DSCOVR: Randomized Primal-Dual Block Coordinate Algorithms for Asynchronous Distributed Optimization

Lin Xiao LIN.XIAO@MICROSOFT.COM
Microsoft Research AI
Redmond, WA 98052, USA

Adams Wei Yu WEIYU@CS.CMU.EDU
Machine Learning Department, Carnegie Mellon University
Pittsburgh, PA 15213, USA

Qihang Lin QIHANG-LIN@UIOWA.EDU
Tippie College of Business, The University of Iowa
Iowa City, IA 52245, USA

Weizhu Chen WZCHEN@MICROSOFT.COM
Microsoft AI and Research
Redmond, WA 98052, USA

October 13, 2017

Abstract

Machine learning with big data often involves large optimization models. For distributed optimization over a cluster of machines, frequent communication and synchronization of all model parameters (optimization variables) can be very costly. A promising solution is to use parameter servers to store different subsets of the model parameters, and update them asynchronously at different machines using local datasets. In this paper, we focus on distributed optimization of large linear models with convex loss functions, and propose a family of randomized primal-dual block coordinate algorithms that are especially suitable for asynchronous distributed implementation with parameter servers. In particular, we work with the saddle-point formulation of such problems which allows simultaneous data and model partitioning, and exploit its structure by doubly stochastic coordinate optimization with variance reduction (DSCOVR). Compared with other first-order distributed algorithms, we show that DSCOVR may require less amount of overall computation and communication, and less or no synchronization. We discuss the implementation details of the DSCOVR algorithms, and present numerical experiments on an industrial distributed computing system.

Keywords: asynchronous distributed optimization, parameter servers, randomized algorithms, saddle-point problems, primal-dual coordinate algorithms, empirical risk minimization

1. Introduction

Algorithms and systems for distributed optimization are critical for solving large-scale machine learning problems, especially when the dataset cannot fit into the memory or storage of a single machine. In this paper, we consider distributed optimization problems of the form

$$\underset{w\in\mathbf{R}^d}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m} f_i(X_i w) + g(w), \tag{1}$$

where $X_i \in \mathbf{R}^{N_i\times d}$ is the local data stored at the ith machine, $f_i : \mathbf{R}^{N_i} \to \mathbf{R}$ is a convex cost function associated with the linear mapping $X_i w$, and $g(w)$ is a convex regularization function. In addition,

we assume that g is separable, i.e., for some integer $n > 0$, we can write

$$g(w) = \sum_{k=1}^{n} g_k(w_k), \tag{2}$$

where $g_k : \mathbf{R}^{d_k} \to \mathbf{R}$, and $w_k \in \mathbf{R}^{d_k}$ for $k = 1, \ldots, n$ are non-overlapping subvectors of $w \in \mathbf{R}^d$ with $\sum_{k=1}^{n} d_k = d$ (they form a partition of w). Many popular regularization functions in machine learning are separable, for example, $g(w) = (\lambda/2)\|w\|_2^2$ or $g(w) = \lambda\|w\|_1$ for some $\lambda > 0$.

An important special case of (1) is distributed empirical risk minimization (ERM) of linear predictors. Let $(x_1, y_1), \ldots, (x_N, y_N)$ be N training examples, where each $x_j \in \mathbf{R}^d$ is a feature vector and $y_j \in \mathbf{R}$ is its label. The ERM problem is formulated as

$$\underset{w \in \mathbf{R}^d}{\text{minimize}} \quad \frac{1}{N} \sum_{j=1}^{N} \phi_j\left(x_j^T w\right) + g(w), \tag{3}$$

where each $\phi_j : \mathbf{R} \to \mathbf{R}$ is a loss function measuring the mismatch between the linear prediction $x_j^T w$ and the label y_j. Popular loss functions in machine learning include, e.g., for regression, the squared loss $\phi_j(t) = (1/2)(t - y_j)^2$, and for classification, the logistic loss $\phi_j(t) = \log(1 + \exp(-y_j t))$ where $y_j \in \{\pm 1\}$. In the distributed optimization setting, the N examples are divided into m subsets, each stored on a different machine. For $i = 1, \ldots, m$, let $\mathcal{I}_i$ denote the subset of $\{1, \ldots, N\}$ stored at machine i and let $N_i = |\mathcal{I}_i|$ (they satisfy $\sum_{i=1}^{m} N_i = N$). Then the ERM problem (3) can be written in the form of (1) by letting X_i consist of x_j^T with $j \in \mathcal{I}_i$ as its rows and defining $f_i : \mathbf{R}^{N_i} \to \mathbf{R}$ as

$$f_i(u_{\mathcal{I}_i}) = \frac{m}{N} \sum_{j \in \mathcal{I}_i} \phi_j(u_j), \tag{4}$$

where $u_{\mathcal{I}_i} \in \mathbf{R}^{N_i}$ is a subvector of $u \in \mathbf{R}^N$, consisting of u_j with $j \in \mathcal{I}_i$.

The nature of distributed algorithms and their convergence properties largely depend on the model of the communication network that connects the m computing machines. A popular setting in the literature is to model the communication network as a graph, and each node can only communicate (in one step) with their neighbors connected by an edge, either synchronously or asynchronously (e.g., Bertsekas and Tsitsiklis, 1989; Nedić and Ozdaglar, 2009). The convergence rates of distributed algorithms in this setting often depend on characteristics of the graph, such as its diameter and the eigenvalues of the graph Laplacian (e.g. Xiao and Boyd, 2006; Duchi et al., 2012; Nedić et al., 2016; Scaman et al., 2017). This is often called the *decentralized* setting.

Another model for the communication network is *centralized*, where all the machines participate synchronous, collective communication, e.g., broadcasting a vector to all m machines, or computing the sum of m vectors, each from a different machine (AllReduce). These collective communication protocols hide the underlying implementation details, which often involve operations on graphs. They are adopted by many popular distributed computing standards and packages, such as MPI (MPI Forum, 2012), MapReduce (Dean and Ghemawat, 2008) and Aparche Spark (Zaharia et al., 2016), and are widely used in machine learning practice (e.g., Lin et al., 2014; Meng et al., 2016). In particular, collective communications are very useful for addressing *data parallelism*, i.e., by allowing different machines to work in parallel to improve the same model $w \in \mathbf{R}^d$ using their local dataset. A disadvantage of collective communications is their synchronization cost: faster machines or machines with less computing tasks have to become idle while waiting for other machines to finish their tasks in order to participate a collective communication.

One effective approach for reducing synchronization cost is to exploit *model parallelism* (here "model" refers to $w \in \mathbf{R}^d$, including all optimization variables). The idea is to allow different machines work in parallel with different versions of the full model or different parts of a common model, with little or no synchronization. The model partitioning approach can be very effective for solving problems with large models (large dimension d). Dedicated *parameter servers* can be set up to store and maintain different subsets of the model parameters, such as the w_k's in (2), and be responsible for coordinating their updates at different workers (Li et al., 2014; Xing et al., 2015). This requires flexible point-to-point communication.

In this paper, we develop a family of randomized algorithms that exploit *simultaneous* data and model parallelism. Correspondingly, we adopt a centralized communication model that support both synchronous collective communication and asynchronous point-to-point communication. In particular, it allows any pair of machines to send/receive a message in a single step, and multiple point-to-point communications may happen in parallel in an event-driven, asynchronous manner. Such a communication model is well supported by the MPI standard. To evaluate the performance of distributed algorithms in this setting, we consider the following three measures.

- *Computation complexity:* total amount of computation, measured by the number of passes over all datasets X_i for $i = 1, \ldots, m$, which can happen in parallel on different machines.
- *Communication complexity:* the total amount of communication required, measured by the equivalent number of vectors in $\mathbf{R}^d$ sent or received across all machines.
- *Synchronous communication:* measured by the total number of vectors in $\mathbf{R}^d$ that requires synchronous collective communication involving all m machines. We single it out from the overall communication complexity as a (partial) measure of the synchronization cost.

In Section 2, we introduce the framework of our randomized algorithms, **D**oubly **S**tochastic **C**oordinate **O**ptimization with **V**ariance **R**eduction (DSCOVR), and summarize our theoretical results on the three measures achieved by DSCOVR. Compared with other first-order methods for distributed optimization, we show that DSCOVR may require less amount of overall computation and communication, and less or no synchronization. Then we present the details of several DSCOVR variants and their convergence analysis in Sections 3-6. We discuss the implementation of different DSCOVR algorithms in Section 7, and present results of our numerical experiments in Section 8.

2. The DSCOVR Framework and Main Results

First, we derive a saddle-point formulation of the convex optimization problem (1). Let f_i^* be the convex conjugate of f_i, i.e., $f_i^*(\alpha_i) = \sup_{u_i \in \mathbf{R}^{N_i}} \left\{\alpha_i^T u_i - f_i(u_i)\right\}$, and define

$$L(w, \alpha) \equiv \frac{1}{m} \sum_{i=1}^{m} \alpha_i^T X_i w - \frac{1}{m} \sum_{i=1}^{m} f_i^*(\alpha_i) + g(w), \tag{5}$$

where $\alpha = [\alpha_1; \ldots; \alpha_m] \in \mathbf{R}^N$. Since both the f_i's and g are convex, $L(w, \alpha)$ is convex in w and concave in α. We also define a pair of *primal* and *dual* functions:

$$P(w) = \max_{\alpha \in \mathbf{R}^N} L(w, \alpha) = \frac{1}{m} \sum_{i=1}^{m} f_i(X_i w) + g(w), \tag{6}$$

$$D(\alpha) = \min_{w \in \mathbf{R}^d} L(w, \alpha) = -\frac{1}{m} \sum_{i=1}^{m} f_i^*(\alpha_i) - g^*\left(-\frac{1}{m} \sum_{i=1}^{m} (X_i)^T \alpha_i\right), \tag{7}$$

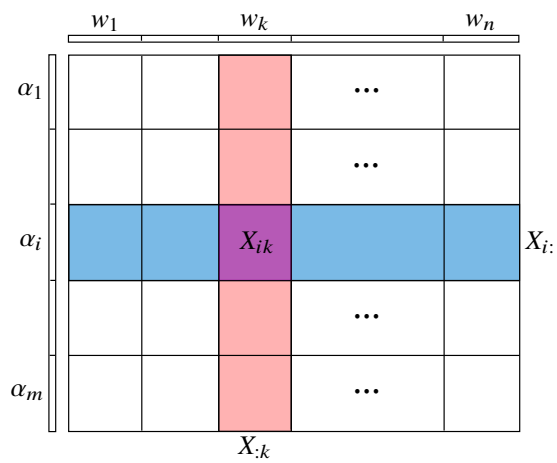

Figure 1: Partition of primal variable w, dual variable α, and the data matrix X.

where $P(w)$ is exactly the objective function in (1)[1] and g^* is the convex conjugate of g. We assume that L has a saddle point $(w^\star, \alpha^\star)$, that is,

$$L(w^\star, \alpha) \le L(w^\star, \alpha^\star) \le L(w, \alpha^\star), \qquad \forall (w, \alpha) \in \mathbf{R}^d \times \mathbf{R}^N.$$

In this case, we have $w^\star = \arg\min\ P(w)$ and $\alpha^\star = \arg\min\ D(\alpha)$, and $P(w^\star) = D(\alpha^\star)$.

The DSCOVR framework is based on solving the convex-concave saddle-point problem

$$\min_{w \in \mathbf{R}^d} \max_{\alpha \in \mathbf{R}^N} L(w, \alpha). \tag{8}$$

Since we assume that g has a separable structure as in (2), we rewrite the saddle-point problem as

$$\min_{w \in \mathbf{R}^d} \max_{\alpha \in \mathbf{R}^N} \left\{ \frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{n} \alpha_i^T X_{ik} w_k - \frac{1}{m} \sum_{i=1}^{m} f_i^*(\alpha_i) + \sum_{k=1}^{n} g_k(w_k) \right\}, \tag{9}$$

where $X_{ik} \in \mathbf{R}^{N_i \times d_k}$ for $k = 1, \ldots, n$ are column partitions of X_i. For convenience, we define the following notations. First, let $X = [X_1; \ldots; X_m] \in \mathbf{R}^{N \times d}$ be the overall data matrix, by stacking the X_i's vertically. Conforming to the separation of g, we also partition X into block columns $X_{:k} \in \mathbf{R}^{N \times d_k}$ for $k = 1, \ldots, n$, where each $X_{:k} = [X_{1k}; \ldots; X_{mk}]$ (stacked vertically). For consistency, we also use $X_{i:}$ to denote X_i from now on. See Figure 1 for an illustration.

We exploit the doubly separable structure in (9) by a doubly stochastic coordinate update algorithm outlined in Algorithm 1. Let $p = \{p_1, \ldots, p_m\}$ and $q = \{q_1, \ldots, q_n\}$ be two probability distributions. During each iteration t, we randomly pick an index $j \in \{1, \ldots, m\}$ with probability p_j, and independently pick an index $l \in \{1, \ldots, n\}$ with probability q_l. Then we compute two vectors $u_j^{(t+1)} \in \mathbf{R}^{N_j}$ and $v_l^{(t+1)} \in \mathbf{R}^{d_l}$ (details to be discussed later), and use them to update the block coordinates α_j and w_l while leaving other block coordinates unchanged. The update formulas in (10) and (11) use the proximal mappings of the (scaled) functions f_j^* and g_l respectively. We

1. More technically, we need to assume that each f_i is convex and lower semi-continuous so that $f_i^{**} = f_i$ (see, e.g., Rockafellar, 1970, Section 12). It automatically holds if f_i is convex and differentiable, which we will assume later.

Algorithm 1 DSCOVR framework

input: initial points $w^{(0)}, \alpha^{(0)}$, and step sizes σ_i for $i = 1, \ldots, m$ and τ_k for $k = 1, \ldots, n$.

1: **for** $t = 0, 1, 2, \ldots,$ **do**

2: pick $j \in \{1, \ldots, m\}$ and $l \in \{1, \ldots, n\}$ randomly with distributions p and q respectively.

3: compute *variance-reduced* stochastic gradients $u_j^{(t+1)}$ and $v_l^{(t+1)}$.

4: update primal and dual block coordinates:

$$\alpha_i^{(t+1)} = \begin{cases} \mathbf{prox}_{\sigma_j f_j^*}\left(\alpha_j^{(t)} + \sigma_j u_j^{(t+1)}\right) & \text{if } i = j, \\ \alpha_i^{(t)}, & \text{if } i \neq j, \end{cases} \tag{10}$$

$$w_k^{(t+1)} = \begin{cases} \mathbf{prox}_{\tau_l g_l}\left(w_l^{(t)} - \tau_l v_l^{(t+1)}\right) & \text{if } k = l, \\ w_k^{(t)}, & \text{if } k \neq l. \end{cases} \tag{11}$$

5: **end for**

recall that the proximal mapping for any convex function $\phi : \mathbf{R}^d \to \mathbf{R} \cup \{\infty\}$ is defined as

$$\mathbf{prox}_\phi(v) \triangleq \arg\min_{u \in \mathbf{R}^d} \left\{ \phi(u) + \frac{1}{2}\|u - v\|^2 \right\}.$$

There are several different ways to compute the vectors $u_j^{(t+1)}$ and $v_l^{(t+1)}$ in Step 3 of Algorithm 1. They should be the partial gradients or stochastic gradients of the bilinear coupling term in $L(w, \alpha)$ with respect to α_j and w_l respectively. Let

$$K(w, \alpha) = \alpha^T X w = \sum_{i=1}^m \sum_{k=1}^n \alpha_i^T X_{ik} w_k,$$

which is the bilinear term in $L(w, \alpha)$ without the factor $1/m$. We can use the following partial gradients in Step 3:

$$\begin{aligned} \bar{u}_j^{(t+1)} &= \frac{\partial K(w^{(t)}, \alpha^{(t)})}{\partial \alpha_j} = \sum_{k=1}^n X_{jk} w_k^{(t)}, \\ \bar{v}_l^{(t+1)} &= \frac{1}{m} \frac{\partial K(w^{(t)}, \alpha^{(t)})}{\partial w_l} = \frac{1}{m} \sum_{i=1}^m (X_{il})^T \alpha_i^{(t)}. \end{aligned} \tag{12}$$

We note that the factor $1/m$ does not appear in the first equation because it multiplies both $K(w, \alpha)$ and $f_j^*(\alpha_j)$ in (9) and hence does not appear in updating α_j. Another choice is to use

$$\begin{aligned} u_j^{(t+1)} &= \frac{1}{q_l} X_{jl} w_l^{(t)}, \\ v_l^{(t+1)} &= \frac{1}{p_j} \frac{1}{m} (X_{jl})^T \alpha_j^{(t)}, \end{aligned} \tag{13}$$

which are unbiased stochastic partial gradients, because

$$\begin{aligned} \mathbf{E}_l\left[u_j^{(t+1)}\right] &= \sum_{k=1}^n q_k \frac{1}{q_k} X_{jk} w_k^{(t)} = \sum_{k=1}^n X_{jk} w_k^{(t)} = \bar{u}_j^{(t+1)}, \\ \mathbf{E}_j\left[v_l^{(t+1)}\right] &= \sum_{i=1}^m p_i \frac{1}{p_i} \frac{1}{m} (X_{il})^T \alpha_i^{(t)} = \frac{1}{m} \sum_{i=1}^m (X_{il})^T \alpha_i^{(t)} = \bar{v}_l^{(t+1)}, \end{aligned}$$

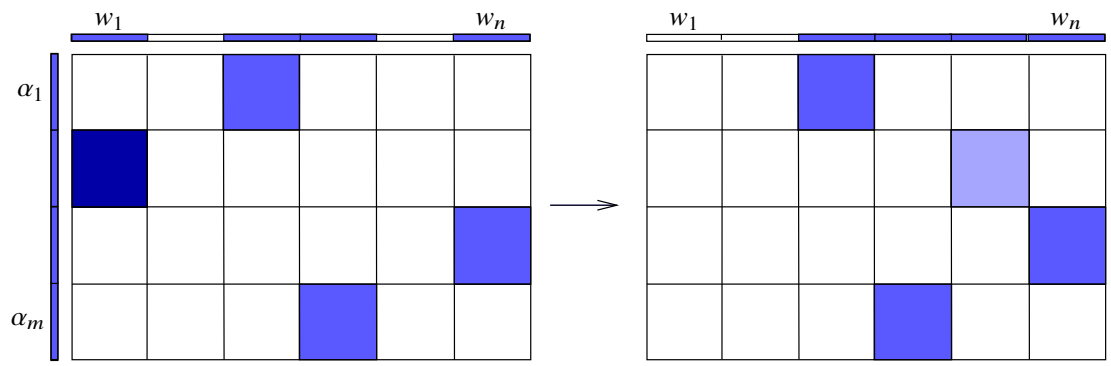

Figure 2: Simultaneous data and model parallelism. At any given time, each machine is busy updating one parameter block and its own dual variable. Whenever some machine is done, it is assigned to work on a random block that is not being updated.

where $\mathbf{E}_j$ and $\mathbf{E}_l$ are expectations with respect to the random indices j and l respectively.

It can be shown that, Algorithm 1 converges to a saddle point of $L(w, \alpha)$ with either choice (12) or (13) in Step 3, and with suitable *step sizes* σ_i and τ_k. It is expected that using the stochastic gradients in (13) leads to a slower convergence rate than applying (12). However, using (13) has the advantage of much less computation during each iteration. Specifically, it employs only one block matrix-vector multiplication for both updates, instead of n and m block multiplications done in (12).

More importantly, the choice in (13) is suitable for parallel and distributed computing. To see this, let $(j^{(t)}, l^{(t)})$ denote the pair of random indices drawn at iteration t (we omit the superscript (t) to simplify notation whenever there is no confusion from the context). Suppose for a sequence of consecutive iterations $t, \ldots, t+s$, there is no common index among $j^{(t)}, \ldots, j^{(t+s)}$, nor among $l^{(t)}, \ldots, l^{(t+s)}$, then these $s+1$ iterations can be done in parallel and they produce the same updates as being done sequentially. Suppose there are $s+1$ processors or machines, then each can carry out one iteration, which includes the updates in (13) as well as (10) and (11). These $s+1$ iterations are independent of each other, and in fact can be done in any order, because each only involve one primal block $w_{l^{(t)}}$ and one dual block $\alpha_{j^{(t)}}$, for both input and output (variables on the right and left sides of the assignments respectively). In contrast, the input for the updates in (12) depend on all primal and dual blocks at the previous iteration, thus cannot be done in parallel.

In practice, suppose we have m machines for solving problem (9), and each holds the data matrix $X_{i:}$ in memory and maintains the dual block α_i, for $i = 1, \ldots, m$. We assume that the number of model partitions n is larger than m, and the n model blocks $\{w_1, \ldots, w_n\}$ are stored at one or more parameter servers. In the beginning, we can randomly pick m model blocks (sampling without replacement) from $\{w_1, \ldots, w_n\}$, and assign each machine to update one of them. If machine i is assigned to update block k, then both α_i and w_k are updated, using only the matrix X_{ik}; moreover, it needs to communicate only the block w_k with the parameter server that are responsible to maintain it. Whenever one machine finishes its update, a scheduler can randomly pick another parameter block that is not currently updated by other machines, and assign it to the free machine. Therefore all machines can work in parallel, in an asynchronous, event-driven manner. Here an event is the completion of a block update at any machine, as illustrated in Figure 2. We will discuss the implementation details in Section 7.

The idea of using doubly stochastic updates for distributed optimization in not new. It has been studied by Yun et al. (2014) for solving the matrix completion problem, and by Matsushima et al. (2014) for solving the saddle-point formulation of the ERM problem. Despite their nice features for parallelization, these algorithms inherit the $O(1/\sqrt{t})$ (or $O(1/t)$ with strong convexity) sublinear convergence rate of the classical stochastic gradient method. They translate into high communication and computation cost for distributed optimization. In this paper, we propose new variants of doubly stochastic update algorithms by using *variance-reduced* stochastic gradients (Step 3 of Algorithm 1). More specifically, we borrow the variance-reduction techniques from SVRG (Johnson and Zhang, 2013) and SAGA (Defazio et al., 2014) to develop the DSCOVR algorithms, which enjoy fast linear rates of convergence. In the rest of this section, we summarize our theoretical results characterizing the three measures for DSCOVR: computation complexity, communication complexity, and synchronization cost. We compare them with distributed implementation of batch first-order algorithms.

2.1 Summary of Main Results

Throughout this paper, we use $\|\cdot\|$ to denote the standard Euclidean norm for vectors. For matrices, $\|\cdot\|$ denotes the operator (spectral) norm and $\|\cdot\|_F$ denotes the Frobenius norm. We make the following assumption regarding the optimization problem (1).

Assumption 1 *Each f_i is convex and differentiable, and its gradient is $(1/\gamma_i)$-Lipschitz continuous, i.e.,*

$$\|\nabla f_i(u) - \nabla f_i(v)\| \le \frac{1}{\gamma_i}\|u - v\|, \quad \forall\, u, v \in \mathbf{R}^{N_i}, \quad i = 1, \ldots, m. \tag{14}$$

In addition, the regularization function g is λ-strongly convex, i.e.,

$$g(w') \ge g(w) + \xi^T(w' - w) + \frac{\lambda}{2}\|w' - w\|^2, \quad \forall\, \xi \in \partial g(w), \quad w', w \in \mathbf{R}^d.$$

Under Assumption 1, each f_i^* is γ_i-strongly convex (see, e.g., Hiriart-Urruty and Lemaréchal, 2001, Theorem 4.2.2), and $L(w, \alpha)$ defined in (5) has a unique saddle point $(w^\star, \alpha^\star)$.

The condition (14) is often referred to as f_i being $1/\gamma_i$-smooth. To simplify discussion, here we assume $\gamma_i = \gamma$ for $i = 1, \ldots, m$. Under these assumptions, each composite function $f_i(X_i w)$ has a smoothness parameter $\|X_i\|^2/\gamma$ (upper bound on the largest eigenvalue of its Hessian). Their average $(1/m)\sum_{i=1}^m f_i(X_i w)$ has a smooth parameter $\|X\|^2/(m\gamma)$, which no larger than the average of the individual smooth parameters $(1/m)\sum_{i=1}^m \|X_i\|^2/\gamma$. We define a condition number for problem (1) as the ratio between this smooth parameter and the convexity parameter λ of g:

$$\kappa_{\text{bat}} = \frac{\|X\|^2}{m\lambda\gamma} \le \frac{1}{m}\sum_{i=1}^m \frac{\|X_{i:}\|^2}{\lambda\gamma} \le \frac{\|X\|_{\max}^2}{\lambda\gamma}, \tag{15}$$

where $\|X\|_{\max} = \max_i\{\|X_{i:}\|\}$. This condition number is a key factor to characterize the iteration complexity of *batch* first-order methods for solving problem (1), i.e., minimizing $P(w)$. Specifically, to find a w such that $P(w) - P(w^\star) \le \epsilon$, the proximal gradient method requires $O\left((1 + \kappa_{\text{bat}})\log(1/\epsilon)\right)$ iterations, and their accelerated variants require $O\left(\left(1 + \sqrt{\kappa_{\text{bat}}}\right)\log(1/\epsilon)\right)$ iterations (e.g., Nesterov, 2004; Beck and Teboulle, 2009; Nesterov, 2013). Primal-dual first order methods for solving the saddle-point problem (8) share the same complexity (Chambolle and Pock, 2011, 2015).

Algorithms	Computation complexity (number of passes over data)	Communication complexity (number of vectors in $\mathbf{R}^d$)
batch first-order methods	$(1+\kappa_{\text{bat}})\log(1/\epsilon)$	$m(1+\kappa_{\text{bat}})\log(1/\epsilon)$
DSCOVR	$(1+\kappa_{\text{rand}}/m)\log(1/\epsilon)$	$(m+\kappa_{\text{rand}})\log(1/\epsilon)$
accelerated batch first-order methods	$\left(1+\sqrt{\kappa_{\text{bat}}}\right)\log(1/\epsilon)$	$m\left(1+\sqrt{\kappa_{\text{bat}}}\right)\log(1/\epsilon)$
accelerated DSCOVR	$\left(1+\sqrt{\kappa_{\text{rand}}/m}\right)\log(1/\epsilon)$	$\left(m+\sqrt{m\cdot\kappa_{\text{rand}}}\right)\log(1/\epsilon)$

Table 1: Computation and communication complexities of batch first-order methods and DSCOVR (for both SVRG and SAGA variants). We omit the $O(\cdot)$ notation in all entries and an extra $\log(1+\kappa_{\text{rand}}/m)$ factor for accelerated DSCOVR algorithms.

A fundamental baseline for evaluating any distributed optimization algorithms is the distributed implementation of batch first-order methods. Let's consider solving problem (1) using the proximal gradient method. During every iteration t, each machine receives a copy of $w^{(t)} \in \mathbf{R}^d$ from a master machine (through Broadcast), and computes the local gradient $z_i^{(t)} = X_i^T \nabla f_i(X_i w^{(t)}) \in \mathbf{R}^d$. Then a collective communication is invoked to compute the batch gradient $z^{(t)} = (1/m)\sum_{i=1}^m z_i^{(t)}$ at the master (Reduce). The master then takes a proximal gradient step, using $z^{(t)}$ and the proximal mapping of g, to compute the next iterate $w^{(t+1)}$ and broadcast it to every machine for the next iteration. We can also use the AllReduce operation in MPI to obtain $z^{(t)}$ at each machine without a master. In either case, the total number of passes over the data is twice the number of iterations (due to matrix-vector multiplications using both X_i and X_i^T), and the number of vectors in $\mathbf{R}^d$ sent/received across all machines is $2m$ times the number of iterations (see Table 1). Moreover, all communications are collective and synchronous.

Since DSCOVR is a family of randomized algorithms for solving the saddle-point problem (8), we would like to find (w, α) such that $\|w^{(t)} - w^\star\|^2 + (1/m)\|\alpha^{(t)} - \alpha^\star\|^2 \le \epsilon$ holds in expectation and with high probability. We list the communication and computation complexities of DSCOVR in Table 1, comparing them with batch first-order methods. Similar guarantees also hold for reducing the duality gap $P(w^{(t)}) - D(\alpha^{(t)})$, where P and D are defined in (6) and (7) respectively.

