FAST-PCA: A Fast and Exact Algorithm for Distributed Principal Component Analysis

Arpita Gang and Waheed U. Bajwa

Abstract—Principal Component Analysis (PCA) is a fundamental data preprocessing tool in the world of machine learning. While PCA is often thought of as a dimensionality reduction method, the purpose of PCA is actually two-fold: dimension reduction and uncorrelated feature learning. Furthermore, the enormity of the dimensions and sample size in the modern day datasets have rendered the centralized PCA solutions unusable. In that vein, this paper reconsiders the problem of PCA when data samples are distributed across nodes in an arbitrarily connected network. While a few solutions for distributed PCA exist, those either overlook the uncorrelated feature learning aspect of the PCA, tend to have high communication overhead that makes them inefficient and/or lack 'exact' or 'global' convergence guarantees. To overcome these aforementioned issues, this paper proposes a distributed PCA algorithm termed *FAST-PCA (Fast and exAct diSTributed PCA)*. The proposed algorithm is efficient in terms of communication and is proven to converge linearly and exactly to the principal components, leading to dimension reduction as well as uncorrelated features. The claims are further supported by experimental results.

Index Terms—Dimension reduction, distributed learning, exact convergence, Krasulina's method, principal component analysis

I. INTRODUCTION

Massive and high-dimensional datasets are becoming an increasingly essential part of the modern world ranging from healthcare to finance, from social media to the Internet-of-Things (IoT) [3], from chemometrics [4] to image and video processing [5], etc. In a related trend, machine learning algorithms are finding their applications in every possible domain because of their data-driven nature and the ability to generalize to new unseen data. But these algorithms need a considerable amount of data preprocessing for their effective and efficient use. One of the major steps in this preprocessing is dimension reduction and feature learning for compression and extraction of useful features from raw data that can be used in downstream machine learning algorithms for classification, clustering, etc. Principal Component Analysis (PCA) [6] is a workhorse tool for such dimension reduction and feature extraction purposes. In a nutshell, PCA transforms a large set

of correlated features to a smaller set of uncorrelated features that contain maximum information of the raw data.

The increasing volume of available data along with concerns like privacy, communication cost, etc., as well as emerging applications such as smart cities, autonomous vehicles, etc., have also led to a significant interest in the last couple of decades in the development of distributed algorithms for PCA on non-collocated data [7]. Data tends to be distributed for a multitude of reasons; it can be inherently distributed like in IoT, sensor networks, etc., or it can be distributed due to storage and/or computational limitations. The ultimate goal of any distributed algorithm is to solve a common problem using data shared among the distributed entities through communication with each other so that all entities collectively reach a solution that is nearly as good as the solution of the centralized algorithms, for which data is available at a single location. Motivated by these reasons, we develop and analyze an effective solution for distributed PCA that is efficient in terms of communication, that does not require exchange of raw data, and that can be proved to converge exactly for any arbitrary network topology and at a linear rate to a solution that is the same as the one returned by *centralized* PCA.

Distributed setups can be largely classified into two types: $i)$ those having a central entity/server that coordinates with the other nodes in a master-slave architecture, and $ii)$ those lacking any central entity, in which the nodes are connected in an arbitrary network. In the first type of setup, the central entity aggregates information from all the nodes and yields the final result. Since the second type of architecture does not rely on any central entity, it is a more general setup and it lacks a single point of failure. The detailed review in [8] discusses these setups along with various algorithms developed for both in more detail. Although the terms distributed and decentralized are used interchangeably for both setups in the literature, we consider the latter scenario for the distributed PCA and call it *distributed* in this paper.

The goal of dimension reduction can be accomplished by learning a low-dimensional subspace spanned by the dominant eigenvectors of the covariance matrix of the distribution to which the data samples belong. Mathematically speaking, for a data point $\mathbf{y} \in \mathbb{R}^d$ sampled from a distribution with zero mean and covariance $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$, dimension reduction can be achieved by projecting $\mathbf{y}$ onto a matrix $\mathbf{X} \in \mathbb{R}^{d \times K}, K \ll d$, such that $\mathbf{X}$ spans a subspace spanned by the leading K eigenvectors of $\mathbf{\Sigma}$ under the constraint $\mathbf{X}^T \mathbf{X} = \mathbf{I}$, that is $\mathbf{X}$ lies on a Stiefel manifold. When $\mathbf{y}$ is compressed as $\tilde{\mathbf{y}} = \mathbf{X}^T \mathbf{y}$ with such an $\mathbf{X}$, its reconstruction $\mathbf{X}\mathbf{X}^T\mathbf{y}$ has minimum error in the Frobenius norm sense. However, this approach can

only be called *principal subspace analysis* as it does not ensure that the resultant K features in $\tilde{\mathbf{y}}$ are uncorrelated. It has been argued in the literature (see, e.g., [9]) that one of the factors that makes a compressed representation of a data sample "good" is having uncorrelated features in the learned representation. Different explanatory features of the data tend to change independently of each other in the input distribution in the real-world settings. This implies that if the learned representations have uncorrelated features, changes or noise in one will not affect the others. Correlated features bring redundant information and in turn lead to unnecessary increase in dimension of learned representations. This ultimately can have consequences in downstream machine learning models. For example, random forests can be good at detecting interactions between different features, but highly correlated features can mask these interactions. Hence, learning uncorrelated feature representation has gained significant attraction in feature learning lately. This uncorrelatedness constraint requires $\mathbb{E}\left[\tilde{\mathbf{y}}\tilde{\mathbf{y}}^T\right] = \mathbb{E}\left[\mathbf{X}^T\mathbf{y}\mathbf{y}^T\mathbf{X}\right]$ to be a diagonal matrix, which is fulfilled only when $\mathbf{X}$ contains the eigenvectors of $\boldsymbol{\Sigma}$, not just any orthogonal basis of the subspace spanned by the said eigenvectors. The true purpose of PCA is thus fulfilled by a specific element of the Stiefel manifold that corresponds to the eigenvectors of $\boldsymbol{\Sigma}$.

An autoencoder is another popular neural-network based tool for data compression. The good generalization capability of neural network-based systems along with their ease of parallelization in the case of massive data make them very attractive and efficient solutions for PCA. A study in [10] showed that the optimum weights of an autoencoder for efficient data compression and decorrelation of features, when the loss function is the reconstruction error, are given by the space spanned by the eigenvectors of the input covariance matrix. It was also noted in [11], [12] that neural networks trained using the Hebbian learning rule [13] extract principal components of the input correlation matrix in the streaming data case. In the same setting of streaming data, an earlier work by Krasulina [14] proposed a similar learning method that converges to the dominant eigenvector of the expectation of input sample covariance matrix. Even though the methods proposed by Krasulina [14] and Oja [11] have many similarities as pointed out in [11], [15], Oja's [11] rule has been studied more extensively than Krasulina's [14] method. The original Krasulina's method is a simple iterative method for the estimation of the top eigenvector in the streaming case and its matrix version called Matrix Krasulina was proposed much later in [16] that extends the original method to estimate the subspace spanned by the top K eigenvectors. Since we aim to find the top K eigenvectors, in this paper we propose a learning method based on the original Krasulina's method that can be shown to converge to the first K eigenvectors (principal components) of the sample covariance matrix, not just the principal subspace, in distributed batch settings. Due to the parallelization potential and an iterative update-based rule, our proposed method is applicable to autoencoder training as well.

A. Relation to Prior Work

The problem of dimension reduction goes back as early as 1901 when Pearson [17] aimed at fitting a line to a set of data points. Later, Hotelling [6] proposed a PCA method for decorrelating and compressing a set of data points by finding their principal components. Since then, many iterative methods like power method, orthogonal iterations [18], and Lanczos method [19] have been proposed to estimate eigenvectors or low-dimensional subspaces of symmetric matrices, a class under which covariance matrices fall. A stochastic approximation algorithm was proposed by Krasulina in [14] for the estimation of the dominant eigenvector in the streaming data case. From the point of view of training neural networks for data compression, an algorithm very similar to Krasulina's method was later proposed by Oja [11], which was then extended for multiple eigenvector estimation by Sanger [12]. Both Oja's and Sanger's method were based on the Hebbian learning rule [13] and it was shown that the weights of an autoencoder trained using this rule converge to the eigenvectors of the input correlation matrix. The works in [20], [21] proved that in deterministic batch settings Oja's rule and the generalized Hebbian rule proposed by Sanger converge to the eigenvectors of a covariance matrix at a linear rate. Krasulina's method was also generalized for the estimation of a subspace of dimension greater than one in [16], although it only guarantees convergence to the principal subspace, instead of principal components, at a linear rate under the low-rank matrix assumption.

Modern data have various aspects that require looking at the PCA problem from different lenses. For example, the presence of outliers or corrupted data led to the development of robust PCA solutions [22]–[24] or the case of sparse PCA [25], [26] when principal components are assumed to be sparse. The problem of PCA in the distributed/decentralized setting is relatively recent. In the decentralized setting, in the presence of a central server, the client nodes do some computations using their local data and the server node aggregates the information from all nodes before passing it back. The works in [27], [28] proposed to perform a local Singular Value Decomposition (SVD) at each node using their partial data, which is then aggregated at the central node. The work proposed in [29] used decentralized PCA to detect anomalies in wireless sensor networks. Aggregation at a central server raises privacy issues, which is tackled in [30] that introduces a differentially private distributed PCA algorithm. Several other works have been proposed for decentralized PCA; e.g., in the case of streaming data [31], in the case of large-scale process monitoring [32], in the case of federated learning [33], etc.

The distributed setting, where nodes are connected in an arbitrary manner, is the main focus of this paper. In any distributed network, data can be distributed by either features or samples and the solutions for these two data distribution types are significantly different. A detailed review of various distributed PCA algorithms for both kinds of data distribution is done in [34]. For the case of feature-wise distribution as in [35]–[37], each node in the network estimates one or a subset of features of the entire subspace. In this paper we focus on

the case of sample-wise data distribution, where each node estimates the entire basis and consensus in the network is a necessary condition. The sample-wise data distribution was considered in [38]–[40], where a power method-based approach was proposed for estimation of the dominant eigenvector ($K = 1$). This method requires an explicit consensus loop [41] in every iteration of the power method and the final error is a function of the number of consensus iterations. The power method-based distributed PCA solutions can be used for multiple ($K > 1$) eigenvector estimation in a sequential manner, where lower-order eigenvectors are estimated using the residue of the covariance matrix left after its projection on the higher-order eigenvectors. Since estimation of any lower-order eigenvector requires that the higher-order eigenvectors are fully estimated, this sequential approach results in a rather slow algorithm. To overcome the issues of the sequential approach, an orthogonal iteration-based solution for the case of $K > 1$ was proposed in [42]. Although this method estimates the K-dimensional subspace simultaneously, its convergence guarantees are in terms of subspace angles and thus it proves convergence to the principal subspace. Moreover, all these aforementioned methods require an explicit consensus loop, making these algorithms inefficient in terms of communication overhead.

PCA is a non-convex problem since the uncorrelated constraint requires the solution to be a specific element on the Stiefel manifold. Recently, some algorithms in the field of distributed optimization were proposed to deal with non-convex problems. While some of those deal with unconstrained problems [43], some are developed for non-convex objectives with convex constraints [44], [45], while some methods guarantee convergence only to a stationary point [46]. For these reasons, none of the existing distributed algorithms for non-convex problems are directly applicable for the PCA objective. A recent work based on perturbation theory for linear operators based on the Picard iteration was proposed for distributed optimization in [47]. The extension of this work in [48] demonstrated the application of the distributed Picard iteration (DPI) method to distributed PCA, but it could only prove local convergence, i.e., if the estimate is already "close enough" to the optimal solution, then it converges to the optimal point at a linear rate. Furthermore, the DPI method suffers from two more limitations in terms of its theoretical analysis, namely, it requires the covariance matrix to be full rank, and the upper bound on the step size required for convergence guarantees is not quantified in terms of problem parameters like eigengap, data dimension, etc. Thus, many gaps still remain to be filled in distributed PCA.

The work in this paper is an extension of our preliminary work in [1] that proposed two fast and efficient algorithms for distributed PCA but did not provide any theoretical guarantees. Both these algorithms were based on the generalized Hebbian algorithm in the case of sample-wise distributed data. The first version called distributed Sanger's algorithm (DSA) used a combine-and-adapt strategy, which was further developed and analyzed in our previous work [49]. Although this strategy has mainly been used in distributed optimization literature for convex and strongly convex problems, we showed using extensive analysis that even for the non-convex PCA problem, each node converges linearly and globally, i.e., starting from any random initial point. Though it is a linearly convergent one-time scale algorithm, it only reaches to a neighborhood of the optimal solution for a fixed step size. The algorithm, however, does converge exactly in the case of decreasing step sizes but with a slower rate of convergence. This result is coherent with the combine-and-update based gradient descent solutions [50] for distributed optimization. To overcome such limitations of simple gradient descent-based algorithms, some new methods have been proposed recently that deploy a technique called "gradient-tracking", which has been shown to converge exactly in the case of convex [51], [52], strongly convex and some non-convex problems [53]. In this paper, we use this gradient-tracking idea to develop an algorithm for the *non-convex* distributed PCA problem that linearly converges to an optimal solution that is the same as its centralized counterpart.

A very recent paper on distributed PCA [54] used this gradient-tracking idea to develop a two-time scale algorithm called DeEPCA for subspace estimation. Our work has three major differences as compared to DeEPCA: firstly, our algorithm guarantees convergence to the eigenvectors of the global covariance matrix and not just any rotated basis of the same subspace, thereby making our algorithm a true PCA and not just a principal subspace analysis (PSA) solution. Secondly, we do not use any explicit consensus loop for ensuring agreement in the network, making it a very communication-efficient solution and finally, DeEPCA requires explicit QR decomposition in every iteration unlike our algorithm, thus requiring more computations. Table I shows the convergence rates of the important PCA/PSA algorithms for the case of sample-wise distributed data. The table provides a comparison of the communication and iteration complexities of various distributed PCA (principal component analysis) and PSA (principal subspace analysis) algorithms in terms of error ϵ and eigengap gap. If λ_l is the lth eigenvalue of the data covariance matrix, then $gap_r := \frac{\lambda_{K+1}}{\lambda_K}$ for PSA and $gap_r := \max_{k=1,...,K} \frac{\lambda_{k+1}}{\lambda_k}$ for PCA algorithms. Also, $gap := \lambda_K - \lambda_{K+1}$ for PSA algorithms and $gap := \min_{k=1,...,K} \lambda_k - \lambda_{k+1}$ for PCA algorithms.

B. Our Contributions

The main contributions of this paper are 1) a novel algorithm for distributed PCA called *Fast and exAct diSTributed PCA* (FAST-PCA) based on a generalization of Krasulina's method, 2) theoretical guarantees that show that the estimates given by our method converge exactly and globally at a linear rate to the eigenvectors of the global covariance matrix, and 3) experimental results that further demonstrate the efficiency of our solution for both synthetic and real-world datasets.

Our primary focus in this paper is to develop a solution for distributed PCA when the data samples are scattered across an arbitrarily connected network with no central node. While PCA is often reduced to dimension reduction, we focus on the dual goal of PCA that requires dimensionality reduction as well as feature decorrelation. To that end, we propose an algorithm based on Krasulina's method using a gradient-tracking approach. Since the original Krasulina's method only finds the dominant eigenvector, we also generalize it to the

TABLE I: Comparison of Communication and Iteration Costs for State-of-the-Art PCA/PSA Solutions

	Comm./Iteration	No. of Iterations	Total Comm.	PCA/PSA
DistSeqPM [38]–[40]	$\mathcal{O}\left(K\frac{1}{\log gap_r^{-1}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(K\frac{1}{\log gap_r^{-1}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(K^2\frac{1}{\log^2 gap_r^{-1}}\log^2\frac{1}{\epsilon}\right)$	PCA
S-DOT [42]	$\mathcal{O}\left(\frac{1}{\log gap_r^{-1}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{\log gap_r^{-1}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{\log^2 gap_r^{-1}}\log^2\frac{1}{\epsilon}\right)$	PSA
DeEPCA [54]	$\mathcal{O}\left(\log\frac{1}{gap}\right)$	$\mathcal{O}\left(\frac{1}{gap}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{gap}\log\frac{1}{gap}\log\frac{1}{\epsilon}\right)$	PSA

distributed setting for the estimation of top K eigenvectors. Our proposed FAST-PCA method is an iterative update algorithm and its main attributes are that it is fast since it lacks any explicit consensus loop and hence reduces the communication overhead, and it converges exactly to the true eigenvectors of the global covariance matrix at a linear rate. We provide detailed convergence analysis to support our claims as well as extensive numerical experiments where we compare our method to centralized orthogonal iteration (OI) as the centralized baseline, as well as distributed PCA algorithms of sequential distributed power method (SeqDistPM), DeEPCA and DSA. We provide the results for different network topologies as well as eigengaps to further solidify our claims.

To the best of our knowledge, this is the first novel algorithm for distributed PCA based on Krasulina's method that achieves fast and exact convergence to the true eigenvectors of the global covariance matrix at every node of an arbitrarily connected network.

C. Notation and Organization

The following notation is used in this paper. Scalars and vectors are denoted by lower-case and lower-case bold letters, respectively, while matrices are denoted by upper-case bold letters. The operator $|\cdot|$ denotes the absolute value of a scalar quantity. The superscript in $\mathbf{a}^{(t)}$ denotes time (or iteration) index, while a^t denotes the exponentiation operation. The superscript $(\cdot)^{\mathrm{T}}$ denotes the transpose operation, the operator $\otimes$ denotes Kronecker product, $\|\cdot\|_F$ denotes the Frobenius norm of matrices, while both $\|\cdot\|$ and $\|\cdot\|_2$ denote the ℓ_2-norm of vectors. Given a matrix $\mathbf{A}$, both a_{ij} and $(\mathbf{A})_{ij}$ denote its entry at the ith row and j^{th} column, while $\mathbf{a}_j$ denotes its jth column. The matrix $\mathbf{I}_a \in \mathbb{R}^{a\times a}$ denotes the identity matrix of dimension a.

The rest of the paper is organized as follows. In Section II, we describe and mathematically formulate the distributed PCA problem, while Section III describes the proposed distributed algorithm, which is based on Krasulina's algorithm. In Section IV-B, we derive an auxiliary result based on Krasulina's method that aids in the convergence analysis of the proposed distributed algorithm, while convergence guarantees for the proposed algorithm are provided in Section IV. Statements and/or proofs of the key lemmas used to derive the main results of this paper are provided as appendices. We provide numerical results in Section V to show efficacy of the proposed method and provide concluding remarks in Section VI.

