ARMY RESEARCH LABORATORY

Minimal Krylov Subspaces for Dimension Reduction

by Alexander M. Breuer

ARL-RP-0480 **May 2014**

A reprint from ProQuest/UMI, Indiana University, 2013; ATT 3552604

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer’s or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5068

ARL-RP-0480 **May 2014**

Minimal Krylov Subspaces for Dimension Reduction

Alexander M. Breuer
Survivability/Lethality Analysis Directorate, ARL

A reprint from ProQuest/UMI, Indiana University, 2013; ATT 3552604

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* May 2014	**2. REPORT TYPE** Reprint	**3. DATES COVERED (From - To)** 1 August 2008–31 December 2012

4. TITLE AND SUBTITLE Minimal Krylov Subspaces for Dimension Reduction	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Alexander M. Breuer	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-SLB-W Aberdeen Proving Ground, MD 21005-5068	**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-RP-0480
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

A reprint from ProQuest/UMI, Indiana University, 2013; ATT 3552604

14. ABSTRACT

Krylov subspaces may be used as an alternative to singular vector spaces or eigenvector spaces for projective dimension reduction for low-rank matrix approximation. Though the truncated spectral or singular value decomposition is optimal for minimizing Frobenius norm error of a low-rank approximation, substituting a Krylov subspace projection may result in marked compute time savings. Previous efforts to apply Krylov subspaces to low-rank approximation problems are extended to block Krylov subspaces. Closely related random projection methods are compared to block Krylov subspaces, and new hybrid approaches are developed. Hybrid random projection Krylov subspace methods offer faster compute times than random projection methods and lower approximation errors when sufficient conditions are met. A novel adaptively blocked Krylov subspace algorithm is developed that offers superior compute times to random projection methods. Stationary inner iteration is considered for accelerating convergence of Krylov subspaces and applied to the low-rank approximation problem; a generalization of eigenvalue approximation bounds is presented for Krylov subspaces augmented with inner iteration. All aforementioned methods are evaluated in terms offloating-point operations and applied to numerous problems.

15. SUBJECT TERMS

Krylov subspaces, random projections, dimension reduction, low-rank approximation, iterative methods

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Alexander M. Breuer
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UU	200	**19b. TELEPHONE NUMBER** (*Include area code*) 410-278-9157

MINIMAL KRYLOV SUBSPACES FOR DIMENSION REDUCTION

Alex Breuer

Submitted to the faculty of the University Graduate School
in partial fulfillment of the requirements
for the degree
Doctor of Philosophy
in the Division of Computer Science
of the School of Informatics and Computing,
Indiana University
January 2013

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Doctoral Committee

Andrew LUMSDAINE, Ph.D.

Gonzalo ARCE, Ph.D.

David CRANDALL, Ph.D.

Esfandiar HAGHVERDI, Ph.D.

Predrag RADIVOJAC, Ph.D.

December 13, 2012

Acknowledgments

I would like to gratefully acknowledge my research advisor, Professor Andrew Lumsdaine, for his support throughout my graduate study. Without his patience and accommodation, this would not have been possible. I would also like to express my appreciation for my entire graduate committee, Professors Gonzalo Arce, David Crandall, Esfandiar Haghverdi and Predrag Radivojac. I would also like to recognize the invaluable support of Lisa Roach, Chief of the Warfighter Survivability Branch in the U. S. Army Research Laboratory, as well as Director Dr. Paul Tanenbaum and Division Chief Robert Bowen for their encouragement and support during this process. I would also like to thank my wife, Stephanie Breuer, whose tireless faith in me has been a constant blessing.

Alex Breuer

Minimal Krylov Subspaces for Dimension Reduction

Krylov subspaces may be used as an alternative to singular vector spaces or eigenvector spaces for projective dimension reduction for low-rank matrix approximation. Though the truncated spectral or singular value decomposition is optimal for minimizing Frobenius norm error of a low-rank approximation, substituting a Krylov subspace projection may result in marked compute time savings. Previous efforts to apply Krylov subspaces to low-rank approximation problems are extended to block Krylov subspaces. Closely-related random projection methods are compared to block Krylov subspaces, and new hybrid approaches are developed. Hybrid random-projection Krylov subspace methods offer faster compute times than random projection methods, and lower approximation errors when sufficient conditions are met. A novel adaptively-blocked Krylov subspace algorithm is developed that offers superior compute times to random projection methods. Stationary inner iteration is considered for accelerating convergence of Krylov subspaces and applied to the low-rank approximation problem; a generalization of eigenvalue approximation bounds is presented for Krylov subspaces augmented with inner iteration. All aforementioned methods are evaluated in terms of floating-point operations and applied to numerous problems.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The dimension reduction problem presents itself in a wide variety of domains, from engineering to biology to economics. Though the domains and context of the reduction problems may be disparate, the presentations of the problem are similar. In all cases, one is faced with data — possibly sparse — that lies in a high-dimensional space. A powerful and popular method to reduce the dimension of the data is to project it into a low-dimensional space spanned by the leading singular vectors of the matrix $A = [a_1\ a_2\ \dots\ a_m]$ where $a_i \in \mathbb{R}^n$ are the data to be reduced. This projection minimizes total Euclidean norm error over all data and is computationally tractable even when the input matrix is large. Nevertheless, computing even a partial singular value decomposition (SVD) is expensive, which has prompted research into relaxed methods that approximate the truncated singular value decomposition.

In particular, Krylov subspace methods can iteratively produce approximations to singular vectors, but at substantially reduced cost [15, 75]. We begin with the definition of a Krylov subspace.

Definition 1. *Given a matrix $A \in \mathbb{R}^{n\times n}$ and a vector $x_0 \in \mathbb{R}^n$, the* **Krylov subspace** *of A and x_0 of dimension i is given by*

$$\mathcal{K}_i(A, x_0) = \operatorname{span}\{x_0, Ax_0, A^2x_0, \dots, A^{i-1}x_0\} \tag{1.1}$$

Krylov subspaces may also be defined when the vector x_0 is replaced by a matrix X_0 with b linearly independent columns.

Definition 2. *Given a matrix $A \in \mathbb{R}^{n\times n}$ and a matrix (or block vector) $X_0 \in \mathbb{R}^{n\times b}$ with b linearly* *independent columns, the* **block Krylov subspace** *of A and X_0 of dimension ib is given by*

$$\mathcal{K}_i(A,X_0) = \text{span}\{X_0, AX_0, A^2X_0, \ldots, A^{i-1}X_0\} \tag{1.2}$$

To distinguish between the two Krylov subspaces, we will refer to the Krylov subspaces in definition 1 as *single-vector Krylov subspaces*, and the Krylov subspaces in definition 2 as *block Krylov subspaces*. When the reference is ambiguous, it is intended to discuss general properties that are shared between the two subspace types.

The substantial reduction in compute time comes at the price of approximation error; as utilized in the literature, Krylov subspace projections may have a non-trivial difference in approximation error when compared to the truncated SVD [15, 75]. These applications all limit the number of iterations performed to produce the least-expensive Krylov subspace possible, which is also likely the least converged [10, 14, 15, 75]. We call such Krylov subspaces *minimal*.

Definition 3. *A single-vector Krylov subspace $\mathcal{K}_i(A,x_0)$ of dimension i is said to be* **minimal** *for a reduction to k dimensions when $k = i$. A block Krylov subspace $\mathcal{K}_i(A,X_0)$ with $X_0 \in \mathbb{R}^{n\times b}$ is said to be* **minimal** *for a reduction to k dimensions when $(i-1)b < k \leq ib$.*

We therefore investigate methods to accelerate the convergence of Krylov subspaces to the truncated SVD in the context of low-rank matrix approximation; in particular, we study block Krylov subspaces with and without deflation, and combination power iteration-Krylov subspace methods to produce low-rank matrix approximations that have smaller error, but are still less expensive than the truncated SVD.

Improvements to Krylov subspaces may be realized by several methods. Block Krylov subspaces [16], deflation [12] and shift-and-invert preconditioning [24] are all used to accelerate convergence for the SVD or spectral decomposition. These methods have not been applied to the generic dimension problem with a restriction on the number of iterations. The limitation on the number of iterations introduces a set of trade-offs which are distinct from the trade-offs for these acceleration methods when applied to SVD or spectral decomposition when iterations are unlimited. For example, implicit restarts are powerful methods for allowing an unlimited number of Lanczos iterations without storage of a large basis set, but offer no advantage when the Krylov subspace must be minimal. Moreover, the extra overhead introduced to execute an implicit restart — which is intended to allow for an unlimited number of stable iterations — adds overhead that may not be necessary if only one restart is ever performed.

1.1 The low-rank approximation problem

Matrix approximation methods are powerful techniques that can cope with high-dimensional data and extract a low-rank approximation that preserves features well [23,39,76]. Matrix approximation methods have a compelling set of features that render them popular choices for dimension reduction; they minimize the sum-of-squares approximation error over all points of data, they are guaranteed to converge to a solution, and they are computationally tractable. These methods grow out of the following basic matrix approximation problem: given a matrix $A \in \mathbb{R}^{n\times m}$ with rank $r \ll \min(n,m)$, find a matrix $\hat{A}^{(k)}$ with rank k such that the norm of the error matrix

$$\|E\| = \|A - \hat{A}^{(k)}\| \tag{1.3}$$

is minimized for some norm $\|\cdot\|$. We define the relevant norms for this effort.

Definition 4. *Given an $m \times n$ matrix A with singular values $\sigma_i(A)$ for $i = 1,2,\dots,r$, the* **Frobenius norm** *of A is defined as*

$$\|A\|_F = \sqrt{\textstyle\sum_{i=1}^{m}\sum_{j=1}^{n}|A_{ij}|^2} = \sqrt{\textstyle\sum_{i=1}^{r}\sigma_i(A)^2}, \tag{1.4}$$

where A_{ij} is the entry of A in column i and row j.

Definition 5. *Given a matrix A with singular values $\sigma_i(A)$ for $i = 1,2,\dots,n$, the* **nuclear norm** *of A is defined as*

$$\|A\|_* = \sum_{i=1}^{n}\sigma_i(A) \tag{1.5}$$

These norms are defined in terms of singular values of A; this implies that the singular value decomposition (SVD) will be consequential in analysis of these norms.

Theorem 1. *Let A be an arbitrary $n \times m$ matrix, and, without loss of generality, that $m \geq n$. Then we* *the* **singular value decomposition** *of A gives*

$$A = U\Sigma V^T, \tag{1.6}$$

where U is $m \times n$ and satisfies $U^T U = I$, V is $n \times n$ and satisfies $V^T V = I$, and $\Sigma = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$, where the **singular values** *$\sigma_1 \geq \sigma_2 \geq \ldots \geq 0$. The columns $u_1, \ldots, u_m$ of U are called* **left singular vectors** *of A, and the columns $v_1, \ldots, v_n$ of V are called* **right singular vectors** *of A. When $m < n$, then the singular vector decomposition may be defined in terms of A^T.*

The spectral decomposition is also a diagonal matrix factorization we will consider herein.

Theorem 2. *Let A be a square $n \times n$ Hermitian matrix. Then the* **spectral decomposition** *(alternately, the* **eigenvalue decomposition***) of A gives*

$$A = U\Lambda U^T, \tag{1.7}$$

where U satisfies $U^T U = I$, and $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and the **eigenvalues** *of A have $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. The columns $u_1, \ldots, u_n$ of U are called* **eigenvectors** *of A.*

The $\hat{A}^{(k)}$ that minimizes the Frobenius norm error $\|\cdot\|_F$ is unique and prescribed by the singular value decomposition (SVD) of A or the spectral decomposition of an associated Gram matrix AA^T or $A^T A$. Likewise, the $\hat{A}^{(k)}$ that is optimal for minimization of the nuclear norm error is given by the truncated SVD of A or spectral decomposition of $A^T A$. SVDs and spectral decompositions have polynomial complexity, may be computed iteratively, and are available in several high-performance libraries.