The key quantity characterizing the complexities of DSCOVR is the condition number κ_{rand}, which can be defined in several different ways. If we pick the data block i and model block k with uniform distribution, i.e., $p_i = 1/m$ for $i = 1, \ldots, m$ and $q_k = 1/n$ for $k = 1, \ldots, n$, then

$$\kappa_{\text{rand}} = \frac{n\|X\|_{m\times n}^2}{\lambda\gamma}, \qquad \text{where} \qquad \|X\|_{m\times n} = \max_{i,k} \|X_{ik}\|. \tag{16}$$

Comparing the definition of κ_{bat} in (15), we have $\kappa_{\text{bat}} \le \kappa_{\text{rand}}$ because

$$\frac{1}{m}\|X\|^2 \le \frac{1}{m}\sum_{i=1}^m \|X_i\|^2 \le \frac{1}{m}\sum_{i=1}^m\sum_{k=1}^n \|X_{ik}\|^2 \le n\|X\|_{m\times n}^2.$$

With $X_{i:} = [X_{i1} \cdots X_{im}] \in \mathbf{R}^{N_i \times d}$ and $X_{:k} = [X_{1k}; \ldots; X_{mk}] \in \mathbf{R}^{N \times d_k}$, we can also define

$$\kappa'_{\text{rand}} = \frac{\|X\|_{\max,F}^2}{\lambda\gamma}, \qquad \text{where} \qquad \|X\|_{\max,F} = \max_{i,k}\left\{\|X_{i:}\|_F,\ \|X_{:k}\|_F\right\}. \tag{17}$$

Algorithms	Synchronous Communication (number of vectors in $\mathbf{R}^d$)	Asynchronous Communication (equiv. number of vectors in $\mathbf{R}^d$)
DSCOVR-SVRG	$m \log(1/\epsilon)$	$\kappa_{\text{rand}} \log(1/\epsilon)$
DSCOVR-SAGA	m	$(m + \kappa_{\text{rand}}) \log(1/\epsilon)$
accelerated DSCOVR-SVRG	$m \log(1/\epsilon)$	$\left(1 + \sqrt{m \cdot \kappa_{\text{rand}}}\right) \log(1/\epsilon)$
accelerated DSCOVR-SAGA	m	$\left(1 + \sqrt{m \cdot \kappa_{\text{rand}}}\right) \log(1/\epsilon)$

Table 2: Breakdown of communication complexities into synchronous and asynchronous communications for two different types of DSCOVR algorithms. We omit the $O(\cdot)$ notation and an extra $\log(1 + \kappa_{\text{rand}}/m)$ factor for accelerated DSCOVR algorithms.

In this case, we also have $\kappa_{\text{bat}} \leq \kappa'_{\text{rand}}$ because $\|X\|_{\max} \leq \|X\|_{\max,F}$. Finally, if we pick the pair (i, k) with non-uniform distribution $p_i = \|X_{i:}\|_F^2 / \|X\|_F^2$ and $q_k = \|X_{:k}\|_F^2 / \|X\|_F^2$, then we can define

$$\kappa''_{\text{rand}} = \frac{\|X\|_F^2}{m\lambda\gamma}. \tag{18}$$

Again we have $\kappa_{\text{bat}} \leq \kappa''_{\text{rand}}$ because $\|X\| \leq \|X\|_F$. We may replace κ_{rand} in Tables 1 and 2 by either κ'_{rand} or κ''_{rand}, depending on the probability distributions p and q and different proof techniques.

From Table 1, we observe similar type of speed-ups in computation complexity, as obtained by variance reduction techniques over the batch first-order algorithms for convex optimization (e.g., Le Roux et al., 2012; Johnson and Zhang, 2013; Defazio et al., 2014; Xiao and Zhang, 2014; Lan and Zhou, 2015; Allen-Zhu, 2017), as well as for convex-concave saddle-point problems (Zhang and Xiao, 2017; Balamurugan and Bach, 2016). Basically, DSCOVR algorithms have potential improvement over batch first-order methods by a factor of m (for non-accelerated algorithms) or $\sqrt{m}$ (for accelerated algorithms), but with a worse condition number. In the worst case, the ratio between κ_{rand} and κ_{bat} may be of order m or larger, thus canceling the potential improvements.

More interestingly, DSCOVR also has similar improvements in terms of communication complexity over batch first-order methods. In Table 2, we decompose the communication complexity of DSCOVR into synchronous and asynchronous communication. The decomposition turns out to be different depending on the variance reduction techniques employed: SVRG (Johnson and Zhang, 2013) versus SAGA (Defazio et al., 2014). We note that DSCOVR-SAGA essentially requires only asynchronous communication, because the synchronous communication of m vectors are only necessary for initialization with non-zero starting point.

The comparisons in Table 1 and 2 give us good understanding of the complexities of different algorithms. However, these complexities are not accurate measures of their performance in practice. For example, collective communication of m vectors in $\mathbf{R}^d$ can often be done in parallel over a spanning tree of the underlying communication network, thus only cost $\log(m)$ times (insted of m times) compared with sending only one vector. Also, for point-to-point communication, sending one vector in $\mathbf{R}^d$ altogether can be much faster than sending n smaller vectors of total length d separately. A fair comparison in term of wall-clock time on a real-world distributed computing system requires customized, efficient implementation of different algorithms. We will shed some light on timing comparisons with numerical experiments in Section 8.

2.2 Related Work

There is an extensive literature on distributed optimization. Many algorithms developed for machine learning adopt the centralized communication setting, due to the wide availability of supporting standards and platforms such as MPI, MapReduce and Spark (as discussed in the introduction). They include parallel implementations of the batch first-order and second-order methods (e.g., Lin et al., 2014; Chen et al., 2014; Lee et al., 2017), ADMM (Boyd et al., 2011), and distributed dual coordinate ascent (Yang, 2013; Jaggi et al., 2014; Ma et al., 2015).

For minimizing the average function $(1/m)\sum_{i=1}^{m} f_i(w)$, in the centralized setting and with only first-order oracles (i.e., gradients of f_i's or their conjugates), it has been shown that distributed implementation of accelerated gradient methods achieves the optimal convergence rate and communication complexity (Arjevani and Shamir, 2015; Scaman et al., 2017). The problem (1) we consider has the extra structure of composition with a linear transformation by the local data, which allows us to exploit simultaneous data and model parallelism using randomized algorithms and obtain improved communication and computation complexity.

Most work on asynchronous distributed algorithms exploit model parallelism in order to reduce the synchronization cost, especially in the setting with parameter servers (e.g., Li et al., 2014; Xing et al., 2015; Aytekin et al., 2016). Besides, delay caused by the asynchrony can be incorporated to the step size to gain practical improvement on convergence (e.g., Agarwal and Duchi, 2011; McMahan and Streeter, 2014; Sra et al., 2016), though the theoretical sublinear rates remain. There are also many recent work on asynchronous parallel stochastic gradient and coordinate-descent algorithms for convex optimization (e.g., Recht et al., 2011; Liu et al., 2014; Shi et al., 2015; Reddi et al., 2015; Richtárik and Takáč, 2016; Peng et al., 2016). When the workloads or computing power of different machines or processors are nonuniform, they may significantly increase *iteration efficiency* (number of iterations done in unit time), but often at the cost of requiring more iterations than their synchronous counterparts (due to delays and stale updates). So there is a subtle balance between iteration efficiency and iteration complexity (e.g., Hannah and Yin, 2017). Our discussions in Section 2.1 show that DSCOVR is capable of improving both aspects.

For solving bilinear saddle-point problems with a finite-sum structure, Zhang and Xiao (2017) proposed a randomized algorithm that works with dual coordinate update but full primal update. Yu et al. (2015) proposed a doubly stochastic algorithm that works with both primal and dual coordinate updates based on equation (12). Both of them achieved accelerated linear convergence rates, but neither can be readily applied to distributed computing. In addition, Balamurugan and Bach (2016) proposed stochastic variance-reduction methods (also based on SVRG and SAGA) for solving more general convex-concave saddle point problems. For the special case with bilinear coupling, they obtained similar computation complexity as DSCOVR. However, their methods require full model updates at each iteration (even though working with only one sub-block of data), thus are not suitable for distributed computing.

With additional assumptions and structure, such as similarity between the local cost functions at different machines or using second-order information, it is possible to obtain better communication complexity for distributed optimization; see, e.g., Shamir et al. (2014); Zhang and Xiao (2015); Reddi et al. (2016). However, these algorithms rely on much more computation at each machine for solving a local sub-problem at each iteration. With additional memory and preprocessing at each machine, Lee et al. (2015) showed that SVRG can be adapted for distributed optimization to obtain low communication complexity.

Algorithm 2 DSCOVR-SVRG

input: initial points $\bar{w}^{(0)}$, $\bar{\alpha}^{(0)}$, number of stages S and number of iterations per stage M.

for $s = 0, 1, 2, \ldots, S-1$ **do**

  $\bar{u}^{(s)} = X\bar{w}^{(s)}$ and $\bar{v}^{(s)} = \frac{1}{m}X^T\bar{\alpha}^{(s)}$

  $w^{(0)} = \bar{w}^{(s)}$ and $\alpha^{(0)} = \bar{\alpha}^{(s)}$

  for $t = 0, 1, 2, \ldots, M-1$ **do**

    pick $j \in \{1, \ldots, m\}$ and $l \in \{1, \ldots, n\}$ randomly with distributions p and q respectively.

    compute variance-reduced stochastic gradients:

$$u_j^{(t+1)} = \bar{u}_j^{(s)} + \frac{1}{q_l}X_{jl}\big(w_l^{(t)} - \bar{w}_l^{(s)}\big), \tag{19}$$

$$v_l^{(t+1)} = \bar{v}_l^{(s)} + \frac{1}{p_j}\frac{1}{m}(X_{jl})^T\big(\alpha_j^{(t)} - \bar{\alpha}_j^{(s)}\big). \tag{20}$$

    update primal and dual block coordinates:

$$\alpha_i^{(t+1)} = \begin{cases} \mathbf{prox}_{\sigma_j f_j^*}\big(\alpha_j^{(t)} + \sigma_j u_j^{(t+1)}\big) & \text{if } i = j, \\ \alpha_i^{(t)}, & \text{if } i \neq j, \end{cases}$$

$$w_k^{(t+1)} = \begin{cases} \mathbf{prox}_{\tau_l g_l}\big(w_l^{(t)} - \tau_l v_l^{(t+1)}\big) & \text{if } k = l, \\ w_k^{(t)}, & \text{if } k \neq l. \end{cases}$$

  end for

  $\bar{w}^{(s+1)} = w^{(M)}$ and $\bar{\alpha}^{(s+1)} = \alpha^{(M)}$.

end for

output: $\bar{w}^{(S)}$ and $\bar{\alpha}^{(S)}$.

3. The DSCOVR-SVRG Algorithm

From this section to Section 6, we present several realizations of DSCOVR using different variance reduction techniques and acceleration schemes, and analyze their convergence properties. These algorithms are presented and analyzed as sequential randomized algorithms. We will discuss how to implement them for asynchronous distributed computing in Section 7.

Algorithm 2 is a DSCOVR algorithm that uses the technique of SVRG (Johnson and Zhang, 2013) for variance reduction. The iterations are divided into stages and each stage has a inner loop. Each stage is initialized by a pair of vectors $\bar{w}^{(s)} \in \mathbf{R}^d$ and $\bar{\alpha}^{(s)} \in \mathbf{R}^N$, which come from either initialization (if $s = 0$) or the last iterate of the previous stage (if $s > 0$). At the beginning of each stage, we compute the batch gradients

$$\bar{u}^{(s)} = \frac{\partial}{\partial \bar{\alpha}^{(s)}}\left((\bar{\alpha}^{(s)})^T X\bar{w}^{(s)}\right) = X\bar{w}^{(s)}, \qquad \bar{v}^{(s)} = \frac{\partial}{\partial \bar{w}^{(s)}}\left(\frac{1}{m}(\bar{\alpha}^{(s)})^T X\bar{w}^{(s)}\right) = \frac{1}{m}X^T\bar{\alpha}^{(s)}.$$

The vectors $\bar{u}^{(s)}$ and $\bar{v}^{(s)}$ share the same partitions as $\alpha^{(t)}$ and $w^{(t)}$, respectively. Inside each stage s, the variance-reduced stochastic gradients are computed in (19) and (20). It is easy to check that they

are unbiased. More specifically, taking expectation of $u_j^{(t+1)}$ with respect to the random index l gives

$$\mathbf{E}_l\left[u_j^{(t+1)}\right] = \bar{u}_j^{(s)} + \sum_{k=1}^{n} q_k \frac{1}{q_k} X_{jk}\left(w_k^{(t)} - \bar{w}_k^{(s)}\right) = \bar{u}_j^{(s)} + X_{j:}w^{(t)} - X_{j:}\bar{w}^{(s)} = X_{j:}w^{(t)},$$

and taking expectation of $v_l^{(t+1)}$ with respect to the random index j gives

$$\mathbf{E}_j\left[v_l^{(t+1)}\right] = \bar{v}_l^{(s)} + \sum_{i=1}^{m} p_i \frac{1}{p_i}\frac{1}{m}(X_{il})^T\left(\alpha_i^{(t)} - \bar{\alpha}_i^{(s)}\right) = \bar{v}_l^{(s)} + \frac{1}{m}(X_{:l})^T\left(\alpha^{(t)} - \bar{\alpha}^{(s)}\right) = \frac{1}{m}(X_{:l})^T\alpha^{(t)}.$$

In order to measure the distance of any pair of primal and dual variables to the saddle point, we define a weighted squared Euclidean norm on $\mathbf{R}^{d+N}$. Specifically, for any pair (w, α) where $w \in \mathbf{R}^d$ and $\alpha = [\alpha_1, \ldots, \alpha_m] \in \mathbf{R}^N$ with $\alpha_i \in \mathbf{R}^{N_i}$, we define

$$\Omega(w, \alpha) = \lambda\|w\|^2 + \frac{1}{m}\sum_{i=1}^{m}\gamma_i\|\alpha_i\|^2. \tag{21}$$

If $\gamma_i = \gamma$ for all $i = 1, \ldots, m$, then $\Omega(w, \alpha) = \lambda\|w\|^2 + \frac{\gamma}{m}\|\alpha\|^2$. We have the following theorem concerning the convergence rate of Algorithm 2.

Theorem 1 *Suppose Assumption 1 holds, and let $(w^\star, \alpha^\star)$ be the unique saddle point of $L(w, \alpha)$. Let Γ be a constant that satisfies*

$$\Gamma \;\geq\; \max_{i,k}\left\{\frac{1}{p_i}\left(1 + \frac{9\|X_{ik}\|^2}{2q_k\lambda\gamma_i}\right),\; \frac{1}{q_k}\left(1 + \frac{9n\|X_{ik}\|^2}{2mp_i\lambda\gamma_i}\right)\right\}. \tag{22}$$

In Algorithm 2, if we choose the step sizes as

$$\sigma_i \;=\; \frac{1}{2\gamma_i(p_i\Gamma - 1)}, \qquad i = 1, \ldots, m, \tag{23}$$

$$\tau_k \;=\; \frac{1}{2\lambda(q_k\Gamma - 1)}, \qquad k = 1, \ldots, n, \tag{24}$$

and the number of iterations during each stage satisfies $M \geq \log(3)\Gamma$, then for any $s > 0$,

$$\mathbf{E}\left[\Omega\left(\bar{w}^{(s)} - w^\star, \bar{\alpha}^{(s)} - \alpha^\star\right)\right] \leq \left(\frac{2}{3}\right)^s \Omega\left(\bar{w}^{(0)} - w^\star, \bar{\alpha}^{(0)} - \alpha^\star\right). \tag{25}$$

The proof of Theorem 1 is given in Appendix A. Here we discuss how to choose the parameter Γ to satisfy (22). For simplicity, we assume $\gamma_i = \gamma$ for all $i = 1, \ldots, m$.

- If we let $\|X\|_{m\times n} = \max_{i,k}\{\|X_{ik}\|\}$ and sample with the uniform distribution across both rows and columns, i.e., $p_i = 1/m$ for $i = 1, \ldots, m$ and $q_k = 1/n$ for $k = 1, \ldots, n$, then we can set

$$\Gamma = \max\{m, n\}\left(1 + \frac{9n\|X\|_{m\times n}^2}{2\lambda\gamma}\right) = \max\{m, n\}\left(1 + \frac{9}{2}\kappa_{\text{rand}}\right),$$

  where $\kappa_{\text{rand}} = n\|X\|_{m\times n}^2/(\lambda\gamma)$ as defined in (16).

- An alternative condition for Γ to satisfy is (shown in Section A.1 in the Appendix)

$$\Gamma \geq \max_{i,k} \left\{ \frac{1}{p_i} \left(1 + \frac{9\|X_{:k}\|_F^2}{2q_k m\lambda\gamma_i} \right), \ \frac{1}{q_k} \left(1 + \frac{9\|X_{i:}\|_F^2}{2p_i m\lambda\gamma_i} \right) \right\}. \tag{26}$$

  Again using uniform sampling, we can set

$$\Gamma = \max\{m, n\} \left(1 + \frac{9\|X\|_{\max,F}^2}{2\lambda\gamma} \right) = \max\{m, n\} \left(1 + \frac{9}{2}\kappa'_{\text{rand}} \right),$$

  where $\|X\|_{\max,F} = \max_{i,k}\{\|X_{i:}\|_F, \|X_{:k}\|_F\}$ and $\kappa'_{\text{rand}} = \|X\|_{\max,F}^2 \big/ (\lambda\gamma)$ as defined in (17).

- Using the condition (26), if we choose the probabilities to be proportional to the squared Frobenius norms of the data partitions, i.e.,

$$p_i = \frac{\|X_{i:}\|_F^2}{\|X\|_F^2}, \qquad q_k = \frac{\|X_{:k}\|_F^2}{\|X\|_F^2}, \tag{27}$$

  then we can choose

$$\Gamma = \frac{1}{\min_{i,k}\{p_i, q_k\}} \left(1 + \frac{9\|X\|_F^2}{2m\lambda\gamma} \right) = \frac{1}{\min_{i,k}\{p_i, q_k\}} \left(1 + \frac{9}{2}\kappa''_{\text{rand}} \right),$$

  where $\kappa''_{\text{rand}} = \|X\|_F^2 \big/ (m\lambda\gamma)$. Moreover, we can set the step sizes as (see Appendix A.1)

$$\sigma_i = \frac{m\lambda}{9\|X\|_F^2}, \qquad \tau_k = \frac{m\gamma_i}{9\|X\|_F^2}.$$

- For the ERM problem (3), we assume that each loss function ϕ_j, for $j = 1, \ldots, N$, is $1/\nu$-smooth. According to (4), the smooth parameter for each f_i is $\gamma_i = \gamma = (N/m)\nu$. Let R be the largest Euclidean norm among all rows of X (or we can normalize each row to have the same norm R), then we have $\|X\|_F^2 \leq NR^2$ and

$$\kappa''_{\text{rand}} = \frac{\|X\|_F^2}{m\lambda\gamma} \leq \frac{NR^2}{m\lambda\gamma} = \frac{R^2}{\lambda\nu}. \tag{28}$$

  The upper bound $R^2/(\lambda\nu)$ is a condition number used for characterizing the iteration complexity of many randomized algorithms for ERM (e.g., Shalev-Shwartz and Zhang, 2013; Le Roux et al., 2012; Johnson and Zhang, 2013; Defazio et al., 2014; Zhang and Xiao, 2017). In this case, using the non-uniform sampling in (27), we can set the step sizes to be

$$\sigma_i = \frac{\lambda}{9R^2}\frac{m}{N}, \qquad \tau_k = \frac{\gamma}{9R^2}\frac{m}{N} = \frac{\nu}{9R^2}. \tag{29}$$

Next we estimate the overall computation complexity of DSCOVR-SVRG in order to achieve $\mathbf{E}\big[\Omega(\bar{w}^{(s)} - w^\star, \bar{\alpha}^{(s)} - \alpha^\star)\big] \leq \epsilon$. From (25), the number of stages required is $\log\big(\Omega^{(0)}/\epsilon\big) \big/ \log(3/2)$, where $\Omega^{(0)} = \Omega(\bar{w}^{(0)} - w^\star, \bar{\alpha}^{(0)} - \alpha^\star)$. The number of inner iterations within each stage is $M = \log(3)\Gamma$. At the beginning of of each stage, computing the batch gradients $\bar{u}^{(s)}$ and $\bar{v}^{(s)}$ requires

going through the whole data set X, whose computational cost is equivalent to $m \times n$ inner iterations. Therefore, the overall complexity of Algorithm 2, measured by total number of inner iterations, is

$$O\left(\left(mn + \Gamma\right) \log\left(\frac{\Omega^{(0)}}{\epsilon}\right)\right).$$

To simplify discussion, we further assume $m \leq n$, which is always the case for distributed implementation (see Figure 2 and Section 7). In this case, we can let $\Gamma = n(1 + (9/2)\kappa_{\text{rand}})$. Thus the above iteration complexity becomes

$$O\big(n(1 + m + \kappa_{\text{rand}}) \log(1/\epsilon)\big). \tag{30}$$

Since the iteration complexity in (30) counts the number of blocks X_{ik} being processed, the number of passes over the whole dataset X can be obtained by dividing it by mn, i.e.,

$$O\left(\left(1 + \frac{\kappa_{\text{rand}}}{m}\right) \log(1/\epsilon)\right). \tag{31}$$

This is the computation complexity of DSCOVR listed in Table 1. We can replace κ_{rand} by κ'_{rand} or κ''_{rand} depending on different proof techniques and sampling probabilities as discussed above. We will address the communication complexity for DSCOVR-SVRG, including its decomposition into synchronous and asynchronous ones, after describing its implementation details in Section 7.

In addition to convergence to the saddle point, our next result shows that the primal-dual optimality gap also enjoys the same convergence rate, under slightly different conditions.

Theorem 2 *Suppose Assumption 1 holds, and let $P(w)$ and $D(\alpha)$ be the primal and dual functions defined in* (6) *and* (7)*, respectively. Let Λ and Γ be two constants that satisfy*

$$\Lambda \;\geq\; \|X_{ik}\|_F^2, \qquad i = 1, \ldots, m, \quad k = 1, \ldots, n,$$

and

$$\Gamma \;\geq\; \max_{i,k} \left\{ \frac{1}{p_i}\left(1 + \frac{18\Lambda}{q_k \lambda \gamma_i}\right),\; \frac{1}{q_k}\left(1 + \frac{18 n \Lambda}{p_i m \lambda \gamma_i}\right) \right\}.$$

In Algorithm 2, if we choose the step sizes as

$$\sigma_i \;=\; \frac{1}{\gamma_i(p_i\Gamma - 1)}, \qquad i = 1, \ldots, m, \tag{32}$$

$$\tau_k \;=\; \frac{1}{\lambda(q_k\Gamma - 1)}, \qquad k = 1, \ldots, n, \tag{33}$$

and the number of iterations during each stage satisfies $M \geq \log(3)\Gamma$, then

$$\mathbf{E}\left[P(\bar{w}^{(s)}) - D(\bar{\alpha}^{(s)})\right] \;\leq\; \left(\frac{2}{3}\right)^s 2\Gamma\left(P(\bar{w}^{(0)}) - D(\bar{\alpha}^{(0)})\right). \tag{34}$$

The proof of Theorem 2 is given in Appendix B. In terms of iteration complexity or total number of passes to reach $\mathbf{E}\big[P(\bar{w}^{(s)}) - D(\bar{\alpha}^{(s)})\big] \leq \epsilon$, we need to add an extra factor of $\log(1 + \kappa_{\text{rand}})$ to (30) or (31), due to the factor Γ on the right-hand side of (34).

Algorithm 3 DSCOVR-SAGA

input: initial points $w^{(0)}, \alpha^{(0)}$, and number of iterations M.

1: $\bar{u}^{(0)} = Xw^{(0)}$ and $\bar{v}^{(0)} = \frac{1}{m}X^T\alpha^{(0)}$

2: $U_{ik}^{(0)} = X_{ik}w_k^{(0)}$, $V_{ik}^{(0)} = \frac{1}{m}(\alpha_i^{(0)})^T X_{ik}$, for all $i = 1, \ldots, m$ and $k = 1, \ldots, K$.

3: **for** $t = 0, 1, 2, \ldots, M-1$ **do**

4: pick $j \in \{1, \ldots, m\}$ and $l \in \{1, \ldots, n\}$ randomly with distributions p and q respectively.

5: compute variance-reduced stochastic gradients:

$$u_j^{(t+1)} = \bar{u}_j^{(t)} - \frac{1}{q_l}U_{jl}^{(t)} + \frac{1}{q_l}X_{jl}w_l^{(t)}, \tag{35}$$

$$v_l^{(t+1)} = \bar{v}_l^{(t)} - \frac{1}{p_j}(V_{jl}^{(t)})^T + \frac{1}{p_j}\frac{1}{m}(X_{jl})^T\alpha_j^{(t)}. \tag{36}$$

6: update primal and dual block coordinates:

$$\alpha_i^{(t+1)} = \begin{cases} \mathbf{prox}_{\sigma_j\Phi_j^*}\left(\alpha_j^{(t)} + \sigma_j u_j^{(t+1)}\right) & \text{if } i = j. \\ \alpha_i^{(t)}, & \text{if } i \neq j, \end{cases}$$

$$w_k^{(t+1)} = \begin{cases} \mathbf{prox}_{\tau_l g_l}\left(w_l^{(t)} - \tau_l v_l^{(t+1)}\right) & \text{if } k = l, \\ w_k^{(t)}, & \text{if } i \neq j. \end{cases}$$

7: update averaged stochastic gradients:

$$\bar{u}_i^{(t+1)} = \begin{cases} \bar{u}_j^{(t)} - U_{jl}^{(t)} + X_{jl}w_l^{(t)} & \text{if } i = j, \\ \bar{u}_i^{(t)} & \text{if } i \neq j, \end{cases}$$

$$\bar{v}_k^{(t+1)} = \begin{cases} \bar{v}_l^{(t)} - (V_{jl}^{(t)})^T + \frac{1}{m}(X_{jl})^T\alpha_j^{(t)} & \text{if } k = l, \\ \bar{v}_k^{(t)} & \text{if } k \neq l, \end{cases}$$

8: update the table of historical stochastic gradients:

$$U_{ik}^{(t+1)} = \begin{cases} X_{jl}w_l^{(t)} & \text{if } i = j \text{ and } k = l, \\ U_{ik}^{(t)} & \text{otherwise.} \end{cases}$$

$$V_{ik}^{(t+1)} = \begin{cases} \frac{1}{m}\left((X_{jl})^T\alpha_j^{(t)}\right)^T & \text{if } i = j \text{ and } k = l, \\ V_{ik}^{(t)} & \text{otherwise.} \end{cases}$$

9: **end for**

output: $w^{(M)}$ and $\alpha^{(M)}$.

4. The DSCOVR-SAGA Algorithm

Algorithm 3 is a DSCOVR algorithm that uses the techniques of SAGA (Defazio et al., 2014) for variance reduction. This is a single stage algorithm with iterations indexed by t. In order to compute the variance-reduced stochastic gradients $u_j^{(t+1)}$ and $v_l^{(t+1)}$ at each iteration, we also need to maintain and update two vectors $\bar{u}^{(t)} \in \mathbf{R}^N$ and $\bar{v}^{(t)} \in \mathbf{R}^d$, and two matrices $U^{(t)} \in \mathbf{R}^{N\times n}$ and $V^{(t)} \in \mathbf{R}^{m\times d}$.

