SUBSPACE ITERATION RANDOMIZATION AND SINGULAR VALUE PROBLEMS

M. GU*

Abstract. A classical problem in matrix computations is the efficient and reliable approximation of a given matrix by a matrix of lower rank. The truncated singular value decomposition (SVD) is known to provide the best such approximation for any given fixed rank. However, the SVD is also known to be very costly to compute. Among the different approaches in the literature for computing low-rank approximations, randomized algorithms have attracted researchers' recent attention due to their surprising reliability and computational efficiency in different application areas. Typically, such algorithms are shown to compute with very high probability low-rank approximations that are within a constant factor from optimal, and are known to perform even better in many practical situations. In this paper, we present a novel error analysis that considers randomized algorithms within the subspace iteration framework and show with very high probability that highly accurate low-rank approximations as well as singular values can indeed be computed quickly for matrices with rapidly decaying singular values. Such matrices appear frequently in diverse application areas such as data analysis, fast structured matrix computations and fast direct methods for large sparse linear systems of equations and are the driving motivation for randomized methods. Furthermore, we show that the low-rank approximations computed by these randomized algorithms are actually rank-revealing approximations, and the special case of a rank-1 approximation can also be used to correctly estimate matrix 2-norms with very high probability. Our numerical experiments are in full support of our conclusions.

key words: low-rank approximation, randomized algorithms, singular values, standard Gaussian matrix.

1. Introduction. Randomized algorithms have established themselves as some of the most competitive methods for rapid low-rank matrix approximation, which is vital in many areas of scientific computing, including principal component analysis [48, 66] and face recognition [61, 79], large scale data compression [21, 22, 36, 57] and fast approximate algorithms for PDEs and integral equations [16, 34, 58, 72, 73, 84, 83]. In this paper, we consider randomized algorithms for low-rank approximations and singular value approximations within the subspace iteration framework, leading to results that simultaneously retain the reliability of randomized algorithms and the typical faster convergence of subspace iteration methods.

Given any $m \times n$ matrix A with $m \geq n$, its singular value decomposition (SVD) is described by the equation

$$A = U\Sigma V^T, \tag{1.1}$$

where U is an $m \times n$ column orthogonal matrix; V is an $n \times n$ orthogonal matrix; and $\Sigma = \mathrm{diag}(\sigma_1, \cdots, \sigma_n)$ with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$. Writing U and V in terms of their columns,

$$U = (u_1, \cdots, u_n) \quad \text{and} \quad V = (v_1, \cdots, v_n),$$

then u_j and v_j are the left and right singular vectors corresponding to σ_j, the j-th largest singular value of A. For any $1 \leq k \leq n$, we let

$$A_k = (u_1, \cdots, u_k)\, \mathrm{diag}(\sigma_1, \cdots, \sigma_k)\, (v_1, \cdots, v_k)^T$$

be the (rank-k) truncated SVD of A. The matrix A_k is unique only if $\sigma_{k+1} < \sigma_k$. The assumption that $m \geq n > \max k, 2$ will be maintained throughout this paper for ease of exposition. Our results still hold for $m < n$ by applying all the algorithms on A^T. Similarly, all our main results are derived under the assumption that $\mathbf{rank}(A) = n$. But they remain unchanged even if $\mathbf{rank}(A) < n$, and hence remain valid by a continuity argument. All our analysis is done without consideration of round-off errors, with the implicit assumption that the user tolerances for the low-rank approximation are always set to be above machine precision levels. Additionally, we assume throughout this paper that all matrices are real. In general, A_k is an ideal rank-k approximation to A, due to the following celebrated property of the SVD:

*This research was supported in part by NSF Awards CCF-0830764 and CCF-1319312, and by the DOE Office of Advanced Scientific Computing Research under contract number DE-AC02-05CH11231. Email: mgu@math.berkeley.edu.

THEOREM 1.1. *(Eckart and Young [24], Golub and van Loan [31])*

$$\min_{\mathbf{rank}(B)\leq k} \|A-B\|_2 = \|A-A_k\|_2 = \sigma_{k+1}. \tag{1.2}$$

$$\min_{\mathbf{rank}(B)\leq k} \|A-B\|_F = \|A-A_k\|_F = \sqrt{\sum_{j=k+1}^{n} \sigma_j^2}. \tag{1.3}$$

Theorem 1.1 states that the truncated SVD provides a rank-k approximation to A with the smallest possible 2-norm error and Frobenius-norm error. In the 2-norm, any rank-k approximation will result in an error no less than σ_{k+1}, and in the Frobenius-norm, any rank-k approximation will result in an error no less than $\sqrt{\sum_{j=k+1}^{n} \sigma_j^2}$. Additionally, the singular values of A_k are exactly the first k singular values of A, and the singular vectors of A_k are the corresponding singular vectors of A. Note, however, that while the solution to problem (1.3) must be A_k, solutions to problem (1.2) are not unique and include, for example, the rank-k matrix B defined below for any $0 \leq \theta \leq 1$:

$$B = A_k - \theta\sigma_{k+1}\left(u_1, \cdots, u_k\right)\left(v_1, \cdots, v_k\right)^T. \tag{1.4}$$

This subtle distinction between the 2-norm and Frobenius norm will later on become very important in our analysis of randomized algorithms (see Remark 3.4.) In Theorem 3.4 we prove an interesting result related to Theorem 1.1 for rank-k approximations that only solve problems (1.2) and (1.3) approximately.

To compute a truncated SVD of a general $m \times n$ matrix A, one of the most straightforward techniques is to compute the full SVD and truncate it, with a standard linear algebra software package like the LAPACK [1]. This procedure is stable and accurate, but it requires $O(mn^2)$ floating point operations, or *flops*. This is prohibitively expensive for applications such as data mining, where the matrices involved are typically sparse with huge dimensions. In other practical applications involving the truncated SVD, often the very objective of computing a rank-k approximation is to avoid excessive computation on A. Hence it is desirable to have schemes that can compute a rank-k approximation more efficiently. Depending on the reliability requirements, a good rank-k approximation can be a matrix that is accurate to within a constant factor from the optimal, such as a rank-revealing factorization (more below), or it can be a matrix that closely approximates the truncated SVD itself.

Many approaches have been taken in the literature for computing low-rank approximations, including rank-revealing decompositions based on the QR, LU, or two-sided orthogonal (aka UTV) factorizations [14, 25, 33, 43, 60, 64, 45]. Recently, there has been an explosion of randomized algorithms for computing low-rank approximations [16, 21, 22, 28, 29, 55, 54, 56, 62, 81, 71]. There is also software package available for computing interpolative decompositions, a form of low-rank approximation, and for computing the PCA, with randomized sampling [59]. These algorithms are attractive for two main reasons: they have been shown to be surprisingly efficient computationally; and like subspace methods, the main operations involved in many randomized algorithms can be optimized for peak machine performance on modern architectures. For a detailed analysis of randomized algorithms and an extended reference list, see [36]; for a survey of randomized algorithms in data analysis, see [57].

The subspace iteration is a classical approach for computing singular values. There is extensive convergence analysis on subspace iteration methods [31, 19, 4, 3] and a large literature on accelerated subspace iteration methods [69]. In general, it is well-suited for fast computations on modern computers because its main computations are in terms of matrix-matrix products and QR factorizations that have been highly optimized for maximum efficiency on modern serial and parallel architectures [19, 31]. There are two well-known weaknesses of subspace iteration, however, that limit its practical use. On one hand, subspace iteration typically requires very good separation between the wanted and unwanted singular values for good convergence. On the other hand, good convergence also often critically depends on the choice of a good start matrix [4, 3].

Another classical class of approximation methods for computing an approximate SVD are the Krylov subspace methods, such as the Lanczos algorithm (see, for example [10, 17, 50, 52, 70, 82].) The computational cost of these methods depends heavily on several factors, including the start vector, properties of the input matrix and the need to stabilize the algorithm. One of the most important part of the Krylov subspace methods, however, is the need to do a matrix-vector product at each iteration. In contrast to matrix-matrix products, matrix-vector products perform very poorly on modern architectures due to the limited data reuse involved in such operations, In fact, one focus of Krylov subspace research is on effective avoidance of matrix-vector operations in Krylov subspace methods (see, for example [32, 68].)

This work focuses on the surprisingly strong performance of randomized algorithms in delivering highly accurate low-rank approximations and singular values. As in Algorithm 1.1, the random matrix Ω in Algorithm 2.2 must be a standard Gaussian matrix (see Remark 1.1.) the The distribution of a standard Gaussian matrix is rotationally invariant: If V is a matrix with orthonormal columns, then $V^T\Omega$ has the same standard Gaussian distribution as Ω. This invariance will allow us to access the vast literature on Gaussian matrices while completely ignoring the matrix V. our analysis is made much easier with the Gaussian matrix due to the existence of the vast literature on the probability density functions of its singular values. To illustrate, we introduce Algorithm 1.1, one of the basic randomized algorithms (see [36].)

ALGORITHM 1.1. **Basic Randomized Algorithm**

Input: *$m \times n$ matrix A with $m \geq n$, integers $k > 0$ and $n > \ell > k$.*
Output: *a rank-k approximation.*

---

1. *Draw a random $n \times \ell$ test matrix Ω.*
2. *Compute $Y = A\,\Omega$.*
3. *Compute an orthogonal column basis Q for Y.*
4. *Compute $B = Q^T A$.*
5. *Compute B_k, the rank-k truncated SVD of B.*
6. *Return QB_k.*

REMARK 1.1. Throughout this paper, a random matrix, such as Ω in Algorithm 1.1, is a standard Gaussian matrix, i.e., its entries are independent standard normal variables of zero mean and standard deviation 1.

While other random matrices might work equally well, the choice of the Gaussian matrix provides two unique advantages: First, the distribution of a standard Gaussian matrix is rotationally invariant: If V is an orthonormal matrix, then $V^T\Omega$ is itself a standard Gaussian matrix with the same statistical properties as Ω [36]. Second, our analysis is much simplified by the vast literature on the singular value probability density functions of the Gaussian matrix.

While Algorithm 1.1 looks deceptively simple, its analysis is long, arduous, and involves very strong doses of statistics [36]. The following theorem establishes an error bound on the accuracy of $QQ^T A$ as a low-rank approximation to A. There are similar results in the Frobenius norm.

THEOREM 1.2. *(Halko, Martinsson, Tropp [36, Corollary 10.9]) The column-orthonormal matrix Q produced*

by Step 3 in Algorithm 1.1 satisfies

$$\| \left(I - QQ^T\right) A\|_2 \leq \left(1 + 17\sqrt{1+\frac{k}{p}}\right)\sigma_{k+1} + \frac{8\sqrt{k+p}}{p+1}\sqrt{\sum_{j=k+1}^{n}\sigma_j^2}, \quad \textit{provided that} \quad p = \ell - k \geq 4,$$

with failure probability at most $6e^{-p}$.

REMARK 1.2. Comparing Theorem 1.2 with Theorem 1.1, it is clear that Algorithm 1.1 could provide a very good low rank approximation to A with probability at least $1 - 6e^{-p}$, despite its simple operations, provided that $\sigma_{k+1} \ll \|A\|_2$. While algorithms [16, 21, 22, 28, 29, 55, 54, 81] differ in their algorithm design, efficiency, and domain applicability, they typically share the same advantages of computational efficiency and approximation accuracy.

Algorithm 1.1 is the combination of Stages A and B of the Proto Algorithm in [36], where the truncated SVD is considered separately from low-rank approximation. In Section 2.3 we will discuss the pros and cons of SVD truncation vs. no truncation. Algorithm 1.1 is a special case of the randomized subspace iteration method (see Algorithm 2.2), for which Halko, Martinsson, Tropp [36] have developed similar results.

However, while the upper bound in Theorem 1.2 can be very satisfactory for many applications, there may be situations where singular value approximations are also desirable. In addition, it is well-known that in practical computations randomized algorithms often far outperform their error bounds [36, 59, 67], whereas the results in [36] do not suggest convergence of the computed rank-k approximation to the truncated SVD in either Algorithm 1.1 or the more general randomized subspace iteration method.

Our entire work is based on novel analysis of the subspace iteration method, and we consider randomized algorithms within the subspace iteration framework. This allows us to take advantage of existing theories and technical machinery in both fields.

Current analysis on randomized algorithms focuses on the errors in the approximation of A by a low rank matrix, whereas classical analysis on subspace iteration methods focuses on the accuracy in the approximate singular values. Our analysis allows us to obtain both kinds of results for both of these methods, leading to the stronger rank-revealing approximations. In terms of randomized algorithms, our matrix approximation bounds are in general tighter and can be drastically better than existing ones; in terms of singular values, our relative convergence lower bounds can be interpreted as simultaneously convergence error bounds and rank-revealing lower bounds.

Our analysis has lead us to some interesting conclusions, all with high probability (more precise statements are in Sections 5 through 7):

- The leading k singular values computed by randomized algorithms are at least a good fraction of the true ones, regardless of how the singular values are distributed, and they converge quickly to the true singular values in case of rapid singular value decay. In particular, this result implies that randomized algorithms can also be used as efficient and reliable condition number estimators.
- The above results, together with the fact that randomized algorithms compute low-rank approximations up to a dimension dependent constant factor from optimal, mean that these low-rank approximations are in fact rank-revealing factorizations. In addition, for rapidly decaying singular values, these approximations can be as accurate as a truncated SVD.
- The subspace iteration method in general and the power method in particular is still slowly convergent without over-sampling in the start matrix. We present an alternative choice of the start matrix based on our analysis, and demonstrate its competitiveness.

The rest of this paper is organized as follows: In Section 2 we discuss subspace iteration methods and their

randomized versions in more detail; in Section 3 we list a number of preliminary as well as key results needed for later analysis; in Section 4 we derive deterministic lower bounds on singular values and upper bounds on low-rank approximations; in Section 5 we provide both average case and large deviation bounds on singular values and low-rank approximations; in Section 6 we compare these approximations with other rank-revealing factorizations; in Section 7 we discuss how randomized algorithms can be used as efficient and reliable condition number estimators; in Section 8 we present supporting numerical experimental results; and in Section 9 we draw some conclusions and point out possible directions for future research.

Much of our analysis has its origin in the analysis of subspace iteration [69] and randomized algorithms [36]. It relies both on linear algebra tools as well as statistical analysis to do some of the needed heavy lifting to reach our conclusions. To limit the length of this paper, we have put the more detailed parts of the analysis as well as some additional numerical experimental results in the *Supplemental Material*, which is accessible at SIAM's on-line portal.

