DiSCO: Distributed Optimization for Self-Concordant Empirical Loss

Yuchen Zhang YUCZHANG@EECS.BERKELEY.EDU
University of California Berkeley, Berkeley, CA 94720, USA

Lin Xiao LIN.XIAO@MICROSOFT.COM
Microsoft Research, Redmond, WA 98053, USA

Abstract

We propose a new distributed algorithm for empirical risk minimization in machine learning. The algorithm is based on an inexact damped Newton method, where the inexact Newton steps are computed by a distributed preconditioned conjugate gradient method. We analyze its iteration complexity and communication efficiency for minimizing self-concordant empirical loss functions, and discuss the results for distributed ridge regression, logistic regression and binary classification with a smoothed hinge loss. In a standard setting for supervised learning, where the n data points are i.i.d. sampled and when the regularization parameter scales as $1/\sqrt{n}$, we show that the proposed algorithm is communication efficient: the required round of communication does not increase with the sample size n, and only grows slowly with the number of machines.

1. Introduction

Many optimization problems in machine learning are formulated with a large amount of data as input. With the amount of data we collect and process growing at a fast pace, it happens more often that the dataset involved in an optimization problem cannot fit into the memory or storage of a single computer (machine). To solve such "big data" optimization problems, we need to use distributed algorithms that rely on inter-machine communication.

In this paper, we focus on distributed algorithms for regularized *empirical risk minimization* (ERM). Suppose that our distributed computing system consists of m machines, and each machine has access to n samples $z_{i,1}, \dots, z_{i,n}$, for $i = 1, \dots, m$. Then each machine can evaluate a local empirical loss function

$$f_i(w) \overset{\text{def}}{=} \frac{1}{n}\sum_{j=1}^{n} \phi(w, z_{i,j}) + \frac{\lambda}{2}\|w\|_2^2, \qquad (1)$$

where $z_{i,j}$ are random vectors whose probability distribution is supported on a set $\mathcal{Z} \subset \mathbb{R}^p$, and the cost function $\phi : \mathbb{R}^d \times \mathcal{Z} \to \mathbb{R}$ is convex in w for every $z \in \mathcal{Z}$. Here the quadratic term $\frac{\lambda}{2}\|w\|_2^2$ is a regularizer, which helps to prevent overfitting. Our goal is to minimize the overall empirical loss defined with all mn samples:

$$f(w) \overset{\text{def}}{=} \frac{1}{m}\sum_{i=1}^{m} f_i(w). \qquad (2)$$

Since the functions $f_i(w)$ can be accessed only locally, we consider distributed algorithms that alternate between a local computation procedure at each machine, and a round of inter-machine communication. Compared with local computation at each machine, the cost of inter-machine communication is much higher in terms of both speed/delay and energy consumption (e.g., Bekkerman et al., 2011; Shalf et al., 2011), thus it is often considered as the bottleneck for distributed computing. Our goal is to develop *communication-efficient* distributed algorithms, which try to use a minimal number of communication rounds to reach certain precision in minimizing $f(w)$.

1.1. Communication Efficiency of Distributed Algorithms

We assume that each communication round requires only simple map-reduce type of operations, such as broadcasting a vector in $\mathbb{R}^d$ to the m machines and computing the sum or average of m vectors in $\mathbb{R}^d$ (Dean & Ghemawat, 2008). Typically, if a distributed iterative algorithm takes T iterations to converge, then it communicates at least T rounds. Therefore, we can measure the communication efficiency of a distributed algorithm by its iteration complexity $T(\epsilon)$, which is the number of iterations required by the algorithm to find a solution w_T such that $f(w_T) - f(w_\star) \le \epsilon$, where $w_\star = \arg\min f(w)$ is the optimal solution.

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

For a concrete discussion, we make the following assumption:

Assumption A. *The function $f : \mathbb{R}^d \to \mathbb{R}$ is twice continuously differentiable, and there exist constants $L \geq \lambda > 0$ such that*

$$\lambda I \preceq f''(w) \preceq LI, \qquad \forall\, w \in \mathbb{R}^d,$$

where $f''(w)$ denotes the Hessian of f at w, and I is the $d \times d$ identity matrix.

Functions that satisfy Assumption A are often called L-smooth and λ-strongly convex. The value $\kappa = L/\lambda \geq 1$ is called the *condition number* of f, which is a key quantity in characterizing the complexity of iterative algorithms. We focus on ill-conditioned cases where $\kappa \gg 1$.

A straightforward approach for minimizing $f(w)$ is distributed implementation of the gradient descent method. More specifically, at each iteration k, each machine computes the local gradient $f_i'(w_k) \in \mathbb{R}^d$ and sends it to a master node to compute $f'(w_k) = (1/m)\sum_{i=1}^m f_i'(w_k)$. The master node takes a gradient step to compute w_{k+1}, and broadcasts it to each machine for the next iteration. The iteration complexity of the classical gradient method is $\mathcal{O}(\kappa \log(1/\epsilon))$, which is linear in the condition number κ. If we use accelerated gradient methods (Nesterov, 2004, Section 2.2), then the iteration complexity can be improved to $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$.

Another popular technique for distributed optimization is to use the alternating direction method of multipliers (ADMM); see, e.g., Boyd et al. (2010). Under Assumption A, the ADMM approach can achieve linear convergence, and the best known complexity is $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$ (Deng & Yin, 2012). This turns out to be the same order as for accelerated gradient methods.

The polynomial dependence of the iteration complexity on κ is unsatifactory. For machine learning applications, both the precision ϵ and the regularization parameter λ should decrease while the overall sample size mn increases, typically on the order of $\Theta(1/\sqrt{mn})$ (Bousquet & Elisseeff, 2002; Shalev-Shwartz et al., 2009). This translates into the condition number κ being $\Theta(\sqrt{mn})$. Consequently, the number of communication rounds scales as $(mn)^{1/4}$ for both accelerated gradient methods and ADMM. This suggests that the number of communication rounds grows with the total sample size.

