Efficient Distributed Second-Order Optimization for Self-Concordant Empirical Losses

Chris Junchi Li$^{\diamond}$

Department of Electrical Engineering and Computer Sciences$^{\diamond}$
University of California, Berkeley

October 8, 2024

Abstract

Solving large-scale optimization problems efficiently is critical in data science applications, where vast datasets need to be processed across multiple machines. We focus on distributed optimization of empirical loss functions that are self-concordant, a property that enables the use of second-order methods for improved convergence rates. This paper proposes a communication-efficient distributed second-order method, leveraging inexact Newton steps and preconditioned conjugate gradient (PCG) techniques. Our algorithm significantly reduces the number of communication rounds compared to classical gradient-based methods while maintaining a favorable iteration complexity. We analyze the performance in terms of communication efficiency and provide theoretical guarantees under stochastic settings. The proposed method is particularly well-suited for machine learning tasks, such as regularized regression and classification, where empirical risk minimization is common.

Keywords: Distributed optimization, self-concordant functions, second-order methods, communication efficiency, machine learning

1 Introduction

With the rapid growth of data across various fields such as machine learning, statistics, and data science, optimization algorithms must handle increasingly larger datasets. Many of these problems are iterative and require accessing the entire dataset at each iteration, making scalability and communication efficiency critical in distributed settings. When the dataset is too large to fit into the memory of a single machine, distributed optimization becomes a necessity. The key challenge here is to minimize communication overhead between machines while maintaining high computational efficiency.

Many optimization problems in data science (including statistics, machine learning, data mining, etc.) are formulated with a large amount of data as input. Typically, such problems are solved by iterative algorithms, which need to access the entire dataset or at least part of it during each iteration. As the volume of data continues to grow at a rapid pace, it becomes increasingly common for the dataset involved in an optimization problem to exceed the memory capacity of a single computer. To solve these ”big data” optimization problems, distributed algorithms, which rely on inter-machine communication, are essential.

In this paper, we focus on distributed optimization problems generated through *sample average approximation* (SAA) of stochastic optimization problems. Specifically, we address the problem of optimizing empirical loss functions that exhibit the property of self-concordance. Self-concordance allows the use of second-order methods, such as Newton's method, which are known for their super-linear convergence rates. However, in a distributed environment, communication between machines can become the bottleneck, especially for second-order methods, which require computation of Hessians. Our goal is to develop algorithms that minimize communication rounds while achieving high convergence efficiency.

We propose a distributed second-order optimization algorithm that leverages inexact Newton methods with preconditioned conjugate gradient (PCG) steps to handle large-scale data efficiently. Our method, termed Distributed Self-Concordant Optimization (DiSCO), is designed to minimize communication overhead by exploiting the stochastic properties of the empirical loss function, while maintaining high computational performance.

Mathematically, we consider the problem

$$\underset{w\in\mathbb{R}^d}{\text{minimize}} \quad \mathbb{E}_z[\phi(w,z)] \tag{1}$$

where z is a random vector 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 $z\in\mathcal{Z}$. In general, evaluating the expected objective function with respect to z is intractable, even if the distribution is given. The idea of SAA is to approximate the solution to (1) by solving a deterministic problem defined over a large number of i.i.d. (independent and identically distributed) samples generated from the distribution of z (see, *e.g.*, [48, Chapter 5]). Suppose our distributed computing system consists of m machines, and each has access to n samples $z_{i,1},\ldots,z_{i,n}$, for $i=1,\ldots,m$. Then each machine can evaluate a local empirical loss function

$$f_i(w) \overset{\text{def}}{=} \frac{1}{n}\sum_{j=1}^n \phi(w,z_{i,j}), \qquad i=1,\ldots,m$$

Our goal is to minimize the overall empirical loss defined with all mn samples:

$$f(w) \overset{\text{def}}{=} \frac{1}{m}\sum_{i=1}^m f_i(w) = \frac{1}{mn}\sum_{i=1}^m\sum_{j=1}^n \phi(w,z_{i,j}) \tag{2}$$

In machine learning applications, the probability distribution of z is usually unknown, and the SAA approach is referred as *empirical risk minimization* (ERM). As a concrete example, we consider ERM of linear predictors for supervised learning. In this case, each sample has the form $z_{i,j}=(x_{i,j},y_{i,j})\in\mathbb{R}^{d+1}$, where $x_{i,j}\in\mathbb{R}^d$ is a feature vector and $y_{i,j}$ can be a target response in $\mathbb{R}$ (for regression) or a discrete label (for classification). Examples of the loss function include

- linear regression: $x\in\mathbb{R}^d$, $y\in\mathbb{R}$, and $\phi(w,(x,y))=(y-w^Tx)^2$.
- logistic regression: $x\in\mathbb{R}^d$, $y\in\{+1,-1\}$, and $\phi(w,(x,y))=\log(1+\exp(-y(w^Tx)))$.
- hinge loss: $x\in\mathbb{R}^d$, $y\in\{+1,-1\}$, and $\phi(w,(x,y))=\max\{0,\ 1-y(w^Tx)\}$.

For stability and generalization purposes, we often add a regularization term $(\lambda/2)\|w\|_2^2$ to make the empirical loss function strongly convex. More specifically, we modify the definition of $f_i(w)$ as

$$f_i(w) \overset{\text{def}}{=} \frac{1}{n}\sum_{j=1}^n \phi(w,z_{i,j}) + \frac{\lambda}{2}\|w\|_2^2, \qquad i=1,\ldots,m \tag{3}$$

For example, when ϕ is the hinge loss, this formulation yields the *support-vector machine* [12].

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 communication round involving simple map-reduce type of operations [13, 32]. 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., [4, 45]); 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 $\mathbb{R}^d$. Typically, if a distributed iterative algorithm takes T iterations to converge, then it communicates at least T rounds (usually one or two communication rounds per iteration). 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 $f(w_T) - f(w_\star) \leq \epsilon$.

For a concrete discussion, we make the following assumption:

Assumption 1. *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 1 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 classical 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 this method is the same as the classical gradient method: $\mathcal{O}(\kappa \log(1/\epsilon))$, which is linear in the condition number κ (*e.g.*, [34]). If we use accelerated gradient methods [34, 35, 28], 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., [8, Section 8]. Under the assumption that each local function f_i is L-smooth and λ-strongly convex, the ADMM approach can achieve linear convergence, and the best known complexity is $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$ [17]. This turns out to be the same order as for accelerated gradient methods. In this case, ADMM can actually be considered as an accelerated primal-dual first-order method; see the discussions in [10, Section 4].

The polynomial dependence of the iteration complexity on the condition number can be 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 $\Theta(1/\sqrt{mn})$ (*e.g.*, [7, 42]). This translates into the condition number κ being $\Theta(\sqrt{mn})$. In this case, the iteration complexity, and thus the number of communication rounds, scales as $(mn)^{1/4}$ for both accelerated gradient methods and ADMM (with careful tuning of the penalty parameter). This suggests that the number of communication rounds grows with the total sample size.

Despite the rich literature on distributed optimization (e.g., [5, 38, 8, 1, 19, 15, 39, 52, 46]), 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. It can be argued that the iteration complexity $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$ cannot be improved in general for distributed first-order methods — after all, it is optimal for centralized first-order methods under the same assumption that $f(w)$ is L-smooth and λ-strongly convex [33, 34]. Thus in order to obtain better communication efficiency, we need to look into further problem structure and/or alternative optimization methods. And we need both in this paper.

First, we note that the above discussion on iteration complexity does not exploit the fact that each function f_i is generated by, or can be considered as, SAA of a stochastic optimization problem. Since the data $z_{i,j}$ are i.i.d. samples from a common distribution, the local empirical loss functions $f_i(w) = (1/n)\sum_{j=1}^{n}\phi(w, z_{i,j})$ will be similar to each other if the local sample size n is large. Under this assumption, Zhang et al. [52] 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 is communication efficient because it achieves optimal dependence on the overall sample size mn (in the sense of statistical lower bounds). But their conclusion doesn't allow the regularization parameter λ to decrease to zero as n goes to infinity (see discussions in [47]).

Exploiting the stochastic nature alone seems not enough to overcome ill-conditioning in the regime of first-order methods. This motivates the development of distributed second-order methods. Recently, Shamir et al. [47] proposed a distributed approximate Newton-type (DANE) method. Their method takes advantage of the fact that, under the stochastic assumptions of SAA, the Hessians $f_1'', f_2'', \ldots, f_m''$ are similar to each other. For quadratic loss functions, DANE is shown to converge in $\widetilde{\mathcal{O}}\big((L/\lambda)^2 n^{-1}\log(1/\epsilon)\big)$ iterations with high probability, where the notation $\widetilde{\mathcal{O}}(\cdot)$ hides additional logarithmic factors involving m and d. If $\lambda \sim 1/\sqrt{mn}$ as in machine learning applications, then the iteration complexity becomes $\widetilde{\mathcal{O}}(m\log(1/\epsilon))$, which scales linearly with the number of machines m, not the total sample size mn. However, the analysis in [47] does not guarantee that DANE has the same convergence rate on non-quadratic functions.

1.2 Outline of our approach

In this paper, we propose a communication-efficient distributed second-order method for minimizing the overall empirical loss $f(w)$ defined in (2). Our method is based on an *inexact* damped Newton method. Assume $f(w)$ is strongly convex and has continuous second derivatives. In the *exact* damped Newton method (e.g., [34, Section 4.1.5]), we first choose an initial point $w_0 \in \mathbb{R}^d$, and then repeat

$$w_{k+1} = w_k - \frac{1}{1+\delta(w_k)}\Delta w_k, \qquad k = 0, 1, 2, \ldots \tag{4}$$

where Δw_k and $\delta(w_k)$ are the Newton step and the Newton decrement, respectively, defined as

$$\begin{aligned} \Delta w_k &= [f''(w_k)]^{-1} f'(w_k)\ , \\ \delta(w_k) &= \sqrt{f'(w_k)^T [f''(w_k)]^{-1} f'(w_k)} = \sqrt{(\Delta w_k)^T f''(w_k) \Delta w_k}\ . \end{aligned} \tag{5}$$

Since f is the average of $f_1, \ldots, f_m$, its gradient and Hessian can be written as

$$f'(w_k) = \frac{1}{m}\sum_{i=1}^{m} f_i'(w_k), \qquad f''(w_k) = \frac{1}{m}\sum_{i=1}^{m} f_i''(w_k) \tag{6}$$

In order to compute Δw_k in a distributed setting, the naive approach would require all the machines to send their gradients and Hessians to a master node (say machine 1). However, the task of transmitting the Hessians (which are $d \times d$ matrices) can be prohibitive for large dimensions d. A better alternative is to use the conjugate gradient (CG) method to compute Δw_k as the solution to a linear system $f''(w_k)\Delta w_k = f'(w_k)$. Each iteration of the CG method requires a matrix-vector product of the form

$$f''(w_k)v = \frac{1}{m}\sum_{i=1}^{m} f_i''(w_k)v$$

where v is some vector in $\mathbb{R}^d$. More specifically, the master node can broadcast the vector v to each machine, each machine computes $f_i''(w_k)v \in \mathbb{R}^d$ locally and sends it back to the master node, which then forms the average $f''(w_k)v$ and performs the CG update. Due to the iterative nature of the CG method, we can only compute the Newton direction and Newton decrement approximately, especially with limited number of communication rounds.

The overall method has two levels of loops: the outer-loop of the damped Newton method, and the inner loop of the CG method for computing the inexact Newton steps. A similar approach (using a distributed truncated Newton method) was proposed in [54, 26] for ERM of linear predictors, and it was reported to perform very well in practice. However, the total number of CG iterations (each takes a round of communication) may still be high.

First, consider the outer loop complexity. It is well-known that Newton-type methods have asymptotic superlinear convergence. However, in classical analysis of Newton's method (e.g., [9, Section 9.5.3]), the number of steps needed to reach the superlinear convergence zone still depends on the condition number; more specifically, it scales quadratically in κ. To solve this problem, we resort to the machinery of self-concordant functions [36, 34]. 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.