II. PROBLEM DESCRIPTION

Principal Component Analysis (PCA) is a widely used data preprocessing tool to find a low-dimensional subspace that would decorrelate data features while retaining maximum information. For data samples $\mathbf{y} \in \mathbb{R}^d$ sampled from a zero-mean distribution with covariance matrix $\mathbf{\Sigma}$, PCA can be mathematically formulated as

$$\mathbf{X} = \underset{\mathbf{X}\in\mathbb{R}^{d\times K},\mathbf{X}^{\mathrm{T}}\mathbf{X}=\mathbf{I}}{\arg\min} \quad \mathbb{E}\left[\|\mathbf{y}-\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{y}\|_2^2\right]$$
$$\text{such that} \quad \forall l \neq q, \ \left(\mathbb{E}\left[\mathbf{X}^{\mathrm{T}}\mathbf{y}\mathbf{y}^{\mathrm{T}}\mathbf{X}\right]\right)_{lq} = 0. \quad (1)$$

The constraint $\left(\mathbb{E}\left[\mathbf{X}^{\mathrm{T}}\mathbf{y}\mathbf{y}^{\mathrm{T}}\mathbf{X}\right]\right)_{lq} = 0, \forall l \neq q$, ensures that $\mathbf{X}$ decorrelates the features of $\mathbf{y}$. It is evident that $\mathbb{E}\left[\mathbf{X}^{\mathrm{T}}\mathbf{y}\mathbf{y}^{\mathrm{T}}\mathbf{X}\right] = \mathbf{X}^{\mathrm{T}}\mathbb{E}\left[\mathbf{y}\mathbf{y}^{\mathrm{T}}\right]\mathbf{X}$ will be a diagonal matrix if and only if $\mathbf{X}$ contains the eigenvectors of $\mathbb{E}\left[\mathbf{y}\mathbf{y}^{\mathrm{T}}\right] = \mathbf{\Sigma}$. Thus the search for a solution of PCA not only requires a minimum reconstruction error solution, which will be given by any basis of the subspace spanned by the dominant K eigenvectors of the covariance matrix $\mathbf{\Sigma}$, but the basis vectors should specifically be the eigenvectors of $\mathbf{\Sigma}$. In practice the actual distribution of the samples and hence $\mathbf{\Sigma}$ is unknown and a sample covariance matrix is used instead for PCA. For a set of samples $\{\mathbf{y}_t\}_{t=1}^N$, the sample covariance matrix is given by $\mathbf{C} = \frac{1}{N-1}\sum_{t=1}^N(\mathbf{y}_t-\bar{\mathbf{y}})(\mathbf{y}_t-\bar{\mathbf{y}})^T$, where $\bar{\mathbf{y}} = \frac{1}{N}\sum_{t=1}^N\mathbf{y}_t$ is the sample mean. Henceforth, we shall assume $\bar{\mathbf{y}} = 0$ without loss of generality because the mean can otherwise be calculated and subtracted from the samples. The empirical formulation of the PCA problem in terms of samples is thus given as

$$\mathbf{X} = \underset{\mathbf{X}\in\mathbb{R}^{d\times K},\mathbf{X}^{\mathrm{T}}\mathbf{X}=\mathbf{I}}{\arg\min} \quad \sum_{t=1}^N\|\mathbf{y}_t-\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{y}_t\|_2^2$$
$$\text{such that} \quad \forall l \neq q, \ \left(\mathbf{X}^{\mathrm{T}}(\sum_{t=1}^N\mathbf{y}_t\mathbf{y}_t^{T})\mathbf{X}\right)_{lq} = 0. \quad (2)$$

A distributed setting implies that the entire data matrix $\mathbf{Y} = \left[\mathbf{y}_1,\ldots,\mathbf{y}_N\right] \in \mathbb{R}^{d\times N}$ is unavailable at a single location. Let us consider an undirected and connected network of M nodes described by a graph $\mathcal{G} = \{\mathcal{V},\mathcal{E}\}$, where $\mathcal{V} = \{1,\ldots,M\}$ is the set of nodes and $\mathcal{E}$ is the set of edges between the nodes. For each node i, the set of its directly connected neighbors is given by $\mathcal{N}_i$. The data can be distributed among the nodes along the rows, i.e., by features, or along the columns, i.e., by samples. In this paper, we consider the case when the samples $\{\mathbf{y}_t\}_{t=1}^N$ are scattered spatially over a network. Thus, each node $i \in \mathcal{V}$ has a non-overlapping subset of the samples $\mathbf{Y}_i \in \mathbb{R}^{d\times N_i}$ such that $\mathbf{Y} = \left[\mathbf{Y}_1,\ldots,\mathbf{Y}_M\right]$. The PCA formulation in this distributed case is:

$$\mathbf{X} = \underset{\mathbf{X}\in\mathbb{R}^{d\times K},\mathbf{X}^{\mathrm{T}}\mathbf{X}=\mathbf{I}}{\arg\min} \quad \sum_{i=1}^M\|\mathbf{Y}_i-\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{Y}_i\|_F^2$$

such that
$$\forall l \neq q, \left(\mathbf{X}^{\mathrm{T}}(\sum_{i=1}^{M} \mathbf{Y}_i \mathbf{Y}_i^{\mathrm{T}})\mathbf{X}\right)_{lq} = 0. \quad (3)$$

Although the formulations (2) and (3) look similar, a major difference is the unavailability of $\mathbf{Y}_i$'s at a single location, rendering the methods for solving (2) unusable directly for solving (3). Since each node carries different local data, there is a difference in local objective function even though the constraint is globally shared. This in turn leads to each node maintaining its own copy $\mathbf{X}_i$ of the variable $\mathbf{X}$. As mentioned before, the goal of distributed PCA is for each node to eventually reach the same solution, i.e., achieve network consensus, given by the eigenvectors of $\mathbf{C}$. Thus, the actual PCA objective for the distributed case is

$$\underset{\mathbf{X}_i \in \mathbb{R}^{d \times K}, \mathbf{X}_i^{\mathrm{T}} \mathbf{X}_i = \mathbf{I}}{\arg \min} \quad \sum_{i=1}^{M} \|\mathbf{Y}_i - \mathbf{X}_i \mathbf{X}_i^{\mathrm{T}} \mathbf{Y}_i\|_F^2 \quad \text{such that}$$

$$\forall j \in \mathcal{N}_i, \mathbf{X}_i = \mathbf{X}_j \quad \text{and} \quad \forall l \neq q, \left(\mathbf{X}_i^{\mathrm{T}}(\sum_{i=1}^{M} \mathbf{Y}_i \mathbf{Y}_i^{\mathrm{T}})\mathbf{X}_i\right)_{lq} = 0.$$
$$(4)$$

Since each node i has access to a subset of data points $\mathbf{Y}_i$ and subsequently has a local covariance matrix $\mathbf{C}_i = \frac{1}{N_i}\mathbf{Y}_i\mathbf{Y}_i^{\mathrm{T}}$, a naive solution is that each node solves its own PCA formulation as follows:

$$\mathbf{X}_i = \underset{\mathbf{X}_i \in \mathbb{R}^{d \times K}, \mathbf{X}_i^{\mathrm{T}} \mathbf{X}_i = \mathbf{I}}{\arg \min} \quad \|\mathbf{Y}_i - \mathbf{X}_i \mathbf{X}_i^{\mathrm{T}} \mathbf{Y}_i\|_F^2$$

$$\text{such that} \quad \forall l \neq q, \left(\mathbf{X}_i^{\mathrm{T}} \mathbf{Y}_i \mathbf{Y}_i^{\mathrm{T}} \mathbf{X}_i\right)_{lq} = 0. \quad (5)$$

However, the naive solution of (5) will have major drawbacks. As explained earlier, PCA ideally aims to find the eigenvectors of covariance $\boldsymbol{\Sigma}$ of the distribution the data points are sampled from but instead uses sample covariance matrix $\mathbf{C}$ because $\boldsymbol{\Sigma}$ is unknown in practice and $\mathbb{E}\left[\mathbf{C}\right] = \boldsymbol{\Sigma}$. Since $\mathbf{C} \to \boldsymbol{\Sigma}$ as the number of samples N increases, using only the local covariance matrices would incur a higher loss in the estimation of the eigenvectors. Furthermore, it is plausible that the samples at a single node are not uniformly sampled from the entire distribution and hence any estimation made using local covariances would result in a biased estimate. These reasons dictate that all the N samples in the network should be incorporated somehow in the estimation of the eigenvectors for dimension reduction and decorrelation at all the nodes of the network. Additionally, in the case of sample-wise distributed data, all nodes should agree and converge to a common solution that is the same as the solution of (2) when all the samples are available at a single location.

The constraint in (4) has two important properties. First, since the solution lies on the Stiefel manifold and particularly, it is a specific element of the manifold, the problem is non-convex. Although this issue can be dealt with through convex approximation of the problem [55], such an approach will result in $\mathcal{O}(d^2)$ computational and memory requirements since it approximates the projection matrix of the $d \times K$ dimensional subspace and that can be restrictive in the case of high-dimensional data. At the same time, such convexification leads to a relaxed constraint that would only give a rotated

basis of the subspace spanned by the eigenvectors of $\mathbf{C}$ and not the eigenvectors themselves. Second, the constraint $\mathbf{X}_i^{\mathrm{T}}(\sum_{i=1}^{M} \mathbf{Y}_i \mathbf{Y}_i^{\mathrm{T}})\mathbf{X}_i$ being diagonal is shared by all nodes due to the reasons explained earlier. Thus meeting this global constraint requires that all nodes of the network collaborate to reach a common solution $\mathbf{X} = \mathbf{X}_i, \forall i \in \mathcal{V}$. Hence, in this paper we propose an iterative algebraic method based on Krasulina's rule [14] for distributed PCA that ensures that all nodes simultaneously converge to the eigenvectors of the global covariance matrix $\mathbf{C}$ without having to share their local covariance $\mathbf{C}_i$. The algorithm converges exactly to the eigenvectors of the global covariance matrix $\mathbf{C}$ at a linear rate when the error is measured in terms of angles between the estimates and the true eigenvectors.

III. PROPOSED ALGORITHM: FAST-PCA

Iterative solutions such as the power method, Oja's rule, and Krasulina's method have proven to be powerful tools for PCA, i.e., dimension reduction and simultaneous feature decorrelation in centralized settings when the data is collocated or streaming at a single location. Although Krasulina's and Oja's method have similar update rules, in this paper we extend the Krasulina's method to develop an algorithm for distributed PCA in batch settings. The original Krasulina's method was developed as a stochastic approximation algorithm for estimating the dominant eigenvector of the expected correlation matrix (which is the same as the covariance matrix for zero-mean inputs) in the case of streaming data. Let $\mathbf{y}_t, t = 1, 2, \ldots$, be data samples drawn from a zero-mean distribution at time t. Then Krasulina's method estimated the leading eigenvector of $\boldsymbol{\Sigma} = \mathbb{E}\left[\mathbf{y}_t\mathbf{y}_t^{\mathrm{T}}\right]$ by the following update equation:

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \alpha_t \left(\mathbf{C}_t \mathbf{x}^{(t)} - \frac{(\mathbf{x}^{(t)})^{\mathrm{T}} \mathbf{C}_t \mathbf{x}^{(t)}}{\|\mathbf{x}^{(t)}\|^2} \mathbf{x}^{(t)}\right), \quad (6)$$

where $\mathbf{C}_t = \mathbf{y}_t\mathbf{y}_t^{\mathrm{T}}$ is the covariance matrix obtained from one sample and α_t is the step size at time t. It was proved in [14] that if the spectral norm of $\mathbb{E}\left[\mathbf{y}_t\mathbf{y}_t^{\mathrm{T}}\right]$ remains bounded and $\sum_t \alpha_t^2$ converges to zero as $t \to \infty$, the update equation (6) yields the dominant eigenvector of $\mathbb{E}\left[\mathbf{C}_t\right]$. One can interpret Krasulina's method as the solution to an optimization problem. The estimation of the top eigenvector can often be posed as the following optimization problem:

$$\underset{\mathbf{x} \in \mathbb{R}^d}{\arg \min} \quad f(\mathbf{x}) = \underset{\mathbf{x} \in \mathbb{R}^d}{\arg \min} - \frac{\mathbf{x}^{\mathrm{T}} \mathbf{C}_t \mathbf{x}}{\|\mathbf{x}\|^2} \quad (7)$$

The gradient of the function $f(\mathbf{x})$ in (7) is:

$$\nabla f(\mathbf{x}) = \frac{1}{\|\mathbf{x}\|^2} \left(- \mathbf{C}_t \mathbf{x}^{(t)} + \frac{(\mathbf{x}^{(t)})^{\mathrm{T}} \mathbf{C}_t \mathbf{x}^{(t)}}{\|\mathbf{x}^{(t)}\|^2} \mathbf{x}^{(t)}\right). \quad (8)$$

Thus, (6) looks similar to applying stochastic gradient descent to the nonconvex problem (7) where a step is taken in the direction of negative of the gradient of the function but the size of the step is scaled with the magnitude of the norm $\|\mathbf{x}\|^2$.

In the distributed setup considered in this paper, samples are not streaming but distributed across a connected network of M nodes, where node i has access to a local covariance matrix $\mathbf{C}_i$ such that $\sum_{i=1}^{M} \mathbf{C}_i = \mathbf{C}$, the global covariance

matrix. It is noteworthy that $\mathbb{E}\left[\mathbf{C}_t\right] = \mathbb{E}\left[\mathbf{C}_i\right] = \mathbf{\Sigma}$ and this similarity between streaming and distributed setting motivates the extrapolation of Krasulina's method for the distributed setting. For the dominant eigenvector $K = 1$, a naive approach would be for each node to estimate an eigenvector using its local data and update rule (6). However, that would result in each node i to only estimate the dominant eigenvector of $\mathbf{C}_i$ whereas the goal of distributed PCA is for every node to estimate the eigenvector of the global covariance matrix $\mathbf{C}$. Furthermore, since Matrix Krasulina [16] only estimates the dominant subspace, Krasulina's method also needs to be generalized for the estimation of $K > 1$ dominant *eigenvectors*.

Estimation of the eigenvectors of $\mathbf{C}$ at every node without sharing raw local covariance matrix $\mathbf{C}_i$ would require some form of collaboration among the nodes of the network. As mentioned earlier, our previous work [49] used a combine-and-adapt strategy in a way that each node converges linearly but only to a neighborhood of the true eigenvectors of the global covariance matrix $\mathbf{C}$. Even though we used the generalized Hebbian algorithm [12], some straightforward calculations and manipulations can show similar results for Krasulina's method. In this paper, we aim to fill that gap of inexact convergence and propose a gradient-tracking based solution [51], [52] that converges exactly and linearly to the true eigenvectors of $\mathbf{C}$ at every node. If $\mathbf{x}_{i,1}^{(t)}$ is the estimate of the dominant eigenvector at node i after the tth iteration, then we define a pseudo-gradient at node i as follows:

$$\mathbf{h}_i(\mathbf{x}_{i,1}^{(t)}) = \mathbf{C}_i\mathbf{x}_{i,1}^{(t)} - \frac{(\mathbf{x}_{i,1}^{(t)})^T\mathbf{C}_i\mathbf{x}_{i,1}^{(t)}}{\|\mathbf{x}_{i,1}^{(t)}\|^2}\mathbf{x}_{i,1}^{(t)}, \qquad (9)$$

which is similar to the update portion of (6). We call this entity pseudo-gradient as this differs from how the gradient would look like (refer to (8)) at node i by a factor of $\frac{1}{\|\mathbf{x}_{i,1}^{(t)}\|^2}$. Additionally, for the estimation of $k^{th}, k = 2, \ldots, K$, eigenvector, we propose to generalize Krasulina's update rule along the lines of the generalized Hebbian algorithm [12] and combine Krasulina's method with Gram–Schmidt orthogonalization to define a general pseudo-gradient as:

$$\mathbf{h}_i(\mathbf{x}_{i,k}^{(t)}) = \mathbf{C}_i\mathbf{x}_{i,k}^{(t)} - \frac{(\mathbf{x}_{i,k}^{(t)})^T\mathbf{C}_i\mathbf{x}_{i,k}^{(t)}}{\|\mathbf{x}_{i,k}^{(t)}\|^2}\mathbf{x}_{i,k}^{(t)}$$
$$- \sum_{p=1}^{k-1}\frac{(\mathbf{x}_{i,p}^{(t)})^T\mathbf{C}_i\mathbf{x}_{i,k}^{(t)}}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\mathbf{x}_{i,p}^{(t)}. \qquad (10)$$

Here, the term $\frac{(\mathbf{x}_{i,p}^{(t)})^T\mathbf{C}_i\mathbf{x}_{i,k}^{(t)}}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\mathbf{x}_{i,p}^{(t)}$ is analogous to Gram–Schmidt orthogonalization and enforces the orthogonality of $\mathbf{x}_{i,k}^{(t)}$ to $\mathbf{x}_{i,p}^{(t)}, p = 1, \ldots, k-1$.

Let $\mathbf{X}_i^{(t)} = \left[\mathbf{x}_{i,1}^{(t)}, \ldots, \mathbf{x}_{i,K}^{(t)}\right] \in \mathbb{R}^{d\times K}$ be the estimate of the K eigenvectors of the global covariance matrix $\mathbf{C}$. A gradient tracking-based algorithm also updates a second variable [52], [53] in every iteration that essentially tracks the average of the gradients at the nodes. In a similar fashion, let us define a *pseudo-gradient tracker* matrix $\mathbf{S}_i^{(t)} = \left[\mathbf{s}_{i,1}^{(t)}, \ldots, \mathbf{s}_{i,K}^{(t)}\right] \in \mathbb{R}^{d\times K}$ that tracks the average of the pseudo-gradients at

each node. These $\mathbf{S}_i^{(t)}$ are updated along with the eigenvector estimates $\mathbf{X}_i^{(t)}$ in each iteration of our algorithm *Fast and exAct diSTributed PCA (FAST-PCA)*, which is described in Algorithm 1. At each node i, the eigenvector estimates $\mathbf{X}_j^{(t)}, j \in \mathcal{N}_i$, where $\mathcal{N}_i$ is the set of neighbors of node i, are combined as a weighted average and updated with the local copy of the gradient tracker $\mathbf{S}_i^{(t)}$ using a constant step size α. Along with that, $\mathbf{S}_i^{(t)}$ is also updated as a weighted average of $\mathbf{S}_j^{(t)}$ and difference of pseudo-gradients. The entity $\mathbf{h}_i(\mathbf{X}_i^{(t)})$ in the algorithm is the matrix of the psuedo-gradients, i.e., $\mathbf{h}_i(\mathbf{X}_i^{(t)}) = \left[\mathbf{h}_i(\mathbf{x}_{i,1}^{(t)}), \ldots, \mathbf{h}_i(\mathbf{x}_{i,K}^{(t)})\right] \in \mathbb{R}^{d\times K}$. The weight matrix $\mathbf{W} = \left[w_{ij}\right]$ is a doubly stochastic matrix that conforms to the underlying graph topology [56], i.e., $w_{ij} \neq 0$ if $(i,j) \in \mathcal{E}$ or $i = j$ and 0 otherwise. A necessary assumption for convergence of the algorithm here is the graph connectivity, which ensures that the magnitude of the second largest eigenvalue of $\mathbf{W}$ is strictly less than 1. The gradient-tracking based solutions are recently being very popular in distributed optimization literature because of their fast and exact convergence guarantees. Our main challenge here was providing theoretical convergence guarantees inspite of the non-convex nature of the problem. In the next section, we provide detailed analysis of our proposed algorithm FAST-PCA and show that the estimates $\mathbf{x}_{i,k}^{(t)}$ at each node i converge at a linear rate $\mathcal{O}(\rho^t), 0 < \rho < 1$, for any random unit-norm initialization and a certain condition on step size, to the eigenvectors $\pm\mathbf{q}_k$ of the global covariance matrix $\mathbf{C}$.

Algorithm 1 Fast and exAct diSTributed PCA (FAST-PCA)

Input: $\mathbf{Y}_1, \mathbf{Y}_2, \ldots \mathbf{Y}_M, \mathbf{W}, \alpha, K$
Initialize: $\forall i, \mathbf{X}_i^{(0)} \leftarrow \mathbf{X}_{\text{init}} : \mathbf{X}_{\text{init}} \in \mathbb{R}^{d\times K}, \mathbf{X}_{\text{init}}^T\mathbf{X}_{\text{init}} = \mathbf{I}$;
$\mathbf{S}_i^{(0)} \leftarrow \mathbf{h}_i(\mathbf{X}_i^{(0)})$
    for $t = 0, 1, \ldots$ **do**
        Communicate $\mathbf{X}_i^{(t)}$ from each node i to its neighbors
        Subspace estimate at node i: $\mathbf{X}_i^{(t+1)} \leftarrow \frac{1}{2}\mathbf{X}_i^{(t)} +$
$\sum_{j\in\mathcal{N}_i}\frac{w_{ij}}{2}\mathbf{X}_j^{(t)} + \alpha\mathbf{S}_i^{(t)}$
        Psuedo-gradient estimate at node i: $\mathbf{S}_i^{(t+1)} \leftarrow \frac{1}{2}\mathbf{S}_i^{(t)} +$
$\sum_{j\in\mathcal{N}_i}\frac{w_{ij}}{2}\mathbf{S}_j^{(t)} + \mathbf{h}_i(\mathbf{X}_i^{(t+1)}) - \mathbf{h}_i(\mathbf{X}_i^{(t)})$
    end for
Return: $\tilde{\mathbf{X}}_i^{(t+1)} = \left[\frac{\mathbf{x}_{i,1}^{(t+1)}}{\|\mathbf{x}_{i,1}^{(t+1)}\|}, \ldots, \frac{\mathbf{x}_{i,K}^{(t+1)}}{\|\mathbf{x}_{i,K}^{(t+1)}\|}\right], i = 1, 2, \ldots, M$

IV. CONVERGENCE ANALYSIS

This section entails detailed analysis for our proposed FAST-PCA algorithm. In the first subsection, we state the main result regarding the convergence of FAST-PCA. The following subsection provides an auxiliary result, which is followed by the detailed proof of the main result.

