Cost-efficient SVRG with Arbitrary Sampling

Anonymous Authors[1]

Abstract

We consider the problem of distributed optimization over a network, using a stochastic variance reduced gradient (SVRG) algorithm, where executing every iteration of the algorithm requires computation and exchange of gradients among network nodes. These tasks always consume network resources, including communication bandwidth and battery power, which we model as a general cost function. In this paper, we consider an SVRG algorithm with arbitrary sampling (SVRG-AS), where the nodes are sampled according to some distribution. We characterize the convergence of SVRG-AS, in terms of this distribution. We determine the distribution that minimizes the costs associated with running the algorithm, with provable convergence guarantees. We show that our approach can substantially outperform vanilla SVRG and its variants in terms of both convergence rate and total cost of running the algorithm. We then show how our approach can optimize the mini-batch size to address the tradeoff between low communication cost and fast convergence rate. Comprehensive theoretical and numerical analyses on real datasets reveal that our algorithm can significantly reduce the cost, especially in large and heterogeneous networks. Our results provide important practical insights for using machine learning over Internet-of-Things.

1. Introduction

Consider the problem of optimizing a sum of differentiable functions $\{f_i : \mathbb{R}^d \mapsto \mathbb{R}\}_{i\in[N]}$, with corresponding gradients $\{\boldsymbol{g}_i : \mathbb{R}^d \mapsto \mathbb{R}^d\}_{i\in[N]}$:

$$\boldsymbol{w}^\star = \min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) = \frac{1}{N}\sum\nolimits_{i\in[N]} f_i(\boldsymbol{w})\,. \qquad (1)$$

Such problems frequently arise in statistical learning, in which each f_i could represent a regularized loss over some sampled data points. In practice, such problems are often solved using a gradient-based algorithm. Due to the large scale of many applications, most modern machine-learning approaches distribute the computation loads of finding the N gradients to some computational nodes (also called workers) (Bottou et al., 2018), to enable parallel computations, or simply due to the data not being available at a single place. That is, at iteration k, a subset of workers compute their gradients $\{\boldsymbol{g}_i(\boldsymbol{w}_k)\}_i$ and send them to a central controller (also called the master node), which updates the model and broadcasts the updated parameter $\boldsymbol{w}_{k+1}$ to the workers. One of the most successful class of methods to solve (1), is the classical stochastic gradient descent and its variance-reduced extensions, including stochastic variance-reduced gradient (SVRG) and stochastic average gradient (SAGA) (Bottou et al., 2018; Johnson & Zhang, 2013; Defazio et al., 2014). In this paper, we focus on SVRG.

In a distributed computation setting, running every iteration of the algorithm involves some costs c_i, which could correspond to the number of bits (or energy or latency) needed to send $\{\boldsymbol{g}_i(\boldsymbol{w}_k)\}_i$ or the computational resources needed to compute $\{\boldsymbol{g}_i(\boldsymbol{w}_k)\}_i$. These costs become of paramount importance when we implement machine learning and distributed optimization algorithms on bandwidth and battery-limited wireless networks. In such networks, tight requirements on low end-to-end latency (in autonomous driving), low energy usage (Internet-of-Things), and high reliability (remote industrial operation) may render the ultimate solution, and consequently the distributed algorithm, useless (Jeschke et al., 2017). Our literature review in Section 2 shows that the existing distributed optimization solutions often ignore these important cost terms. In the case of SVRG, the gradient sampling ignores the heterogeneous costs of obtaining $\boldsymbol{g}_i(\boldsymbol{w})$, for different i, as well as the importance of this gradient for the convergence rate of the algorithm. Here, we address this open research problem.

In this paper, we build on SVRG with arbitrary sampling (SVRG-AS), introduced in (Horváth & Richtárik, 2018) where the gradient sampling follows a generic multinomial distribution. Our algorithm allows for a variable inner loop length, reducing the amount of computations/communications of gradients in the inner loop by

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

half, on average, compared to vanilla SVRG. We show that, when each f_i is strongly convex and L_i-smooth, the convergence rate of SVRG-AS is a function of $\bar{L} := \sum_i L_i/N$ instead of $L_{\max} := \max_i L_i$ in the vanilla SVRG (Johnson & Zhang, 2013). Similar results have been proved for SVRG and SAGA in smooth but nonconvex setting (Horváth & Richtárik, 2018) and for SAGA in convex setting (Qian et al., 2019). We then use our novel convergence bounds to design a minimum-cost SVRG-AS algorithm and transform the resulting optimization problem into a linear program. Comprehensive theoretical and numerical analyses on real datasets reveal that the optimal sampling rate of $\boldsymbol{g}_i$ is a function of L_i. We then consider cost functions that model two important use cases: 1) stragglers in the federated learning case and 2) congestion in wireless communications. In both cases, we show that our minimum cost SVRG-AS can significantly outperform the vanilla SVRG and its state-of-the-art variants, including importance sampling ones, in terms of both total costs of running the algorithm and/or convergence rate. In particular, we show that the optimal mini-batch size depends not only on the computational loads and the number of gradient exchanges but also heavily on the communication protocol.

We conclude that a co-design of distributed optimization algorithms and networking protocols is essential for the success of machine learning over networks, including its applications to edge computing, Internet-of-Things, and intelligent transportation systems.

The rest of the paper is organized as follows. Section 2 reviews the related works. Section 3 presents the problem setting, our proposed algorithm, and its theoretical convergence results. We design a minimum-cost SVRG-AS in Section 4 and apply them to real datasets in Section 5. We conclude the paper in Section 6. For the sake of readability, we have moved extra definitions, lemmas, and all the proofs to the appendix.

Notation: Normal font w or W, bold font small-case $\boldsymbol{w}$, bold-font capital letter $\boldsymbol{W}$, and calligraphic font $\mathcal{W}$ denote scalar, vector, matrix, and distribution function, respectively. We let $[N] = \{1, 2, \ldots, N\}$ for any integer N. We denote by $\|\cdot\|$ the l_2 norm, by $\boldsymbol{x}^{\mathrm{T}}$ the transpose of $\boldsymbol{x}$, and by $\mathbb{1}_x$ the indicator function taking 1 when condition x holds. For easier reference, we have provided a table of notations in the appendix.

2. Literature Review

2.1. Communication-efficient Distributed Optimization

To the best of our knowledge, cost-efficient distributed optimization is addressed in the literature only via the notion of communication-efficiency. Example settings include networked control (Hespanha et al., 2007), distributed optimization (Tsitsiklis & Luo, 1987; Rabbat & Nowak, 2005; Zhang et al., 2012; 2015; Wang & Joshi, 2018), and machine learning (Balcan et al., 2012; Zhang et al., 2013; Jordan et al., 2018; Zhu & Lafferty, 2018; Stich et al., 2018; Karimireddy et al., 2019).

In the literature, there are two classes of approaches relevant to this paper: a) quantization of the parameter and gradient vectors in every iteration, and b) eliminating some communications at every step. The first category includes approaches that reduce the number of bits to represent $\boldsymbol{w}_k$ and $\boldsymbol{g}_i(\boldsymbol{w}_k)$, thereby alleviating the communication between the master node and the workers at every iteration, though more iterations may be needed to compensate for quantization errors, compared to the unquantized version. Recent studies have shown that proper quantization approaches can maintain the convergence to the true minimizer, as well as the convergence rate (Bernstein et al., 2018; Kamilov, 2018; De Sa et al., 2018; Stich et al., 2018; Magnússon et al., 2019; Karimireddy et al., 2019).

The second category includes algorithms that eliminate communication between some of the workers and the master node in some iterations (Chen et al., 2018). Chen et al. (2018) proposed lazily aggregated gradient (LAG) for communication-efficient distributed learning in master-worker architectures. In LAG, each worker reports its gradient vector to the master node only if the changes to the gradient from the last step, measured by $l2$ norm, is large enough. That way, some nodes may skip sending their gradients at some iterations, which saves communication resources. Sun et al. (2019) extended LAG by sending quantized gradient vectors, instead of the true values.

To the best of our knowledge, all existing works assess the convergence in terms of the number of iterations, bits transmitted, or gradients exchanged to achieve a certain solution accuracy. However, when solving a machine learning problem over a wireless channel, the main design objectives are usually latency, total energy usage, and reliability, rather than the number of algorithm iterations or bits involved. For instance, compressing gradients may not be the most best solution for handling a federated learning case with stragglers. Similarly, in the presence of a congested wireless network, where sending more packets (e.g., via a higher mini-batch size in our case) leads to more communication failures, we may need a fundamental redesign of the distributed optimization algorithm, to control the number of active workers based on the communication channel conditions, rather than the gradient norm. This paper addresses the new research problem of cost-aware distributed optimization.

2.2. Arbitrary and Importance Sampling Strategies

There has been a recent wave of works on the importance and arbitrary samplings for various stochastic algorithms.

Using the primal-dual gap as a measure, importance sampling has been successfully developed for randomized coordinate descent algorithms to replace the inefficient random coordinate selection of the updates (Nesterov, 2012; Allen-Zhu et al., 2016; Perekrestenko et al., 2017) Konečnỳ et al. (2017). Stich et al. (2017) extended these results to adaptive importance sampling for coordinate descent, where the sampling probability changes over time to cope with the local geometry of the optimization landscape. Gower et al. (2018) introduced a class of variance reduction algorithms based on Jacobian sketching (JacSketch) in every step and developed importance sampling for SAGA in the strongly convex case. Qian et al. (2019) analyzed SAGA with arbitrary sampling in the non-smooth setting. Horváth & Richtárik (2018) analyzed the importance mini-batch sampling for SVRG and SAGA in the nonconvex setting, and Gazagnadou et al. (2019) used the JacSketch algorithm to find the optimal mini-batch size for SAGA in the strongly convex and smooth setting.