Second, consider the inner loop complexity. The convergence rate of the CG method also depends on the condition number κ: it takes $\mathcal{O}(\sqrt{\kappa}\log(1/\varepsilon))$ CG iterations to compute an ε-precise Newton step. Thus we arrive at the dilemma that the overall complexity of the CG-powered inexact Newton method is no better than accelerated gradient methods or ADMM. To overcome this difficulty, we exploit the stochastic nature of the problem and propose to use a preconditioned CG (PCG) method for solving the Newton system. Roughly speaking, if the local Hessians $f_1''(w_k), \ldots, f_m''(w_k)$ are "similar" to each other, then we can use any local Hessian $f_i''(w_k)$ as a preconditioner. Without loss of generality, let $P = f_1''(w_k) + \mu I$, where μ is an estimate of the spectral norm $\|f_1''(w_k) - f''(w_k)\|_2$. Then we use CG to solve the pre-conditioned linear system

$$P^{-1}f''(w_k)\Delta w_k = P^{-1}f'(w_k)$$

where the preconditioning (multiplication by P^{-1}) can be computed locally at machine 1 (the master node). The convergence rate of PCG depends on the condition number of the matrix $P^{-1}f''(w_k)$, which is close to 1 if the spectral norm $\|f_1''(w_k) - f''(w_k)\|_2$ is small.

To exactly characterize the similarity between $f_1''(w_k)$ and $f''(w_k)$, we rely on stochastic analysis in the framework of SAA or ERM. We show that with high probability, $\|f_1''(w_k) - f''(w_k)\|_2$ decreases as $\widetilde{\mathcal{O}}(\sqrt{d/n})$ in general, and $\widetilde{\mathcal{O}}(\sqrt{1/n})$ for quadratic loss. Therefore, when n is large, the preconditioning is very effective and the PCG method converges to sufficient precision within a small number of iterations. The stochastic assumption is also critical for obtaining an initial point w_0 which further brings down the overall iteration complexity.

Combining the above ideas, we propose and analyze an algorithm for Distributed Self-Concordant Optimization (DiSCO, which also stands for Distributed SeCond-Order method, or Distributed

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 [47]	$m\log(1/\epsilon)$	$(mn)^{1/2}\log(1/\epsilon)$
DiSCO (this paper)	$m^{1/4}\log(1/\epsilon)$	$m^{3/4}d^{1/4} + m^{1/4}d^{1/4}\log(1/\epsilon)$

Table 1. Communication efficiency of several distributed algorithms for ERM of linear predictors, when the regularization parameter λ in (3) is 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). For DiSCO, the dependence on ϵ can be improved to $\log\log(1/\epsilon)$ with superlinear convergence.

Stochastic Convex Optimization). We show that several popular empirical loss functions in machine learning, including ridge regression, regularized logistic regression and a (new) smoothed hinge loss, are actually self-concordant. For ERM with these loss functions, Table 1 lists the number of communication rounds required by DiSCO and several other algorithms to find an ϵ-optimal solution. As the table shows, 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 [47], DiSCO not only improves the communication efficiency on quadratic loss, but also handles non-quadratic classification tasks.

Contributions

- We introduce a communication-efficient distributed second-order method that significantly reduces communication rounds through the use of inexact Newton steps and preconditioned conjugate gradient techniques.
- We provide theoretical analysis demonstrating the algorithm's iteration complexity and communication efficiency under stochastic assumptions.
- The algorithm is shown to be well-suited for regularized empirical risk minimization tasks such as linear regression and binary classification, where we establish improvements over existing methods in communication efficiency.

Organization The rest of this paper is organized as follows. In Section 2, we review the definition of self-concordant functions, and show that several popular empirical loss functions used in machine learning are either self-concordant or can be well approximated by self-concordant functions. In Section 3, we analyze the iteration complexity of an inexact damped Newton method for minimizing self-concordant functions. In Section 4, we show how to compute the inexact Newton step using a distributed PCG method, describe the overall DiSCO algorithm, and discuss its communication complexity. In Section 5, we present our main theoretical results based on stochastic analysis, and apply them to linear regression and classification. Finally, we discuss the extension of DiSCO to distributed minimization of composite loss functions in Section 6, and conclude the paper in Section 7.

2 Self-concordant empirical loss

The theory of self-concordant functions were developed by Nesterov and Nemirovski for the analysis of interior-point methods [36]. 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

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

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

The reader may refer to the books [36, 34] for detailed treatment of self-concordance. In particular, the following lemma [34, Corollary 4.1.2] states that any self-concordant function can be rescaled to become standard self-concordant.

Lemma 1. *If a function f is self-concordant with parameter M_f, then $\frac{M_f^2}{4} f$ is standard self-concordant (with parameter 2).*

In the rest of this section, we show that several popular regularized empirical loss functions for linear regression and binary classification are either self-concordant or can be well approximated by self-concordant functions.

First we consider regularized linear regression (ridge regression) with

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

To simplify notation, here we use a single subscript i running from 1 to $N = mn$, instead of the double subscripts $\{i, j\}$ used in the introduction. Since f is a quadratic function, its third derivatives are all zero. Therefore, it is self-concordant with parameter 0, and by definition is also standard self-concordant.

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

$$\ell(w) \overset{\text{def}}{=} \frac{1}{N}\sum_{i=1}^{N}\varphi(y_i w^T x_i) + \frac{\gamma}{2}\|w\|_2^2 \tag{7}$$

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 loss function which returns 0 if $y_i = \operatorname{sign}(w^T x_i)$ and 1 otherwise. 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. Under this assumption, the following lemma shows that the regularized loss $\ell(w)$ is self-concordant.

Lemma 2. *Assume that $\gamma > 0$ and there exist $Q > 0$ and $\alpha \in [0, 1)$ such that $|\varphi'''(t)| \le Q(\varphi''(t))^{1-\alpha}$ for every $t \in \mathbb{R}$. Then:*

(a) The function $\ell(w)$ defined by equation (7) is self-concordant with parameter $\frac{B^{1+2\alpha}Q}{\gamma^{1/2+\alpha}}$.

(b) The scaled function $f(w) = \frac{B^{2+4\alpha}Q^2}{4\gamma^{1+2\alpha}}\ell(w)$ is standard self-concordant.

Proof. We need to bound the third derivative of ℓ appropriately. Using equation (7) and the assumption on φ, we have

$$\begin{aligned}
\left|u^T(\ell'''(w)[u])u\right| &\le \frac{1}{N}\sum_{i=1}^N \left|\varphi'''(y_i w^T x_i)(y_i u^T x_i)^3\right| \\
&\overset{(i)}{\le} \frac{Q}{N}\sum_{i=1}^N \left((u^T x_i)^2 \varphi''(y_i w^T x_i)\right)^{1-\alpha} (B\|u\|_2)^{1+2\alpha} \\
&\overset{(ii)}{\le} B^{1+2\alpha} Q \left(\frac{1}{N}\sum_{i=1}^N (u^T x_i)^2 \varphi''(y_i w^T x_i)\right)^{1-\alpha} (\|u\|_2)^{1+2\alpha} \\
&\overset{(iii)}{\le} B^{1+2\alpha} Q \left(u^T \ell''(w) u\right)^{1-\alpha} (\|u\|_2)^{1+2\alpha}
\end{aligned}$$

In the above derivation, inequality (i) uses the property that $|y_i| = 1$ and $|u^T x_i| \le B\|u\|_2$, inequality (ii) uses Hölder's inequality and concavity of $(\cdot)^{1-\alpha}$, and inequality (iii) uses the fact that the additional regularization term in $\ell(w)$ is convex.

Since ℓ is γ-strongly convex, we have $u^T \ell''(w) u \ge \gamma \|u\|_2^2$. Thus, we can upper bound $\|u\|_2$ by $\|u\|_2 \le \gamma^{-1/2}(u^T \ell''(w) u)^{1/2}$. Substituting this inequality into the above upper bound completes the proof of part (a). Given part (a), part (b) follows immediately from Lemma 1. □

It is important to note that the self-concordance of ℓ essentially relies on the regularization parameter γ being positive. If $\gamma = 0$, then the function will no longer be self-concordant, as pointed out by Bach [3] on logistic regression. Since we have the freedom to choose φ, Lemma 2 handles a broad class of empirical loss functions. Next, we take the logistic loss and a smoothed hinge loss as two concrete examples.

Logistic regression For logistic regression, we minimize the objective function (7) where φ is the logistic loss: $\varphi(t) = \log(1 + e^{-t})$. We can calculate the second and the third derivatives of $\varphi(t)$:

$$\begin{aligned}
\varphi''(t) &= \frac{e^t}{(e^t+1)^2}\,, \\
\varphi'''(t) &= \frac{e^t(1-e^t)}{(e^t+1)^3} = \frac{1-e^t}{1+e^t}\varphi''(t)
\end{aligned}$$

Since $|\frac{1-e^t}{1+e^t}| \le 1$ for all $t \in \mathbb{R}$, we conclude that $|\varphi'''(t)| \le \varphi''(t)$ for all $t \in \mathbb{R}$. This implies that the condition in Lemma 2 holds with $Q = 1$ and $\alpha = 0$. Therefore, the regularized empirical loss $\ell(w)$ is self-concordant with parameter $B/\sqrt{\gamma}$, and the scaled loss function $f(w) = (B^2/(4\gamma))\ell(w)$ 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

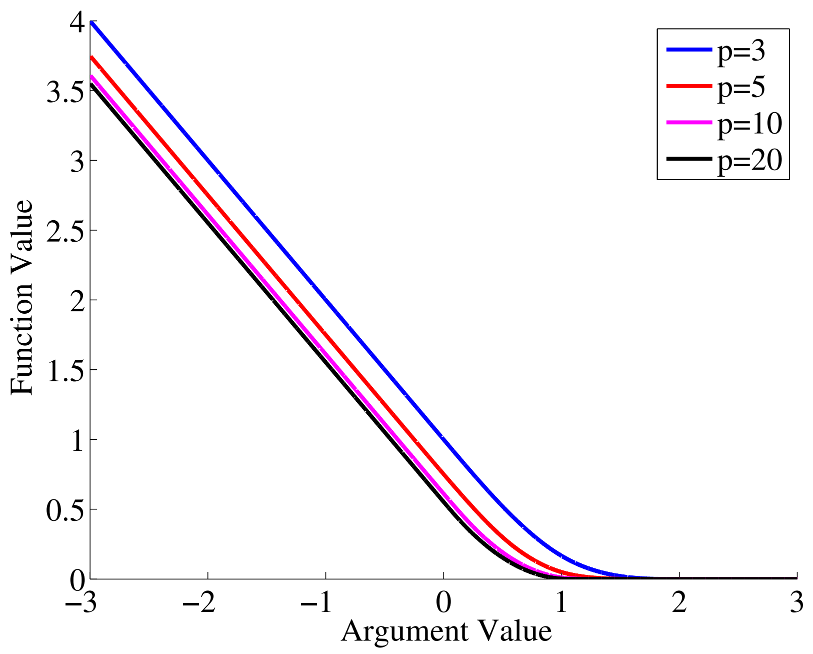

Figure 1: Smoothed hinge loss φ_p with $p = 3, 5, 10, 20$.

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

$$\varphi_p(t) = \begin{cases} \frac{3}{2} - \frac{p-2}{p-1} - t & \text{for } t < -\frac{p-3}{p-1}, \\ \frac{3}{2} - \frac{p-2}{p-1} - t + \frac{(t+(p-3)/(p-1))^p}{p(p-1)} & \text{for } -\frac{p-3}{p-1} \leq t < 1 - \frac{p-3}{p-1}, \\ \frac{p+1}{p(p-1)} - \frac{t}{p-1} + \frac{1}{2}(1-t)^2 & \text{for } 1 - \frac{p-3}{p-1} \leq t < 1, \\ \frac{(2-t)^p}{p(p-1)} & \text{for } 1 \leq t < 2, \\ 0 & \text{for } t \geq 2. \end{cases} \tag{8}$$

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 smoothed hinge loss φ_p satisfies the condition of Lemma 2 with $Q = p - 2$ and $\alpha = \frac{1}{p-2}$. To see this, we note that the third derivative of $\varphi_p(t)$ is nonzero only when $t \in [-\frac{p-3}{p-1}, 1 - \frac{p-3}{p-1}]$ and when $t \in [1, 2]$. On the first interval, we have

$$\varphi_p''(t) = \left(t + \frac{p-3}{p-1}\right)^{p-2}, \qquad \varphi_p'''(t) = (p-2)\left(t + \frac{p-3}{p-1}\right)^{p-3}$$

On the second interval, we have

$$\varphi_p''(t) = (2-t)^{p-2}, \qquad \varphi_p'''(t) = -(p-2)\,(2-t)^{p-3}$$

For both cases we have the inequality

$$|\varphi_p'''(t)| \leq (p-2)(\varphi_p''(t))^{1-\frac{1}{p-2}}$$

```
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 a vector $v_k$ such that $\|f''(w_k)v_k - f'(w_k)\|_2 \le \epsilon_k$.
    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.
```

which means $Q = p - 2$ and $\alpha = \frac{1}{p-2}$. Therefore, according to Lemma 2, the regularized empirical loss $\ell(w)$ is self-concordant with parameter

$$M_p = \frac{(p-2)B^{1+\frac{2}{p-2}}}{\gamma^{\frac{1}{2}+\frac{1}{p-2}}} \tag{9}$$

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 1 holds. Our method is described in Algorithm 1. If we let $\epsilon_k = 0$ for all $k \ge 0$, then $v_k = [f''(w_k)]^{-1} f'(w_k)$ is the exact Newton step and δ_k is the Newton decrement defined in (5), so the algorithm reduces to the exact damped Newton method given in (4). But here we allow the computation of the Newton step (hence also the Newton decrement) to be inexact and contain approximation errors.

