Journal of the American Statistical Association

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uasa20

Distributed Estimation for Principal Component Analysis: An Enlarged Eigenspace Analysis

Xi Chen, Jason D. Lee, He Li & Yun Yang

To cite this article: Xi Chen, Jason D. Lee, He Li & Yun Yang (2021): Distributed Estimation for Principal Component Analysis: An Enlarged Eigenspace Analysis, Journal of the American Statistical Association, DOI: 10.1080/01621459.2021.1886937

To link to this article: https://doi.org/10.1080/01621459.2021.1886937

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION
2021, VOL. 00, NO. 0, 1–12: Theory and Methods
https://doi.org/10.1080/01621459.2021.1886937

Distributed Estimation for Principal Component Analysis: An Enlarged Eigenspace Analysis

Xi Chen[a], Jason D. Lee[b], He Li[a], and Yun Yang[c]

[a]Stern School of Business, New York University, New York, NY; [b]Department of Electrical Engineering, Princeton University, Princeton, NJ; [c]Department of Statistics, University of Illinois Urbana-Champaign, Champaign, IL

ABSTRACT
The growing size of modern datasets brings many challenges to the existing statistical estimation approaches, which calls for new distributed methodologies. This article studies distributed estimation for a fundamental statistical machine learning problem, principal component analysis (PCA). Despite the massive literature on top eigenvector estimation, much less is presented for the top-L-dim ($L > 1$) eigenspace estimation, especially in a distributed manner. We propose a novel multi-round algorithm for constructing top-L-dim eigenspace for distributed data. Our algorithm takes advantage of shift-and-invert preconditioning and convex optimization. Our estimator is communication-efficient and achieves a fast convergence rate. In contrast to the existing divide-and-conquer algorithm, our approach has no restriction on the number of machines. Theoretically, the traditional Davis–Kahan theorem requires the explicit eigengap assumption to estimate the top-L-dim eigenspace. To abandon this eigengap assumption, we consider a new route in our analysis: instead of exactly identifying the top-L-dim eigenspace, we show that our estimator is able to *cover* the targeted top-L-dim population eigenspace. Our distributed algorithm can be applied to a wide range of statistical problems based on PCA, such as principal component regression and single index model. Finally, we provide simulation studies to demonstrate the performance of the proposed distributed estimator.

ARTICLE HISTORY
Received April 2020
Accepted February 2021

KEYWORDS
Convergence analysis; Distributed estimation; Enlarged eigenspace; Principal component analysis; Shift-and-invert preconditioning

1. Introduction

The development of technology has led to the explosive growth in the size of modern datasets. The challenge arises, when memory constraints and computation restrictions make the traditional statistical estimation and inference methods no longer applicable. For example, in a sensor network, the data are collected on each tensor in a distributed manner. The communication cost would be rather high if all the data are transferred and computed on a single (central) machine, and it may be even impossible for the central machine to store and process computation on such large-scale datasets. Distributed statistical approaches have drawn a lot of attentions these days and methods are developed for various statistics problems, such as sparse regression (see, e.g., Lee et al. 2017), likelihood-based inference (see, e.g., Battey et al. 2018; Jordan, Lee, and Yang 2019), kernel ridge regression (Zhang, Duchi, and Wainwright 2015), semiparametric partial linear models (Zhao, Cheng, and Liu 2016), quantile regression (see, e.g., Chen, Liu, and Zhang 2019; Volgushev, Chao, and Cheng 2019; Chen et al. 2020), linear support vector machine (Wang et al. 2019), Newton-type estimator (Chen, Liu, and Zhang 2021), and M-estimators with cubic rate (Shi, Lu, and Song 2018; Banerjee, Durot, and Sen 2019). All these works are seeking for distributed statistical methods that are able to handle massive computation tasks efficiently for large-scale data and achieve the same convergence rate as those classical methods as well.

In a typical distributed environment, each machine has access to a different subset of samples of the whole dataset. The communication and computation follow from a hierarchical master-slave-type architecture, where a central machine acts as a fusion node. Computation tasks for local machines and the central machine are different. After local machines finish their computation, the local results will be transferred to the master machine, where they will be merged together and the fusioned result will be transferred back to all local machines for the next step.

In this article, we study the problem of principal component analysis (PCA) in a distributed environment. PCA (Pearson 1901; Hotelling 1933) is one of the most important and fundamental tools in statistical machine learning. For random vectors $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n$ in $\mathbb{R}^d$ with mean zero and covariance matrix $\boldsymbol{\Sigma}$, its empirical covariance matrix is $\widehat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{a}_i\boldsymbol{a}_i^\top$. The L-PCA ($L \le d$) finds a L-dimension subspace projection that preserves the most variation in the dataset, which is equivalent to the following optimization problem:

$$\max_{\boldsymbol{U}\in\mathbb{R}^{d\times L}:\boldsymbol{U}^T\boldsymbol{U}=\boldsymbol{I}_L} \left\|\widehat{\boldsymbol{\Sigma}}\boldsymbol{U}\right\|_{\mathrm{F}}, \tag{1}$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the matrix Frobenius norm and $\boldsymbol{I}_L$ is the $L\times L$ identity matrix. In other words, $\boldsymbol{U} \in \mathbb{R}^{d\times L}$ is the top-L-dim eigenspace of $\widehat{\boldsymbol{\Sigma}}$. PCA has been widely used in many aspects of statistical machine learning, for example, principal component regression (Jeffers 1967; Jolliffe 1982), single index model (Li

CONTACT Xi Chen xc13@stern.nyu.edu Stern School of Business, New York University, New York, NY 10012.
Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/JASA.

1992), representation learning (Bengio, Courville, and Vincent 2013).

Under distributed regime, Fan et al. (2019) proposed a novel one-shot type of algorithm which is often called divide-and-conquer (DC) method. In Fan et al. (2019), DC method first computes local covariance matrices $\widehat{\boldsymbol{\Sigma}}_i$ on each machine $k = 1, \dots, K$. Eigenspaces $\widehat{\boldsymbol{U}}_k, k = 1, \dots, K$ are then computed locally using the traditional PCA algorithm and transmitted to the central machine. Central machine combines local eigenspaces $\widehat{\boldsymbol{U}}_k$ into an aggregated covariance estimator, $\widetilde{\boldsymbol{\Sigma}} = \frac{1}{K}\sum_{k=1}^{K} \widehat{\boldsymbol{U}}_k \widehat{\boldsymbol{U}}_k^\top$. The final estimator is obtained as the top-L-dim eigenspace of $\widetilde{\boldsymbol{\Sigma}}$. DC method is easy to implement and requires only $\mathcal{O}(dL)$ communications for each local machine, where d denotes the data dimension, n the total sample size, and m the sample size on each local machine. Let us denote the condition number of the population covariance matrix $\boldsymbol{\Sigma}$ by ρ, that is, $\rho = \lambda_1/(\lambda_L - \lambda_{L+1})$, and the effective rank of $\boldsymbol{\Sigma}$ by $r = \mathrm{Tr}(\boldsymbol{\Sigma})/\lambda_1$. For asymmetric innovation distributions, Fan et al. (2019) showed that when the number of machines is not very large (no greater than $\mathcal{O}(m/(\rho^2 r))$), DC method enjoys an optimal statistical convergence rate of order $\mathcal{O}(\rho\sqrt{Lr/n})$. However, when the number of machines becomes larger, DC method only achieves a slow convergence rate of $\mathcal{O}(\rho\sqrt{Lr/n} + \rho^2\sqrt{Lr/m})$. This feature may not be desirable in distributed settings. For example, in a sensor network with a vast number of sensors, the number of machines may exceed the constraint set for the optimal rate. The precise definition of asymmetric innovation above is given in Section 4.2 of Fan et al. (2019). Roughly speaking, a random variable $\boldsymbol{a} \in \mathbb{R}^d$ is distributed under asymmetric innovation if flipping the sign of one component of $\boldsymbol{a}$ changes its distribution.

One question naturally arises from the analysis of DC method, can we possibly relax the restriction on the number of machines? Motivated by this question, our article presents a multi-round distributed algorithm for top-L-dim eigenspace estimation.

The contribution of our method is 2-fold. First, as compared to DC method in Fan et al. (2019), we completely remove the assumption on the number of machines. Our method leverages shift-and-invert preconditioning (a.k.a., Rayleigh quotient iteration) from numerical analysis (Van Loan and Golub 2012) together with quadratic programming and achieves a fast convergence rate. Moreover, most previous convergence analysis of eigenspace estimation relies on the assumption of an explicit eigengap between the Lth and the $(L+1)$th population eigenvalues λ_L and λ_{L+1}, that is, $\lambda_L - \lambda_{L+1} > 0$, or other specific eigen-structures of $\boldsymbol{\Sigma}$. The second contribution of our article is that we propose an enlarged eigenspace estimator that does not require any eigengap assumption.

In particular, let $\boldsymbol{U}_L$ denote the top-L-dim eigenspace of the population covariance matrix $\boldsymbol{\Sigma}$, and $\widehat{\boldsymbol{U}}_L$ the top-L-dim eigenspace of the empirical covariance $\widehat{\boldsymbol{\Sigma}}$. Estimation consistency of $\widehat{\boldsymbol{U}}_L$ is guaranteed by the (variant of) Davis–Kahan theorem (Davis and Kahan 1970; Yu, Wang, and Samworth 2014): there exists an orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{L\times L}$, such that

$$\left\| \boldsymbol{U}_L - \widehat{\boldsymbol{U}}_L \boldsymbol{Q} \right\|_2 \le \frac{\sqrt{2}\, \| \widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma} \|_2}{\min(|\widehat{\lambda}_{L-1} - \lambda_L|, |\widehat{\lambda}_{L+1} - \lambda_L|)}, \quad (2)$$

where $\|\cdot\|_2$ denotes the matrix spectrum norm. Since the empirical eigenvalue $\widehat{\lambda}_l$ is expected to be concentrated around its population counterpart λ_l for all $l \in [d]$, the consistency of $\widehat{\boldsymbol{U}}_L$ relies on an eigengap condition requiring $\min(\lambda_{L-1} - \lambda_L, \lambda_L - \lambda_{L+1})$ to be strictly away from zero. Unfortunately, without such an eigenvalue gap condition, the top-L-dim subspace $\boldsymbol{U}_L$ is not statistically identifiable and estimation error from $\widehat{\boldsymbol{U}}_L$ can be arbitrarily large (see a counter-example provided in Yu, Wang, and Samworth (2014)). Fortunately, in many statistical applications of PCA such as the principal component regression (see Example 1), it suffices to retrieve the variation captured by the top eigenspace rather than exactly recover the top eigenspace to achieve a small in-sample prediction risk. To address the challenge of no explicit eigengap, we choose a different perspective. In particular, we consider an *an enlarged estimator* $\boldsymbol{V}_{>(1-\delta)\widehat{\lambda}_L}$ (see Equation (3)), where δ is a prespecified constant to quantify the amount of enlargement.

$$\underbrace{\boldsymbol{u}_1, \dots, \boldsymbol{u}_L}_{\boldsymbol{U}_L}, \boldsymbol{u}_{L+1}, \dots, \boldsymbol{u}_S, \boldsymbol{u}_{S+1}, \dots, \boldsymbol{u}_d, \quad (3)$$

$$\underbrace{\boldsymbol{v}_1, \dots, \boldsymbol{v}_L, \boldsymbol{v}_{L+1}, \dots, \boldsymbol{v}_S,}_{\boldsymbol{V}_{>(1-\delta)\widehat{\lambda}_L}} \ \underbrace{\boldsymbol{v}_{S+1}, \dots, \boldsymbol{v}_d}_{\boldsymbol{V}_{\le(1-\delta)\widehat{\lambda}_L}}.$$

Roughly speaking, we prove that our distributed estimator $\boldsymbol{V}_{>(1-\delta)\widehat{\lambda}_L}$ satisfies inequality (2) with the following property: the angle between the target $\boldsymbol{U}_L$ and the complement of our estimator $\boldsymbol{V}_{\le(1-\delta)\widehat{\lambda}_L}$ is sufficiently small (please see Theorem 3.3 for more details). Such a property shows that the enlarged estimator $\boldsymbol{V}_{>(1-\delta)\lambda_L}$ almost cover the $\boldsymbol{U}_L$ even without an eigengap condition.