Nevertheless, for sufficiently large data or data with particular spectral characteristics, the singular value or spectral decomposition is costly [15]; many iterations may be required to obtain total convergence of interior eigenvalues as predicted by eigenvalue bounds [17,67]. Slow convergence is particularly problematic for eigenvalues that are not well-separated from their neighbors, but such eigenvalue clustering may be encountered in low-rank approximation problems, especially those requiring a reduction to hundreds of dimensions. Compute time and space savings may be realized

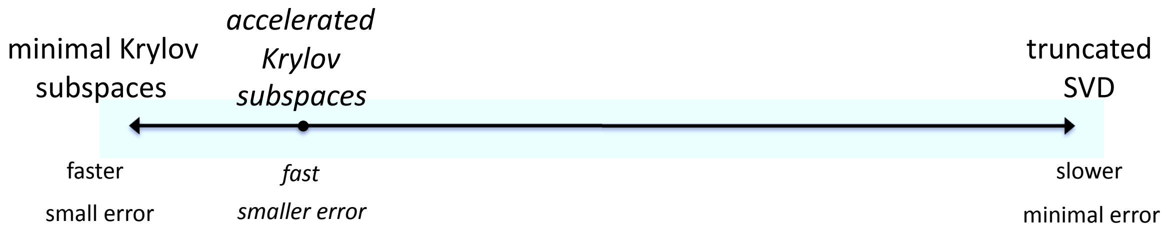

Figure 1.1: Computational costs and singular value approximation errors of subspace projections from minimal Krylov subspaces to singular vector spaces.

by relaxing the requirement for the global minimizer of (1.3) to an $\hat{A}^{(k)}$ that is close to the global minimizer. This sub-optimal $\hat{A}^{(k)}$ may be computed in less time and with less memory than the true minimizer of (1.3). Krylov subspaces may be used as an alternate to a singular vector or eigenvector space for a dimension reduction problem [10,15,63]. Krylov subspaces are widely used for iteratively solving the singular value or spectral decomposition due to their strong convergence properties and modest computational costs. A Krylov subspace will contain good approximations of the singular triplets or eigenpairs needed for minimizing (1.3) even after only a few iterations. One may simply use a Krylov subspace that approximates singular triplets or eigenpairs well to find an $\hat{A}^{(k)}$ that makes (1.3) small. The Krylov subspaces used in the current literature all use *minimal* Krylov subspaces, which require the fewest iterations and are the least expensive to produce, but also have the poorest eigenvalue approximations. Between minimal Krylov subspaces and converged singular vector or eigenvector spaces there is a spectrum of computational possibilities, with differing costs and singular value approximation properties, as illustrated in Figure 1.1. The goal of this research is to apply acceleration to Krylov subspaces that have a minimal or near-minimal number of iterations. These accelerated Krylov subspaces are intended for approximation of the SVD or spectral decomposition as applied to a dimension reduction problem. These accelerated Krylov subspace projections will have error smaller than a minimal Krylov subspace with a modest increase in computational cost. The increase in computational cost will be smaller than the cost of the fully converged SVD or spectral decomposition.

The low-rank matrix approximation attempts to minimize (1.3). As suggested previously, the SVD gives the optimal $\hat{A}^{(k)}$ for any A for the Frobenius norm. This minimization is a natural consequence of the definition of the Frobenius norm in terms of singular values of A. To show how the

SVD can be used to generate the optimal low-rank $\hat{A}^{(k)}$, let A be a $n \times m$ real matrix and $m < n$ without any loss of generality. Write $A = U\Sigma V^T$ as the SVD of A with $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_m, 0, \ldots)$. Let $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m$ be the ordering of the singular values of A. We have the definition of the Frobenius norm from (4). Because of the ordering of singular values, no partial sum of squares of k singular values is greater than $\sum_{i=1}^{k} \sigma_i^2$ and no partial sum of squares of $m-k$ singular values is less than $\sum_{i=k+1}^{m} \sigma_i^2$. Therefore, the optimal $\hat{A}^{(k)}$ is given by

$$\hat{A}^{(k)} = U_k \Sigma_k V_k^T \tag{1.8}$$

where U_k and V_k^T are the first k left and right singular vectors as defined in Theorem 1.6.

The generality of the matrix approximation problem has resulted in its widespread application to dimension reduction in various domains. The generality is a result of the way in which the problem is posed: no domain-specific information or objective function is used. Furthermore, the low-rank matrix approximation problem minimizes squared Euclidean distance over the data, which implies preservation of geometric features. More formally, if $S \subset \mathbb{R}^n$ is a set of data that is high-dimensional, then the dimension reduction problem seeks to find an approximation set $\hat{S}$ such that for any point $a_i \in S$ there is a corresponding approximation $\hat{a}_i \in \hat{S}$ with $\|a_i - \hat{a}_i\|_2^2$ small. The total squared Euclidean norm error for one column is given by

$$\epsilon_i = \|a_i - \hat{a}_i\|_2^2 = \sum_{j=1}^{n} (a_{ij} - \hat{a}_{ij})^2. \tag{1.9}$$

Note that if we assemble a matrix $A = [a_1 \ a_2 \ \ldots \ a_m]$ and a corresponding $\hat{A}^{(k)} = [\hat{a}_1 \ \hat{a}_2 \ \ldots \hat{a}_m]$ then the squared error quantity in (1.3) is

$$\begin{aligned} \|E\|_F^2 &= \|A - \hat{A}^{(k)}\|_F^2 \\ &= \textstyle\sum_{i=1}^{n} \sum_{j=1}^{m} |A_{ij} - \hat{A}_{ij}|^2 \\ &= \textstyle\sum_{i=1}^{n} \epsilon_i \end{aligned} \tag{1.10}$$

where ϵ_i is as in (1.9).

The Frobenius norm of A is also closely related to the nuclear norm of the positive semi-definite

Gram matrices A^TA and AA^T. Eigenvalues of AA^T and A^TA are squares of singular values of A. We have $A = U\Sigma V^T$. Let (v_i, u_i, σ_i) be a singular triplet of A. By construction,

$$AA^T u_i = \sigma_i A v_i = \sigma_i^2 u_i \tag{1.11}$$

by the definition of singular vectors of A. The case for A^TA is similar. Moreover, left and right singular vectors are eigenvectors of AA^T and A^TA. Since AA^T is positive semi-definite, the SVD and spectral decompositions coincide, so singular values of AA^T are also singular values of A^TA. As the nuclear norm is defined as

$$\|A\|_* = \sum_{i=1}^{m} \sigma_i \tag{1.12}$$

it is clear that

$$\|A\|_F^2 = \sum_{i=1}^{m} \sigma_i^2 = \sum_{i=1}^{m} \lambda_i = \|AA^T\|_* = \mathrm{tr}(AA^T), \tag{1.13}$$

where the λ_i are eigenvalues of AA^T. Note that the trace and the nuclear norm coincide for positive semi-definite matrices because the SVD and spectral decompositions coincide. Many existing generic dimension reduction methods use the spectral decomposition of AA^T to produce a dimension reduction [76]; however, these are equivalent to using the SVD of A.

1.2 Approximation of the SVD for low-rank approximation

We have remarked that the posing of the low-rank approximation problem renders the problem tractable, but nevertheless it can consume considerable compute time and memory storage. Though this statement may appear conflicted, its core meaning is that the complexity of the problem does not grow so quickly with problem size that it is unsolvable in practice. Even though the problem is solvable, and there exist well-tuned algorithms for solving the truncated SVD or spectral decomposition, these algorithms — which are iterative — may take many iterations to converge completely. Complete convergence may not be necessary to obtain an approximation with low error.

Numerous methods are available to use for reduced-cost approximation of eigen- or singular vector spaces for low-rank matrix approximation. Krylov subspaces are but one approach; they have been studied previously for low-rank matrix approximation problems [10, 14, 15, 75]. More recently, random projection methods have been studied and contrasted with Krylov subspace approaches for low-rank matrix approximation [36]. These two methods share deep similarities with each other. The success of random projection methods suggests Krylov subspace approaches alternate to the methods proposed previously; these alternate methods may produce better convergence or faster compute times compared to existing Krylov subspace approaches. Much of the inspiration for this work arises out of comparison of Krylov subspace and random projection methods for low-rank approximation.

Though both Krylov subspace methods and random projection approaches have already been proposed and evaluated for low-rank approximation problems, many open questions remain which merit further attention. Common acceleration methods for Krylov subspaces used for eigenvalue or SVD problems have not been employed in generic dimension reduction Krylov subspace applications. Specifically, the following key open questions remain regarding Krylov subspaces as applied to low-rank matrix approximation.

1. Block Krylov subspaces [64] have not been applied to the generic dimension reduction problem.

2. The relaxation of the requirement for a converged SVD or spectral decomposition allows for new algorithms to be developed. Such algorithms may exploit the nonnecessity of eigenvalue convergence to address common deficiencies in Krylov subspace algorithms. In particular, the difficulties caused by round-off error may be ameliorated or eliminated altogether.

3. Loss of orthogonality for the single-vector Lanczos algorithm has been characterized in [73], but an analysis for the block Lanczos algorithm, the principal block Krylov subspace algorithm for Hermitian problems, does not exist.

4. Synthesis of random projections — particularly stationary power iteration — and Krylov subspaces may lead to faster or better low-rank approximation when compute resources are constrained. These constraints may be in the form or either time restrictions of memory limitations. Though synthesis is appealing, no formal mechanisms exist to enable comparative analysis of random projections versus hybrid Krylov-random projection methods.

To address these open questions, we develop the following novel contributions.

A. We develop the novel approaches using short block Krylov subspaces, including the "shrink-and-iterate" method (section 3.2.2) and the hybrid block Krylov-random projection method (Algorithm 5), which addresses items 1 and 4.

B. We present Algorithm 4, which consists of modifications to the Lanczos algorithm [4] that substantially reduce loss of orthogonality when used for the "shrink-and-iterate" method and eliminate loss of orthogonality for the hybrid method. These modifications address item 2.

C. Theorems 7 and 8 permit predictive comparison of the "shrink-and-iterate" and hybrid short block Krylov subspace methods and random projections. These theorems are supported by Lemmas 1 and 3. Lemma 2 and Theorem 4 allow Theorems 7 and 8 to be used for comparison of approximation error between random projections and our new short block Krylov subspace algorithms. Such comparison is not possible using existing bounds. These theorems and lemmas address item 3.

D. In Theorem 9, we adapt the bounds on loss of orthogonality from [73] to the block Lanczos algorithm. This allows one to bound loss of orthogonality to support analysis of item 2.

E. We develop GrABL, presented in algorithm 10, that adaptively changes block size to reduce compute costs. GrABL is distinct from other block method via its deflation criterion, and inflation method described in Algorithm 9. GrABL is tailored for low-rank matrix applications of Krylov subspaces. To support development and analysis of the inflation and deflation criteria of GrABL, we develop Theorem 10. GrABL is a novel algorithm which is also a synthesis of random projection and Krylov subspace methods to address item 4.

F. We analyze Algorithm 11, a variant of the single-vector Lanczos method [4] with inner iteration [69], using stationary powers similar to random projection methods. Bounds to characterize the effects of inner power iteration do not exist. To illustrate the effects of inner power iteration on low-rank approximation, we present Theorem 11, which adapts eigenvalue convergence bounds from [67]. This algorithm and analysis address item 4.

G. We develop Algorithm 13, a synthesis of random projections and Krylov subspaces for inverse spectrum problems to address item 4.

The remainder of this thesis follows the preceding outline. In Chapter 2 we present background material covering both Krylov subspace approaches and random projections for low-rank approximation. We study the convergence of eigenvalues and contrast the asymptotic rates of convergence. We then develop alternate uses of Krylov subspaces for low-rank approximation problem; the use of block Krylov subspaces with large block sizes is suggested by random projection methods, but block Krylov subspaces for generic low-rank approximation have not been investigated or compared to single-vector Krylov subspaces.

In Chapters 3 and 4 present the "shrink-and-iterate" and hybrid block Krylov-random projection methods. These methods are intended to be faster than random projection methods, or to produce better eigenvalue and eigenvector approximations than random projection methods. We have observed that random projections may produce lower approximation error than the single-vector Krylov subspace methods in the literature; therefore, the "shrink-and-iterate" and hybrid methods will also produce better approximations than the single-vector Krylov subspace approaches studied previously. Additionally, these short block Krylov subspace methods have dramatically reduced loss of orthogonality compared to traditional Krylov subspace methods. Loss of orthogonality is a significant problem in all existing Krylov subspace methods; elimination of orthogonality loss also eliminates the need for reorthogonalization methods [57, 73] that add expense and complicatedness to Krylov subspace algorithms.

We then present in Chapters 5 and 6 block methods that use deflation to best leverage the advantages presented by large block sizes while simultaneously managing the additional computational costs imposed. Adaptive deflation of block size allows for asymptotically superior compute complexity compared to random projections. Adaptively-blocked Krylov subspace methods exist [5, 86], but are designed for indefinite iteration. When applied to the low-rank matrix approximation problem with a limited number of iterations, the current adaptive block methods produce larger approximation errors than random projections. Development of new inflation subspaces and deflation criteria produces a new algorithm, GrABL, that produces comparable performance to random projections, while maintaining the compute cost advantages over random projections.