A. Main Result

The main result of this paper shows that FAST-PCA converges at a linear rate and exactly to the eigenvectors of the global sample covariance matrix of the data distributed

in a connected network. Specifically, we have the following theorem about the convergence result.

Theorem 1. *Suppose* $\alpha < \frac{\min_{k=1,\ldots,K}(\lambda_k - \lambda_{k+1})}{(K+5)(K+6)}\left(\frac{1-\beta}{9\lambda_1}\right)^2$, *where* λ_k, λ_{k+1} *are the* k^{th} *and* $(k+1)^{th}$ *largest eigenvalues of* $\mathbf{C}$, $\beta = \max\{|\lambda_2(\mathbf{W})|, |\lambda_M(\mathbf{W})|\}$, $\mathbf{q}_k^T \bar{\mathbf{x}}_k^{(0)} \neq 0$, *and the graph underlying the network is connected. Then the estimate* $\mathbf{x}_{i,k}^{(t)}$ *from FAST-PCA converges to the eigenvector* $\pm c_k \mathbf{q}_k$ *corresponding to the largest eigenvalue* λ_k *of* $\mathbf{C}$ *at each node* $i = 1, \ldots, M$ *at a linear rate of* $\mathcal{O}(\frac{1}{\log(1+\alpha gap)} \log \frac{1}{\epsilon})$, *where* $gap := \min_{k=1,\ldots,K} \lambda_k - \lambda_{k+1}$.

The detailed proof of Theorem 1 is given in Section IV-C. Here we provide a discussion of the implications of the theorem. From Theorem 1, we can see that if $\alpha < \frac{\min_k(\lambda_k - \lambda_{k+1})}{(K+5)(K+6)}\left(\frac{1-\beta}{9\lambda_1}\right)^2$, where λ_1 is the largest eigenvalue of $\mathbf{C}$, K is the number of eigenvectors to be estimated and β is the absolute value of the second-largest eigenvalue of the weight matrix $\mathbf{W}$, then the estimates $\mathbf{x}_{i,k}^{(t)}$ of the kth eigenvector for $k = 1, \ldots, K$ at i^{th} node, $i = 1, \ldots, M$, converge at a linear rate to a multiple of the eigenvector $\mathbf{q}_k$ of $\mathbf{C}$ i.e., $\pm c_k \mathbf{q}_k$. It is clear from the condition on α that with larger eigengap $(\lambda_k - \lambda_{k+1})$, a larger range of step size is possible, which directly affects the rate of convergence. Also, as the connectivity in the network increases, β decreases, which again increases the range of α, thus increasing the rate of convergence.

B. Auxiliary Result

In this subsection, we provide an intermediate result that will help the detailed analysis of our proposed algorithm. Let $\mathbf{C} \in \mathbb{R}^{d \times d}$ be a covariance matrix whose eigenvectors are $\mathbf{q}_l, l = 1, \ldots, d$, with corresponding eigenvalues λ_l. With an aim to estimate the first K eigenvectors of $\mathbf{C}$, we define a general update rule of the following form:

$$\mathbf{x}_{g,k}^{(t+1)} = \mathbf{x}_{g,k}^{(t)} + \alpha\left(\mathbf{C}\mathbf{x}_{g,k}^{(t)} - \frac{(\mathbf{x}_{g,k}^{(t)})^T \mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\mathbf{x}_{g,k}^{(t)}\right.$$
$$\left. - \sum_{p=1}^{k-1} \mathbf{q}_p \mathbf{q}_p^T \mathbf{C}\mathbf{x}_{g,k}^{(t)}\right) \quad (11)$$

$$= \mathbf{x}_{g,k}^{(t)} + \alpha\left(\mathbf{C}\mathbf{x}_{g,k}^{(t)} - \frac{(\mathbf{x}_{g,k}^{(t)})^T \mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\mathbf{x}_{g,k}^{(t)} - \sum_{p=1}^{k-1} \lambda_p \mathbf{q}_p \mathbf{q}_p^T \mathbf{x}_{g,k}^{(t)}\right),$$
$$(12)$$

where α is a constant step size. Looking at Krasulina's method as applying SGD to the optimization problem (7), (11) can be viewed as optimizing the Rayleigh quotient of the residual covariance matrix $\mathbf{C} - \sum_{p=1}^{k-1} \mathbf{q}_p \mathbf{q}_p^T \mathbf{C}$. Note that this is not an algorithm in the true sense of the term as it cannot be implemented because of its dependence on the true eigenvectors $\mathbf{q}_p$. The sole purpose of this update equation is to help in our ultimate goal of providing convergence guarantee for the FAST-PCA algorithm. Also, the subscript 'g' here is simply to denote that $\mathbf{x}_{g,k}^t$ is a general iterate value and any update rule of the form (12) has the same characteristics. On the other hand, $\mathbf{x}_{i,k}^t$ is the iterate value at node i, which has a different update form in the case of our proposed algorithm.

Since $\mathbf{q}_l, l = 1, \ldots, d$, are the eigenvectors of a real symmetric matrix, they form a basis for d-dimensional space and can be used for expansion of any vector $\mathbf{x} \in \mathbb{R}^d$. Let

$$\tilde{\mathbf{x}}_{g,k}^{(t)} = \frac{\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|} = \sum_{l=1}^{d} z_{k,l}^{(t)} \mathbf{q}_l, \quad (13)$$

where $z_{k,l}^{(t)}$ is the coefficient corresponding to the eigenvector $\mathbf{q}_l$ in the expansion of $\tilde{\mathbf{x}}_{g,k}^{(t)}$. The update equation (12) can be re-written as:

$$\frac{\mathbf{x}_{g,k}^{(t+1)}}{\|\mathbf{x}_{g,k}^{(t+1)}\|} = \left(\frac{\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|} + \alpha\left(\mathbf{C}\frac{\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|} - \frac{(\mathbf{x}_{g,k}^{(t)})^T \mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\frac{\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|}\right.\right.$$
$$\left.\left. - \sum_{p=1}^{k-1} \lambda_p \mathbf{q}_p \mathbf{q}_p^T \frac{\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|}\right)\right)\frac{\|\mathbf{x}_{g,k}^{(t)}\|}{\|\mathbf{x}_{g,k}^{(t+1)}\|} \quad (14)$$

$$\tilde{\mathbf{x}}_{g,k}^{(t+1)} = \left(\tilde{\mathbf{x}}_{g,k}^{(t)} + \alpha\left(\mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)} - \frac{(\mathbf{x}_{g,k}^{(t)})^T \mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\tilde{\mathbf{x}}_{g,k}^{(t)}\right.\right.$$
$$\left.\left. - \sum_{p=1}^{k-1} \lambda_p \mathbf{q}_p \mathbf{q}_p^T \tilde{\mathbf{x}}_{g,k}^{(t)}\right)\right)a_k^{(t)}, \quad (15)$$

where $a_k^{(t)} = \frac{\|\mathbf{x}_{g,k}^{(t)}\|}{\|\mathbf{x}_{g,k}^{(t+1)}\|}$. Multiplying both sides of (15) by $\mathbf{q}_l^T$ and using the fact that $\mathbf{q}_l^T \mathbf{q}_{l'} = 0$ for $l \neq l'$, we get

$$z_{k,l}^{(t+1)} = a_k^{(t)}\left(z_{k,l}^{(t)} + \alpha\left(\mathbf{q}_l^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)} - \mathbf{q}_l^T\left(\sum_{p=1}^{k-1} \lambda_p \mathbf{q}_p \mathbf{q}_p^T \tilde{\mathbf{x}}_{g,k}^{(t)}\right)\right.\right.$$
$$\left.\left. - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)} z_{k,l}^{(t)}\right)\right).$$

This gives

$$z_{k,l}^{(t+1)} = a_k^{(t)}\left(z_{k,l}^{(t)} - \alpha(\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)} z_{k,l}^{(t)}\right),$$
$$\text{for} \quad l = 1, \ldots k-1, \quad (16)$$

and $z_{k,l}^{(t+1)} = a_k^{(t)}\left(z_{k,l}^{(t)} + \alpha(\lambda_l - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)})z_{k,l}^{(t)}\right),$
$$\text{for} \quad l = k, \ldots d. \quad (17)$$

In the following theorem, we show that $\mathbf{x}_{g,k}^{(t)}$ converges to a multiple of the true eigenvector $\mathbf{q}_k$ by proving convergence of the coefficients $z_{k,l}^{(t)}$ for $l = 1, \ldots, d$.

Theorem 2. *Suppose* $\mathbf{C}$ *has* K *distinct eigenvalues, i.e.,* $\lambda_1 > \lambda_2 > \cdots > \lambda_K > \lambda_{K+1} \geq \cdots \geq \lambda_d \geq 0$ *and* $\alpha < \frac{1}{\lambda_1}$, $\mathbf{q}_k^T \mathbf{x}_{g,k}^{(0)} \neq 0$, *and* $\|\mathbf{x}_{g,k}^{(0)}\| = 1$ *for all* $k = 1, \ldots, K$. *Then the update equation for* $\mathbf{x}_{g,k}^{(t)}$ *given by* (12) *converges at a linear rate to a multiple of the eigenvector* $\pm\mathbf{q}_k$ *corresponding to the* k^{th} *largest eigenvalue* λ_k *of the covariance matrix* $\mathbf{C}$ *for* $k = 1, \ldots, K$.

Proof. We prove the linear convergence of $\mathbf{x}_{g,k}^{(t)}$ to a multiple of $\mathbf{q}_k$ by proving that $\tilde{\mathbf{x}}_{g,k}^{(t)}$ converges to $\mathbf{q}_k$ at a linear rate. The convergence of $\tilde{\mathbf{x}}_{g,k}^{(t)}$ to $\mathbf{q}_k$ requires convergence of the lower-order coefficients $z_{k,1}^{(t)}, \ldots, z_{k,k-1}^{(t)}$ and the higher-order coefficients $z_{k,k+1}^{(t)}, \ldots, z_{k,d}^{(t)}$ to 0 and convergence of $z_{k,k}^{(t)}$ to ± 1. To this end, Lemma 5 proves linear convergence of the lower-order coefficients $z_{k,1}^{(t)}, \ldots, z_{k,k-1}^{(t)}$ to 0

by showing $\sum_{l=1}^{k-1}(z_{k,l}^{(t)})^2 \le a_1 \gamma_k^t$ for some constants $a_1 > 0, \gamma_k = \frac{1}{1+\alpha\lambda_k} < 1$. Furthermore, Lemma 6 shows that $\sum_{l=k+1}^{d}(z_{k,l}^{(t)})^2 \le a_2\delta_k^t$, where $a_2 > 0$ and $\delta_k = \frac{1+\alpha\lambda_{k+1}}{1+\alpha\lambda_k} < 1$, thereby proving linear convergence of the higher-order coefficients to 0. The formal statements and proofs of Lemma 5 and Lemma 6 are given in Appendix A and Appendix B, respectively. Finally, since $\|\tilde{\mathbf{x}}_{g,k}^{(t)}\|^2 = 1$, we have

$$\sum_{l=1}^{d}(z_{k,l}^{(t)})^2 = 1$$

$$\text{or,}\quad 1 - (z_{k,k}^{(t)})^2 = \sum_{l=1}^{k-1}(z_{k,l}^{(t)})^2 + \sum_{l=k+1}^{d}(z_{k,l}^{(t)})^2$$
$$\le a_1\gamma_k^t + a_2\delta_k^t$$
$$< a_3\delta_k^t,$$

where $a_3 = \max\{a_1, a_2\}$ and $\delta_k = \max\{\gamma_k, \delta_k\}$. This shows $(z_{k,k}^{(t)})^2$ converges to 1 and $(z_{l,k}^{(t)})^2, l \ne k$, converges to 0 at a linear rate of $\mathcal{O}(\delta_k^t)$ where $\delta_k = \frac{1+\alpha\lambda_{k+1}}{1+\alpha\lambda_k}$. Thus, $\tilde{\mathbf{x}}_{g,k}^{(t)} \to \pm\mathbf{q}_k$ and $(\tilde{\mathbf{x}}_{g,k}^{(t)})^T\mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)} \to \lambda_k$. We also know from (12) that

$$\mathbf{x}_{g,k}^{(t+1)} = \mathbf{x}_{g,k}^{(t)}+$$
$$\alpha\Big((\mathbf{C} - \sum_{p=1}^{k-1}\lambda_p\mathbf{q}_p\mathbf{q}_p^T)\mathbf{x}_{g,k}^{(t)} - \frac{(\mathbf{x}_{g,k}^{(t)})^T\mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\mathbf{x}_{g,k}^{(t)}\Big)$$

$$\text{i.e.,}\quad \|\mathbf{x}_{g,k}^{(t+1)}\|^2 = \|\mathbf{x}_{g,k}^{(t)}\|^2 + \alpha^2\|((\mathbf{C} - \sum_{p=1}^{k-1}\lambda_p\mathbf{q}_p\mathbf{q}_p^T)\mathbf{x}_{g,k}^{(t)}$$
$$- \frac{(\mathbf{x}_{g,k}^{(t)})^T\mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\mathbf{x}_{g,k}^{(t)})\|^2 -$$
$$2\alpha(\mathbf{x}_{g,k}^{(t)})^T\big((\mathbf{C} - \sum_{p=1}^{k-1}\lambda_p\mathbf{q}_p\mathbf{q}_p^T)\mathbf{x}_{g,k}^{(t)} - \frac{(\mathbf{x}_{g,k}^{(t)})^T\mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\mathbf{x}_{g,k}^{(t)}\big)$$
$$= \|\mathbf{x}_{g,k}^{(t)}\|^2 + \alpha^2\|((\mathbf{C} - \sum_{p=1}^{k-1}\lambda_p\mathbf{q}_p\mathbf{q}_p^T)\mathbf{x}_{g,k}^{(t)} -$$
$$\frac{(\mathbf{x}_{g,k}^{(t)})^T\mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\mathbf{x}_{g,k}^{(t)})\|^2 + 2\alpha\sum_{p=1}^{k-1}\lambda_p(\mathbf{x}_{g,k}^{(t)})^T\mathbf{q}_p\mathbf{q}_p^T\mathbf{x}_{g,k}^{(t)}$$
$$= \|\mathbf{x}_{g,k}^{(t)}\|^2 + \alpha^2\|(\mathbf{C} - \sum_{p=1}^{k-1}\lambda_p\mathbf{q}_p\mathbf{q}_p^T)\tilde{\mathbf{x}}_{g,k}^{(t)}\|\mathbf{x}_{g,k}^{(t)}\|$$
$$- \frac{(\mathbf{x}_{g,k}^{(t)})^T\mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\tilde{\mathbf{x}}_{g,k}^{(t)}\|\mathbf{x}_{g,k}^{(t)}\|\|^2$$
$$+ 2\alpha\|\mathbf{x}_{g,k}^{(t)}\|^2\sum_{p=1}^{k-1}\lambda_p(\tilde{\mathbf{x}}_{g,k}^{(t)})^T\mathbf{q}_p\mathbf{q}_p^T\tilde{\mathbf{x}}_{g,k}^{(t)}. \quad (18)$$

As $\tilde{\mathbf{x}}_{g,k}^{(t)} \to \pm\mathbf{q}_k$ and $\frac{(\mathbf{x}_{g,k}^{(t)})^T\mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2} \to \lambda_k$, we have

$$(\mathbf{C} - \sum_{p=1}^{k-1}\lambda_p\mathbf{q}_p\mathbf{q}_p^T)\tilde{\mathbf{x}}_{g,k}^{(t)}\|\mathbf{x}_{g,k}^{(t)}\| \to \pm\mathbf{C}\mathbf{q}_k\|\mathbf{x}_{g,k}^{(t)}\| = \pm\lambda_k\mathbf{q}_k\|\mathbf{x}_{g,k}^{(t)}\|$$

and

$$\sum_{p=1}^{k-1}\lambda_p(\tilde{\mathbf{x}}_{g,k}^{(t)})^T\mathbf{q}_p\mathbf{q}_p^T\tilde{\mathbf{x}}_{g,k}^{(t)} \to 0.$$

Thus from (18), we get

$$\|\mathbf{x}_{g,k}^{(t+1)}\|^2 - \|\mathbf{x}_{g,k}^{(t)}\|^2 \to 0,$$

which implies $\|\mathbf{x}_{g,k}^{(t)}\|$ converges to some constant $c_k > 0$ which further implies $\mathbf{x}_{g,k}^{(t)} \to \pm c_k\mathbf{q}_k$. $\blacksquare$

C. Analysis of FAST-PCA

In this subsection, we provide a detailed analysis proving that the FAST-PCA algorithm converges at a linear rate to the true eigenvectors $\mathbf{q}_k, k = 1, \dots, K$, of the global covariance matrix $\mathbf{C}$. Specifically, let $\mathbf{X}_i^{(t)} = \left[\mathbf{x}_{i,1}^{(t)}, \dots, \mathbf{x}_{i,K}^{(t)}\right] \in \mathbb{R}^{d\times K}$ be the estimate of the K eigenvectors at node i, then we show that square of the sine of the angle between $\mathbf{x}_{i,k}^{(t)}, \forall i = 1, \dots, M$, and $\mathbf{q}_k$ for $k = 1, \dots, K$ converges to 0 at the rate of $\mathcal{O}(\rho^t)$ for some $\rho \in (0,1)$.