Most existing theoretical results suggest that a mini-batch of size 1 gives the best solution, disagreeing with practical implementations, where much faster convergence can often be achieved using larger mini-batch sizes. However, Sebbouh et al. (2019) established optimal batch sizes for SVRG, showing that larger batch-sizes can in fact reduce total complexity (number of iterations required to reach target accuracy). Gazagnadou et al. (2019) showed, both theoretically and experimentally, that SAGA may benefit from a larger mini-batch size. In this paper, we extend those results for SVRG and show that, not only the smoothness and strong-convexity parameters of each f_i affect the optimal mini-batch size, but also the communication link between the workers and the master node heavily influence that number. To the best of our knowledge, this is the first paper that optimizes SVRG (and any distributed optimization algorithm) considering a concrete and realistic communication model.

3. SVRG-AS and Convergence Results

In this section, we present our main algorithm and analyze its performance in two scenarios: running the inner loop of SVRG with either a single gradient or a mini-batch.

3.1. SVRG-AS

The inner loop of SVRG (also called epoch) consists of T iterations, in which the parameter is sequentially updated using the gradient of f_ξ for a randomly chosen index ξ (Johnson & Zhang, 2013). In particular, the master node broadcasts $\widetilde{\boldsymbol{w}}_k$ to the workers at the beginning of the inner loop. It then also broadcasts $\boldsymbol{w}_{k,t-1}$, realizes the random variable $\xi := \xi_{k,t-1} \in [N]$, and receives $\boldsymbol{g}_\xi(\boldsymbol{w}_{k,t-1})$ from worker ξ at every inner loop iteration t. At the end of the inner loop, the master node updates $\widetilde{\boldsymbol{g}}_k$ and $\widetilde{\boldsymbol{w}}_k$ for the next epoch.

Algorithm 1 SVRG-AS

1: **Inputs:** Maximum epoch length T, number of epochs K, N, step size sequence $(\alpha_k)_k$, and probabilities $p_1, \ldots, p_N$.
2: **for** $k = 1, 2, \ldots, K-1$ **do**
3: $\quad \widetilde{\boldsymbol{h}}_k \leftarrow \sum_{i\in[N]} p_i \boldsymbol{h}_i\left(\widetilde{\boldsymbol{w}}_k\right)$
4: $\quad \boldsymbol{w}_{k,0} \leftarrow \widetilde{\boldsymbol{w}}_k$
5: $\quad$ Sample $\zeta := \zeta_k$ uniformly from $\{1, 2, \ldots, T\}$
6: $\quad$ **for** $t = 1, 2, \ldots, \zeta$ **do**
7: $\quad\quad$ Sample $\xi := \xi_{k,t-1}$ from $[N]$ with probability distribution $\mathcal{P}$, compute $\boldsymbol{g}_\xi\left(\boldsymbol{w}\right)$ and $\boldsymbol{h}_\xi\left(\boldsymbol{w}\right) := \boldsymbol{g}_\xi\left(\boldsymbol{w}\right)/Np_\xi$, and send it to the master node, for both $\boldsymbol{w} = \boldsymbol{w}_{k,t-1}$ and $\boldsymbol{w} = \widetilde{\boldsymbol{w}}_k$.
8: $\quad\quad$ Compute $\boldsymbol{w}_{k,t} \leftarrow \boldsymbol{w}_{k,t-1} - \alpha_k \sum_{i=1}^N \mathbb{1}_{i\in\{\xi\}} \left(\boldsymbol{h}_i(\boldsymbol{w}_{k,t-1}) - \boldsymbol{h}_i\left(\widetilde{\boldsymbol{w}}_k\right) + \widetilde{\boldsymbol{h}}_k\right)$
9: $\quad\quad$ Broadcast $\boldsymbol{w}_{k,t}$
10: $\quad$ **end for**
11: $\quad \widetilde{\boldsymbol{w}}_{k+1} \leftarrow \boldsymbol{w}_{k,\zeta}$
12: **end for**
13: **Return:** $\widetilde{\boldsymbol{w}}_K$

In SVRG with arbitrary sampling (SVRG-AS), each ξ is an i.i.d. random variable with stationary multinomial distribution $\mathcal{P} := \mathcal{P}(p_1, p_2, \ldots, p_N)$, with $p_j := \Pr(\xi = j)$. Moreover, instead of updating based on $\boldsymbol{g}_\xi(\boldsymbol{w})$, we update based on its scaled version $\boldsymbol{h}_\xi(\boldsymbol{w}) := \boldsymbol{g}_\xi(\boldsymbol{w})/Np_\xi$. The SVRG-AS algorithm is illustrated in Algorithm 1. Vanilla SVRG then corresponds to $p_j = 1/N$ for all $j \in [N]$. We should emphasize that our SVRG-AS allows a variable inner loop length (due to Lines 5 and 6) and computes only the necessary (first ζ) iterates of the inner loop, as opposed to vanilla SVRG and previous SVRG-AS (Horváth & Richtárik, 2018) where such selection was at the end of the inner loop, leading to extra unnecessary computations/communications of gradients and parameter updates. This change reduces the number of inner loop iterations by half, on average, compared to vanilla SVRG, without affecting the convergence rate, as we show later in Section 4.1.

For the sake of mathematical analysis, we limit the class of objective functions to be strongly convex and smooth, though our approach may be applicable to invex (Karimi et al., 2016) and multi-convex (Xu & Yin, 2013) structures (like a deep neural network training optimization problem).

Assumption 1. We assume that $f(\boldsymbol{w})$ is μ-strongly convex and that each gradient $\boldsymbol{g}_i$ is L_i-Lipschitz for all $i \in [N]$. Namely,

$$\left(\boldsymbol{g}(\boldsymbol{v}) - \boldsymbol{g}(\boldsymbol{w})\right)^{\mathrm{T}}\left(\boldsymbol{v} - \boldsymbol{w}\right) \geq \mu\|\boldsymbol{v} - \boldsymbol{w}\|^2\,, \tag{2a}$$

$$\|\boldsymbol{g}_i(\boldsymbol{v}) - \boldsymbol{g}_i(\boldsymbol{w})\| \leq L_i\|\boldsymbol{v} - \boldsymbol{w}\|\,, \tag{2b}$$

for all $i \in [N]$ and $\boldsymbol{v}$ and $\boldsymbol{w}$, where $\boldsymbol{g} = \sum_{i\in[N]} \boldsymbol{g}_i/N$.

Next, we characterize the convergence behavior of SVRG-AS, given in Algorithm 1. The starting point will be the following lemma, which is based on (Gazagnadou et al., 2019, Definition 2):

Lemma 1 (Expected Smoothness)**.** *Let $\boldsymbol{w}^\star$ be the optimal solution of* (1)*. Let ξ be a draw from a mutinomial distribution $\mathcal{P}(p_1, p_2, \ldots, p_N)$ with N outcomes. There exist a positive number L, hereafter called expected smoothness, such that $\mathbb{E}_{\xi\sim\mathcal{P}}\left[\|\boldsymbol{h}_\xi(\boldsymbol{w}) - \boldsymbol{h}_\xi(\boldsymbol{w}^\star)\|^2\right] \leq 2L\left(f(\boldsymbol{w}) - f(\boldsymbol{w}^\star)\right)$.*

Assuming that each of the functions f_i is L_i-smooth, we can set $L = L_{\max} := \max_i L_i$ and Lemma 3 follows from (Johnson & Zhang, 2013). Here, we extend (Johnson & Zhang, 2013) and show that the convergence is a function of expected smoothness, which can be significantly smaller than $L_{\max}$. As a result, we may use a much larger step size to substantially improve the convergence rate.

Lemma 2. *Suppose that each of the functions f_i is L_i-smooth for all $i \in [N]$. Then L in Lemma 1 respects*

$$L \leq \max_{i\in[N]} \left\{ \frac{L_i}{Np_i} \right\} . \tag{3}$$

Proposition 1. *Minimizing the upper bound of the expected smoothness L yields the constrained problem*

$$\underset{p_1,p_2,\ldots,p_N}{\textit{minimize}} \ \max_{i\in[N]} \left\{ \frac{L_i}{Np_i} \right\} \ \textit{subject to} \ \sum_{i\in[N]} p_i = 1 .$$

The solution is $p_i^\star = L_i/(N\bar{L})$, and the optimal L is $\bar{L}$, where $\bar{L} := \sum_i L_i/N$ is the average of L_i's.

As shown in the following proposition, the step size, and consequently the convergence rate is a function of L. Non-uniform sampling can potentially lead to a faster convergence rate than uniform sampling with $L = L_{\max}$, since $\sum_i L_i/N \leq L_{\max}$. The gain would be more prominent as N increases, unless all L_i's are equal. The latter is often not the case in practice, when the data are non-i.i.d. and the network nodes have their own private datasets (Li et al., 2019). Moreover, Proposition 1 implies that we can adaptively change sampling policy based on local geometry. We only need to track the local smoothness of the local functions, and sample according to the probabilities $\{p_i^\star = L_i/\sum_i L_i\}$ at the point $\widetilde{\boldsymbol{w}}_k$. In the following, however, we assume that vector $\boldsymbol{p} = [p_1, p_2, \ldots, p_N]^{\mathrm{T}}$ is fixed for all iterations. Now, we can characterize the convergence of the SVRG-AS algorithm.

Proposition 2. *Let $\alpha_k < 1/4L$ and $T > 1/(\mu\alpha_k(1 - 4L\alpha_k))$, and set $\Delta_k := \mathbb{E}\left[f(\widetilde{\boldsymbol{w}}_k)\right] - f(\boldsymbol{w}^\star)$. The iterates of Algorithm 1 satisfy for any $k \in [0, K-1]$*

$$\Delta_{k+1} \leq \sigma_k \Delta_k , \quad 0 < \sigma_k = \frac{\frac{1}{\mu T \alpha_k} + 2L\alpha_k}{1 - 2L\alpha_k} < 1 . \tag{4}$$

Corollary 1. *To ensure a contraction factor of at most $\sigma_{\max} < 1$, the step size and the maximum epoch length should satisfy*

$$\alpha_k \leq \frac{\sigma_{\max}}{2L\left(1 + \sigma_{\max}\right)}, \ \textit{and}$$
$$T \geq \frac{1}{\min_{k\in\mathbb{N}} \ \mu\alpha_k\left(\sigma_{\max} - 2L\alpha_k\sigma_{\max} - 2L\alpha_k\right)} . \tag{5}$$

Proposition 2 and Corollary 1 suggest that the convergence of SVRG-AS depends heavily on the expected smoothness and therefore on the sampling probability vector $\boldsymbol{p}$.