Random projection methods also suggest the use of power iteration as an acceleration method; we study the theoretical and practical effects of power iteration acceleration in Chapter 7. Power iteration acceleration is far less expensive than shift-and-invert preconditioning, and is well-suited to shifting convergence to the leading extreme of the spectrum and away from the trailing extreme. Inner power iteration is especially suited for generating a Krylov subspace of minimal length when matrix-vector products are inexpensive but memory storage is constrained. No expressions exist to characterize the change in eigenvalue convergence due to inner power iteration. By providing asymptotic, worst-case error expressions, we allow for quantitative comparison of the convergence of eigenvalues in Krylov subspaces formed with inner power iteration to the convergence of eigenvalues in ordinary Krylov subspaces. Such comparisons allow for informed choices that may weigh the convergence advantage of inner power iteration against the extra compute costs inner power iteration imposes.

Shift-and-invert preconditioning may be applied to accelerate the convergence of minimal Krylov subspaces whenever the input matrix admits an efficient sparse factorization. We develop new strategies for applying and combining shifts specifically for low-rank approximation problems, and compare them to exiting shift-and-invert methodologies in Chapter 8. Faster convergence may be obtained by better shift choices. Finally, we compare all methods in an application to model reduction.

Chapter 2

Low-rank approximation background

Krylov subspaces have properties favorable for low-rank approximation problems that are solvable with a truncated SVD or spectral decomposition. Eigenvalues or singular values usually converge rapidly in Krylov subspaces, and the spectral components needed for a truncated singular vectorspace or eigenspace projection are among those that bounds show will converge fastest in a Krylov subspace. This fast convergence implies that a Krylov subspace will contain good approximations to eigenvalues in only a few iterations. Applying a modest limit to the number of iterations should not deteriorate the quality of leading eigenvalue and eigenvectors. One may simply perform k iterations of a Krylov subspace algorithm to obtain the least expensive k-dimensional Krylov subspace approximation possible, rather than compute all k leading eigenvalues or singular values to convergence. This is precisely what has been proposed to date for Krylov subspace dimension reduction and low-rank approximation; the least-expensive Krylov subspace is used.

Apart from their good approximation properties, Krylov subspace methods lend themselves to efficient computation, especially when the input matrix is Hermitian. In these cases, there are Krylov subspace algorithms that produce an orthonormal basis for a Krylov subspace without the need for a Gram-Schmidt processes or Householder rotations [4]. Additionally, Krylov subspace methods can produce a projection of the input matrix in the Krylov subspace as iteration proceeds, rather than first generating a basis and then projecting the input matrix A onto that basis. Krylov subspace methods do have drawbacks; principally that they are prone to corruption by round-off error, and that both leading and trailing eigenvalues converge fastest in a Krylov subspace [17, 67].

The former problem is well-known and has been addressed by methods to stabilize against the effects of round-off error [57,73]. The latter difficulty is less of a problem in traditional applications of Krylov subspaces; indeed, this property is an asset, as it allows for computation of either end of the spectrum with the same algorithm. Yet for low-rank approximation problems in which the number of iterations available is limited, convergence of trailing spectral components can be problematic.

Random projection methods are closely related to Krylov subspace methods; both use matrix-vector products to generate a low-rank subspace in which the important characteristics — approximation of leading eigenpairs or singular triplets, in most cases — are well-preserved. There are important distinctions between Krylov subspace methods and random projection methods, both in terms of computational requirements and convergence of eigenvalues or singular values. Random projection methods admit analysis as Krylov subspace methods, and these asymptotic analyses provide additional insight into the spectral properties of input matrices that drive convergence of the two respective methods.

2.1 Generic dimension reduction background

The generic dimension reduction problem is by no means new, and a great deal of the existing measures reduce to the low-rank matrix approximation problem. An overwhelming number of these generic techniques are direct descendants or reducible to Principal Component Analysis (PCA) [39]. The Krylov methods under consideration here all approximate the PCA or related dimension reduction method. We briefly review the methods dimension reduction methods we aim to approximate with Krylov subspace approximations.

PCA is perhaps the oldest generic dimension reduction method. Originally due to Pearson [59], PCA in its original form is a geometric dimension reduction method, and attempts to solve a minimization problem closely related to (1.3). Instead of the matrix $A = [a_1 \; a_2 \; \ldots a_m]$, PCA uses the centered matrix $A - \mu = [(a_1 - \mu) \; (a_2 - \mu) \; \ldots (a_m - \mu)]$ with $\mu = 1/m \sum_{i=1}^{m} a_i$. The spectral decomposition on the Gram matrix $(A - \mu)(A - \mu)^T = U \Lambda U^T$ provides the projection

$$\hat{A}^{(k)} = U_k^T (A - \mu) = \Sigma_k V_k^T \tag{2.1}$$

with the SVD $(A-\mu) = U\Sigma V^T$. This is equivalent to the truncated SVD approximation derived from $A-\mu$. Statistically, the centered Gram matrix $(A-\mu)(A-\mu)^T$ is identical to the covariance matrix of the underlying distribution of the data points a_i, up to scaling.

Numerous dimension reduction methods are a result of application of PCA or a PCA-like method; the proper orthogonal decomposition (POD) [13, 48, 62] from signals analysis and model reduction, eigenfaces [79, 80] from face recognition and latent semantic indexing (LSI) [19, 21] from information retrieval all use eigenvector or singular vector projections. The chief differences between these methods are due to the preprocessing and formation of the matrix A, such as centering used in PCA or term scaling used in LSI. As these methods all use the SVD or the spectral projection, they all share the same computational advantages and drawbacks: the optimal low-rank approximation is tractable and high-performance libraries exist, but large problems may require large compute times.

Spectral graph methods also use the spectral decomposition of a positive semi-definite matrix to discover properties of a graph. Various graph properties may be determined with spectral methods, from vertex clustering [53, 54, 61, 87] to visualization [37, 43, 70] to partitioning [60]. These methods operate on the graph Laplacian matrix; either the combinatorial Laplacian or normalized Laplacian may be used. Both Laplacian matrices are defined in terms of the adjacency matrix of the graph A, and the diagonal degree matrix of the graph, D. Both methods also require the eigenpairs (λ_i, u_i) with the smallest λ_i rather than the pairs with largest magnitudes as are used in PCA and PCA-like methods.

2.2 Krylov subspace background

We recall the definition of a Krylov subspace from Definition 1:

$$\mathcal{K}_i(A, x_0) = \operatorname{span}\{x_0, Ax_0, A^2x_0, \dots, A^{i-1}x_0\} \tag{2.2}$$

Krylov subspaces are naturally amenable to iteration; one adds basis vectors for $\operatorname{span}\{A^i x_0\}$ one after the other. These basis vectors may be orthogonalized against one another to produce an orthonormal projection. We note that the linear independence of the vectors $A^i x_0$ is only guaranteed in infinitely-precise arithmetic. As realized on a computer, the vectors $A^i x_0$ lose linear independence rapidly, and are not practical for use in generating a basis. Instead, basis vectors are orthogonalized with respect to each other as they are generated to reduce the effects of round-off error [4, 44]. Each basis vector improves the quality of the solution [31, 67, 81, 85]; iteration may be terminated once the solution is of acceptable quality. Krylov subspaces are also defined in block form in which the start vector x_0 is replaced with a matrix $X_0 \in \mathbb{R}^{n \times b}$ with all columns linearly independent.

Krylov subspace methods are widespread choices for iteratively solving a truncated SVD or spectral decomposition. Extremal singular triplets or eigenpairs converge fastest in Krylov subspaces; these are a subset of the leading singular triplets or eigenpairs needed to solve the dimension reduction problem. The rapid convergence of extremal singular triplets or eigenpairs in Krylov subspaces suggests that one may simply terminate iteration before complete convergence of the truncated decomposition. This would produce an approximation that is close to the optimal solution of (1.3) but with reduced costs. The greatest opportunity for reduction of computational costs are given by minimal Krylov subspaces, which perform only as many iterations as are needed to induce a reduction to k dimensions.

Krylov subspaces for low-rank matrix approximation and related dimension reduction problems has been studied previously. Simon et. al. [75] first proposed Krylov subspaces as an alternative to singular or spectral projections for low-rank matrix approximation. Blom and Ruhe [9, 10] applied Krylov subspaces to information retrieval tasks related to low-rank approximation, and showed how to leverage *a priori* information via the start vector x_0. Alternate relaxations to the SVD for information retrieval are proposed in [7, 8]. Chen and Saad [15] studied dimension reduction tasks including information retrieval and eigenfaces [80]. Chen, Fang and Saad [14] applied Krylov subspace approximation to the eigenvector approximation problem for graph partitioning. In addition to these applications, Elden has shown [22] that Krylov subspace projections are equivalent to partial least squares [83], a dimension reduction method widely used in the chemometrics community. All of these applications realized a maximal compute time improvement with minimal Krylov subspaces. More recently, Halko et. al. [36] have investigated random projection methods, which bear deep connections with minimal block Krylov subspaces.

The good results reported in the literature for Krylov subspaces for low-rank approximation are due to the rapid convergence of spectral components used for the optimal projection in Krylov subspaces. Fortunately, many low-rank approximation problems have just the spectral characteristics that drive good convergence in Krylov subspaces. Nevertheless, minimal Krylov subspaces of dimension k will often contain eigenvalue approximations $\lambda_i^{(k)}$ (alternately called Ritz values and denoted θ_i) where $\lambda_i - \theta_i$ is non-trivial. When this is the case, there may be a non-trivial gap in error between the SVD-derived rank-k approximation, which we write as $\hat{A}^{(k)}_{\mathrm{PCA}}$ and the Krylov subspace rank-k approximation, which we write as $\hat{A}^{(k)}_{\mathscr{K}}$. Acceleration of convergence in the Krylov subspace may be realized by application of block methods with deflation, and shift-and-invert preconditioning [4]. To connect the acceleration methods proposed, we will review results on the convergence of Ritz values to eigenvalues in Krylov subspaces.

2.3 Convergence of eigenvalues in Krylov subspaces

When spectral properties are favorable, eigenvalues will converge rapidly in Krylov subspaces. There are two results that provide lower error bounds on eigenvalue estimates in Krylov subspaces, one due to Saad [67] and another due to Underwood [31]. Examination of these bounds provides insight into what spectral properties drive convergence of Ritz values in Krylov subspaces, and transformations that will accelerate convergence. The convergence properties of eigenvalues in Krylov subspaces will also illuminate the difficulties posed by the convergence of both leading and trailing eigenvalues in a Krylov subspace.

The nature of eigenvalue convergence in Krylov subspaces was finally resolved in the 1960s and 1970s, beginning with Kaniel [40], proceeding with Paige [55], Underwood [81] and culminating with Saad [67]. These asymptotic bounds show that Krylov subspace projections indeed do produce good eigenvalue approximations with only a modest number of iterations. We present the bounds due to Underwood, that generalize the Paige results to block Krylov subspaces.

Theorem 3 (Underwood [81])**.** *Consider a Krylov subspace $\mathscr{K}_i(A, X_0)$ with block size r of a Hermitian matrix A with eigenvalues ordered as $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{\mathrm{inf}}$. Let λ_j be an eigenvector of a matrix A with associated eigenvector u_j with $\|u_j\| = 1$. Then for $j = 1, 2, \ldots, r$ the eigenvalue approximation* *(alternately, Ritz value)* $\lambda_j^{(i)}$ *to* λ_j *is bounded as*

$$0 \leq \lambda_j - \lambda_j^{(i)} \leq (\lambda_j - \lambda_j^{(i)}) \frac{\tan^2 \Theta}{T_{i-1}(\hat{\gamma}_j)^2} \tag{2.3}$$

where

$$\hat{\gamma}_j = 1 + 2\frac{\lambda_j - \lambda_{r+1}}{\lambda_{r+1} - \lambda_{\text{inf}}}, \tag{2.4}$$

$T_i(\cdot)$ *is the Chebyshev polynomial of the first kind with order* i*, and* $\Theta = \cos^{-1}\tau$ *where* τ *is the smallest singular value of the matrix* $U^T X_0$ *for* $U = [u_1\ u_2\ \dots\ u_r]$.

