Fifty-fourth Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
September 27 - 30, 2016

Data-Dependent Bounds on Network Gradient Descent

Avleen Bijral[1], Anand D. Sarwate[2], and Nathan Srebro[3]

***Abstract*— We study a consensus-based distributed stochastic gradient method for distributed optimization in a setting common for machine learning applications. Nodes in the network hold disjoint data and seek to optimize a common objective which decomposes into a sum of convex functions of individual data points. We show that the rate of convergence for this method involves the spectral properties of two matrices: the standard spectral gap of a weight matrix from the network topology and a new term depending on the spectral norm of the sample covariance matrix of the data. This result shows the benefit of datasets with small spectral norm. Extensions of the method can identify the impact of limited communication, increasing the number of nodes, and scaling with data set size.**

I. Introduction

This paper studies the problem of minimizing a regularized convex function [1] of the form

$$J(\mathbf{w}) = \sum_{i=1}^{N} \frac{\ell(\mathbf{w}^\top \mathbf{x}_i; y_i)}{N} + \frac{\mu}{2}\|\mathbf{w}\|^2 \qquad (1)$$
$$= \mathbb{E}_{\mathbf{x}\sim\hat{\mathcal{P}}}\left[\ell(\mathbf{w}^\top \mathbf{x}; y)\right] + \frac{\mu}{2}\|\mathbf{w}\|^2,$$

where $\ell(\cdot)$ is convex and Lipschitz and the expectation is with respect to the empirical distribution $\hat{\mathcal{P}}$ corresponding to a given data set with N total data points $\{(\mathbf{x}_i, y_i)\}$. We will assume $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. This *regularized empirical risk minimization* formulation encompasses algorithms such as support vector machine classification, ridge regression, logistic regression, and others [2]. For example $\mathbf{x}$ could represent d pixels in a grayscale image and y a binary label indicating whether the image is of a face: $\mathbf{w}^\top\mathbf{x}$ gives a confidence value about whether the image is of a face or not.

We would like to solve such problems using a network of m processors connected via a network (represented by a graph indicating which nodes can communicate with each other). The system would distribute these N points across the m nodes, inducing local objective functions $J_j(\mathbf{w})$ approximating (1). Decentralized optimization algorithms for statistical computation and machine learning on large data sets try to trade off efficiency (in terms of estimation error) and speed (from parallelization).

This work was supported by the National Science Foundation under award CCF-1440033.

[1]Avleen Bijral is with Microsoft, Redmond, WA 98052, USA. avbijral@microsoft.com

[2]Anand D. Sarwate is with the Department of Electrical and Computer Engineering, Rutgers, The State University of New Jersey, 94 Brett Road, Piscataway, NJ 08854, USA. anand.sarwate@rutgers.edu

[3]Nathan Srebro is with the Toyota Technological Institute at Chicago, 6045 S. Kenwood Ave., Chicago, IL 60637, USA. nati@ttic.edu

The theoretical analysis of distributed optimization methods has focused on providing strong *data-independent* convergence rates under analytic assumptions on the objective function such as convexity and smoothness. We analyze a distributed primal averaging [3] algorithm in which nodes process points sequentially, performing a stochastic gradient descent (SGD) update locally and average the iterates of their neighbors after each gradient step. In this paper we analyze the convergence rate to identify factors corresponding to the underlying data distribution. This work is in the spirit of previous work [4] which studying mini-batching. Here we see that the spectral norm $\rho^2 = \sigma_1(\mathbb{E}_{\hat{\mathcal{P}}}[\mathbf{x}\mathbf{x}^\top])$ affects the rate of convergence: we prove an upper bound on the suboptimality gap for distributed primal averaging that depends on ρ^2. Our result suggests that networks of size $m < \frac{1}{\rho^2}$ gain from parallelization.

Understanding how the data distribution affects the asymptotics of distributed optimization methods is challenging because the statistical assumption on the nodes' data means that in homogenous networks, each node can reach the global optimum by processing its own data [5]. In this sense the network "hurts" the optimization process by distributing the data so that each node converges slower. This is in contrast to mini-batching, in which parallelizing gradient computations leads to speed ups. Empirically, we see that factors such as the sparsity may impact the degree to which distributed processing can help. Our upper bound corroborates that empirical evidence, but without a matching lower bounds we cannot say that ρ^2 determines the rate of convergence.

