Title
Randomized Pivoting and Spectrum-Revealing Bounds in Numerical Linear Algebra

Author
Melgaard, Christopher Blake

Publication Date
2015

Peer reviewed|Thesis/dissertation

Randomized Pivoting and Spectrum-Revealing Bounds in Numerical Linear Algebra

by

Christopher Blake Melgaard

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Mathematics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Ming Gu, Chair
Professor James Demmel
Professor Noureddine El Karoui

Fall 2015

Randomized Pivoting and Spectrum-Revealing Bounds in Numerical Linear Algebra

Abstract

Randomized Pivoting and Spectrum-Revealing Bounds in Numerical Linear Algebra

by

Christopher Blake Melgaard

Doctor of Philosophy in Mathematics

University of California, Berkeley

Professor Ming Gu, Chair

In the first part of this dissertation, we explore a novel randomized pivoting strategy to efficiently improve the reliability and quality of the LU factorization. Gaussian elimination with partial pivoting (GEPP) has long been among the most widely used methods for computing the LU factorization of a given matrix. However, this method is also known to fail for matrices that induce large element growth during the factorization process. We propose a new scheme, Gaussian elimination with randomized complete pivoting (GERCP) for the efficient and reliable LU factorization of a given matrix. GERCP satisfies GECP (Gaussian elimination with complete pivoting) style element growth bounds with high probability, yet costs only marginally more than GEPP in terms of algorithmic complexity and run-time. Our numerical experimental results strongly suggest that GERCP is as reliable as GECP and as efficient as GEPP for computing the LU factorization.

In the second part, this dissertation provides tighter and simplified analyses of various popular low-rank matrix approximation algorithms included randomized subspace iteration and column/row selection based methods. We derive new bounds and unify them with other existing bounds under the title *Spectrum-Revealing Bounds*. These bounds demonstrate how certain structure in the decay of the spectrum of a matrix help to "reveal" an increasingly accurate estimate to the low-rank matrix approximation. We provide real world applications that demonstrate the qualitative value of our bounds for anyone using low-rank matrix approximations. In the case of randomized subspace iteration, we also dramatically improve and simplify the probabilistic analysis from previous works [50, 47] using intuitive and concise techniques.

Lastly, we apply the idea of efficient low-rank matrix approximation in the presence of spectral decay to help speed up sparse principle components analysis (SPCA). We also develop novel lower bounds on the variance captured by each sparse principle component obtained after deflation.

To My Brother

Contents

List of Figures

List of Tables

Acknowledgments

Above all, I wish to thank my advisor Prof. Ming Gu for his patience and advise. He has put a tremendous amount of time into coming up with ideas and projects, which has been invaluable to me. I would not be as well off in my life without him. I miss his talking about math, history and politics with him. I'd also like to thank Prof. Jim Demmel, Prof. John Strain and Prof. Steve Evans for many helpful conversations and discussions over the years. Many thanks to my other collaborators - Dave Anderson, Luming Wang, Simon Du and Kunming Wu - for agreeing to split up our work evenly amongst our different thesis. I also want to thank Prof. Jim Demmel, Prof. Noureddine El Karoui and Prof. Steve Evans for agreeing to be on my qualifying exam and dissertation committee. Special thanks to my long time office mate Christopher Wong for many interesting discussions and a fun working environment.

Part I

Randomized Complete Pivoting for Gaussian Elimination

Chapter 1

Introduction and Motivation

Solving linear systems of equations

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{1.1}$$

where $\mathbf{A} \in \mathbb{R}^{n\times n}$ and $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$, is a fundamental problem in numerical linear algebra and scientific computing. Gaussian Elimination with Partial Pivoting (GEPP) solves this problem by computing the LU factorization of $\mathbf{A}$ and is typically efficient and reliable. Over the years, GEPP has been repeatedly re-designed and re-implemented for better performance, and is the backbone for generations of mathematical software packages, including LINPACK [30], LAPACK [8], PLAPACK [5], SCALAPACK [24], PLASMA [3] and MAGMA [3]. GEPP routines in today's mathematical software libraries such as the Intel `mkl` [58] are capable of solving linear systems of equations with tens of thousands of variables at or near the peak of the machine's speed.

Efficiency aside, an equally important consideration is numerical reliability. While algorithms for solving eigenvalue problems have become significantly more stable over the years, GEPP was known to be, and remains, a method that is mostly stable in practice but unstable for many well-known matrices including some from common integral equations and differential equations applications [38, 100].

Pivoting plays a crucial role in the reliability of Gaussian elimination (GE), which is tied to *element growth* within the LU factorization process. The most naive version of GE, Gaussian elimination without pivoting (GENP), does not perform any pivoting and only requires $\frac{2}{3}n^3 + O(n^2)$ floating point operations with no entry comparisons [28]. However, this method can suffer from uncontrolled element growth and is only known to be reliable in a few instances like diagonally dominant matrices among others. The most popular version of GE is GEPP, which limits element growth to at most exponential by swapping the rows of $\mathbf{A}$ (i.e., partial pivoting) during elimination, and is numerically stable on average. The additional cost, about $\frac{1}{2}n^2$ entry comparisons and the associated data movement, is typically a small fraction of the total GE cost. The most reliable version of GE is Gaussian elimination with complete pivoting (GECP), which swaps both rows and columns for sub-exponential element growth [97] and is universally believed to be always backward stable in practice [28].

However, GECP is prohibitively slow with $\frac{1}{3}n^3 + O(n^2)$ entry comparisons and relatively little memory reuse [28, 51].

Rook pivoting [39, 80, 84] is an attempt to speedup complete pivoting while maintaining the guarantee of sub-exponential element growth. Rook pivoting is part of the `LUSOL` package [40] for sparse LU factorization. Despite having better performance in the "average" case, there are many matrices that still require $O(n^3)$ entry comparisons in the worst case, providing a negligible speedup over complete pivoting [51].

In this thesis, we propose a novel pivoting scheme called *Gaussian elimination with randomized complete pivoting* (GERCP). We show that GERCP satisfies a stability condition similar to that of complete pivoting, suggesting that these methods share similar stability properties. Yet, we also demonstrate that the cost of GERCP is comparable to GEPP in terms of the total number of floating point operations and comparisons. Our numerical experimental results strongly suggest that GERCP is a numerically stable and computationally efficient alternative to GEPP.

Randomization has been used to fix the numerical instability of GEPP in the literature, through GE on the product of random matrices and $\mathbf{A}$ to avoid catastrophically bad pivots. These methods are known to work well in practice in general, but they still lack effective control on element growth, and can be much less accurate than GEPP.

In Section 2, we introduce the necessary notation and background for the first part of the thesis. In Section 3, we introduce GERCP and state/prove some important properties. In section 4, we talk about numerical experiments and implementations of GERCP. The appendix has results needed by the proofs in section 3.

Chapter 2

The Setup and Background

In this thesis, we consider Gaussian elimination on an invertible square matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, although our algorithms and analysis carry over to the cases of singular matrices and rectangular matrices with few modifications.

2.1 Notation

We will follow the familiar slight abuse of notation from scientific computing and numerical linear algebra, mimicking the way that LAPACK overwrites the input matrix with the L and U factors. The diagonal of $\mathbf{A}$ becomes the diagonal of U because the diagonal of L is always 1 and thus does not need to be stored.

Algorithm 1 Classical Gaussian Elimination in Matlab Notation

Inputs: $n \times n$ matrix $\mathbf{A}$

Outputs: lower triangular L with unit diagonal, upper triangular U, row permutation Π_r, column permutation Π_c such that $\Pi_r A \Pi_c^T = LU$.

1: **set** $A = \mathbf{A}$
2: **for** $k = 1, \cdots, n-1$ **do** (i.e. called k^{th} stage of LU)
3: **select** column pivot (INSERT PIVOTING RULE).
4: **swap** (UPDATE A AND Π_c WITH PIVOT DECISION).
5: **select** row pivot (INSERT PIVOTING RULE).
6: **swap** (UPDATE A AND Π_r WITH PIVOT DECISION).
7: **compute** $A(k+1:n, k) = A(k+1:n, k)/A(k, k)$;
8: **compute** $A(k+1:n, k+1:n) = A(k+1:n, k+1:n) - A(k+1:n, k) * A(k, k+1:n)$;
9: **end for**

Remark 2.1.1. While the working matrix A has been overwritten in Algorithm 2.1, in our subsequent discussions we will refer L and U as the triangular matrices stored in A and still refer $\mathbf{A}$ as the original input matrix. We will use $A_k \in \mathbb{R}^{n\times n}$ to refer explicitly to the working

matrix before the k^{th} stage of the outer most loop. Thus, A_n refers to the working matrix after the algorithm terminates, i.e. after the $(n-1)^{th}$ stage of the outer loop.

Remark 2.1.2. For ease of discussion, we have written Algorithm 2.1 in such a way that, for each k, it performs possible column pivoting before any possible row pivoting. GENP, GEPP, GECP and rook pivoting can all be written in this form.

For any appropriate dimension m, we denote $\mathbf{e}_i \in \mathbb{R}^m$ to be the i^{th} standard basis vector, i.e. a vector with all entries equal to 0 except for the i^{th} entry which equals 1; we also denote $\mathbf{e} \in \mathbb{R}^m$ to be the vector with all entries equal to 1. Any permutation matrix $\Pi \in \mathbb{R}^{n\times n}$ is a square matrix with exactly one entry equal to 1 in each row and column, and all other entries equal to 0. We refer to the permutation induced by Π as $\pi : \{1, \cdots, n\} \to \{1, \cdots, n\}$ in the sense that $\pi(i) = j$ if and only if $\Pi\mathbf{e}_i = \mathbf{e}_j$. We will commonly make use of the swap or 2-*cycle* permutation given by $\pi_{(i,j)}$ or $\Pi_{(i,j)}$ in matrix form defined by

$$\pi_{(i,j)}(i) = j, \qquad \pi_{(i,j)}(j) = i \quad \text{and} \quad \pi_{(i,j)}(k) = k, \text{ for all } k \neq i, j$$

We denote the final row and column permutations of an algorithm as Π_r and Π_c respectively. At the k^{th} stage of LU, Algorithm 2.1 will swap the k^{th} column with the α_k^{th} column and the k^{th} row with the β_k^{th} row. As a result, we can write

$$\Pi_c = \Pi_{(n-1,\alpha_{n-1})} \cdots \Pi_{(2,\alpha_2)}\Pi_{(1,\alpha_1)}, \quad \Pi_r = \Pi_{(n-1,\beta_{n-1})} \cdots \Pi_{(2,\beta_2)}\Pi_{(1,\beta_1)}$$

as a product of the individual column/row swaps. Furthermore, we define the next notation to give us the first $k-1$ swaps and the last $n-k$ swaps

$$\Pi_{c,k} = \Pi_{(k-1,\alpha_{k-1})} \cdots \Pi_{(2,\alpha_2)}\Pi_{(1,\alpha_1)}$$
$$\Pi_{c,-k} = \Pi_{(n-1,\alpha_{n-1})} \cdots \Pi_{(k+1,\alpha_{k+1})}\Pi_{(k,\alpha_k)}$$

Also, we will use the analogous definition for $\Pi_{r,k}$ and $\Pi_{r,-k}$.

In Figure 2.1, we describe the use of the MATLAB colon notation in combination with the permutations above to explain our row and column reorderings of a matrix and its selected submatrices.

Let π_c and π_r be the permutation of columns and rows performed by LU respectively. We use the following notation to refer to matrices with the final pivoting applied apriori

$$\mathbf{A}^{\Pi_c} = \Pi_r A \Pi_c^T = A(\pi_r(:), \pi_c(:))$$
$$A_k^{\Pi_c} = \Pi_{r,-k} A \Pi_{c,-k}^T = A_k(\pi_{r,-k}(:), \pi_{c,-k}(:))$$

We do this because all of the pivoting methods discussed in this thesis are *top-heavy* as defined in Definition 3. The row pivots π_r of a top-heavy pivoting strategy are deterministic given the column pivots π_c applied to $\mathbf{A}$ by the LU factorization because each row pivot must satisfy equation (2.8). Therefore, when writing $\mathbf{A}^{\Pi_c}$, it is understood that the row pivots are the unique set of top-heavy row pivots.

Common examples of Matlab notation for $1 \leq i \leq p \leq m$ and $1 \leq j \leq q \leq n$			
Notation	**Pivoted Notation**	**Dimensions**	**Description**
$B(:,:)$	$B(\pi_1(:), \pi_2(:))$	$\mathbb{R}^{n \times n}$	Entire matrix B or $\Pi_1 B \Pi_2^T$ resp.
$B(i,:)$	$B(\pi_1(i), \pi_2(:))$	row vector in $\mathbb{R}^n$	i^{th} row of B or $\Pi_1 B \Pi_2^T$ resp.
$B(:,j)$	$B(\pi_1(:), \pi_2(j))$	column vector in $\mathbb{R}^n$	j^{th} column of B or $\Pi_1 B \Pi_2^T$ resp.
$B(i, j:q)$	$B(\pi_1(i), \pi_2(j:q))$	row vector in $\mathbb{R}^{q-j+1}$	j^{th} through q^{th} entries of i^{th} row of B or $\Pi_1 B \Pi_2^T$ resp.
$B(i:p, j)$	$B(\pi_1(i:p), \pi_2(j))$	column vector in $\mathbb{R}^{p-i+1}$	i^{th} through p^{th} entries of j^{th} column of B or $\Pi_1 B \Pi_2^T$ resp.
$B(i:p, j:q)$	$B(\pi_1(i:p), \pi_2(j:q))$	$\mathbb{R}^{(p-i+1)\times(q-j+1)}$	Submatrix from intersection i^{th} through p^{th} rows and j^{th} through q^{th} columns of B or $\Pi_1 B \Pi_2^T$ resp.

Figure 2.1: Table of Matlab notations

Schur complements form a crucial role in Gaussian elimination. We establish notation for Schur complements as $S_k \in \mathbb{R}^{(k:n)\times(k:n)}$. Notice the use of Matlab notation $(k:n) \times (k:n)$ instead of $(n-k+1) \times (n-k+1)$. The Schur complement S_k will act like a normal $(n-k+1) \times (n-k+1)$ matrix for most operations like matrix multiplication, matrix addition and ect. However, when using the Matlab notation in Figure 2.1 to access entries of S_k, we impose the abuse of notation that rows and columns are enumerated from k to n, instead of 1 to $n-k+1$. For example,

- top left entry of S_k is denoted as $S_k(k,k)$, but NOT $S_k(1,1)$.
- submatrix of last two columns of S_k is denoted by $S_k(:, n-1:n)$, but NOT $S_k(:, n-k:n-k+1)$.

This makes our analysis much cleaner and more straightforward because it synchronizes the enumeration of columns/rows between the k^{th} working matrix A_k and the k^{th} Schur complement S_k which is a submatrix for A_k, i.e. $S_k(k:n, k:n) = A_k(k:n, k:n)$. Given this notation for Schur complements, we formally define the working Schur complement at the k^{th} stage S_k and the fully pivoted k^{th} Schur complement $S_k^{\Pi_c}$

$$S_k(k:n, k:n) = A_k(k:n, k:n)$$
$$S_k^{\Pi_c}(k:n, k:n) = A_k^{\Pi_c}(k:n, k:n)$$

where $A_k \in \mathbb{R}^{n \times n}$ is the working matrix at the k^{th} stage. This implies that $S_k^{\Pi_c}(k:n, k:n) = S_k(\Pi_{r,-k}(k:n), \Pi_{c,-k}(k:n))$, i.e. S_k has the pivots only up to the k^{th} step and S_k^{Π} is already pivoted into the final permutation so that no further pivots are required.

Different A, L and U notations			
Notation	**Dimensions**	**Description**	**Algorithm Pivots**
$\mathbf{A}$, A_1, S_1	$\mathbb{R}^{n\times n}$	Unadulterated input matrix	None
$\mathbf{A}^{\Pi_c}$,$A_1^{\Pi_c}$, $S_1^{\Pi_c}$	$\mathbb{R}^{n\times n}$	Pivoted input matrix	All pivots applied apriori
A	$\mathbb{R}^{n\times n}$	Current working matrix. Overwrite triangular L, U factors and Schur complement in place	Pivots applied as determined by algorithm
A_k	$\mathbb{R}^{n\times n}$	Working matrix at k^{th} stage. Overwrite triangular L_k, U_k factors and Schur complement S_k in place	Pivots applied as determined by algorithm
$A_k^{\Pi_c}$	$\mathbb{R}^{n\times n}$	Working matrix at k^{th} stage, i.e. $\Pi_{r,-k}A_k\Pi_{c,-k}^T$	All pivots applied apriori
S_k	$\mathbb{R}^{(k:n)\times(k:n)}$	k^{th} Schur Complement of $\Pi_{r,k}\mathbf{A}\Pi_{c,k}^T$	Pivots applied as determined by algorithm
$S_k^{\Pi_c}$	$\mathbb{R}^{(k:n)\times(k:n)}$	k^{th} Schur Complement of $\Pi_r\mathbf{A}\Pi_c^T$	All pivots applied apriori

Figure 2.2: Table of Notations

The unit lower triangular matrix $L_{k+1} \in \mathbb{R}^{n\times k}$ and the upper triangular matrix $U_{k+1} \in \mathbb{R}^{k\times n}$ relate to the working matrices A_{k+1}

$$U_{k+1} = \begin{pmatrix} A_{k+1}(1,1) & A_{k+1}(1,2) & \cdots & A_{k+1}(1,k) & \cdots & A_{k+1}(1,n) \\ & A_{k+1}(2,2) & \cdots & A_{k+1}(2,k) & \cdots & A_{k+1}(2,n) \\ & & \ddots & \vdots & \cdots & \vdots \\ & & & A_{k+1}(k,k) & \cdots & A_{k+1}(k,n) \end{pmatrix}$$

$$L_{k+1} = \begin{pmatrix} 1 & & & \\ A_{k+1}(2,1) & 1 & & \\ \vdots & \vdots & \ddots & \\ A_{k+1}(k,1) & A_{k+1}(k,2) & \cdots & 1 \\ A_{k+1}(k{+}1,1) & A_{k+1}(k{+}1,2) & \cdots & A_{k+1}(k{+}1,k) \\ \vdots & \vdots & \vdots & \vdots \\ A_{k+1}(n,1) & A_{k+1}(n,2) & \cdots & A_{k+1}(n,k) \end{pmatrix}$$

for $1 < k \leq n$. Remember that $L = L_{n+1} \in \mathbb{R}^{n\times n}$ and $U = U_{n+1} \in \mathbb{R}^{n\times n}$.

2.2 Useful Preliminary Tools

2.2.1 Tools from Matrix Analysis

In finite dimensions, it is well known that all normed vector spaces are topologically equivalent. The following lemma gives the equivalence between the $\|\cdot\|_2$ and $\|\cdot\|_\infty$ vector norms, which is easily proven from the definitions of the norms. This result can also be seen as a trivial application of the Fritz John Ellipsoid Theorem [59] for convex regions that are symmetric about the origin.

Lemma 1 (Equivalence of ℓ_2 and ℓ_∞ in finite dimensions)**.** *Let $\mathbf{x} \in \mathbb{R}^d$. Then*

$$\frac{1}{\sqrt{d}} \|\mathbf{x}\|_2 \leq \|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_2$$

Definition 1 (Subordinate matrix norms [51, 53])**.** *Let $1 \leq p, q \leq \infty$ and let $\mathbf{B} \in \mathbb{R}^{m \times n}$. A* **subordinate matrix norm** *is a matrix norm of the form*

$$\|\mathbf{B}\|_{q,p} = \sup_{0 \neq \mathbf{x} \in \mathbb{R}^n} \frac{\|\mathbf{B}\mathbf{x}\|_p}{\|\mathbf{x}\|_q} = \max_{\|\mathbf{x}\|_q = 1} \|\mathbf{B}\mathbf{x}\|_p$$

Note that for normal matrix operator norms, we have that $\|\mathbf{B}\|_p = \|\mathbf{B}\|_{p,p}$. We make use of a little known subordinate matrix norm by setting $q = 1$ in the above as in exercise 6.11 of [51].

Lemma 2 (Maximum ℓ_p column norm $\|\cdot\|_{1,p}$)**.** *Let $\mathbf{B} \in \mathbb{R}^{m \times n}$. Then, we have*

$$\|\mathbf{B}\|_{1,p} = \max_{1 \leq i \leq n} \|\mathbf{B}\mathbf{e}_i\|_p = \max_{1 \leq i \leq n} \|\mathbf{B}(:, i)\|_p$$

Proof. Let $0 \neq \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$. Then

$$\|\mathbf{B}\mathbf{x}\|_p = \left\| \sum_{i=1}^n x_i \mathbf{B}(:, i) \right\|_p \leq \sum_{i=1}^n |x_i| \, \|\mathbf{B}(:, i)\|_p \leq \left(\max_j \|\mathbf{B}(:, j)\|_p \right) \|\mathbf{x}\|_1$$

Dividing both sides by $\|\mathbf{x}\|_1$ and taking a supremum, we arrive at

$$\|\mathbf{B}\|_{1,p} \stackrel{def}{=} \sup_{0 \neq \mathbf{x} \in \mathbb{R}^n} \frac{\|\mathbf{B}\mathbf{x}\|_p}{\|\mathbf{x}\|_1} \leq \max_j \|\mathbf{B}(:, j)\|_p$$

Also, let $j^* = \arg\max_j \|\mathbf{B}(:, j)\|_p$ and we arrive at our conclusion by observing

$$\max_j \|\mathbf{B}(:, j)\|_p = \frac{\|\mathbf{B}\mathbf{e}_{j^*}\|_p}{\|\mathbf{e}_{j^*}\|_1} \leq \sup_{0 \neq \mathbf{x} \in \mathbb{R}^n} \frac{\|\mathbf{B}\mathbf{x}\|_p}{\|\mathbf{x}\|_1} \stackrel{def}{=} \|\mathbf{B}\|_{1,p}$$

□

In particular, we make frequent use of the two following subordinate matrix norms:

$$\|\mathbf{B}\|_{1,\infty} = \max_{1\leq j\leq n} \|\mathbf{B}\mathbf{e}_j\|_\infty = \max_{1\leq i,j\leq n} |\mathbf{B}(i,j)| \qquad \textbf{(Maximum entry norm)} \tag{2.1}$$

$$\|\mathbf{B}\|_{1,2} = \max_{1\leq i\leq n} \|\mathbf{B}\mathbf{e}_j\|_2 = \max_{1\leq j\leq n} \|\mathbf{B}(:,j)\|_2 \qquad \textbf{(Maximum } \ell_2 \textbf{ column norm)} \tag{2.2}$$

The next lemma, from line (6.19) in chapter 6.3 of [51], will allow us to control each operator norm $\|\cdot\|_p$ for $1 \leq p \leq \infty$ of our residual error via the largest absolute column and row sum of the residual error.

Theorem 1 (Special case of Riesz-Thorin theorem [51]). *Let* $\mathbf{B} \in \mathbb{R}^{m\times n}$ *and let* $1 \leq p \leq \infty$. *Then*

$$\|\mathbf{B}\|_p \leq \|\mathbf{B}\|_1^{\frac{1}{p}} \|\mathbf{B}\|_\infty^{1-\frac{1}{p}}$$

Setting $p = 2$ in the above theorem, we get a bound on the spectral norm of B. The volume of a parallelepiped (parallelotope) formed from the columns of a matrix $\mathbf{B}$ is given by the absolute value of the determinant of B. A rectangle is formed by forcing the parallelepiped to have only right angles between each of its vectors. The next result states that the volume of parallelepiped (parallelotope) is bounded above by that of the corresponding rectangle (hyperrectangle). This result is crucial to deriving element and column growth factors.

Theorem 2 (Hadamard's Inequality [53]). *Let* $\mathbf{B} \in \mathbb{R}^{m\times m}$. *Then, we have that*

$$|\det(\mathbf{B})| \leq \prod_{j=1}^{m} \|\mathbf{B}(:,j)\|_2$$

2.2.2 Tools from Probability Theory

The union bound is a basic yet important result whose proof is typically left as an exercise in most introductory probability textbooks. We will make use of it in combination with the famous De Morgan laws to consider the probability of an intersection of highly coupled events in the analysis of GERCP.

Lemma 3 (Union Bound or Boole's Inequality). *For events* $E_1, E_2, ..., E_m$, *we have that*

$$\mathbb{P}\left(\bigcup_{i=1}^{m} E_i\right) \leq \sum_{i=1}^{m} \mathbb{P}\left(E_i\right)$$

As GERCP is a randomized algorithm, the factorization it produces will be random. The Law of Total Probability below is the basis on which we analyze the reliability of GERCP regardless of the column and row permutations chosen by GERCP.

Theorem 3 (Law of Total Probability)**.** *Given m mutually exclusive events $E_1, \cdots, E_m$ whose probabilities sum to unity, then*

$$\mathbb{P}(B) = \sum_{i=1}^{m} \mathbb{P}(B|E_i)\,\mathbb{P}(E_i),$$

where B is an arbitrary event, and $\mathbb{P}(B|E_i)$ is the conditional probability of B assuming E_i.

Next, we present a useful generalization to the Johnson-Lindenstrauss concentration of measure. This will allow us to cheaply estimate the Frobenious norm of various matrices within our algorithm.

Theorem 4. *Let $A \in \mathbb{R}^{r_1 \times m}$ and $B \in \mathbb{R}^{n \times r_2}$ be fixed matrices, along with the matrix $\Omega \in \mathbb{R}^{m \times n}$ with iid Gaussian $\mathcal{N}(0,1)$ entries. Then, for any fixed $\epsilon > 0$, we have the tail bounds*

$$\mathbb{P}\left(\|A\Omega B\|_F^2 \geq (1+\epsilon)\|A\|_F^2\|B\|_F^2\right) \leq \exp\left(-\left(\frac{\epsilon^2}{4} - \frac{\epsilon^3}{6}\right)\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}\right) \tag{2.3}$$

and

$$\mathbb{P}\left(\|A\Omega B\|_F^2 \leq (1-\epsilon)\|A\|_F^2\|B\|_F^2\right) \leq \exp\left(-\left(\frac{\epsilon^2}{4} + \frac{\epsilon^3}{6}\right)\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}\right) \tag{2.4}$$

Proof. See section 5.1 for the proof □

The matrix generalization will be of particular use in Section 4.4 and in proving Theorem 9, but most applications of the above concentration of measure will be in the form of the vector-version, which is commonly used to prove the Johnson-Lindenstrauss Theorem on randomized embeddings from probability theory and theoretical computer science. Theorem 5 has been the main theoretical foundation in the recent development of randomized algorithms in numerical linear algebra and data analysis.

Theorem 5 (Random Projection Method (Johnson-Lindenstrauss) [92])**.** *Let $\mathbf{x} \in \mathbb{R}^d$ and $\epsilon > 0$. Assume that the entries in $\Omega \in \mathbb{R}^{r \times d}$ are sampled independently from $\mathcal{N}(0,1)$. Then*

$$\mathbb{P}\left(\left\|\frac{1}{\sqrt{r}}\Omega\mathbf{x}\right\|_2^2 \geq (1+\epsilon)\|\mathbf{x}\|_2^2\right) \leq \exp\left(-\frac{(\epsilon^2-\epsilon^3)r}{4}\right)$$
$$\mathbb{P}\left(\left\|\frac{1}{\sqrt{r}}\Omega\mathbf{x}\right\|_2^2 \leq (1-\epsilon)\|\mathbf{x}\|_2^2\right) \leq \exp\left(-\frac{(\epsilon^2+\epsilon^3)r}{4}\right)$$

and

$$\mathbb{P}\left((1-\epsilon)\|\mathbf{x}\|_2^2 \leq \left\|\frac{1}{\sqrt{r}}\Omega\mathbf{x}\right\|_2^2 \leq (1+\epsilon)\|\mathbf{x}\|_2^2\right) \geq 1 - 2\exp\left(-\frac{(\epsilon^2-\epsilon^3)r}{4}\right). \tag{2.5}$$

Proof. Let $A = I_r \in \mathbb{R}^{r\times r}$ be an identity matrix and $B = \mathbf{x} \in \mathbb{R}^{d\times 1}$ be the vector. See that

$$\frac{\|I_m\|_F^2}{\|I_m\|_2^2} = r \qquad \text{and} \qquad \frac{\|\mathbf{x}\|_F^2}{\|\mathbf{x}\|_2^2} = 1$$

Then applying Theorem 4 gives the desired result. □

Due to the central importance of Theorem 5 in GERCP, we make the next definition

Definition 2. *A given vector $\mathbf{x} \in \mathbb{R}^d$ satisfies the $\epsilon-$JL condition under random mapping Ω if*

$$\sqrt{1-\epsilon}\,\|\mathbf{x}\|_2 \le \left\|\frac{1}{\sqrt{r}}\Omega\mathbf{x}\right\|_2 \le \sqrt{1+\epsilon}\,\|\mathbf{x}\|_2.$$

Remark 2.2.1. Despite its simplicity, Theorem 5 asserts the surprisingly strong norm-preserving abilities under a random projection. For any given $\Delta \in (0,1)$, x satisfies the $\epsilon-$JL condition under random mapping Ω with probability at least $1-\Delta$ for any

$$r \ge \frac{4}{\epsilon^2-\epsilon^3}\log\left(\frac{2}{\Delta}\right). \tag{2.6}$$

In particular, for $\epsilon = \frac{1}{2}$ and $\Delta = 10^{-5}$, $r = 400$ satisfies equation (2.6), regardless of d. In practice, however, one can typically choose a much smaller value of r for x to satisfy $\epsilon-$JL condition.

2.3 Numerical Error and Stability of LU Factorization

Computers make use of a set of real numbers known as *floating point numbers* [28] based off of scientific notation in that they are made up of a (i) sign, a (ii) mantissa, a (iii) base and an (iv) exponent, as shown by the example

$$2.7183 = \underbrace{+}_{(i)}\underbrace{0.27183}_{(ii)}\times\underbrace{10}_{(iii)}\overbrace{1}^{(iv)}$$

The prolific IEEE standard for binary arithmetic, which includes a single precision with 32 bits and a double precision with 64 bits, provides our computers a way to represent floating point numbers with base 2. When we approximating a real number x by a floating point number $fl(x)$ or $\hat{x}$, we either get a relative rounding approximation error within machine precision $\epsilon_{mach} > 0$, i.e.

$$fl(x) = x\,(1+\delta) \text{ for some } |\delta| \le \epsilon_{mach}$$

or we get underflow or overflow when $|x|$ is smaller or larger than the minimum or maximum positive floating point number, respectively. Single precision floating point numbers have

$\epsilon_{mach} \approx 10^{-8}$ and double precision floating point numbers have $\epsilon_{mach} \approx 10^{-16}$. Refer to [28] for more on floating point numbers. With computer roundoff, Theorem 9.3 of [51] gives us that our computed LU factorization obeys the relationship $\mathbf{A} + E = LU$ such that

$$|E_{jk}| \leq \frac{n\epsilon_{mach}}{1 - n\epsilon_{mach}} \left(|L| \cdot |U|\right)_{jk},$$

where $E \in \mathbb{R}^{n \times n}$ and $|\cdot|$ denotes taking the absolute value of each entry of a matrix. For linear systems $LUx = b$, the backwards stability of forward/backward substitution tells us that our computer will calculate $\hat{x}$ which satisfies the following approximation

$$(L + \delta L)(U + \delta U)\hat{x} = b$$

Thus, Gaussian elimination for the linear system is backwards stable if we can provide a tight bound on δA such that $(A + \delta A)\hat{x} = b$ where

$$\delta \mathbf{A} = \delta LU + L\delta U + \delta L \delta U + E$$

Theorem 9.4 of [51] tells us that δA must satisfy

$$|\delta \mathbf{A}| \leq \frac{3n\epsilon_{mach}}{1 - 3n\epsilon_{mach}} |L| \, |U| \tag{2.7}$$

Therefore, in order to bound δA, we need to simply bound L and U of our computed factorization. Next, we define a property that some pivoting strategies enjoy.

Definition 3 (Top-Heavy Pivoting Strategies)**.** *A pivoting strategy for Gaussian elimination or the LU decomposition is called* **top-heavy** *if the pivoting strategy leaves the first entry of the leading column of each Schur complement to be the entry with largest modulus in the leading column. In other words,*

$$\left|S_k^{\Pi_c}(k,k)\right| = \max_{k \leq i \leq n} \left|S_k^{\Pi_c}(i,k)\right| \tag{2.8}$$

or, in other words

$$\left|S_k^{\Pi_c}(k,k)\right| = \|S_k^{\Pi_c}(:,k)\|_\infty$$

for all $1 \leq k \leq n$.

All of the methods discussed in this thesis (partial, complete, rook, ℓ_2 complete and randomized complete pivoting) are **top-heavy** pivoting strategies. Therefore, all of these strategies enjoy the following property

Lemma 4. *Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ and let the lower triangular matrix L be obtained by the LU algorithm above with a top-heavy pivoting strategy. Then, we have*

$$\|L\|_p \leq n$$

where $1 \leq p \leq \infty$.

Proof. First, note that top-heaviness gives us that

$$|l_{jk}| = \frac{|S_k(j,k)|}{|S_k(k,k)|} \leq 1$$

for $k \leq j$. Thus,

$$\|L\|_1 \leq n, \qquad \|L\|_\infty \leq n \quad \text{and} \quad \|L\|_p \leq \|L\|_1^{\frac{1}{p}} \|L\|_\infty^{1-\frac{1}{p}} \leq n$$

where the last inequality was established by Theorem 1. □

It is easy to see that all top-heavy strategies enjoy the bound $\|L\|_p \leq n$ for $1 \leq p \leq \infty$ because all the entries below the diagonal in L will be between -1 and 1 in addition to Lemma 1. Bounding U is a little trickier and historically, the element growth factor has been used to do it.

Definition 4 (Element Growth Factor)**.** *Let $n > 0$ and $\mathbf{A} \in \mathbb{R}^{n\times n}$*

$$\rho_{elem}(\mathbf{A}) \stackrel{def}{=} \frac{\max_k \|S_k\|_{1,\infty}}{\|\mathbf{A}\|_{1,\infty}} = \frac{\max_{i,j,k} |S_k(i,j)|}{\max_{i,j} |\mathbf{A}(i,j)|}$$

In addition to the classical element growth factor, we define the new column growth factor which will be central to our analysis.

Definition 5 (Column Growth Factor)**.** *Let $n > 0$ and $\mathbf{A} \in \mathbb{R}^{n\times n}$*

$$\rho_{col}(\mathbf{A}) \stackrel{def}{=} \frac{\max_k \|S_k\|_{1,2}}{\|\mathbf{A}\|_{1,2}} = \frac{\max_{j,k} \|S_k(:,j)\|_2}{\max_j \|\mathbf{A}(:,j)\|_2}$$

These two definitions of the growth factor are related by the following lemma. It is important to note that the column growth factor commonly attains the lower bound of $\frac{1}{\sqrt{n}}\rho_{elem}$ as we will see with partial and complete pivoting, making ρ_{col} a more informative quantity to control than ρ_{elem} by a factor of $\sqrt{n}$.

Lemma 5. *Let $n > 0$ and $\mathbf{A} \in \mathbb{R}^{n\times n}$, then*

$$\frac{1}{\sqrt{n}}\rho_{elem}(\mathbf{A}) \leq \rho_{col}(\mathbf{A}) \leq \sqrt{n}\rho_{elem}(\mathbf{A})$$

Proof. Easy consequence of Lemma 1. □

Using the definition of element growth, we can bound U in the $\|\cdot\|_1$ and $\|\cdot\|_\infty$ norms as

$$\|U\|_\eta \leq n\rho_{elem}(\mathbf{A}) \|\mathbf{A}\|_{1,\infty} \qquad \text{for } \eta = 1, \infty$$

Plugging both of these estimates into Theorem 1, we get that

$$\|U\|_p \leq n\rho_{elem}(\mathbf{A}) \|\mathbf{A}\|_{1,\infty} \qquad \text{for } 1 \leq p \leq \infty$$

This gives us the following classical result

Theorem 6 (Wilkinson [28, 51]). *Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ and let E and $\delta\mathbf{A}$ be given above, then for any top-heavy pivoting strategy, we have*

$$\begin{aligned} \|\delta\mathbf{A}\|_p &\leq \frac{3n\epsilon_{mach}}{1-3n\epsilon_{mach}} n^2 \rho_{elem}(\mathbf{A}) \|\mathbf{A}\|_{1,\infty} \\ \|E\|_\eta &\leq \frac{n\epsilon_{mach}}{1-n\epsilon_{mach}} n^2 \rho_{elem}(\mathbf{A}) \|\mathbf{A}\|_{1,\infty} \end{aligned}$$

where $1 \leq p \leq \infty$.

The proof of the above is similar to the proof of the corresponding result for the column growth factor.

Theorem 7 (Column growth control on backward error). *Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ and let E and $\delta\mathbf{A}$ be given above, then for any top-heavy pivoting strategy, we have*

$$\begin{aligned} \|\delta\mathbf{A}\|_p &\leq \frac{3n\epsilon_{mach}}{1-3n\epsilon_{mach}} n^2 \rho_{col}(\mathbf{A}) \|\mathbf{A}\|_{1,2} \\ \|E\|_p &\leq \frac{n\epsilon_{mach}}{1-n\epsilon_{mach}} n^2 \rho_{col}(\mathbf{A}) \|\mathbf{A}\|_{1,2} \end{aligned}$$

where $1 \leq p \leq \infty$.

Proof. First, Lemma 5 gives use the desired bound on $\|L\|_p$. Next, please note that by the definition of the LU algorithm, we have $U(m, m:n) = S_m(\beta_m, \pi_{(m,\alpha_m)}(m:n))$. Next, we tackle the U matrix with ρ_{col}.

$$\begin{aligned} \|U\|_\infty &= \max_{1\leq m\leq n} \|U(m,:)\|_1 \leq \sqrt{n} \max_{1\leq m\leq n} \|U(m,:)\|_2 = \sqrt{n} \max_{1\leq m\leq n} \|S_m(\beta_m,:)\|_2 \\ &= \sqrt{n} \max_{1\leq m\leq n} \|S_m\|_F \leq n \max_{1\leq m\leq n} \max_{m\leq j\leq n} \|S_m(:,j)\|_2 = n \max_{1\leq m\leq n} \|S_m\|_{1,2} \end{aligned}$$

and

$$\begin{aligned} \|U\|_1 &= \max_{1\leq j\leq n} \|U(:,j)\|_1 \leq n \max_{1\leq j\leq n} \|U(:,j)\|_\infty = n \max_{1\leq j\leq n} \max_{1\leq m\leq j} |U(m,j)| \\ &= n \max_{1\leq m\leq n} \max_{m\leq j\leq n} |S_m(m,j)| \leq n \max_{1\leq m\leq n} \max_{m\leq j\leq n} \|S_m(:,j)\|_2 = n \max_{1\leq m\leq n} \|S_m\|_{1,2} \end{aligned}$$

Thus, Theorem 1 gives us

$$\|U\|_p \leq \|U\|_1^{\frac{1}{p}} \|U\|_\infty^{1-\frac{1}{p}} \leq n \max_{1\leq m\leq n} \|S_m\|_{1,2}$$

for all $1 \leq p \leq \infty$. Note that

$$\max_{1\leq m\leq n} \|S_m\|_{1,2} = \frac{\max_{1\leq m\leq n} \|S_m\|_{1,2}}{\|\mathbf{A}\|_{1,2}} \|\mathbf{A}\|_{1,2} = \rho_{col}(\mathbf{A}) \|\mathbf{A}\|_{1,2}$$

We arrive at our conclusion by combining this with equation (2.7). □

2.4 Popular Pivoting Strategies

2.4.1 No Pivoting or Static Pivoting

The easiest and most computationally efficient strategy is not to pivot at all, which we call *Gaussian elimination with no pivoting* (GENP). However, this method is not numerically stable for a general $\mathbf{A} \in \mathbb{R}^{n\times n}$. To demonstrate this, we use an example from [51]

$$\mathbf{A} = \begin{bmatrix} \delta & -1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \delta^{-1} & 1 \end{bmatrix} \begin{bmatrix} \delta & -1 \\ 0 & 1+\delta^{-1} \end{bmatrix} = LU$$

where $\delta < \epsilon_{mach}$ and ϵ_{mach} is machine epsilon. However, in floating point arithmetic, we have $fl\left(1+\delta^{-1}\right) = \delta^{-1}$ where $fl(\cdot)$ represents evaluation in floating point arithmetic. Thus,

$$fl\left(LU\right) = fl\left(L\right) fl\left(U\right) = \begin{bmatrix} 1 & 0 \\ \delta^{-1} & 1 \end{bmatrix} \begin{bmatrix} \delta & -1 \\ 0 & \delta^{-1} \end{bmatrix} = \begin{bmatrix} \delta & -1 \\ 1 & 0 \end{bmatrix} \neq \mathbf{A}$$

which is the wrong computed LU factorization with backward error $\|\mathbf{A} - fl(L)fl(U)\|_\infty = 1$. On the other hand, if $\mathbf{A} \in \mathbb{R}^{n\times n}$ is

- Totally nonnegative, i.e. the determinant of every square submatrix is nonnegative,
- Row or column diagonally dominant,
- Symmetric positive definite,

then it is proved in [51] that *no pivoting* is required for a stable computation. In fact, all of these matrices have element growth $\rho_n = O(1)$.

2.4.2 Partial Pivoting

The most common version of LU is *Gaussian elimination with partial pivoting* (GEPP) because it provides some stability at relatively cheap overhead. It is backward stable "in practice" [28], meaning that this method provides a stable LU factorization for most but not all matrices. For partial pivoting, please place the following pivoting rule in Algorithm 1: At the k^{th} stage, the k^{th} row is swapped with the β_k^{th} row, where

$$\beta_k = \underset{k\leq i\leq n}{\arg\max} \left|S_k(i,k)\right|$$

This method requires a number of entry comparisons in addition to the floating point operations required by Gaussian elimination without pivoting. Specifically, it requires

$$\sum_{k=1}^{n}(n-k) = \frac{n(n-1)}{2}$$

comparisons in total, which is one order less asymptotically than the $\frac{2}{3}n^3 + O(n^2)$ flops in GENP. Each row swap requires n individual entry swaps, so the total number of entry swaps required is bounded above by

$$\sum_{k=1}^{n-1} n = n(n-1)$$

The element growth for partial pivoting is bounded by

$$\rho_{elem}^{gepp}(\mathbf{A}) \leq 2^{n-1}$$

It is also easy to show that the column growth for partial pivoting is bounded by

$$\rho_{col}^{gepp}(\mathbf{A}) \leq \frac{1}{\sqrt{n}} 2^{n-1}$$

Both of these bounds are attained by the Wilkinson matrix

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 1 \\ -1 & 1 & 0 & \cdots & 0 & 1 \\ -1 & -1 & 1 & \cdots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -1 & -1 & -1 & \cdots & 1 & 1 \\ -1 & -1 & -1 & \cdots & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ -1 & 1 & 0 & \cdots & 0 & 0 \\ -1 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -1 & -1 & -1 & \cdots & 1 & 0 \\ -1 & -1 & -1 & \cdots & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 1 & 0 & \cdots & 0 & 2 \\ 0 & 0 & 1 & \cdots & 0 & 2^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 2^{n-2} \\ 0 & 0 & 0 & \cdots & 0 & 2^{n-1} \end{bmatrix} = LU$$

where we see that the element growth in U is 2^{n-1} and the column growth $\frac{\|2^{n-1}\|_{\ell_2(\mathbb{R}^1)}}{\|\mathbf{e}\|_{\ell_2(\mathbb{R}^n)}} = \frac{2^{n-1}}{\sqrt{n}}$.

Large element growth and unstable LU factorizations with partial pivoting also occur in many applications. Wright [100] describes a family of two-point boundary value problems that cause GEPP to fail via exponential element growth when attempting to solve the ODE by discretizing it into matrix form. Liu and Russell [68] experience the same phenomenon when attempting to solve the discretized Kuramoto-Sivashinsky PDE, which is used to model laminar flame front propagation, phase dynamics in reaction-diffusion systems, fluctuations in fluid films and instabilities in plasma physics [56, 64]. Foster [38] applies the Newton-Cotes quadrature to discretize the Volterra Integral equation from many areas of applied mathematics including actuarial science, viscoelastic materials and probability theory. This reduces the Volterra Integral equation into a matrix equation that makes GEPP fail. As we will discuss later, the Volterra example is among the most diabolical examples that break GEPP because it induces *passive aggressive element growth*, i.e. barely enough element growth to cause GEPP to fail.

The remainder of the partial pivoting section is spent discussing the *generalized Wilkinson Matrix*, which is a more general class of matrices that can cause exponential element growth in GEPP.

