Distributed Estimation of Generalized Matrix Rank: Efficient Algorithms and Lower Bounds

Yuchen Zhang YUCZHANG@EECS.BERKELEY.EDU
University of California, Berkeley, CA 94720, USA

Martin J. Wainwright WAINWRIG@STAT.BERKELEY.EDU
University of California, Berkeley, CA 94720, USA

Michael I. Jordan JORDAN@CS.BERKELEY.EDU
University of California, Berkeley, CA 94720, USA

Abstract

We study the following generalized matrix rank estimation problem: given an $n \times n$ matrix and a constant $c \geq 0$, estimate the number of eigenvalues that are greater than c. In the distributed setting, the matrix of interest is the sum of m matrices held by separate machines. We show that any deterministic algorithm solving this problem must communicate $\Omega(n^2)$ bits, which is order-equivalent to transmitting the whole matrix. In contrast, we propose a randomized algorithm that communicates only $\widetilde{\mathcal{O}}(n)$ bits. The upper bound is matched by an $\Omega(n)$ lower bound on the randomized communication complexity. We demonstrate the practical effectiveness of the proposed algorithm with some numerical experiments.

1. Introduction

Given a parameter $c \geq 0$, the generalized rank of an $n \times n$ positive semi-definite matrix A corresponds to the number of eigenvalues that are larger than c. It is denoted by $\operatorname{rank}(A, c)$, with the usual rank corresponding to the special case $c = 0$. Estimating the generalized rank of a matrix is useful for many applications. Many machine learning algorithms require knowledge of the approximate rank of the data matrix in order to set hyper-parameters. In the context of large-scale principal component analysis (PCA) (Eckart & Young, 1936; Jolliffe, 2005), it is overly expensive to compute the full eigen-decomposition before deciding when to truncate it. Thus, an important first step is to estimate the rank of the matrix of interest in order to determine how many dimensions will be sufficient to describe the data. The rank also provides useful information for determining the tuning parameter of robust PCA (Candès et al., 2011) and collaborative filtering algorithms for personalized recommendation (Sarwar et al., 2001; Rendle et al., 2009). In the context of numerical linear algebra, a number of eigensolvers (Schofield et al., 2012; Polizzi, 2009; Sakurai & Sugiura, 2003) for large-scale scientific applications are based on divide-and-conquer paradigms. It is a prerequisite of these algorithms to know the approximate number of eigenvalues located in a given interval. Estimating the generalized rank of a matrix is also needed in the context of sampling-based methods for randomized numerical linear algebra (Halko et al., 2011; Mahoney, 2011). For these methods, the rank of a matrix determines the number of samples required for a desired approximation accuracy.

Motivated by large-scale data analysis problems, in this paper we study the generalized rank estimation problem in a distributed setting, in which the matrix A can be decomposed as the the sum of m matrices

$$A := \sum_{i=1}^{m} A_i, \tag{1}$$

where each matrix A_i is stored on a separate machine i. Thus, a distributed algorithm needs to communicate between m machines to perform the estimation. There are other equivalent formulations of this problem. For example, suppose that machine i has a design matrix $X_i \in \mathbb{R}^{n \times N_i}$ and we want to determine the rank of the aggregated design matrix

$$X := (X_1, X_2, \ldots, X_m) \in \mathbb{R}^{n \times N} \text{ where } N := \textstyle\sum_{i=1}^{m} N_i.$$

Recall that the singular values of matrix X are equal to the square root of the eigenvalues of the matrix XX^T. If we

Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

define $A_i := X_i X_i^T$, then equation (1) implies that

$$A = \sum_{i=1}^{m} A_i = \sum_{i=1}^{m} X_i X_i^T = XX^T.$$

Thus, determining the generalized rank of the matrix X reduces to the problem of determining the rank of the matrix A. In this paper, we focus on the formulation given by equation (1).

The standard way of computing the generalized matrix rank, or more generally of computing the number of eigenvalues within a given interval, is to exploit Sylvester's law of inertia (Golub & Van Loan, 2012). Concretely, if the matrix $A - cI$ admits the decomposition $A - cI = LDL^T$, where L is unit lower triangular and D is diagonal, then the number of eigenvalues of matrix A that are greater than c is the same as the number of positive entries in the diagonal of D. While this method yields an exact count, in the distributed setting it requires communicating the entire matrix A. Due to bandwidth limitations and network delays, the $\Theta(n^2)$ communication cost is a significant bottleneck on the algorithmic efficiency. For a matrix of rank r, the power method (Golub & Van Loan, 2012) can be used to compute the top r eigenvalues, which reduces the communication cost to $\Theta(rn)$. However, this cost is still prohibitive for moderate sizes of r. Recently, Napoli et al. (2013) studied a more efficient randomized approach for approximating the eigenvalue counts based on Chebyshev polynomial approximation of high-pass filters. When applying this algorithm to the distributed setting, the communication cost is $\Theta(pn)$, where p is the degree of Chebyshev polynomials. However, the authors note that polynomials of high degree can be necessary.