Related Work. Several authors have proposed distributed algorithms involving nodes computing local gradient steps and averaging iterates, gradients, or other functions of their neighbors [3], [6], [7]. By alternating local updates and consensus with neighbors, estimates at the nodes converge to the optimizer of $J(\cdot)$. In these works no assumption is made on the local objective functions and they can be arbitrary. Consequently the convergence guarantees do not reflect the setting when the data is homogenous (for e.g. when data has the same distribution); in particular, the error increases as we add more machines. This is counterintuitive, especially in the large scale regime, since this suggests that despite homogeneity the methods perform worse than the centralized setting (all data on one node).

We provide a first data-dependent analysis of a consensus based stochastic gradient method in the homogenous setting and demonstrate that there exist regimes where we benefit from having more machines in any network.

In contrast to our stochastic gradient based results, data dependence via the Hessian of the objective has also been

demonstrated in parallel coordinate descent based approaches of Liu et al. [8] and the Shotgun algorithm of Bradley et al. [9]. The assumptions differ from us in that the objective function is assumed to be smooth [8] or $\mathcal{L}_1$ regularized [9]. Most importantly, our results hold for arbitrary networks of compute nodes, while the coordinate descent based results hold only for networks where all nodes communicate with a central aggregator (sometimes referred to as a master-slave architecture, or a star network), which can be used to model shared-memory systems. Another interesting line of work is the impact of delay on convergence in distributed optimization [10]. These results show that delays in the gradient computation for a star network are asymptotically negligible when optimizing smooth loss functions. We study general network topologies but with intermittent, rather than delayed communication. Our result suggest that certain datasets are more tolerant of skipped communication rounds, based on the spectral norm of their covariance.

We take an approach similar to that of Takáč et al. [4] who developed a spectral-norm based analysis of mini-batching for non-smooth functions. We decompose the iterate in terms of the data points encountered in the sample path [11]. This differs from analysis based on smoothness considerations alone [10]–[13] and gives practical insight into how communication (full or intermittent) impacts the performance of these algorithms. Note that our work is fundamentally different in that these other works either assume a centralized setting [11]–[13] or implicitly assume a specific network topology (e.g. [14] uses a star topology). For the main results we only assume strong convexity while the existing guarantees for the cited methods depend on a variety of regularity and smoothness conditions.

Limitation. In the stochastic convex optimization (see for e.g. [15]) setting the quantity of interest is the population objective corresponding to problem 1. When minimizing this population objective our results suggest that adding more machines worsens convergence (See Theorem 1). For finite data our convergence results satisfy the intuition that adding more nodes in an arbitrary network will hurt convergence. The finite homogenous setting is most relevant in settings such as data centers, where the processors hold data which essentially looks the same. In the infinite or large scale data setting, common in machine learning applications, this is counterintuitive since when each node has infinite data, any distributed scheme including one on arbitrary networks shouldn't perform worse than the centralized scheme (all data on one node). Thus our analysis is limited in that it doesn't unify the stochastic optimization and the consensus setting in a completely satisfactory manner.

In this paper we focus on a simple and well-studied protocol [3]. However, our analysis approach and insights may yield data-dependent bounds for other more complex algorithms such as distributed dual averaging [6]. More sophisticated gradient averaging schemes such as that of Mokhtari and Ribeiro [16] can exploit dependence across iterations [17], [18] to improve the convergence rate; analyzing the impact of the data distribution is considerably more complex in these algorithms.

We believe that our results provide a first step towards understanding data-dependent bounds for distributed stochastic optimization in settings common to machine learning. Our analysis coincides with phenomenon seen in practice: for data sets with small ρ, distributing the computation across many machines is beneficial, but for data with larger ρ more machines is not necessarily better. Our work suggests that taking into account the data dependence can improve the empirical performance of these methods.

In this paper we describe our basic result for synchronous networked gradient descent. Full details and proofs are deferred to the full version of our manuscript [19].

II. Model

We will use boldface for vectors. Let $[k] = \{1, 2, \ldots, k\}$. Unless otherwise specified, the norm $\|\cdot\|$ is the standard Euclidean norm. The spectral norm of a matrix A is defined to be the largest singular value $\sigma_1(A)$ of the matrix A or equivalently the square root of the largest eigenvalue of $A^\top A$. For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$, we will denote the neighbors of a vertex $i \in \mathcal{V}$ by $\mathcal{N}(i) \subseteq \mathcal{V}$.

