Title
Convergence Analysis of Randomized Block Lanczos Algorithms for Low-Rank Matrix Approximations

Author
Yuan, Qiaochu

Publication Date
2018

Peer reviewed|Thesis/dissertation

Convergence Analysis of Randomized Block Lanczos Algorithms for Low-Rank Matrix Approximations

by

Qiaochu Yuan

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Applied Mathematics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Ming Gu, Chair
Professor James Demmel
Professor Katherine Yelick

Summer 2018

Convergence Analysis of Randomized Block Lanczos Algorithms for Low-Rank Matrix Approximations

Abstract

Convergence Analysis of Randomized Block Lanczos Algorithms for Low-Rank Matrix Approximations

by

Qiaochu Yuan

Doctor of Philosophy in Applied Mathematics

University of California, Berkeley

Professor Ming Gu, Chair

The low rank approximation of matrices is a crucial component in many data mining applications today. A competitive algorithm for this class of problems is the randomized block Lanczos algorithm - an amalgamation of the traditional block Lanczos algorithm with a randomized component appearing in the form of a randomized starting matrix. While empirically this algorithm performs quite well, there has been scant new theoretical results on its convergence behavior and approximation accuracy, and past results have been restricted to certain parameter settings. In this thesis, we present a unified convergence analysis for this algorithm, for all valid choices of the block size parameter. We give an overview of how the Lanczos algorithm has developed historically and how past and adjacent results in the convergence analysis of these algorithms tie in with the current work. We present novel results on the rate of singular value convergence and show that under certain spectrum regimes, the convergence is superlinear. Additionally, we provide results from numerical experiments that validate our analysis.

To my parents and grandparents,
with all my love.

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I would like to thank my advisor, Prof. Ming Gu. who has shared with me his knowledge of mathematics and his wisdom on how to approach the mathematical process. I am grateful for the many whiteboard discussions we had, and equally grateful for our other conversations regarding career and life. Without his guidance this work would not have been possible.

I would like to thank Prof. James Demmel, Prof. Per-Olof Persson, Prof. John Strain, Prof. Craig Evans, Prof. Katherine Yelick, and Prof. Jon Wilkening for their superb teaching, which helped deepen my understanding of several interesting areas of applied and computational mathematics.

I would like to thank the staff of the Berkeley math department - in particular Barb, Marsha, and Vicky - for helping to take care of the graduate students and for making navigating the department a little bit easier. I would like to thank Chris, Dave, Danny, Jianwei, Ruochen, Bo, Frank, Albert, Minseon, Catherine, Dominique, Yael, and other colleagues in the department. I am glad to have had insightful mathematical and personal discussions with them and they have made the last five years memorable. I would also like to thank Christine, Ania, Thomas, Tegan, Armeen, Bia, and Hijung - physical distance has not diminished the importance of their support or my affection for them.

Finally I would like to thank my parents, who raised me with love and showed me by example the importance of curiosity and hard work. They taught me that learning is a lifelong pursuit. My dad instilled in me a love of mathematics and all things precise, and my mom taught me not only to seek solutions, but also to ask questions. This work is dedicated to them.

Chapter 1

Introduction and Background

Applications of the singular value decomposition abound in traditional areas of scientific computing [3, 22, 1]. More recently, the SVD, and its truncated and approximate variants, have increasingly found uses in big-data applications as well [28, 47, 7, 20]. In this area, the truncated SVD, in particular, plays a role as both a stand-alone matrix processing technique (PCA [9]), and as a computational kernel incorporated into more complex algorithms, such as proximal gradient algorithms applied to objective functions involving the nuclear norm [45]. In recent years, there has been increasing need for, research in, and adoption of fast and accurate approximate k-truncated SVD algorithms in particular [25, 27], and low-rank matrix approximation algorithms in general [15, 2].

While techniques for computing the exact k-truncated SVD have existed and been implemented since the 1960s [24], it has been observed that the recent applications to big-data problems differ in both the computation efficiency requirement and the accuracy requirement of the algorithms. Firstly, whereas traditional scientific computing problems involving the truncated SVD are often applied to moderately sized matrices, big-data problems have much higher computational efficiency demands - the matrices in question are often extraordinarily large, on the order of 10^9 in both dimensions as they may semantically represent quantities such as the number of users or webpages in large, web-scale data sets [44, 12]. Secondly, while the truncated SVD may be the final desired object for scientific computing questions, for big-data applications, it is usually an intermediate representation for the overall classification or regression task. Empirically, the final accuracy of the task only weakly depends on the accuracy of the matrix approximation. Thus, while previous variants of truncated SVD algorithms focused on computing up to full double precision, newer iterations of the algorithm aimed at big-data applications can comfortably get by with only 2-3 digits of accuracy [25].

Both of these observations have driven the development and incorporation of randomization components into traditional SVD algorithms such as power iteration, subspace iteration, and Lanczos and other Krylov subspace family methods [27, 25, 33]. These methods, incorporating either a randomized sketching or projecting operation on the original matrix, have in practice been shown to effectively balance reducing the number of operations needed

to produce the approximation with the probability of producing an acceptably accurate approximation.

In this chapter, we review the historical algorithmic developments and ideas that led to the Randomized Block Lanczos algorithm, which is the subject of this thesis. In the subsequent chapters, we first provide the numerical linear algebraic tools required to derive our main result, then we state and prove a novel convergence result for the algorithm in question. Finally, we use our result as a jumping off point for a practical discussion of the various parameter choices for the algorithm, with accompanying numerical experiments.

Notational Conventions

We establish some notational conventions used in the exposition of this thesis.

We denote matrices by bold-faced uppercase letters, e.g. $\mathbf{A}$, entries of matrices by the plain-faced lowercase letter that the entry belongs to, e.g. a_{11}, and block submatrices by the bold-faced or script-faced uppercase letter that the submatrix belongs to, subscripted by position e.g. $\mathbf{A}_{11}$, $\mathcal{A}_{11}$ or $\mathbf{A}_{a\times b}$. Double numerical subscripts denote the position of the element or the submatrix, i.e. $\mathbf{A}_{11}$ and a_{11} are the topmost leftmost subblock or entry of $\mathbf{A}$ respectively. $m \times n$ subscripts denote the dimensions of a submatrix, when such information is relevant, i.e. $\mathbf{A}_{a\times b}$ denote a subblock of $\mathbf{A}$ that has dimensions $a \times b$.

Constants are denoted by script-faced uppercase or lowercase letters, e.g. $\mathcal{C}$ or α, when it is asymptotically insignificant, i.e. constant with respect to the convergence parameter.

The tilde ($\sim$) accent is generally used to denote approximation to the quantity it decorates, and the parenthetical superscript is generally used to denote an iteration number for intermediary values of an iterative algorithm, e.g. $\widetilde{\sigma}_j^{(q)}$ denotes the value of $\widetilde{\sigma}_j$ at the qth iteration of some algorithm, which itself denotes that it is meant to be an approximation to the value of σ_j.

1.1 Lanczos Iteration Algorithms

The Lanczos algorithm is a classical numerical linear algebra algorithm with a long and illustrious history. In its original formulation by Lanczos in 1960, it was conceived as an iterative algorithm to compute the extremal eigenvalue-eigenvector pairs of symmetric matrices [30]. Since then, numerous variants of the original algorithm - blocked, randomized, nonsymmetric, parallelized, etc - have been developed [38, 13, 10], and many aspects of the algorithm - convergence of eigenvalues, vector, and spaces, numerical stability, etc - has been analyzed [40, 34, 35, 31]. Its popularity stems in part from the ever-growing sizes of matrix problems - generally, the only operations required in Lanczos iteration algorithms are multiplications by the target matrix, and in particular, explicit representation of the matrix is not necessary. This characteristic makes these algorithms ideal for use on large, sparse matrices arising from problems ranging from scientific computing to data science, especially if only the extremal eigenpairs are of interest. Today, variants of the Lanczos algorithm

Table 1.1: Outline of classical Lanczos Iteration

(1). Select an initial vector $\mathbf{v}$.
(2). Construct the **Krylov** subspace $\mathcal{K}(\mathbf{A}, \mathbf{v}, k) = \text{span}\{\mathbf{v}, \mathbf{A}\mathbf{v}, \cdots, \mathbf{A}^k\mathbf{v}\}$.
(3). Restrict and project $\mathbf{A}$ to the Krylov subspace to form $\mathbf{T} = \text{proj}_{\mathcal{K}}\mathbf{A}\big|_{\mathcal{K}}$.
(4). Compute the eigenvalues $\widetilde{\lambda}_1, \cdots, \widetilde{\lambda}_k$ and eigenvectors $\widetilde{\mathbf{u}}_1, \cdots, \widetilde{\mathbf{u}}_k$ of $\mathbf{T}$ as the desired approximations for the first k eigenvalues $\lambda_1, \cdots, \lambda_k$ and eigenvectors $\mathbf{u}_1, \cdots, \mathbf{u}_k$ of $\mathbf{A}$.

are widely used in diverse application domains, and their optimized and numerically stable implementations are available in many software packages [5, 19, 4].

Formally, given a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ with eigenvalues $\lambda_1 > \lambda_2 > \cdots > \lambda_n$ and associated eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n$, the classical Lanczos algorithm provides a way of constructing approximations $\widetilde{\lambda}_1, \cdots, \widetilde{\lambda}_k$ and $\widetilde{\mathbf{u}}_1, \cdots, \widetilde{\mathbf{u}}_k$ to the first k eigenpairs. In typical applications, the number of sought eigenpairs k is much smaller than the dimension of the matrix n.

Mathematically, this class of algorithms work by successively approximating the action of the linear operator represented by $\mathbf{A}$ on a series of specially constructed lower dimensional spaces. The mathematical outline of the steps are listed in Table 1.1

As computations with respect to matrices, the third (3) step above requires the construction of an orthonormal basis $\mathbf{Q}_k = \begin{bmatrix}\mathbf{q}_1 & \cdots & \mathbf{q}_{k+1}\end{bmatrix}$ for the Krylov subspace $\mathcal{K}$, derived from orthonormalizing the columns of the Krylov subspace matrix $\mathbf{K}_k = \begin{bmatrix}\mathbf{v} & \mathbf{A}\mathbf{v} & \cdots & \mathbf{A}^k\mathbf{v}\end{bmatrix}$. Then, the restriction and projection operation can be expressed as

$$\mathbf{T}_k = \mathbf{Q}_k^T \mathbf{A} \mathbf{Q}_k \tag{1.1}$$

where it can be shown that $\mathbf{T}_k$ takes a tridiagonal form

$$\mathbf{T}_k = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_k \\ & & \beta_k & \alpha_{k+1} \end{bmatrix} \tag{1.2}$$

At every iteration, a new basis vector $\mathbf{q}_j$, and entries α_j, β_j are computed using the relation

$$\mathbf{A}\begin{bmatrix}\mathbf{q}_1 & \cdots & \mathbf{q}_j\end{bmatrix} = \left[\begin{array}{ccc|c} \mathbf{q}_1 & \cdots & \mathbf{q}_j & \mathbf{q}_{j+1} \end{array}\right] \left[\begin{array}{cccc} \alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{j-1} \\ \hline & & \beta_{j-1} & \alpha_j \\ & & & \beta_j \end{array}\right] \tag{1.3}$$

and the three-term recurrence equation

$$\mathbf{A}\mathbf{q}_j = \beta_{j-1}\mathbf{q}_{j-1} + \alpha_j\mathbf{q}_j + \beta_j\mathbf{q}_{j+1} \tag{1.4}$$

The classical Lanczos iteration algorithm is given in Alg. 1, corresponding to steps (1)-(3) in Table 1.1. The output of this algorithm are the values $\alpha_1, \cdots, \alpha_{k+1}$ and $\beta_1, \cdots \beta_k$, which together describe the matrix $\mathbf{T}_k$ as represented in Eqn. (1.2). In terms of computational complexity, the bulk of the floating-point operations occur in the matrix-vector multiplication in step (6).

Step (4) is usually carried out via an eigenvalue algorithm such as QR iteration - because $k \ll n$, this post-processing step is performed on a matrix much smaller than the original and its computational cost is considered to be asymptotically insignificant when compared with that of running Alg. 1.

Algorithm 1 Classical Lanczos iteration algorithm for tridiagonalization

Input: symmetric $\mathbf{A} \in \mathbb{R}^{n\times n}$, initial vector $\mathbf{v} \in \mathbb{R}^{n\times 1}$, target rank k.
Output: α_j, β_j, diagonal and subdiagonal elements of the approximation $\mathbf{T}_k \in \mathbb{R}^{(k+1)\times(k+1)}$

1: Initialize $j \leftarrow 0$, $\beta_0 \leftarrow 1$, $\mathbf{q}_0 \leftarrow \mathbf{0}$, $\mathbf{r}_0 \leftarrow \mathbf{v}$.
2: **while** $j \leq k$ and $\beta_j \neq 0$ **do**
3: $\quad \mathbf{q}_{j+1} \leftarrow \frac{\mathbf{r}_j}{\beta_j}$
4: $\quad j \leftarrow j + 1$
5: $\quad \alpha_j \leftarrow \mathbf{q}_j^T \mathbf{A}\mathbf{q}_j$
6: $\quad \mathbf{r}_j \leftarrow (\mathbf{A} - \alpha_j\mathbf{I})\,\mathbf{q}_j - \beta_{j-1}\mathbf{q}_{j-1}$
7: $\quad \beta_j = \|\mathbf{r}_j\|_2$
8: **end while**

We pause here to note that Alg. 1 is best seen as a description of the algorithm in exact arithmetic, and that a successful implementation of the Lanczos iteration algorithm involve numerous additional considerations, such as numerical stability and roundoff properties.

Block Lanczos Iteration Algorithm

While the classical Lanczos Iteration Algorithm already exhibits, both theoretically and empirically, desirable convergence properties, it is known that this algorithm is unable to capture eigenvalues with multiplicity strictly greater than one - the algorithm acts as if all eigenvalues are simple [40]. In addition, it has been well-established that on modern computing hardware, computational speed improve greatly from reorganizing computations as BLAS level-3 operations due to cache efficiency and data reuse. These considerations motivated the development and analysis of the blocked version of the Lanczos Iteration algorithm [13].

The block generalization of the Lanczos Iteration algorithm uses, instead of a single initial vector $\mathbf{v}$, a block of b vectors $\mathbf{V} = \begin{bmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_b\end{bmatrix}$ and builds the Krylov subspace in q iterations as $\mathcal{K}(\mathbf{A}, \mathbf{V}, q) = \text{span}\{\mathbf{V}, \mathbf{A}\mathbf{V}, \cdots, \mathbf{A}^q\mathbf{V}\}$.

The matrix representing the operator $\mathbf{T} = \text{proj}_{\mathcal{K}}\mathbf{A}\big|_{\mathcal{K}}$ is a $(q+1)b \times (q+1)b$ matrix $\mathbf{T}_q$ of block tridiagonal form

$$\mathbf{T}_q = \begin{bmatrix} \mathcal{A}_1 & \mathcal{B}_1^T & & \\ \mathcal{B}_1 & \ddots & \ddots & \\ & \ddots & \ddots & \mathcal{B}_q^T \\ & & \mathcal{B}_q & \mathcal{A}_{q+1} \end{bmatrix} \tag{1.5}$$

Similarly to the classical Lanczos Iteration algorithm, at each iteration, a new block of b basis vectors $\mathcal{Q}_j$, and $b \times b$ blocks $\mathcal{A}_j$, $\mathcal{B}_j$ are computed using the relation

$$\mathcal{A}\begin{bmatrix}\mathbf{Q}_1 & \cdots & \mathbf{Q}_j\end{bmatrix} = \left[\begin{array}{ccc|c}\mathbf{Q}_1 & \cdots & \mathbf{Q}_j & \mathbf{Q}_{j+1}\end{array}\right] \left[\begin{array}{cccc} \mathcal{A}_1 & \mathcal{B}_1^T & & \\ \mathcal{B}_1 & \ddots & \ddots & \\ & \ddots & \ddots & \mathcal{B}_{j-1}^T \\ & & \mathcal{B}_{j-1} & \mathcal{A}_j \\ \hline & & & \mathcal{B}_j \end{array}\right] \tag{1.6}$$

and the three-term recurrence equation

$$\mathbf{A}\mathbf{Q}_j = \mathbf{Q}_{j-1}\mathcal{B}_{j-1}^T + \mathbf{Q}_j\mathcal{A}_j + \mathbf{Q}_{j+1}\mathcal{B}_j \tag{1.7}$$

The Block Lanczos Iteration Algorithm appears in Alg. 2. We draw attention to the different algorithm parameters between Alg. 1 and Alg. 2. In the non-blocked version of the algorithm, the target rank k is required as a way to implicitly determine the number of iterations (= the dimension of the Krylov subspace) to run. In the original formulation of the algorithm, this is fixed at k (i.e. search for approximate eigenpairs in a Krylov subspace of dimension exactly k); however, the algorithm can be run for any number of iterations $q \geq k$. The accuracy of the approximate eigenpairs may improve by running a larger number of iterations, as this allows the original $\mathbf{A}$ to be approximated in a larger subspace. In the blocked version of the algorithm, a choice of the number of iterations q results in general in the construction of a Krylov subspace of dimension qb, and, although the original formulation of the algorithm suggests a choice of $b = k$ and "a few" iterations, we note that the algorithm is valid for any choices of b and q such that $bq \geq k$. For both of these algorithms, selecting the "right" number of iterations q is tightly connected with the convergence theory of these algorithms, which we will discuss shortly.

The blocks $\mathbf{Q}_j \in \mathbb{R}^{n\times b}$ and $\mathcal{B}_j \in \mathbb{R}^{b\times b}$ are produced as a result of performing QR factorization on the residual matrices $\mathbf{R}_j \in \mathbb{R}^{n\times b}$. This implies that the $\mathcal{B}_j$s are upper triangular matrices in and of themselves, and so $\mathbf{T}_q$ in Eqn. (1.5) is in fact a banded matrix with bandwidth $2b-1$. The bulk computation cost of this algorithm comes from the matrix-matrix multiplication performed in step 6. To carry out step (4) in Table 1.1, this matrix $\mathbf{T}_q$ is then further band-reduced to tridiagonal before its eigenvalues and eigenvectors are found by running an eigenvalue algorithm such as QR iteration.

Previous comments regarding the numerical stability of Alg. 1 apply also to Alg. 2. In particular, modifications made to the classical Lanczos iteration are typically also applied to the block Lanczos algorithm [39].

Algorithm 2 Blocked Lanczos iteration algorithm for block tridiagonalization

Input: symmetric $\mathbf{A} \in \mathbb{R}^{n\times n}$, initial block matrix $\mathbf{V} \in \mathbb{R}^{n\times b}$, number of iterations q such that $(q+1)b \geq k$.

Output: $\mathcal{A}_j, \mathcal{B}_j$, diagonal and subdiagonal $b \times b$ block matrices of the approximation $\mathbf{T}_q \in \mathbb{R}^{(q+1)b\times(q+1)b}$

1: Initialize $j \leftarrow 0$, $\mathcal{B}_0 \leftarrow \mathbf{I}$, $\mathbf{Q}_0 \leftarrow \mathbf{0}$, $\mathbf{R}_0 \leftarrow \mathbf{V}$.
2: **while** $j \leq q$ **do**
3: $\quad \mathbf{Q}_{j+1}\mathcal{B}_j \leftarrow \mathrm{qr}\,(\mathbf{R}_j)$
4: $\quad j \leftarrow j+1$
5: $\quad \mathcal{A}_j \leftarrow \mathbf{Q}_j^T\mathbf{A}\mathbf{Q}_j$
6: $\quad \mathbf{R}_j \leftarrow \mathbf{A}\mathbf{Q}_j - \mathbf{Q}_j\mathcal{A}_j - \mathbf{Q}_{j-1}\mathcal{B}_{j-1}^T$
7: **end while**

Table 1.2: Outline of classical Lanczos Iteration for the SVD problem

(1). Select an initial vector $\mathbf{v}$.
(2). Construct the **Krylov** subspaces $\mathcal{K}_V(\mathbf{A}^T\mathbf{A}, \mathbf{v}, k) = \mathrm{span}\{\mathbf{v}, \mathbf{A}^T\mathbf{A}\mathbf{v}, \cdots, (\mathbf{A}^T\mathbf{A})^k\mathbf{v}\}$ and $\mathcal{K}_U(\mathbf{A}\mathbf{A}^T, \mathbf{A}\mathbf{v}, k) = \mathrm{span}\{\mathbf{A}\mathbf{v}, \left(\mathbf{A}\mathbf{A}^T\right)\mathbf{A}\mathbf{v}, \cdots, (\mathbf{A}\mathbf{A}^T)^k\mathbf{A}\mathbf{v}\}$.
(3). Restrict and project $\mathbf{A}$ to the Krylov subspaces to form $\mathbf{B} = \mathrm{proj}_{\mathcal{K}_U}\mathbf{A}\big|_{\mathcal{K}_V}$.
(4). Compute the singular values $\widetilde{\sigma}_1, \cdots, \widetilde{\sigma}_k$ and left and right singular vectors $\widetilde{\mathbf{u}}_1, \cdots, \widetilde{\mathbf{u}}_k$ and $\widetilde{\mathbf{v}}_1, \cdots, \widetilde{\mathbf{v}}_k$ of $\mathbf{B}$ as the desired approximations for the first k singular values $\sigma_1, \cdots, \sigma_k$ and left and right singular vectors $\mathbf{u}_1, \cdots, \mathbf{u}_k$ and $\mathbf{v}_1, \cdots, \mathbf{v}_k$ of $\mathbf{A}$.

Lanczos Iteration Algorithm for the Singular Value Problem

Parallel to the development of various Lanczos tridiagonalization algorithms for the symmetric eigenvalue problem, an analogous class of bidiagonalization algorithms was developed for the SVD problem for arbitrary matrices [24, 36, 21].

The connection between the singular value decomposition of a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and the eigen decomposition of the symmetric matrices $\mathbf{A}\mathbf{A}^T \in \mathbb{R}^{m\times m}$ and $\mathbf{A}^T\mathbf{A} \in \mathbb{R}^{n\times n}$ is well known. In particular, the eigenspaces of $\mathbf{A}\mathbf{A}^T$ and $\mathbf{A}^T\mathbf{A}$ are the left and right singular spaces, respectively, of the original matrix $\mathbf{A}$. The analogous SVD algorithm to the classical Lanczos iteration algorithm constructs Krylov subspaces to approximate the left and right singular spaces of $\mathbf{A}$. The mathematical steps are given in Table 1.2.

Note that while the mathematical description of the algorithm makes use of the subspaces associated with $\mathbf{A}\mathbf{A}^T$ and $\mathbf{A}^T\mathbf{A}$, the preferred matrix to work with is

$$\mathbf{C} = \begin{bmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{bmatrix} \tag{1.8}$$

Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ be a "tall-skinny" matrix, with $m \geq n$ - in the situation of $m < n$, the

following exposition applies to its transpose, $\mathbf{A}^T$. Let $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ be the SVD of $\mathbf{A}$, where $\mathbf{U} \in \mathbb{R}^{m\times n}$, $\mathbf{V} \in \mathbb{R}^{n\times n}$ are column orthogonal matrices and $\mathbf{\Sigma} \in \mathbb{R}^{n\times n}$ is a diagonal matrix. (See Subsection 3.1, Thm 3.1.2 for more details on the SVD). Then, it can be shown that $\mathbf{C} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$ is the eigen decomposition of $\mathbf{C}$, for

$$\mathbf{Q} = \frac{1}{\sqrt{2}}\begin{bmatrix}\mathbf{V} & \mathbf{V} & \mathbf{0} \\ \mathbf{U} & -\mathbf{U} & \sqrt{2}\mathbf{U}^{\perp}\end{bmatrix} \tag{1.9}$$

$$\mathbf{\Lambda} = \begin{bmatrix}\sigma_1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & \sigma_n & & & & & & \\ & & & -\sigma_1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & -\sigma_n & & & \\ & & & & & & 0 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 0\end{bmatrix} \tag{1.10}$$

As computations with respect to matrices, the third (3) step in Table 1.2 requires the construction of orthonormal basis $\mathbf{U}_k = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_{k+1}\end{bmatrix}$ and $\mathbf{V}_k = \begin{bmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_{k+1}\end{bmatrix}$ for the $\mathcal{K}_U$ and $\mathcal{K}_V$ Krylov subspaces. These bases are derived from orthonormalizing the columns of the Krylov subspace matrices $\mathbf{K}_U = \begin{bmatrix}\mathbf{A}\mathbf{v} & \cdots & (\mathbf{A}\mathbf{A}^T)^k\mathbf{A}\mathbf{v}\end{bmatrix}$ and $\mathbf{K}_V = \begin{bmatrix}\mathbf{v} & \cdots & (\mathbf{A}^T\mathbf{A})^k\mathbf{v}\end{bmatrix}$ respectively.

Writing the restriction and projection operation as

$$\mathbf{B}_k = \mathbf{U}_k^T\mathbf{A}\mathbf{V}_k \tag{1.11}$$

we can show that the matrix representing the operator $\mathbf{B} = \text{proj}_{\mathcal{K}_U}\mathbf{A}\big|_{\mathcal{K}_V}$ is bidiagonal of the form

$$\mathbf{B}_k = \begin{bmatrix}\alpha_1 & & & \\ \beta_1 & \alpha_1 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_{k+1}\end{bmatrix} \tag{1.12}$$

At every iteration, new basis vectors $\mathbf{u}_j$, $\mathbf{v}_j$, and entries α_j, β_j are computed using the

relations

$$\mathbf{A}\begin{bmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_j\end{bmatrix} = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_j\end{bmatrix}\begin{bmatrix}\alpha_1 & \beta_1 & & \\ & \ddots & \ddots & \\ & & \ddots & \beta_{j-1} \\ & & & \alpha_j\end{bmatrix} \tag{1.13}$$

$$\mathbf{A}^T\begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_j\end{bmatrix} = \left[\begin{array}{ccc|c}\mathbf{v}_1 & \cdots & \mathbf{v}_j & \mathbf{v}_{j+1}\end{array}\right]\left[\begin{array}{cccc}\alpha_1 & & & \\ \beta_1 & \ddots & & \\ & \ddots & \ddots & \\ \hline & & \beta_{j-1} & \alpha_j \\ & & & \beta_j\end{array}\right] \tag{1.14}$$

and the recurrences

$$\mathbf{A}\mathbf{v}_j = \alpha_j\mathbf{u}_j + \beta_{j-1}\mathbf{u}_{j-1} \tag{1.15}$$

$$\mathbf{A}^T\mathbf{u}_j = \alpha_j\mathbf{v}_j + \beta_j\mathbf{v}_{j+1} \tag{1.16}$$

The algorithms for classical Lanczos iteration for SVD, also known as the Golub-Kahan bidiagonalization algorithm, and its blocked variant appear in Alg. 3 and Alg. 4 respectively.

