Communication-optimal Distributed Principal Component Analysis in the Column-partition Model

Christos Boutsidis
Yahoo Labs
New York, New York
boutsidis@yahoo-inc.com

David P. Woodruff
IBM Research
Almaden, California
dpwoodru@us.ibm.com

Abstract

We study the Principal Component Analysis (PCA) problem in the distributed and streaming models of computation. Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, a rank parameter $k < \text{rank}(\mathbf{A})$, and an accuracy parameter $0 < \varepsilon < 1$, we want to output an $m \times k$ orthonormal matrix $\mathbf{U}$ for which

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\cdot\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2,$$

where $\mathbf{A}_k \in \mathbb{R}^{m\times n}$ is the best rank-k approximation to $\mathbf{A}$.

In the distributed model, each of s machines holds a subset of columns of $\mathbf{A}$ and should output $\mathbf{U}$. Kannan et al. (COLT 2014) achieve an $O(skm/\varepsilon) + \text{poly}(sk/\varepsilon)$ words (of $O(\log(nm))$ bits) communication protocol. We obtain the improved bound of $O(skm) + \text{poly}(sk/\varepsilon)$ words, and show an optimal $\Omega(skm)$ word communication lower bound. Removing the dependence on ε in the leading order term is important for high-precision PCA, and resolves Open Question 3 of Woodruff (NOW, 2014) in the column partition model. Moreover, while previous work could not exploit the *sparsity* of $\mathbf{A}$, we also give an $O(sk\phi/\varepsilon)+\text{poly}(sk/\varepsilon)$ word communication protocol when each column of $\mathbf{A}$ is ϕ-sparse, bypassing the above lower bound whenever $\phi = o(\varepsilon m)$. The communication cost of this protocol is independent of the matrix dimensions. This latter protocol achieves the stronger guarantee that each server, in addition to outputting $\mathbf{U}$, outputs a subset of $O(k/\varepsilon)$ columns of $\mathbf{A}$ containing such a matrix $\mathbf{U}$ in its span (a.k.a. we solve the distributed column subset selection problem). We show a matching $\Omega(sk\phi/\varepsilon)$ word communication lower bound for the distributed column subset selection problem.

In the streaming model, in which the columns arrive one at a time, an algorithm of Liberty (KDD, 2013) with an improved analysis by Ghashami and Phillips (SODA, 2014) shows that $O(km/\varepsilon)$ "real numbers" of space is achievable in a single pass, which we first improve to an $O(km/\varepsilon)+\text{poly}(k/\varepsilon)$ word space upper bound. There is an almost matching $\Omega(km/\varepsilon)$ bit lower bound shown by Woodruff (NIPS 2014). Surprisingly, we show that with two passes one can achieve $O(km) + \text{poly}(k/\varepsilon)$ words of space and (up to the $\text{poly}(k/\varepsilon)$ term and the distinction between words versus bits) this is also optimal for any constant number of passes.

1 Introduction

In distributed-memory computing systems, such as, Hadoop [1] or Spark [3], Principal Component Analysis (PCA) and the related Singular Value Decomposition (SVD) of large matrices is becoming very challenging. Machine learning libraries implemented on top of such systems, for example mahout [2] or mllib [4], provide distributed PCA implementations since PCA is often used as a building block for a learning algorithm. PCA is useful for dimension reduction, noise removal, visualization, etc. In all of these implementations, the bottleneck is the communication; hence, the focus has been on minimizing the communication cost of the related algorithms, and not the computational cost, which is the bottleneck in more traditional batch systems.

The data matrix corresponding to the dataset in hand, e.g., a term-document matrix representing a text collection, or the Netflix matrix representing user's ratings for different movies, could be distributed in many different ways [41]. In this paper, we focus on the so-called column partition model, where an $m \times n$ matrix $\mathbf{A}$ is distributed column-wise among $s < n$ machines:

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s \end{pmatrix};$$

here, for $i = 1 : s$, $\mathbf{A}_i$ is an $m \times w_i$ column submatrix of $\mathbf{A}$ with $\sum_i w_i = n$. This model has been adapted by traditional numerical linear algebra software libraries for distributed memory matrix computations [11], and it is similar in spirit to MapReduce, a programming model for distributed memory machine learning tasks. One way to compute a PCA of such a column-wise distributed matrix is to transfer all the entries of the matrix into a single machine and then perform a standard PCA: (i) for given rank parameter $k < \text{rank}(\mathbf{A})$, compute the matrix $\mathbf{U}_k \in \mathbb{R}^{m \times k}$ with the top k left singular vectors of $\mathbf{A}$; and then, (ii) compute $\mathbf{B} = \mathbf{U}_k^{\mathrm{T}} \mathbf{A}$, which is the so-called low-dimensional PCA projection of $\mathbf{A}$. Alternatively, in the first step, one can use the recent "input-sparsity-time" algorithms [19, 40, 39, 14] and compute an (approximate) PCA in time proportional to $\text{nnz}(\mathbf{A})$. Both of these approaches, however, are too costly from a communication standpoint.

A better approach is to design an algorithm where, first, some computations happen locally in each machine, and then, a central coordinator (server) receives a small amount of information from each machine and computes an (approximate) $m \times k$ orthonormal matrix $\mathbf{U}$, which is subsequently communicated back to each machine. Algorithms along these lines were recently developed in [24, 8, 37, 26, 34, 12] (see Section 2 for a detailed discussion of prior work). All of these algorithms have communication cost which depends on at least one of the dimensions of $\mathbf{A}$; and the reason is that in these algorithms each machine sends to the server some sort of singular vectors of its "local" matrix, which are in general dense high-dimensional vectors, even for a sparse matrix $\mathbf{A}$. Though this is maybe unavoidable for dense matrices, it is not clear if it is necessary for sparse matrices. We note that many matrices are sparse, e.g., Netflix provides a training data set of $100,480,507$ ratings that $480,189$ users give to $17,770$ movies. This means that if users correspond to columns in the matrix, then the average column sparsity is ≈ 200 non-zero elements (out of the $17,770$ possible coordinates). Despite this progress, there are two important questions which remain unanswered:

1. Is the communication cost of existing distributed PCA algorithms necessary?
2. Do there exist better algorithms for sparse matrices?

In this work, we address both questions. In short, we provide:

1. A communication-optimal PCA algorithm for dense matrices along with a matching lower bound (this algorithm improves upon all existing protocols [24, 8, 37, 26, 34, 12]).
2. A PCA algorithm for sparse matrices with dimension-indepedent communication cost.

	Upper bounds	Lower bounds
$\phi = \Omega(\varepsilon \cdot m)$	$O(skm + s \cdot \text{poly}(k/\varepsilon))$ (Theorem 32)	$\Omega(skm)$ (Theorem 66)
$\phi = o(\varepsilon \cdot m)$	$O(sk\phi\varepsilon^{-1} + s \cdot \text{poly}(k/\varepsilon))$ (Theorem 46)	$\Omega(sk\phi)$ (Corollary 67)

Table 1: Communication upper/lower bounds for the Distributed PCA problem of Definition 2.

Before presenting our contributions in detail, we formally define the distributed computing model we consider in this paper and the PCA problem that we would like to solve.

Definition 1 (Column-partition model)**.** *An $m \times n$ matrix $\mathbf{A}$ is partitioned arbitrarily column-wise into s blocks $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$, i.e., for $i = 1, 2, \ldots, s$:*

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \ldots & \mathbf{A}_s \end{pmatrix}.$$

Here, $\sum w_i = n$. There are s machines, and the i-th machine has $\mathbf{A}_i$ as input. There is also another machine, to which we refer to as the "server", which acts as the central coordinator. The model only allows communication between the machines and the server. The communication cost of an algorithm in this model is defined as the total number of words transferred between the machines and the server, where we assume each word is $O(\log(nms/\varepsilon))$ bits.

Notice that in this model the machines can still communicate with each other through the server, and this increases the communication cost only by a factor of 2, plus an additional word per message to identify who to route the message to.

Definition 2 (The Distributed Principal Component Analysis Problem)**.** *Given an $m \times n$ matrix $\mathbf{A}$ partitioned column-wise into s arbitrary blocks $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$ ($i : 1, 2, \ldots, s$): $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \ldots & \mathbf{A}_s \end{pmatrix}$, a rank parameter $k < \text{rank}(\mathbf{A})$, and an accuracy parameter $0 < \varepsilon < 1$, design an algorithm in the model of Definition 1 which, upon termination, leaves on each machine a matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$ with k orthonormal columns such that*

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1 + \varepsilon) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2,$$

and the communication cost of the algorithm is as small as possible.

Notice that a solution to this problem leaves on each machine the same orthonormal matrix $\mathbf{U}$. Hence, each machine can use it locally to compute the PCA projection of its points (columns in $\mathbf{A}$). Also, notice that the statement of the problem does not discuss the running time of the algorithm, but we are interested in the fastest possible algorithms with the minimal communication cost.

1.1 Contributions

To present our contributions, we introduce a parameter to "model" the sparsity in a matrix $\mathbf{A}$. Let us denote by ϕ the maximum number of non-zero elements of a column in $\mathbf{A}$. When we say that $\mathbf{A}$ is ϕ-sparse, we mean that every column of $\mathbf{A}$ has at most ϕ non-zero elements and $\text{nnz}(\mathbf{A}) \leq \phi \cdot n$.

We present novel algorithms and communication lower bounds for two regimes of ϕ. The first regime $\phi = \Omega(\varepsilon \cdot m)$, corresponds to "dense" matrices, while $\phi = o(\varepsilon \cdot m)$ corresponds to "sparse" matrices. We summarize the novel lower/upper communication bounds proven in this paper in Table 2.

1.1.1 Distributed PCA on dense matrices; $\phi = \Omega(\varepsilon \cdot m)$

Upper bound. Previous protocols for distributed PCA in the column partition model [24, 8, 37, 26, 34, 12] have communication cost at least $\Omega\left(skm\varepsilon^{-1}\right)$ words plus low order terms. We note that these protocols work in very different ways; that of [24, 8] is based on coresets, while that of [37, 26] is based on adapting a streaming algorithm to the communication setting, while that of [34] is based on sketching, and that of [12] is based on alternating minimization and it is useful only under some assumptions on the condition number of the matrix (see Section 2 for a more detailed discussion of previous distributed PCA protocols).

We provide an algorithm with communication cost $O\left(skm\right)$ words plus low order terms (see Theorem 32). Thus, our bound breaks the barrier of having a dependence on ε^{-1} in the leading term of the communication cost, which is particularly important for applications with machine precision error requirements. This resolves Open Question 3 of Woodruff [48] for the column partition model. Further, our protocol has no assumption on the condition number and works for arbitrary matrices. We now give an overview of the protocol.

Recall that the input $m \times n$ matrix $\mathbf{A}$ is partitioned arbitrarily column-wise into s blocks $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$, i.e., for $i = 1, 2, \ldots, s$: $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \ldots & \mathbf{A}_s \end{pmatrix}$. Each machine starts by computing $\mathbf{R}_i = \mathbf{W}\mathbf{A}_i$, where $\mathbf{W}$ is a random matrix with $O(k/\varepsilon)$ rows. Let $\mathbf{R} = \sum_i \mathbf{R}_i$ be also a matrix with $O(k/\varepsilon)$ rows. Hence, the matrix $\mathbf{R}_i$ is a "sketch" of $\mathbf{A}_i$ and, correspondingly, the matrix $\mathbf{R}$ is a "sketch" of $\mathbf{A}$. We then prove (see Lemma 21):

$$\min_{\mathrm{rank}(\mathbf{X}) \leq k} \|\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R} - \mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

It is thus natural to project the rows of $\mathbf{A}$ onto the row space of $\mathbf{R}$ in a distributed way. We cannot afford for every machine to learn $\mathbf{R}$ though, as this requires $\Omega(skm/\varepsilon)$ communication. Instead, in the column partition model, to compute $\mathbf{A}_i\mathbf{R}^{\mathrm{T}} = \mathbf{A}_i\mathbf{A}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}$, it is enough for the i-th machine to compute $\mathbf{A}_i\mathbf{A}_i^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}$, as the sum over i coincides with $\mathbf{A}\mathbf{A}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}$. Of course, we cannot communicate each of the $\mathbf{A}_i\mathbf{A}_i^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}$, as this would be $\Omega(skm/\varepsilon)$ words, but we can choose additional "sketching matrices" $\mathbf{T}_{left}$ and $\mathbf{T}_{right}$ to instead solve

$$\mathbf{X}_* = \operatorname*{argmin}_{\mathrm{rank}(\mathbf{X})=k} \|\mathbf{T}_{left}\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\mathbf{T}_{right} - \mathbf{T}_{left}\mathbf{A}\mathbf{T}_{right}\|_{\mathrm{F}}^2,$$

which is a generalization of the subspace embedding technique to "affine embeddings" [48]. Here, $\mathbf{T}_{left}$ has $O(k/\varepsilon^3)$ rows and $\mathbf{T}_{right}$ has $O(k/\varepsilon^3)$ columns. This problem can then be solved in the distributed model with low communication, from which an $O(k/\varepsilon) \times k$ orthonormal basis $\mathbf{U}_{\mathbf{X}_*}$ for the column span of the matrix $\mathbf{X}_*$ can be found by each server. We now use that the matrices $\mathbf{A}_i\mathbf{R}\mathbf{U}_{\mathbf{X}_*}$ are only $m \times k$ matrices, and can be efficiently communicated and combined to find an orthonormal basis $\mathbf{U}$ for the column span of $\mathbf{A}\mathbf{R}\mathbf{X}$. Section 4.3 presents in detail the ideas above.

A technical obstacle in the analysis above is that it may require a large number of machine words to specify the entries of $\mathbf{U}_{\mathbf{X}_*}$, even if each entry in $\mathbf{A}$ is only a single word. In fact, one can show (we omit the details) that the singular values of $\mathbf{X}_*$ can be exponentially large in k/ε, which means one would need to round the entries of $\mathbf{U}_{\mathbf{X}_*}$ to an additive exponentially small precision, which would translate to an undesirable $sm \cdot \mathrm{poly}(k/\epsilon)$ bits of communication.

To counter this, we use the *smoothed analysis* of Tao and Vu [47] to argue that if we add small random Bernoulli noise to $\mathbf{A}$, then its minimum singular value becomes inverse polynomial in n. Moreover, if the rank of $\mathbf{A}$ is at least $2k$ and under the assumption that the entries of $\mathbf{A}$ are reprsentable with a single machine word (so we can identify them with integers in magnitude at most $\mathrm{poly}(nms/\epsilon)$ by scaling), then we can show the additional noise preserves relative error

approximation. To our knowledge, this is the first application of this smoothing technique to distributed linear algebra algorithms. Section 5 discusses the details of this approach.

On the other hand, if the rank of $\mathbf{A}$ is smaller than $2k$, the smoothing technique need not preserve relative error. In this case though, we can learn a basis $\mathbf{C} \in \mathbb{R}^{m \times O(k)}$ for the column span of $\mathbf{A}$ by multiplying by a pseudorandom matrix based on Vandermonde matrices. Since $\mathbf{C}$ has at most $2k$ columns, we can efficiently communicate it to all servers using $O(skm)$ communication. At this point we set up the optimization problem $\min_{\text{rank}(\mathbf{X}) \leq k} \|\mathbf{CXC}^{\mathrm{T}}\mathbf{A} - \mathbf{A}\|_{\mathrm{F}}$. By sketching on the left and right by $\mathbf{T}_{left}$ and $\mathbf{T}_{right}$, the server learns $\mathbf{T}_{left}\mathbf{CXC}^{\mathrm{T}}\mathbf{AT}_{right}$ and $\mathbf{T}_{left}\mathbf{AT}_{right}$, which are small matrices, and can be sent to each machine. Each machine locally solves for the rank-k solution $\mathbf{X}_*$ minimizing $\|\mathbf{T}_{left}\mathbf{CXC}^{\mathrm{T}}\mathbf{AT}_{right} - \mathbf{T}_{left}\mathbf{AT}_{right}\|_{\mathrm{F}}$. Now each machine can find the same $m \times k$ orthonormal basis and output it without further communication.

Lower bound. To the best of our knowledge, there is no known communication lower bound in the literature [1]. We prove an optimal $\Omega(skm)$ word communication lower bound. Theorem 66 argues that there exists an $m \times n$ matrix $\mathbf{A}$ such that for this $\mathbf{A}$, any $k \leq 0.99m$, and any error parameter C with $1 < C < \text{poly}(skm)$, if there exists a protocol to construct an $m \times k$ matrix $\mathbf{U}$ satisfying $\|\mathbf{A} - \mathbf{UU}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq C \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$, with constant probability, then this protocol has communication cost at least $\Omega(skm)$ words. This lower bound is proven for a matrix $\mathbf{A}$ that has k fully dense columns. We should note that the lower bound holds even if each machine only outputs the projection of $\mathbf{A}_i$ onto $\mathbf{U}$, i.e., $\mathbf{UU}^{\mathrm{T}}\mathbf{A}_i$, rather than $\mathbf{U}$ itself. Note that this problem is potentially easier, since given $\mathbf{U}$, a machine could find $\mathbf{UU}^{\mathrm{T}}\mathbf{A}_i$.

The intuition of the proof is that the first machine has a random $m \times k$ matrix $\mathbf{A}_1$ with orthonormal columns (rounded to integer multiples of $1/\text{poly}(n)$), while all other machines have a very small multiple of the identity matrix. When concatenating the columns of these matrices, the best k-dimensional subspace to project the columns onto must be very close to $\mathbf{A}_1$ in spectral norm. Since machine i, $i > 1$, has a tiny multiple of the identity matrix, after projection he/she obtains a projection matrix very close to the projection onto the column span of $\mathbf{A}_1$. By choosing a net of $m \times k$ orthonormal matrices, all of which pairwise have high distance in spectral norm, each machine can reconstruct a random element in this net, which requires a lot of information, and hence communication.

1.1.2 Distributed PCA on sparse matrices; $\phi = o(\varepsilon \cdot m)$

Upper bound. Section 7.2 presents a distributed PCA algorithm that has $O(\phi ks\varepsilon^{-1}) + \text{poly}(sk/\varepsilon)$ words of communication cost. Our idea in order to take advantage of the sparsity in the input matrix $\mathbf{A}$ is to select and transfer to the server machine a small set of "good" columns from each sub-matrix $\mathbf{A}_i$. Specifically, we design an algorithm that first computes, in a distributed way, a matrix $\tilde{\mathbf{C}} \in \mathbb{R}^{m \times c}$ with $c = O(k\varepsilon^{-1})$ columns of $\mathbf{A}$, and then finds $\mathbf{U} \in span(\tilde{\mathbf{C}})$ using a distributed, communication-efficient algorithm developed in [34]. The matrix $\tilde{\mathbf{C}}$ is constructed using optimal algorithms for column sampling [15, 21], extended appropriately to the distributed setting.

We would like to stress that our algorithm uses known column subset selection techniques whose behavior is very well understood in the batch setting [15, 21]. The contribution in this paper (see also next subsection) is a novel combination of those techniques in the distributed setting, and a novel analysis which indicates that a distributed approach for column subset selection can find columns which are as "good" as the columns one can find in the batch setting.

[1] There is a communication lower bound in [34] for a different (more general) distributed protocol where each machine has a matrix $\mathbf{A}_i$ such that $\mathbf{A} = \sum_i \mathbf{A}_i$. This lower bound does not apply to the problem we consider here.

	Upper bounds	Lower bounds
Definition 3	$O(sk\phi\varepsilon^{-1})$ (Theorem 58)	$\Omega(sk\phi\varepsilon^{-1})$ (Theorem 74)
Definition 4	$O\left(sk\phi\varepsilon^{-1} + s \cdot \mathrm{poly}(k/\varepsilon)\right)$ (Theorem 58)	$\Omega(sk\phi\varepsilon^{-1})$ (Corollary 75)

Table 2: Communication upper/lower bounds for the CSSP problems of Definitions 3 and 4.

Our $O(sk\phi/\varepsilon) + \mathrm{poly}(sk/\varepsilon)$ upper bound improves on our $O(skm) + \mathrm{poly}(sk/\varepsilon)$ upper bound once the sparsity is $\phi = o(\varepsilon m)$. A simple extension of this algorithm leads to a faster randomized algorithm with slightly larger communication cost (see Section 8 and Table 4).

Lower bound. Our lower bound of Theorem 66 implies a lower bound of $\Omega(sk\phi)$ words for matrices in which the column sparsity is ϕ (see Corollary 67). Unlike the dense case, where we resolve the communication complexity up to low order terms, in the sparse case there is a factor ε^{-1} gap between upper and lower bounds. It is an intriguing open question to resolve this gap.

1.1.3 The Distributed Column Subset Selection Problem

This paper provides the *first* distributed algorithms for column-based matrix reconstruction [15] [2]. These algorithms are implicit in the distributed PCA algorithms in Section 7.2 and Section 8.2. Before discussing these results in detail, we formally define the underlying optimization problems. In the first problem, upon termination of an algorithm, each machine has a "good" subset of columns of the input matrix, while in the second problem, on top of a subset of "good" columns from the input matrix, each machine has a matrix $\mathbf{U}$ with k orthonormal columns that is in the span of those columns. Notice that a solution to this second problem is also a solution to the PCA problem of Definition 2. Having a small set of columns to represent the data rather than a small set of arbitrary vectors is often useful since if the original matrix is sparse, the small set of columns will also be sparse. It also often leads to better data interpretability.

Definition 3 (The Distributed Column Subset Selection Problem)**.** *Given an $m \times n$ matrix $\mathbf{A}$ partitioned column-wise into s arbitrary blocks $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$ ($i : 1, 2, \ldots, s$): $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \ldots & \mathbf{A}_s \end{pmatrix}$, a rank parameter $k < \mathrm{rank}(\mathbf{A})$, and an accuracy parameter $0 < \varepsilon < 1$, design an algorithm in the model of Definition 1 that, upon termination, leaves on each machine a matrix $\mathbf{C} \in \mathbb{R}^{m \times c}$ with $c < n$ columns of $\mathbf{A}$ such that*

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2,$$

and

1. *The number of selected columns c is as small as possible.*
2. *The communication cost of the algorithm is as small as possible.*

[2] In a recent followup paper to our work, Balcan et al. [9] give a distributed protocol for *kernel* principal component analysis, which also needs to sample columns. The main issue in that paper is how to give such a protocol without a dependence on m or n, since in that paper *both* dimensions are very large. Since they have a kernel structure, it is possible to have communication smaller than both dimensions even for dense matrices.

Definition 4 (The Distributed Column Subset Selection Problem - rank k subspace version)**.** *Given an $m \times n$ matrix $\mathbf{A}$ partitioned column-wise into s arbitrary blocks $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$ $(i : 1, 2, \ldots, s)$: $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \ldots & \mathbf{A}_s \end{pmatrix}$, a rank parameter $k < \text{rank}(\mathbf{A})$, and an accuracy parameter $0 < \varepsilon < 1$, design an algorithm in the model of Definition 1 that, upon termination, leaves on each machine a matrix $\mathbf{C} \in \mathbb{R}^{m \times c}$ with $c < n$ columns of $\mathbf{A}$ and a matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$ with k orthonormal columns with $\mathbf{U} \in span(\mathbf{C})$, such that*

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2,$$

and

1. *The number of selected columns c is as small as possible.*
2. *The communication cost of the algorithm is as small as possible.*

Upper bounds. To the best of our knowledge, before our work, only heuristics for these two problems were known [23]. Here, we provide the first communication efficient algorithms for distributed column-based matrix reconstruction. More specifically, regarding the problem of Definition 3, one can use the Algorithm in Section 8.2 up to step 4-(a). Then, Eqn. 11 in Theorem 58 shows the desired approximation bound. The communication cost is $O(s\phi k/\varepsilon)$ words. Regarding the problem of Definition 4, one can use the Algorithm in Section 8.2. Then, Eqn. 12 in Theorem 58 shows the desired approximation bound. The communication is $O\left(sk\phi\varepsilon^{-1} + sk^3\varepsilon^{-5}\right)$ words.

Lower bounds. Table 2 summarizes the matching communication lower bounds that we obtain for distributed column-based matrix reconstruction. Theorem 74 proves a lower bound for the problem in Definition 3; then, a lower bound for the problem of Definition 4 follows immediately since this is a harder problem (see Corollalry 75).

To illustrate the ideas of the lower bound in Theorem 74, suppose $k = 1$. We start with the canonical hard matrix for column subset selection, namely, an $m \times m$ matrix $\mathbf{A}$ whose first row is all ones, and remaining rows are a subset of the identity matrix. Intuitively the best rank-1 approximation is very aligned with the first row of $\mathbf{A}$. However, given only $o(1/\varepsilon)$ columns of the matrix, there is not a vector in the span which puts a $(1-\varepsilon)$-fraction of its mass on the first coordinate, so $\Omega(1/\varepsilon)$ columns are needed. Such a matrix has $\phi = 2$.

Obtaining a lower bound for general ϕ involves creating a large net of such instances. Suppose we set $m = \phi$ and consider $\tilde{\mathbf{L}}\mathbf{A}$, where $\tilde{\mathbf{L}}$ is formed by choosing a random $\phi \times \phi$ orthonormal matrix, and then rounding each entry to the nearest integer multiple of $1/\text{poly}(n)$. We can show that $o(1/\varepsilon)$ columns of $\tilde{\mathbf{L}}\mathbf{A}$ do not span a rank-1 approximation to $\tilde{\mathbf{L}}\mathbf{A}$. We also need a lemma which shows that if we take two independently random orthonormal matrices, and round their entries to integer multiples of $1/\text{poly}(n)$, then these matrices are extremely unlikely to share a constant fraction of columns. The idea then is that if one server holds $\tilde{\mathbf{L}}\mathbf{A}$, and every other server holds the zero matrix, then every server needs to output the same subset of columns of $\tilde{\mathbf{L}}\mathbf{A}$. By construction of $\mathbf{A}$, each column of $\tilde{\mathbf{L}}\mathbf{A}$ is the sum of the first column of $\tilde{\mathbf{L}}$ and an arbitrary column of $\tilde{\mathbf{L}}$, and since we can assume all servers know the first column of $\tilde{\mathbf{L}}$ (which involves a negligible $O(s\phi)$ amount of communication), this implies that each server must learn $\Omega(1/\varepsilon)$ columns of $\tilde{\mathbf{L}}$. The probability that random discretized orthonormal matrices share $\Omega(1/\varepsilon)$ columns is sufficiently small that we can choose a large enough net for these columns to identify $\tilde{\mathbf{L}}$ in that net, requiring $\Omega(\phi/\varepsilon)$ bits of communication per server, or $\Omega(s\phi/\varepsilon)$ words in total. The argument for general k can be viewed as a "direct sum theorem", giving $\Omega(sk\phi/\varepsilon)$ words of communication in total.

	Upper bounds	Lower bounds
One-pass	$O(mk\varepsilon^{-1} + \text{poly}(k, \varepsilon^{-1}))$ (Theorem 37)	$\Omega(mk\varepsilon^{-1})$ bits [48]
Two-pass	$O(mk + \text{poly}(k, \varepsilon^{-1}))$ (Theorem 39)	$\Omega(mk)$ bits [48]

Table 3: Space upper/lower bounds for the Streaming PCA problem of Definition 6.

1.1.4 Streaming PCA

In Section 6 we use our techniques for dense distributed PCA to develop new PCA algorithms in the *streaming model* of computation. In this model, the columns of an $m \times n$ matrix $\mathbf{A}$ are presented to the algorithm one by one in an arbitrary order and the algorithm is allowed to see the columns only once. During the stream, the algorithm can (and should) keep some information about $\mathbf{A}$ (a "sketch" of $\mathbf{A}$) using as little space as possible; at the end of the stream, using this sketch, the algorithm computes a matrix $\mathbf{U}$ with orthonormal columns such that $\mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}$ is a "good" low-rank approximation to $\mathbf{A}$ (notice that to find the actual PCA projection $\mathbf{U}^{\mathrm{T}}\mathbf{A}$ a second pass over the columns of $\mathbf{A}$ is needed). Formally, we are interested in the following PCA problem.

Definition 5 (Streaming model for Principal Component Analysis)**.** *Let $\mathbf{A}$ be an $m \times n$ matrix consisting of n column vectors: $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$. In the streaming model of computation, an algorithm is allowed a single pass over the columns of $\mathbf{A}$ in an arbitrary order. At the end of the stream the algorithm stores some information regarding $\mathbf{A}$ which we call a "sketch" of $\mathbf{A}$. The space complexity of an algorithm in this model is defined as the total number of words required to describe the information the algorithm stores during the stream including the sketch. We assume each word is $O(\log(nms/\varepsilon))$ bits.*

Definition 6 (The Streaming Principal Component Analysis Problem)**.** *Given an $m \times n$ matrix $\mathbf{A}$, a rank parameter $k <$ rank$(\mathbf{A})$, and an accuracy parameter $0 < \varepsilon < 1$, design an algorithm that, using as little space as possible, first finds a sketch of $\mathbf{A}$ in the streaming model (see Definition 5) and then, using only this sketch constructs $\mathbf{U} \in \mathbb{R}^{m \times k}$ with k orthonormal columns such that*

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

A slightly relaxed version of this problem is to allow a constant number of passes over the matrix; in particular, we discuss a two-pass streaming algorithm in Section 6.

Our distributed PCA algorithm for dense matrices in Section 4 can be implemented in the streaming model. We summarize these results in Table 3, along with known space lower bounds for the problem. Here, as in previous work in the streaming model, we assume that $\mathbf{A}$ may be dense.

We compare our one-pass streaming algorithm in Section 6.1 to the best existing streaming PCA algorithm [37, 26]. The algorithm in [37, 26] requires space equivalent to describe $O(mk\varepsilon^{-1})$ real numbers; however, it is unclear if those numbers can be represented efficiently with a small number of bits, i.e., a word lower bound remained unexplored in [37, 26]. The running time of this deterministic algorithm is $O(mnk^2\varepsilon^{-2})$.

Our algorithm improves upon [37, 26] in two respects. First, our space bound ($O(mk/\varepsilon)$ - see Theorem 37) is in terms of words (we also bound the word size). This matches an $\Omega(km/\varepsilon)$ bit lower bound for 1-pass algorithms in [48], up to the distinction between words versus bits. Second, our algorithm is faster, requiring only $O\left(mn\log(k\varepsilon^{-1}) + mk^2 + n \cdot poly(k\varepsilon^{-1})\right)$ time.

Perhaps more surprisingly, we show that if we allow *two passes*, then there is an algorithm which uses only $O(km) + \mathrm{poly}(k/\varepsilon)$ words of space, and up to the $\mathrm{poly}(k/\varepsilon)$ term and the distinction between words versus bits, this is optimal for any constant number of passes. This is presented in Section 6.3. The optimality follows by a lower bound that applies to any constant number of passes from Appendix A in [48]. A "next natural goal" in [48] was to improve the lower bound of $\Omega(km)$ to a bound closer to the 1-pass $\Omega(km/\varepsilon)$ lower bound established in that paper; our upper bound shows this is not possible.

1.2 Road Map

We present prior results on distributed PCA algorithms in Section 2. Section 3 introduces the notation used in the paper as well as presents basic results from linear algebra that we use in our algorithms. Section 4 presents our distributed PCA algorithm for dense matrices. The communication complexity of the algorithm in this section can be stated only in terms of "real numbers". We resolve this issue in Section 5 where a modified algorithm has communication cost bounded in terms of machine words. Section 6 discusses space-optimal PCA methods in the streaming model of computation. Section 7 presents a distributed PCA algorithm for sparse matrices, while Section 8 extends this algorithm to a faster distributed PCA algorithm for sparse matrices. Section 9 presents communication lower bounds and Section 10 concludes the paper.

2 Related Work

Distributed PCA (or distributed SVD) algorithms have been investigated for a long time. One line of work, developed primarily within the numerical linear algebra literature, studies such algorithms from the perspective of parallelizing existing standard SVD algorithms without sacrificing accuracy. This approach aims at high accuracy implementations with the least possible communication cost. The distributed models of computation go typically beyond the column-partition model that we study in this paper [41]. An extensive survey of this line of work is out of the scope of our paper; we refer the reader to [31, 45, 29] and references therein for more details as well as to popular software for distributed SVD such ScaLAPACK [13] and Elemental [41].

Another line of work for distributed PCA algorithms has emerged within the machine learning and datamining communities. Such algorithms have been motivated by the need to apply SVD or PCA to extremely large matrices encoding enormous amounts of data. The algorithms in this line of research are typically heuristic approaches that work well in practice but come with no rigorous theoretical analysis. We refer the reader to [42, 38, 7, 50] for more details.

Finally, distributed PCA algorithms in the column-partition model have been recently studied within the theoretical computer science community. Perhaps the more intuitive algorithm for PCA in this model appeared in [24, 8]: first, a number of left singular vectors and singular values are computed in each machine; then, the server collects those singular vectors and concatenates them column-wise in a new matrix and then it computes the top k left singular vectors of this "aggregate" matrix. It is shown in [24, 8] that if the number of singular vectors and singular values in the first step is $O(k\varepsilon^{-1})$, then, the approximation error in Frobenius norm is at most $(1+\varepsilon)$ times the optimal Frobenius norm error; the communication cost is $O(skm\varepsilon^{-1})$ *real numbers* because each of the s machines sends $O(k\varepsilon^{-1})$ singular vectors of dimension m; unfortunately, it is unclear how one can obtain a communication cost in terms of words/bits. A different algorithm with the same communication cost (only in terms of real numbers since a word/bit communication bound remained unexplored) is implicit in [37, 26] (see Theorem 3.1 in [26] and the discussion

Reference	$\|\mathbf{A}-\mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_2 \leq$	$\|\mathbf{A}-\mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq$	δ	Communication cost	Total number of arithmetic operations
Implicit in [24]	-	$(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2$	0	-	$O\left(mn\min\{m,n\}+msk\varepsilon^{-1}\min\{m,sk\varepsilon^{-1}\}\right)$
Theorem 2 in [8]	-	$(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2$	0	-	$O\left(mn\min\{m,n\}+msk\varepsilon^{-1}\min\{m,sk\varepsilon^{-1}\}\right)$
Theorem 6 in [8]	-	$(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2$	>0	-	$O\left(\mathrm{nnz}(\mathbf{A})+s\left(\frac{m^3k}{\varepsilon^4}+\frac{k^2m^2}{\varepsilon^6}\right)\log\left(\frac{m}{\varepsilon}\right)\log\left(\frac{sk}{d\varepsilon}\right)\right)$
Implicit in [37, 26]	-	$(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2$	0	-	$O(mnk\varepsilon^{-2})$
Thm 1.1 in [34]	-	$(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2$	$O(1)$	$O\left(skm\varepsilon^{-1}+sk^2\varepsilon^{-4}\right)$	$O(poly(m,n,k,s,\varepsilon^{-1}))$
Thm 5.1 in [12]	$\|\mathbf{A}-\mathbf{A}_k\|_2+\Gamma$	-	>0	$O\left(sm+nk^5\varepsilon^{-2}\Delta\right)$	$O\left(\mathrm{nnz}(\mathbf{A})+\delta^{-1}nk^5\varepsilon^{-2}\sigma_1^2(\mathbf{A})\sigma_k^{-2}(\mathbf{A})\right)$
Remark p. 11 [12]	$\|\mathbf{A}-\mathbf{A}_k\|_2+\Gamma$	-	>0	$O\left(sm+nk^3\varepsilon^{-2}\Delta\right)$	$O\left(\mathrm{nnz}(\mathbf{A})+\delta^{-1}nk^5\varepsilon^{-2}\sigma_1^2(\mathbf{A})\sigma_k^{-2}(\mathbf{A})\right)$
Theorem 32	-	$(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2$	$O(1)$	$O(skm+sk^3\varepsilon^{-5})$	$O\left(mn\log(k\varepsilon^{-1})+mk^2+n\cdot poly(k\varepsilon^{-1})\right)$.
Theorem 46	-	$(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2$	$O(1)$	$O(sk\phi\varepsilon^{-1}+sk^2\varepsilon^{-4})$	$O(mn\min\{m,n\}+mns\cdot poly(k,1/\varepsilon))$
Theorem 58	-	$(1+\varepsilon)\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2$	$O(1)$	$O(sk\phi\varepsilon^{-1}+sk^3\varepsilon^{-5})$	$O\left(\mathrm{nnz}(\mathbf{A})\cdot\log^2(\frac{ns}{\delta})+(m+n)\cdot s\cdot poly(\frac{k}{\varepsilon}\log(\frac{ns}{\delta}))\right)$.

Table 4: Distributed PCA Algorithms in the column-partition model with s machines (see Definition 1). $\mathbf{A}\in\mathbb{R}^{m\times n}$ has rank ρ, $\mathbf{U}\in\mathbb{R}^{m\times k}$ with $k<\rho$ is orthonormal, $0<\varepsilon<1$, and each column of $\mathbf{A}$ contains at most $\phi\leq m$ non-zero elements. δ is the failure probability. Finally, for notational convenience let $\Gamma:=\varepsilon\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}$, $\Delta:=\sigma_1^2(\mathbf{A})\sigma_k^{-2}(\mathbf{A})\log^2\left(\|\mathbf{A}\|_2\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^{-1}\varepsilon^{-1}\right)$.

in Section 2.2 in [37]). Kannan, Vempala and Woodruff study a more general partition model in [34]: each machine has a matrix $\hat{\mathbf{A}}_i$ such that $\mathbf{A}=\sum_i\hat{\mathbf{A}}_i$. Their results, however, apply to the column-partition model, since it is a special case of this general model. They developed a $(1+\varepsilon)$ Frobenius norm error algorithm with communication cost $O\left(skm\varepsilon^{-1}+sk^2\varepsilon^{-4}\right)$ words. Bhojanapalli, Jain, and Sanghavi in [12] developed an algorithm that provides a bound with respect to the spectral norm. Their algorithm is based on sampling elements from the input matrix, but the communication cost is prohibitive if n is large. The cost also depends on the condition number of $\mathbf{A}$. Moreover, to implement the algorithm, one needs to know $\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}$. Finally, [36] discussed a distributed implementation of the "orthogonal iteration" to compute eigenvectors of graphs. The model they consider is different, but perhaps their protocol could be extended to our model.

3 Notation and linear algebra basics

Notation. $\mathbf{A},\mathbf{B},\ldots$ are matrices; $\mathbf{a},\mathbf{b},\ldots$ are column vectors. $\mathbf{I}_n$ is the $n\times n$ identity matrix; $\mathbf{0}_{m\times n}$ is the $m\times n$ matrix of zeros; $\mathbf{1}_n$ is the $n\times 1$ vector of ones; $\boldsymbol{e}_i$ is the standard basis (whose dimensionality will be clear from the context): the ith element of $\boldsymbol{e}_i$ is one and the rest are zeros. $\mathbf{A}^{(i)}$ and $\mathbf{A}_{(j)}$ denotes the ith column and jth row of $\mathbf{A}$, respectively. $\mathbf{A}_{ij}$ is the (i,j)th entry in $\mathbf{A}$.

Sampling Matrices. Let $\mathbf{A}=[\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(n)}]\in\mathbb{R}^{m\times n}$ and let $\mathbf{C}=[\mathbf{A}^{(i_1)},\ldots,\mathbf{A}^{(i_c)}]\in\mathbb{R}^{m\times c}$ consist of $c<n$ columns of $\mathbf{A}$. Note that we can write $\mathbf{C}=\mathbf{A}\boldsymbol{\Omega}$, where the *sampling matrix* is $\boldsymbol{\Omega}=[\mathbf{e}_{i_1},\ldots,\mathbf{e}_{i_c}]\in\mathbb{R}^{n\times c}$ (here $\mathbf{e}_i$ are the standard basis vectors in $\mathbb{R}^n$). If $\mathbf{D}\in\mathbb{R}^{c\times c}$ is a diagonal matrix, then $\mathbf{A}\boldsymbol{\Omega}\mathbf{D}$ contains c columns of $\mathbf{A}$ rescaled with the corresponding elements in $\mathbf{D}$. We abbreviate $\mathbf{S}:=\boldsymbol{\Omega}\mathbf{D}$, hence the matrix $\mathbf{S}\in\mathbb{R}^{n\times c}$ "samples" and "rescales" c columns from $\mathbf{A}$.