The explicit account of approximation errors is essential for distributed optimization. In particular, if $f(w) = (1/m)\sum_{i=1}^m f_i(w)$ and the components f_i locate on separate machines, then we can only perform Newton updates approximately with limited communication budget. Even in a centralized setting on a single machine, analysis of approximation errors can be important if the Newton system is solved by iterative algorithms such as the conjugate gradient method.

Before presenting the convergence analysis, we need to introduce two auxiliary functions

$$\begin{aligned} \omega(t) &= t - \log(1+t), \qquad t \ge 0, \\ \omega_*(t) &= -t - \log(1-t), \qquad 0 \le t < 1 \end{aligned}$$

These two functions are very useful for characterizing the properties of self-concordant functions; see [34, Section 4.1.4] for a detailed account. Here, we simply note that $\omega(0) = \omega_*(0) = 0$, both are strictly increasing for $t \ge 0$, and $\omega_*(t) \to \infty$ as $t \to 1$.

We also need to define two auxiliary vectors

$$\begin{aligned} \widetilde{u}_k &= [f''(w_k)]^{-1/2} f'(w_k), \\ \widetilde{v}_k &= [f''(w_k)]^{1/2} v_k \end{aligned}$$

The norm of the first vector, $\|\widetilde{u}_k\|_2 = \sqrt{f'(w_k)^T [f''(w_k)]^{-1} f'(w_k)}$, is the exact Newton decrement. The norm of the second one is $\|\widetilde{v}_k\|_2 = \delta_k$, which is computed during each iteration of Algorithm 1.

Note that we do *not* compute $\widetilde{u}_k$ or $\widetilde{v}_k$ in Algorithm 1. They are introduced solely for the purpose of convergence analysis. The following Theorem is proved in Appendix A.

Theorem 1. *Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is a standard self-concordant function and Assumption 1 holds. If we choose the sequence $\{\epsilon_k\}_{k\geq 0}$ in Algorithm 1 as*

$$\epsilon_k = \beta(\lambda/L)^{1/2}\|f'(w_k)\|_2 \quad \textit{with } \beta = 1/20 \tag{10}$$

then:

(a) For any $k \geq 0$, we have $f(w_{k+1}) \leq f(w_k) - \frac{1}{2}\omega(\|\widetilde{u}_k\|_2)$.

(b) If $\|\widetilde{u}_k\|_2 \leq 1/6$, then we have $\omega(\|\widetilde{u}_{k+1}\|_2) \leq \frac{1}{2}\omega(\|\widetilde{u}_k\|_2)$.

As mentioned before, when $\epsilon_k = 0$, the vector $v_k = [f''(w_k)]^{-1}f'(w_k)$ becomes the exact Newton step. In this case, we have $\widetilde{v}_k = \widetilde{u}_k$, and it can be shown that $f(w_{k+1}) \leq f(w_k) - \omega(\|\widetilde{u}_k\|_2)$ for all $k \geq 0$ and the exact damped Newton method has quadratic convergence when $\|\widetilde{u}_k\|_2$ is small (see [34, Section 4.1.5]). With the approximation error ϵ_k specified in (10), we have

$$\begin{aligned}\|\widetilde{v}_k - \widetilde{u}_k\|_2 &\leq \|(f''(w_k))^{-1/2}\|_2\|f''(w_k)v_k - f'(w_k)\|_2 \leq \lambda^{-1/2}\epsilon_k \\ &= \beta L^{-1/2}\|f'(w_k)\|_2 \leq \beta\|\widetilde{u}_k\|_2\end{aligned}$$

which implies

$$(1-\beta)\|\widetilde{u}_k\|_2 \leq \|\widetilde{v}_k\|_2 \leq (1+\beta)\|\widetilde{u}_k\|_2 \tag{11}$$

Appendix A shows that when β is sufficiently small, the above inequality leads to the conclusion in part (a). Compared with the exact damped Newton method, the guaranteed reduction of the objective value per iteration is cut by half.

Part (b) of Theorem 1 suggests a linear rate of convergence when $\|\widetilde{u}_k\|_2$ is small. 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, superlinear convergence can be established if we set the tolerances ϵ_k to be small enough; see Appendix B for detailed analysis. However, when v_k is computed through a distributed iterative algorithm (like the distributed PCG algorithm in Section 4.2), a smaller ϵ_k would require more local computational effort and more rounds of inter-machine communication. The choice in equation (10) is a reasonable trade-off in practice.

Using Theorem 1, we can derive the iteration complexity of Algorithm 1 for obtaining an arbitrary accuracy. We present this result as a corollary.

Corollary 1. *Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is a standard self-concordant function and Assumption 1 holds. If we choose the sequence $\{\epsilon_k\}$ in Algorithm 1 as in* (10)*, then for any $\epsilon > 0$, we have $f(w_k) - f(w_\star) \leq \epsilon$ whenever $k \geq K$ where*

$$K = \left\lceil \frac{f(w_0) - f(w_\star)}{\frac{1}{2}\omega(1/6)} \right\rceil + \left\lceil \log_2\left(\frac{2\,\omega(1/6)}{\epsilon}\right) \right\rceil \tag{12}$$

Here $\lceil t \rceil$ denotes the smallest nonnegative *integer that is larger or equal to t.*

Proof. Since $\omega(t)$ is strictly increasing for $t \geq 0$, part (a) of Theorem 1 implies that if $\|\widetilde{u}_k\|_2 > 1/6$, one step of Algorithm 1 decreases the value of $f(w)$ by at least a constant $\frac{1}{2}\omega(1/6)$. So within at most $K_1 = \lceil \frac{f(w_0)-f(w_\star)}{\frac{1}{2}\omega(1/6)} \rceil$ iterations, we are guaranteed that $\|\widetilde{u}_k\|_2 \leq 1/6$.

According to [34, Theorem 4.1.13], if $\|\widetilde{u}_k\|_2 < 1$, then we have

$$\omega(\|\widetilde{u}_k\|_2) \leq f(w_k) - f(w_\star) \leq \omega_*(\|\widetilde{u}_k\|_2) \tag{13}$$

Moreover, it is easy to check that $\omega_*(t) \leq 2\,\omega(t)$ for $0 \leq t \leq 1/6$. Therefore, using part (b) of Theorem 1, we conclude that when $k \geq K_1$,

$$f(w_k) - f(w_\star) \;\leq\; 2\,\omega(\|\widetilde{u}_k\|_2) \;\leq\; 2(1/2)^{k-K_1}\omega(\|\widetilde{u}_{K_1}\|_2) \;\leq\; 2(1/2)^{k-K_1}\omega(1/6)$$

Bounding the right-hand side of the above inequality by ϵ, we have $f(w_k) - f(w_\star) \leq \epsilon$ whenever $k \geq K_1 + \left\lceil \log_2\left(\frac{2\,\omega(1/6)}{\epsilon}\right)\right\rceil = K$, which is the desired result.

We note that when $\|\widetilde{u}_k\|_2 \leq 1/6$ (as long as $k \geq K_1$), we have $f(w_k) - f(w_\star) \leq 2\,\omega(1/6)$. Thus for $\epsilon > 2\,\omega(1/6)$, it suffices to have $k \geq K_1$. □

3.1 Stopping criteria

We discuss two stopping criteria for Algorithm 1. The first one is based on the strong convexity of f, which leads to the inequality (e.g., [34, Theorem 2.1.10])

$$f(w_k) - f(w_\star) \leq \frac{1}{2\lambda}\|f'(w_k)\|_2^2$$

Therefore, we can use the stopping criterion $\|f'(w_k)\|_2 \leq \sqrt{2\lambda\epsilon}$, which implies $f(w_k) - f(w_\star) \leq \epsilon$. However, this choice can be too conservative in practice (see discussions in [9, Section 9.1.2]).

Another choice for the stopping criterion is based on self-concordance. Using the fact that $\omega_*(t) \leq t^2$ for $0 \leq t \leq 0.68$ (see [9, Section 9.6.3]), we have

$$f(w_k) - f(w_\star) \leq \omega_*(\|\widetilde{u}_k\|_2) \leq \|\widetilde{u}_k\|_2^2 \tag{14}$$

provided $\|\widetilde{u}_k\|_2 \leq 0.68$. Since we do not compute $\|\widetilde{u}_k\|_2$ (the exact Newton decrement) directly in Algorithm 1, we can use δ_k as an approximation. Using the inequality (11), and noticing that $\|\widetilde{v}_k\|_2 = \delta_k$, we conclude that

$$\delta_k \leq (1-\beta)\sqrt{\epsilon}$$

implies $f(w_k) - f(w_\star) \leq \epsilon$ when $\epsilon \leq 0.68^2$. Since δ_k is computed at each iteration of Algorithm 1, this can serve as a good stopping criterion.

3.2 Scaling for non-standard self-concordant functions

In many applications, we need to deal with empirical loss functions that are not standard self-concordant; see the examples in Section 2. Suppose a regularized loss function $\ell(w)$ is self-concordant with parameter $M_\ell > 2$. By Lemma 1, the scaled function $f = \eta\ell$ with $\eta = M_\ell^2/4$ is standard self-concordant. We can apply Algorithm 1 to minimize the scaled function f, and rewrite it in terms of the function ℓ and the scaling constant η.

Using the sequence $\{\epsilon_k\}$ defined in (10), the condition for computing v_k in Step 1 is

$$\|f''(w_k)v_k - f'(w_k)\|_2 \leq \beta(\lambda/L)^{1/2}\|f'(w_k)\|_2$$

Let λ_ℓ and L_ℓ be the strong convexity and smoothness parameters of the function ℓ. With the scaling, we have $\lambda = \eta\lambda_\ell$ and $L = \eta L_\ell$, thus their ratio (the condition number) does not change. Therefore the above condition is equivalent to

$$\|\ell''(w_k)v_k - \ell'(w_k)\|_2 \le \beta(\lambda_\ell/L_\ell)^{1/2}\|\ell'(w_k)\|_2 \tag{15}$$

In other words, the precision requirement in Step 1 is *scaling invariant*.

Step 2 of Algorithm 1 can be rewritten as

$$w_{k+1} = w_k - \frac{v_k}{1 + \sqrt{\eta}\cdot\sqrt{v_k^T \ell''(w_k)v_k}} \tag{16}$$

Here, the factor η explicitly appears in the formula. By choosing a larger scaling factor η, the algorithm chooses a smaller stepsize. This adjustment is intuitive because the convergence of Newton-type method relies on local smoothness conditions. By multiplying a large constant to ℓ, the function's Hessian becomes less smooth, so that the stepsize should shrink.

In terms of complexity analysis, if we target to obtain $\ell(w_k) - \ell(w_\star) \le \epsilon$, then the iteration bound in (12) becomes

$$\left\lceil \frac{\eta\big(\ell(w_0) - \ell(w_\star)\big)}{\frac{1}{2}\omega(1/6)} \right\rceil + \left\lceil \log_2\Big(\frac{2\,\omega(1/6)}{\eta\epsilon}\Big) \right\rceil \tag{17}$$

For ERM problems in supervised learning, the self-concordant parameter M_ℓ, and hence the scaling factor $\eta = M_\ell^2/4$, can grow with the number of samples. For example, the regularization parameter γ in (7) often scales as $1/\sqrt{N}$ where $N = mn$ is the total number of samples. Lemma 2 suggests that η grows on the order of $\sqrt{mn}$. A larger η will render the second term in (17) less relevant, but the first term grows with the sample size mn. In order to counter the effect of the growing scaling factor, we need to choose the initial point w_0 judiciously to guarantee a small initial gap. This will be explained further in the next sections.