Our method is motivated by the shift-and-invert preconditioning. The idea of solving PCA via shift-and-invert preconditioning has long history in numerical analysis (Van Loan and Golub 2012). It is an iterative method that sequentially solves linear system to obtain increasingly accurate eigenvector estimates. Its connection with convex optimization has been studied in the past decade. In a single-machine setting, Garber et al. (2016) and Allen-Zhu and Li (2016) formulated each round of shift-and-invert preconditioning as a quadratic optimization problem and it can be solved with first-order deterministic (accelerated) gradient method like Nesterov accelerated method. Garber and Hazan (2015), Shamir (2016), and Xu (2018) also related the same convex optimization problem with variance-reduction stochastic technique (SVRG, see, e.g.„ Johnson and Zhang 2013). Furthermore, in distributed settings, Garber, Shamir, and Srebro (2017) performed a multi-round algorithm but they only consider the estimation task of the first eigenvector. This article proposes a general distributed algorithm that estimates the top-L-dim eigenspace without a restriction on the eigengap.

The proposed algorithm can facilitate many fundamental applications based on PCA in distributed environment. In particular, we illustrate two important applications, namely principal component regression (see Appendix B.1 in the supplementary materials) and single index model (see Appendix B.2 in the supplementary materials).

1.1. Example 1: Principal Component Regression

Introduced by Jeffers (1967) and Jolliffe (1982), principal component regression (PCR) is a regression analysis technique based on PCA. Typically, PCR assumes a linear model $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{\beta}^* + \boldsymbol{\epsilon}$ with the further assumption that coefficient $\boldsymbol{\beta}^*$ lies in the low-rank eigenspace of data covariance matrix. Therefore, PCA can be performed to obtain the principal components $\widehat{\boldsymbol{U}}_L$ of the observed covariance matrix $\widehat{\boldsymbol{\Sigma}} = \frac{1}{n}\boldsymbol{A}^\top\boldsymbol{A}$ and the data matrix $\boldsymbol{A}$ is then projected on $\widehat{\boldsymbol{U}}_L$. The estimator $\widehat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}^*$ is then obtained by regress $\boldsymbol{y}$ on this projected data matrix $\boldsymbol{A}\widehat{\boldsymbol{U}}_L$. Many previous work has analyze the statistical property of PCR, see Frank and Friedman (1993) and Bair et al. (2006). Under a distributed environment, our distributed PCA algorithm can replace the traditional PCA algorithm in the above procedure and lead to a distributed algorithm for PCR. As we will show in Appendix B.1 in the supplementary materials, this distributed estimator achieves a similar error as in the single-machine setting.

1.2. Example 2: Single Index Model

Single index model (Li 1992) considers a semiparametric regression model $y = f(\langle \boldsymbol{\beta}^*, \boldsymbol{a}\rangle) + \epsilon$. Under some mild condition on the link function $f(\cdot)$, we would like to make estimation on the coefficient $\boldsymbol{\beta}^*$ using observed data $\{\boldsymbol{a}_i, y_i\}_{i=1}^n$ without knowing $f(\cdot)$. Some previous methods include semiparametric maximum likelihood estimator (Horowitz 2009) and gradient-based estimator (Hristache, Juditsky, and Spokoiny 2001). Moreover, many works propose to use Stein's identity (Stein 1981; Janzamin, Sedghi, and Anandkumar 2014) to estimate $\boldsymbol{\beta}^*$ (see, e.g., Li 1992; Yang, Balasubramanian, and Liu 2017 and references therein). Specifically, under Gaussian innovation where $\boldsymbol{a}$ is standard multivariate normal random vector, the estimator $\widehat{\boldsymbol{\beta}}$ can be calculated from the top eigenvector of $\frac{1}{n}\sum_{i=1}^n y_i \cdot (\boldsymbol{a}_i\boldsymbol{a}_i^\top - \boldsymbol{I}_d)$. This method can be naturally extended to a distributed manner with a distributed eigen-decomposition of $\frac{1}{n}\sum_{i=1}^n y_i \cdot (\boldsymbol{a}_i\boldsymbol{a}_i^\top - \boldsymbol{I}_d)$.

1.3. Notations

We first introduce the notations related to our work. We write vectors in $\mathbb{R}^d$ in boldface lower-case letters (e.g., $\boldsymbol{a}$), matrices in boldface upper-case letters (e.g., $\boldsymbol{A}$), and scalars are written in lightface letters (e.g., t). Let $\|\cdot\|$ denote vector norm (e.g., $\|\cdot\|_2$ is standard Euclidean norm for vectors). Matrix norm is written as $|||\cdot|||$. For a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times d}$, $|||\boldsymbol{A}|||_2$ and $|||\boldsymbol{A}|||_{\mathrm{F}}$ represent the spectral norm and Frobenius norm, respectively. Furthermore, $\boldsymbol{0}$ represents zero vector with corresponding dimension and identity matrix with dimension $d \times d$ is shortened as $\boldsymbol{I}_d$. We use $\boldsymbol{e}_1, \dots, \boldsymbol{e}_d$ to denote the standard unit vectors in $\mathbb{R}^d$, that is, $\boldsymbol{e}_i = [0, \dots, 0, 1, 0, \dots, 0]$ where only the ith element of $\boldsymbol{e}_i$ is 1.

We use $\mathcal{O}_p$ to describe a high probability bound with constant term omitted. We also use $\widetilde{\mathcal{O}}_p$ to further omit the logarithm factors.

We adopt the standard definition of sub-Gaussian random vectors (see, e.g., Vershynin 2012; Rigollet and Hütter 2015) that a random vector $\boldsymbol{a} \in \mathbb{R}^d$ is said to be a d-dimensional sub-Gaussian with variance proxy σ if $\mathbb{E}[\boldsymbol{a}] = \boldsymbol{0}$ and for any unit vector $\boldsymbol{u}$,

$$\mathbb{E}[\exp(s\boldsymbol{a}^\top\boldsymbol{u})] \le \exp\left(\frac{\sigma^2 s^2}{2}\right),\ \forall s \in \mathbb{R}.$$

1.4. Article Organization

The remainder of this article is organized as follows. In Section 2, we introduce the problem setups of the distributed PCA and give our algorithms. Section 3 develops the convergence analysis of our estimator. Finally, extensive numerical experiments are provided in Section 4. The technical proofs and some additional experimental results are provided in the supplementary materials. We also conduct analysis on two application scenarios, that is, principal component regression and single index model in Appendix B in the supplementary materials where we provide convergence analysis for both single-machine and distributed settings.

2. Problem Setups

In the following section, we collect the setups for our distributed PCA and present the algorithms.

Assume that there are n iid zero mean vectors $\boldsymbol{a}_i$ sampling from some distribution $\mathcal{D}$ in $\mathbb{R}^d$. Let $\boldsymbol{A} = [\boldsymbol{a}_1, \dots, \boldsymbol{a}_n]^\top \in \mathbb{R}^{n\times d}$ be the data matrix. Let $\boldsymbol{\Sigma}$ be the population covariance matrix $\boldsymbol{\Sigma} = \mathbb{E}_{\boldsymbol{a}\sim\mathcal{D}}[\boldsymbol{a}\boldsymbol{a}^\top]$ with the eigenvalues $\lambda_1(\boldsymbol{\Sigma}) \ge \lambda_2(\boldsymbol{\Sigma}) \ge \cdots \lambda_d(\boldsymbol{\Sigma}) \ge 0$ and the associated eigenvectors are $\boldsymbol{U} = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_d] \in \mathbb{R}^{d\times d}$.

In the distributed PCA, for a given number L, $1 \le L \le d$, we are interested in estimating the eigenspace spanned by $\boldsymbol{U}_L := \{\boldsymbol{u}_1, \dots, \boldsymbol{u}_L\}$ in a distributed environment. We assume n samples are split uniformly at random on K machines, where each machine contains m samples, that is, $n = mK$. We note that since our algorithm aggregates gradient information across machines, it can handle the unbalanced data case without any modification. We choose to present the balanced data case only for the ease of presentation (see Remark 3.2 for more details). The data matrix on each machine k is denoted by $\boldsymbol{A}_k \in \mathbb{R}^{m\times d}$ for $k \in [K]$.

Let us first discuss a special case (illustrated in Algorithm 1), where we estimate the top eigenvector, that is, $L = 1$. The basic idea of our Algorithm 1 is as follows.

Let $\boldsymbol{w}^{(0)}$ be the initial estimator of the top eigenvector and $\overline{\lambda}_1$ a crude estimator of an upper bound of the top eigenvalue. Here we propose to compute $\boldsymbol{w}^{(0)}$ and $\overline{\lambda}_1$ only using the data from the first machine, and thus there does not incur any communication cost. For example, $\overline{\lambda}_1$ can be computed with $\overline{\lambda}_1 = \lambda_1(\boldsymbol{A}_1^\top\boldsymbol{A}_1/m) + 3\eta/2$, where $\lambda_1(\boldsymbol{A}_1^\top\boldsymbol{A}_1/m)$ is the top eigenvalue for the empirical covariance matrix on the first machine and η is a special constant defined later in Equation (11). The $\boldsymbol{w}^{(0)}$ can be simply computed via eigenvalue decomposition of $\boldsymbol{A}_1^\top\boldsymbol{A}_1/m$. We note the that Algorithm 1 is almost tuning free. The only parameter in constructing $\overline{\lambda}_1$ is η. According to our theory, we could set $\eta = c_0\sqrt{d/m}$ for some sufficiently large c_0 and the result is not sensitive to c_0. There are other tuning-free ways to obtain a crude top-eigenvalue estimator $\overline{\lambda}_1$ only using the sample on the first machine (e.g., the adaptive Algorithm 1 in Garber et al. (2016) without tuning parameters).

Algorithm 1 Distributed top eigenvector (Distri-Eigen)

```
Input: Data matrix $A_k$ on each machine $k = 1, \ldots, K$. The initial top eigenvalue estimator $\overline{\lambda}_1$ and eigenvector estimator $w^{(0)}$. The number of outer iterations $T$ and the number of inner iterations $T'$.
 1: Distribute $\overline{\lambda}_1$ to each local machine and each local machine computes $\mathbf{H}_k = \overline{\lambda}_1 I - A_k^\top A_k/m$.
 2: for $t = 0, 1, \ldots, (T-1)$ do
 3:   Distribute $w^{(t)}$ to each local machine and each local machine sets $w_0^{(t+1)} = w^{(t)}$
 4:   for $j = 0, 1, \ldots, (T'-1)$ do
 5:     for each local machine $k = 1, \ldots, K$ do
 6:       Compute the local gradient information $g_k = \mathbf{H}_k w_j^{(t+1)} - w^{(t)}$
 7:       Transmit the local gradient information $g_k$ to the central machine.
 8:     end for
 9:     Calculate the global gradient information $g = \frac{1}{K}\sum_{k=1}^K g_k$.
10:     Perform the approximate Newton's step: $w_{j+1}^{(t+1)} = w_j^{(t+1)} - \mathbf{H}_1^{-1} g$.
11:   end for
12:   The central machine updates $w^{(t+1)} = \frac{w_{T'}^{(t+1)}}{\|w_{T'}^{(t+1)}\|_2}$.
13: end for
14: Output: $w^{(T)}$.
```

Given $w^{(0)}$ and $\overline{\lambda}_1$, we perform the *shift-and-invert preconditioning* iteration in a distributed manner. In particular, for each iteration $t = 0, 1, \ldots,$

$$\widetilde{w}^{(t+1)} = \left(\overline{\lambda}_1 I - \frac{1}{n} A^\top A\right)^{-1} w^{(t)}, \quad w^{(t+1)} = \frac{\widetilde{w}^{(t+1)}}{\|\widetilde{w}^{(t+1)}\|_2}. \tag{4}$$

Therefore, the nonconvex eigenvector estimation problem (1) is reduced to solving a sequence of linear system. The key challenge is how to implement $(\overline{\lambda}_1 I - A^\top A/n)^{-1}$ in a distributed setup.