Matrix norms. We use the Frobenius and the spectral matrix-norms: $\|\mathbf{A}\|_{\mathrm{F}}^2=\sum_{i,j}\mathbf{A}_{ij}^2$; $\|\mathbf{A}\|_2=\max_{\|\mathbf{x}\|_2=1}\|\mathbf{A}\mathbf{x}\|_2$. $\|\mathbf{A}\|_\xi$ is used if a result holds for both norms $\xi=2$ and $\xi=\mathrm{F}$. The standard submultiplicativity property for matrix norms implies that for any $\mathbf{A}$ and $\mathbf{B}$: $\|\mathbf{A}\mathbf{B}\|_\xi\leq\|\mathbf{A}\|_\xi\cdot\|\mathbf{B}\|_\xi$. The triangle inequality for matrix norms implies that $\|\mathbf{A}+\mathbf{B}\|_\xi\leq\|\mathbf{A}\|_\xi+\|\mathbf{B}\|_\xi$. A version of the triangle inequality for the norms squared is: $\|\mathbf{A}+\mathbf{B}\|_\xi^2\leq 2\cdot\|\mathbf{A}\|_\xi^2+2\cdot\|\mathbf{B}\|_\xi^2$. A version of the matrix pythagorean theorem is: if $\mathbf{A}^{\mathrm{T}}\mathbf{B}$ is the all-zeros matrix, then $\|\mathbf{A}+\mathbf{B}\|_{\mathrm{F}}^2=\|\mathbf{A}\|_{\mathrm{F}}^2+\|\mathbf{B}\|_{\mathrm{F}}^2$. If $\mathbf{V}$ has orthonormal columns, then $\|\mathbf{A}\mathbf{V}^{\mathrm{T}}\|_\xi=\|\mathbf{A}\|_\xi$, for any $\mathbf{A}$. If $\mathbf{P}$ is a symmetric projection matrix (i.e., $\mathbf{P}=\mathbf{P}^{\mathrm{T}}$ and $\mathbf{P}^2=\mathbf{P}$) then, $\|\mathbf{P}\mathbf{A}\|_\xi\leq\|\mathbf{A}\|_\xi$, for any $\mathbf{A}$.

Singular Value Decomposition (SVD) and Moore-Penrose Pseudo-inverse. The SVD of $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $\text{rank}(\mathbf{A}) = \rho$ is

$$\mathbf{A} = \underbrace{\begin{pmatrix} \mathbf{U}_k & \mathbf{U}_{\rho-k} \end{pmatrix}}_{\mathbf{U}_\mathbf{A}\in\mathbb{R}^{m\times\rho}} \underbrace{\begin{pmatrix} \mathbf{\Sigma}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{\rho-k} \end{pmatrix}}_{\mathbf{\Sigma}_\mathbf{A}\in\mathbb{R}^{\rho\times\rho}} \underbrace{\begin{pmatrix} \mathbf{V}_k^{\mathrm{T}} \\ \mathbf{V}_{\rho-k}^{\mathrm{T}} \end{pmatrix}}_{\mathbf{V}_\mathbf{A}^{\mathrm{T}}\in\mathbb{R}^{\rho\times n}},$$

with singular values $\sigma_1 \geq \ldots \sigma_k \geq \sigma_{k+1} \geq \ldots \geq \sigma_\rho > 0$. Here, $k < \rho$. We will use $\sigma_i(\mathbf{A})$ to denote the i-th singular value of $\mathbf{A}$ when the matrix is not clear from the context. The matrices $\mathbf{U}_k \in \mathbb{R}^{m\times k}$ and $\mathbf{U}_{\rho-k} \in \mathbb{R}^{m\times(\rho-k)}$ contain the left singular vectors of $\mathbf{A}$, and, similarly, the matrices $\mathbf{V}_k \in \mathbb{R}^{n\times k}$ and $\mathbf{V}_{\rho-k} \in \mathbb{R}^{n\times(\rho-k)}$ contain the right singular vectors of $\mathbf{A}$. It is well-known that $\mathbf{A}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^{\mathrm{T}} = \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}} = \mathbf{U}_k\mathbf{U}_k^{\mathrm{T}}\mathbf{A}$ minimizes $\|\mathbf{A} - \mathbf{X}\|_\xi$ over all matrices $\mathbf{X} \in \mathbb{R}^{m\times n}$ of rank at most k. Specifically, $\|\mathbf{A} - \mathbf{A}_k\|_2^2 = \sigma_{k+1}^2(\mathbf{A})$ and $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 = \sum_{i=k+1}^{\rho}\sigma_i^2(\mathbf{A})$ (see [27]). $\mathbf{A}^\dagger = \mathbf{V}_\mathbf{A}\mathbf{\Sigma}_\mathbf{A}^{-1}\mathbf{U}_\mathbf{A}^{\mathrm{T}} \in \mathbb{R}^{n\times m}$ denotes the so-called Moore-Penrose pseudo-inverse of $\mathbf{A} \in \mathbb{R}^{m\times n}$ (here $\mathbf{\Sigma}_\mathbf{A}^{-1}$ is the inverse of $\mathbf{\Sigma}_\mathbf{A}$). By the SVD of $\mathbf{A}$ and $\mathbf{A}^\dagger$, for all $i = 1, \ldots, \rho = \text{rank}(\mathbf{A}) = \text{rank}(\mathbf{A}^\dagger)$: $\sigma_i(\mathbf{A}^\dagger) = 1/\sigma_{\rho-i+1}(\mathbf{A})$.

3.1 The best rank k matrix $\Pi_{\mathbf{V},k}^{\xi}(\mathbf{A})$ within a subspace $\mathbf{V}$

Our analysis uses theory involving computing the best rank k approximation of a matrix $\mathbf{A}$ within a given column subspace $\mathbf{V}$. Let $\mathbf{A} \in \mathbb{R}^{m\times n}$, let $k < n$ be an integer, and let $\mathbf{V} \in \mathbb{R}^{m\times c}$ with $k < c < n$. $\Pi_{\mathbf{V},k}^{\mathrm{F}}(\mathbf{A}) \in \mathbb{R}^{m\times n}$ is the best rank k approximation to $\mathbf{A}$ in the column span of $\mathbf{V}$. Equivalently, we can write

$$\Pi_{\mathbf{V},k}^{\mathrm{F}}(\mathbf{A}) = \mathbf{V}\mathbf{X}_{opt},$$

where

$$\mathbf{X}_{opt} = \underset{\mathbf{X}\in\mathbb{R}^{c\times n}:\text{rank}(\mathbf{X})\leq k}{\text{argmin}} \|\mathbf{A} - \mathbf{V}\mathbf{X}\|_{\mathrm{F}}^2.$$

In order to compute $\Pi_{\mathbf{V},k}^{\mathrm{F}}(\mathbf{A})$ given $\mathbf{A}$, $\mathbf{V}$, and k, one can use the following algorithm:

1: $\mathbf{V} = \mathbf{Y}\mathbf{\Psi}$ is a qr decomposition of $\mathbf{V}$ with $\mathbf{Y} \in \mathbb{R}^{m\times c}$ and $\mathbf{\Psi} \in \mathbb{R}^{c\times c}$. This step requires $O(mc^2)$ arithmetic operations.
2: $\mathbf{\Xi} = \mathbf{Y}^{\mathrm{T}}\mathbf{A} \in \mathbb{R}^{c\times n}$. This step requires $O(mnc)$ arithmetic operations.
3: $\mathbf{\Xi}_k = \mathbf{\Delta}\tilde{\mathbf{\Sigma}}_k\tilde{\mathbf{V}}_k^{\mathrm{T}} \in \mathbb{R}^{c\times n}$ is a rank k SVD of $\mathbf{\Xi}$ with $\mathbf{\Delta} \in \mathbb{R}^{c\times k}$, $\tilde{\mathbf{\Sigma}}_k \in \mathbb{R}^{k\times k}$, and $\tilde{\mathbf{V}}_k \in \mathbb{R}^{n\times k}$. This step requires $O(nc^2)$ arithmetic operations.
4: Return $\mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\mathbf{A} \in \mathbb{R}^{m\times n}$ of rank at most k.

Notice that $\mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\mathbf{A} \in \mathbb{R}^{m\times n}$ is a rank k matrix that lies in the column span of $\mathbf{V}$. The next lemma is a simple corollary of Lemma 4.3 in [18].

Lemma 7. *Given $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{V} \in \mathbb{R}^{m\times c}$ and an integer k, $\mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}$ and $\mathbf{Q}\tilde{\mathbf{U}}_k\tilde{\mathbf{\Sigma}}_k\tilde{\mathbf{V}}_k^{\mathrm{T}}$ satisfy:*

$$\|\mathbf{A} - \mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{Y}\mathbf{\Delta}\tilde{\mathbf{\Sigma}}_k\tilde{\mathbf{V}}_k^{\mathrm{T}}\|_{\mathrm{F}}^2 = \|\mathbf{A} - \Pi_{\mathbf{V},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2$$

The above algorithm requires $O(mnc + nc^2)$ arithmetic operations to construct $\mathbf{Y}, \mathbf{\Psi}$, and $\mathbf{\Delta}$. We will denote the above procedure as $[\mathbf{Y}, \mathbf{\Psi}, \mathbf{\Delta}] = BestColumnSubspaceSVD(\mathbf{A}, \mathbf{V}, k)$.

Proof. The equality was proven in Lemma 4.3 in [18]. To prove the inequality, notice that for any matrix $\mathbf{X}$: $\|\mathbf{A} - \mathbf{Y}\mathbf{\Delta}(\mathbf{Y}\mathbf{\Delta})^\dagger\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{Y}\mathbf{\Delta}\mathbf{X}\|_{\mathrm{F}}^2$. Also, $(\mathbf{Y}\mathbf{\Delta})^\dagger = \mathbf{\Delta}^\dagger\mathbf{Y}^\dagger = \mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}$, because both matrices are orthonormal. ∎

Next, we state a version of the above lemma for the transpose of $\mathbf{A}$, equivalently for the best rank k approximation within the row space of a given subspace $\mathbf{R}$. Let $\mathbf{A} \in \mathbb{R}^{m\times n}$, let $k < n$ be an integer, and let $\mathbf{R} \in \mathbb{R}^{c\times n}$ with $k < c < m$. $\Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A}) \in \mathbb{R}^{m\times n}$ is the best rank k approximation to $\mathbf{A}$ in the row span of $\mathbf{R}$. Equivalently, we can write

$$\Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A}) = \mathbf{X}_{opt}\mathbf{R},$$

where

$$\mathbf{X}_{opt} = \underset{\mathbf{X}\in\mathbb{R}^{m\times c}:\mathrm{rank}(\mathbf{X})\leq k}{\mathrm{argmin}} \|\mathbf{A} - \mathbf{X}\mathbf{R}\|^2_{\mathrm{F}}.$$

In order to compute $\Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A})$ given $\mathbf{A}$, $\mathbf{R}$, and k, one can use the following algorithm:

1: $\mathbf{R}^{\mathrm{T}} = \mathbf{Y}\mathbf{Z}$ is a qr decomposition of $\mathbf{R}^{\mathrm{T}}$ with $\mathbf{Y} \in \mathbb{R}^{n\times c}$ and $\mathbf{Z} \in \mathbb{R}^{c\times c}$. This step requires $O(nc^2)$ arithmetic operations.
2: $\mathbf{\Xi} = \mathbf{A}\mathbf{Y} \in \mathbb{R}^{m\times c}$. This step requires $O(mnc)$ arithmetic operations.
3: $\mathbf{\Xi}_k = \tilde{\mathbf{U}}_k\tilde{\mathbf{\Sigma}}_k\mathbf{\Delta}^{\mathrm{T}} \in \mathbb{R}^{m\times c}$ is a rank k SVD of $\mathbf{\Xi}$ with $\tilde{\mathbf{U}}_k \in \mathbb{R}^{m\times k}$, $\tilde{\mathbf{\Sigma}}_k \in \mathbb{R}^{k\times k}$, and $\mathbf{\Delta} \in \mathbb{R}^{c\times k}$. This step requires $O(mc^2)$ arithmetic operations.
4: Return $\mathbf{A}\mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}} \in \mathbb{R}^{m\times n}$ of rank at most k.

Notice that $\mathbf{A}\mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}} \in \mathbb{R}^{m\times n}$ is a rank k matrix that lies in the row span of $\mathbf{R}$. The next lemma is a corollary of Lemma 7 when applied with $\mathbf{A} := \mathbf{A}^{\mathrm{T}}$ and $\mathbf{V} := \mathbf{R}^{\mathrm{T}}$.

Lemma 8. *Given $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{R} \in \mathbb{R}^{c\times n}$ and an integer k, $\mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}$ and $\tilde{\mathbf{U}}_k\tilde{\mathbf{\Sigma}}_k\mathbf{\Delta}^{\mathrm{T}}$ satisfy:*

$$\|\mathbf{A} - \mathbf{A}\mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\|^2_{\mathrm{F}} \leq \|\mathbf{A} - \tilde{\mathbf{U}}_k\tilde{\mathbf{\Sigma}}_k\mathbf{\Delta}^{\mathrm{T}}\mathbf{R}\|^2_{\mathrm{F}} = \|\mathbf{A} - \Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A})\|^2_{\mathrm{F}}.$$

The above algorithm requires $O(mnc + mc^2)$ arithmetic operations to construct $\mathbf{Y}, \mathbf{Z}$, and $\mathbf{\Delta}$. We will denote the above procedure as $[\mathbf{Y}, \mathbf{Z}, \mathbf{\Delta}] = BestRowSubspaceSVD(\mathbf{A}, \mathbf{R}, k)$.

4 Distributed PCA for dense matrices

This section describes a fast distributed PCA algorithm with total communication $O(msk)$ words plus low order terms, which is optimal for $\phi = \Omega(\varepsilon \cdot m)$, for some fixed $\varepsilon > 0$. The algorithm employs, in a novel way, the notion of sparse *affine embeddings* from [19]. In particular, whereas all previous [43, 19] dimension-reduction-based SVD methods reduce one dimension of the input matrix to compute some approximation to the SVD, our method reduces *both* dimensions and computes an approximation to the SVD from a small almost square matrix. We first present the batch version of the algorithm which offers a new low-rank matrix approximation technique; a specific implementation of this algorithm offers a communication-optimal distributed PCA algorithm (we also discuss in Section 6 two variants of this algorithm that offer space-optimal PCA methods in the streaming model of computation). Before presenting all these new algorithms in detail, we present the relevant results from the previous literature that we employ in the analysis.

4.1 Background material

4.1.1 Low-rank preserving sketching matrices

In this section, we recap a notion of sketching matrices which we call "low-rank preserving sketching matrices". A sketching matrix from this family is a linear matrix transformation and it has the property that it preserves, up to some error, the best rank k approximation of the matrix in hand.

Definition 9 (Low-rank preserving sketching matrices)**.** *We say that a random variable $\xi \times m$ matrix $\mathbf{W}$ is a $(\gamma, \delta, \epsilon, k)$-***low-rank preserving sketching matrix** *if the following two conditions hold simulteously with probability $1 - \delta$:*

1. *For any fixed k-dimensional subspace M, with probability at least $1 - \delta/2$, for all $\mathbf{y} \in M$ it is:*

$$(1 - \gamma)\|\mathbf{y}\|_2 \leq \|\mathbf{W}\mathbf{y}\|_2 \leq (1 + \gamma)\|\mathbf{y}\|_2.$$

2. *There is a constant $C > 0$ so that with probability at least $1 - \delta/2$, for any fixed matrices $\mathbf{A}$ and $\mathbf{B}$ each with m rows,*

$$\|\mathbf{A}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\mathbf{W}\mathbf{B} - \mathbf{A}^{\mathrm{T}}\mathbf{B}\|_{\mathrm{F}}^2 \leq \frac{C\epsilon}{k} \cdot \|\mathbf{A}\|_{\mathrm{F}}^2 \cdot \|\mathbf{B}\|_{\mathrm{F}}^2.$$

The two properties in this definition have nothing to do with low-rank matrix approximation, but as we will see in Lemma 13 later in this section, those two properties is all that we need in order to prove that, for any matrix $\mathbf{A}$ with m rows and ξ sufficiently large, the matrix $\mathbf{W}\mathbf{A}$ "contains" a rank k matrix that is as good as the rank k matrix that can be obtained via the SVD of $\mathbf{A}$. Next, we present two families of random matrices that are low-rank preserving sketching matrices.

Lemma 10 (Gaussians)**.** *For $\xi = \Omega(k/\epsilon)$, if $\mathbf{S}$ is a $\xi \times m$ matrix of i.i.d. normal random variables with variance $1/\xi$, then $\mathbf{S}$ is a $(1/8, 1/500, \epsilon, k)$-low-rank sketching preserving matrix.*

Proof. It is known that if $\mathbf{S}$ has $\xi = \Omega(k)$ rows, then it satisfies the first condition of Definition 9 with probability at least $1 - 1/1000$ and $\gamma = 1/8$ (see p.6 of [49]). The second condition in Definition 9 is given by Theorem 21 of [32] (see also Remark 22 following it, and standard bounds on χ^2-random variables). ■

Lemma 11 (CountSketch)**.** *For $\xi = \Omega(k^2 + k/\epsilon)$, if $\mathbf{T}$ is an $\xi \times m$ sparse subspace embedding matrix (see Definition 49), then $\mathbf{T}$ is a $(1/8, 1/500, \epsilon, k)$-low-rank preserving sketching matrix.*

Proof. For the first property, see Theorem 2.9 of [49], while the second property is Theorem 2.8 of [49] (these theorems originated in [20, 39, 40]). ■

Low-rank preserving sketching matrices are composable and such a composition combines the best of two worlds: fast computation of count-sketch and small dimension of gaussian matrices.

Lemma 12 (Composition of Gaussians with CountSketch)**.** *Let $\xi = O(k/\epsilon)$ and $\xi_2 = O(k^2 + k/\epsilon)$. If $\mathbf{S}$ is a $\xi \times \xi_2$ matrix of i.i.d. normal random variables with variance $1/\xi$, and $\mathbf{T}$ is a $\xi_2 \times m$ sparse subspace embedding matrix, then $\mathbf{W} = \mathbf{S} \cdot \mathbf{T} \in \mathbb{R}^{\xi \times m}$ is a $(1/3, 1/100, \epsilon, k)$-low rank preserving sketching matrix.*

Proof. Consider any k-dimensional space, which we identify with the column space of an $m \times k$ matrix $\mathbf{A}$. By Lemma 10, with probability at least $999/1000$ over the choice of $\mathbf{T}$, $\|\mathbf{T}\mathbf{A}\mathbf{x}\|_2 = (1 \pm 1/8)\|\mathbf{A}\mathbf{x}\|_2$ simultaneously for all $\mathbf{x} \in \mathbb{R}^k$. Similarly, by considering the column space of the matrix $\mathbf{T}\mathbf{A}$ which is again a k-dimensional space, by Lemma 11 with probability at least $999/1000$ over the choice of $\mathbf{S}$,

$$\|\mathbf{S}(\mathbf{T}\mathbf{A})\mathbf{x}\|_2 = (1 \pm 1/8)\|\mathbf{T}\mathbf{A}\mathbf{x}\|_2,$$

simultaneously for all $\mathbf{x}$. It follows that with probability at least $499/500$,

$$\|\mathbf{S}\mathbf{T}\mathbf{A}\mathbf{x}\|_2 = (1 \pm (2 \cdot 1/8 + 1/64))\|\mathbf{A}\mathbf{x}\|_2 = (1 \pm 1/3)\|\mathbf{A}\mathbf{x}\|_2,$$

for all $\mathbf{x}$.

For the second property, for any fixed matrices $\mathbf{SA}$ and $\mathbf{SB}$,

$$\Pr_{\mathbf{S}}[\|\mathbf{A}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{STB} - \mathbf{A}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{TB}\|_{\mathrm{F}} \leq \sqrt{\frac{k}{\epsilon}}\|\mathbf{TA}\|_{\mathrm{F}}\|\mathbf{TB}\|_{\mathrm{F}}] \geq \frac{999}{1000}.$$

Similarly we have,

$$\Pr_{\mathbf{T}}[\|\mathbf{A}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{TB} - \mathbf{A}^{\mathrm{T}}\mathbf{B}\|_{\mathrm{F}}] \leq \sqrt{\frac{k}{\epsilon}}\|\mathbf{A}\|_{\mathrm{F}}\|\mathbf{B}\|_{\mathrm{F}} \geq \frac{999}{1000}.$$

Conditioned on these two events,

$$\|\mathbf{A}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{STB} - \mathbf{A}^{\mathrm{T}}\mathbf{B}\|_{\mathrm{F}} \leq \sqrt{\frac{k}{\epsilon}}\|\mathbf{TA}\|_{\mathrm{F}}\|\mathbf{TB}\|_{\mathrm{F}} + \sqrt{\frac{k}{\epsilon}}\|\mathbf{A}\|_{\mathrm{F}}\|\mathbf{B}\|_{\mathrm{F}}.$$

By Lemma 40 of [20] (which is stated for probability $9/10$ but holds for any constant probability by increasing the number of rows of $\mathbf{T}$ by a constant factor), with probability at least $999/1000$, $\|\mathbf{TA}\|_{\mathrm{F}} \leq 2\|\mathbf{A}\|_{\mathrm{F}}$ and $\|\mathbf{TB}\|_{\mathrm{F}} \leq 2\|\mathbf{B}\|_{\mathrm{F}}$. Hence, by a union bound, with probability at least $1 - 5/1000 = 199/200$, we have

$$\|\mathbf{A}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{STB} - \mathbf{A}^{\mathrm{T}}\mathbf{B}\|_{\mathrm{F}} \leq 5\sqrt{\frac{k}{\epsilon}}\|\mathbf{A}\|_{\mathrm{F}}\|\mathbf{B}\|_{\mathrm{F}},$$

which completes the proof. ■

Finally, we present the main result in this subsection which argues that "sketching" a matrix $\mathbf{A}$ with a low-rank preserving sketching matrix $\mathbf{W}$ preserves the best rank k matrix to $\mathbf{A}$.

Lemma 13 (Good low-rank subspace)**.** *Let $\mathbf{A}$ be an $m \times n$ matrix, $k < \mathrm{rank}(\mathbf{A})$ be a rank parameter, and $\varepsilon > 0$ be an accuracy parameter. Let $\mathbf{W} \in \mathbb{R}^{\xi \times m}$ be a $(1/3, 0.01, \epsilon, k)$-***low-rank preserving sketching matrix** *and let $\mathbf{R} = \mathbf{WA}$ be an $\xi \times n$ matrix. Then, with probability at least 0.99:*

$$\|\mathbf{A} - \Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A})\|^2_{\mathrm{F}} \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|^2_{\mathrm{F}}.$$

$\Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A}) \in \mathbb{R}^{m \times n}$ is the best rank k approximation to $\mathbf{A}$ in the row span of $\mathbf{R}$ (see Definition 3.1).

Proof. This follows from Lemma 4.2 in [49], where the success probability in the proof of Lemma 4.2 there is 0.99 for a $(1/3, .01, \varepsilon, k)$-low rank preserving matrix (this theorem originated in [20]). ■

The following lemma presents a version of Lemma 13 in the case of a specific matrix $\mathbf{W}$. Notice that Lemma 13 is true for any matrix $\mathbf{W}$ that satisfies the two properties of Definition 9.

Lemma 14 (Good low-rank subspace)**.** *Let $\mathbf{A}$ be an $m \times n$ matrix, $k < \mathrm{rank}(\mathbf{A})$ be a rank parameter, and $\varepsilon > 0$ be an accuracy parameter. Let $\xi_1 = O(k/\epsilon)$ and $\xi_2 = O(k^2 + k/\epsilon)$. If $\mathbf{S}$ is a $\xi_1 \times \xi_2$ matrix of i.i.d. normal random variables with variance $1/\xi_1$, and $\mathbf{T}$ is a $\xi_2 \times m$ sparse subspace embedding matrix (see Definition 49), then for $\mathbf{W} = \mathbf{S} \cdot \mathbf{T} \in \mathbb{R}^{\xi_1 \times m}$, let $\mathbf{R} = \mathbf{WA} \in \mathbb{R}^{\xi_1 \times n}$. Then, with probability at least 0.99:*

$$\|\mathbf{A} - \Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A})\|^2_{\mathrm{F}} \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|^2_{\mathrm{F}}.$$

$\Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A}) \in \mathbb{R}^{m \times n}$ is the best rank k approximation to $\mathbf{A}$ in the row span of $\mathbf{R}$ (see Definition 3.1).

Proof. Lemma 12 argues that $\mathbf{W}$ is a $(1/3, 1/100, \epsilon, k)$-low rank preserving sketching matrix. Combining with Lemma 13 concludes the proof. ■

4.1.2 Subsampled Randomized Hadamard Transform and Affine Embeddings

Our algorithms use the following tool, known as "Subsampled Randomized Hadamard Transform" or SRHT for short, to implement efficiently fast dimension reduction in large matrices.

Definition 15 (Normalized Walsh–Hadamard Matrix)**.** *Fix an integer $m = 2^p$, for $p = 1, 2, 3, ...$ The (non-normalized) $m \times m$ matrix of the Walsh–Hadamard transform is defined recursively as,*

$$\mathbf{H}_n = \begin{bmatrix} \mathbf{H}_{m/2} & \mathbf{H}_{m/2} \\ \mathbf{H}_{m/2} & -\mathbf{H}_{m/2} \end{bmatrix}, \qquad \textit{with} \qquad \mathbf{H}_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}.$$

The $m \times m$ normalized matrix of the Walsh–Hadamard transform is equal to $\mathbf{H} = m^{-\frac{1}{2}}\mathbf{H}_m \in \mathbb{R}^{m\times m}$.

Definition 16 (Subsampled Randomized Hadamard Transform (SRHT) matrix)**.** *Fix integers ξ and $m = 2^p$ with $\xi < m$ and $p = 1, 2, 3, ...$ An SRHT matrix is an $\xi \times m$ matrix of the form*

$$\mathbf{T} = \sqrt{\frac{m}{\xi}} \cdot \mathbf{RHD};$$

- $\mathbf{D} \in \mathbb{R}^{m\times m}$ *is a random diagonal matrix whose entries are independent random signs, i.e. random variables uniformly distributed on $\{\pm 1\}$.*
- $\mathbf{H} \in \mathbb{R}^{m\times m}$ *is a normalized Walsh–Hadamard matrix (see Definition 15).*
- $\mathbf{R} \in \mathbb{R}^{\xi\times m}$ *is a subset or r rows from the $n \times n$ identity matrix, where the rows are chosen uniformly at random and without replacement.*

The next lemma argues that an SRHT matrix is a so-called "affine embedding matrix". The SRHT is one of the possible choices of Lemma 32 in [19] that will satisfy the lemma.

Lemma 17 (Affine embeddings - Theorem 39 in [19])**.** *Suppose $\mathbf{G}$ and $\mathbf{H}$ are matrices with m rows, and $\mathbf{G}$ has rank at most r. Suppose $\mathbf{T}$ is a $\xi \times m$ SRHT matrix (see Definition 16) with $\xi = O(r/\varepsilon^2)$. Then, with probability 0.99, for all $\mathbf{X}$ simultaneously:*

$$(1-\varepsilon)\cdot\|\mathbf{GX}-\mathbf{H}\|_{\mathrm{F}}^2 \le \|\mathbf{T}(\mathbf{GX}-\mathbf{H})\|_{\mathrm{F}}^2 \le (1+\varepsilon)\cdot\|\mathbf{GX}-\mathbf{H}\|_{\mathrm{F}}^2.$$

Finally, we note that matrix-vector multiplications with SRHT's are fast.

Lemma 18 (Fast Matrix-Vector Multiplication, Theorem 2.1 in [6])**.** *Given $\mathbf{x} \in \mathbb{R}^m$ and $\xi < m$, one can construct $\mathbf{T} \in \mathbb{R}^{\xi\times m}$ and compute $\mathbf{Tx}$ in at most $2m\log_2(\xi+1))$ operations.*

4.1.3 Generalized rank-constrained matrix approximations

Let $\mathbf{M} \in \mathbb{R}^{m\times n}$, $\mathbf{N} \in \mathbb{R}^{m\times c}$, $\mathbf{L} \in \mathbb{R}^{r\times n}$, and $k \le c, r$ be an integer. Consider the following optimization problem,

$$\mathbf{X}_{opt} \in \underset{\mathbf{X}\in\mathbb{R}^{c\times r},\mathrm{rank}(\mathbf{X})\le k}{\mathrm{argmin}} \|\mathbf{M}-\mathbf{NXL}\|_{\mathrm{F}}^2.$$

Then, the solution $\mathbf{X}_{opt} \in \mathbb{R}^{c\times r}$ with $\mathrm{rank}(\mathbf{X}_{opt}) \le k$ that has the minimum $\|\mathbf{X}_{opt}\|_{\mathrm{F}}$ out of all possible feasible solutions is given via the following formula,

$$\mathbf{X}_{opt} = \mathbf{N}^{\dagger}\left(\mathbf{U_N}\mathbf{U_N}^{\mathrm{T}}\mathbf{MV_L}\mathbf{V_L}^{\mathrm{T}}\right)_k\mathbf{L}^{\dagger}.$$

$\left(\mathbf{U_N}\mathbf{U_N}^{\mathrm{T}}\mathbf{MV_L}\mathbf{V_L}^{\mathrm{T}}\right)_k \in \mathbb{R}^{m\times n}$ of rank at most k denotes the best rank k matrix to $\mathbf{U_N}\mathbf{U_N}^{\mathrm{T}}\mathbf{MV_L}\mathbf{V_L}^{\mathrm{T}} \in \mathbb{R}^{m\times n}$. This result was proven in [25] (see also [44] for the spectral norm version of the problem).

4.2 A batch algorithm for the fast low rank approximation of matrices

In this section, we describe a new method for computing quickly a low-rank approximation to a given matrix. This method does not offer any specific advantages over previous such techniques [43, 19, 16]; however, this new algorithm can be implemented efficiently in the distributed setting (see Section 4.3) and in the streaming model of computation (see Section 6); in fact we are able to obtain communication-optimal and space-optimal results, respectively. For completeness as well as ease of presentation, we first present and analyze the simple batch version of the algorithm. The algorithm uses the low-rank preserving matrix of Lemma 12 and the SRHT matrix of Definition 16 in order to reduce both dimensions of $\mathbf{A}$, before computing some sort of SVD to a $\mathrm{poly}(k/\varepsilon) \times \mathrm{poly}(k/\varepsilon)$ matrix (see Step 7 in the algorithm below).

Consider the usual inputs: a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, a rank parameter $k < \mathrm{rank}(\mathbf{A})$, and an accuracy parameter $0 < \varepsilon < 1$. The algorithm below returns an orthonormal matrix $\mathbf{U} \in \mathbb{R}^{m\times k}$ such that

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\cdot\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2.$$

Algorithm

1. Construct a low-rank preserving sketching matrix $\mathbf{W} \in \mathbb{R}^{\xi\times m}$ with $\xi = O(k\varepsilon^{-1})$ (see Lemma 12); then, construct $\mathbf{R} = \mathbf{W}\mathbf{A} \in \mathbb{R}^{\xi\times n}$.
2. Construct affine embedding matrices $\mathbf{T}_{left} \in \mathbb{R}^{\xi_1\times m}$ and $\mathbf{T}_{right} \in \mathbb{R}^{n\times\xi_2}$ with $\xi_1 = O(k/\varepsilon^3)$ and $\xi_2 = O(k/\varepsilon^3)$ (see Definition 16).
3. Construct $\mathbf{M} = \mathbf{T}_{left}\mathbf{A}\mathbf{T}_{right} \in \mathbb{R}^{\xi_1\times\xi_2}$.
4. Construct $\mathbf{L} = \mathbf{W}\mathbf{A}\mathbf{T}_{right} \in \mathbb{R}^{\xi\times\xi_2}$.
5. Construct $\mathbf{N} = \mathbf{T}_{left}\mathbf{A}\mathbf{R}^{\mathrm{T}} \in \mathbb{R}^{\xi_1\times\xi}$.
6. Construct $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X})\leq k}{\mathrm{argmin}} \|\mathbf{N}\cdot\mathbf{X}\cdot\mathbf{L}-\mathbf{M}\|_{\mathrm{F}}^2$. (Notice that $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X})\leq k}{\mathrm{argmin}} \|\mathbf{T}_{left}\left(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}-\mathbf{A}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2$.)
7. Compute the SVD of $\mathbf{X}_* = \mathbf{U}_{\mathbf{X}_*}\mathbf{\Sigma}_{\mathbf{X}_*}\mathbf{V}_{\mathbf{X}_*}^{\mathrm{T}}$ ($\mathbf{U}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$, $\mathbf{\Sigma}_{\mathbf{X}_*} \in \mathbb{R}^{k\times k}$ $\mathbf{V}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$); $\mathbf{T} = \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*}$.
8. Compute an orthonormal basis $\mathbf{U} \in \mathbb{R}^{m\times k}$ for $span(\mathbf{T})$ (notice that $rank(\mathbf{T}) \leq k$).

Theorem 22 later in this section analyzes the approximation error and the running time of the previous algorithm. First, we prove a few intermediate results.

Lemma 19. *For all matrices $\mathbf{X} \in \mathbb{R}^{\xi\times\xi}$ and with probability at least 0.98:*

$$(1-\varepsilon)^2\cdot\|\mathbf{A}-\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\|_{\mathrm{F}}^2 \leq \|\mathbf{T}_{left}\left(\mathbf{A}-\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)^2\cdot\|\mathbf{A}-\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\|_{\mathrm{F}}^2$$

Proof. Notice that $\mathrm{rank}(\mathbf{A}\mathbf{R}^{\mathrm{T}}) \leq \xi = O(k/\varepsilon^2)$. Then, from Lemma 17 ($\mathbf{G} := \mathbf{A}\mathbf{R}^{\mathrm{T}}$ and $\mathbf{H} := \mathbf{A}$), with probability at least 0.99 for all $\mathbf{Y} \in \mathbb{R}^{\xi\times n}$:

$$(1-\varepsilon)\cdot\|\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{Y}-\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{T}_{left}(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{Y}-\mathbf{A})\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\cdot\|\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{Y}-\mathbf{A}\|_{\mathrm{F}}^2.$$

Replacing $\mathbf{Y} = \mathbf{X}\mathbf{R}$, for an arbitrary $\mathbf{X} \in \mathbb{R}^{\xi\times\xi}$, we obtain that with probability at least 0.99 and for all $\mathbf{X} \in \mathbb{R}^{\xi\times\xi}$ simultaneously:

$$(1-\varepsilon)\cdot\|\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}-\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{T}_{left}(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}-\mathbf{A})\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\cdot\|\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}-\mathbf{A}\|_{\mathrm{F}}^2. \quad (1)$$

Now, notice that $\text{rank}(\mathbf{R}^{\mathrm{T}}) \leq \xi = O(k/\varepsilon^2)$. Then, from Lemma 17 ($\mathbf{G} := \mathbf{R}^{\mathrm{T}}$ and $\mathbf{H} := \mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}}$), we have that with probability at least 0.99 for all $\mathbf{Z} \in \mathbb{R}^{\xi \times \xi_1}$:

$$(1-\varepsilon) \cdot \|\mathbf{R}^{\mathrm{T}}\mathbf{Z} - \mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq \|\mathbf{T}_{right}(\mathbf{R}^{\mathrm{T}}\mathbf{Z} - \mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}})\|_{\mathrm{F}}^2 \leq (1+\varepsilon) \cdot \|\mathbf{R}^{\mathrm{T}}\mathbf{Z} - \mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}}\|_{\mathrm{F}}^2.$$

Replacing $\mathbf{Z} = \mathbf{X}^{\mathrm{T}}\mathbf{R}\mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}}$ for an arbitrary $\mathbf{X} \in \mathbb{R}^{\xi \times \xi}$, we obtain that with probability at least 0.99 and for all $\mathbf{X}$ simultaneously:

$$\begin{aligned}(1-\varepsilon) \cdot \|\mathbf{R}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{R}\mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}} - \mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}}\|_{\mathrm{F}}^2 &\leq \|\mathbf{T}_{right}(\mathbf{R}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{R}\mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}} - \mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}})\|_{\mathrm{F}}^2 \\ &\leq (1+\varepsilon) \cdot \|\mathbf{R}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{R}\mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}} - \mathbf{A}^{\mathrm{T}}\mathbf{T}_{left}^{\mathrm{T}}\|_{\mathrm{F}}^2.\end{aligned}$$

Taking transposes to this last relation and combining with Eqn. (1) concludes the proof. ■

Lemma 20. *Let* $\mathbf{X}_{opt} = \arg\min_{\text{rank}(\mathbf{X}) \leq k} \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\|_{\mathrm{F}}^2$ *with* $\mathbf{X}_{opt} \in \mathbb{R}^{\xi \times \xi}$. *Then, with probability at least* 0.98

$$\min_{\text{rank}(\mathbf{X}) \leq k} \|\mathbf{T}_{left}\left(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R} - \mathbf{A}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_*\mathbf{R}\|_{\mathrm{F}}^2 \leq \frac{(1+\varepsilon)^2}{(1-\varepsilon)^2}\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\|_{\mathrm{F}}^2$$

Proof. From Lemma 19, we have that for all matrices $\mathbf{X} \in \mathbb{R}^{\xi \times \xi}$ and with probability at least 0.98:

$$(1-\varepsilon)^2 \cdot \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\|_{\mathrm{F}}^2 \leq \|\mathbf{T}_{left}\left(\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)^2 \cdot \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\|_{\mathrm{F}}^2$$

Applying this for $\mathbf{X} := \mathbf{X}_{opt}$, we obtain:

$$(1-\varepsilon)^2 \cdot \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\|_{\mathrm{F}}^2 \leq \|\mathbf{T}_{left}\left(\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2 \overset{(e)}{\leq} (1+\varepsilon)^2 \cdot \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\|_{\mathrm{F}}^2$$

Applying this for $\mathbf{X} := \mathbf{X}_*$, we obtain:

$$(1-\varepsilon)^2 \cdot \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_*\mathbf{R}\|_{\mathrm{F}}^2 \overset{(f)}{\leq} \|\mathbf{T}_{left}\left(\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_*\mathbf{R}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)^2 \cdot \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\|_{\mathrm{F}}^2$$

Combining $(f), (e)$ along with the optimality of $\mathbf{X}_*$, formally using the relation:

$$\|\mathbf{T}_{left}\left(\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_*\mathbf{R}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2 \leq \|\mathbf{T}_{left}\left(\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2,$$

shows the claim. ■

In words, the lemma indicates that in order to $(1+\varepsilon)$-approximate the optimization problem $\min_{\text{rank}(\mathbf{X}) \leq k} \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\|_{\mathrm{F}}^2$, it suffices to "sketch" the matrix $\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}$ from left and right with the matrices $\mathbf{T}_{left}$ and $\mathbf{T}_{right}$. Recall that $\mathbf{X}_* = \underset{\text{rank}(\mathbf{X}) \leq k}{\text{argmin}} \|\mathbf{T}_{left}\left(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R} - \mathbf{A}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2$.