2. Algorithms. In this section, we present the main algorithms that are discussed in the rest of this paper. We also discuss subtle differences between our presentation of randomized algorithms and that in [36].

2.1. Basic Algorithms. We start with the classical subspace iteration method for computing the largest few singular values of a given matrix.

ALGORITHM 2.1. **Basic Subspace Iteration**

Input: *$m \times n$ matrix A with $n \leq m$, integers $0 < k \leq \ell < n$, and $n \times \ell$ start matrix Ω.*
Output: *a rank-k approximation.*

---

1. *Compute $Y = \left(AA^T\right)^q A\,\Omega$.*
2. *Compute an orthogonal column basis Q for Y.*
3. *Compute $B = Q^T A$.*
4. *Compute B_k, the rank-k truncated SVD of B.*
5. *Return QB_k.*

Given the availability of Lanczos-type algorithms for the singular value computations, the classical subspace iteration method is not widely used in practice except when $k \ll n$. We present it here for later comparisons with its randomized version. We ignore the vast literature of accelerated subspace iteration methods (see, for example [69]) in this paper since our main goal here is to analyze the convergence behavior of subspace iteration method with and without randomized start matrix Ω.

We have presented Algorithm 2.1 in an over-simplified form above to convey the basic ideas involved. In practice, the computation of Y would be prone to round-off errors. For better numerical accuracy, Algorithm A.1 in the Appendix should be used numerically to compute the Q matrix in Algorithm 2.1. In practical computations, however, Algorithm A.1 is often performed once every few iterations, to balance efficiency and numerical stability (see Saad [69].) In the rest of Section 2, any QR factorization of the matrix $Y = \left(AA^T\right)^q A\Omega$ should be computed numerically through periodic use of Algorithm A.1.

While there is little direct analysis of subspace iteration methods for singular values (Algorithm 2.1) in the literature, one can generalize results of subspace iteration methods for symmetric matrices to the singular value case in a straightforward fashion. The symmetric matrix version of Theorem 2.1 can be found in [4].

THEOREM 2.1. *(Bathe and Wilson) Assume that Algorithm 2.1 converges as $q \to \infty$. Then*

$$|\sigma_j - \sigma_j(Q^T B_k)| \leq O\left(\left(\frac{\sigma_{\ell+1}}{\sigma_k}\right)^{2q+1}\right).$$

Thus convergence is governed by the ratio $\frac{\sigma_{\ell+1}}{\sigma_k}$. The per-iteration cost of Algorithm 2.1 depends linearly on $\ell \geq k$. A choice $\ell > k$ can be economical if the more rapid convergence obtained through the ratio $\frac{\sigma_{\ell+1}}{\sigma_k}$ can more than offset the extra cost per iteration. Another important issue with Algorithm 2.1 is the constant hidden in the O notation. This constant can be exceedingly large for the unfortunate choices of Ω. In fact, an Ω matrix that is almost orthogonal to any leading singular vectors will lead to large number of iterations. Both issues will be made clearer with our relative convergence theory for Algorithm 2.1 in Theorem 4.3.

A special case of Algorithm 2.1 is when $k = \ell = 1$. This is the classical power method for computing the 2-norm of a given matrix. This method, along with its randomized version, is included in Appendix A for later discussion in our numerical experiments (see Section 8.) The power method has the same convergence properties of Algorithm 2.1. More generally, the subspace iteration method is typically run with $k = \ell$.

2.2. Randomized Algorithms. In order to enhance the convergence of Algorithm 2.1 in the absence of any useful information about the leading singular vectors, a sensible approach is to replace the deterministic start matrix with a random one, leading to

ALGORITHM 2.2. **Randomized Subspace Iteration**

Input: *$m \times n$ matrix A with $n \leq m$, integers $0 < k \leq \ell$,*
Output: *a rank-k approximation.*

---

1. *Draw a random $n \times \ell$ start matrix Ω.*
2. *Compute $Y = \left(AA^T\right)^q A\,\Omega$.*
3. *Compute an orthogonal column basis Q for Y.*
4. *Compute $B = Q^T A$.*
5. *Compute B_k, the rank-k truncated SVD of B.*
6. *Return QB_k.*

REMARK 2.1. Since Algorithm 2.2 is the special case of Algorithm 2.1 with Ω being chosen as random, all our results for Algorithm 2.1 equally hold for Algorithm 2.2.

The only difference between Algorithm 2.1 and Algorithm 2.2 is in the choice of Ω, yet this difference will lead to drastically different convergence behavior. One of the main purposes of this paper is to show that the slow or non-convergence of Algorithm 2.1 due to bad choice of Ω vanishes with near certainty in Algorithm 2.2. In particular, a single iteration ($q = 0$ in Algorithm 2.2) in the randomized subspace iteration method is often sufficient to return good enough singular values and low-rank approximations (Section 5).

Our analysis of deterministic and randomized subspace iteration method was in large part motivated by the analysis and discussion of randomized algorithms in [36]. We have chosen to present the algorithms in Section 2 in forms that are not identical to those in [36] for ease of stating our results in Sections 4 through 8.

2.3. To Truncate or not to Truncate. The randomized algorithms in Section 2 are presented in a slight different form than those in [36]. One key difference is in the step of SVD truncation, which is considered an optional postprocessing step there. In this section, we discuss the pros and cons of SVD truncation. We start with the following simple lemma, versions of which appear in [7, 23, 36].

LEMMA 2.2. *Given an $m \times \ell$ matrix with orthonormal columns Q, with $\ell \leq n$, then for any $\ell \times n$ matrix B,*

$$\|A - Q\left(Q^T A\right)\|_2 \leq \|A - QB\|_2 \quad \textit{and} \quad \|A - Q\left(Q^T A\right)\|_F \leq \|A - QB\|_F.$$

Lemma 2.2 makes it obvious that any SVD truncation of $Q^T A$ will only result in a less accurate approximation in the 2-norm and Frobenius norm. This is strong motivation for no SVD truncation. The SVD truncation of $Q^T A$ also involves the computation of the SVD of $Q^T A$ in some form, which also results in extra computation.

On the other hand, some singular values of $Q^T A$ may be poor approximations of those of A, making $QQ^T A$ an inferior rank-ℓ approximation to A, potentially leading to less efficient subsequent computations involving $QQ^T A$. In contrast, for the right choices of k, the rank-k truncated SVD of $Q^T A$ can result in excellent rank-k approximations to A, sometimes almost as good as the rank-k truncated SVD of A itself, with all k singular values of A approximated to very high accuracy. So the choice of whether to truncate the SVD of $Q^T A$ depends on practical considerations of computational efficiency and demands on reliability. This paper focuses on a rank-k approximations obtained from truncated SVD of $Q^T A$.

3. Setup. In this section we build some of the technical machinery needed for our heavy analysis later on. We start by reciting two well-known results in matrix analysis, and then develop a number of theoretical tools that outline our approach in the low-rank approximation analysis. Some of these results may be of interest in their own right. For any matrix X, we use $\sigma_j(X)$ to denote its j-th largest singular value.

The Cauchy interlacing theorem shows the limitations of any approximation with an orthogonal projection.

THEOREM 3.1. *(Golub and van Loan [31, p. 411]) Let A be an $m \times n$ matrix and Q be a matrix with orthonormal columns. Then $\sigma_j(A) \geq \sigma_j(Q^T A)$ for $1 \leq j \leq \min(m, n)$.*

REMARK 3.1. A direct consequence of Theorem 3.1 is that $\sigma_j(A) \geq \sigma_j(\widehat{A})$, where $\widehat{A}$ is any submatrix of A.

Weyl's monotonicity theorem relates singular values of matrices X and Y to those of $X + Y$.

THEOREM 3.2. *(Weyl's monotonicity theorem [44, Thm. 3.3.16]) Let X and Y be $m \times n$ matrices with $m \geq n$. Then*

$$\sigma_{i+j-1}(X + Y) \leq \sigma_i(X) + \sigma_j(Y) \quad \textit{for all } i, j \geq 1 \textit{ such that } i + j - 1 \leq n.$$

The Hoffman-Wielandt theorem bounds the errors in the differences between the singular values of X and those of Y in terms of $\|X - Y\|_F$.

THEOREM 3.3. *(Hoffman and Wielandt [42]) Let X and Y be $m \times n$ matrices with $m \geq n$. Then*

$$\sqrt{\sum_{j=1}^{n} |\sigma_j(X) - \sigma_j(Y)|^2} \leq \|X - Y\|_F.$$

Below we develop a number of theoretical results that will form the basis for our later analysis on low-rank approximations. Theorem 3.4 below is of potentially broad independent interest. Let B be a rank-k approximation to A. Theorem 3.4 below relates the approximation error in the Frobenius norm to that in the 2-norm as well as the approximation errors in the leading k singular values. It will be called the *Reverse Eckart and Young Theorem* due to its complimentary nature with Theorem 1.1 in the Frobenius norm.

THEOREM 3.4. *(Reverse Eckart and Young) Assume that B is a rank-k approximation to A satisfying*

$$\|A - B\|_F \leq \sqrt{\eta^2 + \sum_{j=k+1}^{n} \sigma_j^2} \tag{3.1}$$

for some $\eta \geq 0$. Then we must have

$$\|A - B\|_2 \leq \sqrt{\eta^2 + \sigma_{k+1}^2}, \tag{3.2}$$

$$\sqrt{\sum_{j=1}^{k} (\sigma_j - \sigma_j(B))^2} \leq \eta. \tag{3.3}$$

REMARK 3.2. Notice that

$$\sqrt{\eta^2 + \sigma_{k+1}^2} = \sigma_{k+1} + \frac{\eta^2}{\sqrt{\eta^2 + \sigma_{k+1}^2} + \sigma_{k+1}}.$$

Equation (3.2) can be simplified to

$$\|A - B\|_2 \leq \sigma_{k+1} + \eta \tag{3.4}$$

when η is larger than or close to σ_{k+1}. On the other hand, if $\eta \ll \sigma_{k+1}$, then equation (3.2) simplifies to

$$\|A - B\|_2 \leq \sigma_{k+1} + \frac{\eta^2}{\sigma_{k+1}},$$

where the last ratio can be much smaller than η, implying a much better rank-k approximation in B. Similar comments apply to equation (3.1). This interesting feature of Theorem 3.4 is one of the reasons why our eventual 2-norm and Frobenius norm upper bounds are much better than those in Theorem 1.2 in the event that $\eta \ll \sigma_{k+1}$. This also has made our proofs in Appendix B somewhat involved in places.

REMARK 3.3. Equation (3.3) asserts that a small η in equation (3.1) necessarily means good approximations to all the k leading singular values of A. In particular, $\eta = 0$ means the leading k singular values of A and B must be the same. However, our singular value analysis will not be based on Equation (3.3), as our approach in Section 4 provides us with much better results.

Proof of Theorem 3.4: Write $A = (A - B) + B$. It follows from Theorem 3.2 that for any $1 \leq i \leq n - k$:

$$\sigma_{i+k}(A) \leq \sigma_i(A - B) + \sigma_{k+1}(B) = \sigma_i(A - B),$$

since B is a rank-k matrix. It follows that

$$\|A - B\|_F^2 = \sum_{i=1}^{n} \sigma_i^2(A - B) \geq \sigma_1^2(A - B) + \sum_{i=2}^{n-k} \sigma_i^2(A - B) \geq \sigma_1^2(A - B) + \sum_{i=2}^{n-k} \sigma_{i+k}^2.$$

Plugging this into equation (3.1) yields (3.2).

As to equation (3.3), we observe that the $(k+1)-st$ through the last singular values of B are all zero, given that B has rank k. Hence the result trivially follows from Theorem 3.3,

$$\sum_{j=1}^{k} (\sigma_j - \sigma_j(B))^2 + \sum_{j=k+1}^{n} \sigma_j^2 \leq \|A - B\|_F^2 \leq \eta^2 + \sum_{j=k+1}^{n} \sigma_j^2. \quad \textbf{Q.E.D.}$$

Our next theorem is a generalization of Theorem 1.1.

THEOREM 3.5. *Let Q be an $m \times \ell$ matrix with orthonormal columns, let $1 \leq k \leq \ell$, and let B_k be the rank-k truncated SVD of $Q^T A$. Then B_k is an optimal solution to the following problem*

$$\min_{\mathbf{rank}(B) \leq k, B \text{ is } \ell \times n} \|A - QB\|_F = \|A - QB_k\|_F. \tag{3.5}$$

In addition, we also have

$$\|A - QB_k\|_F^2 \leq \| \left(I - QQ^T\right) A_k\|_F^2 + \sum_{j=k+1}^{n} \sigma_j^2. \tag{3.6}$$

REMARK 3.4. Problem (3.5) in Theorem 3.5 is a type of restricted SVD problem. Oddly enough, this problem becomes much harder to solve for the 2-norm. In fact, B_k might not even be the solution to the corresponding restricted SVD problem in 2-norm. Combining Theorems 3.4 and 3.5, we obtain

$$\|A - QB_k\|_2^2 \leq \| \left(I - QQ^T\right) A_k\|_F^2 + \sigma_{k+1}^2. \tag{3.7}$$

Our low-rank approximation analysis in the 2-norm will be based on equation (3.7). While this is sufficient, it also makes our 2-norm results perhaps weaker than they should be due to the mixture of the 2-norm and the Frobenius norm.

By Theorem 1.1, A_k is the best Frobenius norm approximation to A, whereas by Theorem 3.5 QB_k is the best restricted Frobenius norm approximation to A. This leads to the following interesting consequence

$$\|A - A_k\|_F \leq \|A - QB_k\|_F \leq \|A - QQ^T A_k\|_F. \tag{3.8}$$

Thus we can expect QB_k to also be an excellent rank-k approximation to A as long as Q points to the principle singular vector directions.