Example 2.4.1. [Generalized Wilkinson Matrices $\mathcal{GW}$ [62]] For any integer $r \geq 1$, consider a matrix $\mathbf{A}$ of the following form

$$\mathbf{A} = L + \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & \dots & 0 & 1 \end{pmatrix},$$

where L is a lower triangular matrix with

$$L_{i,i} = 1, \quad \text{and} \quad L_{i,j} = -u_i^T W_{i+1} \cdots W_{j-1} v_j, \quad \text{for any} \quad i > j,$$

with $u_i, v_j \in \mathbb{R}^r$ being any vectors and $W_i \in \mathbb{R}^{r \times r}$ being square matrices. The matrix $\mathbf{A}$ reduces to the Wilkinson matrix for the special case $r = 1$, $u_i = v_j = W_i = 1$ for all i and j. It is straightforward to verify that L^{-1} is a lower triangular matrix with

$$\left(L^{-1}\right)_{i,i} = 1, \quad \text{and} \quad \left(L^{-1}\right)_{i,j} = u_i^T \widehat{W}_{i+1} \cdots \widehat{W}_{j-1} v_j, \quad \text{for any} \quad i > j,$$

where $\widehat{W}_i = W_i + v_i u_i^T$. Now we choose the vectors $\{u_i\}_{i=2}^n$, $\{v_j\}_{j=1}^{n-1}$ and matrices $\{W_i\}_{i=2}^{n-1}$ to contain only positive entries and have 2-norm at most 1. This implies that

$$|L_{i,j}| = |u_i^T W_{i+1} \cdots W_{j-1} v_j| \leq 1 \quad \text{for any} \quad i > j.$$

Consequently LU-factorizing $\mathbf{A}$ with GEPP will incur no row exchanges, and the resulting matrix factorization has the form

$$\mathbf{A} = L\, U, \quad \text{where} \quad U = I + \begin{pmatrix} & & \\ & & \\ L^{-1} & \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 0 \end{pmatrix} & \end{pmatrix} \begin{pmatrix} 0 & \cdots & 0 & 1 \end{pmatrix}.$$

This typically implies exponential element growth in U if the inequality $\|\widehat{W}_i\|_2 > 1$ holds for most matrices $\widehat{W}_i$.

In our numerical experiments, we use this to create a random matrix ensemble that causes GEPP to fail with high probability.

2.4.3 Complete Pivoting

The most reliable version is *Gaussian elimination with complete pivoting* (GECP). Von Neumann and Goldstine [94] referred to this as the "customary procedure." For complete pivoting, please place the following pivoting rule in Algorithm 1: At the k^{th} stage, the k^{th} row and k^{th} column are swapped with the β_k^{th} row and α_k^{th} column respectively, where

$$(\beta_k, \alpha_k) = \underset{\substack{(i,j) \in \mathbb{N}^2 \\ k \leq i,j \leq n}}{\arg\max} |S_k(i,j)|$$

Next, we consider the overhead of complete pivoting, which is broken into entry comparisons and data movement. The number of overall entry comparisons is

$$\sum_{k=1}^{n} \left(k^2 - 1\right) = \frac{n(n+1)(2n+1)}{6} - n = \frac{1}{3}n^3 + \frac{1}{2}n^2 - \frac{5}{6}n$$

In the worst case, the number of entry swaps (or data movement) is

$$\sum_{k=1}^{n-1} 2n = 2n(n-1)$$

Therefore, despite the fact that each entry swap is more expensive on modern computers than each entry comparison, the entry comparisons will form the bulk of the overhead when n is large. This is different than partial pivoting because the total number of comparisons in complete pivoting is $O(n^3)$ instead of the $O(n^2)$ comparisons in partial pivoting. In his seminal work, Wilkinson [97] proves that the element growth in complete pivoting is bounded above by

$$\rho_{elem}^{gecp}(\mathbf{A}) \leq \sqrt{n}\left(2 \cdot 3^{\frac{1}{2}} \cdots n^{\frac{1}{n-1}}\right)^{1/2} \sim cn^{1/2} n^{\frac{1}{4}\ln(n)}$$

Our proof of Theorem 8 can be easily adapted to show that the column growth for complete pivoting is bounded above by

$$\rho_{col}^{gecp}(A) \leq \left(2 \cdot 3^{\frac{1}{2}} \cdots n^{\frac{1}{n-1}}\right)^{1/2} \sim cn^{\frac{1}{4}\ln(n)}$$

These bounds are provably unattainable for $n \geq 3$ [97] because of their proof's reliance on Hadamard's inequality of Theorem 2. It was incorrectly conjectured that $\rho_{elem}^{gecp}(\mathbf{A}) \leq n$ [35, 45]. Nonetheless, it is widely believed that the above element growth bound is wildly pessimistic. However, this bound proves that exponential element growth is impossible as $n^{\log(n)}$ is sub-exponential. Because of this, we call GECP *backwards stable*.

2.4.4 Rook Pivoting

An important attempt to speed up complete pivoting was introduced by Neal and Poole [80] as *Gaussian elimination with rook pivoting* (GERP). Basically, one alternates between partial pivoting on the rows and the columns until arriving at a Nash-equilibrium of sorts. For rook pivoting, place the following pivoting rule in Algorithm 1: At k^{th} stage, initialize $\beta_k = k$ and $\alpha_k = k$. First, choose a new β_k from (2.9) while holding α_k constant, and then choose α_k from (2.10) while holding β_k constant. Repeat (2.9) and (2.10) until the current choice (β_k, α_k) make (2.9) and (2.10) hold simultaneously.

$$\beta_k = \underset{k \leq i \leq n}{\arg\max} \left|S_k\left(i, \alpha_k\right)\right| \tag{2.9}$$

$$\alpha_k = \underset{k \leq j \leq n}{\arg\max} \left|S_k\left(\beta_k, j\right)\right| \tag{2.10}$$

The number of entry comparisons required by rook pivoting depends on the matrix. For example, a diagonally dominant matrix will require no swaps, which causes rook pivoting to only check (2.9) once. This example gives $\sum_{k=1}^{n}(n-k) = \frac{n(n-1)}{2} = O(n^2)$ entry comparisons just like partial pivoting. In fact, there are a few probabilistic arguments [39, 84] that claim for the "average" input matrix $\mathbf{A}$, the user would expect to perform $O(n^2)$ entry comparisons in total. However, the worst case needs a total of $O(n^3)$ entry comparisons (just as bad as complete pivoting) as exemplified by any matrix of the form [51]

$$\begin{pmatrix} \theta_1 & \theta_2 & & & \\ & \theta_3 & \theta_4 & & \\ & & \ddots & \ddots & \\ & & & \theta_{2n-3} & \theta_{2n-2} \\ & & & & \theta_{2n-1} \end{pmatrix}, \qquad |\theta_1| < |\theta_2| < \cdots < |\theta_{2n-1}|$$

The worst case data movement in terms of entry-wise swaps is the same as GECP at $O(n^2)$ because both rows and columns are also being swapped here. It is not clear whether comparisons or swaps are to be considered the dominant overhead cost as it will depend on whether the input matrix requires $O(n^3)$ or $O(n^2)$ comparisons. Foster [39] proves that rook pivoting element growth must obey

$$\rho_{elem}^{gerp}(\mathbf{A}) \leq \frac{3}{2} n^{\frac{3}{4}\ln(n)}$$

and he also shows that the bound is unattainable for $n \geq 3$. This suggests that this method has similar stability properties to complete pivoting and should be considered as a less expensive "cousin."

2.4.5 Prior attempts at randomizing Gaussian elimination

Randomization in the context of Gaussian elimination based direct solvers has been attempted in the past as a way to avoid pivoting altogether [81, 82, 12, 11]. It is important to keep in mind that these methods serve a different purpose than our method GERCP. In other words, prior attempts to randomize Gaussian elimination are meant to be faster that partial pivoting, while our method GERCP is meant to produce high quality solutions that partial pivoting at a marginal expense in run time. These methods pre/post multiply our input matrix $\mathbf{A}$ by random matrices before applying Gaussian elimination without pivoting. This can either be used to solve a linear system or compute the inverse of $\mathbf{A}$. Unfortunately, there are no theoretical guarantees of small backwards error. In fact, many examples cause these methods to produce a large backwards error relative to partial pivoting from rounding errors as shown in the numerical experiments section of [82]. One such linear system $\mathbf{Ax} = \mathbf{b}$ is given as

$$a_{ij} = \begin{pmatrix} i+j-2 \\ j-1 \end{pmatrix} \text{ and } b_i = 1$$

gives the exact solution $\mathbf{x} = \mathbf{e_1}$. This matrix has a condition number of 2^{4n} is spectral norm, so any linear solver will eventually have problems when this system as $n \to \infty$. However, GEPP is able to maintain accuracy for larger n as the performance of the randomized scheme deteriorates almost immediately due to round off error as shown in the numerical experiments section of [82].

Chapter 3

Algorithm GERCP and Main Results

In this section, we first introduce Algorithm 3.1, a deterministic complete pivoting scheme based on the ℓ_2 norm; we then evolve Algorithm 3.1 into GERCP by significantly speeding it up with randomization.

3.1 Deterministic ℓ_2-norm Complete Pivoting

To help motivate our randomized strategy, we propose an intermediate deterministic strategy called *Gaussian elimination with ℓ_2 complete pivoting* (GE2CP).

Algorithm 2 Gaussian Elimination with ℓ_2 norm Complete Pivoting (GE2CP)

Inputs: $n \times n$ matrix **A**

Outputs: lower triangular L with unit diagonal, upper triangular U, row permutation Π_r, column permutation Π_c.

1: **for** $k = 1, \cdots, n-1$ **do**
2: **compute** $\alpha = \mathbf{argmax}_{k \le j \le n} \|A(k:n, j)\|_2^2$.
3: **swap** columns k and α of A.
4: **compute** $\beta = \mathbf{argmax}_{k \le j \le n} |A(j, k)|$.
5: **swap** rows k and β of A.
6: **compute** $A(k+1:n, k) = A(k+1:n, k)/A(k, k)$;
7: **compute** $A(k+1:n, k+1:n) = A(k+1:n, k+1:n) - A(k+1:n, k) * A(k, k+1:n)$;
8: **end for**

Pivoting is done in two steps in Algorithm 3.1 for each k: Step 1 swaps the α^{th} column of the trailing matrix $A(k:n, k:n)$ with the k^{th}, where $A(k:n, \alpha)$ has the largest column 2-norm among all columns of $A(k:n, k:n)$; whereas Step 2 swaps the β^{th} row of $A(k:n, k:n)$ with the k^{th}, where $A(\beta, k)$ is the largest in absolute value among all entries of $A(k:n, k)$. Step 1 controls potentially harmful column norm growth in **A** through column interchanges, and Step 2 performs standard partial pivoting to control potentially harmful element growth

in the k-column $A(k:n,k)$ through row interchanges. Step 2 makes the ℓ_2 norm Complete Pivoting strategy (GE2CP) in Algorithm 3.1 a *top-heavy* pivoting strategy, which allows us to apply Theorem 7 to bound the LU backward error.

In this method, the number of comparisons is reduced in favor of additional floating point operations. The total amount of comparisons for this method is

$$\sum_{k=1}^{n} 2(n-k) = n(n-1) = O(n^2).$$

The total additional floating point operations required to directly compute the 2-norms in Step 1 of Algorithm 3.1 is about

$$\sum_{k=1}^{n-1} 2(n-k+1)(n-k+1) = \frac{n(n+1)(2n+1)}{3} - 2 = \frac{2}{3}n^3.$$

The worst case for entry swaps in GE2CP is the same as in GECP at $O(n^2)$. The ℓ_2 norm complete pivoting strategy (GE2CP) in Algorithm 3.1 enjoys similar element/column growth bounds to complete pivoting and rook pivoting. One of the main results of this thesis is an upper bound on the element growth of our randomized pivoting algorithm in Theorem 8. By omitting references and applications of Lemma 7 from the proof of Theorem 8, one can easily show that

$$\rho_{col}^{ge2cp} \leq n\sqrt{e(n+1)}n^{\frac{1}{2}\ln(n)}$$

The $O(n^3)$ flop overhead makes Algorithm 3.1 an impractical alternative to GEPP. In Section 3.2, we develop GERCP by randomizing Algorithm 3.1 to choose columns with sufficiently large column norms at significantly lowered overhead costs. Furthermore, we will derive a GECP style element growth upper bound for GERCP that holds with an arbitrary user-defined probability $\delta \in (0,1)$ of failure.

3.2 Gaussian Elimination with Randomized Complete Pivoting (GERCP)

Our randomized Gaussian Elimination algorithm, GERCP, is based, in principle, on Algorithm 3.1. However, the key difference is we will replace the column pivoting step, Step 1, by a randomized alternative to significantly reduce its cost. At its core, GERCP relies on a fast random projection scheme to reliably estimate the column norms of *each* observed Schur complement required by line 1 of Algorithm 3.1, based on Theorem 5.

3.2.1 Successive Schur Sketching

We adapt the idea of a *sketching matrix* [99] in this section to speed up the column selection procedure in GE2CP Algorithm 3.1. Let our sampling matrix $\Omega \in \mathbb{R}^{r\times n}$ be a

random matrix with iid standard normal $\mathcal{N}(0,1)$ entries where $r \ll n$, and let $\Psi = \Omega A$ be our first sketching matrix. We refer to the number $r \in \mathbb{N}$ as the **sampling dimension**. One can also make use of a Fast Johnson-Lindenstrauss-like sampling matrix [4] as a sampling matrix. However, we will stick with a Gaussian sampling matrix in this thesis for simplicity of presentation. By Theorem 5, the column norms of A can be reliably estimated via those of Ψ for a large enough choice of r. In other words, we can perform Step 1 of Algorithm 3.1 for $k = 1$ by looking for the column with the largest column norm in Ψ.

However, Step 1 of Algorithm 3.1 must be performed for every value of k. From the single random matrix Ω, below we will construct a collection of matrices $\{\Psi_k\}_{k=1}^{n}$ known as the **Schur sketching matrices**, where $\Psi_k \in \mathbb{R}^{r\times(n-k+1)}$. These matrices are constructed as

$$\Psi_k \stackrel{def}{=} \Omega\left(:, \pi_{r,k}(k:n)\right)\ S_k, \text{ for all } 1 < k \leq n. \tag{3.1}$$

Remark 3.2.1. *For randomized complete pivoting, the choice of the k^{th} pivot column will be based on column norms in the Schur sketching matrix Ψ_k. Since all Schur sketching matrices are based on the same random matrix Ω, the Schur complement $S_k \in \mathbb{R}^{(k:n)\times(k:n)}$ for $1 < k \leq n$ will* **not** *be a deterministic matrix. Indeed, the observed Schur complement S_k is determined by the randomized column pivoting decisions $\alpha_1, \cdots, \alpha_{k-1}$ of GERCP from the previous stages. Given that there are only a finite number of pivot decisions, we conclude that S_k must be a discrete random variable.*

To efficiently continue with all other column pivoting work in Algorithm 3.1, we inductively devise a scheme to use our current Schur sketching matrix Ψ to produce our next Schur sketching matrix $\widehat{\Psi}$. Suppose that we have chosen $p > 0$ rows/columns from the remaining Schur complement $S_k \in \mathcal{R}^{(n-k+1)\times(n-k+1)}$

$$S_k = \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix} = \begin{pmatrix} L_{11} & \\ L_{21} & I \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \\ & \widehat{S}_{22} \end{pmatrix},$$

where $S_{11}, L_{11}, U_{11} \in \mathbb{R}^{p\times p}$, with L_{11} lower triangular and U_{11} upper triangular, respectively; $S_{21}, L_{21} \in \mathbb{R}^{(n-k+p+1)\times p}$; $S_{12}, U_{12} \in \mathbb{R}^{p\times(n-k-p+1)}$; and $S_{22}, \widehat{S}_{22} \in \mathbb{R}^{(n-k-p+1)\times(n-k-p+1)}$ with

$$\widehat{S}_{22} = S_{22} - S_{21}S_{11}^{-1}S_{12} = S_{22} - L_{21}U_{12}$$

being the Schur complement. We call $p > 0$ the **pivot-block size**, namely the number of row/column pivots performed before each update to the Schur sketching matrix Ψ. With the notation established above, we have $\widehat{S}_{22} = S_{n-k-p+1}$.

Step 1 of Algorithm 3.1 requires that we perform column pivoting on $\widehat{S}_{22}$. To do this work, we need to multiply $\widehat{S}_{22}$ by a random matrix. Instead of generating a new random matrix, we introduce a simple and efficient procedure, **Successive Schur Sketching (SSS)**. We partition Ω and Ψ accordingly as $\Omega = \begin{pmatrix} \Omega_P & \Omega_R \end{pmatrix}$ and

$$\begin{aligned} \Psi &= \begin{pmatrix} \Psi_P & \Psi_R \end{pmatrix} = \begin{pmatrix} \Omega_P & \Omega_R \end{pmatrix} \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix} \\ &= \begin{pmatrix} \Omega_P S_{11} + \Omega_R S_{21} & \Omega_P S_{12} + \Omega_R S_{22} \end{pmatrix}. \end{aligned}$$

Now we compute the next Schur sketching matrix $\widehat{\Psi}$ for the Schur complement $\widehat{S}_{22}$ as

$$\begin{aligned}\widehat{\Psi} &\overset{def}{=} \Omega_R \widehat{S}_{22} = \Omega_R \left(S_{22} - L_{21}U_{12}\right) = \left(\Omega_P S_{12} + \Omega_R S_{22}\right) - \left(\Omega_P S_{12} + \Omega_R L_{21}U_{12}\right) \\ &= \Psi_R - \left(\Omega_P L_{11} + \Omega_R L_{21}\right) U_{12}. \end{aligned} \tag{3.2}$$

Thus, in SSS we use Ω_R, a submatrix of Ω, for the new sample matrix $\widehat{\Psi}$. If done directly, it would take $2r(n-k-p+1)^2$ flops to compute $\widehat{\Psi}$ as a matrix-matrix product $\Omega_R \widehat{S}_{22}$. However, since Ψ_R is part of Ψ and was computed in the previous steps, $\widehat{\Psi}$ can instead be computed via equation (3.2) in about $4rp(n-k+1)$ flops, a very large savings for $p \ll n$. Indeed, this random matrix reuse will prove critically important in reducing the overall cost of computing all sample matrices by Algorithm GERCP. Later on we will further show that Algorithm GERCP will be as reliable as sampling the Schur matrices $\widehat{S}_{22}$ without random matrix reuse.

Given Ψ, the new sample matrix $\widehat{\Psi}$ can be updated in about $4rp(n-k+1)$ flops with the above formula, which is much cheaper than the $O(rn^2)$ flops required for a direct computation. It costs another $2r(n-k+1)$ flops to select a column with sufficiently large column norm. If we perform a column pivot once every block elimination step, the total overhead due to column pivoting includes the computation of the initial sampling matrix, its update at every column pivot, and column selection based on the column norms of the updated sample matrix. These costs add up to

$$4rn^2 + \sum_{j=1}^{n/p} \left(4rp + 2r\right)\left(n - (j-1)p\right) \approx \left(6r + r/p\right) n^2$$

flops, which is much smaller than the $O(n^3)$ additional flops required by GE2CP. One can perform a floating point error analysis to show that successively updating Ψ at each stage via equation (3.2) as follows.

Lemma 6. *Suppose equation* (3.2) *is continuously used to produce each sketching matrix* Ψ_k*. Then, each sketching matrix in floating point arithmetic is close to the corresponding sketching matrix in real arithmetic as given by*

$$\begin{aligned}&\left|fl\big(\widehat{\Psi}_{k+1}\big) - \widehat{\Psi}_{k+1}\right| \\ &\quad \le \epsilon_{mach}\left(1+\epsilon_{mach}\right)^k \left(\sum_{i=1}^{k} \left|\Psi_i\left(:, (k+1)p : n\right)\right| + 5\left|\Omega\right| \left|L_{(k+1)p}\right| \left|U_{(k+1)p}(:, (k+1)p : n)\right|\right)\end{aligned}$$

Remark 3.2.2. *Later in the thesis, we will provide a probabilistic analysis of this lemma to get rid of the randomness originating from the sampling matrix* Ω*.*

Proof. A brief floating point analysis proceeds as follows

$$\begin{aligned}\left|fl\big((\Omega_P L_{11} + \Omega_R L_{21})U_{12}\big) - (\Omega_P L_{11} + \Omega_R L_{21})U_{12}\right| &\le 4\epsilon_{mach}(\left|\Omega_P\right| \left|L_{11}\right| + \left|\Omega_R\right| \left|L_{21}\right|) \left|U_{12}\right| \\ &= 4\epsilon_{mach} \begin{pmatrix} \left|\Omega_P\right| & \left|\Omega_R\right| \end{pmatrix} \begin{pmatrix} \left|L_{11}\right| \\ \left|L_{21}\right| \end{pmatrix} \left|U_{12}\right|\end{aligned}$$

Place this together with equation (3.2) in floating point arithmetic to arrive at

$$\begin{aligned}&\left|fl\left(\widehat{\Psi}_{k+1}\right)-\widehat{\Psi}_{k+1}\right| \\ &\quad\leq \underbrace{\left|fl\left(\Psi_{k,R}\right)-\Psi_{k,R}\right|}_{\text{past update errors}}(1+\epsilon_{mach})+\epsilon_{mach}\left|\Psi_{k,R}\right|+5\epsilon_{mach}\begin{pmatrix}|\Omega_P| & |\Omega_R|\end{pmatrix}\begin{pmatrix}|L_{11}|\\|L_{21}|\end{pmatrix}|U_{12}|\end{aligned}$$

Applying a simple induction argument to the inequality above gives the desired result. □

Remark 3.2.3. *This rounding error analysis applies equally well to the L and U factors computed in finite precision arithmetic. We did not make the distinction to avoid introducing yet more notation.*

When U_{11} is well-conditioned, $\widehat{\Psi}$ can be updated by the more efficient formula

$$\begin{aligned}\widehat{\Psi} &\stackrel{def}{=} \Omega_R\widehat{S}_{22}=\Omega_R\left(S_{22}-L_{21}U_{12}\right)=\left(\Omega_P S_{12}+\Omega_R S_{22}\right)-\left(\Omega_P S_{11}+\Omega_R S_{21}\right)S_{11}^{-1}S_{12}\\ &=\Psi_R-\left(\Psi_P U_{11}^{-1}\right)U_{12},\end{aligned}\tag{3.3}$$

which costs about $2rp(n-k+1)+2r(n-k+1)$ flops.

3.2.2 Column pivot quality and column growth factor for GERCP

We present classical Gaussian elimination with randomized complete pivoting (GERCP) below as Algorithm 3.2.2. The section is primarily focused with the development of GERCP from a theoretical perspective, while Chapter 4 will be focused on issues of practical implementation. Provided that the sampling dimension r is large enough, we show that, with high probability, each GERCP pivot column has an ℓ_2 length within a constant factor of the largest column, i.e. the GE2CP pivot column. Using this property, we then prove a sub exponential upper bound on the column growth factor for GERCP that holds with probability not less than $1-\delta$ for any user-defined quantity $\delta>0$. For the sake of simplicity and theoretical guarantees, we will focus on the case with pivot-block size $p=1$ for the rest of the thesis. In line

For ease of notation, we consider a scalar version of GERCP where column pivoting and row pivoting are done one column/one row at a time. Later, in Section 4, we write this algorithm into a blocked version in Algorithm 4 in order to increase the amount of locality in BLAS-3 operation. The column pivots Π_c are chosen from randomized column pivoting. The row pivots Π_r are also random, but if we fix the column pivots Π_c then the row pivots are deterministic and are uniquely determined by the top-heavy property (2.8). Let $S_k^{\Pi_c}\in\mathbb{R}^{(k:n)\times(k:n)}$ be the Schur complement after the first $k-1$ steps of column and row pivoting and Gaussian elimination, and let $\Omega_k^{\Pi_c}=\Omega(:,\pi_r(k:n))\in\mathbb{R}^{r\times(n-k+1)}$ be the

Algorithm 3 Gaussian Elimination with Randomized Complete Pivoting

Inputs: $n \times n$ matrix A, sampling dimension $r > 0$, an optional column threshold $0 < g \leq 1$

Outputs: lower triangular L with unit diagonal, upper triangular U, row permutation Π_r, column permutation Π_c

1: **sample** $\Omega(i,j) \sim \mathcal{N}(0,1)$ for all $1 \leq i \leq r$ and $1 \leq j \leq n$.
2: **compute** $\Psi = \Omega A$.
3: **for** $k = 1, \cdots, n-1$ **do**
4: **compute** $\ell = \underset{k \leq j \leq n}{\arg\max} \|\Psi(:,j)\|_2^2$
5: **set** $\alpha = \begin{cases} k & \text{, if } \|\Psi(:,k)\|_2^2 \geq g^2 \|\Psi(:,\ell)\|_2^2 \\ \ell & \text{, otherwise} \end{cases}$.
6: **swap** columns k and α of A and Ψ.
7: **compute** $\beta = \underset{k \leq i \leq n}{\arg\max} |A(i,k)|$.
8: **swap** rows k and β of A.
9: **compute** $A(k+1:n,k) = A(k+1:n,k)/A(k,k)$
10: **compute** $A(k+1:n,k+1:n) = A(k+1:n,k+1:n) - A(k+1:n,k) * A(k,k+1:n)$
11: **update** $\Psi(:,k:n)$ with either (3.2) or (3.3)
12: **end for**

submatrix of Ω whose columns correspond to the rows of $S_k^{\Pi_c}$. We now define the following events for $1 \leq k \leq i \leq n$:

$$\overline{\mathbf{C}}_{i,k}^{\Pi_c} = \left\{ \left\| \frac{1}{\sqrt{r}} \Omega_k^{\Pi_c} S_k^{\Pi_c}(:,i) \right\|_2 \leq \sqrt{1+\epsilon} \left\| S_k^{\Pi_c}(:,i) \right\|_2 \right\} \tag{3.4}$$

$$\underline{\mathbf{C}}_{i,k}^{\Pi_c} = \left\{ \sqrt{1-\epsilon} \left\| S_k^{\Pi_c}(:,i) \right\|_2 \leq \left\| \frac{1}{\sqrt{r}} \Omega_k^{\Pi_c} S_k^{\Pi_c}(:,i) \right\|_2 \right\} \tag{3.5}$$

$$\mathbf{C}_{i,k}^{\Pi_c} = \left\{ \sqrt{1-\epsilon} \left\| S_k^{\Pi_c}(:,i) \right\|_2 \leq \left\| \frac{1}{\sqrt{r}} \Omega_k^{\Pi_c} S_k^{\Pi_c}(:,i) \right\|_2 \leq \sqrt{1+\epsilon} \left\| S_k^{\Pi_c}(:,i) \right\|_2 \right\} \tag{3.6}$$

$$= \overline{\mathbf{C}}_{i,k}^{\Pi_c} \bigcap \underline{\mathbf{C}}_{i,k}^{\Pi_c} \tag{3.7}$$

By Definition 2, $\mathbf{C}_{i,k}^{\Pi_c}$ describes the event where the i^{th} column of the k^{th} Schur complement $S_k^{\Pi_c}$ satisfies the $\epsilon-$JL condition under random mapping $\Omega_k^{\Pi_c}$. We also define for each k

$$\mathbf{D}_{k,k}^{\Pi_c} = \left\{ \left\| \Omega_k^{\Pi_c} S_k^{\Pi_c}(:,k) \right\|_2 \geq g \left\| \Omega_k^{\Pi_c} S_k^{\Pi_c} \right\|_{1,2} \right\}.$$

which describes the event where no column is swapped under randomized column pivoting at step k with column threshold $0 < g \leq 1$. Remember if $g = 1$ there is no thresholding, which makes $\alpha = \ell$ from line 4 and 5 of Algorithm 3.2.2. We further define for $k < i$ and $\gamma_k \overset{def}{=} \left\| \Omega_k^{\Pi_c} S_k^{\Pi_c} \right\|_{1,2}$,

$$\mathbf{D}_{i,k}^{\Pi_c} = \left\{ \left\| \Omega_k^{\Pi_c} S_k^{\Pi_c}(:,k) \right\|_2 < g\gamma_k,\ \left\| \Omega_k^{\Pi_c} S_k^{\Pi_c}(:,j) \right\|_2 < \gamma_k,\ k<j<i \leq n,\ \left\| \Omega_k^{\Pi_c} S_k^{\Pi_c}(:,i) \right\|_2 = \gamma_k. \right\}$$

Thus, $\mathbf{D}_{i,k}^{\Pi_c}$ is the event where columns k and i are swapped. Thus the event

$$\mathcal{D}\left(\Pi_c\right) \stackrel{def}{=} \bigcap_{1\leq k\leq n} \mathbf{D}_{\alpha_k,k}^{\Pi_c} \tag{3.8a}$$

uniquely defines the column permutation Π_c and by extension, row permutation Π_r, and the event

$$\mathcal{C}\left(\Pi_c\right) \stackrel{def}{=} \bigcap_{1\leq k\leq n} \left(\left(\bigcap_{\substack{k\leq j\leq n \\ j\neq\alpha_k}} \underline{\mathbf{C}}_{j,k}^{\Pi_c} \right) \bigcap \overline{\mathbf{C}}_{\alpha_k,k}^{\Pi_c} \right) \tag{3.8b}$$

defines the Gaussian elimination process where every column in every Schur complement satisfies the $\epsilon-$JL condition for the column permutation Π_c; and the event

$$\mathcal{C} \stackrel{def}{=} \bigcup_{\Pi_c} \left(\mathcal{C}\left(\Pi_c\right) \bigcap \mathcal{D}\left(\Pi_c\right) \right) \tag{3.8c}$$

defines set of Gaussian elimination processes where every column in each Schur complement produced by the algorithm satisfies the $\epsilon-$JL condition during the factorization process. Note that event $\mathcal{C}$ describes the randomized Gaussian elimination process, whereas event $\mathcal{C}^{(\Pi_c)}$ describes a particular incidence of $\mathcal{C}$ conditional on a particular column permutation Π_c.

Since GERCP performs partial pivoting at every step of elimination, it will successfully compute an LU factorization of any given matrix. Additionally, the events $\mathcal{D}^{(\Pi_c)}$ over the set of permutations are mutually exclusive by definition. In other words,

$$\mathbb{P}\left(\bigcup_{\Pi_c} \mathcal{D}\left(\Pi_c\right) \right) = \sum_{\Pi_c} \mathbb{P}\left(\mathcal{D}\left(\Pi_c\right)\right) = 1, \tag{3.9}$$

where the set union is over all possible permutations Π_c. Lemma 7 below shows that with large probability, in GERCP every column in every Schur complement satisfies the $\epsilon-$JL condition during the elimination process regardless of which permutation Π_c is chosen.

Lemma 7 (Randomized norm preservation)**.** *Given $\epsilon, \delta \in (0,1)$ and $g \in (0,1]$. Let Ψ be defined by equation* (3.1)*. Choose $r > \dfrac{4}{\epsilon^2-\epsilon^3} \ln\left(\dfrac{n(n+1)}{2\delta}\right)$. Then we must have*

$$\left\|S_k\right\|_{1,2} \leq \frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}} \left\|S_k(:,\alpha_k)\right\|_2 \tag{3.10}$$

for all $1 \leq k \leq n$ with probability no less than $1-\delta$.

Remark 3.2.4. *The question of choosing ϵ is a balancing act. If $\epsilon \to 0$, then $r \geq 4/(\epsilon^2 - \epsilon^3)\log(...) \to \infty$. But, on the other hand, if $\epsilon \to 1$, the upper bound in* (3.10) *also becomes*

vacuous. For all practical purposes, we can think of $\epsilon \in [1/4, 3/4]$. *Also, this lemma tells us that the factor* $\sqrt{\frac{1-\epsilon}{1+\epsilon}}$ *plays a similar role to the* **column thresholding factor** g *as the two factors appear multiplied together in the same part of the above inequality. Therefore, it is fruitful to view and label the term* $\sqrt{\frac{1-\epsilon}{1+\epsilon}}$ *from Johnson-Lindenstrauss as the* **artificial column thresholding factor**.

Proof. By equations (3.8), the event

$$\bigcup_{\Pi_c} \Big(\mathcal{C}\left(\Pi_c\right) \bigcap \mathcal{D}\left(\Pi_c\right) \Big) \subseteq \left\{ \|S_k\|_{1,2} \leq \frac{1}{g} \sqrt{\frac{1+\epsilon}{1-\epsilon}} \left\|S_k(:,\alpha_k)\right\|_2, \text{ for all } 1 \leq k \leq n \right\}$$

defines a superset of the set of outcomes that satisfies our desired result. To show this take any fixed choice of column pivots Π_c. It is trivial to get $\|S_k(:,\alpha_k)\| \leq \frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}} \left\|S_k(:,\alpha_k)\right\|_2$ always. If our outcome is in both $\mathcal{C}\left(\Pi_c\right)$ and $\mathcal{D}\left(\Pi_c\right)$, then for any $1 \leq k \leq j \leq n$ and $j \neq \alpha_k$

$$\begin{aligned}
(1-\epsilon)\left\|S_k(:,j)\right\|_2^2 &\leq \left\| \frac{1}{\sqrt{r}} \Psi_k(:,j) \right\|_2^2 && \text{(by event } \underline{\mathbf{C}}_{j,k}^{\Pi_c}\text{)} \\
&\leq \frac{1}{g^2} \left\| \frac{1}{\sqrt{r}} \Psi_k(:,\alpha_k) \right\|_2^2 && \text{(by event } \mathcal{D}(\Pi_c)\text{)} \\
&\leq \frac{1}{g^2} (1+\epsilon) \left\|S_k(:,\alpha_k)\right\|_2^2 && \text{(by event } \overline{\mathbf{C}}_{\alpha_k,k}^{\Pi_c}\text{)}
\end{aligned}$$

Choosing $j = \arg\max_{k \leq j \leq n} \left\|S_k(:,j)\right\|_2$ gives us line (3.10) and our desired set containment. Next, we must bound the probability of success from below. It follows from the definition of conditional probability that

$$\begin{aligned}
\mathbb{P}\left(\bigcup_{\Pi_c} \left(\mathcal{C}\left(\Pi_c\right) \cap \mathcal{D}\left(\Pi_c\right) \right) \right) &= \sum_{\Pi_c} \mathbb{P}\left(\mathcal{C}\left(\Pi_c\right) \cap \mathcal{D}\left(\Pi_c\right) \right) \\
&= \sum_{\Pi_c} \mathbb{P}\left(\mathcal{C}\left(\Pi_c\right) | \mathcal{D}\left(\Pi_c\right) \right) \mathbb{P}\left(\mathcal{D}\left(\Pi_c\right) \right) \qquad (3.11)
\end{aligned}$$

Below we derive a lower bound on the right hand side of equation (3.11). Consider any given event $\mathcal{D}\left(\Pi_c\right)$ for which $\mathbb{P}\left(\mathcal{D}\left(\Pi_c\right)\right) > 0$. This implies that the permutation Π_c is given. As before, for each $1 \leq k \leq n-1$, let $S_k^{\Pi_c} \in \mathbb{R}^{(k:n)\times(k:n)}$ be the Schur complement after the first $k-1$ steps of column and row pivoting and Gaussian elimination, and let $\Omega_k^{\Pi_c} = \Omega(:, \Pi_r(k:n)) \in \mathbb{R}^{r \times (n-k+1)}$ be the submatrix of Ω whose columns correspond to the

rows of $S_k^{\Pi_c}$. With this notation, we can write

$$\begin{aligned}
\mathbb{P}\left(\mathcal{C}\left(\Pi_c\right)|\mathcal{D}\left(\Pi_c\right)\right) &= \mathbb{P}\left(\bigcap_{1\le k\le n}\left(\left(\bigcap_{\substack{k\le j\le n\\ j\neq\alpha_k}}\underline{\mathbf{C}}_{j,k}^{\Pi_c}\right)\bigcap\overline{\mathbf{C}}_{\alpha_k,k}^{\Pi_c}\right)\right)\\
&= 1-\mathbb{P}\left(\bigcup_{1\le k\le n}\left(\left(\bigcup_{\substack{k\le j\le n\\ j\neq\alpha_k}}\left(\underline{\mathbf{C}}_{j,k}^{\Pi_c}\right)^c\right)\bigcup\left(\overline{\mathbf{C}}_{\alpha_k,k}^{\Pi_c}\right)^c\right)\right)\\
&\ge 1-\sum_{k=1}^{n}\left(\sum_{\substack{j=k\\ j\neq\alpha_k}}^{n}\mathbb{P}\left(\left(\underline{\mathbf{C}}_{j,k}^{\Pi_c}\right)^c\right)+\mathbb{P}\left(\left(\overline{\mathbf{C}}_{\alpha_k,k}^{\Pi_c}\right)^c\right)\right)\\
&\ge 1-\sum_{k=1}^{n}(n-k+1)\exp\left(-\frac{(\epsilon^2-\epsilon^3)r}{4}\right)\\
&= 1-\frac{n(n+1)}{2}\exp\left(-\frac{(\epsilon^2-\epsilon^3)r}{4}\right)
\end{aligned}$$

where the third line comes from Lemma 3 and the fourth line is from the application of Lemma 5. Plugging this into line (3.11), we have that

$$\begin{aligned}
\mathbb{P}\left(\bigcup_{\Pi_c}\left(\mathcal{C}\left(\Pi_c\right)\cap\mathcal{D}\left(\Pi_c\right)\right)\right) &\ge \left(1-\frac{n(n+1)}{2}\exp\left(-\frac{(\epsilon^2-\epsilon^3)r}{4}\right)\right)\sum_{\Pi_c}\mathbb{P}\left(\mathcal{D}\left(\Pi_c\right)\right)\\
&= 1-\frac{n(n+1)}{2}\exp\left(-\frac{(\epsilon^2-\epsilon^3)r}{4}\right)
\end{aligned}$$

where the last line is achieved by line (3.9). In order to bound the last line from below by $1-\delta$, we require the $r > \dfrac{4}{\epsilon^2-\epsilon^3}\ln\left(\dfrac{n(n+1)}{2\delta}\right)$. □

Below is our main theoretical result on column growth upper bound for GERCP.

Theorem 8 (Column Growth Factor for GERCP)**.** *Choose $\epsilon,\delta\in(0,1)$ and $g\in(0,1]$. If $r > \dfrac{4}{\epsilon^2-\epsilon^3}\ln\left(\dfrac{n(n+1)}{2\delta}\right)$, then the pivot growth factor of Algorithm 1.2 satisfies*

$$\rho_{col}^{gercp}(\mathbf{A}) \le \frac{1}{g^2}\frac{1+\epsilon}{1-\epsilon}\sqrt{e(n+1)}n^{1+\ln\left(g\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)n^{\frac{1}{2}\ln(n)}}$$

with probability greater than $1-\delta$*, otherwise*

$$\rho_{col}^{gercp}(\mathbf{A}) \leq \frac{1}{\sqrt{n}} 2^{n-1}$$

Remark 3.2.5. *What happens if we are extremely unlucky? There is a chance, not exceeding the user-chosen positive probability* δ*, that at least one column in some Schur complement is not well-preserved. In this case, GERCP could end up picking the wrong column pivot. This, however, is far from a disaster as we still perform the deterministic partial pivoting at every step. Therefore, we can view these randomized column swaps as an "insurance policy" against large element growth because with high probability we will attain the growth bounds in Theorem 8, but we will definitely attain GEPP growth bounds.*

Proof. At the k^{th} stage of randomized complete pivoting, our column pivot choice is given by

$$\ell_k = \underset{k\leq j\leq n}{\arg\max} \|\Psi_k(:,j)\|_2^2$$
$$\alpha_k = \begin{cases} \ell_k & \text{if } g\|\Psi_k(:,\alpha_k)\|_2 \geq \|\Psi_k(:,k)\|_2 \\ k & \text{otherwise} \end{cases}$$

Then the row pivot at the k^{th} stage is given as β_k where

$$\beta_k = \underset{k\leq i\leq n}{\arg\max} |S_k(i,\alpha_k)| \tag{3.12}$$

At this point, we proceed in a fashion similar to Wilkinson's element growth proof for complete pivoting [97]. Let $p_k = |S_k(\beta_k,\alpha_k)|$ be the modulus of the pivot element of S_k. Also, let $c_k = \|S_k(:,\alpha_k)\|_2$ be the ℓ_2-norm of the pivot column of S_k. It is important to note that (3.12) along with Lemma 1 implies that $c_k \leq \sqrt{n-k+1}p_k$ (i.e. the last line works because $p_k = U(k,k)$ from our a priori pivoting so p_k is an entry of $S_k^{\Pi_c}$.) Then, we have that

$$\left|\det\left(S_k^{\Pi_c}(k:m,k:m)\right)\right| = \prod_{j=k}^{m} p_j \geq \prod_{j=k}^{m} \frac{1}{\sqrt{n-j+1}} c_j \tag{3.13}$$

which holds from the LU decomposition of $S_k^{\Pi_c}$ since the L factor is unit diagonal and the U factor has the p_j's as its diagonal. We can also apply Theorem 2 (Hadamard's Inequality) to the determinant of $S_k^{\Pi_c}(k:m,k:m)$ to get

$$\begin{aligned}\left|\det\left(S_k^{\Pi_c}(k:m,k:m)\right)\right| &\leq \prod_{j=k}^{m} \left\|S_k^{\Pi_c}(k:m,j)\right\|_2 \leq \prod_{j=k}^{m} \left\|S_k^{\Pi_c}(:,j)\right\|_2 \\ &\leq \left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)^{m-k} \|S_k(:,\alpha_k)\|_2^{m-k+1} = \left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)^{m-k} c_k^{m-k+1}\end{aligned} \tag{3.14}$$

where the last expression of the first line is achieved by taking the ℓ_2 norm of the entire j^{th} column instead of the first few entries of the j^{th} column. Also, the second line of the above is achieved from Theorem 7 by applying (3.10) for each $j \neq \alpha_k$. Combining our inequalities (3.13) and (3.14) for $|\det(S_k(k:m, k:m))|$, we get

$$\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)^{m-k} \sqrt{\frac{(n-k+1)!}{(n-m)!}} c_k^{m-k+1} \geq \prod_{j=k}^{m} c_j$$

for all $1 \leq k \leq m \leq n$. Define $q_k = \ln(c_k)$. Canceling one c_k on both sides and taking logarithms on both sides, we get

$$(m-k)\ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right) + \sum_{j=k}^{m} \ln\sqrt{n-j+1} + (m-k)\,q_k \geq \sum_{j=k+1}^{m} q_j$$

Dividing by $m-k$, moving each term with any q_j for $k \leq j < m$ to one side and everything else to the other,

$$q_k - \frac{1}{m-k}\sum_{j=k+1}^{m-1} q_j \geq \frac{1}{m-k} q_m - \frac{1}{m-k}\sum_{j=k}^{m} \ln\sqrt{n-j+1} - \ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)$$

Next, we combine all the inequalities for each k between $1 \leq k < m$ to get

$$\begin{pmatrix} 1 & -\frac{1}{m-1} & -\frac{1}{m-1} & \cdots & -\frac{1}{m-1} & -\frac{1}{m-1} \\ 0 & 1 & -\frac{1}{m-2} & \cdots & -\frac{1}{m-2} & -\frac{1}{m-2} \\ 0 & 0 & 1 & \cdots & -\frac{1}{m-3} & -\frac{1}{m-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -\frac{1}{2} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ \vdots \\ q_{m-2} \\ q_{m-1} \end{pmatrix} \geq \begin{pmatrix} \frac{1}{m-1} q_m \\ \frac{1}{m-2} q_m \\ \frac{1}{m-3} q_m \\ \vdots \\ \frac{1}{2} q_m \\ q_m \end{pmatrix} - \begin{pmatrix} \frac{1}{m-1}\ln\sqrt{\frac{n!}{(n-m)!}} \\ \frac{1}{m-2}\ln\sqrt{\frac{(n-1)!}{(n-m)!}} \\ \frac{1}{m-3}\ln\sqrt{\frac{(n-2)!}{(n-m)!}} \\ \vdots \\ \frac{1}{2}\ln\sqrt{\frac{(n-m+3)!}{(n-m)!}} \\ \ln\sqrt{\frac{(n-m+2)!}{(n-m)!}} \end{pmatrix} - \begin{pmatrix} \ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right) \\ \ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right) \\ \ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right) \\ \vdots \\ \ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right) \\ \ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right) \end{pmatrix}$$

and we express the above matrix inequality as

$$B \qquad \mathbf{q} \quad \overset{def}{\geq} \quad \mathbf{v}_1 \quad - \quad \mathbf{v}_2 \quad - \quad \mathbf{v}_3$$

Lemma 13 gives us a closed form expression for B^{-1}, which happens to be a non-negative matrix. Since B^{-1} only has non-negative entries, the vector inequality above is preserved after multiplying B^{-1} on both sides. To complete this proof, we only need the top row of this vector inequality. Since q_1 is the first entry of $\mathbf{q}$, we also apply $\mathbf{e}_1^T$ to both sides of the inequality to reduce it to a scalar inequality

$$q_1 = \mathbf{e}_1^T\mathbf{q} \geq \mathbf{e}_1^T B^{-1}\mathbf{v}_1 - \mathbf{e}_1^T B^{-1}\mathbf{v}_2 - \mathbf{e}_1^T B^{-1}\mathbf{v}_3 \tag{3.15}$$

Lemma 13 also tells us that the first row of B^{-1} is

$$\mathbf{e_1}^T B^{-1} = \begin{bmatrix} 1 & \frac{1}{m-1} & \frac{1}{m-2} & \cdots & \frac{1}{3} & \frac{1}{2} \end{bmatrix}$$

which we now use to compute/bound $\mathbf{e}_1^T B^{-1}\mathbf{v}_1$, $\mathbf{e}_1^T B^{-1}\mathbf{v}_2$ and $\mathbf{e}_1^T B^{-1}\mathbf{v}_3$:

$$\mathbf{e}_1^T B^{-1}\mathbf{v}_1 = q_m \left(\frac{1}{m-1} + \sum_{j=1}^{m-2} \frac{1}{(j+1)j} \right) = q_m$$

where the last equality was achieved by Lemma 12. Next, we compute $\mathbf{e}_1^T B^{-1}\mathbf{v}_2$ which we call the Wilkinson term

$$\begin{aligned}
\mathbf{e}_1^T B^{-1}\mathbf{v}_2 &= \frac{1}{m-1} \sum_{j=1}^{m} \ln \sqrt{n-m+j} + \sum_{k=1}^{m-2} \frac{1}{(k+1)k} \sum_{j=1}^{k+1} \ln \sqrt{n-m+j} \\
&= \frac{1}{m-1} \sum_{j=1}^{m} \ln \sqrt{n-m+j} + \sum_{j=1}^{m-1} \sum_{k=\max\{j-1,1\}}^{m-2} \frac{1}{(k+1)k} \ln \sqrt{n-m+j} \\
&= \frac{1}{m-1} \ln \sqrt{n} + \sum_{j=1}^{m-1} \left(\frac{1}{m-1} + \sum_{k=\max\{j-1,1\}}^{m-2} \frac{1}{(k+1)k} \right) \ln \sqrt{n-m+j} \\
&= \frac{1}{m-1} \ln \sqrt{n} + \sum_{j=1}^{m-1} \frac{1}{\max\{j-1,1\}} \ln \sqrt{n-m+j} \qquad \text{(Lemma 12)} \\
&= \ln \sqrt{n-m+1} + \sum_{j=2}^{m} \frac{1}{j-1} \ln \sqrt{n-m+j} \\
&\stackrel{def}{=} \ln \sqrt{n-m+1} + \ln f(m, n-m)
\end{aligned}$$

where we define the *generalized Wilkinson function* to be

$$f(m,t) \stackrel{def}{=} \sqrt{(2+t)^1 (3+t)^{\frac{1}{2}} (4+t)^{\frac{1}{3}} \cdots (m+t)^{\frac{1}{m-1}}}$$

Next comes $\mathbf{e}_1^T B^{-1}\mathbf{v}_3$ or the thresholding term

$$\mathbf{e}_1^T B^{-1}\mathbf{v}_3 = \ln \left(\frac{1}{g} \sqrt{\frac{1+\epsilon}{1-\epsilon}} \right) \sum_{j=1}^{m-1} \frac{1}{j} \leq (1 + \ln(m-1)) \ln \left(\frac{1}{g} \sqrt{\frac{1+\epsilon}{1-\epsilon}} \right)$$

Plugging all of this into (3.15), we have

$$q_1 \geq q_m - \ln \sqrt{n-m+1} - \ln f(m, n-m) - (1 + \ln(m-1)) \ln \left(\frac{1}{g} \sqrt{\frac{1+\epsilon}{1-\epsilon}} \right)$$

Taking the exponential of both sides and rearranging terms, we have that for all $1 \leq m \leq n$

$$\frac{c_m}{c_1} \leq \sqrt{n-m+1} f(m, n-m) \frac{1}{g} \sqrt{\frac{1+\epsilon}{1-\epsilon}} (m-1)^{\ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)} \tag{3.16}$$

Next, we need to relate the ratio $\frac{c_k}{c_1}$ to ρ_{col}^{gercp}

$$\begin{aligned}
\rho_{col}^{gercp} &= \frac{\max_k \|S_k\|_{1,2}}{\|S_1\|_{1,2}} \\
&\leq \frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}} \max_k \frac{\|S_k(:,\alpha_k)\|_2}{\|S_1(:,\alpha_1)\|_2} && \text{(Lemma 7)} \\
&= \frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}} \max_k \frac{c_k}{c_1} \\
&\leq \frac{1}{g^2}\frac{1+\epsilon}{1-\epsilon} \max_m \sqrt{n-m+1} f(m, n-m)(m-1)^{\ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)} && \text{(Eqn (3.16))} \\
&\leq \frac{1}{g^2}\frac{1+\epsilon}{1-\epsilon} \max_m \sqrt{e(n-m+2)}(n-m+1)(m-1)^{\ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)} \\
&\quad m^{\frac{1}{4}\ln(n)} m^{\frac{1}{4}\ln\left(\frac{n}{m}\right)} (n-m+1)^{\frac{1}{4}\ln(n-m+1)} \\
&\leq \frac{1}{g^2}\frac{1+\epsilon}{1-\epsilon} \max_m \left(\sqrt{e(n-m+2)}(n-m+1)(m-1)^{\ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)}\right) \\
&\quad \max_m \left(m^{\frac{1}{4}\ln(n)} m^{\frac{1}{4}\ln\left(\frac{n}{m}\right)}\right) \max_m (n-m+1)^{\frac{1}{4}\ln(n-m+1)} && \text{(Lemma 14)} \\
&\leq \frac{1}{g^2}\frac{1+\epsilon}{1-\epsilon} \sqrt{e(n+1)} n^{1+\ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)} \max_m \left(m^{\frac{1}{4}\ln(n)} m^{\frac{1}{4}\ln\left(\frac{n}{m}\right)}\right) n^{\frac{1}{4}\ln(n)}
\end{aligned}$$

where the third to last line is from Lemma 14. Examining the following derivative

$$\frac{d}{dm} \ln\left(m^{\frac{1}{4}\ln(n)} m^{\frac{1}{4}\ln\left(\frac{n}{m}\right)}\right) = \frac{\ln(n)}{m} - \frac{\ln(m)}{m}$$

which equals zero only when $m = n$, and given the concavity, this point attains the max. Thus, we plug this point in to get

$$\rho_{col}^{gercp} \leq \frac{1}{g^2}\frac{1+\epsilon}{1-\epsilon} \sqrt{e(n+1)} n^{1+\ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon}{1-\epsilon}}\right)} n^{\frac{1}{2}\ln(n)}$$

to get our desired result. □

Next, we slightly add to the statement of the last theorem to add a probabilistic guarantee on the floating point error of the sampling matrix Ψ under the sampling update formula (3.2). The floating error analysis of the sampling matrix must be done under the objective that rounding errors cannot corrupt the column lengths by too much, which is suggested by the following result under small probability of failure. This shows that the floating point error cannot grow quickly or exponentially under the given update scheme.