Lemma 21. *Let* $\mathbf{X}_{opt} = \arg\min_{\text{rank}(\mathbf{X}) \leq k} \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\|_{\mathrm{F}}^2$. *Then, with probability at least* 0.99

$$\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\|_{\mathrm{F}}^2 \leq (1+\varepsilon) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2,$$

Proof. Let $\mathbf{R}^{\mathrm{T}} = \mathbf{Y}\mathbf{Z}_*$ be the QR factorization of $\mathbf{R}^{\mathrm{T}} \in \mathbb{R}^{n\times\xi}$ with $\mathbf{Y} \in \mathbb{R}^{n\times\xi}$ and $\mathbf{Z}_* \in \mathbb{R}^{\xi\times\xi}$, where $\mathbf{Z}_*$ is invertible ($\mathrm{rank}(\mathbf{Z}_*) = \xi$ because $\mathrm{rank}(\mathbf{R}) = \xi$). Let also $\mathbf{\Delta}_* \in \mathbb{R}^{\xi\times k}$ be the matrix in Lemma 8: $[\mathbf{Y}, \mathbf{Z}_*, \mathbf{\Delta}_*] = BestRowSubspaceSVD(\mathbf{A}, \mathbf{R}, k)$. Then,

$$\begin{aligned}
\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\|_{\mathrm{F}}^2 &= \arg\min_{\mathrm{rank}(\mathbf{X})\le k} \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R}\|_{\mathrm{F}}^2 \\
&\le \arg\min_{\mathbf{Z}\in\mathbb{R}^{\xi\times\xi},\mathbf{\Delta}\in\mathbb{R}^{\xi\times k}} \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{Z}^{-1}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}(\mathbf{Z}^{\mathrm{T}})^{-1}\mathbf{R}\|_{\mathrm{F}}^2 \\
&\le \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{Z}_*^{-1}\mathbf{\Delta}_*\mathbf{\Delta}_*^{\mathrm{T}}(\mathbf{Z}_*^{\mathrm{T}})^{-1}\mathbf{R}\|_{\mathrm{F}}^2 \\
&= \|\mathbf{A} - \mathbf{A}\mathbf{Y}\mathbf{\Delta}_*\mathbf{\Delta}_*^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\|_{\mathrm{F}}^2 \\
&= \|\mathbf{A} - \Pi^{\mathrm{F}}_{\mathbf{R},k}(\mathbf{A})\|_{\mathrm{F}}^2 \\
&\le (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2
\end{aligned}$$

The third equality follows from Lemma 8 and the last inequality follows Lemma 14 (there is a failure probability 0.01 in this lemma). ∎

In words, the lemma indicates that $\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R} \in \mathbb{R}^{m\times n}$ is a rank k matrix which approximates the best rank k matrix of $\mathbf{A}$ up to relative error accuracy. Next, we present the main theorem.

Theorem 22. *The matrix $\mathbf{U} \in \mathbb{R}^{m\times k}$ with k orthonormal columns satisfies with probability at least 0.97:*

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \le (1+\varepsilon)\cdot\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

The running time of the algorithm is

$$O\left(mn\log(k\varepsilon^{-1}) + mk^2 + n\cdot\mathrm{poly}(k\varepsilon^{-1})\right).$$

Proof. We manipulate the term $\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2$ as follows:

$$\begin{aligned}
\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 &= \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*}(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*})^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \\
&\le \|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*}\mathbf{\Sigma}_{\mathbf{X}_*}\mathbf{V}_{\mathbf{X}_*}^{\mathrm{T}}\mathbf{W}\mathbf{A}\|_{\mathrm{F}}^2 \\
&\le \frac{(1+\varepsilon)^2}{(1-\varepsilon)^2}\|\mathbf{A} - \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}_{opt}\mathbf{R}\|_{\mathrm{F}}^2 \\
&\le \frac{(1+\varepsilon)^3}{(1-\varepsilon)^2}\cdot\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2
\end{aligned}$$

The first equality follows because $\mathbf{U} \in \mathbb{R}^{m\times k}$ is an orthonormal basis for $span(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*}) \in \mathbb{R}^k$. The first inequality follows from the fact that for any matrices $\mathbf{A}, \mathbf{C}, \mathbf{X}$ it is $\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \le \|\mathbf{A} - \mathbf{C}\mathbf{X}\|_{\mathrm{F}}^2$. The second inequality follows from Lemma 20 and the last inequality uses Lemma 21. The failure probability in the theorem follows from a simple union bound over the failure probabilities of Lemma 20 and Lemma 21. Note that $\frac{(1+\varepsilon)^3}{(1-\varepsilon)^2} \le 1+\varepsilon$, if ε is less than some constant $C < 1$.

Running time. Next, we analyze the running time of the algorithm:

1. This step requires $O(\mathrm{nnz}(\mathbf{A}) + \mathrm{poly}(k/\varepsilon))$ arithmetic operations.
2. $O(\xi_1 + \xi_2)$ arithmetic operations suffice for this step.
3. From Lemma 18, we need $O(mn\log(\xi_1))$ arithmetic operations to compute $\mathbf{T}_{left}\mathbf{A} := \mathbf{Z}$ and another $O(\xi_1 n\log(\xi_2))$ operations to compute $\mathbf{Z}\mathbf{T}_{right}$.

4. We already have $\mathbf{R} = \mathbf{WA}$ from the first step. Hence, $O(\xi n \log(\xi_2))$ additional arithmetic operations suffice to compute $\mathbf{RT}_{right}$, according to Lemma 18.

5. We already have $\mathbf{T}_{left}\mathbf{A} := \mathbf{Z}$ from the third step and $\mathbf{R}$ from the first step, hence $O(n \cdot \mathrm{poly}(k/\varepsilon))$ arithmetic operations suffice to compute $\mathbf{N}$ through a simple matrix multiplication.

6. This step requires $O(\mathrm{poly}(k/\varepsilon))$ operations since all matrices have dimensions $O(\mathrm{poly}(k/\varepsilon)) \times O(\mathrm{poly}(k/\varepsilon))$ and we use the method from Section 4.1.3.

7. This step requires $O(\mathrm{poly}(k/\varepsilon))$ operations since we compute the SVD of a $O(\mathrm{poly}(k/\varepsilon)) \times O(\mathrm{poly}(k/\varepsilon))$ matrix.

8. $O(mk^2)$ operations suffice to compute an orthonormal basis for $\mathbf{T}$, e.g., with a QR factorization.

■

4.3 The distributed PCA algorithm

Recall that the input matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is partitioned column-wise as: $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s \end{pmatrix}$; for $i = 1 : s$, $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$ is a column submatrix of $\mathbf{A}$; also $\sum_i w_i = n$. To describe the algorithm below in a convenient way, let us denote with $\hat{\mathbf{A}}_i \in \mathbb{R}^{m \times n}$ the matrix that contains $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$ in the column submatrix of $\mathbf{A}$ specified by the column indices in $\mathbf{A}_i$ and zeros elsewhere. (Using this notation, it is $\mathbf{A} = \sum_i \hat{\mathbf{A}}_i$.) This convention only helps in describing the algorithm in a simpler way; in an actual implementation, the matrices $\hat{\mathbf{A}}_i$ are never formed and used. The idea in the algorithm below is to implement the algorithm in Section 4.2 in the distributed setting.

Input:

1. $\mathbf{A} \in \mathbb{R}^{m \times n}$ partitioned column-wise $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s \end{pmatrix}$; for $i = 1 : s$, $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$; $\sum_i w_i = n$.
2. rank parameter $k < \mathrm{rank}(\mathbf{A})$
3. accuracy parameter $\varepsilon > 0$

Algorithm

1. Machines agree upon a low-rank preserving sketching matrix $\mathbf{W} \in \mathbb{R}^{\xi \times n}$ with $\xi = O(k\varepsilon^{-1})$ (see Lemma 12). Each machine now locally computes $\mathbf{R}_i = \mathbf{W}\hat{\mathbf{A}}_i \in \mathbb{R}^{\xi \times n}$. ($\mathbf{R} = \sum_i \mathbf{R}_i = \mathbf{WA} \in \mathbb{R}^{\xi \times n}$.)
2. Machines agree upon two affine embedding matrices $\mathbf{T}_{left} \in \mathbb{R}^{\xi_1 \times m}$ and $\mathbf{T}_{right} \in \mathbb{R}^{n \times \xi_2}$ (see Definition 16); $\xi_1 = O(k/\varepsilon^3)$ and $\xi_2 = O(k/\varepsilon^3)$.
3. Each machine locally computes $\mathbf{M}_i = \mathbf{T}_{left} \cdot \hat{\mathbf{A}}_i \cdot \mathbf{T}_{right} \in \mathbb{R}^{\xi_1 \times \xi_2}$ and sends $\mathbf{M}_i$ to the server. Server constructs $\mathbf{M} = \sum_i \mathbf{M}_i \in \mathbb{R}^{\xi_1 \times \xi_2}$. (Notice that $\mathbf{M} = \mathbf{T}_{left} \cdot \mathbf{A} \cdot \mathbf{T}_{right} \in \mathbb{R}^{\xi_1 \times \xi_2}$.)
4. Each machine locally computes $\mathbf{L}_i = \mathbf{R}_i \cdot \mathbf{T}_{right} \in \mathbb{R}^{\xi \times \xi_2}$ and sends $\mathbf{L}_i$ to the server. Server constructs $\mathbf{L} = \sum_{i=1}^{s} \mathbf{L}_i \in \mathbb{R}^{\xi \times \xi_2}$. (Notice that $\mathbf{L} = \mathbf{R} \cdot \mathbf{T}_{right} = \mathbf{WAT}_{right} \in \mathbb{R}^{\xi \times \xi_2}$.)
5. Each machine locally computes $\mathbf{F}_i = \hat{\mathbf{A}}_i\mathbf{R}^{\mathrm{T}} \in \mathbb{R}^{m \times \xi}$, then computes $\mathbf{N}_i = \mathbf{T}_{left}\mathbf{F}_i \in \mathbb{R}^{\xi_1 \times r}$ and sends $\mathbf{N}_i$ to the server. Server constructs $\mathbf{N} = \sum_{i=1}^{s} \mathbf{N}_i \in \mathbb{R}^{\xi_1 \times \xi}$. (Notice that $\mathbf{N} = \mathbf{T}_{left}\mathbf{AR}^{\mathrm{T}} \in \mathbb{R}^{\xi_1 \times \xi}$.)
6. Server sends $\mathbf{M}, \mathbf{N}, \mathbf{L}$ to all the machines and each machine finds $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X}) \le k}{\mathrm{argmin}} \|\mathbf{N} \cdot \mathbf{X} \cdot \mathbf{L} - \mathbf{M}\|_{\mathrm{F}}^2$, where $\mathbf{X}_* \in \mathbb{R}^{\xi \times \xi}$ has rank k (Notice that $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X}) \le k}{\mathrm{argmin}} \|\mathbf{T}_{left}\left(\mathbf{AR}^{\mathrm{T}}\mathbf{XR} - \mathbf{A}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2$.)

7. Each machine computes the SVD of $\mathbf{X}_*$ as $\mathbf{X}_* = \mathbf{U}_{\mathbf{X}_*}\mathbf{\Sigma}_{\mathbf{X}_*}\mathbf{V}_{\mathbf{X}_*}^{\mathrm{T}}$ ($\mathbf{U}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$, $\mathbf{\Sigma}_{\mathbf{X}_*} \in \mathbb{R}^{k\times k}$ $\mathbf{V}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$). Then, each machine computes $\mathbf{T}_i = \hat{\mathbf{A}}_i\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*} \in \mathbb{R}^{m\times k}$ and sends this to the server. Server computes $\mathbf{T} = \sum_i \mathbf{T}_i \in \mathbb{R}^{m\times k}$; (Notice that $\mathbf{T} = \mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*} \in \mathbb{R}^{m\times k}$.)
8. Server sends $\mathbf{T}$ to each machine; now each machine finds the *same* orthonormal basis $\mathbf{U} \in \mathbb{R}^{m\times k}$ for $span(\mathbf{T})$, e.g. with a QR factorization.

Remarks. Several remarks are necessary regarding the above algorithm. In the first step of the algorithm, in order the machines to "agree" upon the same sparse subspace embedding matrix $\mathbf{W}$ we use the implementation described in step 4-*b* of the algorithm in Section 8.2. In the second step of the algorithm, in order the machines to "agree" upon two SRHT matrices $\mathbf{T}_{left}$ and $\mathbf{T}_{right}$ a similar trick is needed: 1) every machine can generate deterministically the Walsh-Hadamard matrix in Definition 15; 2) the server generates $\mathbf{D}$ and $\mathbf{R}$ (of Definition 16) and sends them to all the machines. In the fifth step of the algorithm, every machine is able to compute $\mathbf{F}_i = \hat{\mathbf{A}}_i\mathbf{R}^{\mathrm{T}}$ since $\mathbf{F}_i = \hat{\mathbf{A}}_i\mathbf{R}^{\mathrm{T}} = \hat{\mathbf{A}}_i\sum_i \mathbf{R}_i^{\mathrm{T}} = \hat{\mathbf{A}}_i\mathbf{R}_i^{\mathrm{T}}$, and the outer product of $\hat{\mathbf{A}}_i$ with some $\mathbf{R}_i$ other than the "local" $\mathbf{R}_i$ is all-zeros. The same reasoning is applied to the seventh step of the algorithm when the local machine computes $\mathbf{T}_i$. In the sixth step of the algorithm, the machines use the algorithm in Section 4.1.3 to find $\mathbf{X}_*$.

4.3.1 Main result

The theorem below analyzes the approximation error, the communication complexity, and the running time of the previous algorithm. Notice that the communication cost of this algorithm is only given in terms of "real numbers". The only step where we can not bound the length of a machine word is when the machines communicate $\mathbf{T}_i$ to the server; and this is because the entries of $\mathbf{U}_{\mathbf{X}_*}$ could be unbounded (see the discussion regarding the upper bounds in Section 1.1.1). We resolve this issue in the following section.

Theorem 23. *The matrix $\mathbf{U} \in \mathbb{R}^{m\times k}$ with k orthonormal columns satisfies w.p. 0.97:*

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\cdot\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2. \tag{2}$$

The communication cost of the algorithm is

$$O(msk + s\cdot\mathrm{poly}(k/\varepsilon))$$

"real numbers" and the running time is of the order

$$O\left(mn\log(k\varepsilon^{-1}) + mk^2 + n\cdot\mathrm{poly}(k\varepsilon^{-1})\right).$$

Proof. The matrix $\mathbf{U}$ - up to the randomness in the algorithm - is exactly the same matrix as in the batch algorithm in Section 4.2, hence Theorem 22 proves Eqn. 2.

The algorithm communicates $O(msk + s\cdot\mathrm{poly}(k/\varepsilon))$ real numbers in total: $O(m+n)$ in step 2; $O(s\cdot\mathrm{poly}(k/\varepsilon))$ in steps 3,4,5, and 6, and $O(smk)$ in the last two steps.

The operations in the algorithm are effectively the same operations as in the batch algorithm in Section 4.2, hence the analysis of the running time in Theorem 22 shows the claim. ∎

5 Obtaining bit complexity for the distributed PCA algorithm

For the algorithm in the previous section, we were only able to provide a communication upper bound in terms of "real numbers". In this section, we describe how to obtain a communication upper bound in terms of words for the above protocol, where each word is $O(\log(mnsk/\varepsilon))$ bits.

The basic idea is that we have a case analysis depending on the rank of the matrix $\mathbf{A}$. If the rank of $\mathbf{A}$ is less or equal to $2k$, we follow one distributed protocol and if the rank is at least $2k$ we follow a different protocol. In Section 5.1, Section 5.2, and Section 5.3 we describe the algorithm that tests the rank of a distributed matrix, and the two PCA protocols, respectively. Then, in Section 5.4 we give the details of the overall algorithm and in Section 5.5 we give its analysis.

5.1 Testing the rank of a distributed matrix

Lemma 24. *Given $\mathbf{A} \in \mathbb{R}^{m\times n}$ and a rank parameter $k <$ rank$(\mathbf{A})$, there exists a distributed protocol in the model of Definition 1 to test if the rank of $\mathbf{A}$ is less than or equal to $2k$ using $O(sk^2)$ words of communication and succeeding with probability $1-\delta$ for an arbitrarily small constant $\delta > 0$.*

Proof. This is an immediate implementation of a streaming algorithm due to [18] for testing if an $n \times n$ matrix $\mathbf{A}$ has rank at least $2k$ in the streaming model, using $O(k^2)$ words of space. In that algorithm, there is a fixed $6nk/\delta \times n$ matrix $\mathbf{H}$ whose entries are integers of magnitude at most $\text{poly}(n)$, where $\delta > 0$ is an arbitrarily small constant. The algorithm simply chooses $4k$ random rows from $\mathbf{H}$. Letting $\mathbf{H}'$ be the $2k \times n$ matrix of the first $2k$ random rows, and $\mathbf{H}''$ be the $n \times 2k$ matrix whose columns are the next $2k$ randomly chosen rows, the algorithm just declares that $\mathbf{A}$ has rank at least k iff the rank of $\mathbf{H}'\mathbf{A}\mathbf{H}''$ is $2k$.

The above streaming algorithm can be implemented in the distributed setting by having the coordinator choose $4k$ random rows of the fixed, known matrix, and send the row identities to each of the machines. This only takes $O(sk)$ words of communication. Then machine i computes $\mathbf{H}'\mathbf{A}^i\mathbf{H}''$, and returns this to the coordinator. The coordinator can then add these up to compute $\mathbf{H}'\mathbf{A}\mathbf{H}''$ and compute its rank. The total communication is $O(sk^2)$ words and the protocol succeeds with probability at least $1-\delta$ for an arbitrarily small constant $\delta > 0$. Note that we can assume our input matrix $\mathbf{A}$, which is $m \times n$, is a square $n \times n$ matrix by padding with all-zeros rows. Those rows of course, will never get communicated in the implementation described above. ■

5.2 Distributed PCA protocol when rank$(\mathbf{A}) \leq 2k$

Lemma 25. *Suppose the rank ρ of $\mathbf{A} \in \mathbb{R}^{m\times n}$ satisfies $\rho \leq 2k$, for some rank parameter k. Then, there is a protocol for the Distributed Principal Component Analysis Problem of Definition 2 using $O(smk+sk^2/\varepsilon^2)$ words of communication and succeeding with probability $1-\delta$ for an arbitrarily small constant $\delta > 0$.*

Proof. The $2k\times m$ matrix $\mathbf{H}'$ chosen in the protocol of Lemma 24 satisfies that with probability $1-\delta$, for an arbitrarily small constant $\delta > 0$, the rank of $\mathbf{A}\mathbf{H}''$ is equal to the rank of $\mathbf{A}$ if $\rho < 2k$. Indeed, if this were not true, the algorithm could not be correct, as the rank of $\mathbf{H}'\mathbf{A}\mathbf{H}''$ is at most the rank of $\mathbf{A}\mathbf{H}''$ (and the same algorithm can be used for any $\rho < 2k$). It follows that with probability $1-\delta$, the column span of $\mathbf{A}\mathbf{H}''$ is equal to the column span of $\mathbf{A}$. Hence, as in the protocol of Lemma 24, the coordinator learns the column space of $\mathbf{A}$, which can be described with $O(km)$ words. The coordinator thus communicates this to all machines, using $O(skm)$ total words of communication, assuming the entries of $\mathbf{A}$ are integers of magnitude at most $\text{poly}(mns/\varepsilon)$.

Let $\mathbf{C} = \mathbf{A}\mathbf{H}''$, which is $m \times 2k$. We can set up the optimization problem:

$$\min_{\text{rank}(\mathbf{X})\leq k} \|\mathbf{C}\mathbf{X}\mathbf{C}^{\mathrm{T}}\mathbf{A} - \mathbf{A}\|_{\mathrm{F}}.$$

Since the rank of $\mathbf{C}$ and $\mathbf{A}$ are small, we can sketch on the left and right using affine embeddings $\mathbf{T}_{left}$ and $\mathbf{T}_{right}$. Then machine i sends $\mathbf{C}^{\mathrm{T}}\mathbf{A}^i\mathbf{T}_{right}$ to the coordinator, together with $\mathbf{T}_{left}\mathbf{A}^i\mathbf{T}_{right}$. The coordinator computes $\mathbf{C}^{\mathrm{T}}\mathbf{A}\mathbf{T}_{right}$ and $\mathbf{T}_{left}\mathbf{A}\mathbf{T}_{right}$ by adding up the sketches, and sends these back to all the machines. Each machine can then solve the optimization problem

$$\min_{\mathrm{rank}(\mathbf{X})\leq k} \|\mathbf{T}_{left}\mathbf{C}\mathbf{X}\mathbf{C}^{\mathrm{T}}\mathbf{A}\mathbf{T}_{right} - \mathbf{T}_{left}\mathbf{A}\mathbf{T}_{right}\|_{\mathrm{F}},$$

each obtaining the same $\mathbf{X}_*$. Finally, every machine outputs the same orthonormal basis $\mathbf{U} \in \mathbb{R}^{m\times k}$ for $\mathbf{C}\mathbf{X}_*$.

We can construct affine embedding matrices $\mathbf{T}_{left} \in \mathbb{R}^{\xi_1\times m}$ and $\mathbf{T}_{right} \in \mathbb{R}^{n\times\xi_2}$ with $\xi_1 = O(k/\varepsilon^2)$, $\xi_2 = O(k/\varepsilon^2)$ (see Definition 16). The total communication of this protocol is $O(skm + sk^2/\varepsilon^2)$ words and the success probability can be made $1-\delta$ for an arbitrarily small constant $\delta > 0$. ■

5.3 Distributed PCA protocol when $\mathrm{rank}(\mathbf{A}) > 2k$

The idea here is more involved than in the previous subsection and in order to describe the algorithm we need several intermediate results.

5.3.1 Lower bounds on singular values of matrices with integers entries

The first lemma gives a lower bound on the singular values of a matrix with integer entries with bounded magnitude.

Lemma 26. *(Lemma 4.1 of [18], restated) If an $m \times n$ matrix $\mathbf{A}$ has integer entries bounded in magnitude by γ, and has rank $\rho = \mathrm{rank}(\mathbf{A})$, then the k-th largest singular value $\mathbf{A}$ satisfies*

$$\sigma_k \geq (mn\gamma^2)^{-k/(2(\rho-k))}.$$

Proof. In the proof of Lemma 4.1 of [18], equation (10), it is shown that if λ_k is the k-th largest eigenvalue of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$, then

$$\lambda_k \geq (mn\gamma^2)^{-k/(\rho-k)}.$$

This implies the k-th singular value σ_k of $\mathbf{A}$ satisfies $\sigma_k \geq (mn\gamma^2)^{-k/(2(\rho-k))}$. ■

Next, we state two immediate corollaries of this lemma for future reference.

Corollary 27. *If an $m \times n$ matrix $\mathbf{A}$ has integer entries bounded in magnitude by $\mathrm{poly}(mns/\varepsilon)$ and $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} > 0$, then*

$$\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} > (mns/\varepsilon)^{-O(k)}.$$

Proof. Since $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} > 0$, the rank ρ of $\mathbf{A}$ is at least $k+1$, so by the preceding lemma,

$$\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} \geq \sigma_{k+1} \geq (\mathrm{poly}(mns/\varepsilon))^{-k/2},$$

as desired. ■

Corollary 28. *If an $m \times n$ matrix $\mathbf{A}$ has integer entries bounded in magnitude by $\mathrm{poly}(mns/\varepsilon)$ and rank$(\mathbf{A}) \geq 2k$, then*

$$\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} > 1/\mathrm{poly}(mns/\varepsilon).$$

Proof. This follows by plugging $\rho = 2k$ into Lemma 26. ■

5.3.2 Lower bounds on singular values of integer-perturbed matrices

In this section, we describe a perturbation technique for matrices and provide lower bounds for the smallest singular value of the perturbed matrix. We start with a theorem of Tao and Vu.

Theorem 29. *(Theorem 2.5 of [47]) Let $\mathbf{M}$ be an $n \times n$ matrix with integer entries bounded in magnitude by n^C for a constant $C > 0$. Let $\mathbf{N}_n$ be a matrix with independent entries each chosen to be 1 with probability $1/2$, and -1 with probability $1/2$. Then, there is a constant $B > 0$ depending on C, for which*

$$\Pr[\|(\mathbf{M} + \mathbf{N}_n)^{-1}\|_2 \geq n^B] \leq 1/n.$$

In words, the result indicates that the spectral norm of the inverse of the perturbed matrix is bounded from below with high probability. We now describe a simple corollary of this result.

Corollary 30. *Let $\mathbf{M}$ be an $n \times n$ matrix with integer entries bounded in magnitude by n^C for a constant $C > 0$. Let $\mathbf{N}_n$ be a matrix with independent entries each chosen to be $1/n^D$ with probability $1/2$, and $-1/n^D$ with probability $1/2$, where $D > 0$ is a constant. Then, there is a constant $B > 0$ depending on C and D for which*

$$\Pr[\|(\mathbf{M} + \mathbf{N}_n)^{-1}\|_2 \geq n^B] \leq 1/n.$$

Proof. This follows by Theorem 29 after replacing the constat C in that theorem with $C + D$, and scaling by n^D. ∎

We need to generalize Corollary 30 to rectangular matrices since we will eventually apply this perturbation technique to the matrix $\mathbf{A}$ to which we would like to compute a distributed PCA.

Lemma 31. *Let $\mathbf{M}$ be an $m \times n$ matrix with integer entries bounded in magnitude by n^C for a constant $C > 0$, and suppose $m \leq n$. Let $\mathbf{N}_{m,n}$ be a matrix with independent entries each chosen to be $1/n^D$ with probability $1/2$ and $-1/n^D$ with probability $1/2$, where $D > 0$ is a constant. Then, there is a constant $B > 0$ depending on C and D for which*

$$\Pr[\sigma_m(\mathbf{M} + \mathbf{N}_{m,n}) \geq 1/n^B] \leq 1/n,$$

where $\sigma_m(\mathbf{M} + \mathbf{N}_{m,n})$ denotes the smallest singular value of $\mathbf{M} + \mathbf{N}_{m,n}$.

Proof. Suppose we were to pad $\mathbf{M}$ with $n - m$ zero rows, obtaining a square matrix $\mathbf{M}$. Now consider the $n \times n$ matrix $\mathbf{N}_{n,n}$ with independent entries each chosen to be $1/n^D$ with probability $1/2$ and $-1/n^D$ with probability $1/2$. By Corollary 30, all singular values of $\mathbf{M} + \mathbf{N}_{n,n}$ are at least $1/n^B$. Now consider a unit vector $\mathbf{x} \in \mathbb{R}^n$ which is zero on all but its top m coordinates. Let $\mathbf{y} \in \mathbb{R}^m$ be the unit vector which agrees with $\mathbf{x}$ on its top m coordinates. Then,

$$\begin{aligned} 1/n^B &\leq \|\mathbf{x}(\mathbf{M} + \mathbf{N}_{n,n})\|_2 \\ &= \|\mathbf{y}(\mathbf{M} + \mathbf{N}_{m,n})\|_2, \end{aligned}$$

where the inequality uses the lower bound on the singular values of $\mathbf{M} + \mathbf{N}_{n,n}$, which occurs with probability at least $1 - 1/n$, and the equality follows by definition of the matrices and vectors we have defined. As $\mathbf{y} \in \mathbb{R}^m$ can be chosen to be an arbitrary unit vector, it follows that $\sigma_m(\mathbf{M} + \mathbf{N}_{m,n}) \geq 1/n^B$, which completes the proof. ∎

5.4 Description of algorithm

Using the above results, we are now ready to describe a distributed PCA algorithm whose communication cost can be bounded in terms of machine words and not just in terms of "real numbers". As in the algorithm in Section 4.3, we denote with $\hat{\mathbf{B}}_i \in \mathbb{R}^{m\times n}$ the matrix that contains some matrix $\mathbf{B}_i \in \mathbb{R}^{m\times w_i}$ in the column submatrix of $\mathbf{B}$ specified by the column indices in $\mathbf{B}_i$ and zeros elsewhere. Specifically, the matrices $\mathbf{B}_i$ arise from $\mathbf{A}_i$ after applying the Bernoulli perturbation technique discussed above in Lemma 31. Using this notation, we also have $\mathbf{B} := \sum_i \hat{\mathbf{B}}_i \in \mathbb{R}^{m\times n}$.

Input:

1. $\mathbf{A} \in \mathbb{R}^{m\times n}$ partitioned column-wise $\mathbf{A} = \begin{pmatrix}\mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s\end{pmatrix}$; for $i = 1 : s$, $\mathbf{A}_i \in \mathbb{R}^{m\times w_i}$; $\sum_i w_i = n$.
2. rank parameter $k <$ rank$(\mathbf{A})$
3. accuracy parameter $\varepsilon > 0$

Algorithm

1. Use the protocol of Lemma 24 with $\delta = 0.01$ to test if the rank of $\mathbf{A}$ is less than $2k$.
2. If rank$(\mathbf{A}) \le 2k$, use the protocol of Lemma 25 to find some orthonormal $\mathbf{U} \in \mathbb{R}^{m\times k}$.
3. If rank$(\mathbf{A}) > 2k$,
   (a) Each machine i locally and independently adds $1/n^D$ with probability $1/2$, and $-1/n^D$ with probability $1/2$, to each of the entries in the columns in its matrix $\mathbf{A}_i$, where D is the constant of Lemma 31. Note that this effectively adds the matrix $\mathbf{N}_{m,n}$ of Lemma 31 to the entire matrix $\mathbf{A}$. For notational convenience let $\mathbf{B} = \mathbf{A} + \mathbf{N}_{m,n}$, and $\mathbf{B}_i \in \mathbb{R}^{m\times w_i}$ be the local perturbed matrix.
   (b) Machines agree upon a low-rank preserving sketching matrix $\mathbf{W} \in \mathbb{R}^{\xi\times n}$ with $\xi = O(k\varepsilon^{-1})$ (see Lemma 12). Each machine now computes $\mathbf{R}_i = \mathbf{W}\hat{\mathbf{B}}_i \in \mathbb{R}^{\xi\times n}$. ($\mathbf{R} = \sum_i \mathbf{R}_i = \mathbf{WB} \in \mathbb{R}^{\xi\times n}$.)
   (c) Machines agree upon two affine embedding matrices $\mathbf{T}_{left} \in \mathbb{R}^{\xi_1\times m}$ and $\mathbf{T}_{right} \in \mathbb{R}^{n\times\xi_2}$ (see Definition 16); $\xi_1 = O(k/\varepsilon^3)$ and $\xi_2 = O(k/\varepsilon^3)$.
   (d) Each machine locally computes $\mathbf{M}_i = \mathbf{T}_{left}\cdot\hat{\mathbf{B}}_i\cdot\mathbf{T}_{right} \in \mathbb{R}^{\xi_1\times\xi_2}$ and sends $\mathbf{M}_i$ to the server. Server constructs $\mathbf{M} = \sum_i \mathbf{M}_i \in \mathbb{R}^{\xi_1\times\xi_2}$. (Notice that $\mathbf{M} = \mathbf{T}_{left}\cdot\mathbf{B}\cdot\mathbf{T}_{right} \in \mathbb{R}^{\xi_1\times\xi_2}$.)
   (e) Each machine locally computes $\mathbf{L}_i = \mathbf{R}_i\cdot\mathbf{T}_{right} \in \mathbb{R}^{\xi\times\xi_2}$ and sends $\mathbf{L}_i$ to the server. Server constructs $\mathbf{L} = \sum_{i=1}^{s}\mathbf{L}_i \in \mathbb{R}^{\xi\times\xi_2}$. (Notice that $\mathbf{L} = \mathbf{R}\cdot\mathbf{T}_{right} = \mathbf{WBT}_{right} \in \mathbb{R}^{\xi\times\xi_2}$.)
   (f) Each machine locally computes $\mathbf{F}_i = \hat{\mathbf{B}}_i\mathbf{R}^{\mathrm{T}} \in \mathbb{R}^{m\times\xi}$, then computes $\mathbf{N}_i = \mathbf{T}_{left}\mathbf{F}_i \in \mathbb{R}^{\xi_1\times r}$ and sends $\mathbf{N}_i$ to the server. Server constructs $\mathbf{N} = \sum_{i=1}^{s}\mathbf{N}_i \in \mathbb{R}^{\xi_1\times\xi}$. ($\mathbf{N} = \mathbf{T}_{left}\mathbf{BR}^{\mathrm{T}} \in \mathbb{R}^{\xi_1\times\xi}$.)
   (g) Server sends $\mathbf{M}, \mathbf{N}, \mathbf{L}$ to all the machines and each machine finds $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X})\le k}{\mathrm{argmin}} \|\mathbf{N}\cdot\mathbf{X}\cdot\mathbf{L} - \mathbf{M}\|_{\mathrm{F}}^2$, where $\mathbf{X}_* \in \mathbb{R}^{\xi\times\xi}$ has rank k. Notice that $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X})\le k}{\mathrm{argmin}} \|\mathbf{T}_{left}\left(\mathbf{AR}^{\mathrm{T}}\mathbf{XR} - \mathbf{A}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2$. Specifically, each machine computes three factors, $\mathbf{X}_* = \underbrace{\mathbf{N}^{\dagger}}_{\mathbf{\Phi}_1}\cdot\underbrace{\left(\mathbf{U}_{\mathbf{N}}\mathbf{U}_{\mathbf{N}}^{\mathrm{T}}\mathbf{M}\mathbf{U}_{\mathbf{L}}\mathbf{U}_{\mathbf{L}}^{\mathrm{T}}\right)_k}_{\mathbf{\Phi}_2}\cdot\underbrace{\mathbf{L}^{\dagger}}_{\mathbf{\Phi}_3}$ Furthermore, each machine computes the rank k SVD of $\mathbf{\Phi}_2$ as $\mathbf{\Phi}_2 = \mathbf{U}_{\mathbf{\Phi}_2}\mathbf{\Sigma}_{\mathbf{\Phi}_2}\mathbf{V}_{\mathbf{\Phi}_2}^{\mathrm{T}}$. Now each machine rounds each of the entries in each of the above five matrices to the nearest integer multiple of $1/n^{\gamma}$ for a sufficiently large constant $\gamma > 0$. Let the matrices after the rounding be $\tilde{\mathbf{\Phi}}_1, \tilde{\mathbf{U}}_{\mathbf{\Phi}_2}, \tilde{\mathbf{\Sigma}}_{\mathbf{\Phi}_2}, \tilde{\mathbf{V}}_{\mathbf{\Phi}_2}, \tilde{\mathbf{\Phi}}_3$. Each machine finds $\mathbf{Y}_* = \tilde{\mathbf{\Phi}}_1\cdot\tilde{\mathbf{U}}_{\mathbf{\Phi}_2}\cdot\tilde{\mathbf{\Sigma}}_{\mathbf{\Phi}_2}\cdot\tilde{\mathbf{V}}_{\mathbf{\Phi}_2}^{\mathrm{T}}\cdot\tilde{\mathbf{\Phi}}_3 \in \mathbb{R}^{\xi_1\times\mathbf{x}_2}$ of rank k.
   (h) Each machine finds the same matrix $\mathbf{D} \in \mathbb{R}^{\xi_1\times k}$ with k linearly independent columns of $\mathbf{Y}_*$. Then, each machine computes $\mathbf{T}_i = \hat{\mathbf{B}}_i\mathbf{R}^{\mathrm{T}}\mathbf{D} \in \mathbb{R}^{m\times k}$ and sends this to the server. Server computes $\mathbf{T} = \sum_i \mathbf{T}_i \in \mathbb{R}^{m\times k}$; (Notice that $\mathbf{T} = \mathbf{BR}^{\mathrm{T}}\mathbf{D}$.)
   (i) Server sends $\mathbf{T}$ to each machine; now each machine finds the *same* orthonormal basis $\mathbf{U} \in \mathbb{R}^{m\times k}$ for $span(\mathbf{T})$, e.g. with a QR factorization.

Remarks. Several remarks are necessary regarding the above algorithm. In step $3-b$ of the algorithm, in order the machines to "agree" upon the same sparse subspace embedding matrix $\mathbf{W}$ we use the implementation described in step 4-b of the algorithm in Section 8.2. In step $3-c$ of the algorithm, in order the machines to "agree" upon two SRHT matrices $\mathbf{T}_{left}$ and $\mathbf{T}_{right}$ a similar trick is needed: 1) every machine can generate deterministically the Walsh-Hadamard matrix in Definition 15; 2) the server generates $\mathbf{D}$ and $\mathbf{R}$ (of Definition 16) and sends them to all the machines. In step $3-f$ of the algorithm, every machine is able to compute $\mathbf{F}_i = \hat{\mathbf{B}}_i\mathbf{R}^{\mathrm{T}}$ since

$$\mathbf{F}_i = \hat{\mathbf{B}}_i\mathbf{R}^{\mathrm{T}} = \hat{\mathbf{B}}_i \sum_i \mathbf{R}_i^{\mathrm{T}} = \hat{\mathbf{B}}_i\mathbf{R}_i^{\mathrm{T}},$$

and the outer product of $\hat{\mathbf{B}}_i$ with some $\mathbf{R}_i$ other than the "local" $\mathbf{R}_i$ is all-zeros. The same reasoning is applied to the step 3-h of the algorithm when the local machine computes $\mathbf{T}_i$. In step $3-h$, machines can find the *same* set of linear independent columns after agreeing to the same algorithm, e.g. some Pivoted QR factorization [28].

5.5 Main result

The theorem below analyzes the approximation error, the communication complexity, and the running time of the previous algorithm. Notice that the communication cost of this algorithm is given in terms of machine words.

Theorem 32. *The matrix $\mathbf{U} \in \mathbb{R}^{m\times k}$ with k orthonormal columns satisfies with arbitrarily large constant probability:*

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\cdot\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2. \tag{3}$$

The communication cost of the algorithm is $O(msk + s\cdot\mathrm{poly}(k/\varepsilon))$ words and the running time is $O\left(mn\log(k\varepsilon^{-1}) + mk^2 + n\cdot\mathrm{poly}(k\varepsilon^{-1})\right)$.

5.6 Proof of Theorem 32

Proof of Eqn. 3 If $\mathrm{rank}(\mathbf{A}) \leq 2k$, then Lemma 25 shows the claim. If $\mathrm{rank}(\mathbf{A}) > 2k$, then the situation is more involved and we prove the approximation bound in the following subsection.

Communication Complexity Step 1 requires $O(sk^2)$ words (Lemma 24). Step 2 requires $O(smk + sk^2/\varepsilon^2)$ words (Lemma 25). Step 3 requires $O(msk + s\cdot\mathrm{poly}(k/\varepsilon))$ words and this follows from the analysis in Theorem 23. The only difference with Theorem 23 is that now all matrices communicated in the algorithm have real numbers which can be represented efficiently with one machine word of at most $O(\log(mns/\varepsilon)$ bits. To see this, note that each machine locally computes $\mathbf{Y}_*$, which is of rank k, and just chooses k linearly independent columns of this matrix, which is the matrix denoted by $\mathbf{D}$, each of whose entries are integer multiples of $1/n^{\gamma}$ and bounded in magnitude by $n^{O(B+C)/\gamma}$. Therefore, the entries of $\mathbf{D}$ can each be described using $O(\log n)$ bits, i.e., a constant number of machine words.

Running time The operations in the third step in algorithm are effectively the same operations as in the batch algorithm in Section 4.2 (plus some extra operations whose running time is not asymptotically larger), hence the analysis of the running time in Theorem 22 shows the claim (the number of operations in the first two steps of the algorithm is asymptotically less than the number of operations in the third step).