Data model. Let $\mathcal{P}$ be a distribution on $\mathbb{R}^{d+1}$ such that for $(\mathbf{x}, y) \sim \mathcal{P}$, we have $\|\mathbf{x}\| \leq 1$ almost surely. Let $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ be i.i.d sample of d-dimensional vectors from $\mathcal{P}$ and let $\hat{\mathcal{P}}$ be the empirical distribution of S. Let $\hat{\mathbf{\Sigma}} = \mathbb{E}_{\mathbf{x} \sim \hat{\mathcal{P}}}[\mathbf{x}\mathbf{x}^\top]$ be the sample second-moment matrix of S. Our goal is to express the performance of our algorithms in terms of $\rho^2 = \sigma_1(\hat{\mathbf{\Sigma}})$, the spectral norm of $\hat{\mathbf{\Sigma}}$. The spectral norm ρ^2 can vary significantly across different data sets. For example, for sparse data sets ρ^2 is often small. This can also happen if the data happens to lie in low-dimensional subspace (smaller than the ambient dimension d).

Problem. Our problem is to minimize a particular instance of (1) where the expectation is over a finite collection of data points:

$$\mathbf{w}^* \stackrel{\text{def}}{=} \operatorname*{argmin}_{\mathbf{w}} J(\mathbf{w}) \tag{2}$$

Let $\hat{\mathbf{w}}_j(t)$ be the estimate of $\mathbf{w}^*$ at node $j \in [m]$ in the t-th iteration. We bound the expected gap (over the data distribution) at iteration T between $J(\mathbf{w}^*)$ and the value $J(\hat{\mathbf{w}}_i(T))$ of the global objective $J(\hat{\mathbf{w}}_j(T))$ at the output $\hat{\mathbf{w}}_j(T)$ of each node j in our distributed network. We will denote the subgradient set of $J(\mathbf{w})$ by $\partial J(\mathbf{w})$ and a subgradient of $J(\mathbf{w})$ by $\nabla J(\mathbf{w}) \in \partial J(\mathbf{w})$.

In our analysis we will make the following assumptions about the individual functions $\ell(\mathbf{w}^\top \mathbf{x})$: (a) The loss functions $\{\ell(\cdot)\}$ are convex, and (b) The loss functions $\{\ell(\cdot; y)\}$ are L-Lipschitz for some $L > 0$ and all y. Note that $J(\mathbf{w})$ is μ-strongly convex due to the ℓ_2-regularization. Our analysis will not depend on the the response y except through the Lipschitz bound L so we will omit the explicit dependence on y to simplify the notation in the future.

Network Model. We consider a model in which minimization in (2) must be carried out by m nodes. These nodes

are arranged in a network whose topology is given by a graph $\mathcal{G}$ – an edge (i,j) in the graph means nodes i and j can communicate. A matrix $\mathbf{P}$ is called graph-conformant if $P_{ij} > 0$ only if the edge (i,j) is in the graph. We will consider algorithms which use a doubly stochastic and graph-conformant sequence of matrices $\mathbf{P}(t)$.

Sampling Model. We assume the N data points are divided evenly and uniformly at random among the m nodes, and define $n \stackrel{\text{def}}{=} N/m$ to be the number of points at each node. This is a necessary assumption since our bounds are data dependent and depend on subsampling bounds of spectral norm of certain random submatrices. However our data independent bound holds for arbitrary splits. Let S_i be the subset of n points at node i. The local stochastic gradient procedure consists of each node $i \in [m]$ sampling from S_i with replacement. This is an approximation to the local objective function

$$J_i(\mathbf{w}) = \sum_{j \in S_i} \frac{\ell(\mathbf{w}^T \mathbf{x}_{i,j})}{n} + \frac{\mu}{2} \|\mathbf{w}\|^2. \tag{3}$$

Algorithm. In the subsequent sections we analyze the distributed version (Algorithm 1) of standard SGD. This algorithm is not new [3], [7] and has been analyzed extensively in the literature. The step-size $\eta_t = 1/(\mu t)$ is commonly used for large scale strongly convex machine learning problems like SVMs (e.g.- [20]) and ridge regression: to avoid an extra parameter in the bounds, we take this setting. In Algorithm 1 node i samples a point uniformly with replacement from a local pool of n points and then updates its iterate by computing a weighted sum with its neighbors followed by a local subgradient step. The selection is uniform to guarantee that the subgradient is an unbiased estimate of a true subgradient of the local objective $J_i(\mathbf{w})$, and greatly simplifies the analysis. Different choices of $\mathbf{P}(t)$ will allow us to understand the effect of limiting communication in this distributed optimization algorithm.