Theorem 9 (Stability of unconditionally stable sampling update formula)**.** *For Gaussian Elimination with Randomized Complete Pivoting, we have the two guarantees*

$$\rho_{col}^{gercp}(\mathbf{A}) \leq \frac{1}{g^2}\frac{1+\epsilon_{JL}}{1-\epsilon_{JL}}\sqrt{e(n+1)}n^{1+\ln\left(\frac{1}{g}\sqrt{\frac{1+\epsilon_{JL}}{1-\epsilon_{JL}}}\right)n^{\frac{1}{2}\ln(n)}}$$

and

$$\left\|fl\big(\widehat{\Psi}_{k+1}\big)-\widehat{\Psi}_{k+1}\right\|_{1,2} \leq \epsilon_{mach}(1+\epsilon_{mach})^k\sqrt{kr(1+\epsilon_{JL})}\Big(\sqrt{k}+5n^2\Big)\rho_{col}^{gercp}(A)\|A\|_{1,2}$$

with probability at least $1-\frac{n(n+1)}{2}\exp\left(-\frac{(\epsilon_{JL}^2-\epsilon_{JL}^3)r}{4}\right)-\exp\left(-\frac{(\epsilon_{JL}^2-\epsilon_{JL}^3)nr}{4}\right)$

Proof. First, we apply the relevant norm to both sides to get

$$\begin{aligned}
&\left\|fl\big(\widehat{\Psi}_{k+1}\big)-\widehat{\Psi}_{k+1}\right\|_{1,2} \qquad (3.17)\\
&\leq \epsilon_{mach}\left(1+\epsilon_{mach}\right)^k\left(\sum_{i=1}^{k}\left\|\Psi_i\left(:,(k+1):n\right)\right\|_{1,2}+5\left\||\Omega|\,|L_{k+1}|\right\|_2\left\||U_{k+1}(:,(k+1):n)|\right\|_{1,2}\right)\\
&\leq \epsilon_{mach}\left(1+\epsilon_{mach}\right)^k\left(\sum_{i=1}^{k}\left\|\Psi_i\left(:,(k+1):n\right)\right\|_{1,2}+5\left\|\Omega\right\|_F\left\|L_{k+1}\right\|_F\left\|U_{k+1}(:,(k+1):n)\right\|_{1,2}\right)
\end{aligned}$$

where we bound the 2-norm of the entrywise absolute value by the frobenius norm to get the last line. Next, Theorem 8 and the definition of the event $\overline{\mathbf{C}}_{\alpha_k,k}^{\Pi_c}$ from (3.4) give us both our desired bound on $\rho_{col}^{gercp}(\mathbf{A})$ and

$$\left\|\Psi_k\right\|_{1,2}=\left\|\Psi_k(:,\alpha_k)\right\|_2\leq\sqrt{r(1+\epsilon_{JL})}\left\|S_k\right\|_{1,2}\leq\sqrt{r(1+\epsilon_{JL})}\rho_{col}^{gercp}(\mathbf{A})\left\|\mathbf{A}\right\|_{1,2} \qquad (3.18)$$

with probability of failure bounded above by $\frac{n(n+1)}{2}\exp\left(-\frac{(\epsilon_{JL}^2-\epsilon_{JL}^3)r}{4}\right)$. Since GERCP is a top-heavy method from Definition 3, we have $\|L_{k+1}\|_F\leq\sqrt{kn}$ because L_{k+1} is a $n\times k$ matrix with each entry bounded above by 1 in absolute value. Also, Theorem 7 gives us that $\|U_{k+1}\|_{1,2}\leq n\rho_{col}^{gercp}(\mathbf{A})\|\mathbf{A}\|_{1,2}$. Finally, use Theorem 4 to control the quantity $\|\Omega\|_F$ via

$$\mathbb{P}\left\{\|\Omega\|_F^2\geq nr\left(1+\epsilon_{JL}\right)\right\}\leq\exp\left(-\frac{(\epsilon_{JL}^2-\epsilon_{JL}^3)nr}{4}\right)$$

Then plug this and the inequality (3.18) into the first inequality (3.17) to arrive at our desired conclusion with a union bound. □

Chapter 4

Numerical Experiments

4.1 Block GERCP

Algorithm 3.2.2 was presented in a form for ease of presentation of Theorem 8. For efficient numerical implementation, we need to develop a block version of GERCP in Algorithm 3.2.2 to increase locality and memory/cache re-use with more BLAS-3 calls. Algorithm 4 below is styled after dgetf2.f and dgetrf.f for block GEPP in LAPACK with double precision floating point numbers. Instead of increasing the pivot-block size from $p = 1$, we introduce a loop-blocksize parameter $b \geq 1$. It repeatedly performs b steps of randomized column pivoting and partial pivoting followed by a blocksize b Schur complement update.

When evaluating the norms to make the column pivoting decisions, it is not necessary to take the square root after computing the sum of squares. The motivation for using the 2-norm to make the column pivoting decisions comes from the use of a Gaussian sampling matrix in combination with the Johnson-Lindenstrauss Lemma. As pointed out by Prof. James Demmel, computing the 2-norm is more involved than computing the 1-norm (i.e. sum of absolute values) in order to avoid overflow in floating point arithmetic with the BLAS-1 function snrm2 [16]. This is an interesting direction for future research. For the 2-norm, one follow the example of snrm2 without the final square root in order to obtain maximum reliability. However, given that we use $r = 4$ for all of our examples, we are able to unroll the loops present in snrm2 in order to improve the algorithm runtime as done in our code.

The main work of Algorithm 4 is in the last step, the repeated computation of the matrix $A(\overline{k}+1:n, \overline{k}+1:n)$. The outer loop for $\underline{k}$ is similar to the main loop in dgetrf.f, while the inner loop for k is similar to the main loop in dgetf2.f. The main modifications occur on lines $7, 8$ and the BLAS-3 Schur complement update after the end of the inner loop. Line 7 updates the U factor so that we can use it to update the sketching matrix on line 8. As in dgetf2.f, the inner loop for k is designed to work on a tall-skinny matrix, whereas the outer loop for $\underline{k}$ performs fast BLAS-3 updates on the rest of the matrix. For practical reasons, we stop using the sampling matrix once the dimensions of the Schur complement become

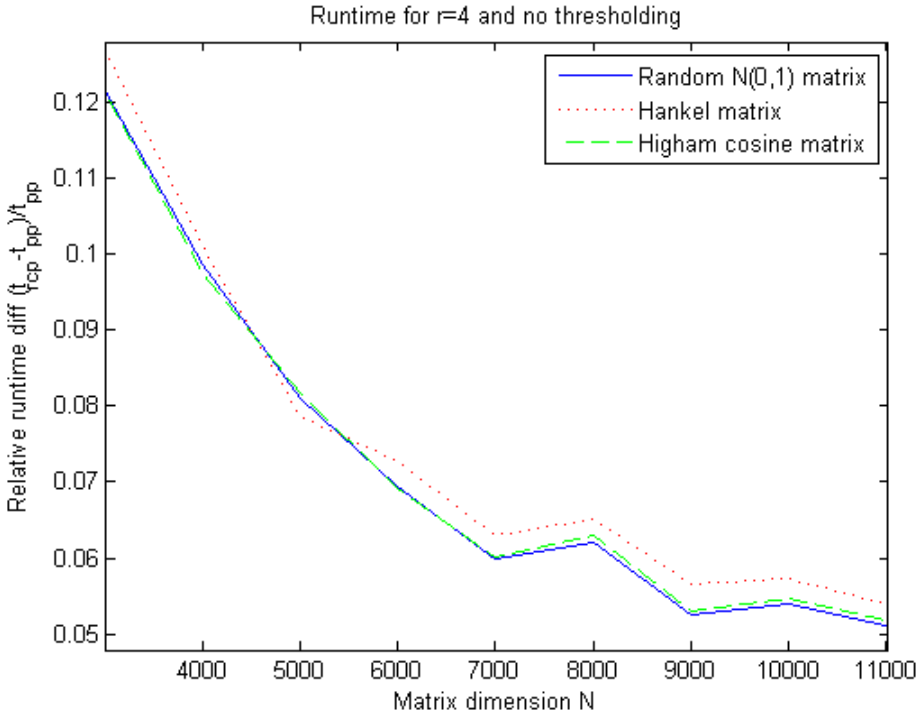


Figure 4.1: Comparing the run times of GERCP and GEPP Fortran code each averaged over 10 different trials

less than or equal to the sampling dimension $r > 0$. It is important to note that the proof of Theorems 8 and 9 is easily modified to apply to this versions of the algorithm with the exact same guarantees on element growth. We present the procedure for updating the Schur sampling matrix Ψ.

Remark 4.1.1. *The first updating formula in Algorithm 5 is slightly more efficient than the second one. While we have not observed it in our numerical computations, it potentially could lead to inaccurate column selections for some highly ill-conditioned matrices in pathological cases. Our numerical experiments also suggest that the execution time of Algorithm 5 is typically a small fraction of the total execution time of of GERCP. Thus, one might use the second updating formula in Algorithm 5 for a more robust numerical implementation.*

4.2 Numerical Results

We ran our experiments on two different machines. The runtime results were performed on a single node of NERSC's Carver machine with two quad-core Intel Xeon X5550 2.67 GHz processors and 24 GB of RAM. This compute resource was courtesy of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by

Algorithm 4 Block GERCP

Inputs: $n \times n$ matrix A, sampling dimension $r > 0$, block size b

Outputs: lower triangular L with unit diagonal, upper triangular U, row permutation Π_r, column permutation Π_c.

sample $\Omega(i,j) \sim \mathcal{N}(0,1)$ for all $1 \le i \le r$ and $1 \le j \le n$
compute $\Psi = \Omega A$
for $\underline{k} = 1 : b : n-1$ **do**
  set $\overline{k} = \underline{k} + \min\{b, n - \underline{k} + 1\} - 1$
  for $k = \underline{k} : \overline{k}$ **do**
    compute $\alpha = \begin{cases} \arg\max_{k \le j \le n} \|\Psi(:,j)\|_2^2 & \text{if } n - k \ge r \\ \arg\max_{k \le j \le n} \|A(k:n,j)\|_2^2 & \text{otherwise} \end{cases}$.
    swap columns k and α of A, Ψ and Ω (*).
    compute $\beta = \arg\max_{k \le i \le n} |A(i,k)|$.
    swap rows k and β of A (*).
    compute $A(k{+}1{:}n, k) = A(k{+}1{:}n, k)/A(k,k)$;
    compute $A(k{+}1{:}n, k{+}1{:}\overline{k}) = A(k{+}1{:}n, k{+}1{:}\overline{k}) - A(k{+}1{:}n, k) * A(k, k{+}1{:}\overline{k})$;
    compute $A(k, \overline{k}{+}1{:}n) = A(k, \overline{k}{+}1{:}n) - A(k, \underline{k}{:}k{-}1) * A(\underline{k}{:}k{-}1, \overline{k}{+}1{:}n)$;
    update $\Psi(:, k{:}n)$ with Algorithm 5
  end for
  compute $A(\overline{k}{+}1{:}n, \overline{k}{+}1{:}n) = A(\overline{k}{+}1{:}n, \overline{k}{+}1{:}n) - A(\overline{k}{+}1{:}n, \underline{k}{:}\overline{k}) * A(\underline{k}{:}\overline{k}, \overline{k}{+}1{:}n)$;
end for

Algorithm 5 Update procedure for Schur sampling matrix Ψ

Inputs: $r \times n$ matrix Ψ, $n \times n$ working matrix A, $r \times n$ random matrix Ω

Outputs: $r \times m$ matrix Ψ

if pivot $|A(k,k)| \ge \sqrt{\epsilon_{mach}} \|\Psi_1\|_{1,2}$ **then** apply Eqn (3.3) with **then**
  $\Psi(:,(k{+}1){:}n) \longleftarrow \Psi(:,(k{+}1){:}n) - \dfrac{\Psi(:,k)A(k,(k{+}1){:}n)}{A(k,k)}$
else apply Eqn (3.2) with
  $\Psi(:,(k{+}1){:}n) \longleftarrow \Psi(:,(k{+}1){:}n) - [\Omega(:,k) + \Omega(:,(k{+}1){:}n)A((k{+}1){:}n,k)]A(k,(k{+}1){:}n)$
end if

N	3000	5000	7000	9000	11000
t_{rcp} (secs)	2.250	9.623	25.506	53.607	96.662
t_{pp} (secs)	2.006	8.902	24.066	50.934	91.967
$\frac{t_{rcp}-t_{pp}}{t_{pp}}$	12.20%	8.10%	6.00%	5.20%	5.10%

Table 4.1: Average run times of GERCP and GEPP over 10 separate trials.

the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. The rest of the numbers are generated on a laptop with an Intel i7-3632QM CPU and 8GB of RAM. All of the code here was run using the Intel MKL BLAS [16, 58] version 11.0.1 with Intel Fortran compiler version 13.0.1. We used the open source Netlib 3.3.1 version of LAPACK GEPP [9]. Our version of GERCP was obtained by modifying the Netlib GEPP Fortran code. This allows for an easy and fair comparison between GEPP and GERCP by insuring that the version of GEPP used to compare against has similar cache optimizations. It is worth noting that the Intel MKL version of GEPP is much faster than both Netlib GEPP and GERCP because of superior cache optimizations. Figure 4.1 from our runtime experiments shows that as the matrix size increases, the percent difference in runtime decreases to a negligible amount. This agrees with our theory, which tells us the $O(n^2)$ operations required to maintain the sampling matrix and pivot columns does not grow as quickly as the $O(n^3)$ operations required to actually factor the matrix A. Even when the relative time difference is high, the absolute time difference is a fraction of a second for a single factorization as shown in by the run times for $N = 3000$ in Table 4.1 below.

In section 2.4.2, we reviewed stability issues associated with to most commonly used Gaussian elimination pivoting strategy, partial pivoting. Now, we produce numerical experiments showing the improved stability properties of GERCP. As described in section 2.3, two metrics for judging the quality of an LU factorization are backwards error and element growth , given as

$$\left\|\Pi_c A \Pi_r^T - LU\right\|_\infty \quad \text{and} \quad \rho_{elem}\left(\mathbf{A}\right) = \frac{\max_{i,j,k}\left|S_k(i,j)\right|}{\max_{i,j}\left|\mathbf{A}(i,j)\right|}$$

The Wilkinson, Generalized Wilkinson and Volterra matrices that we use in Figures 4.2 and 4.3 are as described in Section 2.4.2. The Wilkinson-type matrices serve as the worst case matrix for GEPP instability with entries that grow exponentially, where the standard Wilkinson matrix has the quickest exponential growth with a base of 2. This is exemplified by the dashed blue and red lines in the log-log plots of element growth and backwards error of GEPP in Figures 4.2 and 4.3. However, GERCP fixes this by impeding element growth with column pivots to leave the backwards error near machine precision. As far as Gaussian elimination goes, the most pathological examples, that we characterize as *passive aggressive* element growth, include the Volterra matrix. These matrices exhibit just enough element growth, but no more, to cause an unacceptable level of backwards error. In contrast, the

element/column growth for the Wilkinson-type matrices is so massive that the problem is trivial to detect and fix for any GE algorithm with both row and column pivots to correct. This case too is effortlessly corrected by GERCP as shown by our experiments.

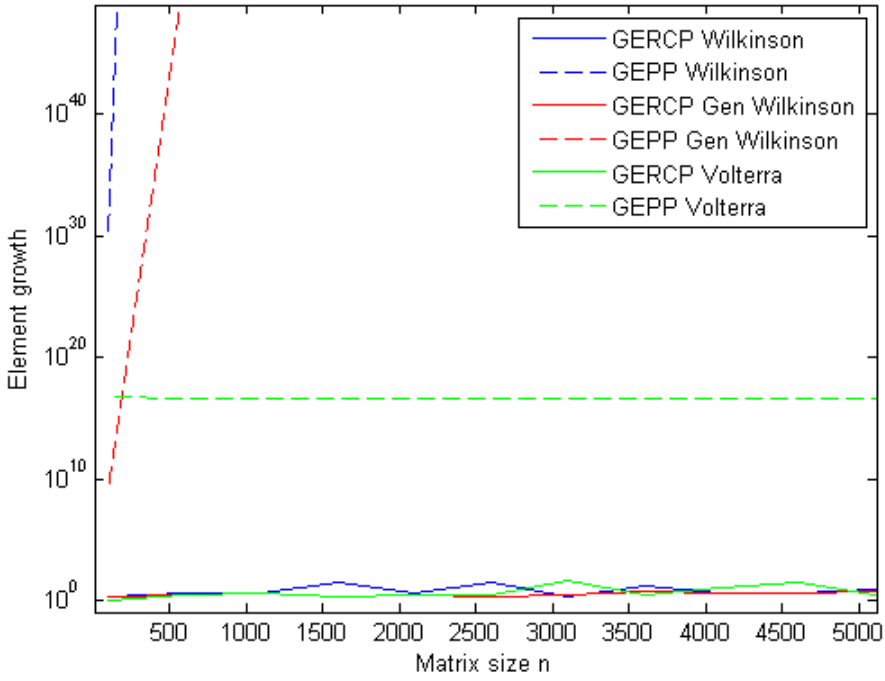


Figure 4.2: Element growth for diabolical matrices with GERCP and GEPP Fortran code

4.3 Backward error for random linear systems

Suppose we wish to solve the linear system $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} \in \mathbb{R}^{n\times n}$ iid $\mathcal{N}(0,1)$ standard normal and $\mathbf{b} \in \mathbb{R}^n$ iid $\mathcal{N}(0,1)$ standard normal. This system is known to be well conditioned [23] and the LU factorization is stable under partial pivoting and complete pivoting [90]. We measure the accuracy of a linear solve with the *relative residual*

$$\frac{\|\mathbf{A}\widehat{\mathbf{x}} - \mathbf{b}\|_\infty}{\|\mathbf{A}\|_\infty \|\widehat{\mathbf{x}}\|_\infty}$$

In Figure 4.4, we plot the relative residual error for GERCP with different sampling parameters $r > 0$, along with competing methods like GEPP, GECP, GERP (rook) and GE2CP. While complete pivoting consistently obtained the smallest residual, GERCP, GERP and GE2CP all produced similar relative residuals which were clearly better than GEPP and not

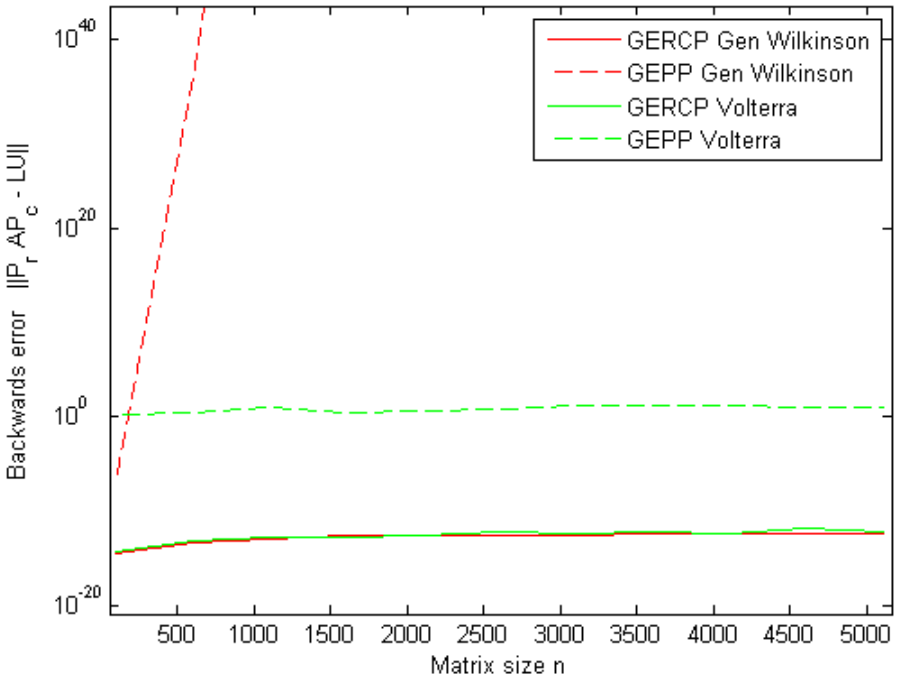


Figure 4.3: Backwards error for diabolical matrices with GERCP and GEPP Fortran code

much worse than that of GECP. This shows that GERCP produces a better quality solution than GEPP even when GEPP is given a well-conditioned system. The Figure 4.4 suggests that the relative residual for the random normal linear system is improved by almost a factor of 2. This savings becomes more important when you work with smaller precision floating point numbers like single or half precision floating point numbers. These smaller precision floating point numbers are becoming commonly used on co-processor platforms like GPGPUs as result in dramatic run time improvements. Also, for different linear systems this improvement in the relative residual can be much higher as in the case of the Wilkinson-type and Volterra matrices.

We also look at the element growth within the LU factors for different pivoting strategies. In [90], Trefethen and Schreiber study the element growth factors for GEPP and GECP on standard normal matrices. They conjecture that $\mathbb{E}\left(\rho_n^{gepp}\right) \approx O\left(n^{2/3}\right)$ and $\mathbb{E}\left(\rho_n^{gecp}\right) \approx O\left(n^{1/2}\right)$. We plot the element growth for GEPP and GECP along with the element growth for GERP (rook), GE2CP and GERCP with different values of the sampling parameter $r > 0$ in Figure 4.5.

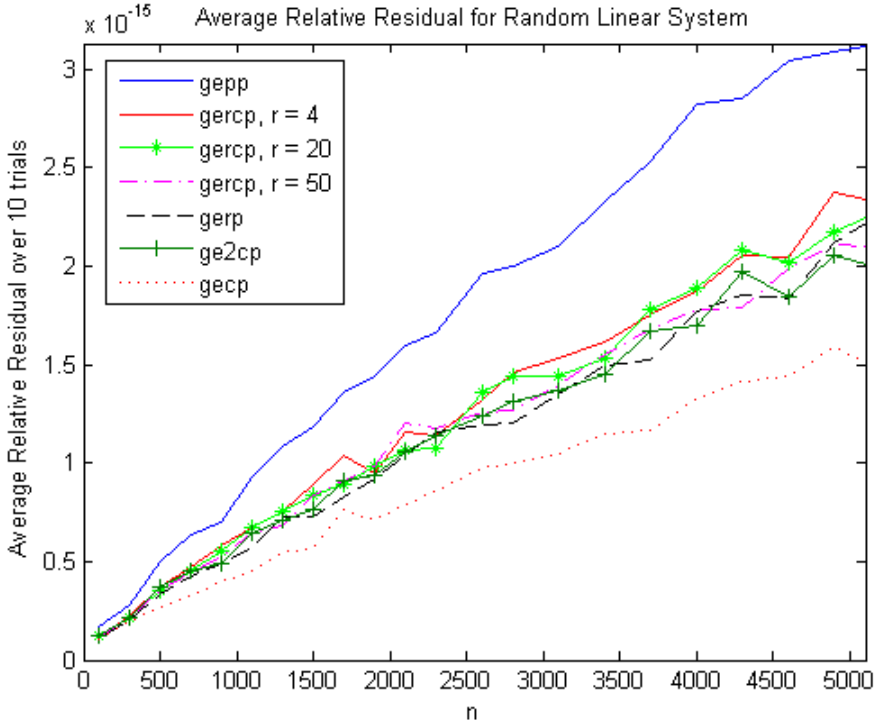


Figure 4.4: Average Relative Residual over 10 trials. This suggests that GERCP should improve the relative residual of a linear solve over GEPP by at least half the improvement that GECP would provide.

4.4 Incomplete LU with Randomized Complete Pivoting

We can also rewrite our GERCP algorithm into a form that makes it more amenable for low-rank incomplete LU factorizations. This actually allows us to use GERCP to produce CUR-style decompositions [31], which are popular in machine learning and randomized numerical linear algebra, as discussed in Part II of this dissertation. The pivoting decisions of GERCP are used to select relevant rows and columns of $\mathbf{A}$. Suppose we wish to perform ℓ steps of LU to form a low-rank approximation $\mathbf{A}^{\Pi_{c,\ell}} \approx L_\ell U_\ell$. In machine learning, researchers [27, 37, 13] have produced efficient algorithms for a similar problem in the context of the incomplete Cholesky factorization for a symmetric $n \times n$ input matrix. The incomplete Cholesky factorization enjoys an impressive complexity of $O(n\ell^2)$, i.e. linear in n. They provide a version where the user stipulates a fixed desired ℓ parameter and an adaptive version that chooses ℓ to be the first number that causes the residual to drop below a user defined tolerance τ, i.e. $\|\mathbf{A}^{\Pi_{c,\ell}} - L_\ell U_\ell\|_F \leq \tau$. In order to make our provably stable GERCP algorithm competitive for this purpose, we must produce a left-looking version [87]

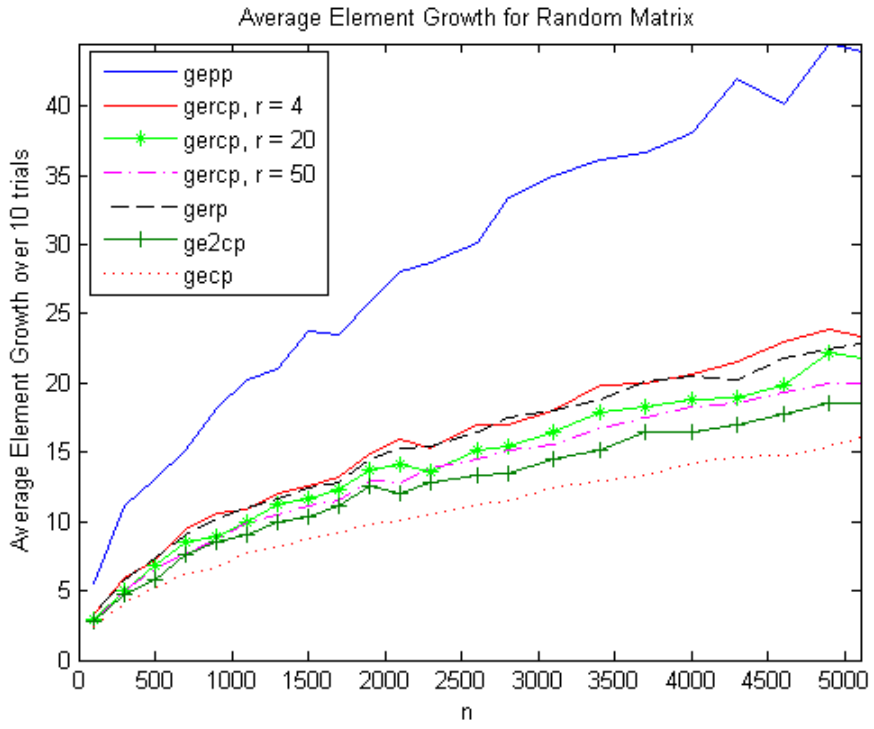


Figure 4.5: Average element growth of iid standard Normal random matrix over 10 trials.

of GERCP as in Algorithm 6. Left-looking Gaussian elimination attempts to leave the Schur complement untouched until it is absolutely needed. Lines $8-10$ of Algorithm 6 are used to update the borders of the Schur complement from scratch. Line 10 is especially needed to update the sampling matrix Ψ for the next stage. The complexity of Left-Looking GERCP is $O\left(6n\ell^2 + 2nr\ell\right)$. If we strictly follow the requirements of Theorem 9 or Theorem 8, then this becomes $O(n\log(n)\ell)$. However, if we let $r = O(1)$ as done in the numerical experiments then we also have complexity $O(n\ell^2)$. For the adaptive version, it is crucial complexity-wise that we make us of the compressed $r \times (n-k+1)$ Schur sketching matrix instead of a larger $(n-k+1)\times(n-k+1)$ matrix, otherwise we would need to replace an r term with an n term in the above complexity. To further justify the use of $\|\Psi(:,k\!:\!n)\|_F$ in line 3 of Algorithm 6

$$\mathbf{A}^{\Pi_{c,k}} = \begin{matrix} \\ k \\ m-k \end{matrix}\!\!\begin{matrix} \begin{matrix} k & n-k \end{matrix} \\ \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \end{matrix} = \begin{matrix} \\ k \\ m-k \end{matrix}\!\!\begin{matrix} \begin{matrix} k & m-k \end{matrix} \\ \begin{pmatrix} L_{11} & 0 \\ L_{21} & I \end{pmatrix} \end{matrix} \begin{matrix} \begin{matrix} k & n-k \end{matrix} \\ \begin{pmatrix} U_{11} & U_{12} \\ 0 & S_\ell \end{pmatrix} \end{matrix} = L_kU_k + \begin{matrix} \begin{matrix} k & n-k \end{matrix} \\ \begin{pmatrix} 0 & 0 \\ 0 & S_k \end{pmatrix} \end{matrix}$$

The application of Frobenious norms then gives $\|\mathbf{A}^{\Pi_{c,k}} - L_kU_k\|_F = \|S_k\|_F$. At this point, it is useful to recall equation 3.1 that $\Psi_\ell = \Omega\left(:, \pi_{r,k}(k:n)\right) S_k$ where Ω has iid standard normal entries. This allows us to apply Theorem 4 to get

Corollary 1 (Scaled Frobenious norm of Schur Sketching Matrix approximates Frobenious

Algorithm 6 Left-Looking GERCP

Inputs: $n \times n$ matrix A, sampling dimension $r \geq 1$ and either $\ell > 0$ or tolerance $\tau > 0$
Outputs: lower triangular L with unit diagonal, upper triangular U, row permutation Π_r, column permutation Π_c

sample $\Omega(i,j) \sim \mathcal{N}(0,1)$ for all $1 \leq i \leq r$ and $1 \leq j \leq n$
compute $\Psi = \Omega A$
while $k \leq \ell$ or $\|\Psi(:, k\!:\!n)\|_F \leq \tau$ **do**
    compute $\alpha = \underset{k \leq j \leq n}{\arg\max} \|\Psi(:,j)\|_2$.
    swap columns k and α of A and Ψ.
    compute $\beta = \underset{k \leq i \leq n}{\arg\max} |A(i,k)|$.
    swap rows k and β of A.
    compute $A(k\!:\!n, k) = A(k\!:\!n, k) - A(k\!:\!n, 1:k-1) * A(1:k-1, k)$;
    compute $A(k\!+\!1\!:\!n, k) = A(k\!+\!1\!:\!n, k)/A(k,k)$;
    compute $A(k, k\!+\!1\!:\!n) = A(k, k\!+\!1\!:\!n) - A(k, 1\!:\!k\!-\!1) * A(1\!:\!k\!-\!1, k\!+\!1\!:\!n)$;
    update $\Psi(:, k\!:\!n)$ with Algorithm 5
end while

norm of Schur complements)**.** *For each LU stage* $1 \leq k \leq n$ *we have that the Frobenious norm of* $\frac{1}{\sqrt{r}}\Psi_k$ *approximates the Frobenious norm of the corresponding Schur complement* S_k

$$\mathbb{P}\left((1-\epsilon_{JL})\|S_k\|_F^2 \geq \|\frac{1}{\sqrt{r}}\Psi_k\|_F^2\right) \leq \exp\left(-r\frac{\epsilon_{JL}^2+\epsilon_{JL}^3}{4}\frac{\|S_k\|_F^2}{\|S_k\|_2^2}\right)$$
$$\mathbb{P}\left((1+\epsilon_{JL})\|S_k\|_F^2 \leq \|\frac{1}{\sqrt{r}}\Psi_k\|_F^2\right) \leq \exp\left(-r\frac{\epsilon_{JL}^2-\epsilon_{JL}^3}{4}\frac{\|S_k\|_F^2}{\|S_k\|_2^2}\right)$$

There are a few interesting observations here. The probability of failure does not depend upon the dimension n of the original matrix! This is because we did not do any union bounding of column norm estimates to arrive at this. Also, these bounds suggest that a larger numerical rank of S_k provides for a more accurate estimate. Even more surprising is that if our adaptive algorithm terminates after ℓ steps, then we know that $\frac{1}{\sqrt{r}}\Psi_k$ provided accurate approximations to the Frobenious norm of the Schur complement all ℓ required steps with probability at least

$$\mathbb{P}\left((1-\epsilon_{JL})\|S_k\|_F^2 \leq \|\frac{1}{\sqrt{r}}\Psi_k\|_F^2 \leq (1+\epsilon_{JL})\|S_k\|_F^2, \forall 1 \leq k \leq \ell\right) \geq 1 - 2\ell\exp\left(-r\frac{\epsilon_{JL}^2-\epsilon_{JL}^3}{4}\frac{\|S_k\|_F^2}{\|S_k\|_2^2}\right)$$

This means that the ability to accurately estimate the Frobenious norm for ℓ stages only requires the sampling dimension r to be $O(\log(\ell))$ instead of $O(\log(n))$. We test this estimation technique of the Frobenious norm of the Schur complement at each step of an imcomplete LU factorization for the KOS blog dataset from the UCI Machine Learning Repository [67]. This dataset gives the standard term-document matrix with rows that represent documents

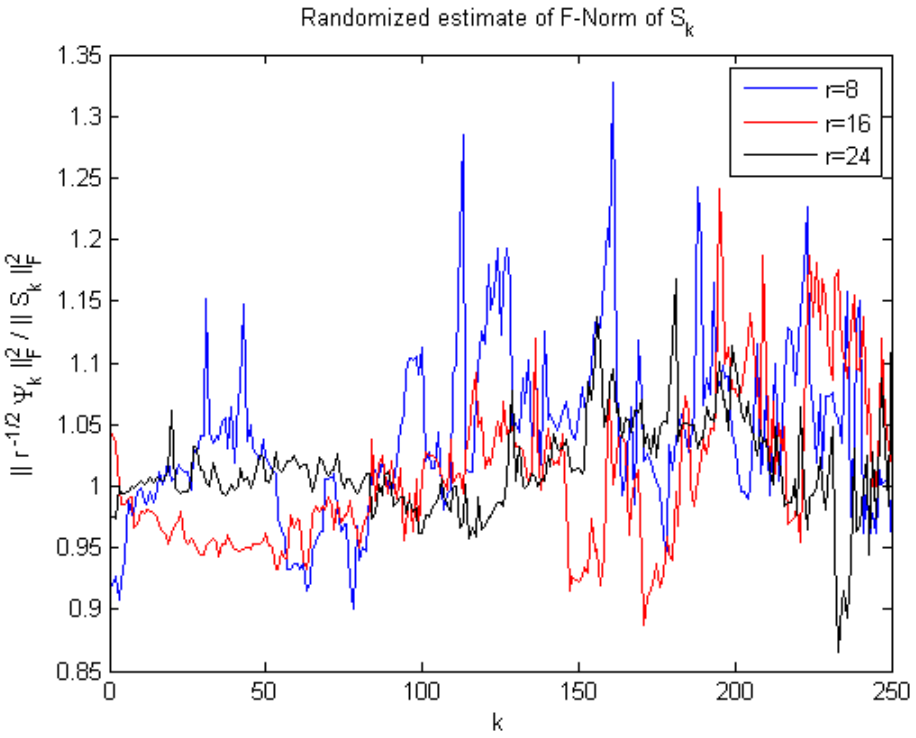


Figure 4.6: One trial incomplete factorization for each value of the sampling dimension r

and columns that represent the number of times a particular word was used in a document. This dataset has 3430 documents and a vocabulary of 6906 words. We run incomplete LU factorizations for 250 steps and Figure 4.6 shows how well the randomized Schur complement norm estimate in terms of the ratio $\frac{\|\frac{1}{\sqrt{r}}\Psi_k\|_F^2}{\|S_k\|_F^2}$ for different values of r.

Future Work. One direction is to study and provide guarantees for variants of GERCP that allow for a large pivot-block size $p > 1$. We believe that such a method could be much faster due to an increase in BLAS-3 operations, and could also provide rank-revealing style guarantees for low-rank approximations generated by incomplete LU factorizations [77]. It will also be important to develop a cache optimized version if this code to be competitive to Intel MKL LAPACK GEPP. Another important avenue for future research is to use these techniques to make communication avoiding tournament pivoting LU (CALU) more stable by adding randomized column pivots [29, 46].

Acknowledgments. I thank my adviser and co-author Prof. Ming Gu for helping me with this work and for give me permission to include this co-authored material in my thesis. We also wish to thank James Demmel and Laura Grigori for many helpful and fruitful discussions.

Chapter 5

Additional Lemmas and Proofs

5.1 Matrix version of Johnson-Lindenstrauss Concentration of Measure

Let $A \in \mathbb{R}^{r_1 \times m}$ and $B \in \mathbb{R}^{n \times r_2}$ be fixed deterministic matrices, where $r_1 < m$ and $r_2 < n$. Let $\Omega \in \mathbb{R}^{m \times n}$ have iid entries distributed standard normal $\mathcal{N}(0,1)$. In this section, we will derive a general concentration of measure result for the random quantity

$$\|A\Omega B\|_F^2$$

which not only generalizes the Johnson-Lindenstrauss concentration of measure result, but is proved using similar methods. First, we start by specifying the SVD for $A = U_1 \Sigma V_1^T$ and $B = U_2 \Theta V_2^T$ where

$$\Sigma = \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_{r_1} \end{pmatrix} \quad \text{and} \quad \Theta = \begin{pmatrix} \theta_1 & & & \\ & \theta_2 & & \\ & & \ddots & \\ & & & \theta_{r_2} \end{pmatrix}$$

Lemma 8 (Rotational invariance and Orthogonality)**.** *We have that*

$$\|A\Omega B\|_F^2 \overset{dist}{=} \sum_{i=1}^{r_1} \sum_{j=1}^{r_2} \sigma_i^2 \theta_j^2 \omega_{i,j}^2$$

where $\overset{dist}{=}$ *denotes equality in distribution and where each* $\omega_{i,j}$ *is independent and identically distributed as* $\omega_{i,j} \sim \mathcal{N}(0,1)$ *for all* $1 \leq i \leq r_1$ *and* $1 \leq j \leq r_2$.

Proof.

$$\begin{aligned} \|A\Omega B\|_F^2 &= \|U_1 \Sigma V_1^T \Omega U_2 \Theta V_2^T\|_F^2 \\ &= \|\Sigma V_1^T \Omega U_2 \Theta\|_F^2 \qquad \text{(Unitary invariance of F-norm)} \\ &= \|\Sigma \widehat{\Omega} \Theta\|_F^2 \end{aligned}$$

where $\widehat{\Omega} = V_1^T \Omega U_2$ is also distributed iid $\mathcal{N}(0,1)$ by the rotational invariance of that random matrix. Let $\omega_{i,j}$ denote the entry of $\widehat{\Omega}$ in the i^{th} row and j^{th} column. Computing the triple product in the Frobenious norm on the last line and applying the entrywise orthogonality of the Frobenious norm, we arrive at our result

$$\|A\Omega B\|_F^2 = \sum_{i=1}^{r_1} \sum_{j=1}^{r_2} \sigma_i^2 \theta_j^2 \omega_{i,j}^2$$

□

To produce our tail bounds, we will rely on Chernoff's inequality, which comes from applying Markov's inequality on the exponentiated version of our random variable. To do this, it is useful to define the moment generating function of a random variable.