In this paper, we study the communication complexity of distributed algorithms for the problem of generalized rank estimation, in both the deterministic and randomized settings. We establish upper bounds by deriving practical, communication-efficient algorithms, and we also establish complexity-theoretic lower bounds. Our first main result shows that no deterministic algorithm is efficient in terms of communication. In particular, communicating $\Omega(n^2)$ bits is necessary for all deterministic algorithms to approximate the matrix rank with constant relative error. That such algorithms cannot be viewed as efficient is due to the fact that by communicating $\mathcal{O}(n^2)$ bits, we are able to compute all eigenvalues and the corresponding eigenvectors. In contrast to the inefficiency of deterministic algorithms, we propose a randomized algorithm that approximates matrix rank by communicating $\widetilde{\mathcal{O}}(n)$ bits. When the matrix is of rank r, the relative approximation error is $1/\sqrt{r}$. Under the same relative error, we show that $\Omega(n)$ bits of communication is necessary, establishing the optimality of our algorithm. This is in contrast with the $\Omega(rn)$ communication complexity lower bound for randomized PCA (Kannan et al., 2014). The difference shows that estimating the eigenvalue count using a randomized algorithm is easier than estimating the top r eigenpairs.

The research on communication complexity has a long history, dating back to the seminal work of Yao (1979) and Abelson (1980). Characterizing the communication complexity of linear algebraic operations is a fundamental question. For the problem of rank testing, Chu and Schnitger (1991; 1995) prove the $\Omega(n^2)$ communication complexity lower bound for deterministically testing the singularity of integer-valued matrices. A successful algorithm for this task is required to distinguish two types of matrices—the singular matrices and the non-singular matrices with arbitrarily small eigenvalues—a requirement that is often too severe for practical applications. Luo and Tsitsiklis (1993) prove an $\Omega(n^2)$ lower bound for computing one entry of A^{-1}, applicable to exact algorithms (with no form of error allowed). In contrast, our deterministic lower bound holds even if we force the non-zero eigenvalues to be bounded away from zero and allow for approximation errors, making it more widely applicable to the inexact algorithms used in practice. For randomized algorithms, Sun et al. (2012) and Li et al. (2014) prove $\Omega(n^2)$ lower bounds for the problems of rank testing, computing a matrix inverse, and solving a set of linear equations over finite fields. To the best of our knowledge, it is not known whether the same lower bounds hold for matrices in the real field. In other related work, Clarkson and Woodruff (2009) give an $\Omega(r^2)$ space lower bound in the streaming model for distinguishing between matrices of rank r and $r-1$. However, such a space lower bound in the streaming model does not imply a communication complexity lower bound in the two-way communication model studied in this paper.

2. Background and problem formulation

In this section, we begin with more details on the problem of estimating generalized matrix ranks, as well as some background on communication complexity.

2.1. Generalized matrix rank

Given an $n \times n$ positive semidefinite matrix A, we use $\sigma_1(A) \geq \sigma_2(A) \geq \cdots \geq \sigma_n(A) \geq 0$ to denote its ordered eigenvalues. For a given constant $c \geq 0$, the generalized rank of order c is given by

$$\operatorname{rank}(A, c) = \sum_{k=1}^{n} \mathbb{I}[\sigma_k(A) > c], \tag{2}$$

where $\mathbb{I}[\sigma_k(A) > c]$ is a 0-1-valued indicator function for the event that $\sigma_k(A)$ is larger than c. Since $\operatorname{rank}(A, 0)$ is equal to the usual rank of a matrix, we see the motivation for using the generalized rank terminology. We assume that

$\|A\|_2 = \sigma_1(A) \leq 1$ so that the problem remains on a standardized scale.

In an m-machine distributed setting, the matrix A can be decomposed as a sum $A = \sum_{i=1}^m A_i$, where the $n \times n$ matrix A_i is stored on machine i. We study distributed protocols, to be specified more precisely in the following section, in which each machine i performs local computation involving the matrix A_i, and the machines then exchange messages so to arrive at an estimate $\widehat{r}(A) \in [n] := \{0, \ldots, n\}$. Our goal is to obtain an estimate that is close to the rank of the matrix in the sense that

$$(1-\delta)\mathrm{rank}(A, c_1) \leq \widehat{r}(A) \leq (1+\delta)\mathrm{rank}(A, c_2), \quad (3)$$

where $c_1 > c_2 \geq 0$ and $\delta \in [0, 1)$ are user-specified constants. The parameter $\delta \in [0, 1)$ upper bounds the relative error of the approximation. The purpose of assuming different thresholds c_1 and c_2 in bound (3) is to handle the ambiguous case when the matrix A has many eigenvalues smaller but very close to c_1. If we were to set $c_1 = c_2$, then any estimator $\widehat{r}(A)$ would be strictly prohibited to take these eigenvalues into account. However, since these eigenvalues are so close to the threshold, distinguishing them from other eigenvalues just above the threshold is obviously difficult (but for an uninteresting reason). Setting $c_1 > c_2$ allows us to expose the more fundamental sources of difficulty in the problem of estimating generalized matrix ranks.

2.2. Basics of communication complexity

To orient the reader, here we provide some very basic background on communication complexity theory; see the books (Lee & Shraibman, 2009; Kushilevitz & Nisan, 1997) for more details. The standard set-up in multiparty communication complexity is as follows: suppose that there are m players (equivalently, agents, machines, etc.), and for $i \in \{1, \ldots, m\}$, player i holds an input string x_i. In the standard form of communication complexity, the goal is to compute a joint function $F(x_1, \ldots, x_m)$ of all m input strings with as little communication between machines as possible. In this paper, we analyze a communication scheme known as the *public blackboard model*, in which each player can write messages on a common blackboard to be read by all other players. A distributed protocol Π consists of a coordinated order in which players write messages on the blackboard. Each message is constructed from the player's local input and the earlier messages on the blackboard. At the end of the protocol, some player outputs the value of $F(x_1, \ldots, x_m)$ based on the information she collects through the process. The communication cost of a given protocol Π, which we denote by $\mathcal{C}(\Pi)$, is the maximum number of bits written on the blackboard given an arbitrary input.