Proof of Theorem 3.5: We first rewrite

$$\|A - QB\|_F^2 = \| \left(I - QQ^T\right) A + Q \left(Q^T A - B\right) \|_F^2 = \| \left(I - QQ^T\right) A\|_F^2 + \| \left(Q^T A - B\right) \|_F^2.$$

Result (3.5) is now an immediate consequence of Theorem 1.1. To prove (3.6), we observe that

$$\begin{aligned}
\|A - QQ^T A_k\|_F^2 &= \mathbf{trace}\left(\left(A - QQ^T A_k\right)^T \left(A - QQ^T A_k\right)\right) \\
&= \mathbf{trace}\left(\left(A - A_k + A_k - QQ^T A_k\right)^T \left(A - A_k + A_k - QQ^T A_k\right)\right) \\
&= \|A - A_k\|_F^2 + \|A_k - QQ^T A_k\|_F^2 + 2\mathbf{trace}\left(\left(A - A_k\right)^T \left(A_k - QQ^T A_k\right)\right) \\
&= \sum_{j=k+1}^{n} \sigma_j^2 + \|A_k - QQ^T A_k\|_F^2 + 2\mathbf{trace}\left(\left(\left(I - QQ^T\right) A_k\right) \left(A - A_k\right)^T\right).
\end{aligned}$$

The third term in the last equation is zero because $A_k \left(A - A_k\right)^T = 0$. Combining this last relation with equation (3.8) gives us relation (3.6). **Q.E.D.**

4. Deterministic Analysis. In this section we perform deterministic convergence analysis on Algorithm 2.1. Theorem 4.3 is a relative convergence lower bound, and Theorem 4.4 is an upper bound on the matrix approximation error. Both appear to be new for subspace iteration. Our approach, while quite novel, was motivated in part by the analysis of subspace iteration methods by Saad [69] and randomized algorithms in [36]. Since Algorithm 1.1 is a special case of Algorithm 2.2 with $q = 0$, which in turn is a special case of Algorithm 2.1 with an initial random matrix, our analysis applies to them as well and will form the basis for additional probabilistic analysis in Section 5.

4.1. A Special Orthonormal Basis. We begin by noticing that the output QB_k in Algorithm 2.1 is also the rank-k truncated SVD of the matrix QQ^TA, due to the fact that Q is column orthonormal. In fact, columns of Q are nothing but an orthonormal basis for the column space of matrix $\left(AA^T\right)^q A\,\Omega$. This is the reason why Algorithm 2.1 is called subspace iteration. Lemma 4.1 below shows how to obtain alternative orthonormal bases for the same column space. We omit the proof.

LEMMA 4.1. *In the notation of Algorithm 2.1, assume that X is a non-singular $\ell \times \ell$ matrix and that Ω has full column rank. Let $\widehat{Q}\widehat{R}$ be the QR factorization of the matrix $\left(AA^T\right)^q A\Omega X$, then*

$$QQ^T = \widehat{Q}\widehat{Q}^T.$$

Since

$$\left(AA^T\right)^q A\Omega = U\Sigma^{2q+1}V^T\Omega,$$

define and partition

$$\widehat{\Omega} \stackrel{def}{=} V^T\Omega = \begin{array}{cc} \ell - p & \{ \\ n-\ell+p & \{ \end{array} \begin{pmatrix} \widehat{\Omega}_1 \\ \widehat{\Omega}_2 \end{pmatrix}, \tag{4.1}$$

where $0 \le p \le \ell - k$. The introduction of the additional parameter p is to balance the need for oversampling for reliability (see Theorem 1.2) and oversampling for faster convergence (see Theorem 2.1). We also partition $\Sigma = \operatorname{diag}(\Sigma_1, \Sigma_2, \Sigma_3)$, where Σ_1, Σ_2, and Σ_3 are $k \times k$, $(\ell - p - k) \times (\ell - p - k)$, and $(n - \ell + p) \times (n - \ell + p)$. This partition allows us to further write

$$\left(AA^T\right)^q A\Omega = U \begin{pmatrix} \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix}^{2q+1} \widehat{\Omega}_1 \\ \\ \Sigma_3^{2q+1}\widehat{\Omega}_2 \end{pmatrix}. \tag{4.2}$$

The matrix $\widehat{\Omega}_1$ has at least as many columns as rows. Assume it is of full row rank so that its pseudo-inverse satisfies

$$\widehat{\Omega}_1\widehat{\Omega}_1^\dagger = I.$$

Below we present a special choice of X that will reveal the manner in which convergence to singular values and low-rank approximations takes place. Ideally, such an X would orient the first k columns of $U \begin{pmatrix} \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix}^{2q+1} \widehat{\Omega}_1 \\ \\ \Sigma_3^{2q+1}\widehat{\Omega}_2 \end{pmatrix} X$ in the directions of the leading k singular vectors in U. We choose

$$X = \left(\widehat{\Omega}_1^\dagger \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix}^{-(2q+1)}, \quad \widehat{X}\right), \tag{4.3}$$

where the $\ell \times p$ matrix $\widehat{X}$ is chosen so that X is non-singular and $\widehat{\Omega}_1 \widehat{X} = 0$. Recalling equation (4.2), this choice of X allows us to write

$$\left(AA^T\right)^q A\Omega X = U \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ H_1 & H_2 & H_3 \end{pmatrix}, \tag{4.4}$$

where

$$H_1 = \Sigma_3^{2q+1} \widehat{\Omega}_2 \widehat{\Omega}_1^\dagger \begin{pmatrix} \Sigma_1^{-(2q+1)} \\ 0 \end{pmatrix}, \quad H_2 = \Sigma_3^{2q+1} \widehat{\Omega}_2 \widehat{\Omega}_1^\dagger \begin{pmatrix} 0 \\ \Sigma_2^{-(2q+1)} \end{pmatrix}, \quad H_3 = \Sigma_3^{2q+1} \widehat{\Omega}_2 \widehat{X}.$$

Notice that we have created a "gap" in H_1: the largest singular value in Σ_3 is $\sigma_{\ell-p+1}$, which is potentially much smaller than σ_k, the smallest singular value in Σ_1. We can expect H_1 to converge to 0 rather quickly when $q \to \infty$, if $\sigma_{\ell-p+1} \ll \sigma_k$ and if the matrix $\widehat{\Omega}_1^\dagger$ is not too large in norm. Our convergence analysis of Algorithms 2.1 and 2.2 will mainly involve deriving upper bounds on various functions related to H_1. Our gap disappears when we choose $p = \ll -k$, in which case our results in Section 5 will be more in line with Theorem 1.2.

By equation (4.4), the QR factorization of $\left(AA^T\right)^q A\Omega X$ can now be written in the following 3×3 partition:

$$U \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ H_1 & H_2 & H_3 \end{pmatrix} = \widehat{Q}\widehat{R} = \begin{pmatrix} \widehat{Q}_1 & \widehat{Q}_2 & \widehat{Q}_3 \end{pmatrix} \begin{pmatrix} \widehat{R}_{11} & \widehat{R}_{12} & \widehat{R}_{13} \\ & \widehat{R}_{22} & \widehat{R}_{23} \\ & & \widehat{R}_{33} \end{pmatrix}. \tag{4.5}$$

We will use this representation to derive convergence upper bounds for singular value and rank-k approximations. In particular, we will make use of the fact that the above QR factorization also embeds another one

$$U \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix} = \widehat{Q}_1 \widehat{R}_{11}. \tag{4.6}$$

We are now ready to derive a lower bound on $\sigma_k(B_k)$.

LEMMA 4.2. *Let H_1 be defined in equation (4.4), and assume that the matrix $\widehat{\Omega}_1$ has full row rank, then the matrix B_k computed in Algorithm 2.1 must satisfy*

$$\sigma_k\left(B_k\right) \geq \frac{\sigma_k}{\sqrt{1 + \|H_1\|_2^2}}. \tag{4.7}$$

REMARK 4.1. It might seem more intuitive in equation (4.3) to choose $X = \begin{pmatrix} X_1 & X_2 \end{pmatrix}$ where X_1 solves the following least squares problem

$$\min_{X_1} \left\| \begin{pmatrix} \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix}^{2q+1} \widehat{\Omega}_1 \\ \\ \Sigma_3^{2q+1} \widehat{\Omega}_2 \end{pmatrix} X_1 - \begin{pmatrix} I \\ 0 \\ 0 \end{pmatrix} \right\|_2.$$

Our choice of X seems as effective and allows simpler analysis.

Proof of Lemma 4.2: We note by Lemma 4.1 that

$$QQ^TA = \widehat{Q}\widehat{Q}^TA = \widehat{Q}\left(\begin{array}{c|c} \widehat{Q}_1^TU\begin{pmatrix}\Sigma_1\\0\\0\end{pmatrix} & \widehat{Q}_1^TU\begin{pmatrix}0&0\\\Sigma_2&\\&\Sigma_3\end{pmatrix} \\ \hline \begin{pmatrix}\widehat{Q}_2^T\\\widehat{Q}_3^T\end{pmatrix}U\begin{pmatrix}\Sigma_1\\0\\0\end{pmatrix} & \begin{pmatrix}\widehat{Q}_2^T\\\widehat{Q}_3^T\end{pmatrix}U\begin{pmatrix}0&0\\\Sigma_2&\\&\Sigma_3\end{pmatrix}\end{array}\right)V^T. \tag{4.8}$$

From equations (4.8) and (4.6), we see that the matrix

$$\widehat{Q}_1^TU\begin{pmatrix}\Sigma_1\\0\\0\end{pmatrix} = \left(U\begin{pmatrix}I\\0\\H_1\end{pmatrix}\widehat{R}_{11}^{-1}\right)^T U\begin{pmatrix}\Sigma_1\\0\\0\end{pmatrix} = \widehat{R}_{11}^{-T}\Sigma_1$$

is simply a submatrix of the middle matrix on the right hand side of equation (4.8). By Remark 3.1, it follows immediately that

$$\sigma_k\left(B_k\right) = \sigma_k\left(\widehat{Q}\widehat{Q}^TA\right) \geq \sigma_k\left(\widehat{R}_{11}^{-T}\Sigma_1\right).$$

On the other hand, we also have

$$\sigma_k = \sigma_k\left(\widehat{R}_{11}^T\left(\widehat{R}_{11}^{-T}\Sigma_1\right)\right) \leq \left\|\widehat{R}_{11}^T\right\|_2\sigma_k\left(\widehat{R}_{11}^{-T}\Sigma_1\right).$$

Combining these two relations, and together with the fact that $\left\|\widehat{R}_{11}^T\right\|_2 = \sqrt{1+\|H_1\|_2^2}$, we obtain (4.7). **Q.E.D.**

4.2. Deterministic Bounds. In this section we develop the analysis in Section 4.1 into deterministic lower bounds for singular values and upper bounds for rank-k approximations.

Since the interlacing theorem 3.1 asserts an upper bound $\sigma_k\left(B_k\right) \leq \sigma_k$, equation (4.7) provides a nice lower bound on $\sigma_k\left(B_k\right)$. These bounds mean that $\sigma_k\left(B_k\right)$ is a good approximation to σ_k as long as $\|H_1\|_2$ is small. This consideration is formalized in the theorem below.

THEOREM 4.3. *Let $A = U\Sigma V^T$ be the SVD of A, let $0 \leq p \leq \ell - k$, and let $V^T\Omega$ be partitioned in equation (4.1). Assume that the matrix $\widehat{\Omega}_1$ has full row rank, then Algorithm 2.1 must satisfy for $j = 1, \cdots, k$:*

$$\sigma_j \geq \sigma_j\left(B_k\right) \geq \frac{\sigma_j}{\sqrt{1+\left\|\widehat{\Omega}_2\right\|_2^2\left\|\widehat{\Omega}_1^\dagger\right\|_2^2\left(\frac{\sigma_{\ell-p+1}}{\sigma_j}\right)^{4q+2}}}.$$

Proof of Theorem 4.3: By the definition of the matrix H_1 in equation (4.4), it is straightforward to get

$$\|H_1\|_2 \leq \left\|\widehat{\Omega}_2\right\|_2\left\|\widehat{\Omega}_1^\dagger\right\|_2\left(\frac{\sigma_{\ell-p+1}}{\sigma_k}\right)^{2q+1}. \tag{4.9}$$

This, together with lower bound (4.7), gives the result in Theorem 4.3 for $j = k$. To prove Theorem 4.3 for any $1 \leq j < k$, we observe that since $\sigma_j\left(B_k\right) = \sigma_j\left(B_j\right)$, all that is needed is to repeat all previous arguments for a rank j truncated SVD. **Q.E.D.**

REMARK 4.2. Theorem 4.3 makes explicit the two key factors governing the convergence of Algorithm 2.1. On one hand, we can expect fast convergence for $\sigma_j(B_k)$ if $\sigma_{\ell-p+1} \ll \sigma_j$. On the other hand, an unfortunate

choice of Ω could potentially make $\|\Omega_1^\dagger\|_2$ very large, leading to slow converge even if the singular values do decay quickly. The main effect of randomization in Algorithm 2.2 is to ensure a reasonably sized $\|\widehat{\Omega}_2\|_2\|\widehat{\Omega}_1^\dagger\|_2$ with near certainty. See Theorem 5.8 for a precise statement and more details.