Definition 6 (Moment Generating Function)**.** *Let $X \in \mathbb{R}$ be a random variable. Then, we define the moment generating function to be*

$$M_X(t) = \mathbb{E}_X\left[e^{tX}\right]$$

Lemma 9 (MGF for linear combinations of independent RVs)**.** *Let $a, b \in \mathbb{R}$ be constants and let $X, Y \in \mathbb{R}$ be two independent random variables with $M_X(t)$ and $M_Y(t)$ as moment generating functions respectively. Then we have that the moment generating function of $Z = aX + bY$ is given as*

$$M_Z(t) = M_X(at) M_Y(bt)$$

Next, we compute the moment generating functionod a squared standard normal random variable

Lemma 10 (MGF for squared standard normal RV)**.** *Let ω be a $\mathcal{N}(0,1)$ random variable. Then the moment generation function for ω^2 is given as*

$$M_{\omega^2}(t) = \left(\frac{1}{1-2t}\right)^{\frac{1}{2}}$$

for all $t < \frac{1}{2}$

Proof. We have that $\omega^2 \propto \mathcal{X}_1^2$ is chi-squared distributed with the PDF $f_{\omega^2}(x) = \frac{1}{\sqrt{2\pi x}} e^{-\frac{1}{2}x}$. With this, we compute the moment generating function

$$M_{\omega^2}(t) = \int_0^\infty e^{tx} \frac{1}{\sqrt{2\pi x}} e^{-\frac{1}{2}x} dx = \sqrt{\frac{1}{1-2t}} \left(\sqrt{\frac{2}{\pi}} \int_0^\infty e^{-\frac{1}{2}y^2} dy\right) = \sqrt{\frac{1}{1-2t}}$$

where $y > 0$ is the change of variables such that $y^2 = (1-2t)x$. □

Before continuing, we need to prove the following technical lemma

Lemma 11. *Let $0 \leq \gamma \leq 1$ and let $|t| < \frac{1}{2}$. Then, we have*

$$\left(\frac{1}{1-2\gamma t}\right)^{1/2} \leq \left(\frac{1}{1-2t}\right)^{\gamma/2}$$

Proof. Fix γ and let us Taylor expand the following function around $t_0 = 0$

$$\begin{aligned}(1-2t)^{\gamma} &= 1 - 2\gamma\,(1-2t_0)^{\gamma-1}\,t + \int_{x=t_0}^{t} (t-x)\gamma\,(\gamma-1)\,(1-2x)^{\gamma-2}\,dx \\ &= 1 - 2\gamma t + \gamma\,(\gamma-1)\int_{x=t_0}^{t} (t-x)\,(1-2x)^{\gamma-2}\,dx \text{ (non-positive integral)} \\ &\leq 1 - 2\gamma t\end{aligned}$$

By division, we can rewrite this as $\frac{1}{1-2\gamma t} \leq \left(\frac{1}{1-2t}\right)^{\gamma}$. We arrive at our result by taking square-roots on both sides. □

Now, we can present the proof of the generalization of Johnson-Lindenstrauss concentration of measure. We wish to prove the following statement of Theorem 4. For any fixed $\epsilon > 0$, we want the tail bounds

$$\mathbb{P}\left(\|A\Omega B\|_F^2 \geq (1+\epsilon)\|A\|_F^2\|B\|_F^2\right) \leq \exp\left(-\left(\frac{\epsilon^2}{4} - \frac{\epsilon^3}{6}\right)\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}\right) \tag{5.1}$$

and

$$\mathbb{P}\left(\|A\Omega B\|_F^2 \leq (1-\epsilon)\|A\|_F^2\|B\|_F^2\right) \leq \exp\left(-\left(\frac{\epsilon^2}{4} + \frac{\epsilon^3}{6}\right)\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}\right) \tag{5.2}$$

Proof of Theorem 4. Lets consider the probability

$$\begin{aligned}\mathbb{P}\left(\|A\Omega B\|_F^2 \geq (1+\epsilon)\|A\|_F^2\|B\|_F^2\right) &= \mathbb{P}\left(\frac{\|A\Omega B\|_F^2}{\|A\|_F^2\|B\|_F^2} \geq 1+\epsilon\right) \\ &= \mathbb{P}\left(\sum_{i=1}^{r_1}\sum_{j=1}^{r_2}\frac{\sigma_i^2}{\|A\|_F^2}\frac{\theta_j^2}{\|B\|_F^2}\omega_{i,j}^2 \geq 1+\epsilon\right)\end{aligned}$$

For convenince, we refer to $Z = \sum_{i=1}^{r_1}\sum_{j=1}^{r_2}\lambda_i\mu_j\omega_{i,j}^2$, where $\lambda_i = \frac{\sigma_i^2}{\|A\|_F^2}$ and $\mu_j = \frac{\theta_j^2}{\|B\|_F^2}$. Proceding with the standard Chernoff inequality trick to get a moment generating function

for Z via Markov's inequality, we get

$$\begin{aligned}
\mathbb{P}\left(Z \geq 1+\epsilon\right) &= \mathbb{P}\left(e^{tZ} \geq e^{(1+\epsilon)t}\right) \\
&\leq \frac{\mathbb{E}\left[e^{tZ}\right]}{e^{(1+\epsilon)t}} \\
&= e^{-(1+\epsilon)t} M_Z(t) \\
&= e^{-(1+\epsilon)t} \prod_{i=1}^{r_1}\prod_{j=1}^{r_2} M_{\omega_{i,j}^2}(\lambda_i \mu_j t) \qquad \text{(Lemma 9)} \\
&= e^{-(1+\epsilon)t} \prod_{i=1}^{r_1}\prod_{j=1}^{r_2} \left(\frac{1}{1-2\lambda_i\mu_j t}\right)^{\frac{1}{2}} \qquad \text{(Lemma 10)}
\end{aligned}$$

where $\lambda_1\mu_1 t < \frac{1}{2}$. Let $s = \lambda_1\mu_1 t$ and replace t with s.

$$\begin{aligned}
\mathbb{P}\left(Z \geq 1+\epsilon\right) &\leq e^{-\frac{(1+\epsilon)}{\lambda_1\mu_1}s} \prod_{i=1}^{r_1}\prod_{j=1}^{r_2} \left(\frac{1}{1-2\frac{\lambda_i}{\lambda_1}\frac{\mu_j}{\mu_1}s}\right)^{\frac{1}{2}} \\
&\leq e^{-\frac{(1+\epsilon)}{\lambda_1\mu_1}s} \prod_{i=1}^{r_1}\prod_{j=1}^{r_2} \left(\frac{1}{1-2s}\right)^{\frac{1}{2}\frac{\lambda_i}{\lambda_1}\frac{\mu_j}{\mu_1}} \qquad \text{(Lemma 11)} \\
&= e^{-\frac{(1+\epsilon)}{\lambda_1\mu_1}s} \left(\frac{1}{1-2s}\right)^{\frac{1}{2}\sum_{i=1}^{r_1}\frac{\lambda_i}{\lambda_1}\sum_{j=1}^{r_2}\frac{\mu_j}{\mu_1}} \\
&= e^{-\frac{(1+\epsilon)}{\lambda_1\mu_1}s} \left(\frac{1}{1-2s}\right)^{\frac{1}{2}\frac{1}{\lambda_1\mu_1}}
\end{aligned}$$

Next, we optimize in s by setting the logarithmic derivative in s of the above line to zero. Preforming this calculation, we get $s = \frac{\epsilon}{2(1+\epsilon)}$. Plugging this in to the above line along with the definitions of λ_1 and μ_1 yields

$$\begin{aligned}
\mathbb{P}\left(Z \geq 1+\epsilon\right) &\leq \left(\sqrt{(1+\epsilon)e^{-\epsilon}}\right)^{\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}} \\
&= \left(e^{-\epsilon/2+\ln(1+\epsilon)/2}\right)^{\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}}
\end{aligned}$$

A Taylor expansion gives us that $\ln(1+x) \leq x - \frac{x^2}{2} + \frac{x^3}{3}$ for all $|x| < 1$ because its an alternating series when $x > 0$ and each additional term is negative when $x < 0$. Plugging this in to the above, we get

$$\mathbb{P}\left(Z \geq 1+\epsilon\right) \leq \exp\left(-\left(\frac{\epsilon^2}{4} - \frac{\epsilon^3}{6}\right)\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}\right)$$

A similar argument gives

$$\begin{aligned}\mathbb{P}\left(\|A\Omega B\|_F^2 \leq (1-\epsilon)\|A\|_F^2\|B\|_F^2\right) &= \mathbb{P}\left(Z \leq 1-\epsilon\right) \\ &= \mathbb{P}\left(e^{-tZ} \geq e^{-(1-\epsilon)t}\right) \\ &\leq e^{(1-\epsilon)t}\prod_{i=1}^{r_1}\prod_{j=1}^{r_2}\left(\frac{1}{1+2\lambda_i\mu_j t}\right)^{\frac{1}{2}}\end{aligned}$$

where the last line comes from Markov's inequality in combination with Lemmas 10 and 9. As before, we apply the transformation of variables $s = \lambda_1\mu_1 t$ along with Lemma 11 to get

$$\mathbb{P}\left(Z \leq 1-\epsilon\right) \leq e^{\frac{(1-\epsilon)}{\lambda_1\mu_1}s}\left(\frac{1}{1+2s}\right)^{\frac{1}{2}\frac{1}{\lambda_1\mu_1}}$$

Then, we minimize the upper bound by setting the logarithmic derivative to zero, which gives $s = \frac{\epsilon}{2(1-\epsilon)}$. Plug this in to get

$$\mathbb{P}\left(Z \leq 1-\epsilon\right) \leq \left(e^{+\epsilon/2+\ln(1-\epsilon)/2}\right)^{\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}} \leq \exp\left(-\left(\frac{\epsilon^2}{4}+\frac{\epsilon^3}{6}\right)\frac{\|A\|_F^2}{\|A\|_2^2}\frac{\|B\|_F^2}{\|B\|_2^2}\right)$$

where the last line makes use of the Taylor expansion of $\ln(1+\epsilon)$ from above. □

5.2 Generalized Wilkinson Function

Lemma 12 (Partial fraction telescoping sum)**.** *Let $q > r > 0$ be positive integers. Then we have*

$$\frac{1}{r} = \frac{1}{q} + \sum_{j=r}^{q-1}\frac{1}{(j+1)j}$$

Proof. Let $s \in \mathbb{N}$ such that $r \leq s < q$. We use common denominators to subtract the two following fractions

$$\frac{1}{s} - \frac{1}{s+1} = \frac{1}{s(s+1)}$$

Then we take the telescoping sum to arrive at our result

$$\frac{1}{r} - \frac{1}{q} = \sum_{s=r}^{q-1}\frac{1}{s} - \frac{1}{s+1} = \sum_{s=r}^{q-1}\frac{1}{s(s+1)}$$

□

Lemma 13 (Special Matrix Inverse)**.** *Let* $B = (b_{ij})_{1\le i,j\le n} \in \mathbb{R}^{n\times n}$ *be an upper triangular matrix given by*

$$b_{ij} = \begin{cases} 0 & \text{if } i > j \\ 1 & \text{if } i = j \\ -\frac{1}{i} & \text{if } i < j \end{cases}$$

or in other words

$$B = \begin{pmatrix} 1 & -\frac{1}{n} & -\frac{1}{n} & \cdots & -\frac{1}{n} & -\frac{1}{n} \\ 0 & 1 & -\frac{1}{n-1} & \cdots & -\frac{1}{n-1} & -\frac{1}{n-1} \\ 0 & 0 & 1 & \cdots & -\frac{1}{n-2} & -\frac{1}{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -\frac{1}{2} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

Then the inverse $B^{-1} = (c_{ij})_{1\le i,j\le n} \in \mathbb{R}^{n\times n}$ *is given by*

$$c_{ij} = \begin{cases} 0 & \text{if } i > j \\ 1 & \text{if } i = j \\ \frac{1}{n-j+2} & \text{if } i < j \end{cases}$$

or

$$B^{-1} = \begin{pmatrix} 1 & \frac{1}{n} & \frac{1}{n-1} & \cdots & \frac{1}{3} & \frac{1}{2} \\ 0 & 1 & \frac{1}{n-1} & \cdots & \frac{1}{3} & \frac{1}{2} \\ 0 & 0 & 1 & \cdots & \frac{1}{3} & \frac{1}{2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & \frac{1}{2} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

Proof. Rewrite B into the product of elementary matrices or atomic triangular matrices

$$B = \underbrace{\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -\frac{1}{2} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}}_{M_2} \cdots \underbrace{\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & -\frac{1}{n-1} & \cdots & -\frac{1}{n-1} & -\frac{1}{n-1} \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}}_{M_{n-1}} \underbrace{\begin{pmatrix} 1 & -\frac{1}{n} & -\frac{1}{n} & \cdots & -\frac{1}{n} & -\frac{1}{n} \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}}_{M_n}$$

$$\overset{def}{=} M_2 \cdots M_{n-1} M_n$$

Taking the inverse, we see that

$$B^{-1} = \underbrace{\qquad M_n^{-1} \qquad}\underbrace{\qquad M_{n-1}^{-1} \qquad}\cdots\underbrace{\qquad M_2^{-1} \qquad}$$
$$= \begin{pmatrix} 1 & \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} & \frac{1}{n} \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \frac{1}{n-1} & \cdots & \frac{1}{n-1} & \frac{1}{n-1} \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \cdots \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & \frac{1}{2} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

We leave computing this product as an exercise to the reader. Lemma 12 will be useful. □

Lemma 14 (Generalized Wilkinson function bound)**.** *Let the generalized Wilkinson function be given as* $f(m,t) := \sqrt{(2+t)(3+t)^{\frac{1}{2}} \cdots (m+t)^{\frac{1}{m-1}}}$*. Then, we have*

$$f(m,t) \leq \sqrt{e(t+2)(t+1)} m^{\frac{1}{4}\ln(m+t)} m^{\frac{1}{4}\ln\left(\frac{m+t}{m}\right)} (t+1)^{\frac{1}{4}\ln(t+1)} \tag{5.3}$$

Proof. We have

$$\ln\left(f^2(m,t)\right) = \sum_{k=2}^{m} \frac{1}{k-1} \ln(t+k) = \sum_{k=1}^{m-1} \frac{1}{k} \ln(t+k+1)$$

Observe the identity $\frac{1}{k} - \frac{1}{k+t+1} = \frac{t+1}{k(k+t+1)}$ and note that the function $\frac{1}{k}\ln(t+k+1)$ is decreasing in k. We use the integral approximation along with integration by parts to get our desired result

$$\begin{aligned}
\ln\left(f^2(m,t)\right) &= \ln(t+2) + \sum_{k=2}^{m-1} \frac{1}{k} \ln(k+t+1) \\
&\leq \ln(t+2) + \int_1^{m-1} \frac{1}{x} \ln(x+t+1)\, dx \\
&= \ln(t+2) + \ln(x)\ln(t+x+1)\Big|_1^{m-1} - \int_1^{m-1} \frac{1}{x+t+1} \ln(x)\, dx \\
&= \ln(t+2) + \ln(x)\ln(t+x+1)\Big|_1^{m-1} - \int_1^{m-1} \frac{1}{x} \ln(x)\, dx + \int_1^{m-1} \frac{t+1}{x(x+t+1)} \ln(x) dx \\
&= \ln(t+2) + \ln(m+1)\ln(m+t) - \frac{1}{2}\ln^2(m+1) + \int_1^{m-1} \frac{t+1}{x(x+t+1)} \ln(x) dx \\
&\leq \ln(t+2) + \ln(m)\ln(m+t) - \frac{1}{2}\ln^2(m) + \int_1^{\infty} \frac{t+1}{x(x+t+1)} \ln(x) dx \\
&= \ln(t+2) + \frac{1}{2}\ln(m)\ln(m+t) + \frac{1}{2}\ln(m)\ln\left(\frac{m+t}{m}\right) + \int_1^{\infty} \frac{t+1}{x(x+t+1)} \ln(x) dx
\end{aligned}$$

where the second to last line follows from the fact that

$$\frac{d}{dx}\ln(x)\ln(m+t)-\frac{1}{2}\ln^2(x)=\frac{\ln(m+t)}{x}-\frac{\ln(x)}{x}\geq 0$$

for all $x \leq m+t$. The inequality (5.3) follows from lemma 15. □

Lemma 15 (Useful inequality for improper integral)**.** *We have the following inequality*

$$\int_1^\infty \frac{t+1}{x(x+c)}\ln(x)\,dx \leq \frac{1}{2}\ln^2(c)+\ln(c)+1 \tag{5.4}$$

Proof.

$$\begin{aligned}
\int_1^\infty \frac{c}{x(x+c)}\ln(x)\,dx &= \int_1^c \frac{c}{x(x+c)}\ln(x)\,dx+\int_c^\infty \frac{c}{x(x+c)}\ln(x)\,dx \\
&\leq \int_1^c \frac{1}{x}\ln(x)\,dx+\int_c^\infty \frac{c}{x^2}\ln(x)\,dx \\
&= \frac{1}{2}\ln^2(x)\Big|_1^c-\frac{c}{x}(\ln(x)+1)\Big|_c^\infty \\
&= \frac{1}{2}\ln^2(c)+\ln(c)+1
\end{aligned}$$

□

Part II

Spectrum Revealing Bounds

Chapter 6

Introduction

The efficient approximation a matrix by another matrix of lower rank is a fundamental problem of numerical linear algebra and matrix computations with applications in machine learning and computer science. Applications include principle components analysis in statistics [60], eigenfaces for facial recognition [91], speeding up support vector machines [98] and speeding up PDE and integral equation solvers [75, 85]. In addition to the large number of applications, there are many different algorithms for performing a low-rank matrix approximation. One's choice of method is extremely important given the application and the user's constraints on speed, accuracy, cache space and amount of computer memory (RAM). Among the methods that we will study in this thesis are (i) Gaussian randomized subspace iteration, (2) Column/row selection based methods like CX decompositions, CUR decompositions and the Nyström method.

Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be our input matrix and let $k \ll \min\{m,n\}$ be the target rank. We write the Singular Value Decomposition (SVD) of $\mathbf{A} = U\Sigma V^T = \sum_{j=1}^{\min\{m,n\}} \sigma_j \mathbf{u}_j \mathbf{v}_j^T$. where the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min\{m,n\}} \geq 0$ are in decreasing order such that $\Sigma = \mathbf{diag}\left(\sigma_1, \sigma_2, \cdots, \sigma_{\min\{m,n\}}\right)$ and the vectors $\mathbf{u}_j$ and $\mathbf{v}_j$ are the columns of U and V, respectively. We define the rank-k truncated SVD of a matrix $(\mathbf{A})_k = \sum_{j=1}^{k} \sigma_j \mathbf{u}_j \mathbf{v}_j^T$, which satisfy the optimal rank-k approximation condition under the two following norms:

Theorem 10 (Eckart-Young Theorem [34])**.** *Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be any matrix and let $1 \leq k \leq \min(m,n)$. Then, the rank-k truncated SVD $\mathbf{A}_k$ attains the minimum of the following two problems*

$$\min_{\mathbf{rank}(\mathbf{B})\leq k} \|\mathbf{A}-\mathbf{B}\|_F = \|\mathbf{A}-\mathbf{A}_k\|_F = \sqrt{\sum_{j=k+1}^{\min(m,n)} \sigma_j^2(\mathbf{A})}$$

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

Computationally, in order to attain a residual error similar to the optimal Eckart-Young error for the rank-k approximation, we allow our fast-approximation algorithms a little wiggle

room in the form of *oversampling*. This means that our algorithms will produce a slightly larger rank-ℓ matrix $\widetilde{\mathbf{A}} \in \mathbb{R}^{m \times n}$, but will still be judged relative the the quality of the optimal rank-k Eckart-Young Truncated SVD. Specifically, we refer to *oversampling* as the difference $\ell - k \geq 0$. While condoning a small amount of oversampling, the accuracy of our approximation when compared against the best rank-k improves dramatically as we can prove.

We present powerful **spectrum revealing bounds** for low-rank matrix approximation algorithms. These bounds show that increased amounts of spectral decay help these algorithms to "reveal" the true spectral structure of the rank-k truncated SVD. Let $\widetilde{\mathbf{A}}$ be a candidate low-rank approximation to our input matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$. We can evaluate the residual error of our approximation in two ways: (i) the *weak-form* residual error and (ii) the *strong-form* residual error

$$(i) \left\| \mathbf{A} - \widetilde{\mathbf{A}} \right\|_\xi^2 \leq \left(1 + O(\tau^2)\right) \|\mathbf{A} - \mathbf{A}_k\|_\xi^2 \tag{6.1}$$

$$(ii) \left\| \mathbf{A} - \left(\widetilde{\mathbf{A}}\right)_k \right\|_\xi^2 \leq \left(1 + O(\tau^2)\right) \|\mathbf{A} - \mathbf{A}_k\|_\xi^2 \tag{6.2}$$

respectively with $\xi = 2, F$. In addition, we also evaluate the accuracy of the weak and strong form problems with singular value lower bounds as in [47] to guarantee that capturing at least a fraction of the directions of largest variance as in

$$\sigma_j\left(\widetilde{\mathbf{A}}\right) \geq \frac{\sigma_j(A)}{\sqrt{1 + O(\tau^2)}} \quad \text{for } j \leq k \tag{6.3}$$

where k is the user's target rank and τ is a quantity that depends on the rate of singular value decay in $\mathbf{A}$ between the target rank k and oversampling parameter $\ell \geq k$. Much work has been done on error bounds that depend solely on the dimensionality of the problem– avoiding the incorporation of spectral decay entirely [18, 32, 50]. Most of these bounds lead to the conclusion that the amount of required oversampling scales with the target rank k and the matrix dimensions m and n. However, the spectrum-revealing bounds show that only a constant amount of oversampling $\ell - k = O(1)$ is needed for matrices with a small enough spectral decay parameter $\tau \ll 1$. We will produce results of this type for various low-rank approximation algorithms and then show these results in data science type applications.

The main contributions of this part of the thesis are (i) improved rates of spectral decay τ for the residual-type error bounds, (ii) a simplified and tighter analysis of subspace iteration removing annoying logarithms from the required amount of oversampling in [50, 47] and (iii) applying these spectrum revealing bounds to a broader class of algorithms involving column/row selection based algorithms.

Chapter 7

Randomized Subspace Iteration

7.1 Basic Setup

First, we fix the desired target rank $k \ll \min\{m, n\}$ and the amount of oversampling $\ell - k \geq 0$. Next, we define a parameter p that varies between $0 \leq p \leq \ell - k$. This parameter p is not known to or involved in the execution of these algorithms–it is merely for the benefit of analysis. We let $\Omega \in \mathbb{R}^{n \times \ell}$ be the starting sampling matrix and we let $\widehat{\Omega} = V^T \Omega \in \mathbb{R}^{n \times \ell}$ denote the starting sampling matrix rotated by the right singular values V^T of A. In order to state our results, we need the following partition of our rotated starting matrix and our singular values

$$\widehat{\Omega} \stackrel{def}{=} V^T \Omega \stackrel{def}{=} \begin{array}{c} \\ \ell - p \\ n - \ell + p \end{array} \begin{array}{c} \ell \\ \begin{pmatrix} \widehat{\Omega}_1 \\ \widehat{\Omega}_2 \end{pmatrix} \end{array} \tag{7.1}$$

and

$$\Sigma \stackrel{def}{=} \begin{array}{c} \\ \ell - p \\ n - \ell + p \end{array} \begin{array}{c} \ell - p \quad n - \ell + p \\ \begin{pmatrix} \Sigma_T & \\ & \Sigma_B \end{pmatrix} \end{array} \tag{7.2}$$

where

$$\Sigma_T = \begin{array}{c} \\ k \\ \ell - p - k \end{array} \begin{array}{c} k \quad \ell - p - k \\ \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix} \end{array} \quad \text{and } \Sigma_B = \begin{array}{c} \\ k \\ n - \ell + p - k \end{array} \begin{array}{c} k \quad n - \ell + p - k \\ \begin{pmatrix} \Sigma_3 & \\ & \Sigma_4 \end{pmatrix} \end{array} \tag{7.3}$$

here, we refer to Σ_T as the *top* singular values and Σ_B as the *bottom* singular values. With regards to the partition of Σ_B, we will make use of the matrix $\Sigma_1^{\downarrow} = \text{diag}(\sigma_k, \sigma_{k-1}, \cdots, \sigma_1)$ which is $\Sigma_1 = \text{diag}(\sigma_1, \sigma_2, \cdots, \sigma_k)$ except with the singular values in reverse or ascending order.

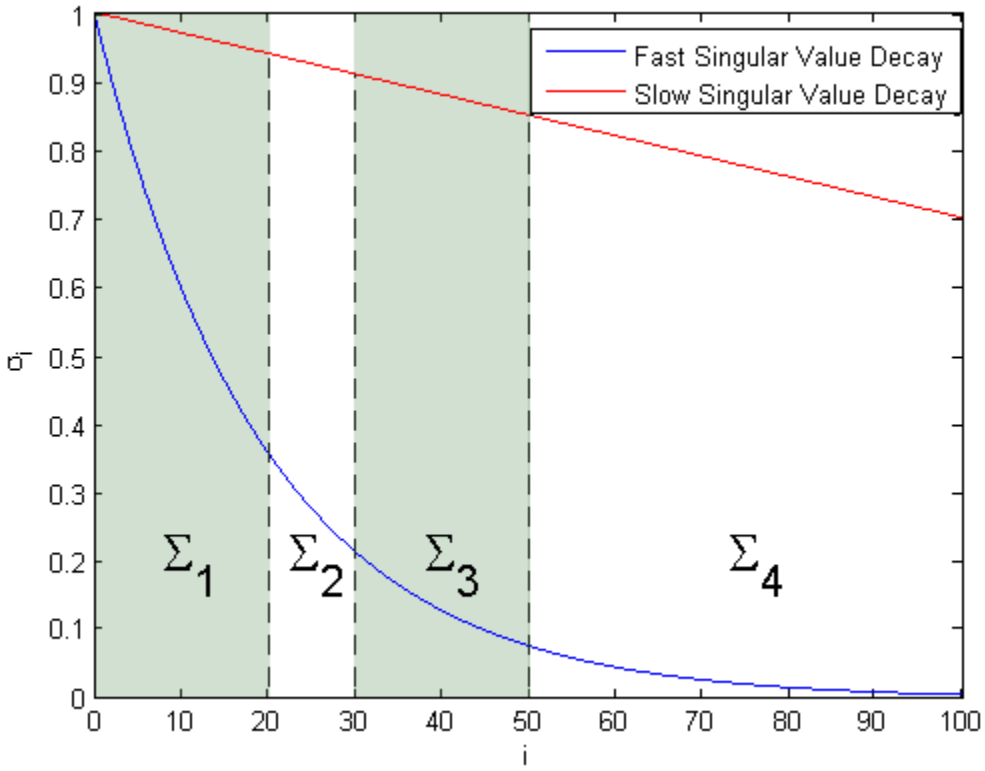


Figure 7.1: The visualization of singular value decay along with our partitions. The entire Σ_1 and Σ_3 components in green of the spectrum contribute to the accelerated decay rate $\tau_F^{(4q)}$ as opposed to τ_k^{4q} where only two singular values σ_k and $\sigma_{\ell-p+1}$ contribute

Next, we introduce the iteration parameter $q \geq 0$. This parameter is only valid for subspace iteration, where it controls the number of power iterations used to improve convergence of the algorithm. All other algorithms must have $q = 0$ because they do not allow this functionality. Then, define the following spectral decay rates

$$\tau_j = \frac{\sigma_{\ell-p+1}}{\sigma_j} \qquad \text{for all } 1 \leq j \leq k \qquad \text{and} \qquad \tau_F^{(4q)} = \frac{1}{k} \sum_{j=1}^{k} \frac{\sigma_{\ell-p+j}^2}{\sigma_{\ell-p+1}^2} \left(\frac{\sigma_{\ell-p+j}}{\sigma_{k+1-j}} \right)^{4q} \tag{7.4}$$

where we have that

$$\tau_k^{4q} = \left(\frac{\sigma_{\ell-p+1}}{\sigma_k} \right)^{4q} = \left\| \frac{1}{\sigma_{\ell-p+1}} \left(\Sigma_1^{\downarrow} \right)^{-2q} \Sigma_3^{2q+1} \right\|_2^2 \geq \frac{1}{k} \left\| \frac{1}{\sigma_{\ell-p+1}} \left(\Sigma_1^{\downarrow} \right)^{-2q} \Sigma_3^{2q+1} \right\|_F^2 = \tau_F^{(4q)}$$

The reason that $\tau_F^{(4q)}$ is less that τ_k^{4q} has to do with a pigeonhole principle for singular value triplets $(\sigma_i, \mathbf{u}_i, \mathbf{v}_i)$ called Lidskii's Theorem as stated in Theorem 17. Essentially, it states

that each singular value triplet is only allowed to occupy one direction or dimension. In the circumstance of the residual error, the spectral decay is a bound of the decay of error over a rank-k space. Thus, a pigenhole principle would tell us that the slowest decay rate $\frac{\sigma_{\ell-p+1}}{\sigma_k}$ can only occupy one of the k dimensions, while the rest of the directions will enjoy faster rates $\frac{\sigma_{\ell-p+j}}{\sigma_{\ell_{k+1-j}}}$ for each direction $1 \leq j \leq k$.

To exemplify this, consider the synthetic experiment of geometric singular value decay. Let $\sigma_j = \alpha\gamma^j$ where $\alpha > 0$ is a fixed constant and $0 \leq \gamma \leq 1$ is the geometric decay rate.

$$\begin{aligned}
\tau_F^{(4q)} &= \frac{1}{k}\sum_{j=1}^{k} \frac{\sigma_{\ell-p+j}^2}{\sigma_{\ell-p+1}^2}\left(\frac{\sigma_{\ell-p+j}}{\sigma_{k+1-j}}\right)^{4q} = \frac{1}{k}\gamma^{4q(\ell-p-k+1)}\sum_{j=0}^{k-1}\gamma^{(8q+2)j} \\
&= \tau_k^{4q}\frac{1}{k}\left(1+\gamma^{(8q+2)}\frac{1-\gamma^{(8q+2)(k-1)}}{1-\gamma^{(8q+2)}}\right) \\
&\leq \tau_k^{4q}\left(\frac{1}{k}+\frac{k-1}{k}\gamma^{(8q+2)}\right) \overset{k \text{ large}}{\approx} \tau_k^{4q}\gamma^{(8q+2)}
\end{aligned}$$

Thus, using the accelerated spectral decay rate $\tau_F^{(4q)}$ is less than the spectral decay rate τ_k^{4q} from previous works by a significant amount. If $\gamma = 0.75$, $q = 1$ and $k = 11$, $\tau_F^{(4)}$ would be less than τ_k^4 by an order of magnitude with $\tau_F^(4) \approx 0.0964\tau_k^4$ independent of the users choice of ℓ and p.

When it comes to the rotated starting matrix $\widehat{\Omega} \in \mathbb{R}^{n\times\ell}$, the bounds for matrix approximation algorithms also depend on the spectral norm of the so-called *sketching interaction matrix*, i.e. $\widehat{\Omega}_2\widehat{\Omega}_1^\dagger$. It can be shown that this quantity $\left\|\widehat{\Omega}_2\widehat{\Omega}_1^\dagger\right\|_2 = \tan(\Omega, V_1)$ is the tangent of the largest principle angle between the column space of Ω and $V_1 = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_{\ell-p} \end{bmatrix}$ as done by Gittens et. al [42]. Typically, we try to bound this quantity by the surrogate interaction quantity $\left\|\widehat{\Omega}_2\right\|_2\left\|\widehat{\Omega}_1^\dagger\right\|_2$.

7.2 Algorithm: Randomized Subspace Iteration

Orthogonal iteration or Subspace Iteration is a popular algorithm for low-rank matrix approximation [44, 50, 47]. In Randomized Subspace Iteration, we sample the entries of the starting matrix Ω as standard normal $\mathcal{N}(0,1)$ random variables and we perform power iterations on it. The algorithm is presented as follows

When computing Y and Q in steps 2 and 3 of the above Algorithm 7 in floating point arithmetic, we want to use the following procedure in Algorithm 8 to orthonormalize Y at each application of A or A^T as mentioned in [50]. If we do not use Algorithm 8, then all of the singular values σ_j below a certain level (about $\epsilon_{mach}^{1/(2q+1)}\sigma_1$) can be totally corrupted by floating point rounding errors.

The rotation invariance of random matrices with iid standard normal $\mathcal{N}(0,1)$ entries [50] gives us that $\widehat{\Omega} = V^T\Omega$, as defined in (7.1), is also distributed as random matrix with iid

Algorithm 7 : Randomized Subspace Iteration

Inputs: $m \times n$ matrix A with $n \leq m$, integers $k > 0$ and $\ell \geq k$.
Outputs: a rank-k approximation.

1: Draw a random $n \times \ell$ test matrix Ω with iid $\mathcal{N}(0,1)$ entries.
2: Compute $Y = \left(AA^T\right)^q A\,\Omega$.
3: Compute an orthogonal column basis Q for Y.
4: Return QQ^TA.

Algorithm 8 : Orthorgonalization with QR

Inputs: $m \times n$ matrix A, $n \times \ell$ start matrix Ω, and integer $q \geq 0$.
Outputs: $Q \in \mathbb{R}^{m \times \ell}$ with orthonormal columns.

1: **compute** $Y = A\Omega$, and QR factorize $QR = Y$.
2: **for** $i = 1, \cdots, q$ **do**
3: $\quad Y = A^T Q$; QR factorize $QR = Y$;
4: $\quad Y = A\,Q$; QR factorize $QR = Y$.
5: **end for**
6: **return** Q and Q^TA

standard normal entries. One of the main contributions of this thesis for Randomized Subspace Iteration is the simplified tail bound for the surrogate sketching iteration in Theorem 29

$$\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq \frac{\mathcal{B}\sqrt{\ell}\,\mathcal{E}}{p+1}\,t\right) \leq t^{-(p+1)}$$

where $\mathcal{E} = \sqrt{n-\ell+p}+\sqrt{\ell}$ is a function of $p < \ell < n$ and $\mathcal{B} \leq 3.0237$ is a universal constant. This random variable has a large tail especially when p is small. As discussed in Section 7.6, prior works [47, 50] use a long analysis with separate tail bounds for $\left\|\widehat{\Omega}_2\right\|_2$ and $\left\|\widehat{\Omega}_1^\dagger\right\|_2$ instead of taking advantage of the natural independence between the two variables.

It is important to keep in mind that the the strong-form residual error will always be larger than the weak-form residual error, i.e.

$$\left\|A - QQ^TA\right\|_\xi^2 \leq \left\|A - Q\left(Q^TA\right)_k\right\|_\xi^2$$

because of Lemma 16 and Theorem 19. To make use of our new spectral decay rate, we introduce the following new structural result from Corollary 5 with $\alpha_F^2 = \frac{\sigma_{\ell-p+1}^2}{\frac{1}{k}\|\Sigma_1\|_F^2}$

$$\left\|A - Q\left(Q^TA\right)_k\right\|_\xi^2 \leq \|A - A_k\|_\xi^2 + k\sigma_{\ell-p+1}^2 \frac{\tau_F^{(4q)} \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + \alpha_F^2 \tau_F^{(4q)} \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}$$

which is tighter than competing bounds [47] due to the small spectral decay rate $\tau_F^{(4q)} \leq \tau_k^{4q}$. We will show in numerical experiments, both from synthetic and real data, that $\tau_F^{(4q)}$ is often much smaller than τ_k^{4q}.

Theorem 11 (Large Deviation: Normed Residual Errors and Singular Value Lower Bounds)**.** *Let the SVD of A and the sampling matrix Ω be given as above. Also, let $0 \leq p \leq \ell$. Let the spectral decay rates τ_k and $\tau_F^{(4q)}$ be as defined in Equation (7.4). In real arithmetic, the output of Randomized Subspace Iteration in Algorithm 7 must also satisfy the following bounds*

$$\left\|A - QQ^TA\right\|_\xi^2 \leq \left\|A - Q\left(Q^TA\right)_k\right\|_\xi^2 \leq \|A - A_k\|_\xi^2 + k\frac{\tau_F^{(4q)}\mathcal{K}_\Delta^2}{1+\alpha_F^2\tau_F^{(4q)}\mathcal{K}_\Delta^2}\sigma_{\ell-p+1}^2 \tag{7.5}$$

where either $\xi = 2, F$ and

$$\left\|A - QQ^TA\right\|_2^2 \leq \sigma_{k+1}^2 + \frac{\tau_k^{4q}\mathcal{K}_\Delta^2}{1+\alpha_2^2\tau_k^{4q}\mathcal{K}_\Delta^2}\sigma_{\ell-p+1}^2 \tag{7.6}$$

as well as,

$$\sigma_j\left(QQ^TA\right) \geq \frac{\sigma_j}{\sqrt{1+\tau_j^{4q+2}\mathcal{K}_\Delta^2}} \quad \text{for } 1 \leq j \leq k \tag{7.7}$$

with probability $1-\Delta$ where $\mathcal{K}_\Delta = \frac{\mathcal{B}\sqrt{\ell}\mathcal{E}}{p}\left(\frac{1}{\Delta}\right)^{p+1}$ with $\mathcal{E} = \sqrt{n-\ell+p}+\sqrt{\ell}$ and the universal constant $\mathcal{B} \leq 3.0237$, and where $\alpha_F^2 = \frac{\sigma_{\ell-p+1}^2}{\frac{1}{k}\|\Sigma_1\|_F^2}$ and $\alpha_2^2 = \frac{\sigma_{\ell-p+1}^2}{\sigma_1^2}$.

Proof. Apply Theorem 29 to Corollary 5, Lemma 20 and Theorem 21, respectively. □

Due to the large tail of $\|\widehat{\Omega}_2\|_2\|\widehat{\Omega}_1^\dagger\|_2$ from Theorem 29, the expected residual errors and singular lower bounds become loose and complicated to calculate when $p = 0, 1$. The work of Gu [47] shows that tractable bounds do exist for $p < 2$ and the smaller spectral decay rate $\tau_F^{(4q)}$ can also be applied in this case. However, we choose not to go into this here for simplicity. Instead, we present the bounds in expectation for $p \geq 2$ as follows

Theorem 12 (Expectation: Normed Residual Errors and Singular value lower bounds)**.** *Let the SVD of A and the sampling matrix Ω be given as above. Also, let $2 \leq p \leq \ell$. Let the spectral decay rates τ_k and $\tau_F^{(4q)}$ be as defined in Equation (7.4). In real arithmetic, the output of Randomized Subspace Iteration in Algorithm 7 must satisfy*

$$\mathbb{E}\left\|A - QQ^TA\right\|_\xi^2 \leq \mathbb{E}\left\|A - Q\left(Q^TA\right)_k\right\|_\xi^2 \leq \|A - A_k\|_\xi^2 + k\frac{\tau_F^{(4q)}\mathcal{K}^2}{1+\alpha_F^2\tau_F^{(4q)}\mathcal{K}^2}\sigma_{\ell-p+1}^2 \tag{7.8}$$

where either $\xi = 2, F$ and

$$\mathbb{E}\left\|A - QQ^TA\right\|_2^2 \leq \sigma_{k+1}^2 + \frac{\tau_k^{4q}\mathcal{K}^2}{1+\alpha_2^2\tau_k^{4q}\mathcal{K}^2}\sigma_{\ell-p+1}^2 \tag{7.9}$$

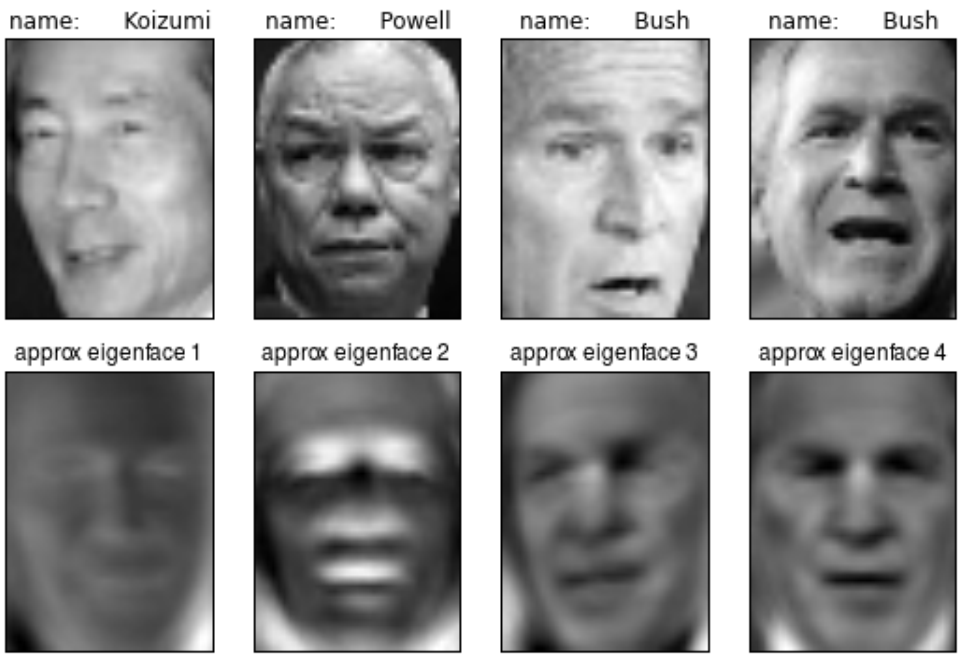


Figure 7.2: This figure shows some example faces from the test set and some example approximate eigenfaces produced from using Algorithm 8 with $\ell = 150$ and $q = 3$ on the training set

as well as,

$$\mathbb{E}\sigma_j\left(QQ^TA\right) \geq \frac{\sigma_j}{\sqrt{1+\tau_j^{4q+2}\mathcal{K}^2}} \quad \textit{for } 1 \leq j \leq k \tag{7.10}$$

in expectation, where $\mathcal{K} = \frac{\mathcal{B}\sqrt{\ell}\mathcal{E}}{\sqrt{(p+1)(p-1)}}$ *with* $\mathcal{E} = \sqrt{n-\ell+p} + \sqrt{\ell}$ *and the universal constant* $\mathcal{B} \leq 3.0237$*, and where* $\alpha_F^2 = \frac{\sigma_{\ell-p+1}^2}{\frac{1}{k}\|\Sigma_1\|_F^2}$ *and* $\alpha_2^2 = \frac{\sigma_{\ell-p+1}^2}{\sigma_1^2}$.

Proof. For bounds (7.8) and (7.9), apply Lemma 23 to Corollary 5 and Lemma 20, respectively. For the lower bound (7.10), use Lemma 24 on Theorem 21. □

7.3 Experiment

In this section, we make use of the dataset *Labeled Faces in the Wild* (LFW) [55] by selecting all the images of people with at least 60 images in the LFW dataset. This gives $1,348$ different images of 8 different people. Each image has $1,850$ grey-style degrees of freedom. We made use of Python's scikit-learn package [83] to efficiently and conveniently load and work with the LFW data. They also provided interesting examples of using LFW

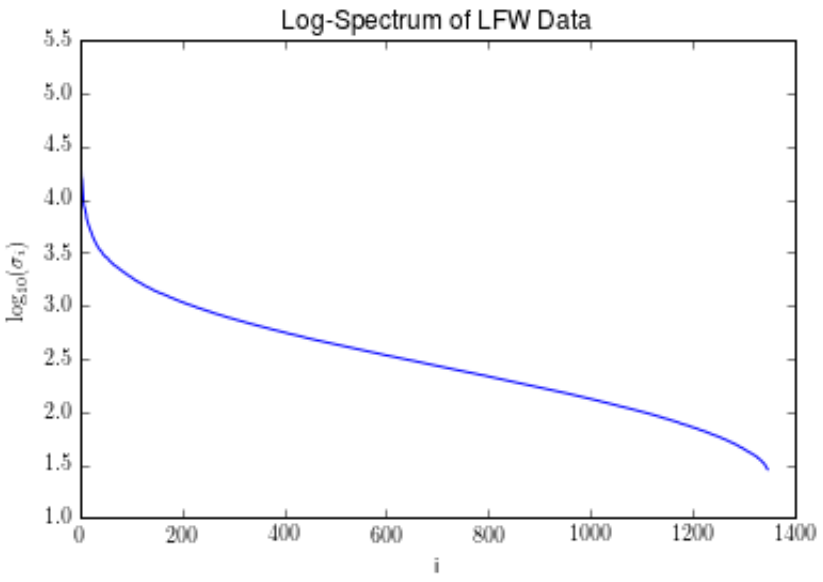


Figure 7.3: The log-spectrum of the data matrix X of images

data that was useful. The typical goal for a data scientist using this dataset is to produce a classifier for the identities of the people in each picture. The novelty of this dataset comes from the fact that it was automatically produced from webscraping images and running the Viola-Jones face detector. Practically speaking, this means that we have no guarantee that our faces will be centered and such for a classification algorithm, as one can see by looking at the pictures in Figure 7.2. Therefore, this will require more sophisticated methods than those proposed in [91]. As per the example implemented in the scikit-learn library, an efficient way to produce accurate classifications on the identity of images is to use randomized subspace iteration –RandomizedPCA and/or TruncatedSVD functions in scikit-learn – to reduce dimensions before using a Radial Basis Function (RBF) Kernel SVM to perform the classification. We are interested in the performance of the randomized subspace iteration step in this context. Let $X \in \mathbb{R}^{1,850\times 1,348}$ be the data matrix of images where each column represents the pixels of a particular image. The true eigenfaces are given by the columns of the matrix U, where $X = U\Sigma V^T$ is the SVD of X. Figure 7.3 gives the log spectrum $\mathbf{diag}\left(\log_{10}(\Sigma)\right)$ of this matrix. Let $X \approx QQ^TX$ be the result of Algorithm 8. Taking the SVD of the short fat matrix $Q^TX = \widetilde{U}\widetilde{\Sigma}\widetilde{V}^T$, we set the approximate eigenfaces to be the columns of $Q\widetilde{U} \in \mathbb{R}^{1,850\times \ell}$. Figure 7.2 gives some approximate eigenfaces from the LFW dataset. In this example, we are interested in demonstrating that our new matrix approximation bound involving the accelerated spectral decay rate $\tau_F^{(4q)}$ is a significant improvement in practice. Table 7.1 shows that this new spectal decay rate $\tau_F^{(4q)}$ can be more than an order of magnitude better than the rate τ_k^{4q} used in prior works [47].