3.2. Mini-batch SVRG-AS

An effective approach for reducing the variance of the gradient error, is to use mini-batching in the inner loop of SVRG-AS (Bottou et al., 2018). This would mean letting ξ be a random mini-batch; see Appendix A for a formal definition. Similar to (Horváth & Richtárik, 2018), we consider a stochastic definition of mini-batch size in the sense that $\mathbb{E}[|\xi|] = b$. Although different from the traditional deterministic definition of the mini-batch size, i.e., $|\xi| = b$, this new stochastic model allows for a distributed implementation of the mini-batch SVRG-AS.

The mini-batch SVRG-AS is almost identical to Algorithm 1 except Lines 7 and 8. Line 7 should be changed to "Every worker i with probability p_i, independent of other workers, computes $\boldsymbol{g}_i\left(\boldsymbol{w}\right)$ and $\boldsymbol{h}_i\left(\boldsymbol{w}\right) := \boldsymbol{g}_i\left(\boldsymbol{w}\right)/Np_i$, and sends it to the master node, for both $\boldsymbol{w} = \boldsymbol{w}_{k,t-1}$ and $\boldsymbol{w} = \widetilde{\boldsymbol{w}}_k$. Moreover, $\mathbb{1}_{i\in\{\xi\}}$ in Line 8 should be changed to $\mathbb{1}_{i\in\xi}$ by redefining ξ to be the set of sampled gradients, i.e., $\xi = \{i \mid \boldsymbol{h}_i(\boldsymbol{w}_{k,t-1})$ is sampled$\}$. We have presented this algorithm in the Appendix.

Remark 1. Convergence of mini-batch SVRG-AS is the same as of Proposition 2 with the same definition for L as of Lemma 2. The only difference is that $\sum_i p_i = b$ instead of being 1 in Propositions 1 and 2.

Next, we show how to use these convergence bounds to optimize the operation of SVRG-AS on a network with limited communication resources.

4. Minimum-cost SVRG-AS

In this section, we focus on the design of minimum cost SVRG-AS whose performance is at least equal to that of SVRG. For notational simplicity, we assume a constant step size, i.e., $\alpha_1 = \alpha_2 = \cdots = \alpha_K = \alpha$.

4.1. Outperforming Vanilla SVRG

Let c_i be non-negative real numbers representing the cost of collecting the corresponding gradient $\boldsymbol{g}_i(\boldsymbol{w})$ for any $\boldsymbol{w}$,

and C_k be the cost of running iteration k. Assume that α and T, satisfying the conditions of Proposition 2, are given. We can formulate cost-efficient SVRG-AS as

$$\underset{p_1,p_2,\dots,p_N}{\text{minimize}} \ \mathbb{E}_{\xi\sim\mathcal{P}}\left[C_k\right] = T\sum_{i\in[N]} c_i p_i \,, \tag{6a}$$

$$\text{subject to} \sum_{i\in[N]} p_i = 1 \,, \ \ p_i \geq 0 \ \ \forall i \in [N] \tag{6b}$$

$$\alpha \leq \frac{1}{4\max_i\left\{L_i/Np_i\right\}} \,, \tag{6c}$$

$$\frac{\frac{1}{\mu T\alpha'} + 2\alpha' \max_i\left\{L_i/Np_i\right\}}{1-2\alpha'\max_i\left\{L_i/Np_i\right\}} \leq \frac{\frac{1}{\mu T\alpha'} + 2\alpha' L_{\max}}{1-2\alpha' L_{\max}}$$
$$\text{for every } \ 0\leq\alpha'<1/4L_{\max} \,, \tag{6d}$$

where the objective function is the average sampling cost, and constraint (6d) ensures that the convergence rate of the SVRG-AS is as good as that of SVRG with uniform sampling (i.e., $L = L_{\max}$) for any admissible step-size α'. Constraint (6c) is equivalent to $p_i \geq 4\alpha L_i/N$ for all $i \in [N]$. Moreover,

$$\frac{x_1}{y_1} \geq \frac{x_2}{y_2} \iff \frac{x_1}{x_1+y_1} \geq \frac{x_2}{x_2+y_2} \tag{7}$$

for all $x_1, x_2 \geq 0$ and $y_1, y_2 > 0$ implies that constraint (6d) is equivalent to $L_{\max} \geq \max_i\{L_i/Np_i\}$ and therefore to $p_i \geq L_i/NL_{\max}$ for all $i \in [N]$.

Now, we can rewrite this optimization problem as

$$\underset{p_1,p_2,\dots,p_N}{\text{minimize}} \ \ T\sum_{i\in[N]} c_i p_i \,, \tag{8a}$$

$$\text{subject to} \sum_{i\in[N]} p_i = 1 \,, \tag{8b}$$

$$p_i \geq \frac{4L_i}{N}\max\left\{\alpha, \frac{1}{4L_{\max}}\right\}, \forall i\in[N]. \tag{8c}$$

As we have shown in Appendix B, a sufficient condition for the feasibility of (8) is that $\alpha \leq 1/4\bar{L}$, where $\bar{L} := \sum_{i=1}^N L_i/N$. Define $c_{\min}$ as the minimum of $\{c_i\}_{i\in[N]}$. Take any index j with $c_j = c_{\min}$. A solution to optimization problem (8) is

$$p_i^\star = \begin{cases} 1 - \sum\limits_{i\in[N]\setminus\{j\}} \dfrac{4L_i}{N}\max\left\{\alpha, \dfrac{1}{4L_{\max}}\right\}, \text{if } i=j, \\ \dfrac{4L_i}{N}\max\left\{\alpha, \dfrac{1}{4L_{\max}}\right\}, \ \ \text{otherwise}, \end{cases} \tag{9}$$

and such sequence $(p_i^\star)_i$ exists when $\alpha \leq 1/4\bar{L}$; see Appendix B for a proof. The solution implies that except the node with minimum sampling cost $c_{\min}$, we sample at a rate that linearly depends on the smoothness parameter, L_i. Namely, SVRG-AS prefers taking fewer samples from nodes with smaller L_i.

Notice that we can easily change optimization problems (6) and (8) to find a sampling strategy that ensures a certain contraction for SVRG-AS, namely $\sigma_k \leq \sigma_{\max}$ for some desired $\sigma_{\max}$. If the resulting problem are feasible, namely there exists a sampling strategy for which $\sigma_k \leq \sigma_{\max}$, the solution would be similar to (9). We further study this case in Section 4.2.2.

Moreover, we should highlight that T and α are given constants to this optimization problem. Corollary 1 shows the interplay among $\sigma_{\max}$, T, and α. Generally speaking, a smaller $\sigma_{\max}$ (faster convergence) implies a smaller α, and consequently a larger T. This leads to a new tradeoff in the objective function, as a smaller α may lead to a smaller p_i, for $i \neq j$, and therefore smaller $\sum c_i p_i$, but also a larger T. We can address this tradeoff by optimizing over step-size, which we leave as our future work. It is also worth mentioning that our experiments show that the bounds on T and α for SVRG-AS as well as the vanilla SVRG may be conservative in general. That is, we can violate the inequalities of Corollary 1 by using a larger α, and a smaller T, and still converge to the optimal solution. This observation suggests that T and α may be optimized as hyper-parameters of the algorithm, independent of $\boldsymbol{p}$.

4.2. Use Cases

Here, we design optimal SVRG-AS for two important use cases.

4.2.1. Stragglers

A major disadvantage of Algorithm 1 is the waiting time for slow devices (i.e., stragglers or stale workers). This problem is prominent in ML over wireless networks, due to the hardware constraints and unreliability of some wireless communication links. For example, a node with low battery power may automatically enter the energy-saving mode and drastically reduce its processing and communication resources, affecting the convergence of distributed optimization (Zhang & Simeone, 2019).

To model stragglers, we assign some a high value to c_i for some straggler nodes i, while keeping the rest at a much lower level. Here, we consider SVRG-AS of Algorithm 1 and focus on the mini-batch SVRG-AS in the next use case. Referring to the optimization problem in (6), we obtain the optimal sampling probability given by (9). In particular, the solution keeps sampling from stragglers at a minimal rate, whose value depends on the smoothness L_i for their private dataset. To further improve the robustness to straggler, we may complement our importance sampling with other approaches, like data duplication (Zhang & Simeone, 2019), asynchronous updates (Xie et al., 2019). We numerically investigate the impact of the straggler nodes on vanilla SVRG and our cost-efficient SVRG-AS in Section 5.

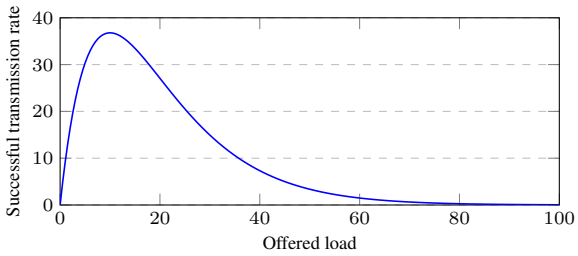


Figure 1: Illustration of rate model for $r_0 = r_1 = 10$.

4.2.2. Congestion in Wireless Communications

In many cases of machine learning over networks, information exchanges happen through a common wireless channel that is shared among all workers. ALOHA and carrier-sense multiple access (CSMA) are important classes of algorithms that regulate how various workers should access the channel and send their data (gradient vectors in this case) without explicit coordination among themselves (Bertsekas et al., 2004). These algorithms are the foundations for connectivity of most modern distributed wireless systems, including Bluetooth and WiFi (Bertsekas et al., 2004).