Now we consider rank-k approximation upper bounds. Toward this end and considering Theorem 3.5, we would like to start with an upper bound on $\| \left(I - QQ^T\right) A_k\|_F$. By Lemma 4.1 and equation (4.5), we have

$$\left\| \left(I - QQ^T\right) A_k \right\|_F = \left\| \left(I - \widehat{Q}\widehat{Q}^T\right) A_k \right\|_F \leq \left\| \left(I - \widehat{Q}_1\widehat{Q}_1^T\right) A_k \right\|_F .$$

Since $A_k = U \operatorname{diag}(\Sigma_1, 0, 0)\, V^T$, and since $\widehat{Q}_1 = U \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix} \widehat{R}_{11}^{-1}$ according to equation (4.6), the above right hand side becomes

$$\left\| \left(I - \widehat{Q}_1\widehat{Q}_1^T\right) A_k \right\|_F = \left\| \left(I - \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix} \left(I + H_1^T H_1\right)^{-1} \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix}^T \right) \begin{pmatrix} \Sigma_1 \\ 0 \\ 0 \end{pmatrix} \right\|_F ,$$

where we have used the fact that (see (4.6))

$$\widehat{R}_{11}^{-1}\widehat{R}_{11}^{-T} = \left(\widehat{R}_{11}^T\widehat{R}_{11}\right)^{-1} = \left(I + H_1^T H_1\right)^{-1} .$$

Continuing,

$$\begin{aligned} \left\| \left(I - \widehat{Q}_1\widehat{Q}_1^T\right) A_k \right\|_F &= \left\| \begin{pmatrix} I - \left(I + H_1^T H_1\right)^{-1} & 0 & -\left(I + H_1^T H_1\right)^{-1} H_1^T \\ 0 & I & 0 \\ -H_1 \left(I + H_1^T H_1\right)^{-1} & 0 & I - H_1 \left(I + H_1^T H_1\right)^{-1} H_1^T \end{pmatrix} \begin{pmatrix} \Sigma_1 \\ 0 \\ 0 \end{pmatrix} \right\|_F \\ &= \left\| \begin{pmatrix} H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 \\ \\ -\left(I + H_1 H_1^T\right)^{-1} H_1 \end{pmatrix} \Sigma_1 \right\|_F \\ &= \sqrt{\mathbf{trace}\left(\Sigma_1 H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 \Sigma_1\right)} \end{aligned} \tag{4.10}$$

We are now ready to prove

THEOREM 4.4. *With the notation of Section 4, we have*

$$\| \left(I - QQ^T\right) A\|_F \leq \|A - QB_k\|_F \leq \sqrt{\left(\sum_{j=k+1}^{n} \sigma_j^2\right) + \frac{\alpha^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + \gamma^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}},$$

$$\| \left(I - QQ^T\right) A\|_2 \leq \|A - QB_k\|_2 \leq \sqrt{\sigma_{k+1}^2 + \frac{\alpha^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + \gamma^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}},$$

where

$$\alpha = \sqrt{k}\sigma_{\ell-p+1} \left(\frac{\sigma_{\ell-p+1}}{\sigma_k}\right)^{2q} \quad \textit{and} \quad \gamma = \left(\frac{\sigma_{\ell-p+1}}{\sigma_1}\right)\left(\frac{\sigma_{\ell-p+1}}{\sigma_k}\right)^{2q} .$$

REMARK 4.3. Theorem 4.4 trivially simplifies to

$$\left\| \left(I - QQ^T\right) A\right\|_F \leq \|A - QB_k\|_F \leq \sqrt{\left(\sum_{j=k+1}^{n} \sigma_j^2\right) + \alpha^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2},$$

$$\left\| \left(I - QQ^T\right) A\right\|_2 \leq \|A - QB_k\|_2 \leq \sqrt{\sigma_{k+1}^2 + \alpha^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}.$$

However, when Ω is taken to be Gaussian, only the bounds in Theorem 4.4 allow average case analysis for all values of p (See Section 5.2.)

REMARK 4.4. Not surprisingly, the two key factors governing the singular value convergence of Algorithm 2.1 also govern the convergence of the low-rank approximation. Hence Remark 4.2 applies equally well to Theorem 4.4.

Proof of Theorem 4.4: We first assume that the matrix H_1 in equation (4.4) has full column rank. Rewrite

$$\begin{aligned}
\Sigma_1 H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 \Sigma_1 &= \left(\left(\left(H_1\Sigma_1\right)^T \left(H_1\Sigma_1\right)\right)^{-1} + \Sigma_1^{-2}\right)^{-1} \\
&= \left(\|H_1\Sigma_1\|_2^{-2} I + \Sigma_1^{-2}\right)^{-1} - \\
&\quad \left(\left(\|H_1\Sigma_1\|_2^{-2} I + \Sigma_1^{-2}\right)^{-1} - \left(\left(\left(H_1\Sigma_1\right)^T \left(H_1\Sigma_1\right)\right)^{-1} + \Sigma_1^{-2}\right)^{-1}\right).
\end{aligned}$$

The last expression is a symmetric positive semi-definite matrix. This allows us to write

$$\begin{aligned}
\left\|\left(I - QQ^T\right) A_k\right\|_F &\leq \sqrt{\mathbf{trace}\left(\Sigma_1 H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 \Sigma_1\right)} \\
&\leq \sqrt{\mathbf{trace}\left(\left(\|H_1\Sigma_1\|_2^{-2} I + \Sigma_1^{-2}\right)^{-1}\right)} = \|H_1\Sigma_1\|_2 \sqrt{\mathbf{trace}\left(\Sigma_1 \left(\|H_1\Sigma_1\|_2^2 I + \Sigma_1^2\right)^{-1} \Sigma_1\right)} \\
&\leq \frac{\sqrt{k}\,\|H_1\Sigma_1\|_2\,\sigma_1}{\sqrt{\sigma_1^2 + \|H_1\Sigma_1\|_2^2}}. \qquad (4.11)
\end{aligned}$$

By a continuity argument, the last relation remains valid even if H_1 does not have full column rank.

Due to the special form of H_1 in equation (4.4), we can write $H_1\Sigma_1$ as

$$H_1\Sigma_1 = \Sigma_3^{2q+1} \widehat{\Omega}_2 \widehat{\Omega}_1^\dagger \begin{pmatrix} \Sigma_1^{-(2q)} \\ 0 \end{pmatrix}.$$

Hence

$$\|H_1\Sigma_1\|_2 \leq \sigma_{\ell-p+1} \left(\frac{\sigma_{\ell-p+1}}{\sigma_k}\right)^{2q} \|\widehat{\Omega}_2\|_2 \|\widehat{\Omega}_1^\dagger\|_2.$$

Plugging this into equation (4.11) and dividing both the nomerator and denominator by σ_1,

$$\left\|\left(I - QQ^T\right) A_k\right\|_F \leq \frac{\alpha \left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2}{\sqrt{1 + \gamma^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}}.$$

Comparing this with Theorems 3.4 and 3.5 proves Theorem 4.4. **Q.E.D.**

5. Statistical Analysis. This section carries out the needed statistical analysis to reach our approximation error bounds. In Section 5.1 we make a list of the statistical tools used in this analysis; in Section 5.2 we perform average value analysis on our error bounds; and in Section 5.3 we provide large deviation bounds.

5.1. Statistical Tools. The simplest of needed statistical results necessary for our analysis is the following proposition from [36].

PROPOSITION 5.1. *For fix matrices S, T and standard Gaussian matrix G, we have*

$$\mathbb{E}\|SGT\|_2 \leq \|S\|_2\|T\|_F + \|S\|_F\|T\|_2.$$

The following large deviation bound for the pseudo-inverse of a Gaussian matrix is also from [36].

LEMMA 5.2. *Let G be an $(\ell - p) \times \ell$ Gaussian matrix where $p \geq 0$ and $\ell - p \geq 2$. Then* $\mathbf{rank}(G) = \ell - p$ *with probability* 1. *For all $t \geq 1$,*

$$\mathbb{P}\left\{\left\|G^{\dagger}\right\|_2 \geq \frac{et\sqrt{\ell}}{p+1}\right\} \leq t^{-(p+1)}.$$

The following theorem provides classical tail bounds for functions of Gaussian matrices. It was taken from [6][Thm. 4.5.7].

THEOREM 5.3. *Suppose that h is a real valued Lipschitz function on matrices:*

$$|h(X) - h(Y)| \leq \mathcal{L}\|X - Y\|_F \quad \textit{for all } X, Y \textit{ and a constant } \mathcal{L} > 0.$$

Draw a standard Gaussian matrix G. Then

$$\mathbb{P}\left\{h(G) \geq \mathbb{E}h(G) + \mathcal{L}u\right\} \leq e^{-u^2/2}.$$

The two propositions below will be used in our average case error bounds analysis, both for singular values and rank-k approximations. Their proofs are lengthy and can be found in the *Supplemental Material*.

PROPOSITION 5.4. *Let $\alpha > 0$, $\beta > 0$, $\gamma > 0$ and $\delta > 0$, and let G be an $m \times n$ Gaussian matrix. Then*

$$\mathbb{E}\left(\frac{1}{\sqrt{1 + \alpha^2\|G\|_2^2}}\right) \geq \frac{1}{\sqrt{1 + \alpha^2\mathcal{C}^2}} \tag{5.1}$$

$$\mathbb{E}\left(\sqrt{\delta^2 + \frac{\alpha^2\|G\|_2^2}{\beta^2 + \gamma^2\|G\|_2^2}}\right) \leq \sqrt{\delta^2 + \frac{\alpha^2\mathcal{C}^2}{\beta^2 + \gamma^2\mathcal{C}^2}}, \tag{5.2}$$

where $\mathcal{C} = \sqrt{m} + \sqrt{n} + 7$.

There are lower and upper bounds similar to Proposition 5.4 for the pseudo-inverse of a Gaussian, with a significant complication. When G is a square Gaussian matrix, it is non-singular with probability 1. However, the probability density function for its pseudo-inverse could have a very long tail according to Lemma 5.2. A similar argument could also be made when G is almost a square matrix. This complication will have

important implications for parameter choices in Algorithm 2.2 (see Sections 5.2 and 5.3.) Function $\log(\cdot)$ below is base-e.

PROPOSITION 5.5. *Let $\alpha > 0$, $\beta > 0$, $\gamma > 0$ and $\delta > 0$, and let G be an $(\ell - p) \times \ell$ Gaussian matrix. Then* $\mathbf{rank}(G) = \ell - p$ *with probability* 1*, and*

$$\mathbb{E}\left(\frac{1}{\sqrt{1+\alpha^2\|G^\dagger\|_2^2}}\right) \geq \begin{cases} \dfrac{1}{\sqrt{1+\alpha^2\mathcal{C}^2}} & \text{for } p \geq 2, \\ \dfrac{1}{1+\alpha^2\mathcal{C}^2\log\frac{2\sqrt{1+\alpha^2\mathcal{C}^2}}{\alpha\mathcal{C}}} & \text{for } p = 1, \\ \dfrac{1}{1+\alpha\mathcal{C}} & \text{for } p = 0. \end{cases} \tag{5.3}$$

$$\mathbb{E}\left(\sqrt{\delta^2+\frac{\alpha^2\|G^\dagger\|_2^2}{\beta^2+\gamma^2\|G^\dagger\|_2^2}}\right) \leq \begin{cases} \sqrt{\delta^2+\dfrac{\alpha^2\mathcal{C}^2}{\beta^2+\gamma^2\mathcal{C}^2}} & \text{for } p \geq 2, \\ \delta+\dfrac{\alpha^2(\ell-1)}{\delta\beta^2}\left(2+\dfrac{1}{2}\log\left(1+\dfrac{\delta^2\beta^2}{\alpha^2}\right)\right) & \text{for } p = 1, \\ \delta+\dfrac{4\sqrt{\ell}\sqrt{\delta^2\gamma^2+\alpha^2}}{\beta}\log\left(1+\left(\dfrac{\alpha}{\delta\gamma}\right)^2\right) & \text{for } p = 0, \end{cases} \tag{5.4}$$

where $\mathcal{C} = \dfrac{4e\sqrt{\ell}}{p+1}$.

5.2. Average Case Error Bounds. This section is devoted to the average case analysis of Algorithm 2.2. This work requires us to study the average case behavior on the upper and lower bounds in Theorems 4.3 and 4.4. As observed in Section 2.2, the distribution of a standard Gaussian matrix is rotationally invariant, and hence the matrices $\widehat{\Omega}_1$ and $\widehat{\Omega}_2$ are themselves independent standard Gaussian matrices. With the tools established in Section 5.1, our analysis here consists mostly of stitching together the right pieces from there.

We first analyze the singular value lower bounds in Theorems 5.6. This will require separate analysis for $p \geq 2$, $p = 1$, and $p = 0$, as suggested in Section 5.1. We then analyze the low-rank approximation bounds in Theorem 4.4, which also requires separate analysis for the same three cases of p. Throughout Section 5.2, we will need the following definition for any $0 \leq p \leq \ell$:

$$\mathcal{C}_1 = \sqrt{n-\ell+p}+\sqrt{\ell}+7, \quad \mathcal{C}_2 = \frac{4e\sqrt{\ell}}{p+1}, \quad \mathcal{C} = \mathcal{C}_1\mathcal{C}_2, \quad \text{and} \quad \tau_j = \frac{\sigma_{\ell-p+1}}{\sigma_j}.$$

THEOREM 5.6. *Let $A = U\Sigma V^T$ be the SVD of A, and let QB_k be a rank-k approximation computed by Algorithm 2.2. Then for $j = 1, \cdots, k$,*

$$\mathbb{E}\left(\sigma_j\left(QB_k\right)\right) \geq \begin{cases} \dfrac{\sigma_j}{\sqrt{1+\mathcal{C}^2\tau_j^{4q+2}}} & \text{for } p \geq 2, \\ \dfrac{\sigma_j}{1+\mathcal{C}^2\tau_j^{4q+2}\log\sqrt{\mathcal{C}^2+\tau_j^{-(4q+2)}}} & \text{for } p = 1, \\ \dfrac{\sigma_j}{1+\mathcal{C}\tau_j^{2q+1}} & \text{for } p = 0. \end{cases} \tag{5.5}$$

REMARK 5.1. The value of p is not part of Algorithm 2.2 and can thus be arbitrarily chosen within $[0, \ell - k]$. Since our bounds for $p \leq 1$ are worse than that for $p \geq 2$, they should probably not be used unless $\ell - k \leq 1$ or unless there is a large singular value gap at σ_ℓ or $\sigma_{\ell+1}$.

REMARK 5.2. Theorem 5.6 strongly suggests that in general some over-sampling in the number of columns can significantly improve convergence in the singular value approximation. This is consistent with the literature [16, 21, 22, 28, 29, 55, 54, 62, 81, 71, 36] and is very significant for practical implementations.

REMARK 5.3. A typical implementation of the classical subspace iteration method in general and the classical power method in particular chooses $\ell = k$, which leads to $p = 0$. Theorem 5.6 implies that this choice in general leads to slower convergence than $p > 0$ and thus should be avoided. We will elaborate this point in more detail in Section 5.3 and provide numerical evidence to support this conclusion in Section 8.

REMARK 5.4. Since $\tau_j \leq 1$ for all j, Theorem 5.6 implies that for $p \geq 2$ and for all $j \leq k$,

$$\mathbb{E}\left(\sigma_j\left(QB_k\right)\right) \geq \frac{\sigma_j}{\sqrt{1+\mathcal{C}^2}}.$$

In other words, Algorithm 2.2 approximates the leading k singular values by a good fraction on average, regardless of how the singular values are distributed, even for $q = 0$. This result is surprising and yet valuable. It will have applications in condition number estimation (see Sections 5.3 and 7 for more discussion.)

REMARK 5.5. For matrices with rapidly decaying singular values, convergence could be so rapid that one could even set $q = 0$ in some cases (Section 5.3.) This is the basis of the excitement about Algorithm 2.2 in that very little work is typically sufficient to realize an excellent low-rank approximation. The faster the singular values decay, the faster Algorithm 2.2 converges.

REMARK 5.6. Kuczyński and Woźniakowski [47] developed probabilistic error bounds for computing the largest eigenvalue of an SPD matrix by the power method for a unit start vector under the uniform distribution. Their results correspond to the case of $\ell = k = 1$ and $p = 0$ in Theorem 5.6. However, our results appear to be much stronger.