Despite the rich literature on distributed optimization (e.g. Bertsekas & Tsitsiklis, 1989; Boyd et al., 2010; Agarwal & Duchi, 2011; Recht et al., 2011; Duchi et al., 2012; Dekel et al., 2012; Jaggi et al., 2014), most algorithms involve high communication cost. In particular, their iteration complexity have similar or worse dependency on the condition number as the methods discussed above. This suggests researchers to look into further structures of the problem. Zhang et al. (2012) make use of the fact that the data $z_{i,j}$ are i.i.d. sampled. Under this assumption, they studied a one-shot averaging scheme that approximates the minimizer of function f by simply averaging the minimizers of f_i. For a fixed condition number, the one-shot approach achieves the optimal statistical accuracy. But their conclusion doesn't allow the regularization parameter λ to decrease to zero as n goes to infinity.

Recently, Shamir et al. (2014) proposed a distributed approximate Newton-type (DANE) method, which also uses the stochastic assumption. For quadratic loss functions, if $\lambda \sim 1/\sqrt{mn}$ as in machine learning applications, DANE is shown to converge in $\widetilde{\mathcal{O}}(m\log(1/\epsilon))$ iterations, where the notation $\widetilde{\mathcal{O}}(\cdot)$ hides additional logarithmic factors involving m and d. This iteration complexity scales independent of the local sample size n. However, DANE does not guarantee the same convergence rate on non-quadratic problem, e.g. logistic regression and suppert vector machines.

1.2. Outline of Our Approach

In this paper, we propose a communication-efficient method for minimizing the overall empirical loss $f(w)$ defined in (2). It contains an outer-loop and an inner loop. The outer-loop employs an inexact damped Newton method to minimize $f(w)$. It is well-known that Newton-type methods have asymptotic superlinear convergence. However, in classical analysis of Newton's method (e.g., Boyd & Vandenberghe, 2004, Section 9.5.3), the number of steps needed to reach the superlinear convergence zone still depends on the condition number (scales quadratically in κ). To solve this problem, we resort to the machinery of self-concordant functions (Nesterov & Nemirovski, 1994; Nesterov, 2004). For self-concordant empirical losses, we show that the iteration complexity of the inexact damped Newton method has a much weaker dependence on the condition number.

In order to compute the Newton step, a straightforward yet naive approach would require all the machines to send their gradients $f_i'(w_k)$ and Hessians $f_i''(w_k)$ to a master node to form the global gradient and global Hessian, and then a damped Newton step is taken to compute the next iterate w_{k+1}. However, in the distributed setting, the task of transmitting the Hessians (which are $d \times d$ matrices) can be prohibitive for large dimensions d. Instead, we propose to use iterative algorithms which requires an inner-loop. In particular, we use a preconditioned conjugate gradient (PCG) method to compute an inexact Newton step. The PCG method only communicates first-order information of size $\mathcal{O}(d)$. We show that by carefully choosing the preconditioning matrix, the inner-loop's iteration complexity will also have weak dependence on the condition number.

Table 1. Communication efficiency of several distributed algorithms for ERM of linear predictors when $\lambda \sim 1/\sqrt{mn}$.

Algorithm	Number of Communication Rounds $\widetilde{\mathcal{O}}(\cdot)$	
	Ridge Regression (quadratic loss)	Binary Classification (logistic loss, smoothed hinge loss)
Accelerated Gradient	$(mn)^{1/4}\log(1/\epsilon)$	$(mn)^{1/4}\log(1/\epsilon)$
ADMM	$(mn)^{1/4}\log(1/\epsilon)$	$(mn)^{1/4}\log(1/\epsilon)$
DANE (Shamir et al., 2014)	$m\log(1/\epsilon)$	$(mn)^{1/2}\log(1/\epsilon)$
DiSCO (this paper)	$m^{1/4}\log(1/\epsilon)$	$m^{3/4}d^{1/4} + m^{1/4}d^{1/4}\log(1/\epsilon)$

We call our approach with inner-outer loops Distributed Self-Concordant Optimization (DiSCO). Table 1 lists the number of communication rounds required by DiSCO and several other algorithms to find an ϵ-optimal solution for linear regression and binary classifiaciton problems. These results are obtained when the regularization parameter λ is set to be on the order of $1/\sqrt{mn}$. All results are deterministic or high probability upper bounds, except that the last one, DiSCO for binary classification, is a bound in expectation (with respect to the randomness in generating the i.i.d. samples).

As shown in Table 1, the communication cost of DiSCO weakly depends on the number of machines m and on the feature dimension d, and is independent of the local sample size n (excluding logarithmic factors). Comparing to DANE (Shamir et al., 2014), DiSCO not only improves the communication efficiency on quadratic loss, but also handles non-quadratic classification tasks.

2. Self-concordant Empirical Loss

The theory of self-concordant functions were developed by Nesterov & Nemirovski (1994) for the analysis of interior-point methods. Roughly speaking, a function is called self-concordant if its third derivative can be controlled, in a specific way, by its second derivative. Suppose the function $f : \mathbb{R}^d \to \mathbb{R}$ has continuous third derivatives. We use $f''(w) \in \mathbb{R}^{d\times d}$ to denote its Hessian at $w \in \mathbb{R}^d$, and use $f'''(w)[u] \in \mathbb{R}^{d\times d}$ to denote the limit

$$f'''(w)[u] \overset{\text{def}}{=} \lim_{t\to 0} \frac{1}{t}\bigl(f''(w+tu) - f''(w)\bigr).$$

Definition 1. *A convex function $f : \mathbb{R}^d \to \mathbb{R}$ is self-concordant with parameter M_f if the inequality*

$$\left|u^T(f'''(w)[u])u\right| \le M_f \left(u^T f''(w)u\right)^{3/2}$$

holds for any $w \in \mathrm{dom}(f)$ and $u \in \mathbb{R}^d$. In particular, a self-concordant function with parameter 2 is called standard self-concordant.