Define the so-called offered load $\ell > 0$ as the average number of packets (gradients) that the workers inject to the wireless channel (Bertsekas et al., 2004). The successful transmission rate model follows $r = r_0 \ell \exp\{-\ell/r_1\}$ for some positive constants r_0 and r_1 describing the wireless channel and communication protocol (Bertsekas et al., 2004). Figure 1 shows the success rate for $r_0 = r_1 = 10$. This rate model implies that the channel can handle the incoming traffic when almost idle. After a certain point, it becomes "congested" and increasing the number of transmitted packets (offered loads) would only add to the congestion and packet drops, leading to a lower success rate.

By the definition of p_i, $\ell = \sum_i p_i$ for mini-batch SVRG-AS. With the natural assumption of fixed packet size for every gradient vector,[1] latency is inversely proportional to the success rate. Assuming a fixed $\boldsymbol{p}$ for all iterations, the expected cost of K iterations of SVRG-AS is then Kr^{-1}, after which $\Delta_K \leq \sigma^K \Delta_0$. Ensuring $\Delta_k \leq \epsilon_1$ for some constant $\epsilon_1 > 0$ implies $\sigma \leq (\epsilon_1/\Delta_0)^{1/K}$, where we have assumed $\alpha_1 = \alpha_2 = \ldots = \alpha_K = \alpha$. Now, using the definition of σ in Proposition 2, definition of L in (3), and equivalence in (7), we can show that $\Delta_K \leq \epsilon_1$ is equivalent to

$$p_i \geq \frac{2\alpha L_i}{N\epsilon_2} \quad \text{for every } i \in [N], \text{ where}$$
$$\epsilon_2 = \left(\frac{(\epsilon_1/\Delta_0)^{1/K}}{1+(\epsilon_1/\Delta_0)^{1/K}}\right)\left(1+\frac{1}{\mu T\alpha}\right) - \frac{1}{\mu T\alpha}. \quad (10)$$

[1] Assuming 32 bits per coordinate, $\boldsymbol{g}$ has $32d$ bits. Compression algorithms, reviewed in Section 2, reduce this number.

Now, we can formulate our minimum latency SVRG-AS problem as

$$\underset{p_1,p_2,\ldots,p_N}{\text{minimize}} \quad KT\frac{\exp\left\{\frac{\sum_{i\in[N]} p_i}{r_1}\right\}}{r_0 \sum_{i\in[N]} p_i}, \quad (11a)$$

$$\text{subject to} \quad p_i \in \left[\frac{2L_i}{N}\max\left\{\frac{\alpha}{\epsilon_2}, \frac{1}{2L_{\max}}\right\}, 1\right], \ \forall i \in [N]. \quad (11b)$$

Ignoring the constraints, the optimal solution is $\sum_i p_i = r_1$ with the objective of $2.72KT/r_0 r_1$. Moreover, the objective is quasi-convex for positive $\sum_i p_i$ and therefore the closer to the optimal point the better objective. When $r_1 > N$, the optimal solution is then $p_i = 1$ for all i, namely all of the nodes should transmit. In other words, the channel capacity is large enough for all workers to simultaneously report their gradient vectors with manageable cost. However, when $r_1 < N$, we need to control the channel congestion by asking some workers to use smaller (yet feasible) p_i such that $\sum_i p_i = r_1$. If r_1 is too small, the optimal solution may become choosing the lower bound for all p_i, leading to an even smaller mini-batch for every iteration.

Our novel cost-efficient optimization problem (11) suggests that higher transmission probabilities and consequently larger mini-batch sizes $\sum_i p_i$ may not necessarily be optimal, even if we ignore the higher computational costs involved in obtaining extra gradients. To the best of our knowledge, this fundamental design insight has never been properly formulated in the literature.

5. Experimental Results

5.1. Settings

In this section, we numerically characterize the convergence of the SVRG-AS algorithm on some real-world dataset and communication channels. We use the MNIST dataset, which has 60,000 training samples of dimension $d = 784$ and 10 classes corresponding to hand-written digits. We split the dataset into N disjoint subsets of size $\{M_i\}_{i\in[N]}$, and assume each node i has access to its own private dataset of size M_i.

Let tuple $(\boldsymbol{x}_{ij}, y_{ij})$ denote data sample $j \in [M_i]$ belonging to node $i \in [N]$. We consider a logistic regression

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}) = \frac{1}{N}\sum_{i\in[N]} \underbrace{\frac{1}{M_i}\sum_{j\in[M_i]} \ln\left(1+e^{-\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_{ij}y_{ij}}\right)}_{f_i(\boldsymbol{w})}.$$

In Appendix B, we have characterized the smoothness L_i and strong convexity parameter μ for every local function f_i, given its local dataset. We applied the one-versus-all

technique to solve 10 independent binary classification problems (using logistic ridge regression) and obtain 10 optimal classifiers, $\boldsymbol{w}^{(l)}$, one for each digit. Given a test data $\boldsymbol{x}$, we choose label l that maximizes $\left(\boldsymbol{w}^{(l)}\right)^{\mathrm{T}}\boldsymbol{x}$.

to measure the performance, we have considered the convergence of loss function (zero-order stopping criterion) and the gradient norm (first-order stopping criterion) for the training data. Moreover, we monitor the F1 score of $\boldsymbol{w}_k$ on the test dataset to assess the generalization performance, though the main focus of this paper is to develop a communication-efficient *training* algorithm. In the following, we focus on our two use cases, introduced in Section 4, for multiple networking scenarios.[2]

5.2. Stragglers

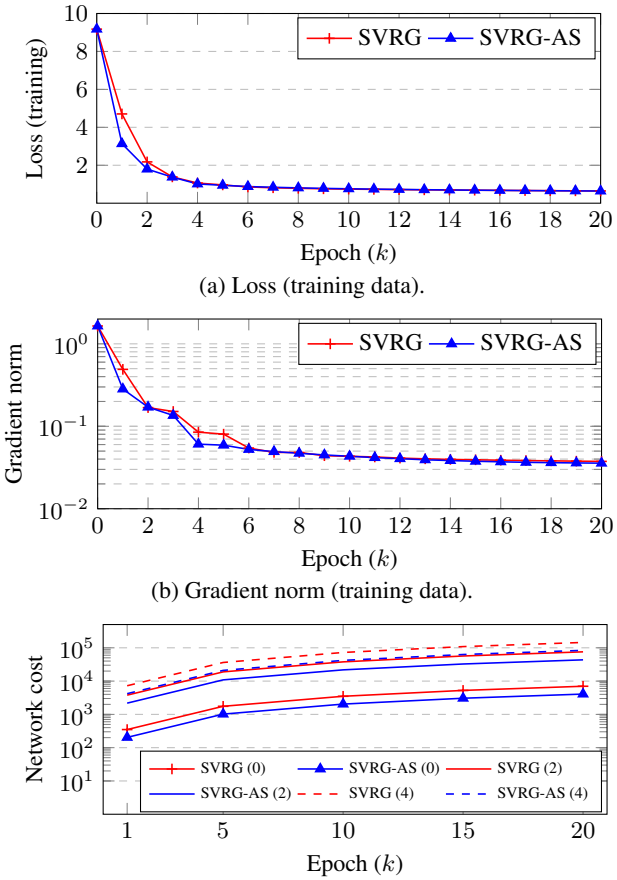
We first assume that $N = 20$. To ensure some statistical difference among the local datasets, so as to ensure different L_i, we keep the samples of only 1 randomly selected class at every node. Consequently, we end up with a training task with around 300 examples in every node (a total of 5927 examples). We then consider three cost models:

- *No straggler:* $c_i = 1$ for all $i \in [N]$;
- *Two straggler:* $c_1 = 0.1, c_{10} = c_{20} = 100$, and for all other $i \in [N]$, $c_i = 1$; and
- *Four stragglers:* $c_1 = 0.1, c_9 = c_{10} = c_{19} = c_{20} = 100$, and for all other $i \in [N]$, $c_i = 1$.

Figure 2 illustrates the convergence of our performance measures in the presence of optimal sampling. SVRG-AS can maintain the convergence to the optimal solution for all cost models, leading to 82% cost reduction of SVRG-AS compared to the vanilla SVRG for the two straggler model. The significant cost reduction in Figure 2(c) is due to optimal sampling as well as better inner loop structure, as discussed in Section 3.

To evaluate the performance of our final solution on all digits, we have reported in Table 1 the F1-score, averaged over all classes. In all cases, the convergence of SVRG-AS was as fast as that of SVRG. We should highlight that we did not try to optimize hyper-parameters to achieve a better F1-score in our experiments.

To further improve the robustness to the stragglers, one may exploit the inherent redundancy of big datasets (Ghadikolaei et al., 2019). In particular, increasing the number of nodes may raise the correlation among the local private datasets, leading to lower dependency to the data stored in a single node. In the example of handwritten digit classification, we can expect less sensitivity of the training task to the lack of information from one straggler node if the information in its local dataset (relevant to the training task) can be captured by others. In the case of the MNIST dataset, when we increase the number of nodes to more than 10, samples of every class will be found in more than one node. Consequently, we can even further reduce p_i for straggler nodes and even ignore their inputs for the outer loop, knowing that their contributions to the training task can be replaced by others with low cost. Using a grid search, we can reduce the cost when $N = 100$ (last row of Table 1) from 476,330 to 20015, leading to 96% less costs to achieve the same solution. We leave the formal design of this extension of cost-efficient SVRG-AS as a future work.



(a) Loss (training data).

(b) Gradient norm (training data).

(c) Network cost.

Figure 2: Convergence results for $T = 15$, assuming digit 3 is the class 1 while all other digits are class -1. In (c), legends (0), (1), and (2) corresponds to no straggler, one straggler, and two stragglers scenarios, respectively.