Proof of Theorem 5.6: As in Theorem 4.3, we will only prove Theorem 5.6 for $j = k$. All other values of j can be proved by simply citing Theorem 5.6 for a rank-j SVD truncation. Since $\widehat{\Omega}_2$ and $\widehat{\Omega}_1$ are independent of each other, we will take expectations over $\widehat{\Omega}_2$ and $\widehat{\Omega}_1$ in turn, based on Propositions 5.4 and 5.5.

Let $\alpha = \left\|\widehat{\Omega}_1^\dagger\right\|_2 \tau_k^{2q+1}$. By Theorem 4.3 and Proposition 5.4,

$$\mathbb{E}\left(\sigma_k\left(QB_k\right) \,\middle|\, \widehat{\Omega}_1\right) \geq \frac{\sigma_k}{\sqrt{1+\alpha^2\mathcal{C}_1^2}}. \tag{5.6}$$

For $p \geq 2$, we further take expectation over $\widehat{\Omega}_1$ according to Proposition 5.5. By equation (5.6),

$$\mathbb{E}\left(\sigma_k\left(QB_k\right)\right) = \mathbb{E}\left(\mathbb{E}\left(\sigma_k\left(QB_k\right) \,\middle|\, \widehat{\Omega}_1\right)\right) \geq \mathbb{E}\left(\frac{\sigma_k}{\sqrt{1+\left(\left\|\widehat{\Omega}_1^\dagger\right\|_2 \tau_k^{2q+1}\right)^2 \mathcal{C}_1^2}}\right) \geq \frac{\sigma_k}{\sqrt{1+\mathcal{C}^2\tau_k^{4q+2}}}.$$

To complete the proof, we note that the results for $p = 1$ and $p = 0$ can be obtained similarly by taking expectation of $\widehat{\Omega}_1$ over equation (5.6) and simplifying. **Q.E.D.**

It is now time for average case analysis of low-rank matrix approximations. Again, we base our arguments on Propositions 5.4 and 5.5. For ease of notation, let

$$\widehat{\delta}_{k+1} = \sqrt{\sum_{j=k+1}^{n} \sigma_j^2}.$$

For the sake of simplicity, in Theorem 5.6 below we have omitted

$$\mathbb{E}\| \left(I - QQ^T\right) A\|_F \leq \mathbb{E}\|A - QB_k\|_F.$$

THEOREM 5.7. *Let QB_k be a rank-k approximation computed by Algorithm 2.2. Then*

$$\mathbb{E}\|A - QB_k\|_F \leq \begin{cases} \sqrt{\widehat{\delta}_{k+1}^2 + k\mathcal{C}^2\sigma_{\ell-p+1}^2\tau_k^{4q}} & \text{for } p \geq 2, \\ \widehat{\delta}_{k+1} + \dfrac{k\mathcal{C}^2\sigma_{\ell-p+1}^2\tau_k^{4q}}{\widehat{\delta}_{k+1}} \log\sqrt{\mathcal{C}^2 + \dfrac{1}{k}\left(\dfrac{\widehat{\delta}_{k+1}}{\sigma_{\ell-p+1}}\right)^2 \tau_k^{-4q}} & \text{for } p = 1, \\ \widehat{\delta}_{k+1} + \sqrt{n}\mathcal{C}\sigma_{\ell-p+1}\tau_k^{2q} \log\left(1 + k\left(\dfrac{\sigma_1}{\widehat{\delta}_{k+1}}\right)^2\right) & \text{for } p = 0. \end{cases}$$

$$\mathbb{E}\|A - QB_k\|_2 \leq \begin{cases} \sqrt{\sigma_{k+1}^2 + k\mathcal{C}^2\sigma_{\ell-p+1}^2\tau_k^{4q}} & \text{for } p \geq 2, \\ \sigma_{k+1} + \dfrac{k\mathcal{C}^2\sigma_{\ell-p+1}^2\tau_k^{4q}}{\sigma_{k+1}} \log\sqrt{\mathcal{C}^2 + \dfrac{1}{k}\left(\dfrac{\sigma_{k+1}}{\sigma_{\ell-p+1}}\right)^2 \tau_k^{-4q}} & \text{for } p = 1, \\ \sigma_{k+1} + \sqrt{(k+1)}\mathcal{C}\sigma_{\ell-p+1}\tau_k^{2q} \log\left(1 + k\left(\dfrac{\sigma_1}{\sigma_{k+1}}\right)^2\right) & \text{for } p = 0. \end{cases}$$

REMARK 5.7. Remark 3.2 applies to Theorem 5.7 as well.

Proof of Theorem 5.7: We only prove Theorem 5.7 for the Frobenius norm. The case for the 2-norm is completely analogous. As in the proof for Theorem 5.6, this one involves taking expectations over $\widehat{\Omega}_2$ first and $\widehat{\Omega}_1$ next. Let $\delta = \widehat{\delta}_{k+1} = \sqrt{\sum_{j=k+1}^{n} \sigma_j^2}$. Fixing $\widehat{\Omega}_1$ in Theorem 4.4 and taking expectation on $\widehat{\Omega}_2$ according to Proposition 5.4, we obtain immediately

$$\mathbb{E}\|A - QB_k\|_F \leq \sqrt{\widehat{\delta}_{k+1}^2 + \frac{\alpha^2\mathcal{C}_1^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + \gamma^2\mathcal{C}_1^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}}, \tag{5.7}$$

with $\alpha = \sqrt{k}\sigma_{\ell-p+1}\tau_k^{2q}$ and $\gamma = \left(\dfrac{\sigma_{\ell-p+1}}{\sigma_1}\right)\tau_k^{2q}$.

For $p \geq 2$, we further take expectation over $\widehat{\Omega}_1$ according to Proposition 5.5. By equation (5.4),

$$\mathbb{E}\left(\mathbb{E}\|A - QB_k\|_F \;\middle|\; \widehat{\Omega}_1\right) \leq \mathbb{E}\left(\sqrt{\widehat{\delta}_{k+1}^2 + \frac{\alpha^2 \mathcal{C}_1^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + \gamma^2 \mathcal{C}_1^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}}\right)$$
$$\leq \sqrt{\widehat{\delta}_{k+1}^2 + \frac{\alpha^2 \mathcal{C}^2}{1 + \gamma^2 \mathcal{C}^2}} \leq \sqrt{\widehat{\delta}_{k+1}^2 + \alpha^2 \mathcal{C}^2},$$

which is the Frobenius norm upper bound in Theorem 5.7.

For $p = 1$, we again take expectation over $\widehat{\Omega}_1$ in equation (5.7) according to Proposition 5.5:

$$\mathbb{E}\|A - QB_k\|_F \leq \mathbb{E}\left(\sqrt{\widehat{\delta}_{k+1}^2 + \frac{\alpha^2 \mathcal{C}_1^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + \gamma^2 \mathcal{C}_1^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}}\right)$$
$$\leq \widehat{\delta}_{k+1} + \frac{\alpha^2 \mathcal{C}_1^2 \,(\ell - 1)}{\widehat{\delta}_{k+1}} \left(2 + \frac{1}{2} \log\left(1 + \frac{\widehat{\delta}_{k+1}^2}{\alpha^2 \mathcal{C}_1^2}\right)\right),$$

which is bounded above by the corresponding expression in Theorem 5.7.

Now, we turn our attention to the case $p = 0$. Taking expectations as before,

$$\mathbb{E}\|A - QB_k\|_F \leq \mathbb{E}\left(\sqrt{\widehat{\delta}_{k+1}^2 + \frac{\alpha^2 \mathcal{C}_1^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + \gamma^2 \mathcal{C}_1^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}}\right)$$
$$\leq \widehat{\delta}_{k+1} + 4\sqrt{\ell}\sqrt{\widehat{\delta}_{k+1}^2 \gamma^2 \mathcal{C}_1^2 + \alpha^2 \mathcal{C}_1^2} \log\left(1 + \left(\frac{\alpha \mathcal{C}_1}{\widehat{\delta}_{k+1}^2 \gamma \mathcal{C}_1}\right)^2\right)$$
$$= \widehat{\delta}_{k+1} + 4\sqrt{\ell}\mathcal{C}_1\sqrt{\widehat{\delta}_{k+1}^2 \gamma^2 + \alpha^2} \log\left(1 + \left(\frac{\alpha}{\widehat{\delta}_{k+1}\gamma}\right)^2\right).$$

Plugging in the expressions for α and γ in equation (5.7),

$$\mathbb{E}\|A - QB_k\|_F \leq \widehat{\delta}_{k+1} + 4\sqrt{\ell}\mathcal{C}_1\sqrt{\left(\frac{\widehat{\delta}_{k+1}}{\sigma_1}\right)^2 + k\;\sigma_{\ell-p+1}\tau_k^{2q}} \log\left(1 + k\left(\frac{\sigma_1}{\widehat{\delta}_{k+1}}\right)^2\right),$$

which is bounded above by the corresponding expression in Theorem 5.7 since $\widehat{\delta}_{k+1} \leq \sqrt{n-k}\;\sigma_1$. **Q.E.D.**

5.3. Large Deviation Bounds. In this section we develop approximation error tail bounds. Theorems 4.3 and 4.4 dictate that our main focus will be in developing probabilistic upper bounds on $\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2$.

THEOREM 5.8. *Let $A = U\Sigma V^T$ be the SVD of A, and $0 \leq p \leq \ell - k$. Further let QB_k be a rank-k approximation computed by Algorithm 2.2. Given any $0 < \Delta \ll 1$, define*

$$\mathcal{C}_\Delta = \frac{e\sqrt{\ell}}{p+1}\left(\frac{2}{\Delta}\right)^{\frac{1}{p+1}}\left(\sqrt{n-\ell+p} + \sqrt{\ell} + \sqrt{2\log\frac{2}{\Delta}}\right).$$

We must have for $j = 1, \cdots, k$,

$$\sigma_j\left(QB_k\right) \geq \frac{\sigma_j}{\sqrt{1+\mathcal{C}_\Delta^2\left(\frac{\sigma_{\ell-p+1}}{\sigma_j}\right)^{4q+2}}},$$

and

$$\left\|\left(I-QQ^T\right)A\right\|_F \leq \|A-QB_k\|_F \leq \sqrt{\left(\sum_{j=k+1}^{n}\sigma_j^2\right)+k\mathcal{C}_\Delta^2\sigma_{\ell-p+1}^2\left(\frac{\sigma_{\ell-p+1}}{\sigma_k}\right)^{4q}},$$

$$\left\|\left(I-QQ^T\right)A\right\|_2 \leq \|A-QB_k\|_2 \leq \sqrt{\sigma_{k+1}^2+k\mathcal{C}_\Delta^2\sigma_{\ell-p+1}^2\left(\frac{\sigma_{\ell-p+1}}{\sigma_k}\right)^{4q}}.$$

with exception probability at most Δ.

REMARK 5.8. Like the average case, the factor $\sigma_{\ell-p+1}^2$ shows up in all three bounds, for all $q \geq 0$. Hence Algorithm 1.1 can make significant progress toward convergence in case of rapidly decaying singular values in A, with probability $1-\Delta$. This is clearly a much stronger result than Theorem 1.2.

REMARK 5.9. While the value of Δ could be set arbitrarily tiny, it can never be set to 0. This implies that there is a chance, however arbitrarily small, that Algorithm 2.2 might not converge according to the bounds in Theorem 5.8. This small exception chance probably has less to do with Algorithm 2.2 and more to do with the inherent complexity of efficiently computing accurate matrix norms. Since Algorithm 2.2 accesses A only through the $2q+2$ matrix-matrix products of the form AX or A^TY for different X and Y matrices, it can be used to efficiently compute $\|A^{-1}\|_2$ (setting $k=1$) provided that a factorization of A is available or if A is itself a non-singular triangular matrix. On the other hand, it is generally expected that even estimating $\|A^{-1}\|_2$ to within a constant factor independent of the matrix A must cost as much, asymptotically, as computing A^{-1}. Demmel, Diament, and Malajovich [20] show that the cost of computing an estimate of $\|A^{-1}\|$ of guaranteed quality is at least the cost of testing whether the product of two $n \times n$ matrices is zero, and performing this test is conjectured to cost as much as actually computing the product [41, p. 288]. Since Algorithm 2.2 costs only $O(n^2q\ell)$ operations to provide a good estimate for $\|A^{-1}\|_2$, it probably can not be expected to work without *any* failure. See Section 7 for more comments.

Proof of Theorem 5.8: Since $\widehat{\Omega}_2$ and $\widehat{\Omega}_1$ are independent from each other, we can study how the error depends on the matrix $\widehat{\Omega}_2$ when $\widehat{\Omega}_1$ is reasonably bounded. To this end, we define an event as follows:

$$\mathbf{E}_t = \left\{\widehat{\Omega}_1 : \left\|\widehat{\Omega}_1^\dagger\right\|_2 \leq t\mathcal{L}\right\}, \quad \text{where} \quad \mathcal{L} = \frac{e\sqrt{\ell}}{p+1}.$$

Invoking the conclusion of Lemma 5.2, we find that

$$\mathbb{P}\left(\mathbf{E}_t^c\right) \leq t^{-(p+1)}. \tag{5.8}$$

In other words, we have just shown that $\left\|\widehat{\Omega}_1^\dagger\right\|_2 \leq t\mathcal{L}$ with probability at least $1-t^{-(p+1)}$.