The vector $\bar{u}^{(t)}$ shares the same partition as $\alpha^{(t)}$ into m blocks, and $\bar{v}^{(t)}$ share the same partitions as $w^{(t)}$ into n blocks. The matrix $U^{(t)}$ is partitioned into $m \times n$ blocks, with each block $U_{ik}^{(t)} \in \mathbf{R}^{N_i \times 1}$. The matrix $V^{(t)}$ is also partitioned into $m \times n$ blocks, with each block $V_{ik}^{(t)} \in \mathbf{R}^{1 \times d_k}$. According to the updates in Steps 7 and 8 of Algorithm 3, we have

$$\bar{u}_i^{(t)} = \sum_{k=1}^{n} U_{ik}^{(t)}, \qquad i = 1, \ldots, m, \tag{37}$$

$$\bar{v}_k^{(t)} = \sum_{i=1}^{m} \big(V_{ik}^{(t)}\big)^T, \qquad k = 1, \ldots, n. \tag{38}$$

Based on the above constructions, we can show that $u_j^{(t+1)}$ is an unbiased stochastic gradient of $(\alpha^{(t)})^T X w^{(t)}$ with respect to α_j, and $v_l^{(t+1)}$ is an unbiased stochastic gradient of $(1/m)\big((\alpha^{(t)})^T X w^{(t)}\big)$ with respect to w_l. More specifically, according to (35), we have

$$\begin{aligned}
\mathbf{E}_l\big[u_j^{(t+1)}\big] &= \bar{u}_j^{(t)} - \sum_{k=1}^{n} q_k \left(\frac{1}{q_k} U_{jk}^{(t)}\right) + \sum_{k=1}^{n} q_k \left(\frac{1}{q_k} X_{jk} w_k^{(t)}\right) \\
&= \bar{u}_j^{(t)} - \sum_{k=1}^{n} U_{jk}^{(t)} + \sum_{k=1}^{n} X_{jk} w_k^{(t)} \\
&= \bar{u}_j^{(t)} - \bar{u}_j^{(t)} + X_{j:} w^{(t)} \\
&= X_{j:} w^{(t)} = \frac{\partial}{\partial \alpha_j} \left(\big(\alpha^{(t)}\big)^T X w^{(t)}\right),
\end{aligned} \tag{39}$$

where the third equality is due to (37). Similarly, according to (36), we have

$$\begin{aligned}
\mathbf{E}_j\big[v_l^{(t+1)}\big] &= \bar{v}_l^{(t)} - \sum_{i=1}^{m} p_i \left(\frac{1}{p_i} (V_{il}^{(t)})^T\right) + \sum_{i=1}^{m} p_i \left(\frac{1}{p_i m} (X_{il})^T \alpha_i^{(t)}\right) \\
&= \bar{v}_l^{(t)} - \sum_{i=1}^{m} V_{il}^{(t)} + \frac{1}{m} \sum_{i=1}^{m} (X_{il})^T \alpha_i^{(t)} \\
&= \bar{v}_l^{(t)} - \bar{v}_l^{(t)} + \frac{1}{m} (X_{:l})^T \alpha^{(t)} \\
&= \frac{1}{m} (X_{:l})^T \alpha^{(t)} = \frac{\partial}{\partial w_l} \left(\frac{1}{m} \big(\alpha^{(t)}\big)^T X w^{(t)}\right),
\end{aligned} \tag{40}$$

where the third equality is due to (38).

Regarding the convergence of DSCOVR-SAGA, we have the following theorem, which is proved in Appendix C.

Theorem 3 *Suppose Assumption 1 holds, and let $(w^\star, \alpha^\star)$ be the unique saddle point of $L(w, \alpha)$. Let Γ be a constant that satisfies*

$$\Gamma \geq \max_{i,k} \left\{ \frac{1}{p_i} \left(1 + \frac{9\|X_{ik}\|^2}{2 q_k \lambda \gamma_i}\right), \ \frac{1}{q_k} \left(1 + \frac{9n\|X_{ik}\|^2}{2 p_i m \lambda \gamma_i}\right), \ \frac{1}{p_i q_k} \right\}. \tag{41}$$

If we choose the step sizes as

$$\sigma_i = \frac{1}{2\gamma_i (p_i \Gamma - 1)}, \qquad i = 1, \ldots, m, \tag{42}$$

$$\tau_k = \frac{1}{2\lambda (q_k \Gamma - 1)}, \qquad k = 1, \ldots, n, \tag{43}$$

Algorithm 4 Accelerated DSCOVR

input: initial points $\tilde{w}^{(0)}, \tilde{\alpha}^{(0)}$, and parameter $\delta > 0$.

1: **for** $r = 0, 1, 2, \ldots,$ **do**

2: find an approximate saddle point of (46) using one of the following two options:

- *option 1:* run Algorithm 2 with $S = \frac{2\log(2(1+\delta))}{\log(3/2)}$ and $M = \log(3)\Gamma_\delta$ to obtain

$$(\tilde{w}^{(r+1)}, \tilde{\alpha}^{(r+1)}) = \text{DSCOVR-SVRG}(\tilde{w}^{(r)}, \tilde{\alpha}^{(r)}, S, M).$$

- *option 2:* run Algorithm 3 with $M = 6\log\left(\frac{8(1+\delta)}{3}\right)\Gamma_\delta$ to obtain

$$(\tilde{w}^{(r+1)}, \tilde{\alpha}^{(r+1)}) = \text{DSCOVR-SAGA}(\tilde{w}^{(r)}, \tilde{\alpha}^{(r)}, M).$$

3: **end for**

Then the iterations of Algorithm 3 satisfy, for $t = 1, 2, \ldots,$

$$\mathbf{E}\left[\Omega\big(w^{(t)} - w^\star, \alpha^{(t)} - \alpha^\star\big)\right] \;\leq\; \left(1 - \frac{1}{3\Gamma}\right)^t \frac{4}{3}\Omega\big(w^{(0)} - w^\star, \alpha^{(0)} - \alpha^\star\big). \tag{44}$$

The condition on Γ in (41) is very similar to the one in (22), except that here we have an additional term $1/(p_i q_k)$ when taking the maximum over i and k. This results in an extra mn term in estimating Γ under uniform sampling. Assuming $m \leq n$ (true for distributed implementation), we can let

$$\Gamma = n\left(1 + \frac{9}{2}\kappa_{\text{rand}}\right) + mn.$$

According to (44), in order to achieve $\mathbf{E}\big[\Omega(w^{(t)} - w^\star, \alpha^{(t)} - \alpha^\star)\big] \leq \epsilon$, DSCOVR-SAGA needs $O\left(\Gamma\log(1/\epsilon)\right)$ iterations. Using the above expression for Γ, the iteration complexity is

$$O\big(n(1 + m + \kappa_{\text{rand}})\log(1/\epsilon)\big), \tag{45}$$

which is the same as (30) for DSCOVR-SVRG. This also leads to the same computational complexity measured by the number of passes over the whole dataset, which is given in (31). Again we can replace κ_{rand} by κ'_{rand} or κ''_{rand} as discussed in Section 3. We will discuss the communication complexity of DSCOVR-SAGA in Section 7, after describing its implementation details.

5. Accelerated DSCOVR Algorithms

In this section, we develop an accelerated DSCOVR algorithm by following the "catalyst" framework (Lin et al., 2015; Frostig et al., 2015). More specifically, we adopt the same procedure by Balamurugan and Bach (2016) for solving convex-concave saddle-point problems.

Algorithm 4 proceeds in rounds indexed by $r = 0, 1, 2, \ldots$. Given the initial points $\tilde{w}^{(0)} \in \mathbf{R}^d$ and $\tilde{\alpha}^{(0)} \in \mathbf{R}^N$, each round r computes two new vectors $\tilde{w}^{(r+1)}$ and $\tilde{\alpha}^{(r+1)}$ using either the DSCOVR-SVRG or DSCOVR-SAGA algorithm for solving a regulated saddle-point problem, similar to the classical proximal point algorithm (Rockafellar, 1976).

Let $\delta > 0$ be a parameter which we will determine later. Consider the following perturbed saddle-point function for round r:

$$L_\delta^{(r)}(w, a) = L(w, \alpha) + \frac{\delta\lambda}{2}\|w - \tilde{w}^{(r)}\|^2 - \frac{\delta}{2m}\sum_{i=1}^{m}\gamma_i\|\alpha_i - \tilde{\alpha}_i^{(r)}\|^2. \tag{46}$$

Under Assumption 1, the function $L_\delta^{(r)}(w, a)$ is $(1+\delta)\lambda$-strongly convex in w and $(1+\delta)\gamma_i/m$-strongly concave in α_i. Let Γ_δ be a constant that satisfies

$$\Gamma_\delta \;\geq\; \max_{i,k}\left\{\frac{1}{p_i}\left(1 + \frac{9\|X_{ik}\|^2}{2q_k\lambda\gamma_i(1+\delta)^2}\right),\ \frac{1}{q_k}\left(1 + \frac{9n\|X_{ik}\|^2}{2p_i m\lambda\gamma_i(1+\delta)^2}\right),\ \frac{1}{p_i q_k}\right\},$$

where the right-hand side is obtained from (41) by replacing λ and γ_i with $(1+\delta)\lambda$ and $(1+\delta)\gamma_i$ respectively. The constant Γ_δ is used in Algorithm 4 to determine the number of inner iterations to run with each round, as well as for setting the step sizes. The following theorem is proved in Appendix D.

Theorem 4 *Suppose Assumption 1 holds, and let $(w^\star, \alpha^\star)$ be the saddle-point of $L(w, \alpha)$. With either options in Algorithm 4, if we choose the step sizes (inside Algorithm 2 or Algorithm 3) as*

$$\sigma_i \;=\; \frac{1}{2(1+\delta)\gamma_i(p_i\Gamma_\delta - 1)}, \qquad i = 1, \ldots, m, \tag{47}$$

$$\tau_k \;=\; \frac{1}{2(1+\delta)\lambda(q_k\Gamma_\delta - 1)}, \qquad k = 1, \ldots, n. \tag{48}$$

Then for all $r \geq 1$,

$$\mathbf{E}\left[\Omega\big(\tilde{w}^{(r)} - w^\star, \tilde{\alpha}^{(r)} - \alpha^\star\big)\right] \;\leq\; \left(1 - \frac{1}{2(1+\delta)}\right)^{2r}\Omega\big(\tilde{w}^{(0)} - w^\star, \tilde{\alpha}^{(0)} - \alpha^\star\big).$$

According to Theorem 4, in order to have $\mathbf{E}\big[\Omega\big(\tilde{w}^{(r)} - w^\star, \tilde{\alpha}^{(r)} - \alpha^\star\big)\big] \leq \epsilon$, we need the number of rounds r to satisfy

$$r \geq (1+\delta)\log\left(\frac{\Omega\big(\tilde{w}^{(0)} - w^\star, \tilde{\alpha}^{(0)} - \alpha^\star\big)}{\epsilon}\right).$$

Following the discussions in Sections 3 and 4, when using uniform sampling and assuming $m \leq n$, we can have

$$\Gamma_\delta = n\left(1 + \frac{9\kappa_{\text{rand}}}{2(1+\delta)^2}\right) + mn. \tag{49}$$

Then the total number of block coordinate updates in Algorithm 4 is

$$O\big((1+\delta)\Gamma_\delta\log(1+\delta)\log(1/\epsilon)\big),$$

where the $\log(1+\delta)$ factor comes from the number of stages S in option 1 and number of steps M in option 2. We hide the $\log(1+\delta)$ factor with the $\widetilde{O}$ notation and plug (49) into the expression above to obtain

$$\widetilde{O}\left(n\left((1+\delta)(1+m) + \frac{\kappa_{\text{rand}}}{(1+\delta)}\right)\log\left(\frac{1}{\epsilon}\right)\right).$$

Now we can choose δ depending on the relative size of κ_{rand} and m:

- If $\kappa_{\text{rand}} > 1 + m$, we can minimizing the above expression by choosing $\delta = \sqrt{\frac{\kappa_{\text{rand}}}{1+m}} - 1$, so that the overall iteration complexity becomes $\widetilde{O}\left(n\sqrt{m\kappa_{\text{rand}}}\log(1/\epsilon)\right)$.
- If $\kappa_{\text{rand}} \leq m + 1$, then no acceleration is necessary and we can choose $\delta = 0$ to proceed with a single round. In this case, the iteration complexity is $O(mn)$ as seen from (49).

Therefore, in either case, the total number of block iterations by Algorithm 4 can be written as

$$\widetilde{O}\left(mn + n\sqrt{m\kappa_{\text{rand}}}\log(1/\epsilon)\right). \tag{50}$$

As discussed before, the total number of passes over the whole dataset is obtained by dividing by mn:

$$\widetilde{O}\left(1 + \sqrt{\kappa_{\text{rand}}/m}\log(1/\epsilon)\right).$$

This is the computational complexity of accelerated DSCOVR listed in Table 1.

5.1 Proximal Mapping for Accelerated DSCOVR

When applying Algorithm 2 or 3 to approximate the saddle-point of (46), we need to replace the proximal mappings of $g_k(\cdot)$ and $f_i^*(\cdot)$ by those of $g_k(\cdot)+(\delta\lambda/2)\|\cdot-\tilde{w}_k^{(r)}\|^2$ and $f_i^*(\cdot)+(\delta\gamma_i/2)\|\cdot-\tilde{\alpha}_i^{(r)}\|^2$, respectively. More precisely, we replace $w_k^{(t+1)} = \mathbf{prox}_{\tau_k g_k}\left(w_k^{(t)} - \tau_k v_k^{(t+1)}\right)$ by

$$\begin{aligned} w_k^{(t+1)} &= \arg\min_{w_k \in \mathbf{R}^{d_k}} \left\{ g_k(w_k) + \frac{\delta\lambda}{2}\left\|w_k - \tilde{w}_k^{(r)}\right\|^2 + \frac{1}{2\tau_k}\left\|w_k - \left(w_k^{(t)} - \tau_k v_k^{(t+1)}\right)\right\|^2 \right\} \\ &= \mathbf{prox}_{\frac{\tau_k}{1+\tau_k\delta\lambda} g_k}\left(\frac{1}{1+\tau_k\delta\lambda}\left(w_k^{(t)} - \tau_k v_k^{(t+1)}\right) + \frac{\tau_k\delta\lambda}{1+\tau_k\delta\lambda}\tilde{w}_k^{(r)}\right), \end{aligned} \tag{51}$$

and replace $\alpha_i^{(t+1)} = \mathbf{prox}_{\sigma_i f_i^*}\left(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\right)$ by

$$\begin{aligned} \alpha_i^{(t+1)} &= \arg\min_{\alpha_i \in \mathbf{R}^{N_i}} \left\{ f_i^*(\alpha_i) + \frac{\delta\gamma_i}{2}\left\|\alpha_i - \tilde{\alpha}_i^{(r)}\right\|^2 + \frac{1}{2\sigma_i}\left\|\alpha_i - \left(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\right)\right\|^2 \right\} \\ &= \mathbf{prox}_{\frac{\sigma_i}{1+\sigma_i\delta\gamma_i} f_i^*}\left(\frac{1}{1+\sigma_i\delta\gamma_i}\left(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\right) + \frac{\sigma_i\delta\gamma_i}{1+\sigma_i\delta\gamma_i}\tilde{\alpha}_i^{(r)}\right). \end{aligned} \tag{52}$$

We also examine the number of inner iterations determined by Γ_δ and how to set the step sizes. If we choose $\delta = \sqrt{\frac{\kappa_{\text{rand}}}{1+m}} - 1$, then Γ_δ in (49) becomes

$$\Gamma_\delta = n\left(1 + \frac{9\kappa_{\text{rand}}}{2(1+\delta)^2}\right) + mn = n\left(1 + \frac{9\kappa_{\text{rand}}}{2\kappa_{\text{rand}}/(m+1)}\right) + mn = 5.5(m+1)n.$$

Therefore a small constant number of passes is sufficient within each round. Using the uniform sampling, the step sizes can be estimated as follows:

$$\sigma_i = \frac{1}{2(1+\delta)\gamma_i(p_i\Gamma_\delta - 1)} \approx \frac{1}{2\sqrt{\kappa_{\text{rand}}/m}\gamma_i(5.5n-1)} \approx \frac{1}{11\gamma_i n\sqrt{\kappa_{\text{rand}}/m}}, \tag{53}$$

$$\tau_k = \frac{1}{2(1+\delta)\lambda(q_k\Gamma_\delta - 1)} \approx \frac{1}{2\sqrt{\kappa_{\text{rand}}/m}\lambda(5.5m-1)} \approx \frac{1}{11\lambda\sqrt{m\cdot\kappa_{\text{rand}}}}. \tag{54}$$

As shown by our numerical experiments in Section 8, the step sizes can be set much larger in practice.

6. Conjugate-Free DSCOVR Algorithms

A major disadvantage of primal-dual algorithms for solving problem (1) is the requirement of computing the proximal mapping of the conjugate function f_i^*, which may not admit closed-formed solution or efficient computation. This is especially the case for logistic regression, one of the most popular loss functions used in classification.

Lan and Zhou (2015) developed "conjugate-free" variants of primal-dual algorithms that avoid computing the proximal mapping of the conjugate functions. The main idea is to replace the Euclidean distance in the dual proximal mapping with a Bregman divergence defined over the conjugate function itself. This technique has been used by Wang and Xiao (2017) to solve structured ERM problems with primal-dual first order methods. Here we use this approach to derive conjugate-free DSCOVR algorithms. In particular, we replace the proximal mapping for the dual update

$$\alpha_i^{(t+1)} = \mathbf{prox}_{\sigma_i f_i^*}\big(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\big) = \underset{\alpha_i \in \mathbf{R}^{n_i}}{\arg\min}\Big\{f_i^*(\alpha_i) - \big\langle \alpha_i,\, u_i^{(t+1)}\big\rangle + \frac{1}{2\sigma_i}\big\|\alpha_i - \alpha_i^{(t)}\big\|^2\Big\},$$

by

$$\alpha_i^{(t+1)} = \underset{\alpha_i \in \mathbf{R}^{n_i}}{\arg\min}\left\{f_i^*(\alpha_i) - \big\langle \alpha_i,\, u_i^{(t+1)}\big\rangle + \frac{1}{\sigma_i}\mathcal{B}_i\big(\alpha_i, \alpha_i^{(t)}\big)\right\}, \tag{55}$$

where $\mathcal{B}_i(\alpha_i, \alpha_i^{(t)}) = f_i^*(\alpha_i) - \big\langle \nabla f_i^*(\alpha_i^{(t)}),\, \alpha_i - \alpha_i^{(t)}\big\rangle$. The solution to (55) is given by

$$\alpha_i^{(t+1)} = \nabla f_i\big(\beta_i^{(t+1)}\big),$$

where $\beta_i^{(t+1)}$ can be computed recursively by

$$\beta_i^{(t+1)} = \frac{\beta_i^{(t)} + \sigma_i u_i^{(t+1)}}{1 + \sigma_i}, \qquad t \geq 0,$$

with initial condition $\beta_i^{(0)} = \nabla f_i^*(\alpha_i^{(0)})$ (see Lan and Zhou, 2015, Lemma 1). Therefore, in order to update the dual variables α_i, we do not need to compute the proximal mapping for the conjugate function f_i^*; instead, taking the gradient of f_i at some easy-to-compute points is sufficient. This conjugate-free update can be applied in Algorithms 1, 2 and 3.

For the accelerated DSCOVR algorithms, we repalce (52) by

$$\alpha_i^{(t+1)} = \underset{\alpha_i \in \mathbf{R}^{n_i}}{\arg\min}\left\{f_i^*(\alpha_i) - \big\langle \alpha_i,\, u_i^{(t+1)}\big\rangle + \frac{1}{\sigma_i}\mathcal{B}_i\big(\alpha_i, \alpha_i^{(t)}\big) + \delta\gamma\mathcal{B}_i\big(\alpha_i, \tilde{\alpha}_i^{(t+1)}\big)\right\}.$$

The solution to the above minimization problem can also be written as

$$\alpha_i^{(t+1)} = \nabla f_i\big(\beta_i^{(t+1)}\big),$$

where $\beta_i^{(t+1)}$ can be computed recursively as

$$\beta_i^{(t+1)} = \frac{\beta_t^{(t)} + \sigma_i u_i^{(t+1)} + \sigma_i\delta\gamma\tilde{\beta}_i}{1 + \sigma_i + \sigma_i\delta\gamma}, \qquad t \geq 0,$$

with the initialization $\beta_i^{(0)} = \nabla f_i^*\big(\alpha_i^{(0)}\big)$ and $\tilde{\beta}_i = \nabla f_i^*\big(\tilde{\alpha}_i^{(r)}\big)$.

The convergence rates and computational complexities of the conjugate-free DSCOVR algorithms are very similar to the ones given in Sections 3–5. We omit details here, but refer the readers to Lan and Zhou (2015) and Wang and Xiao (2017) for related results.

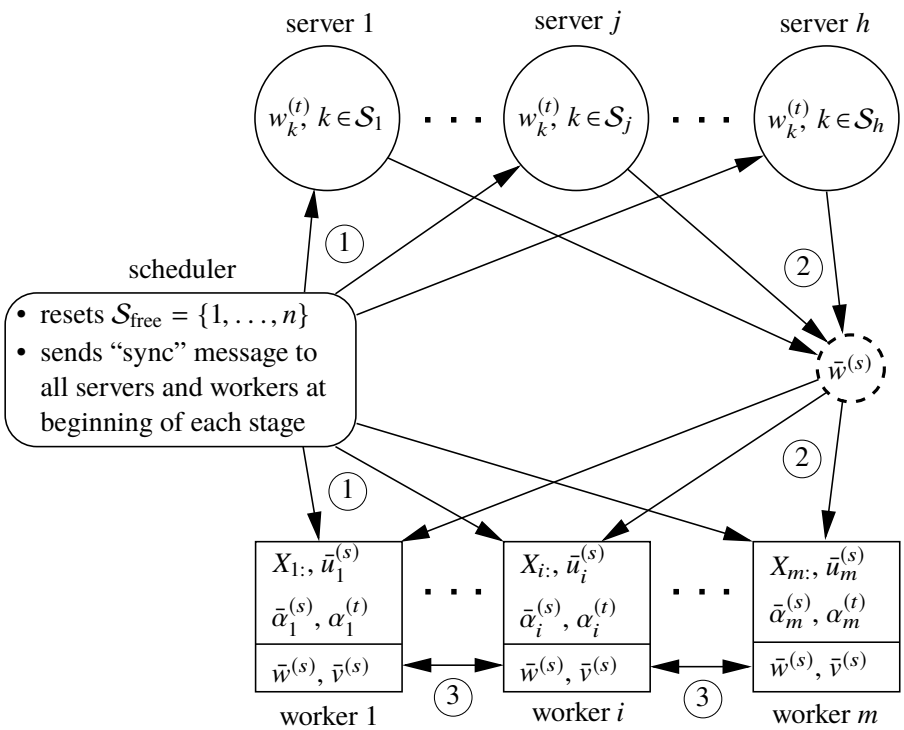

Figure 3: A distributed system for implementing DSCOVR consists of m workers, h parameter servers, and one scheduler. The arrows labeled with the numbers 1, 2 and 3 represent three collective communications at the beginning of each stage in DSCOVR-SVRG.

7. Asynchronous Distributed Implementation

In this section, we show how to implement the DSCOVR algorithms presented in Sections 3–6 in a distributed computing system. We assume that the system provide both synchronous collective communication and asynchronous point-to-point communication, which are all supported by the MPI standard (MPI Forum, 2012). Throughout this section, we assume $m < n$ (see Figure 2).

7.1 Implementation of DSCOVR-SVRG

In order to implement Algorithm 2, the distributed system need to have the following components (see Figure 3):

- *m workers*. Each worker i, for $i = 1, \ldots, m$, stores the following local data and variables :
  - data matrix $X_{i:} \in \mathbf{R}^{N_i \times d}$.
  - vectors in $\mathbf{R}^{N_i}$: $\bar{u}_i^{(s)}$, $\alpha_i^{(t)}$, $\bar{\alpha}_i^{(s)}$.
  - vectors in $\mathbf{R}^d$: $\bar{w}^{(s)}$, $\bar{v}^{(s)}$.
  - extra buffers for computation and communication: $u_j^{(t+1)}$, $v_l^{(t+1)}$, $w_l^{(t)}$ and $w_l^{(t+1)}$.

- *h parameter servers*. Each server j stores a subset of the blocks $\left\{w_k^{(t)} \in \mathbf{R}^{d_k} : k \in \mathcal{S}_j\right\}$, where $\mathcal{S}_1, \ldots, \mathcal{S}_h$ form a partition of the set $\{1, \ldots, n\}$.

- *one scheduler*. It maintains a set of block indices $\mathcal{S}_{\text{free}} \subseteq \{1, \ldots, n\}$. At any given time, $\mathcal{S}_{\text{free}}$ contains indices of parameter blocks that are not currently updated by any worker.

The reason for having $h > 1$ servers is not about insufficient storage for parameters, but rather to avoid the communication overload between only one server and all m workers (m can be in hundreds).

At the beginning of each stage s, the following three collective communications take place across the system (illustrated in Figure 3 by arrows with circled labels 1, 2 and 3):

(1) The scheduler sends a "sync" message to all servers and workers, and resets $\mathcal{S}_{\text{free}} = \{1, \ldots, n\}$.

(2) Upon receiving the "sync" message, the servers aggregate their blocks of parameters together to form $\bar{w}^{(s)}$ and send it to all workers (e.g., through the AllReduce operation in MPI).

(3) Upon receiving $\bar{w}^{(s)}$, each worker compute $\bar{u}_i^{(s)} = X_{i:}\bar{w}^{(s)}$ and $(X_{i:})^T \bar{\alpha}_i^{(s)}$, then invoke a collective communication (AllReduce) to compute $\bar{v}^{(s)} = (1/m) \sum_{i=1}^m (X_{i:})^T \bar{\alpha}_i^{(s)}$.

The number of vectors in $\mathbf{R}^d$ sent and received during the above process is $2m$, counting the communications to form $\bar{w}^{(s)}$ and $\bar{v}^{(s)}$ at m workers (ignoring the short "sync" messages).

After the collective communications at the beginning of each stage, all workers start working on the inner iterations of Algorithm 2 in parallel in an asynchronous, event-driven manner. Each worker interacts with the scheduler and the servers in a four-step loop shown in Figure 4. There are always m iterations taking place concurrently (see also Figure 2), each may at a different phase of the four-step loop:

(1) Whenever worker i finishes updating a block k', it sends the pair (i, k') to the scheduler to request for another block to update. At the beginning of each stage, k' is not needed.

(2) When the scheduler receives the pair (i, k'), it randomly choose a block k from the list of free blocks $\mathcal{S}_{\text{free}}$ (which are not currently updated by any worker), looks up for the server j which stores the parameter block $w_k^{(t)}$ (i.e., $\mathcal{S}_j \ni k$), and then send the pair (i, k) to server j. In addition, the scheduler updates the list $\mathcal{S}_{\text{free}}$ by adding k' and deleting k.

(3) When server j receives the pair (i, k), it sends the vector $w_k^{(t)}$ to worker i, and waits for receiving the updated version $w_k^{(t+1)}$ from worker i.

(4) After worker i receives $w_k^{(t)}$, it computes the updates $\alpha_i^{(t)}$ and $w_k^{(t)}$ following steps 6-7 in Algorithm 2, and then send $w_k^{(t+1)}$ back to server j. At last, it assigns the value of k to k' and send the pair (i, k') to the scheduler, requesting the next block to work on.

The amount of point-to-point communication required during the above process is $2d_k$ float numbers, for sending and receiving $w_k^{(t)}$ and $w_k^{(t+1)}$ (we ignore the small messages for sending and receiving (i, k') and (i, k)). Since the blocks are picked randomly, the average amount of communication per iteration is $2d/n$, or equivalent to $2/n$ vectors in $\mathbf{R}^d$. According to Theorem 1, each stage of Algorithm 2 requires $\log(3)\Gamma$ inner iterations; In addition, the discussions above (30) show that we can take $\Gamma = n(1 + (9/2)\kappa_{\text{rand}})$. Therefore, the average amount of point-to-point communication within each stage is $O(\kappa_{\text{rand}})$ vectors in $\mathbf{R}^d$.