5.7 Proof of Eqn. 3 if $\text{rank}(\mathbf{A}) > 2k$

To proceed, we need to give the details of the construction of the matrix $\mathbf{W}$ in step 3-b of the algorithm. From Lemma 12, $\mathbf{W} \in \mathbb{R}^{\xi \times m}$ is constructed as follows, $\mathbf{W} = \mathbf{ST}$, where, for $\xi = O(k/\epsilon)$ and $\xi_2 = O(k^2 + k/\epsilon)$, $\mathbf{S}$ is a $\xi \times \xi_2$ matrix of i.i.d. normal random variables with variance $1/\xi$, and $\mathbf{T}$ is a $\xi_2 \times m$ sparse subspace embedding matrix. Recall that in the above algorithm, we have manipulated the following matrices in the various stages of the algorithm,

$$\mathbf{N} = \mathbf{T}_{left} \cdot \mathbf{B} \cdot \mathbf{B}^{\mathrm{T}} \cdot \mathbf{T}^{\mathrm{T}} \cdot \mathbf{S}^{\mathrm{T}},$$

$$\mathbf{L} = \mathbf{S} \cdot \mathbf{T} \cdot \mathbf{B} \cdot \mathbf{T}_{right},$$

$$\mathbf{M} = \mathbf{T}_{left} \cdot \mathbf{B} \cdot \mathbf{T}_{right},$$

$$\mathbf{X}_* = \mathbf{N}^{\dagger}(\mathbf{U_N}\mathbf{U_N}^{\dagger}\mathbf{B}\mathbf{V_L}\mathbf{V_L}^{\mathrm{T}})_k\mathbf{L}^{\dagger},$$

$$\mathbf{\Phi}_1 = \mathbf{N}^{\dagger},$$

$$\mathbf{\Phi}_2 = (\mathbf{U_N}\mathbf{U_N}^{\dagger}\mathbf{B}\mathbf{V_L}\mathbf{V_L}^{\mathrm{T}})_k,$$

and

$$\mathbf{\Phi}_3 = \mathbf{L}^{\dagger}.$$

Note also that $\mathbf{\Phi}_2$ is a rank k matrix whose SVD is $\mathbf{\Phi}_2 = \mathbf{U}_{\mathbf{\Phi}_2}\mathbf{\Sigma}_{\mathbf{\Phi}_2}\mathbf{V}_{\mathbf{\Phi}_2}^{\mathrm{T}}$, with $\mathbf{\Sigma}_{\mathbf{\Phi}_2} \in \mathbb{R}^{k \times k}$. To proceed, we need a sequence of lemmas bounding the spectral norm of some of the above matrices.

Lemma 33. *With arbitrarily large constant probability, $\|\mathbf{\Phi}_1\|_2 \leq 4n^{2B}$.*

Proof. Recall that $\mathbf{\Phi}_1 = \mathbf{N}^{\dagger}$. We have $\mathbf{N} = \mathbf{T}_{left} \cdot \mathbf{B} \cdot \mathbf{B}^{\mathrm{T}} \cdot \mathbf{T}^{\mathrm{T}} \cdot \mathbf{S}^{\mathrm{T}}$. Since $\mathbf{T}_{left}$ is an affine subspace embedding, we have with arbitrarily large constant probability, $\sigma_{min}(\mathbf{C}) \geq \frac{1}{2}\sigma_{min}(\mathbf{B}\mathbf{B}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}})$, where σ_{min} denotes the minimum singular value. Writing $\mathbf{B} = \mathbf{U_B}\mathbf{\Sigma_B}\mathbf{V_B}^{\mathrm{T}}$ in its SVD, we have $\mathbf{B}\mathbf{B}^{\mathrm{T}} = \mathbf{U_B}\mathbf{\Sigma_B}^2\mathbf{U_B}^{\mathrm{T}}$. Using Lemma 31, with probability $1 - 1/n$, all singular values of $\mathbf{B}$ are at least $1/n^B$. Further, since $m \leq n$, $\mathbf{U_B}$ is a square matrix with orthonormal rows and columns. Therefore,

$$\sigma_{min}(\mathbf{B}\mathbf{B}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}) \geq \frac{1}{2n^{2B}}\sigma_{min}(\mathbf{T}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}).$$

By standard properties of rectangular Gaussian matrices $\mathbf{S}^{\mathrm{T}}$, we have $\sigma_{min}(\mathbf{G}) \geq 1/2$, with probability $1 - 2^{-\Theta(k)}$, and so $\sigma_{min}(\mathbf{B}\mathbf{B}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}) \geq \frac{1}{4n^{2B}} \cdot \sigma_{min}(\mathbf{T}^{\mathrm{T}})$. But $\mathbf{T}^{\mathrm{T}}$ is just a sparse subspace embedding, which has orthogonal columns, where the squared norm of each column j is the number of $i \in [m]$ for which the single non-zero entry in row i of $\mathbf{T}^{\mathrm{T}}$ is in the j-th column. Since these locations are chosen independently for $i \in [m]$, the expected squared norm of a column is m/c, where $c = \Theta(k^2/\varepsilon^2)$ is the number of columns of $\mathbf{T}^{\mathrm{T}}$. By a Chernoff and a union bound, with probability $1 - 2^{-\Theta(m\varepsilon^2/k^2)}$, each squared column norm is at least $m/(2c) \geq 1$. Therefore, $\sigma_{min}(\mathbf{T}^{\mathrm{T}}) \geq 1$ with this probability.

We conclude $\sigma_{min}(\mathbf{B}\mathbf{B}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}) \geq \frac{1}{4n^{2B}}$ with arbitrarily large constant probability (taking a union bound over all events in the proof). This implies $\|\mathbf{N}^{\dagger}\|_2 \leq 4n^{2B}$. ■

Lemma 34. *With arbitrarily large constant probability, $\|\mathbf{\Phi}_3\|_2 \leq 4n^B$.*

Proof. The proof is similar to that of Lemma 33 and we omit the details. ■

Lemma 35. *With arbitrarily large constant probability, $\|\mathbf{\Sigma}_{\mathbf{\Phi}_2}\|_2 \leq n^{1+C}$.*

Proof. We have

$$\begin{aligned}\|\mathbf{\Sigma}_{\Phi_2}\|_2 = \|\mathbf{\Phi}_2\|_2 &= \|\mathbf{U_N U_N^\dagger B V_L V_L^T}\|_2 \\ &\leq \|\mathbf{U_N U_N^\dagger}\|_2 \cdot \|\mathbf{B}\|_2 \cdot \|\mathbf{V_L V_L^T}\|_2 \\ &\leq \|\mathbf{B}\|_2 \\ &\leq \|\mathbf{B}\|_{\mathrm{F}} \\ &\leq n^C \cdot n \\ &= n^{C+1},\end{aligned}$$

where the first inequality uses submultiplicativity, and the final inequality that all entries of $\mathbf{B}$ are bounded in magnitude by n^C. ∎

Now, recall that in our protocol each machine computes $\mathbf{X}_*$, and its factors

$$\mathbf{X}_* = \mathbf{\Phi}_1 \cdot \mathbf{U}_{\mathbf{\Phi}_2} \cdot \mathbf{\Sigma}_{\mathbf{\Phi}_2} \cdot \mathbf{V}_{\mathbf{\Phi}_2}^{\mathrm{T}} \cdot \mathbf{\Phi}_3.$$

$\mathbf{Y}_*$ of rank k is the matrix obtained by rounding each of the entries in each of the above five factors to the nearest integer multiple of $1/n^\gamma$ for a sufficiently large constant $\gamma > 0$. By Lemma 33, Lemma 34, and Lemma 35, it follows that $\mathbf{Y}_* = \mathbf{X}_* + \mathbf{E}$, where $\mathbf{E}$ is a matrix all of whose entries are bounded in magnitude by $n^{O(B+C)/\gamma}$, and so by choosing $\gamma > 0$ to be a sufficiently large constant, this is at most $1/n^{\gamma/2}$. We manipulate the term $\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}$ as follows:

$$\begin{aligned}\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}} &= \|\mathbf{B} - \mathbf{N}_{m,n} - \mathbf{U}\mathbf{U}^{\mathrm{T}}(\mathbf{B} - \mathbf{N}_{m,n})\|_{\mathrm{F}} \\ &\leq \|\mathbf{B} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{B}\|_{\mathrm{F}} + \|(\mathbf{I}_m - \mathbf{U}\mathbf{U}^{\mathrm{T}})\mathbf{N}_{m,n}\|_{\mathrm{F}} \\ &\leq \|\mathbf{B} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{B}\|_{\mathrm{F}} + \|\mathbf{N}_{m,n}\|_{\mathrm{F}} \\ &= \|\mathbf{B} - \mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{D}(\mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{D})^\dagger\mathbf{B}\|_{\mathrm{F}}^2 + \|\mathbf{N}_{m,n}\|_{\mathrm{F}} \\ &= \|\mathbf{B} - \mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{Y}_*}(\mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{Y}_*})^\dagger\mathbf{B}\|_{\mathrm{F}}^2 + \|\mathbf{N}_{m,n}\|_{\mathrm{F}} \\ &\leq \|\mathbf{B} - \mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{Y}_*}\mathbf{\Sigma}_{\mathbf{Y}_*}\mathbf{V}_{\mathbf{Y}_*}^{\mathrm{T}}\mathbf{W}\mathbf{B}\|_{\mathrm{F}}^2 + \|\mathbf{N}_{m,n}\|_{\mathrm{F}} \\ &= \|\mathbf{B} - \mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{Y}_*\mathbf{R}\|_{\mathrm{F}} + \|\mathbf{N}_{m,n}\|_{\mathrm{F}} \\ &\leq \|\mathbf{B} - \mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{X}_*\mathbf{R}\|_{\mathrm{F}} + \|\mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{E}\mathbf{R}\|_{\mathrm{F}} + \|\mathbf{N}_{m,n}\|_{\mathrm{F}} \\ &\leq (1+\varepsilon)\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}} + \|\mathbf{B}\|_{\mathrm{F}}\|\mathbf{R}^{\mathrm{T}}\|_{\mathrm{F}}\|\mathbf{E}\|_{\mathrm{F}}\|\mathbf{R}\|_{\mathrm{F}} + \|\mathbf{N}_{m,n}\|_{\mathrm{F}} \\ &\leq (1+3\varepsilon)\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}},\end{aligned}$$

The first equality follows by using the relation $\mathbf{B} = \mathbf{A} + \mathbf{N}_{m,n}$. The first inequality uses the triangle inequality for the Frobenius norm. The second inequality uses the fact that $\mathbf{I}_m - \mathbf{U}\mathbf{U}^{\mathrm{T}}$ is a projector matrix and can be dropped without increasing the Frobenius norm. The second equality uses the fact that $\mathbf{U}$ is a basis of $\mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{D}$. The third equality uses the fact that the span of $\mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{D}$ equals the span of $\mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{U}_{\mathbf{Y}_*}$. The third inequality uses the fact that for any matrices $\mathbf{A}, \mathbf{C}, \mathbf{X}$ it is $\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{C}\mathbf{X}\|_{\mathrm{F}}^2$. The fourth equality simply uses the SVD expansion of $\mathbf{Y}_*$. The fourth inequality uses the relation $\mathbf{Y}_* = \mathbf{X}_* + \mathbf{E}$ and the triangle inequality. The fifth inequality uses the relation $\|\mathbf{B} - \mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{X}_*\mathbf{R}\|_{\mathrm{F}} \leq (1+\varepsilon)\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}}$ which follows by Lemma 20 applied on $\mathbf{B}$. The sixth inequality uses that $\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}} \geq 1/\mathrm{poly}(mns/\epsilon)$ (follows from Lemma 31) while $\|\mathbf{B}\|_{\mathrm{F}}, \|\mathbf{R}\|_{\mathrm{F}} \leq \mathrm{poly}(mn)$, yet $\|\mathbf{E}\|_{\mathrm{F}}$ and $\|\mathbf{N}_{m,n}\|_{\mathrm{F}}$ can be made $1/n^t$ for an arbitrarily large integer t.

Overall, after rescaling ε, we have

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}} \leq (1+\varepsilon)\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}}. \tag{4}$$

Finally, we need to relate $\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}}$ to $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}$, which we do in the following lemma.

Lemma 36. *Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be any matrix, $\mathbf{N}_{m,n}$ be the random matrix of Lemma 31, and $\mathbf{B} = \mathbf{A} + \mathbf{N}_{m,n}$. Let $k < \text{rank}(\mathbf{A}), \varepsilon > 0$. Then, for arbitrarily large constant probability,*

$$\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}} \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}.$$

Proof. Since $\mathbf{B} = \mathbf{A} + \mathbf{N}_{m,n}$, the proof idea is to relate the singular values of $\mathbf{B}$ to the singular values of $\mathbf{A}$ using Weyl's inequality [3]. Specifically, for $i : 1 : m$ (recall we assume $m \leq n$) we have

$$|\sigma_i(\mathbf{B}) - \sigma_i(\mathbf{A})| \leq \|\mathbf{N}_{m,n}\|_2 \leq 1/n^D.$$

Rearranging terms in this inequality, and taking squares of the resulting relation we obtain (for $i = 1 : m$),

$$\sigma_i^2(\mathbf{B}) \leq \left(\sigma_i(\mathbf{A}) + 1/n^D\right)^2. \tag{5}$$

We manipulate the term $\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}}$ as follows:

$$\begin{aligned}
\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}} &= \sqrt{\sum_{i=k+1}^{m} \sigma_i^2(\mathbf{B})} \\
&\leq \sqrt{\sum_{i=k+1}^{m} (\sigma_i(\mathbf{A}) + 1/n^D)^2} \\
&= \sqrt{\sum_{i=k+1}^{m} \left(\sigma_i^2(\mathbf{A}) + 1/n^{2D} + 2\sigma_i(\mathbf{A})/n^D\right)} \\
&\leq \sqrt{\sum_{i=k+1}^{m} \sigma_i^2(\mathbf{A})} + \sqrt{1/n^{2D}} + \sqrt{2\sigma_i(\mathbf{A})/n^D} \\
&\leq \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} + \varepsilon \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} + \varepsilon \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} \\
&\leq (1+2\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}.
\end{aligned}$$

The first inequality uses Eqn. (5). In the third inequality, we used

$$\sqrt{1/n^{2D}} \leq \varepsilon \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}},$$

which follows from Corollary 28 for a sufficiently large constant D. Also, we used

$$\sqrt{2\sigma_i(\mathbf{A})/n^D} \leq \sqrt{2\sigma_1(\mathbf{A})/n^D} \leq \sqrt{\text{poly}(nms/\varepsilon)/n^D} \leq \varepsilon \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}},$$

where the second inequality follows because $\|\mathbf{A}\|_2 \leq \text{poly}(nms/\varepsilon)$ and the last inequality uses again Corollary 28 for a sufficiently large constant D. ∎

Completing the proof. Using Lemma 36 in Eqn (4), taking squares in the resulting inequality, and rescaling ε concludes the proof.

[3] [Weyl's inequality for singular values; [30] Corollary 7.3.8] Let $\mathbf{\Phi}, \mathbf{\Psi} \in \mathbb{R}^{m\times n}$. Then, for all $i = 1, \ldots, \min(m,n)$: $|\sigma_i(\mathbf{\Phi}) - \sigma_i(\mathbf{\Psi})| \leq \|\mathbf{\Phi} - \mathbf{\Psi}\|_2$.

6 Streaming Principal Component Analysis

In this section, we are interested in computing a PCA of a matrix which is presented to an algorithm as a *stream* of columns. Let $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n] \in \mathbb{R}^{m\times n}$, where, for $i = 1 : n$, $\mathbf{a}_i \in \mathbb{R}^m$ is the ith column of $\mathbf{A}$. In the streaming setting of computation, we are allowed only one pass over the matrix $\mathbf{A}$, i.e., the algorithm "sees" the columns of $\mathbf{A}$ one by one and only once. Upon termination, the algorithm should return a matrix $\mathbf{U}$ with k orthonormal columns which is a "good" basis for $span(\mathbf{A})$ (a second pass over the columns of $\mathbf{A}$ is neccesary if one wants to compute $\mathbf{U}^{\mathrm{T}}\mathbf{A}$). In Section 6.1 below, we describe a version of the algorithm of Section 4.2 which gives a space-optimal streaming algorithm. In Section 6.2, we relax the problem and we describe a two-pass streaming algorithm which is also a version of the algorithm of Section 4.2; unfortunately, the space complexity of this algorithm can be bounded only in terms of "real numbers". We fix this in Section 6.3 where we describe an optimal two-pass streaming algorithm.

6.1 One-pass streaming PCA

Inputs to the algorithm are a matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n] \in \mathbb{R}^{m\times n}$, a rank parameter $k < \mathrm{rank}(\mathbf{A})$, and an accuracy parameter $0 < \varepsilon < 1$. The algorithm is identical to the algorithm in Section 4.2 except that steps 3, 4, and 5 in that algorithm are now implemented in the streaming setting.

Algorithm

1. Construct a low-rank preserving sketching matrix $\mathbf{W} \in \mathbb{R}^{\xi\times m}$ with $\xi = O(k\varepsilon^{-1})$ (see Lemma 12); then, construct $\mathbf{R} = \mathbf{W}\mathbf{A} \in \mathbb{R}^{\xi\times n}$.
2. Construct affine embedding matrices $\mathbf{T}_{left} \in \mathbb{R}^{\xi_1\times m}$ and $\mathbf{T}_{right} \in \mathbb{R}^{n\times\xi_2}$ with $\xi_1 = O(k/\varepsilon^3)$, $\xi_2 = O(k/\varepsilon^3)$ (see Definition 16).
3. Initialize all-zeros matrices: $\mathbf{M} \in \mathbb{R}^{\xi_1\times\xi_2}$, $\mathbf{L} \in \mathbb{R}^{\xi\times\xi_2}$, $\mathbf{N} \in \mathbb{R}^{\xi_1\times\xi}$, $\mathbf{D} \in \mathbb{R}^{m\times\xi}$, and all-zeros vectors $\mathbf{x} \in \mathbb{R}^{\xi_1\times 1}$, $\mathbf{y} \in \mathbb{R}^{1\times\xi_2}$, $\mathbf{z} \in \mathbb{R}^{\xi\times 1}$.
4. For $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ (one pass over the columns in $\mathbf{A}$)
   (a) $\mathbf{x} = \mathbf{T}_{left}\mathbf{a}_i$, $\mathbf{z} = \mathbf{W}\mathbf{a}_i$, and let $\mathbf{y}$ be the ith row of $\mathbf{T}_{right}$.
   (b) $\mathbf{M} = \mathbf{M} + \mathbf{x}\cdot\mathbf{y}$
   (c) $\mathbf{L} = \mathbf{L} + \mathbf{z}\cdot\mathbf{y}$
   (d) $\mathbf{N} = \mathbf{N} + \mathbf{x}\cdot\mathbf{z}^{\mathrm{T}}$
   (e) $\mathbf{D} = \mathbf{D} + \mathbf{a}_i\cdot\mathbf{z}^{\mathrm{T}}$
5. end
6. Construct $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X})\le k}{\mathrm{argmin}} \|\mathbf{N}\cdot\mathbf{X}\cdot\mathbf{L} - \mathbf{M}\|_{\mathrm{F}}^2$. (Notice that $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X})\le k}{\mathrm{argmin}} \|\mathbf{T}_{left}\left(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R} - \mathbf{A}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2$.)
7. Compute the SVD of $\mathbf{X}_* = \mathbf{U}_{\mathbf{X}_*}\mathbf{\Sigma}_{\mathbf{X}_*}\mathbf{V}_{\mathbf{X}_*}^{\mathrm{T}}$ ($\mathbf{U}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$, $\mathbf{\Sigma}_{\mathbf{X}_*} \in \mathbb{R}^{k\times k}$ $\mathbf{V}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$); then, compute
$$\mathbf{T} = \mathbf{D}\mathbf{U}_{\mathbf{X}_*}.$$
8. Compute an orthonormal basis $\mathbf{U} \in \mathbb{R}^{m\times k}$ for $span(\mathbf{T})$.

The theorem below analyzes the approximation error, the space complexity, and the running time of the previous algorithm.

Theorem 37. *The matrix $\mathbf{U} \in \mathbb{R}^{m\times k}$ with k orthonormal columns satisfies w.p.* 0.97*:*

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1 + O(\varepsilon)) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

The space complexity of the algorithm is $O(mk/\varepsilon + \mathrm{poly}(k, \varepsilon))$ words and the running time is of the order $O\left(mn\log(k\varepsilon^{-1}) + mk^2 + n \cdot \mathrm{poly}(k\varepsilon^{-1})\right)$.

Proof. The matrix $\mathbf{U}$ - up to the randomness in the algorithms - is exactly the same matrix as in the algorithm in Section 4.2, hence Theorem 22 shows the claim.

To see the space complexity of the algorithm, observe that it suffices to maintain the matrices and the vectors in the fourth step of the algorithm. Furthermore, observe that by the end of the stream: $\mathbf{M} = \mathbf{T}_{left} \cdot \mathbf{A} \cdot \mathbf{T}_{right} \in \mathbb{R}^{\xi_1\times\xi_2}$, $\mathbf{L} = \mathbf{W}\mathbf{A}\mathbf{T}_{right} \in \mathbb{R}^{\xi\times\xi_2}$, $\mathbf{N} = \mathbf{T}_{left}\mathbf{A}\mathbf{R}^{\mathrm{T}} \in \mathbb{R}^{\xi_1\times\xi}$, and $\mathbf{D} = \mathbf{A}\mathbf{A}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}} \in \mathbb{R}^{m\times\xi}$. Those matrices form the so called "sketch" of the algorithm.

The operations in the algorithm are effectively the same operations as in the algorithm in Section 4.2, hence the analysis of the running time in Theorem 22 shows the claim. ∎

6.2 Two-pass streaming PCA with "real numbers" space complexity

A simple modification of the one-pass streaming algorithm in the previous section leads to a two-pass streaming algorithm with $O(mk) + \mathrm{poly}(k/\varepsilon)$ "real numbers" space complexity:

Algorithm

1. Construct a low-rank preserving sketching matrix $\mathbf{W} \in \mathbb{R}^{\xi\times m}$ with $\xi = O(k\varepsilon^{-1})$ (see Lemma 12); then, construct $\mathbf{R} = \mathbf{W}\mathbf{A} \in \mathbb{R}^{\xi\times n}$.
2. Construct affine embedding matrices $\mathbf{T}_{left} \in \mathbb{R}^{\xi_1\times m}$ and $\mathbf{T}_{right} \in \mathbb{R}^{n\times\xi_2}$ with $\xi_1 = O(k/\varepsilon^3)$, $\xi_2 = O(k/\varepsilon^3)$ (see Definition 16).
3. Initialize all-zeros matrices: $\mathbf{M} \in \mathbb{R}^{\xi_1\times\xi_2}$, $\mathbf{L} \in \mathbb{R}^{\xi\times\xi_2}$, $\mathbf{N} \in \mathbb{R}^{\xi_1\times\xi}$, $\mathbf{T} \in \mathbb{R}^{m\times k}$, and all-zeros vectors $\mathbf{x} \in \mathbb{R}^{\xi_1\times 1}$, $\mathbf{y} \in \mathbb{R}^{1\times\xi_2}$, $\mathbf{z} \in \mathbb{R}^{\xi\times 1}$, $\mathbf{w} \in \mathbb{R}^{1\times k}$.
4. For $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ (**first pass over the columns in $\mathbf{A}$**)
   (a) $\mathbf{x} = \mathbf{T}_{left}\mathbf{a}_i$, $\mathbf{z} = \mathbf{W}\mathbf{a}_i$, and let $\mathbf{y}$ be the ith row of $\mathbf{T}_{right}$.
   (b) $\mathbf{M} = \mathbf{M} + \mathbf{x} \cdot \mathbf{y}$
   (c) $\mathbf{L} = \mathbf{L} + \mathbf{z} \cdot \mathbf{y}$
   (d) $\mathbf{N} = \mathbf{N} + \mathbf{x} \cdot \mathbf{z}^{\mathrm{T}}$
5. end
6. Construct $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X})\leq k}{\mathrm{argmin}} \|\mathbf{N}\cdot\mathbf{X}\cdot\mathbf{L} - \mathbf{M}\|_{\mathrm{F}}^2$. (Notice that $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X})\leq k}{\mathrm{argmin}} \|\mathbf{T}_{left}\left(\mathbf{A}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R} - \mathbf{A}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2$.)
7. Compute the SVD of $\mathbf{X}_* = \mathbf{U}_{\mathbf{X}_*}\mathbf{\Sigma}_{\mathbf{X}_*}\mathbf{V}_{\mathbf{X}_*}^{\mathrm{T}}$ ($\mathbf{U}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$, $\mathbf{\Sigma}_{\mathbf{X}_*} \in \mathbb{R}^{k\times k}$ $\mathbf{V}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$).
8. Construct $\mathbf{Z} = \mathbf{W}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*} \in \mathbb{R}^{\xi\times k}$.
9. For $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ (**second pass over the columns in $\mathbf{A}$**)
   (a) $\mathbf{w} = \mathbf{a}_i^{\mathrm{T}}\mathbf{Z}$
   (b) $\mathbf{T} = \mathbf{T} + \mathbf{a}_i \cdot \mathbf{w}$
10. end
11. Compute an orthonormal basis $\mathbf{U} \in \mathbb{R}^{m\times k}$ for $span(\mathbf{T})$.

The theorem below analyzes the approximation error, the space complexity, and the running time of the previous algorithm. Notice that the space complexity of this algorithm is only given in terms of "real numbers" - we resolve this issue in the following section.

Theorem 38. *The matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$ with k orthonormal columns satisfies w.p.* 0.97*:*

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1 + O(\varepsilon)) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

The space complexity of the algorithm is

$$O(mk + \mathrm{poly}(k, \varepsilon))$$

"real numbers" and the running time is of the order

$$O\left(mn \log(k\varepsilon^{-1}) + mk^2 + n \cdot \mathrm{poly}(k\varepsilon^{-1})\right).$$

Proof. The matrix $\mathbf{U}$ - up to the randomness in the algorithms - is exactly the same matrix as in the algorithm in Section 4.2, hence Theorem 22 shows the claim.

To see the space complexity of the algorithm, observe that it suffices to maintain the matrices and the vectors in the fourth, eight, and ninth steps of the algorithm. Furthermore, observe that by the end of the stream:

$$\mathbf{M} = \mathbf{T}_{left} \cdot \mathbf{A} \cdot \mathbf{T}_{right} \in \mathbb{R}^{\xi_1 \times \xi_2},$$
$$\mathbf{L} = \mathbf{W}\mathbf{A}\mathbf{T}_{right} \in \mathbb{R}^{\xi \times \xi_2},$$
$$\mathbf{N} = \mathbf{T}_{left}\mathbf{A}\mathbf{R}^{\mathrm{T}} \in \mathbb{R}^{\xi_1 \times \xi},$$

and

$$\mathbf{T} = \mathbf{A}\mathbf{A}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\mathbf{U}_{\mathbf{X}_*} \in \mathbb{R}^{m \times k}.$$

The operations in the algorithm are effectively the same operations as in the algorithm in Section 4.2, hence the analysis of the running time in Theorem 22 shows the claim. ∎

6.3 Two-pass streaming PCA with bit space complexity

The previous algorithm could suffer from large space complexity in case we need a lot of machine words to write down the entries of $\mathbf{U}_{\mathbf{X}_*}$. To fix this issue we use the same idea as in the case of the distributed PCA algorithm in Section 5. This leads to a two-pass streaming algorithm for PCA. Again, as in the distributed case, we need to test if the rank of $\mathbf{A}$ is less than $2k$, and then we use one approach if $\mathrm{rank}(\mathbf{A}) < 2k$ and another approach if $\mathrm{rank}(\mathbf{A}) \geq 2k$. Both of these approaches can be implemented with two passes. In the overall streaming PCA algorithm that we would like to design, we can not wait for the algorithm that tests the rank to finish and then start running one of the two PCA protocols, because this will lead to a three-pass algorithm (one pass to test the rank and two passes for the actual PCA protocol). To keep the number of passes to two, we just start running the PCA protocols in parallel with the protocol that tests the rank of the input matrix. In the end of the first pass over the columns of $\mathbf{A}$, we already know which protocol to follow, and we just do this, disregarding the other protocol.

We already discussed the streaming version of the algorithm that tests the rank of the matrix in Lemma 24. Below, we describe separately the two streaming PCA protocols.

6.3.1 Streaming PCA protocol when $\text{rank}(\mathbf{A}) \leq 2k$

The idea here is to implement a streaming version of the distributed PCA protocol in Lemma 25.

Algorithm

1. Construct an $n \times 2k$ matrix $\mathbf{H}''$ as in Lemma 25.
2. Construct affine embedding matrices $\mathbf{T}_{left} \in \mathbb{R}^{\xi_1 \times m}$ and $\mathbf{T}_{right} \in \mathbb{R}^{n \times \xi_2}$ with $\xi_1 = O(k/\varepsilon^2)$, $\xi_2 = O(k/\varepsilon^2)$ (see Definition 16).
3. Initialize all-zeros matrices: $\mathbf{C} \in \mathbb{R}^{m \times 2k}$, $\mathbf{M} \in \mathbb{R}^{\xi_1 \times \xi_2}$, $\mathbf{L} \in \mathbb{R}^{\xi \times \xi_2}$, $\mathbf{N} \in \mathbb{R}^{\xi_1 \times 2k}$, and all-zeros vectors $\mathbf{x} \in \mathbb{R}^{\xi_1 \times 1}$, $\mathbf{y} \in \mathbb{R}^{1 \times \xi_2}$, $\mathbf{z} \in \mathbb{R}^{1 \times 2k}$, $\mathbf{w} \in \mathbb{R}^{2k \times 1}$.
4. For $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ (**first pass over the columns in A**)
   (a) let $\mathbf{z}$ be the ith row of $\mathbf{H}''$
   (b) $\mathbf{C} = \mathbf{C} + \mathbf{a}_i \mathbf{z}$
5. end
6. For notational convenience, let $\mathbf{B} := \mathbf{C}^{\mathrm{T}} \in \mathbb{R}^{2k \times m}$.
7. For $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ (**second pass over the columns in A**)
   (a) let $\mathbf{z}$ be the ith row of $\mathbf{H}''$, and let $\mathbf{y}$ be the ith row of $\mathbf{T}_{right}$.
   (b) $\mathbf{x} = \mathbf{T}_{left} \mathbf{a}_i$;
   (c) $\mathbf{w} = \mathbf{B}\mathbf{a}_i$
   (d) $\mathbf{M} = \mathbf{M} + \mathbf{x} \cdot \mathbf{y}$
   (e) $\mathbf{L} = \mathbf{L} + \mathbf{w} \cdot \mathbf{y}$
   (f) $\mathbf{N} = \mathbf{N} + \mathbf{x} \cdot \mathbf{z}$
8. end
9. Construct

$$\mathbf{X}_* = \underset{\text{rank}(\mathbf{X}) \leq k}{\text{argmin}} \|\mathbf{N} \cdot \mathbf{X} \cdot \mathbf{L} - \mathbf{M}\|_{\mathrm{F}}^2 := \underset{\text{rank}(\mathbf{X}) \leq k}{\text{argmin}} \|\mathbf{T}_{left} \mathbf{A} \mathbf{H}'' \mathbf{X} (\mathbf{H}'')^{\mathrm{T}} \mathbf{A}^{\mathrm{T}} \mathbf{A} \mathbf{T}_{right} - \mathbf{T}_{left} \mathbf{A} \mathbf{T}_{right}\|_{\mathrm{F}}^2.$$

10. Compute an orthonormal basis $\mathbf{U} \in \mathbb{R}^{m \times k}$ for $span(\mathbf{C}\mathbf{X}_*)$.

6.3.2 Streaming PCA protocol when $\text{rank}(\mathbf{A}) > 2k$

The idea here is to implement a streaming version of step 3-b of the algorithm in Section 5.4.

Algorithm

1. Construct a low-rank preserving sketching matrix $\mathbf{W} \in \mathbb{R}^{\xi \times m}$ with $\xi = O(k\varepsilon^{-1})$ (see Lemma 12); then, construct $\mathbf{R} = \mathbf{W}\mathbf{A} \in \mathbb{R}^{\xi \times n}$.
2. Construct affine embedding matrices $\mathbf{T}_{left} \in \mathbb{R}^{\xi_1 \times m}$ and $\mathbf{T}_{right} \in \mathbb{R}^{n \times \xi_2}$ with $\xi_1 = O(k/\varepsilon^3)$, $\xi_2 = O(k/\varepsilon^3)$ (see Definition 16).
3. Initialize all-zeros matrices: $\mathbf{M} \in \mathbb{R}^{\xi_1 \times \xi_2}$, $\mathbf{L} \in \mathbb{R}^{\xi \times \xi_2}$, $\mathbf{N} \in \mathbb{R}^{\xi_1 \times \xi}$, $\mathbf{T} \in \mathbb{R}^{m \times k}$, and all-zeros vectors $\mathbf{x} \in \mathbb{R}^{\xi_1 \times 1}$, $\mathbf{y} \in \mathbb{R}^{1 \times \xi_2}$, $\mathbf{z} \in \mathbb{R}^{\xi \times 1}$, $\mathbf{w} \in \mathbb{R}^{1 \times k}$.
4. For $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ (**first pass over the columns in A**)

(a) construct $\mathbf{b}_i \in \mathbb{R}^m$ where $\mathbf{b}_i$ is the perturbation of $\mathbf{a}_i$ after adding $1/n^D$ with probability $1/2$, and $-1/n^D$ with probability $1/2$, to each of its entries where D is the constant of Lemma 31.

(b) $\mathbf{x} = \mathbf{T}_{left}\mathbf{b}_i$, $\mathbf{z} = \mathbf{W}\mathbf{b}_i$, and let $\mathbf{y}$ be the ith row of $\mathbf{T}_{right}$.

(c) $\mathbf{M} = \mathbf{M} + \mathbf{x} \cdot \mathbf{y}$

(d) $\mathbf{L} = \mathbf{L} + \mathbf{z} \cdot \mathbf{y}$

(e) $\mathbf{N} = \mathbf{N} + \mathbf{x} \cdot \mathbf{z}^{\mathrm{T}}$

5. end

6. Construct $\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X}) \leq k}{\mathrm{argmin}} \|\mathbf{N} \cdot \mathbf{X} \cdot \mathbf{L} - \mathbf{M}\|_{\mathrm{F}}^2$, where $\mathbf{X}_* \in \mathbb{R}^{\xi \times \xi}$ has rank k. Notice that

$$\mathbf{X}_* = \underset{\mathrm{rank}(\mathbf{X}) \leq k}{\mathrm{argmin}} \|\mathbf{T}_{left}\left(\mathbf{B}\mathbf{R}^{\mathrm{T}}\mathbf{X}\mathbf{R} - \mathbf{B}\right)\mathbf{T}_{right}\|_{\mathrm{F}}^2.$$

For notational convenience, let us also denote

$$\mathbf{X}_* = \underbrace{\mathbf{N}^{\dagger}}_{\mathbf{\Phi}_1} \cdot \underbrace{\left(\mathbf{U}_{\mathbf{N}}\mathbf{U}_{\mathbf{N}}^{\mathrm{T}}\mathbf{M}\mathbf{U}_{\mathbf{L}}\mathbf{U}_{\mathbf{L}}^{\mathrm{T}}\right)_k}_{\mathbf{\Phi}_2} \cdot \underbrace{\mathbf{L}^{\dagger}}_{\mathbf{\Phi}_3}$$

Furthermore, compute the rank k SVD of $\mathbf{\Phi}_2$ as $\mathbf{\Phi}_2 = \mathbf{U}_{\mathbf{\Phi}_2}\mathbf{\Sigma}_{\mathbf{\Phi}_2}\mathbf{V}_{\mathbf{\Phi}_2}^{\mathrm{T}}$. Now round each of the entries in each of the above five matrices to the nearest integer multiple of $1/n^{\gamma}$ for a sufficiently large constant $\gamma > 0$. Let the matrices after the rounding be $\tilde{\mathbf{\Phi}}_1, \tilde{\mathbf{U}}_{\mathbf{\Phi}_2}, \tilde{\mathbf{\Sigma}}_{\mathbf{\Phi}_2}, \tilde{\mathbf{V}}_{\mathbf{\Phi}_2}, \tilde{\mathbf{\Phi}}_3$. Next,

$$\mathbf{Y}_* = \tilde{\mathbf{\Phi}}_1 \cdot \tilde{\mathbf{U}}_{\mathbf{\Phi}_2} \cdot \tilde{\mathbf{\Sigma}}_{\mathbf{\Phi}_2} \cdot \tilde{\mathbf{V}}_{\mathbf{\Phi}_2}^{\mathrm{T}} \cdot \tilde{\mathbf{\Phi}}_3 \in \mathbb{R}^{\xi_1 \times \mathbf{x}_2}.$$

Now, compute $\mathbf{D} \in \mathbb{R}^{\xi_1 \times k}$ with k linearly independent columns of $\mathbf{Y}_*$ and $\mathbf{Z} = \mathbf{W}^{\mathrm{T}}\mathbf{D} \in \mathbb{R}^{\xi \times k}$.

7. For $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ (**second pass over the columns in A**)

(a) construct $\mathbf{b}_i \in \mathbb{R}^m$ where $\mathbf{b}_i$ is the perturbation of $\mathbf{a}_i$ after adding $1/n^D$ with probability $1/2$, and $-1/n^D$ with probability $1/2$, to each of its entries where D is the constant of Lemma 31.

(b) $\mathbf{w} = \mathbf{b}_i^{\mathrm{T}}\mathbf{Z}$

(c) $\mathbf{T} = \mathbf{T} + \mathbf{b}_i \cdot \mathbf{w}$

8. end

9. Compute an orthonormal basis $\mathbf{U} \in \mathbb{R}^{m \times k}$ for $span(\mathbf{T})$.

6.3.3 Main result

The theorem below analyzes the approximation error, the space complexity, and the running time of the previous algorithm. Notice that the space complexity of this algorithm is given in terms of machine words.

Theorem 39. *The matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$ with k orthonormal columns satisfies with arbitrarily large constant probability:*

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1 + \varepsilon) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

The space complexity of the algorithm is $O\left(mk + \mathrm{poly}(k, \varepsilon)\right)$ words and the running time is of the order $O\left(mn\log(k\varepsilon^{-1}) + mk^2 + n \cdot \mathrm{poly}(k\varepsilon^{-1})\right)$.

Proof. The matrix $\mathbf{U}$ - up to the randomness in the algorithms - is exactly the same matrix as in the algorithm in Theorem 32, hence the approximation bound in that theorem shows the claim.

To see the space complexity of the algorithm, observe that it suffices to maintain the matrices and the vectors in the fourth and seventh steps of the two protocols above.

The operations in the algorithm are effectively the same operations as in the algorithm in Theorem 32, hence that theorem shows the claim. ∎

7 Distributed PCA for sparse matrices

Recall that in the problem of Definition 1 we are given 1) an $m \times n$ matrix $\mathbf{A}$ partitioned column-wise as $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s \end{pmatrix}$, with $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$ ($\sum w_i = n$); 2) a rank parameter $k < \text{rank}(\mathbf{A})$; 3) an accuracy parameter $\varepsilon > 0$. We would like to design an algorithm that finds an $m \times k$ matrix $\mathbf{U}$ with $\mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}_k$ and, upon termination, leaves this matrix $\mathbf{U}$ in each machine of the network.

The high level idea of our algorithm is to find, in a distributed way, a small set - $O(k\varepsilon^{-1})$ - of columns from $\mathbf{A}$ and then find $\mathbf{U}$ in the span of those columns. To choose those $O(k\varepsilon^{-1})$ columns of $\mathbf{A}$ in a distributed way we implement the following three-stage sampling procedure:

1. Local sampling: Each machine samples $O(k)$ columns using a deterministic algorithm from [15].
2. Global sampling: The server collects the columns from each machine and down-samples them to $O(k)$ columns using the same deterministic algorithm from [15].
3. Adaptive sampling: the server sends back to each machine those $O(k)$ columns; then, an extra of $O(k\varepsilon^{-1})$ columns are selected randomly from the entire matrix $\mathbf{A}$ using [21].