Errors are driven to zero by the growth of the denominator in (2.3). The definition of $\hat{\gamma}_j$ shows the influence of local gaps in the spectrum of the input matrix A. Well-separated eigenvalues will result in large $\hat{\gamma}_j$, which causes the denominator of (2.3) to be large as well. The denominator may be grown through increased Krylov subspace dimension through the squared Chebyshev polynomial $T_j(\cdot)^2$, which grows rapidly with the increased order resulting from successive iterations. Errors are also driven to zero by the block size increasing $\hat{\gamma}_j$. Note that these bounds are for magnitude only; it may be the case for a Ritz pair (θ_i, v_i) and a corresponding eigenpair (λ_i, u_i) that $\theta_i - \lambda_i$ is small but $\|v_i - u_i\|$ is not small. However, for the low-rank approximation problem, this is less of a problem, as the error in (1.3) is due only to eigenvalues.

These bounds also apply to the trailing component of the spectrum of A; a Krylov subspace will contain approximations to both leading and trailing eigenvalues. This is problematic for minimal Krylov subspace projections if trailing eigenpairs converge at least as quickly as the leading eigenpairs. In general, the quality of a Krylov subspace approximation may be improved by accelerating the convergence of leading eigenvalues. Though one may also generate a Krylov subspace of dimension $2k$ and discard the trailing k Ritz pairs, such an approach may consume a prohibitive amount of memory when the dimension of A is large. Therefore, we restrict ourselves to minimal or near-minimal Krylov subspace methods.

One may note that if the local gaps $\lambda_j - \lambda_{r+1}$ result in rapidly-growing $\hat{\gamma}_j$ for small increases in r, then one may apply a block Krylov subspace method. Increasing block size will accelerate convergence of the eigenvalues. Alternately, applying a transformation function f to the spectrum of A will accelerate convergence of eigenvalues when it makes $\hat{\gamma}_j$ large for leading eigenvalues but small for trailing ones. This is the principle behind shift-and-invert preconditioning: the spectrum of A is preconditioned with $f(\lambda_j) = (\lambda_j - s)^{-1}$ for some shift s. Eigenvalues close to s will have large magnitude and be well-separated from the remainder of the spectrum.

2.4 Low-rank approximation error and eigenvalue approximations

For low-rank matrix approximation in Krylov subspaces, we measure squared Frobenius norm error when the matrix is not positive semi-definite; for positive semi-definite matrices, the matrix trace is a useful error measurement norm. Ideally, we would like to use the squared Frobenius norm of the low-rank approximation to reason about the low-rank approximation error. Indeed, there is a close relationship between the squared Frobenius norm of the approximation error and the squared Frobenius norm of the low rank approximation. We present the following original theorem to show the relationship between low-rank approximation error and the norm of the low-rank approximation $\hat{A}^{(k)}$ formed by the projection of the input matrix A into a subspace.

Theorem 4. *Let A be an arbitrary real rectangular matrix and Q be an orthonormal projection matrix and P be an orthonormal matrix with* $\text{span}\{P\} = \text{span}\{AQ\}$. *Let $M = P^TAQ$ and $\hat{A}^{(k)} = PMQ^T$. Then*

$$\|A - \hat{A}^{(k)}\|_F^2 = \text{tr}(AA^T) - \text{tr}(MM^T). \tag{2.5}$$

Proof. Let A, Q, P, M and $\hat{A}^{(k)}$ be as given. By definition of the Frobenius norm, $\|A\|_F^2 = \text{tr}(AA^T)$. Then

$$\|A - \hat{A}^{(k)}\|_F^2 = \text{tr}((A - \hat{A}^{(k)})(A - \hat{A}^{(k)})^T) = \text{tr}(AA^T) - \text{tr}(2A\hat{A}^{(k)T}) + \text{tr}(\hat{A}^{(k)}\hat{A}^{(k)T}) \tag{2.6}$$

since $\text{tr}(A\hat{A}^{(k)T}) = \text{tr}(\hat{A}^{(k)}A^T)$. Note that $\text{tr}(\hat{A}^{(k)}\hat{A}^{(k)T}) = \text{tr}(PMM^TP^T) = \text{tr}(MM^T)$. Also, as $AQ = PM$,

$$A\hat{A}^{(k)T} = A(QM^TP^T) = PMM^TP^T. \tag{2.7}$$

Then

$$\mathrm{tr}(A\hat{A}^{(k)T}) = \mathrm{tr}(MM^T) \tag{2.8}$$

and (2.6) simplifies to

$$\|A - \hat{A}^{(k)}\|_F^2 = \mathrm{tr}(AA^T) - \mathrm{tr}(MM^T) \tag{2.9}$$

which completes the proof of the theorem. □

This result allows us to use the convergence of eigenvalues to infer the low-rank approximation error.

2.5 Krylov subspace algorithms

There are a multitude of Krylov subspace algorithms for the eigenproblem, and a detailed review of them is beyond the scope of this section. Bai has complied many of them in [4]. In this thesis, we are concerned mainly with the original Hermitian Lanczos algorithm [44], the block Hermitian Lanczos algorithm [16], and the single-vector [30] and block version [31] of the Golub-Kahan algorithm for rectangular matrices.

The focus on the Lanczos algorithm is due to its efficiency. Note that one may generate a basis for $\mathbb{R}^n$ if A has full column rank. We noted that Krylov subspace projections have the property that extremal eigenvalues converge rapidly for Hermitian A with increasing iterations. If we are to generate an orthonormal basis for the Krylov subspace, a naïve approach would simply apply the Gram-Schmidt process to the series $\langle A^i x_0 \rangle_{i=0}^{k-1}$. This method is prone to round-off error; moreover, it has complexity in $O(nk^2)$. The Lanczos algorithm partially addresses round-off error by orthogonalizing as it goes, and leverages the Hermitian-ness of the matrix to eliminate the need for the entire Gram-Schmidt process. Instead of requiring the new vector at step i be orthogonalized against all $i-1$ previous vectors, the Lanczos process only requires the previous 3 vectors to generate an orthonormal basis. This orthogonalization does prevent the Krylov subspace basis vectors from losing linear independence longer than would be for Krylov vectors $A^i x_0$, but round-off error does eventually cause loss of orthogonality between the subspace basis vectors if no extra stabilization is used. More importantly, all iterations have equal cost, and extended iteration is bound only by storage of the basis vectors, rather than by the costs of the orthonormalization process. The basic Lanczos algorithm is presented in Algorithm 1. The Lanczos algorithm may be generalized to accept a block input vector X_0 with r columns. The block Lanczos algorithm is presented in Algorithm 2.

Algorithm 1 Classic Lanczos algorithm

```
Require: a priori chosen start vector $q_1$
$r \leftarrow Aq_1$
for $j = 1 \rightarrow k$ do
    $\alpha_j \leftarrow q_j^T r$
    $q_{j+1} \leftarrow r - q_j \alpha_j$
    $\beta_{j+1} \leftarrow \|q_{j+1}\|$
    $q_{j+1} \leftarrow q_{j+1} / \beta_{j+1}$
    $r \leftarrow Aq_{j+1}$
    $q_{j+1} \leftarrow r - q_j \beta_{j+1}$
end for
return $[q_1\ q_2\ \ldots], \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \alpha_3 & \beta_4 & \\ & & \ddots & \ddots & \ddots \end{bmatrix}$
```

Though both Lanczos algorithms address the round-off error problems of the naïve approach somewhat, they still cannot guarantee orthogonality of all basis vectors in finite-precision arithmetic. Eigenvectors that have converged will re-enter the Krylov subspace and lead to spurious eigenvalue approximations. Some degree of reorthogonalization is required to recover from round-off error, but applying a Gram-Schmidt step at each iteration would wreck the improvement in complexity gained from the Hermitian-ness of A. Selective [57] and partial reorthogonalization [74] both address the instability of the Lanczos algorithm due to round-off error, and have computational costs that are almost certainly less than simply performing a Gram-Schmidt step at each iteration.

The random projection methods in [36] also merit attention at this point. Though they are not true Krylov subspace methods, they may be considered a variant of the acceleration methods we

Algorithm 2 Classic block Lanczos algorithm

```
Require: a priori chosen start block $Q_1$
R ← AQ_1
for $j = 1 \rightarrow k$ do
    $A_j \leftarrow Q_j^T R$
    $R \leftarrow R - Q_j A_j$
    QR factorize $R = Q_{j+1} B_{j+1}$
    $R \leftarrow A Q_{j+1}$
    $R \leftarrow R - Q_j B_{j+1}^T$
end for
return $[Q_1\ Q_2\ \ldots], \begin{bmatrix} A_1 & B_2^T & & & \\ B_2 & A_2 & B_3^T & & \\ & B_3 & A_3 & B_4^T & \\ & & \ddots & \ddots & \ddots \end{bmatrix}$
```

study here. They are near minimal, and never store more than k basis vectors to produce a k-dimensional approximation. These methods use span$\{A^p X_0\}$ with $X_0 \in \mathbb{R}^{n \times k}$ to produce their approximation. These methods are variants of power iteration [4, 32], but with a block instead of a vector. Convergence of the lead eigenvalues is accelerated with block sizes, and the trailing Ritz vector estimates are excluded with a restart. We have observed that this method may produce lower-error approximations than single-vector Lanczos, and serves as a suitable starting point for investigation of accelerated block methods.

2.6 Random projection methods

Random projection methods [36] for approximating the truncated SVD are an alternative to Krylov subspace methods, though they have relevant similarities to Krylov subspace methods. As their name implies, random projection methods use an orthogonal projection to transform a large, likely sparse problem into a small, dense problem that approximates the original. Like Krylov subspace methods, random projection methods form an orthogonal projection basis using matrix products of the input matrix. But unlike Krylov subspaces, random projections do not use a *series* of matrix products; instead, they only use one matrix power A^p for $p \in \mathbb{N}$. The random projection subspace is then

$$S = \operatorname{span}\{A^p\Omega\} \tag{2.10}$$

for a Hermitian input matrix A and a random block vector Ω. The basic algorithm for the random projection method for approximating the leading k eigenvalues of a Hermitian input matrix is given in Algorithm 3. Rather than generating subspace dimensions through matrix powers, random projection methods use a block vector with linearly independent columns. The relationship with Krylov subspaces is through their mutual relationship with the power method. Additionally, random projection methods may be viewed as block Krylov subspace methods, with minimal iteration and maximal block sizes. The random projection space $\operatorname{span}\{A\Omega\}$ will approximate $\mathcal{K}_2(A,\Omega)$, as only the space $\operatorname{span}\{\Omega\} - \operatorname{span}\{A\Omega\}$ differs between them.

Algorithm 3 Direct eigenvalue approximation for Hermitian matrices with random projections

```
Require: random matrix $\Omega$ for multiplication against $A$, optional power $p \in \mathbb{N}$.
QR factorize $QR = A^p\Omega$
$\hat{A} \leftarrow Q^T A Q$
eigendecompose $\hat{A} = U\Theta U^T$
return $\Theta, QU$
```

Random projection methods also have strong properties for low-rank approximation of matrices; in [36] the error of a low-rank approximation of $\hat{A}^{(k)} = Q^T A Q$ where $\operatorname{span}\{Q\} = \operatorname{span}\{A^p\Omega\}$ is bounded as

$$\|A - \hat{A}^{(k)}\| \leq \|\Sigma_2\| + \|\Omega_2\Omega_1^+\Sigma_2\|^{1/p} \tag{2.11}$$

where

$$A = [U_1 U_2] \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} [V_1 V_2]^T \tag{2.12}$$

is the SVD of A with diagonal entries of Σ in descending order, and $\Omega_1 = U_1^T\Omega$ and $\Omega_2 = U_2^T\Omega$. Note that these bounds are in terms of the aggregate error over all eigenvalues of A rather than individual eigenvalues. The motivation for the bounds of this type as opposed to the individual eigenvalue bounds in (2.3) is that they enable convenient specification to the case when Ω is a random matrix and the singular values of Ω_1 and Ω_2 assume expected values. Additionally, bounds of this type clearly express how well a random projection space can be used for approximating a truncated SVD or spectral decomposition.

A further merit of random projection methods is that they do not suffer the problems with loss of orthogonality that afflict Krylov subspace methods. Even when the matrix is badly conditioned, the entire projection basis is orthogonalized at step 1 before it is used to project the input matrix A. Therefore, loss of orthogonality may only result from loss of linear independence of columns of $A^p X_0$. We have not observed the loss of orthogonality between basis vectors generated by the random projection method to be greater than machine error ϵ in any case.

