Communication-Efficient Distributed PCA by Riemannian Optimization

Long-Kai Huang [1] **Sinno Jialin Pan** [1]

Abstract

In this paper, we study the leading eigenvector problem in a statistically distributed setting and propose a communication-efficient algorithm based on Riemannian optimization, which trades local computation for global communication. Theoretical analysis shows that the proposed algorithm linearly converges to the centralized empirical risk minimization solution regarding the number of communication rounds. When the number of data points in local machines is sufficiently large, the proposed algorithm achieves a significant reduction of communication cost over existing distributed PCA algorithms. Superior performance in terms of communication cost of the proposed algorithm is verified on real-world and synthetic datasets.

1. Introduction

Finding top eigenvectors of a symmetric matrix is a fundamental problem for various machine learning problems, such as principal component analysis (PCA) (Hotelling, 1933; Wilkinson, 1965; Bishop, 2006), spectral clustering (Ng et al., 2002; Von Luxburg, 2007), etc. In this paper, we focus on finding the first principal component of PCA, i.e. the leading eigenvector of the data covariance matrix, in a statistical setting where the data instances $x \in \mathbb{R}^d$ are from some unknown but fixed distribution. This goal can be formulated as a population risk minimization problem as

$$\min_{w\in\mathbb{R}^d, \|w\|=1} F(w) = -\tfrac{1}{2} w^\top \mathbb{E}_{x\sim\mathcal{D}}\left[xx^\top\right] w. \quad (1)$$

Given a set of i.i.d. data examples x_i for $i = 1, 2, ..., N$ from the distribution $\mathcal{D}$, we can use the solution of the following empirical risk minimization (ERM) problem

$$w^* = \arg\min_{w\in\mathbb{R}^d, \|w\|=1} \hat{F}(w) = -\tfrac{1}{2} w^\top \hat{A} w, \quad (2)$$

as an approximate minimizer of F in (1), where $\hat{A} = \frac{1}{N}\sum_{i=1}^N x_i x_i^\top$ is the empirically normalized covariance matrix. If the eigengap δ between the first and second eigenvalues of $\hat{A}$ is positive, the optimum of (2) is $w^* = v$ (or $-v$) where v is the leading eigenvector of $\hat{A}$.

In the literature, various algorithms based on the Euclidean space have been developed to solve the PCA problem (Golub & Van Loan, 1996; Oja & Karhunen, 1985; Shamir, 2015; 2016). In recent years, the problem (2) is also studied based on Riemannian optimization (Absil et al., 2008) as the unit norm constraint on the variable in (2) indeed admits the geometry structure as a hypersphere (a Riemannian manifold). On the hypersphere manifold, the objective function (2) becomes an unconstrained problem and thus Riemannian optimization algorithms (Absil et al., 2008), such as Riemannian SGD (Bonnabel, 2013; Zhang & Sra, 2016), Riemannian SVRG (RSVRG) (Zhang et al., 2016), can be directly applied to solve it.

All of the aforementioned algorithms assume that a single machine is able to store and access all the training data to solve the leading eigenvector problem in (2). However, due to the explosive increase of training data, the data size may exceed the storage and computation capacities of a single machine. In addition, the training data may be originally geo-distributed over different local machines. Centralizing all the training data to a single machine is impractical due to the extremely high data transmission cost. Thus, it is crucial to consider (2) in a distributed setting, where N data samples are stored in K machines, each maintaining n instances x_i^k.[1] In the distributed setting, each machine k can construct a local estimate of $F(w)$: $f_k(w) = -\frac{1}{2} w^\top A_k w$, where $A_k = \frac{1}{n}\sum_{i=1}^n x_i^k x_i^{k\top}$. In general, the local empirical estimation $f_k(w)$ is much less accurate than centralized estimation $\hat{F}(w)$. Therefore, the local machines should cooperate and communicate with each other to obtain a solution converging to the centralized ERM solution w^*.

A simple algorithm to solve (2) in the distributed setting is

[1]School of Computer Science and Engineering, Nanyang Technological University, Singapore. Correspondence to: Long-Kai Huang <LHUANG018@e.ntu.edu.sg>, Sinno J. Pan <sinnopan@ntu.edu.sg>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

[1]For simplicity in analysis, we assume the sample size in each machine is the same.

to extend Power Method, Lanczos algorithm or Riemannian Gradient Descent (RGD) to their distributed versions. To achieve ϵ-far solution from w^*, Distributed Power Method or RGD requires $\mathcal{O}\big(\lambda/\delta \log(1/\epsilon)\big)$ communication rounds; Distributed Lanczos algorithm and accelerated RGD requires $\mathcal{O}\big(\sqrt{\lambda/\delta}\log(1/\epsilon)\big)$. Here, λ is the largest eigenvalue of $\hat{A}$ in (2) and $\mathcal{O}(\cdot)$ suppresses logarithmic factors in d and failure probability. When δ is very small, i.e. $\delta = \Omega(1/\sqrt{Kn})$ as discussed in (Garber et al., 2017; Shamir, 2015), the number of communication rounds of these algorithms increases with the sample size. As in real-world systems, the communication speed is limited, which is a bottleneck for distributed optimization (Jaggi et al., 2014), these algorithms are impractical for large-scale datasets.

Most existing communication-efficient distributed algorithms (Shamir et al., 2014; Zhang & Lin, 2015; Jaggi et al., 2014; Saparbayeva et al., 2018) cannot be directly adapted to the PCA problem due to the non-convex structure and unit norm constraints of the objective function. In a recent paper, Garber et al. (2017) proposed to replace the explicit Power Method iterations with a series of convex optimization problems and developed a distributed version, named Distributed Shift-and-Invert Power Method (DSI), by approximately solving these convex problems using existing communication-efficient algorithms designed for convex problems. Assume the squared ℓ_2 norm of the data instances x_i is bounded by b. Then the communication rounds of DSI needed to achieve an ϵ-accurate solution is $\mathcal{O}\big(\sqrt{\frac{b}{\delta\sqrt{n}}}\big[\log^2(1/\epsilon)\log(1/\delta) + \log(1/\epsilon)\log^2(\frac{1}{\delta\sqrt{n}})\big]\big)$. When δ is as small as $\delta = \Omega(1/\sqrt{Kn})$, the communication cost nearly does not increase with the sample size. Therefore, DSI is communication-efficient for getting a low to medium-accuracy solution. However, the quadratic dependence on $\log(1/\epsilon)$ is a concern for getting a high-accuracy solution.