Algorithm 3 Lanczos tridiagonalization algorithm for SVD (Golub-Kahan bidiagonalization)

Input: $\mathbf{A} \in \mathbb{R}^{m\times n}$, initial vector $\mathbf{v} \in \mathbb{R}^{n\times 1}$, target rank k.
Output: α_j, β_j, diagonal and subdiagonal entries of the approximation $\mathbf{B}_k \in \mathbb{R}^{(k+1)\times(k+1)}$

1: Initialize $j \leftarrow 0$, $\beta_0 \leftarrow 1$, $\mathbf{u}_0 \leftarrow \mathbf{0}$, $\mathbf{p}_0 \leftarrow \mathbf{v}$.
2: **while** $j \leq k$ and $\beta_j \neq 0$ **do**
3: $\quad\mathbf{v}_{j+1} = \frac{\mathbf{p}_j}{\beta_j}$
4: $\quad j \leftarrow j + 1$
5: $\quad\mathbf{r}_j \leftarrow \mathbf{A}\mathbf{v}_j - \beta_{j-1}\mathbf{u}_{j-1}$
6: $\quad\alpha_j \leftarrow \|\mathbf{r}_j\|_2$
7: $\quad\mathbf{u}_j \leftarrow \frac{\mathbf{r}_j}{\alpha_j}$
8: $\quad\mathbf{p}_j \leftarrow \mathbf{A}^T\mathbf{u}_j - \alpha_j\mathbf{v}_j$
9: $\quad\beta_j \leftarrow \|\mathbf{p}_j\|_2$
10: **end while**

The dominating computation costs are, again, the matrix-vector multiplications performed in steps 5 and 8 of Alg. 3 and the matrix-matrix multiplications performed in steps 5 and 7 of Alg. 4.

Algorithm 4 Blocked Lanczos tridiagonalization algorithm for SVD

Input: $\mathbf{A} \in \mathbb{R}^{m\times n}$, initial block matrix $\mathbf{V} \in \mathbb{R}^{n\times b}$, number of iterations q, such that $(q+1)b \geq k$.

Output: $\mathcal{A}_j, \mathcal{B}_j$, diagonal and subdiagonal $b \times b$ block matrices of the approximation $\mathbf{B}_q \in \mathbb{R}^{(q+1)b\times(q+1)b}$

1: Initialize $j \leftarrow 0$, $\mathcal{B}_0 \leftarrow \mathbf{I}$, $\mathbf{U}_0 \leftarrow \mathbf{0}$, $\mathbf{P}_0 \leftarrow \mathbf{V}$.
2: **while** $j \leq q$ **do**
3: $\quad \mathbf{V}_{j+1}\mathcal{B}_j \leftarrow \mathrm{qr}\,(\mathbf{P}_j)$
4: $\quad j \leftarrow j+1$
5: $\quad \mathbf{R}_j \leftarrow \mathbf{A}\mathbf{V}_j - \mathbf{U}_{j-1}\mathcal{B}_{j-1}^T$
6: $\quad \mathbf{U}_j\mathcal{A}_j \leftarrow \mathrm{qr}\,(\mathbf{R}_j)$
7: $\quad \mathbf{P}_j \leftarrow \mathbf{A}^T\mathbf{U}_j - \mathbf{V}_j\mathcal{A}_j$
8: **end while**

1.2 Convergence Theory for Lanczos Iteration Algorithms

All variants of the Lanczos iteration algorithms are iterative procedures. Characteristic of such algorithms, they may be run for a user-specified number of steps q, and the results are computed in a successive iteration-by-iteration manner. In particular, for the algorithms we discussed in the previous section, the Krylov subspaces are successively expanded by 1 (resp. b) dimension(s) in each step of the classical (resp. blocked) Lanczos procedure. Furthermore both the accuracy of the approximate eigenvalues and eigenvectors and the computational cost scale with the number of iterations. Therefore, it is vitally important to analyze the convergence behaviors of these algorithms, and the convergence results give answers - at least theoretically - to a variety of questions ranging from the suitability of these algorithms for matrices with different spectrum behaviors, to guidelines on selecting values for the q and b parameters.

In general, convergence theory seeks to answer the following question: how well do the approximate eigenvalues $\widetilde{\lambda}_i^{(q)}$ approximate the actual eigenvalues λ_i, for $i = 1, \cdots, k$? Historically, answers to this question took the form of an inequality bound on the error of the ith approximate singular value:

$$0 \leq \lambda_i - \widetilde{\lambda}_i^{(q)} \leq (\lambda_i - \lambda_n)\{\text{some convergence factor}\} \tag{1.17}$$

where {some convergence factor} dictates the asymptotic convergence behavior of the particular variant of the Lanczos iteration algorithm, and is functionally composed of two multiplicative terms

- one involving q, the number of iterations - typically appearing in the degree of a Chebyshev polynomial term, and

- one involving $\mathbf{v}$ or $\mathbf{V}$, the initial starting space - quantifying the alignment between this space and the relevant eigenspace.

Below we highlight and briefly discuss some representative theoretical convergence results for Algs. 1 and 2. Analogous results for Algs. 3 and 4 hold as well.

We begin with a result by Kaniel and Page.

Theorem 1.2.1 ([37]). *Let $\widetilde{\lambda}_j^{(k)}$ be the jth eigenvalue of the matrix $\mathbf{T}_k$ that results from running Alg. 1, and let $\mathcal{P}_j$ be the projection operator associated with the subspace of the jth eigenvector $\mathbf{u}_j$. Then, if $\mathcal{P}_1\mathbf{v} \neq 0$,*

$$0 \leq \lambda_1 - \widetilde{\lambda}_1^{(k)} \leq (\lambda_1 - \lambda_n) \frac{\tan^2 \theta(\mathbf{u}_1, \mathbf{v})}{T_{k-1}^2(\gamma_1)} \tag{1.18}$$

and for $j = 2, \cdots, k$, if $\mathcal{P}_j\mathbf{v} \neq 0$,

$$0 \leq \lambda_j - \widetilde{\lambda}_j^{(k)} \leq \left(\frac{K}{T_{k-j}(\gamma_j)} \right)^2 + \sum_{i=1}^{j-1} (\lambda_i - \lambda_n) \left\| (\mathcal{I} - \mathcal{P}_i) \widetilde{\mathbf{u}}_i^{(k)} \right\|^2 \tag{1.19}$$

where $T_i(x)$ is the Chebyshev polynomial of degree i,

$$\gamma_j = 1 + 2 \cdot \frac{\lambda_j - \lambda_{j+1}}{\lambda_{j+1} - \lambda_n} \tag{1.20}$$

and K is a constant depending on the eigen-spectrum of $\mathbf{A}$ and $\mathbf{v}$.

Eqn. (1.18) is the convergence bound for the 1st eigenvalue. The factor that determines the convergence is $\frac{\tan^2 \theta(\mathbf{u}_1, \mathbf{v})}{T_{n-1}^2(\gamma_1)}$ - it is composed of a Chebyshev polynomial factor, of which the number of iterations k appear in the degree,

$$T_{k-1}(\gamma_1) \simeq \frac{1}{2} \left(\gamma_1 + \sqrt{\gamma_1^2 - 1} \right)^{-k+1} \tag{1.21}$$

and an alignment term, which measures the magnitude of the component of the initial vector $\mathbf{v}$ in the direction of the first eigenvector $\mathbf{u}_1$,

$$\tan \theta(\mathbf{u}_1, \mathbf{v}) = \frac{\|(\mathcal{I} - \mathcal{P}_1)\mathbf{v}\|}{\|\mathcal{P}_1\mathbf{v}\|} \tag{1.22}$$

Eqn. (1.19) gives the convergence inequality for the remaining eigenvalues $j = 2, \cdots, k$. However, notice that the right-hand side does not directly reference λ_j, and so the quality of this bound depends on the decay of the spectrum. Moreover, the summation term on the right-hand side is hard to estimate and may be non-negligible.

The above theorem was generalized to a bound for all k eigenvalues by Saad, and appears below.

Theorem 1.2.2 ([40]). *Let $\widetilde{\lambda}_j^{(k)}$ be the jth eigenvalue of the matrix $\mathbf{T}_k$ that results from running Alg. 1, and let $\mathcal{P}_j$ be the projection operator associated with the subspace of the jth eigenvector $\mathbf{u}_j$. Then, for $j = 1, \cdots, k$, if $\mathcal{P}_j \mathbf{v} \neq 0$,*

$$0 \leq \lambda_j - \widetilde{\lambda}_j^{(k)} \leq (\lambda_j - \lambda_n) \left(\frac{L_i^{(k)} \tan\theta(\mathbf{u}_j, \mathbf{v})}{T_{k-j}(\gamma_j)} \right)^2 \tag{1.23}$$

where $T_i(x)$ is the Chebyshev polynomial of degree i, $\theta(\cdot,\cdot)$ is the angle between two vectors, and

$$\gamma_j = 1 + 2 \cdot \frac{\lambda_j - \lambda_{j+1}}{\lambda_{j+1} - \lambda_n} \tag{1.24}$$

$$L_j^{(k)} = \begin{cases} \prod_{i=1}^{j-1} \frac{\widetilde{\lambda}_i^{(k)} - \lambda_n}{\widetilde{\lambda}_i^{(k)} - \lambda_j} & \text{if } j \neq 1 \\ 1 & \text{if } j = 1 \end{cases} \tag{1.25}$$

Now, we briefly review some representative convergence bounds for the blocked Lanczos algorithm, Alg. 2.

We begin with a result by Underwood.

Theorem 1.2.3 ([48]). *Let $\widetilde{\lambda}_j^{(q)}$ be the jth eigenvalue of the block matrix $\mathbf{T}_q$ that results from running Alg. 2. Let $\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_b \end{bmatrix}$ be a block matrix whose columns are the b eigenvectors associated with $\lambda_1 \geq \cdots \geq \lambda_b$. If $\mathbf{U}^T\mathbf{V}$ is of full rank b, then for $j = 1, \cdots, b$*

$$0 \leq \lambda_j - \widetilde{\lambda}_j^{(q)} \leq (\lambda_1 - \lambda_n) \frac{\tan^2\Theta(\mathbf{U}, \mathbf{V})}{T_{q-1}^2(\rho_j)} \tag{1.26}$$

where $T_i(x)$ is the Chebyshev polynomial of degree i, $\cos\Theta(\mathbf{U}, \mathbf{V}) = \sigma_{\min}(\mathbf{U}^T\mathbf{V})$, and

$$\rho_j = 1 + 2 \cdot \frac{\lambda_j - \lambda_{b+1}}{\lambda_{b+1} - \lambda_n} \tag{1.27}$$

There are several points to notice. First and foremost, notice that the bound holds only for $j = 1, \cdots, b$. In other words, if we want to use this bound to guarantee a certain rate of convergence for the k largest eigenvalues, in Alg. 2, we must take the block size b to be greater than k.

For simplicity, in Eqn. (1.26), take $j = 1$. The convergence factor can be decomposed into the two familiar factors, similar to the bound for the non-blocked algorithm - the Chebyshev term $T_{q-1}(\rho_j)$, and the alignment term $\tan\Theta(\mathbf{U}, \mathbf{V})$. In the former, we see that the number of iterations q appear in the degree of the polynomial, while the block-size b appears in ρ_1 to create a relative "gap" $\frac{\lambda_1 - \lambda_{b+1}}{\lambda_{b+1} - \lambda_n}$. It is clear that the convergence rate scales positively with both q and b.

The above theorem was also generalized to a bound for all $j = 1, \cdots, b$ eigenvalues by Saad, and appears below.

Theorem 1.2.4 ([40]). *Let $\widetilde{\lambda}_j^{(q)}$ be the jth eigenvalue of the block matrix $\mathbf{T}_q$ that results from running Alg. 2. Let $\mathbf{U}_i = \begin{bmatrix}\mathbf{u}_i & \cdots & \mathbf{u}_{i+b-1}\end{bmatrix}$ be a block matrix whose columns are the b eigenvectors associated with $\lambda_i \geq \cdots \geq \lambda_{i+b-1}$. For $j = 1, \cdots b$, if $\mathbf{U}_j^T\mathbf{V}$ is of full rank b, then*

$$0 \leq \lambda_j - \widetilde{\lambda}_j^{(q)} \leq (\lambda_j - \lambda_n)\left(\frac{L_j^{(q)} \tan\Theta(\mathbf{U}_j, \mathbf{V})}{T_{q-j}(\widehat{\gamma}_j)}\right)^2 \tag{1.28}$$

where $T_i(x)$ is the Chebyshev polynomial of degree i, $\cos\Theta(\mathbf{U}_j, \mathbf{V}) = \sigma_{\min}(\mathbf{U}_j^T\mathbf{V})$, and

$$\widehat{\gamma}_j = 1 + 2 \cdot \frac{\lambda_j - \lambda_{j+b}}{\lambda_{j+b} - \lambda_n} \tag{1.29}$$

$$L_j^{(q)} = \begin{cases} \prod_{i=1}^{j-1} \frac{\widetilde{\lambda}_i^{(q)} - \lambda_n}{\widetilde{\lambda}_i^{(q)} - \lambda_j} & \text{if } j \neq 1 \\ 1 & \text{if } j = 1 \end{cases} \tag{1.30}$$

We end this overview by emphasizing again that while Eqn. (1.28) is generalized to be a bound on $\widetilde{\lambda}_j^{(q)}$ with relative gap $\frac{\lambda_j - \lambda_{j+b}}{\lambda_{j+b} - \lambda_n}$ of size b, it is still applicable only to the eigenvalues $j = 1, \cdots, b$ up to the block size. To our knowledge, in the classical Lanczos convergence theory literature, there are no results that give inequality bounds for eigenvalues $j > b$. In order to achieve guarantees for k eigenvalues, the existing theorems would suggest that one must either run Alg. 1 with some number of iterations $q \geq k$, or run Alg. 2 with a large enough block size $b \geq k$. The former guarantee is driven by the growth of the Chebyshev polynomial, while the latter guarantee comes from the increased "spectrum gap" of size b. Noticeably lacking, however, is an interpolated result, where the combination of these effects are examined (Fig. 1.1).

1.3 Randomized Low-Rank Approximation Algorithms

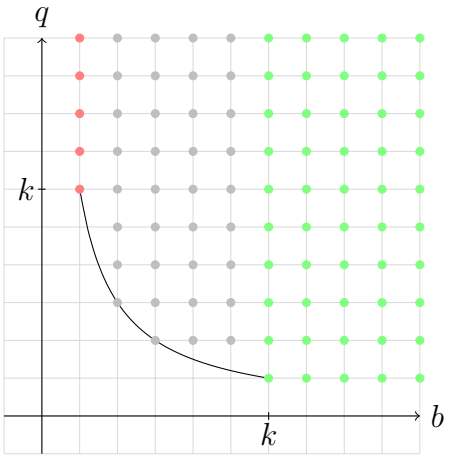
The low rank approximation of matrices, in addition to being a data compression technique in its own right, is an important computational procedure used in various other algorithms.

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, we say that another matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$ is a rank-k approximation to $\mathbf{A}$ if $\text{rank}(\mathbf{B}) \leq k$ and $\|\mathbf{A} - \mathbf{B}\|$ is small for some norm. In the former requirement, $k < \text{rank}(\mathbf{A})$ is typically chosen so that $\mathbf{B}$ is strictly an approximation of $\mathbf{A}$ based on computational and storage requirements. In the latter requirement, the specific choice of norm quantifies the aspect of "small"ness that is domain dependent and applicable to the specific problem. More precisely, the low rank approximation problem can be mathematically formulated as the following minimization problem.

$$\min_{\mathbf{B}} \|\mathbf{A} - \mathbf{B}\| \quad \text{s.t. } \text{rank}(\mathbf{B}) \leq k \tag{1.31}$$

The matrix approximation problem is closely tied to the singular value decomposition, due to the following celebrated optimality result.

Figure 1.1: Parameter space consisting of block size b and number of iterations q. Depicted are applicable parameter regimes for Thm. 1.2.2 (red), Thm. 1.2.4 (green), and results from the current work (grey).



Theorem 1.3.1 (Eckart - Young [17])**.**

$$\min_{\text{rank}(\mathbf{B})\leq k} \|\mathbf{A}-\mathbf{B}\|_2 = \|\mathbf{A}-\text{svd}_k(\mathbf{A})\|_2 = \sigma_{k+1} \tag{1.32}$$

$$\min_{\text{rank}(\mathbf{B})\leq k} \|\mathbf{A}-\mathbf{B}\|_F = \|\mathbf{A}-\text{svd}_k(\mathbf{A})\|_F = \sqrt{\sum_{j=k+1}^{\text{rank}(\mathbf{A})} \sigma_j^2} \tag{1.33}$$

where $\text{svd}_k(\mathbf{A})$ *denotes the* k*-truncated SVD of* $\mathbf{A}$*.*

From this result, we see that the minimization problem in Eqn. (1.31) is exactly solved for the spectral 2-norm and the Frobenius F-norm by first computing the SVD of $\mathbf{A}$, then truncating it to rank-k. Incidentally, this procedure also gives the exact answer to the following measurement of "approximation accuracy":

$$\min_{\mathbf{B}} |\sigma_j(\mathbf{A}) - \sigma_j(\mathbf{B})| \text{ s.t. } \text{rank}(\mathbf{B}) \leq k \text{ for } j = 1, \cdots, k \tag{1.34}$$

By construction, $\sigma_j(\mathbf{A}) = \sigma_j(\text{svd}_k(\mathbf{A}))$ for $j = 1, \cdots, k$, and so by this measure, the approximation error is 0.

While singular value decomposition algorithms provide an exact answer to the rank-k matrix approximation problem, traditional SVD algorithms, such as the Golub-Kahan bidiagonalization procedure followed by QR iterations as outlined in the previous section, are increasingly unable to meet the needs of various data applications. These algorithms have complexity $\mathcal{O}(mn^2)$ in the dimensions of the problem, and for modern datasets, n and m may routinely be on the order of 10^9. Moreover, while traditional scientific computing applications have high accuracy requirements (full double precision or more), this many digits of accuracy

is overkill for some more recent uses of SVD computations in big data computations. Because the low rank approximation functions as a computational subroutine in these algorithms, its accuracy only loosely affects the accuracy of the larger overall procedure, and in some cases merely 2-3 digits of accuracy in the SVD computation is sufficient.

In part driven by these considerations, the approximate low-rank approximation problem have received increased attention. Instead of seeking the exact answer to Eqn. (1.31), we seek an approximate answer, in the following sense: given $\mathbf{A} \in \mathbb{R}^{m \times n}$, target rank k, and approximation tolerance $\epsilon > 0$, compute an approximate rank-k approximation $\mathbf{B}$ such that $\text{rank}(\mathbf{B}) \leq k$ and

$$\|\mathbf{A} - \mathbf{B}\|_2 \leq (1 + \epsilon) \|\mathbf{A} - \text{svd}_k(\mathbf{A})\|_2 \tag{1.35}$$

or

$$\|\mathbf{A} - \mathbf{B}\|_F \leq (1 + \epsilon) \|\mathbf{A} - \text{svd}_k(\mathbf{A})\|_F \tag{1.36}$$

Additionally, instead of seeking an exact answer to Eqn. (1.34), seek an approximate rank-k approximation $\mathbf{B}$ such that for $j = 1, \cdots, k$

$$\sigma_j(\mathbf{A}) \geq \sigma_j(\mathbf{B}) \geq \frac{\sigma_j(\mathbf{A})}{\sqrt{1 + \epsilon}} \tag{1.37}$$

In recent years, numerous low-rank matrix approximation algorithms have been proposed to meet the dual requirement of high computational efficiency and low approximation accuracy. In these algorithms, randomization play a large role, variously

- as a way to randomly sample rows or columns from a given matrix, e.g. CUR decomposition, leverage score sampling [15, 32]
- as a way to randomly project the action of the matrix to a lower dimensional space, e.g. randomized subspace iteration, randomized QRCP, spectrum-revealing LU decomposition [25, 16, 49]

A more complete introduction to randomized low-rank approximation algorithms can be found in the excellent and comprehensive survey paper [27]. Below, we limit our brief overview to the randomized subspace iteration algorithm, which can be seen as a logical precursor to the randomized Lanczos iteration algorithm that is the main topic of this thesis.

Randomized Subspace Iteration

The randomized subspace iteration algorithm is derived from a straightforward extension of the subspace iteration algorithm - also known as the simultaneous iteration or orthogonal iteration algorithm. The subspace iteration algorithm itself is an extension of the power method, and relies on using a monomial $p_q(x) = x^q$ applied to $\mathbf{A}$ to amplify the desired part of the spectrum $\sigma_1, \cdots, \sigma_k$ while suppressing the undesired part of the spectrum $\sigma_{k+1}, \cdots, \sigma_n$.

The mathematical steps are given in Table 1.3 and the algorithmic steps are given in Alg. 5. While the subspace iteration algorithm is simpler, its intermediary matrices do not

Table 1.3: Outline of Subspace Iteration for the SVD problem

(1). Select an initial matrix $\mathbf{V}$.
(2). Construct the subspace $\mathcal{S}(\mathbf{A}, \mathbf{V}, q) = \text{span}\{(\mathbf{A}\mathbf{A}^T)^q\mathbf{V}\}$.
(3). Restrict and project $\mathbf{A}$ to the Krylov subspace to form $\mathbf{O} = \text{proj}_{\mathcal{S}}\mathbf{A}\big|_{\mathcal{S}}$.
(4). Compute the singular values $\widetilde{\sigma}_1, \cdots, \widetilde{\sigma}_k$ and left and right singular vectors $\widetilde{\mathbf{u}}_1, \cdots, \widetilde{\mathbf{u}}_k$ and $\widetilde{\mathbf{v}}_1, \cdots, \widetilde{\mathbf{v}}_k$ of $\mathbf{O}$ as the desired approximations for the first k singular values $\sigma_1, \cdots, \sigma_k$ and left and right singular vectors $\mathbf{u}_1, \cdots, \mathbf{u}_k$ and $\mathbf{v}_1, \cdots, \mathbf{v}_k$ of $\mathbf{A}$.

admit any special form, as the Lanczos method does (tridiagonal or bidiagonal), and the computational complexity for the two class of algorithms are comparable.

Algorithm 5 Pseudocode for subspace iteration for the SVD problem

Input: $\mathbf{A} \in \mathbb{R}^{m\times n}$, initial block matrix $\mathbf{V} \in \mathbb{R}^{n\times b}$ with $b \geq k$ orthogonal columns, number of iterations q.
Output: $\mathbf{B}_k$, a rank-k approximation of $\mathbf{A}$
1: Initialize $\mathbf{Q}_0 \leftarrow \mathbf{V}$.
2: **for** $j = 1, \cdots, q$ **do**
3: $\quad \mathbf{Z}_j \leftarrow \mathbf{A}\mathbf{A}^T\mathbf{Q}_{j-1}$
4: $\quad \mathbf{Q}_j\mathbf{R}_j \leftarrow \text{qr}(\mathbf{Z}_j)$
5: **end for**
6: $\mathbf{B} \leftarrow \mathbf{Q}_q\mathbf{Q}_q^T\mathbf{A}$
7: $\mathbf{B}_k \leftarrow \text{svd}_k(\mathbf{B})$

Note the similarity between the steps outlined in Table 1.2 and those outlined in Table 1.3. The difference of substance between these two classes of algorithms is the projection space - for a fixed q iterations, subspace iteration algorithms work with space constructed from the qth degree monomial applied to $\mathbf{A}$, while Lanczos algorithms work with the space constructed from the optimal qth degree polynomial applied to $\mathbf{A}$. These differences make evident why, intuitively, Lanczos methods converge faster than subspace iteration methods.

The randomized subspace iteration algorithm applies the idea of randomization to the starting matrix of the subspace iteration algorithm. From the convergence results of the previous section, and what we will describe in the next section, it is clear that choosing a starting matrix whose column space more closely aligns with the relevant eigenspace or singular space aids in the convergence of the algorithm. Instead of taking an arbitrary initial set of vectors $\mathbf{V}$ in Table 1.3, an unfortunate choice of which could result in poor convergence, the idea is to choose $\mathbf{V} = \mathbf{A}\boldsymbol{\Omega}$, where $\boldsymbol{\Omega}$ is a matrix with entries drawn from the Gaussian distribution $\mathcal{N}(0, 1)$. This amounts to taking the starting matrix $\mathbf{V}$ as a random projection of the columns of $\mathbf{A}$, and in theory better capture its range space in almost all cases. (Actually, it is better to say that the randomization guards against especially poor choices of the initial

starting matrix, and results in a "non-fatal" choice of $\mathbf{V}$ in almost all cases. This statement will be made more precise in the next section.)

The pseudocode for the random subspace iteration algorithm appears in Alg. 6.

Algorithm 6 Basic subspace iteration algorithm pseudocode

Input:
$\mathbf{A} \in \mathbb{R}^{m \times n}$
$\mathbf{\Omega} \in \mathbb{R}^{n \times b}$, initial starting matrix, typically a random Gaussian matrix
k , target rank , such that $b \geq k$
b , block size
q , number of Lanczos iterations

Output: $\mathbf{B}_k \in \mathbb{R}^{m \times n}$, a rank-k approximation to $\mathbf{A}$

1: Form the power matrix matrix $\mathbf{M} = (\mathbf{A}\mathbf{A}^T)^q \mathbf{A}\mathbf{\Omega}$.
2: Compute an orthonormal basis $\mathbf{Q}$ for the column span of $\mathbf{M}$, using e.g. $\mathbf{QR} \leftarrow \text{qr}(\mathbf{M})$.
3: Project $\mathbf{A}$ onto the Krylov subspace by computing $\mathbf{B} = \mathbf{Q}\mathbf{Q}^T\mathbf{A}$.
4: Compute k-truncated SVD $\mathbf{B}_k = \text{svd}_k(\mathbf{B}) = \text{svd}_k\left(\mathbf{Q}\mathbf{Q}^T\mathbf{A}\right) = \mathbf{Q} \cdot \text{svd}_k\left(\mathbf{Q}^T\mathbf{A}\right)$.
5: Return $\mathbf{B}_k$.

1.4 Convergence Theory for Randomized Low-Rank Approximation Algorithms

In Eqns. (1.35), (1.36), and (1.37), instead of viewing ϵ as an approximation error tolerance parameter, we might view it as a convergence factor determining the rate of convergence. In other words, an inequality of the form

$$\sigma_j \geq \sigma_j(\mathbf{B}_k) \geq \frac{\sigma_j}{\sqrt{1 + \{\text{some convergence factor}\}^2}} \tag{1.38}$$

for $j = 1, \cdots, k$, about Alg. 6, provides an quantitative answer to the question: how well do $\sigma_j(\mathbf{B}_k)$ of the computed rank-k approximation capture the true singular values σ_j?

The theorem below provide one representative bound of such form. Among the various (asymptotically equivalent) convergence theory results, we chose to present the following result because the main theoretical result of this thesis is similar in form, and the proof techniques thereof draws substantially upon those used to derive the following bound.

Theorem 1.4.1 ([25])**.** *Let $\mathbf{B}_k$ be the approximation returned by Alg. 6, and let $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ be the SVD of $\mathbf{A} \in \mathbb{R}^{m \times n}$. Assume that the matrix $\widehat{\mathbf{\Omega}}_1 \in \mathbb{R}^{k \times b}$ has full row rank in the partition $\mathbf{V}^T\mathbf{\Omega} = \begin{bmatrix} \widehat{\mathbf{\Omega}}_1^T & \widehat{\mathbf{\Omega}}_2^T \end{bmatrix}^T$, then Alg. 6 must satisfy for $j = 1, \cdots, k$*

$$\sigma_j \geq \sigma_j(\mathbf{B}_k) \geq \frac{\sigma_j}{\sqrt{1 + \left\|\widehat{\mathbf{\Omega}}_2\right\|_2^2 \left\|\widehat{\mathbf{\Omega}}_1^\dagger\right\|_2^2 \left(\frac{\sigma_{b+1}}{\sigma_j}\right)^{4q+2}}} \tag{1.39}$$

As expected, two terms contribute to the rate of convergence factor. The first, $\left(\frac{\sigma_{b+1}}{\sigma_j}\right)^{2q+1}$, involve q, the number of iterations. Notice that this term is the monomial $p_q(x) = x^{2q+1}$ applied to the spectral "gap" at the jth singular value, and q appears in the degree of this polynomial. The second term, $\left\|\widehat{\mathbf{\Omega}}_2\right\|_2 \left\|\widehat{\mathbf{\Omega}}_1^{\dagger}\right\|_2$, is dependent on the initial starting matrix $\mathbf{\Omega}$, and quantifies the alignment between this space and the right singular space spanned by the columns of $\mathbf{V}$.