2.7 Discussion

Krylov subspace methods and random projections share some important similarities; much of the this thesis explores the similarities between them and the opportunities for hybridization. These hybrid approaches attempt to combine the best attributes of random projection methods and Krylov subspace methods for fast low-rank approximation. The random projection methods suggest use of Krylov subspaces with large blocks may attain better convergence and greater stability than iteration with a single-vector. However, the computational complexity of random projections and block Lanczos iteration shows that costs are asymptotically quadratic with the block size, so smaller block sizes will be faster and scale better than large block sizes. We first study the use of large block sizes with a small number of Lanczos iterations and their effects on convergence relative to the random projection method. We then study how computational costs can be minimized by performing block iteration only when necessary to improve convergence, and reducing the block size adaptively when a wide block is no longer advantageous. The power parameter p on line 1 of Algorithm 3 used to improve convergence in the random projection method motivates study of the use of power refinement to produce better start blocks to reduce $\tan^2 \Theta$ in (2.3) and to act as an acceleration method to improve convergence of the leading eigenvalues or singular values in the Krylov subspace. Alternate preconditioning methods may be drawn from existing techniques used for conditioning Krylov subspaces.

The overarching assumption is that either Krylov subspace projections or random projections will be used for approximation of a truncated SVD or spectral decomposition with an emphasis on computational savings over a converged truncated SVD. The number of power refinements available to the random projection method will be limited, as is the number of iterations available to Krylov subspace methods. This constraint is key to the hybrid approaches developed herein; if power refinements are not limited, or if Krylov iterations are not limited, many of the low-rank approximations considered in this effort will eventually converge to an exact solution. We are interested in solving an approximation of the truncated SVD or spectral decomposition as rapidly as possible, and assume constrained compute time and storage space.

Chapter 3

Short block Krylov subspaces for low-rank approximation

The low-rank matrix approximation problem presents itself in many domains. To find the rank-k approximation that minimizes the Frobenius norm, one may simply apply the SVD to the input matrix A and truncate it to form $\hat{A}^{(k)}$. This truncated SVD solution minimizes the Frobenius norm of $E = A - \hat{A}^{(k)}$. As discussed previously, the SVD is tractable, but expensive. Relaxation of the low-rank problem to allow a solution $\hat{A}^{(k)}$ that may not be optimal but still makes $\|E\|_F$ small may lead to non-negligible computational time savings. Minimal single-vector Krylov subspaces have been proposed for this problem [15, 75], but, as was discussed in Chapter 2, minimal single-vector Krylov subspace projections may have a non-trivial gap in norm between the PCA approximation and the minimal Krylov subspace approximation. Random projection methods [36] are alternate choices for low-rank matrix approximation, and may produce smaller approximation errors than minimal single-vector Krylov subspaces. Random projection methods also enjoy immunity to the loss of orthogonality that is problematic for Lanczos-type methods for generating Krylov subspaces. Nevertheless, random projection methods have computational drawbacks; compute costs excluding sparse matrix-vector costs scale quadratically with the dimension of the subspace produced. These non-sparse operations dominate compute costs when the input matrix is sufficiently structured and sparse. Substitution of a block Krylov subspace with 50% smaller block will lead to compute time improvements; these compute time improvements allow for one to either perform fewer Krylov subspace iterations and be faster than random projections, or perform more iterations in an attempt to produce smaller errors in equivalent compute times. It is possible to produce expressions that predict cases for which a short block Krylov subspace will construct a low-rank approximation with smaller error than random projections. A 50% reduction in block size will also limit loss of orthogonality when the number of Lanczos iterations is limited to between 2 to 4.

An advantage that random projection matrix approximation methods enjoy is robustness to loss of orthogonality issues that afflict the Lanczos-type Krylov subspace algorithms. Indeed, loss of orthogonality due to round-off error has been a constant source of difficulty in Lanczos-type algorithms [32], and prevented the adoption of the Lanczos method [44] until it was shown that loss of orthogonality and eigenvalue convergence were linked [55]. Even after that discovery, methods to correct loss of orthogonality [16, 57, 74] are still necessary to produce accurate eigenvalue calculations. Indeed, incorporation of an explicit reorthgonalization step into the Lanczos process “destroys the simplicity of the Lanczos procedure” [16]. All Lanczos procedures, block or single-vector, will lose orthgonality provided a sufficient number of iterations. However, loss of orthogonality is tied to both eigenvalue convergence and the number of iterations; this was shown for the single-vector case in [73]. We extend this result to the block Lanczos algorithm in Theorem 9; this implies that a limited number of iterations will likewise limit loss of orthogonality.

In contrast to Lanczos methods to generate Krylov subspaces, the projection methods introduced in Algorithm 3 are wholly robust to loss of orthogonality up to machine precision. This robustness is a result of the lack of true iteration in the random projection algorithms. Orthogonality is maintained completely throughout the entire random projection process, provided that round-off error does not lead to loss of linear independence of $A^p\Omega$. This robustness does come at a price; the computational complexity of the random projection method is the same as single-vector Lanczos with a full Gram-Schmidt reorthogonalization step at each iteration. In most cases, the dense linear algebra operations required by the random projection method are more expensive than the (likely sparse) matrix-vector products and dense linear algebra operations in the ordinary Lanczos algorithm. When sparse matrix-vector products used in the random projection method are more expensive than the QR factorizations or matrix-matrix projections, a different set of compute time advantages emerges for the block Lanczos algorithm over random projections. In these cases, the extra work of forming an orthonormal subspace and then re-projecting the input matrix into that subspace puts the random projection method at a disadvantage compared to classical Lanczos iteration.

The computational complexity of random projection methods are a result of their choice of solution subspace: span$\{A^pX_0\}$ for $X_0 \in \mathbb{R}^{n\times k}$ and $p \in \mathbb{N}$. We contrast this with the block Krylov subspace span$\{X_0, AX_0, A^2X_0, A^{i-1}X_0\}$. To generate an orthonormal matrix for projection into span$\{A^pX_0\}$, a Gram-Schmidt or equivalent process must be used to orthogonalize and normalize the columns of A^pX_0. These reorthogonalization methods all have complexity $O(nk^2)$. Block Krylov subspace algorithms, such as the block Lanczos algorithm [16] have equivalent complexity for block size k. Shrinking the block size with s as $b = k/s$ will render each block Lanczos iteration s^2 times as fast as the random projection method, which enables one to either generate a minimal block Krylov subspace and be faster than the random projection method or perform more than s iterations and attempt to produce better eigenvalue approximations than the random projection method. Moreover, if s is small enough, then such a block Lanczos method will not require any reorthogonalization.

To obtain superior compute costs via block size shrinkage, we propose new approaches: the “shrink-and-iterate” approach and the hybrid block Krylov-random projection approach. These methods apply a block size shrinkage factor of s, and perform some number of Krylov subspace iterations to obtain a subspace of dimension k. For practicality, s must be small; we consider only the case of $s = 2$; larger s leads to more loss of orthogonality and more storage of Lanczos vectors. The “shrink-and-iterate” approach uses $s = 2$ and performs 3 or 4 block Lanczos iterations. The hybrid random projection and Krylov subspace method also sets $s = 2$ and uses $\mathcal{K}_2(A, A^2X_0)$ to approximate $\mathcal{K}_4(A, X_0)$ for even more compute savings and more robustness to round-off error. In fact, we show that the block Lanczos algorithm can be modified so that use of $\mathcal{K}_2(A, A^2X_0)$ is as stable as the random projection method, and produce a new modified block Lanczos method that incorporates a refinement step similar to refined Gram-Schmidt [4] that renders the hybrid method as stable as random projections in practice.

We wish to produce expressions that can predict if the random projection method or the shrink-and-iterate approach will produce a low-rank approximation of the input matrix with smaller error.

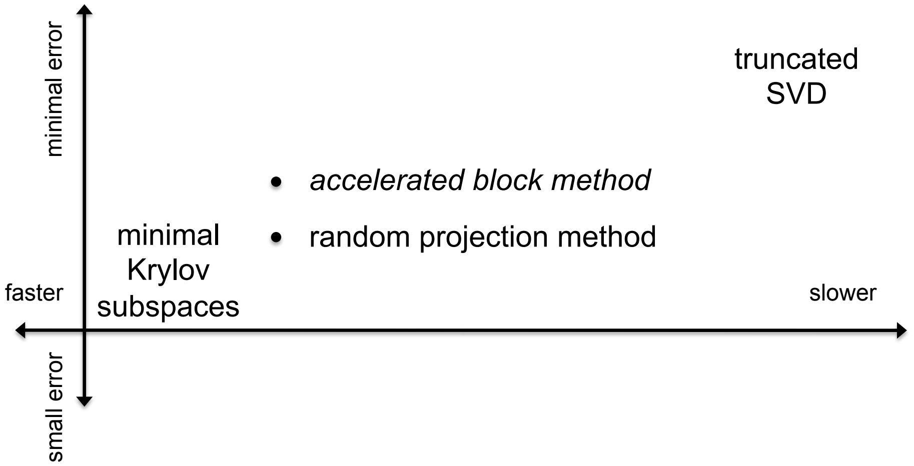

Figure 3.1: Computational costs and singular value approximation errors of truncated SVD versus random projections and proposed block Krylov accelerations. The use of block Krylov subspaces with smaller block sizes may yield smaller error with the same computational effort.

These expressions would also lend insight into what spectral properties will determine if random projections or the shrink-and-iterate approach will produce better approximations. The existing expressions lack the tightness required to compare the two methods in question; we produce a new expression for the norm of the low-rank approximation from the random projection method and the norm of the low-rank approximation from $\mathscr{K}_2(A, A^2X_0)$. The latter expression can be used to predict if either the shrink-and-iterate approach using $\mathscr{K}_4(A, X_0)$ or the hybrid method using $\mathscr{K}_2(A, A^2X_0)$ will produce a better low-rank approximation than the random projection method. Figure 3.1 shows the existing random projection method and the proposed shrink-and-iterate approach in the spectrum of generic dimension reduction methods. When and how eigenvalue approximations from a block Krylov subspace will be better than those from a random projection method depend on the spectrum of the input matrix, the number of dimensions in the subspace and the amount of storage available for iterations beyond s. We will show that cases for which leading eigenvalues converge quickly for the random projection method are also the cases for which the shrink-and-iterate approach or the hybrid approach will produce a better low-rank approximation.

We begin by reviewing the random projection method for low-rank matrix approximation and compare its projections to block Krylov subspaces generated by the block Lanczos algorithm. We note the loss of orthogonality in the block Lanczos method, and propose a block Lanczos variation with refinement akin to refinement in the Gram-Schmidt orthonormalization algorithm. We then investigate convergence of eigenvalues in random projection and block Krylov subspaces and develop new bounds for both algorithms to enable comparison of convergence with respect to low-rank approximation. We conclude with stability analysis of the block Lanczos method.

3.1 Random projections for low-rank matrix approximation

Random projections may be considered quasi-Krylov subspace methods. Halko et. al. proposed a variety of them in [36], but we focus on the "direct eigenvalue approximation for Hermitian matrices with random projections" algorithm. This algorithm is essentially the Raleigh-Ritz orthogonal projection method using normalized power iteration [4, 50] to generate the projection matrix. The algorithm is introduced in Algorithm 3 herein. This method is presented as an eigenvalue approximation method; to effect a low-rank matrix approximation, one may simply use the Ritz vector matrix $V = QU$ and the Ritz value matrix to produce the low-rank approximation as $\hat{A}^{(k)} = V\Theta V^T$. If A is rectangular, then the Ritz vector matrix can either be used for

$$\hat{A}^{(k)} = V^T A \tag{3.1}$$

if $V\Theta V^T \approx AA^T$ or used to approximate A directly as

$$\hat{A}^{(k)} = V\Theta^{\frac{1}{2}} \tag{3.2}$$

if $V\Theta V^T \approx A^T A$. The square root of Θ always exists, provided that no null eigenvector of A is in $\text{span}\{A^p\Omega\}$; this square root is also easy to compute as Θ is diagonal.

This algorithm is simple and elegant. We have also remarked that it is robust to the round-off errors that cause loss of orthogonality in Lanczos algorithms. One may observe that no true iteration occurs in this algorithm; all orthogonality-enforcing operations work on Q as a whole.

The algorithm does not explicitly specify a method for generating the projection basis Q; many candidate methods are studied in [36]. However, the QR factorization $QR = A^p\Omega$ is suggested and forms the basis for the probabilistic analyses of the error of the algorithm presented in [36]. Use of that same Q also allows for more direct analytic comparison against traditional Krylov subspace algorithms. The choice of span$\{A^p\Omega\}$ is further suggested by its relationship to Harmonic Ritz vectors [56]; the space span$\{A^p\Omega\}$ is the minimal basis for Harmonic Ritz vectors, which are minimum-residual approximations to eigenvectors rather than the Galerkin approximations. Harmonic Ritz vectors have been noted as good choices for a subspace for the Rayleigh-Ritz method [52] or choices for restarts of the Lanczos process [41].