	τ_k^0	$\tau_F^{(0)}$	τ_k^4	$\tau_F^{(4)}$	τ_k^8	$\tau_F^{(8)}$	τ_k^{12}	$\tau_F^{(12)}$
$k = 10, l = 15$	1.0	0.87412	0.35983	0.10788	0.12948	0.02542	0.04659	0.00719
$k = 10, l = 20$	1.0	0.88187	0.19464	0.05950	0.03789	0.00761	0.00737	0.00116
$k = 20, l = 25$	1.0	0.82198	0.55295	0.12188	0.30576	0.04148	0.16907	0.01768
$k = 20, l = 30$	1.0	0.86206	0.32015	0.07995	0.10249	0.01590	0.03281	0.00386
$k = 40, l = 45$	1.0	0.82446	0.74128	0.15004	0.54950	0.06601	0.40733	0.03611
$k = 40, l = 50$	1.0	0.83360	0.58878	0.12187	0.34666	0.04293	0.20411	0.01883
$k = 40, l = 55$	1.0	0.84104	0.47417	0.09890	0.22484	0.02751	0.10661	0.00952
$k = 60, l = 65$	1.0	0.80158	0.81309	0.15845	0.66111	0.07474	0.53754	0.04376
$k = 60, l = 70$	1.0	0.80498	0.68826	0.13490	0.47370	0.05386	0.32603	0.02672
$k = 60, l = 80$	1.0	0.81785	0.48473	0.09900	0.23496	0.02826	0.11389	0.00997

Table 7.1: Comparison of spectral decay rates τ_k^{4q} and $\tau_F^{(4q)}$ for $0 \leq q \leq 3$ on the Labeled Faces in the Wild (LFW) dataset.

7.4 The Setup

Let $A = U\Sigma V^T \in \mathbb{R}^{m\times n}$ be the SVD where $m \geq n$. Suppose you want a rank-k approximation to A for $k \leq n$. Instead of calculating the Eckart-Young solution, you want a quicker/faster approximation. So choose an oversampling parameter $\ell > k$. Then let $\Omega \in \mathbb{R}^{n\times\ell}$ be the starting sampling matrix. We repeat the following partitions for ease of presentation

$$\widehat{\Omega} \stackrel{def}{=} V^T\Omega \stackrel{def}{=} \begin{array}{c} \\ \ell-p \\ n-\ell+p \end{array} \begin{array}{c} \ell \\ \begin{pmatrix} \widehat{\Omega}_1 \\ \widehat{\Omega}_2 \end{pmatrix} \end{array} \tag{7.11}$$

and

$$\Sigma \stackrel{def}{=} \begin{array}{c} \ell-p \\ n-\ell+p \end{array} \begin{array}{c} \ell-p \quad n-\ell+p \\ \begin{pmatrix} \Sigma_T & \\ & \Sigma_B \end{pmatrix} \end{array} \tag{7.12}$$

where

$$\Sigma_T = \begin{array}{c} k \\ \ell-p-k \end{array} \begin{array}{c} k \quad \ell-p-k \\ \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix} \end{array} \quad \text{and } \Sigma_B = \begin{array}{c} k \\ n-\ell+p-k \end{array} \begin{array}{c} k \quad n-\ell+p-k \\ \begin{pmatrix} \Sigma_3 & \\ & \Sigma_4 \end{pmatrix} \end{array} \tag{7.13}$$

here, we refer to Σ_T as the *top* singular values and Σ_B as the *bottom* singular values. Let $0 \leq q \in \mathbb{Z}$. We are interested in the column space of

$$\left(AA^T\right)^q A\Omega = U \begin{pmatrix} \Sigma_T^{2q+1}\widehat{\Omega}_1 \\ \Sigma_B^{2q+1}\widehat{\Omega}_2 \end{pmatrix} = U \begin{pmatrix} \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix}^{2q+1} \widehat{\Omega}_1 \\ \begin{pmatrix} \Sigma_3 & \\ & \Sigma_4 \end{pmatrix}^{2q+1} \widehat{\Omega}_2 \end{pmatrix} \tag{7.14}$$

As done in [47], we define a matrix $X \in \mathbb{R}^{\ell \times \ell}$

$$X = \left(\begin{array}{c:c:c} \widehat{\Omega}_1^\dagger \Sigma_T^{-(2q+1)} & \widehat{X} \end{array}\right) = \left(\begin{array}{c:c:c} \widehat{\Omega}_1^\dagger \begin{pmatrix} \Sigma_1^{-(2q+1)} \\ 0 \end{pmatrix} & \widehat{\Omega}_1^\dagger \begin{pmatrix} 0 \\ \Sigma_2^{-(2q+1)} \end{pmatrix} & \widehat{X} \end{array}\right)$$

where $\widehat{X} \in \mathbb{R}^{\ell \times \ell - p}$ satisfies $\widehat{\Omega}_1 \widehat{X} = 0$. This allows us to represent the column space of $\left(AA^T\right)^q A\Omega$ as the span of the following columns

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

where

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

We can also apply the QR factorization to $(AA^T)^q A\Omega X = \widehat{Q}\widehat{R}$ in order to produce a column orthogonal matrix $\widehat{Q}$ that spans the column space of $(AA^T)^q A\Omega$

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

which allows us to represent the span of the first k columns of $(AA^T)^q A\Omega X$ by the column orthogonal matrix $\widehat{Q}_1$, i.e.

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

There is also a lesser known matrix decomposition that can also fulfill this purpose called the Polar Decomposition, which is given as

Theorem 13 (Polar Decomposition [54] Theorem 7.3.1 pg.449)**.** *Let $A \in \mathbb{R}^{m\times n}$ with $m \geq n$. Then, there exists a column orthonormal matrix $U \in \mathbb{R}^{m\times n}$ and a positive semidefinite matrix $P \in \mathbb{R}^{n\times n}$ such that $A = UP$. Also, the matrix P is unique and is given by*

$$P = (A^*A)^{1/2} = V\Sigma V^T$$

where the SVD of $A = U\Sigma V^T$.

Now, we use the polar decomposition to produce another column-orthogonal representation $\widetilde{Q}_1$ of the first k columns of $(AA^T)^q A\Omega X$

$$\widetilde{Q}_1 \left(I + H_1^T H_1\right)^{\frac{1}{2}} = U \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix}, \quad \text{where} \quad \widetilde{Q}_1 = U \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix} \left(I + H_1^T H_1\right)^{-\frac{1}{2}}$$

7.5 Preliminaries

In this section, we review technical results that will be crucial in our analysis of subspace iteration. For the most part, these preliminaries are rooted in the studies of either random Gaussian matrices or deterministic matrix analysis. Our plan of attack will be to (1) apply results from matrix analysis to get tractable bounds in terms of our random start matrix. Then, (2), we will apply results about Gaussian random matrices to arrive at tractable bounds.

7.5.1 Preliminaries from Matrix analysis

7.5.1.1 Inequalities

The singular value version of the Cauchy interlacing theorem shows us that the j^{th} singular value of a projected or "compressed" matrix QQ^TA must lie between two different singular values of the original matrix A.

Theorem 14 (Singular Value Interlacing)**.** *Let $A \in \mathbb{R}^{m\times n}$ be a symmetric matrix, and let $Q \in \mathbb{R}^{m\times(m-k)}$ be column orthogonal. Then*

$$\sigma_j(A) \geq \sigma_j\left(Q^TA\right) \geq \sigma_{j+k}(A)$$

Proof. Immediate consequence of applying the Cauchy Interlacing theorem (Corollary III.1.5 of [14]) to the symmetric matrix A^TA. This is also given as Theorem 3.1 of [47]. □

Next, we cover two different singular value bounds which allow us to bound the singular values of sums and products of matrices that appear in our analysis.

Theorem 15 (Weyl's Inequality and sub-multiplicativity for Singular Values (Problem III.6.5 of [14]))**.** *For any two matrices $A, B \in \mathbb{R}^{m\times n}$ and any two indices i, j such that $i + j \leq \min(m, n) + 1$, we have*

$$\begin{aligned} \sigma_{i+j-1}(A+B) &\leq \sigma_i(A) + \sigma_j(B) \\ \sigma_{i+j-1}(AB) &\leq \sigma_i(A)\,\sigma_j(B) \end{aligned}$$

The following theorem allows to bound the difference in singular values between two different matrices $A, B \in \mathbb{R}^{n\times n}$ by the Frobenious norm of their difference, which is usually easier to bound.

Theorem 16 (Hoffman-Weilandt [52])**.** *For any two matrices $A, B \in \mathbb{R}^{m\times n}$, we have*

$$\sum_{i=1}^{\min(m,n)} (\sigma_i(A) - \sigma_i(B))^2 \leq \|A - B\|_F^2$$

The following version of Weyl's Inequality for Hermitian matrices is frequently used to control *individual* eigenvalues of $A + B$ in terms of eigenvalues of A and B

$$\lambda_j(A) + \lambda_n(B) \leq \lambda_j(A + B) \leq \lambda_j(A) + \lambda_1(B)$$

for A and B Hermitian. If we wanted to bound the sum of different eigenvalues of $A + B$, we could apply this result multiple times to get

$$\sum_{j=1}^{k} \lambda_j(A) + k\lambda_n(B) \leq \sum_{j=1}^{k} \lambda_j(A + B) \leq \sum_{j=1}^{k} \lambda_j(A) + k\lambda_1(B)$$

However, a deep result from matrix analysis states that one does not need to reuse the same $\lambda_1(B)$ and $\lambda_n(B)$ for each term in the sum.

Theorem 17 (Lidskii's Theorem (Theorem III.4.1 of [14] pg.69))**.** *Let $A, B \in \mathbb{C}^{n\times n}$ be Hermitian matrices and let $1 \leq k \leq n$. Then for any choice of indices $1 \leq i_1 < i_2 < \cdots < i_k \leq n$, we have*

$$\sum_{j=1}^{k} \lambda_{i_j}(A) + \lambda_{n-j+1}(B) \leq \sum_{j=1}^{k} \lambda_{i_j}(A + B) \leq \sum_{j=1}^{k} \lambda_{i_j}(A) + \lambda_j(B)$$

If the eigenvalues B of rapidly decaying (i.e. $\lambda_1(B) \gg \lambda_2(B) \gg \cdots \gg \lambda_k(B)$ and so on), then Lidskii's theorem is a huge improvement over blindly reapplying Weyl's Theorem k times. We will exploit this observation to great effect in the development of our spectrum revealing bounds. In the same way that Lidskii's theorem improves the analysis of sums of multiple eigenvalues of $A+B$, the following trace inequality of John von Neumann improves the analysis of sums of multiple eigenvalues of AB over the repeated application of the submultiplicativity of singular values in Theorem 15.

Theorem 18 (John von Neumann's trace inequality [78]). *Let $A, B \in \mathbb{C}^{n\times n}$ be square matrices. Then, we have that*

$$|\mathbf{tr}\,(AB)| \leq \sup_{U,V} |\mathbf{tr}\,(AUBV)| = \sum_{j=1}^{n} \sigma_j\,(A)\,\sigma_j\,(B)$$

where $U, V \in \mathbb{C}^{n\times n}$ vary over the set of Unitary matrices.

A orthogonal projection matrix $P \in \mathbb{R}^{n\times n}$ is a symmetric matrix that satisfies the equation $P^2 = P$. This implies that the eigenvalues of P must be $\lambda = 0, 1$, or in other words, satisfy the equation $\lambda^2 = \lambda$. Therefore, by the spectral theorem for Hermitian matrices [10], if P has $0 \leq k \leq n$ unit eigenvalues then the orthogonal projection matrix can be represented by

$$P = Q_P Q_P^T$$

where $Q_P \in \mathbb{R}^{n\times k}$ is a column-orthonormal matrix. The following Lemma will be useful in analyzing orthogonal projection matrices that show up in the analysis.

Lemma 16 (Orthogonal Projector Lemma (Proposition 8.5 of [50])). *Let $P_1, P_2 \in \mathbb{R}^{n\times n}$ be orthogonal projection matrices. Suppose range$(P_1) \subset$ range(P_2). Then, for each matrix A, it holds that*

$$\begin{aligned} \|P_1 A\|_\xi &\leq \|P_2 A\|_\xi \\ \|(I - P_2)A\|_\xi &\leq \|(I - P_1)A\|_\xi \end{aligned}$$

for both the spectral norm $\xi = 2$ and the Frobenius norm $\xi = F$

Next, we review a clever generalization of Theorem 10 due to Ming Gu [47]. Suppose we want to restrict the range of our rank-k approximation to live within a particular subspace spanned by the orthonormal columns of $Q \in \mathbb{R}^{m\times \ell}$. In other words, we want the best approximation $A \approx QB$, where $B \in \mathbb{R}^{\ell\times n}$ is a rank-k matrix. Then, the following theorem tells us that the best choice is given by $B = \left(Q^T A\right)_k$ under the Frobenious norm measurement of approximation error. This theorem also comes with a tractable bound on the optimal choice of the matrix B

Theorem 19 (Generalized Eckart-Young Theorem (Theorem 3.5 of [47])). *Given any matrix $A \in \mathbb{R}^{m\times n}$ and any column-orthonormal matrix $Q \in \mathbb{R}^{m\times \ell}$, let $B_k \in \mathbb{R}^{\ell\times n}$ be the rank-k truncated SVD of $Q^T A$. Then B_k is an optimal solution to the following problem*

$$\min_{\mathbf{rank}(B)\leq k} \|A - QB\|_F = \|A - QB_k\|_F$$

In addition, we also have

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

Another result of Ming Gu is the Reverse Eckart-Young Theorem. This allows us to take the bound from the last theorem and extends it to the spectral norm.

Theorem 20 (Reverse Eckart-Young (Theorem 3.4 of [47]))**.** *Assume that B is a rank-k approximation to A satisfying*

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

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

$$\begin{aligned} \|A - B\|_2^2 &\leq \sigma_{k+1}^2 + \eta^2 \\ \sum_{j=1}^{k} \left(\sigma_j - \sigma_j(B)\right)^2 &\leq \eta^2 \end{aligned}$$

The last inequality comes from a clever application of the Hoffman-Weilandt Theorem 16. Next, we present the singular value lower bounds from [47], which guarantee that we capture at least a fraction of the true singular values.

Theorem 21 (Gu's Deterministic Singular Value Lower Bound [47])**.** *Let $2 \leq p \leq \ell$ and let $Q \in \mathbb{R}^{m \times \ell}$ be an orthogonal matrix with the same column space as above. Then, for all $1 \leq j \leq \ell - p$, we have*

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

7.5.1.2 Basics of Majorisation and Doubly Stochastic Matrices

This introduction to majorisation and double stochastic matrices is presented in a similar fashion to [14]. Let $\mathbf{x} = \begin{pmatrix} x_1, x_2, \cdots, x_n \end{pmatrix}$ denote a vector in $\mathbb{R}^n$. Define $\mathbf{x}^{\downarrow} \in \mathbb{R}^n$ to be the $\mathbf{x}$ vector with the coordinates permuted so that $x_1^{\downarrow} \geq x_2^{\downarrow} \geq \cdots \geq x_n^{\downarrow}$.

Definition 7 (Majorisation)**.** *Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. We say that $\mathbf{x}$ is* majorised *by $\mathbf{y}$, or $\mathbf{x} \prec \mathbf{y}$ in symbols, if we have that*

$$\sum_{j=1}^{k} \mathbf{x}_j^{\downarrow} \leq \sum_{j=1}^{k} \mathbf{y}_j^{\downarrow} \quad \textit{for all } 1 \leq k \leq n.$$

and

$$\sum_{j=1}^{n} \mathbf{x}_j = \sum_{j=1}^{n} \mathbf{y}_j$$

are both satisfied.

It is important to note that if $\mathbf{x} \prec \mathbf{y}$ the definition above implies the following about the smaller coordinates of these vectors

$$\sum_{j=k}^{n} \mathbf{x}_j^{\downarrow} \geq \sum_{j=k}^{n} \mathbf{y}_j^{\downarrow} \text{ for all } 1 \leq k \leq n.$$

Definition 8 (Doubly Stochastic Matrix)**.** *Let $S \in \mathbb{R}^{n \times n}$. We call S a* **doubly stochastic** *matrix if*

$$s_{ij} \geq 0 \text{ for all } 1 \leq i, j \leq n \tag{7.15}$$

$$\sum_{i=1}^{n} s_{ij} = 1 \text{ for all } 1 \leq i \leq n \tag{7.16}$$

$$\sum_{j=1}^{n} s_{ij} = 1 \text{ for all } 1 \leq j \leq n \tag{7.17}$$

The concepts of majorisation and doubly stochastic matrices are related by the next result from Theorem II.1.10 of Bhatia [14]

Theorem 22 (Doubly stochastic characterization of majorisation [14])**.** *For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. We have $\mathbf{x} \prec \mathbf{y}$ if and only if $\mathbf{x} = S\mathbf{y}$ for some doubly stochastic matrix $S \in \mathbb{R}^{n \times n}$.*

We end this discussion of doubly stochastic matrices with a deep and powerful characterization of double stochastic matrices as the convex combination of permutation matrices from Theorem II.2.3 of [14]

Theorem 23 (Birkhoff's Theorem [14])**.** *The set of $n \times n$ doubly stochastic matrices is a convex set whose extreme points are the permutation matrices. In other words, any doubly stochastic matrix $S \in \mathbb{R}^{n \times n}$ can be written as a convex combination of the $n \times n$ permutation matrices.*

Let $A, B \in \mathbb{C}^{n \times n}$ be Hermitian matrices and let $\lambda(A), \lambda(B), \lambda(A+B) \in \mathbb{R}^n$ be n-dimensional real vectors of A, B and $A + B$, respectively. Each λ vector is in $\mathbb{R}^n$ instead of $\mathbb{C}^n$ by the spectral theorem for Hermitian matrices. Here, we restate Lidskii's theorem in terms of our new notation and tools.

Corollary 2 (Lidskii's theorem [14])**.** *We have that*

$$\lambda^{\downarrow}(A) + \lambda^{\uparrow}(B) \prec \lambda(A+B) \prec \lambda^{\downarrow}(A) + \lambda^{\downarrow}(B)$$

In other words, there exists two doubly stochastic matrices $S_1, S_2 \in \mathbb{R}^{n \times n}$ such that

$$\begin{aligned} \lambda^{\downarrow}(A) + \lambda^{\uparrow}(B) &= S_1 \lambda(A+B) \\ \lambda(A+B) &= S_2 \left(\lambda^{\downarrow}(A) + \lambda^{\downarrow}(B)\right) \end{aligned}$$

Proof. The first line comes from applying Lidskii's Theorem 17 with $i_j = j$. The last two lines come from applying Theorem 22. □

7.5.2 Useful probability results

The probabilistic analysis of randomized subspace iteration depends largely on two random variables: $\left\|\Omega_1^\dagger\right\|_2$ and $\|\Omega_2\|_2$. We will use the following preliminary results to control these random variables.

Theorem 24 (Density function bound for smallest singular value of Gaussian matrix [23])**.** *Let $G \in \mathbb{R}^{m\times n}$ be a standard Gaussian random matrix with $n \geq m$ and let $f_{\sigma^2_{\min}}(x)$ denote the probability density function of $\sigma^2_{\min}(G) = \left\|G^\dagger\right\|_2^{-2}$, then $f_{\sigma^2_{\min}}(x)$ satisfies*

$$L_{m,n}e^{-\frac{mx}{2}}x^{\frac{1}{2}(n-m-1)} \leq f_{\sigma^2_{\min}}(x) \leq L_{m,n}e^{-\frac{x}{2}}x^{\frac{1}{2}(n-m-1)} \tag{7.18}$$

where

$$L_{m,n} = \frac{2^{\frac{n-m-1}{2}}\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{m}{2}\right)\Gamma\left(n-m+1\right)} \tag{7.19}$$

Next, we introduce the framework use to bound the variable $\|\Omega_2\|_2$ in our randomized subspace iteration analysis.

Theorem 25 (Concentration of measure for Lipschitz functions of a Gaussian matrix [65])**.** *Suppose that $f(x)$ is a Lipschitz function on matrices:*

$$|f(A) - f(B)| \leq L_f \|A - B\|_F \qquad \text{for all } A, B \in \mathbb{R}^{m\times n}$$

Sample a matrix $G \in \mathbb{R}^{m\times n}$ with independent standard Gaussian $\mathcal{N}(0,1)$ entries. Then, for all $t \geq 0$,

$$\mathbb{P}\left\{f(G) \geq \mathbb{E}\left[f(G)\right] + L_f t\right\} \leq e^{-\frac{t^2}{2}}$$
$$\mathbb{P}\left\{f(G) \leq \mathbb{E}\left[f(G)\right] - L_f t\right\} \leq e^{-\frac{t^2}{2}}$$

The Lipschitz maps of interest for our analysis will be the singular values of a matrix– most importantly the largest singular value $\|\Omega_2\|_2$. Next, we produce a concentration of measure for matrix singular values.

Corollary 3 (Concentration of measure for singular values of a Gaussian matrix)**.** *Sample a matrix $G \in \mathbb{R}^{m\times n}$ with independent standard Gaussian $\mathcal{N}(0,1)$ entries. Then, for all $t \geq 0$, we have*

$$\mathbb{P}\left\{\sigma_j(G) \geq \mathbb{E}\left[\sigma_j(G)\right] + t\right\} \leq e^{-\frac{t^2}{2}}$$
$$\mathbb{P}\left\{\sigma_j(G) \leq \mathbb{E}\left[\sigma_j(G)\right] - t\right\} \leq e^{-\frac{t^2}{2}}$$

for each $1 \leq j \leq \min(m,n)$.

Proof. The Hoffman-Weilandt theorem 16 gives us

$$|\sigma_j(A) - \sigma_j(B)| \leq \sqrt{\sum_{i=1}^{\min(m,n)} |\sigma_i(A) - \sigma_i(B)|^2} \leq \|A - B\|_F$$

for each $1 \leq j \leq \min(m,n)$. As a result, we have that each map $\sigma_j(A)$ is Lipschitz with constant $L_f = 1$. Thus, we apply Theorem 25 to get the desired result. □

In order to make use of the above concentration of measure, we need to bound the expectation $\mathbb{E}\|\Omega_2\|$ as follows

Theorem 26 (Expected value of norms of scaled Gaussian matrix [50])**.** *Let $A \in \mathbb{R}^{m\times r}$, $B \in \mathbb{R}^{s\times n}$ be fixed matrices and let $\Omega \in \mathbb{R}^{r\times s}$ be an iid Gaussian matrix. Then, we have*

$$\mathbb{E}\|A\Omega B\|_2 \leq \|A\|_F \|B\|_2 + \|A\|_2 \|B\|_F$$
$$\left(\mathbb{E}\|A\Omega B\|_F^2\right)^{\frac{1}{2}} = \|A\|_F \|B\|_F$$

Next, we introduce a fundamental result from probability theory and machine learning involving taking the expectation of convex or concave functions. We will make use of it to both simplify our analysis and to improve the rate of convergence of our low-rank approximation methods.

Theorem 27 (Jensen's Inequality)**.** *Let X be a random variable. If $\phi : \mathbb{R}^n \to \mathbb{R}$ is a concave function, then we have*

$$\mathbb{E}[\phi(X)] \leq \phi(\mathbb{E}[X])$$

Also, if $\psi : \mathbb{R}^n \to \mathbb{R}$ is a convex function, then we get

$$\psi(\mathbb{E}[X]) \leq \mathbb{E}[\psi(X)]$$

An important application of Jensen's Inequality in this work is towards passing discrete arithmetic averages inside concave/convex functions. This will help give spectral decay an even larger impact in our error bounds. This scenario will be handled by the following corollary of Jensen's inequality for the discrete uniform probability measure over a finite outcome space.

Corollary 4. *Let $x_j \in \mathbb{R}^n$ be a fixed number for each $1 \leq j \leq k$. If $\phi : \mathbb{R}^n \to \mathbb{R}$ is a concave function, then we have that*

$$\frac{1}{k}\sum_{j=1}^{k} \phi(x_j) \leq \phi\left(\frac{1}{k}\sum_{j=1}^{k} x_j\right)$$

Also, if $\psi : \mathbb{R}^n \to \mathbb{R}$ is a convex function, then we have that

$$\psi\left(\frac{1}{k}\sum_{j=1}^{k} x_j\right) \leq \frac{1}{k}\sum_{j=1}^{k} \psi(x_j)$$

Proof. Define the uniform random variable $X \in \mathbb{R}$ such that $\mathbb{P}(X = x_j) = \frac{1}{k}$ for each $1 \leq j \leq k$. Then apply Jensen's inequality 27 to arrive at the result. Another common proof would be to recursively apply the standard definition for concave/convex function for two points. □

7.5.3 Miscellaneous

Lemma 17 (Gamma function identities and inequalities)**.** *Let $\Gamma(x) = \int_0^\infty e^{-t}t^{x-1}dt$ be the Gamma function where $x > 0$ then we have that*

$$\Gamma(x+1) = x\Gamma(x) \tag{7.20}$$

along with the following inequalities

$$\Gamma\left(x+\frac{1}{2}\right) \leq \sqrt{x}\Gamma(x) \text{ and } \sqrt{2\pi}x^{x+\frac{1}{2}}e^{-x} < \Gamma(x+1) < \sqrt{2\pi}x^{x+\frac{1}{2}}e^{-x+\frac{1}{12x}} \tag{7.21}$$

7.5.4 Matrix Approximation Error Bounds

We consider the Generalized Eckart-Young Theorem 19 in combination with the Reverse Eckart-Young Theorem 20 [47], which gives the bound

$$\left\|A - Q\left(Q^TA\right)_k\right\|_\xi^2 \leq \|A - A_k\|_\xi^2 + \left\|\left(I - QQ^T\right)A_k\right\|_F^2$$

where $\xi = 2, F$. Therefore, controlling the magnitude of $\left\|\left(I - QQ^T\right)A_k\right\|_F^2$ is a rigourous way of bounding the error of low-rank matrix approximation algorithms. Before proceeding to study the term $\left\|\left(I - QQ^T\right)A_k\right\|_F^2$, we prove this important technical lemma

Lemma 18. *Let $A, B \in \mathbb{R}^{k\times k}$ be symmetric positive definite (PD) matrices. Then, we have that*

$$\mathbf{tr}\left(\left(A^{-1}+B^{-1}\right)^{-1}\right) \leq \sum_{j=1}^{k}\frac{\lambda_j(A)\lambda_j(B)}{\lambda_j(A)+\lambda_j(B)}$$

Proof. Corollary 2 gives us that there exists a doubly stochastic matrix $S \in \mathbb{R}^{n\times n}$ such that

$$\lambda^\downarrow\left(A^{-1}+B^{-1}\right) = S\left(\lambda^\downarrow(A^{-1}) + \lambda^\downarrow(B^{-1})\right)$$

By definition, the row sums of a doubly stochastic matrix all equal one and each entry of a doubly stochastic matrix satisfies $0 \leq s_{ij} \leq 1$. Thus, by matrix-vector multiplication, the above line gives that the i^{th} largest eigenvalue of $A + B$ is given as

$$\lambda_i(A^{-1}+B^{-1}) = \sum_{j=1}^{n} s_{ij}\left(\lambda_j(A^{-1}) + \lambda_j(B^{-1})\right)$$

a convex combination of the entries of the vector $\left(\lambda^{\downarrow}(A) + \lambda^{\downarrow}(B)\right)$. Next, we note that the function $f(t) = \frac{1}{t}$ is convex for $t > 0$, which is seen easily by inspecting the second derivative. Therefore, we have

$$\begin{aligned}
\mathbf{tr}\left(\left(A^{-1} + B^{-1}\right)^{-1}\right) &= \sum_{i=1}^{n} \lambda_i \left(\left(A^{-1} + B^{-1}\right)^{-1}\right) \\
&= \sum_{i=1}^{n} \frac{1}{\lambda_i \left(A^{-1} + B^{-1}\right)} \\
&= \sum_{i=1}^{n} \frac{1}{\sum_{j=1}^{n} s_{ij} \left(\lambda_j \left(A^{-1}\right) + \lambda_j \left(B^{-1}\right)\right)} \\
&\leq \sum_{i=1}^{n} \sum_{j=1}^{n} s_{ij} \frac{1}{\left(\lambda_j \left(A^{-1}\right) + \lambda_j \left(B^{-1}\right)\right)} \\
&= \sum_{j=1}^{n} \frac{1}{\left(\frac{1}{\lambda_j(A)} + \frac{1}{\lambda_j(B)}\right)}
\end{aligned}$$

where the last line is acheived by the fact that the column sums of a doubly stochastic matrix equal one, i.e. $\sum_{i=1}^{n} s_{ij} = 1$. □

Now, this lemma allows us to get a strong control on the quantity $\left\|\left(I - QQ^T\right) A_k\right\|_F$.

Theorem 28. *Given the setup described in Section 7.4, we have that*

$$\left\|A - Q\left(Q^T A\right)_k\right\|_{\xi}^2 \leq \|A - A_k\|_{\xi}^2 + \frac{\|\Sigma_1\|_F^2 \|H_1\Sigma_1\|_F^2}{\|\Sigma_1\|_F^2 + \|H_1\Sigma_1\|_F^2} \tag{7.22}$$

Proof. First, apply Lemma 16 to get

$$\begin{aligned}
\left\|\left(I - QQ^T\right) A_k\right\|_F^2 &\leq \left\|\left(I - Q_1 Q_1^T\right) A_k\right\|_F^2 \\
&= \left\| \left(I - \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix} \left(I + H_1^T H_1\right)^{-1} \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix}^T \right) \begin{pmatrix} \Sigma_1 & & \\ & 0 & \\ & & 0 \end{pmatrix} \right\|_F^2 \\
&= \left\| \begin{pmatrix} H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 & 0 & -H_1^T \left(I + H_1 H_1^T\right)^{-1} \\ 0 & I & 0 \\ -\left(I + H_1 H_1^T\right)^{-1} H_1 & 0 & \left(I + H_1 H_1^T\right)^{-1} \end{pmatrix} \begin{pmatrix} \Sigma_1 & & \\ & 0 & \\ & & 0 \end{pmatrix} \right\|_F^2 \\
&= \mathbf{tr}\left(\Sigma_1 H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 \Sigma_1\right)
\end{aligned}$$

Massaging the expression within the trace gives

$$\Sigma_1 H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 \Sigma_1 = \Sigma_1 \left(\left(H_1^T H_1\right)^{-1} + I\right)^{-1} \Sigma_1 = \left(\left(\left(H_1 \Sigma_1\right)^T \left(H_1 \Sigma_1\right)\right)^{-1} + \Sigma_1^{-2}\right)^{-1}$$

Combine this with the trace expression and apply Lemma 18 to get

$$\begin{aligned}\left\|\left(I-QQ^T\right)A_k\right\|_F^2 &\leq \mathbf{tr}\left(\left(\left(\left(H_1\Sigma_1\right)^T\left(H_1\Sigma_1\right)\right)^{-1}+\Sigma_1^{-2}\right)^{-1}\right)\\ &\leq \sum_{j=1}^{k}\frac{\sigma_j^2(\Sigma_1)\sigma_j^2(H_1\Sigma_1)}{\sigma_j^2(\Sigma_1)+\sigma_j^2(H_1\Sigma_1)}\\ &\stackrel{def}{=}\sum_{j=1}^{k} f\left(\sigma_j^2,\sigma_j^2\left(H_1\Sigma_1\right)\right)\end{aligned}$$

where the function $f:\mathbb{R}_+^2\to\mathbb{R}$ is defined as

$$f(x,y)=\frac{xy}{x+y}. \tag{7.23}$$

This function is concave and monotonically increasing in both variables. This is seen by inspecting the first derivatives of $f(x,y)$

$$\frac{\partial f}{\partial x}(x,y)=\frac{y^2}{(x+y)^2}>0 \text{ and } \frac{\partial f}{\partial y}(x,y)=\frac{x^2}{(x+y)^2}>0$$

to get monotonicity and by looking at the second derivative Hessian

$$\mathbf{H}=\begin{pmatrix}\frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x\partial y}\\ \frac{\partial^2 f}{\partial x\partial y} & \frac{\partial^2 f}{\partial y^2}\end{pmatrix}=-\frac{2}{(x+y)^3}\begin{pmatrix}y^2 & xy\\ xy & x^2\end{pmatrix}=-\frac{2(x^2+y^2)}{(x+y)^3}\begin{bmatrix}\frac{y}{\sqrt{x^2+y^2}}\\ \frac{x}{\sqrt{x^2+y^2}}\end{bmatrix}\begin{bmatrix}\frac{y}{\sqrt{x^2+y^2}} & \frac{x}{\sqrt{x^2+y^2}}\end{bmatrix}$$

to see that it has a zero eigenvalue and another non-positive eigenvalue. Next, we apply Jensen's inequality 4 to get

$$\left\|\left(I-QQ^T\right)A_k\right\|_F^2\leq k\sum_{j=1}^{k}\frac{1}{k}\frac{\sigma_j^2(\Sigma_1)\sigma_j^2(H_1\Sigma_1)}{\sigma_j^2(\Sigma_1)+\sigma_j^2(H_1\Sigma_1)}\leq k\frac{\frac{1}{k}\left\|\Sigma_1\right\|_F^2\frac{1}{k}\left\|H_1\Sigma_1\right\|_F^2}{\frac{1}{k}\left\|\Sigma_1\right\|_F^2+\frac{1}{k}\left\|H_1\Sigma_1\right\|_F^2}=\frac{\left\|\Sigma_1\right\|_F^2\left\|H_1\Sigma_1\right\|_F^2}{\left\|\Sigma_1\right\|_F^2+\left\|H_1\Sigma_1\right\|_F^2}$$

□

Remark 7.5.1. *Given the monotonicity of $f(x,y)$ in equation (7.23) and the fact that $\|C\|_F^2\leq k\|C\|_2^2$ for all $C\in\mathbb{R}^{k\times k}$, the above result implies*

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

where the last expression is equal to the bound in Theorem 4.4 of [47].

In addition to being tighter than previous work in the literature, this result also implies a tractable and easy to understand bound

Corollary 5.

$$\left\|A - Q\left(Q^T A\right)_k\right\|_\xi^2 \leq \|A - A_k\|_\xi^2 + k\sigma_{\ell-p+1}^2 \frac{\tau^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + \alpha^2\tau^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2} \tag{7.24}$$

where $\tau_F^2 = \frac{1}{k}\sum_{j=1}^k \frac{\sigma_{\ell-p+j}^2}{\sigma_{\ell-p+1}^2}\left(\frac{\sigma_{\ell-p+j}}{\sigma_{k+1-j}}\right)^{4q}$ *is the spectral decay rate and where* $\alpha^2 = \frac{\sigma_{\ell-p+1}^2}{\frac{1}{k}\|\Sigma_1\|_F^2}$

Proof. Observe that $H_1\Sigma_1$ has the structure $H_1\Sigma_1 = \Sigma_B^{2q+1}\widehat{\Omega}_2\widehat{\Omega}_1^\dagger \begin{pmatrix} \Sigma_1^{-2q} \\ 0 \end{pmatrix}$. Therefore, we employ the cyclic permutation invariance of the trace (a.k.a. "trace trick") and John von Neumann's trace inequality from Theorem 18 to get

$$\begin{aligned}
\|H_1\Sigma_1\|_F^2 &= \mathbf{tr}\left(\Sigma_1 H_1^T H_1 \Sigma_1\right) \\
&= \mathbf{tr}\left(\left[\widehat{\Omega}_2^T \Sigma_B^{4q+2}\widehat{\Omega}_2\right]\left[\widehat{\Omega}_1^\dagger \begin{pmatrix} \Sigma_1^{-4q} & 0 \\ 0 & 0 \end{pmatrix}\left(\widehat{\Omega}_1^\dagger\right)^T\right]\right) \\
&\leq \sum_{j=1}^k \sigma_j^2\left(\Sigma_B^{2q+1}\widehat{\Omega}_2\right)\sigma_j^2\left(\widehat{\Omega}_1^\dagger \begin{pmatrix} \Sigma_1^{-2q} \\ 0 \end{pmatrix}\right) \\
&\leq \sum_{j=1}^k \sigma_{\ell-p+j}^2 \left(\frac{\sigma_{\ell-p+j}}{\sigma_{k+1-j}}\right)^{4q} \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2
\end{aligned}$$

□

In the case of the 2-norm non-truncated matrix approximation error, we can produce a tighter bound using the following lemma, which is similar to the Matrix Pythagoras result of [25].

Lemma 19 (Quasi-Polarization Inequality for Spectral Norm)**.** *Let* $A, B \in \mathbb{R}^{m\times n}$ *be any two matrices, then*

$$\|A + B\|_2^2 \leq \|A\|_2^2 + 2\min\left\{\left\|B^T A\right\|_2, \left\|BA^T\right\|_2\right\} + \|B\|_2^2 \tag{7.25}$$

Proof. Theorem 15 gives us that

$$\begin{aligned}
\|A + B\|_2^2 &= \sigma_1\left((A+B)^T(A+B)\right) \\
&= \sigma_1\left(A^TA + A^TB + B^TA + B^TB\right) \\
&\leq \sigma_1\left(A^TA\right) + \sigma_1\left(A^TB\right) + \sigma_1\left(B^TA\right) + \sigma_1\left(B^TB\right) \\
&= \|A\|_2^2 + 2\left\|B^TA\right\|_2 + \|B\|_2^2
\end{aligned}$$

A similar argument gives

$$\|A+B\|_2^2 = \sigma_1\left((A+B)(A+B)^T\right) \leq \|A\|_2^2 + 2\left\|BA^T\right\|_2 + \|B\|_2^2$$

□

For the weak-form low rank approximation problem in the 2-norm, we can remove the factor of k in the last term of inequality (7.24) at the expense of a slower spectral decay rate by the following result.

Lemma 20 (Deterministic Non-Truncation Structural Result for Spectral Norm)**.** *Given the setup described in Section 7.4, we have that*

$$\begin{aligned}\left\|A - QQ^TA\right\|_2^2 &\leq \sigma_{k+1}^2 + \frac{\|H_1\Sigma_1\|_2^2\,\sigma_1^2}{\sigma_1^2 + \|H_1\Sigma_1\|_2^2} \\ &\leq \sigma_{k+1}^2 + \frac{\tau_k^{4q}\left\|\widehat{\Omega}_2\right\|_2^2\left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1+\alpha_2^2\tau_k^{4q}\left\|\widehat{\Omega}_2\right\|_2^2\left\|\widehat{\Omega}_1^\dagger\right\|_2^2}\sigma_{\ell-p+1}^2\end{aligned}$$

where $\tau_k = \frac{\sigma_{\ell-p+1}}{\sigma_k}$ *and* $\alpha_2 = \frac{\sigma_{\ell-p+1}}{\sigma_1}$.

Proof. By Lemma 19, we have that

$$\begin{aligned}\left\|A-QQ^TA\right\|_2^2 &= \left\|\left(I-QQ^T\right)(A-A_k) + \left(I-QQ^T\right)A_k\right\|_2^2 \\ &\leq \left\|\left(I-QQ^T\right)(A-A_k)\right\|_2^2 + 2\left\|\left(I-QQ^T\right)A_k(A-A_k)^T\left(I-QQ^T\right)\right\|_2 \\ &\qquad + \left\|\left(I-QQ^T\right)A_k\right\|_2^2 \\ &= \left\|\left(I-QQ^T\right)(A-A_k)\right\|_2^2 + \left\|\left(I-QQ^T\right)A_k\right\|_2^2\end{aligned}$$

where the last line is achieved by the fact that $A_k\left(A-A_k\right)^T = 0$ from the properties of the SVD. Next, we apply Lemma 16 twice to arrive at

$$\left\|\left(I-QQ^T\right)A\right\|_2^2 \leq \|A-A_k\|_2^2 + \left\|\left(I-Q_1Q_1^T\right)A_k\right\|_2^2$$

Then, we bound the last term of the above

$$\begin{aligned}
\left\|\left(I - Q_1 Q_1^T\right) A_k\right\|_2^2 &= \left\| \left(I - \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix} \left(I + H_1^T H_1\right)^{-1} \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix}^T \right) \begin{pmatrix} \Sigma_1 & & \\ & 0 & \\ & & 0 \end{pmatrix} \right\|_2^2 \\
&= \left\| \begin{pmatrix} H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 & 0 & -H_1^T \left(I + H_1 H_1^T\right)^{-1} \\ 0 & I & 0 \\ -\left(I + H_1 H_1^T\right)^{-1} H_1 & 0 & \left(I + H_1 H_1^T\right)^{-1} \end{pmatrix} \begin{pmatrix} \Sigma_1 & & \\ & 0 & \\ & & 0 \end{pmatrix} \right\|_2^2 \\
&= \left\| \Sigma_1 H_1^T \left(I + H_1 H_1^T\right)^{-1} H_1 \Sigma_1 \right\|_2 \\
&\leq \frac{\|H_1 \Sigma_1\|_2^2 \sigma_1^2}{\sigma_1^2 + \|H_1 \Sigma_1\|_2^2}
\end{aligned}$$

Putting this all together, we arrive at our result. □

7.6 Revised Probabilistic Analysis of Subspace iteration: Independence is King

Controlling the magnitude of $\|\widehat{\Omega}_2\|_2$ and $\|\widehat{\Omega}_1^\dagger\|_2$ is a critical part of the analysis of randomized subspace iteration as in the seminal work of Halko, Martinsson, Tropp [50] as well as the seminal work of Gu [47]. Both of these works bound the size of $\|\widehat{\Omega}_2\|_2$ and $\|\widehat{\Omega}_1^\dagger\|_2$ *individually* as two separate random variables. First, conditioned on $\|\widehat{\Omega}_1^\dagger\|_2$, both works apply Corollary 3 to bound $\|\widehat{\Omega}_2\|_2$ either in expectation or large deviation. Then, they apply the following result to bound the remaining terms with $\|\widehat{\Omega}_1^\dagger\|_2$.

Lemma 21 (Large deviation for pseudo-inverted Gaussian matrix [23, 50, 47])**.** *Let G be an $(\ell - p) \times \ell$ Gaussian matrix where $p \geq 0$ and $\ell - p \geq 2$. Then G has full rank with probability 1 and we have that for all $t \geq 1$,*

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

But, why do all of this? All that truly matters for randomized subspace iteration is the *product* of these two *independent* factors. The work of Chen and Dongarra [23] produce results for a similar random variable, the condition number $\|\Omega\|_2 \|\Omega^\dagger\|_2$ of a Gaussian random matrix $\Omega \in \mathbb{R}^{m \times n}$, which is a similar product–except that the two matrices are the same, thereby inducing a natural coupling between the two terms of the product (i.e. the condition number of Ω). In a way, the case that we are faced with is much easier and more tractable because the two terms of the product $\|\widehat{\Omega}_2\|_2 \|\widehat{\Omega}_1^\dagger\|_2$ are independent of each other. The works of Halko et. al. [50] and Gu [47] do not take full advantage of this fact by making excessive

and unnecessary usage of union bounds and conditioning instead of exploiting the inherent structure between these two random variables.

Theorem 29. *Let $\Omega_1 \in \mathbb{R}^{(\ell-p)\times\ell}$ and $\Omega_2 \in \mathbb{R}^{(n-\ell+p)\times\ell}$ be independent random matrices, each with iid standard Gaussian $\mathcal{N}(0,1)$ entries. Then, we have that*

$$\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq \frac{\mathcal{B}\sqrt{\ell}\,\mathcal{E}}{p+1}\,t\right) \leq t^{-(p+1)}$$

where $\mathcal{E} = \sqrt{n-\ell+p} + \sqrt{\ell}$ is a function of $p < \ell < n$ and $\mathcal{B} \leq 3.0237$ is a universal constant.