Another line of distributed PCA algorithms (Kannan et al., 2014; Liang et al., 2014; Boutsidis et al., 2016; Fan et al., 2017) is studied in the deterministic setting where data are arbitrarily partitioned over local machines. To reduce the communication cost, these algorithms first approximate the local data by low-rank approximation or random sketching, and then perform one-shot communication to aggregate all local approximations to a master machine and reconstruct an approximate covariance matrix. The communication rounds required by these algorithms scale polynomially with $1/\delta$ and $1/\epsilon$, which means they are not communication-efficient for getting a high-accuracy solution or for small eigengap.

In this paper, we propose a novel communication-efficient algorithm for distributed stochastic PCA based on Riemannian optimization, named Communication-Efficient Distributed Riemannian Eigensolver (CEDRE). As will be analyzed in Theorem 3 and Corollary 2, the communication rounds of CEDRE needed to achieve an ϵ-accurate solution is $\mathcal{O}\big(\frac{b}{\delta\sqrt{n}}\log(1/\epsilon)\big)$. Compared to distributed versions of Power Method, Lanczos algorithm and RGD, the communication complexity of CEDRE does not scale with the sample size when $\delta = \Omega(1/\sqrt{Kn})$. Besides, for a fixed δ, the number of communication rounds required decreases when the sample size increases. Compared to DSI, the communication complexity of CEDRE depends only logarithmically on the accuracy ϵ. Therefore, for a sufficiently large n, the proposed algorithm is communication-efficient to get a high-accurate solution even when the eigengap δ is small.

The key technique to make CEDRE communication-efficient is to independently update the local variables by a surrogate gradient (defined in (6)) whose difference from the global full gradient is bounded (as analyzed in Lemma 4). With this technique, the local update approximates the global update with small error and therefore only regular communication is required to get consensus on the local variables among different local machines. This technique is inspired by the variance reduced eigensolvers (Shamir, 2015; 2016; Xu et al., 2017). However, their analysis requires an unbiased estimation of the global gradient in each iteration, which cannot be satisfied in the distributed setting. Besides, this technique is also inspired by the communication-efficient frameworks that trade local computation for communication (Shamir et al., 2014; Jaggi et al., 2014; Jordan et al., 2018; Saparbayeva et al., 2018). However, these frameworks rely on the strong convexity of the optimization problem, while our objective function in (2) is not convex in either the Euclidean space or the Riemannian manifold. Therefore, the analysis of our proposed algorithm is new and different from the previous works.

2. Preliminaries

Given a Riemannian manifold $\mathcal{M}$, its tangent space, denoted by $T_w\mathcal{M}$, is a set of all tangent vectors at the point w. The manifold gradient of an objective function $f(w)$, denoted by $\tilde{\nabla} f(w)$, can be obtained by mapping the Euclidean gradient onto the tangent space $T_w\mathcal{M}$ with the projection operator $P_w(\cdot)$. Applying Riemannian GD (Absil et al., 2008) on f, its variable w can be updated while preserving in the manifold $\mathcal{M}$ as $w^+ = R_w(-\eta\tilde{\nabla} f(w))$, where $\eta > 0$ is the step size and $R_w(\xi)$ is the retraction (a.k.a exponential map) at w mapping a tangent vector $\xi \in T_w\mathcal{M}$ to another point on $\mathcal{M}$ along a geodesic. Here, the geodesic is a shortest curve connecting two points on a manifold with zero acceleration, which is a generalization of a straight line in the Euclidean space. The inverse of $R_w(\cdot)$, denoted by $R_w^{-1}(u)$, returns a tangent vector ξ in $T_w\mathcal{M}$ pointing towards $u \in \mathcal{M}$ such that $R_w(\xi) = u$. The distance of two points $w, u \in \mathcal{M}$ is defined as $\mathrm{D}(w,u) = \|R_w^{-1}(u)\|$. Along the geodesic, the parallel transport, $\Gamma_w^u(\xi)$, maps a tangent vector $\xi \in T_w\mathcal{M}$ to $T_u\mathcal{M}$. The vector transport, denoted as $\mathcal{T}_w^u(\xi)$, is a first-

order approximation of the parallel transport.

Following (Zhang & Sra, 2016; Zhang et al., 2016), we define the smoothness, convexity and gradient-dominated properties of functions on manifolds.

Definition 1. ***L-g-smooth:*** *A function $f : \mathcal{M} \to \mathbb{R}$ is geodesically L-smooth if for any $w, u \in \mathcal{M}$, $f(u) \leq f(w) + \langle \tilde{\nabla} f(w), R_w^{-1}(u) \rangle + \frac{L}{2}\|R_w^{-1}(u)\|^2$.*

Definition 2. ***μ-g-convex:*** *A function $f : \mathcal{M} \to \mathbb{R}$ is geodesically μ-strongly-convex if for any $w, u \in \mathcal{M}$, $f(u) \leq f(w) + \langle \tilde{\nabla} f(w), R_w^{-1}(u) \rangle + \frac{\mu}{2}\|R_w^{-1}(u)\|^2$.*

Definition 3. ***τ-gradient-dominated:*** *A function $f : \mathcal{M} \to \mathbb{R}$ is τ-gradient-dominated if for any $w \in \mathcal{M}$, $f(w) - f(w^*) \leq \tau\|\tilde{\nabla} f(w)\|^2$, where w^* is the global minimizer.*

2.1. Riemannian Eigensolver

Due to the unit ℓ_2 norm constraint in (2), w is embedded on the Riemannian manifold, hypersphere. In the rest of this paper, we refer to $\mathcal{M}$ as the hypersphere manifold. On the hypersphere manifold, the projection is defined as $P_w(\xi) = (I - ww^\top)\xi$, and the Riemannian gradient of $\hat{F}(w)$ in (2) is obtained via the projection of the Euclidean gradient $\nabla \hat{F}(w)$ as $\tilde{\nabla} \hat{F}(w) = P_w(\nabla \hat{F}(w)) = -(I - ww^\top)\hat{A}w$. The vector transport is defined as $\mathcal{T}_w^u(\xi) = P_u(\xi)$. And the definition of other Riemannian operators can be found in Appendix A.