4 The DiSCO algorithm

In this section, we adapt the inexact damped Newton method (Algorithm 1) to a distributed system, in order to minimize

$$f(w) = \frac{1}{m}\sum_{i=1}^{m} f_i(w) \tag{18}$$

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. After answering these two questions, we will present the overall DiSCO algorithm and analyze its communication complexity.

4.1 Initialization

In accordance with the averaging structure in (18), we choose the initial point based on averaging. More specifically, we let

$$w_0 = \frac{1}{m}\sum_{i=1}^{m} \widehat{w}_i \tag{19}$$

where each $\widehat{w}_i$ is the solution to a local optimization problem at machine i:

$$\widehat{w}_i = \arg\min_{w\in\mathbb{R}^d} \left\{ f_i(w) + \frac{\rho}{2}\|w\|_2^2 \right\}, \qquad i = 1, \ldots, m \tag{20}$$

Here $\rho \geq 0$ is a regularization parameter, which we will discuss in detail in the context of stochastic analysis in Section 5. Roughly speaking, if each f_i is constructed with n i.i.d. samples as in (3), then we can choose $\rho \sim 1/\sqrt{n}$ to make $\mathbb{E}[f(w_0) - f(w_\star)]$ decreasing as $\mathcal{O}(1/\sqrt{n})$. In this section, we simply regard it as an input parameter.

Here we comment on the computational cost of solving (20) locally at each machine. Suppose each $f_i(w)$ has the form in (3), then the local optimization problems in (20) become

$$\widehat{w}_i = \arg\min_{w\in\mathbb{R}^d} \left\{ \frac{1}{n}\sum_{j=1}^{n} \phi(w, z_{i,j}) + \frac{\lambda+\rho}{2}\|w\|_2^2 \right\}, \qquad i = 1, \ldots, m \tag{21}$$

The finite average structure of the above objective function can be effectively exploited by the stochastic average gradient (SAG) method [40, 41] or its new variant SAGA [14]. Each step of these methods processes only one component function $\phi(w, z_{i,j})$, picked uniformly at random. Suppose $f_i(w)$ is L-smooth, then SAG returns an ϵ-optimal solution with $\mathcal{O}\big((n + \frac{L+\rho}{\lambda+\rho})\log(1/\epsilon)\big)$ steps of stochastic updates. For ERM of linear predictors, we can also use the stochastic dual coordinate ascent (SDCA) method [44], which has the same complexity. We also mention some recent progress in accelerated stochastic coordinate gradient methods [43, 27, 53], which can be more efficient both in theory and practice.

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. When the objective f has the averaging form (18), its Hessian and gradient are given in (6). In the setting of distributed optimization, we propose to use a preconditioned conjugate gradient (PCG) method to solve the Newton system.

To simplify notation, we use H to represent $f''(w_k)$ and use H_i to represent $f_i''(w_k)$. Without loss of generality, we define a preconditioning matrix using the local Hessian at the first machine (the master node):

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

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 particular, the master machine carries out the main steps of the classical PCG algorithm (e.g., [20, Section 10.3]), and all machines (including the master) 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

$$Hu^{(t)} = \frac{1}{m}\sum_{i=1}^{m} f_i''(w_k)u^{(t)}, \qquad Hv^{(t)} = \frac{1}{m}\sum_{i=1}^{m} f_i''(w_k)v^{(t)}$$

```
Algorithm 2: Distributed PCG algorithm (given $w_k$ and $\mu$, compute $v_k$ and $\delta_k$)
master machine ($i = 1$)              machines $i = 1, \dots, m$ 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:;
    broadcasts $w_k$ to other machines;        $\longrightarrow$        compute $f_i'(w_k)$
    aggregate $f_i'(w_k)$ to form $f'(w_k)$.   $\longleftarrow$
initialization: compute $\epsilon_k$ given in (10) and set;
    $v^{(0)} = 0$,            $s^{(0)} = P^{-1} r^{(0)}$,
    $r^{(0)} = f'(w_k)$,      $u^{(0)} = s^{(0)}$.;
repeat for $t = 0, 1, 2 \dots$,
    1. communication:;
           broadcast $u^{(t)}$ and $v^{(t)}$;               $\longrightarrow$    compute $f_i''(w_k) u^{(t)}$
           aggregate to form $Hu^{(t)}$ and $Hv^{(t)}$.     $\longleftarrow$     compute $f_i''(w_k) v^{(t)}$
    2. compute $\alpha_t = \frac{\langle r^{(t)}, s^{(t)} \rangle}{\langle u^{(t)}, Hu^{(t)} \rangle}$ and update
           $v^{(t+1)} = v^{(t)} + \alpha_t u^{(t)}$,
           $r^{(t+1)} = r^{(t)} - \alpha_t H u^{(t)}$.
    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 $\delta_k = \sqrt{v_k^T H v^{(t)} + \alpha^{(t)} v_k^T H u^{(t)}}$. ;
```

We note that the overall Hessian $H = f''(w_k)$ is never formed and the master machine only stores and updates the vectors $Hu^{(t)}$ and $Hv^{(t)}$.

As explained in Section 1.2, the motivation for preconditioning is that when H_1 is sufficiently close to H, the condition number of $P^{-1}H$ might be close to 1, which is much smaller than that of H itself. As a result, the PCG method may converge much faster than CG without preconditioning. The following lemma characterizes the extreme eigenvalues of $P^{-1}H$ based on the closeness between H_1 and H.

Lemma 3. *Suppose Assumption 1 holds. If $\|H_1 - H\|_2 \leq \mu$, then we have*

$$\sigma_{\max}(P^{-1}H) \leq 1, \tag{22}$$

$$\sigma_{\min}(P^{-1}H) \geq \frac{\lambda}{\lambda + 2\mu} \tag{23}$$

Here $\|\cdot\|_2$ denote the spectral norm of a matrix, and $\sigma_{\max}(\cdot)$ and $\sigma_{\min}(\cdot)$ denote the largest and smallest eigenvalues of a diagonalizable matrix, respectively.

Proof. Since both P and H are symmetric and positive definite, all eigenvalues of $P^{-1}H$ are positive real numbers (e.g., [21, Section 7.6]). The eigenvalues of $P^{-1}H$ are identical to that of

$P^{-1/2}HP^{-1/2}$. Thus, it suffices to prove inequalities (22) and (23) for the matrix $P^{-1/2}HP^{-1/2}$. To prove inequality (22), we need to show that $H \preceq P = H_1 + \mu I$. This is equivalent to $H - H_1 \preceq \mu I$, which is a direct consequence of the assumption $\|H_1 - H\|_2 \leq \mu I$.

Similarly, the second inequality (23) is equivalent to $H \succeq \frac{\lambda}{\lambda+2\mu}(H_1 + \mu I)$, which is the same as $\frac{2\mu}{\lambda}H - \mu I \succeq H_1 - H$. Since $H \succeq \lambda I$ (by Assumption 1), we have $\frac{2\mu}{\lambda}H - \mu I \succeq \mu I$. The additional assumption $\|H_1 - H\|_2 \leq \mu I$ implies $\mu I \succeq H_1 - H$, which complete the proof. $\square$

By Assumption 1, the condition number of the Hessian matrix is $\kappa(H) = L/\lambda$, which can be very large if λ is small. Lemma 3 establishes that the condition number of the preconditioned linear system is

$$\kappa(P^{-1}H) = \frac{\sigma_{\max}(P^{-1}H)}{\sigma_{\min}(P^{-1}H)} = 1 + \frac{2\mu}{\lambda} \tag{24}$$

provided that $\|H_1 - H\|_2 \leq \mu$. When μ is small (comparable with λ), the condition number $\kappa(P^{-1}H)$ is close to one and can be much smaller than $\kappa(H)$. Based on classical convergence analysis of the CG method (e.g., [29, 2]), the following lemma shows that Algorithm 2 terminates in $\mathcal{O}(\sqrt{1 + \mu/\lambda})$ iterations. See Appendix C for the proof.

Lemma 4. *Suppose Assumption 1 holds and assume that $\|H_1 - H\|_2 \leq \mu$. Let*

$$T_\mu = \left\lceil \sqrt{1 + \frac{2\mu}{\lambda}} \log\left(\frac{2\sqrt{L/\lambda}\|f'(w_k)\|_2}{\epsilon_k}\right) \right\rceil$$

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

When the tolerance ϵ_k is chosen as in (10), the iteration bound T_μ is independent of $f'(w_k)$, i.e.,

$$T_\mu = \left\lceil \sqrt{1 + \frac{2\mu}{\lambda}} \log\left(\frac{2L}{\beta\lambda}\right) \right\rceil \tag{25}$$

Under Assumption 1, we always have $\|H_1 - H\|_2 \leq L$. If we choose $\mu = L$, then Lemma 4 implies that Algorithm 2 terminates in $\widetilde{\mathcal{O}}(\sqrt{L/\lambda})$ iterations. where the notation $\widetilde{\mathcal{O}}(\cdot)$ hides logarithmic factors. 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 terminating in $\widetilde{\mathcal{O}}(\sqrt{\mu/\lambda})$ iterations. In Section 5, we show that if the local empirical losses f_i are constructed with n i.i.d. samples from the same distribution, then $\|H_1 - H\|_2 \sim 1/\sqrt{n}$ with high probability. As a consequence, the iteration complexity of Algorithm 2 is upper bounded by $\widetilde{\mathcal{O}}(1 + \lambda^{-1/2}n^{-1/4})$.

We wrap up this section by discussing the computation and communication complexities of Algorithm 2. The bulk of computation is at the master machine, especially computing the vector $s^{(t)} = P^{-1}r^{(t)}$ in Step 3, which is equivalent to minimize the quadratic function $(1/2)s^TPs - s^Tr^{(t)}$. Using $P = f_1''(w_k) + \mu I$ and the form of $f_1(w)$ in (3), this is equivalent to

$$s^{(t)} = \arg\min_{s \in \mathbb{R}^d} \left\{ \frac{1}{n} \sum_{j=1}^{n} \frac{s^T \phi''(w_k, z_{i,j}) s}{2} + \langle r^{(t)}, s \rangle + \frac{\lambda + \mu}{2} \|s\|_2^2 \right\} \tag{26}$$

This problem has the same structure as (21), and an ϵ-optimal solution can be obtained with $\mathcal{O}\big((n + \frac{L+\mu}{\lambda+\mu}) \log(1/\epsilon)\big)$ stochastic-gradient type of steps (see discussions at the end of Section 4.1).

As for the communication complexity, we need one round of communication at the beginning of Algorithm 2 to compute $f'(w_k)$. Then, each iteration takes one round of communication to compute $Hu^{(t)}$ and $Hv^{(t)}$. Thus, the total rounds of communication is bounded by $T_\mu + 1$.

```
Algorithm 3: DiSCO
    input: parameters $\rho, \mu \geq 0$ and precision $\epsilon > 0$.
    initialize: compute $w_0$ according to (19) and (20).
    repeat for $k = 0, 1, 2, \ldots$
        1. Run Algorithm 2: given $w_k$ and $\mu$, compute $v_k$ and $\delta_k$.
        2. Update $w_{k+1} = w_k - \frac{1}{1+\delta_k} v_k$.
    until $\delta_k \leq (1-\beta)\sqrt{\epsilon}$.
    output: $\widehat{w} = w_{k+1}$.
```

4.3 Communication efficiency of DiSCO

Putting everything together, we present the DiSCO algorithm in Algorithm 3. Here we study its communication efficiency. Recall that by one round of communication, the master machine broadcasts a message of $\mathcal{O}(d)$ bits to all machines, and every machine processes the aggregated message and sends a message of $\mathcal{O}(d)$ bits back to the master. The following proposition gives an upper bound on the number of communication rounds taken by the DiSCO algorithm.