In a *deterministic protocol*, all messages must be deterministic functions of the local input and previous messages. The deterministic communication complexity computing function F, which we denote by $\mathcal{D}(F)$, is defined by

$$\mathcal{D}(F) := \min\Big\{\mathcal{C}(\Pi) : \Pi \text{ is a deterministic protocol that correctly computes } F\Big\}. \quad (4)$$

In other words, the quantity $\mathcal{D}(F)$ is the communication cost of the most efficient deterministic protocol.

A broader class of protocols are those that allow some form of randomization. In the public randomness model, each player has access to an infinite-length random string, and their messages are constructed from the local input, the earlier messages and the random string. Let $\mathcal{P}_\epsilon(F)$ be the set of randomized protocols that correctly compute the function F on any input with probability at least $1-\epsilon$. The *randomized communication complexity* of computing function F with failure probability ϵ is given by

$$\mathcal{R}_\epsilon(F) := \min\Big\{\mathcal{C}(\Pi) \mid \Pi \in \mathcal{P}_\epsilon(F)\Big\}. \quad (5)$$

In the current paper, we adopt the bulk of the framework of communication complexity, but with one minor twist in how we define "correctness" in computing the function. For our problem, each machine is a player, and the i^{th} player holds the matrix A_i. Our function of interest is given by $F(A_1, \ldots, A_m) = \mathrm{rank}(\sum_{i=1}^m A_i)$. The public blackboard setting corresponds to a broadcast-free model, in which each machine can send messages to a master node, then the master node broadcasts the messages to all other machines without additional communication cost.

Let us now clarify the notion of "correctness" used in this paper. In the standard communication model, a protocol Π is said to correctly compute the function F if the output of the protocol is exactly equal to $F(A_1, \ldots, A_m)$. In this paper, we allow approximation errors in the computation, as specified by the parameters (c_1, c_2), which loosen the matrix rank to the generalized matrix ranks, and the tolerance parameter $\delta \in (0, 1)$. More specifically, we say:

Definition 1. *A protocol* Π *correctly computes the rank of the matrix* A *up to tolerances* (c_1, c_2, δ) *if the output* $\widehat{r}(A)$ *satisfies inequality* (3).

Given this definition of correctness, we denote the deterministic communication complexity of the rank estimation problem by $\mathcal{D}(c_1, c_2, \delta)$, and the corresponding randomized communication complexity by $\mathcal{R}_\epsilon(c_1, c_2, \delta)$. The goal of this paper is to study these two quantities, especially their dependence on the dimension n of matrices.

In addition to allowing for approximation error, our analysis—in contrast to most classical communication complexity—allows the input matrices $\{A_i\}_{i=1}^m$ to take real values. However, doing so does not make the problem substantially harder. Indeed, in order to approximate the matrices in elementwise ℓ_∞-norm up to τ rounding error, it suffices to discretize each matrix entry using $\mathcal{O}(\log(1/\tau))$ bits. As we discuss in more detail in the sequel, this type of discretization has little effect on the communication complexity.

3. Communication complexity of deterministic algorithms

We begin by studying the communication complexity of deterministic algorithms. Here our main result shows that the trivial algorithm—the one in which each machine transmits essentially its whole matrix—is optimal up to logarithmic factors. In the statement of the theorem, we assume that the n-dimensional matrix A is known to have rank in the interval[1] $[r, 2r]$ for some integer $r \leq n/4$.

Theorem 1. *For matrices A with rank in $[r, 2r]$:*

(a) For all $0 \leq c_2 < c_1$ and $\delta \in (0, 1)$, we have $\mathcal{D}(c_1, c_2, \delta) = \mathcal{O}\Big(mrn \log\big(\frac{mrn}{c_1 - c_2}\big)\Big)$.

(b) For two machines $m = 2$, constants $0 \leq c_2 < c_1 < 1/20$ and $\delta \in (0, 1/12)$, we have $\mathcal{D}(c_1, c_2, \delta) = \Omega(rn)$.

When the matrix A has rank r that grows proportionally with its dimension n, the lower bound in part (b) shows that deterministic communication complexity is surprisingly large: it scales as $\Theta(n^2)$, which is as large as transmitting the full matrices. Up to logarithmic factors, this scaling is matched by the upper bound in part (a). It is proved by analyzing an essentially trivial algorithm: for each index $i = 2, \ldots, m$, machine i encodes a reduced rank representation of the matrix A_i, representing each matrix entry by $\log_2\left(\frac{12mrn}{c_1 - c_2}\right)$ bits. It sends this quantized matrix $\widetilde{A}_i$ to the first machine. Given these received messages, the first machine then computes the matrix sum $\widetilde{A} := A_1 + \sum_{i=2}^m \widetilde{A}_i$, and it outputs $\widehat{r}(A)$ to be the largest integer k such that $\sigma_k(\widetilde{A}) > (c_1 + c_2)/2$.

Next, we provide a proof sketch for the lower bound. In order to prove the lower bound, we consider a two-party rank testing problem. Consider two agents holding matrices A_1 and A_2, respectively, such that the matrix sum $A := A_1 + A_2$ has operator norm at most one. Suppose that exactly one of the two following conditions are known to hold:

[1]We use an interval assumption, as the problem becomes trivial if the rank is fixed exactly.