To address this challenge, we formulate (4) into a quadratic optimization problem. In particular, the update $\widetilde{w}^{(t+1)} = (\overline{\lambda}_1 I - A^\top A/n)^{-1} w^{(t)}$ is equivalent to the following problem,

$$\widetilde{w}^{(t+1)} = \arg\min_{w} \left[Q(w) := \frac{1}{2} w^\top \mathbf{H} w - w^\top w^{(t)} \right], \tag{5}$$
$$\mathbf{H} \triangleq \overline{\lambda}_1 I - \frac{1}{n} A^\top A.$$

To solve this quadratic programming, the standard Newton's approach computes a sequence for $j = 0, \ldots,$ with a starting point $w_0^{(t+1)} = w^{(t)}$:

$$w_{j+1}^{(t+1)} = w_j^{(t+1)} - \left(\nabla^2 Q(w_j^{(t+1)})\right)^{-1} \left[\nabla Q(w_j^{(t+1)})\right], \tag{6}$$

where the Hessian matrix $\nabla^2 Q(w_j^{(t+1)})$ is indeed $\mathbf{H}$. If we define, for each machine $k \in [K]$,

$$\mathbf{H}_k = \overline{\lambda}_1 I - \frac{1}{m} A_k^\top A_k, \tag{7}$$
$$Q_k(w) = \frac{1}{2} w^\top \mathbf{H}_k w - w^\top w^{(t)}.$$

It is easy to see that $\mathbf{H} = \sum_{k=1}^K \mathbf{H}_k/K$ and $Q(w) = \sum_{k=1}^K Q_k(w)/K$. Therefore, in the Newton's update (6), computing the full Hessian matrix $\nabla^2 Q(w_j^{(t+1)})$ requires each machine to communicate a $d \times d$ local Hessian matrix $\mathbf{H}_k$ to the central machine. This procedure incurs a lot of communication cost. Moreover, taking the inverse of the whole sample Hessian matrix $\mathbf{H}$ almost solves the original linear system (4). To address this challenge, we adopt the idea from Shamir, Srebro, and Zhang (2014), Jordan, Lee, and Yang (2019), and Fan, Guo, and Wang (2019). In particular, we approximate the Newton's iterates by only using the Hessian information on the *first machine*, which significantly reduces the communication cost. This approximated Newton's update can be written as,

$$\begin{aligned} w_{j+1}^{(t+1)} &= w_j^{(t+1)} - \left(\nabla^2 Q_1(w_j^{(t+1)})\right)^{-1} \left[\nabla Q(w_j^{(t+1)})\right] \\ &= w_j^{(t+1)} - \mathbf{H}_1^{-1} \left[\frac{1}{K} \sum_{k=1}^K (\mathbf{H}_k w_j^{(t+1)} - w^{(t)})\right], \end{aligned} \tag{8}$$

where $\mathbf{H}_1$ is the Hessian matrix of the first machine. This procedure can be computed easily in a distributed manner, that is, each machine computes local gradient $g_k = \mathbf{H}_k w_j^{(t+1)} - w^{(t)}$, and these gradient vectors are communicated to the central machine for a final update $g = \sum_{k=1}^K g_k/K$. Therefore, in each inner iteration, the communication cost for each local nodes is only $\mathcal{O}(d)$. See Algorithm 1 for a complete description.

Remark 2.1. In this remark, we explain why we choose the Newton approach in the inner loop (8), instead of the quasi-Newton method (the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method) or other gradient methods with line search (e.g., Barzilai–Borwein gradient method in Wen and Yin (2013)). Due to the special structure of the PCA problem in our quadratic programming (5), the Hessian matrix is fixed and will not change over iterations. In other words, as shown in Algorithm 1, we compute each local Hessian $\mathbf{H}_k$ for $k = 1, \ldots, m$ (see Equation (7)), and the inverse of the local Hessian on the first machine $\mathbf{H}_1$ only once. Therefore, the distributed Newton method is computationally more efficient for the PCA problem. In comparison, BFGS is often used when the inverse of Hessian matrix $\mathbf{H}$ is hard to compute and changes over iterations, which is not the scenario of our PCA problem. Moreover, as we will show later in Section 3, the Newton method has already achieved a linear convergence rate in the inner loop (see Lemma 3.2), BFGS cannot be faster than that. In fact, although BFGS will eventually achieve a linear convergence rate, it can be quite slow at the very beginning with a crude Hessian inverse estimation.

Remark 2.2. In this remark, we compare the computational and communication costs between our method and the DC approach. Notice that the communication cost of our Algorithm 1 from each local machine is $\mathcal{O}(TT'd)$, where TT' is the total number of iterations. By our theoretical results in Section 3 (see Corollary 3.1), for a targeting error rate ε, we only require T and T' and to be an logarithmic order of $1/\varepsilon$ (i.e., $TT' = \mathcal{O}(\log^2(1/\varepsilon))$). Therefore, the total number of iterations is quite small. While it is more than $\mathcal{O}(d)$ communication cost of the DC approach, it is still considered as a communication efficient

Algorithm 2 Distributed top-L-dim principal subspace

Input: The data matrix $\boldsymbol{A}_k$ on each machine $k = 1, \dots, K$. The number of top-eigenvectors L.

```
1:  Initialize $\boldsymbol{V}_0 = []$, $\boldsymbol{A}_{k,0} = \boldsymbol{A}_k$
2:  for $l = 1, \dots, L$ do
3:    Compute the initial $l$th eigenvalue estimator $\overline{\lambda}_l$ and eigenvector estimator $\boldsymbol{w}_l^{(0)}$.
4:    Call Algorithm 1 with $\{\boldsymbol{A}_{k,l-1}\}_{k=1}^K$ on each local machine to obtain $\boldsymbol{w}_l$ on the central machine.
5:    Project $\boldsymbol{w}_l$ to $\boldsymbol{V}_{l-1}^{\perp}$ by computing $\boldsymbol{v}_l = \frac{(\boldsymbol{I}-\boldsymbol{V}_{l-1}\boldsymbol{V}_{l-1}^\top)\boldsymbol{w}_l}{\|(\boldsymbol{I}-\boldsymbol{V}_{l-1}\boldsymbol{V}_{l-1}^\top)\boldsymbol{w}_l\|_2}$ as the estimated $l$th eigenvector
6:    Update $\boldsymbol{V}_l = [\boldsymbol{V}_{l-1}, \boldsymbol{v}_l]$
7:    Transmit $\boldsymbol{v}_l$ to each local machine.
8:    for each local machine $k = 1, \dots, K$ do
9:      Update the data matrix $\boldsymbol{A}_{k,l} = \boldsymbol{A}_{k,l-1}(\boldsymbol{I} - \boldsymbol{v}_l\boldsymbol{v}_l^\top)$
10:   end for
11: end for
12: Output: $\boldsymbol{V}_L$.
```

protocol. In distributed learning literature (e.g., Jordan, Lee, and Yang 2019), a communication-efficient algorithm usually refers to an algorithm that only transmits an $\mathcal{O}(d)$ vector (instead of $\mathcal{O}(d^2)$ Hessian matrices) at each iteration.

When the full data of n samples can be stored in the memory, the oracle PCA method incurs a computation cost (i.e., runtime) of $\mathcal{O}(nd^2 + d^3)$, where nd^2 is for the computation of the sample covariance matrix and d^3 is for performing the eigendecomposition. In the distributed setting with m samples on each local machine, the DC approach incurs the computation cost of $\mathcal{O}(md^2 + d^3)$ since it is a one-shot algorithm. In comparison, our method incurs the $\mathcal{O}(md^2 + d^3 + TT'd^2)$ computational cost, to achieve the optimal convergence rate. We note that our method incurs one-time computational of the Hessian inverse with $\mathcal{O}(d^3)$ and each iteration only involves the efficient computation of the gradient (i.e., $\mathcal{O}(d^2)$). Therefore, the extra computational overhead over the DC $\mathcal{O}(TT'd^2)$ is a smaller order term in d as compared to $\mathcal{O}(d^3)$. Moreover, the number of iterations TT' is relatively small and thus the extra computation as compared to the DC is rather limited. In practice, one can easily combine two approaches. For example, one can initialize the estimator using the DC method, and further improve its accuracy using our method.

For the top-L-dim eigenspace estimation, we extend a framework from Allen-Zhu and Li (2016) to our distributed settings. In our Algorithm 2, we first compute the leading eigenvector $\boldsymbol{v}_1$ of $\boldsymbol{A}^\top\boldsymbol{A}/n$ in a distributed manner with Algorithm 1. The $\boldsymbol{v}_1$ is then transferred back to local machines and used to right-project data matrix, that is, $\boldsymbol{A}_k(\boldsymbol{I}_d - \boldsymbol{v}_1\boldsymbol{v}_1^\top)$ for $k \in [K]$. The next eigenvector $\boldsymbol{v}_2$ is obtained with these projected data matrices and Algorithm 1. In other words, we estimate the top eigenvector of $(\boldsymbol{I}_d - \boldsymbol{v}_1\boldsymbol{v}_1^\top)\widehat{\boldsymbol{\Sigma}}(\boldsymbol{I}_d - \boldsymbol{v}_1\boldsymbol{v}_1^\top)$ in distributed settings. This procedure is repeated L times until we obtain all the L top eigenvectors $\boldsymbol{V}_L = [\boldsymbol{v}_1, \dots, \boldsymbol{v}_L]$. This deflation technique is quite straight-forward and performs well in our later convergence analysis.

Remark 2.3. Our article, and also the earlier works (Allen-Zhu and Li 2016; Fan et al. 2019) all assume data vectors are centered, that is, zero-mean data vectors $\mathbb{E}[\boldsymbol{a}] = 0$. When the data is noncentered, we could adopt a two stage estimator, where the first stage centralizes the data in a distributed fashion and second stage applies our distributed PCA algorithm. In particular, each local machine k first computes the mean of local samples, that is, $\bar{\boldsymbol{a}}_k = \frac{1}{m_k}\sum_{i\in\mathcal{D}_k}\boldsymbol{a}_i$, where $\mathcal{D}_k$ denotes the sample indices on the k-local machine and $m_k = |\mathcal{D}_k|$. Then each local machine transmits $(\boldsymbol{a}_k, m_k)$ to the center. The center computes their average $\bar{\boldsymbol{a}} = \frac{\sum_{k=1}^K m_k\bar{\boldsymbol{a}}_k}{\sum_{k=1}^K m_k}$, which will be transmitted back to each local machine to center the data (i.e., each sample $\boldsymbol{a}_i$ will be $\boldsymbol{a}_i - \bar{\boldsymbol{a}}$). Given the centralized data, we can directly apply our distributed PCA algorithm. This centralization step only incurs one extra round of communication and each local machine only transmits an $O(d)$ vector to the center (which is the same amount of communication as in our algorithm that transmits the gradient).

3. Theoretical Properties

This section exhibits the theoretical results for our setups in Section 2. The technical proofs will be relegated to the supplementary materials (see Appendix A).