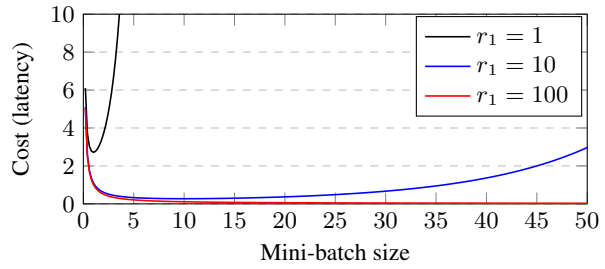
5.3. Congestion in Wireless Communications

Here, we design optimal mini-batch size for SVRG-AS to minimize the network cost for solving our logistic regression problem over a shared wireless media, described

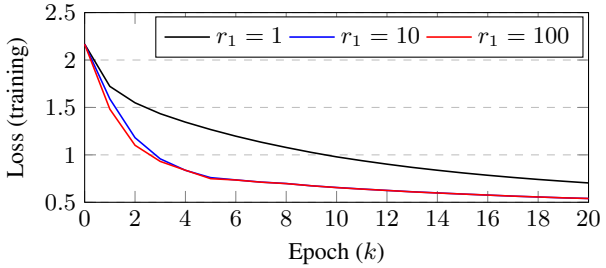
[2] All the simulation codes and datasets are freely available at `https://github.com/hshokrig/papers-SVRG-AS`.

Table 1: F1-score of the MNIST test dataset and cost of training the model using a distributed algorithm. Results are for the two stragglers cost model with $(\alpha_k = 0.2)_k$, $T = 15$, and 20 iterations.

N	SVRG		SVRG-AS	
	F1-score	cost (x1000)	F1-score	cost (x1000)
10	0.864	103.5	0.859	43.5
50	0.860	269.2	0.861	25.9
100	0.841	476.3	0.839	20.0



(a) Network cost of running every inner-loop iteration.



(b) Loss function of optimal mini-batch SVRG-AS, $N = 50$.

Figure 3: Cost and performance of mini-batch updates over a shared wireless network for $T = 15$, assuming digit 3 is the class 1 while all other digits are class -1.

in Section 4.2.2. We consider transmission rate $r = r_0 \ell \exp\{-\ell/r_1\}$ with $\ell = \sum_i p_i$ for mini-batch SVRG-AS. To simulate various network models, we analyze three networking scenarios: high capacity $(r_0 = 1, r_1 = 100)$, medium capacity $(r_0 = 1, r_1 = 10)$, and low capacity $(r_0 = 1, r_1 = 1)$. As a benchmark, we also implement mini-batch SVRG, by picking uniformly at random a subset of cardinality b, among all $\binom{N}{b}$ options, to update every inner loop. Such selection is agnostic to the cost of different subsets.

Figure 3(a) illustrates the cost function for various networking scenarios. The cost function in every inner-loop, shown in Figure 3(a), is quasi-convex in $\sum_i p_i$. Lower network capacity leads to the saturation of the shared wireless media with fewer active nodes. After the network saturation, the costs (latency in our case) of receiving gradients from multiple nodes grows exponentially, making it infeasible to run the iterations in practice. Assuming $N = 100$, a relatively small mini-batch size of 15 leads to per iteration cost of 0.08, 0.3, and around 0.22×10^6 units of cost for high, medium, and low capacity networks. These significantly different costs correspond to the same number of gradients per iteration (namely 15), highlighting the importance of our cost-efficient design compared to the existing approaches that consider only the number of gradients or bits in their designs. Figure 3(b) shows the convergence of mini-batch SVRG-AS with optimal sampling probabilities for $N = 50$. With a low capacity network, our design substantially reduces the mini-batch size to avoid exponentially high usage of the network resources. Consequently, SVRG-AS iterations run with a higher gradient noise, leading to slower convergence. However, this problem can be addressed by exploiting other communication protocols with a higher transmission rate like $r_1 = 10$, in which the optimal mini-batch size is indeed 10 (20% of the nodes) in our experiment. A higher mini-batch size leads to more accurate updates at the inner-iterations and therefore faster convergence. Further increasing the channel capacity to $r_1 = 100$ leads to the optimal mini-batch size of 50 (all nodes). However, the performance improvement is negligible due to a marginal reduction in the variance of the stochastic gradient noise. The latter is because of redundancy in the original dataset and having enough samples from all classes in every mini-batch of size 10 (in the case of $r_1 = 10$).

Our novel cost-efficient optimization problem (11) suggests that higher transmission probabilities and consequently larger mini-batch size $\sum_i p_i$ may not necessarily be optimal, even if we ignore the higher computational costs involved in obtaining extra gradients. To the best of our knowledge, this fundamental design insight has never been properly formulated in the literature.

6. Conclusions

We addressed the problem of minimizing the network costs associated with running a distributed optimization algorithm. In particular, we analyzed the convergence of SVRG-AS with arbitrary sampling and characterized the cost (in terms of the usage of network resources) of finding the solution. We then optimized the sampling probability as well as mini-batch size for SVRG-AS for two networking scenarios: federated learning with straggler nodes and information exchange over a shared wireless network. We have shown that our optimal design can substantially reduce the cost of running SVRG while maintaining an acceptable convergence rate. Our modeling techniques, theoretical results, and detailed discussions, provide important insights to future sustainable networked artificial intelligence and machine learning over large-scale networks, such as Internet-of-Things and cyber-physical systems.

References

Allen-Zhu, Z., Qu, Z., Richtárik, P., and Yuan, Y. Even faster accelerated coordinate descent using non-uniform sampling. In *International Conference on Machine Learning*, pp. 1110–1119, 2016.

Balcan, M. F., Blum, A., Fine, S., and Mansour, Y. Distributed learning, communication complexity and privacy. In *Conference on Learning Theory*, pp. 26–1, 2012.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K., and Anandkumar, A. signSGD: compressed optimisation for non-convex problems. *arXiv preprint arXiv:1802.04434*, 2018.

Bertsekas, D. P., Gallager, R. G., and Humblet, P. *Data networks*, volume 2. Prentice-Hall International New Jersey, 2004.

Bottou, L., Curtis, F., and Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2): 223–311, 2018.

Chen, T., Giannakis, G., Sun, T., and Yin, W. Lag: Lazily aggregated gradient for communication-efficient distributed learning. In *Advances in Neural Information Processing Systems*, pp. 5050–5060, 2018.

De Sa, C., Leszczynski, M., Zhang, J., Marzoev, A., Aberger, C. R., Olukotun, K., and Ré, C. High-accuracy low-precision training. *arXiv preprint arXiv:1803.03383*, 2018.

Defazio, A., Bach, F., and Lacoste-Julien, S. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pp. 1646–1654, 2014.

Gazagnadou, N., Gower, R. M., and Salmon, J. Optimal mini-batch and step sizes for SAGA. *arXiv preprint arXiv:1902.00071*, 2019.

Ghadikolaei, H. S., Ghauch, H., Fischione, C., and Skoglund, M. Learning and data selection in big datasets. In *Proc. International Conference on Machine Learning (ICML)*, pp. 2191–2200, Jun 2019.

Gower, R. M., Richtárik, P., and Bach, F. Stochastic quasi-gradient methods: Variance reduction via Jacobian sketching. *arXiv preprint arXiv:1805.02632*, 2018.

Hespanha, J. P., Naghshtabrizi, P., and Xu, Y. A survey of recent results in networked control systems. *Proceedings of the IEEE*, 95(1):138–162, 2007.

Horváth, S. and Richtárik, P. Nonconvex variance reduced optimization with arbitrary sampling. *arXiv preprint arXiv:1809.04146*, 2018.

Jeschke, S., Brecher, C., Meisen, T., Özdemir, D., and Eschert, T. Industrial internet of things and cyber manufacturing systems. In *Industrial Internet of Things*, pp. 3–19. Springer, 2017.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pp. 315–323, 2013.

Jordan, M. I., Lee, J. D., and Yang, Y. Communication-efficient distributed statistical inference. *Journal of the American Statistical Association*, 2018.

Kamilov, U. S. signProx: One-bit proximal algorithm for nonconvex stochastic optimization. *arXiv preprint arXiv:1807.08023*, 2018.

Karimi, H., Nutini, J., and Schmidt, M. Linear convergence of gradient and proximal-gradient methods under the Polyak-Lojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 795–811. Springer, 2016.

Karimireddy, S. P., Rebjock, Q., Stich, S. U., and Jaggi, M. Error feedback fixes signSGD and other gradient compression schemes. *arXiv preprint arXiv:1901.09847*, 2019.

Konečnỳ, J., Qu, Z., and Richtárik, P. Semi-stochastic coordinate descent. *optimization Methods and Software*, 32(5):993–1005, 2017.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. *arXiv preprint arXiv:1908.07873*, 2019.

Magnússon, S., S. Ghadikolaei, H., and Li, N. On maintaining linear convergence of distributed learning and optimization under limited communication. *arXiv preprint arXiv:1902.11163*, 2019.

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

Perekrestenko, D., Cevher, V., and Jaggi, M. Faster coordinate descent via adaptive importance sampling. *arXiv preprint arXiv:1703.02518*, 2017.

Qian, X., Qu, Z., and Richtárik, P. SAGA with arbitrary sampling. *arXiv preprint arXiv:1901.08669*, 2019.

Rabbat, M. G. and Nowak, R. D. Quantized incremental algorithms for distributed optimization. *IEEE Journal on Selected Areas in Communications*, 23(4):798–808, 2005.

Sebbouh, O., Gazagnadou, N., Jelassi, S., Bach, F., and Gower, R. Towards closing the gap between the theory and practice of SVRG. *arXiv preprint arXiv:1901.09401*, 2019.

Stich, S. U., Raj, A., and Jaggi, M. Safe adaptive importance sampling. In *Advances in Neural Information Processing Systems*, pp. 4381–4391, 2017.

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. Sparsified sgd with memory. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 4447–4458. Curran Associates, Inc., 2018.

Sun, J., Chen, T., Giannakis, G. B., and Yang, Z. Communication-efficient distributed learning via lazily aggregated quantized gradients. In *Advances in Neural Information Processing Systems*, 2019.

Tsitsiklis, J. N. and Luo, Z.-Q. Communication complexity of convex optimization. *Journal of Complexity*, 3(3): 231–243, 1987.