Below we consider the function

$$h(X) = \|X\|_2\left\|\widehat{\Omega}_1^\dagger\right\|_2,$$

where X has the same dimensions as $\widehat{\Omega}_2$. It is straightforward to show that

$$|h(X)-h(Y)| \leq \left\|\widehat{\Omega}_1^\dagger\right\|_2\|X-Y\|_F \leq t\mathcal{L}\|X-Y\|_F,$$

under event $\mathbf{E}_t$. Also under event $\mathbf{E}_t$ and by Proposition 5.1, we have

$$\mathbb{E}h(X) \le \left(\sqrt{n-\ell+p}+\sqrt{\ell}\right)\left\|\widehat{\Omega}_1^\dagger\right\|_2 \le \frac{et\sqrt{\ell}}{p+1}\left(\sqrt{n-\ell+p}+\sqrt{\ell}\right) \stackrel{def}{=} t\mathcal{E}.$$

Applying the concentration of measure equation, Theorem 5.3, conditionally to $\widehat{\Omega}_2$ under event $\mathbf{E}_t$,

$$\mathbb{P}\left\{\left\|\widehat{\Omega}_2\right\|_2\left\|\widehat{\Omega}_1^\dagger\right\|_2 \ge t\mathcal{E}+t\mathcal{L}u \;\middle|\; \mathbf{E}_t\right\} \le e^{-u^2/2}.$$

Use the equation (5.8) to remove the restriction on $\widehat{\Omega}_1$, therefore,

$$\mathbb{P}\left\{\left\|\widehat{\Omega}_2\right\|_2\left\|\widehat{\Omega}_1^\dagger\right\|_2 \ge t\mathcal{E}+t\mathcal{L}u\right\} \le t^{-(p+1)}+e^{-u^2/2},$$

Now we choose

$$t=\left(\frac{2}{\Delta}\right)^{1/(p+1)} \quad \text{and} \quad u=\sqrt{2\log\frac{2}{\Delta}}$$

so that $t^{-(p+1)}+e^{-u^2/2}=\Delta$. With this choice of t and u,

$$t\mathcal{E}+t\mathcal{L}u=\mathcal{C}_\Delta \quad \text{or} \quad \mathbb{P}\left\{\left\|\widehat{\Omega}_2\right\|_2\left\|\widehat{\Omega}_1^\dagger\right\|_2 \ge \mathcal{C}_\Delta\right\} \le \Delta.$$

Plugging this bound into the formulas in Theorem 4.3 and Remark 4.3 proves Theorem 5.8. **Q.E.D.**

While the value of oversampling size p does not look so important in the average case error bounds as long as $p \ge 2$, it makes an oversized difference in large deviation bounds. Consider the case $p=2$ with a tiny $\Delta > 0$. In this case, $\mathcal{C}_\Delta$ may still be quite large, and quite a few extra number of iterations might be necessary to ensure satisfactory convergence with small exception probability.

For $p \le 1$, the large deviation bound is brutal. For very small values of k, such as 1 in the case of the randomized power method (see Algorithm A.3), it seems unreasonable to require a relatively large value of p. On the other hand, a small p value would significantly impact convergence. We will address this conflicting issue of choosing p further in Section 8.

But for any large enough values of k (such as $k=20$ or more, for example,) a reasonable choice would be to choose p so $\left(\frac{2}{\Delta}\right)^{1/(p+1)}$ is a modest number. We will now choose

$$p=\lceil \log_{10}\left(\frac{2}{\Delta}\right)\rceil - 1. \tag{5.9}$$

This choice gives $\left(\frac{2}{\Delta}\right)^{1/(p+1)} \le 10$. For a typical choice of $\Delta = 10^{-16}$, equation (5.9) gives $p=16$. For this value of Δ, the exception probability is smaller than that of matching DNA fingerprints [65]. Given that the "random numbers" generated on modern computers are really only *pseudo random numbers* that may have quite different upper tail distributions than the true Gaussian (see, for example [78, 80]), and given that only finite precision computations are typically done in practice, it is probably meaningless to require Δ to be much less than 10^{-16}, the double precision. Additionally, with this choice of p, the large deviation bounds are very similar to the average case error bounds, suggesting that the typical behavior is also the worst case behavior, with probability $1-\Delta$.

Our final observation on Theorem 5.8 is so important that we present it in the form of a Corollary. We will not prove it because it is a direct consequence.

COROLLARY 5.9. *In the notation of Theorem 5.8, we must have for* $j = 1, \cdots, k$,

$$\sigma_j\left(QB_k\right)) \geq \frac{\sigma_j}{\sqrt{1+\mathcal{C}_\Delta^2}} \quad \textit{and} \quad \|A - QB_k\|_2 \leq \sigma_{k+1}\sqrt{1+k\mathcal{C}_\Delta^2} \tag{5.10}$$

with exception probability at most Δ.

This is a surprisingly strong result. We will discuss its implications in terms of rank-revealing factorizations in Section 6 and condition number estimation in Section 7.

6. Rank-revealing Factorizations. Rank-revealing factorizations were first discussed in Chan [12]. Generally speaking, there are rank-revealing UTV factorizations [26, 77], QR factorizations [13, 14, 33], and LU factorizations [60, 63]. While there is no uniform definition of *the* rank-revealing factorization, a comparison of different forms of rank-revealing factorizations has appeared in Foster and Liu [27]. For the discussions in this section, we make the following definition, which is loosely consistent with those in [27].

DEFINITION 6.1. Given $m \times n$ matrices A and B and integer $k < \min(n, m)$, we call B a *rank-revealing rank-k approximation* to A if $\mathbf{rank}(B) \leq k$ and if there exist polynomials $c_1(m, n)$, and $c_2(m, n)$ such that

$$\sigma_j(B) \geq \frac{\sigma_j(A)}{c_2(m,n)}, \quad j = 1, \cdots, k, \tag{6.1}$$

$$\|A - B\|_2 \leq c_1(m, n)\sigma_{k+1}(A). \tag{6.2}$$

A *rank-revealing rank-k approximation* differs from an ordinary rank-k approximation in the extra condition (6.1), which requires some accuracy in *all* k leading singular values. Therefore a rank-revealing rank-k approximation is likely a stronger approximation than a simple low rank approximation. To see why (6.1) is so important, we consider for an example the case where the leading $k+1$ singular values of A are identical: $\sigma_1(A) = \cdots = \sigma_{k+1}(A)$. This includes the $n \times n$ identity matrix as a special case. Now choose $\theta = 1$ in equation (1.4). It follows that $B = 0$ is an *optimal* rank-k approximation to A, which is likely unacceptable to most users. On the other hand, $B = 0$ obviously does not satisfy condition (6.1) for any polynomial $c_2(m, n)$, and therefore is not a rank-revealing rank-k approximation to A. Similarly, *any* orthogonal matrix Q would satisfy the bound in Theorem 1.2 for such an A matrix, and only the matrix Q from Algorithm 2.2 would satisfy Theorem 5.8.

By definition, Algorithm 2.2 produces a rank-revealing rank-k approximation with probability at least $1-\Delta$. In this section, we compare this approximation with the strong RRQR factorization developed in Gu and Eisenstat [33].

THEOREM 6.1. *(Gu and Eisenstat [33]) Let* A *be an* $m \times n$ *matrix and let* $1 \leq k \leq \min(m, n)$. *For any given parameter* $f > 1$, *there exists a permutation* Π *such that*

$$A\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix},$$

where for any $1 \leq i \leq k$ *and* $1 \leq j \leq n - k$,

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})}, \frac{\sigma_j(R_{22})}{\sigma_{k+j}} \leq \sqrt{1 + f^2k(n-k)}. \tag{6.3}$$

Let $\widehat{A}_k = Q \begin{pmatrix} R_{11} & R_{12} \\ & 0 \end{pmatrix} \Pi^T$. Then $\widehat{A}_k$ is a rank-k matrix. It follows from equation (6.3) that

$$\sigma_j\left(\widehat{A}_k\right) \geq \frac{\sigma_j}{\sqrt{1+f^2k(n-k)}}, \quad j = 1, \cdots, k,$$

$$\left\| A - \widehat{A}_k \right\|_2 \leq \sigma_{k+1} \sqrt{1 + f^2 k(n-k)}.$$

These properties are compatible with the inequalities in Theorem 5.8. The strong RRQR factorization in Theorem 6.1 also includes a permutation Π that selects k linearly independent columns of A such that $\left\| R_{11}^{-1} R_{12} \right\|_2 \leq f$. Such information could be useful in some applications [59].

But the matrix QB_k, being a two-sided orthogonal approximation, does not contain any information about such permutation. On the other hand, it is likely to be cheaper to compute due to the matrix-matrix product operations involved, and for rapidly decaying singular values or by potentially increasing the value of q, it could make a much better approximation than $\widehat{A}_k$.

7. Condition Number Estimation. For any given square non-singular matrix A, define

$$\kappa(A) = \|A\| \|A^{-1}\|,$$

as its condition number. Here $\|\cdot\|$ is any matrix norm, such as the matrix 1-norm, 2-norm, ∞-norm, Frobenius norm, or max-norm. Condition numbers are of central importance in solving many matrix computation problems, such as linear equations, least squares problems, eigenvalue/eigenvector problems, and sparse matrix problems. For a detailed discussion of condition number estimation, see the survey paper by Higham [38] and the references therein. More recent work includes Laub and Xia [51].

A typical condition estimator uses a matrix norm estimator to estimate $\|A\|$ and $\|A^{-1}\|$ separately, and multiply them together to get an estimate for $\kappa(A)$. A typical matrix norm estimator, in turn, only accesses the matrix A through matrix-matrix or matrix-vector multiplications, without the need to directly access entries of A. Thus the costs of estimating $\|A\|$ and $\|A^{-1}\|$ are similar if a factorization for A is available. The goal in matrix norm estimation is to compute a reliable estimate of $\|A\|$ up to a factor that does not grow too fast with the dimension of A, perhaps without direct access to entries of A, at a cost that is considerably less than that of matrix factorization or inversion, something that is believed to be impossible (see Remark 5.9.)

However, by Corollary 5.9, we know Algorithm 2.2 does compute a reliable estimate for $\|A\|_2$ with $k = 1$ and a reasonable choice of $\ell > 1$, due to the randomization of the start matrix. Below we concentrate on estimating $\|A\|_1$. Currently, Hager's method is one of the most popular estimators for $\|A\|_1$, is the default 1-norm estimator of LAPACK [1, 35, 38, 39]. Hager's method is based on a variant of the gradient descent method to find a local maximizer for the following optimization problem:

$$\|A\|_1 = \max_{x \in \mathcal{S}} \|Ax\|_1, \quad \text{where} \quad \mathcal{S} = \{x \in \mathbf{R}^n : \|x\|_1 \leq 1.\} \tag{7.1}$$

ALGORITHM 7.1. **Hager's Method**

Input: *$m \times n$ matrix A, and initial 1-norm unit vector x.*
Output: *An estimate for $\|A\|_1$.*

---

repeat

1. *Compute $y = Ax$, $z = A^T \mathbf{sign}(y)$.*
2. **if** $\|z\|_\infty \leq z^T x$ **then**
   return $\gamma = \|y\|_1$.
3. $x = e_j$, *where* $j = \mathbf{argmax}_k |z_k|$.

The e_j is the j-th unit vector. While it could occasionally take much longer, Hager's method typically takes very few (less than 5) iterations to converge to a local maximum that is within a reasonable factor (like 10 or less) of $\|A\|_1$. As Algorithm 2.2 already computes a reliable estimate for $\|A\|_2$, it is straightforward to combine Algorithms 2.2 and 7.1 to obtain a reliable estimate for $\|A\|_1$, which satisfies $\|A\|_1 \geq \|A\|_2/sqrtn$.

ALGORITHM 7.2. **Randomized Hager's Method**

Input: *$m \times n$ matrix A, and integer $\ell > 1$.*
Output: *An estimate for $\|A\|_1$.*

---

1. *Compute rank-1 approximation QB_1 to A using Algorithm 2.2*
2. *Set $\widehat{u}$ to be the right singular vector of QB_1.*
3. *Run Algorithm 7.1 on A with initial vector $x = \widehat{u}/\|\widehat{u}\|_1$.*
4. *Return γ from Algorithm 7.1.*

Since QB_1 is a rank-1 matrix, $\widehat{u}$ is straightforward to compute. The number of iterations in Algorithm 7.1 can be restricted to as few as 1 or 2. This is because Algorithm 7.1 is only used to find a column whose vector 1-norm provides the estimate for $\|A\|_1$, no local maximum to problem (7.1) is necessary. Corollary 7.1 directly follows from Corollary 5.9.

COROLLARY 7.1. *For any $0 < \Delta \ll 1$, the output γ from Algorithm 7.2 must satisfy*

$$\gamma \geq \frac{\|A\|_1}{\sqrt{n}\sqrt{1+\widehat{\mathcal{C}}_\Delta^2}} \quad \textit{where} \quad \widehat{\mathcal{C}}_\Delta = \frac{e}{\sqrt{\ell}}\left(\frac{2}{\Delta}\right)^{\frac{1}{\ell}}\left(\sqrt{n}+\sqrt{\ell}+\sqrt{2\log\frac{2}{\Delta}}\right),$$

with exception probability at most Δ.

REMARK 7.1. One probably does not need to choose a very tiny Δ for matrix norm estimation. In our numerical experiments, $\ell = 5$ worked very well. For matrices of dimension up to 200, Algorithm 7.2 never under-estimated the true norm by a factor over 10. In general, we can choose $\ell = \lceil \log_2\left(\frac{2}{\Delta}\right) \rceil$, in which case the constants $\widehat{\mathcal{C}}_\Delta$ and γ above satisfy

$$\widehat{\mathcal{C}}_\Delta < 2e\left(\sqrt{\frac{n}{\ell}}+3\right) \quad \text{and} \quad \gamma \geq \frac{\|A\|_1}{2e\sqrt{n}\left(\sqrt{\frac{n}{\ell}}+4\right)}.$$

REMARK 7.2. Hager's method has been generalized by Higham [40] to estimate the matrix p-norm for any $p \geq 1$ and the mixed matrix norm $\|A\|_{\alpha,\beta}$ for $\alpha \geq 1$ and $\beta \geq 1$. In particular, the max-norm is the special case with $\alpha = \infty$ and $\beta = 1$. Algorithm 7.2 can be trivially generalized to those cases as well, by replacing Hager's method in Algorithm 7.2 with its generalized version, leading to a Corollary 7.1-like conclusion for reliability. We omit the details.

REMARK 7.3. Kuczyński and Woźniakowski [46] developed probabilistic error bounds for estimating the condition number using the Lanczos algorithm for unit start vectors under the uniform distribution. However, our results appear to be much stronger.

Below, we demonstrate the robustness of Algorithm 7.2 through the following example. Let

$$A = \begin{pmatrix} \alpha & b^T \\ b & \rho E \widehat{A} E \end{pmatrix}, \quad \text{for} \quad E = I - \frac{1}{n-1} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}^T,$$

where $\alpha > 0, \rho > 0$ are scalars, $b > 0$ is an $n-1$ dimensional vector, and $\widehat{A}$ is an $(n-1) \times (n-1)$ matrix. If we take the initial vector x in Algorithm 7.1 to be the vector of all 1's (the default choice in LAPACK), then Algorithm 7.1 will always return $\alpha + \|b\|_1$ as the 1-norm estimate, regardless of $\rho\widehat{A}$.