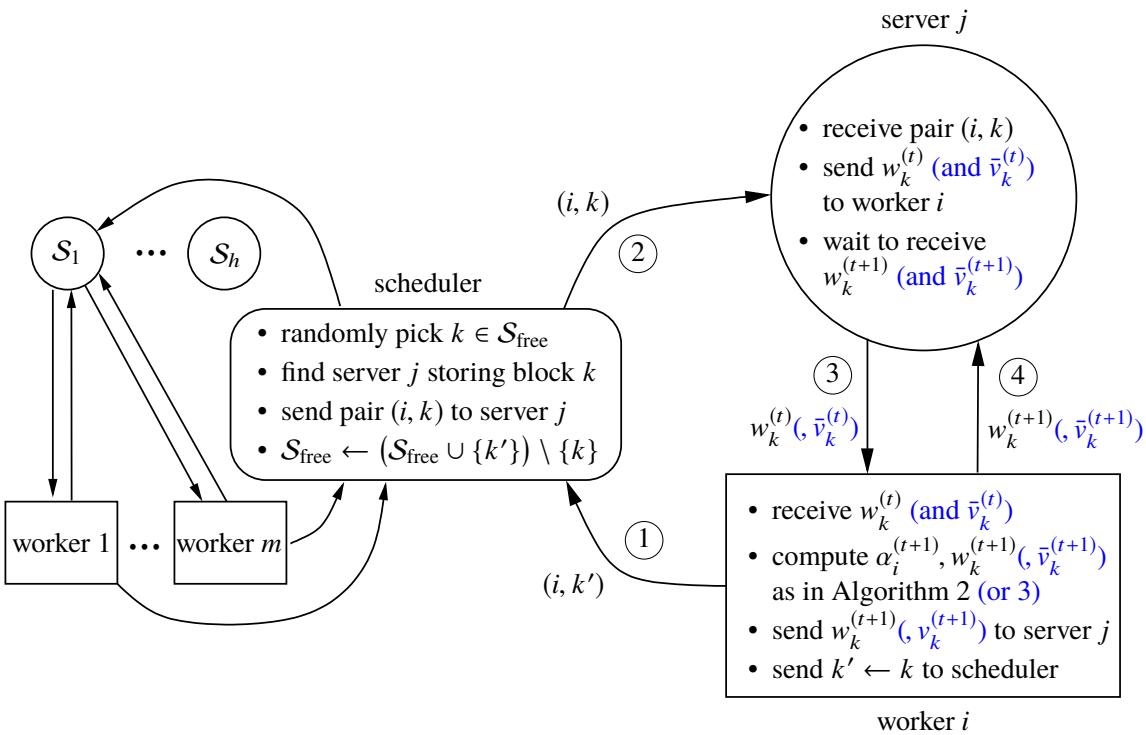

Figure 4: Communication and computation processes for one inner iteration of DSCOVR-SVRG (Algorithm 2). The blue texts in the parentheses are the additional vectors required by DSCOVR-SAGA (Algorithm 3). There are always m iterations taking place in parallel asynchronously, each evolving around one worker. A server may support multiple (or zero) iterations if more than one (or none) of its stored parameter blocks are being updated.

Now we are ready to quantify the communication complexity of DSCOVR-SVRG to find an ϵ-optimal solution. Our discussions above show that each stage requires collective communication of $2m$ vectors in $\mathbf{R}^d$ and asynchronous point-to-point communication of equivalently κ_{rand} such vectors. Since there are total $O(\log(1/\epsilon))$ stages, the total communication complexity is

$$O\left((m + \kappa_{\text{rand}}) \log(1/\epsilon)\right).$$

This gives the communication complexity shown in Table 1, as well as its decomposition in Table 2.

7.2 Implementation of DSCOVR-SAGA

We can implement Algorithm 3 using the same distributed system shown in Figure 3, but with some modifications described below. First, the storage at different components are different:

- *m workers.* Each worker i, for $i = 1, \ldots, m$, stores the following data and variables:
  - data matrix $X_{i:} \in \mathbf{R}^{N_i \times d}$
  - vectors in $\mathbf{R}^{N_i}$: $\alpha_i^{(t)}$, $u_i^{(t)}$, $\bar{u}_i^{(t)}$, and $U_{ik}^{(t)}$ for $k = 1, \ldots, n$.

- vector in $\mathbf{R}^d$: $V_{i:}^{(t)} = \left[V_{i1}^{(t)} \cdots V_{in}^{(t)}\right]^T$ (which is the ith row of $V^{(t)}$, with $V_{ik}^{(t)} \in \mathbf{R}^{1\times d_k}$).
- buffers for communication and update of $w_k^{(t)}$ and $\bar{v}_k^{(t)}$ (both stored at some server).

- *h servers*. Each server j stores a subset of blocks $\left\{w_k^{(t)}, \bar{v}_k^{(t)} \in \mathbf{R}^{d_k} : k \in \mathcal{S}_j\right\}$, for $j = 1, \ldots, n$.
- *one scheduler*. It maintains the set of indices $\mathcal{S}_{\text{free}} \subseteq \{1, \ldots, n\}$, same as in DSCOVR-SVRG.

Unlike DSCOVR-SVRG, there is no stage-wise "sync" messages. All workers and servers work in parallel asynchronously all the time, following the four-step loops illustrated in Figure 4 (including blue colored texts in the parentheses). Within each iteration, the main difference from DSCOVR-SVRG is that, the server and worker need to exchange two vectors of length d_k: $w_k^{(t)}$ and $v_k^{(t)}$ and their updates. This doubles the amount of point-to-point communication, and the average amount of communication per iteration is $4/n$ vectors of length d. Using the iteration complexity in (45), the total amount of communication required (measured by number of vectors of length d) is

$$O\left((m + \kappa_{\text{rand}}) \log(1/\epsilon)\right),$$

which is the same as for DSCOVR-SVRG. However, its decomposition into synchronous and asynchronous communication is different, as shown in Table 2. If the initial vectors $w^{(0)} \neq 0$ or $\alpha^{(0)} \neq 0$, then one round of collective communication is required to propagate the initial conditions to all servers and workers, which reflect the $O(m)$ synchronous communication in Table 2.

7.3 Implementation of Accelerated DSCOVR

Implementation of the accelerated DSCOVR algorithm is very similar to the non-accelerated ones. The main differences lie in the two proximal mappings presented in Section 5.1. In particular, the primal update in (51) needs the extra variable $\tilde{w}_k^{(r)}$, which should be stored at a parameter server together with $w_k^{(t)}$. We modify the four-step loops shown in Figures 4 as follows:

- Each parameter server j stores the extra block parameters $\left\{\tilde{w}_k^{(r)}, k \in \mathcal{S}_j\right\}$. During step (3), $\tilde{w}_k^{(r)}$ is send together with $w_k^{(t)}$ (for SVRG) or $(w_k^{(t)}, v_k^{(t)})$ (for SAGA) to a worker.
- In step (4), no update of $\tilde{w}_k^{(r)}$ is sent back to the server. Instead, whenever switching rounds, the scheduler will inform each server to update their $\tilde{w}_k^{(r)}$ to the most recent $w_k^{(t)}$.

For the dual proximal mapping in (52), each worker i needs to store an extra vector $\tilde{\alpha}_i^{(r)}$, and reset it to the most recent $\alpha_i^{(t)}$ when moving to the next round. There is no need for additional synchronization or collective communication when switching rounds in Algorithm 4. The communication complexity (measured by the number of vectors of length d sent or received) can be obtained by dividing the iteration complexity in (50) by n, i.e., $O\left((m + \sqrt{m\kappa_{\text{rand}}}) \log(1/\epsilon)\right)$, as shown in Table 1.

Finally, in order to implement the conjugate-free DSCOVR algorithms described in Section 6, each worker i simply need to maintain and update an extra vector $\beta_i^{(t)}$ locally.

8. Experiments

In this section, we present numerical experiments on an industrial distributed computing system. This system has hundreds of computers connected by high speed Ethernet in a data center. The hardware

CPU	#cores	RAM	network	operating system
dual Intel® Xeon® processors E5-2650 (v2), 2.6 GHz	16	128 GB 1.8 GHz	10 Gbps Ethernet adapter	Windows® Server (version 2012)

Table 3: Configuration of each machine in the distributed computing system.

and software configurations for each machine are listed in Table 3. We implemented all DSCOVR algorithms presented in this paper, including the SVRG and SAGA versions, their accelerated variants, as well as the conjugate-free algorithms. All implementations are written in C++, using MPI for both collective and point-to-point communications (see Figures 3 and 4 respectively). On each worker machine, we also use OpenMP (OpenMP Architecture Review Board, 2011) to exploit the multi-core architecture for parallel computing, including sparse matrix-vector multiplications and vectorized function evaluations.

Implementing the DSCOVR algorithms requires $m + h + 1$ machines, among them m are workers with local datasets, h are parameter servers, and one is a scheduler (see Figure 3). We focus on solving the ERM problem (3), where the total of N training examples are evenly partitioned and stored at m workers. We partition the d-dimensional parameters into n subsets of roughly the same size (differ at most by one), where each subset consists of randomly chosen coordinates (without replacement). Then we store the n subsets of parameters on h servers, each getting either $\lfloor n/h \rfloor$ or $\lceil n/h \rceil$ subsets. As described in Section 7, we make the configurations to satisfy $n > m > h \geq 1$.

For DSCOVR-SVRG and DSCOVR-SAGA, the step sizes in (29) are very conservative. In the experiments, we replace the coefficient $1/9$ by two tuning parameter η_{d} and η_{p} for the dual and primal step sizes respectively, i.e.,

$$\sigma_i = \eta_{\mathrm{d}} \frac{\lambda}{R^2} \cdot \frac{m}{N}, \qquad \tau_k = \eta_{\mathrm{p}} \frac{\nu}{R^2}. \tag{56}$$

For the accelerated DSCOVR algorithms, we use $\kappa_{\mathrm{rand}} = R^2/(\lambda\nu)$ as shown in (28) for ERM. Then the step sizes in (53) and (54), with $\gamma_i = (m/N)\nu$ and a generic constant coefficient η, become

$$\sigma_i = \frac{\eta_{\mathrm{d}}}{nR} \sqrt{\frac{m\lambda}{\nu}} \cdot \frac{m}{N}, \qquad \tau_k = \frac{\eta_{\mathrm{p}}}{R} \sqrt{\frac{\nu}{m\lambda}}. \tag{57}$$

For comparison, we also implemented the following first-order methods for solving problem 1:

- PGD: parallel implementation of the Proximal Gradient Descent method (using synchronous collective communication over m machines). We use the adaptive line search procedure proposed in Nesterov (2013), and the exact form used is Algorithm 2 in Lin and Xiao (2015).
- APG: parallel implementation of the Accelerated Proximal Gradient method (Nesterov, 2004, 2013). We use a similar adaptive line search scheme to the one for PGD, and the exact form used (with strong convexity) is Algorithm 4 in Lin and Xiao (2015).
- ADMM: the Alternating Direction Method of Multipliers. We use the regularized consensus version in Boyd et al. (2011, Section 7.1.1). For solving the local optimization problems at each node, we use the SDCA method (Shalev-Shwartz and Zhang, 2013).
- CoCoA+: the adding version of CoCoA in Ma et al. (2015). Following the suggestion in Ma et al. (2017), we use a randomized coordinate descent algorithm (Nesterov, 2012; Richtárik and Takáč, 2014) for solving the local optimization problems.

Dataset	#instances (N)	#features (d)	#nonzeros
`rcv1-train`	677,399	47,236	49,556,258
`webspam`	350,000	16,609,143	1,304,697,446
`splice-site`	50,000,000	11,725,480	166,167,381,622

Table 4: Statistics of three datasets. Each feature vector is normalized to have unit norm.

These four algorithms all require m workers only. Specifically, we use the AllReduce call in MPI for the collective communications so that a separate master machine is not necessary.

We conducted experiments on three binary classification datasets obtained from the collection maintained by Fan and Lin (2011). Table 4 lists their sizes and dimensions. In our experiments, we used two configurations: one with $m = 20$ and $h = 10$ for two relatively small datasets, `rcv1-train` and `webspam`, and the other with $m = 100$ and $h = 20$ for the large dataset `splice-site`.

For `rcv1-train`, we solve the ERM problem (3) with a smoothed hinge loss defined as

$$\phi_j(t) = \begin{cases} 0 & \text{if } y_j t \geq 1, \\ \frac{1}{2} - y_j t & \text{if } y_j t \leq 0, \\ \frac{1}{2}(1 - y_j t)^2 & \text{otherwise,} \end{cases} \qquad \text{and} \qquad \phi_j^*(\beta) = \begin{cases} y_j \beta + \frac{1}{2}\beta^2 & \text{if } -1 \leq y_j \beta \leq 0, \\ +\infty & \text{otherwise.} \end{cases}$$

for $j = 1, \ldots, N$. This loss function is 1-smooth, therefore $\nu = 1$; see discussion above (28). We use the ℓ_2 regularization $g(w) = (\lambda/2)\|w\|^2$. Figures 5 and 6 show the reduction of the primal objective gap $P(w^{(t)}) - P(w^\star)$ by different algorithms, with regularization parameter $\lambda = 10^{-4}$ and $\lambda = 10^{-6}$ respectively. All started from the zero initial point. Here the N examples are randomly shuffled and then divided into m subsets. The labels SVRG and SAGA mean DSCOVR-SVRG and DSCOVR-SAGA, respectively, and A-SVRG and A-SAGA are their accelerated versions.

Since PGD and APG both use adaptive line search, there is no parameter to tune. For ADMM, we manually tuned the penalty parameter ρ (see Boyd et al., 2011, Section 7.1.1) to obtain good performance: $\rho = 10^{-5}$ in Figure 5 and $\rho = 10^{-6}$ in Figure 6. For CoCoA+, two passes over the local datasets using a randomized coordinate descent method are sufficient for solving the local optimization problem (more passes do not give meaningful improvement). For DSCOVR-SVRG and SAGA, we used $\eta_{\mathrm{p}} = \eta_{\mathrm{d}} = 20$ to set the step sizes in (56). For DSCOVR-SVRG, each stage goes through the whole dataset 10 times, i.e., the number of inner iterations in Algorithm 2 is $M = 10mn$. For the accelerated DSCOVR algorithms, better performance are obtained with small periods to update the proximal points and we set it to be every 0.2 passes over the dataset, i.e., $0.2mn$ inner iterations. For accelerated DSCOVR-SVRG, we set the stage period (for variance reduction) to be $M = mn$, which is actually longer than the period for updating the proximal points.

From Figures 5 and 6, we observe that the two distributed algorithms based on model averaging, ADMM and CoCoA+, converges relatively fast in the beginning but becomes very slow in the later stage. Other algorithms demonstrate more consistent linear convergence rates. For $\lambda = 10^{-4}$, the DSCOVR algorithms are very competitive compared with other algorithms. For $\lambda = 10^{-6}$, the non-accelerated DSCOVR algorithms become very slow, even after tuning the step sizes. But the accelerated DSCOVR algorithms are superior in terms of both number of passes over data and wall-clock time (with adjusted step size coefficient $\eta_{\mathrm{p}} = 10$ and $\eta_{\mathrm{d}} = 40$).

For ADMM and CoCoA+, each marker represents the finishing of one iteration. It can be seen that they are mostly evenly spaced in terms of number of passes over data, but have large variations in terms of wall-clock time. The variations in time per iteration are due to resource sharing with

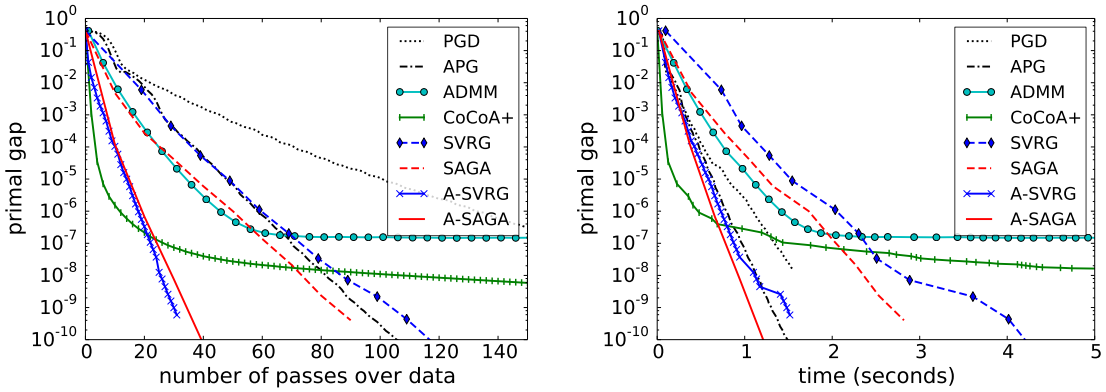

Figure 5: `rcv1-train`: smoothed-hinge loss, $\lambda = 10^{-4}$, randomly shuffled, $m = 20$, $n = 37$, $h = 10$.

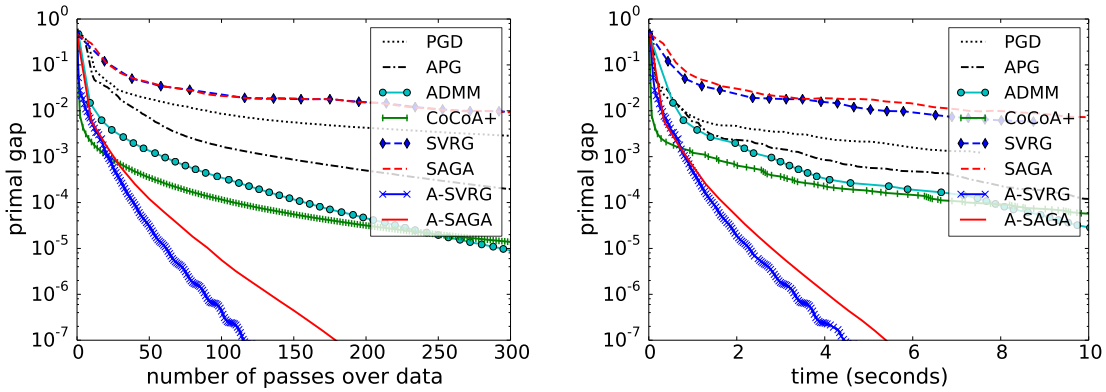

Figure 6: `rcv1-train`: smoothed-hinge loss, $\lambda = 10^{-6}$, randomly shuffled, $m = 20$, $n = 37$, $h = 10$.

other jobs running simultaneously on the distributed computing cluster. Even if we have exclusive use of each machine, sharing communications with other jobs over the Ethernet is unavoidable. This reflects the more realistic environment in cloud computing.

For the `webspam` dataset, we solve the ERM problem with logistic loss $\phi_j(t) = \log(1+\exp(-y_j t))$ where $y_j \in \{\pm 1\}$. The logistic loss is $1/4$-smooth, so we have $\nu = 4$. Since the proximal mapping of its conjugate ϕ_j^* does not have a closed-form solution, we used the conjugate-free DSCOVR algorithms described in Section 6. Figures 7 and 8 shows the reduction of primal objective gap by different algorithms, for $\lambda = 10^{-4}$ and $\lambda = 10^{-6}$ respectively. Here the starting point is no longer the all-zero vectors. Instead, each machine i first computes a local solution by minimizing $f_i(X_i w) + g(w)$, and then compute their average using an AllReduce operation. Each algorithm starts from this average point. This averaging scheme has been proven to be very effective to warm start distributed algorithms for ERM (Zhang et al., 2013). In addition, it can be shown that when starting from the zero initial point, the first step of CoCoA+ computes exactly such an averaged point.

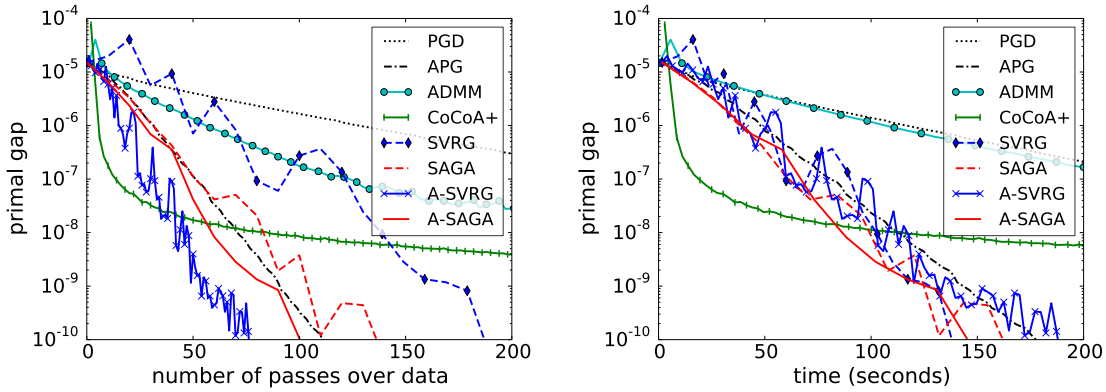

Figure 7: `webspam`: logistic regression, $\lambda = 10^{-4}$, randomly shuffled, $m = 20$, $n = 50$, $h = 10$.

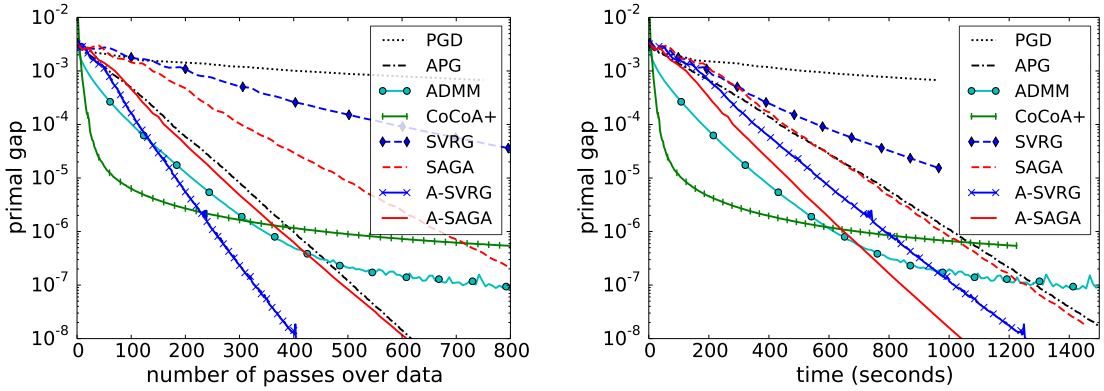

Figure 8: `webspam`: logistic regression, $\lambda = 10^{-6}$, randomly shuffled, $m = 20$, $n = 50$, $h = 10$.

From Figures 7 and 8, we again observe that CoCoA+ has very fast convergence in the beginning but converges very slowly towards higher precision. The DSCOVR algorithms, especially the accelerated variants, are very competitive in terms of both number of iterations and wall-clock time.

In order to investigate the fast initial convergence of CoCoA+ and ADMM, we repeated the experiments on `webspam` without random shuffling. More specifically, we sorted the N examples by their labels, and then partitioned them into m subsets sequentially. That is, most of the machines have data with only $+1$ or -1 labels, and only one machine has mixed ± 1 examples. The results are shown in Figures 9 and 10. Now the fast initial convergence of CoCoA+ and ADMM disappeared. In particular, CoCoA+ converges with very slow linear rate. This shows that statistical properties of random shuffling of the dataset is the main reason for the fast initial convergence of model-averaging based algorithms such as CoCoA+ and ADMM (see, e.g., Zhang et al., 2013).

On the other hand, this should not have any impact on PGD and APG, because their iterations are computed over the whole dataset, which is the same regardless of random shuffling or sorting. The differences between the plots for PGD and APG in Figures 7 and 9 (also for Figures 8 and 10) are due to different initial points computed through averaging local solutions, which does depends on the distribution of data at different machines.

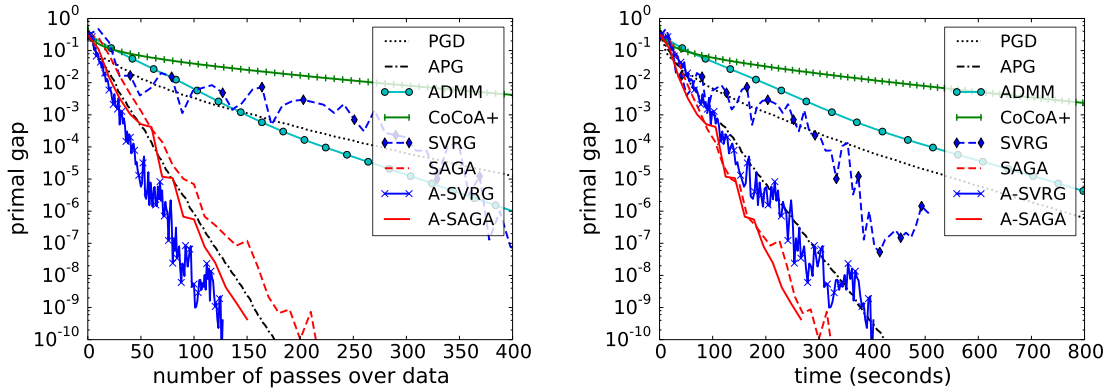

Figure 9: `webspam`: logistic regression, $\lambda = 10^{-4}$, sorted labels, $m = 20$, $n = 50$, $h = 10$.

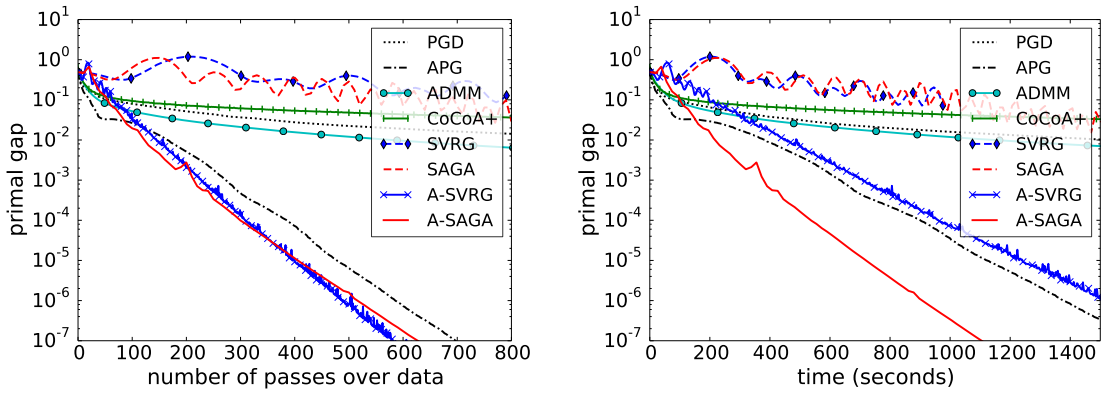

Figure 10: `webspam`: logistic regression, $\lambda = 10^{-6}$, sorted labels, $m = 20$, $n = 50$, $h = 10$.

Different ways for splitting the data over the m workers also affect the DSCOVR algorithms. In particular, the non-accelerated DSCOVR algorithms become very slow, as shown in Figures 9 and 10. However, the accelerated DSCOVR algorithms are still very competitive against the adaptive APG. The accelerated DSCOVR-SAGA algorithm performs best. In fact, the time spent by accelerated DSCOVR-SAGA should be even less than shown in Figures 9 and 10. Recall that other than the initialization with non-zero starting point, DSCOVR-SAGA is completely asynchronous and does not need any collective communication (see Section 7.2). However, in order to record the objective function for the purpose of plotting its progress, we added collective communication and computation to evaluate the objective value for every 10 passes over the data. For example, in Figure 10, such extra collective communications take about 160 seconds (about 15% of total time) for accelerated DSCOVR-SAGA, which can be further deducted from the horizontal time axis.

Finally, we conducted experiments on the `splice-site` dataset with 100 workers and 20 parameter servers. The results are shown in Figure 11. Here the dataset is again randomly shuffled and evenly distributed to the workers. The relative performance of different algorithms are similar to those for the other datasets.