- the matrix A has rank r, or
- the matrix A has rank between $\frac{6r}{5}$ and $2r$, and in addition its $(6r/5)^{th}$ eigenvalue is lower bounded as $\sigma_{\frac{6r}{5}}(A) > \frac{1}{20}$.

The goal is to decide which case is true by exchanging the minimal number of bits between the two agents. Denoting this problem by `RankTest`, the proof of part (a) proceeds by showing that $D(\texttt{RankTest}) = \Omega(rn)$, and then reducing from the `RankTest` problem to the matrix rank estimation problem.

To show that $D(\texttt{RankTest}) = \Omega(rn)$, we construct a set S of $n \times n$ matrices. We prove that there exist a construction such that $|S| = 2^{\Omega(rn)}$, and for any two matrices $A_1, A_2 \in S$, the sum matrix $A := A_1 + A_2$ satisfies the following property:

- If $A_1 = A_2$, then the matrix A has rank r.
- If $A_1 \neq A_2$, then the matrix A has rank between $\frac{6r}{5}$ and $2r$, and in addition its $(6r/5)^{th}$ eigenvalue is lower bounded as $\sigma_{\frac{6r}{5}}(A) > \frac{1}{20}$.

It is well known that in the two-party communication model, determining the equality of two element in a set of cardinality $2^{\Omega(rn)}$ requires at least $\Omega(rn)$ bits of communication. On the other hand, our construction reduces the equality determination problem to the `RankTest` problem, thus establishes the $\Omega(rn)$ lower bound for `RankTest`. The rigorous proof of Theorem 1 is included in the long version of this paper (Zhang et al., 2015).

It is worth noting that we cannot use the same argument to prove an $\Omega(rn)$ lower bound for randomized algorithms. Although the reduction argument also works for randomized algorithms, the randomized communication complexity of determining the equality of two element in a set of cardinality $2^{\Omega(rn)}$ is no longer $\Omega(rn)$. Instead, the randomized communication complexity of that problem is $\Theta(\log(rn))$. This fact suggests that the randomized algorithm might be substantially more efficient than deterministic algorithms. In the next section, we confirm this intuition by presenting a communication-efficient randomized algorithm for estimating the matrix rank, and provide a tight lower bound for randomized algorithms.

4. Communication complexity of randomized algorithms

We now turn to the study of randomized algorithms, for which we see that the communication complexity is substantially lower. In Section 4.1, we propose a randomized algorithm with $\widetilde{\mathcal{O}}(n)$ communication cost, and in Sec-

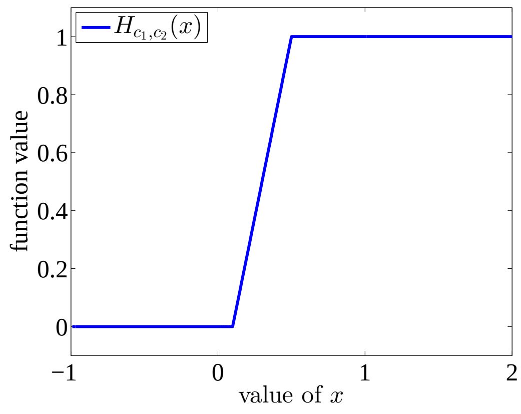


Figure 1. An illustration of the function $x \mapsto H_{c_1,c_2}(x)$ with $c_1 = 0.5$ and $c_2 = 0.1$.

tion 4.3, we establish a lower bound that matches this upper bound in various regimes.

4.1. Upper bound via a practical algorithm

In this section, we present an algorithm based on uniform polynomial approximations for estimating the generalized matrix rank. Let us first provide some intuition for the algorithm before defining it more precisely. For a fixed pair of scalars $c_1 > c_2 \geq 0$, consider the function $H_{c_1,c_2} : \mathbb{R} \to [0, 1]$ given by

$$H_{c_1,c_2}(x) := \begin{cases} 1 & \text{if } x > c_1 \\ 0 & \text{if } x < c_2 \\ \frac{x-c_2}{c_1-c_2} & \text{otherwise.} \end{cases} \tag{6}$$

As illustrated in Figure 1, it is a piecewise linear approximation to a step function. The squared function $H^2_{c_1,c_2}$ is useful in that it can be used to sandwich the generalized ranks of a matrix A. In particular, given a positive semi-definite matrix A with ordered eigenvalues $\sigma_1(A) \geq \sigma_2(A) \geq \ldots \geq \sigma_n(A) \geq 0$, observe that we have

$$\text{rank}(A, c_1) \leq \sum_{i=1}^{n} H^2_{c_1,c_2}(\sigma_i(A)) \leq \text{rank}(A, c_2). \tag{7}$$

Our algorithm exploits this sandwich relation in estimating the generalized rank.

In particular, suppose that we can find a polynomial function $f : \mathbb{R} \to \mathbb{R}$ such that $f \approx H_{c_1,c_2}$, and which is extended to a function on the cone of PSD matrices in the standard way. Observe that if σ is an eigenvalue of A, then the spectral mapping theorem (Bhatia, 1997) ensures that $f(\sigma)$ is an eigenvalue of $f(A)$. Consequently, letting $g \sim N(0, I_{n\times n})$ be a standard Gaussian vector, we have the useful relation

$$\mathbb{E}\Big[\|f(A)g\|_2^2\Big] = \sum_{i=1}^{n} f^2(\sigma_i(A)) \approx \sum_{i=1}^{n} H^2_{c_1,c_2}(\sigma_i(A)). \tag{8}$$

Combined with the sandwich relation (7), we see that a polynomial approximation f to the function H_{c_1,c_2} can be used to estimate the generalized rank.