We argue that if $\tilde{\mathbf{C}}$ contains the columns selected with this three-stage approach, then, w.p. 0.99,

$$\|\mathbf{A} - \tilde{\mathbf{C}}\tilde{\mathbf{C}}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq (1 + O(\varepsilon)) \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

Though we could have used $\Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})$ to be the rank k matrix that approximates $\mathbf{A}$, we are not familiar with any communication-efficient computation of $\Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})$. To address this issue, using an idea from [34], we compute $\mathbf{U} \in span(\tilde{\mathbf{C}})$ such that $\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+O(\varepsilon))\|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2$; this $\mathbf{U}$ can be calculated with small communication cost and it is sufficient for our purposes.

Before presenting the new algorithm in detail, we discuss results from previous literature that we employ in the analysis.

7.1 Background material

7.1.1 Column sampling algorithms

Our distributed PCA algorithm in Section 7 samples columns from the input matrix in three stages:

1. Local sampling: $O(k)$ columns are selected locally in each machine.
2. Global sampling: $O(k)$ columns are selected in the server.
3. Adaptive sampling: an extra $O(k\varepsilon^{-1})$ columns are selected from the entire matrix $\mathbf{A}$.

In the first two stages, we use a deterministic algorithm developed in [15], which itself extends the Batson, Spielman, and Strivastava (BSS) spectral sparsification algorithm [10]. For the actual algorithm we defer the reader to Lemma 3.6 in [15]. Lemma 40 and Lemma 41 below present the relevant results. In the third sampling stage, we use an adaptive sampling algorithm from [21].

Lemma 40 (Dual Set Spectral-Frobenius Sparsification. Lemma 3.6 in [15])**.** *Let $\mathbf{V} \in \mathbb{R}^{w \times k}$ be a matrix with $w > k$ and $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}_k$. Let $\mathbf{E} \in \mathbb{R}^{m \times w}$ be an arbitrary matrix. Then, given an integer ℓ such that $k < \ell \leq w$, there exists a deterministic algorithm that runs in $O\left(\ell w k^2 + mw\right)$ time, and constructs a "sampling/rescaling" $w \times \ell$ matrix $\mathbf{S}$ such that*

$$\sigma_k^2\left(\mathbf{V}^{\mathrm{T}}\mathbf{S}\right) \geq \left(1 - \sqrt{k/\ell}\right)^2, \qquad \|\mathbf{E}\mathbf{S}\|_{\mathrm{F}}^2 \leq \|\mathbf{E}\|_{\mathrm{F}}^2.$$

Specifically, $rank(\mathbf{V}^{\mathrm{T}}\mathbf{S}) = k$. We denote this sampling procedure as $\mathbf{S} = BssSampling(\mathbf{V}, \mathbf{E}, \ell)$.

In words, given $\mathbf{V}$ and $\mathbf{E}$, there exists an algorithm to select (and rescale) a small number of columns from $\mathbf{E}$ and rows from $\mathbf{V}$ such that: 1) the Frobenius norm squared of the sampled $\mathbf{E}$ is less than the Frobenius norm squared of $\mathbf{E}$; 2) the sampled $\mathbf{V}$ has rank equal to the rank of $\mathbf{V}$; and 3) the smallest singular value squared of the sampled $\mathbf{V}$ is bounded from below by $\left(1-\sqrt{k/\ell}\right)^2$.

Lemma 41 (Constant factor column-based matrix reconstruction; Theorem 5 in [15])**.** *Given matrix $\mathbf{G} \in \mathbb{R}^{m\times\alpha}$ of rank ρ and a target rank k [4], there exists a deterministic algorithm that runs in $O\left(\alpha m \min\{\alpha, m\} + \alpha c k^2\right)$ time and selects $c > k$ columns of $\mathbf{G}$ to form a matrix $\mathbf{C} \in \mathbb{R}^{m\times c}$ with*

$$\|\mathbf{G} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{G}\|_{\mathrm{F}}^2 \leq \left(1 + \left(1-\sqrt{k/c}\right)^{-2}\right) \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{G})} \sigma_i^2(\mathbf{G}).$$

The algorithm in this theorem finds $\mathbf{C}$ as $\mathbf{C} = \mathbf{G}\mathbf{S}$, where $\mathbf{S} = BssSampling(\mathbf{V}, \mathbf{G} - \mathbf{G}\mathbf{V}\mathbf{V}^{\mathrm{T}}, c)$ and $\mathbf{V} \in \mathbb{R}^{\alpha\times k}$ contains the top k right singular vectors of $\mathbf{G}$. We denote this sampling procedure as $\mathbf{C} = DeterministicCssFrobenius(\mathbf{G}, k, c)$.

In words, there exists a deterministic algorithm, running in polynomial time, to select any number of $c > k$ columns from the given matrix $\mathbf{G}$, such that the residual error, in Frobenius norm, from projecting $\mathbf{G}$ onto the span of the sampled columns is bounded from above with respect to the residual error of the best rank k matrix from the SVD of $\mathbf{G}$. Notice that

$$\sum_{i=k+1}^{\mathrm{rank}(\mathbf{G})} \sigma_i^2(\mathbf{G}) = \|\mathbf{G} - \mathbf{G}_k\|_{\mathrm{F}}^2,$$

where $\mathbf{G}_k \in \mathbb{R}^{m\times\alpha}$ has rank at most k and is computed via the SVD of $\mathbf{G}$.

Before presenting the adaptive sampling theorem from [21] we introduce some useful notation used in the lemma. Let $\mathbf{A} \in \mathbb{R}^{m\times n}$, $k < n$, and $\mathbf{V} \in \mathbb{R}^{m\times c}$ with $k < c < n$. $\Pi_{\mathbf{V},k}^{\mathrm{F}}(\mathbf{A}) \in \mathbb{R}^{m\times n}$ is the best rank k approximation to $\mathbf{A}$ - wrt the Frobenius norm - in the column span of $\mathbf{V}$. Equivalently, $\Pi_{\mathbf{V},k}^{\mathrm{F}}(\mathbf{A}) = \mathbf{V}\mathbf{X}_{opt}$, where

$$\mathbf{X}_{opt} = \underset{\mathbf{X}\in\mathbb{R}^{c\times n}, \mathrm{rank}(\mathbf{X})\leq k}{\mathrm{argmin}} \|\mathbf{A} - \mathbf{V}\mathbf{X}\|_{\mathrm{F}}^2.$$

Lemma 42 (Adaptive sampling; Theorem 2.1 of [21])**.** *Given $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{V} \in \mathbb{R}^{m\times c_1}$ (with $c_1 \leq n, m$), define the residual matrix $\mathbf{\Psi} = \mathbf{A} - \mathbf{V}\mathbf{V}^{\dagger}\mathbf{A} \in \mathbb{R}^{m\times n}$. For $j = 1, \ldots, n$, let p_j be a probability distribution such that $p_j \geq \beta\|\mathbf{\Psi}^{(j)}\|_2^2/\|\mathbf{\Psi}\|_{\mathrm{F}}^2$, for some $1 > \beta > 0$, where $\mathbf{\Psi}^{(j)}$ is the j-th column of the matrix $\mathbf{\Psi}$. Sample c_2 columns from $\mathbf{A}$ in c_2 i.i.d. trials, where in each trial the j-th column is chosen with probability p_j. Let $\mathbf{C}_2 \in \mathbb{R}^{m\times c_2}$ contain the c_2 sampled columns and let $\mathbf{C} = [\mathbf{V}\ \ \mathbf{C}_2] \in \mathbb{R}^{m\times(c_1+c_2)}$ contain the columns of $\mathbf{V}$ and $\mathbf{C}_2$. Then, for any integer $k > 0$,*

$$\mathbb{E}\left[\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2\right] \leq \mathbb{E}\left[\|\mathbf{A} - \Pi_{\mathbf{C},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2\right] \leq \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}) + \frac{k}{\beta\cdot c_2} \cdot \|\mathbf{A} - \mathbf{V}\mathbf{V}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2.$$

Given $\mathbf{A}$ and $\mathbf{C}$, this method requires $O(c_1 mn)$ time to compute $\mathbf{\Psi}$, another $O(mn)$ time to compute the probabilities p_j's and another $O(n + c_2)$ time for the sampling step - using the method in [46]. In total, the method requires $O(c_1 mn + c_2)$ time to compute $\mathbf{C}_2$. We denote this sampling procedure as $\mathbf{C}_2 = AdaptiveCols(\mathbf{A}, \mathbf{V}, c_2, \beta)$.

[4] The original Theorem 5 in [15] has the assumption that $k < \rho$, but this assumption can be dropped having the result unchanged. The only reason the assumption $k < \rho$ exists is because otherwise column subset selection is trivial.

In words, given the matrix $\mathbf{A}$ and the subspace $\mathbf{V}$, there exists a randomized algorithm to sample an additional of c_2 columns from $\mathbf{A}$, based on probabilities from the residual error matrix $\mathbf{A} - \mathbf{V}\mathbf{V}^\dagger\mathbf{A}$, such that residual error, in Frobenius norm, from projecting $\mathbf{A}$ onto the span of the union of the columns of $\mathbf{V}$ and the sampled columns is bounded from above with respect to the best rank k approximation to $\mathbf{A}$, the residual $\mathbf{A} - \mathbf{V}\mathbf{V}^\dagger\mathbf{A}$, and the number of sampled columns c_2.

7.1.2 Distributed adaptive sampling

In our distributed PCA algorithm below, we also need to use a distributed version of the previous adaptive sampling procedure. We provide some preliminary results for that task in this section.

Lemma 43. *Suppose that the columns of an $m \times n$ matrix $\mathbf{A}$ are partitioned arbitrarily across the machines into matrices $\mathbf{A}_1, \ldots, \mathbf{A}_s$. Let $\mathbf{C}$ be an arbitrary $m \times r$ matrix. Consider the distribution p on n columns in which*

$$p_j = \frac{\|\mathbf{a}_j - \mathbf{C}\mathbf{C}^\dagger\mathbf{a}_j\|_{\mathrm{F}}^2}{\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_{\mathrm{F}}^2},$$

where $\mathbf{a}_j$ is the j-th column of $\mathbf{A}$ ($j = 1 : n$ here).

For each $i \in [s]$, let some value β_i satisfies

$$\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2 \leq \beta^i \leq 2\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2.$$

For each $j \in [n]$, if column $\mathbf{a}_j$ is held on the i-th server (denoted $\mathbf{a}_j^i$), then let

$$q_j = \frac{\beta^i}{\sum_{i'=1}^s \beta_{i'}} \cdot \frac{\|\mathbf{a}_j^i - \mathbf{C}\mathbf{C}^\dagger\mathbf{a}_j^i\|_2^2}{\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2}.$$

Then for each $j \in [n]$,

$$p_j/2 \leq q_j \leq 2p_j.$$

Proof. By definition of the β^i, we have that

$$\frac{\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2}{2\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_{\mathrm{F}}^2} \leq \frac{\beta^i}{\sum_{i'=1}^s \beta_{i'}} \leq \frac{2\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2}{\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_{\mathrm{F}}^2}.$$

Hence,

$$p_j/2 \leq q_j \leq 2p_j.$$

■

Lemma 44. *$O(\log k + \log\log(mns/\varepsilon))$ Suppose the coordinator has an $m \times r$ matrix $\mathbf{C}$ of columns of an $m \times n$ matrix $\mathbf{A}$, where $r = O(k)$. Suppose the entries of $\mathbf{A}$ are integers bounded by $\mathrm{poly}(mns/\varepsilon)$ in magnitude, and let the columns of $\mathbf{A}$ be partitioned arbitrarily across the servers into matrices $\mathbf{A}_1, \ldots, \mathbf{A}_s$.*

There is a protocol using $O(skm)$ machine words of communication for the coordinator to learn values β^i so that for all $i \in [s]$,

$$\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2 \leq \beta^i \leq 2\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2.$$

Proof. The coordinator first sends $\mathbf{C}$ to all machines. The i-th server locally computes $\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2$. If this number is 0, he/she sends 0 to the coordinator, and in this case β^i satisfies the requirement in the statement of the lemma.

Otherwise, consider the matrix $\mathbf{B}_i$ which is formed by concatenating the columns of $\mathbf{A}_i$ with those of $\mathbf{C}$. Then

$$\|\mathbf{B}_i - \mathbf{C}\mathbf{C}^\dagger \mathbf{B}_i\|_{\mathrm{F}}^2 = \|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}_i\|_{\mathrm{F}}^2 > 0, \tag{6}$$

since the columns of $\mathbf{B}_i$ in $\mathbf{C}$ contribute a cost of 0. However, since $\mathbf{B}_i$ contains the columns of $\mathbf{C}$ and its cost is non-zero, it implies the rank of $\mathbf{B}_i$ is at least $r+1$. By Corollary 27, this implies

$$\|\mathbf{B}_i - \mathbf{C}\mathbf{C}^\dagger \mathbf{B}_i\|_{\mathrm{F}}^2 > (mns/\varepsilon)^{-O(k)},$$

which by (6) gives the same lower bound on $\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}_i\|_{\mathrm{F}}^2$. Note also that

$$\|\mathbf{A}_i - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}_i\|_{\mathrm{F}}^2 \leq \mathrm{poly}(mns/\varepsilon),$$

since $\|\mathbf{A}\|_{\mathrm{F}}^2 \leq \mathrm{poly}(mns/\varepsilon)$. This implies if the i-th machine sends β^i to the coordinator, where β^i is the nearest power of 2 to $\|\mathbf{A}^i - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}^i\|_{\mathrm{F}}^2$, there will only be $O(k \log(mns/\varepsilon))$ possible values of β^i, and hence each can be specified using $O(\log k + \log\log(mns/\varepsilon))$ bits, i.e., a single machine word. This completes the proof. ■

7.1.3 Low-rank matrix approximations within a subspace

The final stage of our distributed PCA algorithm below finds $\mathbf{U} \in span(\tilde{\mathbf{C}})$ such that the error of the residual matrix $\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}$ is "small". To implement this step, we employ an algorithm developed in [34].

Lemma 45. *Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be the input matrix and $\mathbf{V} \in \mathbb{R}^{m\times c}$ be the input subspace. We further assume that for some rank parameter $k < c$ and accuracy parameter $0 < \varepsilon < 1$:*

$$\|\mathbf{A} - \Pi_{\mathbf{V},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq (1 + O(\epsilon))\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

Let $\mathbf{V} = \mathbf{Y}\mathbf{\Psi}$ be a qr decomposition of $\mathbf{V}$ with $\mathbf{Y} \in \mathbb{R}^{m\times c}$ and $\mathbf{\Psi} \in \mathbb{R}^{c\times c}$. Let $\mathbf{\Xi} = \mathbf{Y}^{\mathrm{T}}\mathbf{A}\mathbf{W}^{\mathrm{T}} \in \mathbb{R}^{c\times\xi}$, where $\mathbf{W}^{\mathrm{T}} \in \mathbb{R}^{n\times\xi}$ with $\xi = O(c/\varepsilon^2)$, each element of which is chosen i.i.d. to be $\{+1/\sqrt{n}, -1/\sqrt{n}\}$ with probability $1/2$. Let $\mathbf{\Delta} \in \mathbb{R}^{c\times k}$ contain the top k left singular vectors of $\mathbf{\Xi}$. Then, with probability at least $1 - e^{-c}$,

$$\|\mathbf{A} - \mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

$\mathbf{Y}$, and $\mathbf{\Delta}$ can be computed in $O(mn\xi)$ time. We denote this procedure as

$$[\mathbf{Y}, \mathbf{\Delta}] = ApproxSubspaceSVD(\mathbf{A}, \mathbf{V}, k, \varepsilon).$$

Proof. This result was essentially proven inside the proof of Theorem 1.1 in [33]. Specifically, the error bound proven in [33] is for the transpose of $\mathbf{A}$ (also $\mathbf{Y}, \mathbf{\Delta}$ are denoted with U, V in [33]). As for the running time, given $\mathbf{A}, \mathbf{V}$, and k, one can compute (i) $\mathbf{Y}$ in $O(mc^2)$ time; (ii) $\mathbf{\Xi}$ in $O(mn\xi + mc\xi)$ time; and (iii) $\mathbf{\Delta}$ in $O(c^2\xi)$ time. ■

In words, the lemma indicates that given the matrix $\mathbf{A}$ and the subspace $\mathbf{V}$ such that

$$\|\mathbf{A} - \Pi_{\mathbf{V},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq (1 + O(\epsilon))\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2,$$

for some small $\varepsilon > 0$, there exists a randomized algorithm to compute matrices $\mathbf{Y} \in span(\mathbf{V})$ and $\mathbf{\Delta}$ such that the residual error, in Frobenius norm, from projecting $\mathbf{A}$ onto the span of $\mathbf{Y}\mathbf{\Delta}$ is also bounded by $(1 + O(\epsilon))\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$. Notice that Δ contains the top k left singular vectors of a "sketched" version of $\mathbf{Y}^{\mathrm{T}}\mathbf{A}$, a property which makes the computation particularly effective in terms of running time.

7.2 Detailed description of the algorithm

Input:

1. $\mathbf{A} \in \mathbb{R}^{m \times n}$ partitioned column-wise $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s \end{pmatrix}$; for $i = 1 : s$, $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$; $\sum_i w_i = n$.
2. rank parameter $k < \text{rank}(\mathbf{A})$
3. accuracy parameter $\varepsilon > 0$

Algorithm

1. **Local Column Sampling**
   (a) For each sub-matrix $\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$, compute the matrix $\mathbf{V}_i \in \mathbb{R}^{w_i \times k}$ with the top k right singular vectors of $\mathbf{A}_i$. Also, construct the $m \times w_i$ matrix $\mathbf{E}_i = \mathbf{A}_i - \mathbf{A}_i \mathbf{V}_i \mathbf{V}_i^{\mathrm{T}}$. For each sub-matrix $\mathbf{A}_i$, compute $\mathbf{C}_i \in \mathbb{R}^{m \times \ell}$ containing $\ell = 4k$ columns from $\mathbf{A}_i$ as follows: $\mathbf{C}_i = \mathbf{A}_i \mathbf{S}_i$. Here, $\mathbf{S}_i$ has dimensions $w_i \times \ell$ and is constructed as follows: $\mathbf{S}_i = BssSampling(\mathbf{V}_i, \mathbf{E}_i, \ell)$ (see Lemma 40).
   (b) Machine i sends $\mathbf{C}_i$ to the server.
2. **Global Column Sampling**
   (a) Server constructs $m \times (s \cdot \ell)$ matrix $\mathbf{G}$ containing $(s \cdot \ell)$ actual columns from $\mathbf{A}$ as follows: $\mathbf{G} = \begin{pmatrix} \mathbf{C}_1 & \mathbf{C}_2 & \dots & \mathbf{C}_s \end{pmatrix}$. Then, server constructs $\mathbf{C} \in \mathbb{R}^{m \times c_1}$ via choosing $c_1 = 4k$ columns from $\mathbf{G}$ as follows: $\mathbf{C} = DeterministicCssFrobenius(\mathbf{G}, k, c_1)$ (see Lemma 41).
   (b) Server sends $\mathbf{C}$ to all the machines.
3. **Adaptive Column Sampling**
   (a) Machine i computes $\mathbf{\Psi}_i = \mathbf{A}_i - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$ and then computes β_i as it was described in Lemma 44. Machine i sends β_i to server.
   (b) Server computes probability distribution $g_i = \frac{\beta_i}{\sum_i \beta_i}$. Server samples i.i.d. with replacement $\lceil 50k/\varepsilon \rceil$ samples (machines) from g_i. Then, server determines numbers t_i ($i = 1, 2, \dots, s$), where t_i is the number of times the ith machine was sampled. It sends the t_i's to the machines.
   (c) Machine i computes probabilities $q_j^i = \|\mathbf{x}\|_2^2 / \|\mathbf{\Psi}_i\|_{\mathrm{F}}^2$ ($j = 1 : w_i$), where $\mathbf{x}$ is the jth column of $\mathbf{\Psi}_i$. And now machine i samples t_i samples from it's local probability distribution and sends the corresponding columns to the server. Let $c_2 = \sum_i t_i = \lceil 50k/\varepsilon \rceil$.
   (d) Server collects the columns and assigns them to $\hat{\mathbf{C}} \in \mathbb{R}^{m \times c_2}$. Server constructs $\tilde{\mathbf{C}}$ to be the $m \times (c_1 + c_2)$ matrix: $\tilde{\mathbf{C}} = \begin{pmatrix} \mathbf{C} & \hat{\mathbf{C}} \end{pmatrix}$. Let $c = c_1 + c_2 = 4k + \lceil 50k/\varepsilon \rceil$.
4. **Rank-k matrix in the span of $\tilde{\mathbf{C}}$**
   (a) Server sends $\tilde{\mathbf{C}}$ to all the machines and each machine computes (the same) qr factorization of $\tilde{\mathbf{C}}$: $\tilde{\mathbf{C}} = \mathbf{Y}\mathbf{R}$ where $\mathbf{Y} \in \mathbb{R}^{m \times c}$ has orthonormal columns and $\mathbf{R} \in \mathbb{R}^{c \times c}$ is upper triangular.
   (b) Machine i generates $\tilde{\mathbf{W}}_i \in \mathbb{R}^{\xi \times w_i}$ with $\xi = O(c/\varepsilon^2)$ to be i.i.d. $\{+1, -1\}$ w.p. $1/2$. *Implicitly* all machines together generate $\tilde{\mathbf{W}} = \begin{pmatrix} \tilde{\mathbf{W}}_1 & \tilde{\mathbf{W}}_2 & \dots & \tilde{\mathbf{W}}_s \end{pmatrix}$, with $\tilde{\mathbf{W}} \in \mathbb{R}^{\xi \times n}$. Machine i computes $\mathbf{H}_i = \tilde{\mathbf{C}}^{\mathrm{T}}(\mathbf{A}_i \mathbf{W}_i^{\mathrm{T}}) \in \mathbb{R}^{c \times \xi}$. Machine i sends $\mathbf{H}_i$ to the server.
   (c) Server computes $\mathbf{\Xi} = \sum_{i=1}^{s} \mathbf{H}_i \in \mathbb{R}^{c \times \xi}$ and sends this back to all the machines. Now machines compute $\mathbf{\Xi} := \mathbf{R}^{-1} \cdot \mathbf{\Xi} \cdot 1/\sqrt{n} (:= \mathbf{Y}^{\mathrm{T}} \mathbf{A} \mathbf{W}^{\mathrm{T}})$, where $\mathbf{W} = \begin{pmatrix} \mathbf{W}_1 & \mathbf{W}_2 & \dots & \mathbf{W}_s \end{pmatrix} \in \mathbb{R}^{\xi \times n}$ is random matrix each element of which is $\{+1/\sqrt{n}, -1/\sqrt{n}\}$ w.p. $1/2$, and then they compute $\mathbf{\Delta} \in \mathbb{R}^{c \times k}$ to be the top k left singular vectors of $\mathbf{\Xi}$. Each machine computes $\mathbf{U} = \mathbf{Y} \cdot \mathbf{\Delta} \in \mathbb{R}^{m \times k}$.

Discussion. A few remarks are necessary for the last two stages of the algorithm. The third stage (adaptive column sampling), implements the adaptive sampling method of Lemma 42. To see this, note that each column in $\mathbf{A}$ (specifically the jth column in $\mathbf{A}_i$) is sampled with probability

$$g_i \cdot q_j^i \geq \frac{1}{2} \cdot \frac{||\mathbf{x}_j^i||_2^2}{\|\mathbf{\Psi}\|_{\mathrm{F}}^2},$$

where $\mathbf{\Psi} = \mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}$. This follows from Lemma 43. Overall, this stage effectively constructs $\hat{\mathbf{C}}$ such that (see Lemma 42)

$$\hat{\mathbf{C}} = AdaptiveCols(\mathbf{A}, \mathbf{C}, c_2, 1/2).$$

The last stage in the algorithm implements the algorithm in Lemma 45. To see this, note that $\mathbf{W}$ satisfies the properties in the lemma. Hence,

$$[\mathbf{Y}, \mathbf{\Delta}] = ApproxSubspaceSVD(\mathbf{A}, \tilde{\mathbf{C}}, k, \varepsilon).$$

7.3 Main Result

The theorem below analyzes the approximation error, the communication complexity, and the running time of the previous algorithm.

Theorem 46. *The matrix $\tilde{\mathbf{C}} \in \mathbb{R}^{m\times c}$ with $c = O(k/\varepsilon)$ columns of $\mathbf{A}$ satisfies w.p. at least* 0.99,

$$\|\mathbf{A} - \tilde{\mathbf{C}}\tilde{\mathbf{C}}^\dagger\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq (1+200\varepsilon)\cdot\|\mathbf{A} - \mathbf{U}_k\mathbf{U}_k^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2. \tag{7}$$

The matrix $\mathbf{U} \in \mathbb{R}^{m\times k}$ with k orthonormal columns satisfies with probability at least 0.98,

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\cdot\|\mathbf{A} - \mathbf{U}_k\mathbf{U}_k^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2. \tag{8}$$

Let each column of $\mathbf{A}$ has at most ϕ non-zero elements. Then, the communication cost of the algorithm is $O\left(sk\phi\varepsilon^{-1} + sk^2\varepsilon^{-4}\right)$ and the running time is $O\left(mn\min\{m,n\} + mns\cdot poly(k,1/\varepsilon)\right)$.

7.4 Proof of Theorem 46

Recal that $\mathbf{A}_i = \mathbf{A}_i\mathbf{V}_i\mathbf{V}_i^{\mathrm{T}} + \mathbf{E}_i \in \mathbb{R}^{m\times w_i}$, i.e., $\mathbf{E}_i = \mathbf{A}_i - \mathbf{A}_i\mathbf{V}_i\mathbf{V}_i^{\mathrm{T}} \in \mathbb{R}^{m\times w_i}$. Also define $m \times w_i$ matrix $\mathbf{D}_i = \mathbf{C}_i\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right)^\dagger\mathbf{V}_i^{\mathrm{T}}$, and $m \times w_i$ matrix $\hat{\mathbf{E}}_i = \mathbf{A}_i - \mathbf{D}_i$. Hence ($\mathbf{D}, \hat{\mathbf{E}} \in \mathbb{R}^{m\times n}$, $\mathbf{A} = \mathbf{D} + \hat{\mathbf{E}}$),

$$\begin{aligned}\mathbf{A} &= \begin{pmatrix}\mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s\end{pmatrix} \\ &= \underbrace{\begin{pmatrix}\mathbf{C}_1\left(\mathbf{V}_1^{\mathrm{T}}\mathbf{S}_1\right)^\dagger\mathbf{V}_1^{\mathrm{T}} & \mathbf{C}_2\left(\mathbf{V}_2^{\mathrm{T}}\mathbf{S}_2\right)^\dagger\mathbf{V}_2^{\mathrm{T}} & \dots & \mathbf{C}_s\left(\mathbf{V}_s^{\mathrm{T}}\mathbf{S}_s\right)^\dagger\mathbf{V}_s^{\mathrm{T}}\end{pmatrix}}_{\mathbf{D}} + \underbrace{\begin{pmatrix}\hat{\mathbf{E}}_1 & \hat{\mathbf{E}}_2 & \dots & \hat{\mathbf{E}}_s\end{pmatrix}}_{\hat{\mathbf{E}}}\end{aligned}$$

The matrix $\mathbf{D}_i$ has rank k and it is in the span of $\mathbf{C}_i$; hence, we expect that the corresponding error defined by the residual matrix $\hat{\mathbf{E}}_i$ will be "small"; this because $\mathbf{C}_i$ was constructed to exactly minimize this "local" column-based matrix reconstruction error $\|\mathbf{A}_i - \mathbf{C}_i\mathbf{C}_i^\dagger\mathbf{A}_i\|_{\mathrm{F}}^2$.

7.4.1 Proof of Eqn. 7

From Lemma 42:

$$\mathbb{E}\left[\|\mathbf{A}-\tilde{\mathbf{C}}\tilde{\mathbf{C}}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2\right] \leq \mathbb{E}\left[\|\mathbf{A}-\Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2\right] \leq \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A}) + \frac{2\varepsilon}{50}\cdot\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2.$$

We further manipulate $\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2$ as follows,

$$\begin{aligned}\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 = \|(\mathbf{D}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}) + (\mathbf{I}_d-\mathbf{C}\mathbf{C}^{\dagger})\hat{\mathbf{E}}\|_{\mathrm{F}}^2 &\leq 2\cdot\left(\|(\mathbf{D}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2 + \|(\mathbf{I}_d-\mathbf{C}\mathbf{C}^{\dagger})\hat{\mathbf{E}}\|_{\mathrm{F}}^2\right)\\ &\leq 2\cdot\left(\|(\mathbf{D}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2 + \|\hat{\mathbf{E}}\|_{\mathrm{F}}^2\right)\end{aligned}$$

The last inequality holds because $\mathbf{I}_d-\mathbf{C}\mathbf{C}^{\dagger}$ is a projection matrix. Combining the two bounds:

$$\mathbb{E}\left[\|\mathbf{A}-\tilde{\mathbf{C}}\tilde{\mathbf{C}}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2\right] \leq \mathbb{E}\left[\|\mathbf{A}-\Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2\right] \leq \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A}) + \frac{2\varepsilon}{25}\cdot\left(\|\mathbf{D}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2 + \|\hat{\mathbf{E}}\|_{\mathrm{F}}^2\right). \quad (9)$$

Lemma 47. $\|\hat{\mathbf{E}}\|_{\mathrm{F}}^2 \leq 5\sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A})$

Proof.

$$\begin{aligned}
\|\hat{\mathbf{E}}\|_{\mathrm{F}}^2 &= \sum_{i=1}^{s}\|\mathbf{A}_i-\mathbf{D}_i\|_{\mathrm{F}}^2\\
&= \sum_{i=1}^{s}\|\mathbf{A}_i\mathbf{V}_i\mathbf{V}_i^{\mathrm{T}} + \mathbf{E}_i - \mathbf{C}_i\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}\mathbf{V}_i^{\mathrm{T}}\|_{\mathrm{F}}^2\\
&= \sum_{i=1}^{s}\|\mathbf{A}_i\mathbf{V}_i\mathbf{V}_i^{\mathrm{T}} - \underbrace{\left(\mathbf{A}_i\mathbf{V}_i\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i+\mathbf{E}_i\mathbf{S}_i\right)}_{\mathbf{C}_i}\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}\mathbf{V}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 + \sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2\\
&= \sum_{i=1}^{s}\|\mathbf{A}_i\mathbf{V}_i\mathbf{V}_i^{\mathrm{T}} - \mathbf{A}_i\mathbf{V}_i\underbrace{\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}}_{\mathbf{I}_k}\mathbf{V}_i^{\mathrm{T}} + \mathbf{E}_i\mathbf{S}_i\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}\mathbf{V}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 + \sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2\\
&= \sum_{i=1}^{s}\|\mathbf{E}_i\mathbf{S}_i\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}\mathbf{V}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 + \sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2\\
&= \sum_{i=1}^{s}\|\mathbf{E}_i\mathbf{S}_i\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}\|_{\mathrm{F}}^2 + \sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2\\
&\leq \sum_{i=1}^{s}\|\mathbf{E}_i\mathbf{S}_i\|_{\mathrm{F}}^2\cdot\|\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}\|_2^2 + \sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2\\
&\leq \sum_{i=1}^{s}4\cdot\|\mathbf{E}_i\|_{\mathrm{F}}^2 + \sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2\\
&= 5\cdot\sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2\\
&\leq 5\cdot\sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A})
\end{aligned}$$

The third equality follows from the matrix pythagoras theorem. Next, we explain the equality $\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\big(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\big)^{\dagger} = \mathbf{I}_k$. Recall that $\mathbf{V}_i$ is a $w_i \times k$ matrix with $w_i > k$. Also, the construction of $\mathbf{S}_i \in \mathbb{R}^{w_i \times \ell}$ with $\ell > k$ and Lemma 40 imply that $\mathbf{V}_i^{\mathrm{T}}\mathbf{S} := \mathbf{G}_i$ is an $k \times \ell$ matrix with $\mathrm{rank}(\mathbf{G}_i) = k$. Now write the SVD of $\mathbf{G}_i$ as $\mathbf{G}_i = \mathbf{U}_{\mathbf{G}_i}\mathbf{\Sigma}_{\mathbf{G}_i}\mathbf{V}_{\mathbf{G}_i}^{\mathrm{T}}$ with $\mathbf{U}_{\mathbf{G}_i} \in \mathbb{R}^{k\times k}$, $\mathbf{\Sigma}_{\mathbf{G}_i} \in \mathbb{R}^{k\times k}$ and $\mathbf{V}_{\mathbf{G}_i} \in \mathbb{R}^{w_i \times k}$. Also, ${\mathbf{G}_i}^{\dagger} = \mathbf{V}_{\mathbf{G}_i}\mathbf{\Sigma}_{\mathbf{G}_i}^{-1}\mathbf{U}_{\mathbf{G}_i}$. Finally,

$$\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\Big(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger} = \mathbf{U}_{\mathbf{G}_i}\mathbf{\Sigma}_{\mathbf{G}_i}\mathbf{V}_{\mathbf{G}_i}^{\mathrm{T}}\mathbf{V}_{\mathbf{G}_i}\mathbf{\Sigma}_{\mathbf{G}_i}^{-1}\mathbf{U}_{\mathbf{G}_i} = \mathbf{U}_{\mathbf{G}_i}\mathbf{U}_{\mathbf{G}_i}^{\mathrm{T}} = \mathbf{I}_k.$$

The second inequality follows from Lemma 40 (we apply this lemma to each $\mathbf{V}_i$ and notice that we chose $\ell = 4k$, hence $\|\big(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\big)^{\dagger}\|_2^2 = 1/\sigma_k^2\left(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\right) \leq (1 - \sqrt{k/\ell})^{-2} = 4$). The last inequality follows because

$$\sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2 = \sum_{i=1}^{s}\|\mathbf{A}_i - \mathbf{A}_i\mathbf{V}_i\mathbf{V}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq \sum_{i=1}^{s}\|\mathbf{A}_i - \mathbf{A}_i\mathbf{V}_{opt}\mathbf{V}_{opt}^{\mathrm{T}}\|_{\mathrm{F}}^2 = \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A}).$$

Here, $\mathbf{V}_{opt} \in \mathbb{R}^{n\times k}$ contains the top k right singular vectors of $\mathbf{A}$. The inequality follows from the "optimality" of $\mathbf{V}_i$ for each submatrix $\mathbf{A}_i$. ■

Lemma 48. $\|\mathbf{D} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2 \leq 20 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A})$

Proof.

$$\begin{aligned}
\|\mathbf{D} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2 &= \sum_{i=1}^{s}\|\mathbf{C}_i\Big(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger}\mathbf{V}_i^{\mathrm{T}} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{C}_i\Big(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger}\mathbf{V}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \\
&= \sum_{i=1}^{s}\|\mathbf{C}_i\Big(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{C}_i\Big(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger}\|_{\mathrm{F}}^2 \\
&\leq \sum_{i=1}^{s}\|\mathbf{C}_i - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{C}_i\|_{\mathrm{F}}^2 \cdot \|\Big(\mathbf{V}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger}\|_2^2 \\
&\leq \sum_{i=1}^{s}4 \cdot \|\mathbf{C}_i - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{C}_i\|_{\mathrm{F}}^2 \\
&= 4 \cdot \|\mathbf{G} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{G}\|_{\mathrm{F}}^2 \\
&\leq 4 \cdot 5 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{G})}\sigma_i^2(\mathbf{G}) \\
&\leq 20 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{G})}\sigma_i^2(\mathbf{A}) \\
&\leq 20 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A})
\end{aligned}$$

The second inequality follows from Lemma 40 and the third inequality follows from Theorem 41. ■

Concluding the proof. Replacing the bounds for $\|\mathbf{D} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2$, $\|\hat{\mathbf{E}}\|_{\mathrm{F}}^2$ in Eqn. 9 we obtain [5]:

$$\mathbb{E}\left[\|\mathbf{A} - \tilde{\mathbf{C}}\tilde{\mathbf{C}}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2\right] \leq \mathbb{E}\left[\|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2\right] \leq \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}) + 2\varepsilon \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}). \quad (10)$$

The expectation is taken w.r.t. the randomness in constructing $\tilde{\mathbf{C}}$; hence, the term $\sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A})$ is a constant w.r.t. the expectation operation, which means that Eqn. 10 implies the bound:

$$\mathbb{E}\left[\|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 - \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A})\right] \leq 2\varepsilon \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}).$$

Let $Y := \|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 - \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A})$. Y is a random variable with $Y \geq 0$, because $\mathrm{rank}(\Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})) \leq k$. Markov's inequality on Y implies that w.p. 0.99,

$$\|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 - \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}) \leq 200\varepsilon \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}),$$

equivalently,

$$\|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}) + 200\varepsilon \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}).$$

Using $\|\mathbf{A} - \tilde{\mathbf{C}}\tilde{\mathbf{C}}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2$ concludes the proof.

7.4.2 Proof of Eqn. 8

We would like to apply Lemma 45 for the matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$ in the algorithm. Note that we have already proved that the matrix $\tilde{\mathbf{C}}$ in the algorithm satisfies with probability at least 0.99: $\|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq (1 + O(\varepsilon))\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$. Lemma 45 and a simple union bound conclude the proof.

7.4.3 Running time

Next, we give an analysis of the arithmetic operations required in various steps in the algorithm.

1. **Local Column Sampling**: $O(mn \min\{m, n\} + nk^3)$ arithmetic operations in total.
   (a) $O(w_i m \min\{w_i, m)$ for each local SVD; $O(w_i k^3 + w_i m)$ for each $\mathbf{S}_i$
   (b) -
2. **Global Column Sampling**: $O(sk^3 + skm)$ arithmetic operations in total.
   (a) $O(sk^3 + skm)$ to construct $\mathbf{C}$.
   (b) -

[5] An interesting intermediate bound that holds here is $\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq 50 \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A})$; and notice that $\mathbf{C}$ has $4k$ columns of $\mathbf{A}$ which are constructed deterministically and distributed with one round of communication.

3. **Adaptive Sampling** $O(sk^2m + mnk + k/\varepsilon)$ arithmetic operations in total.
   (a) $O(k^2m)$ for each $\mathbf{C}^\dagger$ locally; $O(w_imk)$ for each $\mathbf{\Psi}_i$ and $O(mw_i)$ for each ψ_i.
   (b) $O(s)$ in total.
   (c) $O(k/\varepsilon)$ in total using the method of [46] locally.
   (d) $O(k/\varepsilon)$.
4. **Rank-k matrix in** $span(\tilde{\mathbf{C}})$: $O(mnk/\varepsilon^3 + smk^2/\varepsilon^4 + sk^3/\varepsilon^5)$ arithmetic operations in total.
   (a) $O(smk^2/\varepsilon^2)$ in total
   (b) $O(mw_ik/\varepsilon^3)$ for each $\mathbf{A}_i\mathbf{W}_i^{\mathrm{T}}$ and another $O(mk^2\varepsilon^{-4})$ for each $\tilde{\mathbf{C}}^{\mathrm{T}}(\mathbf{A}_i\mathbf{W}_i^{\mathrm{T}})$
   (c) $O(sk^2\varepsilon^{-3})$ to find $\mathbf{\Xi}$ in the server; then another $O(sk^3\varepsilon^{-3} + sk^3\varepsilon^{-5}))$ to update $\mathbf{\Xi}$ locally in each machine and another $O(sk^3\varepsilon^{-5})$ to find $\mathbf{\Delta}$ locally in each machine; and $O(smk^2/\varepsilon)$ to find $\mathbf{U}$ locally in each machine.

7.4.4 Communication Complexity

Assume that we can represent each element in the input matrix $\mathbf{A}$ with at most b bits. Also, we assume that one word has length b bits and $b = O(\log(mns/\varepsilon))$.