The affect of randomization ensures against an unfortunate choice of $\mathbf{\Omega}$. (In the extreme, a $\mathbf{\Omega}$ might be chosen such that $\widehat{\mathbf{\Omega}}_1 = 0$, which entirely invalidates the inequality in Eqn.(1.39).) Using statistical arguments and machinery from random matrix theory, it can be shown that choosing $\mathbf{\Omega}$ randomly as a Gaussian random matrix, the alignment term, in expectation, is constant. This is stated in the theorem below.

Theorem 1.4.2 ([25])**.** *Let $\mathbf{B}_k$ be the approximation returned by Alg. 6, and let $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ be the SVD of $\mathbf{A}$. If $\mathbf{\Omega}$ is a random Gaussian matrix, then for $j = 1, \cdots, k$*

$$\mathbb{E}(\sigma_j(\mathbf{B}_k)) \geq \begin{cases} \dfrac{\sigma_j}{\sqrt{1+\mathcal{C}^2\left(\frac{\sigma_{b-p+1}}{\sigma_j}\right)^{4q+2}}} & \text{for } p \geq 2 \\ \dfrac{\sigma_j}{1+\mathcal{C}^2\left(\frac{\sigma_{b-p+1}}{\sigma_j}\right)^{4q+2}\log\sqrt{\mathcal{C}^2+\left(\frac{\sigma_{b-p+1}}{\sigma_j}\right)^{-(4q+2)}}} & \text{for } p = 1 \\ \dfrac{\sigma_j}{1+\mathcal{C}\left(\frac{\sigma_{b-p+1}}{\sigma_j}\right)^{2q+1}} & \text{for } p = 0 \end{cases} \tag{1.40}$$

for

$$\mathcal{C} = \left(\sqrt{n-b+p}+\sqrt{b}+7\right)\left(\frac{4e\sqrt{b}}{p+1}\right) \tag{1.41}$$

a constant independent of q, where $p = b - k$ is the oversampling parameter, indirectly determined by the choice of b.

We end by briefly discussing a theoretical convergence result for Alg. 6 of a different flavor.

Theorem 1.4.3 ([33], Thm. 12)**.** *There exists a choice of $q = \Theta\left(\frac{\log n}{\epsilon}\right)$ in Alg. 6 such that, with probability 99/100, the algorithm returns $\mathbf{B}_k$ satisfying*

$$\left|\sigma_j^2 - \sigma_j^2(\mathbf{B}_k)\right| \leq \epsilon\sigma_{b+1}^2 \tag{1.42}$$

In other words, for an appropriately chosen constant, the above convergence rate can be guaranteed with high probability for a choice of q with polynomial dependence on ϵ. Since Alg. 6 requires $b \geq k$, this theorem applies to that particular parameter regime.

The inequality bound of this theorem is unique in the sense that it is spectrum independent - to achieve an error of ϵ, it is enough to iteration for $q = \Theta\left(\frac{\log n}{\epsilon}\right)$ iterations, where the

number of iterations do not depend on spectrum "gaps". While this is certainly advantageous, especially in applications where spectrum decay of the matrix cannot be guaranteed, notice that written in a comparable form

$$\sigma_j \geq \sigma_j(\mathbf{B}_k) \geq \frac{\sigma_j}{\sqrt{1 + \mathcal{C}q^{-1} \log n \frac{\sigma_{b+1}^2}{\sigma_j^2}}} \tag{1.43}$$

it becomes apparent that, without being able to rely on information about the spectrum, the convergence rate is asymptotically worse than the one stated in Eqn. (1.39).

Chapter 2

Randomized Block Lanczos (RBL) Algorithm

2.1 Algorithm

We present the Randomized Block Lanczos (RBL) algorithm. This algorithm is a straightforward combination of the classical block Lanczos algorithm (Table 1.2) with the element of a randomized starting matrix (see discussion accompanying Randomized Subspace Iteration). To our knowledge, while this algorithm has appeared numerous times in the literature, there has been no general and unified convergence analysis of this algorithm, for all parameter settings.

The pseudocode for the basic block Lanczos algorithm is outlined in Algorithm 7. We require the choices of k, b, q to satisfy $(q+1)b \geq k$, to ensure that the Krylov subspace be at least k dimensional.

Algorithm 7 Randomized Block Lanczos (RBL)

Input:
- $\mathbf{A} \in \mathbb{R}^{m \times n}$
- $\mathbf{\Omega} \in \mathbb{R}^{n \times b}$, initial starting matrix, typically a random Gaussian matrix
- k , target rank , such that $(q+1)b \geq k$
- b , block size
- q , number of Lanczos iterations

Output: $\mathbf{B}_k \in \mathbb{R}^{m \times n}$, a rank-k approximation to $\mathbf{A}$

1: Form block column Krylov subspace matrix $\mathbf{K} = \begin{bmatrix} \mathbf{A\Omega} & (\mathbf{AA}^T)\mathbf{A\Omega} & \cdots & (\mathbf{AA}^T)^q\mathbf{A\Omega} \end{bmatrix}$.
2: Compute an orthonormal basis $\mathbf{Q}$ for the column span of $\mathbf{K}$, using e.g. $\mathbf{QR} \leftarrow \mathrm{qr}(\mathbf{K})$.
3: Project $\mathbf{A}$ onto the Krylov subspace by computing $\mathbf{B} = \mathbf{QQ}^T\mathbf{A}$.
4: Compute k-truncated SVD $\mathbf{B}_k = \mathrm{svd}_k(\mathbf{B}) = \mathrm{svd}_k\left(\mathbf{QQ}^T\mathbf{A}\right) = \mathbf{Q} \cdot \mathrm{svd}_k\left(\mathbf{Q}^T\mathbf{A}\right)$.
5: Return $\mathbf{B}_k$.

We present the algorithm pseudocode in this form in order to highlight the mathematical

ideas that are at the core of this algorithm. It is well known that a naive implementation of the Lanczos iteration is plagued by loss of orthogonality of the Lanczos vectors due to roundoff errors, and a realistic implementation of Algorithm 7 should involve at least 1) reorganization of the computation to use the three-term recurrence and bidiagonalization (see previous chapter), and 2) reorthogonalization during steps 1 and 2 [43, 29, 8].

2.2 Problem Statement

We are concerned with the approximation accuracy of the singular values of the Random Block Lanczos algorithm, and with how the accuracy scales with the number of iterations q, the block size b, and various spectrum behaviors.

More precisely, we seek a bound of the form

$$\sigma_j \geq \sigma_j(\mathbf{B}_k) \geq \frac{\sigma_j}{\sqrt{1 + \{\text{some convergence factor}\}^2}} \tag{2.1}$$

and we hope that such a bound, along with supporting numerical experiments, will help to answer such theoretical and practical questions as

- What is the asymptotic convergence of RBL in q?
- What is the effect of choosing different block sizes b?
- For the typical data matrix, what parameter settings encourage the optimal performance of RBL?

A partial answer to our core question was given by the following result.

Theorem 2.2.1 ([33], Thm. 13)**.** *For $b \geq k$, there exists a choice of $q = \Theta\left(\frac{\log n}{\sqrt{\epsilon}}\right)$ in Alg. 7 such that, with probability 99/100, the algorithm returns $\mathbf{B}_k$ satisfying*

$$\left|\sigma_j^2 - \sigma_j^2(\mathbf{B}_k)\right| \leq \epsilon \sigma_{b+1}^2 \tag{2.2}$$

In the event that $\sigma_{b+1} \leq c\sigma_k$ with $c < 1$, taking $q = \Theta\left(\frac{\log(n/\epsilon)}{\sqrt{\min\{1, \sigma_k/\sigma_{b+1} - 1\}}}\right)$ suffices.

As with Theorem 1.4.3 for randomized subspace iteration, this Theorem gives a spectrum independent bound for the convergence of the singular values for the Randomized Block Lanczos algorithm. While the algorithm itself does not require $b \geq k$, the authors of [33] have only stated stated and proven the theorem for this parameter regime. Converted to a familiar form, the spectrum independent version of Eqn. (2.2) reads

$$\sigma_j \geq \sigma_j(\mathbf{B}_k) \geq \frac{\sigma_j}{\sqrt{1 + \mathcal{C}^2 q^{-2} \log(n) \left(\frac{\sigma_{b+1}^2}{\sigma_j^2}\right)}} \tag{2.3}$$

for some constant $\mathcal{C}$. Theorem 2.2.1 also provides a spectrum dependent version of the bound, which is equivalent to

$$\sigma_j \geq \sigma_j(\mathbf{B}_k) \geq \frac{\sigma_j}{\sqrt{1 + \mathcal{C}^2 n \exp\left\{-q\sqrt{\min\{1, \sigma_k/\sigma_{b+1} - 1\}}\right\}\left(\frac{\sigma_{b+1}^2}{\sigma_j^2}\right)}} \tag{2.4}$$

for some other constant $\mathcal{C}$.

A comparison of Eqns. (2.3) and (2.4) reveals that there is an asymptotic loss associated with disregarding the information from the spectrum if the spectrum exhibits sufficient decaying behavior - the spectrum independent bound only guarantees polynomial convergence in q, while the spectrum dependent bound gives the familiar exponential convergence in q.

As a final note, as with all previous bounds for variants of the Block Lanczos algorithm, Theorem 2.2.1 is applicable only for choices of block sizes $b \geq k$.

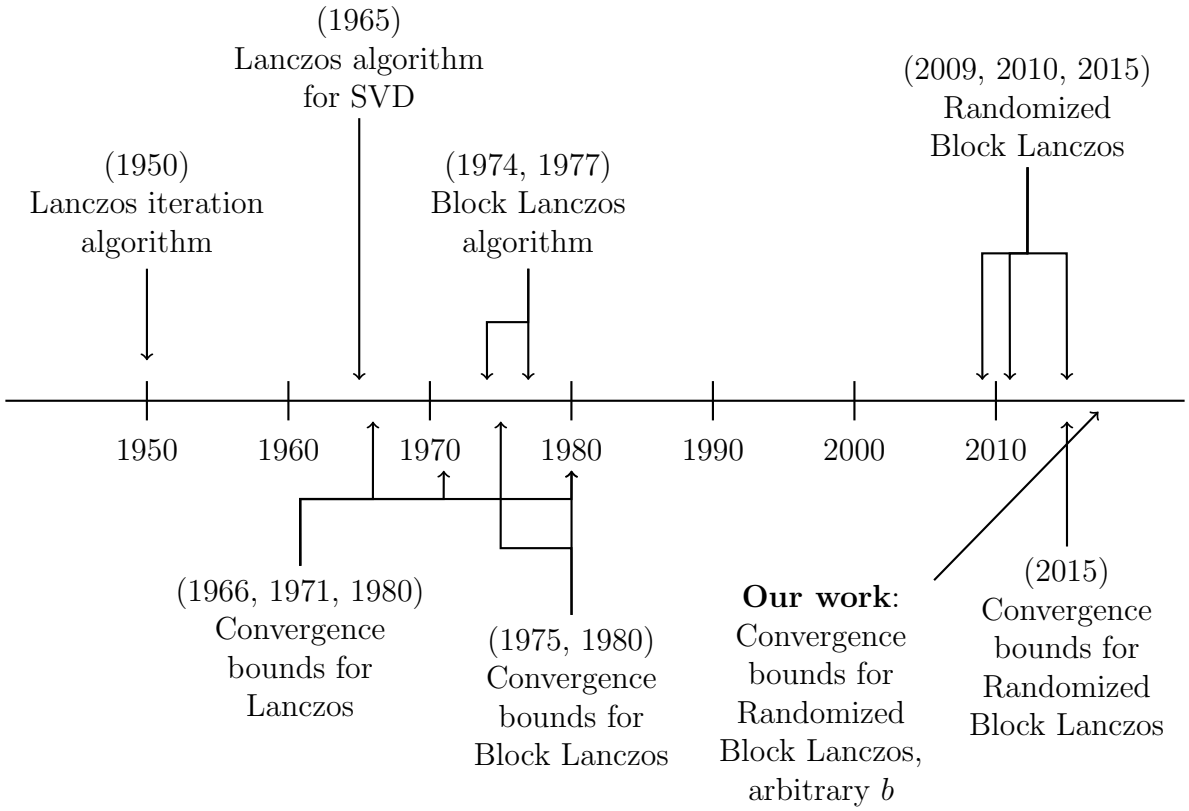
2.3 Discussion of Contributions

In this thesis, we present novel theoretical convergence results concerning the rate of singular value convergence for Algorithm 7, along with numerical experiments supporting these results. Our analysis presents a unified convergence theory for the Block Lanczos and Randomized Block Lanczos algorithms, for all valid parameter choices of block size b. To our knowledge, all previous results in the literature are applicable only with the choice of $b \geq k$, the target rank. We present a generalized theorem, applicable to all block sizes b, which coincide asymptotically with previous results for the case $b \geq k$, while providing equally strong rates of convergence for the case $b < k$. A timeline appears in Figure 2.3 and helps to situate the current work in the context of previous related literature.

Our current work is related primarily to the results laid out in two previous works. First, our analysis derives from the work of Gu ([25]). This work established aggressive multiplicative convergence bounds for the randomized subspace iteration algorithm (Alg. 6), for both singular values and normed (Frobenius, spectral) matrix convergence. As with traditional analysis of SVD algorithms, the bounds depend on both the singular value gap and the number of iterations taken by the algorithm, the former a property of the matrix in question, and the latter being proportional to the number of matrix multiplications and thus the complexity of the algorithm. The analysis presented in Gu's work is linear algebraic in nature, drawing on deterministic matrix analysis, as well as expectation bounds on randomized Gaussian matrices and their concentration of measure characteristics. The current work employs similar methods, and analysis is carried out on the general blocked Lanczos iteration algorithm. While this is more complicated in the details, core ideas, such as blocking, creating an artificial "gap" in the spectrum, and the choice of an opportune orthonormal basis for the analysis, remain the same.

Secondly, this thesis is based in part on the theoretical results presented by Musco and Musco ([33]). To our knowledge, this is the first work that gives "gap"-independent theoretical bounds for the block Lanczos algorithm. We emphasize that while the algorithm we

Figure 2.1: Timeline of relevant historical developments



analyze is identical in exact arithmetic to the one presented in Musco, the analysis found in [33] is restricted to the case for the Lanczos block size, b, chosen at least the size of k, the desired target rank. Our theoretical analysis will give a more generally applicable convergence bound, encompassing the case for both $1 \leq b < k$ and $b \geq k$. In the latter case, our theoretical results coincide with those in [33]. In the former case, we show, in particular, that the rapid convergence of the algorithm for any block size b larger than the largest singular value cluster size is assured. We draw attention to this distinction in choosing the block size parameter b, as, in our numerical results, we observe that it is advantageous to chose smaller block sizes $b < k$. (For the same amount of floating-point computations, choices of smaller block sizes appears to offer better accuracy across a variety of application matrices; see Ch. 4 for details). This is precisely the case for which this work guarantees the convergence rate theoretically and where previous works are lacking.

The current work is largely theoretical in nature, and there continues to be need for quality implementations of the Randomized Block Lanczos algorithm to aid its wider adoptability. To this end, continuations of the current work might include such an (possibly parallelized) implementation, along with further investigations of practical choices for the block size parameter b which balances the evident preference for a smaller b for convergence

with the advantages of a larger b for computational efficiency and numerical stability.

Chapter 3

Theoretical Analysis of RBL

3.1 Numerical Linear Algebra Preliminaries

In this section, we briefly review some definitions and results from Numerical Linear Algebra that are relevant to the discussions at hand and required for the forthcoming derivations. For more in-depth discussions of these topics, consult any standard Numerical Linear Algebra text, e.g. [46, 14, 23].

Throughout this section and the rest of this work, we assume exact arithmetic operations. For convenience, some definitions and results are given for $\mathbf{A} \in \mathbb{R}^{m\times n}$, $m \geq n$. For the case $m < n$, the same definitions and results can be applied to $\mathbf{A}^T$.

QR decomposition

Theorem 3.1.1. *Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $m \geq n$. If $\mathbf{A}$ has full column rank, then there exist a unique orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{m\times n}$ and a unique upper triangular matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$ with positive diagonals $r_{ii} \geq 0$ such that*

$$\mathbf{A} = \mathbf{QR} \tag{3.1}$$

The standard stable procedure for computing the QR factorization of a matrix is by Householder transformation, and the computational complexity of this algorithm is $\mathcal{O}(n^2 m)$.

In the event that $\mathbf{A}$ is rank deficient, say of $\text{rank}(\mathbf{A}) = r < n$, and its first r columns were independent, then its QR factorization would become

$$\mathbf{A} = \begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2\end{bmatrix} \begin{bmatrix}\mathbf{R}_{11} & \mathbf{R}_{12} \\ \mathbf{0} & \mathbf{0}\end{bmatrix} \tag{3.2}$$

where $\begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2\end{bmatrix}$ is an orthogonal matrix, $\mathbf{R}_{11} \in \mathbb{R}^{r\times r}$ is upper triangular and $\mathbf{R}_{12} \in \mathbb{R}^{r\times(n-r)}$.

In general, algorithms such as QR with column pivoting, or any rank-revealing QR factorization algorithms [26] may be used for the stable computation of the above.

Singular value decomposition

Theorem 3.1.2. *Let $\mathbf{A} \in \mathbb{R}^{m\times n}$, $m \geq n$ be an arbitrary matrix. Then, there exists orthogonal matrix $\mathbf{U} \in \mathbb{R}^{m\times n}$ (i.e. $\mathbf{U}^T\mathbf{U} = \mathbf{I}$), orthogonal matrix $\mathbf{V} \in \mathbb{R}^{n\times n}$ (i.e. $\mathbf{V}^T\mathbf{V} = \mathbf{I}$), and diagonal matrix $\mathbf{\Sigma} \in \mathbb{R}^{n\times n}$ with entries $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$, such that*

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{3.3}$$

The columns of $\mathbf{U} = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_n\end{bmatrix}$ are called the left singular vectors, the columns of $\mathbf{V} = \begin{bmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_n\end{bmatrix}$ are called the right singular vectors, and $\sigma_1, \cdots, \sigma_n$ are called the singular values. If $m < n$, the SVD is defined by considering $\mathbf{A}^T$.

A related entity to the SVD of $\mathbf{A}$ is the rank-k truncated SVD of $\mathbf{A}$.

Definition 3.1.1. *For $k \leq \text{rank}(\mathbf{A})$, given the SVD of $\mathbf{A}$ as defined above, the rank-k truncated SVD of $\mathbf{A} \in \mathbb{R}^{m\times n}$ is*

$$\text{svd}_k(\mathbf{A}) \equiv \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T \tag{3.4}$$

where $\mathbf{U}_k \equiv \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_k\end{bmatrix} \in \mathbb{R}^{m\times k}$, $\mathbf{V}_k \equiv \begin{bmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_k\end{bmatrix} \in \mathbb{R}^{n\times k}$, and $\mathbf{\Sigma}_k = \text{diag}(\sigma_1, \cdots, \sigma_k) \in \mathbb{R}^{k\times k}$.

The rank-k truncated SVD is optimal in the following sense.

Theorem 3.1.3 (Eckart - Young [17])**.**

$$\min_{\text{rank}(\mathbf{B})\leq k} \|\mathbf{A} - \mathbf{B}\|_2 = \|\mathbf{A} - \text{svd}_k(\mathbf{A})\|_2 = \sigma_{k+1} \tag{3.5}$$

$$\min_{\text{rank}(\mathbf{B})\leq k} \|\mathbf{A} - \mathbf{B}\|_F = \|\mathbf{A} - \text{svd}_k(\mathbf{A})\|_F = \sqrt{\sum_{j=k+1}^{\text{rank}(\mathbf{A})} \sigma_j^2} \tag{3.6}$$

Another related entity which can be defined in terms of the SVD is the pseudoinverse of $\mathbf{A}$.

Definition 3.1.2. *Let $r = \text{rank}(\mathbf{A})$, given the SVD of $\mathbf{A}$ as defined above, the Moore-Penrose pseudoinverse of $\mathbf{A} \in \mathbb{R}^{m\times n}$ is*

$$\mathbf{A}^\dagger \equiv \mathbf{U}_r\mathbf{\Sigma}_r^{-1}\mathbf{V}_r^T \tag{3.7}$$

where $\mathbf{\Sigma}_r^{-1} \equiv \text{diag}(\sigma_1^{-1}, \cdots, \sigma_r^{-1})$.

Due to the implicit relationship between the eigen decomposition and the SVD, in general eigen decomposition algorithms may be used to compute the SVD. Algorithms that stably compute the SVD of a matrix include bidiagonalization followed by QR iteration (especially suited to large, sparse matrices, see last chapter for an overview), divide-and-conquer, and Jacobi's method. These algorithms are generally $\mathcal{O}(mn^2)$ in computational complexity.

Interlacing results

The following interlacing results derive from the min-max formulation of of singular values, and relate the singular values of a matrix with those of the matrix projected to a subspace.

Theorem 3.1.4 (Cauchy Interlacing Property)**.** *Let* $\mathbf{A} = \begin{bmatrix}\mathbf{a}_1 & \cdots & \mathbf{a}_n\end{bmatrix} \in \mathbb{R}^{m\times n}$ *be a column partitioning with* $m \geq n$*. If* $\mathbf{A}_r = \begin{bmatrix}\mathbf{a}_1 & \cdots & \mathbf{a}_r\end{bmatrix}$*, then for* $r = 1, \cdots n-1$*,*

$$\sigma_1(\mathbf{A}_{r+1}) \geq \sigma_1(\mathbf{A}_r) \geq \sigma_2(\mathbf{A}_{r+1}) \geq \cdots \geq \sigma_r(\mathbf{A}_{r+1}) \geq \sigma_r(\mathbf{A}_r) \geq \sigma_{r+1}(\mathbf{A}_{r+1}) \tag{3.8}$$

More generally:

Theorem 3.1.5. *Let* $\mathbf{A} \in \mathbb{R}^{m\times n}$ *and let* $\mathbf{Q} \in \mathbb{R}^{m\times s}$ *be an orthonormal matrix. Then, for* $1 \leq j \leq \min(m, n)$

$$\sigma_j(\mathbf{A}) \geq \sigma_j\left(\mathbf{Q}^T\mathbf{A}\right) \tag{3.9}$$

Chebyshev polynomials

Definition 3.1.3. *The pth degree Chebyshev polynomial is defined by the recurrence*

$$T_0(x) \equiv 1 \tag{3.10}$$
$$T_1(x) \equiv x \tag{3.11}$$
$$T_p(x) \equiv 2pT_{p-1}(x) - T_{p-2}(x) \tag{3.12}$$

Alternatively, they may be expressed as

Theorem 3.1.6. *For* $|x| > 1$*,*

$$T_p(x) = \frac{1}{2}\left(\left(x + \sqrt{x^2-1}\right)^p + \left(x + \sqrt{x^2-1}\right)^{-p}\right) \tag{3.13}$$

The property that we will make use of, for Chebyshev polynomials, is that they are bounded between $-1 \leq x \leq 1$ and grow rapidly outside this interval. See Figure 3.1 for illustrative comparison between the growth of a Chebyshev polynomial, and the growth of a monomial of the same degree.

Theorem 3.1.7. *For any* $p > 0$*,*

$$|T_p(x)| \leq 1 \qquad \text{for } |x| \leq 1 \tag{3.14}$$

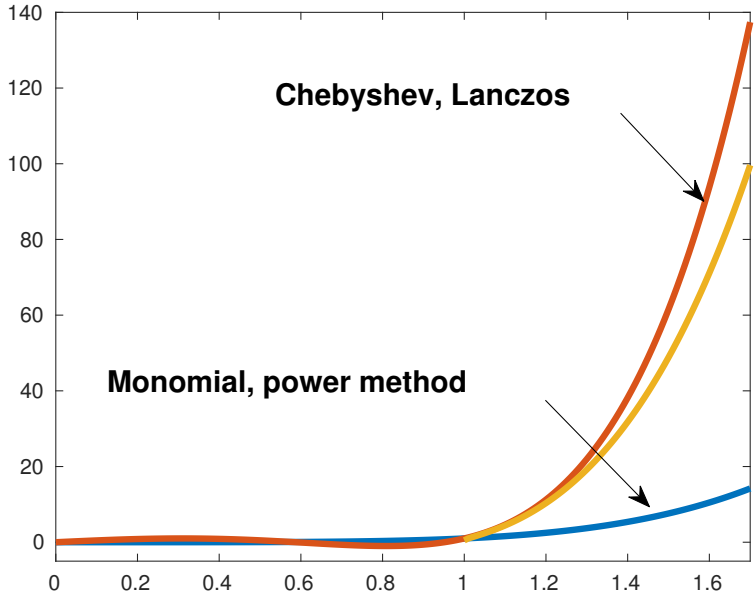
and

$$T_p(1+\epsilon) \geq 2^{p\sqrt{\epsilon}-2} \qquad \text{for } \epsilon > 0 \tag{3.15}$$

Moreover, the following informal estimate is enlightening. For p large and ϵ small,

$$T_p(1+\epsilon) \approx \frac{1}{2}\left(1 + \epsilon + \sqrt{2\epsilon}\right)^p \tag{3.16}$$

Figure 3.1: Comparison of the growth of the Chebyshev polynomial $T_n(x)$ (red), its approximation given by Eqn. (3.16) (orange), and the monomial of the same degree $M_n(x) = x^n$ (blue).



Statistical tools

We cite two lemmas needed to make arguments about the expectations of certain expressions involving Gaussian random matrices. Their proofs are given in [25].

Lemma 3.1.8. *Let $\alpha > 0$ and $\mathbf{G} \in \mathbb{R}^{m\times n}$ be a Gaussian random matrix. Then,*

$$\mathbb{E}\left(\frac{1}{\sqrt{1+\alpha^2\left\|\mathbf{G}\right\|_2^2}}\right) \geq \frac{1}{\sqrt{1+\alpha^2\mathcal{C}^2}} \tag{3.17}$$

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

Lemma 3.1.9. *Let $\alpha > 0$ and $\mathbf{G} \in \mathbb{R}^{(l-p)\times l}$ be a Gaussian random matrix, then with probability* 1, $\operatorname{rank}(\mathbf{G}) = l - p$ *and*

$$\mathbb{E}\left(\frac{1}{\sqrt{1+\alpha^2\left\|\mathbf{G}^\dagger\right\|_2^2}}\right) \geq \begin{cases} \frac{1}{\sqrt{1+\alpha^2\mathcal{C}^2}} & \text{for } p \geq 2 \\ \frac{1}{1+\alpha^2\mathcal{C}^2\log\frac{2\sqrt{1+\alpha^2\mathcal{C}^2}}{\alpha\mathcal{C}}} & \text{for } p = 1 \\ \frac{1}{1+\alpha\mathcal{C}} & \text{for } p = 0 \end{cases} \tag{3.18}$$

where $\mathcal{C} = \frac{4e\sqrt{l}}{p+1}$.