3.1. Distributed Top Eigenvector Estimation

We first investigate the theoretical properties of the top eigenvector estimation in Algorithm 1. Let $\widehat{\boldsymbol{\Sigma}}_k = \boldsymbol{A}_k^\top\boldsymbol{A}_k/m \in \mathbb{R}^{d\times d}$ denote the local sample covariance matrix on machine $k \in [K]$, and $\widehat{\boldsymbol{\Sigma}} = K^{-1}\sum_{k=1}^K\widehat{\boldsymbol{\Sigma}}_k$ the global sample covariance matrix using all data. Let $\widehat{\lambda}_1 \geq \widehat{\lambda}_2 \geq \cdots \geq \widehat{\lambda}_d \geq 0$ and $\widehat{\boldsymbol{u}}_1, \widehat{\boldsymbol{u}}_2, \dots, \widehat{\boldsymbol{u}}_d$ denote the sorted eigenvalues and associated eigenvectors of $\widehat{\boldsymbol{\Sigma}}$. We are interested in quantifying the quality of some estimator $\boldsymbol{w}^{(t)}$. More specifically, we will reserve the letter δ to denote the relative eigenvalue gap threshold, and will measure the closeness between $\boldsymbol{w}^{(t)}$ and the top eigenvector $\widehat{\boldsymbol{u}}_1$ via proving

$$\sum_{l:\,\widehat{\lambda}_l\leq(1-\delta)\,\widehat{\lambda}_1} |\langle\widehat{\boldsymbol{u}}_l, \boldsymbol{w}^{(t)}\rangle|^2 \leq \frac{\varepsilon^2}{\delta^2}, \tag{9}$$

for error $\varepsilon > 0$ and any constant $\delta \in (0, 1)$. In particular, the result (9) is always stronger (modulo constants) than the usual bound

$$\widehat{\theta}^{(t)} := \arccos|\langle\widehat{\boldsymbol{u}}_1, \boldsymbol{w}^{(t)}\rangle| \leq C\varepsilon\frac{\widehat{\lambda}_1}{\widehat{\lambda}_1 - \widehat{\lambda}_2}, \tag{10}$$

that involves the relative gap between the first two eigenvalues of $\widehat{\boldsymbol{\Sigma}}$. Here C is a constant. To see this, we can simply choose $\delta = (\widehat{\lambda}_1 - \widehat{\lambda}_2)/\widehat{\lambda}_1$ in Equation (9). Then $\sin^2\widehat{\theta}^{(t)} = 1 - |\langle\widehat{\boldsymbol{u}}_1, \boldsymbol{w}^{(t)}\rangle|^2 = \sum_{l:\widehat{\lambda}_l\leq(1-\delta)\widehat{\lambda}_1}|\langle\widehat{\boldsymbol{u}}_l, \boldsymbol{w}^{(t)}\rangle|^2 \leq \varepsilon^2/\delta^2$, implying $\widehat{\theta}^{(t)} \leq \arcsin(\varepsilon/\delta) \leq C\varepsilon\widehat{\lambda}_1/(\widehat{\lambda}_1 - \widehat{\lambda}_2)$ for some universal constant $C > 0$. Moreover, it has to be assumed that $\widehat{\lambda}_1 > \widehat{\lambda}_2$ in the usual bound (10), which may not be held in some applications.

From our enlarged eigenspace viewpoint, the result in Equation (9) indicates that the top eigenvector estimator $\boldsymbol{w}^{(t)}$ is almost covered by the span of $\{\widehat{\boldsymbol{u}}_l : \widehat{\lambda}_l > (1-\delta)\widehat{\lambda}_1\}$.

As we will show in the theoretical analysis later, the success of our algorithm relies on the initial values of both eigenvalue and eigenvector. We first clarify our choice of initial eigenvalue estimates.

For the top eigenvector estimation in Algorithm 1, since we have the following high probability bound (see Equation (A.2) in Lemma A.2 in the supplementary appendix for the justification), $\left\|\widehat{\mathbf{\Sigma}} - \widehat{\mathbf{\Sigma}}_1\right\|_2 \leq \eta/2$ for some constant $\eta > 0$. If we choose $\overline{\lambda}_1^{(0)} = \lambda_1(\widehat{\mathbf{\Sigma}}_1) + 3\eta/2$, then it is guaranteed that, $2\eta \geq \overline{\lambda}_1^{(0)} - \widehat{\lambda}_1 \geq \eta$. Lemma A.2 (Equation (A.3) in the supplementary materials) also provides a concentration bound on our initial value of eigenvectors, $\sum_{l:\widehat{\lambda}_l\leq(1-\delta)\widehat{\lambda}_1} \left|\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{w}^{(0)}\rangle\right|^2 \leq 3/4$ with high probability. Here $\{\widehat{\boldsymbol{u}}_l : \widehat{\lambda}_l \leq (1-\delta)\widehat{\lambda}_1\}$ are all the eigenvectors for the full sample covariance matrix $\widehat{\mathbf{\Sigma}}$ whose associated eigenvalues have a relative gap δ from the largest eigenvalue $\widehat{\lambda}_1$ and $\boldsymbol{w}^{(0)}$ is the top eigenvector for the sample covariance matrix on the first machine.

Given our initial estimators $\overline{\lambda}_1^{(0)}$ and $\boldsymbol{w}^{(0)}$, we have the following convergence guarantee for our Algorithm 1. With the above guarantees of initial estimator $\overline{\lambda}_1$ and $\boldsymbol{w}^{(0)}$, our first lemma characterizes the convergence rate of the *outer loop* in Algorithm 1.

Lemma 3.1. Suppose the initial estimator $\overline{\lambda}_1$ satisfies

$$\eta \leq \overline{\lambda}_1 - \widehat{\lambda}_1 \leq 2\eta \quad \text{for some } \eta > 0. \tag{11}$$

For any $\boldsymbol{w} \in \mathbb{R}^d$, and $\boldsymbol{v} \in \mathbb{R}^d$ that satisfies

$$\|\boldsymbol{w}\|_2 = 1, \qquad \sum_{l:\widehat{\lambda}_l\leq(1-\delta)\widehat{\lambda}_1} \left|\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{w}\rangle\right|^2 \leq \frac{3}{4}, \tag{12}$$

and

$$\|\boldsymbol{v} - \mathbf{H}^{-1}\boldsymbol{w}\|_2 \leq \varepsilon \leq (8\eta)^{-1}, \tag{13}$$

and for each index $l = 1, \ldots$ such that $\widehat{\lambda}_l \leq (1-\delta)\widehat{\lambda}_1$, we have

$$\frac{|\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{v}\rangle|}{\|\boldsymbol{v}\|_2} \leq \frac{8\eta}{\delta\widehat{\lambda}_1}\frac{|\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{w}\rangle|}{\|\boldsymbol{w}\|_2} + 8\eta\varepsilon. \tag{14}$$

Moreover, we have

$$\sum_{l:\widehat{\lambda}_l\leq(1-\delta)\widehat{\lambda}_1} \frac{|\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{v}\rangle|^2}{\|\boldsymbol{v}\|_2^2} \leq \frac{128\eta^2}{\delta^2\widehat{\lambda}_1^2} \sum_{l:\widehat{\lambda}_l\leq(1-\delta)\widehat{\lambda}_1} \frac{|\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{w}\rangle|^2}{\|\boldsymbol{w}\|_2^2} + 128\eta^2\varepsilon^2. \tag{15}$$

For the outer loop in our Algorithm 1, $\boldsymbol{w}$ and $\boldsymbol{v}/\|\boldsymbol{v}\|_2$ in Lemma 3.1 can be explained as the tth round and $(t+1)$th round estimators $\boldsymbol{w}^{(t)}$ and $\boldsymbol{w}^{(t+1)}$, respectively. This lemma implies that up to a numerical tolerance ε for inverting $\mathbf{H}$ (Condition (13)), each application of the outer loop reduces the magnitude of the projection of $\boldsymbol{w}^{(t)}$ onto $\widehat{\boldsymbol{u}}_l$ by a factor of $\mathcal{O}\big((\delta\widehat{\lambda}_1)^{-1}\eta\big) \ll 1$ given $\eta \ll 1$ (if we have a good initial estimator of $\widehat{\lambda}_1$ and $\delta\widehat{\lambda}_1 = \mathbf{\Omega}(1)$). Notice that if $\boldsymbol{w}^{(t)}$ satisfies condition (12), our Equation (15) claims that $\boldsymbol{w}^{(t+1)} = \boldsymbol{v}/\|\boldsymbol{v}\|_2$ satisfies Condition (12) as well. This condition is justified if $\boldsymbol{w}^{(0)}$ satisfies Condition (12), which is a conclusion from Lemma A.2 in the supplementary appendix.

Our second lemma characterizes the convergence rate of distributively solving the linear system $\mathbf{H}\boldsymbol{w} = \boldsymbol{w}^{(t)}$ in the *inner loop* of Algorithm 1. Recall that in Equation (4), $\widetilde{\boldsymbol{w}}^{(t+1)} = \mathbf{H}^{-1}\boldsymbol{w}^{(t)}$ denote the exact solution of this linear system.

Lemma 3.2. Suppose the initial estimator $\overline{\lambda}_1$ satisfies

$$\overline{\lambda}_1 - \widehat{\lambda}_1 \geq \eta \geq \frac{1}{2}\left\|\widehat{\mathbf{\Sigma}} - \widehat{\mathbf{\Sigma}}_1\right\|_2.$$

Then for each $j = 0, 1, \ldots, (T'-1)$, we have

$$\|\boldsymbol{w}_{j+1}^{t+1} - \widetilde{\boldsymbol{w}}^{(t+1)}\|_2 \leq \frac{2\left\|\widehat{\mathbf{\Sigma}} - \widehat{\mathbf{\Sigma}}_1\right\|_2}{\eta}\|\boldsymbol{w}_j^{t+1} - \widetilde{\boldsymbol{w}}^{(t+1)}\|_2. \tag{16}$$

Here $\left\|\widehat{\mathbf{\Sigma}} - \widehat{\mathbf{\Sigma}}_1\right\|_2$ on the RHS of (16) is due to the approximation using the Hessian matrix $\mathbf{H}_1$ on the first machine in place of original Hessian matrix $\mathbf{H}$. As we will show later, by standard matrix concentration inequalities, we have $\left\|\widehat{\mathbf{\Sigma}} - \widehat{\mathbf{\Sigma}}_1\right\|_2 = \mathcal{O}\left(\sqrt{d/m}\right)$ with high probability. As a consequence, the inner loop of Algorithm 1 has a contraction rate of order $\mathcal{O}(\eta^{-1}\sqrt{d/m})$, which is inversely proportional to the gap $\overline{\lambda}_1 - \widehat{\lambda}_1$ (due to the condition number of the Hessian $\mathbf{H}$).

Combining these two lemmas, we come to our first main theoretical result for the convergence rate of Algorithm 1.

Theorem 3.1. Let $\kappa := \left\|\widehat{\mathbf{\Sigma}} - \widehat{\mathbf{\Sigma}}_1\right\|_2 = \mathcal{O}_P(\sqrt{d/m})$. Assume

$$2\eta \geq \overline{\lambda}_1 - \widehat{\lambda}_1 \geq \eta \geq \frac{1}{2}\kappa,$$

and the initial eigenvector estimator $\boldsymbol{w}^{(0)}$ satisfies

$$\sum_{l:\widehat{\lambda}_l\leq(1-\delta)\widehat{\lambda}_1} \left|\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{w}^{(0)}\rangle\right|^2 \leq \frac{3}{4}.$$

Then for each T and T' as the outer and inner iterations in Algorithm 1, respectively, and the relative eigenvalue gap $\delta \in (0,1)$, we have

$$\sum_{l:\widehat{\lambda}_l\leq(1-\delta)\widehat{\lambda}_1} |\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{w}^{(t)}\rangle|^2 \leq \left(\frac{128\eta^2}{\delta^2\widehat{\lambda}_1^2}\right)^T + \frac{512\,\eta}{1 - 128\eta^2/(\delta\widehat{\lambda}_1)^2}\left(\frac{4\kappa^2}{\eta^2}\right)^{T'}. \tag{17}$$

We can further simplify Equation (17) by choosing proper η and T'.

Corollary 3.1. In particular, if $\eta \leq \delta\widehat{\lambda}_1/16$, and we choose $T' = T$, and $\eta = \left(\kappa\delta\widehat{\lambda}_1\right)^{1/2}/3 = \mathcal{O}_P(\sqrt[4]{d/m})$, then the final output $\boldsymbol{w}^{(T)}$ satisfies

$$\sum_{l:\widehat{\lambda}_l\leq(1-\delta)\widehat{\lambda}_1} |\langle \widehat{\boldsymbol{u}}_l, \boldsymbol{w}^{(T)}\rangle|^2 \leq 257\left(\frac{6\kappa}{\delta\widehat{\lambda}_1}\right)^{2T}. \tag{18}$$

As indicated in (18), when $6\kappa/\delta\widehat{\lambda}_1 \ll 1$, our Algorithm 1 enjoys a linear convergence rate. Moreover, to ensure this convergence, when the absolute eigengap $\delta\widehat{\lambda}_1$ is small, $\kappa = \mathcal{O}_P(\sqrt{d/m})$ needs to be smaller, that is, $\kappa = o(\delta\widehat{\lambda}_1)$ Recall that $\kappa := \left\|\widehat{\mathbf{\Sigma}} - \widehat{\mathbf{\Sigma}}_1\right\|_2$, which is defined in Theorem 3.1. This indicates that more samples are needed on each local machine.