If f is a polynomial function of degree p, then the vector $f(A)g$ can be computed through p rounds of communication. In more detail, in one round of communication, we can first compute the matrix-vector product $Ag = \sum_{i=1}^{m} A_i g$. Given the vector Ag, a second round of communication suffices to compute the quantity $A^2 g$. Iterating a total of p times, the first machine is equipped with the collection of vectors $\{g, Ag, A^2 g, \ldots, A^p g\}$, from which it can compute $f(A)g$.

Let us now consider how to obtain a suitable polynomial approximation of the function H_{c_1,c_2}. The most natural choice is a Chebyshev polynomial approximation of the first kind: more precisely, since H_{c_1,c_2} is a continuous function with bounded variation, classical theory (Mason & Handscomb, 2010, Theorem 5.7) guarantees that the Chebyshev expansion converges uniformly to H_{c_1,c_2} over the interval $[0, 1]$. Consequently, we may assume that there is a finite-degree Chebyshev polynomial q_1 of the first kind such that

$$\sup_{x\in[0,1]} |q_1(x) - H_{c_1,c_2}(x)| \leq 0.1. \tag{9}$$

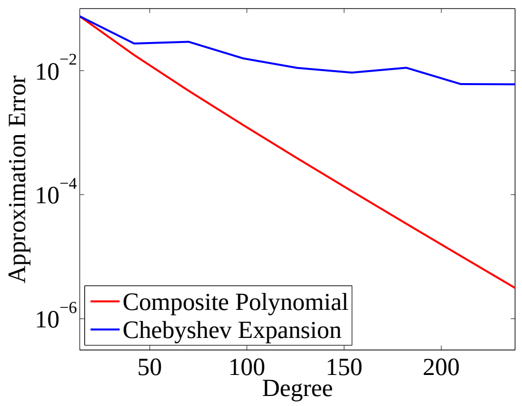
By increasing the degree of the Chebyshev polynomial, we could reduce the approximation error (set to 0.1 in the expansion (9)) to an arbitrarily small level. However, a very high degree could be necessary to obtain an arbitrary accuracy. Instead, our strategy is to start with the Chebyshev polynomial q_1 that guarantees the 0.1-approximation error (9), and then construct a second polynomial q_2 such that the composite polynomial function $f = q_2 \circ q_1$ has an approximation error, when measured over the intervals $[0, c_2]$ and $[c_1, 1]$ of interest, that converges linearly in the degree of function f. More precisely, consider the polynomial of degree $2p + 1$ given by

$$q_2(x) = \frac{1}{B(p + 1, p + 1)} \int_0^x t^p (1 - t)^p dt \tag{10}$$

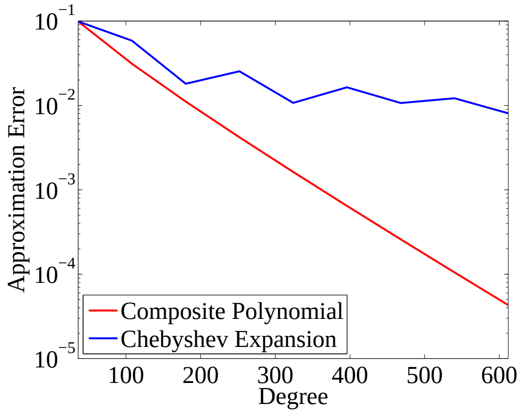
where $B(\cdot, \cdot)$ is the Beta function.

Lemma 1. *Consider the composite polynomial $f(x) := q_2(q_1(x))$, where the base polynomials q_1 and q_2 were previously defined in equations (9) and (10) respectively. Then $f(x) \in [0, 1]$ for all $x \in [0, 1]$, and moreover*

$$|f(x) - H_{c_1,c_2}(x)| \leq 2^{-p} \text{ for } x \in [0, c_2] \cup [c_1, 1]. \tag{11}$$



(a) Thresholds $(c_1, c_2) = (0.2, 0.1)$



(b) Thresholds $(c_1, c_2) = (0.02, 0.01)$

Figure 2. Comparison of the composite polynomial approximation in Algorithm 2 with the Chebyshev polynomial expansion. The error is measured with the ℓ_∞-norm on the interval $[0, c_2] \cup [c_1, 1]$. The composite polynomial approximation achieves a linear convergence rate as the degree is increased, while the Chebyshev expansion converges at a much slower rate.

See Zhang et al. (2015) for the proof.

Figure 2 provides a comparison of the error in approximating H_{c_1,c_2} for the standard Chebyshev polynomial and the composite polynomial. In order to conduct a fair comparison, we show the approximations obtained by Chebyshev and composite polynomials of the same final degree, and we evaluate the ℓ_∞-norm approximation error on interval $[0, c_2] \cup [c_1, 1]$—namely, for a given polynomial approximation h, the quantity

$$\text{Error}(h) := \sup_{x\in[0,c_2]\cup[c_1,1]} |h(x) - H_{c_1,c_2}(x)|.$$

As shown in Figure 2 shows, the composite polynomial function achieves a linear convergence rate with respect to its degree. In contrast, the convergence rate of the Chebyshev expansion is sub-linear, and substantially slower than that of the composite function. The comparison highlights the advantage of our approach over the method only based on Chebyshev expansions.