Proof.

$$\begin{aligned}
\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq t\right) &= \mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2^2 \geq t^2\sigma_{\min}^2\left(\widehat{\Omega}_1\right)\right) \\
&= \int_0^\infty \int_{t^2 z}^\infty f_{\sigma_{\min}^2(\widehat{\Omega}_1)}(z)\, f_{\sigma_{\max}^2(\widehat{\Omega}_2)}(y)\, dydz \\
&= \int_0^\infty f_{\sigma_{\min}^2(\widehat{\Omega}_1)}(z)\, \mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2^2 \geq t^2 z\right) dz \\
&= \int_0^\infty 2x f_{\sigma_{\min}^2(\widehat{\Omega}_1)}(x^2)\, \mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2^2 \geq t^2 x^2\right) dx
\end{aligned}$$

where the $2x$ in the integrand comes from the Jacobian for the change of variables $z = x^2$. Next, we apply Theorem 24 to upper bound the probability density function $f_{\sigma_{\min}^2}$ from above.

$$\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq t\right) \leq 2L_{\ell-p,\ell}\int_0^\infty e^{-\frac{x^2}{2}} x^p \mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2^2 \geq t^2x^2\right) dx \quad \text{where} \quad L_{\ell-p,\ell} = \frac{2^{\frac{p-1}{2}}\Gamma\left(\frac{\ell+1}{2}\right)}{\Gamma\left(\frac{\ell-p}{2}\right)\Gamma(p+1)} \tag{7.26}$$

Next, we employ Theorem 25 in order to bound the tail distribution for $\left\|\widehat{\Omega}_2\right\|_2$. To be able to use this, we rely on Theorem 26 to bound the expected value of

$$\mathbb{E}\left\|\widehat{\Omega}_2\right\|_2 = \mathbb{E}\left\|I_{n-\ell+p}\widehat{\Omega}_2 I_\ell\right\|_2 \leq \|I_{n-\ell+p}\|_F\|I_\ell\|_2 + \|I_{n-\ell+p}\|_2\|I_\ell\|_F = \sqrt{n-\ell+p} + \sqrt{\ell} \stackrel{def}{=} \mathcal{E}$$

so that we can actually use the tail bound from Theorem 25. The spectral norm is a Lipschitz map with Lipschitz constant equal to 1. Thus, Theorem 25 gives

$$\mathbb{P}\left\{\left\|\widehat{\Omega}_2\right\|_2 \geq \mathcal{E} + u\right\} \leq \mathbb{P}\left\{\left\|\widehat{\Omega}_2\right\|_2 \geq \mathbb{E}\left\|\widehat{\Omega}_2\right\|_2 + u\right\} \leq e^{-\frac{u^2}{2}}$$

for all $u \geq 0$. Using this result, we construct the following upper bound to the entire tail distribution

$$\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \geq tx\right) \leq \begin{cases} \exp\left(-\frac{(tx-\mathcal{E})^2}{2}\right) & \text{, if } x \geq \mathcal{C}\frac{\mathcal{E}}{t} \\ 1 & \text{, otherwise} \end{cases}$$

where $\mathcal{C} \geq 1$ is a constant to be determined later. We now apply this to equation (7.26) to get

$$\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq t\right) \leq 2L_{\ell-p,\ell}\left[\underbrace{\int_0^{\mathcal{C}\frac{\mathcal{E}}{t}} e^{-\frac{x^2}{2}} x^p dx}_{I_1} + \underbrace{\int_{\mathcal{C}\frac{\mathcal{E}}{t}}^{\infty} e^{-\frac{x^2}{2}} x^p \exp\left(-\frac{t^2}{2}\left(x-\frac{\mathcal{E}}{t}\right)^2\right) dx}_{I_2}\right] \tag{7.27}$$

First, we tackle the first integral

$$I_1 = \int_0^{\frac{\mathcal{C}\mathcal{E}}{t}} e^{-\frac{x^2}{2}} x^p dx \leq \int_0^{\frac{\mathcal{C}\mathcal{E}}{t}} x^p dx \leq \frac{1}{p+1}\left(\frac{\mathcal{C}\mathcal{E}}{t}\right)^{p+1}$$

Next, comes a delicate computation for the second integral, which effectively reduces to bounding a truncated moment of a Gaussian random variable.

$$I_2 \leq \int_{\frac{\mathcal{C}\mathcal{E}}{t}}^{\infty} x^p \exp\left(-\frac{(tx-\mathcal{E})^2}{2}\right) dx = \left(\frac{1}{t}\right)^{p+1} \int_{\mathcal{C}\mathcal{E}}^{\infty} x^p \exp\left(-\frac{(x-\mathcal{E})^2}{2}\right) dx \tag{7.28}$$

Now, we employ a technique from [26] to bound this integral. Consider the following inequality

$$x^p = e^{p\ln(x)} = (\mathcal{C}\mathcal{E})^p \exp\left(p\ln\left(\frac{x}{\mathcal{C}\mathcal{E}}\right)\right) \leq (\mathcal{C}\mathcal{E})^p \exp\left(p\frac{x}{\mathcal{C}\mathcal{E}} - p\right)$$

where we use the first order condition for a concave function [19] to get that $\ln\left(\frac{x}{\mathcal{C}\mathcal{E}}\right) \leq \frac{x}{\mathcal{C}\mathcal{E}} - 1$. Applying this to equation (7.28), we see that the new integral is more malleable

$$\begin{aligned}
I_2 &\leq \left(\frac{1}{t}\right)^{p+1} (\mathcal{C}\mathcal{E})^p \int_{\mathcal{C}\mathcal{E}}^{\infty} \exp\left(-\frac{(x-\mathcal{E})^2}{2} + p\frac{x}{\mathcal{C}\mathcal{E}} - p\right) dx \\
&= \left(\frac{1}{t}\right)^{p+1} (\mathcal{C}\mathcal{E})^p \int_{\mathcal{C}\mathcal{E}}^{\infty} \exp\left(-\frac{\left(x-\mathcal{E}-\frac{p}{\mathcal{C}\mathcal{E}}\right)^2}{2} + \frac{p}{\mathcal{C}} - p + \frac{p^2}{2\mathcal{C}^2\mathcal{E}^2}\right) dx \\
&= \left(\frac{1}{t}\right)^{p+1} (\mathcal{C}\mathcal{E})^p \exp\left(\frac{p}{\mathcal{C}} - p + \frac{p^2}{2\mathcal{C}^2\mathcal{E}^2}\right) \int_{\mathcal{C}\mathcal{E}-\mathcal{E}-\frac{p}{\mathcal{C}\mathcal{E}}}^{\infty} e^{\frac{-x^2}{2}} dx \\
&\leq \frac{\sqrt{2\pi}}{\mathcal{C}\mathcal{E}}\left(\frac{\mathcal{C}\mathcal{E}}{t}\right)^{p+1} \exp\left(\frac{p}{\mathcal{C}} - p + \frac{p^2}{2\mathcal{C}^2\mathcal{E}^2}\right) \\
&\leq \frac{\sqrt{2\pi}}{\mathcal{C}\mathcal{E}}\left(\frac{\mathcal{C}\mathcal{E}}{t}\right)^{p+1} \exp\left(\frac{p}{\mathcal{C}^2}\left[\mathcal{C} - \mathcal{C}^2 + \frac{1}{8}\right]\right)
\end{aligned}$$

where the second to last line was achieved by using the identity $\int_{-\infty}^{\infty} e^{-\frac{x^2}{2}} dx = \sqrt{2\pi}$ and the last line comes from applying $\mathcal{E} = \sqrt{n-\ell+p} + \sqrt{\ell} \geq 2\sqrt{p}$. By applying the quadratic

equation, we deduce that if $\mathcal{C} = \frac{\sqrt{2}+\sqrt{3}}{2\sqrt{2}} \leq 1.1124$ then $\mathcal{C} - \mathcal{C}^2 + \frac{1}{8} = 0$, which results in

$$I_2 \leq \frac{\sqrt{2\pi}}{\mathcal{C}\mathcal{E}} \left(\frac{\mathcal{C}\mathcal{E}}{t}\right)^{p+1}$$

Placing this together with equation (7.27) followed by equation (7.19), we get

$$\begin{aligned}
\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq t\right) &\leq 2L_{\ell-p,\ell} \left(\frac{1}{p+1} + \sqrt{2\pi}\frac{1}{\mathcal{C}\mathcal{E}}\right) \left(\frac{\mathcal{C}\mathcal{E}}{t}\right)^{p+1} \\
&= \frac{2^{\frac{p+1}{2}}\Gamma\left(\frac{\ell+1}{2}\right)}{\Gamma\left(\frac{\ell-p}{2}\right)\Gamma\left(p+2\right)} \left(1 + \sqrt{2\pi}\frac{p+1}{\mathcal{C}\mathcal{E}}\right) \left(\frac{\mathcal{C}\mathcal{E}}{t}\right)^{p+1}
\end{aligned}$$

where we use the Gamma function identity $x\Gamma(x) = \Gamma(x+1)$ of Lemma 17 on the last line. Next, we employ the inequalities (7.21) from Lemma 17 to get

$$\begin{aligned}
\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq t\right) &\leq \frac{2^{\frac{p+1}{2}}\left(\frac{\ell}{2}\right)^{\frac{p+1}{2}}}{\Gamma\left(p+2\right)} \left(1 + \sqrt{2\pi}\frac{p+1}{\mathcal{C}\mathcal{E}}\right) \left(\frac{\mathcal{C}\mathcal{E}}{t}\right)^{p+1} \\
&\leq \left(\frac{1}{\sqrt{2\pi(p+1)}} + \frac{\sqrt{p+1}}{\mathcal{C}\mathcal{E}}\right) \left[\frac{e\mathcal{C}\sqrt{\ell}\mathcal{E}}{p+1}\right]^{p+1} \left(\frac{1}{t}\right)^{p+1} \\
&\leq \left(\frac{1}{\sqrt{2\pi(p+1)}} + \frac{1}{2\mathcal{C}}\right) \left[\frac{e\mathcal{C}\sqrt{\ell}\mathcal{E}}{p+1}\right]^{p+1} \left(\frac{1}{t}\right)^{p+1} \\
&\leq \left[\frac{e\mathcal{C}\sqrt{\ell}\mathcal{E}}{p+1}\right]^{p+1} \left(\frac{1}{t}\right)^{p+1}
\end{aligned}$$

In the line above, we use the strict inequality assumption between $p < \ell < n$ to get that $n - \ell \geq 1$ and $\ell - p \geq 1$, permitting us to bound

$$\mathcal{E} = \sqrt{n-\ell+p} + \sqrt{(\ell-p)+p} \geq 2\sqrt{p+1}$$

We define $\mathcal{B} = e\mathcal{C} \leq 3.0237$ to arrive at our conclusion

$$\mathbb{P}\left(\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq t\right) \leq \left[\frac{\mathcal{B}\sqrt{\ell}\mathcal{E}}{p+1}\right]^{p+1} \left(\frac{1}{t}\right)^{p+1}$$

□

7.6.1 Average Case Error Bounds for Subspace Iteration

We develop a fundamental lemma for the analysis of normed residual matrix approximation bounds and singular value lower bounds, which is to bound the second moment of the random variable $\left\|\widehat{\Omega}_2\right\|_2 \left\|\widehat{\Omega}_1^\dagger\right\|_2$ in Lemma 29

Lemma 22. *Let $\widehat{\Omega}_1 \in \mathbb{R}^{(\ell-p)\times\ell}$ and $\widehat{\Omega}_2 \in \mathbb{R}^{(n-\ell+p)\times\ell}$ be two independent random matrices each with iid $\mathcal{N}(0,1)$ standard normal entries. Also, let $\ell > p \geq 2$. Define*

$$\mathcal{K} \stackrel{def}{=} \beta\sqrt{\frac{\ell\mathcal{E}^2}{(p+1)(p-1)}}$$

where $\beta \leq 3.0237$ is a universal constant. Then, we have the bound on the second moment

$$\mathbb{E}\left\|\widehat{\Omega}_2\right\|_2^2\left\|\widehat{\Omega}_1^\dagger\right\|_2^2 \leq \frac{\beta^2\ell\mathcal{E}^2}{(p+1)(p-1)} \stackrel{def}{=} \mathcal{K}^2$$

Proof. Apply the law of the unconscious statistician with an arbitrary fixed constant $c \in \mathbb{R}_+$ to get

$$\begin{aligned}\mathbb{E}\left\|\widehat{\Omega}_2\right\|_2^2\left\|\widehat{\Omega}_1^\dagger\right\|_2^2 &= \int_0^\infty 2x\mathbb{P}\left\{\left\|\widehat{\Omega}_2\right\|_2\left\|\widehat{\Omega}_1^\dagger\right\|_2 \geq x\right\}dx \\ &\leq \int_0^c 2xdx + \int_c^\infty 2x\left(\frac{p+1}{\beta\sqrt{\ell}\mathcal{E}}x\right)^{-(p+1)} = c^2\left(1 + \frac{2}{p-1}\left(\frac{\beta\sqrt{\ell}\mathcal{E}}{p+1}\right)^{p+1}c^{-(p+1)}\right)\end{aligned}$$

Let $c = \frac{\beta\sqrt{\ell}\mathcal{E}}{p+1}$ and proceed

$$\mathbb{E}\left\|\widehat{\Omega}_2\right\|_2^2\left\|\widehat{\Omega}_1^\dagger\right\|_2^2 \leq \frac{\beta^2\ell\mathcal{E}^2}{(p+1)^2}\left(\frac{p+1}{p-1}\right) = \frac{\beta^2\ell\mathcal{E}^2}{(p+1)(p-1)} = \mathcal{K}^2$$

□

Let us define two functions $g : \mathbb{R}_+ \to \mathbb{R}_+$ and $h : \mathbb{R}_+ \to \mathbb{R}_+$

$$\begin{aligned} g(x) &= \frac{x}{1+d^2x} & & & h(x) &= \frac{1}{\sqrt{1+d^2x}} \\ g'(x) &= \frac{1}{(1+d^2x)^2} & & \text{and} & h'(x) &= -\frac{d^2}{2\left(1+d^2x\right)^{(3/2)}} \\ g''(x) &= -\frac{2d^2}{\left(1+d^2x\right)^3} & & & h''(x) &= \frac{3d^4}{4\left(1+d^2x\right)^{(5/2)}} \end{aligned} \tag{7.29}$$

with $d > 0$ fixed. From the above, we conclude that $g(\cdot)$ and $h(\cdot)$ are monotonically increasing and decreasing, respectively, by the first derivative test. Also, we deduce that $g(\cdot)$ and $h(\cdot)$ are concave and convex, respectively, by the second derivative test. The function $g(\cdot)$ is important in the analysis of normed residual matrix approximation bounds, while $h(\cdot)$ is fundamental to the analysis of singular value lower bounds. The identity $d^2g(x) + h^2(x) = 1$ is important to understanding the trigonometric structure and relationship between normed residual matrix approximation bounds and singular value lower bounds.

Lemma 23. *Let $b, d \geq 0$ and let $\widehat{\Omega}_1 \in \mathbb{R}^{(\ell-p)\times\ell}$ and $\widehat{\Omega}_2 \in \mathbb{R}^{(n-\ell+p)\times\ell}$ be two independent random matrices each with iid $\mathcal{N}(0,1)$ standard normal entries. Let $\ell > p \geq 2$ from the Setup section, then we have*

$$\mathbb{E}\frac{b^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + d^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\Omega_1^\dagger\right\|_2^2} \leq \frac{b^2\mathcal{K}^2}{1+d^2\mathcal{K}^2}$$

where $\mathcal{K} = \beta\sqrt{\frac{\ell\mathcal{E}^2}{(p+1)(p-1)}}$ and $\beta \leq 3.0237$ is a universal constant.

Proof. Note that $g(\cdot)$ from equation (7.29) is concave, which allows us to apply Jensen's inequality to get

$$\mathbb{E}\frac{b^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}{1 + d^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2} = \mathbb{E}b^2 g\left(\left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2\right) \leq b^2 g\left(\mathbb{E}\left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2\right) \leq b^2 g\left(\mathcal{K}^2\right)$$

where the last inequality comes from the monotonicity of $g(\cdot)$ and the application of Lemma 22. □

Lemma 24. *Let $d \geq 0$ and let $\widehat{\Omega}_1 \in \mathbb{R}^{(\ell-p)\times\ell}$ and $\widehat{\Omega}_2 \in \mathbb{R}^{(n-\ell+p)\times\ell}$ be two independent random matrices each with iid $\mathcal{N}(0,1)$ standard normal entries. Let $\ell > p \geq 2$ from the Setup section, then we have*

$$\mathbb{E}\frac{1}{\sqrt{1 + d^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}} \geq \frac{1}{1+d^2\mathcal{K}^2}$$

where $\mathcal{K} = \beta\sqrt{\frac{\ell\mathcal{E}^2}{(p+1)(p-1)}}$ and $\beta \leq 3.0237$ is a universal constant.

Proof. We start by applying Jensen's inequality of Theorem 27 to the convex function $h(\cdot)$ from equation (7.29) under the random variable of interest to get

$$\mathbb{E}\frac{1}{\sqrt{1 + d^2 \left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2}} = \mathbb{E}h\left(\left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2\right) \geq h\left(\mathbb{E}\left\|\widehat{\Omega}_2\right\|_2^2 \left\|\widehat{\Omega}_1^\dagger\right\|_2^2\right) \geq h\left(\mathcal{K}^2\right)$$

where lemma 22 is invoked along with the monotonicity of $h(\cdot)$ from equation (7.29) to arrive at the last inequality. □

Acknowledgments. I would like to thank my adviser Prof. Ming Gu for helpful comments on this randomized subspace iteration and for getting me interested in this area of research.

Chapter 8

Spectrum Revealing Bounds for Column/Row Selection Based Methods

8.1 Introduction

This work originates from a collaboration with Dave Anderson, Simon S. Du, Kunming Wu, Michael W. Mahoney and Ming Gu [6]. We credit them for their contributions to this work and thank them for allowing us to publish this work in this dissertation. The CUR matrix decomposition approximates an arbitrary data matrix by selecting a subset of columns and a subset of rows to form a low-rank approximation [32, 72]. This method overcomes a fundamental drawback of standard PCA analysis: that the principal components and the loading vectors are dense. Dense components and loadings suffer from two main disadvantages: a loss of sparsity and reduced interpretability. The CUR decomposition is a product of three matrices: two ($\mathbf{C}$ and $\mathbf{R}^T$ with c sampled columns and r sampled rows of $\mathbf{A}$ respectively) are tall and skinny and preserve the sparsity of the data matrix, while the third ($\mathbf{U}$) is a relatively small dense matrix. Thus the CUR approximation is cheaper to work with and to store.

Notable applications of CUR include bioinformatics, document classification, image and video processing, securities trading, and web graphs [15, 72, 74, 89, 96]. The Nyström method is a special case of CUR decomposition for symmetric matrices where $\mathbf{R} = \mathbf{C}^T$ so that the same rows and columns are selected. The Nyström method approximates large kernel matrices that are used for kernel methods, manifold learning, and dimension reduction [31, 96, 95, 98, 103, 104]. In particular, the recent work of [42] introduced an efficient leverage-based random sampling algorithm for Nyström approximation that is analyzed simultaneously for both the spectral and Frobenius norms, while other recent work requires separate algorithms depending on the choice of norm. CUR is also a natural extension of the CX decomposition, which selects either columns or rows, but not both, of the data matrix, and which has been

studied in [17, 48]. The CX decomposition is formed by selecting a subset of columns c to form a tall-skinny matrix C and forming a short fat matrix X to get $A \approx CX$. In general, these works seek to obtain improved *multiplicative error bounds*, which are of the form

$$\|\mathbf{A} - \mathbf{CUR}\|_\xi \leq f\left(m, n, k, c, r\right) \|\mathbf{A} - \mathbf{A}_k\|_\xi,$$

where $\xi \in \{2, F\}$, and where f is a polynomial function and $\mathbf{A}_k$ is an optimal rank-k approximation to a given $\mathbf{A} \in \mathbb{R}^{m \times n}$. When f does not depend on m and n, these bounds are called *constant factor bounds* [71]. Recent works have also established *relative error bounds*, where $f \approx 1 + \epsilon$ for a selection of roughly $O(k/\epsilon)$ rows and columns [18, 32, 42, 71, 96, 95].

Regardless of the form of the guarantee, there are two main drawbacks to the practical use of these existing approaches to column/row selection methods: choosing $\ell \gtrsim O(k/\epsilon)$ columns/rows is often not practical, and thus one typically chooses $\ell = k + O(1)$, i.e., many fewer columns/rows than the sufficient conditions required by the worst-case theory; and, additionally, no known results adapt these methods specifically to matrices with rapidly decaying singular values. Because most data matrices to which CUR decompositions have been applied have decaying singular values, and because a decaying spectrum facilitates better approximations, CUR decompositions would greatly benefit from analysis comparing the quality of the approximation to the rate of spectral decay.

In this thesis, we introduce powerful *spectrum revealing error bounds* that solve these two related problems. This method performs a more refined analysis based on the spectrum of the input data, and it can achieve bounds of the form

$$\|\mathbf{A} - \mathbf{CUR}\|_\xi^2 \leq \left(1 + O\left(\tau^2\right)\right) \|\mathbf{A} - \mathbf{A}_k\|_\xi^2,$$

for $\xi \in \{2, F\}$, where k is the target rank and τ is a quantity that depends on the singular value rate of decay of $\mathbf{A}$ and the amount of oversampling. For matrices with rapidly decaying singular values, and as a function of the amount of oversampling, $\tau \ll 1$. Thus, unlike previous work, our error bounds are near-optimal for matrices with rapidly decaying spectra, and the approximations achieve optimality in the limit as the rate of decay of the spectra increase. (Such a result is a natural requirement for a good approximation method, but none have proved this.) These bounds also help explain why it is acceptable to use a constant $O(1)$ amount of oversampling, i.e., why, given a desired rank k, one can sample $c = k + O(1)$ columns and/or $r = k + O(1)$ rows.

We also show that CUR can be unstable, and we develop a novel algorithm, *StableCUR*, that completely avoids this instability. This algorithm accepts any $\mathbf{C}$ and $\mathbf{R}$ matrices from any row and column selection algorithm, and avoids calculating $\mathbf{U}$, which we show can be ill-conditioned. We apply the column selection algorithm from [7] to determine $\mathbf{C}$ and $\mathbf{R}$, and then we apply our algorithm to compute a CUR decomposition in a stable form. Also, we compare the performance of the combination of these two algorithms to existing randomized CUR algorithms. We also provide a brief empirical illustration of how deterministic and randomized CUR decompositions perform as a function of the oversampling parameter for matrices for which the spectrum decays quickly, as well as when it decays slowly.

8.2 Preliminaries

In this section we review previous results and important theorems to be used in our main results.

8.2.1 The CUR, CX and Nyström Decompositions

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ with rank ρ and a target rank k, we choose a subset of columns $\mathbf{C} \in \mathbb{R}^{m\times c}$, a subset of of rows $\mathbf{R} \in \mathbb{R}^{r\times n}$ and compute a matrix $\mathbf{U} \in \mathbb{R}^{c\times r}$ to form the *CUR decomposition* $\tilde{\mathbf{A}} = \mathbf{CUR}$ that approximates $\mathbf{A}$, where $k < c \ll n$ and $k < r \ll m$. Thus only $\mathbf{C}$, $\mathbf{U}$, and $\mathbf{R}$ need to be stored, which are much smaller than the original matrix $\mathbf{A}$. Additionally, $\mathbf{C}$ and $\mathbf{R}$ retain the sparsity of the original matrix.

We could also take the subset of columns $\mathbf{C} \in \mathbb{R}^{m\times c}$ and compute a short-fat matrix $\mathbf{X} \in \mathbb{R}^{c\times n}$ to form the *CX decomposition* $\tilde{\mathbf{A}} = \mathbf{CX}$ that approximates $\mathbf{A}$, where $k < c \ll n$. The CX decomposition can also be formed using the subset of rows instead of columns. When the input matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ is symmetric, the *Nyström method* is formed by $\tilde{\mathbf{A}} = \mathbf{CUC^T}$ where $C = R^T$ and where $\mathbf{U} \in \mathbb{R}^{c\times c}$ is a matrix chosen to make $\tilde{\mathbf{A}} \approx \mathbf{A}$.

8.2.2 Notation

In this chapter, we adopt slightly different notation than the previous Chapter 7 in order to present results easier. The main difference is the parameter p which serves a similar purpose as before, but now varies between $k \le p \le \ell$. This parameter p can also be an input variable into some subset selection algorithms. We also consider the possibility of a low rank input matrix with rank $\rho \le \min\{m, n\}$ as it improves the guarantees given by some column/row selection methods. As before, we exploit the potential decay in the singular values of $\mathbf{A}$ for better computational efficiency and decomposition reliability. Consider a parameter p such that $k \le p < \min(c, r)$. In the SVD of $\mathbf{A} = \mathbf{U\Sigma V}^T$, we partition $\mathbf{U}$ and $\mathbf{V}$ as

$$\mathbf{U} = \begin{matrix} \\ m \end{matrix} \overset{\begin{matrix} p & \rho-p \end{matrix}}{\begin{pmatrix} \mathbf{U}_1 & \mathbf{U}_2 \end{pmatrix}}, \ \mathbf{V} = \begin{matrix} \\ n \end{matrix} \overset{\begin{matrix} p & \rho-p \end{matrix}}{\begin{pmatrix} \mathbf{V}_1 & \mathbf{V}_2 \end{pmatrix}}. \tag{8.1}$$

Let $\mathbf{\Sigma} = \mathbf{diag}(\sigma_1, \cdots, \sigma_\rho)$, $\sigma_1 \ge \cdots \ge \sigma_\rho > 0$ with

$$\mathbf{\Sigma} = \begin{matrix} p \\ \rho-p \end{matrix} \overset{\begin{matrix} p & \rho-p \end{matrix}}{\begin{pmatrix} \mathbf{\Sigma}_T & \\ & \mathbf{\Sigma}_B \end{pmatrix}}, \ \mathbf{\Sigma}_T = \begin{matrix} k \\ p-k \end{matrix} \overset{\begin{matrix} k & p-k \end{matrix}}{\begin{pmatrix} \mathbf{\Sigma}_1 & \\ & \mathbf{\Sigma}_2 \end{pmatrix}}.$$

We can also use Figure 7.1 to visualize this partition in the singular values.

In equation (8.1), $\mathbf{U}_1$ and $\mathbf{V}_1$ comprise p orthonormal columns spanning the leading p-dimensional row space and column space respectively. The largest k singular values of $\mathbf{A}$ are contained in the diagonal matrix $\mathbf{\Sigma}_1$, which in turn is contained in $\mathbf{\Sigma}_T$; the $(p+1)$-th through the ρ-th singular values of $\mathbf{A}$ are contained in $\mathbf{\Sigma}_B$. The value of p is chosen to

create a "spectrum gap" between the kth and $(p+1)$th singular values of $\mathbf{A}$. To the best of our knowledge, such a partition was first introduced in [47]. Section 8.3 will show that if this gap is large, then the rank-k CUR approximation differs from the best possible rank-k approximation by a negligible amount.

Based on the SVD, the row statistical leverage scores and the row coherence relative to the best rank-p approximation to $\mathbf{A}$ are defined through the p leading left singular vectors in $\mathbf{U_1}$:

$$l_j^r = ||\mathbf{U}_1(j,:)||^2, \quad \mu_r = \frac{m}{p} \times \max_{j\in\{1,\dots m\}} l_i^r \tag{8.2}$$

Similarly, the column statistical leverage scores and the column coherence relative to the best rank-p approximation to $\mathbf{A}$ are defined through the p leading right singular vectors in $\mathbf{V_1}$:

$$l_j^c = ||\mathbf{V}_1(j,:)||^2, \quad \mu_c = \frac{n}{p} \times \max_{j\in\{1,\dots n\}} l_i^c \tag{8.3}$$

The *Moore-Penrose inverse* of $\mathbf{A}$ is denoted by $\mathbf{A}^\dagger = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T$.

Finally, we discuss the time complexities of the matrix operations. For $\mathbf{A} \in \mathbb{R}^{m\times n}$(assume $m > n$) it takes $O(mn^2)$ flops to compute the full SVD and QR decomposition and $O(mnk)$ to compute the truncated SVD of rank-k. Computation of $\mathbf{A}^\dagger$ takes $O(mn^2)$. Leverage scores can be computed in approximately $O(mn\ln n)$[33].

8.2.3 The Sketching Model

Let $\mathbf{\Pi}_r \in \mathbb{R}^{m\times r}$ and $\mathbf{\Pi}_c \in \mathbb{R}^{n\times c}$ be row and column sketching matrices. Examples include sampling matrices that select a subset of columns and rows of $\mathbf{A}$ and Gaussian matrices which produce matrices $\mathbf{C}$ and $\mathbf{R}$ that are Gaussian mixtures of columns and rows of $\mathbf{A}$. Take $\mathbf{C} = \mathbf{A}\mathbf{\Pi}_c$ and $\mathbf{R} = \mathbf{\Pi}_r^T\mathbf{A}$ and $\mathbf{U} = \mathbf{C}^\dagger\mathbf{A}\mathbf{R}^\dagger$. Then the CUR approximation is defined as $\widetilde{\mathbf{A}} = \mathbf{CUR}$, and $\widetilde{\mathbf{A}}_k = (\mathbf{CUR})_k$ is an approximation to $\mathbf{A}$ with rank at most k. Following [42], and for completeness, we formulate our main theoretical result in terms of arbitrary "sketching" matrices.

Note that by equations (8.1)

$$\mathbf{\Psi}_1 := \mathbf{U}_1^T\mathbf{\Pi}_r \quad \text{and} \quad \mathbf{\Psi}_2 := \mathbf{U}_2^T\mathbf{\Pi}_r$$

capture the intersections of the space spanned by the columns of the left sketching matrix with the top and bottom column spaces of $\mathbf{A}$, respectively; and $\mathbf{\Psi_2}\mathbf{\Psi_1}^\dagger$ defines the tangents of the angles between the spaces spanned by $\mathbf{U}_1$ and $\mathbf{\Pi}_r$ [42]. These angles should be sufficiently acute for $\mathbf{\Pi}_r$ to be a good sketch matrix. Similarly,

$$\mathbf{\Omega}_1 := \mathbf{V}_1^T\mathbf{\Pi}_c \quad \text{and} \quad \mathbf{\Omega}_2 := \mathbf{V}_2^T\mathbf{\Pi}_c$$

capture the intersections of the space spanned by the columns of the right sketching matrix with the top and bottom column spaces of $\mathbf{A}^T$, respectively; and $\mathbf{\Omega_2}\mathbf{\Omega_1}^\dagger$ defines the tangents of the angles between the spaces spanned by $\mathbf{V}_1$ and $\mathbf{\Pi}_c$.

When considering the modified Nyström method for positive semi-definite $\mathbf{A}$ instead of the CUR approximation, we will only use $\mathbf{\Pi}_c$ and $\mathbf{\Omega}$, and we set the other side by $\mathbf{R} = \mathbf{C}^T$. This sketching gives the Nyström method the approximation $\mathbf{A} \approx \mathbf{CUC}^T$ where $\mathbf{U} = C^\dagger \mathbf{A} \left(C^T\right)^\dagger$. Theoretically, the CX decomposition is the same as letting $R = I_n$ or $R = A$ in the CUR decomposition to get the desired sketching $\mathbf{X} = \mathbf{C}^\dagger \mathbf{A}$ for the approximation $\mathbf{A} \approx \mathbf{CX}$.

8.2.4 Deterministic Column-Selection

In this section we describe the deterministic Unweighted Column Selection (UCS) algorithm of [7], which will be used in our main results. Applied to a given a matrix $\mathbf{V}^T \in \mathbb{R}^{p \times n}$ with orthonormal rows, this greedy algorithm attempts to choose a subset π of columns to maximize $\sigma_{\min}\left(\mathbf{V}^T(:, \pi)\right)$. The previous column selection algorithm of [17] requires two input matrices and outputs a weighted column selection, for which the weights could be arbitrary. The algorithm of [7] requires a single, relatively small input matrix and outputs an unweighted column selection, while also proving tighter error bounds. The fact that column selection algorithm of [17] requires two matrices to work on makes it less efficient than UCS in complexity and memory use. Consider the matrix $\mathbf{V}_1^T$ in equation (8.1). We refer to the i^{th} column of $\mathbf{V}_1^T$ as $\vec{u}_i \in \mathbb{R}^p$. Then the UCS algorithm is summarized as follows: starting with a p-by-p matrix $B = 0$ and a parameter $T > 0$, the UCS algorithm iteratively selects ℓ columns of $\mathbf{V}_1^T$ by iterating:

- solve for the unique $\lambda < \lambda_{\min}(B)$ such that

$$\mathbf{tr}\left(B - \lambda I\right)^{-1} = T, \tag{8.4}$$

- solve for the unique $\widehat{\lambda} < \lambda_k$ that satisfies

$$\begin{aligned}\left(\widehat{\lambda} - \lambda\right)\left(n - r + \sum_{j=1}^{p} \frac{1 - \lambda_j}{\lambda_j - \lambda}\right) \\ = \frac{\sum \frac{1-\lambda_j}{(\lambda_j - \lambda)(\lambda_j - \widehat{\lambda})}}{\sum \frac{1}{(\lambda_j - \lambda)(\lambda_j - \widehat{\lambda})}},\end{aligned} \tag{8.5}$$

  where λ_j is the j^{th} eigenvalue of B,

- find an index i, not already selected, such that

$$\mathbf{tr}\left(B - \widehat{\lambda} I + \vec{u}_i \vec{u}_i^T\right) \leq \mathbf{tr}\left(B - \lambda I\right)^{-1} \tag{8.6}$$

- reset $B := B + \vec{u}_i \vec{u}_i^T$.

Theorem 30. *An index $i \notin \Pi$ can always be found to satisfy condition (8.6).*

Carried out efficiently, each i can be computed in $O(p^2 n)$ operations. We summarize the above procedure in Algorithm 9. It can be shown that

$$\lambda_{\min}(B_\ell) \geq \frac{(\sqrt{\ell} - \sqrt{p})^2}{(\sqrt{n-p} + \sqrt{\ell})^2 + (\sqrt{\ell} - \sqrt{p})^2}. \tag{8.7}$$

Algorithm 9 Unweighted Column Selection (UCS)

Inputs: Row-orthonormal matrix $\mathbf{V}_1^T \in \mathbb{R}^{p \times n}$, $T \in \mathbb{R}^+$, $\ell, p \in \mathbb{N}$ s.t. $k \leq p < \ell$
Outputs: Index set Π and matrix B.

```
Set B_0 = 0_{p×p}, Π_0 = φ
for t = 0, ··· , ℓ − 1 do
    Solve for λ using equation (8.4)
    Calculate λ̂ using equation (8.5)
    Find i ∉ Π such that inequality (8.6) is satisfied with u_i
    Update B_{t+1} := B_t + u_i u_i^T and Π := Π ∪ {i}.
end for
```

8.3 Theoretical Results

In this section, we present our *StableCUR* algorithm and our *spectrum revealing error bounds*.

8.3.1 The StableCUR Algorithm

Directly computing $\widetilde{\mathbf{A}}$ by multiplying $\mathbf{C}, \mathbf{U}, \mathbf{R}$ together is not numerically stable. Each step of this procedure is numerically stable, and standard libraries exist for both QR and SVD.

In Figure 8.1 we compare the naive procedure and our stable procedure on a synthesized matrix whose i^{th} singular value is 2^{-i}. Note that we are evaluating these methods in the weak-form of the low-rank approximation problem so that $\mathbf{CUR}$ is allowed to have a rank larger than k, i.e. $\frac{\|A-CUR\|}{\|A-A_k\|} \to 0$ as $c, r \to \infty$. The naive computations could lead to inaccurate results because as the number of columns and rows in $\mathbf{C}$ and $\mathbf{R}$ increase, these matrices capture a greater amount of the singular values of $\mathbf{A}$, and so $\mathbf{U} = \mathbf{C}^\dagger \mathbf{A} \mathbf{R}^\dagger$ can be ill-conditioned. Although the algorithm above performs QR on both $\mathbf{C}$ and $\mathbf{R}$, QR for either $\mathbf{C}$ or $\mathbf{R}$ is all that is necessary to make it stable. We define both StableNyström and StableCX in a similar manner.

Algorithm 10 StableCUR

Inputs: $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{R} \in \mathbb{R}^{r\times n}$, $\mathbf{C} \in \mathbb{R}^{m\times c}$, target rank k
Outputs: $\widetilde{\mathbf{A}} \in \mathbb{R}^{m\times n}$ and $\widetilde{\mathbf{A}}_k \in \mathbb{R}^{m\times n}$

1: Do QR factorization on $\mathbf{R}^T$ to obtain a basis of rows of $\mathbf{R}$, $\mathbf{R} = \mathbf{R}_r\mathbf{Q}_r$
2: Do QR factorization on $\mathbf{C}$ to obtain a basis of columns of $\mathbf{C}$, $\mathbf{C} = \mathbf{Q}_c\mathbf{R}_c$
3: $\mathbf{B} = \mathbf{Q}_c^T\mathbf{A}\mathbf{Q}_r^T$
4: $\widetilde{\mathbf{A}} = \mathbf{Q}_c\mathbf{B}\mathbf{Q}_r$
5: Do SVD on $\mathbf{B}$ to Compute $\mathbf{B}_k$.
6: $\widetilde{\mathbf{A}}_\mathbf{k} = \mathbf{Q}_c\mathbf{B}_k\mathbf{Q}_r$

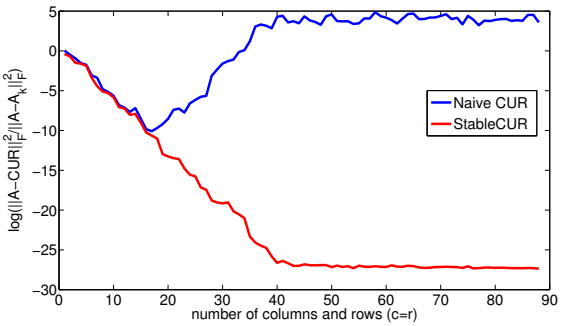


Figure 8.1: Stability comparison of the naive CUR algorithm and our proposed stable sketch algorithm for the weak-form residual error, i.e. **CUR** can have rank $c \geq k$. Also, we use the natural logarithm.

Algorithm 11 StableNyström

Inputs: $\mathbf{A} \in \mathbb{R}^{n\times n}$ symmetric positive semi-definite, $\mathbf{C} \in \mathbb{R}^{n\times c}$, target rank k
Outputs: $\widetilde{\mathbf{A}} \in \mathbb{R}^{n\times n}$ and $\widetilde{\mathbf{A}}_k \in \mathbb{R}^{n\times n}$

1: Do QR factorization on $\mathbf{C}$ to obtain a basis of columns of $\mathbf{C}$, $\mathbf{C} = \mathbf{Q}_c\mathbf{R}_c$
2: $\mathbf{B} = \mathbf{Q}_c^T\mathbf{A}\mathbf{Q}_c$
3: $\widetilde{\mathbf{A}} = \mathbf{Q}_c\mathbf{B}\mathbf{Q}_c^T$
4: Do SVD on $\mathbf{B}$ to Compute $\mathbf{B}_k$.
5: $\widetilde{\mathbf{A}}_\mathbf{k} = \mathbf{Q}_c\mathbf{B}_k\mathbf{Q}_c^T$

8.3.2 Deterministic Structural Results

Here, we introduce theorems about accuracy in the individual singular values and error bounds in the spectral and Frobenius norms for the CUR sketching model. Theorems 31 and 32 below are stated in terms of the following upper bounds:

$$\mathcal{C}_\Omega \geq \left|\left|\mathbf{\Omega_2}\mathbf{\Omega_1^\dagger}\right|\right|_2, \quad \mathcal{C}_\Psi \geq \left|\left|\mathbf{\Psi_2}\mathbf{\Psi_1^\dagger}\right|\right|_2. \tag{8.8}$$

Algorithm 12 StableCX

Inputs: $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{C} \in \mathbb{R}^{m\times c}$, target rank k
Outputs: $\widetilde{\mathbf{A}} \in \mathbb{R}^{m\times n}$ and $\widetilde{\mathbf{A}}_k \in \mathbb{R}^{m\times n}$

1: Do QR factorization on $\mathbf{C}$ to obtain a basis of columns of $\mathbf{C}$, $\mathbf{C} = \mathbf{Q}_c\mathbf{R}_c$
2: $\mathbf{B} = \mathbf{Q}_c^T\mathbf{A}$
3: $\widetilde{\mathbf{A}} = \mathbf{Q}_c\mathbf{B}$
4: Do SVD on $\mathbf{B}$ to Compute $\mathbf{B}_k$.
5: $\widetilde{\mathbf{A}}_\mathbf{k} = \mathbf{Q}_c\mathbf{B}_k$

We start by examining bounds on individual singular values of the low-rank matrix approximations.

Theorem 31 (Singular Value Bounds)**.** *Let $\tau_j = \sigma_{p+1}/\sigma_j$. Then, the output of the CUR Algorithm 10 must satisfy*

$$\sigma_j(\mathbf{CUR}) \geq \frac{\sigma_j\left(1-\tau_j^3\,\mathcal{C}_\Omega\,\mathcal{C}_\Psi\right)}{\sqrt{1+\tau_j^2\mathcal{C}_\Omega^2}\sqrt{1+\tau_j^2\mathcal{C}_\Psi^2}}, \quad \textit{for all } 1\leq j\leq k.$$

Also, the output of the Nyström Algorithm 11 must satisfy

$$\sigma_j(\mathbf{CUC}^T) \geq \frac{\sigma_j}{1+\tau_j^2\mathcal{C}_\Omega^2}, \quad \textit{for all } 1\leq j\leq k.$$

In addition the output of the CX Algorithm 12 must satisfy

$$\sigma_j(\mathbf{CX}) \geq \frac{\sigma_j}{\sqrt{1+\tau_j^2\mathcal{C}_\Omega^2}}, \quad \textit{for all } 1\leq j\leq k.$$

Proof. For the CX Decomposition, the lower bound is a simple application of Theorem 21 from Chapter 7. The CUR and Nyström decomposition lower bounds result from an application of Theorem 37. □

Next, we present error bounds in the spectral and Frobenius norms. Remember that $\|\Sigma_1\|_F^2 = \sum_{j=1}^k \sigma_j^2 \leq k\sigma_1^2 = k\,\|\Sigma_1\|_2^2$ and $\|\Sigma_3\|_F^2 = \sum_{j=1}^k \sigma_{p+j}^2 \leq k\sigma_{p+1}^2 = k\,\|\Sigma_3\|_2^2$

Theorem 32 (Residual Error Bounds)**.** *Assume the notation setup above and let $\xi = 2$ or F. The CUR Algorithm 10 must satisfy*

$$\|\mathbf{A}-(\mathbf{CUR})_k\|_\xi^2 \leq \|\mathbf{A}-\mathbf{A}_k\|_\xi^2 + \left(\frac{\mathcal{C}_\Omega}{\sqrt{1+\frac{\|\mathbf{\Sigma}_3\|_F^2}{\|\mathbf{\Sigma}_1\|_F^2}\mathcal{C}_\Omega^2}} + \frac{\mathcal{C}_\Psi}{\sqrt{1+\frac{\|\mathbf{\Sigma}_3\|_F^2}{\|\mathbf{\Sigma}_1\|_F^2}\mathcal{C}_\Psi^2}}\right)^2 \|\mathbf{\Sigma}_3\|_F^2$$

Also, the Nyström Method in Algorithm 11 must satisfy

$$\left\|\mathbf{A} - (\mathbf{CUC^T})_k\right\|_\xi^2 \leq \|\mathbf{A} - \mathbf{A}_k\|_\xi^2 + 4\left(\frac{\mathcal{C}_\Omega^2}{1 + \frac{\|\mathbf{\Sigma}_3\|_F^2}{\|\mathbf{\Sigma}_1\|_F^2}\mathcal{C}_\Omega^2}\right)\|\mathbf{\Sigma}_3\|_F^2$$

In addition, the CX Algorithm 12 must satisfy

$$\|\mathbf{A} - (\mathbf{CX})_k\|_\xi^2 \leq \|\mathbf{A} - \mathbf{A}_k\|_\xi^2 + \left(\frac{\mathcal{C}_\Omega^2}{1 + \frac{\|\mathbf{\Sigma}_3\|_F^2}{\|\mathbf{\Sigma}_1\|_F^2}\mathcal{C}_\Omega^2}\right)\|\mathbf{\Sigma}_3\|_F^2$$

Proof. We start with the structural result of Theorem 8.5.1 for the CUR and Nyström Decompositions. We then use the work of Theorem 28 and Corollary 5 in order to bound $\left\|\left(I - Q_c Q_c^T\right) A_k\right\|_F$ and $\left\|A_k\left(I - Q_r Q_r^T\right)\right\|_F$ in the desired way with $q = 0$ as column/row selection based methods do not have the iteration feature of randomized subspace iteration. The result for the CX decomposition is a straightforward application of Theorem 28 with Q_c and $q = 0$. □

Discussion

A good CUR decomposition heavily depends on how the sketch matrices are chosen; Theorems 31 and 32 point out the connection between sketch matrices and the quality of the CUR decomposition through quantities $\mathcal{C}_\Omega$ and $\mathcal{C}_\Psi$.