Remark 3.1. Under the setting without an explicit eigengap, our goal is not to construct a good estimator of the top eigenvector. Instead, we aim to construct an estimator that captures a similar amount of variability in the sample data as the top eigenvector. Recall that by Theorem 3.1, we construct an estimator $\boldsymbol{w}$ such that $\sum_{l:\widehat{\lambda}_l\le(1-\delta)\widehat{\lambda}_1}|\langle\widehat{\boldsymbol{u}}_l,\boldsymbol{w}\rangle|^2\le\varepsilon$ for some error term $\varepsilon>0$. We can see that,

$$\boldsymbol{w}^\top\widehat{\boldsymbol{\Sigma}}\boldsymbol{w}>(1-\delta)(1-\varepsilon)\widehat{\lambda}_1. \tag{19}$$

This fact can be easily derived as follows (see also the proof of Theorem 3.1 in Allen-Zhu and Li (2016)),

$$\begin{aligned}\boldsymbol{w}^\top\widehat{\boldsymbol{\Sigma}}\boldsymbol{w}=\sum_{i=1}^d\widehat{\lambda}_i(\boldsymbol{w}^\top\widehat{\boldsymbol{u}}_i)^2&\ge\sum_{l:\widehat{\lambda}_l>(1-\delta)\widehat{\lambda}_1}\widehat{\lambda}_l(\boldsymbol{w}^\top\widehat{\boldsymbol{u}}_l)^2\ge(1-\delta)\widehat{\lambda}_1\\&\quad\times\sum_{l:\widehat{\lambda}_l>(1-\delta)\widehat{\lambda}_1}(\boldsymbol{w}^\top\widehat{\boldsymbol{u}}_l)^2\\&\ge(1-\delta)(1-\varepsilon)\widehat{\lambda}_1.\end{aligned}$$

According to (19), our estimator $\boldsymbol{w}$ captures almost the same amount of variability of the sampled data (up to a $(1-\delta)(1-\varepsilon)$ multiplicative factor). This type of results are also known as the gap-free bound in some optimization literature (see e.g., Allen-Zhu and Li 2016).

When the eigengap is extremely small, identifying the top eigenvector is an information-theoretically difficult problem. As an extreme case, when the gap is zero, it is impossible to distinguish between the top and the second eigenvectors. In contrast, our setting is favorable in practice since the main goal of PCA/dimension reduction is to capture the variability of the data.

Moreover, we note that the parameter δ is a prespecified parameter that measures the proportion of the variability explain by the estimator $\boldsymbol{w}$. For example, when setting $\delta=\varepsilon$, the estimator $\boldsymbol{w}$ will capture at least $(1-2\varepsilon)$ of the variability captured by the top eigenvector according to (19). We can also choose $\delta=c_0/\widehat{\lambda}_1$ for some constant c_0 so that $\delta\widehat{\lambda}_1=\Omega(1)$.

3.2. Distributed Top-L-Dim Principal Subspace Estimation

With the theoretical results for the top eigenvector estimation in place, we further present convergence analysis on the top-L-dim eigenspace estimation in Algorithm 2.

Let $\widehat{\boldsymbol{U}}_{\le(1-\delta)\widehat{\lambda}_L}=[\widehat{\boldsymbol{u}}_{S+1},\ldots,\widehat{\boldsymbol{u}}_d]$ denote the column orthogonal matrix composed of all eigenvectors of $\widehat{\boldsymbol{\Sigma}}$ whose associated eigenvalues have a relative gap δ from the Lth largest eigenvalue $\widehat{\lambda}_L$, that is, $S:=\arg\max\{l:\widehat{\lambda}_l>(1-\delta)\widehat{\lambda}_L\}$. We also denote $\widehat{\boldsymbol{U}}_{>(1-\delta)\widehat{\lambda}_L}=[\widehat{\boldsymbol{u}}_1,\ldots,\widehat{\boldsymbol{u}}_S]$ to be the enlarged eigenspace corresponding to the eigenvalues larger than $(1-\delta)\widehat{\lambda}_L$.

We also use the notation $\widehat{\boldsymbol{\Sigma}}^{(l)}=(\boldsymbol{I}-\boldsymbol{V}_{l-1}\boldsymbol{V}_{l-1}^\top)\widehat{\boldsymbol{\Sigma}}(\boldsymbol{I}-\boldsymbol{V}_{l-1}\boldsymbol{V}_{l-1}^\top)$ for $l=0,1,\ldots,L-1$. Here $\boldsymbol{V}_l=[\boldsymbol{v}_1,\ldots,\boldsymbol{v}_l]$ consists of all the top-l eigenvector estimations and $\boldsymbol{V}_0=\boldsymbol{0}$. Notice that $\widehat{\boldsymbol{\Sigma}}^{(l)}$ is just the matrix $\boldsymbol{A}^{(l)}\boldsymbol{A}^{(l)}/n$ where $\boldsymbol{A}^{(l)}:=[\boldsymbol{A}_{1,l}^\top,\ldots,\boldsymbol{A}_{K,l}^\top]^\top$ and $\boldsymbol{A}_{k,l}^T$ is the projected data matrix on machine k ($k\in[K]$) for the lth eigenvector estimation.

We first provide our choices of initial eigenvalue estimates. For Algorithm 2 for the top-L-dim principal, let $\widehat{\boldsymbol{\Sigma}}_k^{(l)}=\boldsymbol{A}_{k,l}^\top\boldsymbol{A}_{k,l}/m$ and $\widehat{\boldsymbol{\Sigma}}^{(l)}=K^{-1}\sum_{k=1}^K\widehat{\boldsymbol{\Sigma}}_k^{(l)}$ denote the local and global projected sample covariance matrices at the outer iteration l. For the same constant η defined above in (11), we choose $\overline{\lambda}_l=\lambda_1(\widehat{\boldsymbol{\Sigma}}_1^{(l)})+3\eta/2$ for $l\in[L]$. This follows from

$$\begin{aligned}\left\|\widehat{\boldsymbol{\Sigma}}^{(l)}-\widehat{\boldsymbol{\Sigma}}_1^{(l)}\right\|_2&=\left\|(\boldsymbol{I}-\boldsymbol{V}_l\boldsymbol{V}_l^\top)(\widehat{\boldsymbol{\Sigma}}-\widehat{\boldsymbol{\Sigma}}_1)(\boldsymbol{I}-\boldsymbol{V}_l\boldsymbol{V}_l^\top)\right\|_2\\&\le\left\|\widehat{\boldsymbol{\Sigma}}-\widehat{\boldsymbol{\Sigma}}_1\right\|_2\le\frac{\eta}{2},\end{aligned}$$

which implies that,

$$2\eta\ge\overline{\lambda}_l-\lambda_1(\widehat{\boldsymbol{\Sigma}}^{(l)})\ge\eta.$$

Our main result is summarized as follows.

Theorem 3.2. Let $\kappa=\left\|\widehat{\boldsymbol{\Sigma}}-\widehat{\boldsymbol{\Sigma}}_1\right\|_2=\mathcal{O}_P(\sqrt{d/m})$. Assume

$$2\eta\ge\overline{\lambda}_l-\lambda_1(\widehat{\boldsymbol{\Sigma}}^{(l)})\ge\eta\ge\frac{1}{2}\kappa,$$

for each $l\in[L]$, where $\lambda_1(\widehat{\boldsymbol{\Sigma}}^{(l)})$ denotes the largest eigen value of $\widehat{\boldsymbol{\Sigma}}^{(l)}$. Then we have

$$\begin{aligned}&\left\|\widehat{\boldsymbol{U}}_{\le(1-\delta)\widehat{\lambda}_L}^\top\boldsymbol{V}_L\right\|_2^2\\&\le\frac{64\widehat{\lambda}_1L^2}{\widehat{\lambda}_L\delta}\sqrt{\left(\frac{128\eta^2}{\delta^2\widehat{\lambda}_L^2}\right)^T+\frac{512\,\eta}{1-128\eta^2/(\delta\widehat{\lambda}_L)^2}\left(\frac{4\kappa^2}{\eta^2}\right)^{T'}}.\end{aligned}\tag{20}$$

By choosing specific settings of some parameters, the result in Theorem 3.2 can be simplified as shown in the following corollary.

Corollary 3.2. Similarly, if $\eta\le\delta\widehat{\lambda}_1/16$, and we choose $T'=T$, and $\eta=\left(\kappa\delta\widehat{\lambda}_1\right)^{1/2}/3=\mathcal{O}_P(\sqrt[4]{d/m})$, then our estimator $\boldsymbol{V}_L$ satisfies

$$\left\|\widehat{\boldsymbol{U}}_{\le(1-\delta)\widehat{\lambda}_L}^\top\boldsymbol{V}_L\right\|_2^2=\mathcal{O}\left(\frac{\widehat{\lambda}_1L^2}{\widehat{\lambda}_L\delta}\left(\frac{6\kappa}{\delta\widehat{\lambda}_L}\right)^T\right).$$

Here we could also interpret our results in Theorem 3.2 from an "angle" point of view corresponding to the classical $\sin\Theta$ result. Since there is no eigengap assumption, it is impossible to directly estimate $\boldsymbol{U}_L$. Therefore, we choose a parameter δ, and consider *an enlarged eigenspace* $\boldsymbol{U}_{>(1-\delta)\widehat{\lambda}_L}$. Our theoretical results (see Theorem 3.2 and Corollary 3.2) imply that the "angle" between our estimator $\boldsymbol{V}_L$ and $\widehat{\boldsymbol{U}}_{\le(1-\delta)\widehat{\lambda}_L}$ is sufficiently small. This result extends the classical $\sin\Theta$ result.

Similar to the top-eigenvector case in Remark 3.1, our estimator $\boldsymbol{V}_L$ can also capture a similar amount of variability in the sampled data to $\widehat{\boldsymbol{U}}_L:=\{\widehat{\boldsymbol{u}}_1,\ldots,\widehat{\boldsymbol{u}}_L\}$. We further describe this property in the following Corollary 3.3.

Corollary 3.3. Assume our estimator $\boldsymbol{V}_L$ from Algorithm 2 satisfies $\left\|\widehat{\boldsymbol{U}}_{\le(1-\delta)\widehat{\lambda}_L}^\top\boldsymbol{V}_L\right\|_2\le\frac{\delta}{16\widehat{\lambda}_1/\widehat{\lambda}_{L+1}}$, then we have,

$$\widehat{\lambda}_{L+1}\le\left\|\left(\boldsymbol{I}_d-\boldsymbol{V}_L\boldsymbol{V}_L^\top\right)\widehat{\boldsymbol{\Sigma}}\left(\boldsymbol{I}_d-\boldsymbol{V}_L\boldsymbol{V}_L^\top\right)\right\|_2\le\frac{\widehat{\lambda}_{L+1}}{1-\delta},\tag{21}$$

$$(1-\delta)\widehat{\lambda}_l\le\boldsymbol{v}_l^\top\widehat{\boldsymbol{\Sigma}}\boldsymbol{v}_l\le\frac{1}{1-\delta}\widehat{\lambda}_l,\quad\forall l\in[L].\tag{22}$$

Now we further extend the result in Corollary 3.2 to quantify the "angle" between our estimator $\boldsymbol{V}_L$ and the population eigenspace $\boldsymbol{U}_{\leq(1-2\delta)\lambda_L}$.

Corollary 3.4. Assume our estimator $\boldsymbol{V}_L$ from Algorithm 2 satisfies $\left\|\widehat{\boldsymbol{U}}_{\leq(1-\delta)\widehat{\lambda}_L}^\top \boldsymbol{V}_L\right\|_2 \leq \varepsilon$ for some error term $\varepsilon > 0$, then we have,

$$\left\|\boldsymbol{U}_{\leq(1-2\delta)\lambda_L}^\top \boldsymbol{V}_L\right\|_2 \leq \frac{\|\boldsymbol{\Sigma} - \widehat{\boldsymbol{\Sigma}}\|_2}{(1-\delta)(\widehat{\lambda}_L - \lambda_L) + \delta\lambda_L} + \varepsilon, \quad (23)$$

where $\boldsymbol{U}_{\leq(1-2\delta)\lambda_L}$ is the eigenvectors of the population covariance matrix $\boldsymbol{\Sigma}$ corresponding to eigenvalues less than or equal to $(1-2\delta)\lambda_L$.