We then introduce the smoothness and the gradient-dominated properties of the objective function $\hat{F}(w)$ in (2) on the hypersphere manifold.

Lemma 1. *$\hat{F}(w)$ is geodesically λ-smooth.*

Lemma 2. *For any $w \in \mathcal{B} = \{w : w \in \mathcal{M}, (w^\top v)^2 > 0\}$, $\hat{F}(w)$ is $\frac{2}{\delta}$-gradient-dominated, where δ is the eigengap and v is the leading eigenvector.*

The proof of Lemmas can be found in Appendix B.

3. Algorithm

Our proposed algorithm CEDRE is summarized in Algorithm 1. Each global iteration consists of two main stages: 1) communication round (Steps 4-6 and 14-17) and 2) local computation round (Steps 7-13). In each communication round, the master server communicates with all local machines to update the global variable $\tilde{w}^s$ and its corresponding full gradient $\tilde{\nabla} \hat{F}(\tilde{w}^s)$.

Between two communication rounds is a local computation round. In each computation round, each local machine independently updates its local variable based on its local information, including local data and the information received from the master server in the latest communication round. The local variable is then updated by a surrogate gradient as described in Step 10. To construct this surrogate gradient, it is required to pass through all local data, which is computationally expensive when n is large. In practice, it is preferred to update the variables using the mini-batch stochastic gradient, which is computationally flexible, and the computation of g_t becomes

$$g_t = \frac{1}{B}\sum_{i \in \mathcal{I}_t} \tilde{\nabla} f_{k,i}(w_t^{s,k}) - \mathcal{T}_{\tilde{w}_s}^{w_t^{s,k}}\Big(\frac{1}{B}\sum_{i \in \mathcal{I}_t} \tilde{\nabla} f_{k,i}(\tilde{w}_s) - \tilde{\nabla} \hat{F}(\tilde{w}_s)\Big), \quad (3)$$

where $\mathcal{I}_t \subset [n]$ is a stochastic mini-batch set with size B and $f_{k,i}(w) = -\frac{1}{2} w^\top x_i^k {x_i^k}^\top w$.

In Algorithm 1, the update of the global variable (Step 15) and the choice of output (Step 19) are both by random selection. This is a common technique for analyzing nonconvex objective functions (Zhang et al., 2016; Allen-Zhu, 2017; Reddi et al., 2016). However, in practice, we can generate the output using the latest global variable $\tilde{w}_S$; and update global variable by sign-fixed averaging (Option II in Step 16) as:

$$\tilde{w}_{s+1} = \frac{\sum_{k=1}^K \text{sign}({w_m^{s,k}}^\top w_m^{s,1}) w_m^{s,k}}{\|\sum_{k=1}^K \text{sign}({w_m^{s,k}}^\top w_m^{s,1}) w_m^{s,k}\|}, \quad (4)$$

or by geodesically averaging (Option III in Step 17) as:

$$\tilde{w}_{s+1} = \bar{w}_K, \quad (5)$$

where $\bar{w}_1 = w_m^{s,1}$ and $\bar{w}_k = R_{\bar{w}_{k-1}}(\frac{1}{k} R_{\bar{w}_{k-1}}^{-1}(w_m^{s,k}))$ for $k = 2, 3, ..., K$. Detailed discussion about the update strategies can be found in Section 4.2 and Section 6.4.

4. Convergence Analysis

To analyze the convergence of Algorithm 1, we start by introducing the bounds of A_k and g_t defined in Algorithm 1. All the proofs of Lemmas and Theorems can be found in Appendix B and C.

Lemma 3. *Assume the data instances in every local machine are i.i.d., and sampled from some unknown distribution with squared ℓ_2 norm at most b. Then for each local machine k, with probability at least $1 - p$, we have $\|A_k - \hat{A}\|_2^2 \leq \alpha^2 b^2$, where $\alpha^2 = \frac{32log(d/p)}{n}$, and $p \in (0, 1)$.*

Lemma 3 is a corollary of matrix Hoeffding's inequality (Tropp, 2012). When the data are not i.i.d., we can still have the same upper bound of $\|A_k - \hat{A}\|_2^2$ by applying the without-replacement version of Bernstein's inequality for matrices (Gross & Nesme, 2010) if the data are randomly partitioned over all local machines. Based on the spectral norm bound of $\|A_k - \hat{A}\|_2^2$, we derive the bound for g_t defined in Step 10 of Algorithm 1. To avoid confusion with (3), we redefine it as

$$G(w) = \tilde{\nabla} f_k(w) - \mathcal{T}_{\tilde{w}_s}^w\Big(\tilde{\nabla} f_k(\tilde{w}_s) - \tilde{\nabla} \hat{F}(\tilde{w}_s)\Big). \quad (6)$$

Algorithm 1 CEDRE(w_0, S, m, η)

Input: initial variable w_0, global iteration length S, local iteration length m, learning rate η
Initialize $\tilde{w}_0 = w_0$
for $s = 0, 1, 2, ..., S-1$ **do**
  Broadcast $\tilde{w}_s$ to all machines
  Aggregate local gradient $\tilde{\nabla} f_k(\tilde{w}_s)$ from all machines
  Broadcast full gradient $\tilde{\nabla} F(\tilde{w}_s) = \frac{1}{K}\sum_{k=1}^{K} \tilde{\nabla} f_k(\tilde{w}_s)$ to all machines
  for local machine $k = 1, 2, ..., K$ in parallel **do**
    Set $w_0^{s,k} = \tilde{w}_s$
    for $t = 0, 1, 2, ..., m-1$ **do**
      $g_t = \tilde{\nabla} f_k(w_t^{s,k}) - \mathcal{T}_{\tilde{w}_s}^{w_t^{s,k}}\big(\tilde{\nabla} f_k(\tilde{w}_s) - \tilde{\nabla}\hat{F}(\tilde{w}_s)\big)$
      $w_{t+1}^{s,k} = R_{w_t^{s,k}}(-\eta g_t)$
    end for
  end for
  Aggregate local variable $w_m^{s,k}$ from all machines
  option I: $\tilde{w}_{s+1} = w_m^{s,k}$ for randomly chosen $k \in \{1, 2, ..., K\}$
  option II: Update $\tilde{w}_{s+1}$ by sign-fixed averaging as in (4).
  option III: Update $\tilde{w}_{s+1}$ by geodesically averaging as in (5).
end for
Output: w_a is chosen uniformly randomly from $\big\{\big\{\{w_t^{s,k}\}_{t=0}^{m-1}\big\}_{k=1}^{K}\big\}_{s=0}^{S-1}$