Detailed account of self-concordance can be found in the books by Nesterov & Nemirovski (1994) and Nesterov (2004). In this section, we show that several popular regularized empirical loss functions for linear regression and binary classification are self-concordant.

First we consider regularized linear regression (ridge regression) with

$$f(w) = \frac{1}{N}\sum_{i=1}^{N}(y_i - w^T x_i)^2 + \frac{\lambda}{2}\|w\|_2^2, \qquad (3)$$

In the setting of distributed optimization, we have $N = mn$ where m is the number of machines and n is the number of samples on each machine. In terms of the definition in (1) and (2), we have $z_i = (x_i, y_i)$ where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. Since $f(w)$ is a quadratic function, its third derivatives are all zero. Therefore, it is self-concordant with parameter 0, and by definition it is also standard self-concordant.

For binary classification, we consider the following regularized empirical loss function

$$\ell(w) \overset{\text{def}}{=} \frac{1}{N}\sum_{i=1}^{N}\varphi(y_i w^T x_i) + \frac{\gamma}{2}\|w\|_2^2, \qquad (4)$$

where $x_i \in \mathcal{X} \subset \mathbb{R}^d$, $y_i \in \{-1, 1\}$, and $\varphi : \mathbb{R} \to \mathbb{R}$ is a convex surrogate function for the binary $0/1$ loss. We further assume that the elements of $\mathcal{X}$ are bounded, that is, we have $\sup_{x\in\mathcal{X}} \|x\|_2 \le B$ for some finite B.

Logistic regression For logistic regression, we minimize the objective function (4) where φ is the logistic loss: $\varphi(t) = \log(1 + e^{-t})$. The following lemma shows that the regularized loss is self-concordant. See Zhang & Xiao (2015) for the proof.

Lemma 1. *For logistic regression, the empirical loss $\ell(w)$ is self-concordant with parameter $B/\sqrt{\gamma}$, and the scaled loss $f(w) = (B^2/(4\gamma))\ell(w)$ is standard self-concordant.*

Smoothed hinge loss In classification tasks, it is sometimes more favorable to use the hinge loss $\varphi(t) = \max\{0, 1-t\}$ than using the logistic loss. We consider a family of smoothed hinge loss functions φ_p parametrized

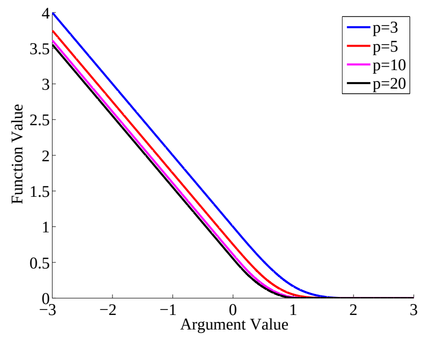

Figure 1. Smoothed hinge loss φ_p with $p = 3, 5, 10, 20$.

by a positive number $p \geq 3$. The function is defined by

$$\varphi_p(t) = \begin{cases} \frac{3}{2} - \frac{p-2}{p-1} - t & \text{for } t \in (-\infty, -\frac{p-3}{p-1}], \\ \frac{3}{2} - \frac{p-2}{p-1} - t + \frac{(t+(p-3)/(p-1))^p}{p(p-1)} & \text{for } t \in (-\frac{p-3}{p-1}, 1 - \frac{p-3}{p-1}], \\ \frac{p+1}{p(p-1)} - \frac{t}{p-1} + \frac{1}{2}(1-t)^2 & \text{for } t \in (1 - \frac{p-3}{p-1}, 1], \\ \frac{(2-t)^p}{p(p-1)} & \text{for } t \in (1, 2], \\ 0 & \text{for } t \in (2, +\infty). \end{cases} \quad (5)$$

We plot the functions φ_p for $p = 3, 5, 10, 20$ on Figure 1. As the plot shows, $\varphi_p(t)$ is zero for $t > 2$, and it is a linear function with unit slope for $t < -\frac{p-3}{p-1}$. These two linear zones are connected by three smooth non-linear segments on the interval $[-\frac{p-3}{p-1}, 2]$.

The following lemma shows that the regularized loss is self-concordant. See Zhang & Xiao (2015) for the proof.

Lemma 2. *For the smoothed hinge loss defined in* (5)*, the empirical loss $\ell(w)$ is self-concordant with parameter*

$$M_p = \frac{(p-2)B^{1+\frac{2}{p-2}}}{\gamma^{\frac{1}{2}+\frac{1}{p-2}}}, \quad (6)$$

and the scaled loss function $f(w) = (M_p^2/4)\ell(w)$ is standard self-concordant.

3. Inexact Damped Newton Method

In this section, we propose and analyze an inexact damped Newton method for minimizing self-concordant functions. Without loss of generality, we assume the objective function $f : \mathbb{R}^d \to \mathbb{R}$ is standard self-concordant. In addition, we assume that Assumption A holds. Our method is described in Algorithm 1. If we let $\epsilon_k = 0$ for all $k \geq 0$, then Algorithm 1 reduces to the exact damped Newton method (e.g., Nesterov, 2004, Section 4.1.5). The explicit account of approximation errors is essential for distributed optimization, because with limited communication budget, we can only perform Newton updates approximately.

The following theorem upper bounds the iteration complexity of Algorithm 1 for obtaining an arbitrary accuracy. The theorem is proved in the long version of this paper (Zhang & Xiao, 2015).