The algorithmic complexity of the method is dominated by the QR factorization on line 1 and the formation of $\hat{A}$ on line 2. The matrix $A^p\Omega$ will be $n \times k$ and each basis vector $i \le k$ must be orthogonalized against all $i-1$ previous vectors; therefore the resulting complexity is $O(nk^2)$. The formation of $\hat{A}$ on line 2 also requires k pairwise inner products, for a complexity of $O(nk^2)$ as well. Overall, the method requires $4k^3 + 4k^2n + 4kn_{\text{nnz}} - kn$ FLOPS to generate a k-dimensional approximation, where A has n_{nnz} non-zero elements.

3.2 Block Lanczos for low-rank approximation

Block Krylov subspaces may be used to generate low-rank matrix approximations much in the way that single-vector Krylov subspaces are used for low-rank matrix approximations in [15, 75]. An advantage of a Krylov subspace method is that it produces both an orthonormal basis Q for the projection subspace $\mathscr{K}_i(A, x_0)$ and the projection $T_{k,k}$ of A in that subspace. This eliminates the need for an expensive $O(nk^2)$ operation to project A onto the solution subspace. Moreover, the projected matrix $T_{k,k}$ has band tridiagonal structure, which makes manipulation of it more efficient than if it had arbitrary structure. Block Lanczos methods are widely used for solving eigenproblems [16,31,33, 88], and are also used in low-rank matrix approximation problems for model reduction [25,27,34,35]. The block Lanczos method may exhibit faster convergence of eigenvalues, especially when they are tightly clustered [67]. The block Lanczos method requires $4b^2kn + 2bkn_{\text{nnz}}$ FLOPS to produce a bk-dimensional subspace, where b is the block size. If $b = k/2$, then the block Lanczos routine requires $3k^2n + 2kn_{\text{nnz}}$ FLOPS.

In the case that A is square and Hermitian, then we may simply approximate A with

$$\hat{A}^{(k)} = QT_{k,k}Q^T. \tag{3.3}$$

If A is rectangular, then we generate $\mathscr{K}_i(A^TA, x_0)$ and may use the Cholesky factorization $LL^T = T_{k,k}$, as $T_{k,k}$ is positive-definite provided that no null eigenvector of A^TA has converged in the Krylov subspace. Then we may use

$$\hat{A}^{(k)} = Q^TL^T. \tag{3.4}$$

These formulas hold for either single-vector or block Krylov subspaces.

We note that the literature examples of Krylov subspaces for low-rank matrix approximation use minimal Krylov subspaces $\mathscr{K}_i(A, x_0)$ with $i = k$. That is, to produce a k-dimensional approximation, the Krylov subspace $\mathscr{K}_k(A, x_0)$ is used. This is by no means the only way to apply a Krylov subspace to a low-rank approximation problem. One may generate a Krylov subspace $\mathscr{K}_i(A, x_0)$ with $i > k$ and truncate using the leading k Ritz vectors. The low-rank approximation becomes

$$\hat{A}^{(k)} = V_k\Theta_k \tag{3.5}$$

where $T_{i,i} = U\Theta U^T$ with Ritz pairs (u_j, θ_j), $\Theta_k = \text{diag}(\theta_1, \theta_2, \ldots, \theta_k)$ and $V_k = [Qu_1\ Qu_2\ \ldots\ Qu_k]$. Intuitively, we expect that more iterations will lead to more converged Ritz values which then will have larger magnitudes, and that the resulting low-rank approximations will be better.

The block Lanczos algorithm [4, 16] is the block version of the single-vector Lanczos algorithm used in [15] for generating a orthonormal basis for the Krylov subspace $\mathscr{K}_i(A, x_0)$. The algorithm is presented in Algorithm 2. The salient difference in use is that the single-vector Lanczos method accepts a vector x_0 and produces an orthonormal basis for $\mathscr{K}_i(A, x_0)$ and the block Lanczos method accepts a matrix X_0 to produce $\mathscr{K}_i(A, X_0)$. Both methods are based on the *Lanczos recurrence*

$$Q_{i+1}B_{i+1} = AQ_i - Q_iA_i - Q_iB_{i-1} \tag{3.6}$$

where A_i and B_i are from the Lanczos algorithm, with $A_i = Q_i^TAQ_i$. The Lanczos recurrence gives a three-term relation for enforcing orthogonality between basis vectors of the Krylov subspace. Note that the block Krylov subspace algorithm is simply a generalization of the single vector Lanczos algorithm, and reduces to it when X_0 has 1 column. Though the Lanczos recurrence holds in exact arithmetic, in finite-precision floating point arithmetic the equality is not exact. Relying solely on (3.6) will lead to a gradual loss of orthgonality in the columns of Q as iterations proceed.

3.2.1 Block Lanczos for with refinement for improved stability

Round-off error causes loss of orthogonality for several reasons, but lost orthogonality compounds over iterations. Moreover, round-off error may cause adjacent Lanczos blocks to be non-orthogonal. Therefore minimizing round-off error produced early in Lanczos iteration will help maintain stability. The steps on lines 3 and 4 are a source of initial error. These lines are equivalent to a classical Gram-Schimdt orthgonalization of R against span$\{Q_j\}$. However, $R - Q_1 A_1$ will not be completely orthogonal to Q_1 with round-off error. Adding an extra refinement step similar to the modified Gram Schmidt process results in Algorithm 4, the block Lanczos method with a modified orthogonalization step. The addition of the extra stabilization step at line 6 has apparently not been suggested previously. The extra stability added by the refinement would only serve to delay the loss of orthogonality between Lanczos basis vectors; round-off error would compound anyway and some reorthogonalization routine would still be necessary. We suspect that this is the reason that a refinement step augmentation for the Lanczos algorithm has not been proposed before.

The addition of refinement may only maintain acceptable orthogonality for a few extra iterations, which is not significant if the number of Lanczos iterations is assumed to be large. However, if the number of iterations is fixed beforehand and small, then the addition of a refinement step may be sufficient to maintain orthgonality and eliminate the need for reorthogonalization. This modification adds a non-trivial cost to the classical Lanczos algorithm, as the projections $Q_j^T R$ are among the most expensive operations per iteration, assuming the cost of the sparse matrix-vector products AQ_j is negligible. This method does reduce round-off error in the initial phase of the algorithm; even if only two iterations are performed, the error due to lines 3 and 4 in Algorithm 2 may leave the maximum cosine between basis vector much larger than machine precision, but lines 5 and 6 in

Algorithm 4 Block Lanczos with refinement

```
Require: a priori chosen start block $Q_1$
$R \leftarrow AQ_1$
for $j = 1 \rightarrow k$ do
    $A_j \leftarrow Q_j^T R$
    $R \leftarrow R - Q_j A_j$
    $S \leftarrow Q_j^T R$
    $R \leftarrow R - Q_j S$
    QR factorize $R = Q_{j+1} B_{j+1}$
    $R \leftarrow AQ_{j+1}$
    $R \leftarrow R - Q_j B_{j+1}^T$
end for
return $[Q_1\, Q_2\, \ldots], \begin{bmatrix} A_1 & B_2^T & & \\ B_2 & A_2 & B_3^T & \\ & B_3 & A_3 & B_4^T \\ & & \ddots & \ddots & \ddots \end{bmatrix}$
```

Algorithm 4 reduce this error. The minimization of initial error not only improves the case for 2 iterations, but implies slower loss of orthogonality in the next few iterations. The extra refinement adds $2b^2kn$ FLOPS to the ordinary block Lanczos method.

3.2.2 The shrink-and-iterate approach

The complexity of the block Lanczos algorithm is determined by the block size b, the number of columns of $Q_1 = X_0$. The matrix-matrix product on line 4 and the QR-factorization on line 5 require $O(nb^2)$ operations. This is the same complexity as the random projection methods in Algorithm 3 when $b = k$ and the block Lanczos algorithm is asymptotically as expensive as the random projection method. If one uses a smaller block size, then each iteration of the block Lanczos routine will be accelerated. For example, if the block size is $b/2$, then each iteration of block Lanczos will be roughly four times as fast as with a block size of b. One may expect to perform four Lanczos iterations in roughly the same amount of time as is required to form a k-dimensional subspace using the random projection method in Algorithm 3; the resulting Krylov subspace will be $2k$-dimensional. In general, one may shrink the block size by a block shrinkage factor of s and generate $\mathscr{K}_{s^2}(A, x_0)$.

The above complexity analysis suggests that shrinking block sizes will always lead to computational gains, provided that the sparse matrix-vector products needed to form the AQ_i are not expensive. The irresistible conclusion that setting the block shrinkage factor $s = k$, resulting in single-vector Lanczos and generating a k^2-dimensional subspace is always the best choice may be tempered by two important observations:

1. the increasingly long Krylov subspaces generated by the shrink-and-iterate scheme described above will require storage of all k^2 Lanczos basis vectors, and
2. the Lanczos algorithm will require reorthogonalization at some point to recover orthgonality lost due to round-off error.

Item 1 is a non-trivial limitation; memory constraints led to the restarted Lanczos routines [3, 41] that limit the number of Lanczos vectors stored. We eschew restarted methods due to their computational costs and algorithmic complicatedness; our overarching goal is to develop a block Lanczos method that requires no stabilization and does not perform many iterations. It is also possible to simply not generate all s^2 Lanczos vectors, but item 2 limits the number of iterations available without any stabilization. In order to both present modest storage requirements and simultaneously require no stabilization, any increases in iteration must be limited to small s. Spectral properties govern both the convergence of eigenvalues in Krylov subspaces and loss of orthgonality in the basis vectors. Therefore, spectral properties of the input matrix A determine the cases for which block Lanczos with a block size shrinkage of s will produce a better low-rank approximation than Algorithm 3 and the cases for which block Lanczos will not require stabilization.

From the above complexity analysis, it is not difficult to see that the shrink-and-iterate method previously described can allow the block Lanczos method to generate a Krylov subspace $\mathscr{K}_i(A,X_0)$ of dimension $i > k$ in the same amount of time necessary for the random projection method to produce a subspace of size k. Though truncation by Ritz vectors may lead to better low-rank approximations of A when the leading k Ritz values of the Krylov subspace are larger than the Ritz values produced by the random projection method and the number of iterations beyond k available to a block Lanczos routine are constrained both by memory and by loss of orthogonality, the number of iterations and block size also influence convergence of eigenvalues in the Krylov subspace. Through analysis of the spectrum of A we may develop sufficient conditions for a block Lanczos method with more iterations to lose negligible orthogonality and still produce a better low-rank approximation than the random projection method. However, we first investigate a hybrid random-projections Krylov subspace method that aims to combine the good stability of the random projection method and the improved computational complexity of the shrink-and-iterate approach.

3.2.3 The hybrid random projection-Krylov subspace method

Both random projections and block Lanczos algorithms have their own particular advantages and drawbacks. Random projections can produce good approximations with no loss of orthogonality in the basis they produce, but the computational complexity is quadratic in the number of dimensions produced. Block Krylov subspaces can have better computational complexity from shrinking the block size by a factor of s which allows for a larger solution subspace to be produced with the same complexity as random projections. Then one can either perform s^2 iterations and potentially get a better low-rank approximation, or perform fewer than s^2 iterations and simply be faster than random projections. Ideally, we would like to produce a low-rank approximation with smaller error than the random projection method, avoid storage of an s^2-dimensional subspace, have no loss of orthogonality, and be faster. The hybrid random projection-Krylov subspace method attempts to combine the two algorithms to get the best qualities of both. The hybrid approach is presented in Algorithm 5.

The hybrid approach is rather straightforward; the random projection method is used to produce a start block for the block Lanczos method with refinement. The motivation for the hybrid approach is to approximate $\mathscr{K}_{s^2}(A,X_0)$ with $\mathscr{K}_{s^2-p}(A,A^pX_0)$. Increasing the value of p reduces the number of basis vectors stored and also reduces the number of Lanczos iterations. Clearly, it is desirable to store fewer basis vectors, and intuitively, less orthogonality will be lost with fewer Lanczos iterations. As was noted in the discussion of Algorithm 4, the block Lanczos algorithm run for only 2 iterations is equivalent to a full Gram-Schmidt process when refinement is used. Setting $p = s = 2$ and using Algorithm 4 will result in no more loss of orthogonality than the random projection method.