We know from Theorem 2 in the previous section that for estimation of the kth eigenvector, any general iterate of the form

$$\mathbf{x}_{g,k}^{(t+1)} = \mathbf{x}_{g,k}^{(t)}+\alpha\Big(\mathbf{C}\mathbf{x}_{g,k}^{(t)} - \frac{(\mathbf{x}_{g,k}^{(t)})^T\mathbf{C}\mathbf{x}_{g,k}^{(t)}}{\|\mathbf{x}_{g,k}^{(t)}\|^2}\mathbf{x}_{g,k}^{(t)} - \sum_{p=1}^{k-1}\mathbf{q}_p\mathbf{q}_p^T\mathbf{C}\mathbf{x}_{g,k}^{(t)}\Big) \quad (19)$$

converges at a linear rate to a scalar multiple of the eigenvector $\mathbf{q}_k$ of $\mathbf{C}$ if the top K eigenvalues of $\mathbf{C}$ are distinct, i.e., $\lambda_1 > \lambda_2 > \cdots > \lambda_K > \lambda_{K+1} \ge \cdots \ge \lambda_d \ge 0$ as well as if $\alpha < \frac{1}{\lambda_1}$ and $\mathbf{q}_k^T\mathbf{x}_{g,k}^{(0)} \ne 0$. Specifically, $\mathbf{x}_{g,k}^{(t)}$ converges to either $c_k\mathbf{q}_k$ or $-c_k\mathbf{q}_k$ at a linear rate in this case. Mathematically,

$$\|\mathbf{x}_{g,k}^{(t+1)} - \mathbf{x}_k^*\| \le \delta_k\|\mathbf{x}_{g,k}^{(t)} - \mathbf{x}_k^*\|, \quad \text{for}$$
$$0 < \delta_k = \frac{1+\alpha\lambda_{k+1}}{1+\alpha\lambda_k} < 1 \quad \text{and} \quad \mathbf{x}_k^* = c_k\mathbf{q}_k \quad \text{or} \quad -c_k\mathbf{q}_k. \quad (20)$$

Now, if $\mathbf{W} = [w_{ij}]$ is the weight matrix underlying the graph representing the network, then the iterates of FAST-PCA for the estimation of the kth eigenvector are given as follows:

$$\mathbf{x}_{i,k}^{(t+1)} = \frac{1}{2}\mathbf{x}_{i,k}^{(t)} + \sum_{j\in\mathcal{N}_i}\frac{w_{ij}}{2}\mathbf{x}_{j,k}^{(t)} + \alpha\mathbf{s}_{i,k}^{(t)} \quad (21)$$

$$\mathbf{s}_{i,k}^{(t+1)} = \frac{1}{2}\mathbf{s}_{i,k}^{(t)} + \sum_{j\in\mathcal{N}_i}\frac{w_{ij}}{2}\mathbf{s}_{j,k}^{(t)} + \mathbf{h}_i(\mathbf{x}_{i,k}^{(t+1)}) - \mathbf{h}_i(\mathbf{x}_{i,k}^{(t)}), \quad (22)$$

where $\mathbf{x}_{i,k}^{(t)}$ is the estimate of the kth eigenvector, $\mathbf{h}_i(\mathbf{x}_{i,k}^{(t)})$ is the pseudo-gradient given as $\mathbf{h}_i(\mathbf{x}_{i,k}^{(t)}) = \mathbf{C}_i\mathbf{x}_{i,k}^{(t)} - \frac{(\mathbf{x}_{i,k}^{(t)})^T\mathbf{C}_i\mathbf{x}_{i,k}^{(t)}}{\|\mathbf{x}_{i,k}^{(t)}\|^2}\mathbf{x}_{i,k}^{(t)} - \sum_{p=1}^{k-1}\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T\mathbf{C}_i}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\mathbf{x}_{i,k}^{(t)}$ and $\mathbf{s}_{i,k}^{(t)}$ is the estimate of the average pseudo-gradients. Let us define the

following stacked versions of the quantities $\mathbf{x}_{i,k}^{(t)}, \mathbf{s}_{i,k}^{(t)}$, and $\mathbf{h}_i(\mathbf{x}_{i,k}^{(t)})$ for $i = 1, \ldots, M$ as

$$
\mathbf{x}_k^{(t)} = \begin{bmatrix} \mathbf{x}_{1,k}^{(t)} \\ \mathbf{x}_{2,k}^{(t)} \\ \vdots \\ \mathbf{x}_{M,k}^{(t)} \end{bmatrix}, \quad \mathbf{h}(\mathbf{x}_k^{(t)}) = \begin{bmatrix} \mathbf{h}_1(\mathbf{x}_{1,k}^{(t)}) \\ \mathbf{h}_2(\mathbf{x}_{2,k}^{(t)}) \\ \vdots \\ \mathbf{h}_M(\mathbf{x}_{M,k}^{(t)}) \end{bmatrix}, \quad \mathbf{s}_k^{(t)} = \begin{bmatrix} \mathbf{s}_{1,k}^{(t)} \\ \mathbf{s}_{2,k}^{(t)} \\ \vdots \\ \mathbf{s}_{M,k}^{(t)} \end{bmatrix}.
$$

Let $\bar{\mathbf{x}}_k^{(t)}$ and $\bar{\mathbf{s}}_k^{(t)}$ denote the average of $\{\mathbf{x}_{i,k}^{(t)}\}_{i=1}^M$ and $\{\mathbf{s}_{i,k}^{(t)}\}_{i=1}^M$, respectively. Taking the average of (21) and (22) over all nodes $i = 1, \ldots, M$, we get

$$
\frac{1}{M} \sum_{i=1}^M \mathbf{x}_{i,k}^{(t+1)} = \bar{\mathbf{x}}_k^{(t+1)} = \bar{\mathbf{x}}_k^{(t)} + \alpha \bar{\mathbf{s}}_k^{(t)} \tag{23}
$$

$$
\frac{1}{M} \sum_{i=1}^M \mathbf{s}_{i,k}^{(t+1)} = \bar{\mathbf{s}}_k^{(t+1)}
$$
$$
= \bar{\mathbf{s}}_k^{(t)} + \mathbf{g}(\mathbf{x}_k^{(t+1)}) - \mathbf{g}(\mathbf{x}_k^{(t)}) = \mathbf{g}(\mathbf{x}_k^{(t+1)}), \tag{24}
$$

where $\mathbf{g}(\mathbf{x}_k^{(t)}) = \frac{1}{M} \sum_{i=1}^M \mathbf{h}_i(\mathbf{x}_{i,k}^{(t)}) \in \mathbb{R}^d$. Additionally, we also define the following stacked versions (denoted by subscript 's') such that all these are in $\mathbb{R}^{Md}$:

$$
\bar{\mathbf{x}}_{s,k}^{(t)} = \begin{bmatrix} \bar{\mathbf{x}}_k^{(t)} \\ \bar{\mathbf{x}}_k^{(t)} \\ \vdots \\ \bar{\mathbf{x}}_k^{(t)} \end{bmatrix}, \quad \bar{\mathbf{s}}_{s,k}^{(t)} = \begin{bmatrix} \bar{\mathbf{s}}_k^{(t)} \\ \bar{\mathbf{s}}_k^{(t)} \\ \vdots \\ \bar{\mathbf{s}}_k^{(t)} \end{bmatrix}, \quad \mathbf{g}_s(\mathbf{x}_k^{(t)}) = \begin{bmatrix} \mathbf{g}(\mathbf{x}_k^{(t)}) \\ \mathbf{g}(\mathbf{x}_k^{(t)}) \\ \vdots \\ \mathbf{g}(\mathbf{x}_k^{(t)}) \end{bmatrix}.
$$

Using these definitions, (21) and (22) can be re-written as

$$
\mathbf{x}_k^{(t+1)} = \frac{1}{2}((\mathbf{I}_M + \mathbf{W}) \otimes \mathbf{I}_d)\mathbf{x}_k^{(t)} + \alpha \mathbf{s}_k^{(t)}, \tag{25}
$$

$$
\mathbf{s}_k^{(t+1)} = \frac{1}{2}((\mathbf{I}_M + \mathbf{W}) \otimes \mathbf{I}_d)\mathbf{s}_k^{(t)} + \mathbf{h}(\mathbf{x}_k^{(t+1)}) - \mathbf{h}(\mathbf{x}_k^{(t)}). \tag{26}
$$

Also,

$$
\bar{\mathbf{s}}_{s,k}^{(t+1)} = \bar{\mathbf{s}}_{s,k}^{(t)} + \mathbf{g}_s(\mathbf{x}_k^{(t+1)}) - \mathbf{g}_s(\mathbf{x}_k^{(t)}) = \mathbf{g}_s(\mathbf{x}_k^{(t+1)}) \tag{27}
$$

$$
\bar{\mathbf{x}}_{s,k}^{(t+1)} = \bar{\mathbf{x}}_{s,k}^{(t)} + \alpha \bar{\mathbf{s}}_{s,k}^{(t)} = \bar{\mathbf{x}}_{s,k}^{(t)} + \alpha \mathbf{g}_s(\mathbf{x}_k^{(t)}) \tag{28}
$$

$$
\mathbf{g}(\bar{\mathbf{x}}_{s,k}^{(t)}) = \frac{1}{M} \sum_{i=1}^M \left(\mathbf{C}_i \bar{\mathbf{x}}_k^{(t)} - \frac{(\bar{\mathbf{x}}_k^{(t)})^T \mathbf{C}_i \bar{\mathbf{x}}_k^{(t)}}{\|\bar{\mathbf{x}}_k^{(t)}\|^2} \bar{\mathbf{x}}_k^{(t)} - \sum_{p=1}^{k-1} \frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T \mathbf{C}_i}{\|\mathbf{x}_{i,p}^{(t)}\|^2} \bar{\mathbf{x}}_k^{(t)} \right)
$$
$$
= \frac{1}{M} \left(\mathbf{C}\bar{\mathbf{x}}_k^{(t)} - \frac{(\bar{\mathbf{x}}_k^{(t)})^T \mathbf{C}\bar{\mathbf{x}}_k^{(t)}}{\|\bar{\mathbf{x}}_k^{(t)}\|^2} \bar{\mathbf{x}}_k^{(t)} - \sum_{i=1}^M \sum_{p=1}^{k-1} \frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T \mathbf{C}_i}{\|\mathbf{x}_{i,p}^{(t)}\|^2} \bar{\mathbf{x}}_k^{(t)} \right). \tag{29}
$$

Now, we first show that $\mathbf{x}_{i,1}^{(t)}$ converges to a multiple of $\mathbf{q}_1$ at a linear rate and then proceed with the proof for $k = 2, \ldots, K$ through induction.

Case I for Induction – $k = 1$: The iterates of FAST-PCA for estimation of the dominant eigenvector are

$$
\mathbf{x}_{i,1}^{(t+1)} = \frac{1}{2}\mathbf{x}_{i,1}^{(t)} + \sum_{j \in \mathcal{N}_i} \frac{w_{ij}}{2}\mathbf{x}_{j,1}^{(t)} + \alpha \mathbf{s}_{i,1}^{(t)} \tag{30}
$$

$$
\mathbf{s}_{i,1}^{(t+1)} = \frac{1}{2}\mathbf{s}_{i,1}^{(t)} + \sum_{j \in \mathcal{N}_i} \frac{w_{ij}}{2}\mathbf{s}_{j,1}^{(t)} + \mathbf{h}_i(\mathbf{x}_{i,1}^{(t+1)}) - \mathbf{h}_i(\mathbf{x}_{i,1}^{(t)}), \tag{31}
$$

where $\mathbf{h}_i(\mathbf{x}_{i,1}^{(t)}) = \mathbf{C}_i \mathbf{x}_{i,1}^{(t)} - \frac{(\mathbf{x}_{i,1}^{(t)})^T \mathbf{C}_i \mathbf{x}_{i,1}^{(t)}}{\|\mathbf{x}_{i,1}^{(t)}\|^2}\mathbf{x}_{i,1}^{(t)}$.

Lemma 1. *The function* $\mathbf{h}_i : \mathbb{R}^d \to \mathbb{R}^d$ *with* $\mathbf{h}_i(\mathbf{v}) = \mathbf{C}_i\mathbf{v} - \frac{(\mathbf{v})^T \mathbf{C}_i \mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{v}$ *is Lipschitz continuous with Lipschitz constant* $L_1 = 6\lambda_1$.

The proof of this lemma is deferred to Appendix C. For Lipschitz continuous functions $\mathbf{h}(\mathbf{x}_1)$ and $\mathbf{g}(\mathbf{x}_1)$ defined above, the following lemma holds true, the proof of which is deferred to Appendix D.

Lemma 2. *The following inequalities hold for* $L_1 = 6\lambda_1$:
1) $\|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\|_2 \le L_1\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|_2$
2) $\|\mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\mathbf{x}_1^{(t-1)})\|_2 \le \frac{L_1}{\sqrt{M}}\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|_2$
3) $\|\mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t)})\|_2 \le \frac{L_1}{\sqrt{M}}\|\mathbf{x}_1^{(t)} - \bar{\mathbf{x}}_{s,1}^{(t)}\|_2$

These inequalities aid the proof of our main theorem presented next that shows the convergence of the iterate $\mathbf{x}_{i,1}^{(t)}$ at node i to $\mathbf{x}_1^* = \pm c_1 \mathbf{q}_1$, where c_1 is a constant.

Proof of Theorem 1 for $k = 1$:
For proving the convergence of $\mathbf{x}_{i,1}^{(t)}$ to $\mathbf{x}_1^* = \pm c_1\mathbf{q}_1, \forall i = 1, \ldots, M$, we prove that the distance of average $\bar{\mathbf{x}}_1^{(t)}$ from $\mathbf{x}_1^*$, the consensus error as well as the distance of $\mathbf{s}_{i,1}^{(t)}$ from the average pseudo-gradient $\mathbf{g}(\mathbf{x}_1^{(t)})$ decay to zero at a linear rate. From (26), we have

$$
\mathbf{s}_1^{(t)} - \mathbf{g}_s(\mathbf{x}_1^{(t)}) = \frac{1}{2}((\mathbf{I}_M + \mathbf{W}) \otimes \mathbf{I}_d)\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}) +
$$
$$
\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)}) - (\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})).
$$

From the definitions of $\bar{\mathbf{s}}_1^{(t)}$, $\mathbf{g}(\mathbf{x}_1^{(t-1)})$ and $\mathbf{g}_s(\mathbf{x}_1^{(t-1)})$, it is obvious that $(\frac{1}{M}\mathbf{1}\mathbf{1}^T \otimes \mathbf{I}_d)\mathbf{s}_1^{(t-1)} = \mathbf{1}\bar{\mathbf{s}}_1^{(t-1)} = \bar{\mathbf{s}}_{s,1}^{(t-1)} = \mathbf{g}_s(\mathbf{x}_1^{(t-1)})$. Thus,

$$
\|\mathbf{s}_1^{(t)} - \mathbf{g}_s(\mathbf{x}_1^{(t)})\|
$$
$$
= \|\frac{1}{2}((\mathbf{I}_M + \mathbf{W}) \otimes \mathbf{I}_d)\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}) + \mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})
$$
$$
- (\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))\|
$$
$$
= \|\frac{1}{2}((\mathbf{I}_M + \mathbf{W}) \otimes \mathbf{I}_d)\mathbf{s}_1^{(t-1)} - (\frac{1}{M}\mathbf{1}\mathbf{1}^T \otimes \mathbf{I}_d)\mathbf{s}_1^{(t-1)} + \mathbf{g}_s(\mathbf{x}_1^{(t-1)})
$$
$$
- \mathbf{g}_s(\mathbf{x}_1^{(t-1)}) + \mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)}) - (\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))\|
$$
$$
= \|(\frac{1}{2}((\mathbf{I}_M + \mathbf{W}) \otimes \mathbf{I}_d) - \frac{1}{M}\mathbf{1}\mathbf{1}^T \otimes \mathbf{I}_d)(\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})) +
$$
$$
\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)}) - (\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))\|
$$
$$
\le \|((\frac{1}{2}(\mathbf{I}_M + \mathbf{W}) - \frac{1}{M}\mathbf{1}\mathbf{1}^T) \otimes \mathbf{I}_d)(\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))\| +
$$
$$
\|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)}) - (\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))\|. \tag{32}
$$

Next, we simplify the second term of the above inequality (32) as follows:

$$\|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)}) - (\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))\|^2$$
$$= \|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\|^2 + \|\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\|^2$$
$$- 2\langle \mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)}), \mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\rangle$$
$$= \|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\|^2 + \|\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\|^2$$
$$- 2\sum_{i=1}^{M}\langle \mathbf{h}_i(\mathbf{x}_{i,1}^{(t)}) - \mathbf{h}_i(\mathbf{x}_{i,1}^{(t-1)}), \mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\mathbf{x}_1^{(t-1)})\rangle$$
$$= \|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\|^2 + M\|\mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\mathbf{x}_1^{(t-1)})\|^2$$
$$- 2M\langle \mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\mathbf{x}_1^{(t-1)}), \mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\mathbf{x}_1^{(t-1)})\rangle$$
$$= \|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\|^2 - M\|\mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\mathbf{x}_1^{(t-1)})\|^2$$
$$\leq \|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\|^2.$$

Thus,

$$\|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)}) - (\mathbf{g}_s(\mathbf{x}_1^{(t)}) - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))\|$$
$$\leq \|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\| \leq L_1\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|. \quad (33)$$

From (32) and (33), we have the following

$$\|\mathbf{s}_1^{(t)} - \mathbf{g}_s(\mathbf{x}_1^{(t)})\|$$
$$\leq \|((\frac{1}{2}(\mathbf{I}_M + \mathbf{W}) - \frac{1}{M}\mathbf{1}\mathbf{1}^T)\otimes\mathbf{I}_d)(\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))\|$$
$$+ L_1\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|$$
$$\leq \frac{1+\beta}{2}\|\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\| + L_1\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|, \quad (34)$$

where β is absolute value of the second largest eigenvalue of the weight matrix $\mathbf{W}$, i.e., $\beta = \max\{|\lambda_2(\mathbf{W})|, |\lambda_M(\mathbf{W})|\}$. As pointed out before, network connectivity ensures that $\beta < 1$. Next, from (25) and (28), we have

$$\mathbf{x}_1^{(t)} - \bar{\mathbf{x}}_{s,1}^{(t)} = \frac{1}{2}((\mathbf{I}_M + \mathbf{W})\otimes\mathbf{I}_d)\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)} +$$
$$\alpha(\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}))$$
$$= ((\frac{1}{2}(\mathbf{I}_M + \mathbf{W}) - \frac{1}{M}\mathbf{1}\mathbf{1}^T)\otimes\mathbf{I}_d)(\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)})$$
$$+ \alpha(\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})).$$

Thus,

$$\|\mathbf{x}_1^{(t)} - \bar{\mathbf{x}}_{s,1}^{(t)}\| \leq \frac{1+\beta}{2}\|\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)}\| + \alpha\|\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\|. \quad (35)$$

Next, we bound $\|\bar{\mathbf{x}}_1^{(t)} - \mathbf{x}_1^*\|$. We know from (23)

$$\bar{\mathbf{x}}_1^{(t)} = \bar{\mathbf{x}}_1^{(t-1)} + \alpha\bar{\mathbf{s}}_1^{(t-1)} = \bar{\mathbf{x}}_1^{(t-1)} + \alpha\mathbf{g}(\mathbf{x}_1^{(t-1)})$$
$$= \bar{\mathbf{x}}_1^{(t-1)} + \alpha\mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t-1)}) + \alpha(\mathbf{g}(\mathbf{x}_1^{(t-1)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t-1)}))$$
$$= \bar{\mathbf{x}}_1^{(t-1)} + \frac{\alpha}{M}(\mathbf{C}\bar{\mathbf{x}}_1^{(t-1)} - \frac{(\bar{\mathbf{x}}_1^{(t-1)})^T\mathbf{C}\bar{\mathbf{x}}_1^{(t-1)}}{\|\bar{\mathbf{x}}_1^{(t-1)}\|^2}\bar{\mathbf{x}}_1^{(t-1)})$$
$$+ \alpha(\mathbf{g}(\mathbf{x}_1^{(t-1)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t-1)})).$$

Using (19) and (20), we know that an iterate of the form

$$\bar{\mathbf{x}}_1^{(t)} = \bar{\mathbf{x}}_1^{(t-1)} + \alpha(\mathbf{C}\bar{\mathbf{x}}_1^{(t-1)} - \frac{(\bar{\mathbf{x}}_1^{(t-1)})^T\mathbf{C}\bar{\mathbf{x}}_1^{(t-1)}}{\|\bar{\mathbf{x}}_1^{(t-1)}\|^2}\bar{\mathbf{x}}_1^{(t-1)})$$

converges linearly as

$$\|\bar{\mathbf{x}}_1^{(t)} - \mathbf{x}_1^*\| \leq \delta_1\|\bar{\mathbf{x}}_1^{(t-1)} - \mathbf{x}_1^*\|,$$

where $\mathbf{x}_1^* = \pm c_1\mathbf{q}_1$ and $\delta_1 = \frac{1+\alpha\lambda_2}{1+\alpha\lambda_1}$. Thus,

$$\|\bar{\mathbf{x}}_1^{(t)} - \mathbf{x}_1^*\| \leq \delta_1\|\bar{\mathbf{x}}_1^{(t-1)} - \mathbf{x}_1^*\| + \alpha\|\mathbf{g}(\mathbf{x}_1^{(t-1)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t-1)})\|$$
$$\leq \delta_1\|\bar{\mathbf{x}}_1^{(t-1)} - \mathbf{x}_1^*\| + \alpha\frac{L_1}{\sqrt{M}}\|\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)}\|. \quad (36)$$