Wang, J. and Joshi, G. Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

Xie, C., Koyejo, S., and Gupta, I. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019.

Xu, Y. and Yin, W. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–1789, 2013.

Zhang, J. and Simeone, O. LAGC: Lazily aggregated gradient coding for straggler-tolerant and communication-efficient distributed learning. *arXiv preprint arXiv:1905.09148*, 2019.

Zhang, S., Choromanska, A. E., and LeCun, Y. Deep learning with elastic averaging SGD. In *Advances in Neural Information Processing Systems*, pp. 685–693, 2015.

Zhang, Y., Wainwright, M. J., and Duchi, J. C. Communication-efficient algorithms for statistical optimization. In *Advances in Neural Information Processing Systems*, pp. 1502–1510, 2012.

Zhang, Y., Duchi, J., Jordan, M. I., and Wainwright, M. J. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In *Advances in Neural Information Processing Systems*, pp. 2328–2336, 2013.

Zhu, Y. and Lafferty, J. Distributed nonparametric regression under communication constraints. In *Proc. International Conference on Machine Learning (ICML)*, pp. 6009–6017, Jul 2018.

Appendices for "Cost-efficient SVRG with Arbitrary Sampling"

A. Useful Definitions and Lemmas

Table 2 lists the main symbols used in the paper.

Table 2: Summary of main notations.

Symbol	Definition
N	Number of worker nodes
K	Number of (outer) iterations
T	Maximum epoch length
$\boldsymbol{w}$	Global model (parameters)
f_i	Local loss function of node i
$\boldsymbol{g}_i(\boldsymbol{w})$	Gradient of node i at point $\boldsymbol{w}$
$\boldsymbol{g}(\boldsymbol{w})$	Average gradient of all the nodes $(\sum_i \boldsymbol{g}_i(\boldsymbol{w})/N)$
$(\boldsymbol{x}_{ij}, y_{ij})$	Data sample j of node i and its label
$\mathcal{P}$	Sampling policy of the master node
p_i	Sampling probability of node i
α_k	Step size (learning rate) at iteration k
L	Expected smoothness
$\xi_{k,t-1}$	Sampled node(s) at iteration k, inner loop iteration t

As we recall, the random variables $\xi_{k,t} \sim \mathcal{P}$ are all pairwise independent and identically distributed (i.i.d) and represent a sampling from the set $[N]$ according to the distribution $\mathcal{P}$, taken at each inner loop iteration of Algorithm 1. For the sake of analysis, set

$$\boldsymbol{v}_{k,t} = \begin{cases} \widetilde{\boldsymbol{w}}_k, & t = 0, \\ \boldsymbol{v}_{k,t-1} - \alpha_k \left(\boldsymbol{h}_{\xi_{k,t-1}}\left(\boldsymbol{v}_{k,t-1}\right) - \boldsymbol{h}_{\xi_{k,t-1}}\left(\widetilde{\boldsymbol{w}}_k\right) + \widetilde{\boldsymbol{h}}_k \right), & 1 \le t \le T+1, \end{cases}$$

where $\widetilde{\boldsymbol{w}}_k$ and $\widetilde{\boldsymbol{h}}_k = \sum_{i\in[N]} p_i \boldsymbol{h}_i\left(\widetilde{\boldsymbol{w}}_k\right)$ are given by Algorithm 1, and $\boldsymbol{h}_i(\boldsymbol{w}) := \boldsymbol{g}_i(\boldsymbol{w})/Np_i$. Notice that these new variables $\boldsymbol{v}_{k,t}$ have the same values as the variables $\boldsymbol{w}_{k,t}$ in Algorithm 1. We will use the new variables in the convergence proof of the algorithm, since doing so will simplify some parts of the proof.

Definition 1 (Sampling (Qian et al., 2019)). A mini-batch sampling ξ is a random set-valued mapping, with possible values being the subsets of $[N]$. Given a sampling ξ, we let $p_i := \Pr(i \in \xi)$. We say ξ is proper if $p_i > 0$ for all $i \in [N]$.

Lemma 3. *For any set of vectors $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n\} \subseteq \mathbb{R}^d$, we have that $\| \sum_{i=1}^n \boldsymbol{w}_i \|^2 \le n \sum_{i=1}^n \|\boldsymbol{w}_i\|^2$.*

Lemma 4 (Variance decomposition inequality). *Let w be a random variable. Then*

$$\mathbb{E}\left[\|\boldsymbol{w} - \mathbb{E}[\boldsymbol{w}]\|^2\right] = \mathbb{E}\left[\|\boldsymbol{w}\|^2\right] - \|\mathbb{E}[\boldsymbol{w}]\|^2 \le \mathbb{E}\left[\|\boldsymbol{w}\|^2\right].$$

Lemma 5. *Consider the variables $\boldsymbol{v}_{k,t}$ above, for $1 \le t \le T$. We have that*

$$\mathbb{E}\left[\left\| \boldsymbol{h}_{\xi_{k,t-1}}\left(\boldsymbol{v}_{k,t-1}\right) - \boldsymbol{h}_{\xi_{k,t-1}}\left(\widetilde{\boldsymbol{w}}_k\right) + \widetilde{\boldsymbol{h}}_k \right\|^2 \mid \xi_{k,t-1} \right] \le 4L\left(f\left(\boldsymbol{v}_{k,t-1}\right) + f\left(\widetilde{\boldsymbol{w}}_k\right) - 2f\left(\boldsymbol{w}^\star\right)\right),$$

where L is the expected smoothness constant in Lemma 1.

Lemma 6. *Consider the variables $\boldsymbol{v}_{k,t}$ above, for $1 \le t \le T$, and suppose that $\boldsymbol{w}^\star$ is the minimizer of f, i.e., that $\nabla f(\boldsymbol{w}^\star) = 0$. We have that*

$$\mathbb{E}\left[\left\|\boldsymbol{v}_{k,t} - \boldsymbol{w}^\star\right\|^2 \mid \xi_{k,t-1}\right] \le \left\|\boldsymbol{v}_{k,t-1} - \boldsymbol{w}^\star\right\|^2 + 4L\alpha_k^2\left(f\left(\boldsymbol{v}_{k,t-1}\right) + f\left(\widetilde{\boldsymbol{w}}_k\right) - 2f\left(\boldsymbol{w}^\star\right)\right) - 2\alpha_k\left(f\left(\boldsymbol{v}_{k,t-1}\right) - f\left(\boldsymbol{w}^\star\right)\right).$$

B. Proofs

B.1. Lemma 2

Since each function f_i is L_i-smooth, it follows that

$$f_i(\boldsymbol{y}) \geq f_i(\boldsymbol{x}) + \langle \boldsymbol{g}_i(\boldsymbol{x}), \boldsymbol{y} - \boldsymbol{x} \rangle + \frac{1}{2L_i}\|\boldsymbol{g}_i(\boldsymbol{y}) - \boldsymbol{g}_i(\boldsymbol{x})\|^2,$$

for any i, $\boldsymbol{x}$ and $\boldsymbol{y}$. If we set $\boldsymbol{x} = \boldsymbol{w}^\star, \boldsymbol{y} = \boldsymbol{w}$, and rearrange this inequality, we end up with

$$\|\boldsymbol{g}_i(\boldsymbol{w}) - \boldsymbol{g}_i(\boldsymbol{w}^\star)\|^2 \leq 2L_i \left(f_i(\boldsymbol{w}) - f_i(\boldsymbol{w}^\star) - \langle \boldsymbol{g}_i(\boldsymbol{w}^\star), \boldsymbol{w} - \boldsymbol{w}^\star \rangle\right).$$

Therefore, for any random variable $\xi \sim \mathcal{P}$,

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{h}_\xi(\boldsymbol{w}) - \boldsymbol{h}_\xi(\boldsymbol{w}^\star)\|^2 \mid \xi\right] &= \mathbb{E}\left[\left\|\frac{\boldsymbol{g}_\xi(\boldsymbol{w}) - \boldsymbol{g}_\xi(\boldsymbol{w}^\star)}{Np_\xi}\right\|^2 \mid \xi\right] \\
&\leq \sum_{i\in[N]} \frac{2L_i}{N^2 p_i}\left(f_i(\boldsymbol{w}) - f_i(\boldsymbol{w}^\star) - \langle \boldsymbol{g}_i(\boldsymbol{w}^\star), \boldsymbol{w} - \boldsymbol{w}^\star\rangle\right) \\
&\leq 2\max_{1\leq i\leq N}\left\{\frac{L_i}{N^2 p_i}\right\}\sum_{i=1}^n f_i(\boldsymbol{w}) - f_i(\boldsymbol{w}^\star) - \langle \boldsymbol{g}_i(\boldsymbol{w}^\star), \boldsymbol{w} - \boldsymbol{w}^\star\rangle \\
&= 2\max_{i\in[N]}\left\{\frac{L_i}{Np_i}\right\}\left(f(\boldsymbol{w}) - f(\boldsymbol{w}^\star)\right),
\end{aligned}$$

leading to $L \leq \max\limits_{i\in[N]}\{L_i/Np_i\}$, where in the last step we used that $f = \frac{1}{N}\sum_{i\in[N]} f_i$ and $\boldsymbol{g}(\boldsymbol{w}^\star) = \boldsymbol{0}$.

B.2. Lemma 3

Write $\|\sum_{i=1}^n w_i\|^2 = n^2\|\frac{1}{n}\sum_{i=1}^n w_i\|^2$ and use Jensen's inequality with the convexity of the norm squared to conclude the lemma.