1. **Local Column Sampling**: $O(sk\phi)$ words.
   (a) -
   (b) $O(sk\phi)$ elements of $\mathbf{A}$.
2. **Global Column Sampling**:$O(sk\phi)$ words.
   (a) -
   (b) $O(sk\phi)$ elements of $\mathbf{A}$.
3. **Adaptive Sampling**: $O(s + \phi k/\varepsilon)$ words.
   (a) s integers each of which is representable with $O(\log k + \log\log(mns/\varepsilon))$ (from Lemma 44).
   (b) s integers (the t_i's) each with magnitude at most n, hence representable with b bits.
   (c) $O(\phi k/\varepsilon)$ elements of $\mathbf{A}$.
   (d) -
4. **Best rank-k matrix in the span of** $\tilde{\mathbf{C}}$: $O(s\phi k/\varepsilon + sk^2\varepsilon^{-4})$ words.
   (a) $O(sk\phi\varepsilon^{-1})$ elements of $\mathbf{A}$.
   (b) $O(sk^2\varepsilon^{-4})$ numbers each of which can be represented with b bits.
   (c) $O(sk^2\varepsilon^{-4})$ numbers each of which can be represented with b bits.

In total the communication complexity is $O(sk\phi/\varepsilon + sk^2/\varepsilon^4)$ words.

8 Faster Distributed PCA for sparse matrices

We now explain how to modify certain steps of the distributed PCA algorithm in Section 7 in order to improve the total running time spent to compute the matrix $\mathbf{U}$; this, at the cost of increasing the communication cost *slightly*. Specifically, we replace the parts 1-(a), 2-(a), 3-(a), and 4-(b) with similar procedures, which are almost as accurate as the original procedures but run in time proportional to the number of the non-zero elements of the underlying matrices. Before presenting the new algorithm in detail, we discuss results from previous literature that we employ in the analysis. We remark that in the analysis below we have not attempted to optimize the constants.

8.1 Background material

8.1.1 Constant probability sparse subspace embeddings and sparse SVD

First, we present the so-called sparse subspace embedding matrices of Clarskon and Woodruff [19].

Definition 49. *[Sparse Subspace Embedding [19]] We call $\mathbf{W} \in \mathbb{R}^{\xi \times n}$ a sparse subspace embedding of dimension $\xi < n$ if it is constructed as follows, $\mathbf{W} = \mathbf{\Psi} \cdot \mathbf{Y}$, where*

- *$h : [n] \to [\xi]$ is a random map so that for each $i \in [n], h(i) = \xi'$, for $\xi' \in [\xi]$ w.p. $1/\xi$.*
- *$\mathbf{\Psi} \in \mathbb{R}^{\xi \times n}$ is a binary matrix with $\mathbf{\Psi}_{h(i),i} = 1$, and all remaining entries 0.*
- *$\mathbf{Y} \in \mathbb{R}^{n \times n}$ is a random diagonal matrix, with each diagonal entry independently chosen to be $+1$ or -1, with equal probability.*

Such sparse subspace embedding matrices have favorable properties which we summarize below:

Lemma 50 ([19])**.** *Let $\mathbf{A}^{\mathrm{T}} \in \mathbb{R}^{n \times m}$ have rank ρ and let $\mathbf{W} \in \mathbb{R}^{\xi \times n}$ be a randomly chosen sparse subspace embedding with dimension $\xi = \Omega(\rho^2 \varepsilon^{-2})$, for some $0 < \varepsilon < 1$. Then, 1) computing $\mathbf{A}\mathbf{W}^{\mathrm{T}}$ requires $O(\mathrm{nnz}(\mathbf{A}))$ time; 2) $\mathrm{nnz}(\mathbf{A}\mathbf{W}^{\mathrm{T}}) \leq \mathrm{nnz}(\mathbf{A})$; and 3) with probability at least 0.99, and for all vectors $\mathbf{y} \in \mathbb{R}^m$ simultaneously,*

$$(1-\varepsilon)\|\mathbf{A}^{\mathrm{T}}\mathbf{y}\|_2^2 \leq \|\mathbf{W}\mathbf{A}^{\mathrm{T}}\mathbf{y}\|_2^2 \leq (1+\varepsilon)\|\mathbf{A}^{\mathrm{T}}\mathbf{y}\|_2^2.$$

Notice that the third property in the lemma fails with constant probability. Based on the above sparse subspace embeddings, Clarkson and Woodruff [19] described a low-rank matrix approximation algorithm for $\mathbf{A}$ that runs in time $\mathrm{nnz}(\mathbf{A})$ plus low-order terms.

Lemma 51 (Theorem 47 in [19]; for the exact statement see Lemma 3.4 in [17])**.** *Given $\mathbf{A} \in \mathbb{R}^{m \times n}$ of rank ρ, a target rank $1 \leq k < \rho$, and $0 < \epsilon \leq 1$, there exists a randomized algorithm that computes $\mathbf{Z} \in \mathbb{R}^{n \times k}$ with $\mathbf{Z}^{\mathrm{T}}\mathbf{Z} = \mathbf{I}_k$ and with probability at least 0.99,*

$$\|\mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq (1+\epsilon)\,\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

The proposed algorithm requires $O\left(\mathrm{nnz}(\mathbf{A}) + (m+n)\cdot poly(k, \varepsilon^{-1})\right)$ arithmetic operations. We denote this procedure as $\mathbf{Z} = SparseSVD(\mathbf{A}, k, \varepsilon)$.

Notice that the algorithm in the lemma fails with constant probability.

8.1.2 High probability sparse SVD

Lemma 51 above presents a randomized algorithm to compute quickly an SVD of a sparse matrix; however, this lemma fails with constant probability. Since in our fast distributed PCA algorithm we will use the lemma multiple times (specifically, s times, since we need such a fast SVD locally in each machine), the overall failure probability would be of the order $O(s)$ (by a union bound). To boost this failure probability, we develop a simple scheme to obtain a high probability result.

Before we proceed, we quickly review the Johnson Lindestrauss transform, which we need in the analysis.

Lemma 52. *[Theorem 1 in [5] for fixed $\varepsilon = 1/2$] Let $\mathbf{B} \in \mathbb{R}^{m\times n}$. Given $\beta > 0$, let $r = \frac{4+2\beta}{(1/2)^2-(1/2)^3}\log n$. Construct a matrix $\mathbf{S} \in \mathbb{R}^{r\times m}$, each element of which is a random variable which takes values $\pm 1/\sqrt{r}$ with equal probability. Let $\tilde{\mathbf{B}} = \mathbf{SB}$. Then, if $\mathbf{b}_i$ and $\tilde{\mathbf{b}}_i$ denote the ith column of $\mathbf{B}$ and $\tilde{\mathbf{B}}$, respectively, with probability at least $1-n^{-\beta}$, and for all $i = 1, ...n$, $(1-\frac{1}{2})\|\mathbf{b}_i\|_2^2 \leq \|\tilde{\mathbf{b}}_i\|_2^2 \leq (1+\frac{1}{2})\|\mathbf{b}_i\|_2^2$. Given $\mathbf{B}$, it takes $O(\mathrm{nnz}(\mathbf{B})\log n)$ arithmetic operations to construct $\tilde{\mathbf{B}}$. We will denote this procedure as $\tilde{\mathbf{B}} = JLT(\mathbf{B}, \beta)$.*

Now, we are fully equipped to design the "high-probability" analog of Lemma 51:

Lemma 53. *Given $\mathbf{A} \in \mathbb{R}^{m\times n}$ of rank ρ, a target rank $1 \leq k < \rho$, and $0 < \delta, \epsilon \leq 1$, there exists a randomized algorithm that computes $\mathbf{Z} \in \mathbb{R}^{n\times k}$ with $\mathbf{Z}^{\mathrm{T}}\mathbf{Z} = \mathbf{I}_k$ and with probability at least $1-\delta-1/n$, $\|\mathbf{A} - \mathbf{AZZ}^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq 3(1+\epsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$. The proposed algorithm requires $O\left(\mathrm{nnz}(\mathbf{A})\log^2(\frac{n}{\delta})\right) + n \cdot poly(k, \varepsilon, \log(\frac{n}{\delta}))$ arithmetic operations. We denote this procedure as $\mathbf{Z} = SparseSVDBoosting(\mathbf{A}, k, \varepsilon, \delta)$.*

Proof. The idea is to use the algorithm in Lemma 51 and generate i.i.d. $r = O(\log(\frac{1}{\delta}))$ matrices $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_r$, where $\mathbf{Z}_i = SparseSVD(\mathbf{A}, k, \varepsilon)$, and $\|\mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$, with probability at least 0.99 for a single $\mathbf{Z}_i$.

Now, we could have chosen $\mathbf{Z} := \mathbf{Z}_i$ which minimizes $\|\mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2$, and this would have implied that $\|\mathbf{A} - \mathbf{AZZ}^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$, with probability $1-\delta$ (via a simple chernoff argument), but evaluating the errors $\|\mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2$ is computationally intensive for the overall running time that we would like to obtain.

Selection of Z. Instead, we use the Johnson Lindestrauss transform to speedup this computation (evaluating the errors). Hence, for every $\mathbf{Z}_i$, we (implicitly) set $\mathbf{B}_i := \mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}$ and then we compute $\tilde{\mathbf{B}}_i = JLT(\mathbf{B}_i, 1)$ (see Lemma 52). Now, we choose the $\mathbf{Z}_i$ that minimizes $\|\tilde{\mathbf{B}}_i\|_{\mathrm{F}}^2$.

Correctness. For a fixed $\mathbf{Z}_i$, from Lemma 52 it follows that with probability $1-1/n$ it is: $\frac{1}{2}\|\mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq \|\tilde{\mathbf{B}}_i\|_{\mathrm{F}}^2 \leq \frac{3}{2}\|\mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2$, where the failure probability is over the randomness of $\mathbf{S}$ in Lemma 52. Now, for some $\mathbf{Z}_i$ with probability at least 0.99 it is $\|\mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$, where the failure probability is over the randomness in constructing $\mathbf{Z}_i$. Overall, given that $n > 1$, with probability at least 0.49 (over the randomness of both $\mathbf{S}$ and $\mathbf{Z}_i$; the failure probability follows via a union bound) for each single $\mathbf{Z}_i$ it is: $\|\tilde{\mathbf{B}}_i\|_{\mathrm{F}}^2 \leq \frac{2}{3}(1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$. Hence, since $r = O(\log\frac{1}{\delta})$ and we have picked $\mathbf{Z}$ as the $\mathbf{Z}_i$ that minimizes $\|\tilde{\mathbf{B}}_i\|_{\mathrm{F}}^2$, it follows that with probability at least $1-\delta$, $\|\tilde{\mathbf{B}}_i\|_{\mathrm{F}}^2 \leq \frac{2}{3}(1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$. Combining with $\frac{1}{2}\|\mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq \|\tilde{\mathbf{B}}_i\|_{\mathrm{F}}^2$, and using a union bound, it follows that with probability $1-1/n-\delta$: $\|\mathbf{A} - \mathbf{AZ}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq \frac{6}{2}(1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$.

Running time. The following costs occur $O(\log(\frac{1}{\delta}))$ times: (i) $O\left(\mathrm{nnz}(\mathbf{A}) + (m+n)\cdot poly(k, \varepsilon^{-1})\right)$ to find $\mathbf{Z}_i$ (via Lemma 51); (ii) $O(\mathrm{nnz}(\mathbf{A})\log n + nk\log(n))$ to evaluate each cost, i.e., find $\|\tilde{\mathbf{B}}_i\|_{\mathrm{F}}^2$.

■

8.1.3 Fast column sampling techniques

Section 7.1.1 presented the column sampling algorithms that we used in our distributed PCA algorithm in Section 7. Next, we present the "fast" analogs of those algorithms. To design such fast analogs we employ the sparse subspace embedding matrices in the previous section. All the results in this section developed in [17]. There is a small difference in Lemma 54 below, hence we present a short proof for completeness. Specifically, the original result in [17] has a constant failure probability; here, via using standard arguments, we extended this result to a high probability bound.

Lemma 54 (Input-Sparsity-Time Dual-Set Spectral-Frobenius Sparsification [17].)**.** *Let $\mathbf{V} \in \mathbb{R}^{w\times k}$ be a matrix with $w > k$ and $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}_k$. Let $\mathbf{E} \in \mathbb{R}^{m\times w}$ be an arbitrary matrix. Let δ be a failure probability parameter.*

For $i = 1, 2, \ldots, r = O(\log \frac{1}{\delta})$, let $\mathbf{B}_i = \mathbf{E}\mathbf{W}_i^{\mathrm{T}} \in \mathbb{R}^{m\times\xi}$, where $\mathbf{W}_i \in \mathbb{R}^{\xi\times w}$ is a randomly chosen sparse subspace embedding with $\xi = O(k^2/\varepsilon^2) < w$, for some $0 < \varepsilon < 1$ (see Definition 49 and Lemma 50), and run the algorithm of Lemma 40 with $\mathbf{V}$, $\mathbf{B}_i$, and some ℓ with $k < \ell \leq w$, to construct $\mathbf{S}_i \in \mathbb{R}^{w\times\ell}$. Choose some $\mathbf{S}$ from the $\mathbf{S}_i$'s - see the proof for the details.

Then, with probability at least $1-\delta$, $\sigma_k^2\left(\mathbf{V}^{\mathrm{T}}\mathbf{S}\right) \geq \left(1-\sqrt{k/\ell}\right)^2$ and $\|\mathbf{E}\mathbf{S}\|_{\mathrm{F}}^2 \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2 \cdot \|\mathbf{E}\|_{\mathrm{F}}^2$. The matrix $\mathbf{S}$ can be computed in $O\left(\log\left(\frac{1}{\delta}\right)\cdot\left(\mathrm{nnz}(\mathbf{A}) + \ell w k^2 + mk^2/\varepsilon^2 + m\ell\right)\right)$ time. We denote this procedure as $\mathbf{S} = BssSamplingSparse(\mathbf{V}, \mathbf{A}, r, \varepsilon, \delta)$.

Proof. The algorithm constructs the $\mathbf{S}_i$'s as follows, $\mathbf{S}_i = BssSampling(\mathbf{V}, \mathbf{B}_i, r)$ - see Lemma 40. From that lemma, we also have that $\sigma_k^2\left(\mathbf{V}^{\mathrm{T}}\mathbf{S}_i\right) \geq \left(1-\sqrt{k/\ell}\right)^2$ and $\|\mathbf{B}_i^{\mathrm{T}}\mathbf{S}\|_{\mathrm{F}}^2 \leq \|\mathbf{B}_i^{\mathrm{T}}\|_{\mathrm{F}}^2$, i.e., $\|\mathbf{W}_i\mathbf{A}^{\mathrm{T}}\mathbf{S}_i\|_{\mathrm{F}}^2 \leq \|\mathbf{W}_i\mathbf{A}^{\mathrm{T}}\|_{\mathrm{F}}^2$. Since $\mathbf{W}_i$ is a subspace embedding, from Lemma 50 we have that with probability at least 0.99 and for *all* vectors $\mathbf{y} \in \mathbb{R}^n$ simultaneously, $(1-\varepsilon)\|\mathbf{A}^{\mathrm{T}}\mathbf{y}\|_2^2 \leq \|\mathbf{W}_i\mathbf{A}^{\mathrm{T}}\mathbf{y}\|_2^2$. Apply this r times for $\mathbf{y} \in \mathbb{R}^n$ being columns from $\mathbf{S}_i \in \mathbb{R}^{n\times r}$ and take a sum on the resulting inequalities: $(1-\varepsilon)\|\mathbf{A}^{\mathrm{T}}\mathbf{S}_i\|_{\mathrm{F}}^2 \leq \|\mathbf{W}\mathbf{A}^{\mathrm{T}}\mathbf{S}_i\|_{\mathrm{F}}^2$; also, apply this n times for the basis vectors in $\mathbb{R}^n$ and take a sum on the resulting inequalities: $\|\mathbf{W}\mathbf{A}^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathbf{A}^{\mathrm{T}}\|_{\mathrm{F}}^2$. Combining all these inequalities together, we conclude that with probability at least 0.99, $\|\mathbf{A}^{\mathrm{T}}\mathbf{S}_i\|_{\mathrm{F}}^2 \leq \frac{1+\varepsilon}{1-\varepsilon}\cdot\|\mathbf{A}^{\mathrm{T}}\|_{\mathrm{F}}^2$.

To summarize, for each $\mathbf{S}_i$ and with probability at least 0.99, $\sigma_k^2\left(\mathbf{V}^{\mathrm{T}}\mathbf{S}_i\right) \geq \left(1-\sqrt{k/\ell}\right)^2$ and $\|\mathbf{E}\mathbf{S}_i\|_{\mathrm{F}}^2 \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\cdot\|\mathbf{E}\|_{\mathrm{F}}^2$.

Selection of S. Now that we have r matrices $\mathbf{S}_i$ each of which satisfying the above bounds with probability 0.99 we select one of them as follows: 1) we compute all $\sigma_k^2\left(\mathbf{V}^{\mathrm{T}}\mathbf{S}_i\right)$ and sort them from the largest to the smallest value; 2) we compute all $\|\mathbf{A}^{\mathrm{T}}\mathbf{S}_i\|_{\mathrm{F}}^2$ and sort them from the smallest to the largest value; 3) we choose an $\mathbf{S}_i$ which occurs in the top $2/3$ fraction of each of the lists.

Correctness. Let X_i be an indicator random variable: $X_i = 1$ if the ith matrix $\mathbf{S}_i$ satisfies $\sigma_k^2\left(\mathbf{V}^{\mathrm{T}}\mathbf{S}_i\right) \geq \left(1-\sqrt{k/\ell}\right)^2$, and $X_i = 0$, otherwise. Then, $\Pr[X_i] \geq 0.99$. Let $X = \sum_i^r X_i$. The expected value of X is $\mu_X = \mathbb{E}[X] = 0.99r$. A standard Chernoff bound applied on X gives: $\Pr[X \leq (1-\alpha)\mu_X] \leq e^{-\frac{\mu_X\alpha}{2}}$. Let $(1-\alpha)0.99r = 0.75r$, i.e., $\alpha = 0.14/0.99$. Applying this to the Chernoff bound, we get: $\Pr[X \leq 0.75r] \leq e^{-O(r)}$. Replacing $r = O(\log(1/\delta))$: $\Pr[X \leq 0.75r] \leq \delta/2$. This means precisely that $3/4$ of the fraction of the $\mathbf{S}_i$'s satisfy with probability $1-\delta/2$: $\sigma_k^2\left(\mathbf{V}^{\mathrm{T}}\mathbf{S}_i\right) \geq \left(1-\sqrt{k/\ell}\right)^2$.

Similarly, let Y_i be an indicator random variable: $Y_i = 1$ if the ith matrix $\mathbf{S}_i$ satisfies $\|\mathbf{E}\mathbf{S}_i\|_{\mathrm{F}}^2 \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2 \cdot \|\mathbf{E}\|_{\mathrm{F}}^2$, and $Y_i = 0$, otherwise. Then, $\Pr[Y_i] \geq 0.99$. Let $Y = \sum_i^r X_i$. The expected value of Y is $\mu_Y = \mathbb{E}[Y] = 0.99r$. A standard Chernoff bound applied on Y gives: $\Pr[Y \leq (1-\alpha)\mu_Y] \leq e^{-\frac{\mu_Y \alpha}{2}}$. Let $(1-\alpha)0.99r = 0.75r$, i.e., $\alpha = 0.14/0.99$. Applying this to the Chernoff bound, we get: $\Pr[Y \leq 0.75r] \leq e^{-O(r)}$. Replacing $r = O(\log(1/\delta))$: $\Pr[Y \leq 0.75r] \leq \delta/2$. This means precisely that $3/4$ of the fraction of the $\mathbf{S}_i$'s satisfy w.p. $1 - \delta/2$: $\|\mathbf{E}\mathbf{S}_i\|_{\mathrm{F}}^2 \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2 \cdot \|\mathbf{E}\|_{\mathrm{F}}^2$.

Now a simple union bound indicates that with probability at least $1 - \delta$, the following two events happen simultaneously: $X > 0.75r$ and $Y > 0.75r$. Since we select an element $\mathbf{S}_i$ on top of the two sorted lists, it follows that this element should have $X_i = Y_i = 1$.

Running time. The following costs occur $O(\log(\frac{1}{\delta}))$ times: (i) $O(\mathrm{nnz}(\mathbf{A}))$ to find $\mathbf{B}_i$. (i) $O(\ell w k^2 + m\xi)$ to find $\mathbf{S}_i$ (via Lemma 40); and (iii) $O(k^2\ell + m\ell)$ to evaluate each cost and, i.e., find $\sigma_k^2\left(\mathbf{V}^{\mathrm{T}}\mathbf{S}_i\right)$ and $\|\mathbf{E}\mathbf{S}_i\|_{\mathrm{F}}^2$. Additionally, $O(\log(\frac{1}{\delta})\log\log(\frac{1}{\delta}))$ time in total is spent in sorting. ■

The following Lemma is the "fast" analog of Lemma 41. The failure probability in the lemma is constant, but this is sufficient for our purposes since we apply this lemma only once in the global column sampling step of the distributed PCA algorithm in Section 8.2.

Lemma 55 (Input-Sparsity-Time constant factor column-based matrix reconstruction; Lemma 6.3 in [17])**.** *Given matrix $\mathbf{G} \in \mathbb{R}^{m\times\alpha}$ of rank ρ and a target rank k [6], there exists a randomized algorithm that runs in $O\left(\mathrm{nnz}(\mathbf{A}) \cdot \log\alpha + m \cdot poly(\log\alpha, k, \varepsilon^{-1})\right)$ time and selects $c = 4k$ columns of $\mathbf{G}$ to form a matrix $\mathbf{C} \in \mathbb{R}^{m\times c}$ such that with probability at least 0.69:*

$$\|\mathbf{G} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{G}\|_{\mathrm{F}}^2 \leq 4820 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{G})} \sigma_i^2(\mathbf{G}).$$

We denote this procedure as $\mathbf{C} = DeterministicCssFrobeniusSparse(\mathbf{G}, k, c)$.

Finally, the lemma below presents the "fast" analog of Lemma 42. The failure probability in the lemma is, again, constant; we have not attempted to obtain a high probability bound since we employ this lemma only once in the algorithm in Section 8.2.

Lemma 56 (Input-sparsity-time Adaptive Sampling [17])**.** *Given $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{V} \in \mathbb{R}^{m\times c_1}$ (with $c_1 \leq n, m$), define the residual matrix $\mathbf{\Psi} = \mathbf{A} - \mathbf{V}\mathbf{V}^{\dagger}\mathbf{A} \in \mathbb{R}^{m\times n}$. Let $\tilde{\mathbf{\Psi}} = JLT(\mathbf{\Psi}, 1)$. For $j = 1, \ldots, n$, let p_j be a probability distribution such that, for some positive constant $\beta \leq 1$, $p_j \geq \beta\|\tilde{\mathbf{\Psi}}^{(j)}\|_2^2/\|\tilde{\mathbf{\Psi}}\|_{\mathrm{F}}^2$, where $\tilde{\mathbf{\Psi}}^{(j)}$ is the j-th column of the matrix $\tilde{\mathbf{\Psi}}$. Sample c_2 columns from $\mathbf{A}$ in c_2 i.i.d. trials, where in each trial the j-th column is chosen with probability p_i. Let $\mathbf{C}_2 \in \mathbb{R}^{m\times c_2}$ contain the c_2 sampled columns and let $\mathbf{C} = [\mathbf{V}\ \ \mathbf{C}_2] \in \mathbb{R}^{m\times(c_1+c_2)}$ contain the columns of $\mathbf{V}$ and $\mathbf{C}_2$. Then, for any integer $k > 0$, and with probability $0.9 - \frac{1}{n}$*

$$\|\mathbf{A} - \Pi_{\mathbf{C},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 + \frac{30 \cdot k}{\beta \cdot c_2}\|\mathbf{A} - \mathbf{V}\mathbf{V}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2.$$

Given $\mathbf{A}$ and $\mathbf{V}$, the algorithm takes $O(\mathrm{nnz}(\mathbf{A})\log n + mc_1\log n + mc_1^2)$ time to find $\mathbf{C}_2$. We denote this sampling procedure as $\mathbf{C}_2 = AdaptiveColsSparse(\mathbf{A}, \mathbf{V}, c_2, \beta)$.

[6] The original Lemma 6.3 in [17] has the assumption that $k < \rho$, but this assumption can be dropped having the result unchanged. The only reason the assumption $k < \rho$ exists is because otherwise column subset selection is trivial.

8.1.4 Fast Low-rank matrix approximations within a subspace

Finally, we present the fast analog of the result in Section 7.1.3. The failure probability in the lemma is, again, constant; we have not attempted to obtain a high probability bound since we employ this lemma only once in the algorithm in Section 8.2.

Lemma 57. *Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be the input matrix and $\mathbf{V} \in \mathbb{R}^{m\times c}$ be the input subspace. We further assume that for some rank parameter $k < c$ and accuracy parameter $0 < \varepsilon < 1$:*

$$\|\mathbf{A} - \Pi^{\mathrm{F}}_{\mathbf{V},k}(\mathbf{A})\|^2_{\mathrm{F}} \leq (1 + O(\varepsilon))\|\mathbf{A} - \mathbf{A}_k\|^2_{\mathrm{F}}.$$

Let $\mathbf{V} = \mathbf{Y}\mathbf{\Psi}$ be a qr decomposition of $\mathbf{V}$ with $\mathbf{Y} \in \mathbb{R}^{m\times c}$ and $\mathbf{\Psi} \in \mathbb{R}^{c\times c}$. Let $\mathbf{\Xi} = \mathbf{Y}^{\mathrm{T}}\mathbf{A}\mathbf{W}^{\mathrm{T}} \in \mathbb{R}^{c\times\xi}$, where $\mathbf{W}^{\mathrm{T}} \in \mathbb{R}^{n\times\xi}$ with $\xi = O(c/\varepsilon^2)$, is a sparse subspace embedding matrix (see Definition 49 and Lemma 50). Let $\mathbf{\Delta} \in \mathbb{R}^{c\times k}$ contain the top k left singular vectors of $\mathbf{\Xi}$. Then, with probability at least 0.99,

$$\|\mathbf{A} - \mathbf{Y}\mathbf{\Delta}\mathbf{\Delta}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\mathbf{A}\|^2_{\mathrm{F}} \leq (1 + \varepsilon)\|\mathbf{A} - \mathbf{A}_k\|^2_{\mathrm{F}}.$$

$\mathbf{Y}$ and $\mathbf{\Delta}$ can be computed in $O(\mathrm{nnz}(\mathbf{A}) + mc\xi)$ time. We denote this procedure as

$$[\mathbf{Y}, \mathbf{\Delta}] = ApproxSubspaceSVDSparse(\mathbf{A}, \mathbf{V}, k, \varepsilon).$$

Proof. This result was proven inside the proof of Theorem 1.5 in [33]. Specifically, the error bound proven in [33] is for the transpose of $\mathbf{A}$ (also $\mathbf{Y}, \mathbf{\Delta}$ are denoted with U, V in [33]). The only requirement for the embedding matrix $\mathbf{W}$ (denoted with P in the proof of Theorem 1.5 in [33]) is to be a subspace embedding for $\mathbf{Y}^{\mathrm{T}}\mathbf{A}$, in the sense that $\mathbf{W}$ is a subspace embedding for $\mathbf{A}$ in Lemma 50. Since our choice of $\mathbf{W}$ with $\xi = O(c^2/\varepsilon^2)$ satisfies this requirement we omit the details of the proof. The running time is $O(\mathrm{nnz}(\mathbf{A})+mc\xi)$: one can compute (i) $\mathbf{Y}$ in $O(mc^2)$; (ii) $\mathbf{\Xi}$ in $O(\mathrm{nnz}(\mathbf{A})+mc\xi)$; and (iii) $\mathbf{\Delta}$ in $O(c\xi\min\{c, \xi\})$. The failure probability 0.01 from Lemma 50. ■

8.2 Detailed description of the algorithm

This algorithm is very similar to the algorithm in Section 7.2; we only replace the parts 1-(a), 2-(a), 3-(a), and 4-(b) with faster procedures.

Input:

1. $\mathbf{A} \in \mathbb{R}^{m\times n}$ partitioned column-wise $\mathbf{A} = \begin{pmatrix}\mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s\end{pmatrix}$; for $i = 1 : s$, $\mathbf{A}_i \in \mathbb{R}^{m\times w_i}$; $\sum_i w_i = n$.
2. rank parameter $k <$ rank$(\mathbf{A})$
3. accuracy parameter $\varepsilon > 0$
4. failure probability δ

Algorithm

1. **Local Column Sampling**
   (a) For each $\mathbf{A}_i \in \mathbb{R}^{m\times w_i}$, compute $\mathbf{Z}_i \in \mathbb{R}^{w_i\times k}$ as $\mathbf{Z}_i = SparseSVDBoosting(\mathbf{A}_i, k, 1/3, \delta/s)$ (see Lemma 53). Also, construct the $m \times w_i$ matrix $\mathbf{E}_i = \mathbf{A}_i - \mathbf{A}_i\mathbf{Z}_i\mathbf{Z}_i^{\mathrm{T}}$. For each $\mathbf{A}_i$, compute $\mathbf{C}_i \in \mathbb{R}^{m\times\ell}$ containing $\ell = 4k$ columns from $\mathbf{A}_i$ as follows: $\mathbf{C}_i = \mathbf{A}_i\mathbf{S}_i$. Here, $\mathbf{S}_i$ has dimensions $w_i \times \ell$ and is constructed as follows: $\mathbf{S}_i = BssSamplingSparse(\mathbf{Z}_i, \mathbf{E}_i, \ell, \frac{1}{2}, \delta/s)$ (see Lemma 54).
   (b) Machine i sends $\mathbf{C}_i$ to the server.
2. **Global Column Sampling**

(a) Server constructs $m \times (s \cdot \ell)$ matrix $\mathbf{G}$ containing $(s \cdot \ell)$ actual columns from $\mathbf{A}$ as follows: $\mathbf{G} = \begin{pmatrix} \mathbf{C}_1 & \mathbf{C}_2 & \dots & \mathbf{C}_s \end{pmatrix}$. Then, server constructs $\mathbf{C} \in \mathbb{R}^{m \times c_1}$ via choosing $c_1 = 4k$ columns from $\mathbf{G}$ as follows: $\mathbf{C} = DeterministicCssFrobeniusSparse(\mathbf{G}, k, c_1)$ (see Lemma 55).

(b) Server sends $\mathbf{C}$ to all the machines.

3. **Adaptive Column Sampling**

(a) Server initializes the random seed and communicates it to all the machines. Each machine constructs the same $\mathbf{S} \in \mathbb{R}^{r \times m}$, with $r = \frac{4+2}{(1/2)^2-(1/2)^3} \log n$, each element of which is a random variable which takes values $\pm 1/\sqrt{r}$ with equal probability. Machine i finds $\tilde{\mathbf{\Psi}}_i = \mathbf{S}\mathbf{A}_i - \mathbf{S}\mathbf{C}\mathbf{C}^\dagger\mathbf{A}_i \in \mathbb{R}^{m \times w_i}$ and then computes β_i as it was described in Lemma 44. Machine i sends β_i to server.

(b) Server computes probability distribution $g_i = \frac{\beta_i}{\sum_i \beta_i}$. Server samples i.i.d. with replacement $\lceil 50k/\varepsilon \rceil$ samples (machines) from g_i. Then, server determines numbers t_i $(i = 1, 2, \dots, s)$, where t_i is the number of times the ith machine was sampled. It sends the t_i's to the machines.

(c) Machine i computes probabilities $q_j^i = \|\mathbf{x}\|_2^2 / \|\tilde{\mathbf{\Psi}}_i\|_{\mathrm{F}}^2$ $(j = 1 : w_i)$, where $\mathbf{x}$ is the jth column of $\tilde{\mathbf{\Psi}}_i$. And now machine i samples t_i samples from it's local probability distribution and sends the corresponding columns to the server. Let $c_2 = \sum_i t_i = \lceil 50k/\varepsilon \rceil$.

(d) Server collects the columns and assigns them to $\hat{\mathbf{C}} \in \mathbb{R}^{m \times c_2}$. Server constructs $\tilde{\mathbf{C}}$ to be the $m \times (c_1 + c_2)$ matrix: $\tilde{\mathbf{C}} = \begin{pmatrix} \mathbf{C}; & \hat{\mathbf{C}} \end{pmatrix}$. Let $c = c_1 + c_2 = 4k + \lceil 50k/\varepsilon \rceil$.

4. **Rank-k matrix in the span of $\tilde{\mathbf{C}}$**

(a) Server sends $\tilde{\mathbf{C}}$ to all the machines and each machine computes (the same) qr factorization of $\tilde{\mathbf{C}}$: $\tilde{\mathbf{C}} = \mathbf{Y}\mathbf{R}$ where $\mathbf{Y} \in \mathbb{R}^{m \times c}$ has orthonormal columns and $\mathbf{R} \in \mathbb{R}^{c \times c}$ is upper triangular.

(b) Server initializes the random seed and sends this to each machine, such that each machine generates the same matrix $\mathbf{\Psi} \in \mathbb{R}^{\xi \times n}$, for $\xi = O(c^2/\varepsilon^2)$, (see Definition 49). Machine i generates $\mathbf{D}_i \in \mathbb{R}^{n \times w_i}$ such that implicitly $\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & \mathbf{D}_2 & \dots & \mathbf{D}_s \end{pmatrix}$ is an $n \times n$ matrix each element of which is ± 1, with probability $1/2$. Each machine constructs implicitly $\mathbf{W}_i = \mathbf{\Psi} \cdot \mathbf{D}_i \in \mathbb{R}^{\xi \times w_i}$. *Implicitly* all machines together generate $\mathbf{W} = \begin{pmatrix} \mathbf{W}_1 & \mathbf{W}_2 & \dots & \mathbf{W}_s \end{pmatrix}$, with $\mathbf{W} \in \mathbb{R}^{\xi \times n}$ and $\mathbf{W}$ is a subspace embedding as in Definition 49. Machine i computes $\mathbf{H}_i = \tilde{\mathbf{C}}^{\mathrm{T}}(\mathbf{A}_i\mathbf{W}_i^{\mathrm{T}}) \in \mathbb{R}^{c \times \xi}$. Machine i sends $\mathbf{H}_i$ to the server.

(c) Server computes $\mathbf{\Xi} = \sum_{i=1}^s \mathbf{H}_i \in \mathbb{R}^{c \times \xi}$ and sends this back to all the machines. Now machines compute $\mathbf{\Xi} := \mathbf{R}^{-1} \cdot \mathbf{\Xi} (:= \mathbf{Y}^{\mathrm{T}}\mathbf{A}_i\mathbf{W}_i^{\mathrm{T}})$, and then they compute $\mathbf{\Delta} \in \mathbb{R}^{c \times k}$ to be the top k left singular vectors of $\mathbf{\Xi}$. Each machine computes $\mathbf{U} = \mathbf{Y} \cdot \mathbf{\Delta} \in \mathbb{R}^{m \times k}$.

Discussion. A few remarks are necessary for the last two stages of the algorithm. The third stage (adaptive column sampling), implements the adaptive sampling method of Lemma 56. To see this note that each column in $\mathbf{A}$ is sampled with probability

$$q_j^i \cdot g_i \geq \frac{1}{2} \cdot ||\mathbf{x}||_2^2 / \|\tilde{\mathbf{\Psi}}\|_{\mathrm{F}}^2,$$

where $\mathbf{x}$ is this column in $\tilde{\mathbf{\Psi}}$. This follows from Lemma 43. Overall, this stage constructs $\hat{\mathbf{C}}$ such that $\hat{\mathbf{C}} = AdaptiveColsSparse(\mathbf{A}, \mathbf{C}, c_2, 1/2)$.

The last stage in the algorithm implements the algorithm in Lemma 57. To see this, note that $\mathbf{W}$ satisfies the properties in the lemma. Hence,

$$[\mathbf{Y}, \mathbf{\Delta}] = ApproxSubspaceSVDSparse(\mathbf{A}, \tilde{\mathbf{C}}, k, \varepsilon).$$

8.3 Main result

The theorem below analyzes the approximation error, the communication complexity, and the running time of the previous algorithm.

Theorem 58. *The matrix $\tilde{\mathbf{C}} \in \mathbb{R}^{m \times c}$ with $c = O(k/\varepsilon)$ columns of $\mathbf{A}$ satisfies w.p. $0.59 - \frac{s+1}{n} - 2\delta$:*

$$\|\mathbf{A} - \tilde{\mathbf{C}}\tilde{\mathbf{C}}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq (1 + O(\varepsilon)) \cdot \left(\sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}) \right). \tag{11}$$

The matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$ with k orthonormal columns satisfies w.p. $0.58 - \frac{s+1}{n} - 2\delta$:

$$\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1 + \varepsilon) \cdot \left(\sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}) \right). \tag{12}$$

Let each column of $\mathbf{A}$ has at most ϕ non-zero elements. Then, the communication cost of the algorithm is $O\left(sk\phi\varepsilon^{-1} + sk^3\varepsilon^{-5}\right)$ and the running time is

$$O\left(\mathrm{nnz}(\mathbf{A}) \cdot \log^2\left(\frac{ns}{\delta}\right) + (m+n)s \cdot poly(k, \varepsilon^{-1}, \log\left(\frac{ns}{\delta}\right)\right).$$

8.4 Proof of Theorem 58

Recall that $\mathbf{A}_i = \mathbf{A}_i\mathbf{Z}_i\mathbf{Z}_i^{\mathrm{T}} + \mathbf{E}_i \in \mathbb{R}^{m \times w_i}$, i.e., $\mathbf{E}_i = \mathbf{A}_i - \mathbf{A}_i\mathbf{Z}_i\mathbf{Z}_i^{\mathrm{T}} \in \mathbb{R}^{m \times w_i}$. Also define $m \times w_i$ matrix $\mathbf{D}_i = \mathbf{C}_i\left(\mathbf{Z}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}\mathbf{Z}_i^{\mathrm{T}}$, and $m \times w_i$ matrix $\hat{\mathbf{E}}_i = \mathbf{A}_i - \mathbf{D}_i$. Hence ($\mathbf{D}, \hat{\mathbf{E}} \in \mathbb{R}^{m \times n}$, $\mathbf{A} = \mathbf{D} + \hat{\mathbf{E}}$),

$$\begin{aligned}
\mathbf{A} &= \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_s \end{pmatrix} \\
&= \underbrace{\begin{pmatrix} \mathbf{C}_1\left(\mathbf{Z}_1^{\mathrm{T}}\mathbf{S}_1\right)^{\dagger}\mathbf{Z}_1^{\mathrm{T}} & \mathbf{C}_2\left(\mathbf{Z}_2^{\mathrm{T}}\mathbf{S}_2\right)^{\dagger}\mathbf{Z}_2^{\mathrm{T}} & \dots & \mathbf{C}_s\left(\mathbf{Z}_s^{\mathrm{T}}\mathbf{S}_s\right)^{\dagger}\mathbf{Z}_s^{\mathrm{T}} \end{pmatrix}}_{\mathbf{D}} + \underbrace{\begin{pmatrix} \hat{\mathbf{E}}_1 & \hat{\mathbf{E}}_2 & \dots & \hat{\mathbf{E}}_s \end{pmatrix}}_{\hat{\mathbf{E}}}
\end{aligned}$$

The matrix $\mathbf{D}_i$ has rank k and it is in the span of $\mathbf{C}_i$; hence, we expect that the corresponding error defined by the residual matrix $\hat{\mathbf{E}}_i$ will be "small"; this because $\mathbf{C}_i$ was constructed to exactly minimize this "local" column-based matrix reconstruction error.