Lemma 4. *For $G(w)$ defined in (6), given any $w \in \mathcal{M}$, it holds with probability at least $1-p$ that $\|G(w) - \tilde{\nabla}\hat{F}(w)\|^2 \le 6\alpha^2 b^2\, D^2(w, \tilde{w}_s)$, where $\alpha^2 = \frac{32log(d/p)}{n}$.*

Lemma 4 describes the key difference of the update rule between the distributed setting and the single machine setting. In the single machine setting, $f_k(w) = \hat{F}(w)$ and we have $G(w) = \tilde{\nabla}\hat{F}(w)$. In this case, the variable is updated by Riemannian GD. However, in distributed setting, $G(w)$ is a biased estimation of the global full $\tilde{\nabla}\hat{F}(w)$, and the bias becomes large with the increment of distance between $\tilde{w}_s$ and w. To bound the bias, it requires careful control over the distance between $\tilde{w}_s$ and w by choosing proper step size η and inner iteration length m.

Based on the observation in Lemma 4, we derive the following theorems for convergence analysis of CEDRE.

Theorem 1. *Consider Algorithm 1 with option I. Set the step size $\eta = \rho/\lambda$ with ρ satisfying $12\alpha^2 b^2 \rho^2 m^2/\lambda^2 + \rho \le 1$, where α is defined in Lemma 3. Then for the output w_a of Algorithm 1, it holds with probability at least $1-p$ that*

$$\mathbb{E}[\|\tilde{\nabla}\hat{F}(w_a)\|^2] \le \frac{2\lambda}{\rho m S}\big(\hat{F}(w_0) - \hat{F}(w^*)\big),$$

where $w^* = v$ is the global minimizer of $\hat{F}(w)$.

Corollary 1. *Set $\rho = 1/4$ and $m = \lfloor \lambda/\alpha b \rfloor$ in Theorem 1. Then the output of the proposed algorithm converges to the leading eigenvector with a rate of $\mathcal{O}(\lambda/mS)$. That is it takes $\mathcal{O}(\alpha b/\epsilon) = \mathcal{O}(\frac{b}{\epsilon\sqrt{n}})$ communication rounds to obtain an ϵ-accurate solution.*

Corollary 1 demonstrates that the convergence rate of Algorithm 1 is sublinear. With the same parameters setting in Corollary 1, a larger size of local data set indicates less communication cost to converge to the same accuracy of solution due to the factor $n^{-1/2}$. When the local variable is updated by mini-batch stochastic gradient shown in (3), its convergence analysis is studied in the following theorem.

Algorithm 2 RST-CEDRE($\hat{w}_0, R, S, m, \eta$)

Input: initial variable w_0, global iteration length S, local iteration length m, learning rate η and restart iteration length R
Initialize $\hat{w}_0 = w_0$
for $r = 0, 1, 2, ..., R-1$ **do**
  $\hat{w}_{r+1} = \text{CEDRE}(\hat{w}_r, S, m, \eta)$
end for
Output: $\hat{w}_R$

Theorem 2. *Consider Algorithm 1 with option I, replace the computation of g_t in Step 10 with (3), and set the step size $\eta = \rho/\lambda$ with ρ satisfying $12(\alpha^2 + 1/B) b^2 \rho^2 m^2/\lambda^2 + \rho \le 1$, where α is defined in Lemma 3. Then for the output w_a of Algorithm 1, it holds with probability at least $1-p$ that $\mathbb{E}[\|\tilde{\nabla}\hat{F}(w_a)\|^2] \le 2\lambda/\rho m S\big(\hat{F}(w_0) - \hat{F}(w^*)\big)$, where $w^* = v$ is the global minimizer of $\hat{F}(w)$.*

Theorem 2 tells that even the local variables are updated by stochastic local first-order information, the algorithm still achieves sublinear convergence rate of $\mathcal{O}(\frac{\lambda}{mS})$ with proper setting of parameters. The difference is that it admits a smaller step size η or fewer local computation iterations m.

The convergence analysis presented in Theorems 1-2 is eigengap-free but sublinear. In the next theorem, we present that with a restart strategy as summarized in Algorithm 2, the algorithm enjoys an eigengap-dependent linear convergence rate. The analysis is based on the gradient-dominated property of the objective function in (2). For simplification,

we assume the initial variable w_0 satisfies $(v^\top w_0)^2 > 0$. Based on Theorem 3 and Lemma 5, the variable generally approaches v. Therefore, $(v^\top w)^2$ is always positive during update and $\hat{F}(w)$ is always $\frac{2}{\delta}$-gradient-dominated.

Theorem 3. *Consider Algorithm 2 plugged with CEDRE using option I. Assume the eigengap of $\hat{A}$ is δ and set the step size $\eta = \rho/\lambda$ with ρ satisfying $12\alpha^2 b^2 \rho^2 m^2/\lambda^2 + \rho \leq 1$, where α is defined in Lemma 3. Then given any initial variable w_0 satisfying $(v^\top w_0)^2 > 0$, for the output $\hat{w}_R$ of Algorithm 2, it holds with probability at least $1-p$ that*

$$\mathbb{E}\big[\hat{F}(\hat{w}_R) - \hat{F}(w^*)\big] \leq \big(\tfrac{4\lambda}{\rho\delta m S}\big)^R \big(\hat{F}(w_0) - \hat{F}(w^*)\big),$$

where $w^* = v$ is the global minimizer of $\hat{F}(w)$.