B.3. Lemma 5

Lemma 3 yields

$$\begin{aligned}
\mathbb{E}\left[\left\|\boldsymbol{h}_{\xi_{k,t-1}}(\boldsymbol{v}_{k,t-1}) - \boldsymbol{h}_{\xi_{k,t-1}}(\widetilde{\boldsymbol{w}}_k) + \widetilde{\boldsymbol{h}}_k\right\|^2 \mid \xi_{k,t-1}\right] &\leq 2\mathbb{E}\left[\left\|\boldsymbol{h}_{\xi_{k,t-1}}(\boldsymbol{v}_{k,t-1}) - \boldsymbol{h}_{\xi_{k,t-1}}(\boldsymbol{w}^\star)\right\|^2 \mid \xi_{k,t-1}\right] \\
&\quad + 2\mathbb{E}\left[\left\|\boldsymbol{h}_{\xi_{k,t-1}}(\boldsymbol{w}^\star) - \boldsymbol{h}_{\xi_{k,t-1}}(\widetilde{\boldsymbol{w}}_k) + \widetilde{\boldsymbol{h}}_k\right\|^2 \mid \xi_{k,t-1}\right] \\
&\overset{(a)}{\leq} 2\mathbb{E}\left[\left\|\boldsymbol{h}_{\xi_{k,t-1}}(\boldsymbol{v}_{k,t-1}) - \boldsymbol{h}_{\xi_{k,t-1}}(\boldsymbol{w}^\star)\right\|^2 \mid \xi_{k,t-1}\right] \\
&\quad + 2\mathbb{E}\left[\left\|\boldsymbol{h}_{\xi_{k,t-1}}(\boldsymbol{w}^\star) - \boldsymbol{h}_{\xi_{k,t-1}}(\widetilde{\boldsymbol{w}}_k)\right\|^2 \mid \xi_{k,t-1}\right] \\
&\overset{(b)}{\leq} 4L\left(f(\boldsymbol{v}_{k,t-1}) + f(\widetilde{\boldsymbol{w}}_k) - 2f(\boldsymbol{w}^\star)\right),
\end{aligned}$$

where (a) is due to the variance decomposition inequality (Lemma 4) and the definition of $\boldsymbol{h}(\widetilde{\boldsymbol{w}})$, and (b) is due to Lemma 1. For (a), recall that $\widetilde{\boldsymbol{h}}_k = \sum_i p_i \boldsymbol{h}_i(\widetilde{\boldsymbol{w}}_k) = \mathbb{E}\left[\boldsymbol{h}_\xi(\widetilde{\boldsymbol{w}}_k) \mid \xi\right]$, for any random variable $\xi \sim \mathcal{P}$, which applies to $\xi_{k,t-1}$.

B.4. Lemma 6

The inner loop of Algorithm 1 yields

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{v}_{k,t}-\boldsymbol{w}^\star\|^2 \mid \xi_{k,t-1}\right] &= \mathbb{E}\left[\|\boldsymbol{v}_{k,t-1}-\alpha_k\left(\boldsymbol{h}_{\xi_{k,t-1}}\left(\boldsymbol{v}_{k,t-1}\right)-\boldsymbol{h}_{\xi_{k,t-1}}\left(\widetilde{\boldsymbol{w}}_k\right)+\widetilde{\boldsymbol{h}}_k\right)-\boldsymbol{w}^\star\|^2 \mid \xi_{k,t-1}\right] \\
&= \mathbb{E}\Bigg[\|\boldsymbol{v}_{k,t-1}-\boldsymbol{w}^\star\|^2+\alpha_k^2\left\|\boldsymbol{h}_{\xi_{k,t-1}}\left(\boldsymbol{v}_{k,t-1}\right)-\boldsymbol{h}_{\xi_{k,t-1}}\left(\widetilde{\boldsymbol{w}}_k\right)+\widetilde{\boldsymbol{h}}_k\right\|^2 \\
&\qquad -2\alpha_k\left(\boldsymbol{v}_{k,t-1}-\boldsymbol{w}^\star\right)^{\mathrm{T}}\left(\boldsymbol{h}_{\xi_{k,t-1}}\left(\boldsymbol{v}_{k,t-1}\right)-\boldsymbol{h}_{\xi_{k,t-1}}\left(\widetilde{\boldsymbol{w}}_k\right)+\widetilde{\boldsymbol{h}}_k\right) \mid \xi_{k,t-1}\Bigg] \\
&\overset{(a)}{\leq} \|\boldsymbol{v}_{k,t-1}-\boldsymbol{w}^\star\|^2+4L\alpha_k^2\left(f\left(\boldsymbol{v}_{k,t-1}\right)+f\left(\widetilde{\boldsymbol{w}}_k\right)-2f\left(\boldsymbol{w}^\star\right)\right) \\
&\quad -2\alpha_k\left(\boldsymbol{v}_{k,t-1}-\boldsymbol{w}^\star\right)^{\mathrm{T}}\mathbb{E}\left[\boldsymbol{h}_\xi\left(\boldsymbol{v}_{k,t-1}\right)-\boldsymbol{h}_\xi\left(\widetilde{\boldsymbol{w}}_k\right)+\widetilde{\boldsymbol{h}}_k \mid \xi_{k,t-1}\right] \\
&\overset{(b)}{=} \|\boldsymbol{v}_{k,t-1}-\boldsymbol{w}^\star\|^2+4L\alpha_k^2\left(f\left(\boldsymbol{v}_{k,t-1}\right)+f\left(\widetilde{\boldsymbol{w}}_k\right)-2f\left(\boldsymbol{w}^\star\right)\right) \\
&\quad -2\alpha_k\left(\boldsymbol{v}_{k,t-1}-\boldsymbol{w}^\star\right)^{\mathrm{T}}\boldsymbol{g}\left(\boldsymbol{v}_{k,t-1}\right) \\
&\overset{(c)}{\leq} \|\boldsymbol{v}_{k,t-1}-\boldsymbol{w}^\star\|^2+4L\alpha_k^2\left(f\left(\boldsymbol{v}_{k,t-1}\right)+f\left(\widetilde{\boldsymbol{w}}_k\right)-2f\left(\boldsymbol{w}^\star\right)\right) \\
&\quad -2\alpha_k\left(f\left(\boldsymbol{v}_{k,t-1}\right)-f\left(\boldsymbol{w}^\star\right)\right)\,,
\end{aligned}$$

where (a) is due to Lemma 5, and the independence of $\boldsymbol{w}_{k,t-1}$ from $\xi_{k,t-1}$, and (c) is due to the convexity of f. Equality (b) follows from

$$\mathbb{E}\left[\boldsymbol{h}_{\xi_{k,t-1}}\left(\boldsymbol{v}_{k,t-1}\right)-\boldsymbol{h}_{\xi_{k,t-1}}\left(\widetilde{\boldsymbol{w}}_k\right)+\widetilde{\boldsymbol{h}}_k \mid \xi_{k,t-1}\right]=\sum_{i=1}^N p_i\boldsymbol{h}_i\left(\boldsymbol{v}_{k,t-1}\right)=\boldsymbol{g}\left(\boldsymbol{w}_{k,t-1}\right),$$

where we have used the fact that $\widetilde{\boldsymbol{h}}_k=\mathbb{E}\left[\boldsymbol{h}_\xi\left(\widetilde{\boldsymbol{w}}_k\right) \mid \xi\right]$, where $\xi\sim\mathcal{P}$ is a sampling of the set $[N]$ according to the distribution $\mathcal{P}$.

B.5. Proposition 1

Using proof by contradiction, it is relatively easy to establish that $p_i^\star/L_i=a$ for all $i\in[N]$ and some constant $a>0$. Constraint $\sum_i p_i^\star=1$ yields $a=1/\sum_i L_i=1/N\bar{L}$, completing the proof.

B.6. Proposition 2

The proof sketch is similar to the convergence of the original SVRG algorithm (Johnson & Zhang, 2013). Recall that ζ_k is uniformly sampled from $\{1,\ldots,T\}$, and $\widetilde{\boldsymbol{w}}_{k+1}=\boldsymbol{w}_{k,\zeta_k}=\boldsymbol{v}_{k,\zeta_k}$. That is,

$$\mathbb{E}\left[f(\widetilde{\boldsymbol{w}}_{k+1})\right]=\frac{1}{T}\sum_{t=1}^T f(\boldsymbol{v}_{k,t})\,. \tag{A.1}$$

Use the inequality for $\mathbb{E}\left[\|\boldsymbol{v}_{k,t}-\boldsymbol{w}^\star\|^2 \mid \xi_{t-1}\right]$ in Lemma 6, together with the tower law $\mathbb{E}[X]=\mathbb{E}[\mathbb{E}[X \mid Y]]$, and sum them over the T iterations of one epoch:

$$\begin{aligned}
\sum_{t=1}^{T+1}\mathbb{E}\left[\|\boldsymbol{v}_{k,t}-\boldsymbol{w}^\star\|^2\right] &\leq \sum_{t=1}^{T+1}\mathbb{E}\left[\|\boldsymbol{v}_{k,t-1}-\boldsymbol{w}^\star\|^2\right] \\
&\quad +\mathbb{E}\left[\sum_{t=1}^{T+1}4L\alpha_k^2\left(f\left(\boldsymbol{v}_{k,t-1}\right)+f\left(\widetilde{\boldsymbol{w}}_k\right)-2f\left(\boldsymbol{w}^\star\right)\right)-2\alpha_k\left(f\left(\boldsymbol{v}_{k,t-1}\right)-f\left(\boldsymbol{w}^\star\right)\right)\right] \\
&\overset{(A.1)}{=} \sum_{t=0}^{T}\mathbb{E}\left[\|\boldsymbol{v}_{k,t}-\boldsymbol{w}^\star\|^2\right]+\mathbb{E}\left[4LT\alpha_k^2\left(f(\widetilde{\boldsymbol{w}}_{k+1})+f\left(\widetilde{\boldsymbol{w}}_k\right)-2f\left(\boldsymbol{w}^\star\right)\right)-2T\alpha_k\left(f(\widetilde{\boldsymbol{w}}_{k+1})-f\left(\boldsymbol{w}^\star\right)\right)\right. \\
&\quad \left.+\,4L\alpha_k^2\left(f(\boldsymbol{w}_{k,0})+f\left(\widetilde{\boldsymbol{w}}_k\right)-2f\left(\boldsymbol{w}^\star\right)\right)-2\alpha_k\left(f(\boldsymbol{w}_{k,0})-f\left(\boldsymbol{w}^\star\right)\right)\right]\,.
\end{aligned}$$