Theorem 2. *Assume that f is a standard self-concordant function and it satisfies Assumption 1. Suppose the input parameter μ in Algorithm 3 is an upper bound on $\|f_1''(w_k) - f''(w_k)\|_2$ for all $k \geq 0$. Then for any $\epsilon > 0$, in order to find a solution $\widehat{w}$ satisfying $f(\widehat{w}) - f(w_\star) < \epsilon$, the total number of communication rounds T is bounded by*

$$T \leq 1 + \left(\left\lceil \frac{f(w_0) - f(w_\star)}{\frac{1}{2}\omega(1/6)} \right\rceil + \left\lceil \log_2\left(\frac{2\omega(1/6)}{\epsilon} \right) \right\rceil \right) \left(2 + \sqrt{1 + \frac{2\mu}{\lambda}} \log\left(\frac{2L}{\beta\lambda} \right) \right) \tag{27}$$

Ignoring logarithmic terms and universal constants, the rounds of communication T is bounded by

$$\widetilde{\mathcal{O}}\left(\left(f(w_0) - f(w_\star) + \log(1/\epsilon) \right) \sqrt{1 + 2\mu/\lambda} \right)$$

Proof. First we notice that the number of communication rounds in each call of Algorithm 2 is no more than $1 + T_\mu$, where T_μ is given in (25), and the extra 1 accounts for the communication round to form $f'(w_k)$. Corollary 1 states that in order to guarantee $f(w_k) - f(w_\star) \leq \epsilon$, the total number of calls of Algorithm 2 in DiSCO is bounded by K given in (12). Thus the total number of communication rounds is bounded by $1 + K(1 + T_\mu)$, where the extra one count is for computing the initial point w_0 defined in (19). □

It can be hard to give a good *a priori* estimate of μ that satisfies the condition in Theorem 2. In practice, we can adjust the value of μ adaptively while running the algorithm. Inspired by a line search procedure studied in [35], we propose an adaptive DiSCO method, described in Algorithm 4. The following proposition bounds the rounds of communication required by this algorithm.

Theorem 3. *Assume that f is a standard self-concordant function and it satisfies Assumption 1. Let $\mu_{\max}$ be the largest value of μ_k generated by Algorithm 4, i.e., $\mu_{\max} = \max\{\mu_0, \mu_1, \ldots, \mu_K\}$ where K is the number of outer iterations. Then for any $\epsilon > 0$, in order to find a solution $\widehat{w}$ satisfying $f(\widehat{w}) - f(w_\star) < \epsilon$, the total number of communication rounds T is bounded by*

$$T \leq 1 + \left(2\left\lceil \frac{f(w_0) - f(w_\star)}{\omega(1/6)} \right\rceil + 2\left\lceil \log_2\left(\frac{2\omega(1/6)}{\epsilon} \right) \right\rceil + \log_2\left(\frac{\mu_{\max}}{\mu_0} \right) \right) \left(2 + \sqrt{1 + \frac{2\mu_{\max}}{\lambda}} \log\left(\frac{2L}{\beta\lambda} \right) \right)$$

Algorithm 4: Adaptive DiSCO

```
input: parameters $\rho \geq 0$ and $\mu_0 > 0$, and precision $\epsilon > 0$.
initialize: compute $w_0$ according to (19) and (20).
repeat for $k = 0, 1, 2, \ldots$
    1. Run Algorithm 2 up to $T_{\mu_k}$ PCG iterations, with output $v_k$, $\delta_k$, $r_k$ and $\epsilon_k$.
    2. if $\|r_k\|_2 > \epsilon_k$ then
            set $\mu_k := 2\mu_k$ and go to Step 1;
       else
            set $\mu_{k+1} := \mu_k/2$ and go to Step 3.
    3. Update $w_{k+1} = w_k - \frac{1}{1+\delta_k} v_k$.
until $\delta_k \leq (1-\beta)\sqrt{\epsilon}$.
output: $\widehat{w} = w_{k+1}$.
```

Proof. Let n_k be the number of calls to Algorithm 2 during the kth iteration of Algorithm 4. We have

$$\mu_{k+1} = \frac{1}{2}\mu_k 2^{n_k - 1} = \mu_k 2^{n_k - 2}$$

which implies

$$n_k = 2 + \log_2 \frac{\mu_{k+1}}{\mu_k}$$

The total number of calls to Algorithm 2 is

$$N_K = \sum_{k=0}^{K-1} n_k = \sum_{k=0}^{K-1} \left(1 + \log_2 \frac{\mu_{k+1}}{\mu_k}\right) = 2K + \log_2 \frac{\mu_K}{\mu_0} \leq 2K + \log_2 \frac{\mu_{\max}}{\mu_0}$$

Since each call of Algorithm 2 involves no more than $T_{\mu_{\max}} + 1$ communication rounds, we have

$$T \leq 1 + N_K(T_{\mu_{\max}} + 1)$$

Plugging in the expression of K in (12) and $T_{\mu_{\max}}$ in (25), we obtain the desired result. $\square$

From the above proof, we see that the average number of calls to Algorithm 2 at each iteration is $2 + \frac{1}{K}\log_2\left(\frac{\mu_K}{\mu_0}\right)$, roughly twice as the non-adaptive Algorithm 3. Ignoring logarithmic terms and universal constants, the number of communication round T used by Algorithm 4 is bounded by

$$\widetilde{\mathcal{O}}\left(\left(f(w_0) - f(w_\star) + \log_2(1/\epsilon)\right)\sqrt{1 + 2\mu_{\max}/\lambda}\right)$$

In general, we can update μ_k in Algorithm 4 as follows:

$$\mu_k := \begin{cases} \theta_{\text{inc}}\mu_k & \text{if } \|r_k\|_2 > \epsilon_k, \\ \mu_k/\theta_{\text{dec}} & \text{if } \|r_k\|_2 \leq \epsilon_k, \end{cases}$$

with any $\theta_{\text{inc}} > 1$ and $\theta_{\text{dec}} \geq 1$ (see [35]). We have used $\theta_{\text{inc}} = \theta_{\text{dec}} = 2$ to simplify presentation.

4.4 A simple variant without PCG iterations

We consider a simple variant of DiSCO where the approximate Newton step v_k is computed without using the PCG method described in Algorithm 2. Instead, we simply set

$$v_k = P^{-1}f'(w_k) = (f_1''(w_k) + \mu I)^{-1}f'(w_k) \tag{28}$$

which is equivalent to setting $v_k = s^{(0)}$ in the initialization phase of Algorithm 2, or forcing it to always exit during the first PCG iteration. (The latter choice gives the same search direction but with a slightly different scaling.) In this variant, each iteration of the inexact damped Newton method requires two communication rounds: one to form $f'(w_k)$ and another to compute the stepsize parameter $\delta_k = (v_k^T f''(w_k)v_k)^{1/2}$.

A distributed algorithm that is similar to this variant of DiSCO is proposed in [31]. It does not compute δ_k; instead it uses line search to determine the step size, which also requires extra round(s) of communication. It is shown in [31] that this method works well in experiments, requiring less number of iterations to converge than ADMM. However, according to their theoretical analysis, its iteration complexity still depends badly on the condition number.

Here we examine the theoretical conditions under which this variant of DiSCO enjoys a low iteration complexity. Recall the two auxiliary vectors defined in Section 3:

$$\widetilde{u}_k = H^{-1/2}f'(w_k), \qquad \widetilde{v}_k = H^{1/2}v_k$$

The norm of their difference can be bounded as

$$\begin{aligned}\|\widetilde{v}_k - \widetilde{u}_k\|_2 &= \left\|H^{1/2}P^{-1}f'(w_k) - \widetilde{u}_k\right\|_2 = \left\|H^{1/2}P^{-1}H^{1/2}\widetilde{u}_k - \widetilde{u}_k\right\|_2 \\ &\leq \left\|I - H^{1/2}P^{-1}H^{1/2}\right\|_2 \cdot \|\widetilde{u}_k\|_2 = \left\|I - P^{-1}H\right\|_2 \cdot \|\widetilde{u}_k\|_2\end{aligned}$$

From Lemma 3, we know that when $\|H_1 - H\|_2 \leq \mu$, the eigenvalues of $P^{-1}H$ are located within the interval $[\frac{\lambda}{\lambda+2\mu}, 1]$. Therefore, we have

$$\|\widetilde{v}_k - \widetilde{u}_k\|_2 \leq \left(1 - \frac{\lambda}{\lambda + 2\mu}\right)\|\widetilde{u}_k\|_2 = \frac{2\mu}{\lambda + 2\mu}\|\widetilde{u}_k\|_2$$

The above inequality implies

$$\left(1 - \frac{2\mu}{\lambda + 2\mu}\right)\|\widetilde{u}_k\|_2 \leq \|\widetilde{v}_k\|_2 \leq \left(1 + \frac{2\mu}{\lambda + 2\mu}\right)\|\widetilde{u}_k\|_2$$

This inequality has the same form as (11), which is responsible to obtain the desired low complexity result if $\frac{2\mu}{\lambda+\mu}$ is sufficiently small. Indeed, if $\frac{2\mu}{\lambda+2\mu} \leq \beta = \frac{1}{20}$ as specified in (10), the same convergence rate and complexity result stated in Theorem 1 and Corollary 1 apply. Since each iteration of the damped Newton method involves only two communication rounds (to compute $f'(w_k)$ and δ_k respectively), we have the following corollary.

Corollary 2. *Assume that f is a standard self-concordant function and it satisfies Assumption 1. In the DiSCO algorithm, we compute the inexact Newton step using* (28). *Suppose $\frac{2\mu}{\lambda+2\mu} \leq \frac{1}{20}$ and $\|f_1''(w_k) - f''(w_k)\|_2 \leq \mu$ for all $k \geq 0$. Then for any $\epsilon > 0$, in order to find a solution $\widehat{w}$ satisfying $f(\widehat{w}) - f(w_\star) < \epsilon$, the total number of communication rounds T is bounded by*

$$T \leq 1 + 2\left(\left\lceil \frac{f(w_0) - f(w_\star)}{\frac{1}{2}\omega(1/6)} \right\rceil + \left\lceil \log_2\left(\frac{2\omega(1/6)}{\epsilon}\right) \right\rceil\right) \tag{29}$$

In Corollary 2, the requirement on μ, which upper bounds $\|f_1''(w_k) - f''(w_k)\|_2$ for all $k \geq 0$, is quite strong. In particular, it requires μ to be a small fraction of λ in order to satisfy $\frac{2\mu}{\lambda+2\mu} \leq \frac{1}{20}$. As we will see from the stochastic analysis in the next section, the spectral bound μ decreases on the order of $1/\sqrt{n}$. Therefore, in the standard setting where the regularization parameter $\lambda \sim 1/\sqrt{mn}$, the condition in Corollary 2 cannot be satisfied, and the convergence of this simple variant may be slow. In contrast, DiSCO with PCG iterations is much more tolerant of a relatively large μ, and can achieve superlinear convergence with a smaller ϵ_k.

5 Stochastic analysis

From Theorems 2 and 3 of the previous section, we see that the communication complexity of the DiSCO algorithm mainly depends on two quantities: the initial objective gap $f(w_0) - f(w_\star)$ and the upper bound μ on the spectral norms $\|f_1''(w_k) - f''(w_k)\|_2$ for all $k \geq 0$. As we discussed in Section 3.2, 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 to exploit the stochastic origin of the problem (SAA or ERM, as explained in Section 1) to mitigate the effect of objective scaling and quantify the notion of similarity between local and global Hessians. These lead to improved complexity results.

We focus on the setting of distributed optimization of *regularized* empirical loss. That is, our goal is to minimize $f(w) = (1/m)\sum_{i=1}^m f_i(w)$, where

$$f_i(w) = \frac{1}{n}\sum_{j=1}^{n} \phi(w, z_{i,j}) + \frac{\lambda}{2}\|w\|_2^2, \qquad i = 1, \ldots, m \tag{30}$$

We assume that $z_{i,j}$ are i.i.d. samples from a common distribution. Our theoretical analysis relies on refined assumptions on the smoothness of the loss function ϕ. In particular, we assume that for any z in the sampling space $\mathcal{Z}$, the function $\phi(\cdot, z)$ has bounded first derivative in a compact set, and its second derivatives are bounded and Lipschitz continuous. We formalize these statements in the following assumption.

Assumption 2. *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) $\|\phi''(u, z) - \phi''(w, z)\|_2 \leq M\|u - w\|_2$ *for any* $u, w \in \mathbb{R}^d$.

For the regularized empirical loss in (30), 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 1.

Recall that the initial point w_0 is obtained as the average of the solutions to m regularized local optimization problems; see equations (19) and (20). The following lemma shows that 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. The proof uses the notion and techniques of *uniform stability* for analyzing the generalization performance of ERM [7]. See Appendix D for the proof.