We further provide a different "angle" result on quantifying the complement of an enlarged space of V_L. Recall our definition $S := \arg\max\{l : \widehat{\lambda}_l > (1-\delta_L)\widehat{\lambda}_L\}$. We can classify $\widehat{\boldsymbol{u}}_1, \ldots, \widehat{\boldsymbol{u}}_d$ and correspondingly our estimators $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d$ from Algorithm 2 into three regimes:

$$\overbrace{\underbrace{\widehat{\boldsymbol{u}}_1, \ldots, \widehat{\boldsymbol{u}}_L}_{\widehat{\boldsymbol{U}}_L}, \widehat{\boldsymbol{u}}_{L+1}, \ldots, \widehat{\boldsymbol{u}}_S}^{\widehat{\boldsymbol{U}}_S}, \underbrace{\widehat{\boldsymbol{u}}_{S+1}, \ldots, \widehat{\boldsymbol{u}}_d}_{\widehat{\boldsymbol{U}}_{\leq(1-\delta)\widehat{\lambda}_L}}, \quad (24)$$

$$\overbrace{\underbrace{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_L}_{\boldsymbol{V}_L}, \boldsymbol{v}_{L+1}, \ldots, \boldsymbol{v}_S}^{\boldsymbol{V}_S}, \underbrace{\boldsymbol{v}_{S+1}, \ldots, \boldsymbol{v}_d}_{\boldsymbol{V}_{\leq(1-\delta)\widehat{\lambda}_L}}.$$

Corollary 3.2 shows that the "angle" between our estimator $\boldsymbol{V}_L$ and $\widehat{\boldsymbol{U}}_{\leq(1-\delta)\widehat{\lambda}_L}$ is sufficiently small. Similarly, we can show the counterpart of this result, which indicates that the "angle" between $\widehat{\boldsymbol{U}}_L$ and $\boldsymbol{V}_{\leq(1-\delta)\widehat{\lambda}_L}$ is also very small. This result will be useful in our principal component regression example. To introduce our result, we denote $\widehat{\lambda}_S$ to be the Sth largest eigenvalue of $\widehat{\boldsymbol{\Sigma}}$.

Theorem 3.3. By running Algorithm 2 for obtaining the distributed top-S-dim principal subspace estimator $\boldsymbol{V}_S$, if there exists $\delta_S < \delta$ such that $\left\|\widehat{\boldsymbol{U}}_{\leq(1-\delta_S)\widehat{\lambda}_S}^\top \boldsymbol{V}_S\right\|_2 \leq \frac{\delta_S}{16\widehat{\lambda}_1/\widehat{\lambda}_{S+1}}$, then we have for the empirical eigenspace $\widehat{\boldsymbol{U}}_L$

$$\left\|\widehat{\boldsymbol{U}}_L^\top \boldsymbol{V}_{\leq(1-\delta)\widehat{\lambda}_L}\right\|_2 \leq S\,\frac{\delta_S\widehat{\lambda}_1}{\widehat{\lambda}_L(1-\delta_S) - \widehat{\lambda}_S}. \quad (25)$$

Furthermore, for the population eigenspace $\boldsymbol{U}_L$, we can derive that

$$\left\|\boldsymbol{U}_L^\top \boldsymbol{V}_{\leq(1-2\delta)\widehat{\lambda}_L}\right\|_2 \leq S\,\frac{\delta_S\widehat{\lambda}_1}{\widehat{\lambda}_L(1-\delta_S) - \widehat{\lambda}_S} + \frac{\|\boldsymbol{\Sigma} - \widehat{\boldsymbol{\Sigma}}\|_2}{(1-\delta)(\widehat{\lambda}_L - \lambda_L) + \delta\lambda_L}. \quad (26)$$

The reason why we impose the upper bound on $\left\|\widehat{\boldsymbol{U}}_{\leq(1-\delta_S)\widehat{\lambda}_S}^\top \boldsymbol{V}_S\right\|_2$ is mainly to obtain the result in Equation (21) for $\boldsymbol{V}_S$. We also note that this upper bound can be easily satisfied as long as we run Algorithm 2 for sufficiently large number of iterations.

Let us recall the classical Davis-Kahan result for PCA in Equation (2). As we explained in the introduction, without an eigengap condition, the estimation error can be arbitrarily large. However, our enlarged eigenspace estimator $\boldsymbol{V}_S$ in (24) (i.e., $\boldsymbol{V}_{>(1-\delta)\widehat{\lambda}_L}$) will almost contain the top-L-dim eigenspace of the population covariance matrix. In particular, by Equation (26) and Lemma A.1 in the supplementary appendix, we have shown that there exists a matrix $\boldsymbol{Q}$ satisfying $\|\boldsymbol{Q}\|_2 \leq 1$ such that the error bound $\left\|\boldsymbol{U}_L - \boldsymbol{V}_{>(1-2\delta)\widehat{\lambda}_L}\boldsymbol{Q}\right\|_2$ is sufficiently small.

Our enlarged eigenspace results find important applications to many statistical problems. In particular, in Appendix B in the supplementary materials, we illustrate how the theoretical results can be applied to the principal component regression (Example 1) and the single index model (Example 2). We also provide simulation studies of these two applications in Appendix D in the supplementary materials.

Remark 3.2. It is also worthwhile to note that we assume the data are evenly split only for the ease of discussions. In fact, the local sample size m in our theoretical results is the sample size on the first machine (or any other machine that used to compute the estimation of Hessian $\mathbf{H}$) in Algorithms 1 and 2. As long as the sample size m on the first machine is specified, our method does not depend on the partition of the entire dataset.

4. Numerical Study

In this section, we provide simulation experiments to illustrate the empirical performance of our distributed PCA algorithm.

Our data follows a normal distribution, $\mathbb{E}[\boldsymbol{a}] = 0$ and the population covariance matrix $\mathbb{E}[\boldsymbol{a}\boldsymbol{a}^\top] = \boldsymbol{\Sigma}$ is generated as follows:

$$\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T,$$

where $\boldsymbol{U}$ is an orthogonal matrix generated randomly and $\boldsymbol{\Lambda}$ is a diagonal matrix. Since our experiments mainly estimate the top-3 eigenvectors, $\boldsymbol{\Lambda}$ has the following form,

$$\boldsymbol{\Lambda} = \text{diag}\,(1+3\delta, 1+2\delta, 1+\delta, 1, \ldots, 1)\,. \quad (27)$$

For example, when the relative eigengap δ is 1, $\boldsymbol{\Lambda} = \text{diag}(4, 3, 2, 1, \ldots, 1)$.

For orthogonal matrix $\boldsymbol{U} = [u_{ij}] \in \mathbb{R}^{d\times d}$, we first generate all elements $u_{ij}, i,j = 1, \ldots, d$ such that they are iid standard normal variables. We then use Gram–Schmidt process to orthonormalize the matrix and obtain the $\boldsymbol{U}$.

We will compare our estimator with the following two estimators:

1. Oracle estimator: the PCA estimator is computed in the single-machine setting with pooled data, that is, we gather all the sampled data and compute the top eigenspace of $\widehat{\boldsymbol{\Sigma}} = \frac{1}{n}\boldsymbol{A}\boldsymbol{A}^\top$, where $\boldsymbol{A} \in \mathbb{R}^{n\times d}$ i the data matrix.

2. DC estimator (Algorithm 1 in Fan et al. (2019)): it first computes the top-L-dim eigenspace estimation $\widehat{\boldsymbol{U}}_L^{(k)}, k = 1, \ldots, K$ on each machine, and merges every local result together with $\widetilde{\boldsymbol{\Sigma}} = \frac{1}{K}\sum_{k=1}^K \widehat{\boldsymbol{U}}_L^{(k)}\widehat{\boldsymbol{U}}_L^{(k)\top}$. The final estimator is given by the eigenvalue decomposition of $\widetilde{\boldsymbol{\Sigma}}$.

Note that all the reported estimation errors are computed based on the average of 100 Monte Carlo simulations. Since the standard deviations of Monte Carlo estimators for all the methods are similar and sufficiently small, we omit standard deviation terms in the following figures and only report the average errors for better visualization. As shown in the following subsections, our distributed algorithm gets to a very close performance with the oracle one when the number of outer iterations T is large enough and outperforms its divide-and-conquer counterpart.

For distributed PCA, we adopt the following error measurements from the bound (17) and bound (20) with population eigenvectors replacing the oracle estimator. To be more specific, for the top eigenvector case, with the estimator $\widehat{\boldsymbol{u}}_1$, population eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d$, population eigenvalues $\lambda_1, \ldots, \lambda_d$ and relative eigenvalue gap $\delta \in (0, 1)$, the error measurement is defined as

$$\operatorname{error}(\widehat{\boldsymbol{u}}_1) = \sum_{l:\lambda_l \le (1-\delta)\lambda_1} \left|\langle \boldsymbol{u}_l, \widehat{\boldsymbol{u}}_1 \rangle\right|^2. \tag{28}$$

As for the top-L-dim eigenspace estimation, let $\widetilde{\boldsymbol{U}} = [\boldsymbol{u}_{l_\delta}, \ldots, \boldsymbol{u}_d]$ be the column orthogonal matrix composed of all eigenvectors of population covariance $\boldsymbol{\Sigma}$ whose associated eigenvalues have a relative gap δ from the Lth largest eigenvalue λ_L. That is, $l_\delta := \arg\min\{l : \widehat{\lambda}_l \le (1-\delta)\widehat{\lambda}_L\}$. Recall that $\widehat{\boldsymbol{U}}_L$ is the estimator the top-L eigenvectors. Then the corresponding error should be

$$\operatorname{error}(\widehat{\boldsymbol{U}}_L) = \left\| \widetilde{\boldsymbol{U}}^\top \widehat{\boldsymbol{U}}_L \right\|_2^2. \tag{29}$$

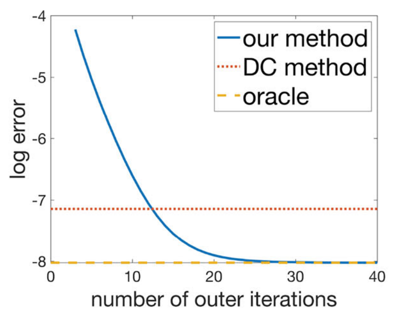
4.1. Varying the Number of Outer Iterations

In this section, we present tests on how the performance of our distributed PCA changes with the number of outer iterations T in Algorithm 1. Consider data dimension d to be 50, sample size on each machine to be 500, and the number of machines to be 200, that is, $\boldsymbol{a} \in \mathbb{R}^{50}$, $m = 500$ and $K = 200$.

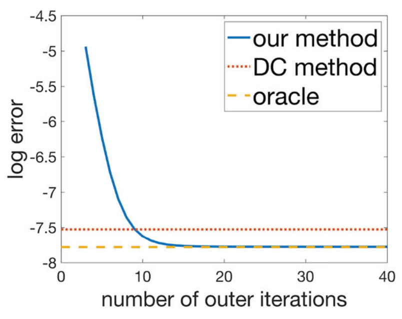
We will report the *logarithmic error*. As shown in Theorem 3.1, the logarithmic error follows an approximately linear decrease with respect to the number of outer iterations. A linear relationship between the number of outer iterations and logarithmic error verifies our theoretical findings.

We now check the performance of these three approaches (oracle one, our method and DC method) under the setting of a small eigengap. Specifically, we let eigengap δ to be 1.0 and 2.0. Our data are drawn independently, and $\boldsymbol{a}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$ for $i = 1, \ldots, mK$. We vary the number of outer iterations T to evaluate the performance.

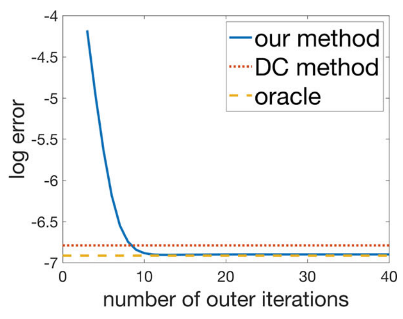
As we fix the total sample size $n = 10^5$, the errors of oracle estimator and DC estimator should be constants (illustrated by two horizontal dash lines in the graphs since they are not iterative algorithms). As shown below in Figures 1 and 2, our method converges to the oracle estimator in around 20 iterations and outperforms the DC method. Moreover, as expected, we observe an approximately linear relation between logarithmic error and the number of outer iterations. We also observe that, empirically, setting the number of inner iterations $T' = 5$ in Algorithm 1 is good enough for most cases.