3.2 Setup and Intuition

We briefly overview the core pieces of our analysis. Our convergence analysis will show that if the Lanczos iteration in Algorithm 7 converges, then the k desired singular values of the approximation $\mathbf{B}_k$ converges to the corresponding true singular values of $\mathbf{A}$ exponentially in the number of iteration q. Moreover, convergence occurs as long as the block size b is chosen to be larger than the maximum cluster size for the k relevant singular values. Our analysis makes use of the following three ideas:

- the growth behavior of Chebyshev polynomials, a traditional ingredient in the analysis of Lanczos iteration methods,
- the choice of a clever orthonormal basis for the Krylov subspace, an idea adapted from [25]
- the creation of a spectrum "gap", by separating the spectrum of $\mathbf{A}$ into those singular values that are "close" to σ_k, and those that are sufficiently smaller in magnitude.

3.3 Convergence Theory

We begin with an analysis of the deterministic basic block Lanczos algorithm (Alg. 7). This analysis assumes a deterministic initial starting matrix $\mathbf{\Omega}$ with certain characteristics; in subsequently sections, we will use these results as a basis for proving probabilistic convergence results for special cases of the algorithm when it is run with random Gaussian initial starting matrices.

We are interested in the column span of the Krylov subspace matrix $\mathbf{K}$. Let the singular value decomposition of $\mathbf{A}$ be denoted as $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Then, we may write

$$\begin{aligned}
\mathbf{K} &= \begin{bmatrix} \mathbf{A}\mathbf{\Omega} & (\mathbf{A}\mathbf{A}^T)\mathbf{A}\mathbf{\Omega} & \cdots & (\mathbf{A}\mathbf{A}^T)^q\mathbf{A}\mathbf{\Omega} \end{bmatrix} && (3.19)\\
&= \begin{bmatrix} \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{\Omega} & \mathbf{U}\mathbf{\Sigma}^{2+1}\mathbf{V}^T\mathbf{\Omega} & \cdots & \mathbf{U}\mathbf{\Sigma}^{2q+1}\mathbf{V}^T\mathbf{\Omega} \end{bmatrix} && (3.20)\\
&= \mathbf{U}\mathbf{\Sigma}\begin{bmatrix} \widehat{\mathbf{\Omega}} & \widehat{\mathbf{\Sigma}}\widehat{\mathbf{\Omega}} & \cdots & \widehat{\mathbf{\Sigma}}^q\widehat{\mathbf{\Omega}} \end{bmatrix} && (3.21)
\end{aligned}$$

where for notational convenience we have defined the quantities $\widehat{\mathbf{\Omega}} \equiv \mathbf{V}^T\mathbf{\Omega}$ and $\widehat{\mathbf{\Sigma}} \equiv \mathbf{\Sigma}^2$.

As previously discussed, we want to separate the component of the Krylov subspace that drives convergence as the number of iterations q increases, from the component that is related to the initial starting subspace but independent of q. With this in mind, we define the following matrix.

Definition 3.3.1. *For $0 \leq p \leq q$, let*

$$\mathbf{K}_p \equiv \mathbf{U}T_{2p+1}(\mathbf{\Sigma})\begin{bmatrix} \widehat{\mathbf{\Omega}} & \widehat{\mathbf{\Sigma}}\widehat{\mathbf{\Omega}} & \cdots & \widehat{\mathbf{\Sigma}}^{q-p}\widehat{\mathbf{\Omega}} \end{bmatrix} \tag{3.22}$$

where $T_{2p+1}(x)$ is a shifted and scaled Chebyshev polynomial of degree $2p+1$.

The matrices $\mathbf{K}$ and $\mathbf{K}_p$ are related as follows.

Lemma 3.3.1. *For all $0 \leq p \leq q$, for any matrices $\mathbf{M}$, $\mathbf{N}$ and polynomial $p(x)$ of degree p,*

$$\operatorname{span}\left\{p(\mathbf{M})\begin{bmatrix}\mathbf{N} & \mathbf{MN} & \cdots & \mathbf{M}^{q-p}\mathbf{N}\end{bmatrix}\right\} \subseteq \operatorname{span}\left\{\begin{bmatrix}\mathbf{N} & \mathbf{MN} & \cdots \mathbf{M}^{q}\mathbf{N}\end{bmatrix}\right\} \tag{3.23}$$

In particular,

$$\operatorname{span}\{\mathbf{K}_p\} \subseteq \operatorname{span}\{\mathbf{K}\} \tag{3.24}$$

Proof. The first statement follows from the definition of the column span of matrices. For the second statement, take $\mathbf{M} = \mathbf{A}\mathbf{A}^T$, $\mathbf{N} = \mathbf{A}\mathbf{\Omega}$. Then, we have, firstly,

$$\operatorname{span}\{\mathbf{K}\} \supseteq \operatorname{span}\left\{(\mathbf{A}\mathbf{A}^T)^p\begin{bmatrix}\mathbf{A}\mathbf{\Omega} & (\mathbf{A}\mathbf{A}^T)\mathbf{A}\mathbf{\Omega} & \cdots & (\mathbf{A}\mathbf{A}^T)^{q-p}\mathbf{A}\mathbf{\Omega}\end{bmatrix}\right\} \tag{3.25}$$

for any $0 \leq p \leq q$ by the first statement, and,

$$\operatorname{span}\left\{(\mathbf{A}\mathbf{A}^T)^p\begin{bmatrix}\mathbf{A}\mathbf{\Omega} & (\mathbf{A}\mathbf{A}^T)\mathbf{A}\mathbf{\Omega} & \cdots & (\mathbf{A}\mathbf{A}^T)^{q-p}\mathbf{A}\mathbf{\Omega}\end{bmatrix}\right\} \tag{3.26}$$

$$= \operatorname{span}\left\{\mathbf{U}\mathbf{\Sigma}\mathcal{P}_p(\widehat{\mathbf{\Sigma}})\begin{bmatrix}\widehat{\mathbf{\Omega}} & \widehat{\mathbf{\Sigma}}\widehat{\mathbf{\Omega}} & \cdots & \widehat{\mathbf{\Sigma}}^{q-p}\widehat{\mathbf{\Omega}}\end{bmatrix}\right\} \tag{3.27}$$

$$\supseteq \operatorname{span}\left\{\mathbf{U}T_{2p+1}(\mathbf{\Sigma})\begin{bmatrix}\widehat{\mathbf{\Omega}} & \widehat{\mathbf{\Sigma}}\widehat{\mathbf{\Omega}} & \cdots & \widehat{\mathbf{\Sigma}}^{q-p}\widehat{\mathbf{\Omega}}\end{bmatrix}\right\} \tag{3.28}$$

where in the second equation, $\mathcal{P}_p$ is the set of polynomials of degree at most p. While as sets of polynomials $x\mathcal{P}_p(x^2) \neq \mathcal{P}_{2p+1}(x)$, it is well known that the odd degree Chebyshev polynomials $T_{2p+1}(x)$ belong to the set $x\mathcal{P}_p(x^2)$, a fact that was used from the second to the third equation. □

Using the variational characterization of singular values (Thm. 3.1.4), it is not hard to show that

Lemma 3.3.2. *In step 1 of algorithm 7, if instead of $\mathbf{K}$ we formed the factored matrix $\mathbf{K}_p$, then the singular values of the approximation returned by the modified algorithm bounds the singular values of the approximation returned by the original algorithm from below.*

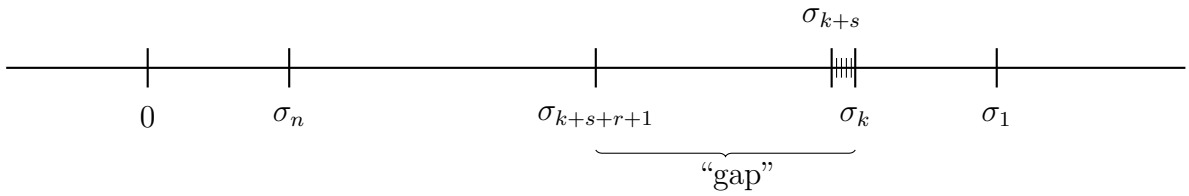
For the rest of the analysis, Lemma 3.3.2 allows us to work with $\mathbf{K}_p$ instead of the more complicated $\mathbf{K}$. To proceed further in seeing the manner of convergence, we must choose a helpful orthonormal basis $\mathbf{Q}$ for the column span of $\mathbf{K}_p$.

Lemma 3.3.3 ([25])**.** *For any non-singular matrix $\mathbf{X}$ of the correct dimensions, assume $\mathbf{\Omega}$ is chosen such that the $\mathbf{K}_p$ matrix as we have defined it is full rank. Let $\widehat{\mathbf{Q}}\widehat{\mathbf{R}}$ be the QR factorization of $\mathbf{K}_p\mathbf{X}$, and let $\mathbf{QR}$ be the QR factorization of $\mathbf{K}_p$, then*

$$\mathbf{Q}\mathbf{Q}^T = \widehat{\mathbf{Q}}\widehat{\mathbf{Q}}^T \tag{3.29}$$

Below we choose an obliging $\mathbf{X}$, which as much as possible orients the first k columns of $\mathbf{K}_p$ in the directions of the leading k singular vectors.

Figure 3.2: Illustration of parameters r, s.



Definition 3.3.2. *For all $0 \leq p \leq q$, define the generalized Vandermonde matrix*

$$\mathbf{V}_p \equiv \begin{bmatrix} \widehat{\mathbf{\Omega}} & \widehat{\mathbf{\Sigma}}\widehat{\mathbf{\Omega}} & \cdots & \widehat{\mathbf{\Sigma}}^{q-p}\widehat{\mathbf{\Omega}} \end{bmatrix} \tag{3.30}$$

and partition this matrix as follows:

$$\mathbf{V}_p = \begin{bmatrix} \mathbf{V}_{11} & \mathbf{V}_{12} \\ \mathbf{V}_{21} & \mathbf{V}_{22} \\ \mathbf{V}_{31} & \mathbf{V}_{32} \\ \mathbf{V}_{41} & \mathbf{V}_{42} \end{bmatrix} \tag{3.31}$$

where the blocks in the first dimension are sized $k, s, r, t = n-(k+s+r)$ and the blocks in the second dimension are sized k, r. We partition the generalized Vandermonde matrix in such a way to create a "gap" in the analysis that allows us to examine the convergence behavior viewed as an accentuation of the "gap" by the appropriate polynomial. An overview of the purpose of these parameters is as follows:

$k =$ target rank, algorithm parameter

$s =$ used to handle duplicate / clustered singular values, analysis parameter

$r =$ used to create the "gap" that drives convergence, analysis parameter

$t = n - (k + s + r),$ the "rest" of the spectrum

See Figure 3.2 for an illustration.

We show the existence of (at least one) special non-singular $\mathbf{X} \in \mathbb{R}^{(k+r)\times(k+r)}$ which reveals a "gap" of size r. More precisely, with the notation as previously given, we claim that there exists non-singular $\mathbf{X}$ such that

$$\mathbf{K}_p\mathbf{X} = \mathbf{U}T_{2p+1}(\mathbf{\Sigma})\mathbf{V}_p\mathbf{X} \tag{3.32}$$

$$= \mathbf{U} \begin{bmatrix} \mathbf{Q}_{11} & \widehat{\mathbf{V}}_{12} \\ \mathbf{Q}_{21} & \widehat{\mathbf{V}}_{22} \\ \mathbf{0} & \widehat{\mathbf{V}}_{32} \\ \mathbf{H} & \widehat{\mathbf{V}}_{42} \end{bmatrix} \tag{3.33}$$

with $\begin{bmatrix} \mathbf{Q}_{11} \\ \mathbf{Q}_{21} \end{bmatrix}$ a column orthogonal matrix. Notice the "gap" in the $(3,1)$ block of size r is created by using $\mathbf{X}$ to align the columns of $\mathbf{K}_p$.

Below, we explicitly construct such an $\mathbf{X}$. Partition

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} \\ \mathbf{X}_{21} & \mathbf{X}_{22} \end{bmatrix} \tag{3.34}$$

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_1 & & & \\ & \mathbf{\Sigma}_2 & & \\ & & \mathbf{\Sigma}_3 & \\ & & & \mathbf{\Sigma}_4 \end{bmatrix} \tag{3.35}$$

where each dimension of $\mathbf{X}$ is sized k, r, and each dimension of $\mathbf{\Sigma}$ is sized $k, s, r, t = n - (k + s + r)$. Then,

$$T_{2p+1}(\mathbf{\Sigma})\mathbf{V}_p\mathbf{X} = \begin{bmatrix} T_{2p+1}(\mathbf{\Sigma}_1) & & & \\ & T_{2p+1}(\mathbf{\Sigma}_2) & & \\ & & T_{2p+1}(\mathbf{\Sigma}_3) & \\ & & & T_{2p+1}(\mathbf{\Sigma}_4) \end{bmatrix} \begin{bmatrix} \mathbf{V}_{11} & \mathbf{V}_{12} \\ \mathbf{V}_{21} & \mathbf{V}_{22} \\ \mathbf{V}_{31} & \mathbf{V}_{32} \\ \mathbf{V}_{41} & \mathbf{V}_{42} \end{bmatrix} \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} \\ \mathbf{X}_{21} & \mathbf{X}_{22} \end{bmatrix} \tag{3.36}$$

$$= \left[\begin{array}{cc|c} \begin{pmatrix} T_{2p+1}(\mathbf{\Sigma}_1) & \\ & T_{2p+1}(\mathbf{\Sigma}_2) \end{pmatrix} \begin{pmatrix} \mathbf{V}_{11} & \mathbf{V}_{12} \\ \mathbf{V}_{21} & \mathbf{V}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{X}_{11} \\ \mathbf{X}_{21} \end{pmatrix} & \cdots \\ \hline T_{2p+1}(\mathbf{\Sigma}_3)(\mathbf{V}_{31}\mathbf{X}_{11} + \mathbf{V}_{32}\mathbf{X}_{21}) & \cdots \\ \hline T_{2p+1}(\mathbf{\Sigma}_4)(\mathbf{V}_{41}\mathbf{X}_{11} + \mathbf{V}_{42}\mathbf{X}_{21}) & \cdots \end{array}\right] \tag{3.37}$$

$$\equiv \left[\begin{array}{c|c} \begin{pmatrix} \widehat{\mathbf{V}}_{11} \\ \widehat{\mathbf{V}}_{21} \end{pmatrix} & \cdots \\ \hline \widehat{\mathbf{V}}_{31} & \cdots \\ \hline \widehat{\mathbf{V}}_{41} & \cdots \end{array}\right] \tag{3.38}$$

Setting

$$\mathbf{X}_{21} = -\mathbf{V}_{32}^{-1}\mathbf{V}_{31}\mathbf{X}_{11} \tag{3.39}$$

ensures that the $(2, 1)$ block of dimensions $r \times k$ in Eqn. (3.37) is the zero block (i.e. ensures $\widehat{\mathbf{V}}_{31} = \mathbf{0}$), and causes the $(1, 1)$ block of dimensions $(k + s) \times k$ to become

$$\begin{pmatrix} \widehat{\mathbf{V}}_{11} \\ \widehat{\mathbf{V}}_{21} \end{pmatrix} = \begin{pmatrix} T_{2p+1}(\mathbf{\Sigma}_1) & \\ & T_{2p+1}(\mathbf{\Sigma}_2) \end{pmatrix} \begin{pmatrix} \mathbf{V}_{11} & \mathbf{V}_{12} \\ \mathbf{V}_{21} & \mathbf{V}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{I} \\ -\mathbf{V}_{32}^{-1}\mathbf{V}_{31} \end{pmatrix} \mathbf{X}_{11} \tag{3.40}$$

$$= \begin{bmatrix} T_{2p+1}(\mathbf{\Sigma}_1) & \\ & T_{2p+1}(\mathbf{\Sigma}_2) \end{bmatrix} \begin{bmatrix} \mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31} \\ \mathbf{V}_{21} - \mathbf{V}_{22}\mathbf{V}_{32}^{-1}\mathbf{V}_{31} \end{bmatrix} \mathbf{X}_{11} \tag{3.41}$$

We then take the QR factorization

$$\widetilde{\mathbf{Q}}\widetilde{\mathbf{R}} = \begin{bmatrix} T_{2p+1}(\mathbf{\Sigma}_1) & \\ & T_{2p+1}(\mathbf{\Sigma}_2) \end{bmatrix} \begin{bmatrix} \mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31} \\ \mathbf{V}_{21} - \mathbf{V}_{22}\mathbf{V}_{32}^{-1}\mathbf{V}_{31} \end{bmatrix} \tag{3.42}$$

and set

$$\mathbf{X}_{11} = \widetilde{\mathbf{R}}^{-1} \tag{3.43}$$

which ensures that

$$\begin{pmatrix} \widehat{\mathbf{V}}_{11} \\ \widehat{\mathbf{V}}_{21} \end{pmatrix}^T \begin{pmatrix} \widehat{\mathbf{V}}_{11} \\ \widehat{\mathbf{V}}_{21} \end{pmatrix} = \left(\widetilde{\mathbf{Q}}\widetilde{\mathbf{R}}\widetilde{\mathbf{R}}^{-1} \right)^T \left(\widetilde{\mathbf{Q}}\widetilde{\mathbf{R}}\widetilde{\mathbf{R}}^{-1} \right) = \mathbf{I} \tag{3.44}$$

Let Eqn. (3.43) and Eqn. (3.39) define $\mathbf{X}_{11}$ and $\mathbf{X}_{21}$ respectively.

$$\begin{bmatrix} \mathbf{X}_{11} \\ \mathbf{X}_{21} \end{bmatrix} = \begin{bmatrix} \mathbf{I} \\ -\mathbf{V}_{32}^{-1}\mathbf{V}_{31} \end{bmatrix} \widetilde{\mathbf{R}}^{-1} \tag{3.45}$$

In order to completely specify the existence of the desired $\mathbf{X}$ in Eqn. (3.34), we must specify the remaining columns $\begin{bmatrix} \mathbf{X}_{12} \\ \mathbf{X}_{22} \end{bmatrix}$ such that $\mathbf{X}$ is non-singular. To this end, we can simply specify

$$\begin{bmatrix} \mathbf{X}_{12} \\ \mathbf{X}_{22} \end{bmatrix} \equiv \begin{bmatrix} \mathbf{X}_{11} \\ \mathbf{X}_{21} \end{bmatrix}^{\perp} \tag{3.46}$$

and this completes the description of $\mathbf{X}$.

Remark 3.3.1. *In order for the above derivation and thus Eqn. (3.46) and Eqn. (3.45) to be valid, the following conditions must hold: that* $\mathbf{\Omega}$ *is chosen to allow*

- $\mathbf{V}_{32}$ *to be non-singular and thus invertible,*
- $\mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31}$ *to be non-singular and thus* $\widetilde{\mathbf{R}}$ *to be invertible. Note that this expression is the Schur complement of the* $(k+r) \times (k+r)$ *matrix* $\begin{bmatrix} \mathbf{V}_{11} & \mathbf{V}_{12} \\ \mathbf{V}_{31} & \mathbf{V}_{32} \end{bmatrix}$ *with respect to the* $\mathbf{V}_{32}$ *block.*

For $\mathbf{\Omega}$ *drawn from the distribution of standard random Gaussian matrices with dimension* $n \times b$*, these conditions are satisfied with probability* 1 *([18, 11]).*

With this $\mathbf{X}$, we have shown (as desired in Eqn. (3.33)) that the first block column of $\mathbf{K}_p\mathbf{X}$ of size k is

$$\mathbf{K}_p\mathbf{X} = \mathbf{U} \left[\begin{array}{c|c} \mathbf{Q}_{11} & \cdots \\ \mathbf{Q}_{21} & \cdots \\ \mathbf{0} & \cdots \\ \mathbf{H} & \cdots \end{array}\right] \tag{3.47}$$

where $\begin{bmatrix} \mathbf{Q}_{11} \\ \mathbf{Q}_{21} \end{bmatrix}$ is column orthonormal and

$$\mathbf{H} = T_{2p+1}(\mathbf{\Sigma}_4)(\mathbf{V}_{41}\mathbf{X}_{11} + \mathbf{V}_{42}\mathbf{X}_{21}) \tag{3.48}$$

$$= T_{2p+1}(\mathbf{\Sigma}_4)(\mathbf{V}_{41} - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})\widetilde{\mathbf{R}}^{-1} \tag{3.49}$$

We present the first lower bound on the singular value of $\mathbf{B}_k$.

Lemma 3.3.4. *Let $\mathbf{B}_k$ be the matrix returned by Alg. 7, let $\mathbf{H}$ be as defined in Eqn. (3.49), and assume that the two conditions in Remark 3.3.1 hold. Then,*

$$\sigma_k(\mathbf{B}_k) \geq \frac{\sigma_{k+s}}{\sqrt{1 + \|\mathbf{H}\|_2^2}} \tag{3.50}$$

Proof. The matrix returned by Alg. 7 is the k-truncated SVD of $\mathbf{Q}\mathbf{Q}^T\mathbf{A}$, where the columns of $\mathbf{Q}$ are an orthonormal basis for the column span of $\mathbf{K}$. By Lemmas 3.3.2 and 3.3.3, it follows that

$$\sigma_k(\mathbf{B}_k) \geq \sigma_k\left(\widehat{\mathbf{Q}}_p\widehat{\mathbf{Q}}_p^T\mathbf{A}\right) \tag{3.51}$$

where $\widehat{\mathbf{Q}}_p$ contains columns that form an orthonormal basis for the column span of $\mathbf{K}_p\mathbf{X}$.

In particular, let $\widehat{\mathbf{Q}}_p\widehat{\mathbf{R}}_p$ be the QR factorization of $\mathbf{K}_p\mathbf{X}$, partitioned as follows:

$$\mathbf{K}_p\mathbf{X} = \widehat{\mathbf{Q}}_p\widehat{\mathbf{R}}_p = \begin{bmatrix}\widehat{\mathbf{Q}}_1 & \widehat{\mathbf{Q}}_2\end{bmatrix}\begin{bmatrix}\widehat{\mathbf{R}}_{11} & \widehat{\mathbf{R}}_{12} \\ & \widehat{\mathbf{R}}_{22}\end{bmatrix} \tag{3.52}$$

where the block dimensions are sized k, s, as appropriate.

We can then write

$$\widehat{\mathbf{Q}}_p\widehat{\mathbf{Q}}_p^T\mathbf{A} = \widehat{\mathbf{Q}}_p\begin{bmatrix}\widehat{\mathbf{Q}}_1^T \\ \widehat{\mathbf{Q}}_2^T\end{bmatrix}\mathbf{U}\left[\begin{array}{cc|cc} \begin{pmatrix}\boldsymbol{\Sigma}_1 & \\ & \boldsymbol{\Sigma}_2 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{pmatrix} & & \begin{pmatrix}\mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \boldsymbol{\Sigma}_3 & \\ & \boldsymbol{\Sigma}_4\end{pmatrix} & \end{array}\right]\mathbf{V}^T \tag{3.53}$$

$$= \widehat{\mathbf{Q}}_p\left[\begin{array}{cc|cc} \widehat{\mathbf{Q}}_1^T\mathbf{U}\begin{pmatrix}\boldsymbol{\Sigma}_1 & \\ & \boldsymbol{\Sigma}_2 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{pmatrix} & & \widehat{\mathbf{Q}}_1^T\mathbf{U}\begin{pmatrix}\mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \boldsymbol{\Sigma}_3 & \\ & \boldsymbol{\Sigma}_4\end{pmatrix} \\ \hline \widehat{\mathbf{Q}}_2^T\mathbf{U}\begin{pmatrix}\boldsymbol{\Sigma}_1 & \\ & \boldsymbol{\Sigma}_2 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{pmatrix} & & \widehat{\mathbf{Q}}_2^T\mathbf{U}\begin{pmatrix}\mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \boldsymbol{\Sigma}_3 & \\ & \boldsymbol{\Sigma}_4\end{pmatrix} \end{array}\right]\mathbf{V}^T \tag{3.54}$$

We are specifically interested in the $(1,1)$ block of the center matrix. By the Cauchy interlacing theorem for singular values (Thm. 3.1.5), it follows that

$$\sigma_k\left(\widehat{\mathbf{Q}}_p\widehat{\mathbf{Q}}_p^T\mathbf{A}\right) \geq \sigma_k\left(\widehat{\mathbf{Q}}_1^T\mathbf{U}\begin{pmatrix}\boldsymbol{\Sigma}_1 & \\ & \boldsymbol{\Sigma}_2 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{pmatrix}\right) \tag{3.55}$$

We can compare the first k columns of Eqn. (3.52) with the expression in Eqn. (3.47) to see that

$$\widehat{\mathbf{Q}}_1\widehat{\mathbf{R}}_{11} = \mathbf{U}\begin{bmatrix}\mathbf{Q}_{11} \\ \mathbf{Q}_{21} \\ \mathbf{0} \\ \mathbf{H}\end{bmatrix} \tag{3.56}$$

which helps us to write

$$\widehat{\mathbf{Q}}_1^T\mathbf{U}\begin{pmatrix}\boldsymbol{\Sigma}_1 & \\ & \boldsymbol{\Sigma}_2 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{pmatrix} = \left(\mathbf{U}\begin{pmatrix}\mathbf{Q}_{11} \\ \mathbf{Q}_{21} \\ \mathbf{0} \\ \mathbf{H}\end{pmatrix}\widehat{\mathbf{R}}_{11}^{-1}\right)^T \mathbf{U}\begin{pmatrix}\boldsymbol{\Sigma}_1 & \\ & \boldsymbol{\Sigma}_2 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{pmatrix} \tag{3.57}$$

$$= \widehat{\mathbf{R}}_{11}^{-T}\begin{pmatrix}\mathbf{Q}_{11} \\ \mathbf{Q}_{21} \\ \mathbf{0} \\ \mathbf{H}\end{pmatrix}^T\begin{pmatrix}\boldsymbol{\Sigma}_1 & \\ & \boldsymbol{\Sigma}_2 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{pmatrix} \tag{3.58}$$

$$= \widehat{\mathbf{R}}_{11}^{-T}\begin{bmatrix}\mathbf{Q}_{11}^T\boldsymbol{\Sigma}_1 & \mathbf{Q}_{21}^T\boldsymbol{\Sigma}_2\end{bmatrix} \tag{3.59}$$

On the other hand, we have

$$\sigma_{k+s} = \sigma_k\left(\sigma_{k+s}\begin{bmatrix}\mathbf{Q}_{11}^T & \mathbf{Q}_{21}^T\end{bmatrix}\right) \tag{3.60}$$

$$\leq \sigma_k\left(\begin{bmatrix}\mathbf{Q}_{11}^T\boldsymbol{\Sigma}_1 & \mathbf{Q}_{21}^T\boldsymbol{\Sigma}_2\end{bmatrix}\right) \tag{3.61}$$

$$= \sigma_k\left(\widehat{\mathbf{R}}_{11}^T\widehat{\mathbf{R}}_{11}^{-T}\begin{bmatrix}\mathbf{Q}_{11}^T\boldsymbol{\Sigma}_1 & \mathbf{Q}_{21}^T\boldsymbol{\Sigma}_2\end{bmatrix}\right) \tag{3.62}$$

$$\leq \|\widehat{\mathbf{R}}_{11}^T\|_2\,\sigma_k\left(\widehat{\mathbf{R}}_{11}^{-T}\begin{bmatrix}\mathbf{Q}_{11}^T\boldsymbol{\Sigma}_1 & \mathbf{Q}_{21}^T\boldsymbol{\Sigma}_2\end{bmatrix}\right) \tag{3.63}$$

Combining Eqns. (3.51), (3.55), (3.59), and (3.63), we obtain

$$\sigma_k(\mathbf{B}_k) \geq \frac{\sigma_{k+s}}{\|\widehat{\mathbf{R}}_{11}^T\|_2} \tag{3.64}$$

and, with the help of Eqn. (3.56), and noting that

$$\widehat{\mathbf{R}}_{11}^T\widehat{\mathbf{R}}_{11} = \widehat{\mathbf{R}}_{11}^T\left(\mathbf{U}^T\widehat{\mathbf{Q}}_1\right)^T\left(\mathbf{U}^T\widehat{\mathbf{Q}}_1\right)\widehat{\mathbf{R}}_{11} \tag{3.65}$$

$$= \begin{bmatrix}\mathbf{Q}_{11} \\ \mathbf{Q}_{21}\end{bmatrix}^T\begin{bmatrix}\mathbf{Q}_{11} \\ \mathbf{Q}_{21}\end{bmatrix} + \mathbf{H}^T\mathbf{H} \tag{3.66}$$

$$= \mathbf{I} + \mathbf{H}^T\mathbf{H} \tag{3.67}$$

completes the proof. □

From here, we can obtain the following general deterministic singular value convergence result.