Cancelling similar terms from both sides, noting that $\boldsymbol{w}_{k,0} = \widetilde{\boldsymbol{w}}_k$, and setting $\Delta_k := \mathbb{E}\left[f(\widetilde{\boldsymbol{w}}_k)\right] - f(\boldsymbol{w}^\star)$, yields

$$0 \leq \mathbb{E}\left[\|\boldsymbol{v}_{k,T} - \boldsymbol{w}^\star\|^2\right] \leq \mathbb{E}\left[\|\widetilde{\boldsymbol{w}}_k - \boldsymbol{w}^\star\|^2\right] + 4LT\alpha_k^2\left(\Delta_{k+1} + \Delta_k\right) - 2T\alpha_k\Delta_{k+1} + 8L\alpha_k^2\Delta_k - 2\alpha_k\Delta_k$$
$$\overset{\text{(a)}}{\leq} \frac{2\Delta_k}{\mu} + 4LT\alpha_k^2\left(\Delta_{k+1} + \Delta_k\right) - 2T\alpha_k\Delta_{k+1} + \left(8L\alpha_k^2 - 2\alpha_k\right)\Delta_k\,.$$

where (a) follows from strong convexity of f. After rearranging, we end up with

$$\Delta_{k+1} \leq \left(\frac{\frac{2}{\mu} + 4LT\alpha_k^2 + 8L\alpha_k^2 - 2\alpha_k}{2T\alpha_k - 4LT\alpha_k^2}\right)\Delta_k = \left(\frac{\frac{1}{\mu T\alpha_k} + 2L\alpha_k + \frac{4L\alpha_k - 1}{T}}{1 - 2L\alpha_k}\right)\Delta_k\,, \tag{A.2}$$

assuming $\alpha_k \in (0, 1/2L)$ to ensure a positive denominator. Forcing the contraction factor to be in interval $(0, 1)$ yields $T > 0$ and $\alpha < 1/2L$ to meet the lower limit and

$$T > \frac{1 + \mu\alpha_k(4L\alpha_k - 1)}{\mu\alpha_k\left(1 - 4L\alpha_k\right)}\,, \tag{A.3}$$

to meet the upper limit. Moreover, $T > 0$ in (A.3) implies $\alpha_k < 1/4L$, and therefore $4L\alpha_k - 1 \leq 0$, which allows us to use the bounds

$$T > \frac{1}{\mu\alpha_k\left(1 - 4L\alpha_k\right)}\,, \quad \text{and} \tag{A.4}$$

$$\Delta_{k+1} \leq \left(\frac{\frac{1}{\mu T\alpha_k} + 2L\alpha_k}{1 - 2L\alpha_k}\right)\Delta_k\,, \tag{A.5}$$

which is identical to the one in the original SVRG paper (Johnson & Zhang, 2013).

B.7. Corollary 1

From Proposition 2, we set

$$\frac{\frac{1}{\mu T\alpha_k} + 2L\alpha_k}{1 - 2L\alpha_k} \leq \sigma_{\max}$$

and obtain

$$T \geq \frac{1}{\mu\alpha_k\left(\sigma_{\max} - 2L\alpha_k\sigma_{\max} - 2L\alpha_k\right)}\,.$$

Equation (5) immediately follows. Notice that positivity of the denominator implies

$$\alpha_k < \frac{\sigma_{\max}}{2L(1 + \sigma_{\max})}\,,$$

for some positive $\sigma_{\max} < 1$. This condition automatically implies $\alpha_k < 1/4L$ (from $\sigma/(1+\sigma)$ being monotonically increasing for $0 < \sigma \leq 1$), under which Proposition 2 holds.

B.8. Optimization Problem (8)

Let $\beta_i := 4L_i \max\left\{\alpha, 1/4L_{\max}\right\}/N$ and $\bar{L} := \sum_{i=1}^N L_i/N$. Feasibility of (8) is ensured if and only if $\sum_{i\in[N]}\beta_i \leq 1$, since the constraint (8c) is equivalent to $\beta_i \leq p_i$, and we should have $\sum_i p_i = 1$. Consequently,

$$\max\left\{4\alpha\bar{L}, \frac{\bar{L}}{L_{\max}}\right\} = \sum_{i\in[N]}\beta_i \leq 1\,. \tag{A.6}$$

Recall that $\bar{L} \leq L_{\max}$, so that $\bar{L}/L_{\max} \leq 1$. Since $\alpha \leq 1/4\bar{L}$, it follows that (A.6) is satisfied, meaning that the problem is always feasible. Let $(p_1, \ldots, p_N)$ be the probability distribution given in (9), and notice that it satisfies $p_i = \beta_i$ for every

$i \neq j$, and $p_j = 1 - \sum_{i \neq j} \beta_i \geq \beta_j$. That is, it satisfies the constraint (8c), meaning that it is a feasible solution. We will now show that it is also an optimal solution to the problem.

Any other feasible probability distribution $(p'_1, \cdots, p'_N)$ can be obtained as $p'_i = p_i + e_i$, where $e_i \geq 0$ for $i \neq j$, and $e_j = -\sum_{i \neq j} e_i \leq \beta_j - p_j$, due to the constraints $p'_i \geq \beta_i$, and $\sum_i p'_i = 1 = \sum_i p_i$. However, the cost associated with the distribution $(p'_1, \cdots, p'_N)$ is

$$\sum_i p'_i c_i = \sum_i p_i c_i + e_j c_j + \sum_{i \neq j} e_i c_i \geq \sum_i p_i c_i + e_j c_j - e_j c_{\min} = \sum_i p_i c_i,$$

since $c_i \geq c_{\min}$, and $c_j = c_{\min}$. That is, the cost is not lower than the one associated with the distribution $(p_1, \ldots, p_N)$. This shows that the probability distribution given by (9) is an optimal solution.

B.9. Remark 1

It is straightforward to show that all main lemmas and proof steps would remain the same for the mini-batch SVRG-AS algorithm.

B.10. Smoothness and Strong Convexity Parameters For Logistic Regression

Consider function

$$f_i(\boldsymbol{w}) = \frac{1}{M_i} \sum_{j \in [M_i]} \ln\left(1 + e^{-\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_{ij} y_{ij}}\right) .$$

Define $\boldsymbol{z}_{ij} := \boldsymbol{x}_{ij} y_{ij}$. Let $\mathrm{eig}_{\max}(\boldsymbol{X})$ and $\mathrm{eig}_{\min}(\boldsymbol{X})$ denote the largest and smallest eigenvalues of matrix $\boldsymbol{X}$. Now, we characterize the geometry of our problem using $L_i \geq \mathrm{eig}_{\max}\left(\nabla^2 f_i(\boldsymbol{w})\right)$ and $\mu \leq \mathrm{eig}_{\min}\left(\nabla^2 f(\boldsymbol{w})\right)$ inequalities. We have,

- Smoothness parameter L_i:

$$\begin{aligned} \mathrm{eig}_{\max}\left(\nabla^2 f_i(\boldsymbol{w})\right) &\leq \frac{1}{M_i} \sum_{j \in [M_i]} \mathrm{eig}_{\max}\left(\nabla^2 f_i(\boldsymbol{w})\right) \\ &\leq \frac{1}{4M_i} \sum_{j \in [M_i]} \|\boldsymbol{z}_{ij}\|_2^2 := L_i \, . \end{aligned}$$

- Strong convexity parameter μ:

$$\mathrm{eig}_{\min}\left(\nabla^2 f(\boldsymbol{w})\right) \geq \frac{1}{M_i} \sum_{j \in [M_i]} \mathrm{eig}_{\min}\left(\nabla^2 f_i(\boldsymbol{w})\right) := \mu \, .$$

B.11. Mini-batch SVRG-AS

Algorithm 2 shows SVRG-AS with mini-batch updates.

Algorithm 2 Mini-batch SVRG-AS

Inputs: Maximum epoch length T, number of epochs K, N, step size sequence $(\alpha_k)_k$, and probabilities $p_1, \ldots, p_N$.
for $k = 1, 2, \ldots, K-1$ **do**
  $\widetilde{\boldsymbol{h}}_k \leftarrow \sum_{i\in[N]} p_i \boldsymbol{h}_i\left(\widetilde{\boldsymbol{w}}_k\right)$
  $\boldsymbol{w}_{k,0} \leftarrow \widetilde{\boldsymbol{w}}_k$
  Sample $\zeta := \zeta_k$ uniformly from $\{1, 2, \ldots, T\}$
  for $t = 1, 2, \ldots, \zeta$ **do**
    Every worker i with probability p_i, independent of other workers, computes $\boldsymbol{g}_i\left(\boldsymbol{w}\right)$ and $\boldsymbol{h}_i\left(\boldsymbol{w}\right) := \boldsymbol{g}_i\left(\boldsymbol{w}\right)/Np_i$, and sends it to the master node, for both $\boldsymbol{w} = \boldsymbol{w}_{k,t-1}$ and $\boldsymbol{w} = \widetilde{\boldsymbol{w}}_k$.
    Define $\xi := \{i \mid \boldsymbol{h}_i(\boldsymbol{w}_{k,t-1}) \text{ is sampled}\}$.
    Compute $\boldsymbol{w}_{k,t} \leftarrow \boldsymbol{w}_{k,t-1} - \alpha_k \sum_{i=1}^{N} \mathbb{1}_{i\in\xi}\left(\boldsymbol{h}_i\left(\boldsymbol{w}_{k,t-1}\right) - \boldsymbol{h}_i\left(\widetilde{\boldsymbol{w}}_k\right) + \widetilde{\boldsymbol{h}}_k\right)$
    Broadcast $\boldsymbol{w}_{k,t}$
  end for
  $\widetilde{\boldsymbol{w}}_{k+1} \leftarrow \boldsymbol{w}_{k,\zeta}$
end for
Return: $\widetilde{\boldsymbol{w}}_K$