Algorithm 5 Hybrid random projection-Krylov subspace approach

```
Require: a priori chosen random start block $X_0$ with linearly-independent columns, power factor $p$
QR factorize $X_1, R = A^p X_0$
return $Q, T$ from 2 iterations of Algorithm 4 or 2 with start block $X_1$.
```

We conclude our discussion of all block Lanczos methods by noting that our complexity analysis has assumed that sparse matrix-vector products are of negligible cost compared to the dense matrix operations in algorithms 3 and 2. This is not always the case; if A is dense then forming AQ_j will be more expensive than the cost of QR factorizing the block vector AQ_j. Also, if A is implicitly defined in terms of a rectangular $n \times m$ B with $m \gg n$, then the product AQ_j may be more expensive than the QR factorization of AQ_j even when B is sparse. If the costs of sparse matrix-vector products are the dominant compute costs and no substantial computation advantages are realized for dense matrix operations over sparse matrix-vector operations, 3 iterations of block Lanczos with a 50% block size may still be less computationally expensive than random projections to produce a k-dimensional approximation. The block Lanczos method simply requires fewer sparse matrix-vector products than the random projection method.

3.3 Convergence Analyses

We have observed that shrinking the block size can lead to Lanczos iteration that is faster than the random projection algorithm. This speed advantage alone is insufficient to recommend Lanczos iteration with a shrunken block over the random projection algorithm. Rather, the complexity observations must be combined with error bounding analyses on both Algorithm 3 and Algorithm 2 to determine those cases for which block Lanczos with a smaller block will produce better low-rank approximations than Algorithm 3.

Bounds for low-rank matrix approximation error exist for Algorithm 3 and for eigenvalue approximations from Krylov subspaces. These existing bounds are not useful in combination with each other to determine cases for which the shrink-and-iterate approach will produce a low-rank approximation with lower error than the random projections method. Both the bounds on the random projections error and the eigenvalue approximation error bounds for Krylov subspaces produce lower bounds, and for comparison of the two algorithms we require upper bounds for one and lower for the other. Moreover, existing Krylov subspace bounds were originally intended to be elegant expressions for the asymptotic worst-case error. Their derivation uses simplifications that drastically improve clarity at the expense of tightness. In some cases, they are rather pessimistic; we have observed that they are so pessimistic for short Krylov sequences as to not be useful for comparison with the random projections method.

Due to these limitations of existing bounds, we derive results that allow for direct comparison of random projections and the shrink-and-iterate application of Krylov subspaces. These results give exact formulations of the norm of low-rank approximations for the random projection method and the shrink-and-iterate approach, so that tightness is not an issue. These formulations discard the asymptotic form of previous bounds of eigenvalue approximations from Krylov subspaces to exploit the shortness of the Krylov sequence in the shrink-and-iterate approach. In particular, we aim to show cases for which a Krylov sequence of length 2 and a start block of size b will produce a better low-rank approximation than random projections with a start block of size $2b$. Before deriving these new expressions, we review existing bounds on random projections errors from Halko [36] and eigenvalue approximations.

3.3.1 Review of existing bounds for random projections and Krylov subspaces

We first review the available bounds for low-rank approximations for random projections and Krylov subspaces. For random projections, we have an elegant result due to Halko et. al.

Theorem 5 (Halko, Martinsson and Tropp [36]). *Let $A \in \mathbb{R}^{n\times m}$ be the input matrix with partitioned singular value decomposition*

$$A = [U_1\ U_2] \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} [V_1^T\ V_2^T] \tag{3.7}$$

with U_1, Σ_1 and V_1, having k columns.Then the error in low rank approximation $\hat{A}^{(k)}$ using the subspace generated by Algorithm 3 operating on AA^T is bounded as

$$\|A - \hat{A}^{(k)}\| \leq \|\Sigma_2\| + \|\Sigma_1\Omega_2\Omega_1^+\|^{1/p} \tag{3.8}$$

with $\Omega_1 = U_1^T\Omega$, $\Omega_2 = U_2^T\Omega$ and random sample matrix Ω for $\|\cdot\|$ being either the spectral or Frobenuis norm.

Eigenvalue approximation bounds exist in many forms for Krylov subspace approximations. These eigenvalue approximation results may be used to construct low-rank approximation bounds for $\|A-\hat{A}^{(k)}\|$ indirectly for cases which produce $\hat{A}^{(k)}$ by restricting it to a subspace. From Theorem 4, we have $\|A-\hat{A}^{(k)}\|_F^2 = \mathrm{tr}(AA^T) - \mathrm{tr}(\hat{A}^{(k)}\hat{A}^{(k)T})$. When A is square and positive semidefinite, we have $\|A-\hat{A}^{(k)}\|_* = \mathrm{tr}(A-\hat{A}^{(k)}) = \mathrm{tr}(A) - \mathrm{tr}(\hat{A}^{(k)})$. In both cases, we can infer the improvement in low-rank approximation error directly from the improvement in the norm of $\hat{A}^{(k)}$; therefore convergence of eigenvalue approximations suggests low-rank approximation error. The following theorem is a generalization of Theorem 3 to convergence of singular values of A in the spaces $\mathcal{K}_i(AA^T, x_0)$ and $\mathcal{K}_i(A^TA, Ax_0)$.

Theorem 6 (Golub, Luk and Overton [31]). *Let $A \in \mathbb{R}^{n\times m}$ be the input matrix with singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\text{inf}}$. Consider the matrix AA^T with spectral decomposition $AA^T = U\Lambda U^T$ and eigenvalues $\lambda_1 = \sigma_1^2 \geq \lambda_2 = \sigma_2^2 \geq \ldots \geq \lambda_{\text{inf}}^2 = \sigma_{\text{inf}}^2$ and corresponding eigenvectors $u_1, u_2, \ldots, u_{\text{inf}}$. Apply s steps of the block Lanczos algorithm in Algorithm 2 with block size b to generate $\mathcal{K}_s(AA^T, X_0)$ and the projection $T_{s,s}$ of $AA^T \in \mathcal{K}_s(AA^T, X_0)$. Use $\lambda_i^{(s)}$ to denote the i^{th} eigenvalue of $T_{s,s}$. Then for $i = 1,2,\ldots,b$ the eigenvalue estimate $\lambda_i^{(s)} = \sigma_i^{(s)2}$ is bounded as*

$$0 \leq \sigma_i^2 - \sigma_i^{(s)2} \leq (\sigma_i^2 - \sigma_{\text{inf}}^2)\frac{\tan^2\theta}{T_{s-1}\left(\frac{1+\gamma_i}{1-\gamma_i}\right)^2} \tag{3.9}$$

for

$$\gamma_i = \frac{\sigma_i^2 - \sigma_{b+1}^2}{\sigma_i^2 - \sigma_{\text{inf}}^2} \tag{3.10}$$

and $\theta = cos^{-1}\tau$ where τ is the smallest singular value of $[u_1\ u_2\ \ldots\ u_b]^T X_0$.

A nearly-identical result is also available for the *trailing* singular values of A [17]. It bounds errors of approximations of small eigenvalues; it therefore gives upper bounds on the trailing b eigenvalue approximations from the Krylov subspace. Then simply using these bounds on $\mathcal{K}_2(A, X_0)$ does not result in bounds which may be used in conjunction an upper bound on $\|\hat{A}_{\text{RP}}^{(k)}\|$, where $\|\hat{A}_{\text{RP}}^{(k)}\|$ is the rank-k approximation produced by random projections, to determine which method will produce a smaller error. Another serious limitation on these bounds for comparing the shrink-and-iterate approach to random projections is that they do not provide for bounds for all sb eigenvalue approximations from $\mathcal{K}_s(A, X_0)$; bounds on all sb eigenvalue approximations is required for comparison against $\|\hat{A}_{\text{RP}}^{(k)}\|$. The limitation on bounding interior eigenvalues is an inevitable consequence of the methods used to produce the bounds, and bounding interior eigenvalues beyond s using the same methods is rather difficult.

3.3.2 Derivation of new bounds

Unfortunately, neither the bounds in Theorem 5 or those in 6 are useful for comparing the block Lanczos and random projection methods. The bound in (3.8) gives upper bounds for the error, while we require *lower* bounds to determine when block Lanczos will perform better. Although we do require lower bounds on the eigenvalue estimates of $\hat{A}^{(k)}\hat{A}^{(k)T}$ for block Lanczos as given by (3.9), the bounds only provide for the first b eigenvalue estimates. We require bounds on all k eigenvalue estimates, not just the leading b of them. Nevertheless, the Halko bounds do present us with a useful starting point to derive the necessary devices to compare random projections and Krylov subspaces.

We begin with a formula that characterizes exactly $\|\hat{A}^{(k)}_{\mathrm{RP}}\|$ for the random projection method. We note that this expression may be combined with the Kaniel-Paige-Saad bounds to produce an expression for predicting which method will produce a better low-rank approximation error, but the pessimism in the Kaniel-Paige-Saad bounds limits its usefulness. However, the expression for $\|\hat{A}^{(k)}_{\mathrm{RP}}\|$ not only gives an expression for the value of random projections, but also may be applied to generate a formula for part of $\|\hat{A}^{(k)}_{\mathscr{K}}\|$. With that, we then present an extension that gives an expression for all of $\|\hat{A}^{(k)}_{\mathscr{K}}\|$, by using the identities $\|A\|_F = \mathrm{tr}(A^TA)$ and $\mathrm{tr}(A) = \sum_{i=1}^n a_{ii}$ with a block decomposition of A^TA. The bounds for random projections can be used for the upper-left-hand block of A restricted to $\mathscr{K}_2(A, A^2X_0)$, and the extension gives the trace of the lower-right-hand block of A restricted to $\mathscr{K}_2(A, A^2X_0)$. As $\mathscr{K}_2(A, A^2X_0) \subset \mathscr{K}_4(A, X_0)$, the eigenvalue approximations from A restricted to $\mathscr{K}_2(A, A^2X_0)$ are at least as good as the approximations restricted to $\mathscr{K}_4(A, X_0)$. We begin with proof of a new theorem characterizing the error of random projections or one iteration of the block Lanczos process.

Theorem 7. *Let A be given as in Theorem 5. For a random sampling matrix $\Omega \in \mathbb{R}^{n\times k}$, with QR factorization $\Omega = X_0S$, let $\hat{A}^{(k)}$ be the low-rank approximation obtained by Algorithm 3 with input AA^T. Let $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\inf}$ be the singular values of A with corresponding left singular vectors $u_1, u_2, \ldots, u_{\inf}$. Then $\|\hat{A}^{(k)}\|_F^2$ is given by*

$$\|\hat{A}^{(k)}\|_F^2 = \sum_{i=1}^{\min\{m,n\}} \sigma_i^{6p} \|(\Sigma^{2p}U^T\Omega)^{+T}\Omega^T u_i\|_2^2 \tag{3.11}$$

To prove this theorem, we first introduce and prove the following lemma.

Lemma 1. *Let A be an arbitrary real matrix and $AA^T = U\Sigma^2U^T$ be the spectral decomposition of the positive semi-definite Gram matrix AA^T. Then for any eigenvector u_i, the norm of the image $\hat{u}_i$ of u_i*

in the space $S = \operatorname{span}\{(AA^T)^p\Omega\}$ *is given by*

$$\|\hat{u}_i\|_2 = \sigma_i^{2p}\|(\Sigma^{2p}U^T\Omega)^{+T}\Omega^T u_i\|_2. \tag{3.12}$$

Proof. Let $QR = (AA^T)^p\Omega$ be the QR factorization of $(AA^T)^p\Omega$. Then Q is an orthonormal basis for $\operatorname{span}\{(AA^T)^p\Omega\} = S$ and

$$Q = (AA^T)^p\Omega R^{-1} \tag{3.13}$$

as R is nonsingular, provided that no null eigenvector is in $\operatorname{span}\{\Omega\}$. Then

$$\|\hat{u}_i\|_2 = \|((AA^T)^p\Omega R^{-1})^T u_i\|_2 = \|R^{-T}\Omega^T(AA^T)^p u_i\|_2 = \sigma_i^{2p}\|R^{-T}\Omega^T u_i\|_2 \tag{3.14}$$

We have $QR = (AA^T)^p\Omega$, so

$$\sigma(QR) = \sigma(U\Sigma^{2p}U^T\Omega) = \sigma(\Sigma^{2p}U^T\Omega) \tag{3.15}$$

where $\sigma(A)$ represents the singular values of A, as singular values are invariant under the orthonormal rotations Q and U. Combining (3.14) and (3.15) we get

$$\|\hat{u}_i\|_2 = \sigma_i^{2p}\|(\Sigma^{2p}U^T\Omega)^{+T}\Omega^T u_i\|_2 \tag{3.16}$$

which completes the proof. □

This lemma gives a *a priori* formula for the norms of the projections of eigenvectors of A in the space $\operatorname{span}\{(AA^T)^p\Omega\}$. With that, we can bound the norm of $\hat{A}^{(k)}$.