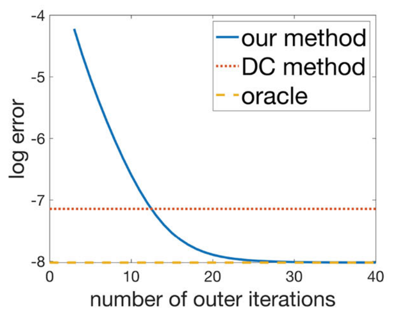
(a) Top-1-dim eigenvector



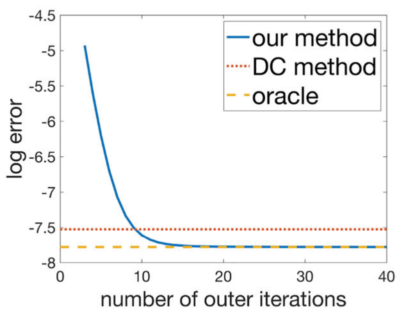
(b) Top-2-dim eigenspace



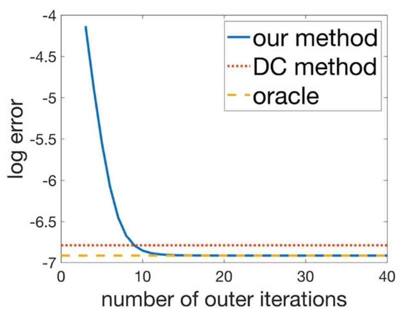
(c) Top-3-dim eigenspace



(d) Top-1-dim eigenvector

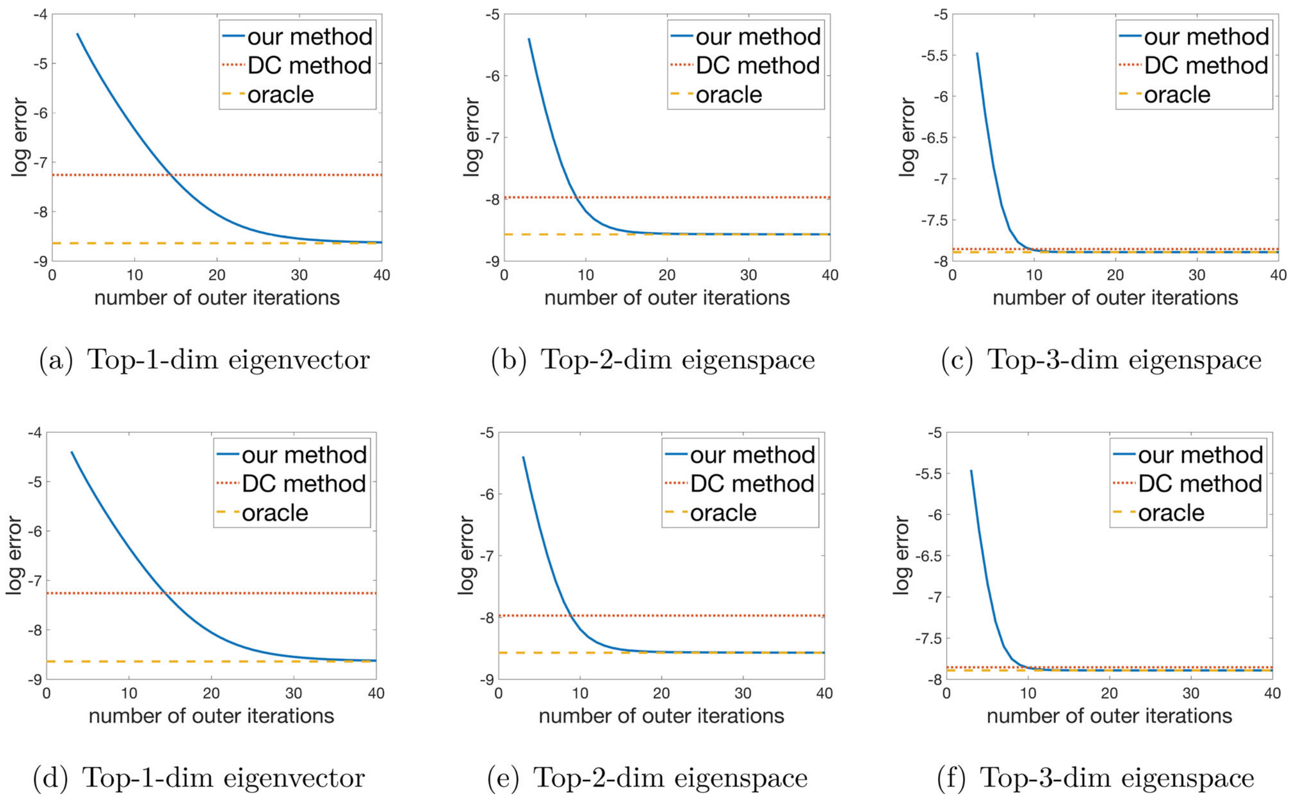


(e) Top-2-dim eigenspace



(f) Top-3-dim eigenspace

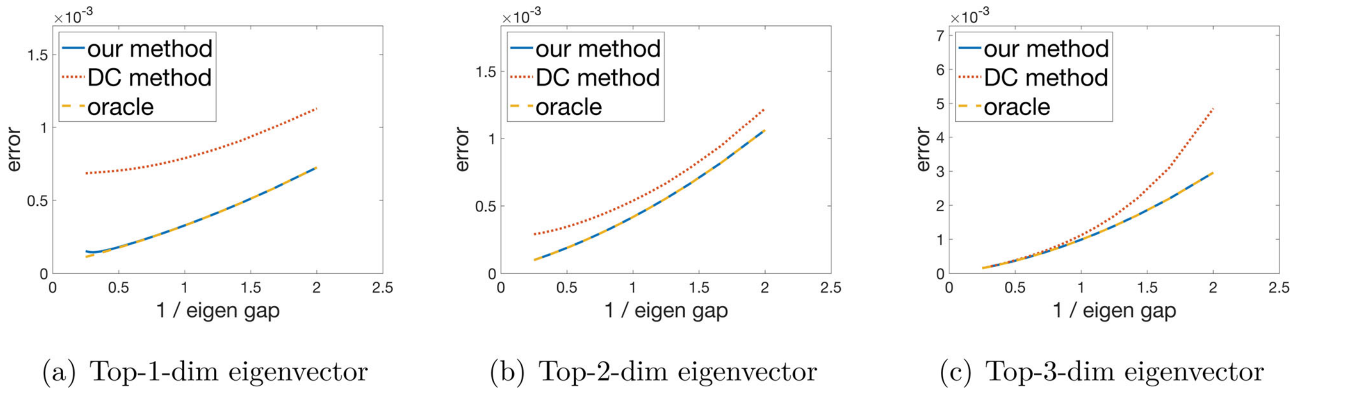
Figure 1. Comparison between algorithms when the number of outer iterations varies. The x-axis is the number of outer iterations and the y-axis is the *logarithmic error*. The blue line is our error, the red line is the DC method performance and the yellow one is logarithmic error for the oracle estimator. (a)–(c) The experiments with 5 inner loops. (d)–(f) The experiments with 10 inner loops. Eigengap δ is fixed to be 1.0.



(a) Top-1-dim eigenvector (b) Top-2-dim eigenspace (c) Top-3-dim eigenspace

(d) Top-1-dim eigenvector (e) Top-2-dim eigenspace (f) Top-3-dim eigenspace

Figure 2. Comparison between algorithms when the number of outer iterations varies, under the same setting as in Figure 1. (a)–(c) The experiments with 5 inner loops. (d)–(f) The experiments with 10 inner loops. Eigengap δ is fixed to be 2.0.



(a) Top-1-dim eigenvector (b) Top-2-dim eigenvector (c) Top-3-dim eigenvector

Figure 3. Comparison between algorithms when the eigengap varies. The *x*-axis is the reciprocal of eigengap and the *y*-axis is the logarithmic error.

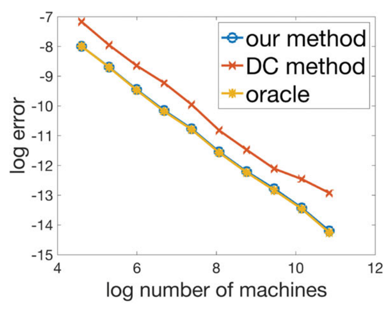
4.2. Varying the Eigengap

In the convergence analysis of both our distributed algorithm and DC method, eigengap plays a central role in the error bound. When the eigengap between λ_L and λ_{L+1} becomes smaller, the estimation task turns to be harder and more rounds are needed for the same error. Theorem 4 in Fan et al. (2019) also shows a similar conclusion. In this part, we continue our experiment in Section 4.1, and examine the relationship between estimation error and eigengap.

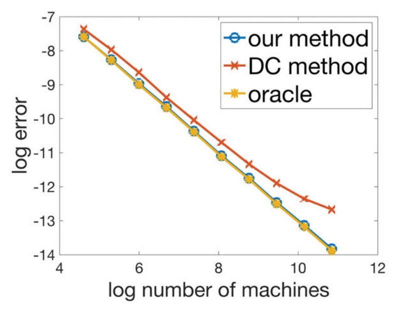
We fix the number of inner iterations to be 10, and the number of outer iterations to be 40, which, from Section 4.1, is large enough for top-3-dim eigenspace. We still consider data dimension d to be 50, sample size on each machine to be 500, and the number of machines to be 200, that is, $\boldsymbol{a} \in \mathbb{R}^{50}$, $m = 500$ and $K = 200$. Under this setting, we vary δ in (27) and the results is shown in Figure 3. In Figure 3, the logarithmic error increases with respect to $1/\delta$, which agrees with our theoretical findings. Furthermore, our estimator has the same performance as the oracle one.

4.3. Varying the Number of Machines for Asymmetric Innovation Distributions

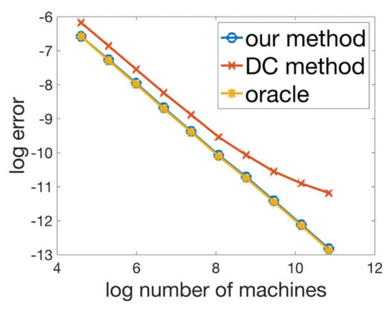
In this section, we compare our method to the DC method by varying the number of local machines. As mentioned in Theorem 4 in Fan et al. (2019), DC method has a slower convergence rate (of order $\mathcal{O}(\rho\sqrt{Lr/n}) + \mathcal{O}(\rho^2\sqrt{L}r/m)$ instead of the optimal rate $\mathcal{O}(\rho\sqrt{Lr/n})$) when the number of machines is greater than $\mathcal{O}\left(m/(\rho^2 r)\right)$ in the asymmetric innovation distributions (defined in Section 1.3) setting. Here ρ is the condition number



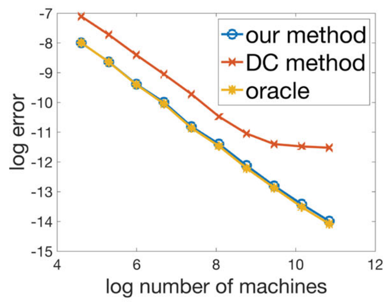
(a) Top-1-dim eigenvector



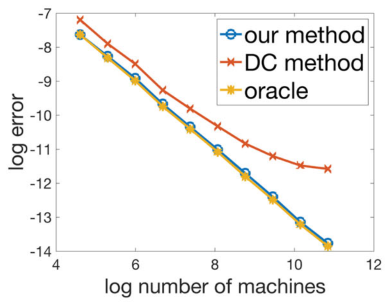
(b) Top-2-dim eigenvector



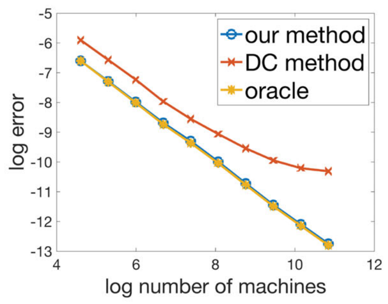
(c) Top-3-dim eigenvector



(d) Top-1-dim eigenvector



(e) Top-2-dim eigenvector



(f) Top-3-dim eigenvector

Figure 4. Comparison between algorithms when the number of machines varies. The *x*-axis is the log the number of machines and the *y*-axis is the logarithmic error. (a)–(c) The experiments of top-1-dim to top-3-dim eigenvector estimation with skewness 4.0 and (d)–(f) 6.0.

of the population covariance matrix, that is, $\rho = \lambda_1/(\lambda_L - \lambda_{L+1})$, and $r = \mathrm{Tr}(\boldsymbol{\Sigma})/\lambda_1$ is the effective rank of $\boldsymbol{\Sigma}$.