Theorems 31 and 32 also exhibit a surprisingly strong connection between the rate at which the singular values of matrix $\mathbf{A}$ might decay and the quality of the CUR decomposition. For the sake of argument assume for the moment that $\mathcal{C}_\Omega = O(1)$ and $\mathcal{C}_\Psi = O(1)$. When singular values of $\mathbf{A}$ decay rapidly, as they often do in many large data matrices, we can expect $\tau_j \ll 1$ for a choice of p that is somewhat larger than k. Theorem 31 suggests that the leading singular values of $\widetilde{\mathbf{A}}$, $\sigma_j(\widetilde{\mathbf{A}})$ for $1 \leq j \leq k$, differ from the corresponding singular values of $\mathbf{A}$ by a negligible relative amount. Similarly, since

$$\left(\sum_{j=k+1}^{\rho} \sigma_j^2\right) \geq \sigma_{k+1}^2 \gg \sigma_{p+1}^2$$

when singular values rapidly decay, Theorems 1 and 32 suggest that the approximation error in $\widetilde{\mathbf{A}}$ differs from that in $\mathbf{A}_k$, the best rank-k approximation, by a negligible additional amount in both the Frobenius norm and spectral norm.

In the remainder of this section, we show that the UCS algorithm from [7] and two sampling algorithms are able to bring both $\mathcal{C}_\Omega$ and $\mathcal{C}_\Psi$ under effective control in their magnitude, leading to high quality CUR decompositions. It is important to note that when using the modified Nyström method, the above bounds still hold with $\mathcal{C}_\Psi := \mathcal{C}_\Omega$.

8.3.3 Bounds of the Deterministic Unweighted Column Selection

We apply Theorems 31 and 32 to bound the singular value errors and the low-rank approximation error in the spectral and Frobenius norms for the matrix constructed by Algorithm 9.

Theorem 33. (Unweighted Column Selection)
Let $\mathbf{\Pi}_r$ and $\mathbf{\Pi}_c$ be constructed with Algorithm 9, then Theorems 31 and 32 hold with

$$\begin{aligned} \mathcal{C}_\Omega^{-1} &= \frac{\sqrt{c}-\sqrt{p}}{\sqrt{(\sqrt{n-p}+\sqrt{c})^2+(\sqrt{c}-\sqrt{p})^2}}, \\ \mathcal{C}_\Psi^{-1} &= \frac{\sqrt{r}-\sqrt{p}}{\sqrt{(\sqrt{m-p}+\sqrt{r})^2+(\sqrt{r}-\sqrt{p})^2}}. \end{aligned}$$

When applying the result above to the Nyström method (i.e. $\mathbf{R} := \mathbf{C}^T$), one simply needs to ignore the discussion of sampling rows. Simple algebra reveals as c and r increase, $\mathcal{C}_\Omega$ and $\mathcal{C}_\Psi$ will decrease as well. This suggests a tradeoff between controlling the $\mathcal{C}$ terms and improving the spectral gap τ_{k+1}.

8.3.4 Stochastic Bounds of Sampling Based Algorithms

We apply Theorems 31 and 32 to bound errors in the random sampling methods. μ_r and μ_c in Theorem 34 refer to the row coherence in equation (8.2) and column coherence in equation (8.3). The failure probabilities below are squared for the CUR because the rows and columns are sampled independently. When applying the two theorems below to the Nyström method (i.e. $\mathbf{R} := \mathbf{C}^T$), one needs to ignore the discussion of sampling rows and to take the square root of the failure probability by the point above.

Theorem 34. (Uniform Sampling) [41]. *Let $\mathbf{\Pi}_r \in \mathbb{R}^{r\times m}, \mathbf{\Pi}_c \in \mathbb{R}^{n\times c}$ be sketching matrices corresponding to sampling rows and columns uniformly at random, respectively. Fix a failure probability $0 < \delta \ll 1$ and an accuracy factor $\epsilon \in (0,1)$. If*

$$r \geq 2\epsilon^{-2}\mu_r p \ln(p/\delta), \quad c \geq 2\epsilon^{-2}\mu_c p \ln(p/\delta),$$

then Theorems 31 and 32 hold with

$$\mathcal{C}_\Omega = \sqrt{\frac{n}{(1-\epsilon)\,c}}, \quad \mathcal{C}_\Psi = \sqrt{\frac{m}{(1-\epsilon)\,r}}$$

with probability at least $(1-\delta)^2$.

Theorem 35. (Leverage Score Sampling) [32] *Let* $\mathbf{\Pi}_r \in \mathbb{R}^{r\times m}, \mathbf{\Pi}_c \in \mathbb{R}^{n\times c}$ *be generated with probability distributions based on the row leverage scores* $\{l_j^r\}$ *in equation (8.2) and column leverage scores* $\{l_j^c\}$ *in equation (8.3):*

$$p_{rj} = \frac{l_j^r}{p} \quad and \quad p_{cj} = \frac{l_j^r}{p}$$

for an accuracy factor $\epsilon \in (0,1)$*. If*

$$r \geq 400\epsilon^{-2} p \ln(p), \quad c \geq 400\epsilon^{-2} p \ln(p),$$

then Theorems 31 and 32 hold with

$$\mathcal{C}_\Omega = \sqrt{\frac{1}{1-\epsilon}}, \quad \mathcal{C}_\Psi = \sqrt{\frac{1}{1-\epsilon}}$$

with probability at least $0.9^2 = 0.81$*.*

8.4 Numerical Results

In this section, we provide a summary of our empirical evaluation. We start in Section 8.4.1 with a description of our data sets and our evaluation metrics; then, in Section 8.4.2, we show how oversampling affects reconstruction error for deterministic and randomized CUR/Nyström on two data sets with different spectrum properties; and then, in Section 8.4.3, we compare our *Stable* algorithms using input matrices determined by the deterministic UCS algorithm with other related decompositions.

8.4.1 Data Sets

We used data sets from the recent analysis of [42]. The data sets include matrices constructed from bag-of-words data (Dexter) and dense matrices constructed from a Gaussian Radial Basis Function (RBF) Kernel (Abalone). The description of data sets is presented in Table 8.1. Here, m and n are numbers of columns and rows of the data matrix, %nnz is the percentage of number of non-zero entries, k is the target rank, and μ_c and μ_r are the coherence of the rows and columns of $\mathbf{A}$ respectively. Recall that, for a set of data points $\mathbf{x_1}, \ldots, \mathbf{x_n} \in \mathbb{R}^d$, the Gaussian RBF Kernel matrix $\mathbf{A}^\sigma$ is given by $\mathbf{A}^\sigma_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{\sigma^2}\right)$.

These data matrices are chosen because of their different spectral decay properties. In particular, by adjusting the σ in the Gaussian RBF Kernel, we can change the speed of the decay in a controlled manner. Observe that $\frac{\sigma_{2k}(\mathbf{A})}{\sigma_k(\mathbf{A})}$ increases from 0.156 to 0.801 as σ is decreased from 5 to 0.1. In more detail, Figure 8.2 shows that for Abalone kernel matrix, decreasing values of σ from 5 to 0.1 slows down the singular value decay, reducing the domination by the top-k eigenspace; and Table 8.1 shows that when $\sigma = 0.1$, best rank-20

Table 8.1: Dataset Summary.

Data Set	m	n	%nnz	k	$\frac{\|\mathbf{A}\|_F^2}{\|\mathbf{A}\|_2^2}$	$100\ \frac{\|\mathbf{A}-\mathbf{A_k}\|_F}{\|\mathbf{A}\|_F}$	μ_c	μ_r	$\frac{\sigma_{2k}(\mathbf{A})}{\sigma_k(\mathbf{A})}$
Abalone($\sigma = 5$)	4177	4177	100	20	1.09	0.17	10.6	10.6	0.156
Abalone($\sigma = 2$)	4177	4177	100	20	1.88	4.39	2.67	2.67	0.285
Abalone($\sigma = 0.2$)	4177	4177	84.6	20	14.9	79.6	17.6	17.6	0.62
Abalone($\sigma = 0.1$)	4177	4177	40.74	20	174.7	97.47	59.9	59.9	0.801
Dexter	2000	20000	0.48	10	7.16	88.6	197.2	1945	0.806

approximation is far from the original matrix (and thus low-rank approximation cannot be expected to yield good results), while for $\sigma = 5$, the matrix is very well approximated by a rank-20 matrix.

Since the RBF Kernel matrices are postive semi-definite, we apply StableNyström to the Abalone data in kernel form. Next, we apply the StabelCUR algorithm to the Dexter data as the data matrix takes the form of a general non-symmetric matrix. In our empirical evaluation, we consider the following measures to compare different approximation algorithms:

- $\sigma_k(\widetilde{\mathbf{A}})/\sigma_k(\mathbf{A})$, k^{th} singular value ratio
- $\|\mathbf{A} - \widetilde{\mathbf{A}}\|_F / \|\mathbf{A} - \mathbf{A}_k\|_F$, weak-form Frobenius norm error
- $\|\mathbf{A} - \widetilde{\mathbf{A}}_k\|_F / \|\mathbf{A} - \mathbf{A}_k\|_F$, strong-form rank-$k$ Frobenius norm error
- $\|\mathbf{A} - \widetilde{\mathbf{A}}\|_2 / \|\mathbf{A} - \mathbf{A}_k\|_2$, weak-form spectrum norm error
- $\|\mathbf{A} - \widetilde{\mathbf{A}}_k\|_2 / \|\mathbf{A} - \mathbf{A}_k\|_2$, strong-form rank-$k$ spectrum norm error

In addition, the legends in the following plots correspond to the four CUR algorithms we consider:

- RandLeverage: CUR/Nyström Decomposition of [32] constructed from Leverage Score Sampling
- RandUniform: CUR/Nyström Decomposition constructed from Uniform Sampling
- NearOptimal: CUR/Nyström via the Near-Optimal Column Selection Algorithm of [17, 96]
- DetUCS: Deterministic Unweighted Column Selection (UCS) algorithm of [7].

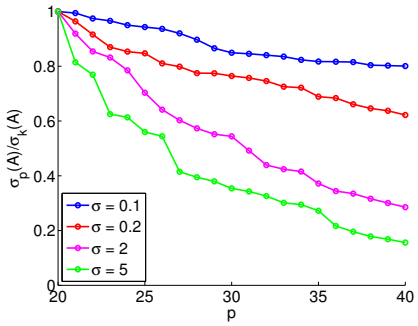


Figure 8.2: Singular value decay of Abalone kernel matrices with different σ's. The reported value is the ratio between σ_p and σ_k, where $k = 20$ and p varies from 20 to 40.

8.4.2 Oversampling Experiments

Here, we test how the spectrum gap can affect the performance of DetUCS and RandLeverage. Our main results are presented in Figures 8.3 and 8.4. We choose target rank $k = 20$, $c = r = 80$ and vary p from k to $2k$. Recall from our deterministic structural results from Section 8.3 that increasing p will decrease $\sigma_{p+1}(\mathbf{A})$ and thus improve the approximation accuracy. However, in Theorem 33 and 35, we showed that increasing p may increase $\mathcal{C}_\Omega$ and $\mathcal{C}_\Psi$. By our bounds, for matrices whose singular values decay rapidly, increase in p could be beneficial.

Figure 8.3 shows the effects of different values of p on the Frobenius norm reconstruction error. For Abalone kernel matrix with $\sigma = 5$, both DetUCS and RandLeverage behave better as p increases. On the other hand, when $\sigma = 0.1$, the reconstruction error is much larger and there is no systematic evident performance improvement as p increases.

Figure 8.4 shows the effects of different values of p on the spectral norm reconstruction error. These plots are qualitatively similar to the Frobenius norm error: for Abalone kernel matrix with $\sigma = 5$, increase in p reduces reconstruction error for both algorithms; while, when $\sigma = 0.1$, increase in p would not improve DetUCS. Interestingly, however, for RandLeverage, increase in p may even decrease the reconstruction accuracy. The reason for this is likely that we do not have control on $\mathcal{C}_\Omega$ and $\mathcal{C}_\Psi$ as we increase p.

8.4.3 Comparing Different CUR/Nyström Methods

We now compare the performance of different CUR/Nystrom algorithms (RandLeverage, RandUniform, NearOptimal, and DetUCS) with same number of columns and rows. To take advantage of the spectrum gap, we choose oversampling parameter $p = k + 10$ for matrix (Figure 8.5) with rapid singular value decay. While for matrices (Figures 8.6 and 8.7) with slow singular value decay, there is no need to oversample and to decrease $\mathcal{C}_\Omega$ and $\mathcal{C}_\Psi$ we choose $p = k$.

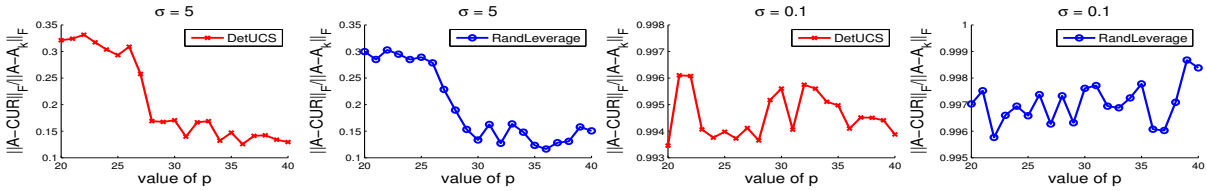


Figure 8.3: Reconstruction error in Frobenius norm for DETUCS and RANDLEVERAGE running on Abalone kernel matrix with $\sigma = 5$ and 0.1. When $\sigma = 5$, both algorithms perform better as we increase p. When $\sigma = 0.1$, the reconstruction errors are less consistent.

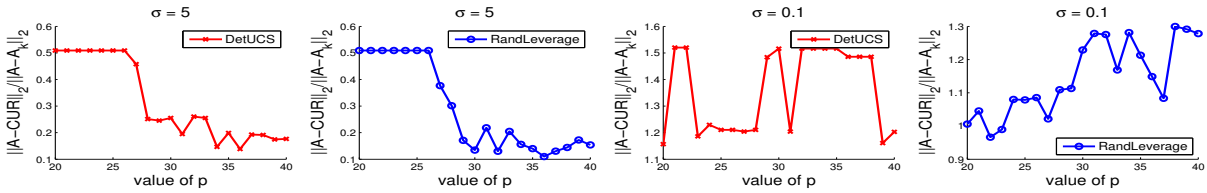


Figure 8.4: Reconstruction error in spectral norm for DETUCS and RANDLEVERAGE running on Abalone kernel matrix with $\sigma = 5$ and 0.1. The results are very similar to figure 8.4.

Figure 8.5 shows the performance of different Nyström algorithms on Abalone matrix with $\sigma = 5$, whose singular values decay rapidly. Since $\mathbf{A}_k$ contains most of the information, low rank approximation is a reasonable model. The reconstruction matrix is able to capture most singular values and the residual errors in both spectral and Frobenius norm decrease rapidly as more columns and rows are sampled. Since the leverage scores are fairly uniform, i.e., the coherence is fairly small, RANDUNIFORM performs well in this case, even though it is still worse than other algorithms.

Figure 8.6 shows the performance of different Nyström algorithms on Abalone matrix with $\sigma = 0.1$, whose singular values decay slowly. Since $\mathbf{A}_k$ only contains a small portion of information of $\mathbf{A}$, the curves are flatter in this case. Since the coherence is large, RANDUNIFORM performs poorly and RANDLEVERAGE performs best under most metrics. However, sampling with more columns and rows only increases approximation accuracy marginally, because the leverage score distribution is extremely imbalanced due to the high coherence of the matrix.

Figure 8.7 shows the performance of different CUR algorithms on non-symmetric Dexter data matrix. This data set is "worse" than Abalone kernel matrix with $\sigma = 5$, because of its slow decay in singular values and large coherence, and our empirical results are consistent with this.

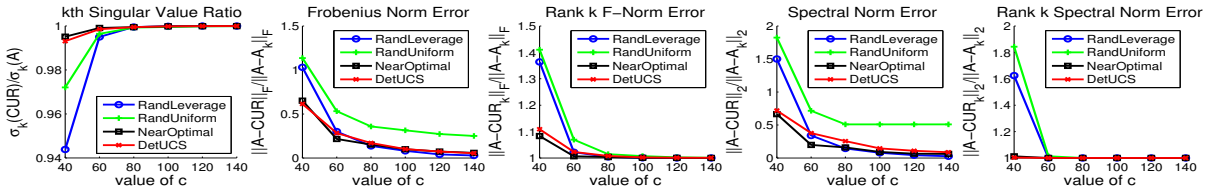


Figure 8.5: Results of algorithms comparison on RBF kernel($\sigma = 5$) of the Abalone data set. In this matrix, singular values decay very fast, which results in rapid decrease in residual errors and rapid increase in singular value ratio for all algorithms.

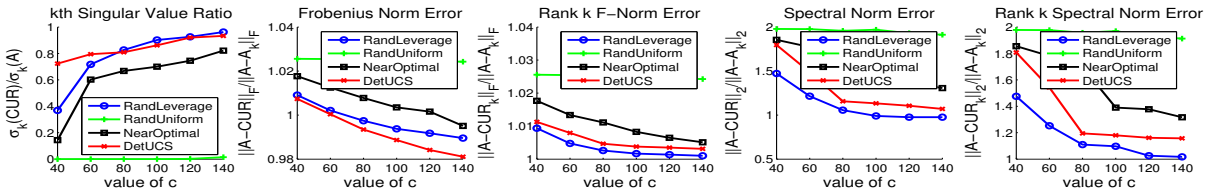


Figure 8.6: Results of algorithms comparison on RBF kernel($\sigma = 0.1$) of the Abalone data set. In this matrix, singular values decay very slowly. All curves are flatter than the ones in Figure 8.5.

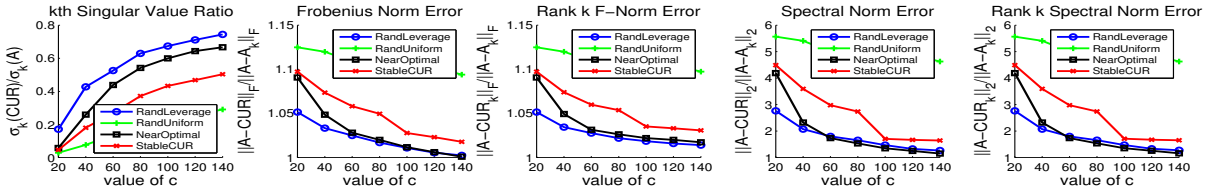


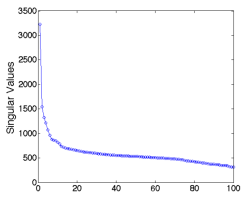
Figure 8.7: Results of CUR algorithms comparison on Dexter data matrix. This is a non-symmetric matrix with slow decay in its singular values. The performance of algorithms are similar to the ones in Figure 8.6.

8.4.4 Experiments with CX Algorithm

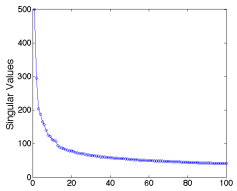
We test our algorithms on the Jester Joke Data [43], a data matrix containing numeric ratings from 24,983 people for 100 jokes. Each row corresponds to a person; each column a joke. We have removed its first column, which represents the number of jokes rated, and we have changed any NA value, indicated by 99, to 0 (meaning neutral) in order to make it consistent with other entries. Therefore, the size of the matrix is 24,983 by 100, and the entries are ratings ranging from -10.00 to 10.00. Positive ratings indicate the joke is favored, while negative ratings indicate the opposite. Additionally we test on a second data matrix, which comes from the Reuters bag of words data [21]. The matrix is modified into a sparse matrix of size 8,293 by 21,578 with binary entries, with "1" meaning the word is present in the document and "0" meaning the word is absent. For each target sparsity value, we let

the number of selected columns, ℓ, range from $k+1$ to $k+20$. In these experiments, we let $p = k$ and $s = \ell$.

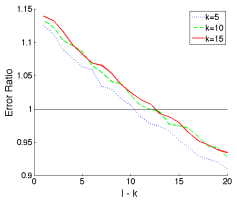
Our tests exhibit rapid decay of the error ratio. Figure 8.9 shows heat maps of the



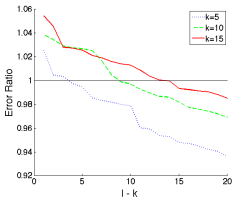
(a) Jester data spectrum



(b) Reuters data spectrum



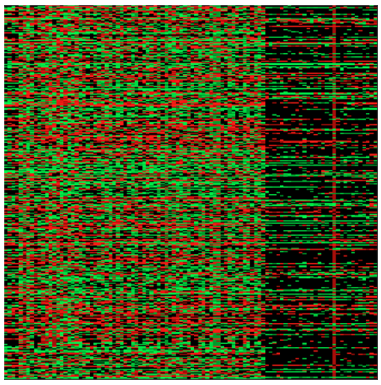
(c) Jester data CX residual error



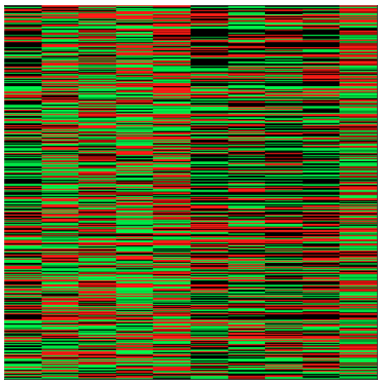
(d) Reuters data CX residual error

Figure 8.8: Real Data Matrices

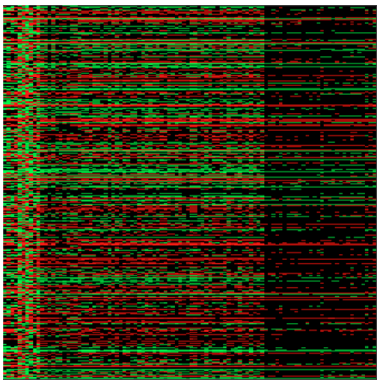
Jester jokes data matrix A, and the columns subset C with $l = 10$, and the reconstruction $\tilde{A} = CX$. Green corresponds to a positive rating, while red a negative rating. The map in 8.9(a) shows the original data matrix, 8.9(b) shows the columns selected from this data, and 8.9(c) shows the reconstruction $\tilde{A} = CX$. While the reconstruction loses some data as expected, it preserves the structure and the pattern of the original matrix A. Also, the heat maps suggest that the algorithm chose mutually independent and informative columns.



(a) Original Data Matrix



(b) Selected Columns Subset



(c) Reconstruction

Figure 8.9: Heat Maps of Jester Joke Data Matrix

8.5 Proofs

8.5.1 Preliminaries

First we prove a useful theorem, which is similar to [47] Theorem 3.5.

Theorem 36. *Let Q_c be an $m \times c$ column-orthonormal matrix matrix. Let Q_r be a $n \times r$ column-orthonormal matrix. Let B_k be the rank-k truncated SVD of $Q_c^T A Q_r$. We have:*

$$\min_{rank(B)\leq k, B\in\mathbb{R}^{c\times r}} \left\|A - Q_c B Q_r^T\right\|_F^2 = \left\|A - Q_c B_k Q_r^T\right\|_F^2 \tag{8.9}$$

In addition:

$$\left\|A - Q_c B_k Q_r^T\right\|_F^2 \leq \|A - A_k\|_F^2 + \left(\left\|\left(I - Q_c Q_c^T\right) A_k\right\|_F + \left\|A_k \left(I - Q_r Q_r^T\right)\right\|_F\right)^2 \tag{8.10}$$

Proof. We start by taking column-orthogonal matrices of dimensions $m\times(m-c)$ and $n\times(n-r)$ labeled $\hat{Q}_c$ and $\hat{Q}_r$, respectively, so that $\begin{pmatrix} Q_c & \hat{Q}_c \end{pmatrix}$ and $\begin{pmatrix} Q_r & \hat{Q}_r \end{pmatrix}$ are both orthogonal matrices. Then, the unitary invariance of the Frobenious norm and orthogonality give

$$\begin{aligned}
\left\|A - Q_c B Q_r^T\right\|_F^2 &= \left\| \begin{pmatrix} Q_c^T A Q_r - B & Q_c^T A \hat{Q}_r \\ \hat{Q}_c^T A Q_r & \hat{Q}_c^T A \hat{Q}_r \end{pmatrix} \right\|_F^2 \\
&= \left\|A - Q_c \left(Q_c^T A Q_r\right) Q_r^T\right\|_F^2 + \left\|Q_c^T A Q_r - B\right\|_F^2
\end{aligned}$$

Thus, the last term in the expression above is minimized when $B = B_k$, which gives us (8.9). Since B_k is the minimizer, we can replace it with $Q_c^T A_k Q_r$ to get the inequality

$$\begin{aligned}
\|A - Q_c B_k Q_r^T\|_F^2 &\leq \|A - Q_c \left(Q_c^T A_k Q_r\right) Q_r^T\|_F^2 \\
&= \|A - Q_c Q_c^T A_k + Q_c Q_c^T A_k - Q_c \left(Q_c^T A_k Q_r\right) Q_r^T\|_F^2 \\
&= \|A - A_k + A_k - Q_c Q_c^T A_k\|_F^2 + \|Q_c Q_c^T \left(A_k - A_k Q_r Q_r^T\right)\|_F^2 \\
&\quad + 2\mathbf{tr}\left((A - A_k)^T Q_c Q_c^T A_k \left(I - Q_r Q_r^T\right)\right) \\
&= \|A - A_k\|_F^2 + 2\mathbf{tr}\left((A - A_k) A_k^T (I - Q_c Q_c^T)\right) \\
&\quad + \left\|\left(I - Q_c Q_c^T\right) A_k\right\|_F^2 + \|Q_c Q_c^T \left(A_k - A_k Q_r Q_r^T\right)\|_F^2 \\
&\quad + 2\mathbf{tr}\left((A - A_k)^T Q_c Q_c^T A_k \left(I - Q_r Q_r^T\right)\right) \\
&\leq \|A - A_k\|_F^2 + \|\left(I - Q_c Q_c^T\right) A_k\|_F^2 + \|A_k \left(I - Q_r Q_r^T\right)\|_F^2 \\
&\quad - 2\mathbf{tr}\left((A - A_k)^T \left(I - Q_c Q_c^T\right) A_k \left(I - Q_r Q_r^T\right)\right),
\end{aligned}$$

which is (8.10). In the last inequality, we have used once the fact that $Q_c Q_c^T$ is an orthogonal projection and twice the fact that $(A - A_k)A_k^T = 0$ via the SVD. Now, use the identity

$A_k A_k^\dagger A_k = A_k$ from the definition of the Moore-Penrose psuedo-inverse to get

$$\begin{aligned}
&\mathbf{tr}\Big(\big(I-Q_cQ_c^T\big)A_k\big(I-Q_rQ_r^T\big)(A-A_k)^T\Big) = \mathbf{tr}\Big(\big(I-Q_cQ_c^T\big)A_kA_k^\dagger A_k\big(I-Q_rQ_r^T\big)(A-A_k)^T\Big)\\
&\leq \left\|\left(I-Q_cQ_c^T\right)A_k\right\|_F \left\|A_k^\dagger A_k\left(I-Q_rQ_r^T\right)(A-A_k)^T\right\|_F\\
&\leq \left\|\left(I-Q_cQ_c^T\right)A_k\right\|_F \left\|A_k\left(I-Q_rQ_r^T\right)\right\|_F \left\|A_k^\dagger\right\|_2 \|A-A_k\|_2\\
&= \left\|\left(I-Q_cQ_c^T\right)A_k\right\|_F \left\|A_k\left(I-Q_rQ_r^T\right)\right\|_F \frac{\sigma_{k+1}}{\sigma_k}\\
&\leq \left\|\left(I-Q_cQ_c^T\right)A_k\right\|_F \left\|A_k\left(I-Q_rQ_r^T\right)\right\|_F
\end{aligned}$$

Putting everything together, we arrive at our desired result. □

8.5.2 Deterministic Analysis

We begin with some notes about partitioning A by columns and rows. Let $\Pi_c \in \mathbb{R}^{n\times c}$ and $\Pi_r \in \mathbb{R}^{m\times r}$ be matrices that represent the column and row choices, respectively, of our algorithm such that $\begin{pmatrix}\Pi_c & \Pi_c^\perp\end{pmatrix} \in \mathbb{R}^{n\times n}$ and $\begin{pmatrix}\Pi_r & \Pi_r^\perp\end{pmatrix} \in \mathbb{R}^{m\times m}$ are a permutation matrices.

$$\begin{aligned}
\begin{pmatrix}\Pi_r & \Pi_r^\perp\end{pmatrix}^T A \begin{pmatrix}\Pi_c & \Pi_c^\perp\end{pmatrix} &= \begin{pmatrix}\Pi_r & \Pi_r^\perp\end{pmatrix}^T U\Sigma V^T \begin{pmatrix}\Pi_c & \Pi_c^\perp\end{pmatrix}\\
&= \begin{pmatrix}U_{11} & U_{12}\\ U_{21} & U_{22}\end{pmatrix}\begin{pmatrix}\begin{pmatrix}\Sigma_1 & 0\\ 0 & \Sigma_2\end{pmatrix} & 0\\ 0 & \Sigma_B\end{pmatrix}\begin{pmatrix}V_{11}^T & V_{21}^T\\ V_{12}^T & V_{22}^T\end{pmatrix}
\end{aligned}$$

From this point on, we refer to

$$\Omega = \begin{pmatrix}\Omega_1\\ \Omega_2\end{pmatrix} \stackrel{def}{=} \begin{pmatrix}V_{11}^T\\ V_{12}^T\end{pmatrix}$$
$$\Psi = \begin{pmatrix}\Psi_1\\ \Psi_2\end{pmatrix} \stackrel{def}{=} \begin{pmatrix}U_{11}^T\\ U_{12}^T\end{pmatrix}$$

We change notation at this point because these principles go far beyond column and row selection. For example, if either Π_c or Π_r were an iid Gaussian matrix, the following results will still hold.

By definition, the matrix $C \in \mathbb{R}^{m\times c}$ produced by our algorithm is $A\Pi_c$.

$$C = A\Pi_c = U\begin{pmatrix}\begin{pmatrix}\Sigma_1 & 0\\ 0 & \Sigma_2\end{pmatrix} & 0\\ 0 & \Sigma_B\end{pmatrix}\begin{pmatrix}\Omega_1\\ \Omega_2\end{pmatrix}$$

Now, we are interested in the matrix $CX = CC^\dagger A$ In order to get a grip on the orthogonal projector $CC^\dagger$, we will study the column space of C via post-multiplying by a judiciously

chosen square invertible matrix $Y_c \in \mathbb{R}^{c\times c}$ (cf. [47].) This may change the matrix, but it preserves the column space.

$$\begin{aligned} CY_c &:= C \left[\begin{array}{c|c} \Omega_1^\dagger \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix}^{-1} & Z_c \end{array} \right] \\ &= U \begin{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix} & 0 \\ 0 & \Sigma_B \end{pmatrix} \begin{pmatrix} \Omega_1 \\ \Omega_2 \end{pmatrix} \left[\begin{array}{c|c} (\Omega_1)^\dagger \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix}^{-1} & Z_c \end{array} \right] \\ &= U \begin{pmatrix} I_k & 0 & 0 \\ 0 & I_{p-k} & 0 \\ H_1 & H_2 & H_3 \end{pmatrix} \end{aligned}$$

where we assume that $\Omega_1 \in \mathbb{R}^{c\times p}$ is full rank and $Z_c \in \mathbb{R}^{c\times(c-p)}$ is a matrix such that $\Omega_1 Z_c = 0$. This gives us that

$$H_1 = \Sigma_B \Omega_2 \Omega_1^\dagger \begin{pmatrix} \Sigma_1^{-1} \\ 0 \end{pmatrix},\ H_2 = \Sigma_B \Omega_2 \Omega_1^\dagger \begin{pmatrix} 0 \\ \Sigma_2^{-1} \end{pmatrix},\ H_3 = \Sigma_B \Omega_2 Z_c$$

By the same procedure we can select rows from A to form $R = \Pi_r^T A$. As before, there is an invertible matrix $Y_r \in \mathbb{R}^{r\times r}$ such that

$$Y_r R = \begin{pmatrix} I_k & 0 & 0 \\ 0 & I_{p-k} & 0 \\ G_1 & G_2 & G_3 \end{pmatrix}^T V^T$$

where

$$G_1 = \Sigma_B \Psi_2 \Psi_1^\dagger \begin{pmatrix} \Sigma_1^{-1} \\ 0 \end{pmatrix},\ G_2 = \Sigma_B \Psi_2 \Psi_1^\dagger \begin{pmatrix} 0 \\ \Sigma_2^{-1} \end{pmatrix},\ G_3 = \Sigma_B \Psi_2 Z_r$$

Following [47], we are interested in upper bounds on $||H_1||_2$ and $||G_1||_2$.

$$||H_1||_2 \leq \frac{\sigma_{p+1}}{\sigma_k} \left|\left|\Omega_2 \Omega_1^\dagger\right|\right|_2, \text{ and } \left|\left|\left(I + H_1^T H_1\right)^{-1/2}\right|\right|_2 \geq \frac{1}{\sqrt{1 + \left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2 \left|\left|\Omega_2 \Omega_1^\dagger\right|\right|_2^2}}$$

$$||G_1||_2 \leq \frac{\sigma_{p+1}}{\sigma_k} \left|\left|\Psi_2 \Psi_1^\dagger\right|\right|_2, \text{ and } \left|\left|\left(I + G_1^T G_1\right)^{-1/2}\right|\right|_2 \geq \frac{1}{\sqrt{1 + \left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2 \left|\left|\Psi_2 \Psi_1^\dagger\right|\right|_2^2}}.$$

To develop lower bounds on computed singular values, let

$$U \begin{pmatrix} I_k & 0 & 0 \\ 0 & I_{p-k} & 0 \\ H_1 & H_2 & H_3 \end{pmatrix} =: \widehat{Q}\widehat{R} =: \begin{pmatrix} \widehat{Q}_1 & \widehat{Q}_2 & \widehat{Q}_3 \end{pmatrix} \begin{pmatrix} \widehat{R}_{11} & \widehat{R}_{12} & \widehat{R}_{13} \\ 0 & \widehat{R}_{22} & \widehat{R}_{23} \\ 0 & 0 & \widehat{R}_{33} \end{pmatrix}, \tag{8.11}$$

$$V \begin{pmatrix} I_k & 0 & 0 \\ 0 & I_{p-k} & 0 \\ G_1 & G_2 & G_3 \end{pmatrix} =: \widetilde{Q}\widetilde{R} =: \begin{pmatrix} \widetilde{Q}_1 & \widetilde{Q}_2 & \widetilde{Q}_3 \end{pmatrix} \begin{pmatrix} \widetilde{R}_{11} & \widetilde{R}_{12} & \widetilde{R}_{13} \\ 0 & \widetilde{R}_{22} & \widetilde{R}_{23} \\ 0 & 0 & \widetilde{R}_{33} \end{pmatrix}. \tag{8.12}$$

It follows from (cf. [47]) that

$$Q_c Q_c^T = \widehat{Q}\widehat{Q}^T,\ Q_r Q_r^T = \widetilde{Q}\widetilde{Q}^T.$$

Consider the first k columns of the above expression, i.e.

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

Since $R_{11}^T R_{11} = I + H_1^T H_1$, the polar decomposition tells us that R_{11} can be written in the form

$$\widehat{R}_{11} = W_c \left(I + H_1^T H_1\right)^{1/2}$$

for some orthogonal matrix $W_c \in \mathbb{R}^{k\times k}$. Thus, we can write

$$\widehat{Q}_1 = U \begin{pmatrix} I \\ 0 \\ H_1 \end{pmatrix} \left(I + H_1^T H_1\right)^{-1/2} W_c^T$$

By the same reasoning, we also have

$$\widetilde{Q}_1 = V \begin{pmatrix} I \\ 0 \\ G_1 \end{pmatrix} \left(I + G_1^T G_1\right)^{-1/2} W_r^T$$

for some orthogonal matrix $W_r \in \mathbb{R}^{k\times k}$.

Theorem 37 (Two-Sided Singular Value Lower Bounds)**.** *The CUR Algorithm 10 must satisfy*

$$\sigma_k\left(CUR\right) \geq \frac{\sigma_k - \sigma_{p+1}\left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2 \left|\left|\Omega_2\Omega_1^\dagger\right|\right|_2 \left|\left|\Psi_2\Psi_1^\dagger\right|\right|_2}{\sqrt{1+\left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2 \left|\left|\Omega_2\Omega_1^\dagger\right|\right|_2^2}\sqrt{1+\left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2 \left|\left|\Psi_2\Psi_1^\dagger\right|\right|_2^2}}$$

However, the Nyström Decomposition Algorithm 11 satisfies a slightly better guarantee due to the symmetric positive semi-definite input and output

$$\sigma_k\left(CUC^T\right) \geq \frac{\sigma_k}{1+\left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2\left|\left|\Omega_2\Omega_1^\dagger\right|\right|_2^2}$$

Proof. Next, by the interlacing theorem for singular values, we have

$$\begin{aligned}
\sigma_k\left(CUR\right) &= \sigma_k\left(\widehat{Q}^T A\widetilde{Q}\right)\\
&\geq \sigma_k\left(\widehat{Q}_1^T A\widetilde{Q}_1\right)\\
&= \sigma_k\left(\left(I+H_1^TH_1\right)^{-1/2}\left(\Sigma_1+H_1^T\Sigma_3G_1\right)\left(I+G_1G_1^T\right)^{-1/2}\right)\\
&\geq \frac{\sigma_k-\sigma_{p+1}\left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2\left|\left|\Omega_2\Omega_1^\dagger\right|\right|_2\left|\left|\Psi_2\Psi_1^\dagger\right|\right|_2}{\sqrt{1+\left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2\left|\left|\Omega_2\Omega_1^\dagger\right|\right|_2^2}\sqrt{1+\left(\frac{\sigma_{p+1}}{\sigma_k}\right)^2\left|\left|\Psi_2\Psi_1^\dagger\right|\right|_2^2}}.
\end{aligned}$$

However for the Nyström Decomposition we get

$$\begin{aligned}
\sigma_k\left(CUC^T\right) &\geq \sigma_k\left(\left(I+H_1^TH_1\right)^{-1/2}\left(\Sigma_1+H_1^T\Sigma_3H_1\right)\left(I+H_1H_1^T\right)^{-1/2}\right)\\
&= \sigma_k\left(\left(I+H_1^TH_1\right)^{-1/2}\Sigma_1^{1/2}\left(I+\Sigma_1^{-1/2}H_1^T\Sigma_3H_1\Sigma_1^{-1/2}\right)\Sigma_1^{1/2}\left(I+H_1H_1^T\right)^{-1/2}\right)\\
&\geq \sigma_k\left(\left(I+H_1^TH_1\right)^{-1/2}\Sigma_1\left(I+H_1H_1^T\right)^{-1/2}\right)
\end{aligned}$$

where the last line is achieved by noting that the matrix $\Sigma_1^{-1/2}H_1^T\Sigma_3H_1\Sigma_1^{-1/2}$ is symmetric positive definite matrix, i.e. $\left(I+\Sigma_1^{-1/2}H_1^T\Sigma_3H_1\Sigma_1^{-1/2}\right)\succ I$ in the sense of Loewner ordering. This gives the desired result. □

Acknowledgments. I would like to thank my co-authors Dave Anderson, Simon S. Du, Kunming Wu, Michael W. Mahoney and Ming Gu for giving permission to publish this work on column and row selection in this thesis. The error bounds presented in this part of the thesis, which were derived independently, are tighter than the bounds presented in our original publication [6] because of the use of Lidskii's Theorem to improve bounds. The influence and work of my co-authors was invaluable in researching and exploring these methods.

Chapter 9

Sparse PCA via Secular Backwards Elimination

9.1 Background and Motivation

This work originated as a collaboration with Dave Anderson, Luming Wang and my adviser Ming Gu. Principal Component Analysis (PCA) is widely used in many areas of data analysis as an effective tool for dimensionality reduction. Performed via SVD, PCA extracts orthogonal linear combinations of the data variables that best explain the variance in the data. However, the principle components are typically dense, even when the data matrix is sparse. For a high dimensional data set, a dense loading vector may not be sufficiently informative to meet application needs. Motivated by this, sparse PCA algorithms impose sparseness on the loading vectors so that the user can see which individual variables play a role in principal directions of high variance and thus interpret data better. These directions give an intuition as to which variables are *principal variables* [76].

Early work to promote sparsity includes [61], which suggested using rotations to facilitate understanding of the principal components. A simple thresholding approach was discussed in [20], whereby small elements of the loading vectors are set to 0. The idea of restricting the loadings to a small set of values, including 0, was discussed in [63, 93]. The CUR decomposition is proposed in [73] to create sparsity by expressing the decomposition as a combination of a small number of columns and rows of the data matrix. SCotTLASS was introduced in [57] to enforce sparsity through a LASSO penalty function approach. Additional research into penalty functions to create sparsity includes sPCA-rSVD [86]. Sparse PCA (SPCA), [49], finds sparse approximations of the loading vectors by reworking PCA as a regression-type optimization problem. The Generalized Power Method [69] recasts the optimization of a non-convex function as an optimization problem on a Euclidean sphere or Stiefel manifold. The largest eigenvalues and corresponding eigenvectors are then calculated by using a gradient-type scheme. Deflation algorithms for sparse PCA have been discussed in [101, 102, 70].

A seminal work on sparse PCA appears in [2], which formulated DSPCA, a convex relaxation to convert the highly non-convex sparsity enforcing problem into a semi-definite program. The resulting problem benefits from the well-researched area of interior point methods, which can be used to solve the semi-definite program. More recently, a block coordinate ascent variant of DSPCA algorithm has been developed in [105] to approximately solve DSPCA much more efficiently. The authors of [1] additionally study forward-searching greedy algorithms and provide optimality conditions. These greedy algorithms allow one to build approximations of increasing rank, but the sequential nature of forward building algorithms implies they will miss possible optimal combinations of the loading vectors. A related work [79] suggested that backwards elimination algorithm has the ability to find sparse approximations that the forward search may miss. By eliminating rows sequentially, unobvious groups of loading vectors that are near-sparse will remain in consideration. But it is also claimed that the computational complexity is $O(n^4)$, which makes the backward algorithm prohibitively expensive for large n. Thus, that work focused on forward column selection instead, where a simple implementation yields an $O(n^3)$ algorithm.

We propose an efficient *backwards elimination* algorithm which exploits accurate low rank matrix approximations and secular equations for rank-1 updates to eigenvalue or singular value problems. To further improve the performance, a root test is introduced to bypass solving many unnecessary equations. We also provide theoretical bounds for explained variances (sparse singular values). A number of numerical tests show that compared with various competing approaches, our method can efficiently generate local optima to effectively explain the variances, often better than competing SPCA algorithms.

The rest of the chapter is organized as follows: Section 2 outlines notation and briefly reviews linear algebra preliminaries. Section 3 presents our efficient backwards elimination algorithm, along with theoretical bounds. The efficacy of the our algorithm is illustrated with numerical tests in Section 4, using both artificial and real-world data sets.

9.2 Problem Formulation

9.2.1 Sparse PCA

We denote $X \in \mathbb{R}^{m\times n}$ to be a given centered data matrix with m experiments and n variables. The corresponding covariance matrix is $A = X^T X$. Given a user defined degree of sparsity ℓ, the first sparse principal component or loading vector is commonly defined as $\vec{v} \in \mathbb{R}^n$ such that

$$\begin{array}{ll} \arg\max_{\vec{v}\in\mathbb{R}^n} \frac{\vec{v}^T A\vec{v}}{\vec{v}^T\vec{v}} & \text{or equivalently} \quad \arg\max_{\vec{v}\in\mathbb{R}^n} \frac{\|X\vec{v}\|_2}{\|\vec{v}\|} \\ \text{s.t. } \mathbf{card}\,(\vec{v}) \le \ell, & \text{s.t. } \mathbf{card}\,(\vec{v}) \le \ell. \end{array}$$

where $\mathbf{card}(\vec{v})$ is defined to be the number of non-zero entries in the vector $\vec{v} \in \mathbb{R}^n$. We can use this to define the corresponding sparse singular value $\sigma^s = \|X\vec{v}\|_2$. Also, note that there exists a unit vector $\vec{u} = \frac{1}{\sigma^s} X\vec{v} \in \mathbb{R}^m$ such that $\vec{u}^T X\vec{v} = \|X\vec{v}\|_2 := \sigma^s$. After finding the first

principal vector with $X^{(1)} = X$, we then deflate X in the following way

$$X^{(i+1)} = X^{(i)} - \sigma_i^s uv^T$$

and solve the same maximization problem on the deflated data matrix $X^{(2)}$, $X^{(3)}$, and so on. A systematic study of deflation schemes is done in [70]. The deflation method used in this thesis is equivalent to *projection deflation* because we are deflating directly on the data matrix. An advantage of this approach is that the covariance matrix $A^{(i)} = X^{(i)T}X^{(i)}$ is always positive semi-definite.