Theorem 3.3.5. *Let $\mathbf{B}_k$ be the matrix returned by Alg. 7. Assume that $\boldsymbol{\Omega}$ is chosen such that the two conditions in Remark 3.3.1 hold. For any parameter choices r, s, and any parameter choice b satisfying $k + r = (q - p + 1)b \geq k$, with notation as before, for $j = 1, \cdots, k$,*

$$\sigma_j \geq \sigma_j(\mathbf{B}_k) \geq \frac{\sigma_{j+s}}{\sqrt{1 + \mathcal{C}^2 T_{2p+1}^{-2}\left(1 + 2\cdot\frac{\sigma_j - \sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)}} \tag{3.68}$$

where

$$\mathcal{C}^2 \equiv \|(\mathbf{V}_{41} - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})(\mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})^{-1}\|_2^2 \tag{3.69}$$

is a constant that is independent of q.

Proof. With an eye toward Lemma 3.3.4, we proceed by providing a bound for $\|\mathbf{H}\|_2^2$.

$$\|\mathbf{H}\|_2^2 \tag{3.70}$$
$$= \sigma_1^2\left(\mathbf{H}\mathbf{H}^T\right) \tag{3.71}$$
$$= \sigma_1^2\left(T_{2p+1}(\mathbf{\Sigma}_4)(\mathbf{V}41 - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})\left(\widetilde{\mathbf{R}}^T\widetilde{\mathbf{R}}\right)^{-1}(\mathbf{V}41 - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})^T T_{2p+1}(\mathbf{\Sigma}_4)\right) \tag{3.72}$$
$$= \sigma_1^2\Bigg(T_{2p+1}(\mathbf{\Sigma}_4)(\mathbf{V}41 - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31}) \left(\begin{bmatrix}\mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31}\\ \mathbf{V}_{21} - \mathbf{V}_{22}\mathbf{V}_{32}^{-1}\mathbf{V}_{31}\end{bmatrix}^T \begin{bmatrix}T_{2p+1}^2(\mathbf{\Sigma}_1) & \\ & T_{2p+1}^2(\mathbf{\Sigma}_2)\end{bmatrix} \begin{bmatrix}\mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31}\\ \mathbf{V}_{21} - \mathbf{V}_{22}\mathbf{V}_{32}^{-1}\mathbf{V}_{31}\end{bmatrix}\right)^{-1} (\mathbf{V}41 - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})^T T_{2p+1}(\mathbf{\Sigma}_4)\Bigg) \tag{3.73}$$
$$\leq \sigma_1^2\Bigg(T_{2p+1}(\mathbf{\Sigma}_4)(\mathbf{V}41 - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31}) \left((\mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})^T T_{2p+1}^2(\mathbf{\Sigma}_1)(\mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})\right)^{-1} (\mathbf{V}41 - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})^T T_{2p+1}(\mathbf{\Sigma}_4)\Bigg) \tag{3.74}$$
$$= \|T_{2p+1}(\mathbf{\Sigma}_4)(\mathbf{V}_{41} - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})(\mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})^{-1}T_{2p+1}^{-1}(\mathbf{\Sigma}_1)\|_2^2 \tag{3.75}$$
$$\leq T_{2p+1}^{-2}\left(1 + 2\cdot\frac{\sigma_k - \sigma_{k+s+r+1}}{\sigma_{k+s+r+1}}\right)\|(\mathbf{V}_{41} - \mathbf{V}_{42}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})(\mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{32}^{-1}\mathbf{V}_{31})^{-1}\|_2^2 \tag{3.76}$$

In the last equation, we have used the fact that $\mathbf{\Sigma}_1$ is a diagonal matrix whose elements are $\sigma_1, \cdots, \sigma_k$, and $\mathbf{\Sigma}_4$ is a diagonal matrix whose elements are $\sigma_{k+s+r+1}, \cdots, \sigma_n$. The $1 + 2\cdot\frac{\sigma_k - \sigma_{k+s+r+1}}{\sigma_{k+s+r+1}}$ factor is best seen as shifting the Chebyshev polynomial T_{2p+1} onto the interval $[0, \sigma_{k+s+r+1}]$, so that the tail of the singular spectrum is bounded by 1 and convergence is driven by the growth of the Chebyshev polynomial on the $\sigma_1, \cdots, \sigma_k$ part of the spectrum that we are interested in.

Repeating the previous argument for $1 \leq j \leq k$ completes the proof for the bound on $\sigma_j(\mathbf{B}_k)$. □

Remark 3.3.2. *Recall that the intuition for choosing s is that it be chosen as large as possible so that $\sigma_{j+s} \approx \sigma_j$. Typically, when σ_j is not a multiple singular value, we may choose $s = 0$,*

so that the bound Eqn. (3.68) takes the more familiar form where there is an oversampling parameter r and the bound is a relative inequality between σ_j and $\sigma_j(\mathbf{B}_k)$.

Remark 3.3.3. *In the statement of Theorem 3.3.5, we specified the condition $k + r = (q - p + 1)b \geq k$. Note that these are constraints of the analysis, and not of the algorithm. The algorithm requires only that the block Krylov subspace matrix spans a space large enough, which translates to the condition $(q + 1)b \geq k$. The condition $(q - p + 1)b \geq k$ derives from the fact that we use $\mathbf{K}_p$ in our analysis instead of the original block Krylov matrix.*

Remark 3.3.4. *An easy way to see that the constant $\mathcal{C}$ is independent of the iteration parameter q is to note that $\mathcal{C}$ is defined in terms of blocks of the generalized Vandermonde matrix $\mathbf{V}_p \in \mathbb{R}^{n\times(k+r)}$, which depends only on $\mathbf{\Omega}$, and $\mathbf{\Sigma}, \cdots, \mathbf{\Sigma}^{k+r}$.*

3.4 Special Case: $b = 1$

Previously, in Theorem 3.3.5, we assumed a general choice for the algorithm parameter block size b. Now, we will examine some specific cases for the choice of b, under which it will be possible to express the entries of $\mathbf{H}$, and so the constant $\mathcal{C}$ explicitly.

When the block size parameter b is taken to be 1, the initial random starting block matrix is in fact a random vector.

$$\mathbf{\Omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_n \end{bmatrix} \in \mathbb{R}^{n\times 1}, \qquad \widehat{\mathbf{\Omega}} = \mathbf{V}^T\mathbf{\Omega} = \begin{bmatrix} \widehat{\omega}_1 \\ \widehat{\omega}_2 \\ \vdots \\ \widehat{\omega}_n \end{bmatrix} \in \mathbb{R}^{n\times 1} \tag{3.77}$$

Then, the $\mathbf{K}_p$ as defined in Def. 3.3.1 can be expressed as

$$\mathbf{K}_p = \mathbf{U}T_{2p+1}(\mathbf{\Sigma}) \begin{bmatrix} \widehat{\omega}_1 & \widehat{\sigma}_1\widehat{\omega}_1 & \cdots & \widehat{\sigma}_1^{q-p}\widehat{\omega}_1 \\ \widehat{\omega}_2 & \widehat{\sigma}_2\widehat{\omega}_2 & \cdots & \widehat{\sigma}_2^{q-p}\widehat{\omega}_2 \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \widehat{\omega}_n & \widehat{\sigma}_n\widehat{\omega}_n & \cdots & \widehat{\omega}_n^{q-p}\widehat{\omega}_n \end{bmatrix} \tag{3.78}$$

$$= \mathbf{U}T_{2p+1}(\mathbf{\Sigma})\,\mathrm{diag}(\widehat{\mathbf{\Omega}}) \begin{bmatrix} 1 & \widehat{\sigma}_1 & \widehat{\sigma}_1^2 & \cdots & \widehat{\sigma}_1^{q-p} \\ 1 & \widehat{\sigma}_2 & \widehat{\sigma}_2^2 & \cdots & \widehat{\sigma}_2^{q-p} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \widehat{\sigma}_n & \widehat{\sigma}_n^2 & \cdots & \widehat{\sigma}_n^{q-p} \end{bmatrix} \tag{3.79}$$

where $\mathrm{diag}(\widehat{\mathbf{\Omega}})$ is the $n \times n$ diagonal matrix with entries $\widehat{\omega}_1, \cdots, \widehat{\omega}_n$ on the diagonal. Now, with the $\widehat{\omega}_i$ entries factored out, the remaining matrix is exactly the classical Vandermonde matrix.

As previously, we require $k + r = (q - p + 1)b = q - p + 1$, where r is an oversampling parameter. (See comments after Def. 3.3.2.) Analogous to Def. 3.3.2, we define the following $\widetilde{\mathbf{V}}_p$.

Definition 3.4.1. *For $0 \leq p \leq q$, let*

$$\widetilde{\mathbf{V}}_p \equiv \begin{bmatrix} 1 & \widehat{\sigma}_1 & \widehat{\sigma}_1^2 & \cdots & \widehat{\sigma}_1^{q-p} \\ 1 & \widehat{\sigma}_2 & \widehat{\sigma}_2^2 & \cdots & \widehat{\sigma}_2^{q-p} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \widehat{\sigma}_n & \widehat{\sigma}_n^2 & \cdots & \widehat{\sigma}_n^{q-p} \end{bmatrix} \tag{3.80}$$

$$\equiv \left[\begin{array}{cccc|ccc} 1 & \widehat{\sigma}_1 & \cdots & \widehat{\sigma}_1^{k-1} & \widehat{\sigma}_1^{k} & \cdots & \widehat{\sigma}_1^{k+r-1} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 1 & \widehat{\sigma}_k & \cdots & \widehat{\sigma}_k^{k-1} & \widehat{\sigma}_k^{k} & \cdots & \widehat{\sigma}_k^{k+r-1} \\ \hline 1 & \widehat{\sigma}_{k+1} & \cdots & \widehat{\sigma}_{k+1}^{k-1} & \widehat{\sigma}_{k+1}^{k} & \cdots & \widehat{\sigma}_{k+1}^{k+r-1} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 1 & \widehat{\sigma}_{k+r} & \cdots & \widehat{\sigma}_{k+r}^{k-1} & \widehat{\sigma}_{k+r}^{k} & \cdots & \widehat{\sigma}_{k+r}^{k+r-1} \\ \hline 1 & \widehat{\sigma}_{k+r+1} & \cdots & \widehat{\sigma}_{k+r+1}^{k-1} & \widehat{\sigma}_{k+r+1}^{k} & \cdots & \widehat{\sigma}_{k+r+1}^{k+r-1} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 1 & \widehat{\sigma}_n & \cdots & \widehat{\sigma}_n^{k-1} & \widehat{\sigma}_n^{k} & \cdots & \widehat{\sigma}_n^{k+r-1} \end{array}\right] \tag{3.81}$$

$$\equiv \begin{bmatrix} \widetilde{\mathbf{V}}_{11} & \widetilde{\mathbf{V}}_{12} \\ \widetilde{\mathbf{V}}_{21} & \widetilde{\mathbf{V}}_{22} \\ \widetilde{\mathbf{V}}_{41} & \widetilde{\mathbf{V}}_{42} \end{bmatrix} \tag{3.82}$$

Note that the three equations express three representations for the same matrix: 1) as originally expressed in the definition of $\mathbf{K}_p$, 2) re-indexed in terms of k, r, and 3) expressed in block form with dimensions $(k + r + (n - (k + r))) \times (k + r)$. In the block representation, it is not a typo that the block indices are $1, 2, 4$ respectively - this representation is intended to correspond to Def. 3.3.2 and Eqn. 3.31, where in the current definition we have made the simplifying assumption that the spectrum of $\mathbf{A}$ contains no multiple singular values. In turn, the parameter s can be taken to be 0, rendering the third block unnecessary. For the exposition that follows, the case for $s \neq 0$ can be reduced to the default case.

Now, once again, we seek a non-singular $\mathbf{X} \in \mathbb{R}^{(k+r)\times(k+r)}$ to reveal a "gap" of size r. Below, we show, explicitly, the construction of a non-singular $\mathbf{X}$ such that

$$\mathbf{K}_p\mathbf{X} = \mathbf{U}T_{2p+1}(\mathbf{\Sigma})\,\mathrm{diag}(\widehat{\mathbf{\Omega}})\widetilde{\mathbf{V}}_p\mathbf{X} \tag{3.83}$$

$$= \mathbf{U}\begin{bmatrix} \mathbf{I}_{k\times k} & \mathbf{0}_{k\times r} \\ \mathbf{0}_{r\times k} & \mathbf{I}_{r\times r} \\ \mathbf{H} & \mathbf{G} \end{bmatrix} \tag{3.84}$$

We will make use of the following lemma.

Lemma 3.4.1. *Let* $\mathbf{V}$ *be the* $n \times (l+1)$ *Vandermonde matrix*

$$\mathbf{V} = \begin{bmatrix} 1 & v_1 & v_1^2 & \cdots & v_1^l \\ 1 & v_2 & v_2^2 & \cdots & v_2^l \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & v_n & v_n^2 & \cdots & v_n^l \end{bmatrix} \tag{3.85}$$

If the constants v_j *are distinct, and* $n > l+1$*, then there exist non-singular matrix* $\mathbf{X} \in \mathbb{R}^{(l+1)\times(l+1)}$ *such that*

$$\mathbf{VX} = \begin{bmatrix} \mathbf{I} \\ \mathbf{H} \end{bmatrix} \tag{3.86}$$

where the entries of $\mathbf{H} \in \mathbb{R}^{(n-(l+1))\times(l+1)}$ *are given by*

$$\mathbf{H}_{ij} = \frac{(v_i - v_1)\cdots(v_i - v_{j-1})(v_i - v_{j+1})\cdots(v_i - v_{l+1})}{(v_j - v_1)\cdots(v_j - v_{j-1})(v_j - v_{j+1})\cdots(v_j - v_{l+1})} = \prod_{\substack{t=1 \\ t\neq j}}^{l+1} \frac{v_i - v_t}{v_j - v_t} \tag{3.87}$$

for

$$l+2 \leq i \leq n$$
$$1 \leq j \leq l+1$$

(For notational convenience, the indices of $\mathbf{H}$ *run as the absolute position of the entry in the* $\mathbf{VX}$ *matrix.)*

Proof. See appendix. □

Now, let us construct the $\mathbf{X}$ required in Eqn. (3.84). Applying the previous Lemma to $\widetilde{\mathbf{V}}_p$ gives some $\widetilde{\mathbf{X}}$ such that

$$\widetilde{\mathbf{V}}_p \widetilde{\mathbf{X}} = \begin{bmatrix} \mathbf{I}_{k\times k} & \\ & \mathbf{I}_{r\times r} \\ \widetilde{\mathbf{H}}_1 & \widetilde{\mathbf{H}}_2 \end{bmatrix} \tag{3.88}$$

Building upon this $\widetilde{\mathbf{X}}$, let

$$\mathbf{X} \equiv \widetilde{\mathbf{X}} \begin{bmatrix} \widehat{\omega}_1^{-1} & & \\ & \ddots & \\ & & \widehat{\omega}_{k+r}^{-1} \end{bmatrix} T_{2p+1} \left(\begin{bmatrix} \sigma_1^{-1} & & \\ & \ddots & \\ & & \sigma_{k+r}^{-1} \end{bmatrix} \right) \tag{3.89}$$

Then

$$\mathbf{K}_p\mathbf{X} \tag{3.90}$$

$$= \mathbf{U}T_{2p+1}(\mathbf{\Sigma})\,\mathrm{diag}(\widehat{\mathbf{\Omega}})\widetilde{\mathbf{V}}_p\mathbf{X} \tag{3.91}$$

$$= \mathbf{U}T\left(\begin{bmatrix}\sigma_1 & & \\ & \ddots & \\ & & \sigma_n\end{bmatrix}\right)\begin{bmatrix}\widehat{\omega}_1 & & \\ & \ddots & \\ & & \widehat{\omega}_n\end{bmatrix}\widetilde{\mathbf{V}}_p\widetilde{\mathbf{X}}\begin{bmatrix}\widehat{\omega}_1^{-1} & & \\ & \ddots & \\ & & \widehat{\omega}_{k+r}^{-1}\end{bmatrix}T\left(\begin{bmatrix}\sigma_1^{-1} & & \\ & \ddots & \\ & & \sigma_{k+r}^{-1}\end{bmatrix}\right) \tag{3.92}$$

$$= \mathbf{U}\begin{bmatrix} & \begin{pmatrix}\mathbf{I}_{k\times k} & \\ & \mathbf{I}_{r\times r}\end{pmatrix} & \\ \begin{pmatrix}T_{2p+1}(\sigma_1)\widehat{\omega}_1 & & \\ & \ddots & \\ & & T_{2p+1}(\sigma_n)\widehat{\omega}_n\end{pmatrix}\begin{pmatrix}\widetilde{\mathbf{H}}_1 & \widetilde{\mathbf{H}}_2\end{pmatrix}\begin{pmatrix}T_{2p+1}^{-1}(\sigma_1)\widehat{\omega}_1^{-1} & & \\ & \ddots & \\ & & T_{2p+1}^{-1}(\sigma_{k+r})\widehat{\omega}_{k+r}^{-1}\end{pmatrix} & \end{bmatrix} \tag{3.93}$$

$$= \mathbf{U}\begin{bmatrix}\mathbf{I}_{k\times k} & \\ & \mathbf{I}_{r\times r} \\ \mathbf{H} & \mathbf{G}\end{bmatrix} \tag{3.94}$$

where the entries of $\mathbf{H} \in \mathbb{R}^{(n-(k+r))\times(k+r)}$ can be expressed as

$$\mathbf{H}_{i,j} = \prod_{\substack{t=1\\ t\neq j}}^{k+r} \frac{\sigma_{k+r+i}^2 - \sigma_t^2}{\sigma_j^2 - \sigma_t^2} \cdot \frac{T_{2p+1}(\sigma_{k+r+i})}{T_{2p+1}(\sigma_j)} \cdot \frac{\widehat{\omega}_{k+r+i}}{\widehat{\omega}_j} \tag{3.95}$$

for

$$1 \leq i \leq n-(k+r)$$
$$1 \leq j \leq k$$

We now state and prove the singular value convergence bound for the special case block size $b = 1$.

Corollary 3.4.2 (of Thm. 3.3.5)**.** *Let $\mathbf{B}_k$ be the matrix returned by Alg. 7. Assume $\mathbf{\Omega} = \begin{bmatrix}\omega_1 & \omega_2 & \cdots & \omega_n\end{bmatrix}^T$ is chosen such that $\widehat{\omega}_1, \cdots \widehat{\omega}_k$ are non-zero in Eqn. (3.77). For any parameter choices r, s satisfying $k+r = (q-p+1) \geq k$, with notation as before, for $j = 1, \cdots, k$,*

$$\sigma_j \geq \sigma_j\left(\mathbf{B}_k\right) \geq \frac{\sigma_{j+s}}{\sqrt{1 + \mathcal{C}_{b=1}T_{2p+1}^{-2}\left(1 + 2\cdot\frac{\sigma_j - \sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)}} \tag{3.96}$$

where

$$\mathcal{C}_{b=1} = \left(\max_{\substack{1\le s\le k \\ j+r+1\le r\le n}} \frac{\widehat{\omega}_r}{\widehat{\omega}_s} \right)^2 \cdot \left(\sum_{s=1}^{j} \sum_{r=j+r+1}^{n} \prod_{\substack{t=1 \\ t\neq s}}^{j+r} \left(\frac{\sigma_r^2 - \sigma_t^2}{\sigma_s^2 - \sigma_t^2} \right)^2 \right) \tag{3.97}$$

is a constant independent of q.

Proof. Recalling the expression for, and derivation leading up to, $\mathbf{H}_{i,j}$ in Eqn. (3.95), we see that there exists a non-singular $\mathbf{X}$ such that

$$\mathbf{K}_p\mathbf{X} = \mathbf{U} \begin{bmatrix} \mathbf{I} & \\ & \mathbf{I} \\ \mathbf{H} & \mathbf{G} \end{bmatrix} \tag{3.98}$$

In this form, Lemma 3.3.4 applies, and it only remains to make an estimate of the size of $\|\mathbf{H}\|_2^2$.

$$\|\mathbf{H}\|_2^2 \le \|\mathbf{H}\|_F^2 \tag{3.99}$$

$$= \sum_{i=1}^{n-(k+r)} \sum_{j=1}^{k} \mathbf{H}_{i,j}^2 \tag{3.100}$$

$$= \sum_{i=1}^{n-(k+r)} \sum_{j=1}^{k} \left(\prod_{\substack{t=1 \\ t\neq j}}^{k+r} \frac{\sigma_{k+r+i}^2 - \sigma_t^2}{\sigma_j^2 - \sigma_t^2} \cdot \frac{T_{2p+1}(\sigma_{k+r+1})}{T_{2p+1}(\sigma_j)} \cdot \frac{\widehat{\omega}_{k+r+i}}{\widehat{\omega}_j} \right)^2 \tag{3.101}$$

$$= \sum_{s=1}^{k} \sum_{r=k+r+1}^{n} \left(\prod_{\substack{t=1 \\ t\neq s}}^{k+r} \frac{\sigma_r^2 - \sigma_t^2}{\sigma_s^2 - \sigma_t^2} \cdot \frac{T_{2p+1}(\sigma_r)}{T_{2p+1}(\sigma_s)} \cdot \frac{\widehat{\omega}_r}{\widehat{\omega}_s} \right)^2 \tag{3.102}$$

$$\le T_{2p+1}^{-2} \left(1 + 2 \cdot \frac{\sigma_k - \sigma_{k+r+1}}{\sigma_{k+r+1}} \right) \left(\max_{\substack{1\le s\le k \\ k+r+1\le r\le n}} \frac{\widehat{\omega}_r}{\widehat{\omega}_s} \right)^2 \left(\sum_{s=1}^{k} \sum_{r=k+r+1}^{n} \prod_{\substack{t=1 \\ t\neq s}}^{k+r} \left(\frac{\sigma_r^2 - \sigma_t^2}{\sigma_s^2 - \sigma_t^2} \right)^2 \right) \tag{3.103}$$

and this completes the proof for σ_k. We can repeat the same argument for the j-truncated matrix $\mathbf{B}_j$, for the bound on σ_j, $j = 1, \cdots k$. □

Remark 3.4.1. *While the above argument was made with the assumption that $s = 0$, i.e. that no singular value in the spectrum of $\mathbf{A}$ is a multiple singular value, the corollary as stated is valid for the general choice of s. To see this is the case, note that in the definition for $\widetilde{\mathbf{V}}_p$, had there been multiple singular values, we could have reindexed the spectrum such that the first $\sigma_1, \cdots, \sigma_k$ singular values are unique, and the subsequent $\sigma_{k+r+1}, \cdots, \sigma_{k+r+s}$ singular*

values contain multiples of those from the first k singular values. With this reordering, $\widetilde{\mathbf{V}}_p$ would become

$$\widetilde{\mathbf{V}}_p = \begin{bmatrix} \widetilde{\mathbf{V}}_{11} & \widetilde{\mathbf{V}}_{12} \\ \widetilde{\mathbf{V}}_{21} & \widetilde{\mathbf{V}}_{22} \\ \widetilde{\mathbf{V}}_{31} & \widetilde{\mathbf{V}}_{32} \\ \widetilde{\mathbf{V}}_{41} & \widetilde{\mathbf{V}}_{42} \end{bmatrix} \tag{3.104}$$

with blocking dimensions $k, r, s, n-(k+r+s)$ and k, r respectively. The subsequent reduction in Lemma 3.4.1 would have reduced this matrix to

$$\mathbf{K}_p\mathbf{X} = \mathbf{U} \begin{bmatrix} \mathbf{I}_{k\times k} & \mathbf{0}_{k\times r} \\ \mathbf{0}_{r\times k} & \mathbf{I}_{r\times r} \\ \mathbf{0}_{s\times k} & \mathbf{0}_{s\times r} \\ \mathbf{H} & \mathbf{G} \end{bmatrix} \tag{3.105}$$

and the remaining details of the derivation would be unchanged.

Remark 3.4.2. *Looking at the form of the constant $\mathcal{C}_{b=1}$, we see that it has two components, which are familiar from the traditional Lanczos bounds. The first, involving the ratios $\widehat{\omega}_r/\widehat{\omega}_s$, quantifies the alignment of the initial starting vector with the relevant singular subspace. If, as intended, the $\begin{bmatrix}\omega_1 & \cdots & \omega_n\end{bmatrix}^T$ initial starting vector is taken to have random Gaussian entries, then with probabilistic arguments we can guarantee that the non-zero condition in the statement of the corollary is satisfied with high probability. The second component is a "Lagrange multiples" factor.*

3.5 Special Case: $b \geq k + r$

When the block size parameter b is taken to be greater than the sum of the target rank k and the oversampling parameter r, it becomes possible to disentangle the effects of the spectrum decay and the random initial starting matrix $\mathbf{\Omega}$ in the convergence bounds. From this, we will also derive a spectrum independent convergence bound.

By taking $b \geq k + r$, we may choose $p = q$ in Definition 3.3.2 to satisfy the condition $(q-p+1)b \geq k+r$. In this case $\mathbf{V}_p = \begin{bmatrix}\widehat{\mathbf{\Omega}} & \widehat{\mathbf{\Sigma}}\widehat{\mathbf{\Omega}} & \cdots & \widehat{\mathbf{\Sigma}}^{q-p}\widehat{\mathbf{\Omega}}\end{bmatrix}$ will in fact consist of a single block.

$$\mathbf{V}_p = \widehat{\mathbf{\Omega}} = \begin{bmatrix} \widehat{\mathbf{\Omega}}_{11} & \widehat{\mathbf{\Omega}}_{12} \\ \widehat{\mathbf{\Omega}}_{21} & \widehat{\mathbf{\Omega}}_{22} \\ \widehat{\mathbf{\Omega}}_{31} & \widehat{\mathbf{\Omega}}_{32} \\ \widehat{\mathbf{\Omega}}_{41} & \widehat{\mathbf{\Omega}}_{42} \end{bmatrix} \tag{3.106}$$

where the block dimensions are again $k, s, r, n-(k+s+r)$ and k, r respectively.