Algorithm 1 Inexact damped Newton method

```
input: initial point $w_0$ and specification of a nonnegative sequence $\{\epsilon_k\}$.
repeat for $k = 0, 1, 2, \ldots$:
  1. Find $v_k \in \mathbb{R}^d$ such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$.
  2. Compute $\delta_k = \sqrt{v_k^T f''(w_k) v_k}$ and update
     $w_{k+1} = w_k - \frac{1}{1+\delta_k} v_k$.
until a stopping criterion is satisfied.
```

Theorem 1. *Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is a standard self-concordant function and Assumption A holds. If we choose the sequence $\{\epsilon_k\}$ in Algorithm 1 as*

$$\epsilon_k = \beta(\lambda/L)^{1/2}\|f'(w_k)\|_2 \quad \textit{with } \beta = 1/20, \quad (7)$$

then for any $\epsilon > 0$, we have $f(w_k) - f(w_\star) \leq \epsilon$ whenever $k \geq K$ where $w_\star = \arg\min f(w)$ and

$$K = \left\lceil \frac{f(w_0) - f(w_\star)}{\frac{1}{2}\omega(1/6)} \right\rceil + \left\lceil \log_2\left(\frac{2\,\omega(1/6)}{\epsilon}\right) \right\rceil. \quad (8)$$

Here, $\omega(t) = t - \log(1+t)$ (which appears often in the literature on interior-point methods) and $\lceil t \rceil$ denotes the smallest nonnegative *integer that is larger or equal to t.*

Theorem 1 shows that after a constant number of steps (proportional to the initial gap $f(w_0) - f(w_\star)$), Algorithm 1 has a linear rate of convergence characterized by the term $\log(1/\epsilon)$. This is slower than the quadratic convergence rate of the exact damped Newton method, due to the allowed approximation errors in computing the Newton step. Actually, if we set the tolerances ϵ_k to be sufficiently small, then superlinear convergence can be established, i.e., the second term in (8) can be replaced by $\log(\log(1/\epsilon))$ (Zhang & Xiao, 2015). The choice in equation (7) is a reasonable trade-off in practice. We note that the self-concordance property is essential in the proof of Theorem 1. The inexact damped Newton method won't have convergence rate (8) unless the objective function is self-concordance.

Theorem 1 states that the iteration complexity of the damped Newton method is proportional to

$$f(w_0) - f(w_\star) + \log(1/\epsilon).$$

In many applications, the function $f(w)$ is obtained via scaling to be standard self-concordant (see examples of logistic regression and smoothed hinge loss in Section 2). When the scaling factor is large, we need to choose the initial point w_0 judiciously to guarantee that the initial gap $f(w_0) - f(w_\star)$ is small.

Algorithm 2 Distributed PCG algorithm

```
input: $w_k \in \mathbb{R}^d$ and $\mu \geq 0$.
Let $H = f''(w_k)$ and $P = f_1''(w_k) + \mu I$.
communication: The master machine broadcasts $w_k$ to other machines to compute $f_i'(w_k)$, for $i = 1, \ldots, m$; then it aggregates $f_i'(w_k)$ to form $f'(w_k)$.
initialization: Compute $\epsilon_k$ given in (7) and set
    $v^{(0)} = 0, \quad s^{(0)} = P^{-1}r^{(0)},$
    $r^{(0)} = f'(w_k), \quad u^{(0)} = s^{(0)}.$
repeat for $t = 0, 1, 2 \ldots,$
    1. communication: The master machine broadcasts $u^{(t)}$ to other machines to compute $f_i''(w_k)u^{(t)}$; then aggregates them to form the vector $Hu^{(t)}$.
    2. Compute $\alpha_t = \frac{\langle r^{(t)}, s^{(t)} \rangle}{\langle u^{(t)}, Hu^{(t)} \rangle}$ and update:
        $v^{(t+1)} = v^{(t)} + \alpha_t u^{(t)}, \; Hv^{(t+1)} = Hv^{(t)} + \alpha_t Hu^{(t)}$
        and $r^{(t+1)} = r^{(t)} - \alpha_t Hu^{(t)}.$
    3. Compute $\beta_t = \frac{\langle r^{(t+1)}, s^{(t+1)} \rangle}{\langle r^{(t)}, s^{(t)} \rangle}$ and update:
        $s^{(t+1)} = P^{-1}r^{(t+1)},$
        $u^{(t+1)} = s^{(t+1)} + \beta_t u^{(t)}.$
until $\|r^{(t+1)}\|_2 \leq \epsilon_k$
return: $v_k = v^{(t+1)}, r_k = r^{(t+1)}$, and
    $\delta_k = \sqrt{v_k^T Hv^{(t)} + \alpha^{(t)} v_k^T Hu^{(t)}}.$
```

4. The DiSCO Algorithm

In this section, we adapt the inexact damped Newton method (Algorithm 1) to a distributed system, in order to minimize $f(w) = \frac{1}{m}\sum_{i=1}^m f_i(w)$, where each function f_i can only be evaluated locally at machine i (see background in Section 1). This involves two questions: (1) how to set the initial point w_0 and (2) how to compute the inexact Newton step v_k in a distributed manner.

4.1. Initialization

In accordance with the averaging structure in the objective function, we choose the initial point based on averaging:

$$w_0 = \frac{1}{m}\sum_{i=1}^m \widehat{w}_i, \tag{9}$$

where each $\widehat{w}_i$ is computed locally at machine i as

$$\widehat{w}_i = \arg\min_{w \in \mathbb{R}^d} \left\{ f_i(w) + \frac{\rho}{2}\|w\|_2^2 \right\}. \tag{10}$$

Here $\rho \geq 0$ is a regularization parameter, which we will discuss in Section 5. Roughly speaking, we can choose $\rho \sim 1/\sqrt{n}$ to make $\mathbb{E}[f(w_0) - f(w_\star)]$ decreasing as $\mathcal{O}(1/\sqrt{n})$.