Lemma 5. *Suppose that Assumption 2 holds and* $\mathbb{E}[\|w_\star\|_2^2] \le D^2$ *for some constant* $D > 0$*. If we choose* $\rho = \frac{\sqrt{6}G}{\sqrt{n}D}$ *in* (20) *to compute* $\widehat{w}_i$*, then the initial point* $w_0 = \frac{1}{m}\sum_{i=1}^m \widehat{w}_i$ *satisfies*

$$\max\{\|w_\star\|_2, \|w_0\|_2\} \le \sqrt{\frac{2V_0}{\lambda}} \tag{31}$$

and

$$\mathbb{E}[f(w_0) - f(w_\star)] \le \frac{\sqrt{6}GD}{\sqrt{n}} \tag{32}$$

Here the expectation is taken with respect to the randomness in generating the i.i.d. data.

Next, we show that with high probability, $\|f_i''(w) - f''(w)\|_2 \sim \sqrt{d/n}$ for any $i \in \{1, \ldots, m\}$ and for any vector w in an ℓ_2-ball. Thus, if the number of samples n is large, the Hessian matrix of f can be approximated well by that of f_i. The proof uses random matrix concentration theories [30]. We defer the proof to Appendix E.

Lemma 6. *Suppose Assumption 2 holds. For any* $r > 0$ *and any* $i \in \{1, \ldots, m\}$*, we have with probability at least* $1 - \delta$*,*

$$\sup_{\|w\|_2 \le r} \|f_i''(w) - f''(w)\|_2 \le \mu_{r,\delta}$$

where

$$\mu_{r,\delta} \stackrel{\text{def}}{=} \min\left\{ L, \sqrt{\frac{32L^2 d}{n}} \cdot \sqrt{\log\left(1 + \frac{rM\sqrt{2n}}{L}\right) + \frac{\log(md/\delta)}{d}} \right\} \tag{33}$$

If $\phi(w, z_{i,j})$ are quadratic functions in w, then we have $M = 0$ in Assumption 2. In this case, Lemma 6 implies $\|f_i''(w) - f''(w)\|_2 \sim \sqrt{1/n}$. For general non-quadratic loss, Lemma 6 implies $\|f_i''(w) - f''(w)\|_2 \sim \sqrt{d/n}$. We use this lemma to obtain an upper bound on the spectral norm of the Hessian distances $\|f_1''(w_k) - f''(w_k)\|_2$, where the vectors w_k are generated by Algorithm 1.

Corollary 3. *Suppose Assumption 2 holds and the sequence* $\{w_k\}_{k\ge 0}$ *is generated by Algorithm 1. Let* $r = \left(\frac{2V_0}{\lambda} + \frac{2G}{\lambda}\sqrt{\frac{2V_0}{\lambda}}\right)^{1/2}$*. Then with probability at least* $1 - \delta$*, we have for all* $k \ge 0$*,*

$$\|f_1''(w_k) - f''(w_k)\|_2 \le \min\left\{ L, \sqrt{\frac{32L^2 d}{n}} \cdot \sqrt{\log\left(1 + \frac{rM\sqrt{2n}}{L}\right) + \frac{\log(md/\delta)}{d}} \right\} \tag{34}$$

Proof. We begin by upper bounding the ℓ_2-norm of w_k, for $k = 0, 1, 2 \ldots$, generated by Algorithm 1. By Theorem 1, we have $f(w_k) \le f(w_0)$ for all $k \ge 0$. By Assumption 2 (i), we have $\phi(w, z) \ge 0$ for all $w \in \mathbb{R}^d$ and $z \in \mathcal{Z}$. As a consequence,

$$\frac{\lambda}{2}\|w_k\|_2^2 \le f(w_k) \le f(w_0) \le f(0) + G\|w_0\|_2 \le V_0 + G\|w_0\|_2$$

Substituting $\|w_0\|_2 \le \sqrt{2V_0/\lambda}$ (see Lemma 5) into the above inequality yields

$$\|w_k\|_2 \le \left(\frac{2V_0}{\lambda} + \frac{2G}{\lambda}\sqrt{\frac{2V_0}{\lambda}}\right)^{1/2} = r$$

Thus, we have $\|w_k\|_2 \le r$ for all $k \ge 0$. Applying Lemma 6 establishes the corollary. $\square$

Here we remark that the dependence on d of the upper bound in (34) comes from Lemma 6, where the bound needs to hold for all point in a d-dimensional ball with radius r. However, for the analysis of the DiSCO algorithm, we only need the matrix concentration bound to hold for a finite number of vectors $w_0, w_1, \ldots, w_K$, instead of for all vectors satisfying $\|w\|_2 \le r$. Thus we conjecture that the bound in (34), especially its dependence on the dimension d, is too conservative and very likely can be tightened.

We are now ready to present the main results of our stochastic analysis. The following theorem provides an upper bound on the expected number of communication rounds required by the DiSCO algorithm to find an ϵ-optimal solution. Here the expectation is taken with respect to the randomness in generating the i.i.d. data set $\{z_{i,j}\}$.

Theorem 4. *Let Assumption 2 hold. Assume that the regularized empirical loss function f is standard self-concordant, and its minimizer $w_\star = \arg\min_w f(w)$ satisfies $\mathbb{E}[\|w_\star\|_2^2] \le D^2$ for some constant $D > 0$. Let the input parameters to Algorithm 3 be $\rho = \frac{\sqrt{6}G}{\sqrt{n}D}$ and $\mu = \mu_{r,\delta}$ in (33) with*

$$r = \left(\frac{2V_0}{\lambda} + \frac{2G}{\lambda}\sqrt{\frac{2V_0}{\lambda}}\right)^{1/2}, \qquad \delta = \frac{GD}{\sqrt{n}} \cdot \frac{\sqrt{\lambda/(4L)}}{4V_0 + 2G^2/\lambda} \tag{35}$$

Then for any $\epsilon > 0$, the total number of communication rounds T required to reach $f(\widehat{w}) - f(w_\star) \le \epsilon$ is bounded by

$$\mathbb{E}[T] \le 1 + \left(C_1 + \frac{6}{\omega(1/6)} \cdot \frac{GD}{\sqrt{n}}\right)\left(2 + C_2\left(1 + 2\sqrt{\frac{32L^2 d\, C_3}{\lambda^2 n}}\right)^{1/2}\right)$$

where C_1, C_2, C_3 are $\widetilde{\mathcal{O}}(1)$ or logarithmic terms:

$$\begin{aligned}
C_1 &= \left(1 + \left\lceil \log_2\left(\frac{2\omega(1/6)}{\epsilon}\right)\right\rceil\right)\left(1 + \frac{1}{\sqrt{n}} \cdot \frac{GD}{4V_0 + 2G^2/\lambda}\right),\\
C_2 &= \log\left(\frac{2L}{\beta\lambda}\right),\\
C_3 &= \log\left(1 + \frac{rM\sqrt{2n}}{L}\right) + \frac{\log(dm/\delta)}{d}
\end{aligned}$$

In particular, ignoring numerical constants and logarithmic terms, we have

$$\mathbb{E}[T] = \widetilde{\mathcal{O}}\left(\left(\log(1/\epsilon) + \frac{GD}{n^{1/2}}\right)\left(1 + \frac{L^{1/2}d^{1/4}}{\lambda^{1/2}n^{1/4}}\right)\right)$$

Proof. Suppose Algorithm 3 terminates in K iterations, and let t_k be the number of conjugate gradient steps in each call of Algorithm 2, for $k = 0, 1, \ldots, K-1$. For any given $\mu > 0$, we define T_μ as in (25). Let $\mathcal{A}$ denotes the event that $t_k \le T_\mu$ for all $k \in \{0, \ldots, K-1\}$. Let $\overline{\mathcal{A}}$ be the complement of $\mathcal{A}$, *i.e.*, the event that $t_k > T_\mu$ for some $k \in \{0, \ldots, K-1\}$. In addition, let the probabilities of the events $\mathcal{A}$ and $\overline{\mathcal{A}}$ be $1-\delta$ and δ respectively. By the law of total expectation, we have

$$\mathbb{E}[T] = \mathbb{E}[T|\mathcal{A}]\mathbb{P}(\mathcal{A}) + \mathbb{E}[T|\overline{\mathcal{A}}]\mathbb{P}(\overline{\mathcal{A}}) = (1-\delta)\mathbb{E}[T|\mathcal{A}] + \delta\,\mathbb{E}[T|\overline{\mathcal{A}}]$$

When the event $\mathcal{A}$ happens, we have $T \leq 1 + K(T_\mu + 1)$ where T_μ is given in (25); otherwise we have $T \leq 1 + K(T_L + 1)$, where

$$T_L = \sqrt{2 + \frac{2L}{\lambda}} \log\left(\frac{2L}{\beta\lambda}\right) \tag{36}$$

is an upper bound on the number of PCG iterations in Algorithm 2 when the event $\overline{\mathcal{A}}$ happens (see the analysis in Appendix F). Since Algorithm 2 always return a v_k such that $\|f''(w_k)v_k - f'(w_k)\|_2 \leq \epsilon_k$, the outer iteration count K share the same bound (12), which depends on the random variable $f(w_0) - f(w_\star)$. However, T_μ and T_L are deterministic constants. So we have

$$\begin{aligned}\mathbb{E}[T] &\leq 1 + (1-\delta)\mathbb{E}[K(T_\mu + 1)|\mathcal{A}] + \delta\,\mathbb{E}[K(T_L+1)|\overline{\mathcal{A}}] \\ &= 1 + (1-\delta)(T_\mu+1)\mathbb{E}[K|\mathcal{A}] + \delta(T_L+1)\mathbb{E}[K|\overline{\mathcal{A}}].\end{aligned} \tag{37}$$

Next we bound $\mathbb{E}[K|\mathcal{A}]$ and $\mathbb{E}[K|\overline{\mathcal{A}}]$ separately. To bound $\mathbb{E}[K|\mathcal{A}]$, we use

$$\mathbb{E}[K] = (1-\delta)\mathbb{E}[K|\mathcal{A}] + \delta\,\mathbb{E}[K|\overline{\mathcal{A}}] \geq (1-\delta)\mathbb{E}[K|\mathcal{A}]$$

to obtain

$$\mathbb{E}[K|\mathcal{A}] \leq \mathbb{E}[K]/(1-\delta) \tag{38}$$

In order to bound $\mathbb{E}[K|\overline{\mathcal{A}}]$, we derive a deterministic bound on $f(w_0) - f(w_\star)$. By Lemma 5, we have $\|w_0\|_2 \leq \sqrt{2V_0/\lambda}$, which together with Assumption 2 (ii) yields

$$\|f'(w)\|_2 \leq G + \lambda\|w\|_2 \leq G + \sqrt{2\lambda V_0}$$

Combining with the strong convexity of f, we obtain

$$f(w_0) - f(w_\star) \leq \frac{1}{2\lambda}\|f'(w_0)\|_2^2 \leq \frac{1}{2\lambda}\left(G + \sqrt{2\lambda V_0}\right)^2 \leq 2V + \frac{G^2}{\lambda}$$

Therefore by Corollary 1,

$$K \leq K_{\max} \overset{\text{def}}{=} 1 + \frac{4V_0 + 2G^2/\lambda}{\omega(1/6)} + \left\lceil \log_2\left(\frac{2\omega(1/6)}{\epsilon}\right)\right\rceil \tag{39}$$

where the additional 1 counts compensate for removing one $\lceil\cdot\rceil$ operator in (12).