Here, the key is that $\mathbf{V}_p$ no longer depends on the spectrum $\mathbf{\Sigma}$.

In addition, it follows that

$$\mathbf{K}_p = \mathbf{U} T_{2p+1}(\mathbf{\Sigma}) \widehat{\mathbf{\Omega}} \tag{3.107}$$

As before, for our analysis we seek a non-singular $\mathbf{X} \in \mathbb{R}^{(k+r)\times(k+r)}$ to reveal a "gap" of size r.

We begin by rotating $\widehat{\mathbf{\Omega}}$ into alignment as follows. Take the RQ-factorization of the third block row

$$\begin{bmatrix} \widehat{\mathbf{\Omega}}_{31} & \widehat{\mathbf{\Omega}}_{32} \end{bmatrix} = \mathbf{R}^T \mathbf{Q}^T = \begin{bmatrix} \mathbf{R}_{\widehat{\Omega}}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{V}_{\widehat{\Omega}}^T \\ \mathbf{W}_{\widehat{\Omega}}^T \end{bmatrix} \tag{3.108}$$

Then, form the rotation matrix

$$\widehat{\mathbf{Q}} = \begin{bmatrix} \mathbf{W}_{\widehat{\Omega}} & \mathbf{V}_{\widehat{\Omega}} \end{bmatrix} \tag{3.109}$$

Note that $\widehat{\mathbf{\Omega}}$ thus rotated becomes

$$\widehat{\mathbf{\Omega}}\widehat{\mathbf{Q}} = \begin{bmatrix} \begin{pmatrix} \widehat{\mathbf{\Omega}}_{11} & \widehat{\mathbf{\Omega}}_{12} \end{pmatrix} \mathbf{W}_{\widehat{\Omega}} & \begin{pmatrix} \widehat{\mathbf{\Omega}}_{11} & \widehat{\mathbf{\Omega}}_{12} \end{pmatrix} \mathbf{V}_{\widehat{\Omega}} \\ \begin{pmatrix} \widehat{\mathbf{\Omega}}_{21} & \widehat{\mathbf{\Omega}}_{22} \end{pmatrix} \mathbf{W}_{\widehat{\Omega}} & \begin{pmatrix} \widehat{\mathbf{\Omega}}_{21} & \widehat{\mathbf{\Omega}}_{22} \end{pmatrix} \mathbf{V}_{\widehat{\Omega}} \\ \begin{pmatrix} \widehat{\mathbf{\Omega}}_{31} & \widehat{\mathbf{\Omega}}_{32} \end{pmatrix} \mathbf{W}_{\widehat{\Omega}} & \begin{pmatrix} \widehat{\mathbf{\Omega}}_{31} & \widehat{\mathbf{\Omega}}_{32} \end{pmatrix} \mathbf{V}_{\widehat{\Omega}} \\ \begin{pmatrix} \widehat{\mathbf{\Omega}}_{41} & \widehat{\mathbf{\Omega}}_{42} \end{pmatrix} \mathbf{W}_{\widehat{\Omega}} & \begin{pmatrix} \widehat{\mathbf{\Omega}}_{41} & \widehat{\mathbf{\Omega}}_{42} \end{pmatrix} \mathbf{V}_{\widehat{\Omega}} \end{bmatrix} = \begin{bmatrix} \widetilde{\mathbf{\Omega}}_{11} & \widetilde{\mathbf{\Omega}}_{12} \\ \widetilde{\mathbf{\Omega}}_{21} & \widetilde{\mathbf{\Omega}}_{22} \\ \mathbf{0} & \mathbf{R}_{\widehat{\Omega}}^T \\ \widetilde{\mathbf{\Omega}}_{41} & \widetilde{\mathbf{\Omega}}_{42} \end{bmatrix} \tag{3.110}$$

Plugging back into Eqn. (3.107), we have

$$\mathbf{K}_p \widehat{\mathbf{Q}} = \mathbf{U} T_{2p+1} \left(\begin{bmatrix} \mathbf{\Sigma}_1 & & & \\ & \mathbf{\Sigma}_2 & & \\ & & \mathbf{\Sigma}_3 & \\ & & & \mathbf{\Sigma}_4 \end{bmatrix} \right) \begin{bmatrix} \widetilde{\mathbf{\Omega}}_{11} & \widetilde{\mathbf{\Omega}}_{12} \\ \widetilde{\mathbf{\Omega}}_{21} & \widetilde{\mathbf{\Omega}}_{22} \\ \mathbf{0} & \mathbf{R}_{\widehat{\Omega}}^T \\ \widetilde{\mathbf{\Omega}}_{41} & \widetilde{\mathbf{\Omega}}_{42} \end{bmatrix} \tag{3.111}$$

$$= \left[\begin{array}{c|c} \mathbf{U} \begin{pmatrix} T_{2p+1}(\mathbf{\Sigma}_1)\widetilde{\mathbf{\Omega}}_{11} \\ T_{2p+1}(\mathbf{\Sigma}_2)\widetilde{\mathbf{\Omega}}_{21} \\ \mathbf{0} \\ T_{2p+1}(\mathbf{\Sigma}_4)\widetilde{\mathbf{\Omega}}_{41} \end{pmatrix} & \cdots \end{array} \right] \tag{3.112}$$

where in the last expression, we have isolated the first k columns of $\mathbf{K}_p$.

We further push the "mass" of the top $k+s$ rows into the top k rows. Consider the QR-factorization of the $(k+s)\times k$ block

$$\begin{bmatrix} T_{2p+1}(\mathbf{\Sigma}_1)\widetilde{\mathbf{\Omega}}_{11} \\ T_{2p+1}(\mathbf{\Sigma}_2)\widetilde{\mathbf{\Omega}}_{21} \end{bmatrix} = \widetilde{\mathbf{Q}}\widetilde{\mathbf{R}} = \begin{bmatrix} \widetilde{\mathbf{Q}}_1 & \widetilde{\mathbf{Q}}_2 \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{R}}_1 \\ \mathbf{0} \end{bmatrix} \tag{3.113}$$

We have

$$\begin{bmatrix} T_{2p+1}(\mathbf{\Sigma}_1)\widetilde{\mathbf{\Omega}}_{11} \\ T_{2p+1}(\mathbf{\Sigma}_2)\widetilde{\mathbf{\Omega}}_{21} \\ \mathbf{0} \\ T_{2p+1}(\mathbf{\Sigma}_4)\widetilde{\mathbf{\Omega}}_{41} \end{bmatrix} \widetilde{\mathbf{R}}_1^{-1} = \begin{bmatrix} \begin{pmatrix} \widetilde{\mathbf{Q}}_1 \end{pmatrix} \\ \mathbf{0} \\ T_{2p+1}(\mathbf{\Sigma}_4)\widetilde{\mathbf{\Omega}}_{41}\widetilde{\mathbf{R}}_1^{-1} \end{bmatrix} \tag{3.114}$$

where by construction $\widetilde{\mathbf{Q}}_1$ is column orthogonal.

Finally, we take as a non-singular $\mathbf{X} \in \mathbb{R}^{(k+s)\times(k+s)}$

$$\mathbf{X} = \widehat{\mathbf{Q}}\widetilde{\mathbf{R}}_1^{-1} \tag{3.115}$$

With this choice, combining Eqn. (3.112) and Eqn. (3.114), we have

$$\mathbf{K}_p\mathbf{X} = \left[\begin{array}{c|c} \mathbf{U} \begin{pmatrix} \begin{pmatrix} \widetilde{\mathbf{Q}}_1 \end{pmatrix} \\ \mathbf{0} \\ \mathbf{H} \end{pmatrix} & \cdots \end{array}\right] \tag{3.116}$$

where

$$\mathbf{H} \equiv T_{2p+1}(\mathbf{\Sigma}_4)\widetilde{\mathbf{\Omega}}_{41}\widetilde{\mathbf{R}}_1^{-1} \tag{3.117}$$

Note the similarity between Eqn. (3.116) and Eqn. (3.47). Lemma 3.3.4 holds as well with $\mathbf{H}$ as defined in Eqn. (3.117).

We are now ready to state and prove the singular value convergence bound for the special case of block size $b \geq k + r$.

Corollary 3.5.1 (of Thm. 3.3.5)**.** *Let $\mathbf{B}_k$ be the matrix returned by Alg. 7. Assume $\mathbf{\Omega}$ is chosen such that $\widetilde{\mathbf{\Omega}}_{11}$ is nonsingular in Eqn. (3.110). For any parameter choices r, s, with notation as before, for $j = 1, \cdots, k$*

$$\sigma_j \geq \sigma_j\left(\mathbf{B}_k\right) \geq \frac{\sigma_{j+s}}{\sqrt{1 + \mathcal{C}_{b\geq k+r}^2 T_{2q+1}^{-2}\left(1 + 2 \cdot \frac{\sigma_j - \sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)}} \tag{3.118}$$

where

$$\mathcal{C}_{b\geq k+r} = \left\|\widetilde{\mathbf{\Omega}}_{41}\right\|_2 \left\|\widetilde{\mathbf{\Omega}}_{11}^{-1}\right\|_2 \tag{3.119}$$

is a constant independent of both q, the iteration parameter, and $\mathbf{\Sigma}$, the spectrum of $\mathbf{A}$.

Proof. We estimate $\|\mathbf{H}\|_2$ as

$$\|\mathbf{H}\|_2 = \left\|T_{2p+1}(\mathbf{\Sigma}_4)\widetilde{\mathbf{\Omega}}_{41}\widetilde{\mathbf{R}}_1^{-1}\right\|_2 \leq T_{2p+1}(\sigma_{k+s+r+1}) \left\|\widetilde{\mathbf{\Omega}}_{41}\right\|_2 \left\|\widetilde{\mathbf{R}}_1^{-1}\right\|_2 \tag{3.120}$$

and we estimate $\left\|\widetilde{\mathbf{R}}_1^{-1}\right\|_2$ as

$$\left\|\widetilde{\mathbf{R}}_1^{-1}\right\|_2 = \left\|\left(\widetilde{\mathbf{R}}_1^T\widetilde{\mathbf{R}}_1\right)^{-1}\right\|_2^{1/2} = \left(\sigma_{\min}\left(\widetilde{\mathbf{R}}_1^T\widetilde{\mathbf{R}}_1\right)\right)^{-1/2} \tag{3.121}$$

Since

$$\widetilde{\mathbf{R}}_1^T\widetilde{\mathbf{R}}_1 = \left(\widetilde{\mathbf{Q}}\widetilde{\mathbf{R}}\right)^T\left(\widetilde{\mathbf{Q}}\widetilde{\mathbf{R}}\right) \tag{3.122}$$

$$= \begin{bmatrix}\widetilde{\mathbf{\Omega}}_{11}^T & \widetilde{\mathbf{\Omega}}_{21}^T\end{bmatrix}\begin{bmatrix}T_{2p+1}(\mathbf{\Sigma}_1) & \\ & T_{2p+1}(\mathbf{\Sigma}_2)\end{bmatrix}\begin{bmatrix}T_{2p+1}(\mathbf{\Sigma}_1) & \\ & T_{2p+1}(\mathbf{\Sigma}_2)\end{bmatrix}\begin{bmatrix}\widetilde{\mathbf{\Omega}}_{11} \\ \widetilde{\mathbf{\Omega}}_{21}\end{bmatrix} \tag{3.123}$$

$$= \widetilde{\mathbf{\Omega}}_{11}^T T_{2p+1}^2(\mathbf{\Sigma}_1)\widetilde{\mathbf{\Omega}}_{11} + \widetilde{\mathbf{\Omega}}_{21}^T T_{2p+1}^2(\mathbf{\Sigma}_2)\widetilde{\mathbf{\Omega}}_{21} \tag{3.124}$$

it follows that

$$\sigma_{\min}\left(\widetilde{\mathbf{R}}_1^T\widetilde{\mathbf{R}}_1\right) = \sigma_k\left(\widetilde{\mathbf{\Omega}}_{11}^T T_{2p+1}^2(\mathbf{\Sigma}_1)\widetilde{\mathbf{\Omega}}_{11} + \widetilde{\mathbf{\Omega}}_{21}^T T_{2p+1}^2(\mathbf{\Sigma}_2)\widetilde{\mathbf{\Omega}}_{21}\right) \tag{3.125}$$

$$\geq \sigma_k\left(\widetilde{\mathbf{\Omega}}_{11}^T T_{2p+1}^2(\mathbf{\Sigma}_1)\widetilde{\mathbf{\Omega}}_{11}\right) \tag{3.126}$$

$$\geq \frac{\sigma_k\left(T_{2p+1}^2(\mathbf{\Sigma}_1)\right)}{\left\|\widetilde{\mathbf{\Omega}}_{11}^{-1}\right\|_2^2} \tag{3.127}$$

$$= \frac{T_{2p+1}^2(\sigma_k)}{\left\|\widetilde{\mathbf{\Omega}}_{11}^{-1}\right\|_2^2} \tag{3.128}$$

Combining the above, we have

$$\|\mathbf{H}\|_2 \leq \frac{T_{2p+1}(\sigma_{k+s+r+1})}{T_{2p+1}(\sigma_k)}\left\|\widetilde{\mathbf{\Omega}}_{41}\right\|_2\left\|\widetilde{\mathbf{\Omega}}_{11}^{-1}\right\|_2 \tag{3.129}$$

$$\leq T_{2p+1}^{-1}\left(1 + 2\cdot\frac{\sigma_k - \sigma_{k+s+r+1}}{\sigma_{k+s+r+1}}\right)\left\|\widetilde{\mathbf{\Omega}}_{41}\right\|_2\left\|\widetilde{\mathbf{\Omega}}_{11}^{-1}\right\|_2 \tag{3.130}$$

Finally, recalling our choice of $p = q$ gives the desired bound for σ_k. Making the same argument for $\sigma_j(\mathbf{B}_j) = \sigma_j(\mathbf{B}_k)$ gives the bound for $1 \leq j \leq k$. □

While in general we cannot say much about $\widetilde{\mathbf{\Omega}}_{41}$ and $\widetilde{\mathbf{\Omega}}_{11}$ in relation to the initial starting matrix $\mathbf{\Omega}$, when $\mathbf{\Omega}$ is a Gaussian random matrix, it is distributionally invariant under rotations, and so $\widetilde{\mathbf{\Omega}}$ is also a Gaussian random matrix. Under these assumptions, $\widetilde{\mathbf{\Omega}}_{11}$ is non-singular with high probability. In fact, the following result holds.

Theorem 3.5.2. *Let $\mathbf{B}_k$ be the matrix returned by Alg. 7. Assume that $\mathbf{\Omega}$ is a random Gaussian matrix. Then, for any parameter choices r, s, with notation as before, for $j =$*

$1, \cdots, k$,

$$\mathbb{E}(\sigma_j(\mathbf{B}_k)) \geq \begin{cases} \frac{\sigma_j}{\sqrt{1+\mathcal{C}^2 T_{2q+1}^{-2}\left(1+2\cdot\frac{\sigma_j-\sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)}} & \text{for } r \geq 2 \\ \frac{\sigma_j}{1+\mathcal{C}^2 T_{2q+1}^{-2}\left(1+2\cdot\frac{\sigma_j-\sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)\log\sqrt{\mathcal{C}^2+T_{2q+1}^{2}\left(1+2\cdot\frac{\sigma_j-\sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)}} & \text{for } r = 1 \\ \frac{\sigma_j}{1+\mathcal{C} T_{2q+1}^{-1}\left(1+2\cdot\frac{\sigma_j-\sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)} & \text{for } r = 0 \end{cases} \tag{3.131}$$

for

$$\mathcal{C} = \left(\sqrt{n-b+r}+\sqrt{r}+7\right)\left(\frac{4e\sqrt{b}}{r+1}\right) \tag{3.132}$$

Note the similarity between the randomized block Lanczos results Cor. 3.5.1 and Thm. 3.5.2 with the randomized subspace iteration results Thms. 1.4.1 and 1.4.2.

Since Corollary 3.5.1 is valid for all parameter choices of s, we may optimize over it to achieve a spectrum independent singular value convergence bound. We state and prove this result in the next Theorem.

Theorem 3.5.3. *Let $\mathbf{B}_k$ be the matrix returned by running Alg. 7 with the block size $b = k$. Assume $\mathbf{\Omega}$ is chosen such that $\widetilde{\mathbf{\Omega}}_{11}$ is nonsingular. Then, for $j = 1, \cdots, k$*

$$\sigma_j \geq \sigma_j\left(\mathbf{B}_k\right) \geq \sigma_j e^{\mathcal{O}\left(-\frac{\log(\mathcal{A}(4q+2))^2}{(4q+2)^2}\right)} \tag{3.133}$$

where

$$\mathcal{A} = 2\left\|\widetilde{\mathbf{\Omega}}_{41}\right\|_2\left\|\widetilde{\mathbf{\Omega}}_{11}^{-1}\right\|_2 \tag{3.134}$$

is a constant independent of q.

Proof. For every $f > 0$, choose s in Corollary 3.5.1 to be the largest index such that

$$\sigma_{j+s} \geq \frac{\sigma_j}{1+f} \tag{3.135}$$

Since s is the largest such index, it follows that

$$\sigma_{j+s+r+1} \leq \frac{\sigma_j}{1+f} \tag{3.136}$$

Then

$$\frac{\sigma_j - \sigma_{j+s+r+1}}{\sigma_{j+s+r+1}} \geq \frac{\sigma_j - \frac{\sigma_j}{1+f}}{\frac{\sigma_j}{1+f}} = f \tag{3.137}$$

Using the Chebyshev polynomial estimation in Theorem 3.1.7, we have

$$T_{2q+1}\left(1+2\cdot\frac{\sigma_j-\sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)\geq 2^{-1}\cdot 2^{(2q+1)\sqrt{2f}} \tag{3.138}$$

$$\Longrightarrow T_{2q+1}^{-2}\left(1+2\cdot\frac{\sigma_j-\sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)\leq 2^2\cdot 2^{-(4q+2)\sqrt{2f}} \tag{3.139}$$

With the definition $\mathcal{A}=2\left\|\widetilde{\mathbf{\Omega}}_{41}\right\|_2\left\|\widetilde{\mathbf{\Omega}}_{11}^{-1}\right\|_2$, the singular value inequality bound in Corollary 3.5.1 becomes

$$\sigma_j\geq\sigma_j(\mathbf{B}_k)\geq\frac{\sigma_{j+s}}{\sqrt{1+\mathcal{C}_{b\geq k+r}^2T_{2q+1}^{-2}\left(1+2\cdot\frac{\sigma_j-\sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)}} \tag{3.140}$$

$$\geq\frac{\frac{\sigma_j}{1+f}}{\sqrt{1+2^{-2}\mathcal{A}^2\cdot 2^2\cdot 2^{-(4q+2)\sqrt{2f}}}} \tag{3.141}$$

$$=\frac{\sigma_j}{\sqrt{(1+f)^2(1+\mathcal{A}^2\cdot 2^{-(4q+2)\sqrt{2f}})}} \tag{3.142}$$

, valid for any choice of $f>0$. In particular, take

$$f=\frac{1}{2}\left(\frac{\log(\mathcal{A}^2(4q+2)^2)}{\log 2}\right)^2\frac{1}{(4q+2)^2} \tag{3.143}$$

Then, we can estimate the right-hand side of Eqn. (3.142) as follows.

$$(1+f)^2\leq e^{\left(\frac{\log(\mathcal{A}^2(4q+2)^2)}{\log 2}\right)^2\frac{1}{(4q+2)^2}} \tag{3.144}$$

and

$$\left(1+\mathcal{A}^2\cdot 2^{-(4q+2)\sqrt{2f}}\right)\leq e^{\frac{1}{(4q+2)^2}} \tag{3.145}$$

imply

$$\left((1+f)^2\left(1+\mathcal{A}^2\cdot 2^{-(4q+2)\sqrt{2f}}\right)\right)^{-1/2}\geq\left(e^{-\frac{1}{(4q+2)^2}-\left(\frac{\log\left(\mathcal{A}^2(4q+2)^2\right)}{\log 2}\right)^2\frac{1}{(4q+2)^2}}\right)^{1/2} \tag{3.146}$$

$$\geq e^{-\left(\frac{\log\left(\mathcal{A}^2(4q+2)^2\right)}{\log 2}\right)^2\frac{1}{(4q+2)^2}} \tag{3.147}$$

Asymptotically in q, this bound states

$$\sigma_j\geq\sigma_j\left(\mathbf{B}_k\right)\geq\sigma_j e^{\mathcal{O}\left(-\frac{\log(\mathcal{A}(4q+2))^2}{(4q+2)^2}\right)} \tag{3.148}$$

as desired.

□

Remark 3.5.1. *Expressed another way, this asymptotic lower bound implies that to attain a* $1+\epsilon$ *multiplicative error, i.e.* $\sigma_j(\mathbf{B}_k) \geq (1+\epsilon)\sigma_j$*, the algorithm should be allowed to iterate for* $q = \mathcal{O}\left(\frac{n}{\sqrt{\epsilon}}\right)$ *iterations. This is in line with the result stated and proved in [33].*

3.6 Superlinear Convergence

We show that under certain assumptions about the singular value spectrum of $\mathbf{A}$, our result in Theorem 3.3.5 implies that the block Lanczos algorithm converges superlinearly.

Recall that a sequence $\{a_n\}$ converges **superlinearly** to a if

$$\lim_{n\to\infty} \frac{|a_{n+1} - a|}{|a_n - a|} = 0 \tag{3.149}$$

Theorem 3.6.1. *Assume the singular value spectrum of* $\mathbf{A}$ *decays such that* $\sigma_j \to 0$*. Let* $\mathbf{B}_k$ *be the rank* k *approximation of* $\mathbf{A}$ *returned by Alg. 7. Assume additionally that the hypothesis and notation of Theorem 3.3.5 hold. Then*

$$\sigma_j(\mathbf{B}_k) \to \sigma_j \tag{3.150}$$

superlinearly in q*, the number of iterations.*

Proof. The statement of the lemma is equivalent to the statement that

$$\mathcal{C}\, T_{2p+1}^{-1}\left(1 + 2\cdot\frac{\sigma_j - \sigma_{j+r+1}}{\sigma_{j+r+1}}\right) \to 0 \tag{3.151}$$

superlinearly. For notational convenience we assume σ_j is not a multiple singular value and we have chosen $s = 0$. (See Remark 3.3.2; otherwise, our following argument can be made for the largest choice of s such that $\sigma_{j+s} = \sigma_j$.)

For any fixed $j = 1, \cdots, k$, define

$$a_q \equiv \mathcal{C}(r) T_{2p+1}^{-1}\left(1 + 2\cdot\frac{\sigma_j - \sigma_{j+r+1}}{\sigma_{j+r+1}}\right) \tag{3.152}$$

$$= \mathcal{C}(r) T_{2\left(q+1-\frac{k+r}{b}\right)+1}^{-1}\left(1 + 2\cdot\left(\left(\frac{j+r+1}{j}\right)^{\tau} - 1\right)\right) \tag{3.153}$$

where we have explicitly specified the dependence of the constant $\mathcal{C}$ on the analysis parameter r, and made use of the assumption that $\sigma_j \sim 1/j^{\tau}$. Using Theorem 3.1.7 we can approximate

$$a_q \approx \mathcal{C}(r)\cdot\frac{1}{2}\left(1 + g + \sqrt{2g}\right)^{-\left(2\left(q+1-\frac{k+r}{b}\right)+1\right)} \tag{3.154}$$

$$\approx \frac{1}{2}\cdot\mathcal{C}(r)\cdot\left(1 + g + \sqrt{2g}\right)^{-2\left(1-\frac{k+r}{b}\right)+1}\cdot\left(1 + g + \sqrt{2g}\right)^{-2q} \tag{3.155}$$

$$\text{where } g = 2\cdot\frac{\sigma_j - \sigma_{j+r+1}}{\sigma_{j+r+1}} = 2\cdot\left(\frac{\sigma_j}{\sigma_{j+r+1}} - 1\right) \tag{3.156}$$

Table 3.1: Summary of convergence results

citation	bound	req. on b
Saad	$\lambda_j^{(q)} \geq \frac{\lambda_j}{1+{L_j^{(q)}}^2 \tan^2\Theta(\mathbf{U},\mathbf{V}) T_{q-j}^{-2}\left(1+2\frac{\lambda_j-\lambda_{j+b}}{\lambda_{j+b}}\right)}$	$b \geq k$
Musco - spec. indep.	$\sigma_j^{(q)} \geq \frac{\sigma_j}{\sqrt{1+\mathcal{C}^2\log^2(n)\, q^{-2}\frac{\sigma_{b+1}^2}{\sigma_j^2}}}$	$b \geq k$
Musco - spec. dep.	$\sigma_j^{(q)} \geq \frac{\sigma_j}{\sqrt{1+\mathcal{C}ne^{-q\sqrt{\min(1,\sigma_k/\sigma_{b+1}-1)}}\frac{\sigma_{b+1}^2}{\sigma_j^2}}}$	$b \geq k$
Current Work	$\sigma_j^{(q)} \geq \frac{\sigma_j}{\sqrt{1+\mathcal{C}^2 T_{2q+1-2(k+r)/b}^{-2}\left(1+2\frac{\sigma_j-\sigma_{j+r+1}}{\sigma_{j+r+1}}\right)}}$	$b \geq 1$, $bq \geq k+r$

Then we argue that $a_{q+1}/a_q \to 0$ as follows.

$$\frac{a_{q+1}}{a_q} = \frac{1}{\left(1+g+\sqrt{2g}\right)^2} \leq \frac{1}{(1+g)^2} \tag{3.157}$$

Since we assumed a spectrum such that $\sigma_j \to 0$ eventually, it is possible to choose r sufficiently large such that $1/(1+g)^2$ is arbitrarily small.

It may seem strange that here in order to make our argument we chose the oversampling parameter r instead of the parameter q, on which the sequence is indexed by. Recall that r and q are related by $k+r = (q-p+1)b$. Therefore, choosing r to be sufficiently large is in effect choosing q to be so. Furthermore, since the inequality in Theorem 3.3.5 holds for all choices of r and r itself is only a parameter in the analysis and not the algorithm, the conclusion holds. □

Remark 3.6.1. *Rigorously, the above argument applies only to infinite dimensional operators, as in the finite dimensional case, $r \leq n$ cannot be chosen to be arbitrarily large. However, numerous previous works have noted that in practice, the convergence does tend to exhibit superlinear behavior [41].*

Remark 3.6.2. *See the next chapter for an illustration of some spectrums of matrices from real data applications - by and large, they do exhibit the kind of characteristics that are assumed in this theorem.*

3.7 Summary

A summary of all applicable results for the Randomized Subspace Iteration algorithm 7 along with their restrictions, translated to comparable form, appears in Table 3.1.

Chapter 4

Numerical Experiments

In the previous chapter, one of our main results was the following generalized bound, proved in Thm. 3.3.5, and reproduced below for convenience.

$$\sigma_j \geq \sigma_j\left(\mathbf{B}_k\right) \geq \frac{\sigma_{j+s}}{\sqrt{1+\mathcal{C}^2 T_{2p+1}^{-2}\left(1+2\cdot\frac{\sigma_j-\sigma_{j+s+r+1}}{\sigma_{j+s+r+1}}\right)}} \tag{4.1}$$

where p is given by the expression $k+r=(q-p+1)b$.