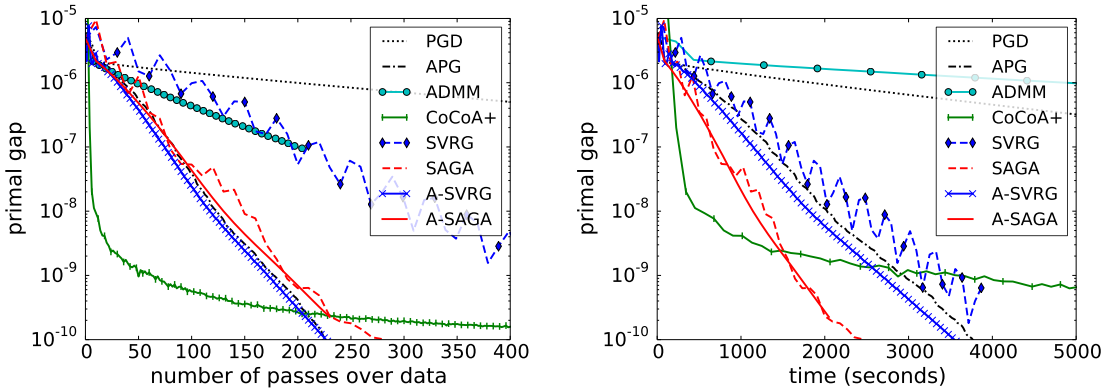

Figure 11: `splice-site`: logistic loss, $\lambda = 10^{-6}$. randomly shuffled, $m = 100$, $n = 150$, $h = 20$.

9. Conclusions and Discussions

We proposed a class of DSCOVR algorithms for asynchronous distributed optimization of large linear models with convex loss functions. They avoid dealing with delays and stale updates in an asynchronous, event-driven environment by exploiting simultaneous data and model parallelism. Compared with other first-order distributed algorithms, DSCOVR may require less amount of overall communication and computation, and especially much less or no synchronization. These conclusions are well supported by our computational experiments on a distributed computing cluster.

We note that there is still some gap between theory and practice. In our theoretical analysis, we assume that the primal and dual block indices in different iterations of DSCOVR are i.i.d. random variables, sampled sequentially with replacement. But the parallel implementations described in Section 7 impose some constraints on how they are generated. In particular, the parameter block to be updated next is randomly chosen from the set of blocks that are not being updated by any worker simultaneously, and the next worker available is event-driven, depending on the loads and processing power of different workers as well as random communication latency. These constraints violate the i.i.d. assumption, but our experiments show that they still lead to very competitive performance. Intuitively some of them can be potentially beneficial, reminiscent of the practical advantage of sampling without replacement over sampling with replacement in randomized coordinate descent methods (e.g., Shalev-Shwartz and Zhang, 2013). This is an interesting topic worth future study.

In our experiments, the parallel implementation of Nesterov's accelerated gradient method (APG) is very competitive on all the datasets we tried and for different regularization parameters used. In addition to the theoretical justifications in Arjevani and Shamir (2015) and Scaman et al. (2017), the adaptive line-search scheme turns out to be critical for its good performance in practice. The accelerated DSCOVR algorithms demonstrated comparable or better performance than APG in the experiments, but need careful tuning of the constants in their step size formula. On one hand, it supports our theoretical results that DSCOVR is capable of outperforming other first-order algorithms including APG, in terms of both communication and computation complexity (see Section 2.1). On the other hand, there are more to be done in order to realize the full potential of DSCOVR in practice. In particular, we plan to follow the ideas in Wang and Xiao (2017) to develop adaptive schemes that can automatically tune the step size parameters, as well as exploit strong convexity from data.

Acknowledgment

Adams Wei Yu is currently supported by NVIDIA PhD Fellowship. The authors would like to thank Chiyuan Zhang for helpful discussion of the system implementation.

Appendix A. Proof of Theorem 1

We first prove two lemmas concerning the primal and dual proximal updates in Algorithm 2. Throughout this appendix, $\mathbf{E}_t[\cdot]$ denotes the conditional expectation taken with respect to the random indices j and l generated during the tth inner iteration in Algorithm 2, conditioned on all quantities available at the beginning of the tth iteration, including $w^{(t)}$ and $\alpha^{(t)}$. Whenever necessary, we also use the notation $j^{(t)}$ and $l^{(t)}$ to denote the random indices generated in the tth iteration.

Lemma 5 *For each $i = 1, \ldots, m$, let $u_i^{(t+1)} \in \mathbf{R}^{N_i}$ be a random variable and define*

$$\tilde{\alpha}_i^{(t+1)} = \mathbf{prox}_{\sigma_i f_i^*}\big(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\big). \tag{58}$$

We choose an index j randomly from $\{1, \ldots, m\}$ with probability distribution $\{p_j\}_{j=1}^m$ and let

$$\alpha_i^{(t+1)} = \begin{cases} \tilde{\alpha}_i^{(t+1)} & \text{if } i = j, \\ \alpha_i^{(t)} & \text{otherwise}. \end{cases}$$

If each $u_i^{(t+1)}$ is independent of j and satisfies $\mathbf{E}_t\left[u_i^{(t+1)}\right] = X_{i:}w^{(t)}$ for $i = 1, \ldots, m$, then we have

$$\begin{aligned} & \sum_{i=1}^m \left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right) - \gamma_i\right) \|\alpha_i^{(t)} - \alpha_i^\star\|^2 \\ \geq\ & \sum_{i=1}^m \frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right) \mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \sum_{i=1}^m \frac{1}{p_i}\left(\frac{1}{2\sigma_i} - \frac{1}{a_i}\right) \mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] \\ & - \sum_{i=1}^m \frac{a_i}{4} \mathbf{E}_t\left[\|u_i^{(t+1)} - X_{i:}w^{(t)}\|^2\right] + \left\langle w^{(t)} - w^\star,\ X^T(\alpha^\star - \alpha^{(t)})\right\rangle \\ & - \sum_{i=1}^m \frac{1}{p_i} \mathbf{E}_t\left[\left\langle \alpha_i^{(t+1)} - \alpha_i^{(t)},\ X_{i:}(w^{(t)} - w^\star)\right\rangle\right], \end{aligned} \tag{59}$$

where $(w^\star, \alpha^\star)$ is the saddle point of $L(w, \alpha)$ defined in (5), and the a_i's are arbitrary positive numbers.

Proof First, consider a fixed index $i \in \{1, \ldots, m\}$. The definition of $\tilde{\alpha}_i^{(t+1)}$ in (58) is equivalent to

$$\tilde{\alpha}_i^{(t+1)} = \operatorname*{arg\,min}_{\beta \in \mathbf{R}^{N_i}} \left\{ f_i^*(\beta) - \left\langle \beta, u_i^{(t+1)}\right\rangle + \frac{\|\beta - \alpha_i^{(t)}\|^2}{2\sigma_i} \right\}. \tag{60}$$

By assumption, $f_i^*(\beta)$ and $\frac{1}{2\sigma_i}\|\beta - \alpha_i^{(t)}\|^2$ are strongly convex with convexity parameters γ_i and $\frac{1}{\sigma_i}$ respectively. Therefore, the objective function in (60) is $(\frac{1}{\sigma_i} + \gamma_i)$-strongly convex, which implies

$$\begin{aligned} & \frac{\|\alpha_i^\star - \alpha_i^{(t)}\|^2}{2\sigma_i} - \left\langle \alpha_i^\star,\ u_i^{(t+1)}\right\rangle + f_i^*(\alpha_i^\star) \\ \geq\ & \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2}{2\sigma_i} - \left\langle \tilde{\alpha}_i^{(t+1)},\ u_i^{(t+1)}\right\rangle + f_i^*(\tilde{\alpha}_i^{(t+1)}) + \left(\frac{1}{\sigma_i} + \gamma_i\right) \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2}{2}. \end{aligned} \tag{61}$$

In addition, since $(w^\star, \alpha^\star)$ is the saddle-point of $L(w, \alpha)$, the function $f_i^*(\alpha_i) - \langle \alpha_i, X_{i:} w^\star \rangle$ is γ_i-strongly convex in α_i and attains its minimum at $\alpha_i^\star$. Thus we have

$$f_i^*\big(\tilde{\alpha}_i^{(t+1)}\big) - \big\langle \tilde{\alpha}_i^{(t+1)},\, X_{i:} w^\star \big\rangle \geq f_i^*(\alpha_i^\star) - \big\langle \alpha_i^\star,\, X_{i:} w^\star \big\rangle + \frac{\gamma_i}{2} \|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2.$$

Summing up the above two inequalities gives

$$\begin{aligned}
\frac{\|\alpha_i^\star - \alpha_i^{(t)}\|^2}{2\sigma_i} &\geq \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2}{2\sigma_i} + \Big(\frac{1}{2\sigma_i} + \gamma_i\Big)\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2 + \Big\langle \alpha_i^\star - \tilde{\alpha}_i^{(t+1)},\, u_i^{(t+1)} - X_{i:} w^\star \Big\rangle \\
&= \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2}{2\sigma_i} + \Big(\frac{1}{2\sigma_i} + \gamma_i\Big)\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2 + \Big\langle \alpha_i^\star - \tilde{\alpha}_i^{(t+1)},\, X_{i:}(w^{(t)} - w^\star) \Big\rangle \\
&\quad + \Big\langle \alpha_i^\star - \alpha_i^{(t)},\, u_i^{(t+1)} - X_{i:} w^{(t)} \Big\rangle + \Big\langle \alpha_i^{(t)} - \tilde{\alpha}_i^{(t+1)},\, u_i^{(t+1)} - X_{i:} w^{(t)} \Big\rangle \\
&\geq \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2}{2\sigma_i} + \Big(\frac{1}{2\sigma_i} + \gamma_i\Big)\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2 + \Big\langle \alpha_i^\star - \tilde{\alpha}_i^{(t+1)},\, X_{i:}(w^{(t)} - w^\star) \Big\rangle \\
&\quad + \Big\langle \alpha_i^\star - \alpha_i^{(t)},\, u_i^{(t+1)} - X_{i:} w^{(t)} \Big\rangle - \frac{\|\alpha_i^{(t)} - \tilde{\alpha}_i^{(t+1)}\|^2}{a_i} - \frac{a_i \|u_i^{(t+1)} - X_{i:} w^{(t)}\|^2}{4},
\end{aligned}$$

where in the last step we used Young's inequality with a_i being an arbitrary positive number. Taking conditional expectation $\mathbf{E}_t$ on both sides of the above inequality, and using the assumption $\mathbf{E}_t[u_i^{(t+1)}] = X_{i:} w^{(t)}$, we have

$$\begin{aligned}
\frac{\|\alpha_i^\star - \alpha_i^{(t)}\|^2}{2\sigma_i} \geq{}& \frac{\mathbf{E}_t\big[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2\big]}{2\sigma_i} + \Big(\frac{1}{2\sigma_i} + \gamma_i\Big)\mathbf{E}_t[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2] + \mathbf{E}_t\Big[\Big\langle \alpha_i^\star - \tilde{\alpha}_i^{(t+1)}, X_{i:}(w^{(t)} - w^\star) \Big\rangle\Big] \\
&- \frac{\mathbf{E}_t[\|\alpha_i^{(t)} - \tilde{\alpha}_i^{(t+1)}\|^2]}{a_i} - \frac{a_i \mathbf{E}_t[\|u_i^{(t+1)} - X_{i:} w^{(t)}\|^2]}{4}. \qquad (62)
\end{aligned}$$

Notice that each $\tilde{\alpha}_i^{(t+1)}$ depends on the random variable $u_i^{(t+1)}$ and is independent of the random index j. But $\alpha_i^{(t+1)}$ depends on both $u_i^{(t+1)}$ and j. Using the law of total expectation,

$$\mathbf{E}_t[\,\cdot\,] \;=\; \mathbf{P}(j = i)\mathbf{E}_t[\,\cdot\,|j = i] + \mathbf{P}(j \neq i)\mathbf{E}_t[\,\cdot\,|j \neq i],$$

we obtain

$$\begin{aligned}
\mathbf{E}_t[\alpha_i^{(t+1)}] &= p_i \mathbf{E}_t[\tilde{\alpha}_i^{(t+1)}] + (1 - p_i)\alpha_i^{(t)}, && (63) \\
\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] &= p_i \mathbf{E}_t[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2], && (64) \\
\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] &= p_i \mathbf{E}_t[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2] + (1 - p_i)\mathbf{E}_t[\|\alpha_i^{(t)} - \alpha_i^\star\|^2]. && (65)
\end{aligned}$$

Next, using the equalities (63), (64) and (65), we can replace each term in (62) containing $\tilde{\alpha}_i^{(t+1)}$ with terms that contain only $\alpha_i^{(t)}$ and $\alpha_i^{(t+1)}$. By doing so and rearranging terms afterwards, we obtain

$$\begin{aligned}
& \left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)-\gamma_i\right)\|\alpha_i^{(t)}-\alpha_i^\star\|^2 \\
\geq\ & \frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2+\frac{1}{p_i}\left(\frac{1}{2\sigma_i}-\frac{1}{a_i}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{(t)}\|^2 \\
& -\frac{a_i\mathbf{E}_t[\|u_i^{(t+1)}-X_{i:}w^{(t)}\|^2]}{4}+\left\langle \alpha_i^\star-\alpha_i^{(t)},\ X_{i:}(w^{(t)}-w^\star)\right\rangle \\
& -\mathbf{E}_t\left[\left\langle \frac{1}{p_i}(\alpha_i^{(t+1)}-\alpha_i^{(t)}),\ X_{i:}(w^{(t)}-w^\star)\right\rangle\right].
\end{aligned}$$

Summing up the above inequality for $i = 1, \ldots, m$ gives the desired result in (59). ■

Lemma 6 *For each $k = 1, \ldots, n$, let $v_k^{(t+1)} \in \mathbf{R}^{d_i}$ be a random variable and define*

$$\tilde{w}_k^{(t+1)} = \mathbf{prox}_{\tau_k g_k}\left(w_k^{(t)} - \tau_k v_k^{(t+1)}\right).$$

We choose an index l randomly from $\{1, \ldots, n\}$ with probability distribution $\{q_l\}_{l=1}^n$ and let

$$w_k^{(t+1)} = \begin{cases} \tilde{w}_k^{(t+1)} & \text{if } k = l, \\ w_k^{(t)} & \text{otherwise}. \end{cases}$$

If each $v_k^{(t+1)}$ is independent of l and satisfies $\mathbf{E}_t\left[v_k^{(t+1)}\right] = \frac{1}{m}(X_{:k})^T\alpha^{(t)}$, then we have

$$\begin{aligned}
& \sum_{k=1}^n\left(\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)-\lambda\right)\|w_k^{(t)}-w_k^\star\|^2 \\
\geq\ & \sum_{k=1}^n\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]+\sum_{k=1}^n\frac{1}{q_k}\left(\frac{1}{2\tau_k}-\frac{1}{b_k}\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^{(t)}\|^2] \\
& -\sum_{k=1}^n\frac{b_k}{4}\mathbf{E}_t\left[\left\|v_k^{(t+1)}-\frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right]+\frac{1}{m}\left\langle X(w^{(t)}-w^\star),\ \alpha^{(t)}-\alpha^\star\right\rangle \\
& +\sum_{k=1}^n\frac{1}{q_k}\mathbf{E}_t\left[\left\langle w_k^{(t+1)}-w_k^{(t)},\ \frac{1}{m}(X_{:k})^T(\alpha^{(t)}-\alpha^\star)\right\rangle\right], \qquad (66)
\end{aligned}$$

where $(w^\star, \alpha^\star)$ is the saddle point of $L(w, \alpha)$ defined in (5), and the b_i's are arbitrary positive numbers.

Lemma 6 is similar to Lemma 5 and can be proved using the same techniques. Based on these two lemmas, we can prove the following proposition.

Proposition 7 *The t-th iteration within the s-th stage of Algorithm 2 guarantees*

$$\begin{aligned}
&\sum_{i=1}^{m}\frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)-\gamma_i+\sum_{k=1}^{n}\frac{3\tau_k\|X_{ik}\|^2}{mp_i}\right]\|\alpha_i^{(t)}-\alpha_i^\star\|^2+\sum_{i=1}^{m}\sum_{k=1}^{n}\frac{2\tau_k\|X_{ik}\|^2}{m^2p_i}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2\\
&+\sum_{k=1}^{n}\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)-\lambda+\sum_{i=1}^{m}\frac{3\sigma_i\|X_{ik}\|^2}{mq_k}\right]\|w_k^{(t)}-w_k^\star\|^2+\sum_{i=1}^{m}\sum_{k=1}^{n}\frac{2\sigma_i\|X_{ik}\|^2}{mq_k}\|\bar{w}_k^{(s)}-w_k^\star\|^2\\
\geq\ &\sum_{i=1}^{m}\frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t\left[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2\right]+\sum_{k=1}^{n}\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t\left[\|w_k^{(t+1)}-w_k^\star\|^2\right]. \qquad (67)
\end{aligned}$$

Proof Multiplying both sides of the inequality (59) by $\frac{1}{m}$ and adding to the inequality (66) gives

$$\begin{aligned}
&\sum_{i=1}^{m}\frac{1}{m}\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)-\gamma_i\right)\|\alpha_i^{(t)}-\alpha_i^\star\|^2+\sum_{k=1}^{n}\left(\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)-\lambda\right)\|w_k^{(t)}-w_k^\star\|^2\\
\geq\ &\sum_{i=1}^{m}\frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]+\sum_{k=1}^{n}\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]\\
&+\sum_{i=1}^{m}\frac{1}{mp_i}\left(\frac{1}{2\sigma_i}-\frac{1}{a_i}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{(t)}\|^2]+\sum_{k=1}^{n}\frac{1}{q_k}\left(\frac{1}{2\tau_k}-\frac{1}{b_k}\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^{(t)}\|^2]\\
&-\sum_{k=1}^{n}\frac{b_k}{4}\mathbf{E}_t\left[\left\|v_k^{(t+1)}-\frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right]+\sum_{k=1}^{n}\frac{1}{q_k}\mathbf{E}_t\left[\left\langle w_k^{(t+1)}-w_k^{(t)},\ \frac{1}{m}(X_{:k})^T(\alpha^{(t)}-\alpha^\star)\right\rangle\right]\\
&-\sum_{i=1}^{m}\frac{a_i}{4m}\mathbf{E}_t[\|u_i^{(t+1)}-X_{i:}w^{(t)}\|^2]-\sum_{i=1}^{m}\frac{1}{mp_i}\mathbf{E}_t\left[\left\langle\alpha_i^{(t+1)}-\alpha_i^{(t)},\ X_{i:}(w^{(t)}-w^\star)\right\rangle\right]. \qquad (68)
\end{aligned}$$

We notice that the terms containing $\frac{1}{m}\left\langle X(w^{(t)}-w^\star),\ \alpha^{(t)}-\alpha^\star\right\rangle$ from (59) and (66) canceled each other. Next we bound the last four terms on the right-hand side of (68).

As in Algorithm 2, for each $i=1,\ldots,m$, we define a random variable

$$u_i^{(t+1)}=\bar{u}_i^{(s)}-\frac{1}{q_l}X_{il}\bar{w}_l^{(s)}+\frac{1}{q_l}X_{il}w_l^{(t)},$$

which depends on the random index $l\in\{1,\ldots,n\}$. Taking expectation with respect to l yields

$$\mathbf{E}_t[u_i^{(t+1)}]=\sum_{k=1}^{n}q_k\left(\bar{u}_i^{(s)}-\frac{1}{q_k}X_{ik}\bar{w}_k^{(s)}+\frac{1}{q_k}X_{ik}w_k^{(t)}\right)=X_{i:}w^{(t)},\qquad i=1,2,\ldots,m.$$

Therefore $u_i^{(t+1)}$ satisfies the assumption in Lemma 5. In order to bound its variance, we notice that

$$\sum_{k=1}^{n}q_k\left(\frac{1}{q_k}X_{ik}\bar{w}_k^{(s)}-\frac{1}{q_k}X_{ik}w_k^{(t)}\right)=X_{i:}\bar{w}^{(s)}-X_{i:}w^{(t)}=\bar{u}_i^{(s)}-X_{i:}w^{(t)}.$$

Using the relation between variance and the second moment, we have

$$\begin{aligned}
\mathbf{E}_t\left[\|u_i^{(t+1)} - X_{i:}w^{(t)}\|^2\right] &= \sum_{k=1}^{n} q_k \left\|\bar{u}_i^{(s)} - \frac{1}{q_k}X_{ik}\bar{w}_k^{(s)} + \frac{1}{q_k}X_{ik}w_k^{(t)} - X_{i:}w^{(t)}\right\|^2 \\
&= \sum_{k=1}^{n} \frac{1}{q_k}\left\|X_{ik}\bar{w}_k^{(s)} - X_{ik}w_k^{(t)}\right\|^2 - \|\bar{u}_i^{(s)} - X_{i:}w^{(t)}\|^2 \\
&\leq \sum_{k=1}^{n} \frac{1}{q_k}\left\|X_{ik}(\bar{w}_k^{(s)} - w_k^{(t)})\right\|^2 \\
&\leq \sum_{k=1}^{n} \frac{2\|X_{ik}\|^2}{q_k}\left(\|\bar{w}_k^{(s)} - w_k^{\star}\|^2 + \|w_k^{(t)} - w_k^{\star}\|^2\right). \qquad (69)
\end{aligned}$$

Similarly, for $k = 1, \ldots, n$, we have

$$\mathbf{E}_t[v_k^{(t+1)}] = \sum_{i=1}^{m} p_i \left(\bar{v}_k^{(s)} - \frac{1}{p_i}\frac{1}{m}(X_{ik})^T\bar{\alpha}_i^{(s)} + \frac{1}{p_i}\frac{1}{m}(X_{ik})^T\alpha_i^{(t)}\right) = \frac{1}{m}(X_{:k})^T\alpha^{(t)}.$$

Therefore $v_k^{(t+1)}$ satisfies the assumption in Lemma 6. Furthermore, we have

$$\begin{aligned}
\mathbf{E}_t\left[\left\|v_k^{(t+1)} - \frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right] &= \sum_{i=1}^{m} p_i\left\|\bar{v}_k^{(s)} - \frac{1}{p_i}\frac{1}{m}(X_{ik})^T\bar{\alpha}_i^{(s)} + \frac{1}{p_i}\frac{1}{m}(X_{ik})^T\alpha_i^{(t)} - \frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2 \\
&= \sum_{i=1}^{m} \frac{1}{p_i}\left\|\frac{1}{m}(X_{ik})^T\bar{\alpha}_i^{(s)} - \frac{1}{m}(X_{ik})^T\alpha_i^{(t)}\right\|^2 - \left\|\bar{v}_k^{(s)} - \frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2 \\
&\leq \sum_{i=1}^{m} \frac{1}{p_i}\left\|\frac{1}{m}(X_{ik})^T\left(\bar{\alpha}_i^{(s)} - \alpha_i^{(t)}\right)\right\|^2 \\
&\leq \sum_{i=1}^{m} \frac{2\|X_{ik}\|^2}{p_i m^2}\left(\|\bar{\alpha}_i^{(s)} - \alpha_i^{\star}\|^2 + \|\alpha_i^{(t)} - \alpha_i^{\star}\|^2\right). \qquad (70)
\end{aligned}$$

Now we consider the two terms containing inner products in (68). Using the conditional expectation relation (63), we have

$$\begin{aligned}
\mathbf{E}_t\left[-\left\langle \alpha_i^{(t+1)} - \alpha_i^{(t)},\, X_{i:}(w^{(t)} - w^{\star})\right\rangle\right] &= p_i\mathbf{E}_t\left[-\left\langle \tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)},\, X_{i:}(w^{(t)} - w^{\star})\right\rangle\right] \\
&\geq p_i\mathbf{E}_t\left[-\frac{1}{c_i}\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2 - \frac{c_i}{4}\|X_{i:}(w^{(t)} - w^{\star})\|^2\right] \\
&= -\frac{p_i}{c_i}\mathbf{E}_t\left[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2\right] - \frac{c_i p_i}{4}\|X_{i:}(w^{(t)} - w^{\star})\|^2 \\
&= -\frac{1}{c_i}\mathbf{E}_t\left[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2\right] - \frac{c_i p_i}{4}\|X_{i:}(w^{(t)} - w^{\star})\|^2, \qquad (71)
\end{aligned}$$

where we used Young's inequality with c_i being an arbitrary positive number, and the last equality used (64). We note that for any n vectors $z_1, \ldots, z_n \in \mathbf{R}^{N_i}$, it holds that

$$\left\|\sum_{k=1}^{n} z_k\right\|^2 \leq \sum_{k=1}^{n} \frac{1}{q_k}\|z_k\|^2.$$

To see this, we let $z_{k,j}$ denote the jth component of z_k and use the Cauchy-Schwarz inequality:

$$\begin{aligned}
\left\|\sum_{k=1}^{n} z_k\right\|^2 &= \sum_{j=1}^{N_i}\left(\sum_{k=1}^{n} z_{k,j}\right)^2 = \sum_{j=1}^{N_i}\left(\sum_{k=1}^{n}\frac{z_{k,j}}{\sqrt{q_k}}\sqrt{q_k}\right)^2 \\
&\le \sum_{j=1}^{N_i}\left(\sum_{k=1}^{n}\left(\frac{z_{k,j}}{\sqrt{q_k}}\right)^2\right)\left(\sum_{k=1}^{n}(\sqrt{q_k})^2\right) \\
&= \sum_{j=1}^{N_i}\left(\sum_{k=1}^{n}\frac{z_{k,j}^2}{q_k}\right) = \sum_{k=1}^{n}\frac{1}{q_k}\sum_{j=1}^{N_i} z_{k,j}^2 = \sum_{k=1}^{n}\frac{1}{q_k}\|z_k\|^2.
\end{aligned}$$

Applying this inequality to the vector $X_{i:}(w^{(t)} - w^\star) = \sum_{k=1}^{n} X_{ik}(w_k^{(t)} - w_k^\star)$, we get

$$\|X_{i:}(w^{(t)} - w^\star)\|^2 \le \sum_{k=1}^{n}\frac{1}{q_k}\|X_{ik}(w_k^{(t)} - w_k^\star)\|^2.$$

Therefore we can continue the inequality (71), for each $i = 1, \ldots, m$, as

$$\begin{aligned}
&\mathbf{E}_t\left[-\left\langle \alpha_i^{(t+1)} - \alpha_i^{(t)},\ X_{i:}(w^{(t)} - w^\star)\right\rangle\right] \\
\ge\ & -\frac{1}{c_i}\mathbf{E}_t\left[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2\right] - \frac{c_i p_i}{4}\sum_{k=1}^{n}\frac{1}{q_k}\|X_{ik}(w_k^{(t)} - w_k^\star)\|^2 \\
\ge\ & -\frac{1}{c_i}\mathbf{E}_t\left[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2\right] - \frac{c_i p_i}{4}\sum_{k=1}^{n}\frac{1}{q_k}\|X_{ik}\|^2\|w_k^{(t)} - w_k^\star\|^2. \qquad (72)
\end{aligned}$$

Using similarly arguments, we can obtain, for each $k = 1, \ldots, n$ and arbitrary $h_k > 0$,

$$\begin{aligned}
&\mathbf{E}_t\left[\left\langle w_k^{(t+1)} - w_k^{(t)},\ \frac{1}{m}(X_{:k})^T(\alpha^{(t)} - \alpha^\star)\right\rangle\right] \\
\ge\ & -\frac{1}{h_k}\mathbf{E}_t\left[\|w_k^{(t+1)} - w_k^{(t)}\|^2\right] - \frac{h_k q_k}{4m^2}\sum_{i=1}^{m}\frac{1}{p_i}\|X_{ik}\|^2\|\alpha_i^{(t)} - \alpha_i^\star\|^2. \qquad (73)
\end{aligned}$$