8.4.1 Proof of Eqn. 11

From Lemma 56, we have that with probability at least $0.9 - \frac{1}{n}$:

$$\|\mathbf{A} - \tilde{\mathbf{C}}\tilde{\mathbf{C}}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}) + \frac{60\varepsilon}{50} \cdot \|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2.$$

We further manipulate $\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2$ as follows,

$$\begin{aligned}
\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 = \|(\mathbf{D} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}) + (\mathbf{I}_d - \mathbf{C}\mathbf{C}^{\dagger})\hat{\mathbf{E}}\|_{\mathrm{F}}^2 &\leq 2 \cdot \left(\|(\mathbf{D} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2 + \|(\mathbf{I}_d - \mathbf{C}\mathbf{C}^{\dagger})\hat{\mathbf{E}}\|_{\mathrm{F}}^2 \right) \\
&\leq 2 \cdot \left(\|(\mathbf{D} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2 + \|\hat{\mathbf{E}}\|_{\mathrm{F}}^2 \right)
\end{aligned}$$

The last inequality holds because $\mathbf{I}_d - \mathbf{C}\mathbf{C}^\dagger$ is a projection matrix. Combining the two bounds, we have that with probability at least $0.9 - \frac{1}{n}$:

$$\|\mathbf{A} - \tilde{\mathbf{C}}\tilde{\mathbf{C}}^\dagger \mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A}) + \frac{60\varepsilon}{25} \cdot \left(\|\mathbf{D} - \mathbf{C}\mathbf{C}^\dagger \mathbf{D}\|_{\mathrm{F}}^2 + \|\hat{\mathbf{E}}\|_{\mathrm{F}}^2\right). \quad (13)$$

Lemma 59. *With probability at least* $1 - \frac{s}{n} - 2\delta$: $\|\hat{\mathbf{E}}\|_{\mathrm{F}}^2 \leq 10 \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A})$

Proof.

$$\begin{aligned}
\|\hat{\mathbf{E}}\|_{\mathrm{F}}^2 &= \sum_{i=1}^{s} \|\mathbf{A}_i - \mathbf{D}_i\|_{\mathrm{F}}^2 \\
&= \sum_{i=1}^{s} \|\mathbf{A}_i \mathbf{Z}_i \mathbf{Z}_i^{\mathrm{T}} + \mathbf{E}_i - \mathbf{C}_i \left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger \mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \\
&= \sum_{i=1}^{s} \|\mathbf{A}_i \mathbf{Z}_i \mathbf{Z}_i^{\mathrm{T}} - \underbrace{\left(\mathbf{A}_i \mathbf{Z}_i \mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i + \mathbf{E}_i \mathbf{S}_i\right)}_{\mathbf{C}_i} \left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger \mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 + \sum_{i=1}^{s} \|\mathbf{E}_i\|_{\mathrm{F}}^2 \\
&= \sum_{i=1}^{s} \|\mathbf{A}_i \mathbf{Z}_i \mathbf{Z}_i^{\mathrm{T}} - \mathbf{A}_i \mathbf{Z}_i \underbrace{\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i \left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger}_{\mathbf{I}_k} \mathbf{Z}_i^{\mathrm{T}} + \mathbf{E}_i \mathbf{S}_i \left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger \mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 + \sum_{i=1}^{s} \|\mathbf{E}_i\|_{\mathrm{F}}^2 \\
&= \sum_{i=1}^{s} \|\mathbf{E}_i \mathbf{S}_i \left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger \mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 + \sum_{i=1}^{s} \|\mathbf{E}_i\|_{\mathrm{F}}^2 \\
&= \sum_{i=1}^{s} \|\mathbf{E}_i \mathbf{S}_i \left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger\|_{\mathrm{F}}^2 + \sum_{i=1}^{s} \|\mathbf{E}_i\|_{\mathrm{F}}^2 \\
&\leq \sum_{i=1}^{s} \|\mathbf{E}_i \mathbf{S}_i\|_{\mathrm{F}}^2 \cdot \|\left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger\|_2^2 + \sum_{i=1}^{s} \|\mathbf{E}_i\|_{\mathrm{F}}^2 \\
&\leq \sum_{i=1}^{s} 4 \cdot \|\mathbf{E}_i\|_{\mathrm{F}}^2 + \sum_{i=1}^{s} \|\mathbf{E}_i\|_{\mathrm{F}}^2 \\
&= 5 \cdot \sum_{i=1}^{s} \|\mathbf{E}_i\|_{\mathrm{F}}^2 \\
&\leq 20 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A})
\end{aligned}$$

The third equality follows from the matrix pythagoras theorem. Next, we explain the equality $\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i \left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger = \mathbf{I}_k$. Recall that $\mathbf{Z}_i$ is a $w_i \times k$ matrix with $w_i > k$. Also, the construction of $\mathbf{S}_i \in \mathbb{R}^{w_i \times \ell}$ with $\ell > k$ and Lemma 54 imply that with probability at least $1 - \delta/s$: $\mathbf{Z}_i^{\mathrm{T}} \mathbf{S} := \mathbf{G}_i$ is an $k \times \ell$ matrix with $\mathrm{rank}(\mathbf{G}_i) = k$. Now write the SVD of $\mathbf{G}_i$ as $\mathbf{G}_i = \mathbf{U}_{\mathbf{G}_i} \mathbf{\Sigma}_{\mathbf{G}_i} \mathbf{V}_{\mathbf{G}_i}^{\mathrm{T}}$ with $\mathbf{U}_{\mathbf{G}_i} \in \mathbb{R}^{k \times k}$, $\mathbf{\Sigma}_{\mathbf{G}_i} \in \mathbb{R}^{k \times k}$ and $\mathbf{V}_{\mathbf{G}_i} \in \mathbb{R}^{w_i \times k}$. Also, ${\mathbf{G}_i}^\dagger = \mathbf{V}_{\mathbf{G}_i} \mathbf{\Sigma}_{\mathbf{G}_i}^{-1} \mathbf{U}_{\mathbf{G}_i}$. Finally,

$$\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i \left(\mathbf{Z}_i^{\mathrm{T}} \mathbf{S}_i\right)^\dagger = \mathbf{U}_{\mathbf{G}_i} \mathbf{\Sigma}_{\mathbf{G}_i} \mathbf{V}_{\mathbf{G}_i}^{\mathrm{T}} \mathbf{V}_{\mathbf{G}_i} \mathbf{\Sigma}_{\mathbf{G}_i}^{-1} \mathbf{U}_{\mathbf{G}_i} = \mathbf{U}_{\mathbf{G}_i} \mathbf{U}_{\mathbf{G}_i}^{\mathrm{T}} = \mathbf{I}_k.$$

The second inequality follows from Lemma 54 (we apply this lemma to each $\mathbf{Z}_i$ and notice that we chose $\ell = 4k$, hence $\|\left(\mathbf{Z}_i^{\mathrm{T}}\mathbf{S}_i\right)^{\dagger}\|_2^2 = 1/\sigma_k^2\left(\mathbf{Z}_i^{\mathrm{T}}\mathbf{S}_i\right) \leq (1-\sqrt{k/\ell})^{-2} = 4$).

The last inequality follows because

$$\sum_{i=1}^{s}\|\mathbf{E}_i\|_{\mathrm{F}}^2 = \sum_{i=1}^{s}\|\mathbf{A}_i - \mathbf{A}_i\mathbf{Z}_i\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq 4\sum_{i=1}^{s}\|\mathbf{A}_i - \mathbf{A}_i\mathbf{V}_i\mathbf{V}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \leq 4\sum_{i=1}^{s}\|\mathbf{A}_i - \mathbf{A}_i\mathbf{V}_{opt}\mathbf{V}_{opt}^{\mathrm{T}}\|_{\mathrm{F}}^2 = 4\sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A}).$$

Here, $\mathbf{V}_{opt} \in \mathbb{R}^{n\times k}$ contains the top k right singular vectors of $\mathbf{A}$. The first inequality follows from Lemma 51 (there is a failure probability $\delta + \frac{s}{n}$ at this step). The second inequality follows from the "optimality" of $\mathbf{V}_i$ for each submatrix $\mathbf{A}_i$. The failure probability follows from a union bound. ■

Lemma 60. *With probability at least* $0.69 - \delta$*:* $\|\mathbf{D} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2 \leq 57840 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A})$

Proof.

$$\begin{aligned}
\|\mathbf{D} - \mathbf{D}\mathbf{R}\mathbf{R}^{\dagger}\|_{\mathrm{F}}^2 &= \sum_{i=1}^{s}\|\mathbf{C}_i\Big(\mathbf{Z}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger}\mathbf{Z}_i^{\mathrm{T}} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{C}_i\Big(\mathbf{Z}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger}\mathbf{Z}_i^{\mathrm{T}}\|_{\mathrm{F}}^2 \\
&= \sum_{i=1}^{s}\|\mathbf{C}_i\Big(\mathbf{Z}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{C}_i\Big(\mathbf{Z}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger}\|_{\mathrm{F}}^2 \\
&\leq \sum_{i=1}^{s}\|\mathbf{C}_i - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{C}_i\|_{\mathrm{F}}^2 \cdot \|\Big(\mathbf{Z}_i^{\mathrm{T}}\mathbf{S}_i\Big)^{\dagger}\|_2^2 \\
&\leq \sum_{i=1}^{s}4 \cdot \|\mathbf{C}_i - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{C}_i\|_{\mathrm{F}}^2 \\
&= 4 \cdot \|\mathbf{G} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{G}\|_{\mathrm{F}}^2 \\
&\leq 4 \cdot 4820 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{G})}\sigma_i^2(\mathbf{G}) \\
&\leq 19280 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{G})}\sigma_i^2(\mathbf{A}) \\
&\leq 19280 \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A})
\end{aligned}$$

The second inequality follows from Lemma 54 and the third inequality follows from Theorem 55. The failure probability in the lemma follows via a simple union bound over the failure probabilities of those two lemmas. ■

Concluding the proof. Replacing the bounds for $\|\mathbf{D} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{D}\|_{\mathrm{F}}^2$ and $\|\hat{\mathbf{E}}\|_{\mathrm{F}}^2$ in Eqn. 9 we obtain that with probability $0.59 - \frac{s+1}{n} - 2\delta$ [7]:

$$\|\mathbf{A} - \tilde{\mathbf{C}}\tilde{\mathbf{C}}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \Pi_{\tilde{\mathbf{C}},k}^{\mathrm{F}}(\mathbf{A})\|_{\mathrm{F}}^2 \leq \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A}) + \frac{60}{25}57850\varepsilon \cdot \sum_{i=k+1}^{\mathrm{rank}(\mathbf{A})}\sigma_i^2(\mathbf{A}). \qquad (14)$$

[7] This probability follows by a union bound over the randomness of the whole algorithm; also, notice that the failure probabilities in the previous two lemmas both depend on the construction of the $\mathbf{S}_i$'s, hence we count those failure probabilities once when it comes to the failure probability of the whole algorithm.

8.4.2 Proof of Eqn. 12

We would like to apply Lemma 57 for the matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$ in the algorithm. Note that we already proved that the matrix $\tilde{\mathbf{C}}$ in the algorithm satisfies with probability $0.59 - \frac{s+1}{n} - 2\delta$:

$$\|\mathbf{A} - \Pi^{\mathrm{F}}_{\tilde{\mathbf{C}},k}(\mathbf{A})\|_{\mathrm{F}}^2 \leq (1 + O(\varepsilon))\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

Lemma 57 and a simple union bound conclude the proof.

8.4.3 Running time

1. **Local Column Sampling**: $O\left(\mathrm{nnz}(\mathbf{A}) \cdot \log^2(\frac{ns}{\delta}) + (m+n) \cdot s \cdot poly(k, \varepsilon^{-1}, \log(\frac{ns}{\delta})\right)$.
   (a) $O\left(\mathrm{nnz}(\mathbf{A}_i) \log^2(\frac{ns}{\delta})\right) + w_i \cdot \mathrm{poly}(k, \varepsilon, \log(\frac{ns}{\delta}))$ for each $\mathbf{Z}_i$ - from Lemma 53; and another $O\left(\log\left(\frac{s}{\delta}\right) \cdot \left(\mathrm{nnz}(\mathbf{A}_i) + m \cdot \mathrm{poly}(k, \varepsilon^{-1}, \log(\frac{s}{\delta}))\right)\right)$ for each $\mathbf{S}_i$ - from Lemma 54.
   (b) -
2. **Global Column Sampling**: $O\left(\mathrm{nnz}(\mathbf{A}) \cdot \log k + m \cdot \mathrm{poly}(k, \varepsilon^{-1})\right)$.
   (a) $O\left(\mathrm{nnz}(\mathbf{A}) \cdot \log k + m \cdot poly(k, \varepsilon^{-1})\right)$ to construct $\mathbf{C}$ - from Lemma 55.
   (b) -
3. **Adaptive Sampling**: $O(\mathrm{nnz}(\mathbf{A}) \cdot \log n + m \cdot s \cdot \mathrm{poly}(k, \varepsilon^{-1}, \log n))$.
   (a) First, we analyze the cost to compute the matrix $\tilde{\mathbf{\Psi}}_i$. $O(k^2 m)$ for each $\mathbf{C}^\dagger$ locally; then, we compute $(\mathbf{SA}_i) - (((\mathbf{SC}_i)\mathbf{C}_i^\dagger)\mathbf{A}_i)$. The costs are: $\mathbf{SA}_i := \mathbf{D}$ takes $O(\mathrm{nnz}(\mathbf{A}_i))$ time, $\mathbf{SC}_i := \mathbf{G}$ takes $O(\mathrm{nnz}(\mathbf{A}_i)\log(n))$ time, $\mathbf{GC}_i^\dagger := \mathbf{H}$ takes $O(m \log(n) k/\varepsilon^2)$ time, $\mathbf{HA} := \mathbf{L}$ takes $O(\mathrm{nnz}(\mathbf{A}_i)\log(n))$ time, and $\mathbf{D} - \mathbf{L}$ takes $O(n \log n)$ time. So, the total time to compute one $\tilde{\mathbf{\Psi}}_i$ is $O(\mathrm{nnz}(\mathbf{A}_i) \log n + m \cdot poly(k, \varepsilon^{-1}, \log n))$.
   Also, there is a cost of computing ψ_i, which is $O(\log n \cdot w_i)$ arithmetic operations.
   (b) $O(s)$ in total.
   (c) $O(k/\varepsilon)$ in total using the method of [46] locally.
   (d) $O(k/\varepsilon)$.
4. **Rank-k matrix in** $span(\tilde{\mathbf{C}})$: $O(\mathrm{nnz}(\mathbf{A}) + m \cdot s \cdot poly(k, \varepsilon^{-1}))$ arithmetic operations in total.
   (a) $O(smk^2/\varepsilon^2)$ in total.
   (b) $O(sn)$ in total to generate s times the same $\mathbf{\Psi}$. Then another $O(n)$ in total to generate the $\mathbf{D}_i$'s. For all $\mathbf{A}_i\mathbf{W}_i^{\mathrm{T}}$ we need $O(\mathrm{nnz}(\mathbf{A}))$ operations and then another $O(mk^3\varepsilon^{-7})$ for each $\mathbf{Y}^{\mathrm{T}} \cdot (\mathbf{A}_i\mathbf{W}_i^{\mathrm{T}})$
   (c) $O(sk^3\varepsilon^{-5})$ to find $\mathbf{\Xi}$; then another $O(sk^3\varepsilon^{-3} + sk^4\varepsilon^{-6}))$ to update $\mathbf{\Xi}$ locally in each machine and another $O(sk^4\varepsilon^{-6})$ to find $\mathbf{\Delta}$; and $O(smk^2/\varepsilon)$ to find $\mathbf{U}$ (s times).

8.4.4 Communication Complexity

Assume that we can represent each element in the input matrix $\mathbf{A}$ with at most b bits. Also, we assume that one word has length b bits and $b = O(\log(mns/\varepsilon))$.

1. **Local Column Sampling**: $O(sk\phi)$ words.

(a) -
(b) $O(sk\phi)$ elements of $\mathbf{A}$.

2. **Global Column Sampling**:$O(sk\phi)$ words.
   (a) -
   (b) $O(sk\phi)$ elements of $\mathbf{A}$.
3. **Adaptive Sampling**: $O(s + \phi k/\varepsilon)$ words.
   (a) s integers each of which is representable with $O(\log k + \log\log(mns/\varepsilon))$ (from Lemma 44).
   (b) s integers (the t_i's) each with magnitude at most n, hence representable with b bits.
   (c) $O(\phi k/\varepsilon)$ elements of $\mathbf{A}$.
   (d) -
4. **Best rank-k matrix in the span of $\tilde{\mathbf{C}}$**: $O(s\phi k/\varepsilon + sk^3\varepsilon^{-5})$ words.
   (a) $O(sk\phi\varepsilon^{-1})$ elements of $\mathbf{A}$.
   (b) $O(sk^3\varepsilon^{-5})$ numbers each of which can be represented with b bits.
   (c) $O(sk^3\varepsilon^{-5})$ numbers each of which can be represented with b bits.

In total the communication complexity is $O(sk\phi/\varepsilon + sk^3/\varepsilon^5)$ words.

9 Lower bounds

9.1 Lower bounds for distributed PCA on dense matrices

This section provides a communication cost lower bound for the Distributed PCA problem of Definition 2. Specifically, we describe the construction of an $m \times n$ matrix $\mathbf{A}$ (hard instance), and formally argue that for this $\mathbf{A}$, any $k \leq 0.99m$, and any error parameter C with $1 < C < poly(skm)$, if there exists some algorithm to construct an $m \times k$ matrix $\mathbf{U}$ such that, with constant probability, $\|\mathbf{A} - \mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq C \cdot \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$, then this algorithm has communication cost $\Omega(skm)$ words.

9.1.1 Preliminaries

We use the notation $\mathbf{G}_{k,m}$ to denote the set of k-dimensional subspaces of $\mathbb{R}^m$, which we identify with corresponding projector matrices $\mathbf{Q} \in \mathbb{R}^{m\times m}$, with $\mathrm{rank}(\mathbf{Q}) = k \leq m$, onto the subspaces, i.e.,

$$\mathbf{G}_{k,m} = \{\mathbf{Q} \in \mathbb{R}^{m\times m} : \mathbf{Q}^2 = \mathbf{Q}, \mathrm{rank}(\mathbf{Q}) = k \leq m.\}$$

We also need a description of a subset from $\mathbf{G}_{k,m}$:

$$C_\delta^k(\mathbf{P}) = \{\mathbf{Q} \in \mathbf{G}_{k,m} : \|\mathbf{P} - \mathbf{Q}\|_2 \leq \delta\}.$$

This is the set of those projectors $\mathbf{Q} \in \mathbf{G}_{k,m}$ which differ in operator norm by at most δ from another fixed projector $\mathbf{P} \in \mathbf{G}_{k,m}$. In our analysis below we also need a result from [35]:

Theorem 61. *(Net Bound - Corollary 5.1 of [35]) For any $m, k, \delta > 0$, there is a family $\mathcal{N} = \{\mathbf{P}^1, \ldots, \mathbf{P}^N\}$, $N = 2^{\Omega(k(m-k)\log(1/\delta))}$, where $\mathbf{P}^i \in \mathbf{G}_{k,m}$ and $C_\delta^k(\mathbf{P}^i) \cap C_\delta^k(\mathbf{P}^j) = \emptyset$, for all $i \neq j$.*

Theorem 61 proves the existence of a large, but finite, set $\mathcal{N}$ of projection matrices, such that if one considers the ball of matrices of operator norm at most δ centered at each matrix in $\mathcal{N}$, then these balls are disjoint. Our hard instance matrix $\mathbf{A}$ is constructed from some member in $\mathcal{N}$.

9.1.2 Hard instance construction

Recall that in the column-partition model of Definition 1 there are s servers holding matrices $\mathbf{A}_1, \ldots, \mathbf{A}_s$, respectively, where $\mathbf{A}_i$ has m rows and some subset of w_i columns of some $m \times n$ matrix $\mathbf{A}$. Notice that $\sum w_i = n$. First of all, for arbitrary m, s we assume[8] that $n \geq sm$. Below, we describe a specific construction for a matrix $\mathbf{A}$.

First of all, we set $\delta = 1/(skm)$, the parameter to be used in Theorem 61. Next, fix a family $\mathcal{N}$ of subspaces of $\mathbf{G}_{k,m}$ with the guarantees of Theorem 61. Let $\mathbf{P_R} = \mathbf{R}\mathbf{R}^{\mathrm{T}}$ be a uniformly random member of $\mathcal{N}$, where $\mathbf{R} \in \mathbb{R}^{m \times k}$ has orthonormal columns (any projector can be expressed as such a matrix, where the columns of $\mathbf{R}$ span the subspace that $\mathbf{P}$ projects onto). The entries of $\mathbf{A}_1$ are equal to the entries of $\mathbf{R}$, each rounded to the nearest integer multiple of $1/B$, where $B = \text{poly}(skm)$ is a large enough parameter specified later. We denote this rounded matrix with $\tilde{\mathbf{R}} \in \mathbb{R}^{m \times k}$. So, if α_{ij} is the (i,j)th entry in $\mathbf{A}_1$, then $\alpha_{ij} = \frac{k_{min}}{B}$, with $k_{min} = \operatorname{argmin}_{k \in \mathbb{Z}} |R_{ij} - \frac{k}{B}|$. Each $\mathbf{A}_i$ for $i = 2, 3, \ldots, s-1$ is $1/B$ times the $m \times m$ identity matrix. Finally, $\mathbf{A}_s$ is the $m \times t$ matrix of all zeros with $t = n - (s-1)m - k$. I.e.,

$$\mathbf{A} = \begin{pmatrix} \tilde{\mathbf{R}} & \frac{1}{B}\mathbf{I}_m & \frac{1}{B}\mathbf{I}_m & \ldots & \frac{1}{B}\mathbf{I}_m & \mathbf{0}_{m \times t} \end{pmatrix}.$$

9.1.3 Intermediate results

First, we give a bound regarding the best rank k approximation for the above matrix $\mathbf{A}$.

Lemma 62. *For the matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ in Section 9.1.2, and any $k \leq 0.99m$: $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 < \frac{sm}{B^2}$.*

Proof. We have,

$$\begin{aligned} \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 &\leq \|\mathbf{A} - \mathbf{A}_1\mathbf{A}_1^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \\ &= \sum_{i>1} \|(\mathbf{I} - \mathbf{A}_1\mathbf{A}_1^{\dagger})\mathbf{A}_i\|_{\mathrm{F}}^2 \\ &\leq \sum_{i>1} \|\mathbf{A}_i\|_{\mathrm{F}}^2 \\ &= (s-2) \cdot \frac{m}{B^2} \\ &< \frac{sm}{B^2}, \end{aligned}$$

where the first inequality uses the fact that $\mathbf{A}_1\mathbf{A}_1^{\dagger}$ is a matrix of rank at most k, the first equality uses the fact that $(\mathbf{I} - \mathbf{A}_1\mathbf{A}_1^{\dagger})\mathbf{A}_i$ is the all-zeros matrix, the second inequality uses that a projector cannot increase a unitarily invariant norm, and the third inequality follows by construction. ■

Next, we bound, in the operator norm, the difference of $\mathbf{A}_1$ from the matrix $\mathbf{R}$. The lemma follows from the fact that $\mathbf{A}_1 \in \mathbb{R}^{m \times k}$ is obtained by $\mathbf{R} \in \mathbb{R}^{m \times k}$ by rounding each entry to the nearest integer multiple of $1/B$, and then summing the squared differences across all km entries.

Lemma 63. *(Precision Lemma)* $\|\mathbf{A}_1 - \mathbf{R}\|_2 \leq \frac{\sqrt{km}}{B}$.

Proof. $\|\mathbf{A}_1 - \mathbf{R}\|_2^2 \leq \|\mathbf{A}_1 - \mathbf{R}\|_{\mathrm{F}}^2 \leq \frac{km}{B^2}$. ■

[8] As, otherwise we can choose a value $s' < s$ so that $n - m \leq s'm \leq n$ and apply the argument of Theorem 66 with s replaced with s'. Then, the lower bound in Theorem 66 will then be $\Omega(s'mk)$, which assuming $n \geq 2d$, is an $\Omega(n)$ communication lower bound

Next, we prove a pure linear algebraic result. The following lemma captures the fact that if some $\mathbf{P} \in \mathbf{G}_{k,m}$ is close to $\mathbf{QP}$ (in Frobenius norm) for $\mathbf{Q} \in \mathbf{G}_{k,m}$, then $\mathbf{P}$ is also close to $\mathbf{Q}$ (in Frobenius norm). We will use this lemma later for $\mathbf{P} = \mathbf{P}_{\mathbf{R}}$.

Lemma 64. *(Error Measure) Let $\mathbf{P}, \mathbf{Q} \in \mathbf{G}_{k,m}$ with $\|\mathbf{P} - \mathbf{QP}\|_{\mathrm{F}}^2 \leq \Delta$. Then $\|\mathbf{P} - \mathbf{Q}\|_{\mathrm{F}}^2 \leq 2\Delta$.*

Proof. By the matrix Pythagorean theorem,

$$k = \|\mathbf{Q}\|_{\mathrm{F}}^2 = \|\mathbf{QP}\|_{\mathrm{F}}^2 + \|\mathbf{Q}(\mathbf{I} - \mathbf{P})\|_{\mathrm{F}}^2. \tag{15}$$

Also by the matrix Pythagorean theorem,

$$k = \|\mathbf{P}\|_{\mathrm{F}}^2 = \|\mathbf{QP}\|_{\mathrm{F}}^2 + \|(\mathbf{I} - \mathbf{Q})\mathbf{P}\|_{\mathrm{F}}^2,$$

and so using the premise of the lemma, $\|\mathbf{QP}\|_{\mathrm{F}}^2 \geq k - \Delta$. Combining with (15),

$$\|\mathbf{Q}(\mathbf{I} - \mathbf{P})\|_{\mathrm{F}}^2 \leq \Delta. \tag{16}$$

Hence,

$$\begin{aligned} \|\mathbf{P} - \mathbf{Q}\|_{\mathrm{F}}^2 &= \|(\mathbf{P} - \mathbf{Q})\mathbf{P}\|_{\mathrm{F}}^2 + \|(\mathbf{P} - \mathbf{Q})(\mathbf{I} - \mathbf{P})\|_{\mathrm{F}}^2 \\ &= \|\mathbf{P} - \mathbf{QP}\|_{\mathrm{F}}^2 + \|\mathbf{Q} - \mathbf{QP}\|_{\mathrm{F}}^2 \\ &\leq \Delta + \|\mathbf{Q}(\mathbf{I} - \mathbf{P})\|_{\mathrm{F}}^2 \\ &\leq 2\Delta, \end{aligned}$$

where the first equality follows by the matrix Pythagorean theorem, the second equality uses $\mathbf{P}^2 = \mathbf{P}$ (since $\mathbf{P}$ is a projector), the third equality uses the bound in the premise of the lemma, and the fourth inequality uses (16). ∎

9.1.4 Main Argument

Before presenting the main theorem, we give an intermediate technical lemma.

Lemma 65. *(Implication of Correctness) Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be the hard instance matrix in Section 9.1.2 and let $\mathbf{A}$ be distributed in s machines in the way it was described in Section 9.1.2:*

$$\mathbf{A} = \begin{pmatrix} \tilde{\mathbf{R}} & \frac{1}{B}\mathbf{I}_m & \frac{1}{B}\mathbf{I}_m & \dots & \frac{1}{B}\mathbf{I}_m & \mathbf{0}_{m \times t} \end{pmatrix}.$$

Suppose $n = \Omega(sm)$, $k < \mathrm{rank}(\mathbf{A})$, and $1 < C < \mathit{poly}(skm)$. $\mathbf{A}$ is revealed to the machines by just describing the entries in $\mathbf{A}$ and without specifying any other detail regarding the construction. Given this $\mathbf{A}$ and assuming further that each machine knows some projector matrix $\mathbf{Q} \in \mathbb{R}^{m \times m}$ with rank at most k such that $\|\mathbf{A} - \mathbf{QA}\|_{\mathrm{F}} \leq C\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}$, there is a deterministic algorithm (protocol) which, upon termination, leaves on each machine the matrix $\mathbf{R} \in \mathbb{R}^{m \times k}$ which was used to construct $\mathbf{A}_1$.

Proof. First, we prove that $\|\mathbf{RR}^{\mathrm{T}} - \mathbf{Q}\|_2 < \frac{1}{2skm}$. Then, using this bound, we describe a protocol that deterministically reveals the matrix $\mathbf{R}$ which was used to construct $\mathbf{A}_1$.

Using Lemma 62 and the premise in the lemma, it follows that: $\|\mathbf{A} - \mathbf{QA}\|_{\mathrm{F}}^2 \leq C\frac{sm}{B^2}$, which in particular implies that

$$\|\mathbf{A}_1 - \mathbf{QA}_1\|_{\mathrm{F}}^2 \leq C\frac{sm}{B^2}. \tag{17}$$

We further manipulate the term $\|\mathbf{A}_1 - \mathbf{Q}\mathbf{A}_1\|_{\mathrm{F}}$ as follows:

$$\begin{aligned}
\|\mathbf{A}_1 - \mathbf{Q}\mathbf{A}_1\|_{\mathrm{F}} &= \|(\mathbf{I} - \mathbf{Q})\mathbf{A}_1\|_{\mathrm{F}} \\
&= \|(\mathbf{I} - \mathbf{Q})\mathbf{R}\mathbf{R}^{\mathrm{T}} + (\mathbf{I} - \mathbf{Q})(\mathbf{A}_1 - \mathbf{R}\mathbf{R}^{\mathrm{T}})\|_{\mathrm{F}} \\
&\geq \|(\mathbf{I} - \mathbf{Q})\mathbf{R}\mathbf{R}^{\mathrm{T}}\|_{\mathrm{F}} - \|(\mathbf{I} - \mathbf{Q})(\mathbf{A}_1 - \mathbf{R}\mathbf{R}^{\mathrm{T}})\|_{\mathrm{F}} \\
&\geq \|(\mathbf{I} - \mathbf{Q})\mathbf{R}\mathbf{R}^{\mathrm{T}}\|_{\mathrm{F}} - \|\mathbf{I} - \mathbf{Q}\|_{\mathrm{F}}\|(\mathbf{A}_1 - \mathbf{R}\mathbf{R}^{\mathrm{T}})\|_2 \\
&\geq \|(\mathbf{I} - \mathbf{Q})\mathbf{R}\mathbf{R}^{\mathrm{T}}\|_{\mathrm{F}} - \frac{\sqrt{km}}{B}(\sqrt{m} + \sqrt{k}),
\end{aligned}$$

where the first inequality is the triangle inequality, the second inequality uses sub-multiplicativity, and the third inequality uses the triangle inequality, i.e., that $\|\mathbf{I}-\mathbf{Q}\|_{\mathrm{F}} \leq \|\mathbf{I}\|_{\mathrm{F}}+\|\mathbf{Q}\|_{\mathrm{F}}$, and Lemma 63. Combining this bound with (17), it follows that $\|(\mathbf{I}-\mathbf{Q})\mathbf{R}\mathbf{R}^{\mathrm{T}}\|_{\mathrm{F}} \leq \frac{\sqrt{Csm}}{B} + \frac{2m\sqrt{k}}{B}$. At this point we would like to apply Lemma 64 to conclude that $\|\mathbf{R}\mathbf{R}^{\mathrm{T}} - \mathbf{Q}\|_{\mathrm{F}}$ is small. To do so, we need $\mathbf{R}\mathbf{R}^{\mathrm{T}}$ and $\mathbf{Q}$ to be of rank k. While $\mathbf{R}$ has rank k by construction, $\mathbf{Q}$ may not. However, increasing the rank of $\mathbf{Q}$ cannot increase the error $\|\mathbf{A}-\mathbf{Q}\mathbf{A}\|_{\mathrm{F}}$. Hence, if $\mathrm{rank}(\mathbf{Q}) < k$, given $\mathbf{Q}$, the protocol in the premise of the lemma allows each server to locally add standard basis vectors to the column space of $\mathbf{Q}$ until the column space of $\mathbf{Q}$ becomes k. This involves no communication and each server ends up with the same new setting of $\mathbf{Q}$, which for simplicity we still denote with $\mathbf{Q}$. Therefore, $rank(\mathbf{Q}) = k$. Applying Lemma 64, it now follows that

$$\|\mathbf{R}\mathbf{R}^{\mathrm{T}} - \mathbf{Q}\|_2 \leq \|\mathbf{R}\mathbf{R}^{\mathrm{T}} - \mathbf{Q}\|_{\mathrm{F}} \leq \sqrt{2}\left(\frac{\sqrt{Csm}}{B} + \frac{2m\sqrt{k}}{B}\right).$$

By setting B to be a sufficiently large $\mathrm{poly}(skm)$, we have $\|\mathbf{R}\mathbf{R}^{\mathrm{T}} - \mathbf{Q}\|_2 < \frac{1}{2skm}$.

Therefore, given $\mathbf{Q}$, by Theorem 61 there is a unique matrix $\mathbf{P}^i$ in $\mathcal{N}$ for which $\mathbf{Q} \in C_\delta^k(\mathbf{P}^i)$, with $\delta = \frac{1}{skm}$. Indeed, since $\|\mathbf{R} - \mathbf{Q}\|_2 \leq \frac{1}{skm} < \delta/2$, there cannot be two matrices $\mathbf{P}^i$ and $\mathbf{P}^j$ for $i \neq j$ for which $\mathbf{Q} \in C_\delta^k(\mathbf{P}^i)$ and $\mathbf{Q} \in C_\delta^k(\mathbf{P}^j)$ since then by the triangle inequality $\|\mathbf{P}^i - \mathbf{P}^j\|_2 < \delta$, contradicting the construction of $\mathcal{N}$. Hence, it follows that each machine can deterministically identify this $\mathbf{P}^i$, by enumeration over $\mathcal{N}$, and therefore compute $\mathbf{R}$, via, for example, an SVD. ■

We are now fully equipped to present the main argument about the communication cost lower bound for distributed PCA.

Theorem 66. *(Main) Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be the hard instance matrix in Section 9.1.2 and let $\mathbf{A}$ be distributed in s machines in the way it was described in Section 9.1.2:*

$$\mathbf{A} = \begin{pmatrix} \tilde{\mathbf{R}} & \frac{1}{B}\mathbf{I}_m & \frac{1}{B}\mathbf{I}_m & \dots & \frac{1}{B}\mathbf{I}_m & \mathbf{0}_{m\times t} \end{pmatrix}.$$

Suppose $n = \Omega(sm)$, $k \leq .99m$, and $1 < C < \mathrm{poly}(skm)$. Assume that there is an algorithm (protocol) which succeeds with probability at least $2/3$ in having the i-th server output $\mathbf{Q}\mathbf{A}_i$, for all i, where $\mathbf{Q} \in \mathbb{R}^{m\times m}$ is a projector matrix with rank at most k such that $\|\mathbf{A} - \mathbf{Q}\mathbf{A}\|_{\mathrm{F}} \leq C\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}$. Then, this algorithm requires $\Omega(skm\log(skm))$ bits of communication.

Further, the bound holds even if the input matrices $\mathbf{A}_i$ to the servers have all entries which are integer multiples of $1/B$, for a value $B = \mathrm{poly}(skm)$, and bounded in magnitude by 1, and therefore all entries can be specified with $O(\log(skm))$ bits.

Note that assuming a word size of $\Theta(\log(skm))$ bits, we obtain an $\Omega(skm)$ word lower bound.

Proof. If the protocol succeeds, then, since each server outputs $\mathbf{Q}\mathbf{A}_i$, and $\mathbf{A}_i$ is $1/B$ times the identity matrix for all $i = 2, 3, ...s-1$, each of those servers can compute $\mathbf{Q}$. One of those servers

can now send this $\mathbf{Q}$ to the first and the last server. It follows by Lemma 65, that each server can identify $\mathbf{P_R}$ and $\mathbf{R}$.

Letting $\mathcal{E}$ be the event that the protocol succeeds, and letting Π^i be the ordered sequence of all incoming and outgoing messages at the i-th server, it follows that

$$I(\Pi^i; \mathbf{Q} \mid \mathcal{E}) = \Omega(\log |\mathcal{N}|) = \Omega(km \log(smk)),$$

where $I(X; Y \mid \mathcal{F}) = H(X \mid \mathcal{F}) - H(X \mid Y, \mathcal{F})$ is the mutual information between random variables X and Y conditioned on $\mathcal{F}$. To see the first equality in the above, observe that $I(\Pi^i; \mathbf{Q} \mid \mathcal{E}) = H(\mathbf{Q} \mid \mathcal{E}) - H(\mathbf{Q} \mid \Pi^i, \mathcal{E})$, and conditioned on $\mathcal{E}$, there is a uniquely determined matrix $\mathbf{P_R}$ each server identifies, which is uniform over a set of size $|\mathcal{N}|$, and so $H(\mathbf{Q} \mid \mathcal{E}) \geq H(\mathbf{P_R}) = \Omega(\log_2 |\mathcal{N}|)$.

Here, $H(X)$ is the Shannon entropy of a random variable X, given by $H(X) = \sum_x \Pr[X = x] \log(1/\Pr[X = x])$, and $H(X \mid Y) = \sum_y \Pr[Y = y] H(X \mid Y = y)$ is the conditional Shannon entropy.

Hence, if Z is an indicator random variable which is 1 if and only if $\mathcal{E}$ occurs, then using that $I(X; Y) \geq I(X; Y \mid W) - H(W)$ for any random variables X, Y, and W, and that $I(X; Y \mid W) = \sum_w \Pr[W = w] I(X; Y \mid W = w)$, we have

$$\begin{aligned} I(\Pi^i; \mathbf{Q}) &\geq I(\Pi^i; \mathbf{Q} \mid Z) - 1 \\ &\geq I(\Pi^i; \mathbf{Q} \mid Z = 1) \Pr[Z = 1] - 1 \\ &= \Omega(km \log(smk)), \end{aligned}$$

mapping X to Π^i, Y to $\mathbf{Q}$, and W to Z in the mutual information bound above. It follows that

$$\Omega(km \log(smk)) \leq I(\Pi^i; \mathbf{Q}) \leq H(\Pi^i) \leq \mathbf{E}_{\mathbf{P_R}, rand}[|\Pi^i|],$$

where $|\Pi^i|$ denotes the length, in bits, of the sequence Π^i, and $rand$ is the concatenation of the private randomness of all s players Note that the entropy of Π^i is a lower bound on the expected encoding length of $|\Pi^i|$, by the Shannon coding theorem. By linearity of expectation, $\sum_i \mathbf{E}_{R, rand} \|\Pi^i| = \Omega(skm \log(smk))$, which implies there exists an $\mathbf{R}$ and setting of $rand$ for which $\sum_i \|\Pi^i| = \Omega(skm \log(smk))$. It follows that the total communication is $\Omega(smk \log(smk))$ bits. ■

9.2 Lower bounds for distributed PCA on sparse matrices

Applying the previous theorem with $m = \phi$ gives a communication lower bound for sparse matrices.

Corollary 67. *Let $\mathbf{A} \in \mathbb{R}^{\phi \times n}$ be the hard instance matrix in Section 9.1.2 (with m replaced with ϕ in noting the number of rows in $\mathbf{A}$):*

$$\mathbf{A} = \begin{pmatrix} \tilde{\mathbf{R}} & \frac{1}{B}\mathbf{I}_\phi & \frac{1}{B}\mathbf{I}_\phi & \dots & \frac{1}{B}\mathbf{I}_\phi & \mathbf{0}_{\phi \times t} \end{pmatrix}.$$

Suppose $n = \Omega(s\phi)$, $k \leq .99\phi$, and $1 < C < poly(sk\phi)$. Now, for arbitrary m consider the matrix $\hat{\mathbf{A}} \in \mathbb{R}^{m \times n}$ which has $\mathbf{A}$ in the top part and the all-zeros $(m - \phi) \times n$ matrix in the bottom part. $\hat{\mathbf{A}}$ admits the same column partition as $\mathbf{A}$ after padding with zeros:

$$\hat{\mathbf{A}} = \begin{pmatrix} \tilde{\mathbf{R}} & \frac{1}{B}\mathbf{I}_\phi & \frac{1}{B}\mathbf{I}_\phi & \dots & \frac{1}{B}\mathbf{I}_\phi & \mathbf{0}_{\phi \times t} \\ \mathbf{0}_{(m-\phi) \times k} & \mathbf{0}_{(m-\phi) \times \phi} & \mathbf{0}_{(m-\phi) \times \phi} & \dots & \mathbf{0}_{(m-\phi) \times \phi} & \mathbf{0}_{(m-\phi) \times t} \end{pmatrix}.$$

We denote the new sub-matrices with $\hat{\mathbf{A}}_i \in \mathbb{R}^{m \times w_i}$.