Algorithm 1 Consensus Strongly Convex Optimization

Input: $\{\mathbf{x}_{i,j}\}$,where $i \in [m]$ and $j \in [n]$ and $N = mn$, matrix sequence $\mathbf{P}(t)$, $\mu > 0$, $T \geq 1$

{Each $i \in [m]$ executes}
Initialize: set $\mathbf{w}_i(1) = \mathbf{0} \in \mathbb{R}^d$.
for $t = 1$ **to** T **do**
  Sample $\mathbf{x}_{i,t}$ uniformly with replacement from S_i.
  Compute $\mathbf{g}_i(t) \in \partial \ell(\mathbf{w}_i(t)^T \mathbf{x}_{i,t}) \mathbf{x}_{i,t} + \mu \mathbf{w}_i(t)$
  $\mathbf{w}_i(t+1) = \sum_{j=1}^m \mathbf{w}_j(t) P_{ij}(t) - \eta_t \mathbf{g}_i(t)$
end for
Output: $\hat{\mathbf{w}}_i(T) = \frac{1}{T} \sum_{t=1}^T \mathbf{w}_i(t)$ for any $i \in [m]$.

Expectations and probabilities. There are two sources of stochasticity in our model: the first in the split of data points to the individual nodes, and the second in sampling the points during the gradient descent procedure. We assume that the split is done uniformly at random, which implies that the expected covariance matrix at each node is the same as the population covariance matrix $\hat{\mathbf{\Sigma}}$. Conditioned on the split, we assume that the sampling at each node is uniformly at random from the data point at that node, which makes the stochastic subgradient an unbiased estimate of the subgradient of the local objective function. Let $\mathcal{F}_t$ be the sigma algebra generated by the random point selections of the algorithm up to time t, so that the iterates $\{\mathbf{w}_i(t) : i \in [m]\}$ are measurable with respect to $\mathcal{F}_t$.

III. Main result

Methods like Algorithm 1, also referred to as primal averaging, have been analyzed previously [3], [7], [21]. In these works it is shown that the convergence properties depend on the structure of the underlying network via the second largest eigenvalue of $\mathbf{P}$. We consider in this section the case when $\mathbf{P}(t) = \mathbf{P}$ for all t where $\mathbf{P}$ is a fixed Markov matrix. This corresponds to a synchronous setting where communication occurs at every iteration.

We analyze the use of the step-size $\eta_t = 1/(\mu t)$ in Algorithm 1 and show that the convergence depends on the spectral norm $\rho^2 = \sigma_1(\hat{\mathbf{\Sigma}})$ of the sample covariance matrix.

Theorem 1: Fix a Markov matrix $\mathbf{P}$ and let $\rho^2 = \sigma_1(\hat{\mathbf{\Sigma}})$ denote the spectral norm of the covariance matrix of the data distribution. Consider Algorithm 1 when the objective $J(\mathbf{w})$ is strongly convex, $\mathbf{P}(t) = \mathbf{P}$ for all t, and $\eta_t = 1/(\mu t)$. Let $\lambda_2(\mathbf{P})$ denote the second largest eigenvalue of $\mathbf{P}$. Then if the number of samples on each machine n satisfies

$$n > \frac{4}{3\rho^2} \log(d) \tag{4}$$

and the number of iterations T satisfies

$$T > 2e \log(1/\sqrt{\lambda_2(\mathbf{P})}) \tag{5}$$

$$\frac{T}{\log(T)} > \max\left(\frac{4}{3\rho^2} \log(d), \frac{\left(\frac{8}{5}\right)^{\frac{1}{4}} \sqrt{m/\rho}}{\log(1/\lambda_2(\mathbf{P}))} \right), \tag{6}$$

then the expected error for each node i satisfies

$$\mathbb{E}\left[J(\hat{\mathbf{w}}_i(T)) - J(\mathbf{w}^*)\right] \leq \left(\frac{1}{m} + \frac{100\sqrt{m\rho^2} \cdot \log T}{1 - \sqrt{\lambda_2(\mathbf{P})}} \right) \cdot \frac{L^2}{\mu} \cdot \frac{\log T}{T}. \tag{7}$$

Remark 1: Theorem 1 indicates that the number of machines should be chosen as a function of ρ. We can identify three sub-cases of interest:

Case (a): $m \leq \frac{1}{\rho^{2/3}}$: In this regime since $1/m > \sqrt{m\rho^2}$ (ignoring the constants and the $\log T$ term) we always benefit from adding more machines.