Applying the bounds in (69), (70), (72) and (73) to (68) and rearranging terms, we have

$$\begin{aligned}
&\sum_{i=1}^{m}\frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right) - \gamma_i + \sum_{k=1}^{n}\frac{b_k\|X_{ik}\|^2}{2mp_i} + \sum_{k=1}^{n}\frac{h_k\|X_{ik}\|^2}{4mp_i}\right]\|\alpha_i^{(t)} - \alpha_i^\star\|^2 \\
&+ \sum_{k=1}^{n}\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right) - \lambda + \sum_{i=1}^{m}\frac{a_i\|X_{ik}\|^2}{2mq_k} + \sum_{i=1}^{m}\frac{c_i\|X_{ik}\|^2}{4mq_k}\right]\|w_k^{(t)} - w_k^\star\|^2 \\
&+ \sum_{i=1}^{m}\sum_{k=1}^{n}\frac{b_k\|X_{ik}\|^2}{2m^2p_i}\|\bar{\alpha}_i^{(s)} - \alpha_i^\star\|^2 + \sum_{i=1}^{m}\sum_{k=1}^{n}\frac{a_i\|X_{ik}\|^2}{2mq_k}\|\bar{w}_k^{(s)} - w_k^\star\|^2 \\
\ge\ & \sum_{i=1}^{m}\frac{1}{mp_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \sum_{k=1}^{n}\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2] \\
&+ \sum_{i=1}^{m}\frac{1}{mp_i}\left(\frac{1}{2\sigma_i} - \frac{1}{a_i} - \frac{1}{c_i}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] \\
&+ \sum_{k=1}^{n}\frac{1}{q_k}\left(\frac{1}{2\tau_k} - \frac{1}{b_k} - \frac{1}{h_k}\right)\mathbf{E}_t[\|w_k^{(t+1)} - w_k^{(t)}\|^2].
\end{aligned}$$

The desired result (67) is obtained by choosing $a_i = c_i = 4\sigma_i$ and $b_k = h_k = 4\tau_k$. ■

Finally, we are ready to prove Theorem 1. Let $\theta \in (0,1)$ be a parameter to be determined later, and let Γ and η be two constants such that

$$\Gamma \geq \max_{i,k} \left\{ \frac{1}{p_i}\left(1 + \frac{3\|X_{ik}\|^2}{2\theta q_k \lambda \gamma_i}\right), \frac{1}{q_k}\left(1 + \frac{3n\|X_{ik}\|^2}{2\theta m p_i \lambda \gamma_i}\right) \right\}, \tag{74}$$

$$\eta = 1 - \frac{1-\theta}{\Gamma}. \tag{75}$$

It is easy to check that $\Gamma > 1$ and $\eta \in (0,1)$. By the choices of σ_i and τ_k in (23) and (24) respectively, we have

$$\frac{1}{p_i}\left(1 + \frac{1}{2\sigma_i\gamma_i}\right) = \frac{1}{q_k}\left(1 + \frac{1}{2\tau_k\lambda}\right) = \Gamma, \tag{76}$$

for all $i = 1, \ldots, m$ and $k = 1, \ldots, n$. Comparing the above equality with the definition of Γ in (74), we have

$$\frac{3\|X_{ik}\|^2}{2\theta q_k \lambda \gamma_i} \leq \frac{1}{2\sigma_i\gamma_i} \qquad \text{and} \qquad \frac{3n\|X_{ik}\|^2}{2\theta m p_i \lambda \gamma_i} \leq \frac{1}{2\tau_k\lambda},$$

which implies

$$\frac{3\sigma_i\|X_{ik}\|^2}{q_k} \leq \theta\lambda \qquad \text{and} \qquad \frac{3n\tau_k\|X_{ik}\|^2}{m p_i} \leq \theta\gamma_i,$$

for all $i = 1, \ldots, m$ and $k = 1, \ldots, n$. Therefore, we have

$$\sum_{k=1}^{n} \frac{3\tau_k\|X_{ik}\|^2}{m p_i} = \frac{1}{n}\sum_{k=1}^{n} \frac{3n\tau_k\|X_{ik}\|^2}{m p_i} \leq \frac{1}{n}\sum_{k=1}^{n} \theta\gamma_i = \theta\gamma_i, \qquad i = 1, \ldots, m, \tag{77}$$

$$\sum_{i=1}^{m} \frac{3\sigma_i\|X_{ik}\|^2}{m q_k} = \frac{1}{m}\sum_{i=1}^{m} \frac{3\sigma_i\|X_{ik}\|^2}{q_k} \leq \frac{1}{m}\sum_{i=1}^{m} \theta\lambda = \theta\lambda, \qquad k = 1, \ldots, n. \tag{78}$$

Now we consider the inequality (67), and examine the ratio between the coefficients of $\|\alpha_i^{(t)} - \alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2]$. Using (77) and (76), we have

$$\frac{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right) - \gamma_i + \sum_{k=1}^{n} \frac{3\tau_k\|X_{ik}\|^2}{m p_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)} \leq 1 - \frac{(1-\theta)\gamma_i}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)} = 1 - \frac{1-\theta}{\Gamma} = \eta. \tag{79}$$

Similarly, the ratio between the coefficients of $\|w_k^{(t)} - w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2]$ can be bounded using (78) and (76):

$$\frac{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right) - \lambda + \sum_{i=1}^{m} \frac{3\sigma_i\|X_{ik}\|^2}{m q_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)} \leq 1 - \frac{(1-\theta)\lambda}{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)} = 1 - \frac{1-\theta}{\Gamma} = \eta. \tag{80}$$

In addition, the ratio between the coefficients of $\|\bar{\alpha}_i^{(s)} - \alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2]$ and that of $\|\bar{w}_k^{(s)} - w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2]$ can be bounded as

$$\frac{\sum_{k=1}^k \frac{2\tau_k \|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)} \leq \frac{(2/3)\theta\gamma_i}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)} = \frac{(2/3)\theta}{\Gamma} = \frac{2\theta(1-\eta)}{3(1-\theta)}, \tag{81}$$

$$\frac{\sum_{i=1}^m \frac{3\sigma_i \|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)} \leq \frac{(2/3)\theta\lambda}{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)} = \frac{(2/3)\theta}{\Gamma} = \frac{2\theta(1-\eta)}{3(1-\theta)}. \tag{82}$$

Using (76) and the four inequalities (79), (80), (81) and (82), we conclude that the inequality (67) implies

$$\begin{aligned}
& \eta \sum_{i=1}^m \frac{\Gamma\gamma_i}{m}\|\alpha_i^{(t)} - \alpha_i^\star\|^2 + \frac{2\theta(1-\eta)}{3(1-\theta)}\sum_{i=1}^m \frac{\Gamma\gamma_i}{m}\|\bar{\alpha}_i^{(s)} - \alpha_i^\star\|^2 \\
& +\eta \sum_{k=1}^n \Gamma\lambda\|w_k^{(t)} - w_k^\star\|^2 + \frac{2\theta(1-\eta)}{3(1-\theta)}\sum_{k=1}^n \Gamma\lambda\|\bar{w}_k^{(s)} - w_k^\star\|^2 \\
\geq \quad & \sum_{i=1}^m \frac{\Gamma\gamma_i}{m}\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \sum_{k=1}^n \Gamma\lambda\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2].
\end{aligned}$$

Using the definite of $\Omega(\cdot)$ in (21), the inequality above is equivalent to

$$\begin{aligned}
& \eta\,\Omega\big(w^{(t)} - w^\star, \alpha^{(t)} - \alpha^\star\big) + \frac{2\theta(1-\eta)}{3(1-\theta)}\Omega\big(\bar{w}^{(s)} - w^\star, \bar{\alpha}^{(s)} - \alpha^\star\big) \\
\geq \quad & \mathbf{E}_t\left[\Omega\big(w^{(t+1)} - w^\star, \alpha^{(t+1)} - \alpha^\star\big)\right].
\end{aligned} \tag{83}$$

To simplify further derivation, we define

$$\begin{aligned}
\Delta^{(t)} &= \mathbf{E}\left[\Omega\big(w^{(t)} - w^\star, \alpha^{(t)} - \alpha^\star\big)\right], \\
\bar{\Delta}^{(s)} &= \mathbf{E}\left[\Omega\big(\bar{w}^{(s)} - w^\star, \bar{\alpha}^{(s)} - \alpha^\star\big)\right],
\end{aligned}$$

where the expectation is taken with respect to all randomness in the sth stage, that is, the random variables $\{(j^{(0)}, l^{(0)}), (j^{(1)}, l^{(1)}), \ldots, (j^{(M-1)}, l^{(M-1)})\}$. Then the inequality (83) implies

$$\frac{2\theta(1-\eta)}{3(1-\theta)}\bar{\Delta}^{(s)} + \eta\Delta^{(t)} \geq \Delta^{(t+1)}.$$

Dividing both sides of the above inequality by η^{t+1} gives

$$\frac{2\theta(1-\eta)}{3(1-\theta)}\frac{\bar{\Delta}^{(s)}}{\eta^{t+1}} + \frac{\Delta^{(t)}}{\eta^t} \geq \frac{\Delta^{(t+1)}}{\eta^{t+1}}.$$

Summing for $t = 0, 1, , \ldots, M-1$ gives

$$\left(\frac{1}{\eta} + \frac{1}{\eta^2} + \cdots + \frac{1}{\eta^M}\right)\frac{2\theta(1-\eta)}{3(1-\theta)}\bar{\Delta}^{(s)} + \Delta^{(0)} \geq \frac{\Delta^{(T)}}{\eta^M},$$

which further leads to

$$(1-\eta^M)\frac{2\theta}{3(1-\theta)}\bar{\Delta}^{(s)} + \eta^M \Delta^{(0)} \geq \Delta^{(M)}.$$

Now choosing $\theta = 1/3$ and using the relation $\bar{\Delta}^{(s)} = \Delta^{(0)}$ for each stage, we obtain

$$\left(\frac{1}{3} + \frac{2}{3}\eta^M\right)\Delta^{(0)} \geq \Delta^{(M)}.$$

Therefore if we choose M large enough such that $\eta^M \leq \frac{1}{2}$, then

$$\Delta^{(M)} \leq \frac{2}{3}\Delta^{(0)}, \quad \text{or equivalently,} \quad \bar{\Delta}^{(s+1)} \leq \frac{2}{3}\bar{\Delta}^{(s)}.$$

The condition $\eta^M \leq \frac{1}{2}$ is equivalent to $M \geq \frac{\log(2)}{\log(1/\eta)}$, which can be guaranteed by

$$M \geq \frac{\log(2)}{1-\eta} = \frac{\log(2)}{1-\theta}\Gamma = \frac{3\log(2)}{2}\Gamma = \log(\sqrt{8})\Gamma.$$

To further simplify, it suffices to have $M \geq \log(3)\Gamma$. Finally, we notice that $\bar{\Delta}^{(s+1)} \leq (2/3)\bar{\Delta}^{(s)}$ implies $\bar{\Delta}^{(s)} \leq (2/3)^s\, \bar{\Delta}^{(0)}$. which is the desired result in Theorem 1.

A.1 Alternative bounds and step sizes

Alternatively, we can let Γ to satisfy

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1 + \frac{3\|X_{:k}\|_F^2}{2\theta m q_k \lambda \gamma_i}\right),\ \frac{1}{q_k}\left(1 + \frac{3\|X_{i:}\|_F^2}{2\theta m p_i \lambda \gamma_i}\right)\right\}, \tag{84}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. Then by choosing σ_i and τ_k that satisfy (76), we have

$$\frac{3\sigma_i\|X_{:k}\|_F^2}{mq_k} \leq \theta\lambda \qquad \text{and} \qquad \frac{3\tau_k\|X_{i:}\|_F^2}{mp_i} \leq \theta\gamma_i,$$

We can bound the left-hand sides in (77) and (78) using Hölder's inequality, which results in

$$\sum_{k=1}^{n}\frac{3\tau_k\|X_{ik}\|^2}{mp_i} \leq \sum_{k=1}^{n}\frac{3\tau_k\|X_{ik}\|_F^2}{mp_i} \leq \frac{3\max_k\{\tau_k\}\|X_{i:}\|_F^2}{mp_i} \leq \theta\gamma_i, \qquad i = 1,\ldots,m, \tag{85}$$

$$\sum_{i=1}^{m}\frac{3\sigma_i\|X_{ik}\|^2}{mq_k} \leq \sum_{i=1}^{m}\frac{3\sigma_i\|X_{ik}\|_F^2}{mq_k} \leq \frac{3\max_i\{\sigma_i\}\|X_{:k}\|_F^2}{mq_k} \leq \theta\lambda, \qquad k = 1,\ldots,n. \tag{86}$$

The rest of the proof hold without any change. Setting $\theta = 1/3$ gives the condition on Γ in (26).

In Theorem 1 and the proof above, we choose Γ as a uniform bound over all combinations of (i,k) in order to obtain a uniform convergence rates on all blocks of the primal and dual variables $w_k^{(t)}$ and $\alpha_i^{(t)}$, so we have a simple conclusion as in (25). In practice, we can use different bounds on different blocks and choose step sizes to allow them to converge at different rates.

For example, we can choose the step sizes σ_i and τ_k such that

$$\begin{aligned}
\frac{1}{p_i}\left(1+\frac{1}{2\sigma_i\gamma_i}\right) &= \max_k\left\{\frac{1}{p_i}\left(1+\frac{3\|X_{:k}\|_F^2}{2\theta m q_k\lambda\gamma_i}\right)\right\}, \qquad i=1,\ldots,m,\\
\frac{1}{q_k}\left(1+\frac{1}{2\tau_k\lambda}\right) &= \max_i\left\{\frac{1}{q_k}\left(1+\frac{3\|X_{i:}\|_F^2}{2\theta m p_i\lambda\gamma_i}\right)\right\}, \qquad k=1,\ldots,n.
\end{aligned}$$

Then the inequalities (85) and (86) still hold, and we can still show linear convergence with a similar rate. In this case, the step sizes are chosen as

$$\begin{aligned}
\sigma_i &= \min_k\left\{\frac{\theta m q_k\lambda}{3\|X_{:k}\|_F^2}\right\}, \qquad i=1,\ldots,m,\\
\tau_k &= \min_i\left\{\frac{\theta m p_i\gamma_i}{3\|X_{i:}\|_F^2}\right\}, \qquad k=1,\ldots,n.
\end{aligned}$$

If we choose the probabilities to be proportional to the norms of the data blocks, i.e.,

$$p_i=\frac{\|X_{i:}\|_F^2}{\|X\|_F^2}, \qquad q_k=\frac{\|X_{:k}\|_F^2}{\|X\|_F^2},$$

then we have

$$\sigma_i=\frac{\theta m\lambda}{3\|X\|_F^2}, \qquad \tau_k=\frac{\theta m\gamma_i}{3\|X\|_F^2}.$$

If we further normalize the rows of X, and let R be the norm of each row, then (with $\theta=1/3$)

$$\sigma_i=\frac{\theta\lambda}{3R^2}\frac{m}{N}=\frac{\lambda}{9R^2}\frac{m}{N}, \qquad \tau_k=\frac{\theta\gamma_i}{3R^2}\frac{m}{N}=\frac{\gamma_i}{9R^2}\frac{m}{N}.$$

For distributed ERM, we have $\gamma_i=\frac{N}{m}\gamma$, thus $\tau_k=\frac{\gamma}{9R^2}$ as in (29).

Appendix B. Proof of Theorem 2

Consider the following saddle-point problem with doubly separable structure:

$$\min_{w\in\mathbf{R}^D}\ \max_{\alpha\in\mathbf{R}^N}\left\{L(w,\alpha)\ \equiv\ \frac{1}{m}\sum_{i=1}^m\sum_{k=1}^n\alpha_i^TX_{ik}w_k-\frac{1}{m}\sum_{i=1}^m f_i^*(\alpha_i)+\sum_{k=1}^n g_k(w_k)\right\}. \tag{87}$$

Under Assumption 1, L has a unique saddle point $(w^\star,\alpha^\star)$. We define

$$\widetilde{P}_k(w_k)\ \equiv\ \frac{1}{m}(\alpha^\star)^TX_{:k}w_k+g_k(w_k)-\frac{1}{m}(\alpha^\star)^TX_{:k}w_k^\star-g_k(w_k^\star), \qquad k=1,\ldots,n, \tag{88}$$

$$\widetilde{D}_i(\alpha_i)\ \equiv\ \frac{1}{m}\left(\alpha_i^TX_{i:}w^\star-f_i^*(\alpha_i)-(\alpha_i^\star)^TX_{i:}w^\star+f_i^*(\alpha_i^\star)\right), \qquad i=1,\ldots,m. \tag{89}$$

We note that $w_k^\star$ is the minimizer of $\widetilde{P}_k$ with $\widetilde{P}_k(w_k^\star)=0$ and $\alpha_i^\star$ is the maximizer of $\widetilde{D}_i$ with $\widetilde{D}_i(\alpha_i^\star)=0$. Moreover, by the assumed strong convexity,

$$\widetilde{P}_k(w_k)\ \geq\ \frac{\lambda}{2}\|w_k-w_k^\star\|^2, \qquad \widetilde{D}_i(\alpha_i)\ \leq\ -\frac{\gamma_i}{2m}\|\alpha_i-\alpha_i^\star\|^2. \tag{90}$$

Moreover, we have the following lower bound for the duality gap $P(w) - D(\alpha)$:

$$\sum_{k=1}^{n} \widetilde{P}_k(w_k) - \sum_{i=1}^{m} \widetilde{D}_i(\alpha_i) \;=\; L(w, \alpha^\star) - L(w^\star, \alpha) \;\leq\; P(w) - D(\alpha). \tag{91}$$

We can also use them to derive an upper bound for the duality gap, as in the following lemma.

Lemma 8 *Suppose Assumption 1 holds. Let $(w^\star, \alpha^\star)$ be the saddle-point of $L(w, \alpha)$ and define*

$$P(w) = \sup_{\alpha} L(w, \alpha), \qquad D(\alpha) = \inf_{w} L(w, \alpha).$$

Then we have

$$P(w) - D(\alpha) \;\leq\; L(w, \alpha^\star) - L(w^\star, \alpha) + \left(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\right)\|w - w^\star\|^2 + \frac{\|X\|^2}{2m^2\lambda}\|\alpha - \alpha^\star\|^2.$$

Proof By definition, the primal function can be written as $P(w) = F(w) + g(w)$, where

$$F(w) = \frac{1}{m}\sum_{i=1}^{m} f_i(X_{i:}w) = \frac{1}{m}\max_{\alpha}\left\{\alpha^T X w - \sum_{i=1}^{m} f_i^*(\alpha_i)\right\}.$$

From the optimality conditions satisfied by the saddle point $(w^\star, \alpha^\star)$, we have

$$\nabla F(w^\star) = \frac{1}{m} X^T \alpha^\star.$$

By assumption, $\nabla F(w)$ is Lipschitz continuous with smooth constant $\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{\gamma_i}$, which implies

$$\begin{aligned}
F(w) \;&\leq\; F(w^\star) + \langle \nabla F(w^\star),\, w - w^\star\rangle + \left(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\right)\|w - w^\star\|^2 \\
&=\; \frac{1}{m}\left((\alpha^\star)^T X w^\star - \sum_{i=1}^{m} f_i^*(\alpha_i^\star)\right) + \frac{1}{m}(\alpha^\star)^T X(w - w^\star) + \left(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\right)\|w - w^\star\|^2 \\
&=\; \frac{1}{m}\left((\alpha^\star)^T X w - \sum_{i=1}^{m} f_i^*(\alpha_i^\star)\right) + \left(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\right)\|w - w^\star\|^2.
\end{aligned}$$

Therefore,

$$\begin{aligned}
P(w) \;&=\; F(w) + g(w) \\
&\leq\; \frac{1}{m}(\alpha^\star)^T X w - \frac{1}{m}\sum_{i=1}^{m} f_i^*(\alpha_i^\star) + g(w) + \left(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\right)\|w - w^\star\|^2 \\
&=\; L(w, \alpha^\star) + \left(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\right)\|w - w^\star\|^2.
\end{aligned}$$

Using similar arguments, especially that $\nabla g^*(\alpha)$ has Lipschitz constant $\frac{\|X\|}{m^2\lambda}$, we can show that

$$D(\alpha) \;\geq\; L(w^\star, \alpha) - \frac{\|X\|^2}{2m^2\lambda}\|\alpha - \alpha^\star\|^2.$$

Combining the last two inequalities gives the desired result. ■

The rest of the proof follow similar steps as in the proof of Theorem 1. The next two lemmas are variants of Lemmas 5 and 6.

Lemma 9 *Under the same assumptions and setup in Lemma 5, we have*

$$\begin{aligned}
&\sum_{i=1}^{m}\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)-\frac{\gamma_i}{2}\right)\|\alpha_i^{(t)}-\alpha_i^{\star}\|^2-\sum_{i=1}^{m}\left(\frac{1}{p_i}-1\right)m\widetilde{D}_i(\alpha_i^{(t)})\\
\geq\ &\sum_{i=1}^{m}\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{\star}\|^2]+\sum_{i=1}^{m}\frac{1}{2p_i\sigma_i}\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{(t)}\|^2]-\sum_{i=1}^{m}\frac{m}{p_i}\mathbf{E}_t\left[\widetilde{D}_i(\alpha_i^{(t+1)})\right]\\
&+\left\langle w^{(t)}-w^{\star},\ X^T(\alpha^{\star}-\alpha^{(t)})\right\rangle-\sum_{i=1}^{m}\frac{1}{p_i}\mathbf{E}_t\left[\left\langle\alpha_i^{(t+1)}-\alpha_i^{(t)},\ u_i^{(t+1)}-X_{i:}w^{\star}\right\rangle\right]. \qquad (92)
\end{aligned}$$

Proof We start by taking conditional expectation $\mathbf{E}_t$ on both sides of the inequality (61), and would like to replace every term containing $\tilde{\alpha}_i^{(t+1)}$ with terms that contain only $\alpha_i^{(t)}$ and $\alpha_i^{(t+1)}$. In addition to the relations in (63), (64) and (65), we also need

$$\mathbf{E}_t\left[f_i^*(\alpha_i^{(t+1)})\right]\ =\ p_i f_i^*(\tilde{\alpha}_i^{(t+1)})+(1-p_i)f_i^*(\alpha_i^{(t)}).$$

After the substitutions and rearranging terms, we have

$$\begin{aligned}
&\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)-\frac{\gamma_i}{2}\right)\|\alpha_i^{(t)}-\alpha_i^{\star}\|^2+\left(\frac{1}{p_i}-1\right)\left(f_i^*(\alpha_i^{(t)})-f_i^*(\alpha_i^{\star})\right)\\
\geq\ &\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{\star}\|^2]+\frac{1}{2p_i\sigma_i}\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{(t)}\|^2]+\frac{1}{p_i}\mathbf{E}_t\left[\left(f_i^*(\alpha_i^{(t+1)})-f_i^*(\alpha_i^{\star})\right)\right]\\
&\mathbf{E}_t\left[\langle\alpha_i^{\star}-\alpha_i^{(t)},\ u_i^{(t+1)}\rangle\right]-\frac{1}{p_i}\mathbf{E}_t\left[\left\langle\alpha_i^{(t+1)}-\alpha_i^{(t)},\ u_i^{(t+1)}\right\rangle\right].
\end{aligned}$$

Next, we use the assumption $\mathbf{E}_t\left[u_i^{(t+1)}\right]=X_{i:}w^{(t)}$ and the definition of $\widetilde{D}_i(\cdot)$ in (89) to obtain

$$\begin{aligned}
&\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)-\frac{\gamma_i}{2}\right)\|\alpha_i^{(t)}-\alpha_i^{\star}\|^2-\left(\frac{1}{p_i}-1\right)m\widetilde{D}_i(\alpha_i^{(t)})\\
\geq\ &\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{\star}\|^2]+\frac{1}{2p_i\sigma_i}\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{(t)}\|^2]-\frac{m}{p_i}\mathbf{E}_t\left[\widetilde{D}_i(\alpha_i^{(t+1)})\right]\\
&+\left\langle\alpha_i^{\star}-\alpha_i^{(t)},\ X_{i:}\left(w^{(t)}-w^{\star}\right)\right\rangle-\frac{1}{p_i}\mathbf{E}_t\left[\left\langle\alpha_i^{(t+1)}-\alpha_i^{(t)},\ u_i^{(t+1)}-X_{i:}w^{\star}\right\rangle\right].
\end{aligned}$$

Summing up the above inequality for $i=1,\ldots,m$ gives the desired result (92). ■

Lemma 10 *Under the same assumptions and setup in Lemma 6, we have*

$$\begin{aligned}
&\sum_{k=1}^{K}\left(\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)-\frac{\lambda}{2}\right)\|w_k^{(t)}-w_k^{\star}\|^2+\sum_{k=1}^{n}\left(\frac{1}{q_k}-1\right)\widetilde{P}_k(w_k^{(t)})\\
\geq\ &\sum_{k=1}^{n}\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^{\star}\|^2]+\sum_{k=1}^{n}\frac{1}{2q_k\tau_k}\mathbf{E}_t[\|w_k^{(t+1)}-w_k^{(t)}\|^2]+\sum_{k=1}^{n}\frac{1}{q_k}\mathbf{E}_t\left[\widetilde{P}_k(w_k^{(t+1)})\right]\\
&+\frac{1}{m}\left\langle X(w^{(t)}-w^{\star}),\ \alpha^{(t)}-\alpha^{\star}\right\rangle+\sum_{k=1}^{n}\frac{1}{q_k}\mathbf{E}_t\left[\left\langle w_k^{(t+1)}-w_k^{(t)},\ v_k^{(t+1)}-\frac{1}{m}(X_{:k})^T\alpha^{\star}\right\rangle\right].
\end{aligned}$$

Based on Lemma 9 and Lemma 10, we can prove the following proposition. The proof is very similar to that of Proposition 7, thus we omit the details here.