4.2. Distributed Computing of the Inexact Newton Step

In each iteration of Algorithm 1, we need to compute an inexact Newton step v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$. This boils down to solving the Newton system $f''(w_k)v_k = f'(w_k)$ approximately. We propose to use a distributed preconditioned conjugate gradient (PCG) method to solve the Newton system.

To simplify notation, we use H to represent $f''(w_k)$ and H_i to represent $f_i''(w_k)$. We define a preconditioning matrix using the local Hessian at the first machine (the master):

$$P \overset{\text{def}}{=} H_1 + \mu I,$$

where $\mu > 0$ is a small regularization parameter. Algorithm 2 describes our distributed PCG method for solving the preconditioned linear system

$$P^{-1}Hv_k = P^{-1}f'(w_k).$$

In Algorithm 2, the master machine carries out the the classical PCG algorithm (e.g., Golub & Van Loan, 1996, Section 10.3), and other machines compute the local gradients and Hessians and perform matrix-vector multiplications. Communication between the master and other machines are used to form the overall gradient $f'(w_k)$ and the matrix-vector products $Hu^{(t)} = \frac{1}{m}\sum_{i=1}^m f_i''(w_k)u^{(t)}$. We note that the overall Hessian $H = f''(w_k)$ is never formed and the master only stores and updates the vector $Hu^{(t)}$.

When H_1 is sufficiently close to H and μ is chosen to be sufficiently small, then the condition number of $P^{-1}H$ will be close to 1. This makes the PCG method converging much faster than the standard conjugate gradient method. See Zhang & Xiao (2015) for the proof of the following lemma.

Lemma 3. *Suppose Assumption A holds and assume that $\|H_1 - H\|_2 \leq \mu$. Let β be the constant in (7) and define*

$$T_\mu = \left\lceil \sqrt{1 + \frac{2\mu}{\lambda}} \log\left(\frac{2L}{\beta\lambda}\right) \right\rceil. \tag{11}$$

Then Algorithm 2 terminates in T_μ iterations and the output v_k satisfies $\|Hv_k - f'(w_k)\|_2 \leq \epsilon_k$.

Under Assumption A, we always have $\|H_1 - H\|_2 \leq L$. If we let $\mu = L$, then Lemma 3 implies that Algorithm 2 terminates in $\widetilde{\mathcal{O}}(\sqrt{L/\lambda})$ iterations. In practice, however, the matrix norm $\|H_1 - H\|_2$ is usually much smaller than L due to the stochastic nature of f_i. Thus, we can choose μ to be a tight upper bound on $\|H_1 - H\|_2$, and expect the algorithm to terminate in $\widetilde{\mathcal{O}}(\sqrt{1 + \mu/\lambda})$ iterations.

A similar approach was proposed by Zhuang et al. (2014) and Lin et al. (2014) for ERM of linear predictors, where they use a distributed truncated Newton method and conjugate gradient (CG) method for solving the Newton system. However, they did not employ preconditioning in the CG

Algorithm 3 DiSCO

```
input: parameters $\rho, \mu \geq 0$ and number of iterations $K$.
initialize: compute $w_0$ according to (9) and (10).
repeat for $k = 0, 1, 2, \ldots, K$:
    1. Run Algorithm 2: given $w_k$, compute $v_k$ and $\delta_k$.
    2. Update $w_{k+1} = w_k - \frac{1}{1+\delta_k} v_k$.
output: $\widehat{w} = w_{K+1}$.
```

method; consequently, the total number of the CG iterations may still be high for ill-conditioned problems.

4.3. DiSCO and its communication efficiency

Putting all pieces together, we summarize the DiSCO method in Algorithm 3. DiSCO combines the inexact damped Newton method (Algorithm 1) and the distributed PCG method (Algorithm 2). For each execution of the PCG method, if Algorithm 2 runs for T iterations, then there will be $T+1$ rounds of communication.

The following theorem provides an upper bound on the number of communication rounds required by the DiSCO algorithm to find an ϵ-optimal solution. It combines the theoretical results given in Theorem 1 and Lemma 3.

Theorem 2. *Assume that $f(w) = (1/m)\sum_{i=1}^m f_i(w)$ is standard self-concordant and it satisfies Assumptions A. Suppose the input parameter μ is chosen such that $\|f_1''(w_k) - f''(w_k)\|_2 \leq \mu$ for all $k \geq 0$. Then for any $\epsilon > 0$, the total number of communication rounds T required by DiSCO to reach $f(\widehat{w}) - f(w_\star) \leq \epsilon$ is bounded by*

$$T = \widetilde{\mathcal{O}}\left(\big(f(w_0) - f(w_\star) + \log(1/\epsilon)\big)\sqrt{1+\mu/\lambda}\right).$$

where $\widetilde{\mathcal{O}}(\cdot)$ hides logarithmic terms involving L and λ.

We wrap up this section by studying the computation complexity of DiSCO. For the initialization step, each machine needs to compute (10). For Algorithm 2, the bulk of computation is at computing the vector $s^{(t)} = P^{-1}r^{(t)}$ in Step 3, which is equivalent to minimize the quadratic function $(1/2)s^T P s - s^T r^{(t)}$. Both are convex optimization problems whose objective functions are the average of a finite number of local component functions. They can be solved to high accuracy by the stochastic average gradient (SAG) method (Le Roux et al., 2012) or SAGA (Defazio et al., 2014) in $\widetilde{\mathcal{O}}(dn)$ computation time. This is roughly the same time complexity for loading the dataset.

5. Stochastic Analysis

From Theorem 2 we see that the communication efficiency of DiSCO mainly depends on two quantities: the initial objective gap $f(w_0) - f(w_\star)$ and the upper bound μ on the spectral norms $\|f_1''(w_k) - f''(w_k)\|_2$. The initial gap $f(w_0) - f(w_\star)$ may grow with the number of samples due to the scaling used to make the objective function standard self-concordant. On the other hand, the upper bound μ may decrease as the number of samples increases based on the intuition that the local Hessians and the global Hessian become similar to each other.