We set data dimension d to be 50, local sample size to be 500, that is, $\boldsymbol{a} \in \mathbb{R}^{50}$, $m = 500$. We choose eigengap δ to be 0.5, thus $\boldsymbol{\Lambda} = \mathrm{diag}(2.5, 2, 1.5, 1, \dots, 1)$. Here, without sticking on our Gaussian setting, we consider to use skew-distributed random variables. In particular, we generate $\boldsymbol{a} = [a_1, \dots, a_d]^\top \in \mathbb{R}^d$ from beta distribution family such that for each $a_i, i = 1, \dots, d$, we set its mean to be zero, variance to be $\boldsymbol{\Lambda}_{ii}$ and skewness to be 4 or 6, respectively.

We set the iteration parameters as in Section 4.2 and the number of machines is varied from 100 to 51,200. Our results are shown in Figure 4. As can be seen from Figure 4, our method achieves the same statistical convergence rate as the oracle one. When the number of machines is small, the estimation error of the DC method also decreases at the same rate as the number of machines increases. However, the estimation error the of DC method becomes flat (or decreases at a much slower rate) when the number of machines is larger than a certain threshold. In that regime, our approach is still comparable to its oracle counterpart.

We also conduct simulation studies on principal component regression and Gaussian single index model cases and compare our approach with the oracle and the DC ones. Due to the space limitation, we defer these results to Appendix D in the supplementary materials.

5. Discussions and Future Work

In this article, we address the problem of distributed estimation for principal eigenspace. Our proposed multi-round method achieves fast convergence rate. Furthermore, we establish an error bound for our method from an enlarged eigenspace viewpoint, which can be seen as an extension to the traditional error bound. The insight behind our work is the combination of shift-and-invert preconditioning and convex optimization, with the adaption into distributed environment. This distributed PCA algorithm refines the divide-and-conquer scheme and removes the constraint on the number of machines from previous methods.

One important future direction is to further investigate the principal eigenspace problem under distributed settings. Specifically, computational approaches and theoretical tools can be established for other types of PCA problems, such as PCA in high dimension (see, e.g., Johnstone 2001; Fan and Wang 2017; Cai and Zhang 2018) and sparse PCA (see, e.g., Johnstone and Lu 2009; Cai, Ma, and Wu 2013; Vu and Lei 2013).

Supplementary Materials

In the supplementary material, we first provide detailed proofs of all our theoretical results in Section 3. We then study two applications of our distributed PCA: principal component regression and single index model. We provide theoretical findings and conduct additional numerical experiments for these two applications.

Acknowledgments

The authors would like to thank the editor-in-chief, the anonymous associated editor, and two anonymous referees for many useful suggestions and feedback that greatly improve the article.

Funding

Xi Chen is supported by NSF grant IIS-1845444. Jason D. Lee is supported by NSF grant CCF-2002272. Yun Yang is supported by NSF grant DMS-1810831.

References

Allen-Zhu, Z., and Li, Y. (2016), "LazySVD: Even Faster SVD Decomposition Yet Without Agonizing Pain," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. [2,5,7]

Bair, E., Hastie, T., Paul, D., and Tibshirani, R. (2006), "Prediction by Supervised Principal Components," *Journal of the American Statistical Association*, 101, 119–137. [3]

Banerjee, M., Durot, C., and Sen, B. (2019), "Divide and Conquer in Nonstandard Problems and the Super-Efficiency Phenomenon," *The Annals of Statistics*, 47, 720–757. [1]

Battey, H., Fan, J., Liu, H., Lu, J., and Zhu, Z. (2018), "Distributed Testing and Estimation Under Sparse High Dimensional Models," *The Annals of Statistics*, 46, 1352. [1]

Bengio, Y., Courville, A., and Vincent, P. (2013), "Representation Learning: A Review and New Perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35, 1798–1828. [2]

Cai, T. T., Ma, Z., and Wu, Y. (2013), "Sparse PCA: Optimal Rates and Adaptive Estimation," *The Annals of Statistics*, 41, 3074–3110. [11]

Cai, T. T., and Zhang, A. (2018), "Rate-Optimal Perturbation Bounds for Singular Subspaces With Applications to High-Dimensional Statistics," *The Annals of Statistics*, 46, 60–89. [11]

Chen, X., Liu, W., Mao, X., and Yang, Z. (2020), "Distributed High-Dimensional Regression Under a Quantile Loss Function," *Journal of Machine Learning Research*, 21, 1–43. [1]

Chen, X., Liu, W., and Zhang, Y. (2021), "First-Order Newton-Type Estimator for Distributed Estimation and Inference," *Journal of the American Statistical Association*, in press, DOI: 10.1080/01621459.2021.1891925. [1]

Chen, X., Liu, W., and Zhang, Y. (2019), "Quantile Regression Under Memory Constraint," *The Annals of Statistics*, 47, 3244–3273. [1]

Davis, C., and Kahan, W. M. (1970), "The Rotation of Eigenvectors by a Perturbation. III," *SIAM Journal on Numerical Analysis*, 7, 1–46. [2]

Fan, J., Guo, Y., and Wang, K. (2019), "Communication-Efficient Accurate Statistical Estimation," arXiv no. 1906.04870. [4]

Fan, J., Wang, D., Wang, K., and Zhu, Z. (2019), "Distributed Estimation of Principal Eigenspaces," *The Annals of Statistics*, 47, 3009–3031. [2,5,8,10]

Fan, J., and Wang, W. (2017), "Asymptotics of Empirical Eigen-Structure for Ultra-High Dimensional Spiked Covariance Model," *The Annals of Statistics*, 45, 1342–1374. [11]

Frank, L. E., and Friedman, J. H. (1993), "A Statistical View of Some Chemometrics Regression Tools," *Technometrics*, 35, 109–135. [3]

Garber, D., and Hazan, E. (2015), "Fast and Simple PCA via Convex Optimization," arXiv no. 1509.05647. [2]

Garber, D., Hazan, E., Jin, C., Kakade, S. M., Musco, C., Netrapalli, P., and Sidford, A. (2016), "Faster Eigenvector Computation via Shift-and-Invert Preconditioning," in *Proceedings of the International Conference on Machine Learning (ICML)*. [2,3]

Garber, D., Shamir, O., and Srebro, N. (2017), "Communication-Efficient Algorithms for Distributed Stochastic Principal Component Analysis," in *Proceedings of the International Conference on Machine Learning (ICML)*. [2]

Horowitz, J. L. (2009), *Semiparametric and Nonparametric Methods in Econometrics* (Vol. 12), New York: Springer. [3]

Hotelling, H. (1933), "Analysis of a Complex of Statistical Variables Into Principal Components," *Journal of Educational Psychology*, 24, 417–441. [1]

Hristache, M., Juditsky, A., and Spokoiny, V. (2001), "Direct Estimation of the Index Coefficient in a Single-Index Model," *The Annals of Statistics*, 29, 595–623. [3]

Janzamin, M., Sedghi, H., and Anandkumar, A. (2014), "Score Function Features for Discriminative Learning: Matrix and Tensor Framework," arXiv no. 1412.2863. [3]

Jeffers, J. (1967), "Two Case Studies in the Application of Principal Component Analysis," *Journal of the Royal Statistical Society*, Series C, 16, 225–236. [1,3]

Johnson, R., and Zhang, T. (2013), "Accelerating Stochastic Gradient Descent Using Predictive Variance Reduction," in *Advances in Neural Information Processing Systems (NIPS)*. [2]

Johnstone, I. M. (2001), "On the Distribution of the Largest Eigenvalue in Principal Components Analysis," *The Annals of Statistics*, 29, 295–327. [11]

Johnstone, I. M., and Lu, A. Y. (2009), "On Consistency and Sparsity for Principal Components Analysis in High Dimensions," *Journal of the American Statistical Association*, 104, 682–693. [11]

Jolliffe, I. T. (1982), "A Note on the Use of Principal Components in Regression," *Journal of the Royal Statistical Society*, Series C, 31, 300–303. [1,3]

Jordan, M. I., Lee, J. D., and Yang, Y. (2019), "Communication-Efficient Distributed Statistical Inference," *Journal of the American Statistical Association*, 114, 668–681. [1,4,5]

Lee, J. D., Liu, Q., Sun, Y., and Taylor, J. E. (2017), "Communication-Efficient Sparse Regression," *Journal of Machine Learning Research*, 18, 1–30. [1]

Li, K.-C. (1992), "On Principal Hessian Directions for Data Visualization and Dimension Reduction: Another Application of Stein's Lemma," *Journal of the American Statistical Association*, 87, 1025–1039. [2,3]

Pearson, K. (1901), "LIII. On Lines and Planes of Closest Fit to Systems of Points in Space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2, 559–572. [1]

Rigollet, P., and Hütter, J.-C. (2015), "High Dimensional Statistics," Lecture Notes for Course 18S997. [3]

Shamir, O. (2016), "Fast Stochastic Algorithms for SVD and PCA: Convergence Properties and Convexity," in *Proceedings of the International Conference on Machine Learning (ICML)*. [2]

Shamir, O., Srebro, N., and Zhang, T. (2014), "Communication Efficient Distributed Optimization Using an Approximate Newton-Type Method," in *Proceedings of the International Conference on Machine Learning (ICML)*. [4]

Shi, C., Lu, W., and Song, R. (2018), "A Massive Data Framework for *M*-Estimators With Cubic-Rate," *Journal of American Statistical Association*, 113, 1698–1709. [1]

Stein, C. M. (1981), "Estimation of the Mean of a Multivariate Normal Distribution," *The Annals of Statistics*, 9, 1135–1151. [3]

Van Loan, C., and Golub, G. (2012), *Matrix Computations* (3rd ed.), Baltimore, MD: Johns Hopkins University Press. [2]

Vershynin, R. (2012), "Introduction to the Non-Asymptotic Analysis of Random Matrices," in *Compressed Sensing*, pp. 210–268. [3]

Volgushev, S., Chao, S.-K., and Cheng, G. (2019), "Distributed Inference for Quantile Regression Processes," *The Annals of Statistics*, 47, 1634–1662. [1]

Vu, V. Q., and Lei, J. (2013), "Minimax Sparse Principal Subspace Estimation in High Dimensions," *The Annals of Statistics*, 41, 2905–2947. [11]

Wang, X., Yang, Z., Chen, X., and Liu, W. (2019), "Distributed Inference for Linear Support Vector Machine," *Journal of Machine Learning Research*, 20, 1–41. [1]

Wen, Z., and Yin, W. (2013), "A Feasible Method for Optimization With Orthogonality Constraints," *Mathematical Programming*, 142, 397–434. [4]

Xu, Z. (2018), "Gradient Descent Meets Shift-and-Invert Preconditioning for Eigenvector Computation," in *Advances in Neural Information Processing Systems (NIPS)*. [2]

Yang, Z., Balasubramanian, K., and Liu, H. (2017), "On Stein's Identity and Near-Optimal Estimation in High-Dimensional Index Models," arXiv no. 1709.08795. [3]

Yu, Y., Wang, T., and Samworth, R. J. (2014), "A Useful Variant of the Davis–Kahan Theorem for Statisticians," *Biometrika*, 102, 315–323. [2]

Zhang, Y., Duchi, J., and Wainwright, M. (2015), "Divide and Conquer Kernel Ridge Regression: A Distributed Algorithm With Minimax Optimal Rates," *Journal of Machine Learning Research*, 16, 3299–3340. [1]

Zhao, T., Cheng, G., and Liu, H. (2016), "A Partially Linear Framework for Massive Heterogeneous Data," *The Annals of Statistics*, 44, 1400–1437. [1]