Proposition 11 *The t-th iteration within the s-th stage of Algorithm 2 guarantees*

$$\begin{aligned}
&\sum_{k=1}^{n}\left(\frac{1}{q_k}-1\right)\widetilde{P}_k(w_k^{(t)})-\sum_{i=1}^{m}\left(\frac{1}{p_i}-1\right)\widetilde{D}_i(\alpha_i^{(t)})\\
&+\sum_{i=1}^{m}\frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)-\frac{\gamma_i}{2}+\sum_{k=1}^{n}\frac{3\tau_k\|X_{ik}\|^2}{mp_i}\right]\|\alpha_i^{(t)}-\alpha_i^{\star}\|^2+\sum_{i=1}^{m}\sum_{k=1}^{n}\frac{2\tau_k\|X_{ik}\|^2}{m^2p_i}\|\bar{\alpha}_i^{(s)}-\alpha_i^{\star}\|^2\\
&+\sum_{k=1}^{n}\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)-\frac{\lambda}{2}+\sum_{i=1}^{m}\frac{3\sigma_i\|X_{ik}\|^2}{mq_k}\right]\|w_k^{(t)}-w_k^{\star}\|^2+\sum_{i=1}^{m}\sum_{k=1}^{n}\frac{2\sigma_i\|X_{ik}\|^2}{mq_k}\|\bar{w}_k^{(s)}-w_k^{\star}\|^2\\
\geq\ &\sum_{k=1}^{n}\frac{1}{q_k}\mathbf{E}_t\left[\widetilde{P}_k(w_k^{(t+1)})\right]-\sum_{i=1}^{m}\frac{1}{p_i}\mathbf{E}_t\left[\widetilde{D}_i(\alpha_i^{(t+1)})\right]\\
&+\sum_{i=1}^{m}\frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)\mathbf{E}_t\left[\|\alpha_i^{(t+1)}-\alpha_i^{\star}\|^2\right]+\sum_{k=1}^{n}\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)\mathbf{E}_t\left[\|w_k^{(t+1)}-w_k^{\star}\|^2\right]. \qquad (93)
\end{aligned}$$

Now we proceed to prove Theorem 2. Let $\theta\in(0,1)$ be a parameter to be determined later, and let Γ and η be two constants such that

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1+\frac{6\Lambda}{\theta q_k\lambda\gamma_i}\right),\ \frac{1}{q_k}\left(1+\frac{6n\Lambda}{\theta p_i m\lambda\gamma_i}\right)\right\}, \qquad (94)$$

$$\eta = 1-\frac{1-\theta}{\Gamma}. \qquad (95)$$

It is easy to check that $\Gamma>1$ and $\eta\in(0,1)$. The choices of σ_i and τ_k in (32) and (33) satisfy

$$\frac{1}{p_i}\left(\frac{1}{2}+\frac{1}{2\sigma_i\gamma_i}\right) = \frac{\Gamma}{2}, \qquad i=1,\ldots,m, \qquad (96)$$

$$\frac{1}{q_k}\left(\frac{1}{2}+\frac{1}{2\tau_k\lambda}\right) = \frac{\Gamma}{2}, \qquad k=1,\ldots,n. \qquad (97)$$

Comparing them with the definition of Γ in (94), and using the assumption $\Lambda\geq\|X_{ik}\|_F^2\geq\|X_{ik}\|^2$, we get

$$\frac{6\|X_{ik}\|^2}{\theta q_k\lambda\gamma_i}\leq\frac{6\Lambda}{\theta q_k\lambda\gamma_i}\leq\frac{1}{\sigma_i\gamma_i} \qquad \text{and} \qquad \frac{6n\|X_{ik}\|^2}{\theta p_i m\lambda\gamma_i}\leq\frac{6n\Lambda}{\theta p_i m\lambda\gamma_i}\leq\frac{1}{\tau_k\lambda},$$

which implies

$$\frac{3\sigma_i\|X_{ik}\|^2}{q_k}\leq\theta\frac{\lambda}{2} \qquad \text{and} \qquad \frac{3n\tau_k\|X_{ik}\|^2}{mp_i}\leq\theta\frac{\gamma_i}{2},$$

for all $i=1,\ldots,m$ and $k=1,\ldots,n$. Therefore, we have

$$\sum_{k=1}^{n}\frac{3\tau_k\|X_{ik}\|^2}{mp_i}=\frac{1}{n}\sum_{k=1}^{n}\frac{3n\tau_k\|X_{ik}\|^2}{mp_i}\leq\theta\frac{\gamma_i}{2}, \qquad i=1,\ldots,m, \qquad (98)$$

$$\sum_{i=1}^{m}\frac{3\sigma_i\|X_{ik}\|^2}{mq_k}=\frac{1}{m}\sum_{i=1}^{m}\frac{3\sigma_i\|X_{ik}\|^2}{q_k}\leq\theta\frac{\lambda}{2}, \qquad k=1,\ldots,n. \qquad (99)$$

Now we consider the inequality (93), and examine the ratio between the coefficients of $\|\alpha_i^{(t)}-\alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]$. Using (98) and (96), we have

$$\frac{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)-\frac{\gamma_i}{2}+\sum_{k=1}^n\frac{3\tau_k\|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)}\le 1-\frac{(1-\theta)\frac{\gamma_i}{2}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)}=1-\frac{1-\theta}{\Gamma}=\eta. \tag{100}$$

Similarly, the ratio between the coefficients of $\|w_k^{(t)}-w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]$ can be bounded using (99) and (97):

$$\frac{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)-\frac{\lambda}{2}+\sum_{i=1}^m\frac{3\sigma_i\|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)}\le 1-\frac{(1-\theta)\frac{\lambda}{2}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)}=1-\frac{1-\theta}{\Gamma}=\eta. \tag{101}$$

In addition, the ratio between the coefficients of $\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]$ and that of $\|\bar{w}_k^{(s)}-w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]$ can be bounded as

$$\frac{\sum_{k=1}^n\frac{2\tau_k\|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)}\le\frac{(2/3)\theta\frac{\gamma_i}{2}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)}=\frac{(2/3)\theta}{\Gamma}=\frac{2\theta(1-\eta)}{3(1-\theta)}, \tag{102}$$

$$\frac{\sum_{i=1}^m\frac{2\sigma_i\|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)}\le\frac{(2/3)\theta\frac{\lambda}{2}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)}=\frac{(2/3)\theta}{\Gamma}=\frac{2\theta(1-\eta)}{3(1-\theta)}. \tag{103}$$

Also, the ratios between the coefficients of $\widetilde{P}_k(w_k^{(t)})$ and $\mathbf{E}_t\left[\widetilde{P}_k(w_k^{(t+1)})\right]$ is $1-q_k$, and that of $\widetilde{D}_k(\alpha_i^{(t)})$ and $\mathbf{E}_t\left[\widetilde{D}_i(\alpha_i^{(t+1)})\right]$ is $1-p_i$. From the definition of Γ and η in (94) and (95), we have

$$1-p_i\le\eta\quad\text{for}\quad i=1,\dots,m,\quad\text{and}\quad 1-q_k\le\eta\quad\text{for}\quad k=1,\dots,n. \tag{104}$$

Using the relations in (96) and (97) and the inequalities (100), (101), (102), (103) and (104), we conclude that the inequality (93) implies

$$\begin{aligned}
&\eta\left(\sum_{k=1}^n\frac{1}{q_k}\widetilde{P}_k(w_k^{(t)})-\sum_{i=1}^m\frac{1}{p_i}\widetilde{D}_i(\alpha_i^{(t)})\right)+\eta\left(\sum_{i=1}^m\frac{\Gamma\gamma_i}{2m}\|\alpha_i^{(t)}-\alpha_i^\star\|^2+\sum_{k=1}^n\frac{\Gamma\lambda}{2}\|w_k^{(t)}-w_k^\star\|^2\right)\\
&+\frac{2\theta(1-\eta)}{3(1-\theta)}\left(\sum_{i=1}^m\frac{\Gamma\gamma_i}{2m}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2\sum_{k=1}^n\frac{\Gamma\lambda}{2}\|\bar{w}_k^{(s)}-w_k^\star\|^2\right)\\
\ge\;&\sum_{k=1}^n\frac{1}{q_k}\mathbf{E}_t\left[\widetilde{P}_k(w_k^{(t+1)})\right]-\sum_{i=1}^m\frac{1}{p_i}\mathbf{E}_t\left[\widetilde{D}_i(\alpha_i^{(t+1)})\right]\\
&+\sum_{i=1}^m\frac{\Gamma\gamma_i}{2m}\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]+\sum_{k=1}^n\frac{\Gamma\lambda}{2}\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2],
\end{aligned}$$

which is equivalent to

$$\begin{aligned}
&\eta\left(\sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(w_k^{(t)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\alpha_i^{(t)})+\frac{\Gamma\lambda}{2}\|w^{(t)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\alpha_i^{(t)}-\alpha_i^\star\|^2\right)\\
&+\frac{2\theta(1-\eta)}{3(1-\theta)}\left(\frac{\Gamma\lambda}{2}\|\bar{w}^{(s)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2\right) \qquad (105)\\
\geq\ &\mathbf{E}_t\left[\sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(w_k^{(t+1)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\alpha_i^{(t+1)})+\frac{\Gamma\lambda}{2}\|w^{(t+1)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2\right].
\end{aligned}$$

To simplify further derivation, we define

$$\begin{aligned}
\Delta^{(t)} &= \sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(w_k^{(t)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\alpha_i^{(t)})+\frac{\Gamma\lambda}{2}\|w^{(t)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\alpha_i^{(t)}-\alpha_i^\star\|^2,\\
\bar{\Delta}^{(s)} &= \sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(\bar{w}_k^{(s)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\bar{\alpha}_i^{(s)})+\frac{\Gamma\lambda}{2}\|\bar{w}^{(s)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2.
\end{aligned}$$

Using the facts that $\widetilde{P}_k(\bar{w}_k^{(s)})\geq 0$ and $-\widetilde{D}_i(\bar{\alpha}_i^{(s)})\geq 0$, the inequality (105) implies

$$\frac{2\theta(1-\eta)}{3(1-\theta)}\bar{\Delta}^{(s)}+\eta\mathbf{E}\big[\Delta^{(t)}\big]\ \geq\ \mathbf{E}\big[\Delta^{(t+1)}\big],$$

where the expectation is taken with respect to all randomness in the s-th stage, that is, the random variables $\{(j^{(0)},l^{(0)}),(j^{(1)},l^{(1)}),\ldots,(j^{(M-1)},l^{(M-1)})\}$. Next we choose $\theta=1/3$ and follow the same arguments as in the proof for Theorem 1 to obtain $\mathbf{E}\big[\Delta^{(M)}\big]\ \leq\ \frac{2}{3}\Delta^{(0)}$, provided $M\geq\log(3)\Gamma$. This further implies

$$\mathbf{E}\big[\bar{\Delta}^{(s)}\big]\ \leq\ \left(\frac{2}{3}\right)^s\bar{\Delta}^{(0)}. \qquad (106)$$

From the definition of Γ in (94), we have $\frac{1}{q_k}<\Gamma$ for $k=1,\ldots,n$ and $\frac{1}{p_i}<\Gamma$ for $i=1,\ldots,m$. Therefore,

$$\begin{aligned}
\bar{\Delta}^{(0)} &\leq \Gamma\left(\sum_{k=1}^{n}\widetilde{P}_k(\bar{w}_k^{(0)})-\sum_{i=1}^{m}\widetilde{D}_i(\bar{\alpha}_i^{(0)})+\frac{\lambda}{2}\|\bar{w}^{(0)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\gamma_i}{2}\|\bar{\alpha}_i^{(0)}-\alpha_i^\star\|^2\right)\\
&\leq 2\Gamma\left(\sum_{k=1}^{n}\widetilde{P}_k(\bar{w}_k^{(0)})-\sum_{i=1}^{m}\widetilde{D}_i(\bar{\alpha}_i^{(0)})\right)\\
&\leq 2\Gamma\left(P(\bar{w}^{(0)})-D(\bar{\alpha}^{(0)})\right), \qquad (107)
\end{aligned}$$

where the second inequality used (90) and the last inequality used (91). On the other hand, we can also lower bound $\bar{\Delta}^{(s)}$ using $P(\bar{w}^{(s)})-D(\bar{\alpha}^{(s)})$. To this end, we notice that with $\theta=1/3$,

$$\Gamma\ \geq\ \max_{i,k}\left\{\frac{1}{p_i}\left(1+\frac{18\Lambda}{q_k\lambda\gamma_i}\right),\ \frac{1}{q_k}\left(1+\frac{18n\Lambda}{p_i m\lambda\gamma_i}\right)\right\}\ \geq\ \max_{i,k}\left\{\frac{18\Lambda}{p_i q_k\lambda\gamma_i},\ \frac{18n\Lambda}{p_i q_k m\lambda\gamma_i}\right\}.$$

Noticing that $\max_k\{1/q_k\}\geq n$ and $n\Lambda\geq\|X_{i:}\|_F^2$ for all $i=1,\ldots,m$, we have

$$\Gamma\geq\max_{i,k}\left\{\frac{18\Lambda}{q_k\lambda\gamma_i}\right\}\geq\max_{i}\left\{\frac{18n\Lambda}{\lambda\gamma_i}\right\}\geq\frac{18}{m\lambda}\sum_{i=1}^{m}\frac{n\Lambda}{\gamma_i}\geq\frac{18}{m\lambda}\sum_{i=1}^{m}\frac{\|X_{i:}\|_F^2}{\gamma_i}\geq\frac{18}{m\lambda}\sum_{i=1}^{m}\frac{\|X_{i:}\|^2}{\gamma_i}.$$

Moreover, since $\Gamma \geq \max_k \left\{\frac{18\Lambda}{p_i q_k \lambda \gamma_i}\right\} \geq \frac{18n\Lambda}{p_i \lambda \gamma_i}$ for all i and $mn\Lambda \geq \|X\|_F^2$, we have

$$\begin{aligned}
\frac{1}{m}\sum_{i=1}^{m}\Gamma\gamma_i\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2 &\geq \frac{1}{m}\sum_{i=1}^{m}\frac{18n\Lambda}{p_i\lambda\gamma_i}\gamma_i\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2 = \frac{18mn\Lambda}{m^2\lambda}\sum_{i=1}^{m}\frac{\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2}{p_i} \\
&\geq \frac{18\|X\|_F^2}{m^2\lambda}\left(\sum_{i=1}^{m}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|\right)^2 \\
&\geq \frac{18\|X\|^2}{m^2\lambda}\sum_{i=1}^{m}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2 = \frac{18\|X\|^2}{m^2\lambda}\|\bar{\alpha}^{(s)}-\alpha^\star\|^2.
\end{aligned}$$

Therefore, from the definition of $\bar{\Delta}^{(s)}$,

$$\begin{aligned}
\bar{\Delta}^{(s)} &= \sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(\bar{w}_k^{(s)}) - \sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\bar{\alpha}_i^{(s)}) + \frac{\Gamma\lambda}{2}\|\bar{w}^{(s)}-w^\star\|^2 + \frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2 \\
&\geq \sum_{k=1}^{n}\widetilde{P}_k(\bar{w}_k^{(s)}) - \sum_{i=1}^{m}\widetilde{D}_i(\bar{\alpha}_i^{(s)}) + \left(\frac{18}{m}\sum_{i=1}^{m}\frac{\|X_{i:}\|^2}{\gamma_i}\right)\|\bar{w}^{(s)}-w^\star\|^2 + \frac{18\|X\|^2}{m^2\lambda}\|\bar{\alpha}^{(s)}-\alpha^\star\|^2 \\
&= L(\bar{w}^{(s)},\alpha^\star) - L(w^\star,\bar{\alpha}^{(s)}) + \left(\frac{18}{m}\sum_{i=1}^{m}\frac{\|X_{i:}\|^2}{\gamma_i}\right)\|\bar{w}^{(s)}-w^\star\|^2 + \frac{18\|X\|^2}{m^2\lambda}\|\bar{\alpha}^{(s)}-\alpha^\star\|^2 \\
&\geq P(\bar{w}^{(s)}) - D(\bar{\alpha}^{(s)}), \qquad (108)
\end{aligned}$$

where the last inequality is due to Lemma 8. Combining (106), (107) and (108) gives the desired result:

$$\mathbf{E}\left[P(\bar{w}^{(s)}) - D(\bar{\alpha}^{(s)})\right] \leq \left(\frac{2}{3}\right)^s 2\Gamma\left(P(\bar{w}^{(0)}) - D(\bar{\alpha}^{(0)})\right).$$

Appendix C. Proof of Theorem 3

To facilitate the analysis of DSCOVR-SAGA in Algorithm 3, we define two sequences of matrices recursively. The first is $\{W^{(t)}\}_{t\geq 0}$, where each $W^{(t)} \in \mathbf{R}^{m\times d}$. They are partitioned into $m\times n$ blocks, and we denote each block as $W_{ik}^{(t)} \in \mathbf{R}^{1\times d_k}$. The recursive updates for $W^{(t)}$ are as follows:

$$\begin{aligned}
W^{(0)} &= \mathbf{1}_m \otimes \left(w^{(0)}\right)^T, \\
W_{ik}^{(t+1)} &= \begin{cases} \left(w_l^{(t)}\right)^T & \text{if } i=j \text{ and } k=l, \\ W_{ik}^{(t)} & \text{otherwise,} \end{cases} \qquad t = 0,1,2,\ldots, \qquad (109)
\end{aligned}$$

where $\mathbf{1}_m$ denotes the vector of all ones in $\mathbf{R}^m$. and $\otimes$ denotes the Kronecker product of two matrices. The second sequence is $\{A^{(t)}\}_{t\geq 0}$, where each $A^{(t)} \in \mathbf{R}^{N\times n}$. They are partitioned into $m\times n$ blocks, and we denote each block as $A_{ik}^{(t)} \in \mathbf{R}^{N_i\times 1}$. The recursive updates for $A^{(t)}$ are as follows:

$$\begin{aligned}
A^{(0)} &= \alpha^{(0)} \otimes \mathbf{1}_n^T, \\
A_{ik}^{(t+1)} &= \begin{cases} \alpha_j^{(t)} & \text{if } i=j \text{ and } k=l, \\ A_{ik}^{(t)} & \text{otherwise,} \end{cases} \qquad t = 0,1,2,\ldots. \qquad (110)
\end{aligned}$$

The matrices $W^{(t)}$ and $A^{(t)}$ consist of most recent values of the primal and dual block coordinates, updated at different times, up to time t.

Notice that in Algorithm 3, the matrices $U^{(t)} \in \mathbf{R}^{N\times n}$ follow the same partitioning as the matrices $A^{(t)}$, and the matrices $V^{(t)} \in \mathbf{R}^{m\times d}$ follow the same partitioning as the matrices $W^{(t)}$. According to the updates of $U^{(t)}$, $V^{(t)}$, $\bar{u}^{(t)}$ and $\bar{v}^{(t)}$ in Algorithm 3, we have for each $t \geq 0$,

$$\begin{aligned} U_{ik}^{(t)} &= X_{ik}\big(W_{ik}^{(t)}\big)^T, \quad i = 1,\ldots,m, \quad k = 1,\ldots,n, \\ V_{ik}^{(t)} &= \frac{1}{m}\big(A_{ik}^{(t)}\big)^T X_{ik}, \quad i = 1,\ldots,m, \quad k = 1,\ldots,n. \end{aligned}$$

Proposition 12 *Suppose Assumption 1 holds. The t-th iteration of Algorithm 3 guarantees*

$$\begin{aligned} &\sum_{i=1}^m \frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)-\gamma_i+\sum_{k=1}^n \frac{3\tau_k\|X_{ik}\|^2}{mp_i}\right]\|\alpha_i^{(t)}-\alpha_i^\star\|^2+\sum_{i=1}^m\sum_{k=1}^n\frac{2\tau_k\|X_{ik}\|^2}{m^2p_i}\|A_{ik}^{(t)}-\alpha_i^\star\|^2 \\ &+\sum_{k=1}^n\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)-\lambda+\sum_{i=1}^m\frac{3\sigma_i\|X_{ik}\|^2}{mq_k}\right]\|w_k^{(t)}-w_k^\star\|^2+\sum_{i=1}^m\sum_{k=1}^n\frac{2\sigma_i\|X_{ik}\|^2}{mq_k}\|\big(W_{ik}^{(t)}\big)^T-w_k^\star\|^2 \\ \geq\ &\sum_{i=1}^m\frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t\left[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2\right]+\sum_{k=1}^n\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t\left[\|w_k^{(t+1)}-w_k^\star\|^2\right] \end{aligned} \tag{111}$$

Proof The main differences between Algorithm 2 and Algorithm 3 are the definitions of $u_j^{(t+1)}$ and $v_l^{(t+1)}$. We start with the inequality (68) and revisit the bounds for the following two quantities:

$$\mathbf{E}_t\left[\|u_i^{(t+1)}-X_{i:}w^{(t)}\|^2\right] \qquad \text{and} \qquad \mathbf{E}_t\left[\left\|v_k^{(t+1)}-\frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right].$$

For Algorithm 3, we have

$$\begin{aligned} u_i^{(t+1)} &= \bar{u}_i^{(t)}-\frac{1}{q_l}U_{il}^{(t)}+\frac{1}{q_l}X_{il}w_l^{(t)}, \qquad i = 1,\ldots,m, \\ v_k^{(t+1)} &= \bar{v}_k^{(t)}-\frac{1}{p_j}(V_{jk}^{(t)})^T+\frac{1}{p_j}\frac{1}{m}(X_{jk})^T\alpha_j^{(t)}, \qquad k = 1,\ldots,n. \end{aligned}$$

We can apply the reasoning in (39) and (40) to every block coordinate and obtain

$$\begin{aligned} \mathbf{E}_t\left[u_i^{(t+1)}\right] &= X_{i:}w^{(t)}, \qquad i = 1,\ldots,m, \\ \mathbf{E}_t\left[v_k^{(t+1)}\right] &= \frac{1}{m}(X_{:k})^T\alpha^{(t)}, \qquad k = 1,\ldots,n. \end{aligned}$$

Therefore they satisfy the assumptions in Lemma 5 and Lemma 6, respectively. Moreover, following similar arguments as in (69) and (70), we have

$$\begin{aligned} \mathbf{E}_t\left[\|u_i^{(t+1)}-X_{i:}w^{(t)}\|^2\right] &\leq \sum_{k=1}^n\frac{2\|X_{ik}\|^2}{q_k}\left(\left\|\big(W_{ik}^{(t)}\big)^T-w_k^\star\right\|^2+\|w_k^{(t)}-w_k^\star\|^2\right), \\ \mathbf{E}_t\left[\left\|v_k^{(t+1)}-\frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right] &\leq \sum_{i=1}^m\frac{2\|X_{ik}\|^2}{m^2p_i}\left(\left\|\big(A_{ik}^{(t)}\big)^T-\alpha_i^\star\right\|^2+\|\alpha_i^{(t)}-\alpha_i^\star\|^2\right). \end{aligned}$$

The rest of the proof are the same as in the proof of Proposition 7. ■

Now we are ready to prove Theorem 3. By the definition of $W^{(t)}$ in (109) and $A^{(t)}$ in (110), we have

$$\mathbf{E}_t\left[\left\|(W_{ik}^{(t+1)})^T - w_k^\star\right\|^2\right] = p_i q_k\left\|w_k^{(t)} - w_k^\star\right\|^2 + (1-p_iq_k)\left\|\left(W_{ik}^{(t)}\right)^T - w_k^\star\right\|^2, \tag{112}$$

$$\mathbf{E}_t\left[\left\|A_{ik}^{(t+1)} - \alpha_i^\star\right\|^2\right] = p_i q_k\left\|\alpha_i^{(t)} - \alpha_i^\star\right\|^2 + (1-p_iq_k)\left\|A_{ik}^{(t)} - \alpha_i^\star\right\|^2. \tag{113}$$

For all $i = 1,\ldots,m$ and $k = 1,\ldots,n$, let

$$\xi_{ik} = \frac{3\sigma_i\|X_{ik}\|^2}{mp_iq_k^2} \qquad \text{and} \qquad \zeta_{ik} = \frac{3\tau_k\|X_{ik}\|^2}{m^2p_i^2q_k}. \tag{114}$$

We multiply (112) by ξ_{ik} and (113) by ζ_{ik} and add them to (111) to obtain

$$\begin{aligned}
&\sum_{i=1}^m \frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right) - \gamma_i + \sum_{k=1}^n \frac{6\tau_k\|X_{ik}\|^2}{mp_i}\right]\|\alpha_i^{(t)} - \alpha_i^\star\|^2 \\
&+\sum_{k=1}^n\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right) - \lambda + \sum_{i=1}^m \frac{6\sigma_i\|X_{ik}\|^2}{mq_k}\right]\|w_k^{(t)} - w_k^\star\|^2 \\
&+\sum_{i=1}^m\sum_{k=1}^n\left(1-\frac{1}{3}p_iq_k\right)\zeta_{ik}\left\|A_{ik}^{(t)} - \alpha_i^\star\right\|^2 + \sum_{i=1}^m\sum_{k=1}^n\left(1-\frac{1}{3}p_iq_k\right)\xi_{ik}\left\|\left(W_{ik}^{(t)}\right)^T - w_k^\star\right\|^2 \\
\geq\ &\sum_{i=1}^m \frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t\left[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2\right] + \sum_{k=1}^n \frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t\left[\|w_k^{(t+1)} - w_k^\star\|^2\right] \\
&+\sum_{i=1}^m\sum_{k=1}^n \zeta_{ik}\mathbf{E}_t\left[\left\|A_{ik}^{(t+1)} - \alpha_i^\star\right\|^2\right] + \sum_{i=1}^m\sum_{k=1}^n \xi_{ik}\mathbf{E}_t\left[\left\|\left(W_{ik}^{(t+1)}\right)^T - w_k^\star\right\|^2\right].
\end{aligned} \tag{115}$$

Let $\theta \in (0,1)$ be a parameter to be determined later, and Γ be a constant such that

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1+\frac{3\|X_{ik}\|^2}{2\theta q_k\lambda\gamma_i}\right),\ \frac{1}{q_k}\left(1+\frac{3n\|X_{ik}\|^2}{2\theta p_i m\lambda\gamma_i}\right),\ \frac{1}{p_iq_k}\right\}. \tag{116}$$

The choices of σ_i in (42) and τ_k in (43) satisfy

$$\frac{1}{p_i}\left(1+\frac{1}{2\sigma_i\gamma_i}\right) = \frac{1}{q_k}\left(1+\frac{1}{2\tau_k\lambda}\right) = \Gamma. \tag{117}$$

Comparing the above equality with the definition of Γ in (116), we have

$$\frac{3\|X_{ik}\|^2}{2\theta q_k\lambda\gamma_i} \leq \frac{1}{2\sigma_i\gamma_i} \qquad \text{and} \qquad \frac{3n\|X_{ik}\|^2}{2\theta p_i m\lambda\gamma_i} \leq \frac{1}{2\tau_k\lambda},$$

which implies that

$$\frac{6\sigma_i\|X_{ik}\|^2}{q_k} \leq 2\theta\lambda \qquad \text{and} \qquad \frac{6n\tau_k\|X_{ik}\|^2}{mp_i} \leq 2\theta\gamma_i \tag{118}$$

hold for all $i = 1,\ldots,m$ and $k = 1,\ldots,n$. Therefore, we have

$$\sum_{k=1}^n \frac{6\tau_k\|X_{ik}\|^2}{mp_i} = \frac{1}{n}\sum_{k=1}^n \frac{6n\tau_k\|X_{ik}\|^2}{mp_i} \leq 2\theta\gamma_i, \qquad i = 1,\ldots,m, \tag{119}$$

$$\sum_{i=1}^m \frac{6\sigma_i\|X_{ik}\|^2}{mq_k} = \frac{1}{m}\sum_{i=1}^m \frac{6\sigma_i\|X_{ik}\|^2}{q_k} \leq 2\theta\lambda, \qquad k = 1,\ldots,n. \tag{120}$$

Now we consider the inequality (115), and examine the ratio between the coefficients of $\|\alpha_i^{(t)} - \alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2]$. Using (119) and (117), we have

$$\frac{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right) - \gamma_i + \sum_{k=1}^n \frac{6\tau_k \|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)} \leq 1 - \frac{(1-2\theta)\gamma_i}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)} = 1 - \frac{1-2\theta}{\Gamma}. \tag{121}$$

Similarly, the ratio between the coefficients of $\|w_k^{(t)} - w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2]$ can be bounded using (120) and (117):

$$\frac{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right) - \lambda + \sum_{i=1}^m \frac{6\sigma_i \|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)} \leq 1 - \frac{(1-2\theta)\lambda}{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)} = 1 - \frac{1-2\theta}{\Gamma}. \tag{122}$$

We notice that in (115), the ratios between the coefficients of $\zeta_{ik}\left\|A_{ik}^{(t)} - \alpha_i^\star\right\|^2$ and $\zeta_{ik}\mathbf{E}_t\left[\left\|A_{ik}^{(t+1)} - \alpha_i^\star\right\|^2\right]$, as well as and that of $\xi_{ik}\left\|\left(W_{ik}^{(t)}\right)^T - w_k^\star\right\|^2$ and $\xi_{ik}\mathbf{E}_t\left[\left\|\left(W_{ik}^{(t+1)}\right)^T - w_k^\star\right\|^2\right]$, are all $1 - \frac{1}{3}p_i q_k$. By definition of Γ in (116), we have

$$1 - \frac{1}{3}p_i q_k \;\leq\; 1 - \frac{1}{3\Gamma}, \qquad i = 1, \ldots, m, \quad k = 1, \ldots, n. \tag{123}$$

We choose $\theta = 1/3$ so that the ratios in (121) and (122) have the same bound $1 - \frac{1}{3\Gamma}$. Therefore, it follows from inequality (115) that

$$\begin{aligned}
&\sum_{i=1}^m \frac{\Gamma\gamma_i}{m}\mathbf{E}_t\left[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2\right] + \sum_{k=1}^n \Gamma\lambda\mathbf{E}_t\left[\|w_k^{(t+1)} - w_k^\star\|^2\right] \\
&+ \sum_{i=1}^m\sum_{k=1}^n \zeta_{ik}\mathbf{E}_t\left[\left\|A_{ik}^{(t+1)} - \alpha_i^\star\right\|^2\right] + \sum_{i=1}^m\sum_{k=1}^n \xi_{ik}\mathbf{E}_t\left[\left\|\left(W_{ik}^{(t+1)}\right)^T - w_k^\star\right\|^2\right]. \\
\leq\;& \left(1 - \frac{1}{3\Gamma}\right)\Bigg(\sum_{i=1}^m \frac{\Gamma\gamma_i}{m}\|\alpha_i^{(t)} - \alpha_i^\star\|^2 + \sum_{k=1}^n \Gamma\lambda\|w_k^{(t)} - w_k^\star\|^2 \\
&\qquad + \sum_{i=1}^m\sum_{k=1}^n \zeta_{ik}\left\|A_{ik}^{(t)} - \alpha_i^\star\right\|^2 + \sum_{i=1}^m\sum_{k=1}^n \xi_{ik}\left\|\left(W_{ik}^{(t)}\right)^T - w_k^\star\right\|^2\Bigg).
\end{aligned} \tag{124}$$

Let's define

$$\Delta^{(t)} = \lambda\|w^{(t)} - w^\star\|^2 + \frac{1}{m}\sum_{i=1}^m \gamma_i\|\alpha_i^{(t)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{\zeta_{ik}}{\Gamma}\left\|A_{ik}^{(t)} - \alpha_i^\star\right\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{\xi_{ik}}{\Gamma}\left\|\left(W_{ik}^{(t)}\right)^T - w_k^\star\right\|^2.$$

Then (124) implies

$$\mathbf{E}\left[\Delta^{(t)}\right] \leq \left(1 - \frac{1}{3\Gamma}\right)^t \Delta^{(0)}, \tag{125}$$

where the expectation is taken with respect to all random variables generated by Algorithm 3 up to iteration t.