Given the composite polynomial $f = q_2 \circ q_1$, we first evaluate the vector $f(A)g$ in a two-stage procedure. In the first stage, we evaluate $q_1(A)g$, $q_1^2(A)g$, $\ldots$, $q_1^{2p+1}(A)g$ using the Clenshaw recurrence (Clenshaw, 1955), a procedure proven to be numerically stable (Mason & Handscomb, 2010). The details are given in Algorithm 1. In the second stage, we substitute the coefficients of q_2 so as to evaluate $q_2(q_1(A))b$. The overall procedure is summarized in Algorithm 2.

The following result provides a theoretical guarantee for the overall procedure (combination of Algorithm 1 and Algorithm 2). Roughly speaking, if the degree p is chosen by $p = \lceil \log_2(2n) \rceil$, the function f will be sufficiently close to the function H_{c_1,c_2}. As a consequence, inequality (7) and equation (8) imply that $\|f(A)g\|_2^2$ provides a nearly unbiased estimator to the generalized matrix rank.

Algorithm 1 Evaluation of Chebyshev Polynomial

```
Input: m machines hold $A_1, A_2, \ldots, A_m \in \mathbb{R}^{n\times n}$; vector $v \in \mathbb{R}^d$; Chebyshev polynomial expansion $q(x) = \frac{1}{2}a_0T_0(x) + \sum_{i=1}^d a_iT_i(x)$.
Output: matrix-vector product $q(A)v$.
  1. Initialize vector $b_{d+1} = b_{d+2} = \mathbf{0} \in \mathbb{R}^n$.
  2. For $j = d, \ldots, 1, 0$: the first machine broadcasts $b_{j+1}$ to all other machines. Machine $i$ computes $A_ib_{j+1}$ and sends it back to the first machine. The first machine computes
        $b_j := \left(4\sum_{i=1}^m A_ib_{j+1}\right) - 2b_{j+1} - b_{j+2} + a_jv.$
  3. Output $\frac{1}{2}(a_0v + b_1 - b_3)$;
```

Theorem 2. *For any $0 \le \delta < 1$, with probability at least $1 - 2\exp\left(-\frac{T\delta^2 \text{rank}(A,c_1)}{32}\right)$, the output of Algorithm 2 satisfies the bounds*

$$(1-\delta)\text{rank}(A, c_1) - 1 \le \widehat{r}(A) \le (1+\delta)(\text{rank}(A, c_2) + 1). \quad (12)$$

Moreover, we have the following upper bound on the randomized communication complexity of estimating the generalized matrix rank.

$$\mathcal{R}_\epsilon\left(c_1, c_2, 1/\sqrt{\text{rank}(A, c_1)}\right) = \widetilde{\mathcal{O}}(mn). \quad (13)$$

Algorithm 2 Randomized Algorithm for Rank Estimation

```
Input: $m$ machines hold matrices $A_1, A_2, \ldots, A_m \in \mathbb{R}^{n\times n}$. Tolerance parameters $(c_1, c_2)$, polynomial degree $p$, and number of repetitions $T$.
Output: rank estimate $\widehat{r}(A)$.
    1. [(a)]
        Find a Chebyshev expansion $q_1$ of the function $H_{c_1,c_2}$ satisfying the uniform bound (9).
       (b) Define the degree $2p+1$ polynomial function $q_2$ by equation (10).
    2. [(a)]
        Generate random Gaussian vector $g \sim N(0, I_{n\times n})$.
       (b) Apply Algorithm 1 to compute $q_1(A)g$, and sequentially apply the same algorithm to compute $q_1^2(A)g, \ldots, q_1^{2p+1}(A)g$.
       (c) Evaluate the vector $y := f(A)g = q_2(q_1(A))g$ on the first machine.
    3. Repeat Step 2 for $T$ times, obtaining a collection of $n$-vectors $\{y_1, \ldots, y_T\}$, and output the estimate $\widehat{r}(A) = \frac{1}{T}\sum_{i=1}^T \|y_i\|_2^2$.
```

The rigorous proof of Theorem 2 is included in the long version of this paper (Zhang et al., 2015).

We show in Section 4.3 that the upper bound (13) is unimprovable up to the logarithmic pre-factors. For now, let us turn to the results of some numerical experiments using Algorithm 2, which show that in addition to being an order-optimal algorithm, it is also practically useful.

4.2. Numerical experiments

Given $m = 2$ machines, suppose that machine i (for $i = 1, 2$) receives $N_i = 1000$ data points of dimension $n = 1000$. Each data point x is independently generated as $x = a+\varepsilon$, where $a \sim N(0, \lambda\Sigma)$ and $\varepsilon \sim N(0, \sigma^2 I_{n\times n})$ are random Gaussian vectors. Here $\Sigma \in \mathbb{R}^{n\times n}$ is a low-rank covariance matrix of the form $\Sigma := \sum_{i=1}^r u_i u_i^T$, where $\{u_i\}_{i=1}^r$ are an orthonormal set of vectors in $\mathbb{R}^n$ drawn uniformly at random. The goal is to estimate the rank r from the observed $N_1 + N_2 = 2000$ data points.

Let us now describe how to estimate the rank using the covariance matrix of the samples. Notice that $\mathbb{E}[xx^T] = \lambda^2\Sigma + \sigma^2 I_{n\times n}$, of which there are r eigenvalues equal to $\lambda + \sigma^2$ and the remaining eigenvalues are equal to σ^2. Letting $x_{i,j} \in \mathbb{R}^n$ denote the j-th data point received by machine i, that machine can compute the local sample covariance matrix

$$A_i = \frac{1}{N_1 + N_2}\sum_{j=1}^{N_i} x_{i,j}x_{i,j}^T, \quad \text{for } i = 1, 2.$$

The full sample covariance matrix is given by the sum $A := A_1 + A_2$, and its rank can be estimated using Algorithm 2.