Now we will bound $\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|$. We know from (29)

$$\mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t)}) = \frac{1}{M}(\mathbf{C}\bar{\mathbf{x}}_1^{(t)} - \frac{(\bar{\mathbf{x}}_1^{(t)})^T\mathbf{C}\bar{\mathbf{x}}_1^{(t)}}{\|\bar{\mathbf{x}}_1^{(t)}\|^2}\bar{\mathbf{x}}_1^{(t)}).$$

Thus $\mathbf{g}(\mathbf{x}_{s,1}^*) = \frac{1}{M}(\mathbf{C}\mathbf{x}_1^* - \frac{(\mathbf{x}_1^*)^T\mathbf{C}\mathbf{x}_1^*}{\|\mathbf{x}_1^*\|^2}\mathbf{x}_1^*) = 0$, where $\mathbf{x}_{s,1}^* = \left[(\mathbf{x}_1^*)^T, \dots, (\mathbf{x}_1^*)^T\right]^T$. Hence,

$$\|\mathbf{g}_s(\bar{\mathbf{x}}_{s,1}^{(t-1)})\| = \sqrt{M}\|\mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t-1)})\| = \sqrt{M}\|\mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t-1)}) - \mathbf{g}(\mathbf{x}_{s,1}^*)\|$$
$$\leq L_1\sqrt{M}\|\bar{\mathbf{x}}_1^{(t-1)} - \mathbf{x}_1^*\|.$$

Using the above inequality and Lemma 2, we get

$$\|\mathbf{s}_1^{(t-1)}\| = \|\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)}) + \mathbf{g}_s(\mathbf{x}_1^{(t-1)}) - \mathbf{g}_s(\bar{\mathbf{x}}_{s,1}^{(t-1)})$$
$$+ \mathbf{g}_s(\bar{\mathbf{x}}_{s,1}^{(t-1)})\| \quad (37)$$
$$\leq \|\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\| + \|\mathbf{g}_s(\mathbf{x}_1^{(t-1)}) - \mathbf{g}_s(\bar{\mathbf{x}}_{s,1}^{(t-1)})\|$$
$$+ \|\mathbf{g}_s(\bar{\mathbf{x}}_{s,1}^{(t-1)})\| \quad (38)$$
$$\leq \|\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\| + L_1\|\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)}\|$$
$$+ L_1\sqrt{M}\|\bar{\mathbf{x}}_1^{(t-1)} - \mathbf{x}_1^*\|. \quad (39)$$

Thus,

$$\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\| = \|\frac{1}{2}((\mathbf{I}_M + \mathbf{W})\otimes\mathbf{I}_d)\mathbf{x}_1^{(t-1)} - \mathbf{x}_1^{(t-1)} + \alpha\mathbf{s}_1^{(t-1)}\|$$
$$= \|(\frac{1}{2}((\mathbf{I}_M + \mathbf{W})\otimes\mathbf{I}_d) - \mathbf{I}_{Md})(\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)})$$
$$+ \alpha\mathbf{s}_1^{(t-1)}\|$$
$$\leq 2\|\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)}\| + \alpha\|\mathbf{s}_1^{(t-1)}\|$$
$$\leq \alpha\|\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\| + (2 + \alpha L_1)\|\mathbf{x}_1^{(t-1)}$$
$$- \bar{\mathbf{x}}_{s,1}^{(t-1)}\| + \alpha L_1\sqrt{M}\|\bar{\mathbf{x}}_1^{(t-1)} - \mathbf{x}_1^*\| \quad \text{using (39)},$$

where the second last inequality is because $\|\frac{1}{2}((\mathbf{I}_M + \mathbf{W})\otimes \mathbf{I}_d) - \mathbf{I}_{Md}\| \leq \|\frac{1}{2}(\mathbf{I}_M + \mathbf{W})\| + \|\mathbf{I}_{Md}\| = 2$. Using the above inequality in (34), we get

$$\|\mathbf{s}_1^{(t)} - \mathbf{g}_s(\mathbf{x}_1^{(t)})\| \leq (\frac{1+\beta}{2} + \alpha L_1)\|\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\|$$
$$+ L_1(2 + \alpha L_1)\|\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)}\|$$
$$+ \alpha L_1^2\sqrt{M}\|\bar{\mathbf{x}}_1^{(t-1)} - \mathbf{x}_1^*\|. \quad (40)$$

Writing a system of equations from (35), (36) and (40), we have the following:

$$
\begin{bmatrix}
\|\mathbf{s}_1^{(t)} - \mathbf{g}_s(\mathbf{x}_1^{(t)})\| \\
\|\mathbf{x}_1^{(t)} - \bar{\mathbf{x}}_{s,1}^{(t)}\| \\
\sqrt{M}\|\bar{\mathbf{x}}_1^{(t)} - \mathbf{x}_1^*\|
\end{bmatrix}
\leq
\begin{bmatrix}
(\frac{1+\beta}{2} + \alpha L_1) & L_1(2+\alpha L_1) & \alpha L_1^2 \\
\alpha & \frac{1+\beta}{2} & 0 \\
0 & \alpha L_1 & \delta_1
\end{bmatrix}
$$
$$
\times
\begin{bmatrix}
\|\mathbf{s}_1^{(t-1)} - \mathbf{g}_s(\mathbf{x}_1^{(t-1)})\| \\
\|\mathbf{x}_1^{(t-1)} - \bar{\mathbf{x}}_{s,1}^{(t-1)}\| \\
\sqrt{M}\|\bar{\mathbf{x}}_1^{(t-1)} - \mathbf{x}_1^*\|
\end{bmatrix}, \quad (41)
$$

where $\leq$ implies element-wise inequalities. Let us define
$$
\mathbf{P}(\alpha) =
\begin{bmatrix}
(\frac{1+\beta}{2} + \alpha L_1) & L_1(2+\alpha L_1) & \alpha L_1^2 \\
\alpha & \frac{1+\beta}{2} & 0 \\
0 & \alpha L_1 & \delta_1
\end{bmatrix}. \text{ Since } \mathbf{P}(\alpha)
$$
has non-negative entries and $\mathbf{P}^2(\alpha)$ has all positive entries, each entry of $\mathbf{P}^t(\alpha)$ will be $\mathcal{O}(\rho(\mathbf{P}(\alpha))^t)$, where $\rho(\mathbf{P}(\alpha))$ is the spectral radius of $\mathbf{P}(\alpha)$. If we choose α such that $\rho(\mathbf{P}(\alpha))$ is < 1, then that implies $\|\mathbf{s}_1^{(t)} - \mathbf{g}_{v,1}^{(t)}\|$, $\|\mathbf{x}_1^{(t)} - \bar{\mathbf{x}}_{v,1}^{(t)}\|$ and $\|\bar{\mathbf{x}}_1^{(t)} - \mathbf{x}_1^*\|$ converge at a linear rate. To find the required condition on α, we show in Lemma 7 provided in Appendix E that if $\alpha < \frac{\lambda_1 - \lambda_2}{42}(\frac{1-\beta}{9\lambda_1})^2$, the spectral radius of $\mathbf{P}(\alpha)$ is strictly less than 1. This implies that if $\alpha < \frac{\lambda_1 - \lambda_2}{42}(\frac{1-\beta}{9\lambda_1})^2$, then $\|\bar{\mathbf{x}}_1^{(t)} - \mathbf{x}_1^*\|$, $\|\mathbf{x}_{i,1}^{(t)} - \bar{\mathbf{x}}_1^{(t)}\|$ and $\|\mathbf{s}_{i,1}^{(t)} - \mathbf{g}_1^{(t)}\|$ converge at a linear rate to 0. In other words, $\mathbf{x}_{i,1}^{(t)}$ converges linearly to $\mathbf{x}_1^* = \pm c_1 \mathbf{q}_1$, where c_1 is some constant. ∎

Case II for Induction – $2 \leq k \leq K$:
We proceed with the proof of convergence for the rest of the eigenvectors through induction. Assume that $\mathbf{x}_{i,p}^{(t)}$ converges to $\pm c_p \mathbf{q}_p$ for $p = 1, \ldots, k-1$ linearly, i.e., there exist constants $b_i > 0$ and $\nu_i < 1$ such that

$$
\|\sum_{p=1}^{k-1} \Big(\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2} - \mathbf{q}_p \mathbf{q}_p^T \Big)\| \leq b_i \nu_i^t. \quad (42)
$$

In Case I, we proved $\mathbf{x}_{i,1}^{(t)}$ converges to $\pm c_1 \mathbf{q}_1$ linearly. By induction, we assume $\mathbf{x}_{i,p}^{(t)}$ converges to $\pm c_p \mathbf{q}_p$ for $p = 1, \ldots, (k-1)$ at a linear rate, which leads to the inequality. We use (42) to prove $\mathbf{x}_{i,k}^{(t)}$ converges to $\pm c_k \mathbf{q}_k$.

Lemma 3. *The function* $\mathbf{h}_{i,t} : \mathbb{R}^d \to \mathbb{R}^d$ *with* $\mathbf{h}_{i,t}(\mathbf{v}) = \mathbf{C}_i \mathbf{v} - \frac{(\mathbf{v})^T \mathbf{C}_i \mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{v} - - \sum_{p=1}^{k-1} \frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\mathbf{C}_i \mathbf{v}$ *is Lipschitz continuous with constant* $L_k = \lambda_1(k+5)$.

The proof of this lemma is deferred to Appendix F. Using this lemma and the definition of $\mathbf{g}(\mathbf{x}_k)$, the following lemma holds true, the proof of which is the same as that of Lemma 2.

Lemma 4. *The following inequalities hold with* $L_k = \lambda_1(k+5)$:

1) $\|\mathbf{h}(\mathbf{x}_k^{(t)}) - \mathbf{h}(\mathbf{x}_k^{(t-1)})\|_2 \leq L_k \|\mathbf{x}_k^{(t)} - \mathbf{x}_k^{(t-1)}\|_2$
2) $\|\mathbf{g}(\mathbf{x}_k^{(t)}) - \mathbf{g}(\mathbf{x}_k^{(t-1)})\|_2 \leq \frac{L_k}{\sqrt{M}}\|\mathbf{x}_k^{(t)} - \mathbf{x}_k^{(t-1)}\|_2$
3) $\|\mathbf{g}(\mathbf{x}_k^{(t)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,k}^{(t)})\|_2 \leq \frac{L_k}{\sqrt{M}}\|\mathbf{x}_k^{(t)} - \mathbf{x}_{s,k}^{(t)}\|_2$

Proof of Theorem 1 for $k > 1$:

Using the definitions of $\mathbf{x}_k^{(t)}$, $\mathbf{s}_k^{(t)}$, $\mathbf{g}_s(\mathbf{x}_k^{(t)})$, $\mathbf{h}(\mathbf{x}_k^{(t)})$ and same algebraic manipulations as in proof for the case of $k = 1$, we get

$$
\|\mathbf{s}_k^{(t)} - \mathbf{g}_s(\mathbf{x}_k^{(t)})\| \leq \frac{1+\beta}{2}\|\mathbf{s}_k^{(t-1)} - \mathbf{g}_s(\mathbf{x}_k^{(t-1)})\|
$$
$$
+ L_k \|\mathbf{x}_k^{(t)} - \mathbf{x}_k^{(t-1)}\| \quad (43)
$$

and

$$
\|\mathbf{x}_k^{(t)} - \bar{\mathbf{x}}_{s,k}^{(t)}\| \leq \frac{1+\beta}{2}\|\mathbf{x}_k^{(t-1)} - \bar{\mathbf{x}}_{s,k}^{(t-1)}\| + \alpha\|\mathbf{s}_k^{(t-1)} - \mathbf{g}_s(\mathbf{x}_k^{(t-1)})\|. \quad (44)
$$

Now, we bound $\|\bar{\mathbf{x}}_k^{(t)} - \mathbf{x}_k^*\|$. We know

$$
\mathbf{g}(\bar{\mathbf{x}}_{s,k}^{(t)})
$$
$$
= \frac{1}{M}\Big(\mathbf{C}\bar{\mathbf{x}}_k^{(t)} - \frac{(\bar{\mathbf{x}}_k^{(t)})^T \mathbf{C}\bar{\mathbf{x}}_k^{(t)}}{\|\bar{\mathbf{x}}_k^{(t)}\|^2}\bar{\mathbf{x}}_k^{(t)} - \sum_{i=1}^{M}\sum_{p=1}^{k-1} \frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\mathbf{C}_i \bar{\mathbf{x}}_k^{(t)} \Big)
$$
$$
= \frac{1}{M}\Big(\mathbf{C}\bar{\mathbf{x}}_k^{(t)} - \frac{(\bar{\mathbf{x}}_k^{(t)})^T \mathbf{C}\bar{\mathbf{x}}_k^{(t)}}{\|\bar{\mathbf{x}}_k^{(t)}\|^2}\bar{\mathbf{x}}_k^{(t)} - \sum_{i=1}^{M}\sum_{p=1}^{k-1} \mathbf{q}_p \mathbf{q}_p^T \mathbf{C}_i \bar{\mathbf{x}}_k^{(t)} \Big)
$$
$$
- \frac{1}{M}\sum_{i=1}^{M}\sum_{p=1}^{k-1} \Big(\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2} - \mathbf{q}_p \mathbf{q}_p^T \Big)\mathbf{C}_i \bar{\mathbf{x}}_k^{(t)}
$$
$$
= \frac{1}{M}\Big(\mathbf{C}\bar{\mathbf{x}}_k^{(t)} - \frac{(\bar{\mathbf{x}}_k^{(t)})^T \mathbf{C}\bar{\mathbf{x}}_k^{(t)}}{\|\bar{\mathbf{x}}_k^{(t)}\|^2}\bar{\mathbf{x}}_k^{(t)} - \sum_{p=1}^{k-1} \mathbf{q}_p \mathbf{q}_p^T \mathbf{C}\bar{\mathbf{x}}_k^{(t)} \Big)
$$
$$
- \frac{1}{M}\sum_{i=1}^{M}\sum_{p=1}^{k-1} \Big(\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2} - \mathbf{q}_p \mathbf{q}_p^T \Big)\mathbf{C}_i \bar{\mathbf{x}}_k^{(t)}
$$
$$
= \mathbf{g}'(\bar{\mathbf{x}}_{s,k}^{(t)}) - \mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t)}),
$$

where $\mathbf{g}'(\bar{\mathbf{x}}_{s,k}^{(t)}) = \frac{1}{M}(\mathbf{C}\bar{\mathbf{x}}_k^{(t)} - \frac{(\bar{\mathbf{x}}_k^{(t)})^T \mathbf{C}\bar{\mathbf{x}}_k^{(t)}}{\|\bar{\mathbf{x}}_k^{(t)}\|^2}\bar{\mathbf{x}}_k^{(t)} - \sum_{p=1}^{k-1} \mathbf{q}_p \mathbf{q}_p^T \mathbf{C}\bar{\mathbf{x}}_k^{(t)})$ and $\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t)}) = \frac{1}{M}\sum_{i=1}^{M}\sum_{p=1}^{k-1}(\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2} - \mathbf{q}_p \mathbf{q}_p^T)\mathbf{C}_i \bar{\mathbf{x}}_k^{(t)}$. From (23), we have

$$
\bar{\mathbf{x}}_k^{(t)} = \bar{\mathbf{x}}_k^{(t-1)} + \alpha\bar{\mathbf{s}}_k^{(t-1)} = \bar{\mathbf{x}}_k^{(t-1)} + \alpha\mathbf{g}(\mathbf{x}_k^{(t-1)})
$$
$$
= \bar{\mathbf{x}}_k^{(t-1)} + \alpha\mathbf{g}(\bar{\mathbf{x}}_{s,k}^{(t-1)}) + \alpha(\mathbf{g}(\mathbf{x}_k^{(t-1)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,k}^{(t-1)}))
$$
$$
= \bar{\mathbf{x}}_k^{(t-1)} + \frac{\alpha}{M}(\mathbf{C}\bar{\mathbf{x}}_k^{(t-1)} - \frac{(\bar{\mathbf{x}}_k^{(t-1)})^T \mathbf{C}\bar{\mathbf{x}}_k^{(t-1)}}{\|\bar{\mathbf{x}}_k^{(t-1)}\|^2}\bar{\mathbf{x}}_k^{(t-1)} -
$$
$$
\sum_{p=1}^{k-1} \mathbf{q}_p \mathbf{q}_p^T \mathbf{C}\bar{\mathbf{x}}_k^{(t-1)}) - \alpha\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t-1)}) + \alpha(\mathbf{g}(\mathbf{x}_k^{(t-1)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,k}^{(t-1)})).
$$

Using (19) and (20), we know that an iterate of the form

$$
\bar{\mathbf{x}}_k^{(t)} = \bar{\mathbf{x}}_k^{(t-1)} + \alpha(\mathbf{C}\bar{\mathbf{x}}_k^{(t-1)} - \frac{(\bar{\mathbf{x}}_k^{(t-1)})^T \mathbf{C}\bar{\mathbf{x}}_k^{(t-1)}}{\|\bar{\mathbf{x}}_k^{(t-1)}\|^2}\bar{\mathbf{x}}_k^{(t-1)}
$$
$$
- \sum_{p=1}^{k-1} \mathbf{q}_p \mathbf{q}_p^T \mathbf{C}\bar{\mathbf{x}}_k^{(t-1)})
$$

converges linearly for $\alpha < \frac{1}{\lambda_1}$ and $\mathbf{q}_k^T \bar{\mathbf{x}}_k^{(0)} \neq 0$ as

$$
\|\bar{\mathbf{x}}_k^{(t)} - \mathbf{x}_k^*\| \leq \delta_k \|\bar{\mathbf{x}}_k^{(t-1)} - \mathbf{x}_k^*\|,
$$

where $\mathbf{x}_k^* = \pm c_k \mathbf{q}_k$ and $\delta_k = \frac{1+\alpha\lambda_{k+1}}{1+\alpha\lambda_k}$. Thus, using (19) and (20), we know

$$\|\bar{\mathbf{x}}_k^{(t)} - \mathbf{x}_k^*\| \le \delta_k \|\bar{\mathbf{x}}_k^{(t-1)} - \mathbf{x}_k^*\| + \alpha\|\mathbf{g}(\mathbf{x}_k^{(t-1)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,k}^{(t-1)})\| + \alpha\|\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t-1)})\| \quad (45)$$

$$\le \delta_k \|\bar{\mathbf{x}}_k^{(t-1)} - \mathbf{x}_k^*\| + \alpha\frac{L_k}{\sqrt{M}}\|\mathbf{x}_k^{(t-1)} - \bar{\mathbf{x}}_{s,k}^{(t-1)}\| + \alpha\|\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t-1)})\|. \quad (46)$$

Now, we will bound $\|\mathbf{x}_k^{(t)} - \mathbf{x}_k^{(t-1)}\|$. Since $\mathbf{g}'(\bar{\mathbf{x}}_{s,k}^{(t)}) = \frac{1}{M}(\mathbf{C}\bar{\mathbf{x}}_k^{(t)} - \frac{(\bar{\mathbf{x}}_k^{(t)})^T\mathbf{C}\bar{\mathbf{x}}_k^{(t)}}{\|\bar{\mathbf{x}}_k^{(t)}\|^2}\bar{\mathbf{x}}_k^{(t)} - \sum_{p=1}^{k-1}\mathbf{q}_p\mathbf{q}_p^T\mathbf{C}\bar{\mathbf{x}}_k^{(t)})$, we have $\mathbf{g}'(\mathbf{x}_{s,k}^*) = \frac{1}{M}(c_k\mathbf{C}\mathbf{q}_k - \frac{c_k\mathbf{q}_k^T\mathbf{C}c_k\mathbf{q}_k}{c_k^2\|\mathbf{q}_k\|^2}\mathbf{q}_k - \sum_{p=1}^{k-1}\mathbf{q}_p\mathbf{q}_p^Tc_k\mathbf{q}_k) = 0$. Hence,

$$\|\mathbf{g}'_s(\bar{\mathbf{x}}_{s,k}^{(t-1)})\| = \sqrt{M}\|\mathbf{g}'(\bar{\mathbf{x}}_{s,k}^{(t-1)})\|$$
$$= \sqrt{M}\|\mathbf{g}'(\bar{\mathbf{x}}_{s,k}^{(t-1)}) - \mathbf{g}'(\mathbf{x}_{s,k}^*)\|$$
$$\le L_k\sqrt{M}\|\bar{\mathbf{x}}_k^{(t-1)} - \mathbf{x}_k^*\|.$$