Using inequality (37), the bound on $\mathbb{E}[K|\mathcal{A}]$ in (38) and the bound on $\mathbb{E}[K|\overline{\mathcal{A}}]$ in (39), we obtain

$$\mathbb{E}[T] \leq 1 + (T_\mu + 1)\mathbb{E}[K] + \delta(T_L + 1)K_{\max}$$

Now we can bound $\mathbb{E}[K]$ by Corollary 1 and Lemma 5. More specifically,

$$\mathbb{E}[K] \leq \frac{\mathbb{E}[f(w_0) - f(w_\star)]}{\frac{1}{2}\omega(1/6)} + \left\lceil \log_2\left(\frac{2\omega(1/6)}{\epsilon}\right)\right\rceil + 1 = C_0 + \frac{2\sqrt{6}}{\omega(1/6)} \cdot \frac{GD}{\sqrt{n}} \tag{40}$$

where $C_0 = 1 + \lceil \log_2(2\omega(1/6)/\epsilon)\rceil$. With the choice of δ in (35) and the definition of T_L in (36), we have

$$\delta(T_L+1)K_{\max} = \frac{GD}{\sqrt{n}} \cdot \frac{\sqrt{\lambda/(4L)}}{4V_0 + 2G^2/\lambda}\left(2 + \sqrt{2 + \frac{2L}{\lambda}}\log\left(\frac{2L}{\beta\lambda}\right)\right)\left(C_0 + \frac{4V_0 + 2G^2/\lambda}{\omega(1/6)}\right)$$

$$
\begin{aligned}
&= \left(\frac{C_0}{\sqrt{n}} \cdot \frac{GD}{4V_0 + 2G^2/\lambda} + \frac{1}{\omega(1/6)} \cdot \frac{GD}{\sqrt{n}}\right) \left(\sqrt{\frac{\lambda}{L}} + C_2\sqrt{\frac{\lambda}{2L} + \frac{1}{2}}\right) \\
&\leq \left(\frac{C_0}{\sqrt{n}} \cdot \frac{GD}{4V_0 + 2G^2/\lambda} + \frac{1}{\omega(1/6)} \cdot \frac{GD}{\sqrt{n}}\right) \left(2 + C_2\sqrt{1 + \frac{2\mu}{\lambda}}\right) \\
&= \left(\frac{C_0}{\sqrt{n}} \cdot \frac{GD}{4V_0 + 2G^2/\lambda} + \frac{1}{\omega(1/6)} \cdot \frac{GD}{\sqrt{n}}\right) (T_\mu + 1)
\end{aligned}
$$

Putting everything together, we have

$$
\begin{aligned}
\mathbb{E}[T] &\leq 1 + \left(C_0 + \frac{C_0}{\sqrt{n}} \cdot \frac{GD}{4V_0 + 2G^2/\lambda} + \frac{2\sqrt{6}+1}{\omega(1/6)} \cdot \frac{GD}{\sqrt{n}}\right) (T_\mu + 1) \\
&\leq 1 + \left(C_1 + \frac{6}{\omega(1/6)} \cdot \frac{GD}{\sqrt{n}}\right) (T_\mu + 1)
\end{aligned}
$$

Replacing T_μ by its expression in (25) and applying Corollary 3, we obtain the desired result. □

According to Theorem 4, we need to set the two input parameters ρ and μ in Algorithm 3 appropriately to obtain the desired communication efficiency. Using the adaptive DiSCO method given in Algorithm 4, we can avoid the explicit specification of $\mu = \mu_{r,\delta}$ defined in (33) and (35). This is formalized in the following theorem.

Theorem 5. *Let Assumption 2 hold. Assume that the regularized empirical loss function f is standard self-concordant, and its minimizer $w_\star = \arg\min_w f(w)$ satisfies $\mathbb{E}[\|w_\star\|_2^2] \leq D^2$ for some constant $D > 0$. Let the input parameters to Algorithm 4 be $\rho = \frac{\sqrt{6}G}{\sqrt{n}D}$ and any $\mu_0 > 0$. Then the total number of communication rounds T required to reach $f(\widehat{w}) - f(w_\star) \leq \epsilon$ is bounded by*

$$
\mathbb{E}[T] = \widetilde{\mathcal{O}}\left(\left(\log(1/\epsilon) + \frac{GD}{n^{1/2}}\right)\left(1 + \frac{L^{1/2}d^{1/4}}{\lambda^{1/2}n^{1/4}}\right)\right)
$$

Proof. In Algorithm 4, the parameter μ_k is automatically tuned such that the number of PCG iterations in Algorithm 2 is no more than T_{μ_k}. By Corollary 3, with probability at least $1-\delta$, we have

$$
\max\{\mu_0, \ldots, \mu_K\} \leq 2\mu_{r,\delta}
$$

where $\mu_{r,\delta}$ is defined in (33), and r and δ are given in (35). Therefore we can use the same arguments in the proof of Theorem 4 to show that

$$
\mathbb{E}[T] \leq 1 + \left(\widetilde{C}_1 + \frac{6}{\omega(1/6)} \cdot \frac{GD}{\sqrt{n}}\right) \left(2 + C_2\left(1 + 4\sqrt{\frac{32L^2 d\, C_3}{\lambda^2 n}}\right)^{1/2}\right)
$$

where

$$
\widetilde{C}_1 = \left(2 + 2\left\lceil \log_2\left(\frac{2\omega(1/6)}{\epsilon}\right)\right\rceil + \log_2\left(\frac{L}{\mu_0}\right)\right)\left(1 + \frac{1}{\sqrt{n}} \cdot \frac{GD}{4V_0 + 2G^2/\lambda}\right)
$$

and C_2 and C_3 are the same as given in Theorem 4. Ignoring constants and logarithmic terms, we obtain the desired result. □

In both Theorems 4 and 5, the parameter $\rho = \frac{\sqrt{6}G}{\sqrt{n}D}$ depends on a constant D such that $\mathbb{E}[\|w_\star\|_2^2] \leq D^2$. In practice, it may be hard to give a tight estimate of $\mathbb{E}[\|w_\star\|_2^2]$. An alternative is to fix a desired value of D and consider the constrained optimization problem

$$\underset{\|w\|_2 \leq D}{\text{minimize}} \; f(w)$$

To handle the constraint $\|w\|_2 \leq D$, we need to replace the inexact damped Newton method in DiSCO with an inexact *proximal* Newton method, and replace the distributed PCG method for solving the Newton system with a preconditioned accelerated proximal gradient method. Further details of such an extension are given in Section 6.

Remarks The expectation bounds on the rounds of communication given in Theorems 4 and 5 are obtained by combining two consequences of averaging over a large number of i.i.d. local samples. One is the expected reduction of the initial gap $f(w_0) - f(w_\star)$ (Lemma 5), which helps to mitigate the effect of objective scaling used to make f standard self-concordant. The other is a high-probability bound that characterizes the similarity between the local and global Hessians (Corollary 3). If the empirical loss f is standard self-concordant without scaling, then we can regard $f(w_0) - f(w_\star)$ as a constant, and only need to use Corollary 3 to obtain a high-probability bound. This is demonstrated for the case of linear regression in Section 5.1.

For applications where the loss function needs to be rescaled to be standard self-concordant, the convexity parameter λ as well as the "constants" (V_0, G, L, M) in Assumption 2 also need to be rescaled. If the scaling factor grows with n, then we need to rely on Lemma 5 to balance the effects of scaling. As a result, we only obtain bounds on the expected number of communication rounds. These are demonstrated in Section 5.2 for binary classification with logistic regression and a smoothed hinge loss.

5.1 Application to linear regression

We consider linear regression with quadratic regularization (ridge regression). More specifically, we minimize the overall empirical loss function

$$f(w) = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}(y_{i,j} - w^T x_{i,j})^2 + \frac{\lambda}{2}\|w\|_2^2 \tag{41}$$

where the i.i.d. instances $(x_{i,j}, y_{i,j})$ are sampled from $\mathcal{X}\times\mathcal{Y}$. We assume that $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}$ are bounded: there exist constants B_x and B_y such that $\|x\|_2 \leq B_x$ and $|y| \leq B_y$ for any $(x, y) \in \mathcal{X}\times\mathcal{Y}$. It can be shown that the least-squares loss $\phi(w, (x, y)) = (y - w^T x)^2$ satisfies Assumption 2 with

$$V_0 = B_y^2, \qquad G = 2B_x\left(B_y + B_x B_y\sqrt{2/\lambda}\right), \qquad L = \lambda + 2B_x^2, \qquad M = 0$$

Thus we can apply Theorems 4 and 5 to obtain an expectation bound on the number of communication rounds for DiSCO. For linear regression, however, we can obtain a stronger result.

Since f is a quadratic function, it is self-concordant with parameter 0, and by definition also standard self-concordant (with parameter 2). In this case, we do not need to rescale the objective function, and can regard the initial gap $f(w_0) - f(w_\star)$ as a constant. As a consequence, we can directly apply Theorem 2 and Corollary 3 to obtain a high probability bound on the communication complexity, which is stronger than the expectation bounds in Theorems 4 and 5. In particular, Theorem 2 states that if

$$\left\|f_1''(w_k) - f''(w_k)\right\|_2 \le \mu, \quad \text{for all} \quad k = 0, 1, 2, \ldots \tag{42}$$

then the number of communication rounds T is bounded as

$$T \le 1 + \left(\left\lceil \frac{f(w_0) - f(w_\star)}{\omega(1/6)} \right\rceil + \left\lceil \log_2\left(\frac{2\omega(1/6)}{\epsilon}\right) \right\rceil\right)\left(2 + \sqrt{1 + \frac{2\mu}{\lambda}} \log\left(\frac{2L}{\beta\lambda}\right)\right)$$

Since there is no scaling, the initial gap $f(w_0) - f(w_\star)$ can be considered as a constant. For example, we can simply pick $w_0 = 0$ and have

$$f(0) - f(w_\star) \le f(0) = \frac{1}{N}\sum_{i=1}^{N} y_i^2 \le B_y^2$$

By Corollary 3 and the fact that $M = 0$ for quadratic functions, the condition (42) holds with probability at least $1 - \delta$ if we choose

$$\mu = \sqrt{\frac{32L^2 d}{n}}\sqrt{\frac{\log(md/\delta)}{d}} = \frac{8L}{\sqrt{n}}\sqrt{2\log(md/\delta)} \tag{43}$$

Further using $L \le \lambda + 2B_x^2$, we obtain the following corollary.

Corollary 4. *Suppose we apply DiSCO (Algorithm 3) to minimize $f(w)$ defined in* (41) *with the input parameter μ in* (43)*, and let T be the total number of communication rounds required to find an ϵ-optimal solution. With probability at least $1 - \delta$, we have*

$$T = \widetilde{\mathcal{O}}\Big(\Big(1 + \frac{B_x}{\lambda^{1/2} n^{1/4}}\Big)\log(1/\epsilon)\log(md/\delta)\Big) \tag{44}$$

We note that the same conclusion also holds for the adaptive DiSCO algorithm (Algorithm 4), where we do not need to specify the input parameter μ based on (43). For the adaptive DiSCO algorithm, the bound in (44) holds for any $\delta \in (0, 1)$.

The communication complexity guaranteed by Corollary 4 is strictly better than that of distributed implementation of the accelerated gradient method and ADMM (*cf.* Table 1). If we choose $\lambda = \Theta(1/\sqrt{mn})$, then Corollary 4 implies

$$T = \widetilde{\mathcal{O}}\left(m^{1/4}\log(1/\epsilon)\right)$$

with high probability. The DANE algorithm [47], under the same setting, converges in $\widetilde{\mathcal{O}}(m\log(1/\epsilon))$ iterations with high probability (and each iteration requires two rounds of communication). Thus DiSCO enjoys a better communication efficiency.

5.2 Application to binary classification

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

$$\ell(w) \overset{\text{def}}{=} \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\varphi(y_{i,j}w^T x_{i,j}) + \frac{\gamma}{2}\|w\|_2^2 \tag{45}$$

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

Under the above assumptions, Lemma 2 gives conditions on φ for ℓ to be self-concordant. As we have seen in Section 2, the function ℓ usually needs to be scaled by a large factor to become standard self-concordant. Let the scaling factor be η, we can use DiSCO to minimize the scaled function $f(w) = \eta\ell(w)$. Next we discuss the theoretical implications for logistic regression and the smoothed hinge loss constructed in Section 2. These results are summarized in Table 1.

Logistic Regression For logistic regression, we have $\varphi(t) = \log(1 + e^{-t})$. In Section 2, we have shown that the logistic loss satisfies the condition of Lemma 2 with $Q = 1$ and $\alpha = 0$. Consequently, with the factor $\eta = \frac{B^2}{4\gamma}$, the rescaled function $f(w) = \eta\ell(w)$ is standard self-concordant. If we express f in the standard form

$$f(w) = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\phi(y_{i,j}w^T x_{i,j}) + \frac{\lambda}{2}\|w\|_2^2 \tag{46}$$

then we have $\phi(w, (x, y)) = \eta\varphi(yw^T x)$ and $\lambda = \eta\gamma$. It is easy to check that Assumption 2 holds with

$$V_0 = \eta\log(2), \qquad G = \eta B, \qquad L = \eta(B^2/4 + \gamma), \qquad M = \eta B^3/10$$

which all containing the scaling factor η. Plugging these scaled constants into Theorems 4 and 5, we have the following corollary.