In order to generate the data, we choose the parameters $r = 100$, $\lambda = 0.4$ and $\sigma^2 = 0.1$. These choices motivate the thresholds $c_1 = \lambda + \sigma^2 = 0.5$ and $c_2 = \sigma^2 = 0.1$ in Algorithm 2. We illustrate the behavior of the algorithm for three different choices of the degree parameter p—specifically, $p \in \{0, 1, 5\}$—and for a range of repetitions $T \in \{1, 2, \ldots, 30\}$. Letting $\widehat{r}(A)$ denote the output of the algorithm, we evaluate the mean squared error, $\mathbb{E}[(\widehat{r}(A) - r)^2]$, based on 100 independent runs of the algorithm.

We plot the results of this experiment in Figure 3. Panel (a) shows the distribution of eigenvalues of the matrix A. In this plot, there is a gap between the large eigenvalues generated by the low-rank covariance matrix Σ, and small eigenvalues generated by the random Gaussian noise, showing that the problem is relatively easy to solve in the centralized setting. Panel (b) shows the estimation error achieved by the communication-efficient distributed algorithm; notice how the estimation error stabilizes after $T = 30$ repetitions or iterations. We compare our algorithm for $p \in \{0, 1, 5\}$, corresponding to polynomial approximations with degree in $\{4, 12, 44\}$. For the case $p = 0$, the polynomial approximation is implemented by the Chebyshev expansion. For the case $p = 1$ and $p = 5$, the approximation is achieved by the composite function f. As a baseline method, we also implement Napoli et al.'s algorithm (2013) in the distributed setting. In particular, their method replaces the function f in Algorithm 2 by a Chebyshev expansion of the high-pass filter $\mathbb{I}(x \geq \frac{c_1+c_2}{2})$. It is observed that both the Chebyshev expansion with $p = 0$ and the baseline method incur a large bias in the rank estimate, while the composite function's estimation errors are substantially smaller. After $T = 30$ iterations, Algorithm 2 with $p = 1$ achieves a mean squared error close to 10, which means that the relative error of the estimation is around 3%.

4.3. Lower bound

It is natural to wonder if the communication efficiency of Algorithm 2 is optimal. The following theorem shows that, in order to achieve the same $1/\sqrt{r}$ relative error, it is necessary to send $\Omega(n)$ bits. As in our upper bound, we assume that the matrix A satisfies the spectral norm bound $\|A\|_2 \leq 1$. Given an arbirary integer r in the interval $[16, n/4]$, suppose that the generalized matrix ranks satisfy the sandwich relation $r \leq \operatorname{rank}(A, c_1) \leq \operatorname{rank}(A, c_2) \leq 2r$. Under these conditions, we have the following guarantee:

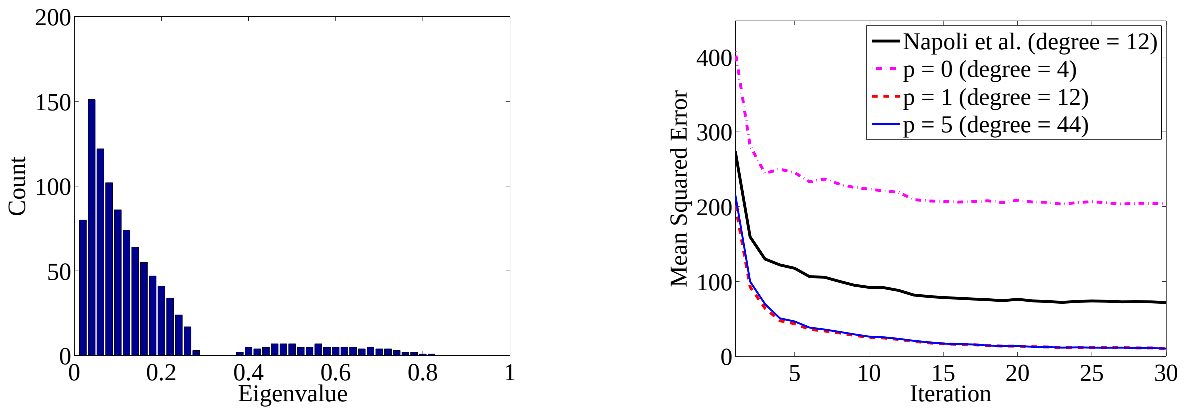


Figure 3. Panel (a): distribution of eigenvalues of matrix A. Panel (b): mean squared error of rank estimation versus the number of iterations for the baseline method by Napoli et al. (Di Napoli et al., 2013), and three versions of Algorithm 2 (with parameters $p \in \{0, 1, 5\}$).

Theorem 3. *For any c_1, c_2 satisfying $c_1 < 2c_2 \leq 1$ and any $\epsilon \leq \epsilon_0$ for some numerical constant ϵ_0, we have*

$$\mathcal{R}_\epsilon\Big(c_1, c_2, 1/\sqrt{r}\Big) = \Omega(n). \tag{14}$$

We prove Theorem 3 by reducing the 2-SUM problem (Woodruff & Zhang, 2014) to the generalized rank estimation problem. The former problem is known to have $\Theta(n)$ randomized communication complexity. See Zhang et al. (2015) for the rigorous proof of Theorem 3.