9.2.2 Accurate Low Rank Truncation

There are many efficient SVD-based low-rank approximation methods [44, 50]. Performing a rank-k SVD truncation $A \approx A_k = U_k\Sigma_k V_k^T$ will reduce the computational complexity of backwards elimination algorithm by trying to sparsify $\Sigma_k V_k^T \in \mathbb{R}^{n\times k}$ instead of from a potentially much larger matrix $A \in \mathbb{R}^{m\times n}$. We propose choosing k based upon the spectral decay of the data matrix X, which is typically very rapid for real-world data. The following theorem guarantees that the solution will keep almost the same accuracy with a judicious choice of k.

Theorem 38. *Let $X = U\Sigma V^T$ be the SVD of our data matrix. Let $0 \leq \tau \ll 1$ be a user defined tolerance and ℓ a user defined sparsity. Now, select k such that*

$$\sigma_{k+1} \leq \tau\sigma_1\sqrt{\frac{\ell}{n}}. \tag{9.1}$$

Let $X_k^{sparse} = \sigma^s\vec{u}\vec{v}^T$ be the output of the greedy backwards elimination *algorithm on X_k. Then*

$$\left|\|X - X_k^{sparse}\|_2 - \|X_k - X_k^{sparse}\|_2\right| \leq \tau\left\|X_k^{sparse}\right\|_2.$$

Theorem 38 ensures that the additional error introduced from an accurate low-rank matrix trunction on the data matrix can be negligible. However, this matrix truncation can significantly speed up the work of backwards elimination. We will prove Theorem 38 at the end of Section 9.3.

9.2.3 The Secular Equation

The secular equation is based on a formula for rank-1 updates of spectral problems, which, in turn, relies on the well-known determinant formula in Lemma 25.

Lemma 25 (Determinant Lemma [28])**.**

$$\det\left(A + \rho uv^T\right) = \det\left(A\right)\left(1 + \rho v^T A^{-1}u\right).$$

Given the SVD of matrix $X \in \mathbb{R}^{m \times n}$, we are interested in the top singular value of the matrix X with the j^{th} column removed or zeroed out, i.e. $X_{\setminus j} = X - x_j e_j^T$, where x_j is the j^{th} column. Thus, we need the top eigenvalue of $X_{\setminus j} X_{\setminus j}^T = XX^T - x_j x_j^T$. Its characteristic polynomial satisfies

$$\begin{aligned} P_{X_{\setminus j}}(\sigma^2) &:= \det\left(XX^T - x_j x_j^T - \sigma^2 I\right) \\ &= \det\left(XX^T - \sigma^2 I\right)\left(1 - x_j^T \left(XX^T - \sigma^2 I\right)^{-1} x_j\right). \end{aligned}$$

By setting $P_{X_{\setminus j}}(\sigma^2) = 0$, we arrive at the *secular function*

$$\begin{aligned} s_j(\sigma^2) &:= 1 - x_j^T \left(XX^T - \sigma^2 I\right)^{-1} x_j = 1 - (U^T x_j)^T \left(\Sigma\Sigma^T - \sigma^2 I\right)^{-1} U^T x_j \\ &= 1 - \sum_{i=1}^{m} \frac{(U^T x_j)_i^2}{\sigma_i^2 - \sigma^2} = 0. \end{aligned}$$

The roots of $s_j(\sigma^2)$ give us each $\sigma_k\left(X_{\setminus j}\right)$, which is known to satisfy

$$\sigma_{k+1}(X) \leq \sigma_k\left(X_{\setminus j}\right) \leq \sigma_k(X).$$

Many efficient solvers for the secular equations, such as "The Middle Way" [66], dramatically reduce the bottom line running time of our algorithm over naive solvers. An important note about the secular equations is that $s_j(\sigma^2)$ is increasing between the intervals and asymptote off to $-\infty$ and $+\infty$. Therefore, by intermediate value theorem, we have that for $\sigma_k(X) > \sigma > \sigma_{k+1}(X)$,

$$s_j\left(\sigma^2\right) \leq 0 \quad \text{if and only if} \quad \sigma \geq \sigma_k\left(X_{\setminus j}\right).$$

This fact will be used in our algorithm to skip solving many secular equations when we try to find

$$j^* = \arg\max_{j \in I} \sigma_1\left(X_{\setminus j}\right).$$

Roots of $P_X(\sigma^2)$ correspond to eigenvalues that have not changed, and thus need not be considered.

9.3 Algorithm and Main Results

The rank-k truncated SVD and the secular equations for rank-1 updates motivate Algorithm 13.

9.3.1 Singular Value Bounds

Next, we talk about theoretical guarantees.

Algorithm 13 SPCA via Secular Backwards Elimination (BEPCA)

Inputs: $m \times n$ data matrix X, sparsity ℓ, τ tolerance
Outputs: U_s, Σ_s and V_s s.t. $U_s \Sigma_s V_s^T \approx X$

Let $I = \{1, \cdots, n\}$
Compute a rank-k truncated SVD s.t. $U_k \Sigma_k V_k^T \approx A$, with k chosen according to (9.1).
Set $W_k \leftarrow \Sigma_k V_k^T$
for $i = n : -1 : \ell$ **do**
    Find j^* s.t. $j^* = \arg\max_{j \in I} \sigma_1 \left((W_k)_{\setminus j} \right)$ (Note: each secular eqn solve costs $O(k)$)
    Set $I \leftarrow I \setminus \{j^*\}$
    Zero out column j^* of W_k, i.e. $V_k(j^*, :) = 0$
end for
Solve for top singular value/vector pair of remaining matrix $U_k W_k$.

Theorem 39 (Existence of Special Sub-matrices)**.** *Let $F \in \mathbb{R}^{m \times n}$ and $F_{\setminus j} \in \mathbb{R}^{m \times (n-1)}$ be the matrix F with the j^{th} column removed. Then we have that for each $i \in \{1, \cdots, n\}$*

$$\max_{1 \leq j \leq n} \sigma_i^2 \left(F_{\setminus j} \right) \geq \frac{n-i}{n} \sigma_i^2 \left(F \right) + \frac{i}{n} \sigma_n^2 \left(F \right)$$

A sketch of the proof is as follows.
Let u_j denote the j^{th} column of F. Using Lemma 25, we examine

$$\begin{aligned} \det \left(F_{\setminus j} F_{\setminus j}^T - \sigma^2 I \right) &= \det \left(F F^T - \sigma^2 I \right) \left(1 - u_j^T \left(F F^T - \sigma^2 I \right)^{-1} u_j \right) \\ &= \det \left(F F^T - \sigma^2 I \right) f_j \left(\sigma^2 \right). \end{aligned}$$

We look for the roots of f_j, which correspond to eigenvalues that have changed. Using the trace trick on f_j, we average over j

$$f_{avg} \left(\sigma^2 \right) := \frac{1}{n} \sum_{j=1}^{n} f_j \left(\sigma^2 \right) = \frac{1}{n} \mathrm{tr} \left(I - \left(F^T F - \sigma^2 I \right)^{-1} F^T F \right) = -\frac{1}{n} \sum_{s=1}^{n} \frac{\sigma^2}{\sigma_s^2 - \sigma^2}$$

Now, let $j^* = \arg\max_{1 \leq j \leq n} \sigma_i \left(F_{\setminus j} \right)$. By interlacing, we have $\sigma_{i+1} \left(F \right) \leq \sigma_i \left(F_{\setminus j} \right) \leq \sigma_i \left(F_{\setminus j^*} \right) \leq \sigma_i \left(F \right)$. Thus, each secular equation has $0 = f_j \left(\sigma_i^2 \left(F_{\setminus j} \right) \right) \geq f_j \left(\sigma_i^2 \left(F_{\setminus j^*} \right) \right)$. So averaging over j yields

$$0 \;\geq\; f_{avg} \left(\sigma_i^2 \left(F_{\setminus j^*} \right) \right) \;=\; -\frac{1}{n} \sum_{s=1}^{n} \frac{\sigma^2}{\sigma_s^2 - \sigma^2}.$$

This implies

$$0 \;\leq\; \frac{1}{n} \sum_{s=i+1}^{n} \frac{\sigma^2}{\sigma_s^2 - \sigma^2} + \frac{1}{n} \sum_{s=1}^{i} \frac{\sigma^2}{\sigma_s^2 - \sigma^2} \;\leq\; \frac{n-i}{n} \frac{\sigma^2}{\sigma_n^2 - \sigma^2} + \frac{i}{n} \frac{\sigma^2}{\sigma_i^2 - \sigma^2}$$

which simplifies to the desired result.

Corollary 6. *Let $F \in \mathbb{R}^{m\times n}$ be a matrix. Let the set $J^* = \left\{j_1^*, \cdots, j_{n-\ell}^*\right\} \subset \{1, \cdots, n\}$ be defined as the set of $n-\ell$ columns removed by Algorithm 13, i.e.*

$$j_i^* = arg \max_{1\leq j\leq n-i+1} \sigma_1\left(F_{\setminus\left\{j_1^*,\cdots,j_{i-1}^*\right\}}\right).$$

Let $F_{\setminus J^*} \in \mathbb{R}^{m\times \ell}$ be the matrix with the columns removed. Then, we have that

$$\sigma_1\left(F_{\setminus J^*}\right) \;=\; \sigma_1\left(F_{\setminus\left\{j_1^*,\cdots,j_{n-\ell}^*\right\}}\right) \;\geq\; \sqrt{\frac{\ell}{n}}\sigma_1\left(F\right).$$

Corollary 6 follows by applying Theorem 39 $n-\ell$ times with $i=1$. Next, we highlight an important tool before proceeding:

Theorem 40 (Weyl's Inequality for Singular Values)**.** *Let $Y, Z \in R^{m\times n}$ be any matrices and $i, j \in \mathbb{N}$ such that $i+j \leq n+1$. Then*

$$\sigma_{i+j-1}\left(Y+Z\right) \leq \sigma_i\left(Y\right) + \sigma_j\left(Z\right)$$

This is exercise III.6.5 in [14]. Theorem 40 will be used in our final Theorem concerning sparse singular values. The following theorem is particularly important because it gives us theoretical guarantees of using the orthogonal deflation process in [70] with backwards elimination for the *every* sparse principle component instead of the simply the first one. To the author's knowledge, this is the first theoretical guarantee of its kind in the literature for SPCA.

Theorem 41 (Sparse Singular Value Bounds)**.** *Let $X \in R^{m\times n}$. Then, the r^{th} sparse singular value obeys the following inequality*

$$\sigma_r^s \geq \sqrt{\frac{\ell}{n}} \max_{1\leq j\leq r}\left(\sigma_j\left(X\right) - \sum_{i=j}^{r-1}\sigma_i^s\right).$$

Proof. We apply Corollary 6 to the sparse singular value found by Backwards Elimination

$$\sigma_r^s = \sigma_1\left(\left[X - \sum_{t=1}^{r-1}\sigma_t^s u_t v_t^T\right]_{\setminus J^*}\right) \geq \sqrt{\frac{\ell}{n}}\sigma_1\left(X - \sum_{t=1}^{r-1}\sigma_t^s u_t v_t^T\right)$$

The remainder of the proof will be done by induction on r. The base case of $r=1$ is immediate, i.e. $\sigma_1(X) = \sigma_1(X)$. For the inductive step, we simply need to prove that

$$\sigma_1\left(X - \sum_{t=1}^{r}\sigma_t^s u_t v_t^T\right) \geq \max\left(\sigma_1\left(X - \sum_{t=1}^{r-1}\sigma_t^s u_t v_t^T\right) - \sigma_r^s, \sigma_{r+1}\left(X\right)\right)$$

However, this is just two simple applications of Weyl's Inequality. The first one set $Y = X - \sum_{t=1}^{r} \sigma_t^s u_t v_t^T$ and $Z = \sigma_r^s u_r v_r^T$ with $i = j = 1$ to get

$$\begin{aligned}\sigma_1\left(X - \sum_{t=1}^{r} \sigma_t^s u_t v_t^T\right) &= \sigma_1(Y) \geq \sigma_1(Y+Z) - \sigma_1(Z) \\ &= \sigma_1\left(X - \sum_{t=1}^{r-1} \sigma_t^s u_t v_t^T\right) - \sigma_1\left(\sigma_r^s u_r v_r^T\right) = \sigma_1\left(X - \sum_{t=1}^{r-1} \sigma_t^s u_t v_t^T\right) - \sigma_r^s\end{aligned}$$

The second set $Y = X - \sum_{t=1}^{r} \sigma_t^s u_t v_t^T$ and $Z = \sum_{t=1}^{r} \sigma_t^s u_t v_t^T$ with $i = 1$ and $j = r+1$

$$\begin{aligned}\sigma_1\left(X - \sum_{t=1}^{r} \sigma_t^s u_t v_t^T\right) &= \sigma_1(Y) \geq \sigma_{r+1}(Y+Z) - \sigma_{r+1}(Z) \\ &= \sigma_{r+1}(X) - \sigma_{r+1}\left(\sum_{t=1}^{r} \sigma_t^s u_t v_t^T\right) = \sigma_{r+1}(X)\end{aligned}$$

Taking the maximum over the two lower bounds establishes the recursion. □

An important corollary is the sparse singular values always capture at least a fraction of the true ones.

Corollary 7 (Sparse σ_r^s at least a fraction of true $\sigma_r(X)$)**.** *Let $X \in R^{m \times n}$. Then, the r^{th} sparse singular value obeys the following inequality*

$$\sigma_r^s \geq \sqrt{\frac{\ell}{n}} \sigma_r(X).$$

We are now ready to prove Theorem 38 as a Corollary of Theorem 41.

Proof of Theorem 38. Consider $r = 1$ in Theorem 41, we have

$$\|X_k^{sparse}\|_2 \geq \sigma^s \geq \sqrt{\frac{\ell}{n}} \sigma_1(X).$$

It follows that

$$\begin{aligned}\|X - X_k^{sparse}\|_2 &= \|X - X_k + X_k - X_k^{sparse}\|_2 \leq \sigma_{k+1} + \|X_k - X_k^{sparse}\|_2 \\ &\leq \tau\sigma_1\sqrt{\frac{\ell}{n}} + \|X_k - X_k^{sparse}\|_2 \leq \tau \|X_k^{sparse}\|_2 + \|X_k - X_k^{sparse}\|_2\end{aligned}$$

Analogously,

$$\|X_k - X_k^{sparse}\|_2 \leq \tau \|X_k^{sparse}\|_2 + \|X - X_k^{sparse}\|_2.$$

□

9.4 Numerical Experiments

We compare our sparse PCA algorithm with four other popular methods: the DSPCA algorithm (DSPCA) [2], the approximation greedy forward-search algorithm (FSPCA) [1], and the single-unit generalized power methods with $\mathcal{L}_0$ (PowerL0) or $\mathcal{L}_1$ penalty terms (PowerL1)[69]. Both DSPCA algorithm and approximation greedy search algorithm are implemented by using the software package from `http://www.di.ens.fr/~aspremon/software.html`. Given covariance matrices, we construct the artificial data matrix via Cholesky factorization or the matrix square-root using eigenvalue decomposition.

Table 9.1: Results for synthetic test 4.1.

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	Variance
BEPCA, PC1	0	0.4082	0	-0.4082	-0.4082	-0.4082	0	0	0.4082	0	0	0.4082	90
FSPCA, PC1	-0.6220	0	0	0.0399	-0.4295	-0.2487	-0.5595	0	0	0	-0.2284	0	88.68
DSPCA, PC1	-0.7706	0	0	0	-0.2310	-0.0856	-0.5878	-0.0001	0	0.0003	0	0	87.19
PowerL0, PC1	0.5774	0	0	0	0.2887	0.2887	0.5774	0.2887	0	-0.2887	0	0	90
PowerL1, PC1	0.5774	0	0	0	0.2887	0.2887	0.5774	0.2887	0	-0.2887	0	0	90

9.4.1 Synthetic Example with Dense Leading Eigenvectors

We choose 4 mutually orthonormal vectors $\{\tilde{v_1}, \tilde{v_2}, \tilde{v_3}, \tilde{v_4}\}$ whose entries are either $+1$ or -1 (i.e. rows of Hadamard matrix):

$$\tilde{v_1} = (1,1,1,1,1,1,1,1,1,1,1,1)^T, \qquad \tilde{v_2} = (1,-1,1,-1,1,1,1,-1,-1,-1,1,-1)^T$$
$$\tilde{v_3} = (1,-1,-1,1,-1,1,1,1,-1,-1,-1,1)^T, \quad \tilde{v_4} = (1,1,-1,-1,1,-1,1,1,1,-1,-1,-1)^T$$

We generate another 8 random vectors $\{\tilde{v}_5, \tilde{v}_6, \ldots, \tilde{v}_{12}\}$ whose entries are drawn from uniform distribution $U(0,1)$, and form a matrix $\tilde{V} = \{\tilde{v_1}, \tilde{v_2}, \ldots, \tilde{v}_{12}\}$ with full rank. Then, we compute an orthonormal matrix $V = \{v_1, v_2, \ldots, v_{12}\}$ by applying Gram-Schmidt orthogonalization to $\tilde{V}$. Using columns of V as eigenvectors, we obtain the covariance matrix with the eigenvalues as $90, 90, 90, 90, 80, 79, 70, 50, 20, 18, 15, 15$.

We test all 5 methods by generating the primary loading vector with cardinality 6. Note that all eigenvectors are dense in this test, but the linear combination of the first 4 eigenvectors is sparse. As the first 4 largest eigenvalues are all equal to 90, an ideal sparse PCA algorithm should be able to detect the implicit sparsity and return a linear combination of the first 4 eigenvectors.

In table 9.1, BEPCA and the generalized power methods with $\mathcal{L}_0$ penalty or $\mathcal{L}_1$ penalty successfully identified the potential sparsity pattern and return the first loading vector as a linear combination of $\{v_1, v_2, v_3, v_4\}$ with explained variance 90 (e.g. the first loading vector for BEPCA is actually $\sqrt{2}(v_3 - v_2)$). However, FSPCA and DSPCA fail to recognize the

Table 9.2: Results for synthetic test 4.2.

	First Loading Vector				Second Loading Vector			
	Median Angle(^{o})	Median Variance(%)	Correct(%)	Incorrect(%)	Median Angle(^{o})	Median Variance(%)	Correct(%)	Incorrect(%)
	Sample Size 50							
BEPCA	4.89	47.55	98.75	0.83	13.34	24.48	84.0	10.67
FSPCA	5.17	47.51	93.25	4.50	13.75	24.44	76.5	15.67
PowerL0	4.98	47.55	96.75	2.17	13.53	24.48	79.75	13.50
PowerL1	4.99	47.55	96.25	2.50	13.37	24.48	79.63	13.58
	Sample Size 200							
BEPCA	2.36	47.86	100	0	5.11	23.86	94.87	3.42
FSPCA	2.48	47.86	96	2.67	5.33	23.86	91.50	5.67
PowerL0	2.44	47.86	98	1.33	5.35	23.86	92	5.33
PowerL1	2.44	47.86	97	2	5.32	23.86	92.63	4.92

linear combination over 4 different eigenvectors and produce a sparse loading vector with the same cardinality but less variance.

If the leading eigenvectors are sparse, BEPCA and FSPCA can both recover the sparsity pattern and output the same accurate solution. However, when the leading eigenvectors are dense, FSPCA shows less accuracy than BEPCA. In this example, FSPCA looks for additional columns to maximize the leading eigenvalue of its selected sub-matrix and wrongly chooses columns from the last 8 random eigenvectors–giving a sub-optimal explained variance. By eliminating columns iteratively, BEPCA retains the first 4 eigenvectors and then achieves the largest possible eigenvalue. This illustrates that the backward elimination is able to obtain more accuracy than forward selection.

9.4.2 Synthetic Example for Data Matrices

This test was proposed by [86], where two sparse eigenvectors were chosen as

$$v_1 = (0.422, 0.422, 0.422, 0.422, 0, 0, 0, 0, 0.38, 0.38)^T$$
$$v_2 = (0, 0, 0, 0, 0.489, 0.489, 0.489, 0.489, -0.147, -0.147)^T$$

and then use the same trick from last numerical test finding the 8 random vectors and obtain an orthonormal matrix $V = \{v_1, v_2, v_3, \ldots, v_{10}\}$. We choose covariance matrix $\Sigma = VSV^T$ with 10 eigenvalues as $200, 100, 50, 50, 6, 5, 4, 3, 2, 1$. We generate sample data matrices of size 50 from the artificial covariance matrix above and calculate the first two sparse loading vectors with cardinality 6. Such a test is simulated 200 times, and then we investigate the medians of explained variance and the angles between the extracted loading vectors and the corresponding real eigenvectors, as well as the percentage of correctly/incorrectly identified zero loadings for loading vectors.

We perform this test on all sparse PCA algorithms except for DSPCA because the penalty parameter was unable to achieve a cardinality to 6 every time. Table 9.2 shows that BEPCA returns the smallest median angles, the best percentage of correctly/incorrectly identified zeros. We repeat the same test 100 times but with sample size 200 at each time, and the results are significantly improved for every algorithm. BEPCA identifies the zeros perfectly when recovering the first sparse eigenvectors. Based on this test, we can see that BEPCA quickly figures out the sparsity pattern of loading vectors with much fewer samples than any of the other methods.

9.4.3 Pit Props Data

Pit Props data is a classic benchmark example to test sparse PCA algorithms due to the difficulty of interpreting principal components and lack of sparsity. It consists 180 observations and 13 measured variables. We apply all the candidate methods to extract the first 6 loading vectors with similar cardinality restriction.

Table 9.3 shows that compared with DSPCA and FSPCA, BEPCA explains the most cumulative variance using the same number of sparse loading vectors and the same cardinalities. As for generalized power methods, BEPCA can interpret nearly the same variance for the first few loading vectors but with less cardinality. The cumulative explained variance exceeds those of generalized power methods as more loading vectors are generated.

Table 9.3: Results for PitProps test.

	Cumulative Cardinality						Cumulative Explained Variance					
	1PC	2PCs	3PCs	4PCs	5PCs	6PCs	1PC	2PCs	3PCs	4PCs	5PCs	6PCs
BEPCA	6	8	11	12	13	14	29.01	43.48	56.12	63.81	71.50	79.19
FSPCA	6	8	11	12	13	14	29.01	39.23	54.10	61.79	69.48	77.18
DSPCA	6	8	11	12	13	14	26.60	41.08	54.23	61.92	69.61	77.30
PowerL0	7	8	10	13	14	15	30.74	38.43	52.91	63.63	71.32	79.02
PowerL1	7	9	12	13	14	15	30.74	45.22	55.94	63.63	71.32	79.02

9.4.4 Gene Expression Data

We examine our algorithm on a large gene expression data matrix obtained from Gene Expression Omnibus with GEO accession number GSE10006 [36, 88]. The data was originally used to study the effect of smoking on the gene expressions of the intestinal lactoferrin receptor in a particular tissue of the human airway [22]. The matrix has 87 samples and $54,675$ measured variables. We test BEPCA by generating the first sparse loading vector with cardinality 200, based on the truncated SVD with rank $k = 10, 30, 50, 70$ and 87 (i.e. full SVD). For each k, the run-time and explained variance are compared in the left plot of

Fig 9.1. As k decreases, the explained variance only gradually decays. The performance, nevertheless, benefits remarkably from small values of k. This corroborates Theorem 38 and allows us to apply BEPCA efficiently to large data.

We want sparse principle components that can explain the variance based on different small groups of variables and avoid the appearance of collinearity and linear dependence among loading vectors [69, 105]. Therefore, for the second part of this test, we extract the first three loading vectors $\{v_1, v_2, v_3\}$ and the corresponding right singular vectors $\{u_1, u_2, u_3\}$ based on BEPCA, and compute the angles between them. From table 9.4, we can clearly see that both u's and v's are almost orthogonal with each other, which means that multiple principal components can interpret the original data from different directions.

Table 9.4: Angles between singular vectors (degree)

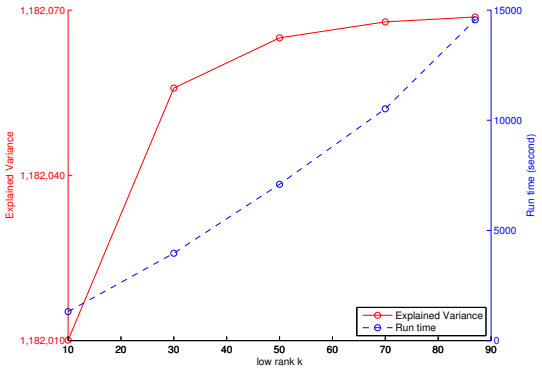
	1st vs 2nd	1st vs 3rd	2nd vs 3rd
Loading Vectors v	89.7797	87.6663	87.3315
Right Singular Vectors u	88.5587	86.4175	85.8174

Various tests suggested that PowerL1 turns out to be more efficient algorithms but suffers from at least two shortcomings: first, penalty parameters in PowerL1 must be tuned to obtain the desired cardinality of loading vectors, which reduces the performance by repeating the program a number of times; second, as showed in most of tests, PowerL1 is less accurate than BEPCA and BEPCA can select better variables (See the result of synthetic test 2). Therefore, we can use PowerL1 as a preprocessor to zero out a fair amount of components and continue to run BEPCA until achieving the desired degree of sparsity. By this, no parameter tuning is required. We test this hybrid algorithm still by extracting the first loading vector with cardinality 200. The right plot of 9.1 reports the run time and variance for different sparsity levels induced by PowerL1. We can see from the plot that the hybrid algorithm is able to explain more variances with almost no extra cost on time.

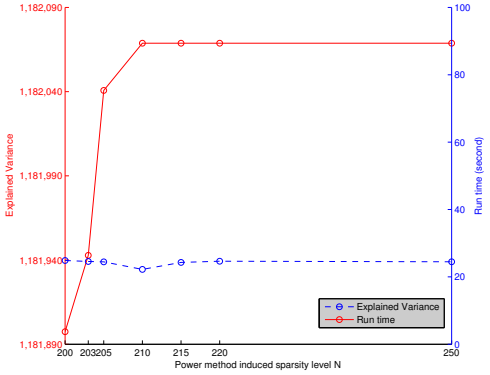
9.5 Conclusion

We have presented a sparse PCA algorithm using backwards column selection. Utilizing a low rank truncated SVD and solving the secular equations for eigenvalues significantly improves performance. Also, several singular value bounds were derived to guarantee accuracy. Numerical experiments demonstrate that our algorithm is also able to extract more accurate sparse loading vectors and explain more variances comparing with some other popular techniques.

Acknowledgments. I would like to thank my co-authors Dave Anderson, Luming Wang and Ming Gu for giving me permission to use this work in my thesis.



Run time and variances vs rank k



Run time and variances for hybrid algorithm

Figure 9.1: Test for Gene Expression Data

Bibliography

[1] F. R. Bach A. d'Aspremont and L. El Ghaoui. "Optimal solutions for sparse principal component analysis". In: *Journal of Machine Learning Research* 9:1269-1294 (2008).

[2] M. I. Jordan A. d'Aspremont L. El Ghaoui and G. R. G. Lanckriet. "A Direct Formulation for Sparse PCA using Semidefinite Programming". In: *Siam Review* 49:434-448 (2007).

[3] Emmanuel Agullo et al. "Numerical linear algebra on emerging architectures: The PLASMA and MAGMA projects". In: *Journal of Physics: Conference Series*. Vol. 180. 1. IOP Publishing. 2009, p. 012037.

[4] Nir Ailon and Bernard Chazelle. "The fast Johnson-Lindenstrauss transform and approximate nearest neighbors". In: *SIAM Journal on Computing* 39.1 (2009), pp. 302–322.

[5] Phillip Alpatov et al. "PLAPACK: Parallel Linear Algebra Package." In: *PPSC*. Citeseer. 1997.

[6] David Anderson et al. "Spectral Gap Error Bounds for Improving CUR Matrix Decomposition and the Nyström". In: *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*. 2015, pp. 19–27.

[7] David G Anderson, Ming Gu, and Christopher Melgaard. "An Efficient Algorithm for Unweighted Spectral Graph Sparsification". In: *arXiv preprint arXiv:1410.4273* (2014).

[8] E. Anderson et al. *LAPACK Users' Guide*. Second. Philadelphia, PA: SIAM, 1994.

[9] Edward Anderson et al. *LAPACK Users' guide*. Vol. 9. Siam, 1999.

[10] Sheldon Jay Axler. *Linear algebra done right*. Vol. 2. Springer, 1997.

[11] Marc Baboulin, Xiaoye S Li, and François-Henry Rouet. "Using Random Butterfly Transformations to Avoid Pivoting in Sparse Direct Methods". In: *High Performance Computing for Computational Science–VECPAR 2014*. Springer, 2014, pp. 135–144.

[12] Marc Baboulin et al. "Accelerating linear system solutions using randomization techniques". In: *ACM Transactions on Mathematical Software (TOMS)* 39.2 (2013), p. 8.

[13] Francis R Bach and Michael I Jordan. "Kernel independent component analysis". In: *The Journal of Machine Learning Research* 3 (2003), pp. 1–48.

[14] R. Bhatia. *Matrix Analysis*. New York, NY: Springer, 1997.

[15] Jacob Bien, Ya Xu, and Michael W Mahoney. "CUR from a sparse optimization viewpoint". In: *Advances in Neural Information Processing Systems*. 2010, pp. 217–225.

[16] L Susan Blackford et al. "An updated set of basic linear algebra subprograms (BLAS)". In: *ACM Transactions on Mathematical Software* 28.2 (2002), pp. 135–151.

[17] Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. "Near-optimal column-based matrix reconstruction". In: *SIAM Journal on Computing* 43.2 (2014), pp. 687–717.

[18] Christos Boutsidis and David P Woodruff. "Optimal cur matrix decompositions". In: *arXiv preprint arXiv:1405.7910* (2014).

[19] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[20] J. Cadima and I. T. Jolliffe. "Loadings and correlations in the interpretation of principal components". In: *Journal of Applied Statistics* 22:203-214 (1995).

[21] D. Cai. "Text datasets in matlab format". In: `http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html`, formatted version of: D. D. Lewis. Reuters-21578 Text Categorization Collection Data Set. AT&T Labs. `http://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection` (1987).

[22] De BP Carolan BJ Harvey BG and Vanni H et al. "Decreased expression of intelectin 1 in the human airway epithelium of smokers compared to nonsmokers". In: *J Immunol* 181 (8 2008 Oct 15), pp. 5760–7.

[23] Z. Chen and J. Dongarra. "Condition numbers of Gaussian random matrices". In: *SIAM J. Matrix Anal. Appl.* 27 (2005), pp. 603–620.

[24] Jaeyoung Choi et al. "ScaLAPACK: A portable linear algebra library for distributed memory computersDesign issues and performance". In: *Applied Parallel Computing Computations in Physics, Chemistry and Engineering Science*. Springer, 1996, pp. 95–106.

[25] Petros Drineas Christos Boutsidis and Malik Magdon-Ismail. "Near-Optimal Column-Based Matrix Reconstruction". In: (2011). arXiv: `1103.0995v3`.

[26] John D Cook. "Upper bounds on non-central chi-squared tails and truncated normal moments". In: (2010).

[27] Nello Cristianini, John Shawe-Taylor, and Huma Lodhi. "Latent semantic kernels". In: *Journal of Intelligent Information Systems* 18.2-3 (2002), pp. 127–152.

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

[29] Simplice Donfack, Stanimire Tomov, and Jack Dongarra. "Dynamically balanced synchronization-avoiding LU factorization with multicore and GPUs". In: *Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International*. IEEE. 2014, pp. 958–965.

[30] J. Dongarra et al. *Linpack User's Guide*. Philadelphia, PA: SIAM, 1979.

[31] Petros Drineas and Michael W Mahoney. "On the Nyström method for approximating a Gram matrix for improved kernel-based learning". In: *The Journal of Machine Learning Research* 6 (2005), pp. 2153–2175.

[32] Petros Drineas, Michael W Mahoney, and S Muthukrishnan. "Relative-error CUR matrix decompositions". In: *SIAM Journal on Matrix Analysis and Applications* 30.2 (2008), pp. 844–881.

[33] Petros Drineas et al. "Fast approximation of matrix coherence and statistical leverage". In: *Journal of Machine Learning Research* 13.1 (2012), pp. 3475–3506.

[34] C. Eckart and G. Young. "The approximation of one matrix by another of lower rank". In: *Psychometrika* 1 (1936), pp. 211–218.

[35] Alan Edelman. "The complete pivoting conjecture for Gaussian elimination is false". In: *Mathematica Journal* 2.2 (1992), pp. 58–61.

[36] Lash AE. Edgar R Domrachev M. "Gene Expression Omnibus: NCBI gene expression and hybridization array data repository". In: *Nucleic Acids Res.* 30 (1 2002), pp. 207–10.

[37] Shai Fine and Katya Scheinberg. "Efficient SVM training using low-rank kernel representations". In: *The Journal of Machine Learning Research* 2 (2002), pp. 243–264.

[38] Leslie V. Foster. "Gaussian Elimination with Partial Pivoting Can Fail in Practice". In: *SIAM Journal on Matrix Analysis and Applications* 15.4 (1994), pp. 1354–1362.

[39] Leslie V. Foster. "The Growth Factor and Efficiency of Gaussian Elimination with Rook Pivoting". In: *Journal of Computational and Applied Mathematics* 86 (1997). Corrigendum in JCAM, 98:177, 1998, pp. 177–194.

[40] Philip E Gill, Walter Murray, and Michael A Saunders. "SNOPT: An SQP algorithm for large-scale constrained optimization". In: *SIAM journal on optimization* 12.4 (2002), pp. 979–1006.

[41] Alex Gittens. "The spectral norm error of the naive Nyström extension". In: *arXiv preprint arXiv:1110.5305* (2011).

[42] Alex Gittens and Michael W Mahoney. "Revisiting the Nyström method for improved large-scale machine learning". In: *arXiv preprint arXiv:1303.1849* (2013).

[43] K. Goldberg. "Anonymous Ratings from the Jester Online Joke Recommender System". In: `http://eigentaste.berkeley.edu/dataset/` (2003).

[44] G. Golub and C. Van Loan. *Matrix Computations*. 3nd. Baltimore, MD: Johns Hopkins University Press, 1996.

[45] Nick Gould. "On growth in Gaussian elimination with complete pivoting". In: *SIAM Journal on Matrix Analysis and Applications* 12.2 (1991), pp. 354–361.

[46] Laura Grigori, James W Demmel, and Hua Xiang. "CALU: a communication optimal LU factorization algorithm". In: *SIAM Journal on Matrix Analysis and Applications* 32.4 (2011), pp. 1317–1350.

[47] Ming Gu. "Subspace Iteration Randomization and Singular Value Problems". In: *SIAM Journal on Scientific Computing* 37.3 (2015), A1139–A1173.

[48] Venkatesan Guruswami and Ali Kemal Sinop. "Optimal column-based low-rank matrix reconstruction". In: *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2012, pp. 1207–1214.

[49] T. Hastie H. Zou and R. Tibshirani. "Sparse principal component analysis". In: *J. Comput. Graphical Statist.* 15:265-286 (2006).

[50] N. Halko, P.-G. Martinsson, and J. A. Tropp. "Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions". In: *SIAM Review* 53 (2011), pp. 217–288.

[51] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 1996.

[52] A. J. Hoffman and H. W. Wielandt. "The variation of the spectrum of a normal matrix". In: *Duke Mathematics* 20 (1953), pp. 37–39.

[53] R. A. HORN and C. R. JOHNSON. *Matrix Analysis*. CAMBRIDGE UNIVERSITY PRESS, 1985.

[54] R. A. HORN and C. R. JOHNSON. *Matrix Analysis*. CAMBRIDGE UNIVERSITY PRESS, 2013.

[55] Gary B Huang et al. *Labeled faces in the wild: A database for studying face recognition in unconstrained environments*. Tech. rep. Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[56] J. M. Hyman and B. Nicolaenko. "The Kuramoto-Sivashinsky equation: a bridge between PDE's and dynamical systems". In: *Physica D* 18 (1986), pp. 113–126.

[57] N. T. Trendafilov I. T. Jolliffe and M. Uddin. "A modified principal component technique based on the LASSO". In: *Journal of Computational and Graphical Statistics* 12(3):531-547 (2003).

[58] MKL Intel. *Intel math kernel library*. 2007.

[59] Fritz John. "Extremum problems with inequalities as subsidiary conditions". In: *Traces and Emergence of Nonlinear Programming*. Springer, 2014, pp. 197–215.

[60] I. T. Jolliffe. *Principal Component Analysis*. New York: Springer Verlag, 1986.

[61] I.T. Jolliffe. "Rotation of principal components: choice of normalization constraints". In: *Journal of Applied Statistics* 22:29-35 (1995).

[62] Amal Khabou et al. "LU factorization with panel rank revealing pivoting and its communication avoiding version". In: *SIAM Journal on Matrix Analysis and Applications* 34.3 (2013), pp. 1401–1429.

[63] T. G. Kolda and D. P. O'Leary. "Computation and uses of the semidiscrete matrix decomposition". In: 26:415-435 (2000).

[64] Robert Eugene LaQuey et al. "Nonlinear saturation of the trapped-ion mode". In: *Physical Review Letters* 34.7 (1975), p. 391.

[65] Michel Ledoux. *The concentration of measure phenomenon*. 89. American Mathematical Soc., 2005.

[66] R.-C. Li. *Solving secular equations stably and efficiently*. Computer Science Dept. Technical Report CS-94-260. (LAPACK Working Note #89). Knoxville: University of Tennessee, 1994.

[67] M. Lichman. *UCI Machine Learning Repository*. 2013. URL: `http://archive.ics.uci.edu/ml`.

[68] Lixin Liu and Robert D Russell. "Linear system solvers for boundary value ODEs". In: *Journal of Computational and Applied Mathematics* 45.1 (1993), pp. 103–117.

[69] P. Richtárik M. Journée Y. Nesterov and R. Sepulchre. "Generalized Power Method for Sparse Principal Component Analysis". In: *Journal of Machine Learning Research* 11:517-553 (2010).

[70] L. Mackey. "Deflation Methods for Sparse PCA". In: *Adv. NIPS*. 2009.

[71] Michael W Mahoney. "Randomized algorithms for matrices and data". In: *Foundations and Trends® in Machine Learning* 3.2 (2011), pp. 123–224.

[72] Michael W Mahoney and Petros Drineas. "CUR matrix decompositions for improved data analysis". In: *Proceedings of the National Academy of Sciences* 106.3 (2009), pp. 697–702.

[73] Michael W Mahoney and Petros Drineas. "CUR matrix decompositions for improved data analysis". In: *Proceedings of the National Academy of Sciences* 106.3 (2009), pp. 697–702.

[74] Michael W Mahoney, Mauro Maggioni, and Petros Drineas. "Tensor-CUR decompositions for tensor-based data". In: *SIAM Journal on Matrix Analysis and Applications* 30.3 (2008), pp. 957–987.

[75] Per-Gunnar Martinsson. "A fast randomized algorithm for computing a hierarchically semiseparable representation of a matrix". In: *SIAM Journal on Matrix Analysis and Applications* 32.4 (2011), pp. 1251–1274.

[76] G. McCabe. "Principle Variables". In: *Technometrics* 26 (1984), pp. 137–144.

[77] L Miranian and Ming Gu. "Strong rank revealing LU factorizations". In: *Linear algebra and its applications* 367 (2003), pp. 1–16.

[78] L. Mirsky. "A Trace Inequality of John von Neumann". eng. In: *Monatshefte fr Mathematik* 79 (1975), pp. 303–306. URL: http://eudml.org/doc/177697.

[79] B. Moghaddam, Y. Weiss, and S. Avidan. "Spectral bounds for sparse PCA: Exact and greedy algorithms". In: *Adv. NIPS*. 2006.

[80] Larry Neal and George Poole. "A geometric analysis of Gaussian elimination. II". In: *Linear Algebra Appl.* 173 (1992), pp. 239–264. ISSN: 0024-3795. DOI: 10.1016/0024-3795(92)90432-A. URL: http://dx.doi.org/10.1016/0024-3795(92)90432-A.

[81] Victor Y Pan, Guoliang Qian, and Ai-Long Zheng. "Randomized preprocessing versus pivoting". In: *Linear Algebra and Its Applications* 438.4 (2013), pp. 1883–1899.

[82] D Stott Parker. "Random butterfly transformations with applications in computational linear algebra". In: (1995).

[83] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[84] George Poole and Larry Neal. "The rook's pivoting strategy". In: *J. Comput. Appl. Math.* 123.1-2 (2000). Numerical analysis 2000, Vol. III. Linear algebra, pp. 353–369. ISSN: 0377-0427. DOI: 10.1016/S0377-0427(00)00406-4. URL: http://dx.doi.org/10.1016/S0377-0427(00)00406-4.

[85] Phillip G Schmitz and Lexing Ying. "A fast direct solver for elliptic problems on general meshes in 2D". In: *Journal of Computational Physics* 231.4 (2012), pp. 1314–1338.

[86] H. Shen and J. Z. Huang. "Sparse principal component analysis via regularized low rank matrix approximation". In: *Journal of Multivariate Analysis* 99(6):1015-1034 (2008).

[87] GW Stewart. "Matrix Algorithms: Basic Decompositions (Volume 1)". In: *Society for Industrial and Applied Math* (1998).

[88] Barrett T et al. "NCBI GEO: archive for functional genomics data sets–update." In: *Nucleic Acids Res.* 41 (Database issue 2013), pp. D991–5.

[89] Christian Thurau, Kristian Kersting, and Christian Bauckhage. "Deterministic CUR for Improved Large-Scale Data Analysis: An Empirical Study." In: *SDM*. SIAM. 2012, pp. 684–695.

[90] L. Trefethen and R. Schreiber. "Average case analysis of Gaussian elimination". In: *SIAM J. Mat. Anal. Appl.* 11.3 (1990), pp. 335–360.

[91] M. Turk and A. Pentland. "Eigenfaces for recognition". In: *Journal of Cognitive Neuroscience* 3.1 (1991), pp. 71–86.

[92] Santosh Srinivas Vempala. *The random projection method*. Vol. 65. DIMACS series in discrete mathematics and theoretical computer science. Appendice p.101-105. Providence, R.I. American Mathematical Society, 2004. ISBN: 0-8218-2018-4.

[93] S. Vines. "Simple principal components". In: *Appl. Statist.* 49:441-451 (2000).

[94] John Von Neumann and Herman H Goldstine. "Numerical inverting of matrices of high order". In: *Bulletin of the American Mathematical Society* 53.11 (1947), pp. 1021–1099.

[95] Shusen Wang and Zhihua Zhang. "Efficient Algorithms and Error Analysis for the Modified Nyström Method". In: *arXiv preprint arXiv:1404.0138* (2014).

[96] Shusen Wang and Zhihua Zhang. "Improving CUR matrix decomposition and the Nyström approximation via adaptive sampling". In: *Journal of Machine Learning Research* 14.1 (2013), pp. 2729–2769.

[97] J. H. Wilkinson. "ERROR ANALYSIS OF DIRECT METHODS OF MATRIX INVERSION". In: *J. ACM* 10 (1961), pp. 281–330.

[98] Christopher Williams and Matthias Seeger. "Using the Nyström method to speed up kernel machines". In: *Proceedings of the 14th Annual Conference on Neural Information Processing Systems*. EPFL-CONF-161322. 2001, pp. 682–688.

[99] David P Woodruff. "Sketching as a tool for numerical linear algebra". In: *arXiv preprint arXiv:1411.4357* (2014).

[100] Stephen J. Wright. "A Collection of Problems for Which Gaussian Elimination with Partial Pivoting is Unstable". In: *SIAM J. Sci. Comput.* 14.1 (Jan. 1993), pp. 231–238. ISSN: 1064-8275. DOI: 10.1137/0914013. URL: http://dx.doi.org/10.1137/0914013.

[101] H. Zha Z. Zhang and H. Simon. "Low-rank approximations with sparse factors I: Basic algorithms and error analysis". In: *SIAM J. Matrix Anal. Appl.* 23:706-727 (2002).

[102] H. Zha Z. Zhang and H. Simon. "Low-rank approximations with sparse factors II: Penalized methods with discrete Newton-like iterations". In: *SIAM J. Matrix Anal. Appl.* 25:901-920 (2004).

[103] Kai Zhang and James T Kwok. "Clustered Nyström method for large scale manifold learning and dimension reduction". In: *Neural Networks, IEEE Transactions on* 21.10 (2010), pp. 1576–1587.

[104] Kai Zhang et al. "Scaling up kernel svm on limited resources: A low-rank linearization approach". In: *International Conference on Artificial Intelligence and Statistics*. 2012, pp. 1425–1434.

[105] Youwei Zhang and Laurent E. Ghaoui. "Large-Scale Sparse Principal Component Analysis with Application to Text Data". In: *Advances in Neural Information Processing Systems 24*. 2011, pp. 532–539. URL: http://papers.nips.cc/paper/4337-large-scale-sparse-principal-component-analysis-with-application-to-text-data.pdf.