In this section, we show how the choice of w_0 in Section 4.1 can mitigate the effect of objective scaling by reducing the initial gap, and also quantify the similarity between local and global Hessians. Both relies on the assumption that the random vectors $z_{i,j}$ in the definition of the empirical loss in (1) are i.i.d. samples from a common distribution. Throughout this section, we take expectation with respect to the randomness in generating the i.i.d. data. All theoretical results of this section are proved in the long version of this paper (Zhang & Xiao, 2015).

To formalize the argument, we need additional assumptions on the smoothness of the loss function ϕ in equation (1).

Assumption B. *There are finite constants (V_0, G, L, M), such that for any $z \in \mathcal{Z}$:*

(i) $\phi(w,z) \geq 0$ *for all* $w \in \mathbb{R}^d$, *and* $\phi(0,z) \leq V_0$;

(ii) $\|\phi'(w,z)\|_2 \leq G$ *for any* $\|w\|_2 \leq \sqrt{2V_0/\lambda}$;

(iii) $\|\phi''(w,z)\|_2 \leq L - \lambda$ *for any* $w \in \mathbb{R}^d$;

(iv) $\|\phi''(u,z) - \phi''(w,z)\|_2 \leq M\|u-w\|_2$, $\forall\, u, w \in \mathbb{R}^d$.

For the regularized empirical loss defined in (1) and (2), condition (iii) in the above assumption implies $\lambda I \preceq f_i''(w) \preceq LI$ for $i = 1, \ldots, m$, which in turn implies Assumption A. These assumptions are satisfied by popular loss functions, including linear regression, logistic regression and the smoothed hinge loss defined in Section 2.

The following lemma shows that the expected value of the initial gap $f(w_0) - f(w_\star)$ decreases with order $1/\sqrt{n}$ as the local sample size n increases.

Lemma 4. *Suppose that Assumption B holds and $\mathbb{E}[\|w_\star\|_2^2] \leq D^2$ for some constant $D > 0$. If we choose $\rho = \frac{\sqrt{6}G}{\sqrt{n}D}$ in equation (10) to compute $\widehat{w}_i$, then the initial point $w_0 = \frac{1}{m}\sum_{i=1}^m \widehat{w}_i$ satisfies*

$$\mathbb{E}[f(w_0) - f(w_\star)] \leq \frac{\sqrt{6}GD}{\sqrt{n}}.$$

The next lemma quantifies the similarity between the local and global Hessians.

Lemma 5. *Suppose Assumption B holds and the sequence $\{w_k\}_{k\geq 0}$ is generated by Algorithm 1. Let*

$$r = \Big(\frac{2V_0}{\lambda} + \frac{2G}{\lambda}\sqrt{\frac{2V_0}{\lambda}}\Big)^{1/2}.$$

Then with probability at least $1-\delta$, we have for all $k \geq 0$,

$$\|f_1''(w_k) - f''(w_k)\|_2 \leq \sqrt{\frac{32L^2 d}{n}} \cdot \sqrt{\log\Big(1 + \frac{rM\sqrt{2n}}{L}\Big) + \frac{\log(md/\delta)}{d}}.$$

Here the high probability bound is also with respect to the randomness in generating the i.i.d. data.

If $\phi(w, z_{i,j})$ are quadratic functions in w, then we have $M = 0$ in Assumption B. In this case, Lemma 5 implies

$$\|f_i''(w_k) - f''(w_k)\|_2 \leq \widetilde{\mathcal{O}}(\sqrt{L^2/n}).$$

For general non-quadratic loss, Lemma 5 implies

$$\|f_i''(w_k) - f''(w_k)\|_2 \leq \widetilde{\mathcal{O}}(\sqrt{L^2 d/n}).$$

Combining Lemma 4 and Lemma 5 with Theorem 2, we obtain the following main result.

Theorem 3. *Assume that Assumptions B holds and the function $f(w)$ defined by (1) and (2) is standard self-concordant. Let ρ be chosen as in Lemma 4, and let μ be the upper bound in Lemma 5. Let $\kappa = L/\lambda$ be the condition number of the problem. Then for any $\epsilon > 0$, the expected number of communication rounds T required by DiSCO to reach $f(\widehat{w}) - f(w_\star) \leq \epsilon$ is bounded as*

$$\mathbb{E}[T] = \widetilde{\mathcal{O}}\Big(\Big(\frac{GD}{\sqrt{n}} + \log(1/\epsilon)\Big)\Big(1 + \frac{\kappa^2}{n}\Big)^{1/4}\Big)$$

for the quadratic loss, and

$$\mathbb{E}[T] = \widetilde{\mathcal{O}}\Big(\Big(\frac{GD}{\sqrt{n}} + \log(1/\epsilon)\Big)\Big(1 + \frac{\kappa^2 d}{n}\Big)^{1/4}\Big)$$

for other convex loss functions.

To apply the above results to specific loss functions, we assume the standard setting for supervised learning where the regularization parameter is on the order of $1/\sqrt{mn}$, thus the condition number $\kappa = \Theta(\sqrt{mn})$.

Linear Regression For linear regression, the loss function $f(w)$ takes the form (3). Since f is a quadratic function, it is standard self-concordant without scaling. This means that G and D are bounded constants, and we have $GD/\sqrt{n} = o(1)$. Then Theorem 3 implies

$$\mathbb{E}[T] = \widetilde{\mathcal{O}}(m^{1/4}\log(1/\epsilon)).$$

In fact, since we do not need to rescale the objective function, we can regard the initial gap $f(w_0) - f(w_\star)$ as a constant. As a consequence, we can directly apply Theorem 2 and Lemma 5 to show that $T = \widetilde{\mathcal{O}}(m^{1/4}\log(1/\epsilon))$ holds with high probability.