Corollary 2. *Set $\rho = 1/4$ and $m = \lfloor \lambda/\alpha b \rfloor$ and $S = \lceil 32\alpha b/\delta \rceil$ in Theorem 3. We then have $\mathbb{E}\big[\hat{F}(\hat{w}_R) - \hat{F}(w^*)\big] \leq 2^{-R}\big(\hat{F}(w_0) - \hat{F}(w^*)\big)$. And it takes $\mathcal{O}\big(\alpha b/\delta \log(1/\epsilon)\big) = \mathcal{O}\big(\frac{b}{\delta\sqrt{n}} \log(1/\epsilon)\big)$ communication rounds to obtain an ϵ-accurate solution.*

The analysis in Collory 2 presents that even when δ is as small as $\delta = \Omega(1/\sqrt{Kn})$, the communication cost does not increase with the sample size. For Algorithm 2 plugged with CEDRE using mini-batch stochastic update, its analysis is similar to Theorem 3, but with a condition on ρ as presented in Theorem 2.

4.1. Another Accuracy Measure

In the above convergence analysis of CEDRE, the optimization accuracy is measured by either the squared norm of the gradient (as in Theorems 1-2) or the difference from the optimal function value (as in Theorem 3). For the leading eigenvector problem studied in Euclidean space (Shamir, 2015; 2016; Xu et al., 2017; Garber et al., 2017), the optimization accuracy is often measured by the squared sine distance, $1-(w^\top v)^2$, which is the squared sine value of the angle between w and the optimal solution v. Since the sign of w does not affect the value of $\hat{F}(w)$, we assume $w^\top u \geq 0$ is satisfied by default for any $w, u \in \mathcal{M}$ in the discussion of this paper. Then the accuracy measure used in Theorem 3, $\hat{F}(w) - \hat{F}(v)$, can be transformed to the accuracy measure $1-(w^\top v)^2$ based on the following lemma.

Lemma 5. *For any vector $w \in \mathcal{M}$, it holds that*

$$\tfrac{\delta}{2}(1-(w^\top v)^2) \leq \hat{F}(w) - \hat{F}(v) \leq \tfrac{\lambda}{2}(1-(w^\top v)^2).$$

With Lemma 5 and Theorem 3, we have

$$\mathbb{E}[1-(v^\top \hat{w}_R)^2] \leq \Big(\tfrac{4\lambda}{\rho\delta m S}\Big)^R \tfrac{\lambda}{\delta}\, \mathbb{E}[1-(v^\top w_0)^2].$$

Applying the same settings in Corollary 2, it takes $\mathcal{O}\big(\frac{b}{\delta\sqrt{n}} \log(\frac{\lambda}{\delta\epsilon})\big)$ communication rounds to obtain an ϵ-accurate solution if using $1-(w^\top v)^2$ as the accuracy measure.

4.2. The Averaging Strategy

In the aforementioned convergence analysis of Algorithm 1, the global variable is assumed to be updated by random selection (option I). In this part, we discuss the convergence analysis with another two averaging strategies. As will be presented in Lemma 6, if the global variable is updated by sign-fixed averaging in (4), the optimization accuracy, if measured by the squared sine distance to the optimal solution, does not decrease after averaging.

Lemma 6. *Given K variables $w_k \in \mathcal{M}$ which are close enough to the leading eigenvector v of A such that $1-(v^\top w_k)^2 < 1/2$ for $k = 1, 2, ..., K$. Consider the following unit norm vector (averaging with sign-fixed),*

$$\bar{w} = \frac{\sum_{k=1}^K sign(w_k^\top w_1) w_k}{\|\sum_{k=1}^K sign(w_k^\top w_1) w_k\|},$$

we then have $1-(v^\top \bar{w})^2 \leq \frac{1}{K}\sum_{k=1}^K \big\{1-(v^\top w_k)^2\big\}$.

Theorem 4. *Consider Algorithm 2 plugged with CEDRE using option II and set the parameters the same as presented in Theorem 3. Then given any initial variable w_0 satisfying $(v^\top w_0)^2 > 1/2$, for the output $\hat{w}_R$ of Algorithm 2, it holds with probability at least $1-p$ that*

$$\mathbb{E}\big[\hat{F}(\hat{w}_R) - \hat{F}(w^*)\big] \leq \Big(\tfrac{4\lambda^2}{\rho\delta^2 m S}\Big)^R \big(\hat{F}(w_0) - \hat{F}(w^*)\big).$$

The geodesically averaging presented (5) is a geodesically convex combination of K variables. If the objective function is g-convex and the global variable is updated by geodesically averaging (option III), the objective value will not increase after averaging. As studied in (Shamir, 2016), if the objective function in (2) is formulated as a negative Rayleigh quotient problem in the Euclidean space, it is convex within a small area around the optima. However, the convexity of $\hat{F}(w)$ has not been studied on Riemannian manifolds. Here, we present the locally geodesical convexity of $\hat{F}(w)$ in Lemma 7 and derive the convergence analysis of CEDRE with option III in Theorem 5.

Lemma 7. *For any $w \in \mathcal{A} = \{w \in \mathcal{M} : (w^\top v)^2 \geq 1 - \frac{\delta}{\lambda+\delta}\}$, $\hat{F}(w)$ is geodesically-convex.*

Theorem 5. *Consider Algorithm 2 plugged with CEDRE using option III and set the parameters the same as presented in Theorem 3. Then given any initial variable $w_0 \in \mathcal{A}$, for the output $\hat{w}_R$ of Algorithm 2, it holds with probability at least $1-p$ that*

$$\mathbb{E}\big[\hat{F}(\hat{w}_R) - \hat{F}(w^*)\big] \leq \Big(\tfrac{4\lambda}{\rho\delta m S}\Big)^R \big(\hat{F}(w_0) - \hat{F}(w^*)\big).$$

The analysis depends on a high-accurate initialization of w_0, which might be achieved via a warm start.

4.3. Warm Start

To accelerate the convergence of the algorithm, we can initialize the algorithm with the leading eigenvector of a local covariance matrix, e.g., A_1, if the local data size n is sufficiently large. The theoretic analysis of the warm start is presented in Lemma 8.