Case (b): $\frac{1}{\rho^{2/3}} < m \leq \frac{1}{\rho^2}$: The result tells us that there is no degradation in the error and the bound improves by a factor $\sqrt{m}\rho$. Sparse data sets generally have a smaller value of ρ^2 (as seen in Takáč et al. [4]); Theorem 1 suggests that for such data sets we can use a larger number of machines without losing performance. However the requirements on the number of iterations also increases. This provides additional perspective on the observation by Takáč et al [4] that sparse datasets are more amenable to parallelization via

TABLE I

DATA SETS AND PARAMETERS FOR EXPERIMENTS

data set	training	test	dim.	λ	ρ^2
RCV1	781,265	23,149	47,236	10^{-4}	0.01
Covertype	522,911	58,001	47,236	10^{-6}	0.21

mini-batching. The same holds for our type of parallelization as well.

Case (c): $m > \frac{1}{\rho^2}$: In this case we pay a penalty $\sqrt{m\rho^2} \geq 1$ suggesting that for datasets with large ρ we should expect to lose performance even with relatively fewer machines.

Note that $m > 1$ is implicit in the condition $T > 2e\log(1/\sqrt{\lambda_2}))$ since $\lambda_2 = 0$ for $m = 1$. This excludes the single node Pegasos [4] case. Additionally in the case of general strongly convex losses (not necessarily dependent on $\mathbf{w}^\top\mathbf{x}$) we can obtain a convergence rate of $\mathcal{O}(\log^2(T)/T)$. We do not provide the proof here.

IV. EXPERIMENTS

Our goals in our experimental evaluation are to validate the theoretical dependence of the convergence rate on ρ^2 and to see if the conclusions hold when the assumptions we make in the analysis are violated. Note that all our experiments are based on simulations on a multicore computer.

A. Data sets, tasks, and parameter settings

The data sets used in our experiments are summarized in Table (IV-A). `Covertype` is the forest covertype dataset [22] used in [20] obtained from the UC Irvine Machine Learning Repository [23], and rcv1 is from the Reuters collection [23] obtained from libsvm collection [24]. The `RCV1` data set has a small value of ρ^2, whereas `Covertype` has a larger value. In all the experiments we looked at ℓ_2-regularized classification objectives for problem (1). Each plot is averaged over 5 runs.

The data consists of pairs $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$. In all experiments we optimize the ℓ_2-regularized empirical hinge loss where $\ell(\mathbf{w}^\top\mathbf{x}) = (1 - \mathbf{w}^\top\mathbf{x}y)_+$. The values of the regularization parameter μ are chosen from to be the same as those in Shalev-Shwarz et al. [20].

We simulated networks of compute nodes of varying size (m) arranged in a k-regular graph with $k = \lfloor 0.25m \rfloor$ or a fixed degree (not dependent on m). Note that the dependence of the convergence rate of procedures like Algorithm (1) on the properties of the underlying network has been investigated before and we refer the reader to Agarwal and Duchi [10] for more details. In this paper we experiment only with k-regular graphs. The weights on the Markov matrix $\mathbf{P}$ are set by using the max-degree Markov chain (see [25]). One can also optimize for the fastest mixing Markov chain ([25], [26]). Each node is randomly assigned $n = \lfloor N/m \rfloor$ points.

B. Intermittent Communication

In this experiment we show the objective function for `RCV1` and `Covertype` as we change the frequency of communication (Figure 1), communicating after every $1, 10, 50$ and 500 iterations. Indeed as predicted we see that the dataset with the larger ρ^2 appears to be affected more by intermittent communication. This indicates that network bandwidth can be conserved for datasets with a smaller ρ^2.

C. Comparison of Different Schemes

We compare the three different schemes proposed in this paper. On a network of $m = 64$ machines we plot the performance of the mini batch extension of Algorithm (1) with batch size 128 against the intermittent scheme that communicates after every 128 iterations and also the standard version of the algorithm. In Figure 3-(a) we see that mini-batching does better than the vanilla and the intermittent scheme.

D. Infinite Data

We generated a very large ($N = 10^7$) synthetic dataset from a multivariate normal distribution and created a simple binary classification task using a random hyperplane. As we can see in Figure 2 for the SVM problem and a k-regular network we continue to gain as we add more machines and then eventually we stabilize but never lose from more machines. We only show the first few thousand iterations for clarity.