According to Theorem 3, for matrices with true rank in the interval $[16, n/2]$, the communication complexity for estimating the rank with relative error $1/\sqrt{r}$ is lower bounded by $\Omega(n)$. This lower bound matches the upper bound provided by Theorem 2. In particular, choosing $r = 16$ yields the worst-case lower bound

$$\mathcal{R}_\epsilon(c_1, c_2, 1/4) = \Omega(n),$$

showing that $\Omega(n)$ bits of communication are necessary for achieving a constant relative error. This lower bound is not trivial relative to the coding length of the correct answer: given that the matrix rank is known to be between r and $2r$, this coding length scales only as $\Omega(\log r)$.

There are several open problems suggested by the result of Theorem 3. First, it would be interesting to strengthen the lower bound (14) from $\Omega(n)$ to $\Omega(mn)$, incorporating the natural scaling with the number of machines m. Doing so requires a deeper investigation into the multi-party structure of the problem. Another open problem is to lower bound the communication complexity for arbitrary values of the tolerance parameter δ, say as small as $\delta = 0$. When $\delta = 0$, communicating $\mathcal{O}(mn^2)$ bits is an obvious upper bound, and we are not currently aware of better upper bounds. On the other hand, whether it is possible to prove an $\Omega(n^2)$ lower bound for small δ remains an open question.

5. Conclusions

In this paper, we have studied the problem of estimating the generalized rank of matrices. Our main results are to show that in the deterministic setting, sending $\Theta(n^2)$ bits is both necessary and sufficient in order to obtain any constant relative error. In contrast, when randomized algorithms are allowed, this scaling is reduced to $\widetilde{\Theta}(n)$.

There are many interesting open problems in the study of distributed computing for linear algebraic functions. In Section 4.3, we have proposed an open question asking if there is a communication-efficient algorithm which estimates the matrix rank with tolerance parameter $\delta = 0$. In the long version of this paper (Zhang et al., 2015), we have shown that if this question could be answered, then we will be able to tightly bound the communication complexity of a broad class of important problems, including the matrix singularity test problem, the problem of solving linear equations and the problem of convex optimization. We view the conclusion of this paper as a step towards exploring this research direction.

References

Abelson, H. Lower bounds on information transfer in distributed computations. *Journal of the ACM*, 27(2):384–392, 1980.

Bhatia, R. *Matrix Analysis*. Graduate Texts in Mathematics. Springer-Verlag, New York, NY, 1997.

Candès, E. J., Li, X., Ma, Y., and Wright, J. Robust principal component analysis? *Journal of the ACM*, 58(3):11, 2011.

Chu, J. I. and Schnitger, G. The communication complexity of several problems in matrix computation. *Journal of Complexity*, 7(4):395–407, 1991.

Chu, J. I. and Schnitger, G. Communication complexity of matrix computation over finite fields. *Mathematical Systems Theory*, 28(3):215–228, 1995.

Clarkson, K. L. and Woodruff, D. P. Numerical linear algebra in the streaming model. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, pp. 205–214. ACM, 2009.

Clenshaw, C. A note on the summation of Chebyshev series. *Mathematics of Computation*, 9(51):118–120, 1955.

Di Napoli, E., Polizzi, E., and Saad, Y. Efficient estimation of eigenvalue counts in an interval. *arXiv:1308.4275*, 2013.

Eckart, C. and Young, G. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Golub, G. H. and Van Loan, C. F. *Matrix Computations*, volume 3. JHU Press, 2012.

Halko, N., Martinsson, P.-G., and Tropp, J. A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

Jolliffe, I. *Principal Component Analysis*. Wiley Online Library, 2005.

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

Kushilevitz, E. and Nisan, N. *Communication Complexity*. Cambridge University Press, 1997.

Lee, T. and Shraibman, A. *Lower Bounds in Communication Complexity*. Now Publishers, 2009.

Li, Y., Sun, X., Wang, C., and Woodruff, D. P. On the communication complexity of linear algebraic problems in the message passing model. In *Distributed Computing*, pp. 499–513. Springer, 2014.

Luo, Z.-Q. and Tsitsiklis, J. N. On the communication complexity of distributed algebraic computation. *Journal of the ACM*, 40(5):1019–1047, 1993.

Mahoney, M. W. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3 (2):123–224, 2011.

Mason, J. C. and Handscomb, D. C. *Chebyshev Polynomials*. CRC Press, 2010.

Polizzi, E. Density-matrix-based algorithm for solving eigenvalue problems. *Physical Review B*, 79(11): 115112, 2009.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 452–461. AUAI Press, 2009.

Sakurai, T. and Sugiura, H. A projection method for generalized eigenvalue problems using numerical integration. *Journal of Computational and Applied Mathematics*, 159(1):119–128, 2003.

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on the World Wide Web*, pp. 285–295. ACM, 2001.

Schofield, G., Chelikowsky, J. R., and Saad, Y. A spectrum slicing method for the Kohn–Sham problem. *Computer Physics Communications*, 183(3):497–505, 2012.

Sun, X. and Wang, C. Randomized communication complexity for linear algebra problems over finite fields. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 14. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2012.

Woodruff, D. P. and Zhang, Q. An optimal lower bound for distinct elements in the message passing model. In *Symposium on Discrete Algorithms*, pp. 718–733. SIAM, 2014.

Yao, A. C.-C. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, pp. 209–213. ACM, 1979.

Zhang, Y., Wainwright, M. J., and Jordan, M. I. Distributed estimation of generalized matrix rank: Efficient algorithms and lower bounds. *arXiv preprint arXiv:1502.01403*, 2015.