While this bound exhibits attractive asymptotic convergence behavior, it remains to be seen that the desirable convergence behavior manifests itself in the regime of practical data problems. For one, it is possible that the constant $\mathcal{C}$ is large enough to totally overwhelm the Chebyshev convergence term in all practical settings. The purpose of our numerical experiments is in part to show that this does not occur. Additionally, the above bound holds for all valid choices of the oversampling parameter r, whose effect enters the bound in both the spectral "gap" term and in the degree of the Chebyshev polynomial. In our experiments, we show that it is generally beneficial to select an oversampling of $r>0$. Finally, we show that for practical data matrices, the assumptions we have made about the decay of their spectra in Section 3.6 is true in most cases, and the Randomized Block Lanczos does exhibit superlinear convergence when applied to these matrices.

For our numerical experiments, we implemented the RSI and RBL Algorithms (Alg. 6 and 7 respectively) as MATLAB routines. As outlined, Algorithm 7 consists of the pseudocode steps, whereas the actual implementation reorganizes the computations into a block bidiagonalization step followed by an SVD step on the bidiagonal matrix. Both algorithms were implemented with full orthogonalization at every iteration for simplicity, although numerous other orthogonalization schemes exist. In both the computational complexity discussion and the numerical experiments below, we refer to the actual implementation and not the pseudocode. See Appendix B for the relevant code snippets.

4.1 Computational Complexity

We give an arithmetic complexity accounting of the Randomized Block Lanczos algorithm. The initialization of the random starting matrix $\mathbf{\Omega}$ takes $\mathcal{O}(nb)$ floating-point operations (flops). The formation of the Krylov matrix $\mathbf{K}$ consists of 1 matrix multiplications of $\mathbf{A\Omega}$ along with $2q$ accumulated applications of either $\mathbf{A}$ or $\mathbf{A}^T$ for a total dominating cost of $\mathcal{O}(mnbq)$ flops. (There are also $2q$ matrix-multiplications involving the $\mathcal{A}$ and $\mathcal{B}$ blocks of the bidiagonal matrix, but because of the smaller matrix dimensions, this cost is insignificant.) The orthonormal basis $\mathbf{Q}$ of $\mathbf{K}$ is computed implicitly at each iteration using a QR factorization, the standard Householder implementation of which has complexity $\mathcal{O}(m(b)^2)$, for a total cost of $\mathcal{O}(mb^2q)$. Finally, the truncated SVD factorization is performed on the bidiagonal matrix. Because the dimension of this matrix is $((q+1)b) \times ((q+1)b)$ and we expect $(q+1)b \approx k$ to be small, we assume its SVD computation is performed with a non-specialized dense matrix algorithm, using $\mathcal{O}((bq)^3)$ flops. The final step of forming the approximation matrix $\mathbf{B}_k$ is an additional $\mathcal{O}(m(bq)^2)$ flops.

Overall, the computational complexity of Algorithm 7 is $\mathcal{O}(mnbq + m(bq)^2)$. The first dominating term is the result of performing the matrix multiplications for the computation of the Lanczos block vectors. Fortunately, matrix multiplication is a highly optimized and highly tuned part of many matrix computation libraries, especially for suitably chosen block sizes.

We draw attention to the fact that that the parameters b and q only appear together as the quantity bq in our computational complexity accounting. This suggests that we may freely vary b, q and as long as they vary inversely and the quantity bq remains constant, the time for running Algorithm 7 remains comparable. (In practice, the story is more complicated. For dense matrices, due to the efficiency of BLAS3 operations compared with BLAS1/2 operations, choices for b larger than the CPU cache block size, and input matrices that are more "square", will generally be more performant. For sparse matrices, the time complexity will instead be determined by the number of nonzeros entries of the matrix, in addition to the memory layout of the storage of these entries.) Given the comparable computational complexity, and assuming the conditions for the convergence of Algorithm 7 is met, we need not privilege the block size choice $b = k$. In fact, the next section, we show empirically that in many cases, it is beneficial in terms of approximation accuracy to choose block sizes b strictly smaller than k.

4.2 Constructed Matrices

We perform two sets of numerical experiments to empirically profile the convergence behavior of Algorithm 7. Firstly, we observe the convergence properties of the algorithm on a set of constructed matrices. We aim to show the effect of varying the block size b on matrices with different spectrum decay behaviors.

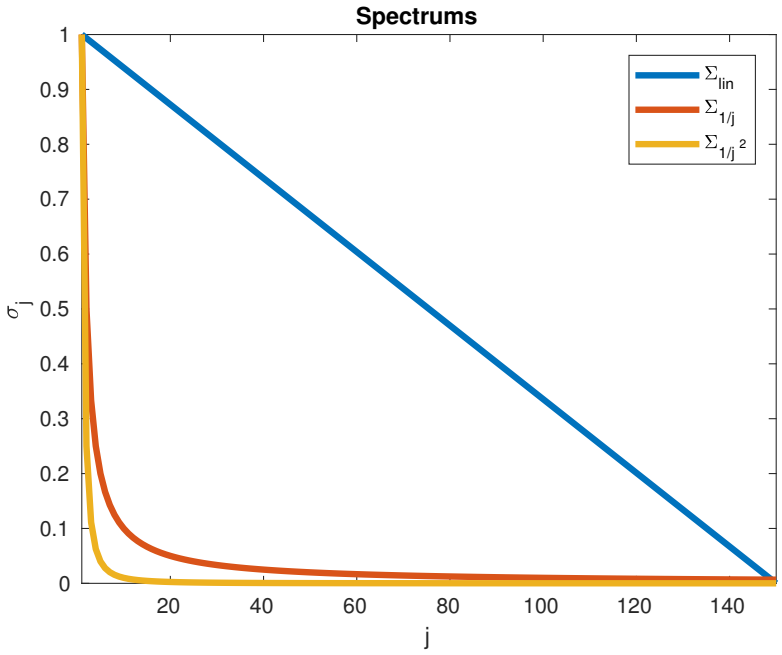


Figure 4.1: The singular value spectrums of $\mathbf{A}_{\text{lin}}$, $\mathbf{A}_{1/j}$ and $\mathbf{A}_{1/j^2}$.

To construct matrices with specific singular spectrums, we form them as

$$\mathbf{A}_{(\cdot)} = \mathbf{U}\mathbf{\Sigma}_{(\cdot)}\mathbf{V}^T \tag{4.2}$$

with random orthonormal matrices $\mathbf{U}$ and $\mathbf{V}$. All matrices that we test are of dimensions $\mathbf{A}_{(\cdot)} \in \mathbb{R}^{10000\times 150}$.

Our test matrices $\mathbf{A}_{\text{lin}}$, $\mathbf{A}_{1/j}$, and $\mathbf{A}_{1/j^2}$ have singular spectrums decaying linearly, and as $\sigma_j = \frac{1}{j}$ and $\sigma_j = \frac{1}{j^2}$ respectively. See Figure 4.1 for plots of $\mathbf{\Sigma}_{\text{lin}}$, $\mathbf{\Sigma}_{1/j}$ and $\mathbf{\Sigma}_{1/j^2}$.

For this set of experiments, we target a rank of $k = 10$. To compare runs of the algorithm with comparable number of flops, we fix $bq = 20$. To that end, we vary the block size parameter as $b = 20, 10, 5, 4, 2, 1$, and the Lanczos iteration parameter as $q = 1, 2, 4, 5, 10, 20$ respectively. This means, for example, for a block size of $b = 5$, $q = 4$ iterations of the RBL algorithm were run. For each (b, q) parameter setting, 500 runs of the algorithm were performed and the results averaged. The results appear in Figures 4.2, 4.3, 4.4 respectively.

Each line on this set of plots represents the result of running RBL with a single set of parameters - block size b, and $q = 20/b$ iterations. Each marker on the line represents the relative error of a single singular value, $\widetilde{\sigma}_1, \cdots, \widetilde{\sigma}_{10}$. The y-axis is in log scale, and so further down (less relative error) is good. As expected, the faster the spectrum decays, the more converged the singular values. We observe that for all parameter settings and all spectrums, $\widetilde{\sigma}_1$ converges the most, followed by $\widetilde{\sigma}_2$, etc. Comparing across block sizes, we see that smaller block sizes generally perform better, and this is especially pronounced for the faster decaying spectrums $\mathbf{\Sigma}_{1/j}$ and $\mathbf{\Sigma}_{1/j^2}$.

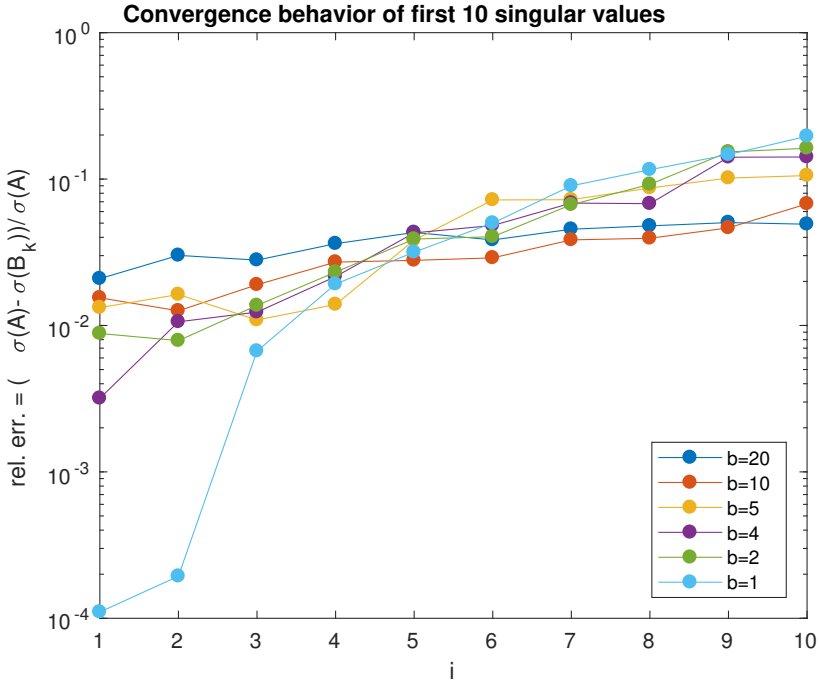


Figure 4.2: First $k = 10$ singular values convergence behavior, for the matrix $\mathbf{A}_{\text{lin}}$.

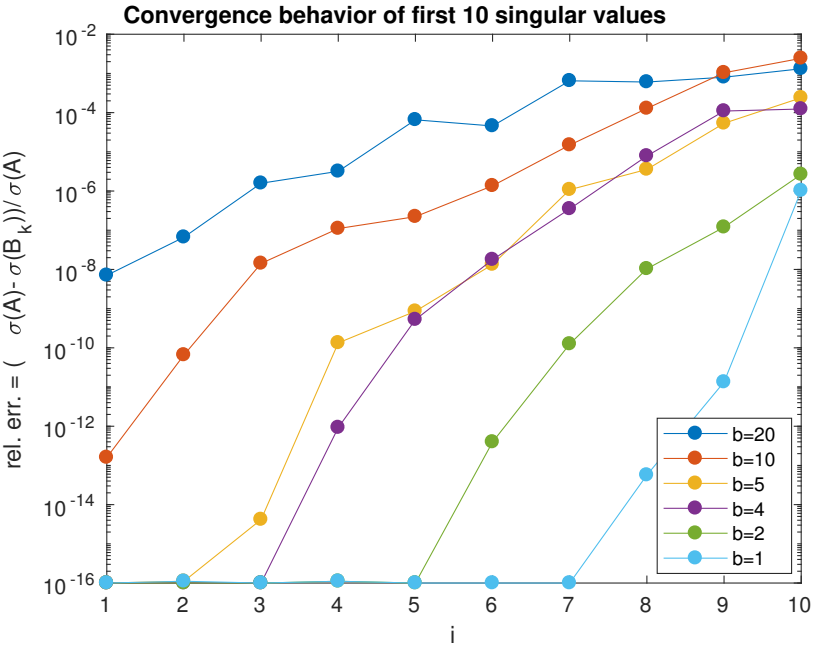


Figure 4.3: First $k = 10$ singular values convergence behavior, for the matrix $\mathbf{A}_{1/j}$.

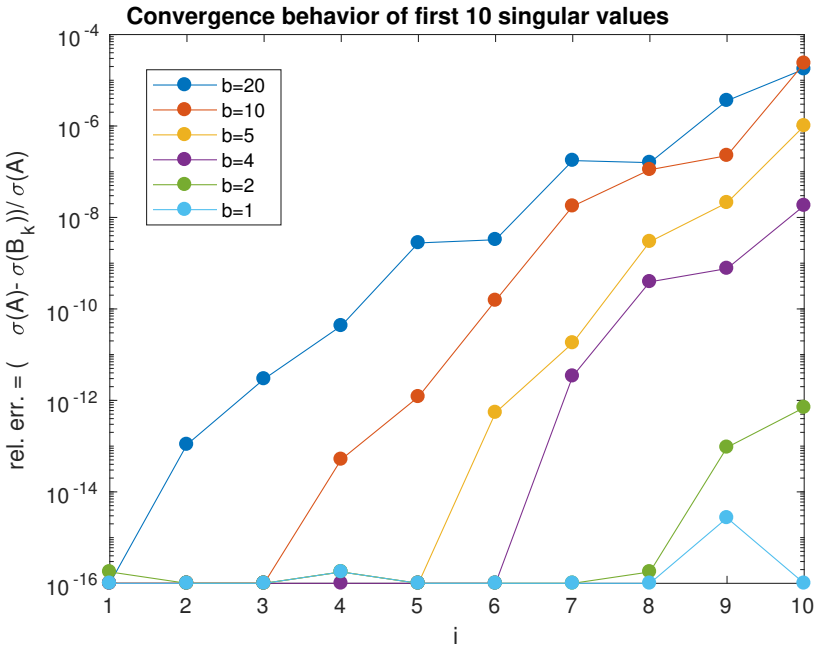


Figure 4.4: First $k = 10$ singular values convergence behavior, for the matrix $\mathbf{A}_{\text{lin}}$.

4.3 Activities and Sports Dataset

The Activities and Sports Dataset is a dataset consisting of motion sensor data for 8 subjects performing 19 daily/sports activities, for 5 minutes, sampled at 25Hz frequency. This dataset can be found at [6].

The matrix associated with this dataset is dense and of dimension $\mathbf{A} \in \mathbb{R}^{9120\times 5625}$, where each row is a sample and each entry is a double precision float. Figure 4.5 shows a plot of the first 500 singular values of $\mathbf{A}$. As is typically for data matrices, this matrix exhibits spectrum decay on the order of $\sigma_j = \frac{1}{j^\tau}$, for constant $\tau > 1$, and our theory suggests that in this case, we should observe superlinear convergence for RBL.

In this set of experiments, we investigate the convergence of a single singular value with respect to the number of iterations, in addition to the affect of the block size on convergence. We run the RSI and RBL algorithms on the Activities and Sports Dataset matrix with a target rank of $k = 200$, and examine the convergence of σ_1, σ_{100}, and σ_{200}. The results of these experiments are in Figures 4.6, 4.7, and 4.8.

Each of these plots represent the convergence of a particular singular value. In each plot, each line represents a single parameter setting for the block size b, for either the RSI or the RBL algorithm. The y-axis is in log scale, and denotes the relative error of the particular singular value we are examining. The x-axis is in linear scale, and denotes the value of bq (a proxy measure for computational complexity). Markers on each line represent successive iterations of the algorithm. In this plot, down and to the left is good - we seek parameter

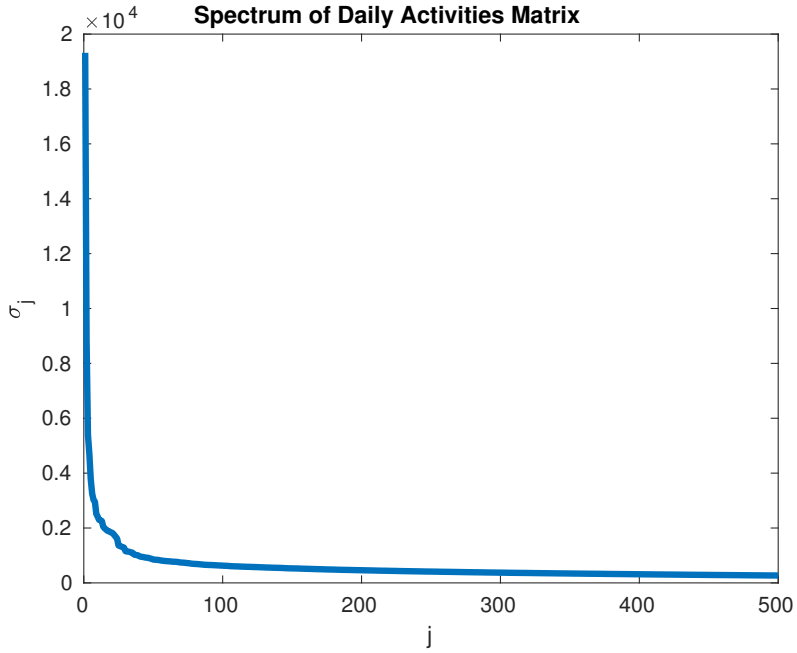


Figure 4.5: First 500 singular values of the Daily Activities and Sports Matrix.

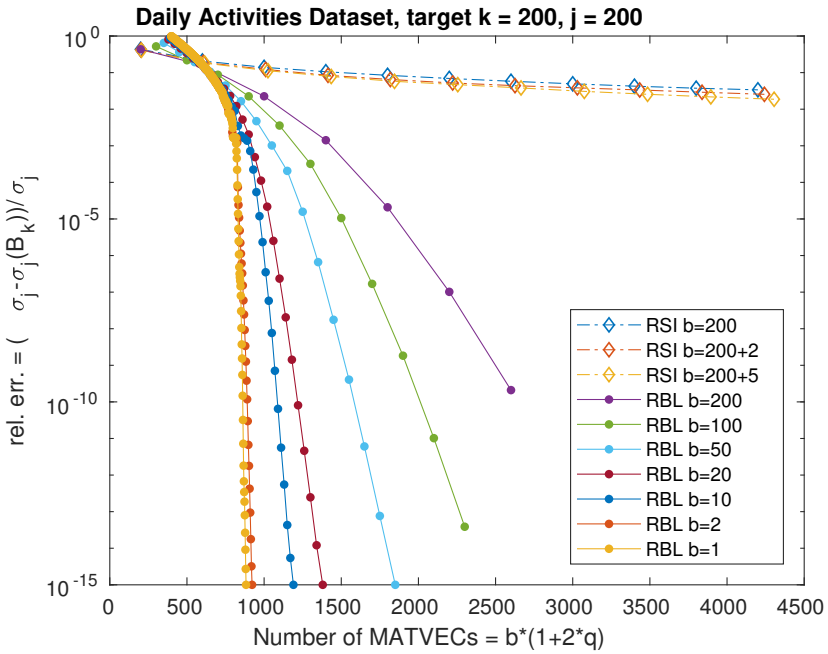


Figure 4.6: $k = 200$ approximation of the Daily Activities Dataset, convergence of σ_{200}.

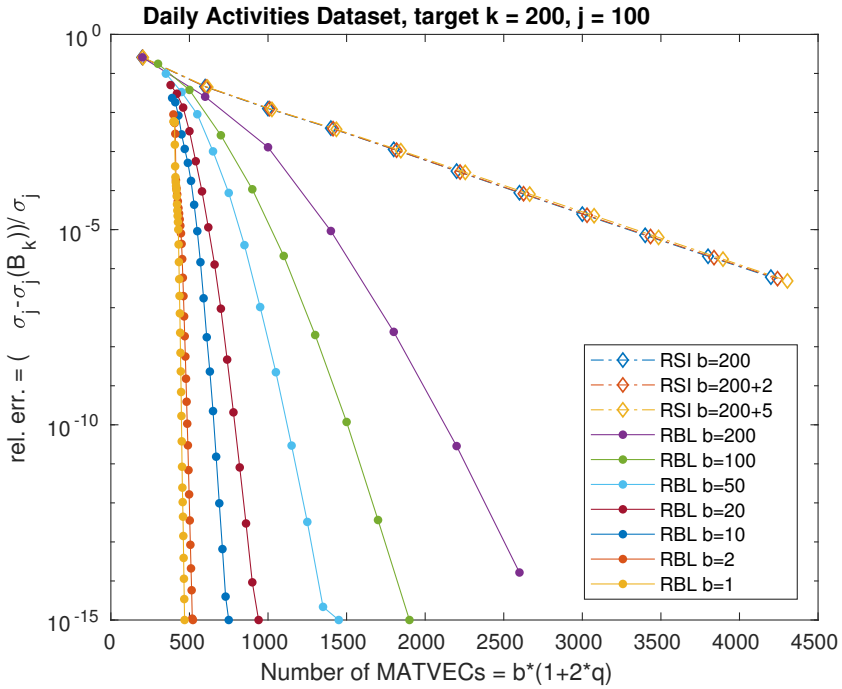


Figure 4.7: $k = 200$ approximation of the Daily Activities Dataset, convergence of σ_{100}.

settings that give good convergence for less computational complexity. We observe that, as expected, RSI converges linearly and RBL converges superlinearly. These trends are most clearly seen in Figure 4.6 and is also present in Figure 4.7. The convergence of σ_1 is extremely rapid in Figure 4.8, and reaches double precision in 2-5 iterations for all block sizes. In fact, the convergence is so rapid for the first singular value that, by the time the last singular value, σ_{200}, becomes available, the relative error for σ_1 has dropped to machine epsilon for block sizes $b = 20, 10, 2, 1$. In all cases, for both RBL and RSI, it appears that, at the same computational complexity, choosing a smaller block size b, leads to more rapid convergence. For example, in Figure 4.6, we observe that in order for σ_j to converge to a relative error of $\sim 10^{-5}$, taking $b = 1$ uses $1/2$ the number of flops as taking $b = k = 200$.

4.4 Eigenfaces Dataset

The Eigenfaces dataset is available from the AT&T Laboratories Cambridge's Database of Faces [42], and consists of 10 different face images of 40 different subjects at 92×112 pixels resolution, varying in light, facial expressions, and other details. The widely cited technique for processing this data is via PCA [47], where it was observed that each face can composed in large part from a few prominent "Eigenfaces".

The associated matrix is a dense matrix, which is formed by vectorizing each different

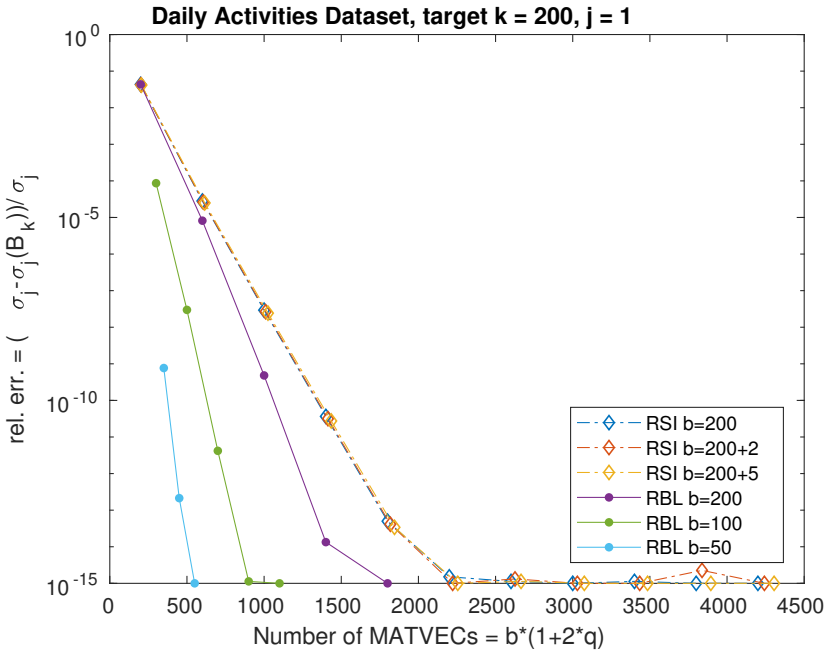


Figure 4.8: $k = 200$ approximation of the Daily Activities Dataset, convergence of σ_1. Smaller values of b are omitted as, for the first singular value σ_1, all have converged to machine epsilon by the iteration at which the last singular value at the target rank, σ_{200}, is first computed.

face image as a column vector, and has dimensions $\mathbf{A} \in \mathbb{R}^{10304\times 400}$ and it is of full numerical rank. The spectrum of this matrix spans 5 orders of magnitude but decays extremely rapidly, typical of data matrices. In fact, as seen in Figure 4.9, the noise level appears to begin around the 50th singular value.

We repeat the experiments performed for the Activities and Sports Dataset. For this set of experiments, we use the RSI and RBL algorithms to compute rank-$k = 100$ approximations for the Eigenfaces matrix, and examine the convergence of σ_{100}. The result appears in Figure 4.10.

We observe similar behavior as those observed for the Daily Activities and Sports Matrix: the RSI algorithm exhibits linear convergence while the RBL algorithm exhibits superlinear convergence; smaller block sizes b appear to converge more quickly for a fixed number of flops.

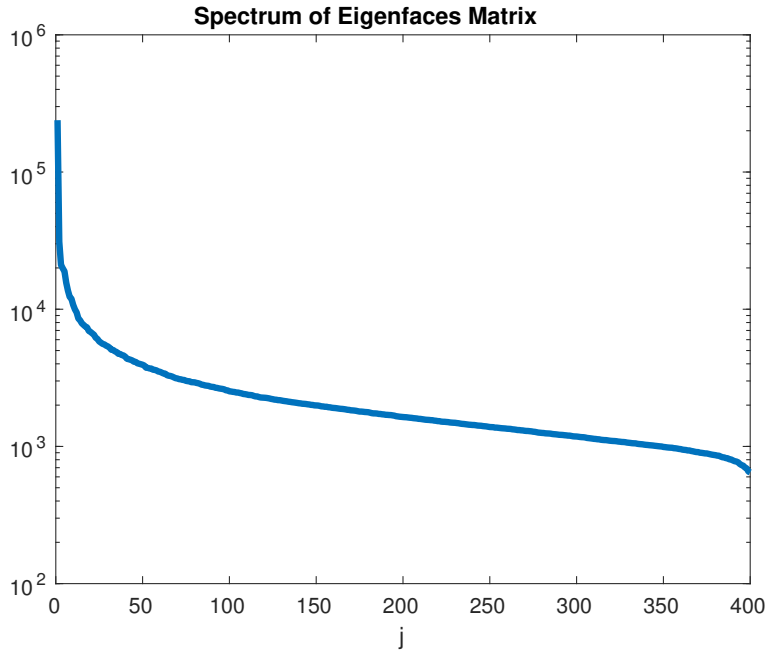


Figure 4.9: Spectrum of the Eigenfaces Matrix.