E. Diminishing Communication

To test if our conclusions apply when the i.i.d assumption for the matrices $\mathbf{P}(t)$ does not hold we simulate a diminishing communication regime. Such a scheme can be useful when the nodes are already close to the optimal solution and communicating their respective iterate is wasteful. Intuitively it is in the beginning the nodes should communicate more frequently. To formalize the intuition we propose the following communication model

$$\mathbf{P}(t) = \begin{cases} \mathbf{P} & \text{w.p. } Ct^{-p} \\ \mathbf{I} & \text{w.p. } 1 - Ct^{-p} \end{cases} \tag{8}$$

where $C, p > 0$. Thus the sequence of matrices are not identically distributed and the conclusions of Theorem (**??**) do not apply.

However in Figure 3-(b) ($C = 1, p = 0.5$) we see that on a network of $m = 128$ nodes the performance for the diminishing regime is similar to the full communication case and we can hypothesize that our results also hold for non i.i.d communication matrices.

V. DISCUSSION

In this paper we described a consensus stochastic gradient descent algorithm and analyzed its performance in terms of the spectral norm ρ^2 of the data covariance matrix under a homogenous assumption. The extended version of this paper contains additional results on intermittent communication and the asymptotic performance of the method.

- We can extend the conclusions of Theorem 1 to the case of stochastic communication schemes, thus allowing for the data dependent interpretations of convergence in a

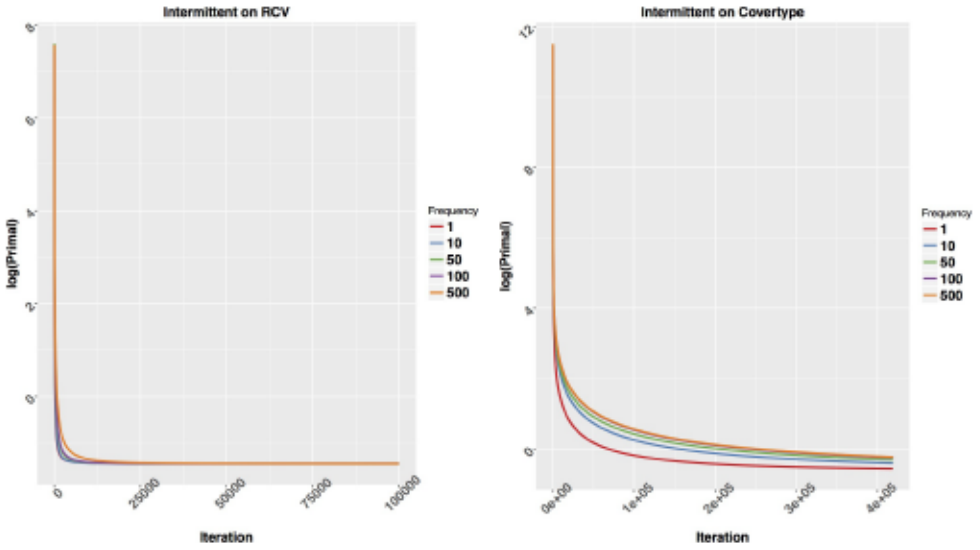

Fig. 1. Performance of Algorithm (1) with intermittent communication scheme on datasets with very different ρ^2. The algorithm works better for smaller ρ^2 and there is less decay in performance for `RCV1` as we decrease the number of communication rounds as opposed to `Covertype` ($\rho^2 = 0.01$ vs $\rho^2 = 0.21$).

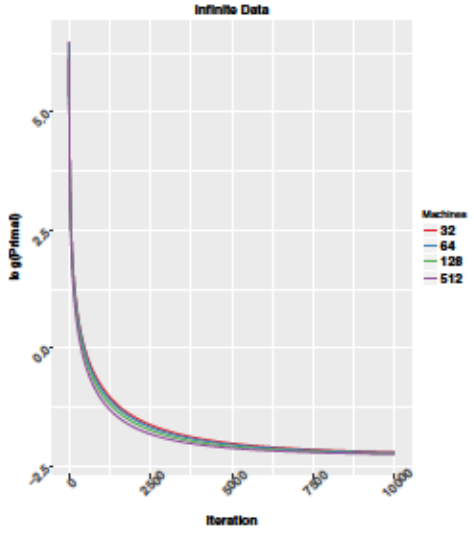

Fig. 2. No network effect with increasing benefit of adding more machines in the case of infinite data.

more general setting. For example, we could analyze the effect of reducing the communication overhead of Algorithm 1. This reduction can improve the overall running time ("wall time") of the algorithm because communication latency can hinder the convergence of many algorithms in practice [27]. A natural way of limiting communication is to communicate only a fraction ν of the T total iterations; at other times nodes simply perform local gradient steps. A deterministic schedule of communication every $1/\nu$ iterations would allow local nodes to perform minibatch steps:

$$\mathbf{w}_i(t+1) = \sum_{j \in \mathcal{N}_i} \mathbf{w}_j(t) P_{ij}(t) - \eta_t \nu \sum_{i \in \mathcal{I}_i} \mathbf{g}_i(t). \quad (9)$$