Logistic Regression For logistic regression, consider the loss function $\ell(w)$ defined in (4) with $\gamma = \Theta(1/\sqrt{mn})$. By Lemma 1, the scaled function $f(w) = \frac{B^2}{4\gamma}\ell(w)$ is standard self-concordant. As a result, the constants (V_0, G, L, M) in Assumption B also need to be scaled by $\frac{B^2}{4\gamma} = \Theta(\sqrt{mn})$. So we have $GD/\sqrt{n} = \Theta(\sqrt{m})$. Then Theorem 3 implies

$$\mathbb{E}[T] = \widetilde{\mathcal{O}}\Big(m^{3/4}d^{1/4} + m^{1/4}d^{1/4}\log(1/\epsilon)\Big). \tag{12}$$

Smoothed Hinge Loss We consider the function $\ell(w)$ defined in (3) with the smoothed hinge loss φ_p defined in (5). By Lemma 2, the function $f(w) = \frac{M_p^2}{4}\ell(w)$ is standard self-concordant, where M_p is defined in (6). By choosing $p = 2 + \log(1/\gamma)$, we obtain $M_p^2/4 = \widetilde{\mathcal{O}}(1/\gamma) = \widetilde{\mathcal{O}}(\sqrt{mn})$. This scaling factor is on the same order as for logistic regression. Thus, the smoothed hinge loss enjoys the same communication efficiency in (12).

6. Numerical Experiments

In this section, we conduct numerical experiments to compare the DiSCO algorithm with several state-of-the-art distributed optimization algorithms: the ADMM algorithm (Boyd et al., 2010), the accelerated full gradient method (AFG) (Nesterov, 2004), the L-BFGS quasi-Newton method (Nocedal & Wright, 2006, Section 7.2), and the DANE algorithm (Shamir et al., 2014).

Table 2. Summary of three binary classification datasets.

Dataset	# samples N	# features d
Covtype	581,012	54
RCV1	20,242	47,236
News20	19,996	1,355,191

For comparison, we solve three binary classification tasks using logistic regression. The datasets are obtained from the LIBSVM datasets (Chang & Lin, 2011) and summarized in Table 2. These datasets are selected to cover different relations between the sample size $N = mn$ and the feature dimensionality d. For each dataset, our goal is to minimize the regularized logistic loss (4). The regularization parameter is set to be $\gamma = 10^{-5}$.

We describe some implementation details. Altough the theoretical analysis suggests that we scale the function $\ell(w)$ by a factor $\eta = B^2/(4\gamma)$. In practice, we find that DiSCO converges faster without rescaling. Thus, we use $\eta = 1$ for all experiments. For Algorithm 3, we choose the input parameters $\mu = m^{1/2}\mu_0$, where μ_0 is manually chosen to be $\mu_0 = 0$ for Covtype, $\mu_0 = 4 \times 10^{-4}$ for RCV1, and $\mu_0 = 2 \times 10^{-4}$ for News20. To monitor the progress of DiSCO, after every iteration of the inner loop (Algorithm 2), we take $v^{(t)}$ to compute an intermediate solution

$$\widehat{w}_k^t = w_k - \frac{v^{(t)}}{1 + \sqrt{(v^{(t)})^T \ell''(w_k) v^{(t)}}},$$