Using the above inequality and Lemma 4, we get

$$\|\mathbf{s}_k^{(t-1)}\| = \|\mathbf{s}_k^{(t-1)} - \mathbf{g}_s(\mathbf{x}_k^{(t-1)}) + \mathbf{g}_s(\mathbf{x}_k^{(t-1)}) - \mathbf{g}_s(\bar{\mathbf{x}}_{s,k}^{(t-1)}) + \mathbf{g}'_s(\bar{\mathbf{x}}_{s,k}^{(t-1)}) - \mathbf{f}_s(\bar{\mathbf{x}}_{s,k}^{(t-1)})\|$$
$$\le \|\mathbf{s}_k^{(t-1)} - \mathbf{g}_s(\mathbf{x}_k^{(t-1)})\| + \|\mathbf{g}_s(\mathbf{x}_k^{(t-1)}) - \mathbf{g}_s(\bar{\mathbf{x}}_{s,k}^{(t-1)})\| + \|\mathbf{g}'_s(\bar{\mathbf{x}}_{s,k}^{(t-1)})\| + \|\mathbf{f}_s(\bar{\mathbf{x}}_{s,k}^{(t-1)})\|$$
$$\le \|\mathbf{s}_k^{(t-1)} - \mathbf{g}_s(\mathbf{x}_k^{(t-1)})\| + L_k\|\mathbf{x}_k^{(t-1)} - \bar{\mathbf{x}}_{s,k}^{(t-1)}\| + L_k\sqrt{M}\|\bar{\mathbf{x}}_k^{(t-1)} - \mathbf{x}_k^*\| + \sqrt{M}\|\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t-1)})\|. \quad (47)$$

Thus,

$$\|\mathbf{x}_k^{(t)} - \mathbf{x}_k^{(t-1)}\| = \|(\mathbf{W}\otimes\mathbf{I})\mathbf{x}_k^{(t-1)} - \mathbf{x}_k^{(t-1)} + \alpha\mathbf{s}_k^{(t-1)}\|$$
$$= \|(\mathbf{W}\otimes\mathbf{I} - \mathbf{I})(\mathbf{x}_k^{(t-1)} - \bar{\mathbf{x}}_{s,k}^{(t-1)}) + \alpha\mathbf{s}_k^{(t-1)}\|$$
$$\le 2\|\mathbf{x}_k^{(t-1)} - \bar{\mathbf{x}}_{s,k}^{(t-1)}\| + \alpha\|\mathbf{s}_k^{(t-1)}\|$$
$$\le \alpha\|\mathbf{s}_k^{(t-1)} - \mathbf{g}_s(\mathbf{x}_k^{(t-1)})\| + (2 + \alpha L_k)\times \quad (48)$$
$$\|\mathbf{x}_k^{(t-1)} - \bar{\mathbf{x}}_{s,k}^{(t-1)}\| + \alpha L_k\sqrt{M}\|\bar{\mathbf{x}}_k^{(t-1)} - \mathbf{x}_k^*\| + \alpha\sqrt{M}\|\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t-1)})\| \quad \text{using (47).} \quad (49)$$

Using the above inequality in (43), we get

$$\|\mathbf{s}_k^{(t)} - \mathbf{g}_s(\mathbf{x}_k^{(t)})\| \le (\frac{1+\beta}{2} + \alpha L_k)\|\mathbf{s}_k^{(t-1)} - \mathbf{g}_s(\mathbf{x}_k^{(t-1)})\| + L_k(2 + \alpha L_k)\|\mathbf{x}_k^{(t-1)} - \bar{\mathbf{x}}_{s,k}^{(t-1)}\| + \alpha L_k^2\sqrt{M}\|\bar{\mathbf{x}}_k^{(t-1)} - \mathbf{x}_k^*\| + \alpha L_k\sqrt{M}\|\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t-1)})\|. \quad (50)$$

Writing a system of equations from (50), (44) and (46), we have the following:

$$\begin{bmatrix} \|\mathbf{s}_k^{(t)} - \mathbf{g}_s(\mathbf{x}_k^{(t)})\| \\ \|\mathbf{x}_k^{(t)} - \bar{\mathbf{x}}_k^{(t)}\| \\ \sqrt{M}\|\bar{\mathbf{x}}_k^{(t)} - \mathbf{x}_k^*\| \end{bmatrix} \le \begin{bmatrix} (\frac{1+\beta}{2} + \alpha L_k) & L_k(2+\alpha L_k) & \alpha L_k^2 \\ \alpha & \frac{1+\beta}{2} & 0 \\ 0 & \alpha L_k & \delta_k \end{bmatrix}$$
$$\times \begin{bmatrix} \|\mathbf{s}_k^{(t-1)} - \mathbf{g}_s(\mathbf{x}_k^{(t-1)})\| \\ \|\mathbf{x}_k^{(t-1)} - \bar{\mathbf{x}}_k^{(t-1)}\| \\ \sqrt{M}\|\bar{\mathbf{x}}_k^{(t-1)} - \mathbf{x}_k^*\| \end{bmatrix} + \|\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t-1)})\|\begin{bmatrix} \alpha L_k\sqrt{M} \\ 0 \\ \alpha\sqrt{M} \end{bmatrix}. \quad (51)$$

Let us define $\mathbf{P}_k(\alpha) = \begin{bmatrix} (\frac{1+\beta}{2} + \alpha L_k) & L_k(2+\alpha L_k) & \alpha L_k^2 \\ \alpha & \frac{1+\beta}{2} & 0 \\ 0 & \alpha L_k & \rho_k \end{bmatrix}$. Since $\mathbf{P}_k(\alpha)$ has non-negative entries and $\mathbf{P}_k^2(\alpha)$ has all positive entries, each entry of $\mathbf{P}_k^t(\alpha)$ will be $\mathcal{O}(\rho(\mathbf{P}_k(\alpha))^t)$, where $\rho(\mathbf{P}_k(\alpha))$ is the spectral radius of $\mathbf{P}_k(\alpha)$. From Lemma 7 in Appendix E, we know if we choose $\alpha < \frac{\lambda_k - \lambda_{k+1}}{(k+5)(k+6)}(\frac{1-\beta}{9\lambda_1})^2$, then $\rho(\mathbf{P}_k(\alpha)) < 1$. Also, we know $\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t)}) = \frac{1}{M}\sum_{i=1}^M\sum_{p=1}^{k-1}(\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2} - \mathbf{q}_p\mathbf{q}_p^T)\mathbf{C}_i\bar{\mathbf{x}}_k^{(t)}$. Thus,

$$\|\mathbf{f}(\bar{\mathbf{x}}_{s,k}^{(t)})\| = \|\frac{1}{M}\sum_{i=1}^M\sum_{p=1}^{k-1}(\frac{\mathbf{x}_{i,p}^{(t-1)}(\mathbf{x}_{i,p}^{(t-1)})^T}{\|\mathbf{x}_{i,p}^{(t-1)}\|^2} - \mathbf{q}_p\mathbf{q}_p^T)\mathbf{C}_i\bar{\mathbf{x}}_k^{(t-1)}\|$$
$$\le \frac{1}{M}\sum_{i=1}^M\|\sum_{p=1}^{k-1}(\frac{\mathbf{x}_{i,p}^{(t-1)}(\mathbf{x}_{i,p}^{(t-1)})^T}{\|\mathbf{x}_{i,p}^{(t-1)}\|^2} - \mathbf{q}_p\mathbf{q}_p^T)\|\|\mathbf{C}_i\|\|\bar{\mathbf{x}}_k^{(t-1)}\|.$$

From (42), we know $\|\sum_{p=1}^{k-1}(\frac{\mathbf{x}_{i,p}^{(t-1)}(\mathbf{x}_{i,p}^{(t-1)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2} - \mathbf{q}_p\mathbf{q}_p^T)\| \le b_i\nu_i^t$. Let $b = (\max_i b_i)\lambda_1\|\bar{\mathbf{x}}_k^{(t-1)}\| > 0$ and $\nu = \max_i\nu_i < 1$. Thus the system of equations becomes

$$\begin{bmatrix} \|\mathbf{s}_k^{(t)} - \mathbf{g}_s(\mathbf{x}_k^{(t)})\| \\ \|\mathbf{x}_k^{(t)} - \bar{\mathbf{x}}_{s,k}^{(t)}\| \\ \sqrt{M}\|\bar{\mathbf{x}}_k^{(t)} - \mathbf{x}_k^*\| \end{bmatrix} \le \rho(\mathbf{P}_k(\alpha))^t \begin{bmatrix} \|\mathbf{s}_k^{(0)} - \mathbf{g}_s(\mathbf{x}_k^{(0)})\| \\ \|\mathbf{x}_k^{(0)} - \bar{\mathbf{x}}_{s,k}^{(0)}\| \\ \sqrt{M}\|\bar{\mathbf{x}}_k^{(0)} - \mathbf{x}_k^*\| \end{bmatrix} + b\nu^t\begin{bmatrix} \alpha L_k\sqrt{M} \\ 0 \\ \alpha\sqrt{M} \end{bmatrix}. \quad (52)$$

This implies that if $\alpha < \frac{\lambda_k - \lambda_{k+1}}{(k+5)(k+6)}(\frac{1-\beta}{9\lambda_1})^2$, then $\|\bar{\mathbf{x}}_k^{(t)} - \mathbf{x}_k^*\|$, $\|\mathbf{x}_{i,k}^{(t)} - \bar{\mathbf{x}}_k^{(t)}\|$ and $\|\mathbf{s}_{i,k}^{(t)} - \mathbf{g}_k^{(t)}\|$ converge at a linear rate to 0. In other words, $\mathbf{x}_{i,k}^{(t)}$ converges linearly to $\mathbf{x}_k^* = \pm c_k\mathbf{q}_k$, where c_k is some constant. ∎

In summary, FAST-PCA converge exactly to the true eigenvectors whilst completely doing away with the need of explicit consensus loop. Table II provides a comparison of the communication and iteration complexities of various distributed PCA and PSA algorithms in terms of error ϵ and eigengap gap. Since our proposed algorithm has a reduced dependence of the total iteration complexity on gap, our solutions are significantly faster than other algorithms as also shown through numerical experiments in the next section. It is worth noting here that the total iteration complexity of DeEPCA and FAST-PCA become comparable in the case of small gap since $\log(1+x) \approx x$ for small x. Nonetheless, the total communication cost—a major performance metric for any distributed algorithm—of FAST-PCA in this case would still be less than that of DeEPCA by a factor of $\log\frac{1}{gap}$, which would be a significant factor for small gap. The computational complexity per iteration per node of the proposed method is $\mathcal{O}(K^2d)$. Additionally, in the statement of Theorem 1, it requires that the average of the initial $\bar{\mathbf{x}}_k^{(0)}$ be not orthogonal to $\mathbf{q}_k$. Such condition is easy to meet in practical applications, since any small disturbance can make the initialization meet this condition. Thus, random initialization at each node will work for all practical purposes.

TABLE II: Comparison of Communication and Iteration Costs of FAST-PCA With Related Methods

	Comm./Iteration	No. of Iterations	Total Comm.	PCA/PSA
DistSeqPM [38]–[40]	$\mathcal{O}\left(K\frac{1}{\log gap_r^{-1}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(K\frac{1}{\log gap_r^{-1}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(K^2\frac{1}{\log^2 gap_r^{-1}}\log^2\frac{1}{\epsilon}\right)$	PCA
S-DOT [42]	$\mathcal{O}\left(\frac{1}{\log gap_r^{-1}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{\log gap_r^{-1}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{\log^2 gap_r^{-1}}\log^2\frac{1}{\epsilon}\right)$	PSA
DeEPCA [54]	$\mathcal{O}\left(\log\frac{1}{gap}\right)$	$\mathcal{O}\left(\frac{1}{gap}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{gap}\log\frac{1}{gap}\log\frac{1}{\epsilon}\right)$	PSA
DSA [49]	1	$\mathcal{O}\left(\frac{1}{\log(1+\alpha gap)}\log\frac{1}{\epsilon}\right)$ up to $\epsilon=\mathcal{O}(\alpha)$	$\mathcal{O}\left(\frac{1}{\log(1+\alpha gap)}\log\frac{1}{\epsilon}\right)$	PCA
FAST-PCA (This Work)	2	$\mathcal{O}\left(\frac{1}{\log(1+\alpha gap)}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{\log(1+\alpha gap)}\log\frac{1}{\epsilon}\right)$	PCA

Furthermore, the convergence results can be extended to the case of repeated eigenvalues through some straightforward but tedious calculations. In that case the iterates can be proved to converge to a vector in the subspace spanned by the eigenvectors corresponding to the (repeated) eigenvalue. Due to space constraints, we leave that extension for future work.

V. EXPERIMENTAL RESULTS

In this section, we demonstrate the efficacy of our proposed FAST-PCA algorithm through experiments on synthetic as well as real-world data. We compare the performance of our algorithm with existing algorithms of (centralized) orthogonal iteration (OI), (centralized) sequential power method (SeqPM), distributed sequential power method (SeqDistPM), distributed orthogonal iteration algorithms (S-DOT, SA-DOT) [42], an orthogonal iteration+gradient tracking based method DeEPCA [54] and our previously proposed distributed Sanger's algorithm (DSA) [49]. In the case of OI and SeqPM, we assume that all the samples are available at a single location and, for the estimation of K dominant eigenvectors of $\mathbf{C}$, SeqPM performs power method K times sequentially starting from the most dominant eigenvector. SeqDistPM is the distributed version of SeqPM, which uses an explicit consensus loop with a fixed number T_c of consensus iterations per iteration of the power iteration [38], [39], whereas S-DOT and SA-DOT are distributed versions of OI using fixed and increasing number of consensus iterations per orthogonal iteration. The DSA is a distributed generalized Hebbian algorithm that converges linearly to a neighborhood of the true eigenvectors of the global covariance matrix. Assuming that the cost of communicating $\mathbb{R}^{d\times K}$ matrices across the network in one (outer loop) iteration is one unit, the x-axes of all the plots indicate the total communication cost, i.e., total inner and outer loop communications. In the algorithms with one time scale, this is the same as the number of total outer loop iterations (since inner iterations = 0). The y-axes of the plots express the average angle between the estimated eigenvectors $\mathbf{x}_{i,k}^{(t)}$ and the true eigenvectors $\pm\mathbf{q}_k$ across all the M nodes in the network given by

$$\mathcal{E} = \frac{1}{MK}\sum_{i=1}^{M}\sum_{k=1}^{K}\left(1 - \left(\frac{\mathbf{x}_{i,k}^T\mathbf{q}_k}{\|\mathbf{x}_{i,k}\|}\right)^2\right). \tag{53}$$

A. Synthetic Data

We study the effects of factors like eigengap and distinct/repeated eigenvalues on the performance of our algorithm in comparison to various other existing PCA and distributed PCA algorithms. To that end, we generate Erdos–Renyi graphs ($p = 0.5$) and cyclic graphs to simulate the distributed setup with $M = 20$ nodes. We also generate synthetic data with different (ratio) eigengaps of $gap_r = \frac{\lambda_{K+1}}{\lambda_K} \in \{0.8, 0.97\}$. The data is generated such that each node has 5000 i.i.d samples, i.e., $N_i = 5000$ with $d = 20$ drawn from a multivariate Gaussian distribution with zero mean and fixed covariance matrix $\boldsymbol{\Sigma}$. The number of eigenvectors to be estimated is set to $K = 5$. For SeqPM, SeqDistPM and S-DOT, the number of consensus iterations per outer loop iteration is $T_c = 50$ and the number of maximum consensus iterations in the case of SA-DOT is set to 50 as well. For the Erdos–Renyi topology, we use a step size of $\alpha = 0.7$ for our algorithm and for cyclic graph, we use $\alpha = 0.1$. These values correspond to the best performing step sizes chosen after trial-and-error. The results reported are an average of 10 Monte-Carlo simulations each for a different random initialization.

Figure 1 compares the performance of our proposed FAST-PCA algorithm with centralized OI, SeqPM, SeqDistPM, S-DOT, SA-DOT, DeEPCA and DSA when the subspace eigenvalues $\lambda_1, \ldots, \lambda_K$ are distinct, i.e., $\lambda_1 > \lambda_2 > \ldots > \lambda_K$. It is clear that our algorithm significantly outperforms SeqPM and SeqDistPM since estimating one eigenvector at a time slows down the convergence of these methods. Also, the requirement of an explicit consensus loop implies the communication cost of these methods is high as indicated by the plots. Even though S-DOT and SA-DOT estimate the whole subspace (but not necessarily the eigenvectors) simultaneously, an explicit consensus loop makes those relatively slow as well. As expected, since DSA converges only to a neighborhood of the true solutions, our new proposed algorithm outperforms it. The performance of FAST-PCA is better than of DeEPCA in case of Erdos–Renyi graph, but deprecates for a cycle graph. It is desired from any distributed algorithm to perform similar to their centralized counterparts and it is clear from the figures that our algorithm FAST-PCA performs very similar to centralized OI.

Figure 2 shows a similar performance comparison when the subspace eigenvalues are very close to each other, i.e. $\lambda_1 \approx \lambda_2 \approx \ldots \approx \lambda_K$. The Gaussian distribution generated in this case has covariance matrix $\boldsymbol{\Sigma}$ with equal subspace eigenvalues but due to the finite number of samples, the eigenvalues of $\mathbf{C}$ are not exactly equal albeit almost equal. It is evident that the performance of every algorithm significantly deprecates in this

(a) Erdos–Renyi, $gap_r = 0.8$

(b) Erdos–Renyi, $gap_r = 0.97$

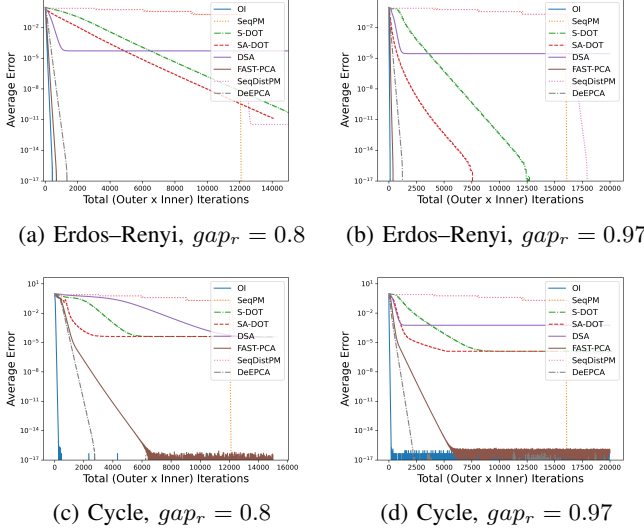(c) Cycle, $gap_r = 0.8$

(d) Cycle, $gap_r = 0.97$

Fig. 1: Performance comparison of FAST-PCA with various algorithms for two different eigengaps and two graph topologies. Here, the top $K + 1$ eigenvalues of $\mathbf{C}$ are distinct.

(a) Erdos–Renyi, $gap_r = 0.8$

(b) Erdos–Renyi, $gap_r = 0.97$

(c) Cycle, $gap_r = 0.8$

(d) Cycle, $gap_r = 0.97$

Fig. 2: Performance comparison of FAST-PCA with various algorithms for two different eigengaps and two graph topologies in the case of first K (almost) equal eigenvalues.

TABLE III: Effect of Network Size on the Runtime

M	10	20	50
Runtime (in secs)	0.98	1.75	4

scenario. Nonetheless, in this case FAST-PCA outperforms all other algorithms including DeEPCA, while still being close to centralized OI in terms of performance.