In our numerical experiment, we set $n = 100$, $\rho = 10^{10}$ and chose α, b and $\widehat{A}$ to be random, with $\|A\|_1 \approx 8.35 \times 10^{11}$. For $\ell = 5$, we obtained $\|A\|_1 \approx 2.46 \times 10^{11}$ from Algorithm 7.2. On the other hand, Algorithm 7.1 returned $\|A\|_1 \approx 4.72 \times 10^1$, which was completely wrong.

8. Numerical Experiments. In this section we perform numerical experiments to shed more light on randomized algorithms. Our main purpose of these experiments is to provide numerical support to our probabilistic analysis and to demonstrate that different applications can lead to different singular value distributions in the matrix and impose different accuracy requirements, and thus demand different levels of computational effort on the randomized algorithms.

8.1. Improved Randomized Power Iteration. In the case of a small k, it seems unreasonable to require a potentially large value of p as suggested in equation (5.9). However, for a truely small value of p, going random is still not enough to overcome the potential problem of slow convergence associated with a poor start matrix in Algorithm 2.2, and some additional work maybe needed (see Sections 5.)

This discussion is particularly relevant for $k = 1$, which corresponds to the classical power method, Algorithm A.2, and its randomized version, Algorithm A.3, in Appendix A. Any value of $p > 0$ seems to be too much work, but $p = 0$ does not lead to fast enough convergence.

According to Corollary 5.9, Algorithm 2.2 can already compute order of magnitude approximations to all the leading singular values with $q = 0$. Thus, an obvious improvement of Algorithm 2.2 for small values of k would be to compute Ω with Algorithm 1.1 and then compute a subspace approximation with Algorithm 2.1. Algorithm 8.1 below is designed for subspace computations where $k = O\left(\lceil \log_{10}\left(\frac{2}{\Delta}\right) \rceil\right)$ or smaller.

ALGORITHM 8.1. **Improved Randomized Subspace Iteration for small k**

Input: *$m \times n$ matrix A with $n \leq m$, integers q and $\ell_1 > \ell_2 \geq k$.*
Output: *a rank-k approximation.*

---

1. *Run Algorithm 1.1 with $\ell = \ell_1$ for a rank-ℓ_2 approximation.*
2. *Set Ω to be approximate right singular vector matrix.*
3. *Run Algorithm 2.1 with Ω and $\ell = \ell_2$ for a rank-k approximation.*

We perform our experiments with 4000×4000 matrices of the form

$$A = \left(\log \|X_i - Y_j\|_2\right),$$

where $\{X_i\}$ are n-dimensional Gaussian random variables with mean 0 and standard deviation 1, and where $\{Y_j\}$ are n-dimensional Gaussian random variables with mean μ and standard deviation 1. We choose different μ values to control the ratio of the two leading singular values of A.

We ran Algorithm 8.1 with $\ell_1 = 5$ and $\ell_2 = k = 1$. We also ran Randomized Power Method, Algorithm A.3, to compute $\|A\|_2$. We choose $\mu = 1$ for a large σ_2/σ_1 ratio and $\mu = 2.5$ for a small ratio. The results are summarized in Figure 8.1.

For the case of large σ_2/σ_1 ratio, Algorithm 8.1 converged to $\|A\|_2$ in about 250 steps, as opposed to about 350 steps for Algorithm A.3. For the case of a small σ_2/σ_1 ratio, both algorithms performed equally well. Algorithm 8.1 converged slightly more quickly, but that is offset by the extra work needed to compute the initial Ω.

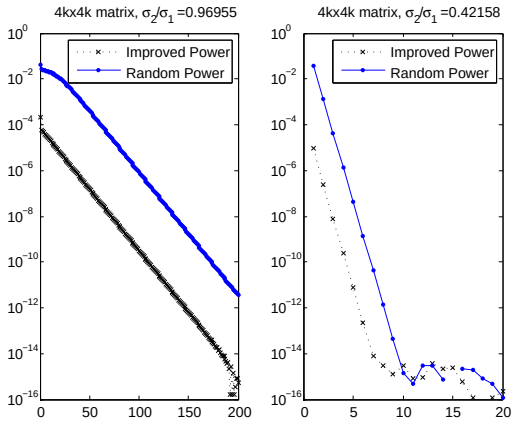

FIG. 8.1. *Faster Convergence of Algorithm 8.1 due to Better Choice of Start Vector*

Figure 8.1 confirms our analysis. At the cost of the initial step to obtain a good start vector, Algorithm 8.1 can converge significantly faster than Algorithm A.3.

8.2. low-rank approximation. In this experiment, we consider a 4000×4000 matrix of the form

$$A = \left(\log \|X_i - Y_j\|_2\right),$$

where $\{X_i\}$ are equi-spaced points on the edge of the disc $\|X - \begin{pmatrix} -1 \\ -1 \end{pmatrix}\|_2 = \sqrt{2}$ and $\{Y_j\}$ equi-spaced points on the edge of the disc $\|Y - \begin{pmatrix} 2 \\ 2 \end{pmatrix}\|_2 = 2\sqrt{2}$ (see Figure 8.2.) We compare the performance of Algorithms 1.1 and 2.2 against that of `svds`, the matlab version of ARPACK [52] for finding a few selected singular values of large matrices. We choose $k = 50$. The results are summarized in Table 8.1.

TABLE 8.1
Numbers of Matrix-Vector Multiplies

Tolerance	$q = 0$	$q = 2$	$q = 4$	svds
10^{-6}	143	5×96	9×79	500
10^{-8}	180	5×96	9×87	600
10^{-10}	190	5×96	9×93	600

Since the singular values of this matrix decay relatively quickly, Algorithm 1.1 seems to out-perform Algorithm 2.2 for any values of $q > 0$. Algorithm 1.1 also outperforms `svds`. As Algorithm 1.1 mostly computes

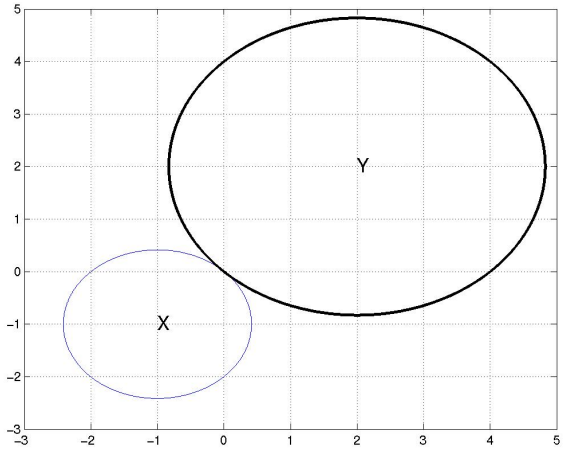

FIG. 8.2. *X and Y points in A*

matrix-matrix products whereas each step of svds involves a matrix-vector product, we would expect Algorithm 1.1 to have even better performance than `svds` on modern serial and parallel architectures. This example demonstrates that for matrices with fast decaying singular values, randomized algorithms can be as competitive as the best methods for computing highly accurate low-rank approximations.

8.3. Structured Matrix Computations. In this example, we demonstrate the effectiveness of randomized algorithms for low-rank approximation in the context of structured matrix computations. $\texttt{G3}_{\texttt{circuit}}$ is a 1585478×1585478 sparse SPD matrix arising from circuit simulations. It is publicly available in the *University of Flordia Sparse Matrix Collection* [18]. Figure 8.3 depicts its sparsity pattern in the symmetric minimum degree ordering [30]. A direct factorization of this matrix creates a large amount of fill-in. In particular, the Schur complement of the leading 1582178×1582178 principal submatrix, to be called A, is a 3300×3300 dense submatrix. Here we compute hierarchical semiseparable (HSS) preconditioners to A with the techniques in [53] and report the numbers of preconditioned conjugate gradient (PCG) steps to iteratively solve for a linear system of equations $Ax = b$ for a random right hand side b. The PCG is a very popular technique for solving large SPD systems of equations [37, 2]. We refer the reader to [53, 58] for details about the HSS matrix structure and its numerical construction, but emphasize that the key and most time-consuming step for computing HSS preconditioners is to approximate various off-diagonal blocks of the matrix A by matrices of rank k or less. We choose convergence tolerance $\delta = 10^{-12}$. The conjugate gradient method (CG) without any preconditioning takes 878 iterations to reduce the residual below this tolerance.

TABLE 8.2
Numbers of PCG Iterations

Maximum off-diagonal rank k	$p = 10$	$p = 20$	$p = 40$
20	75	77	72
40	69	69	69
60	64	61	61

Table 8.2 summarizes our results. We can see that all choices of p drastically decrease the number of CG iterations. Howver, the additional reduction in the number of CG iterations is typically small for higher values of p. Considering the extra cost involved in higher p values in the construction of HSS preconditioners, it seems that higher p values are ineffective for this application. This example suggests that for the purpose

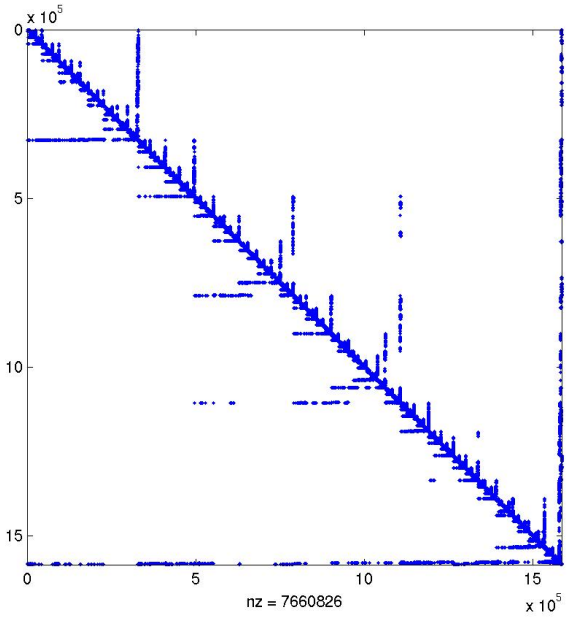

FIG. 8.3. *A Sparse Matrix in Symmetric Minimum Degree Ordering*

of constructing preconditioners in structured matrix computations, a small p value is typically sufficient to develop highly effective preconditioners. This is consistent with the rule of thumb that typically randomized algorithms require very little oversampling and a value of p in between 10 to 20 suffices [58, 59, 67].

TABLE 8.3
Comparison of Numbers of Incorrect Matches

Rank k	$p = 10$	$p = 20$	$p = 40$	Truncated SVD
10	32	25	23	24
20	25	26	25	21
30	21	20	18	17
40	20	17	17	16

8.4. Eigenfaces. Eigenfaces is a well studied method of face recognition based on principal component analysis (PCA), popularised by the seminal work of Turk and Pentland [79]. For more recent work and survey, see [8, 49, 74, 75, 76] and the references therein. In this experiment we demonstrate the effects of randomized algorithms on face recognition.

Typical face recognition starts with a data base of training images, which are then processed as follows:

1. Calculate the mean of the training images.
2. Subtract the mean from the training images, obtaining the mean-shifted images.
3. Calculate a truncated SVD of the mean-shifted images.
4. Project the mean-shifted images into the singular vector space using the retained singular vectors, obtaining feature vectors.

To classify a new face, one does the following calculations:

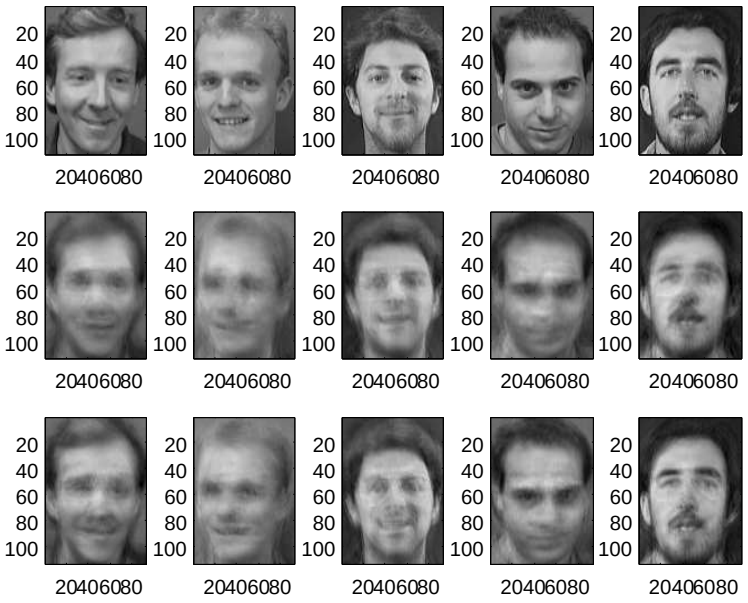

FIG. 8.4. *Original Faces and their Eigenfaces*

1. Subtract the mean from the new image, obtaining the mean-shifted image.
2. Project the mean-shifted image into the singular vector space, obtaining a new feature vector.
3. Find the feature vector in the data base that best matches the new feature vector.

Our face data are obtained from the Database of Faces maintained at the AT&T Laboratories Cambridge [11]. All faces are greyscale images with a consistent resolution. There are ten different images of each of 40 distinct subjects. The size of each image is 92×112 pixels, with 256 grey levels per pixel. We use 200 of these images, 5 from each individual, as training images, and the remaining ones for classification.

In Figure 8.4, the first row are the original face images; the second row are eigenfaces with a rank-10 truncated SVD, and the third row eigenfaces with a rank-20.

In addition to the exact truncated SVD, we also perform image training and classification using Algorithm 1.1 with different p values. The results are summarized in Table 8.3. It is clear that smaller p values give worse results than truncated SVD, but $p = 40$ gives results that are very similar to truncated SVD, even though some of the singular values are accurate to only within 1 to 2 digits. This example demonstrates that limited accuracy that goes beyond being correct to within a constant factor is sufficient for some applications.

9. Conclusions and Future Work. We have presented some interesting results on randomized algorithms within the framework of the subspace iteration method for singular value and low-rank matrix approximations. While randomized algorithms have been primarily considered as an efficient tool to compute low-rank approximations, our results further suggest that they actually compute the much stronger rank-revealing factorizations, and can be used to reliably estimate condition numbers. We have also presented numerical experimental results that support our analysis.

This work opens up many directions for future research. Most immediate is the convergence analysis on singular vectors. We expect results compatible to those for singular values. Variations of subspace iteration methods exist for computing eigenvalues of symmetric and non-symmetric matrices. It would be interesting to extend our results to these methods.