Lemma 8. *Define the leading eigenvector of A_1 as v_1. We then have $\hat{F}(v_1) - \hat{F}(v) \leq \alpha b = \mathcal{O}(n^{-1/2})$.*

Assume we run Algorithm 2 with parameters settings as in Corollary 2 and set $w_0 = v_1$. Then based on the observation in Lemma 8, we have that it takes $\mathcal{O}\big(\alpha b/\delta\big)\log\big(\alpha b/\epsilon\big)\big)$ communication rounds to achieve an ϵ-accurate solution. Moreover, by combining Lemma 8 and Lemma 5, we have $(v^\top v_1)^2 \geq 1 - 2\alpha b/\delta$. For a sufficiently large n, it is with high probability that $2\alpha b/\delta < 1/2$ and therefore the assumption $1 - (v^\top w_0)^2 < 1/2$ in Lemma 6 is satisfied.

5. Related Work

Besides the distributed PCA algorithms (Garber et al., 2017; Fan et al., 2017; Liang et al., 2014; Boutsidis et al., 2016) discussed in Section 1, recently an extension of distributed power method for sparse PCA is proposed by Ge et al. (2018). Different from our algorithm, their focus is privacy preservation. Without considering privacy preservation, its communication efficiency is not better than the distributed power method. Another way to develop distributed PCA algorithms is to apply existing communication-efficient Riemannian algorithms to solve (2). To the best of our knowledge, there exists only one communication-efficient Riemannian algorithm, named Iterative Local Estimation Algorithm (ILEA) (Saparbayeva et al., 2018). It approximates the global objective function in one local machine with first-order global information and higher-order local information, and optimizes the surrogate function locally. However, the PCA problem may not satisfy the assumptions of ILEA, e.g., strongly geodesically convexity around the optimal point and high-order moment bounds. Thus, whether ILEA is applicable to PCA is still under exploration.

6. Experiments

6.1. Datasets and Settings

We implement CEDRE with a manifold optimization toolbox manopt (Boumal et al., 2014) on a distributed computing platform MATLAB Parallel Server with multiple computers. The empirical performance is evaluated on three real-world datasets with different scales. These datasets are a9a, CIFAR-10 and rcv1. For a9a and CIFAR-10, we only use the training sets. For rcv1, we combine the original training sets and testing sets of rcv1.binary and rcv1.multiclass datasets to construct a large-scale dataset. The details are summarized in Table 1. The data instances are randomly and evenly partitioned over $K = 100$ local machines. And they are normalized to have zero mean by subtracting the mean of all data points. The global mean can be obtained by aggregating the means of local data and then averaging these local means. This requires sending and receiving only one vector for all machines and thus it is not an issue in practice. Besides, for convenience of results comparison across different datasets, the data in three datasets are re-scaled such that their largest eigenvalues are the same. As for the settings of CEDRE, we run the stochastic update, which means Step 10 in Algorithm 1 is replaced by (3) with batch size $B = 1$ and with option II. In addition, we set the local iteration length $m = 5n$ and choose the step size η based on the best training loss with one communication round.

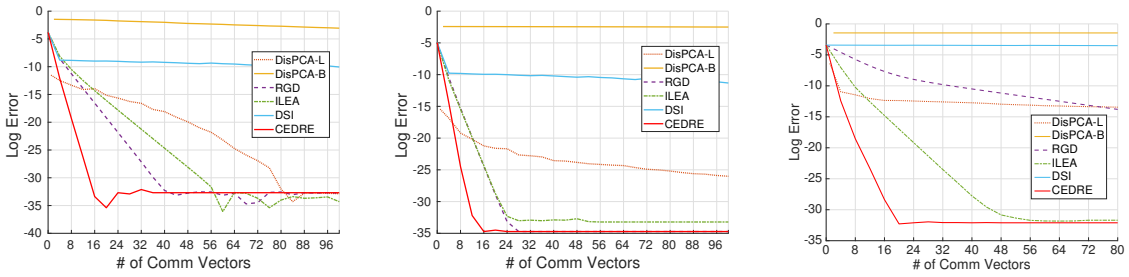
Table 1. Summary of Datasets

Datasets	a9a	CIFAR-10	rcv1
# of Samples (N)	$32,561$	$60,000$	$1,231,776$
# of features (d)	123	3072	$47,236$
Leading Eigenvalue (λ)	1.00	1.00	1.00
Eigengap (δ)	0.3691	0.6152	0.2833

In the experiments, the communication cost is measured by the number of communicated vectors, where the vector is with the same dimension as the data instances. And we count one communicated vector for the following two types of communication: 1) the master server sends one vector to all local machines; and 2) each local machine sends one vector to the master server. Note that for CEDRE, 4 vectors will be communicated in one communication round.

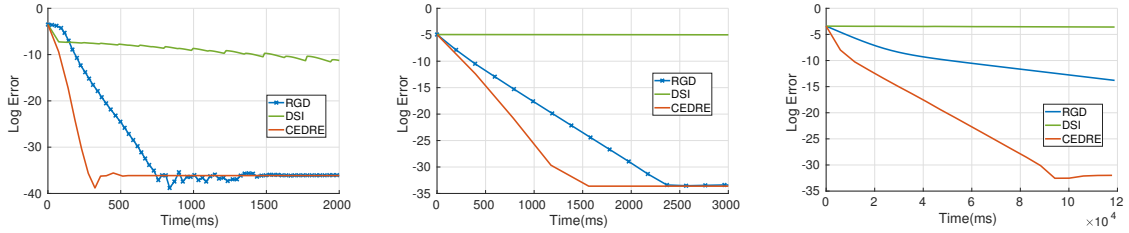
6.2. Comparison on Real-World Datasets

We compare CEDRE with two deterministic distributed PCA algorithms, i.e. (Boutsidis et al., 2016), denoted by DisPCA-B, (Liang et al., 2014), denoted by DisPCA-L, and three iterative-update distributed PCA algorithms, i.e. the distributed implementation of accelerated RGD (Absil et al., 2008), ILEA (Saparbayeva et al., 2018) and DSI (Garber et al., 2017). The distributed power method and Lanczos algorithm are not selected as a baseline because their performance is similar to but a little worse than RGD. The algorithms in (Fan et al., 2017; Liang et al., 2014; Boutsidis et al., 2016) all focus on approximating the local covariance matrix by a small set of vectors and communicating these vectors to master machines to reconstruct the global covariance matrix. (Fan et al., 2017) and (Liang et al., 2014) both apply PCA to construct local covariance matrix approximation, but (Fan et al., 2017) sends the eigenvectors of local covariance matrix while (Liang et al., 2014) sends eigenvectors together with eigenvalues. (Boutsidis et al., 2016) applies random projection to map the local covariance matrix to low dimension space. Since the accuracy of (Fan et al., 2017) decreases with more communication



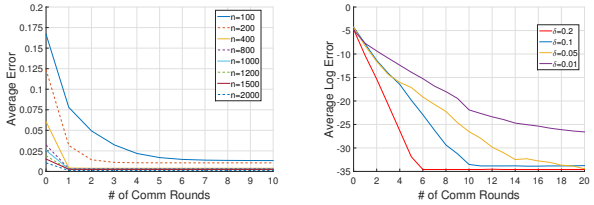
(a) a9a (b) CIFAR-10 (c) rcv1

Figure 1. Communication cost comparison results of different distributed optimization algorithms on real-world datasets.