As already discussed, one of the key attributes of our proposed FAST-PCA algorithm is that it is a one time-scale algorithm. As evident from Table II, all algorithms including FAST-PCA require more iterations when subspace eigenvalues are close to each other. The main advantage of FAST-PCA over other competing algorithms including DeEPCA is reduced dependence on eigengap. This is further illustrated through Figure 3a which shows the effect of change in eigengap on the iteration complexity of FAST-PCA and DeEPCA. Reducing the eigengap by half increases the convergence time of both algorithms but the increase in DeEPCA is nearly 3 times, whereas the increase in FAST-PCA is less than 2. Here, $d = 200$, $K = 5$. Figure 3b shows the effect of graph connectivity on the performance of FAST-PCA. As expected, smaller β i.e., stronger graph connectivity leads to faster convergence. Furthermore, the performance for a fully connected graph is same as that of the centralized solution. Here, we used $d = 20, K = 1, gap = 0.2$.

Another network parameter that we study is the network size, i.e., number of nodes in the network M. For an Erdos–Renyi graph, if we keep the connectivity factor p constant, say $p = 0.5$, then as M increases the number of connections in the network increases, thereby reducing β, which in turn decreases the iteration complexity. But more connections imply more communications and bigger network size imply more computations per round, which increases the overall runtime of the algorithm. Table III shows the effect of network size on the runtime of the proposed FAST-PCA algorithm.
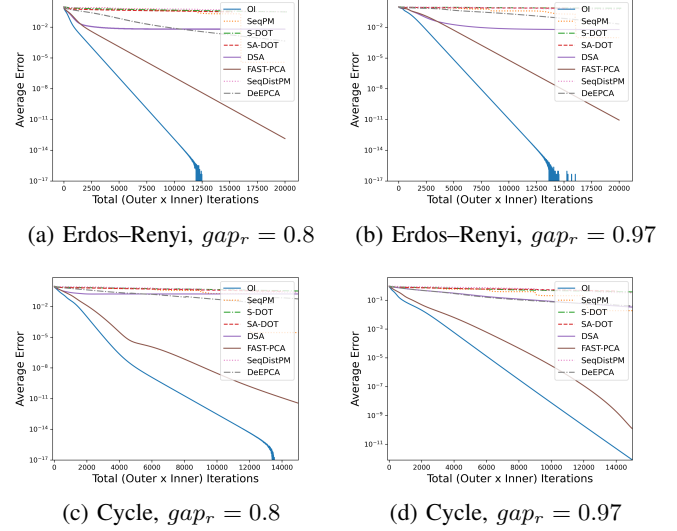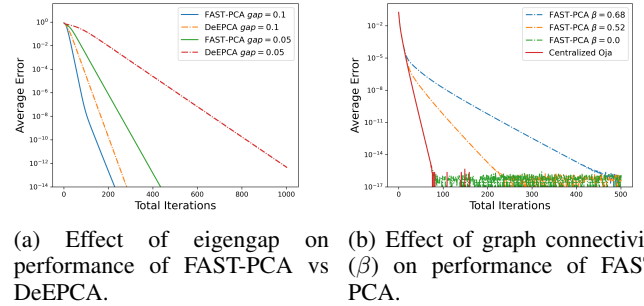

(a) Effect of eigengap on performance of FAST-PCA vs DeEPCA.

(b) Effect of graph connectivity (β) on performance of FAST-PCA.

Fig. 3: Effect of various parameters on the performance of FAST-PCA.

B. Real-World Data

We also provide some results for the real-world datasets of MNIST [57] and CIFAR10 [58]. We simulate the distributed setup with an Erdos–Renyi graph with $p = 0.5$ and $M = 20$ nodes. Both these datasets have $N = 60,000$ samples distributed equally among the nodes, making $N_i = 3000$. The data dimensions are $d = 784$ for MNIST and $d = 1024$ for CIFAR10. Figure 4a shows the comparison of the various PCA algorithms for MNIST dataset when $K = 10$ dominant eigenvectors are estimated. The step size used for FAST-PCA and DSA in this case is $\alpha = 0.1$. Similar results for CIFAR10 are shown in Figure 4b when $K = 5$ and $\alpha = 0.8$ is used.

VI. CONCLUSION

In this paper, we proposed and analyzed a novel algorithm for distributed Principal Component Analysis (PCA) that truly serves the complete purpose of dimension reduction and uncorrelated feature learning in the scenario where data samples are distributed across a network. We provided detailed theoretical analysis to prove that our proposed algorithm converges linearly, exactly and globally, i.e., starting from any random unit
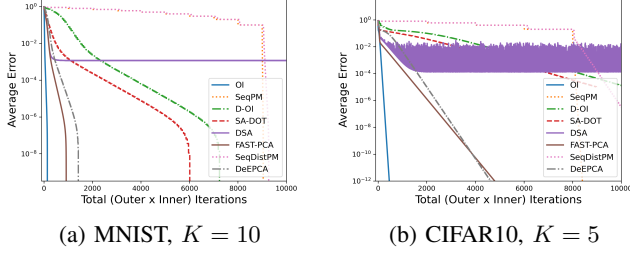
(a) MNIST, $K = 10$ (b) CIFAR10, $K = 5$

Fig. 4: Performance comparison of FAST-PCA with various algorithms for MNIST and CIFAR10.

vectors, to the eigenvectors of the global covariance matrix. We also provided experimental results that further validate our claims and demonstrate the communication efficiency and overall effectiveness of our solution. In the future, we aim to solve the problem of distributed PCA for estimation of multiple eigenvectors in the case of streaming data. Other possible directions are considering asynchronicity in the network and the case of directed and time-varying graphs.

APPENDIX A
STATEMENT AND PROOF OF LEMMA 5

Lemma 5. *Suppose $z_{k,k}^{(0)} \neq 0$ and $\alpha < \frac{1}{\lambda_1}$. Then the following is true for $\gamma_k = \left(\frac{1}{1+\alpha\lambda_k}\right)^2 < 1$ and some constant $a_1 > 0$:*

$$\sum_{l=1}^{k-1} (z_{k,l}^{(t+1)})^2 \leq a_1 \gamma_k^{t+1}. \tag{54}$$

Proof. For $l = 1, \ldots, k-1$ we know from (17) that $z_{k,l}^{(t+1)} = a_k^{(t)} \big(1 - \alpha (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C} \tilde{\mathbf{x}}_{g,k}^{(t)} \big) z_{k,l}^{(t)}$. Since $(\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C} \tilde{\mathbf{x}}_{g,k}^{(t)} \leq \lambda_1 < \frac{1}{\alpha}$, we have $1 + \alpha(\lambda_k - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C} \tilde{\mathbf{x}}_{g,k}^{(t)}) > \alpha\lambda_k \geq 0$.

Thus, we have for $l = 1, \cdots k-1$,

$$\left(\frac{z_{k,l}^{(t+1)}}{z_{k,k}^{(t+1)}}\right)^2 = \left(\frac{1 - \alpha(\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)}}{1 + \alpha(\lambda_k - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)})}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$= \left(1 - \frac{\alpha\lambda_k}{1 + \alpha(\lambda_k - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)})}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$\leq \left(1 - \frac{\alpha\lambda_k}{1 + \alpha\lambda_k}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$= \left(\frac{1}{1 + \alpha\lambda_k}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$= \gamma_k \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2, \quad \gamma_k = \left(\frac{1}{1 + \alpha\lambda_k}\right)^2 < 1.$$

Therefore, for $l = 1, \ldots, k-1$, $(z_{k,l}^{(t+1)})^2 \leq \gamma_k^{t+1} \left(\frac{z_{k,l}^{(0)}}{z_{k,k}^{(0)}}\right)^2 (z_{k,k}^{(t+1)})^2$. Since $\|\tilde{\mathbf{x}}_k^{(t+1)}\|^2 = 1$ and $\|\tilde{\mathbf{x}}_k^{(0)}\|^2 = 1$, hence $(z_{k,k}^{(t+1)})^2 \leq 1$ and $z_{k,l}^{(0)} \leq 1$. Also, because of the

assumption $z_{k,k}^{(0)} \neq 0$, let us assume $(z_{k,k}^{(0)})^2 > \tilde{\eta}$. Thus, we can write

$$\sum_{l=1}^{k-1} (z_{k,l}^{t+1})^2 \leq \gamma_k^{t+1} \sum_{l=1}^{k-1} \frac{1}{\tilde{\eta}} = a_1 \gamma_k^{t+1}. \tag{55}$$

■

APPENDIX B
STATEMENT AND PROOF OF LEMMA 6

Lemma 6. *Suppose $z_{k,k}^{(0)} \neq 0$ and $\alpha < \frac{1}{\lambda_1}$. Then the following is true for $\rho_k = \left(\frac{1+\alpha\lambda_{k+1}}{1+\alpha\lambda_k}\right)^2 < 1$ and some constant $a_2 > 0$:*

$$\sum_{l=k+1}^{d} (z_{k,l}^{(t+1)})^2 \leq a_2 \rho_k^{t+1}. \tag{56}$$

Proof. For $l = k, \ldots, d$ we know from (17) that $z_{k,l}^{(t+1)} = a_k^{(t)} \big(1 + \alpha(\lambda_l - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)})\big) z_{k,l}^{(t)}$. Since $(\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)} \leq \lambda_1 < \frac{1}{\alpha}$, we have $1 + \alpha(\lambda_l - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)}) > \alpha\lambda_l \geq 0, \forall l = k, \ldots, d$.

Thus, we have for $l = k+1, \cdots d$,

$$\left(\frac{z_{k,l}^{(t+1)}}{z_{k,k}^{(t+1)}}\right)^2 = \left(\frac{1 + \alpha(\lambda_l - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)})}{1 + \alpha(\lambda_k - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)})}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$= \left(1 - \frac{\alpha(\lambda_k - \lambda_l)}{1 + \alpha(\lambda_k - (\tilde{\mathbf{x}}_{g,k}^{(t)})^T \mathbf{C}\tilde{\mathbf{x}}_{g,k}^{(t)})}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$\leq \left(1 - \frac{\alpha(\lambda_k - \lambda_l)}{1 + \alpha\lambda_k}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$= \left(\frac{1 + \alpha\lambda_l}{1 + \alpha\lambda_k}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$\leq \left(\frac{1 + \alpha\lambda_{k+1}}{1 + \alpha\lambda_k}\right)^2 \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2$$

$$= \rho_k \left(\frac{z_{k,l}^{(t)}}{z_{k,k}^{(t)}}\right)^2, \quad \rho_k = \left(\frac{1 + \alpha\lambda_{k+1}}{1 + \alpha\lambda_k}\right)^2 < 1.$$

Therefore, for $l = k + 1, \ldots, d$, $(z_{k,l}^{(t+1)})^2 \leq \rho_k^{t+1} \left(\frac{z_{k,l}^{(0)}}{z_{k,k}^{(0)}}\right)^2 (z_{k,k}^{(t+1)})^2$. Since $\|\tilde{\mathbf{x}}_k^{(t+1)}\|^2 = 1$ and $\|\tilde{\mathbf{x}}_k^{(0)}\|^2 = 1$, hence $(z_{k,k}^{(t+1)})^2 \leq 1$ and $z_{k,l}^{(0)} \leq 1$. Also, since $z_{k,k}^{(0)} \neq 0$, let us assume $(z_{k,k}^{(0)})^2 > \tilde{\eta}$. Thus, we can write

$$\sum_{l=k+1}^{d} (z_{k,l}^{t+1})^2 \leq \rho_k^{t+1} \sum_{l=k+1}^{d} \frac{1}{\tilde{\eta}} = a_2 \rho_k^{t+1}. \tag{57}$$

■

APPENDIX C
PROOF OF LEMMA 1

For a continuous and differentiable function $f : \mathbb{R}^d \to \mathbb{R}^d$, we know $\|f(x) - f(y)\| \leq \|\nabla f(x)\|\|x - y\|$. Thus, the Lips-

chitz constant can be given by the upper bound of $\|\nabla f(x)\|$. For the following function,

$$\mathbf{h}_i(\mathbf{v}) = \mathbf{C}_i\mathbf{v} - \frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{v},$$

taking derivative on both sides gives

$$\frac{\partial}{\partial\mathbf{v}}\mathbf{h}_i(\mathbf{v}) = \mathbf{C}_i - \frac{\partial}{\partial\mathbf{v}}\left(\frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\right)\mathbf{v}^T - \frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{I}$$

$$= \mathbf{C}_i - \frac{2\|\mathbf{v}\|^2\mathbf{C}_i\mathbf{v} - 2(\mathbf{v})^T\mathbf{C}_i\mathbf{v}\mathbf{v}}{\|\mathbf{v}\|^4}\mathbf{v}^T - \frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{I}$$

$$= \mathbf{C}_i - \frac{2\mathbf{C}_i\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^2} + \frac{2(\mathbf{v})^T\mathbf{C}_i\mathbf{v}\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^4} - \frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{x}_{i,1}\|^2}\mathbf{I}.$$

$$\left\|\frac{\partial}{\partial\mathbf{v}}\mathbf{h}_i(\mathbf{v})\right\| \leq \|\mathbf{C}_i\| + \left\|\frac{2\mathbf{C}_i\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^2}\right\| + \left\|\frac{2(\mathbf{v})^T\mathbf{C}_i\mathbf{v}\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^4}\right\|$$

$$+ \left\|\frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{I}\right\|$$

$$\leq \lambda_{i,1} + 2\frac{\lambda_{i,1}\|\mathbf{v}\|^2}{\|\mathbf{v}\|^2} + 2|(\mathbf{v})^T\mathbf{C}_i\mathbf{v}|\frac{\|\mathbf{v}\|^2}{\|\mathbf{v}\|^4}$$

$$+ \frac{|(\mathbf{v})^T\mathbf{C}_i\mathbf{v}|}{\|\mathbf{v}\|^2} \quad \text{where,} \quad \lambda_{i,1} = \|\mathbf{C}_i\|$$

$$\leq \lambda_{i,1} + 2\lambda_{i,1} + 2\lambda_{i,1} + \lambda_{i,1} = 6\lambda_{i,1}.$$

Thus,

$$\|\mathbf{h}_i(\mathbf{v}_1) - \mathbf{h}_i(\mathbf{v}_2)\| \leq 6\lambda_{i,1}\|\mathbf{v}_1 - \mathbf{v}_2\| \leq 6\lambda_1\|\mathbf{v}_1 - \mathbf{v}_2\|, \tag{58}$$

where the last inequality uses the fact that $\mathbf{C}_i \preceq \mathbf{C}$, hence $\lambda_{i,1} \leq \lambda_1$. ∎

APPENDIX D
PROOF OF LEMMA 2

This lemma uses Lemma 1 to prove three inequalities that aid in the proof of Theorem 1.

Proof. 1. First, we prove the Lipschitz continuity of the stacked function $\mathbf{h}(\mathbf{x}_1^{(t)})$.

$$\|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\|_2^2 = \sum_{i=1}^M \|\mathbf{h}_i(\mathbf{x}_{i,1}^{(t)}) - \mathbf{h}_i(\mathbf{x}_{i,1}^{(t-1)})\|^2$$

$$\leq L_1^2 \sum_{i=1}^M \|\mathbf{x}_{i,1}^{(t)} - \mathbf{x}_{i,1}^{(t-1)}\|^2$$

$$= L_1^2\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|^2$$

$$\|\mathbf{h}(\mathbf{x}_1^{(t)}) - \mathbf{h}(\mathbf{x}_1^{(t-1)})\|_2 \leq L_1\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|.$$

2. Here, we prove the Lipschitz continuity of the function $\mathbf{g}(\mathbf{x}_1^{(t)})$.

$$\|\mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\mathbf{x}_1^{(t-1)})\|_2^2 = \frac{1}{M^2}\|\sum_{i=1}^M(\mathbf{h}_i(\mathbf{x}_{i,1}^{(t)}) - \mathbf{h}_i(\mathbf{x}_{i,1}^{(t-1)}))\|^2$$

$$\leq \frac{1}{M^2}M\sum_{i=1}^M\|\mathbf{h}_i(\mathbf{x}_{i,1}^{(t)}) - \mathbf{h}_i(\mathbf{x}_{i,1}^{(t-1)})\|^2$$

$$\leq \frac{L_1^2}{M}\sum_{i=1}^M\|\mathbf{x}_{i,1}^{(t)} - \mathbf{x}_{i,1}^{(t-1)}\|^2$$

$$= \frac{L_1^2}{M}\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|^2$$

$$\|\mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\mathbf{x}_1^{(t-1)})\|_2 \leq \frac{L_1}{\sqrt{M}}\|\mathbf{x}_1^{(t)} - \mathbf{x}_1^{(t-1)}\|_2.$$

3. Using the Lipschitz continuous property of $\mathbf{g}(\mathbf{x}_1^{(t)})$, we get

$$\|\mathbf{g}(\mathbf{x}_1^{(t)}) - \mathbf{g}(\bar{\mathbf{x}}_{s,1}^{(t)})\|_2 \leq \frac{L_1}{\sqrt{M}}\|\mathbf{x}_1^{(t)} - \bar{\mathbf{x}}_{s,1}^{(t)}\|_2.$$

∎

APPENDIX E
STATEMENT AND PROOF OF LEMMA 7

Lemma 7. *For a matrix $\mathbf{P}_k(\alpha)$ such that*

$$\mathbf{P}(\alpha) = \begin{bmatrix} (\frac{1+\beta}{2} + \alpha L_k) & L_k(2 + \alpha L_k) & \alpha L_k^2 \\ \alpha & \frac{1+\beta}{2} & 0 \\ 0 & \alpha L_k & \delta_k \end{bmatrix}$$

where $L_k = (k+5)\lambda_1$ and $\delta_k = \frac{1+\alpha\lambda_{k+1}}{1+\alpha\lambda_k}$, the spectral radius $\rho(\mathbf{P}_k(\alpha))$ is strictly less than 1 if $\alpha < \frac{\lambda_1-\lambda_2}{42}(\frac{1-\beta}{9\lambda_1})^2$.