- We also examined the sub-optimality of distributed primal averaging when $T \to \infty$ for the case of smooth strongly convex objectives. Our results suggest that we never gain from adding more machines in any network. An asymptotic analysis of Algorithm 1 shows that the network effect disappears and we do indeed gain from more machines in any network. The result follows from the asymptotic normality analysis of Bianchi et al. [28, Theorem 5]. The main differences between Algorithm 1 and the consensus algorithm of Bianchi et al. is that we average the iterates before making the local update.

References

[1] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization," ArXiV, extended version of ICML paper arXiv:1109.5647 [cs.LG], 2012. [Online]. Available: http://arxiv.org/abs/1109.5647

[2] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, UK: Cambridge, 2014.

[3] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, January 2009. [Online]. Available: http://dx.doi.org/10.1109/TAC.2008.2009515

[4] M. Takáč, A. Bijral, P. Richtárik, and N. Srebro, "Mini-batch primal and dual methods for SVMs," in *Proceedings of the 30th International Conference on Machine Learning (ICML)*, ser. JMLR Workshop and Conference Proceedings, S. Dasgupta and D. McAllester, Eds., vol. 28, 2013, pp. 1022–1030. [Online]. Available: http://jmlr.org/proceedings/papers/v28/takac13.html

[5] O. Shamir and N. Srebro, "Distributed stochastic optimization and learning," in *52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Monticello, IL, USA, September 2014, pp. 850–857.

[6] J. Duchi, A. Agarwal, and M. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, March 2011. [Online]. Available: http://dx.doi.org/10.1109/TAC.2011.2161027

[7] S. S. Ram, A. Nedic, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *Journal of Optimization Theory and Applications*, vol. 147, no. 3, pp. 516–545, December 2011. [Online]. Available: http://dx.doi.org/10.1007/s10957-010-9737-7

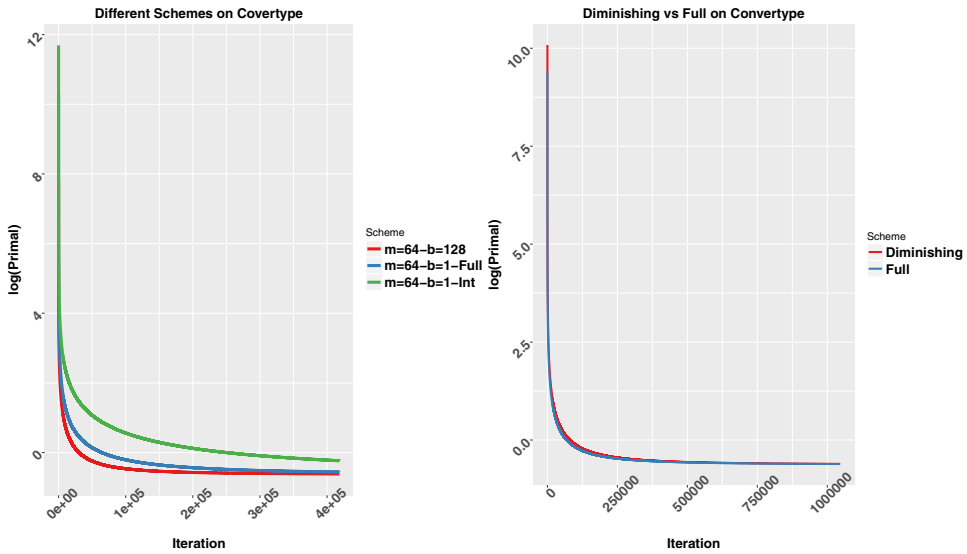

Fig. 3. a) Comparison of three different schemes a) Algorithm (1) with Mini-Batching b) Standard c) Intermittent with $b = (1/\nu) = 128$. As predicted the mini-batch scheme performs much better than the others. b) The performance on Covertype with a full and a diminishing communication scheme is similar.