(a) a9a (b) CIFAR-10 (c) rcv1

Figure 2. Running time comparison results of different distributed optimization algorithms on real-world datasets.



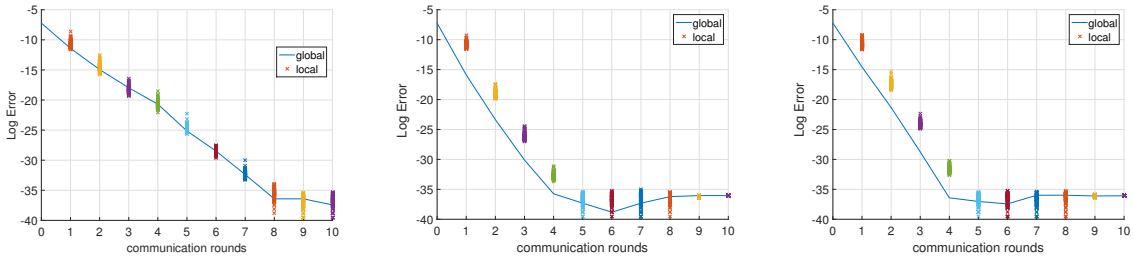
(a) (b)

Figure 3. Results on synthetic datasets. (a) displays convergence of CEDRE regarding different size of dataset in local machines. (b) displays convergence of CEDRE regarding different eigengaps.

rounds, it is not selected as a baseline. To avoid the effect of randomness in (Boutsidis et al., 2016), we run it 100 times and report the averaged results. DSI and ILEA have been introduced.

The comparison results are presented in Figure 1. On all the datasets, the proposed algorithm CEDRE converges to high accuracy with the least number of communicated vectors, which verifies that CEDRE is communication-efficient. And the gain of CEDRE over the competitors becomes larger when the scale of the datasets increases. To be specific, on a9a, CEDRE converges to about -32 log error after 24 vectors are communicated while its best competitor, RGD, takes 66% more communication cost to converges to the same accuracy. On CIFAR-10, CEDRE achieves high accuracy with 16 vectors communicated while the best competitor RGD obtains the same accuracy with 75% more vectors communicated. On rcv1, CEDRE achieves high accuracy after 20 vectors are communicated. The best competitor ILEA achieves the same accuracy with more than 56 vectors communicated, which is 175% more than CEDRE. Specially, with extremely few vectors communicated (i.e. only 4 or fewer vectors are communicated), DisPCA-L achieves better accuracy than CEDRE. But with a few more vectors communicated, the accuracy of CEDRE increases by a large margin while the accuracy of DisPCA-L improves slowly. Moreover, the accuracy improvement of DisPCA-L regarding the increase of communication cost depends on the dimension of data instances. If the number of features is small, the improvement is large as shown on a9a dataset. If the d is large, which is the case of rcv1 dataset, the improvement of accuracy is slow. Due to the uncertainty of random projection, the performance of DisPCA-B is not good. As for DSI, its convergence is slow because it requires high communication cost to solve a series of convex objectives to high accuracy. Compared to CEDRE, RGD does not perform the local update and therefore its convergence is slower than CEDRE, especially when the size of the dataset is large. For example, RGD converges to only -14 log error after communicating 80 vectors on the rcv1 dataset. And for ILEA, it converges slower than CEDRE because the distance to the optimal solution from an exact solution of the local surrogate function may be farther than an early-stop solution of the local surrogate function.

We also compare the running time of CEDRE with distributed RGD and DSI. ILEA is not compared because it fails to complete one iteration within the maximal time we show in the figure. And DisPCA-L, DisPCA-B require to solve a PCA problem on an approximate covariance matrix in the master machine, which is computationally expensive and cannot complete within the maximal time. The results



(a) By random selection (b) By sign-fixed averaging (c) By geodesically averaging

Figure 4. Comparison of different options to update the global variable on a9a dataset. The $\times$ symbols represent the accuracy of variables returned by local machines.

are presented in Figure 2. The results show that CEDRE costs significantly less wall-clock time to achieve a high-accuracy solution.

6.3. Convergence Analysis on Synthetic Datasets

We generate synthetic datasets using normal distribution with zero-mean and specially designed covariance matrix $X = U\Sigma U^\top$ with U being a random $d \times d$ orthonormal matrix, where $d = 1,000$, and Σ being a diagonal matrix satisfying: $\Sigma(1,1) = 1$, $\Sigma(2,2) = 1-\delta$ and $\Sigma(j,j) = 0.9\Sigma(j-1,j-1)$, $\forall j \geq 3$. To reduce the uncertainty effect of random sampling, we generate 10 datasets for each type of synthetic dataset and show the averaged results run on the 10 datasets.

Convergence Regarding Different Sizes of Local Datasets: We first examine the convergence speed of the proposed algorithm CEDRE with different sizes of local datasets n on the synthetic dataset by varying $n = 100$ to $n = 2,000$ with eigengap $\delta = 0.2$. Specially, we define the error by the gap between current function value to the population optimal function value. The empirical results are displayed in Figure 3(a). When $n \geq 400$, the algorithm converges to a low-error solution almost after only 1 round of communication. When $n = 200$, the algorithm converges after about 3 communication rounds. And when $n = 100$, it converges after about 7 rounds. These results show that the number of communication rounds it takes to obtain a high accuracy solution is approximately polynomial to $1/\sqrt{n}$, which meets the theoretic analysis in Theorem 1 and Theorem 3 that the number of communication rounds is linear to λ/m, where $m = 5n$ in our experiments. Moreover, we can observe from Figure 3(a) that the error after convergence decreases with n increases. This meets the theoretic analysis that the error of ERM optimal solution to the population risk minimal solution is $\Omega(1/\sqrt{n})$.