By the definition of ξ_{ik} in (114), we have

$$\frac{\xi_{ik}}{\Gamma} = \frac{3\sigma_i \|X_{ik}\|^2}{m p_i q_k^2} \frac{1}{\Gamma} \leq \frac{\theta\lambda}{m p_i q_k} \frac{1}{\Gamma} \leq \frac{\theta\lambda}{m} = \frac{\lambda}{3m},$$

where the first inequality is due to (118) and the second inequality is due to the relation $\Gamma \geq \frac{1}{p_i q_k}$ from the definition of Γ in (116). Similarly, we have

$$\frac{\zeta_{ik}}{\Gamma} = \frac{3\tau_k \|X_{ik}\|^2}{m^2 p_i^2 q_k} \frac{1}{\Gamma} \leq \frac{\theta\gamma_i}{m n p_i q_k} \frac{1}{\Gamma} \leq \frac{\theta\gamma_i}{3mn} = \frac{\gamma_i}{3mn}.$$

Moreover, by the construction in (109) and (110), we have for $t = 0$,

$$\begin{aligned} A_{ik}^{(0)} &= \alpha_i^{(0)}, \quad \text{for} \quad k = 1, \ldots, n \quad \text{and} \quad i = 1, \ldots, m, \\ \left(W_{ik}^{(0)}\right)^T &= w_k^{(0)}, \quad \text{for} \quad i = 1, \ldots, m \quad \text{and} \quad k = 1, \ldots, n. \end{aligned}$$

Therefore, the last two terms in the definition of $\Delta^{(0)}$ can be bounded as

$$\begin{aligned} & \sum_{i=1}^m \sum_{k=1}^n \frac{\zeta_{ik}}{\Gamma} \left\| A_{ik}^{(0)} - \alpha_i^\star \right\|^2 + \sum_{i=1}^m \sum_{k=1}^n \frac{\xi_{ik}}{\Gamma} \left\| \left(W_{ik}^{(0)}\right)^T - w_k^\star \right\|^2 \\ \leq \; & \sum_{i=1}^m \sum_{k=1}^n \frac{\gamma_i}{3mn} \left\| \alpha_i^{(0)} - \alpha_i^\star \right\|^2 + \sum_{i=1}^m \sum_{k=1}^n \frac{\lambda}{3m} \left\| w_k^{(0)} - w_k^\star \right\|^2 \\ = \; & \frac{1}{3m} \sum_{i=1}^m \gamma_i \left\| \alpha_i^{(0)} - \alpha_i^\star \right\|^2 + \frac{\lambda}{3} \left\| w^{(0)} - w^\star \right\|^2, \end{aligned}$$

which implies

$$\Delta^{(0)} \leq \frac{4}{3} \left(\lambda \left\| w^{(0)} - w^\star \right\|^2 + \frac{1}{m} \sum_{i=1}^m \gamma_i \left\| \alpha_i^{(0)} - \alpha_i^\star \right\|^2 \right).$$

Finally, combining with (125), we have

$$\mathbf{E}\left[\Delta^{(t)}\right] \leq \left(1 - \frac{1}{3\Gamma}\right)^t \frac{4}{3} \left(\lambda \left\| w^{(0)} - w^\star \right\|^2 + \frac{1}{m} \sum_{i=1}^m \gamma_i \left\| \alpha_i^{(0)} - \alpha_i^\star \right\|^2 \right),$$

which further implies the desired result.

Appendix D. Proof of Theorem 4

To simplify the presentation, we present the proof for the case $\gamma_i = \gamma$ for all $i = 1, \ldots, m$. It is straightforward to generalize to the case where the γ_i's are different.

Lemma 13 *Let $g : \mathbf{R}^D \to \mathbf{R}$ be λ-strongly convex, and $f_i^* : \mathbf{R}^{N_i} \to \mathbf{R} \cup \{\infty\}$ be γ-strongly convex over its domain. Given any $\tilde{w} \in \mathbf{R}^d$ and $\tilde{\alpha} \in \mathbf{R}^N$, we define the following two functions:*

$$L(w, \alpha) = g(w) + \frac{1}{m} \alpha^T X w - \frac{1}{m} \sum_{i=1}^m f_i * (\alpha_i), \tag{126}$$

$$L_\delta(w, \alpha) = L(w, \alpha) + \frac{\delta\lambda}{2} \|w - \tilde{w}\|^2 - \frac{\delta\gamma}{2m} \|\alpha - \tilde{\alpha}\|^2. \tag{127}$$

Let $(w^\star, \alpha^\star)$ and $(\tilde{w}^\star, \tilde{\alpha}^\star)$ be the (unique) saddle points of $L(w, \alpha)$ and $L_\delta(w, \alpha)$, respectively. Then we have

$$\lambda\|\tilde{w} - \tilde{w}^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \tilde{\alpha}^\star\|^2 \quad \leq \quad \lambda\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \alpha^\star\|^2, \tag{128}$$

$$\left(\lambda\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \alpha^\star\|^2\right)^{1/2} \quad \leq \quad \frac{\delta}{1+\delta}\left(\lambda\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \alpha^\star\|^2\right)^{1/2}. \tag{129}$$

Proof This lemma can be proved using the theory of monotone operators (e.g., Rockafellar, 1976; Ryu and Boyd, 2016), as done by Balamurugan and Bach (2016). Here we give an elementary proof based on first-order optimality conditions.

By assumption, we have

$$\begin{aligned}(w^\star, \alpha^\star) &= \arg\min_w \max_\alpha L(w, \alpha),\\ (\tilde{w}^\star, \tilde{\alpha}^\star) &= \arg\min_w \max_\alpha L_\delta(w, \alpha).\end{aligned}$$

Optimality conditions for $(\tilde{w}^\star, \tilde{\alpha}^\star)$ as a saddle point of L_δ:

$$-\frac{1}{m}X^T\tilde{\alpha}^\star - \delta\lambda\left(\tilde{w}^\star - \tilde{w}\right) \quad \in \quad \partial g(\tilde{w}^\star), \tag{130}$$

$$X\tilde{w}^\star - \delta\gamma\left(\tilde{\alpha}^\star - \tilde{\alpha}\right) \quad \in \quad \partial\sum_{i=1}^m f_i^*(\tilde{\alpha}^\star). \tag{131}$$

For any $\xi \in \partial g(\tilde{w}^\star)$, it holds that $\xi + \frac{1}{m}X^T\alpha^\star \in \partial_w L(\tilde{w}^\star, \alpha^\star)$. Therefore using (130) we have

$$\frac{1}{m}X^T(\alpha^\star - \tilde{\alpha}^\star) - \delta\lambda\left(\tilde{w}^\star - \tilde{w}\right) \quad \in \quad \partial_w L(\tilde{w}^\star, \alpha^\star).$$

Since $L(w, \alpha^\star)$ is strongly convex in w with convexity parameter λ, we have

$$L(\tilde{w}^\star, \alpha^\star) + \left(\frac{1}{m}X^T(\alpha^\star - \tilde{\alpha}^\star) - \delta\lambda\left(\tilde{w}^\star - \tilde{w}\right)\right)^T(w^\star - \tilde{w}^\star) + \frac{\lambda}{2}\|\tilde{w}^\star - w^\star\|^2 \leq L(w^\star, \alpha^\star). \tag{132}$$

Similarly, we have

$$\frac{1}{m}X^T(\tilde{w}^\star - w^\star) - \frac{\delta\gamma}{m}\left(\tilde{\alpha}^\star - \tilde{\alpha}\right) \quad \in \quad \partial_\alpha\left(-L(w^\star, \tilde{\alpha}^\star)\right),$$

and since $-L(w^\star, \alpha)$ is strongly convex in α with convexity parameter $\frac{\gamma}{m}$, we have

$$-L(w^\star, \tilde{\alpha}^\star) + \left(\frac{1}{m}X^T(\tilde{w}^\star - w^\star) - \frac{\delta\gamma}{m}\left(\tilde{\alpha}^\star - \tilde{\alpha}\right)\right)^T(\alpha^\star - \tilde{\alpha}^\star) + \frac{\gamma}{2m}\|\tilde{\alpha}^\star - \alpha^\star\|^2 \leq -L(w^\star, \alpha^\star). \tag{133}$$

Adding inequalities (132) and (133) together gives

$$\begin{aligned}&L(\tilde{w}^\star, \alpha^\star) - L(w^\star, \tilde{\alpha}^\star)\\ &+ \delta\lambda(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) + \frac{\delta\gamma}{m}(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) + \frac{\lambda}{2}\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{2m}\|\tilde{\alpha}^\star - \alpha^\star\|^2 \leq 0.\end{aligned}$$

Combining with the inequality

$$L(\tilde{w}^\star, \alpha^\star) - L(w^\star, \tilde{\alpha}^\star) \geq \frac{\lambda}{2}\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{2m}\|\tilde{\alpha}^\star - \alpha^\star\|^2,$$

we obtain

$$\lambda\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \alpha^\star\|^2 + \delta\lambda(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) + \frac{\delta\gamma}{m}(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) \leq 0. \quad (134)$$

Proof of the first claim. We can drop the nonnegative terms on the left-hand side of (134) to obtain

$$\lambda(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) + \frac{\gamma}{m}(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) \leq 0.$$

The two inner product terms on the left-hand side of the inequality above can be expanded as follows:

$$\begin{aligned}
(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) &= (\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - \tilde{w} + \tilde{w} - w^\star) = \|\tilde{w}^\star - \tilde{w}\|^2 + (\tilde{w}^\star - \tilde{w})^T(\tilde{w} - w^\star), \\
(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) &= (\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \tilde{\alpha} + \tilde{\alpha} - \alpha^\star) = \|\tilde{\alpha}^\star - \tilde{\alpha}\|^2 + (\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha} - \alpha^\star).
\end{aligned}$$

Combining them with the last inequality, we have

$$\begin{aligned}
\lambda\|\tilde{w}^\star - \tilde{w}\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \tilde{\alpha}\|^2 &\leq -\lambda(\tilde{w}^\star - \tilde{w})^T(\tilde{w} - w^\star) - \frac{\gamma}{m}(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha} - \alpha^\star) \\
&\leq \frac{\lambda}{2}\left(\|\tilde{w}^\star - \tilde{w}\|^2 + \|\tilde{w} - w^\star\|^2\right) + \frac{\gamma}{2m}\left(\|\tilde{\alpha}^\star - \tilde{\alpha}\|^2 + \|\tilde{\alpha} - \alpha^\star\|^2\right),
\end{aligned}$$

which implies

$$\frac{\lambda}{2}\|\tilde{w}^\star - \tilde{w}\|^2 + \frac{\gamma}{2m}\|\tilde{\alpha}^\star - \tilde{\alpha}\|^2 \leq \frac{\lambda}{2}\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{2m}\|\tilde{\alpha} - \alpha^\star\|^2.$$

Proof of the second claim. We expand the two inner product terms in (134) as follows:

$$\begin{aligned}
(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) &= (\tilde{w}^\star - w^\star + w^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) = \|\tilde{w}^\star - w^\star\|^2 + (w^\star - \tilde{w})^T(\tilde{w}^\star - w^\star), \\
(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) &= (\tilde{\alpha}^\star - \alpha^\star + \alpha^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) = \|\tilde{\alpha}^\star - \alpha^\star\|^2 + (\alpha^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star).
\end{aligned}$$

Then (134) becomes

$$\begin{aligned}
& (1+\delta)\lambda\|\tilde{w}^\star - w^\star\|^2 + (1+\delta)\frac{\Gamma}{m}\|\tilde{w}^\star - w^\star\|^2 \\
\leq\ & \delta\lambda(\tilde{w} - w^\star)^T(\tilde{w}^\star - w^\star) + \frac{\delta\gamma}{m}(\tilde{\alpha} - \alpha^\star)^T(\tilde{\alpha}^\star - \alpha^\star) \\
\leq\ & \delta\left(\lambda\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \alpha^\star\|^2\right)^{1/2}\left(\lambda\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \alpha^\star\|^2\right)^{1/2},
\end{aligned}$$

where in the second inequality we used the Cauchy-Schwarz inequality. Therefore we have

$$\left(\lambda\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \alpha^\star\|^2\right)^{1/2} \leq \frac{\delta}{1+\delta}\left(\lambda\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \alpha^\star\|^2\right)^{1/2},$$

which is the desired result. ■

To simplify notations in the rest of the proof, we let $z = (w, \alpha)$ and define

$$\|z\| = \left(\lambda\|w\|^2 + \frac{\gamma}{m}\|\alpha\|^2\right)^{1/2}.$$

The results of Lemma 13 can be written as

$$\|\tilde{z} - \tilde{z}^\star\| \leq \|\tilde{z} - z^\star\|, \tag{135}$$

$$\|\tilde{z}^\star - z^\star\| \leq \frac{\delta}{1+\delta}\|\tilde{z} - z^\star\|. \tag{136}$$

Next consider the convergence of Algorithm 4, and follow the proof ideas in Balamurugan and Bach (2016, Section D.3).

If we use DSCOVR-SVRG (option 1) in each round of Algorithm 4, then Algorithm 2 is called with initial point $\tilde{z}^{(r)} = (\tilde{w}^{(r)}, \tilde{\alpha}^{(r)})$ and after S stages, it outputs $\tilde{z}^{(r+1)}$ as an approximate saddle point of $L_\delta^{(r)}(w, \alpha)$, which is defined in (46). Then Theorem 1 implies

$$\mathbf{E}\big[\|\tilde{z}^{(r+1)} - \tilde{z}^{\star(r)}\|^2\big] \leq \left(\frac{2}{3}\right)^S \mathbf{E}\big[\|\tilde{z}^{(r)} - \tilde{z}^{\star(r)}\|^2\big], \tag{137}$$

where $\tilde{z}^{\star(r)}$ denotes the unique saddle point of $L_\delta^{(r)}(w, \alpha)$. By Minkowski's inequality, we have

$$\left(\mathbf{E}\big[\|\tilde{z}^{(r+1)} - z^\star\|^2\big]\right)^{1/2} \leq \left(\mathbf{E}\big[\|\tilde{z}^{(r+1)} - \tilde{z}^{\star(r)}\|^2\big]\right)^{1/2} + \left(\mathbf{E}\big[\|\tilde{z}^{\star(r)} - z^\star\|^2\big]\right)^{1/2},$$

where $z^\star$ is the unique saddle point of $L(w, \alpha)$. Using (137), (135) and (136), we obtain

$$\begin{aligned}
\left(\mathbf{E}\big[\|\tilde{z}^{(r+1)} - z^\star\|^2\big]\right)^{1/2} &\leq \left(\frac{2}{3}\right)^{S/2} \left(\mathbf{E}\big[\|\tilde{z}^{(r)} - \tilde{z}^{\star(r)}\|^2\big]\right)^{1/2} + \left(\mathbf{E}\big[\|\tilde{z}^{\star(r)} - z^\star\|^2\big]\right)^{1/2} \\
&\leq \left(\frac{2}{3}\right)^{S/2} \left(\mathbf{E}\big[\|\tilde{z}^{(r)} - z^\star\|^2\big]\right)^{1/2} + \frac{\delta}{1+\delta}\left(\mathbf{E}\big[\|\tilde{z}^{(r)} - z^\star\|^2\big]\right)^{1/2} \\
&= \left[\left(\frac{2}{3}\right)^{S/2} + \frac{\delta}{1+\delta}\right]\left(\mathbf{E}\big[\|\tilde{z}^{(r)} - z^\star\|^2\big]\right)^{1/2},
\end{aligned} \tag{138}$$

Therefore, if $S \geq \frac{2\log(2(1+\delta))}{\log(3/2)}$, we have

$$\left(\frac{2}{3}\right)^{S/2} + \frac{\delta}{1+\delta} \leq \frac{1}{2(1+\delta)} + \frac{\delta}{1+\delta} = \frac{1+2\delta}{2(1+\delta)} = 1 - \frac{1}{2(1+\delta)},$$

which implies

$$\mathbf{E}\big[\|\tilde{z}^{(r+1)} - z^\star\|^2\big] \leq \left(1 - \frac{1}{2(1+\delta)}\right)^2 \mathbf{E}\big[\|\tilde{z}^{(r)} - z^\star\|^2\big]. \tag{139}$$

If we use DISCOVR-SAGA (option 2) in Algorithm 4, then Algorithm 3 is called with initial point $\tilde{z}^{(r)} = (\tilde{w}^{(r)}, \tilde{\alpha}^{(r)})$ and after M steps, it outputs $\tilde{z}^{(r+1)}$ as an approximate saddle point of $L_\delta^{(r)}(w, \alpha)$. Then Theorem 3 implies

$$\mathbf{E}\big[\|\tilde{z}^{(r+1)} - \tilde{z}^{\star(r)}\|^2\big] \leq \frac{4}{3}\left(1 - \frac{1}{3\Gamma_\delta}\right)^M \mathbf{E}\big[\|\tilde{z}^{(r)} - \tilde{z}^{\star(r)}\|^2\big].$$

Using similar arguments as in (138), we have

$$\left(\mathbf{E}\left[\|\tilde{z}^{(r+1)} - z^\star\|\right]^2\right)^{1/2} \;\leq\; \left[\frac{4}{3}\left(1 - \frac{1}{3\Gamma_\delta}\right)^{M/2} + \frac{\delta}{1+\delta}\right]\left(\mathbf{E}\left[\|\tilde{z}^{(r)} - z^\star\|^2\right]\right)^{1/2}.$$

Therefore, if $M \geq 6\log\left(\frac{8(1+\delta)}{3}\right)\Gamma_\delta$, we have

$$\frac{4}{3}\left(1 - \frac{1}{3\Gamma_\delta}\right)^{M/2} + \frac{\delta}{1+\delta} \leq \frac{1}{2(1+\delta)} + \frac{\delta}{1+\delta} = \frac{1+2\delta}{2(1+\delta)} = 1 - \frac{1}{2(1+\delta)},$$

which implies the same inequality in (139).

In summary, using either option 1 or option 2 in Algorithm 4, we have

$$\mathbf{E}\left[\|\tilde{z}^{(r)} - z^\star\|^2\right] \;\leq\; \left(1 - \frac{1}{2(1+\delta)}\right)^{2r} \|\tilde{z}^{(0)} - z^\star\|^2.$$

In order to have $\mathbf{E}\left[\|\tilde{z}^{(r)} - z^\star\|^2\right] \leq \epsilon$, it suffices to have $r \geq (1+\delta)\log\left(\|\tilde{z}^{(0)} - z^\star\|^2/\epsilon\right)$.

References

A. Agarwal and J. C. Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems (NIPS) 24*, pages 873–881, 2011.

Z. Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of 49th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 1200–1205, 2017.

Y. Arjevani and O. Shamir. Communication complexity of distributed convex learning and optimization. In *Advances in Neural Information Processing Systems (NIPS) 28*, pages 1756–1764. 2015.

A. Aytekin, H. R. Feyzmahdavian, and M. Johansson. Analysis and implementatino of an asynchronous optimization algorithm for the parameter server. arXiv:1610.05507, 2016.

P. Balamurugan and F. Bach. Stochastic variance reduction methods for saddle-point problems. In *Advances in Neural Information Processing Systems (NIPS) 29*, pages 1416–1424, 2016.

A. Beck and M. Teboulle. A fast iterative shrinkage-threshold algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, 1989.

S. P. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

A. Chambolle and T. Pock. On the ergodic convergence rates of a first-order primal–dual algorithm. *Mathematical Programming*, pages 1–35, 2015.

W. Chen, Z. Wang, and J. Zhou. Large-scale L-BFGS using MapReduce. In *Advances in Neural Information Processing Systems (NIPS) 27*, pages 1332–1340. 2014.

J. Dean and S. Ghemawat. MapReduce: Simplfied data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems (NIPS) 27*, pages 1646–1654. 2014.

J. C. Duchi, A. Agarwal, and M. J. Wainwright. Dual averaging for distributed optimization: convergence analysis and network scaling. *IEEE Transactions on Automatic Control*, 57(3): 592–606, 2012.

R.-E. Fan and C.-J. Lin. LIBSVM data: Classification, regression and multi-label. URL: http://www.csie.ntu.edu.tw/˜cjlin/libsvmtools/datasets, 2011.

R. Frostig, R. Ge, S. Kakade, and A. Sidford. Un-regularizing: approximate proximal point and faster stochastic algorithms for empirical risk minimization. In *Proceedings of The 32nd International Conference on Machine Learning (ICML)*, pages 2540–2548. 2015.

R. Hannah and W. Yin. More iterations per second, same quality — why asynchronous algorithms may drastically outperform traditional ones. CAM Report 17-50, University of California at Los Angeles, 2017.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of Convex Analysis*. Springer, 2001.

M. Jaggi, V. Smith, M. Takac, J. Terhorst, S. Krishnan, T. Hofmann, and M. I. Jordan. Communication-efficient distributed dual coordinate ascent. In *Advances in Neural Information Processing Systems (NIPS) 27*, pages 3068–3076. 2014.

R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems (NIPS) 26*, pages 315–323. 2013.

G. Lan and Y. Zhou. An optimal randomized incremental gradient method. Technical report, Department of Industrial and System Engineering, University of Florida, July 2015.

N. Le Roux, M. Schmidt, and F. Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems 25*, pages 2672–2680. 2012.

C.-P. Lee, P.-W. Wang, W. Chen, and C.-J. Lin. Limited-memory common-directions method for distributed optimization and its application on empirical risk minimization. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 732–740, 2017.

J. D. Lee, Q. Lin, T. Ma, and T. Yang. Distributed stochastic variance reduced gradient methods and a lower bound for communication complexity. arXiv:1507.07595, 2015.

M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su. Scaling distributed machine learning with the parameter server. In *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 583–598, 2014.

C.-Y. Lin, C.-H. Tsai, C.-P. Lee, and C.-J. Lin. Large-scale logistic regression and linear support vector machines using Spark. In *Proceedings of the IEEE Conference on Big Data*, Washington DC, USA, 2014.

H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems (NIPS) 28*, pages 3384–3392. 2015.

Q. Lin and L. Xiao. An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization. *Computatoinal Optimization and Applications*, 60(3):633–674, 2015.

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

C. Ma, V. Smith, M. Jaggi, M. I. Jordan, P. Richtárik, and M. Takáč. Adding vs. averaging in distributed primal-dual optimization. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning (ICML)*, pages 1973–1982, 2015.

C. Ma, V. Smith, M. Jaggi, M. I. Jordan, P. Richtárik, and M. Takáč. Distributed optimization with arbitrary local solvers. *Optimization Methods and Software*, 32(4):813–848, 2017.

S. Matsushima, H. Yun, X. Zhang, and S. V. N. Vishwanathan. Distributed stochastic optimization of the regularized risk. arXiv:1406.4363, 2014.

H. B. McMahan and M. J. Streeter. Delay-tolerant algorithms for asynchronous distributed online learning. In *Advances in Neural Information Processing Systems (NIPS) 27*, pages 2915–2923, 2014.

X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talwalkar. MLlib: Machine learning in Apache Spark. *Journal of Machine Learning Research*, 17(34):1–7, 2016.

MPI Forum. MPI: a message-passing interface standard, Version 3.0. Document available at `http://www.mpi-forum.org`, 2012.

A. Nedić and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, January 2009.

A. Nedić, A. Olshevsky, and W. Shi. Achieving geometric convergence for distributed optimizaiton over time-varying graphs. arXiv:1607.03218, 2016.

Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer, Boston, 2004.

Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming, Ser. B*, 140:125–161, 2013.

OpenMP Architecture Review Board. OpenMP Application Program Interface, Version 3.1. Available at `http://www.openmp.org`, July 2011.

Z. Peng, Y. Xu, M. Yan, and W. Yin. ARock: An algorithmic framework for asynchronous parallel coordinate updates. *SIAM Journal on Scientific Computing*, 38(5):2851–2879, 2016.

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 (NIPS) 24*, pages 693–701, 2011.

S. J. Reddi, A. Hefny, S. Sra, B. Póczós, and A. J. Smola. On variance reduction in stochastic gradient descent and its asynchronous variants. In *Advances in Neural Information Processing Systems (NIPS) 28*, pages 2647–2655. 2015.

S. J. Reddi, J. Konečný, P. Richtárik, B. Póczós, and A. Smola. AIDE: Fast and communication efficient distributed optimization. arXiv:1608.06879, 2016.

P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1):1–38, 2014.

P. Richtárik and M. Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1):433–484, 2016.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5), 1976.

E. K. Ryu and S. P. Boyd. A primer on monotone operator methods. *Applied and Computational Mathematics: an International Journal*, 15(1):3–43, 2016.

K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3027–3036, Sydney, Australia, 2017.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14:567–599, 2013.

O. Shamir, N. Srebro, and T. Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1000–1008, Bejing, China, 2014.

W. Shi, Q. Ling, G. Wu, and W. Yin. EXTRA: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

S. Sra, A. W. Yu, M. Li, and A. J. Smola. Adadelay: Delay adaptive distributed stochastic optimization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 957–965, 2016.

J. Wang and L. Xiao. Exploiting strong convexity from data with primal-dual first-order algorithms. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3694–3702, Sydney, Australia, 2017.

L. Xiao and S. P. Boyd. Optimal scaling of a gradient method for distributed resource allocation. *Journal of Optimization Theory and Applications*, 129(3):469–488, June 2006.

L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

E. P. Xing, Q. Ho, W. Dai, J. K. Kim, J. Wei, S. Lee, X. Zheng, P. Xie, A. Kumar, and Y. Yu. Petuum: A new platform for distributed machine learning on big data. *IEEE Transactions on Big Data*, 1 (2):49–67, 2015.

T. Yang. Trading computation for communication: Distributed stochastic dual coordinate ascent. In *Advances in Neural Information Processing Systems (NIPS) 26*, pages 629–637. 2013.

A. W. Yu, Q. Lin, and T. Yang. Doubly stochastic primal-dual coordinate method for bilinear saddle-point problem. arXiv:1508.03390, 2015.

H. Yun, H.-F. Yu, C.-J. Hsieh, S. V. N. Vishwanathan, and I. Dhillon. NOMAD: Non-locking, stochastic multi-machine algorithm for asynchronous and decentralized matrix completion. In *Proceedings of the VLDB Endowment*, volume 7, pages 975–986, 2014.

M. Zaharia, R. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica. Apache Spark: A unified engine for big data processing. *Communications of the ACM*, 59(11):56–65, 2016.

Y. Zhang and L. Xiao. DiSCO: Distributed optimization for self-concordant empirical loss. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 362–370, Lille, France, 2015.

Y. Zhang and L. Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. *Journal of Machine Learning Research*, 18(84):1–42, 2017.

Y. Zhang, J. C. Duchi, and M. J. Wainwright. Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research*, 14:3321–3363, 2013.