Acknowledgments. The author would like to thank Shengguo Li, Michael Mahoney, Vladimir Rokhlin, Mark Tygert, Jianlin Xia and Chao Yang for many helpful discussions on this subject. He would especially like to thank Joel Tropp, whose interesting talk at UC Berkeley in the Spring of 2010 sparked the author's interest on the subject that eventually led to this work, and Chris Melgaard, with whom he had extensive discussions about the material presented in this work. Finally, the author would like to thank the anonymous referees who go out of their ways to provide numerous helpful suggestions that greatly improved the presentation of this paper, including a shorter proof for Theorem 3.4.

Appendix. For numerical stability, Algorithm A.1 below is often performed once every few iterations in subspace iteration methods, to balance efficiency and numerical stability (see Saad [69].)

ALGORITHM A.1. **Orthorgonalization with QR**

Input: $m \times n$ *matrix* A, $n \times \ell$ *start matrix* Ω, *and integer* $q \geq 0$.
Output: $Y = \left(AA^T\right)^q A\Omega$.

Compute $Y = A\Omega$, *and* QR *factorize* $QR = Y$.
for $i = 1, \cdots, q$ **do**
  $Y = A^T Q$; QR *factorize* $QR = Y$;
  $Y = AQ$; QR *factorize* $QR = Y$.
endfor

Below is the classical power method for computing the 2-norm of a given matrix.

ALGORITHM A.2. **Basic Power Method**

Input: $m \times n$ *matrix* A *with* $n \leq m$,
and $n \times 1$ *start vector* Ω.
Output: *approximation to* $\|A\|_2$.

---

1. *Compute* $Y = \left(AA^T\right)^q A\,\Omega$.
2. *Compute an orthogonal column basis* Q *for* Y.
3. *Compute* $B = Q^T A$.
4. *Return* $\|B\|_2$.

In situations where no useful information about the leading right singular vector is available, the vector Ω in Algorithm A.2 can also be chosen to be random, to enhance convergence, leading to

ALGORITHM A.3. **Randomized Power Method**

Input: $m \times n$ *matrix* A *with* $n \leq m$,
Output: *approximation to* $\|A\|_2$.

1. *Draw a random* $n \times 1$ *vector* Ω.
2. *Compute* $Y = \left(AA^T\right)^q A\,\Omega$.
3. *Compute an orthogonal column basis* Q *for* Y.
4. *Compute* $B = Q^T A$.
5. *Return* $\|B\|_2$.

REFERENCES

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, second edition, 1994.

[2] O. Axelsson and L. Yu. Kolotilina. *Preconditioned Conjugate Gradient Methods*. Springer Verlag, Berlin, 1990.

[3] Z.-J. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst. *Templates for the solution of Algebraic Eigenvalue Problems*. SIAM, Philadelphia, PA, 2000.

[4] K.-J. Bathe and E. L. Wilson. *Numerical Methods in Finite Element Analysis*. Prentice Hall, Englewood Cliffs and NJ, 1976.

[5] M. W. Berry, S. T. Dumais, and G. W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37:575–595, 1995.

[6] V. Bogdanov. *Gaussian Measures*. American Mathematical Society, Providence, RI, 1998.

[7] C. Boutsidis, P. Drineas, and M. W. Mahoney. An improved approximation algorithm for the column subset selection problem. *arXiv preprint arXiv:0812.4293v2*, 2008.

[8] R. Brunelli and T. Poggio. Face recognition: Features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052, 1993.

[9] D. Cai. Text datasets in matlab format. http://www.zjucadcg.cn/dengcai/Data/TextData.html, 2009.

[10] D. Calvetti, L. Reichel, and D.C. Sorensen. An implicitly restarted Lanczos method for large symmetric eigenvalue problems. *ETNA*, 2:1–21, 1994.

[11] AT&T Laboratories Cambridge. Database of faces. http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html, 2002.

[12] T. F. Chan. Rank revealing QR factorizations. *Lin. Alg. Appl.*, 88/89:67–82, 1987.

[13] T. F. Chan and P. C. Hansen. Some applications of the rank revealing QR factorization. *SIAM J. Sci. Stat. Comput.*, 13:727–741, September 1992.

[14] S. Chandrasekaran and I. Ipsen. On rank-revealing QR factorizations. *SIAM J. Matrix Anal. Appl.*, 15:592–622, 1994.

[15] Z. Chen and J. Dongarra. Condition numbers of Gaussian random matrices. *SIAM J. Matrix Anal. Appl.*, 27:603–620, 2005.

[16] H. Cheng, Z. Gimbutas, P.-G. Martinsson, and V. Rokhlin. On the compression of low rank matrices. *SIAM J. Sci. Comput.*, 26:1389–1404, 2005.

[17] J. Cullum and R. A. Willoughby. *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. I: Theory*. SIAM, Philadelphia, PA, 2002.

[18] T. Davis. University of Florida sparse matrix collection. http://www.cise.ufl.edu/research/sparse/matrices.

[19] J. Demmel. *Applied Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997.

[20] J. W. Demmel, B. Diament, and G. Malajovich. On the complexity of computing error bounds. *Found. Comp. Math.*, 1:101–125, 2001.

[21] P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices, II. computing a low-rank approximation. *SIAM J. Comput.*, 36:158–183, 2006.

[22] P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Subspace sampling and relative-error matrix approximation: Column-based methods. In J. Diaz and *et al.*, editors, *Approximation, Randomization, Combinatorial Optimization*, volume 4110 of *LNCS*, pages 321–326, Berlin, 2006. Springer.

[23] P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM J. Matrix Anal. Appl.*, 30:844–881, 2008.

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

[25] R. D. Fierro and P. C. Hansen. Low-rank revealing UTV decompositions. *Numerical Algorithms*, 15:37–55, 1997.

[26] R. D. Fierro and P. C. Hansen. Low-rank revealing UTV decompositions. *Numerical Algorithms*, 15:37–55, 1997.

[27] L. V. Foster and X. Liu. Comparison of rank revealing algorithms applied to matrices with well defined numerical ranks. http://www.researchgate.net/publication/228523390_Comparison_of_rank_revealing_algorithms_applied_to_matrices_with_well_defined_nu

[28] A. Frieze, R. Kannan, and S. Vempala. Fast Monte Carlo algorithms for finding low-rank approximations. In *Proc. 39th*

Ann. IEEE Symp. Foundations of Computer Science (FOCS), pages 370–378, 1998.
[29] A. Frieze, R. Kannan, and S. Vempala. Fast Monte Carlo algorithms for finding low-rank approximations. *J. Assoc. Comput. Mach.*, 51:1025–1041, 2004.
[30] A. George and J. Liu. The evolution of the minimum degree ordering algorithm. *SIAM Review*, 31:1–19, 1989.
[31] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, 3nd edition, 1996.
[32] R. G. Grimes, J. G. Lewis, and H. D. Simon. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *SIAM J. Matrix Anal. Appl.*, 15:228–272, 1994.
[33] M. Gu and S. C. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM J. Sci. Comput.*, 17:848–869, 1996.
[34] W. Hackbusch. A sparse matrix arithmetic based on ⟨-matrices. part I: introduction to ⟨-matrices. *Computing*, 62:89–108, 1999.
[35] W. W. Hager. Condition estimators. *SIAM J. Sci. Stat. Comput.*, 5:311–316, 1984.
[36] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53:217–288, 2011.
[37] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Natl. Bur. Stand.*, 49:409–436, 1954.
[38] N. J. Higham. A survey of condition number estimation for triangular matrices. *SIAM Review*, 29:575–596, 1987.
[39] N. J. Higham. Experience with a matrix norm estimator. *SIAM J. Sci. Stat. Comput.*, 11:804–809, 1990.
[40] N. J. Higham. Estimating the matrix p-norm. *Numer. Math.*, 62:539–555, 1992.
[41] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 1996.
[42] A. J. Hoffman and H. W. Wielandt. The variation of the spectrum of a normal matrix. *Duke Mathematics*, 20:37–39, 1953.
[43] P. Hong and C.-T. pan. The rank revealing QR decomposition and SVD. *Math. Comp.*, 58:213–232, 1992.
[44] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University press, 1991.
[45] S. Van Huffel and H. Zha. An efficient total least squares algorithm based on a rank revealing two-sided orthogonal decomposition. *Numerical Algorithms*, 4:101–133, 1993.
[46] H. Woźniakowski J. Kuczyński. Probabilistic bounds on the extremal eigenvalues and condition number by the lanczos algorithm. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.34.4243.pdf.
[47] H. Woźniakowski J. Kuczyński. Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start. Dept. of Computer Science Report CUCS-465-89, University of Columbia, 1989.
[48] I. T. Jolliffe. *Principal Component Analysis*. Springer Verlag, New York, 1986.
[49] M. Kirby and L. Sirovich. Application of the Karhunen-Loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):103–108, 1990.
[50] E. Kokiopoulou, C. Bekas, and E. Gallopoulos. Computing smallest singular triplets with implicitly restarted Lanczos bidiagonalization. *Appl. Numer. Math.*, 49:39–61, 2004.
[51] A. Laub and J. Xia. Rapplications of statistical condition estimation to the solution of linear systems. *Numerical Linear Algebra with Applications*, 15:489–513, 2008.
[52] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, Philadelphia, PA, 1998.
[53] S. Li, M. Gu, C. J. Wu, and J. Xia. New efficient and robust HSS cholesky factorization of spd matrices. *SIAM J. Matrix Anal. Appl.*, 33:886–904, 2012.
[54] E. Liberty. *Accelerated dense random projections*. PhD thesis, Department of Computer Science, Yale University, 2009.
[55] E. Liberty, N. Ailon, and A. Singer. Dense fast random projections and lean walsh transforms. In A. Goel, K. Jansen, J. Rolim, and R. Rubinfeld, editors, *Approximation and Randomization and Combinatorial Optimization*, volume 5171 of *Lecture Notes in Computer Science*, pages 512–522, Berlin, 2008. Springer.
[56] E. Liberty, F. F. Woolfe, V. Rokhlin P.-G. Martinsson, and M. Tygert. Randomized algorithms for the low-rank approximation of matrices. *Proc. Natl. Acad. Sci. USA*, 104:2016–2017, 2007.
[57] M. Mahoney. Randomized algorithms for matrices and data. http://arxiv.org/abs/1104.5557, 2011.
[58] P. G. Martinsson. A fast randomized algorithm for computing a hierarchically semi-separable representation of a matrix. amath.colorado.edu/faculty/martinss/Pubs/2010_randomhudson.pdf, 2010.
[59] P.-G. Martinsson, V. Rokhlin, Y. Shkolnisky, and M. Tygert. ID: A software package for low-rank approximation of matrices via interpolative decompositions, 2008. version 0.2.
[60] L. Miranian and M. Gu. Strong rank-revealing LU factorizations. *Linear Algebra Appl.*, 367:1–16, 2003.
[61] N. Muller, L. Magaia, and B. M. Herbst. Singular value decomposition, eigenfaces, and 3D reconstructions. *SIAM Review*, 46:518–545, 2004.
[62] Nguyen, T. T. Do, and T. D. Tran. A fast and efficient algorithm for low-rank approximation of a matrix. In *STOC: Proc. 41st Ann. ACM Symp. Theory of Computing*, 2009.
[63] C.-T. Pan. On the existence and computation of rank-revealing LU factorizations. In *Householder Symposium XIII*, pages 166–168. 1996. Pontresina, Switzerland.
[64] C.-T. Pan. On the existence and computation of rank-revealing LU factorizations. *Linear Algebra Appl.*, 316:199–222, 2000.
[65] N. J. Risch and B. Devlin. On the probability of matching DNA fingerprints. *Science*, 255:717–720, 1992.
[66] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM J. Matrix Anal. Appl.*, 31:1100–1124, 2009.
[67] V. Rokhlin and M. Tygert. A fast randomized algorithm for overdetermined linear least-squares regression. *Proc. Natl. Acad. Sci. USA*, 105:13212–13217, 2008.
[68] A. Ruhe. Implementation aspects of band lanczos algorithms for computation of eigenvalues of large sparse symmetric matrices. *Math. Comp.*, 33:680–687, 1979.

[69] Y. Saad. *Numerical methods for large eigenvalue problems*. SIAM, 2 edition, 2011.
[70] Y. Saad. *Numerical Methods for Large Eigenvalue Problems*. SIAM, Philadelphia, PA, second edition, 2011.
[71] T. T. Sarlós. Improved approximation algorithms for large matrices via random projections. In *Proc. 47th Ann. IEEE Symp. Foundations of Computer Science (FOCS)*, pages 143–152, 2006.
[72] P. Schmitz and L. Ying. A fast direct solver for elliptic problems on general meshes in 2D. http://www.ma.utexas.edu/users/lexing/publications/direct2d.pdf, 2011.
[73] P. Schmitz and L. Ying. A fast direct solver for elliptic problems on general meshes in 3D. http://www.ma.utexas.edu/users/lexing/publications/direct3d.pdf, 2011.
[74] P. Sinha, B. Balas, Y. Ostrovsky, and Russell. Face recognition by humans: 19 results all computer vision researchers should know about. *Proceedings of the IEEE*, 94(11):1948–1962, 2006.
[75] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A - Optics, Image Science and Vision*, 4(3):519–524, 1987.
[76] L. Sirovich and M. Meytlis. Symmetry, probability, and recognition in face space. *PNAS - Proceedings of the National Academy of Sciences*, 106(17):6895–6899, 2009.
[77] G. W. Stewart. Updating a rank-revealing ULV decomposition. *SIAM J. Mat. Anal. Appl.*, 14(2):494–499, April 1993.
[78] D. B. Thomas, W. Luk, P. Leong, and J. D. Villasenor. Gaussian random number generators. *ACM Computing Surveys*, 39, 2007.
[79] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.
[80] B. A. Wichmann and I. D. Hill. Algorithm AS 183: An efficient and portable pseudo-random number generator. *J. of the Royal Statistical Society, Series C*, 31:188–190, 1982.
[81] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert. A fast randomized algorithm for the approximation of matrices. *Appl. Comp. Harmon. Anal.*, 25:335–366, 2008.
[82] K. Wu and H. Simon. Thick-restart Lanczos method for large symmetric eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 22:602–616, 2000.
[83] J. Xia, S. Chandrasekaran, M. Gu, and X. S. Li. Superfast multifrontal method for large structured linear systems of equations. *SIAM J. Matrix Anal. Appl.*, 31:1382–1411, 2009.
[84] J. Xia and M. Gu. Robust approximate Cholesky factorization of rank-structured symmetric positive definite matrices. *SIAM J. Matrix Anal. Appl.*, 31:2899–2920, 2010.