Corollary 5. *For logistic regression, the number of communication rounds required by DiSCO to find an ϵ-optimal solution is bounded by*

$$\mathbb{E}[T] = \widetilde{\mathcal{O}}\Big(\Big(\log(1/\epsilon) + \frac{B^3 D}{\gamma n^{1/2}}\Big)\Big(1 + \frac{Bd^{1/4}}{\gamma^{1/2}n^{1/4}}\Big)\Big)$$

In the specific case when $\gamma = \Theta(1/\sqrt{mn})$, Corollary 5 implies

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

If we ignore logarithmic terms, then the expected number of communication rounds is independent of the sample size n, and only grows slowly with the number of machines m.

Smoothed Hinge Loss We consider minimizing $\ell(w)$ in (45) where the loss function φ is the smoothed hinge loss defined in (8), which depends on a parameter $p \ge 3$. Using Lemma 2, we have shown in Section 2 that $\ell(w)$ is self-concordant with parameter M_p given in (9). As a consequence, by choosing

$$\eta = \frac{M_p^2}{4} = \frac{(p-2)^2 B^{2+\frac{4}{p-2}}}{4\gamma^{1+\frac{2}{p-2}}}$$

the function $f(w) = \eta\ell(w)$ is standard self-concordant. If we express f in the form of (46), then $\phi(w, (x, y)) = \eta\varphi_p(yw^T x)$ and $\lambda = \eta\gamma$. It is easy to verify that Assumption 2 holds with

$$V_0 = \eta, \qquad G = \eta B, \qquad L = \eta(B^2 + \lambda), \qquad M = \eta(p-2)B^3$$

If we choose $p = 2 + \log(1/\gamma)$, then applying Theorems 4 and 5 yields the following result.

Corollary 6. *For the smoothed hinge loss φ_p defined in* (8) *with* $p = 2+\log(1/\gamma)$*, the total number of communication rounds required by DiSCO to find an ϵ-optimal solution is bounded by*

$$\mathbb{E}[T] = \widetilde{\mathcal{O}}\Big(\Big(\log(1/\epsilon) + \frac{B^3 D}{\gamma n^{1/2}}\Big)\Big(1 + \frac{Bd^{1/4}}{\gamma^{1/2} n^{1/4}}\Big)\Big)$$

Thus, the smoothed hinge loss enjoys the same communication efficiency as the logistic loss.

6 Extension to distributed composite minimization

Thus far, we have studied the problem of minimizing empirical loss functions that are standard self-concordant. In this section, we sketch how to extend the DiSCO algorithm to solve distributed composite minimization problems. By composite minimization, we consider the minimization of

$$F(w) \overset{\text{def}}{=} f(w) + \Psi(w) \tag{47}$$

where f is a standard self-concordant function taking the form of (2), and Ψ a closed convex function with a simple structure (see discussions in [35]). For solving the Lasso [49], for example, the ℓ_1-penalty $\Psi(w) = \sigma\|w\|_1$ with $\sigma > 0$ is nonsmooth but admits a simple proximal mapping.

We modify Algorithm 1 and Algorithm 2 to minimize the composite function $F(w)$. To modify Algorithm 1, we update w_{k+1} using an inexact version of the proximal-Newton method (*e.g.*, [24, 50]). More specifically, the two steps in each iteration of Algorithm 1 are replaced with:

1. Find a vector v_k that is an approximate solution of

$$\underset{v\in\mathbb{R}^d}{\text{minimize}} \quad \Big\{\frac{1}{2} v^T f''(w_k) v - v^T f'(w_k) + \Psi(w_k - v)\Big\} \tag{48}$$

2. Update $w_{k+1} = w_k - \frac{v_k}{1+\sqrt{v^T f''(w_k) v_k}}$.

Note that for $\Psi(w) \equiv 0$, the above proximal-Newton method reduces to Algorithm 1. Since v_k only needs to be an inexact solution to problem (48), we need a measure to quantify the approximation error. For this purpose, we define the following gradient mapping

$$g(v_k) \overset{\text{def}}{=} \arg\min_{g\in\mathbb{R}^d} \Big\{\frac{L}{2}\|g\|_2^2 + \langle f''(w_k)v_k - f'(w_k), g\rangle + \Psi(w_k - v_k + g)\Big\}$$

If v_k is an exact minimizer of (48), then we have $\|g(v_k)\|_2 = 0$. In the distributed setting, we only need to find a vector v_k such that $\|g(v_k)\|_2 \le \epsilon_k$.

It remains to devise an distributed algorithm to compute an inexact minimizer v_k. Since the objective function in (48) is not quadratic, we can no longer employ the distributed PCG method in Algorithm 2. Instead, we propose a preconditioned accelerated proximal gradient method. In particular, we modify the algorithm on the master machine in Algorithm 2 as follows:

$$\begin{aligned} v^{(t+1)} &= \arg\min_{v\in\mathbb{R}^d} \Big\{\frac{1}{2}(v - s^{(t)})^T [f_1''(w_k) + \mu I](v - s^{(t)}) \\ &\qquad\qquad + \langle f''(w_k) s^{(t)} - f'(w_k), v - s^{(t)}\rangle + \Psi(w_k + v)\Big\}, \qquad (49) \\ s^{(t+1)} &= v^{(t+1)} + \frac{\sqrt{1+2\mu/\lambda} - 1}{\sqrt{1+2\mu/\lambda}+1}(v^{(t+1)} - v^{(t)}), \end{aligned}$$

where $s^{(t+1)}$ is an auxiliary vector. We output $v_k = v^{(t+1)}$ once the condition $\|g(v^{(t+1)})\|_2 \le \epsilon_k$ is satisfied. Each update takes one round of communication to compute the vector $f''(w_k)s^{(t)}$. Then, the sub-problem (49) is locally solved by the master machine. This problem has similar structure as problems (20) and (26), and can be solved in $\mathcal{O}\big((n + \frac{L+\mu}{\lambda+\mu})\log(1/\epsilon)\big)$ time using the methods proposed in [44, 51, 14].

If we replace the first term on the right-hand side of equation (49) by $\frac{L}{2}\|v - v^{(t)}\|_2^2$ and set $\mu = L$, then the above algorithm is exactly the accelerated proximal gradient algorithm [35, 28], which converges in $\widetilde{\mathcal{O}}(\sqrt{L/\lambda})$ iterations. By utilizing the similarity between $f_1''(w_k)$ and $f''(w_k)$, and assuming $\|f_1''(w_k) - f''(w_k)\|_2 \le \mu$ for all $k \ge 0$, it can be shown that our algorithm in (49) converges in $\widetilde{\mathcal{O}}(1 + \sqrt{\mu/\lambda})$ iterations, which is of the same order as the PCG algorithm.

In summary, to minimize the composite function $f(w) + \Psi(w)$, we replace Algorithm 1 by the inexact proximal Newton method, and replace Algorithm 2 by a distributed implementation of the above preconditioned accelerated proximal gradient method. Under the same assumptions on f, we can obtain similar guarantees on the communication efficiency as stated in Theorems 4 and 5.

7 Conclusion

We addressed the problem of distributed convex optimization in the context of sample average approximation (SAA) or empirical risk minimization (ERM) commonly found in machine learning tasks. These problems involve large datasets, distributed across multiple machines, where communication efficiency becomes critical due to the high cost of inter-machine communication. Traditional first-order methods, such as accelerated gradient methods and ADMM, suffer from an increasing number of communication rounds as the condition number of the objective function grows, particularly when weaker regularization is required in large-scale problems. This results in communication costs scaling with the overall sample size.

To overcome these challenges, we proposed and analyzed the Distributed Self-Concordant Optimization (DiSCO) algorithm. DiSCO is a communication-efficient distributed algorithm based on an inexact damped Newton method, where inexact Newton steps are computed using a distributed preconditioned conjugate gradient (PCG) method. Our analysis showed that DiSCO's communication rounds do not increase with the sample size and only grow slowly with the number of machines, making it highly scalable and efficient for large-scale distributed optimization.

Key contributions of our work include:

- *Self-concordant analysis:* We demonstrated that several popular empirical loss functions used in machine learning, such as those for linear regression and classification, are either self-concordant or can be well approximated by self-concordant functions. We provided a complexity analysis of the inexact damped Newton method and characterized conditions for linear and superlinear convergence.

- *Preconditioned conjugate gradient (PCG) method:* We developed a distributed implementation of the PCG method to compute the inexact Newton steps. The preconditioner, which leverages the similarity between local and global Hessians, effectively reduces the number of communication rounds, achieving significant improvements in both theory and practice.

- *Stochastic analysis of communication efficiency:* We provided theoretical results based on averaging over large i.i.d. samples, showing two key effects: (1) an expected reduction in the initial objective value that offsets the scaling required for self-concordance, and (2) a high-probability bound that captures the similarity between local and global Hessians, further reducing communication overhead.

Additionally, we proposed extensions of DiSCO to handle composite empirical loss functions, further broadening its applicability in distributed optimization problems.

References

[1] A. Agarwal and J. C. Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 873–881, 2011.

[2] M. Avriel. *Nonlinear Programming: Analysis and Methods*. Prentice-Hall, 1976.

[3] F. Bach. Self-concordant analysis for logistic regression. *Electronic Journal of Statistics*, 4:384–414, 2010.

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

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

[6] J. A. Blackard, D. J. Dean, and C. W. Anderson. Covertype data set. In K. Bache and M. Lichman, editors, *UCI Machine Learning Repository*, URL: http://archive.ics.uci.edu/ml, 2013. University of California, Irvine, School of Information and Computer Sciences.

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

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

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

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

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

[12] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

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

[14] A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1646–1654. Curran Associates, Inc., 2014.

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

[16] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM Journal on Numerical Analysis*, 19(2):400–408, April 1982.

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

[18] J. E. Dennis and J. J. Moré. A characterization of superlinear convergence and its application to quasi-Newton methods. *Mathematics of Computation*, 28(126):549–560, April 1974.

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

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

[21] R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, 1985.

[22] S. S. Keerthi and D. DeCoste. A modified finite Newton method for fast solution of large scale linear svms. *Journal of Machine Learning Research*, 6:341–361, 2005.

[23] K. Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning (ICML)*, pages 331–339, 1995.

[24] J. D. Lee, Y. Sun, and M. Saunders. Proximal Newton-type methods for minimizing composite functions. *SIAM Journal on Optimization*, 24(3):1420–1443, 2014.

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

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

[27] Q. Lin, Z. Lu, and L. Xiao. An accelerated proximal coordinate gradient method and its application to regularized empirical risk minimization. Technical Report MSR-TR-2014-94, Microsoft Research, 2014. arXiv:1407.1296.

[28] Q. Lin and L. Xiao. An adaptive accelerated proximal gradient method and its homotopy contiuation for sparse optimization. *Computational Optimization and Applications*, published online, September 2014.

[29] D. G. Luenberger. *Introduction to Linear and Nonlinear Programming.* Addison-Wesley, New York, 1973.

[30] L. Mackey, M. I. Jordan, R. Y. Chen, B. Farrell, J. A. Tropp, et al. Matrix concentration inequalities via the method of exchangeable pairs. *The Annals of Probability*, 42(3):906–945, 2014.

[31] D. Mahajan, S. S. Keerthi, S. Sundararajan, and L. Bottou. A functional approximation based distributed learning algorithm. arXiv:1310.8418, 2013.

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

[33] A. Nemirovsky and D. Yudin. *Problem Complexity and Method Efficiency in Optimization.* J. Wiley & Sons, New York, 1983.

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

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

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

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

[38] S. S. Ram, A. Nedić, and V. V. Veeravalli. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of optimization theory and applications*, 147(3):516–545, 2010.

[39] B. Recht, C. Re, S. Wright, and F. Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 693–701, 2011.

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

[41] M. Schmidt, N. L. Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. Technical Report HAL 00860051, INRIA, Paris, France, 2013.

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

[43] S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. arXiv:1309.2375.

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

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

[46] O. Shamir and N. Srebro. On distributed stochastic optimization and learning. In *Proceedings of the 52nd Annual Allerton Conference on Communication, Control, and Computing*, 2014.

[47] O. Shamir, N. Srebro, and T. Zhang. Communication efficient distributed optimization using an approximate Newton-type method. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*. JMLR: W&CP volume 32, 2014.

[48] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory*. MPS-SIAM Series on Optimization. SIAM-MPS, Philadelphia, PA, 2009.

[49] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[50] Q. Tran-Dinh, A. Kyrillidis, and V. Cevher. Composite self-concordant minimization. *arXiv preprint arXiv:1308.2867*, 2013.

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

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

[53] Y. Zhang and L. Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. Technical Report MSR-TR-2014-123, Microsoft Research, 2014. arXiv:1409.3257.

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