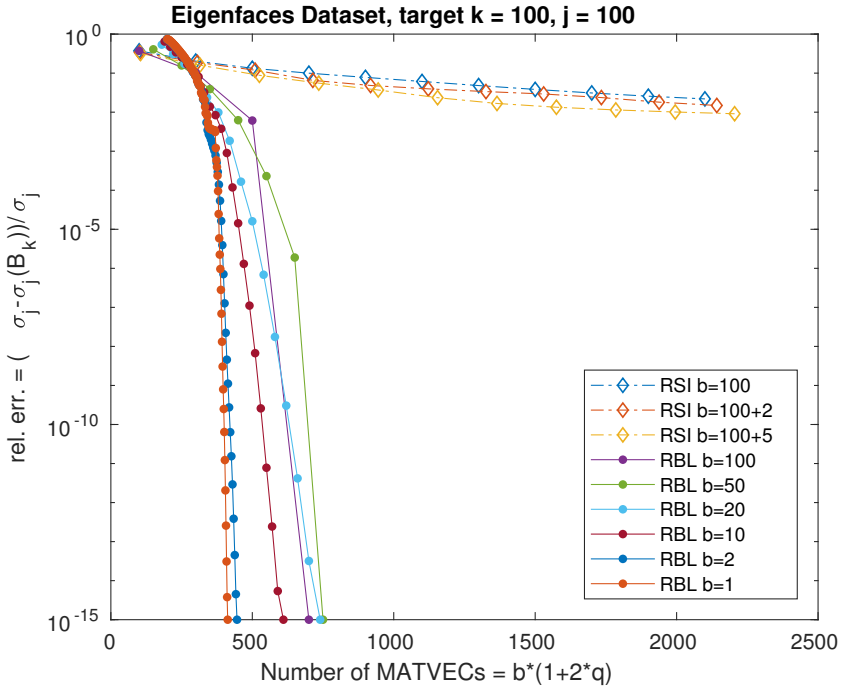


Figure 4.10: $k = 100$ approximation of the Eigenfaces Dataset, convergence of σ_{100}.

Appendix A

Proofs of Lemmas

For completeness, here we include the proofs of some lemmas that were used in the course of this thesis, but whose proofs are rather computational and not illuminative of the main ideas.

A.1 Proof of Vandermonde matrix reduction Lemma

Lemma (3.4.1)**.** *Let* $\mathbf{V}$ *be the* $n \times (l+1)$ *Vandermonde matrix*

$$\mathbf{V} = \begin{bmatrix} 1 & v_1 & v_1^2 & \cdots & v_1^l \\ 1 & v_2 & v_2^2 & \cdots & v_2^l \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & v_n & v_n^2 & \cdots & v_n^l \end{bmatrix} \tag{A.1}$$

If the constants v_j *are distinct, and* $n > l+1$*, then there exist non-singular matrix* $\mathbf{X} \in \mathbb{R}^{(l+1)\times(l+1)}$ *such that*

$$\mathbf{VX} = \begin{bmatrix} \mathbf{I} \\ \mathbf{H} \end{bmatrix} \tag{A.2}$$

where the entries of $\mathbf{H} \in \mathbb{R}^{(n-(l+1))\times(l+1)}$ *are given by*

$$\mathbf{H}_{ij} = \frac{(v_i - v_1)\cdots(v_i - v_{j-1})(v_i - v_{j+1})\cdots(v_i - v_{l+1})}{(v_j - v_1)\cdots(v_j - v_{j-1})(v_j - v_{j+1})\cdots(v_j - v_{l+1})} = \prod_{\substack{t=1 \\ t\neq j}}^{l+1} \frac{v_i - v_t}{v_j - v_t} \tag{A.3}$$

for

$$l+2 \leq i \leq n$$
$$1 \leq j \leq l+1$$

Proof. We use a series of elimination steps to introduce successive zeros and reduce the Vandermonde matrix to the desired form. To begin, define

$$\mathbf{X}_{ij}^{k} = \mathbf{I} - v_k e_i e_j^T = \begin{bmatrix} 1 & & & & \\ & 1 & & -v_k & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix} \tag{A.4}$$

i.e. $\mathbf{X}_{ij}^{k} \in \mathbb{R}^{(l+1)\times(l+1)}$ is a matrix with 1s down the diagonal, and $-v_k$ in the (i,j)th entry.

Now, we introduce successive zeros. First, in the $(1, l+1)$ entry:

$$\mathbf{V}\mathbf{X}_{l,l+1}^{1} = \begin{bmatrix} 1 & v_1 & \cdots & v_1^{l-1} & 0 \\ 1 & v_2 & \cdots & v_2^{l-1} & v_2^{l-1}(v_2 - v_1) \\ \vdots & \vdots & & \vdots & \vdots \\ \vdots & \vdots & & \vdots & \vdots \\ 1 & v_n & \cdots & v_n^{l-1} & v_n^{l-1}(v_n - v_1) \end{bmatrix} \tag{A.5}$$

Next, in the $(1, l)$ entry:

$$\mathbf{V}\mathbf{X}_{l,l+1}^{1}\mathbf{X}_{l-1,l}^{1} = \begin{bmatrix} 1 & v_1 & \cdots & v_1^{l-2} & 0 & 0 \\ 1 & v_2 & \cdots & v_2^{l-2} & v_2^{l-2}(v_2 - v_1) & v_2^{l-1}(v_2 - v_1) \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ 1 & v_n & \cdots & v_n^{l-2} & v_n^{l-2}(v_n - v_1) & v_n^{l-1}(v_n - v_1) \end{bmatrix} \tag{A.6}$$

Proceeding like this, we can successively multiplying by $\mathbf{X}$s to introduce zeros in the $(1, l-1)$ to $(1, 2)$ entries, completing the first row:

$$\mathbf{V}\mathbf{X}_{l,l+1}^{1}\mathbf{X}_{l-1,l}^{1}\cdots\mathbf{X}_{1,2}^{1} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & (v_2 - v_1) & v_2(v_2 - v_1) & \cdots & v_2^{l-2}(v_2 - v_1) & v_2^{l-1}(v_2 - v_1) \\ 1 & (v_3 - v_1) & v_3(v_3 - v_1) & \cdots & v_3^{l-2}(v_3 - v_1) & v_3^{l-1}(v_3 - v_1) \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 1 & (v_n - v_1) & v_n(v_n - v_1) & \cdots & v_n^{l-2}(v_n - v_1) & v_n^{l-1}(v_n - v_1) \end{bmatrix} \tag{A.7}$$

Next, we use the same procedure to complete the second row, by successively introducing

zeros in the $(2, l+1)$ to $(2,3)$ entries, in that order.

$$\mathbf{V}\left(\mathbf{X}^1_{l,l+1}\cdots\mathbf{X}^1_{1,2}\right)\left(\mathbf{X}^2_{l,l+1}\cdots\mathbf{X}^2_{2,3}\right) \tag{A.8}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & (v_2-v_1) & 0 & 0 & \cdots & 0 \\ 1 & (v_3-v_1) & (v_3-v_1)(v_3-v_2) & v_3(v_3-v_1)(v_3-v_2) & \cdots & v_3^{l-2}(v_3-v_1)(v_3-v_2) \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & (v_n-v_1) & (v_n-v_1)(v_n-v_2) & v_n(v_n-v_1)(v_n-v_2) & \cdots & v_n^{l-2}(v_n-v_1)(v_n-v_2) \end{bmatrix} \tag{A.9}$$

Proceeding in this manner, successively working on the rows from top to bottom, introducing zeros in each row from right to the diagonal, we can create a matrix whose upper triangular part is uniformly zero. More precisely,

$$\mathbf{V}\left(\mathbf{X}^1_{l,l+1}\cdots\mathbf{X}^1_{1,2}\right)\left(\mathbf{X}^2_{l,l+1}\cdots\mathbf{X}^2_{2,3}\right)\cdots\left(\mathbf{X}^{l-1}_{l,l+1}\mathbf{X}^{l-1}_{l-1,l}\right)\left(\mathbf{X}^l_{l,l+1}\right) \tag{A.10}$$

$$= \left[\begin{array}{cccc} 1 & & & \\ \vdots & (v_2-v_1) & & \\ \vdots & \vdots & (v_3-v_1)(v_3-v_2) & \\ \vdots & \vdots & \vdots & \ddots \\ 1 & (v_{l+1}-v_1) & (v_{l+1}-v_1)(v_{l+1}-v_2) & \cdots \quad (v_{l+1}-v_1)(v_{l+1}-v_2)\cdots(v_{l+1}-v_l) \\ \hline 1 & (v_{l+2}-v_1) & (v_{l+2}-v_1)(v_{l+2}-v_2) & \cdots \quad (v_{l+2}-v_1)(v_{l+2}-v_2)\cdots(v_{l+2}-v_l) \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ 1 & (v_n-v_1) & (v_n-v_1)(v_n-v_2) & \cdots \quad (v_n-v_1)(v_n-v_2)\cdots(v_n-v_l) \end{array}\right] \tag{A.11}$$

Denote

$$\mathbf{X}^{(l+1)} \equiv \frac{1}{(v_{l+1}-v_1)(v_{l+1}-v_2)\cdots(v_{l+1}-v_l)}\left(\mathbf{X}^1_{l,l+1}\cdots\mathbf{X}^1_{1,2}\right)\left(\mathbf{X}^2_{l,l+1}\cdots\mathbf{X}^2_{2,3}\right)\cdots\left(\mathbf{X}^{l-1}_{l,l+1}\mathbf{X}^{l-1}_{l-1,l}\right)\left(\mathbf{X}^l_{l,l+1}\right) \tag{A.12}$$

Then, the last $(l+1\text{st})$ column in the equality in Eqn. (A.11) gives the following relation:

$$\mathbf{V}\left[\mathbf{X}^{(l+1)}\right]_{:,l+1} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ \frac{(v_{l+2}-v_1)(v_{l+2}-v_2)\cdots(v_{l+2}-v_l)}{(v_{l+1}-v_1)(v_{l+2}-v_2)\cdots(v_{l+2}-v_l)} \\ \vdots \\ \frac{(v_n-v_1)(v_n-v_2)\cdots(v_n-v_l)}{(v_{l+1}-v_1)(v_{l+1}-v_2)\cdots(v_{l+1}-v_l)} \end{bmatrix} \tag{A.13}$$

Note this is the last column of our the desired Eqn. (A.2), where the division in Eqn. (A.12) makes use of the condition that the v_js are distinct.

Now, with the previous computation in mind, we show, in general, how to derive any column j in the desired Eqn. (A.2). The idea is to use the same $\mathbf{X}$s as before to introduce zeros, in all but the jth row.

We first introduce zeros to the right of the diagonal in the first $j-1$ rows.

$$\mathbf{V}\left(\mathbf{X}_{l,l+1}^{1}\cdots\mathbf{X}_{1,2}^{1}\right)\cdots\left(\mathbf{X}_{l,l+1}^{j-1}\cdots\mathbf{X}_{j-1,j}^{j-1}\right) \tag{A.14}$$

$$=\begin{bmatrix}
1 & & & & & & \\
1 & (v_2-v_1) & & & & & \\
\vdots & \vdots & \ddots & & & & \\
1 & (v_j-v_1) & \cdots & (v_j-v_1)\cdots(v_j-v_{j-1}) & \cdots & v_j^{l-(j-1)}(v_j-v_1)\cdots(v_j-v_{j-1}) \\
\vdots & \vdots & & \vdots & & \vdots \\
\vdots & \vdots & & \vdots & & \vdots \\
1 & (v_n-v_1) & \cdots & (v_n-v_1)\cdots(v_n-v_{j-1}) & \cdots & v_n^{l-(j-1)}(v_n-v_1)\cdots(v_n-v_{j-1})
\end{bmatrix} \tag{A.15}$$

Next, instead of introducing zeros in the jth row, we skip the jth row and introduce zeros from right to left in the $j+1$st row until the $(j+1,j)$ entry.

$$\mathbf{V}\left(\mathbf{X}_{l,l+1}^{1}\cdots\mathbf{X}_{1,2}^{1}\right)\cdots\left(\mathbf{X}_{l,l+1}^{j-1}\cdots\mathbf{X}_{j-1,j}^{j-1}\right)\left(\mathbf{X}_{l,l+1}^{j+1}\cdots\mathbf{X}_{j,j+1}^{j+1}\right) \tag{A.16}$$

$$=\begin{bmatrix}
1 & & & & & & \\
1 & (v_2-v_1) & & & & & \\
\vdots & \vdots & \ddots & & & & \\
1 & (v_j-v_1) & \cdots & (v_j-v_1)\cdots(v_j-v_{j-1})(v_j-v_{j+1}) & \cdots & v_j^{l-j}(v_j-v_1)\cdots(v_j-v_{j-1})(v_j-v_{j+1}) \\
1 & (v_{j+1}-v_1) & \cdots & 0 & \cdots & 0 \\
\vdots & \vdots & & \vdots & & \vdots \\
\vdots & \vdots & & \vdots & & \vdots \\
1 & (v_n-v_1) & \cdots & (v_n-v_1)\cdots(v_n-v_{j-1})(v_n-v_{j+1}) & \cdots & v_n^{l-j}(v_n-v_1)\cdots(v_n-v_{j-1})(v_n-v_{j+1})
\end{bmatrix} \tag{A.17}$$

Continue to introduce zeros in the subsequent rows until row $l+1$, in each row i reducing

from right to left until the $(i, i-1)$ entry.

$$\mathbf{V}\left(\mathbf{X}_{l,l+1}^{1}\cdots\mathbf{X}_{1,2}^{1}\right)\cdots\left(\mathbf{X}_{l,l+1}^{j-1}\cdots\mathbf{X}_{j-1,j}^{j-1}\right)\left(\mathbf{X}_{l,l+1}^{j+1}\cdots\mathbf{X}_{j,j+1}^{j+1}\right)\cdots\left(\mathbf{X}_{l,l+1}^{l+1}\right) \tag{A.18}$$

$$= \begin{bmatrix} 1 & & & & & & \\ 1 & (v_2-v_1) & & & & & \\ \vdots & \vdots & \ddots & & & & \\ 1 & (v_j-v_1) & \cdots & (v_j-v_1)\cdots(v_j-v_{j-1})(v_j-v_{j+1}) & \cdots & (v_j-v_1)\cdots(v_j-v_{j-1})(v_j-v_{j+1})\cdots(v_j-v_{l+1}) \\ \vdots & & & 0 & & 0 \\ \vdots & & & \vdots & & \vdots \\ \vdots & & & \vdots & & 0 \\ 1 & (v_{l+2}-v_1) & \cdots & (v_{l+2}-v_1)\cdots(v_{l+2}-v_{j-1})(v_{l+2}-v_{j+1}) & \cdots & (v_{l+2}-v_1)\cdots(v_{l+2}-v_{j-1})(v_{l+2}-v_{j+1})\cdots(v_{l+2}-v_{l+1}) \\ \vdots & \vdots & & \vdots & & \vdots \\ \vdots & \vdots & & \vdots & & \vdots \\ 1 & (v_n-v_1) & \cdots & (v_n-v_1)\cdots(v_n-v_{j-1})(v_n-v_{j+1}) & \cdots & (v_n-v_1)\cdots(v_n-v_{j-1})(v_n-v_{j+1})\cdots(v_n-v_{l+1}) \end{bmatrix} \tag{A.19}$$

(See Figure TODO for the non-zero pattern of this matrix.)

Analogous to Eqn. (A.11), denote

$$\mathbf{X}^{(j)} \equiv \frac{1}{(v_j-v_1)\cdots(v_j-v_{j-1})(v_j-v_{j+1})\cdots(v_j-v_{l+1})} \cdot\left(\mathbf{X}_{l,l+1}^{1}\cdots\mathbf{X}_{1,2}^{1}\right)\cdots\left(\mathbf{X}_{l,l+1}^{j-1}\cdots\mathbf{X}_{j-1,j}^{j-1}\right)\left(\mathbf{X}_{l,l+1}^{j+1}\cdots\mathbf{X}_{j,j+1}^{j+1}\right)\cdots\left(\mathbf{X}_{l,l+1}^{l+1}\right) \tag{A.20}$$

Then, the last $(l+1\text{st})$ column in the equality in Eqn. (A.19) gives the following relation:

$$\mathbf{V}\left[\mathbf{X}^{(j)}\right]_{:,l+1} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ \frac{(v_{l+2}-v_1)\cdots(v_{l+2}-v_{j-1})(v_{l+2}-v_{j+1})\cdots(v_{l+2}-v_{l+1})}{(v_j-v_1)\cdots(v_j-v_{j-1})(v_j-v_{j+1})\cdots(v_j-v_{l+1})} \\ \vdots \\ \frac{(v_n-v_1)\cdots(v_n-v_{j-1})(v_n-v_{j+1})\cdots(v_n-v_{l+1})}{(v_j-v_1)\cdots(v_j-v_{j-1})(v_j-v_{j+1})\cdots(v_j-v_{l+1})} \end{bmatrix} \tag{A.21}$$

Finally, applying Eqn. (A.21) for each $j = 1, \cdots, l+1$, we can column-wise construct a

matrix $\mathbf{X}$ as

$$\mathbf{VX} \equiv \mathbf{V}\left[\left[\mathbf{X}^{(1)}\right]_{:,l+1} \quad \cdots \quad \left[\mathbf{X}^{(j)}\right]_{:,l+1} \quad \cdots \quad \left[\mathbf{X}^{(l+1)}\right]_{:,l+1}\right] \tag{A.22}$$

$$= \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \\ h_{l+2,1} & \cdots & h_{l+2,l+1} \\ \vdots & & \vdots \\ h_{n,1} & \cdots & h_{n,l+1} \end{bmatrix} \tag{A.23}$$

where

$$h_{i,j} = \frac{(v_i - v_1)\cdots(v_i - v_{j-1})(v_i - v_{j+1})\cdots(v_i - v_{l+1})}{(v_j - v_1)\cdots(v_j - v_{j-1})(v_j - v_{j+1})\cdots(v_j - v_{l+1})} \tag{A.24}$$

as desired. □

Appendix B

Code reference

Below is the relevant MATLAB code snippet for the implementation of Algorithm 7 used for all numerical experiments.

```
function [s] = RBLbidiag(A, Omega, k, q, reorthog)

    [B] = lanczos_bidiag(A', A*Omega, q, reorthog);
    s = svds(B,k)';

end

function [B] = lanczos_bidiag(A, B0, q_iter, reorthog)

    [m,n] = size(A);
    b = size(B0,2);

    % preallocate space
    U = zeros(m,b*q_iter);
    V = zeros(n,b*q_iter);
    B = zeros(b*q_iter,b*(q_iter+1));

    u = zeros(m,b);
    [v,beta] = qr(B0,0);
    V(:,1:b) = v;

    for i=1:q_iter+1
        r = A*v-u*beta';
        if reorthog && i>1
            r = r-U(:,1:(i-1)*b)*(U(:,1:(i-1)*b)'*r);
        end
```

```
        [u,alpha] = qr(r,0);
        U(:,(i-1)*b+1:i*b) = u;
        B((i-1)*b+1:i*b,(i-1)*b+1:i*b) = alpha;
        p = A'*u-v*alpha;
        if reorthog
            p = p-V(:,1:i*b)*(V(:,1:i*b)'*p);
        end
        [v,beta] = qr(p,0);
        V(:,i*b+1:(i+1)*b) = v;
        B((i-1)*b+1:i*b,i*b+1:(i+1)*b) = beta';
    end
    B = B(1:b*(q_iter+1),1:b*(q_iter+1));

end
```

Bibliography

[1] Orly Alter, Patrick O Brown, and David Botstein. "Singular value decomposition for genome-wide expression data processing and modeling". In: *Proceedings of the National Academy of Sciences* 97.18 (2000), pp. 10101–10106.

[2] David Anderson and Ming Gu. "An Efficient, Sparsity-Preserving, Online Algorithm for Low-Rank Approximation". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, June 2017, pp. 156–165.

[3] Harry Andrews and C Patterson. "Singular value decompositions and digital image processing". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 24.1 (1976), pp. 26–53.

[4] James Baglama and Lothar Reichel. "Augmented implicitly restarted Lanczos bidiagonalization methods". In: *SIAM Journal on Scientific Computing* 27.1 (2005), pp. 19–42.

[5] Zhaojun Bai et al. *Templates for the solution of algebraic eigenvalue problems: a practical guide*. SIAM, 2000.

[6] Billur Barshan and Murat Cihan Yüksek. "Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units". In: *The Computer Journal* 57.11 (2014), pp. 1649–1667.

[7] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. "A singular value thresholding algorithm for matrix completion". In: *SIAM Journal on Optimization* 20.4 (2010), pp. 1956–1982.

[8] Daniela Calvetti, L Reichel, and Danny Chris Sorensen. "An implicitly restarted Lanczos method for large symmetric eigenvalue problems". In: *Electronic Transactions on Numerical Analysis* 2.1 (1994), p. 21.

[9] Emmanuel J Candès et al. "Robust principal component analysis?" In: *Journal of the ACM (JACM)* 58.3 (2011), p. 11.

[10] Erin Carson, Nicholas Knight, and James Demmel. "Avoiding communication in nonsymmetric Lanczos-based Krylov subspace methods". In: *SIAM Journal on Scientific Computing* 35.5 (2013), S42–S61.

[11] Zizhong Chen and Jack J Dongarra. "Condition numbers of Gaussian random matrices". In: *SIAM Journal on Matrix Analysis and Applications* 27.3 (2005), pp. 603–620.

[12] Sara Cohen, Benny Kimelfeld, and Georgia Koutrika. "A survey on proximity measures for social networks". In: *Search computing*. Springer, 2012, pp. 191–206.

[13] Jane Cullum and William E Donath. "A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace of large, sparse, real symmetric matrices". In: *Decision and Control including the 13th Symposium on Adaptive Processes, 1974 IEEE Conference on*. Vol. 13. IEEE. 1974, pp. 505–509.

[14] James W Demmel. *Applied numerical linear algebra*. Vol. 56. Siam, 1997.

[15] Petros Drineas, Ravi Kannan, and Michael W Mahoney. "Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix". In: *SIAM Journal on computing* 36.1 (2006), pp. 158–183.

[16] Jed A Duersch and Ming Gu. "Randomized QR with column pivoting". In: *SIAM Journal on Scientific Computing* 39.4 (2017), pp. C263–C291.

[17] Carl Eckart and Gale Young. "The approximation of one matrix by another of lower rank". In: *Psychometrika* 1.3 (1936), pp. 211–218.

[18] Alan Edelman. "Eigenvalues and condition numbers of random matrices". In: *SIAM Journal on Matrix Analysis and Applications* 9.4 (1988), pp. 543–560.

[19] *Fast Randomized SVD*. `https://research.fb.com/fast-randomized-svd/`. Accessed: 2018-05-30.

[20] Ross Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[21] Gene H Golub, Franklin T Luk, and Michael L Overton. "A block Lanczos method for computing the singular values and corresponding singular vectors of a matrix". In: *ACM Transactions on Mathematical Software (TOMS)* 7.2 (1981), pp. 149–169.

[22] Gene H Golub and Christian Reinsch. "Singular value decomposition and least squares solutions". In: *Numerische mathematik* 14.5 (1970), pp. 403–420.

[23] Gene H Golub and Charles F Van Loan. *Matrix computations*. Vol. 3. JHU Press, 2012.

[24] Gene Golub and William Kahan. "Calculating the singular values and pseudo-inverse of a matrix". In: *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis* 2.2 (1965), pp. 205–224.

[25] Ming Gu. "Subspace iteration randomization and singular value problems". In: *SIAM Journal on Scientific Computing* 37.3 (2015), A1139–A1173.

[26] Ming Gu and Stanley C Eisenstat. "Efficient algorithms for computing a strong rank-revealing QR factorization". In: *SIAM Journal on Scientific Computing* 17.4 (1996), pp. 848–869.

[27] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions". In: *SIAM review* 53.2 (2011), pp. 217–288.

[28] Peg Howland, Moongu Jeon, and Haesun Park. "Structure preserving dimension reduction for clustered text data based on the generalized singular value decomposition". In: *SIAM Journal on Matrix Analysis and Applications* 25.1 (2003), pp. 165–179.

[29] W Kahan and Beresford N Parlett. "How far should you go with the Lanczos process?" In: *Sparse matrix computations*. Elsevier, 1976, pp. 131–144.

[30] Cornelius Lanczos. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA, 1950.

[31] Ren-Cang Li. "Sharpness in rates of convergence for the symmetric Lanczos method". In: *Mathematics of Computation* 79.269 (2010), pp. 419–435.

[32] Michael W Mahoney and Petros Drineas. "CUR matrix decompositions for improved data analysis". In: *Proceedings of the National Academy of Sciences* (2009), pnas–0803205106.

[33] Cameron Musco and Christopher Musco. "Randomized block Krylov methods for stronger and faster approximate singular value decomposition". In: *Advances in Neural Information Processing Systems*. 2015, pp. 1396–1404.

[34] Chris C Paige, Beresford N Parlett, and Henk A Van der Vorst. "Approximate solutions and eigenvalue bounds from Krylov subspaces". In: *Numerical linear algebra with applications* 2.2 (1995), pp. 115–133.

[35] Christopher C Paige. "Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix". In: *IMA Journal of Applied Mathematics* 18.3 (1976), pp. 341–349.

[36] Christopher C Paige and Michael A Saunders. "LSQR: An algorithm for sparse linear equations and sparse least squares". In: *ACM Transactions on Mathematical Software (TOMS)* 8.1 (1982), pp. 43–71.

[37] Christopher Conway Paige. "The computation of eigenvalues and eigenvectors of very large sparse matrices." PhD thesis. University of London, 1971.

[38] Beresford N Parlett, Derek R Taylor, and Zhishun A Liu. "A look-ahead Lanczos algorithm for unsymmetric matrices". In: *Mathematics of computation* 44.169 (1985), pp. 105–124.

[39] Axel Ruhe. "Implementation aspects of band Lanczos algorithms for computation of eigenvalues of large sparse symmetric matrices". In: *Mathematics of Computation* 33.146 (1979), pp. 680–687.

[40] Yousef Saad. "On the rates of convergence of the Lanczos and the block-Lanczos methods". In: *SIAM Journal on Numerical Analysis* 17.5 (1980), pp. 687–706.

[41] Yousef Saad. "Theoretical error bounds and general analysis of a few Lanczos-type algorithms". In: *Proceedings of the Cornelius Lanczos International Centenary Conference (JD Brown, MT Chu, DC Ellison and RJ Plemmons, eds), SIAM, Philadelphia, PA*. 1994, pp. 123–134.

[42] Ferdinando S Samaria and Andy C Harter. "Parameterisation of a stochastic model for human face identification". In: *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*. IEEE. 1994, pp. 138–142.

[43] Horst D Simon. "Analysis of the symmetric Lanczos algorithm with reorthogonalization methods". In: *Linear algebra and its applications* 61 (1984), pp. 101–131.

[44] Ameet Talwalkar et al. "Large-scale SVD and manifold learning". In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 3129–3152.

[45] Kim-Chuan Toh and Sangwoon Yun. "An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems". In: *Pacific Journal of optimization* 6.615-640 (2010), p. 15.

[46] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*. Vol. 50. Siam, 1997.

[47] Matthew A Turk and Alex P Pentland. "Face recognition using eigenfaces". In: *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*. IEEE. 1991, pp. 586–591.

[48] Richard Ray Underwood. "An Iterative Block Lanczos Method for the Solution of Large Sparse Symmetric Eigenproblems." AAI7525622. PhD thesis. Stanford, CA, USA, 1975.

[49] Jianwei Xiao and Ming Gu. "Spectrum-revealing Cholesky factorization for kernel methods". In: *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE. 2016, pp. 1293–1298.