Convergence Regarding Different Eigengaps: We then test the convergence speed of CEDRE on the synthetic dataset with different eigengaps. Based on the analysis in Theorem 3 and Corollary 2, the number of communication rounds to obtain high accuracy solution is linear to $1/\delta$. By varying the eigengap from $\delta = 0.01$ to $\delta = 0.2$ on the synthetic dataset with $n = 2,000$, we observe from Figure 3(b) that the convergence of CEDRE becomes slow with the decrease of eigengap δ. To be specific, when $\delta = 0.2$, CEDRE achieves -25 log error in 4 communication rounds and converges after 6 communication rounds. When $\delta = 0.01$, CEDRE uses 14 communication rounds to obtain -25 log error, which converges much slower. The empirical results meet the theoretic analysis in Corollary 2 that the communication cost of CEDRE to converge to the same accuracy increases when the eigengap δ becomes small.

Extra experimental results can be found in Appendix D.

6.4. Averaging Strategy

In this section, we compare the performance using three different options to update the global variable (Steps 15-17 in Algorithm 1) on a9a. The results in Figure 4 show that the losses of global variables generated by three options are not worse than the largest loss among all local variables. Specifically, the global variable obtained by sign-fixed averaging or by geodesically averaging may achieve a smaller loss than all the local variables (as shown in Figures 4(b) and 4(c)). Therefore, with these two options (Options II and III), CEDRE converges faster than that with random selection (Option I). Besides, the sign-fixed averaging slightly outperforms the geodesically averaging.

7. Conclusion

We propose a Communication-Efficient Distributed Riemannian Eigensolver (CEDRE) algorithm, which converges to the optimal ERM solution with a linear rate regarding the number of communication rounds. The theoretic analysis shows CEDRE is more communication-efficient than the distributed PCA algorithms in previous works. The numerical experiments on real-world and synthetic datasets verified that CEDRE achieves competitive performance to the existing communication-efficient distributed eigensolvers.

Acknowledgements

This work is supported by NTU Singapore Nanyang Assistant Professorship (NAP) grant M4081532.020, and Singapore MOE AcRF Tier-2 grant MOE2016-T2-2-06.

References

Absil, P.-A., Mahony, R., and Sepulchre, R. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

Allen-Zhu, Z. Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 89–97. JMLR. org, 2017.

Bishop, C. M. *Pattern recognition and machine learning*. springer, 2006.

Bonnabel, S. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58 (9):2217–2229, 2013.

Boumal, N., Mishra, B., Absil, P.-A., and Sepulchre, R. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15:1455–1459, 2014. URL http://www.manopt.org.

Boutsidis, C., Woodruff, D. P., and Zhong, P. Optimal principal component analysis in distributed and streaming models. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pp. 236–249, 2016.

Fan, J., Wang, D., Wang, K., and Zhu, Z. Distributed estimation of principal eigenspaces. *arXiv preprint arXiv:1702.06488*, 2017.

Garber, D., Shamir, O., and Srebro, N. Communication-efficient algorithms for distributed stochastic principal component analysis. In *International Conference on Machine Learning*, pp. 1203–1212, 2017.

Ge, J., Wang, Z., Wang, M., and Liu, H. Minimax-optimal privacy-preserving sparse pca in distributed systems. In *International Conference on Artificial Intelligence and Statistics*, pp. 1589–1598, 2018.

Golub, G. H. and Van Loan, C. F. *Matrix computations*, volume 3. Johns Hopkins University Press, 1996.

Gross, D. and Nesme, V. Note on sampling without replacing from a finite collection of matrices. *arXiv preprint arXiv:1001.2738*, 2010.

Hotelling, H. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

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

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

Kannan, R., Vempala, S., and Woodruff, D. Principal component analysis and higher correlations for distributed data. In *Conference on Learning Theory*, pp. 1040–1057, 2014.

Liang, Y., Balcan, M.-F. F., Kanchanapally, V., and Woodruff, D. Improved distributed principal component analysis. In *Advances in Neural Information Processing Systems*, pp. 3113–3121, 2014.

Ng, A. Y., Jordan, M. I., and Weiss, Y. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pp. 849–856, 2002.

Oja, E. and Karhunen, J. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of mathematical analysis and applications*, 106(1):69–84, 1985.

Reddi, S. J., Hefny, A., Sra, S., Poczos, B., and Smola, A. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pp. 314–323, 2016.

Saparbayeva, B., Zhang, M., and Lin, L. Communication efficient parallel algorithms for optimization on manifolds. In *Advances in Neural Information Processing Systems*, pp. 3578–3588, 2018.

Shamir, O. A stochastic pca and svd algorithm with an exponential convergence rate. In *International Conference on Machine Learning*, pp. 144–152, 2015.

Shamir, O. Fast stochastic algorithms for svd and pca: Convergence properties and convexity. In *International Conference on Machine Learning*, pp. 248–256, 2016.

Shamir, O., Srebro, N., and Zhang, T. Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pp. 1000–1008, 2014.

Tropp, J. A. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12 (4):389–434, 2012.

Von Luxburg, U. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Wilkinson, J. H. *The algebraic eigenvalue problem*, volume 87. Clarendon Press Oxford, 1965.

Xu, Z., Ke, Y., and Gao, X. A fast stochastic riemannian eigensolver. In *Conference on Uncertainty in Artificial Intelligence*, 2017.

Zhang, H. and Sra, S. First-order methods for geodesically convex optimization. In *Conference on Learning Theory*, pp. 1617–1638, 2016.

Zhang, H., Reddi, S. J., and Sra, S. Riemannian svrg: fast stochastic optimization on riemannian manifolds. In *Advances in Neural Information Processing Systems*, pp. 4592–4600, 2016.

Zhang, Y. and Lin, X. Disco: Distributed optimization for self-concordant empirical loss. In *International conference on machine learning*, pp. 362–370, 2015.