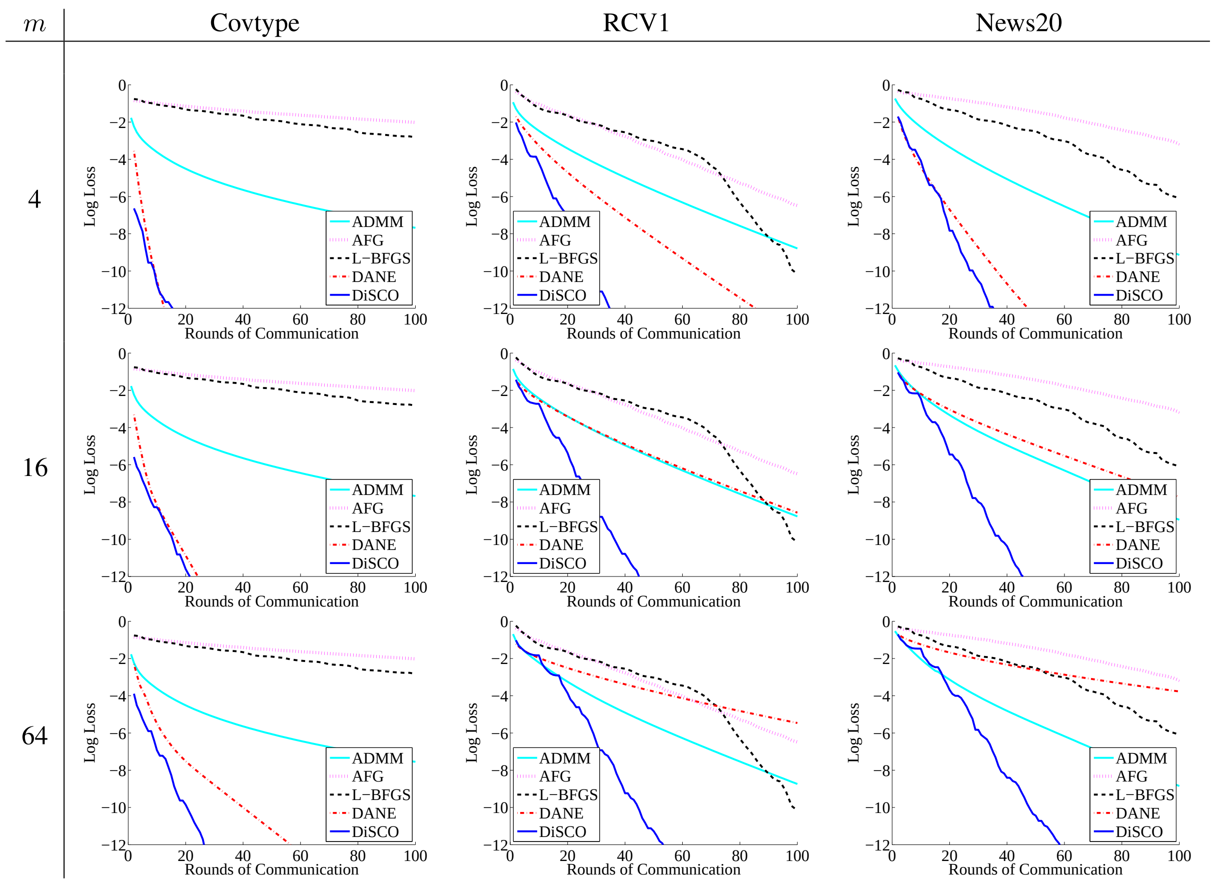

Figure 2. Comparing DiSCO with other distributed optimization algorithms. We splits each dataset evenly to m machines, with $m \in \{4, 16, 64\}$. Each plot above shows the reduction of the logarithmic gap $\log_{10}(\ell(\widehat{w}) - \ell(w_\star))$ (the vertical axis) versus the number of communication rounds (the horizontal axis) taken by each algorithm.

and evaluate the associated objective function $\ell(\widehat{w}_k^t)$.

For fair comparison, we manually tune the penalty parameter of ADMM and the regularization parameter μ for DANE to optimize their performance. For AFG, we used an adaptive line search scheme (Nesterov, 2013) to speed up its convergence. For L-BFGS, we adopted the memory size 30, as suggested in Nocedal & Wright (2006).

It is important to note that different algorithms take different number of communication rounds per iteration. ADMM requires one round of communication per iteration. For AFG and L-BFGS, each iteration consists of at least two rounds of communications: one for finding the descent direction, and another one or more for searching the stepsize. For DANE, there are also two rounds of communications per iteration, for computing the gradient and for aggregating the local solutions. For DiSCO, each iteration in the inner loop takes one round of communication, and there is an additional round of communication at the beginning of each inner loop. Since we are interested in the communication efficiency of the algorithms, we plot in Figure 2 their progress in reducing the objective value versus the number of communication rounds taken.

According to the plots in Figure 2, DiSCO converges substantially faster than ADMM and AFG. It is also notably faster than L-BFGS and DANE. In particular, the convergence speed (and the communication efficiency) of DiSCO is more robust to the number of machines in the distributed system. For $m = 4$, the performance of DiSCO is somewhat comparable to that of DANE. As m grows to 16 and 64, the convergence of DANE becomes significantly slower, while the performance of DiSCO only degrades slightly. This coincides with the theoretical analysis: the iteration complexity of DANE is proportional to m, but the iteration complexity of DiSCO is proportional to $m^{1/4}$.

Thus, our experiments on real datasets confirmed the superior communication efficiency of the DiSCO algorithm. We note that when comparing to ADMM, AFG, L-BFGS and DANE (in the case of large m), the DiSCO algorithm is not only communication efficient but also faster in the sense of computation, because DiSCO requires much fewer iterations to converge to a high-accuracy solution.

References

Agarwal, A. and Duchi, J. C. Distributed delayed stochastic optimization. In *Advances in NIPS*, pp. 873–881, 2011.

Bekkerman, R., Bilenko, M., and Langford, J. *Scaling up Machine Learning: Parallel and Distributed Approaches*. Cambridge University Press, 2011.

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

Bousquet, O. and Elisseeff, A. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

Chang, C.-C. and Lin, C.-J. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

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

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 NIPS 27*, pp. 1646–1654. 2014.

Dekel, O., Gilad-Bachrach, R., Shamir, O., and Xiao, L. Optimal distributed online prediction using mini-batches. *The Journal of Machine Learning Research*, 13 (1):165–202, 2012.

Deng, W. and Yin, W. On the global and linear convergence of the generalized alternating direction method of multipliers. CAAM Technical Report 12-14, Rice University, 2012.

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

Golub, G. H. and Van Loan, C. F. *Matrix Computations*. The John Hopkins University Press, Baltimore, MD, third edition, 1996.

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

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

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

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

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

Nesterov, Y. and Nemirovski, A. *Interior Point Polynomial Time Methods in Convex Programming*. SIAM, Philadelphia, 1994.

Nocedal, J. and Wright, S. J. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

Recht, B., Re, C., Wright, S. J., and Niu, F. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in NIPS*, pp. 693–701, 2011.

Shalev-Shwartz, S., Shamir, O., Srebro, N., and Sridharan, K. Stochastic convex optimization. In *Proceedings of the 22nd Annual Conference on Learning Theory*, 2009.

Shalf, J., Dosanjh, S., and Morrison, J. Exascale computing technology challenges. In *Proceedings of the 9th International Conference on High Performance Computing for Computational Science*, VECPAR'10, pp. 1–25. Springer-Verlag, 2011.

Shamir, O., Srebro, N., and Zhang, T. Communication efficient distributed optimization using an approximate Newton-type method. In *Proceedings of ICML*. JMLR: W&CP volume 32, 2014.

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

Zhang, Y. and Xiao, L. Communication-efficient distributed optimization of self-concordant empirical loss. *arXiv preprint arXiv:1501.00263*, 2015.

Zhuang, Y., Chin, W.-S., Juan, Y.-C., and Lin, C.-J. Distributed newton method for regularized logistic regression. Technical report, Department of Computer Science, National Taiwan University, 2014.