Assume that there is an algorithm (protocol) which succeeds with probability at least $2/3$ in having the i-th server output $\hat{\mathbf{Q}}\hat{\mathbf{A}}_i$, for all i, where $\hat{\mathbf{Q}} \in \mathbb{R}^{m \times m}$ is a projector matrix with rank at most k such that $\|\hat{\mathbf{A}} - \hat{\mathbf{Q}}\hat{\mathbf{A}}\|_{\mathrm{F}} \leq C\|\hat{\mathbf{A}} - \hat{\mathbf{A}}_k\|_{\mathrm{F}}$. Then, this algorithm requires $\Omega(sk\phi \log(sk\phi))$ bits of communication.

Note that assuming a word size of $\Theta(\log(sk\phi))$ bits, we obtain an $\Omega(sk\phi)$ word lower bound.

9.3 Lower bounds for distributed column-based matrix reconstruction

In this section we develop a communication lower bound for the Distributed Column Subset Selection Problem of Definition 4.

9.3.1 A Net of discretized Deshpande-Vempala hard instances

We start by describing the construction of a special sparse matrix from the work of Desphande and Vempala [22], which was further studied by [15]. This matrix gives a lower bound (on the number of columns need to be selected in order to achieve a certain accuracy) for the standard column subset selection problem (see the following Fact). Suppose $\phi \geq 2k/\epsilon$. Consider a $(\phi+1) \times \phi$ matrix $\mathbf{B}$ for which the i-th column is $\mathbf{e}_1 + \mathbf{e}_{i+1}$, where $\mathbf{e}_i$ is the i-th standard basis vector in $\mathbb{R}^{\phi+1}$. Now define an $((\phi+1)k) \times (\phi k)$ matrix $\mathbf{A}$ with k blocks along the diagonal, where each block is equal to the matrix $\mathbf{B}$. Let $n = (\phi+1)k$.

Fact 68. *(Proposition 4 in [22], extended in Theorem 37 of [15]) Let $\mathbf{A}$ be as above. If $\mathbf{P}$ is an $n \times n$ projection operator onto any subset of at most $k/(2\epsilon)$ columns of $\mathbf{A}$, then $\|\mathbf{A} - \mathbf{PA}\|_{\mathrm{F}}^2 > (1+\epsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$.*

To prove a communication lower bound for distributed column subset selection, we will need to transform the above matrix $\mathbf{A}$ by multiplying by a rounded orthonormal matrix, as follows. We also need to prove a similar lower bound as in the previous fact for this transformed matrix.

Lemma 69. *Let $\mathbf{A}$, k, ϕ, $1/\varepsilon$, and n be as above. For any $n \times n$ orthonormal matrix $\mathbf{L}$, let $\tilde{\mathbf{L}}$ denote the matrix formed by rounding each of the entries of $\mathbf{L}$ to the nearest integer multiple of M, where $M \leq 1/((\phi+1)k)^c$, for a sufficiently large constant $c > 0$. If $\mathbf{P}$ is an $n \times n$ projection operator onto any subset of at most $k/(6\epsilon)$ columns of $\tilde{\mathbf{L}}\mathbf{A}$, then $\|\tilde{\mathbf{L}}\mathbf{A} - \mathbf{P}\tilde{\mathbf{L}}\mathbf{A}\|_{\mathrm{F}}^2 > (1+\epsilon)\|\tilde{\mathbf{L}}\mathbf{A} - [\tilde{\mathbf{L}}\mathbf{A}]_k\|_{\mathrm{F}}^2$.*

Proof. We will show that, assuming $1/\varepsilon < n/3$, if $\mathbf{P}$ is an $n \times n$ projection operator onto any subset of at most $k/(2\epsilon)$ columns of $\tilde{\mathbf{L}}\mathbf{A}$, then $\|\tilde{\mathbf{L}}\mathbf{A} - \mathbf{P}\tilde{\mathbf{L}}\mathbf{A}\|_{\mathrm{F}}^2 > (1+\epsilon/3)\|\tilde{\mathbf{L}}\mathbf{A} - [\tilde{\mathbf{L}}\mathbf{A}]_k\|_{\mathrm{F}}^2$. The lemma will then follow by replacing ε with 3ε.

Suppose S is a subset of $k/(2\epsilon)$ columns of $\tilde{\mathbf{L}}\mathbf{A}$ for which

$$\|\tilde{\mathbf{L}}\mathbf{A} - \mathbf{P}\tilde{\mathbf{L}}\mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\epsilon/3)\|\tilde{\mathbf{L}}\mathbf{A} - [\tilde{\mathbf{L}}\mathbf{A}]_k\|_{\mathrm{F}}^2, \tag{18}$$

where $\mathbf{P}$ is the projection onto the columns in S and $[\tilde{\mathbf{L}}\mathbf{A}]_k$ is the best rank-k approximation to $\tilde{\mathbf{L}}\mathbf{A}$ The proof strategy is to relate the left hand side of (18) to $\|\mathbf{A} - \mathbf{QA}\|_{\mathrm{F}}^2$ where $\mathbf{Q}$ is the projection onto the columns of $\mathbf{A}$ indexed by the same index set S, and simultaneously to relate the right hand side of (18) to $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$.

We start by looking at $\|\tilde{\mathbf{L}}\mathbf{A} - [\tilde{\mathbf{L}}\mathbf{A}]_k\|_{\mathrm{F}}$. Notice that $\tilde{\mathbf{L}}\mathbf{A} = \mathbf{L}\mathbf{A} + \mathbf{E}\mathbf{A}$, where $\mathbf{E}$ is a matrix whose entries are all bounded in magnitude by M. Therefore, $\tilde{\mathbf{L}}\mathbf{A} = \mathbf{L}\mathbf{A} + \mathbf{F}$, where $\mathbf{F} = \mathbf{E}\mathbf{A}$ and

$$\|\mathbf{F}\|_2 \leq \|\mathbf{E}\|_{\mathrm{F}}\|\mathbf{A}\|_{\mathrm{F}} \leq \left(n^2M^2\right)^{1/2} \cdot (2(n-1))^{1/2} \leq \left(2n^3M^2\right)^{1/2},$$

where we have used $\|\mathbf{A}\|_{\mathrm{F}} \leq (2(n-1))^{1/2}$. By Weyl's inequality (Corollary 7.3.8 of [30]), for all $i \leq \phi k$,

$$|\sigma_i(\mathbf{LA}) - \sigma_i(\mathbf{LA} + \mathbf{F})| \leq \|\mathbf{F}\|_2 \leq \left(2n^3M^2\right)^{1/2},$$

where $\sigma_i(\mathbf{X})$ denotes the i-th singular value of a matrix $\mathbf{X}$. Since $\mathbf{L}$ is orthonormal, $\sigma_i(\mathbf{LA}) = \sigma_i(\mathbf{A})$ for all i. It follows that

$$\begin{aligned}
\|\tilde{\mathbf{L}}\mathbf{A} - [\tilde{\mathbf{L}}\mathbf{A}]_k\|_{\mathrm{F}}^2 &= \sum_{i>k} \sigma_i^2(\tilde{\mathbf{L}}\mathbf{A}) \\
&= \sum_{i>k} \left(\sigma_i(\mathbf{LA}) \pm \left(2n^3M^2\right)^{1/2}\right)^2 \\
&= \sum_{i>k} \left(\sigma_i(\mathbf{A}) \pm \left(2n^3M^2\right)^{1/2}\right)^2 \\
&= \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 \pm O(n\|\mathbf{A}\|_2 \left(n^3M^2\right)^{1/2} + n^4M^2) \\
&= \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 \pm O(1/n^2), \qquad (19)
\end{aligned}$$

where the final line follows for a sufficiently small choice of $M \leq 1/n^c$, using that $\|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_{\mathrm{F}} = O(\sqrt{n})$.

We now look at $\|\tilde{\mathbf{L}}\mathbf{A} - \mathbf{P}\tilde{\mathbf{L}}\mathbf{A}\|_{\mathrm{F}}$. By the triangle inequality,

$$\begin{aligned}
\|\tilde{\mathbf{L}}\mathbf{A} - \mathbf{P}\tilde{\mathbf{L}}\mathbf{A}\|_{\mathrm{F}} &\geq \|\mathbf{LA} - \mathbf{PLA}\|_{\mathrm{F}} - \|\mathbf{EA}\|_{\mathrm{F}} - \|\mathbf{PEA}\|_{\mathrm{F}} \\
&\geq \|\mathbf{LA} - \mathbf{PLA}\|_{\mathrm{F}} - \|\mathbf{E}\|_{\mathrm{F}}\|\mathbf{A}\|_{\mathrm{F}} - \|\mathbf{P}\|_2\|\mathbf{E}\|_{\mathrm{F}}\|\mathbf{A}\|_{\mathrm{F}} \\
&= \|\mathbf{A} - \mathbf{L}^{\mathrm{T}}\mathbf{PLA}\|_{\mathrm{F}} - O(1/n^3), \qquad (20)
\end{aligned}$$

where the equality uses that multiplying by $\mathbf{L}^{\mathrm{T}}$ preserves norms, as well as that $\|\mathbf{E}\|_{\mathrm{F}} \leq 1/\mathrm{poly}(n)$ for an arbitrarily small $\mathrm{poly}(n)$ by making $M \leq 1/n^c$ sufficiently small, whereas $\|\mathbf{P}\|_2 \leq 1$ and $\|\mathbf{A}\|_{\mathrm{F}} = O(\sqrt{n})$. We claim $\mathbf{L}^{\mathrm{T}}\mathbf{PL} = \mathbf{Q}$, where $\mathbf{Q}$ is the projection onto the columns in $\mathbf{A}$ indexed by the set S. To see this, let $\mathbf{U}$ be an orthonormal basis for the columns in $\mathbf{A}$ spanned by the index set S, so that $\mathbf{P} = \mathbf{LUU}^{\mathrm{T}}\mathbf{L}^{\mathrm{T}}$. Then $\mathbf{L}^{\mathrm{T}}\mathbf{PL} = \mathbf{UU}^{\mathrm{T}} = \mathbf{Q}$. Hence, using that $\|\mathbf{A} - \mathbf{QA}\|_{\mathrm{F}} \leq \|\mathbf{A}\|_{\mathrm{F}} = O(\sqrt{n})$ and squaring (20), we have,

$$\|\tilde{\mathbf{L}}\mathbf{A} - \mathbf{P}\tilde{\mathbf{L}}\mathbf{A}\|_{\mathrm{F}}^2 \geq \|\mathbf{A} - \mathbf{QA}\|_{\mathrm{F}}^2 - O(1/n^2). \qquad (21)$$

Combining (18), (19), and (21),

$$\|\mathbf{A} - \mathbf{QA}\|_{\mathrm{F}}^2 \leq (1 + \epsilon/3)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 + O(1/n^2). \qquad (22)$$

Note that $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} \geq 1$ for any $k < n$, since if we remove the top row of $\mathbf{A}$, creating a matrix $\mathbf{B}$, then $\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}} \leq \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}$, yet $\mathbf{B}$ is the identity matrix, and so $\|\mathbf{B} - \mathbf{B}_k\|_{\mathrm{F}} \geq 1$. It follows that the additive $O(1/n^2)$ error in (22) translates into a relative error, provided $k < n$ and for $1/\varepsilon < n/3$. Hence, $\|\mathbf{A} - \mathbf{QA}\|_{\mathrm{F}}^2 \leq (1 + \epsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2$, and therefore by Fact 68, we have $|S| > k/(2\epsilon)$. ■

9.3.2 Column sharing of discretized orthonormal matrices

We need the following technical lemma about sampling random orthonormal matrices, concerning the number of columns they have in common after rounding.

Lemma 70. *Let $1 \leq r \leq n$, and let $M \leq 1/n^c$ for a sufficiently large absolute constant $c > 0$. Suppose we choose two independently random orthogonal matrices $\mathbf{A} \in \mathbb{R}^{n\times n}$ and $\mathbf{B} \in \mathbb{R}^{n\times n}$ from the Haar measure (so they each have orthonormal columns and orthonormal rows). We then round each of the entries in $\mathbf{A}$ to the nearest integer multiple of M, obtaining a matrix $\tilde{\mathbf{A}}$, and similarly obtain $\tilde{\mathbf{B}}$. Let $\mathcal{E}$ denote the event that there exist r distinct indices $i_1, \ldots, i_r \in [n]$ and r distinct indices $j_1, \ldots, j_r \in [n]$ for which $\tilde{\mathbf{A}}_{*,i_\ell} = \tilde{\mathbf{B}}_{*,j_\ell}$ for all $\ell = 1, 2, \ldots, r$, where $\tilde{\mathbf{A}}_{*,i_\ell}$ denotes the i_ℓ-th column of $\tilde{\mathbf{A}}$. Then, $\Pr[\mathcal{E}] \leq e^{-\Theta(nr\log M)}$.*

Proof. We fix a subset S of r distinct indices $i_1, \ldots, i_r \in [n]$ and a subset T of r distinct indices $j_1, \ldots, j_r \in [n]$ and show that the probability $\tilde{\mathbf{A}}_{*,i_\ell} = \tilde{\mathbf{B}}_{*,j_\ell}$ simultaneously for all $\ell \in [r]$ is at most $e^{-\Theta(nr \log M)}$. It then follows by a union bound that

$$\Pr[\mathcal{E}] \leq \binom{n}{r} \cdot \binom{n}{r} \cdot e^{-\Theta(nr \log M)} \leq \left(\frac{ne}{r}\right)^{2r} e^{-\Theta(nr \log M)} \leq e^{-\Theta(nr \log M)},$$

as desired.

Since $\mathbf{A}$ and $\mathbf{B}$ are independent and their distribution is permutation-invariant, we can assume $i_\ell = j_\ell = \ell$ for $\ell \in [r]$. Let $\mathcal{F}$ be the event that $\tilde{\mathbf{A}}_{*,i_\ell} = \tilde{\mathbf{B}}_{*,j_\ell}$ occurs for all $\ell \in [r]$. If $\mathcal{F}$ occurs, then we need $\|\mathbf{A}_{*,i_\ell} - \mathbf{B}_{*,j_\ell}\|_\infty \leq M$ for all $\ell \in [r]$, which implies $\|\mathbf{A}_{*,i_\ell} - \mathbf{B}_{*,j_\ell}\|_2 \leq \sqrt{n}M$ for all $\ell \in [r]$. Let $\mathbf{A}^r$ be the leftmost r columns of $\mathbf{A}$, and $\mathbf{B}^r$ the leftmost r columns of $\mathbf{B}$. Then, if $\mathcal{F}$ occurs

$$\|\mathbf{A}^r - \mathbf{B}^r\|_2^2 \leq \|\mathbf{A}^r - \mathbf{B}^r\|_{\mathrm{F}}^2 = \sum_{\ell=1}^{r} \|\mathbf{A}_{*,\ell} - \mathbf{B}_{*,\ell}\|_2^2 \leq rnM^2.$$

Note that $\mathbf{A}^r$ and $\mathbf{B}^r$ have orthonormal columns, and so $\mathbf{P}_{\mathbf{A}^r} = (\mathbf{A}^r)(\mathbf{A}^r)^{\mathrm{T}}$ and $\mathbf{P}_{\mathbf{B}^r} = (\mathbf{B}^r)(\mathbf{B}^r)^{\mathrm{T}}$ are projection matrices. Then,

$$\begin{aligned}
\|\mathbf{A}^r - \mathbf{B}^r\|_2 &\geq \|(\mathbf{A}^r)(\mathbf{A}^r)^{\mathrm{T}} - \mathbf{B}^r(\mathbf{A}^r)^{\mathrm{T}}\|_2/2 + \|(\mathbf{A}^r)(\mathbf{B}^r)^{\mathrm{T}} - (\mathbf{B}^r)(\mathbf{B}^r)^{\mathrm{T}}\|_2/2 \\
&= \|(\mathbf{A}^r)(\mathbf{A}^r)^{\mathrm{T}} - (\mathbf{A}^r)(\mathbf{B}^r)^{\mathrm{T}}\|_2/2 + \|(\mathbf{A}^r)(\mathbf{B}^r)^{\mathrm{T}} - (\mathbf{B}^r)(\mathbf{B}^r)^{\mathrm{T}}\|_2/2 \\
&\geq \|(\mathbf{A}^r)(\mathbf{A}^r)^{\mathrm{T}} - (\mathbf{B}^r)(\mathbf{B}^r)^{\mathrm{T}}\|_2/2, \qquad (23)
\end{aligned}$$

where the first inequality follows since multiplying by a matrix on the right with orthonormal rows cannot increase the spectral norm, the equality follows by taking transposes, and the second inequality follows by the triangle inequality.

We need another result from [35].

Theorem 71. *(Probability Bound - Claim 5.2 of [35]) Suppose we choose two independent random subspaces Y and S of $\mathbb{R}^n$ each of dimension r from the Haar measure, where we denote the projection operators onto the subspaces by $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ and $\mathbf{S}\mathbf{S}^{\mathrm{T}}$, respectively, where $\mathbf{Y}$ and $\mathbf{S}$ have r orthonormal columns. Then for any $\delta \in (0,1)$, $\Pr[\|\mathbf{Y}\mathbf{Y}^{\mathrm{T}} - \mathbf{S}\mathbf{S}^{\mathrm{T}}\|_2 \leq \delta] \leq e^{-\Theta(r(n-r)\log(1/\delta))}$.*

Combining (23) with Theorem 71, we have that for $r \leq n/2$, $\Pr[\mathcal{F}] \leq e^{-\Theta(rn \log(M))}$, since by taking $M \leq 1/n^c$ sufficiently small, we can ensure $\|\mathbf{A}^r - \mathbf{B}^r\|_2 \leq \sqrt{rn}M \leq M^{1/2}$, and so $2M^{1/2}$ upper bounds $\|(\mathbf{A}^r)(\mathbf{A}^r)^{\mathrm{T}} - (\mathbf{B}^r)(\mathbf{B}^r)^{\mathrm{T}}\|_2$.

Note that if $r > n/2$, then in particular we still have that $\tilde{\mathbf{A}}_{*,i_\ell} = \tilde{\mathbf{B}}_{*,j_\ell}$ for all $\ell \in [r/2]$, and therefore in this case we also have $\Pr[\mathcal{F}] \leq e^{-\Theta(rn \log M)}$. Hence, $\Pr[\mathcal{E}] \leq e^{-\Theta(rn \log M)}$, which completes the proof. ∎

We now extend Lemma 70 to a k-fold version.

Lemma 72. *Let $M \leq 1/n^c$ for a sufficiently large absolute constant $c > 0$. Suppose we independently choose k pairs of matrices $\mathbf{A}^z, \mathbf{B}^z \in \mathbb{R}^{n \times n}$, $z \in [k]$, each with orthonormal columns (and hence also orthonormal rows). We then round each of the entries in each $\mathbf{A}^z$ and $\mathbf{B}^z$ to the nearest integer multiple of M, obtaining matrices $\tilde{\mathbf{A}}^z$, and $\tilde{\mathbf{B}}^z$. Let $r = (r_1, \ldots, r_k) \in (\mathbb{Z}^{\geq 0})^k$ and let $\mathcal{E}$ be the event that for each $z \in [k]$, there exist distinct indices $i_1, \ldots, i_{r_z} \in [n]$ and $j_1, \ldots, j_{r_z} \in [n]$ for which $\tilde{\mathbf{A}}^z{}_{*,i_\ell} = \tilde{\mathbf{B}}^z{}_{*,j_\ell}$ for all $\ell = 1, 2, \ldots, r_z$. Then, $\Pr[\mathcal{E}] \leq e^{-\Theta(n \sum_{z=1}^k r_z \log M)}$.*

Proof. This follows by Lemma 70, and the fact that the matrices $\mathbf{A}^1, \ldots, \mathbf{A}^k, \mathbf{B}^1, \ldots, \mathbf{B}^k$ are jointly independent. Here we also use a union bound. ∎

Corollary 73. *Suppose $1 \leq k < c_0 n$ for an absolute constant $c_0 > 0$. Let $r = (r_1, \ldots, r_k) \in (\mathbb{Z}^{\geq 0})^k$. Let $M \leq 1/n^c$ for a sufficiently large constant $c > 0$. There exists a set $\mathcal{N}$ of $e^{\Theta(nt \log M)}$ k-tuples $(\tilde{\mathbf{A}}_1, \ldots, \tilde{\mathbf{A}}_k)$ of matrices each of dimensions $n \times n$ with entries that are integer multiples of M and bounded in magnitude by 1, such that for every vector $r = (r_1, \ldots, r_k) \in (\mathbb{Z}^{\geq 0})^k$ with $\sum_{z=1}^k r_z \leq t$, it holds that for all distinct k-tuples $(\tilde{\mathbf{A}}_1, \ldots, \tilde{\mathbf{A}}_k), (\tilde{\mathbf{B}}_1, \ldots, \tilde{\mathbf{B}}_k)$, there exists a $z \in [k]$ for which there are no sets $S = \{i_1, \ldots, i_{r_z}\}, T = \{j_1, \ldots, j_z\} \subset [n]$ for which $\tilde{\mathbf{A}}_{*,i_\ell} = \tilde{\mathbf{B}}_{*,j_\ell}$ for all $\ell = 1, 2, \ldots, r_z$.*

Proof. For a fixed vector r and a random choice of $(\tilde{\mathbf{A}^1}, \ldots, \tilde{\mathbf{A}^k}), (\tilde{\mathbf{B}^1}, \ldots, \tilde{\mathbf{B}^k})$, by the guarantee of Lemma 72, the complement of the event in the statement of the corollary happens with probability at most $e^{-\Theta(nt \log M)}$. The number of vectors r with $\sum_{z=1}^k r_z \leq t$ is at most t^k, and $t^k e^{-\Theta(nt \log M)} \leq e^{-\Theta(nt \log M)}$ provided $k < c_0 n$ for an absolute constant $c_0 > 0$, so we can apply a union bound over all pairs in a random choice of $e^{\Theta(nt \log M)}$ k-tuples. ■

9.3.3 Hard instance construction

We have s servers holding matrices $\mathbf{A}_1, \ldots, \mathbf{A}_s$, respectively, where $\mathbf{A}_i$ has m rows and a subset of w_i columns of an $m \times n$ matrix $\mathbf{A}$, where $\sum w_i = n$. For our lower bound we assume $2/\varepsilon \leq \phi$ and $k\phi \leq m$. Note that for ε less than a sufficiently small constant, this implies that $k < c_0 k\phi$, and so the assumption in Corollary 73 is valid (since the n of that corollary is equal to $(k+1)\phi$ We set $M = 1/(mn)^c$ for a sufficiently large constant $c > 0$.

We will have $\mathbf{A}_1$ being an $m \times k(\phi - 1)$ matrix for which all but the first $k\phi$ rows are zero. We assume $2/\varepsilon \leq \phi$. On the first $k\phi$ rows, $\mathbf{A}_1$ is block diagonal containing k blocks, each block being a $\phi \times (\phi - 1)$ matrix.

To specify $\mathbf{A}_1$, let the parameters t and n of Corollary 73 equal $k/(2\varepsilon)$ and ϕ, respectively. That corollary gives us a net $\mathcal{N}$ of $e^{\Theta(k\phi(\log(mn))/\varepsilon)}$ k-tuples of matrices each of dimensions $\phi \times \phi$ with entries that are integer multiple of M and bounded in magnitude by 1.

For a random k-tuple $(\mathbf{B}^1, \ldots, \mathbf{B}^k)$ in $\mathcal{N}$, we create an $m \times m$ block diagonal matrix $\mathbf{B}$, which is 0 on all but its first $k\phi$ rows and first $k\phi$ columns. On its first $k\phi$ rows, it is a block diagonal matrix $\mathbf{B}$ whose blocks along the diagonal are $\mathbf{B}^1, \ldots, \mathbf{B}^k$, respectively. Our matrix $\mathbf{A}_1$ is then the matrix product of $\mathbf{B}$ and an matrix $\mathbf{D}$, where $\mathbf{D}$ is $m \times k(\phi - 1)$ is formed by taking the $k\phi \times k(\phi - 1)$ matrix of Fact 68 and padding it with $m - k\phi$ rows which are all zeros. By Lemma 69, no $k/(2\epsilon)$ columns of $\mathbf{A}_1$ contain a k-dimensional subspace in their span which is a $(1+\epsilon)$-approximation to the best rank-k approximationt to $\mathbf{A}_1$.

Each $\mathbf{A}_i$ for $i = 2, 3, \ldots, s-1$ is the $m \times m$ matrix of all zeros. Finally, $\mathbf{A}_s$ is the $m \times t$ matrix of all zeros with $t = n - (s-1)m - k(\phi - 1)$, i.e., $\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{0}_{m \times m} & \mathbf{0}_{m \times m} & \ldots & \mathbf{0}_{m \times m} & \mathbf{0}_{m \times t} \end{pmatrix}$. By construction, each column of $\mathbf{A}$ has at most ϕ non-zero entries, since the only non-zero columns are those in $\mathbf{A}_1$, and each column in $\mathbf{A}_1$ has the form $\mathbf{B}^z_{*,1} + \mathbf{B}^z_{*,i}$ for a $z \in [k]$ and an $i \in [\phi]$, so it has at most ϕ non-zero entries.

9.3.4 Main theorem

In the Distributed Column Subset Selection Problem in Definition 3, a correct protocol should have each of the s machines simultaneously output a matrix $\mathbf{C} \in \mathbb{R}^{m \times c}$ with $c < n$ columns of $\mathbf{A}$ such that

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\|_{\mathrm{F}}^2 \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

Theorem 74. *Assume $2/\varepsilon \leq \phi$ and $k\phi \leq \min(m, n)$. Then, any, possibly randomized, protocol Π which succeeds with probability at least $2/3$ in solving the Distributed Column Subset Selection Problem of Definition 3, under the promise that each column of $\mathbf{A}$ has at most ϕ non-zero entries which are integer*

multiples of $1/(mn)^c$ and bounded in magnitude by 1, for a constant $c > 0$, requires $\Omega(s\phi k(\log(mn))/\varepsilon)$ bits of communication. If the word size is $\Theta(\log(mn))$, this implies the communication cost of the porticol is $\Omega(s\phi k/\varepsilon)$ words.

Proof. By construction of $\mathbf{A}_1$, whenever Π succeeds, each of the s machines outputs the same set $\mathbf{C}$ of at least $t \geq k/(2\epsilon)$ columns of $\mathbf{A}_1$.

Let Π^i denote the transcript (sequence of all messages) between the coordinator and the i-th player. We look at the information Π^i reveals about $\mathbf{A}_1$, given the event $\mathcal{E}$ that Π succeeds, that is, $I(\Pi^i; \mathbf{A}_1 \mid \mathcal{E})$.

A critical observation is that any column $\mathbf{C}_{*,i}$ of $\mathbf{C}$, is equal to $\mathbf{B}^z_{*,1} + \mathbf{B}^z_{*,i}$ for some $z \in [k]$, where $\mathbf{B}^z$ is as defined above (i.e., $\mathbf{A}_1 = \mathbf{B} \cdot \mathbf{D}$). Hence, given $\mathbf{B}^z_{*,1}$ and $\mathbf{C}_{*,i}$, one can reconstruct $\mathbf{B}^z_{*,i}$. Now, for any random variable W,

$$I(\Pi^i; \mathbf{A}_1 \mid \mathcal{E}) \geq I(\Pi^i; \mathbf{A}_1 \mid \mathcal{E}, W) - H(W). \tag{24}$$

We let $W = (\mathbf{B}^1_{*,1}, \ldots, \mathbf{B}^k_{*,1})$, and observe that $H(W) \leq \log_2((2M)^{k\phi})$ since there are $2M$ choices for each coordinate of each $\mathbf{B}^z_{*,1}$. Hence, $H(W) = O(k\phi \log(mn))$. Note that $I(\Pi^i; \mathbf{A}_1 \mid \mathcal{E}, W) = H(\mathbf{A}_1) = \log_2(|\mathcal{N}|)$, since conditioned on the protocol succeeding, and given W, by Corollary 73 we can reconstruct $\mathbf{A}_1$. By (24), it follows that

$$(\Pi^i; \mathbf{A}_1 \mid \mathcal{E}) \geq \Theta(\phi k(\log(mn))/\epsilon) - O(k\phi \log(mn)) = \Omega(\phi k(\log(mn))/\epsilon).$$

Note that if Z is an indicator random variable that indicates that $\mathcal{E}$ occurs, we have $H(Z) \leq 1$, and so

$$I(\Pi^i; \mathbf{A}_1) \geq I(\Pi^i; \mathbf{A}_1 \mid \mathcal{E}) \Pr[\mathcal{E}] - 1 = \Omega(\phi k(\log(mn))/\epsilon),$$

since $\Pr[\mathcal{E}] \geq 2/3$.

It follows that

$$\Omega(\phi k(\log(mn))/\epsilon) \leq I(\Pi^i; \mathbf{A}_1) \leq H(\Pi^i) \leq \mathbf{E}_{\mathbf{A}_1, rand}[|\Pi^i|],$$

where $|\Pi^i|$ denotes the length, in bits, of the sequence Π^i of messages, and $rand$ is the concatenation of the private randomness of all s machines. Note that the entropy of Π^i is a lower bound on the expected encoding length of $|\Pi^i|$, by the Shannon coding theorem. By linearity of expectation,

$$\sum_i \mathbf{E}_{\mathbf{A}_1, rand}|\Pi^i| = \Omega(\phi k(\log(mn))/\epsilon),$$

which implies there exists an $\mathbf{A}_1$ and setting of $rand$ for which

$$\sum_i |\Pi^i| = \Omega(s\phi k(\log(mn))/\epsilon).$$

It follows that the total communication is $\Omega(s\phi k(\log(mn))/\epsilon)$ bits. ∎

Corollary 75. *Assume $2/\varepsilon \leq \phi$ and $k\phi \leq \min(m, n)$. Then, any, possibly randomized, protocol Π which succeeds with probability at least $2/3$ in solving the Distributed Column Subset Selection Problem - rank k subspace version (see Definition 4), promise that each column of $\mathbf{A}$ has at most ϕ non-zero entries which are integer multiples of $1/(mn)^c$ and bounded in magnitude by 1, for a constant $c > 0$, requires $\Omega(s\phi k(\log(mn))/\varepsilon)$ bits of communication. If the word size is $\Theta(\log(mn))$, this implies the total communication across all s machines is $\Omega(s\phi k/\varepsilon)$ words.*

Proof. As any such protocol is also a protocol for the Distributed Column Subset Selection Problem of Definition 3 (the problem in Definition 4 is more general/difficult than the problem of Definition 3), the proof follows immediately from Theorem 74. ∎

10 Conclusions

This paper makes substantial progress in resolving the communication complexity of the Distributed PCA problem (see Definition 2). First, when the matrix is dense, we provide a novel distributed PCA algorithm improving upon all previous approaches [24, 8, 37, 26, 34, 12]. Second, we provide a matching communication lower bound, indicating that our algorithm is communication-optimal (up to low order terms). Third, we provide the first algorithm for distributed PCA on sparse matrices that has communication cost which is independent of the matrix dimensions.

Along the way, we resolve the communication complexity of the Distributed Column Subset Selection problem (see Definitions 3 and 4) by providing distributed algorithms and matching communication lower bounds. The Column Subset Selection problem has been studied exhaustively within the numerical linear algebra and theoretical computer science communities and has been very well understood in the batch setting [15]. Here, we study the problem in the distributed setting and resolve its communication complexity up to low order terms.

Furthermore, we demonstrate that simple variants of our distributed PCA algorithms can be implemented efficiently in the streaming model of computation to provide the first space-optimal (up to low order terms and the distinction between words versus bits) streaming PCA algorithms. Our one-pass algorithm improves upon the best such one-pass algorithm in the literature [37, 26]. Surprisingly, our two-pass algorithm demonstrates that with just an additional pass over the data matrix, one can achieve considerably better space complexity for streaming PCA.

An important open problem in our work is to resolve the communication complexity of the Distributed PCA problem in the case of a sparse matrix. Our upper and lower bounds have an $O(\varepsilon^{-1})$ gap (see Table 1). Designing a better upper bound or a better lower bound would give the complete picture of the communication complexity of Distributed PCA. An interesting future direction is to investigate whether our results generalize to the arbitrary partition model of [34].

References

[1] `http://hadoop.apache.org/`.

[2] `https://mahout.apache.org`.

[3] `https://spark.apache.org/`.

[4] `https://spark.apache.org/mllib/`.

[5] D. Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4):671–687, 2003.

[6] N. Ailon and E. Liberty. Fast dimension reduction using rademacher series on dual bch codes. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2008.

[7] Z.-J. Bai, R. H. Chan, and F. T. Luk. Principal component analysis for distributed data sets with updating. In *Advanced Parallel Processing Technologies*, pages 471–483. Springer, 2005.

[8] M.-F. Balcan, V. Kanchanapally, Y. Liang, and D. Woodruff. Improved distributed principal component analysis. *arXiv preprint arXiv:1408.5823, to appear in NIPS*, 2014.

[9] M.-F. Balcan, Y. Liang, L. Song, D. Woodruff, and B. Xie. Distributed kernel principal component analysis. *arxiv preprint*, 2015.

[10] J. Batson, D. Spielman, and N. Srivastava. Twice-ramanujan sparsifiers. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 255–262. ACM, 2009.

[11] A. R. Benson, D. F. Gleich, and J. Demmel. Direct qr factorizations for tall-and-skinny matrices in mapreduce architectures. In *Big Data, 2013 IEEE International Conference on*, pages 264–272. IEEE, 2013.

[12] S. Bhojanapalli, P. Jain, and S. Sanghavi. Tighter low-rank approximation via sampling the leveraged element. `http://uts.cc.utexas.edu/~bsrinadh/main.pdf`, *to appear in SODA*, 2015.

[13] L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, et al. *ScaLAPACK users' guide*, volume 4. siam, 1997.

[14] J. Bourgain and J. Nelson. Toward a unified theory of sparse dimensionality reduction in euclidean space. In *manuscriptg*, 2014.

[15] C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near optimal column based matrix reconstruction. *SIAM Journal on Computing (SICOMP)*, 2013.

[16] C. Boutsidis and A. Gittens. Improved matrix algorithms via the subsampled randomized hadamard transform. *SIAM Journal on Matrix Analysis and Applications*, 34(3):1301–1340, 2013.

[17] C. Boutsidis and D. P. Woodruff. Optimal cur matrix decompositions. *STOC*, 2014.

[18] K. Clarkson and D. Woodruff. Numerical linear algebra in the streaming model. In *Proceedings of the 41st annual ACM symposium on Theory of computing (STOC)*, 2009.

[19] K. L. Clarkson and D. P. Woodruff. Low rank approximation and regression in input sparsity time. *ArxiV report:* `http://arxiv.org/pdf/1207.6365v4.pdf`, 2013.

[20] K. L. Clarkson and D. P. Woodruff. Low rank approximation and regression in input sparsity time. In *In STOC*, 2013.

[21] A. Deshpande, L. Rademacher, S. Vempala, and G. Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(12):225–247, 2006.

[22] A. Deshpande and S. Vempala. Adaptive sampling and fast low-rank matrix approximation. In *RANDOM - APPROX*, 2006.

[23] A. K. Farahat, A. Elgohary, A. Ghodsi, and M. S. Kamel. Distributed column subset selection on mapreduce. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, pages 171–180. IEEE, 2013.

[24] D. Feldman, M. Schmidt, and C. Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *SODA*, pages 1434–1453. SIAM, 2013.

[25] S. Friedland and A. Torokhti. Generalized rank-constrained matrix approximations. *SIAM Journal on Matrix Analysis and Applications*, 29(2):656–659, 2007.

[26] M. Ghashami and J. Phillips. Relative errors for deterministic low-rank matrix approximations. *In SODA*, 2013.

[27] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[28] M. Gu and S. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM Journal on Scientific Computing*, 17:848–869, 1996.

[29] M. Gu and S. C. Eisenstat. A divide-and-conquer algorithm for the bidiagonal svd. *SIAM Journal on Matrix Analysis and Applications*, 16(1):79–92, 1995.

[30] R. Horn and C. Johnson. *Matrix Analysis*. Cambridge University Press, New York, 1985.

[31] E. Jessup and D. Sorensen. A parallel algorithm for computing the singular value decomposition of a matrix. *Siam Journal on Matrix Analysis and Applications*, 15(2):530–548, 1994.

[32] D. M. Kane and J. Nelson. Sparser johnson-lindenstrauss transforms. *J. ACM*, 61(1):4, 2014.

[33] R. Kannan, S. Vempala, and D. P. Woodruff. Nimble algorithms for cloud computing. *arXiv preprint arXiv:1304.3162, v3*, 2013.

[34] R. Kannan, S. S. Vempala, and D. P. Woodruff. Principal component analysis and higher correlations for distributed data. In *Proceedings of The 27th Conference on Learning Theory*, pages 1040–1057, 2014.

[35] M. Kapralov and K. Talwar. On differentially private low rank approximation. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1395–1414, 2013.

[36] D. Kempe and F. McSherry. A decentralized algorithm for spectral analysis. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 561–568. ACM, 2004.

[37] E. Liberty. Simple and deterministic matrix sketching. In *KDD*, pages 581–588. ACM, 2013.

[38] S. V. Macua, P. Belanovic, and S. Zazo. Consensus-based distributed principal component analysis in wireless sensor networks. In *Signal Processing Advances in Wireless Communications (SPAWC), 2010 IEEE Eleventh International Workshop on*, pages 1–5. IEEE, 2010.

[39] X. Meng and M. W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 91–100. ACM, 2013.

[40] J. Nelson and H. L. Nguyên. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 117–126. IEEE, 2013.

[41] J. Poulson, B. Marker, R. A. van de Geijn, J. R. Hammond, and N. A. Romero. Elemental: A new framework for distributed memory dense matrix computations. *ACM Transactions on Mathematical Software (TOMS)*, 39(2):13, 2013.

[42] Y. Qu, G. Ostrouchov, N. Samatova, and A. Geist. Principal component analysis for dimension reduction in massive distributed data sets. In *Proceedings of IEEE International Conference on Data Mining (ICDM)*, 2002.

[43] T. Sarlos. Improved approximation algorithms for large matrices via random projections. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.

[44] K. C. Sou and A. Ranzer. On generalized matrix approximation problem in the spectral norm. *Linear Algebra and its Applications*, 436(7):2331–2341, 2012.

[45] F. Tisseur and J. Dongarra. A parallel divide and conquer algorithm for the symmetric eigenvalue problem on distributed memory architectures. *SIAM Journal on Scientific Computing*, 20(6):2223–2236, 1999.

[46] M. D. Vose. A linear algorithm for generating random numbers with a given distribution. *Software Engineering, IEEE Transactions on*, 17(9):972–975, 1991.

[47] V. H. Vu and T. Tao. The condition number of a randomly perturbed matrix. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 248–255, 2007.

[48] D. Woodruff. Low rank approximation lower bounds in row-update streams. In *Advances in Neural Information Processing Systems*, pages 1781–1789, 2014.

[49] D. P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.

[50] S. R. Yann-Ael Le Borgne and G. Bontempi. Distributed principal component analysis for wireless sensor networks. *Sensors*, 2008.