[8] J. Liu, S. J. Wright, C. Re, V. Bittorf, and S. Sridhar, "An asynchronous parallel stochastic coordinate descent algorithm," in *Proceedings of the 31st International Conference on Machine Learning*, ser. JMLR Workshop and Conference Proceedings, L. Getoor and T. Scheffer, Eds., vol. 32, 2014. [Online]. Available: http://jmlr.org/proceedings/papers/v32/liud14.pdf

[9] J. K. Bradley, A. Kyrola, D. Bickson, and C. Guestrin, "Parallel coordinate descent for l_1-regularized loss minimization," in *Proceedings of the 28th International Conference on Machine Learning*, ser. JMLR Workshop and Conference Proceedings, L. Getoor and T. Scheffer, Eds., vol. 28, 2011. [Online]. Available: http://www.select.cs.cmu.edu/publications/paperdir/icml2011-bradley-kyrola-bickson-guestrin.pdf

[10] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., 2011, pp. 873–881. [Online]. Available: http://books.nips.cc/papers/files/nips24/NIPS2011_0574.pdf

[11] A. Cotter, O. Shamir, N. Srebro, and K. Sridharan, "Better mini-batch algorithms via accelerated gradient methods," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., 2011, pp. 1647–1655. [Online]. Available: http://papers.nips.cc/paper/4432-better-mini-batch-algorithms-via-accelerated-gradient-methods

[12] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *Journal of Machine Learning Research*, vol. 13, pp. 165–202, January 2012. [Online]. Available: http://jmlr.org/papers/v13/dekel12a.html

[13] 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*, ser. JMLR Workshop and Conference Proceedings, E. P. Xing and T. Jebara, Eds., vol. 32, 2014, pp. 1000–1008. [Online]. Available: http://jmlr.org/proceedings/papers/v32/shamir14.html

[14] Y. Zhang, J. Duchi, and M. Wainwright, "Communication-efficient algorithms for statistical optimization," in *Advances in Neural Information Processing Systems 25*, P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., 2012, pp. 1511–1519. [Online]. Available: http://books.nips.cc/papers/files/nips25/NIPS2012_0716.pdf

[15] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan, "Stochastic convex optimization," in *Proceedings Conference on Learning Theory*, P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., 2009.

[16] A. Mokhtari and A. Ribeiro, "DSA: decentralized double stochastic averaging gradient algorithm," *Journal of Machine Learning Research*, vol. 17, no. 61, pp. 1–35, 2016. [Online]. Available: http://jmlr.org/papers/v17/15-292.html

[17] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[18] M. Schmidt, N. L. Roux, and F. Bach, "Minimizing finite sums with the stochastic average gradient," HAL, Tech. Rep. hal-00860051, January 2015. [Online]. Available: https://hal.inria.fr/hal-00860051v2

[19] A. Bijral, A. Sarwate, and N. Srebro, "On data dependence in distributed stochastic optimization," ArXiV, Tech. Rep. arXiv:1603.04379 [math.OC], September 2016. [Online]. Available: http://arxiv.org/abs/1603.04379

[20] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal Estimated sub-GrAdient SOlver for SVM," *Mathematical Programming, Series B*, vol. 127, no. 1, pp. 3–30, October 2011. [Online]. Available: http://dx.doi.org/10.1007/s10107-010-0420-4

[21] K. I. Tsianos and M. G. Rabbat, "Distributed strongly convex optimization," ArXiV, Tech. Rep. arXiv:1207.3031 [cs.DC], July 2012. [Online]. Available: http://arxiv.org/abs/1207.3031

[22] J. A. Blackard and D. J. Dean, "Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables," *Computers and Electronics in Agriculture*, vol. 24, no. 3, pp. 131–151, December 1999. [Online]. Available: http://dx.doi.org/10.1016/S0168-1699(99)00046-0

[23] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[24] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27:27, May 2011. [Online]. Available: http://dx.doi.org/10.1145/1961189.1961199

[25] S. Boyd, L. Xiao, and P. Diaconis, "Fastest mixing markov chain on a graph," *SIAM Review*, vol. 46, no. 4, pp. 667–689, 2004. [Online]. Available: http://dx.doi.org/10.1137/S0036144503423264

[26] A. S. Bijral and N. Srebro, "On doubly stochastic graph optimization," in *NIPS Workshop on Analyzing Networks and Learning with Graphs*, 2009.

[27] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Communication/computation tradeoffs in consensus-based distributed optimization," in *Advances in Neural Information Processing Systems 25*, P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., 2012, pp. 1952–1960.

[28] P. Bianchi, G. Fort, and W. Hachem, "Performance of a distributed stochastic approximation algorithm," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7405–7418, 2013. [Online]. Available: http://dx.doi.org/10.1109/TIT.2013.2275131