Proof. Since $\mathbf{P}_k(\alpha)$ is a non-negative matrix, by Perron-Frobenius theorem its characteristic polynomial has a simple positive real root r such that $\rho(\mathbf{P}_k(\alpha)) = r$. We know $\delta_k = \frac{1+\alpha\lambda_{k+1}}{1+\alpha\lambda_k} < 1$ and $L_k = (k+5)\lambda_1$. Now, the characteristic polynomial $p(\gamma)$ of $\mathbf{P}(\alpha)$ is given as

$$p(\gamma) = |\gamma\mathbf{I} - \mathbf{P}(\alpha)|$$

$$= \begin{vmatrix} \gamma - (\frac{1+\beta}{2} + \alpha L_k) & -L_k(\alpha L_k + 2) & -\alpha L_k^2 \\ -\alpha & \gamma - \frac{1+\beta}{2} & 0 \\ 0 & -\alpha L_k & \gamma - \delta_1 \end{vmatrix}$$

$$= (\gamma - \frac{1+\beta}{2} - \alpha L_k)(\gamma - \frac{1+\beta}{2})(\gamma - \delta_k) +$$

$$\alpha(-L_k(\alpha L_k + 2)(\gamma - \delta_k) - \alpha^2 L_k^3)$$

$$= ((\gamma - \frac{1+\beta}{2} - \alpha L_k)(\gamma - \frac{1+\beta}{2}) -$$

$$\alpha L_k(\alpha L_k + 2))(\gamma - \delta_k) - \alpha^3 L_k^3$$

$$= p_0(\gamma)(\gamma - \delta_k) - \alpha^3 L_k^3,$$

where

$$p_0(\gamma) = (\gamma - \frac{1+\beta}{2} - \alpha L_k)(\gamma - \frac{1+\beta}{2}) - \alpha L_k(\alpha L_k + 2)$$

$$= \gamma^2 - (1 + \beta + \alpha L_k)\gamma + \frac{1+\beta}{2}(\frac{1+\beta}{2} + \alpha L_k) - \alpha L_k(\alpha L_k + 2)$$

$$= (\gamma - \gamma_1)(\gamma - \gamma_2),$$

with γ_1, γ_2 being the roots of $p_0(\gamma)$, given as

$$\gamma_1, \gamma_2 = \frac{1 + \beta + \alpha L_k \pm \sqrt{5\alpha^2 L_k^2 + 8\alpha L_k}}{2}. \quad (59)$$

If $0 < \alpha < \frac{1}{L_k}$, $\alpha L_k < 1$ and it implies $\alpha^2 L_k^2 < \alpha L_k < \sqrt{\alpha L_k}$. Thus,

$$
\begin{aligned}
\gamma_1, \gamma_2 &= \frac{1 + \beta + \alpha L_k \pm \sqrt{5\alpha^2 L_k^2 + 8\alpha L_k}}{2} \\
&< \frac{1 + \beta + \alpha L_k + \sqrt{5\alpha^2 L_k^2 + 8\alpha L_k}}{2} \\
&< \frac{1 + \beta + \sqrt{\alpha L_k} + \sqrt{5\alpha L_k + 8\alpha L_k}}{2} \\
&< \frac{1 + \beta + \sqrt{\alpha L_k} + \sqrt{16\alpha L_k}}{2} \\
&= \frac{1 + \beta + 5\sqrt{\alpha L_k}}{2} = \gamma_0.
\end{aligned}
$$

For $\gamma \geq \gamma_0$, $p_0(\gamma) \geq (\gamma - \gamma_0)^2$. Now, let $\gamma^* = \max\{1 - \frac{1}{2}\frac{\alpha(\lambda_k - \lambda_{k+1})}{1 + \alpha\lambda_k}, \frac{1+\beta}{2} + 4.5\sqrt{\alpha L_1}\sqrt{\frac{(1+\alpha\lambda_k)L_1}{\lambda_k - \lambda_{k+1}}}\} > \gamma_0$. Then

$$
\begin{aligned}
p(\gamma^*) &\geq \frac{1}{2}\frac{\alpha(\lambda_k - \lambda_{k+1})}{1 + \alpha\lambda_k}\Big(4.5\sqrt{\alpha L_k}\sqrt{\frac{(1+\alpha\lambda_k)L_k}{\lambda_k - \lambda_{k+1}}} \\
&\quad - 2.5\sqrt{\alpha L_k}\Big)^2 - \alpha^3 L_k^3 \\
&\geq \frac{1}{2}\frac{\alpha(\lambda_k - \lambda_{k+1})}{1 + \alpha\lambda_{k+1}}\Big(4.5\sqrt{\alpha L_k}\sqrt{\frac{(1+\alpha\lambda_k)L_k}{\lambda_k - \lambda_{k+1}}} \\
&\quad - 2.5\sqrt{\alpha L_k}\sqrt{\frac{(1+\alpha\lambda_k)L_k}{\lambda_k - \lambda_{k+1}}}\Big)^2 - \alpha^3 L_k^3 \\
&\geq \frac{1}{2}\frac{\alpha(\lambda_k - \lambda_{k+1})}{1 + \alpha\lambda_k}\Big(2\sqrt{\alpha L_k}\sqrt{\frac{(1+\alpha\lambda_k)L_k}{\lambda_k - \lambda_{k+1}}}\Big)^2 - \alpha^3 L_k^3 \\
&= 2\alpha^2 L_k^2 - \alpha^3 L_k^3 \geq 0.
\end{aligned}
$$

Since $p(\gamma) = p_0(\gamma)(\gamma - \delta_k) - \alpha^3 L_k^3 \geq (\gamma - \gamma_0)^2(\gamma - \delta_k) - \alpha^3 L_k^3$, evidently it is a strictly increasing function on $[\max\{\delta_k, \gamma_0\}, +\infty)$ and since this interval includes γ^*, $p(\gamma)$ has no real roots on $(\gamma^*, +\infty)$. Thus, the real root of the characteristic polynomial is $\leq \gamma^*$. Hence $\rho(\mathbf{P}_k(\alpha)) \leq \gamma^*$. If we choose α such that $\gamma^* < 1$, then $\rho(\mathbf{P}_k(\alpha)) < 1$ and the convergence would be linear. For $\gamma^* < 1$, we need

$$\frac{1+\beta}{2} + 4.5\sqrt{\alpha L_k}\sqrt{\frac{(1+\alpha\lambda_k)L_k}{\lambda_k - \lambda_{k+1}}} < 1$$

$$\sqrt{\alpha L_k}\sqrt{\frac{(1+\alpha\lambda_k)L_k}{\lambda_k - \lambda_{k+1}}} < \frac{1-\beta}{9}$$

$$(\alpha L_k)\frac{(1+\alpha\lambda_k)L_k}{\lambda_k - \lambda_{k+1}} < (\frac{1-\beta}{9})^2$$

$$\alpha(1 + \alpha\lambda_k) < \frac{\lambda_k - \lambda_{k+1}}{L_k^2}(\frac{1-\beta}{9})^2$$

$$= \frac{\lambda_k - \lambda_{k+1}}{\lambda_k^2(k+5)^2}(\frac{1-\beta}{9})^2.$$

If $\alpha < \frac{1}{L_k} = \frac{1}{(k+5)\lambda_1} \leq \frac{1}{(k+5)\lambda_k}$, then $\alpha\lambda_k < \frac{1}{k+5}$. Thus, $1 + \alpha\lambda_k < \frac{k+6}{k+5}$. If $\alpha < \frac{\lambda_k - \lambda_{k+1}}{(k+5)(k+6)}(\frac{1-\beta}{9\lambda_1})^2 < \frac{1}{L_k} = \frac{1}{(k+5)\lambda_1}$, that is $\frac{k+6}{k+5}\alpha < \frac{\lambda_k - \lambda_{k+1}}{\lambda_1^2(k+5)^2}(\frac{1-\beta}{9})^2$, then $\alpha(1 + \alpha\lambda_k) < \frac{k+6}{k+5}\alpha < \frac{\lambda_k - \lambda_{k+1}}{\lambda_1^2(k+5)^2}(\frac{1-\beta}{9})^2$. ∎

APPENDIX F
PROOF OF LEMMA 3

For a continuous function $f : \mathbb{R}^d \to \mathbb{R}^d$, we know $\|f(x) - f(y)\| \leq \|\nabla f(x)\|\|x - y\|$. Thus, the Lipschitz constant can be given by the upper bound of $\|\nabla f(x)\|$. Taking derivative on both sides of the function

$$\mathbf{h}_{i,t}(\mathbf{v}) = \mathbf{C}_i\mathbf{v} - \frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{v} - \sum_{p=1}^{k-1}\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\mathbf{C}_i\mathbf{v}$$

we get

$$
\begin{aligned}
\frac{\partial}{\partial\mathbf{v}}\mathbf{h}_{i,t}(\mathbf{v}) &= \mathbf{C}_i - \frac{\partial}{\partial\mathbf{v}}\Big(\frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\Big)\mathbf{v}^T - \frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{I} \\
&\quad - \Big(\sum_{p=1}^{k-1}\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\mathbf{C}_i\Big)^T \\
&= \mathbf{C}_i - \frac{2\|\mathbf{v}\|^2\mathbf{C}_i\mathbf{v} - 2(\mathbf{v})^T\mathbf{C}_i\mathbf{v}\mathbf{v}}{\|\mathbf{v}\|^4}\mathbf{v}^T - \frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{I} \\
&\quad - \Big(\sum_{p=1}^{k-1}\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\mathbf{C}_i\Big)^T \\
&= \mathbf{C}_i - \frac{2\mathbf{C}_i\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^2} + \frac{2(\mathbf{v})^T\mathbf{C}_i\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^4} - \frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{I} \\
&\quad - \sum_{p=1}^{k-1}\mathbf{C}_i\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2}
\end{aligned}
$$

$$
\begin{aligned}
\|\frac{\partial}{\partial\mathbf{v}}\mathbf{h}_{i,t}(\mathbf{v})\| &\leq \|\mathbf{C}_i\| + \|\frac{2\mathbf{C}_i\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^2}\| + \|\frac{2(\mathbf{v})^T\mathbf{C}_i\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|^4}\| + \\
&\quad \|\frac{(\mathbf{v})^T\mathbf{C}_i\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{I}\| + \sum_{p=1}^{k-1}\|\mathbf{C}_i\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\| \\
&\leq \lambda_{i,1} + 2\frac{\lambda_{i,1}\|\mathbf{v}\|^2}{\|\mathbf{v}\|^2} + 2|(\mathbf{v})^T\mathbf{C}_i\mathbf{v}|\frac{\|\mathbf{v}\|^2}{\|\mathbf{v}\|^4} + \\
&\quad \frac{|(\mathbf{v})^T\mathbf{C}_i\mathbf{v}|}{\|\mathbf{v}\|^2} + \sum_{p=1}^{k-1}\|\mathbf{C}_i\|\|\frac{\mathbf{x}_{i,p}^{(t)}(\mathbf{x}_{i,p}^{(t)})^T}{\|\mathbf{x}_{i,p}^{(t)}\|^2}\| \\
&\leq \lambda_{i,1} + 2\lambda_{i,1} + 2\lambda_{i,1} + \lambda_{i,1} + \lambda_{i,1}(k-1) \\
&= (k+5)\lambda_{i,1}
\end{aligned}
$$

Thus,

$$
\begin{aligned}
\|\mathbf{h}_{i,t}(\mathbf{v}_1) - \mathbf{h}_{i,t}(\mathbf{v}_2)\| &\leq \lambda_{i,1}(k+5)\|\mathbf{v}_1 - \mathbf{v}_2\| \\
&\leq \lambda_1(k+5)\|\mathbf{v}_1 - \mathbf{v}_2\|
\end{aligned}
$$

∎

REFERENCES

[1] A. Gang, H. Raja, and W. U. Bajwa, "Fast and communication-efficient distributed PCA," in *Proc. 2019 IEEE International Conf. Acoustics, Speech and Signal Process. (ICASSP)*, 2019, pp. 7450–7454.

[2] A. Gang and W. U. Bajwa, "The best of both worlds: Distributed PCA that is both exact and communication efficient," in *Proc. 2022 30th European Signal Process. Conf. (EUSIPCO)*, 2022, pp. 732–736.

[3] M. Marjani, F. Nasaruddin, A. Gani, A. Karim, I. A. T. Hashem, A. Siddiqa, and I. Yaqoob, "Big IoT data analytics: Architecture, opportunities, and open research challenges," *IEEE Access*, vol. 5, pp. 5247–5261, 2017.

[4] M. Hubert, P. J. Rousseeuw, and S. Verboven, "A fast method for robust principal components with applications to chemometrics," *Chemometrics Intell. Laboratory Syst.*, vol. 60, no. 1, pp. 101–111, 2002.

[5] T. Bouwmans, S. Javed, H. Zhang, Z. Lin, and R. Otazo, "On the applications of robust PCA in image and video processing," *Proc. IEEE*, vol. 106, no. 8, pp. 1427–1457, 2018.

[6] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *J. Educational Psychology*, vol. 24, no. 6, pp. 417–441, 1933.

[7] W. U. Bajwa, V. Cevher, D. Papailiopoulos, and A. Scaglione, "Machine learning from distributed, streaming data," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 11–13, May 2020.

[8] Z. Yang, A. Gang, and W. U. Bajwa, "Adversary-resilient distributed and decentralized statistical inference and machine learning: An overview of recent advances under the Byzantine threat model," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 146–159, 2020.

[9] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798—1828, August 2013.

[10] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural Netw.*, vol. 2, no. 1, p. 53–58, Jan. 1989.

[11] E. Oja and J. Karhunen, "On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix," *J. Math. Anal. Applicat.*, vol. 106, no. 1, pp. 69 – 84, 1985.

[12] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Netw.*, vol. 2, no. 6, pp. 459 – 473, 1989.

[13] D. O. Hebb, *The Organization of Behavior : A Neuropsychological Theory*. Wiley New York, 1949.

[14] T. P. Krasulina, "Method of stochastic approximation in the determination of the largest eigenvalue of the mathematical expectation of random matrices," *Autom. Remote Control*, vol. 1970, pp. 215–221, 1970.

[15] A. Balsubramani, S. Dasgupta, and Y. Freund, "The fast convergence of incremental PCA," in *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013.

[16] C. Tang, "Exponentially convergent stochastic k-PCA without variance reduction," in *NeurIPS*, 2019.

[17] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Mag.*, vol. 2, pp. 559–572, 1901.

[18] G. H. Golub and C. F. Van Loan, *Matrix Computations (3rd Ed.)*. Baltimore, MD, USA: Johns Hopkins University Press, 1996.

[19] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Research Nat. Bureau Standards*, 1950.

[20] Z. Yi, M. Ye, J. C. Lv, and K. K. Tan, "Convergence analysis of a deterministic discrete time system of Oja's PCA learning algorithm," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1318–1328, Nov 2005.

[21] J. C. Lv, Z. Yi, and K. K. Tan, "Global convergence of GHA learning algorithm with nonzero-approaching adaptive learning rates," *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1557–1571, 2007.

[22] N. Vaswani, Y. Chi, and T. Bouwmans, "Rethinking PCA for modern data sets: Theory, algorithms, and applications," *Proc. IEEE*, vol. 106, no. 8, pp. 1274–1276, 2018.

[23] N. Vaswani, T. Bouwmans, S. Javed, and P. Narayanamurthy, "Robust subspace learning: Robust PCA, robust subspace tracking and robust subspace recovery," *IEEE Signal Process. Mag.*, vol. 35, pp. 32–55, 07 2018.

[24] S. Hauberg, A. Feragen, R. Enficiaud, and M. J. Black, "Scalable robust principal component analysis using Grassmann averages," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 11, pp. 2298–2311, 2016.

[25] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *J. Computational and Graphical Stat.*, vol. 15, no. 2, pp. 265–286, 2006.

[26] H. Zou and L. Xue, "A selective overview of sparse principal component analysis," *Proc. IEEE*, vol. 106, no. 8, pp. 1311–1320, 2018.

[27] Z.-J. Bai, R. H. Chan, and F. T. Luk, "Principal component analysis for distributed data sets with updating," in *Advanced Parallel Process. Technologies*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 471–483.

[28] N. An and S. Weber, "On the performance overhead tradeoff of distributed principal component analysis via data partitioning," in *Proc. Annu. Conf. Inform. Sci. Syst. (CISS)*, 2016, pp. 578–583.

[29] M. A. Livani and M. Abadi, "Distributed PCA-based anomaly detection in wireless sensor networks," in *Proc. Int. Conf. Internet Technology Secured Transactions*, 2010, pp. 1–8.

[30] H. Imtiaz and A. D. Sarwate, "Distributed differentially private algorithms for matrix and tensor factorization," *IEEE J. Select. Topics Signal Process.*, vol. 12, no. 6, pp. 1449–1464, 2018.

[31] D. A. Tarzanagh, M. K. S. Faradonbeh, and G. Michailidis, "Online distributed estimation of principal eigenspaces," in *Proc. 2019 IEEE Data Sci. Workshop (DSW)*, 2019, pp. 27–31.

[32] B. Xiao, Y. Li, B. Sun, C. Yang, K. Huang, and H. Zhu, "Decentralized PCA modeling based on relevance and redundancy variable selection and its application to large-scale dynamic process monitoring," *Process Safety and Environmental Protection*, vol. 151, pp. 85–100, 2021.

[33] A. Grammenos, R. Mendoza Smith, J. Crowcroft, and C. Mascolo, "Federated principal component analysis," in *Proc. Adv. Neural Inform. Process. Syst.*, vol. 33. Curran Associates, Inc., 2020, pp. 6453–6464.

[34] S. X. Wu, H.-T. Wai, L. Li, and A. Scaglione, "A review of distributed algorithms for principal component analysis," *Proc. IEEE*, vol. 106, no. 8, pp. 1321–1340, 2018.

[35] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," *J. Comput. and Syst. Sci.*, vol. 74, no. 1, pp. 70 – 83, 2008.

[36] A. Scaglione, R. Pagliari, and H. Krim, "The decentralized estimation of the sample covariance," in *Proc. 2008 42nd Asilomar Conf. on Signals, Syst. and Comput.*, 2008, pp. 1722–1726.

[37] L. Li, A. Scaglione, and J. H. Manton, "Distributed principal subspace estimation in wireless sensor networks," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 725–738, Aug 2011.

[38] H. Raja and W. U. Bajwa, "Cloud K-SVD: Computing data-adaptive representations in the cloud," in *Proc. 2013 51st Annual Allerton Conf. Commun., Control and Computing (Allerton)*, 2013, pp. 1474–1481.

[39] H. Raja and W. U. Bajwa, "Cloud-K-SVD: A collaborative dictionary learning algorithm for big, distributed data," *IEEE Trans. Signal Process.*, vol. 64, no. 1, pp. 173–188, Jan 2016.

[40] H. Wai, A. Scaglione, J. Lafond, and E. Moulines, "Fast and privacy preserving distributed low-rank regression," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Process., (ICASSP)*, 2017, pp. 4451–4455.

[41] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[42] A. Gang, B. Xiang, and W. U. Bajwa, "Distributed principal subspace analysis for partitioned big data: Algorithms, analysis, and implementation," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 7, pp. 699–715, 2021.

[43] M. Hong, D. Hajinezhad, and M.-M. Zhao, "Prox-PDA: The proximal primal-dual algorithm for fast distributed nonconvex optimization and learning over networks," in *Proc. 34th Int. Conf. Mach. Learning*, vol. 70. PMLR, 06–11 Aug 2017, pp. 1529–1538.

[44] P. Bianchi and J. Jakubowicz, "Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization," *IEEE Trans. Autom. Control*, vol. 58, no. 2, pp. 391–405, 2013.

[45] H. Wai, A. Scaglione, J. Lafond, and E. Moulines, "A projection-free decentralized algorithm for non-convex optimization," in *Proc. 2016 IEEE Global Conf. Signal and Inform. Process. (GlobalSIP)*, 2016, pp. 475–479.

[46] S. Chen, A. Garcia, M. Hong, and S. Shahrampour, "Decentralized riemannian gradient descent on the Stiefel manifold," *arXiv preprint arXiv:2102.07091*, 2021.

[47] F. L. Andrade, M. A. Figueiredo, and J. Xavier, "Distributed Picard iteration," *arXiv preprint arXiv:2104.00131*, 2021.

[48] ——, "Distributed Picard iteration: Application to distributed EM and distributed PCA," *arXiv preprint arXiv:2106.10665*, 2021.

[49] A. Gang and W. U. Bajwa, "A linearly convergent algorithm for distributed principal component analysis," *Signal Processing*, vol. 193, p. 108408, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016516842100445X

[50] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan 2009.

[51] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: an exact first-order algorithm for decentralized consensus optimization," *SIAM J. Optim.*, vol. 25, no. 2, pp. 944–966, 2015.

[52] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 3, pp. 1245–1260, 2018.

[53] P. D. Lorenzo and G. Scutari, "NEXT: In-network nonconvex optimization," *IEEE Trans. Signal Inform. Process. Netw.*, vol. 2, no. 2, pp. 120–136, 2016.

[54] H. Ye and T. Zhang, "DeEPCA: Decentralized exact PCA with linear convergence rate," *Journal of Machine Learning Research*, vol. 22, no. 238, pp. 1–27, 2021. [Online]. Available: http://jmlr.org/papers/v22/21-0298.html

[55] R. Arora, A. Cotter, and N. Srebro, "Stochastic optimization of PCA with capped MSG," in *Advances Neural Inform. Process. Systs.*, 2013, pp. 1815–1823.

[56] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing Markov chain on a graph," *SIAM REVIEW*, vol. 46, pp. 667–689, 2003.

[57] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database," *ATT Labs*, vol. 2, 2010.

[58] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.