Proof of Theorem 7. We use $\lambda_i(A)$ to represent the i^{th} eigenvalue of A. By definition, $\|A\|_F^2 = \operatorname{tr}(A^TA) = \sum_{i=1}^n \lambda_i(A^TA)$ for any A. Also $AA^T \sim A^TA$, so $\|A\|_F^2 = \operatorname{tr}(A^TA) = \operatorname{tr}(AA^T)$. From the definition of $\hat{A}^{(k)}$ in (3.1), we have

$$\hat{A}^{(k)} = Q^TA \tag{3.17}$$

for any basis Q of span$\{(AA^T)^p\Omega\}$. Then

$$\|\hat{A}^{(k)}\|_F^2 = \mathrm{tr}(A^TQQ^TA) = \mathrm{tr}(V\Sigma^TU^TQQ^TU\Sigma V^T). \tag{3.18}$$

where $A = U\Sigma V^T$ is the singular value decomposition of A. With the similarity transform $V^T \cdot V$,

$$\|\hat{A}^{(k)}\|_F^2 = \mathrm{tr}(\Sigma U^TQQ^TU\Sigma). \tag{3.19}$$

Without loss of generality, we order the diagonal entries of Σ and columns of U such that $\sigma_{11} \geq \sigma_{22} \geq \ldots \geq \sigma_{\inf}$ and $U = [u_1\ u_2\ \ldots\ u_{\inf}]$. Let $\hat{U} = Q^TU$. As Q is a basis for span$\{(AA^T)^p\Omega\}$, we have

$$\|\hat{u}_i\|_2 = \sigma_i^{2p}\|(\Sigma^{2p}U^T\Omega)^{+T}\Omega^Tu_i\|_2 \tag{3.20}$$

from Lemma 1. Substituting $\hat{U}$ into (3.19), we get

$$\|\hat{A}^{(k)}\|_F^2 = \mathrm{tr}(\Sigma\hat{U}^T\hat{U}\Sigma) = \sum_{i=1}^{\infty}\sigma_i^{2p}\hat{u}_i^T\hat{u}_i = \sum_{i=1}^{\infty}\sigma_i^{2p}\|\hat{u}_i\|_2^2 = \sum_{i=1}^{\infty}\sigma_i^{2p}(\sigma_i^{2p}\|(\Sigma^{2p}U^T\Omega)^{+T}\Omega^Tu_i\|_2)^2 \tag{3.21}$$

which completes the proof. □

Theorem 7 is similar to Theorem 5 from Halko et. al. [36] but gives an exact formulation of the Frobenius norm of $\hat{A}^{(k)}$. The theorem does give a more direct means of comparison of the difference in Frobenius norm of the random projection method and block Krylov subspaces for low-rank approximation.

With Theorem 7, we may begin drawing conclusions regarding matrices for which a shrink-and-iterate scheme will produce better low-rank approximations than the random projection method. Combination of (3.11) and the lower bounds on eigenvalue approximations in (3.9) yields the following proposition.

Proposition 1. *Let A be a positive semi-definite matrix, X_0 be a real matrix with $2b$ orthonormal columns and AX_0 defined, let $\hat{A}^{(k)}_{\mathrm{RP}}$ be the low-rank approximation computed by the random projections method in Algorithm 3, let $\hat{A}^{(k)}_{\mathcal{K}}$ be the low-rank approximation generated by k iterations of the* *block Lanczos method described in Algorithm 2 with inputs $A = A$ and $Q_1 = X_S$ where X_S is composed of some arbitrary b columns of X_0, let γ_i defined as in Theorem 6, let $\theta = \cos^{-1} \min_S \min_{\sigma_{\text{inf}}} U_b^T X_S$ where S is a subset of columns of X_0 with $|S| = b$ and $X_s = [x_i \ x_j \ \ldots]$ for $x_i \in S$, $x_j \in S$. Then the squared Frobenius norm of $\hat{A}_{\text{RP}}^{(k)}$ is strictly less than the squared Frobenius norm of $\hat{A}_{\mathscr{K}}^{(k)}$ whenever*

$$\|\hat{A}_{\text{RP}}^{(k)}\|_F^2 = \sum_{i=1}^{\infty} \sigma_i^4 \|(\Sigma^2 U^T X_0)^{+T} X_0^T u_i\|_2^2 < \sum_{i=1}^{b} (\lambda_i - \lambda_{\text{inf}}) \frac{\tan^2\theta}{T_{k-1}\left(\frac{1+\gamma_i}{1-\gamma_i}\right)^2} \leq \|\hat{A}_{\mathscr{K}}^{(k)}\| \tag{3.22}$$

We remark that

- when A is rectangular, we may substitute the positive semi-definite Gram matrix AA^T,
- for $s = 4$ and $k = b$ the Krylov subspace generated in this proposition is the same Krylov subspace generated by the "shrink-and-iterate" method,
- that this result may be valid in theory, but the pessimistic nature of the Kaniel-Paige-Saad bounds from Theorem 6 limits their usefulness. In fact, we have observed no case in which the bounds produce sufficient tightness to ever predict the shrink-and-iterate method will produce a better approximation, even when it outperforms the random projections method by a substantial margin.

For further tightness we consider an alternative to the Kaniel-Page-Saad bounds in the right-hand side of (3.22). These new bounds will not use the polynomial methods used in the Kaniel-Page-Saad bounds, but will instead use a more direct approach with the assumption that the Krylov sequence length is at most 2. This assumption sacrifices generality in numbers of iterations in exchange for producing a formula that, when combined with (3.11) gives the value for $\|\hat{A}_{\mathscr{K}}^{(k)}\|$. We intend to use this formula to give a value for $\|\hat{A}_{\mathscr{K}}^{(k)}\| \in \mathscr{K}_2(A, A^2X_0)$, which in turn bounds $\|\hat{A}_{\mathscr{K}}^{(k)}\| \in \mathscr{K}_4(A, X_0)$.

The Kaniel-Paige-Saad bounds provide elegant expressions that overcome the recurrence relations that would be otherwise necessary to formulate *a priori* bounds on eigenvalue approximations in Krylov subspaces. The bounds are applicable to Krylov subspaces of any dimension. When one is considering bounds for Krylov subspaces of arbitrary length, the direct expressions for eigenvalue approximations would be prohibitively cumbersome or inaccurate. However, for small Krylov sequence length i, reasonable bounds may be generated directly. To that end, we present the following proposition.

Proposition 2. *Let A be a positive semi-definite matrix, and X_0 be a matrix with orthonormal columns and AX_0 defined. Consider the Krylov subspace $\mathscr{K}_2(A, A^p X_0)$; then the projection of A into $\mathscr{K}_2(A, A^p X_0)$ is*

$$T_{2,2} = \begin{bmatrix} A_1 & B_2^T \\ B_2 & A_2 \end{bmatrix} \tag{3.23}$$

and A_1 and B_2 are the matrices produced by the block Lanczos algorithm.

We note that

- the block A_1 is the low-rank approximation generated by the random projection method in Algorithm 3 with inputs $A = A$ and $\Omega = A^p X_0$,
- the trace $\mathrm{tr}(T_{2,2}) = \mathrm{tr}(A_1) + \mathrm{tr}(A_2)$,
- if A is rectangular, and the block Lanczos method operates on AA^T, then $\|\hat{A}^{(k)}\|_F^2 = \mathrm{tr}(T_{2,2})$.

To produce bounds on nuclear norm or squared Frobenius norm error, we simply require a corresponding formula for $\mathrm{tr}(A_2)$ and a provision that eigenvalue approximations from $\mathscr{K}_i(A, A^p X_0)$ are never better than eigenvalue approximations from $\mathscr{K}_{i+p}(A, X_0)$. Therefore, we require implements to reason about how eigenvalue approximations from $\mathscr{K}_i(A, X_0)$ compare against eigenvalue approximations from $\mathscr{K}_{i-1}(A, AX_0)$. We introduce and prove the following lemma to that end.

Lemma 2. *Let A be a symmetric positive semi-definite matrix and $\lambda_i^{(j)}$ be the i^{th} eigenvalue estimate from A restricted to $\mathscr{K}_j(A, X_0)$ and $\lambda_i^{(n-1)}$ be the i^{th} eigenvalue approximation of A restricted to $\mathscr{K}_{j-1}(A, X_0)$. Then*

$$\lambda_i^{(j)} \geq \lambda_i^{(j-1)}. \tag{3.24}$$

Proof. Suppose by contradiction otherwise. Let v_i be Ritz vectors of A with $v_i \in \mathscr{K}_j(A, X_0)$ and u_i be Ritz vectors of A with $u_i \in \mathscr{K}_{j-1}(A, X_0)$. Since $\mathscr{K}_{j-1}(A, X_0) \subset \mathscr{K}_j(A, X_0)$ and the Ritz vectors v_i

form an orthonormal basis for $\mathscr{K}_j(A,X_0)$, any Ritz vector $u_i \in \mathscr{K}_{j-1}(A,X_0)$ may be written as a linear combination of v_i:

$$u_i = \sum_{i=1}^{j} c_i v_i. \tag{3.25}$$

Let $V = [v_1 \; \dots \; v_j]$. Then $V^T AV$ is the restriction of A to $\mathscr{K}_j(A,X_0)$, and $V^T AV = \Lambda^{(j)}$. Since u_i may be expressed in terms of the vectors v_i, $VV^T u_i = u_i$. Let $U = [u_1 \; \dots \; u_{j-1}]$ and $\bar{U} = V^T U$. Then we may write Ritz values $\lambda^{(j-1)}$ as the diagonal of

$$\operatorname{diag}(U^T AU) = \operatorname{diag}(U^T VV^T AVV^T U) = \operatorname{diag}(\bar{U}^T \Lambda^{(j)} \bar{U}). \tag{3.26}$$

Since the matrix $\Lambda^{(j)}$ is also diagonal, then its eigenvalues are simply its diagonal entries and its eigenvectors are the element vectors e_i. Assume without loss of generality that diagonal entries of $\Lambda^{(j)}$ are ordered such that $e_1^T \Lambda^{(n)} e_1 \geq e_2^T \Lambda^{(j)} e_2 \geq \ldots \geq e_n^T \Lambda^{(j)} e_j$. Then, by the Courant characterization of eigenvectors, there is no other set of orthonormal vectors $\{x_i \mid 1 \leq i \leq j\}$ with $x_1^T \Lambda^{(j)} x_1 \geq x_2^T \Lambda^{(n)} x_2 \geq \ldots \geq x_j^T \Lambda^{(j)} x_n$ such that $x_i^T \Lambda^{(j)} x_i > e_i^T \Lambda^{(j)} e_i$. But from our assumption, at least one Ritz value had $\lambda^{(j-1)} > \lambda^{(jx)}$ which implies that at least one Ritz vector has $\bar{u}_i^T \Lambda^{(j)} \bar{u}_i > e_i^T \Lambda^{(j)} e_i$, which is a contradiction as the Ritz vectors $\bar{u}_i$ are orthonormal.

□

Now that we have shown that the eigenvalue approximations from A restricted to $\mathscr{K}_4(A,X_0)$ are at least as good as those from A restricted to $\mathscr{K}_2(A,A^2X_0)$, we may use the expression from Theorem 7 to quantify the trace of A_1 from (3.35). Now we develop an expression for $\operatorname{tr}(A_2)$, which is simply $\operatorname{tr}(Q_2^T AA^T Q_2)$. We use a strategy similar to that of Theorem 7; first we develop a formula for the image $\bar{u}_i$ of a left singular vector u_i of A in $\operatorname{span}\{Q_2\}$, and then use that to generate a formula for $\|Q_2^T A\|_F^2$. We begin by presenting and proving the following lemma.

Lemma 3. *Let $A \in \mathbb{R}^{n \times m}$ be an arbitrary matrix with singular value decomposition $A = U\Sigma V^T$ and singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\text{inf}}$, and X_0 be an orthonormal $n \times b$ matrix. Let $Q_1 R = (AA^T)^p X_0$ and Q_2 be the orthonormal matrices generated by 2 iterations of the block Lanczos algorithm in presented in Algorithm 2 applied to AA^T. Then the norm $\|\bar{u}_i\|_2^2$ of a left singular vector u_i of A in* $\operatorname{span}\{Q_2\}$ *is* *given by*

$$\|\bar{u}_i\|_2^2 = \|(\sigma_i^{2p+2}\hat{u}_i^T R^{-1} - \sigma_i^{2p}\hat{u}_i^T R^{-1} A_1)B_2^{-1}\|_2^2 \tag{3.27}$$

Proof. Let A and X_0 be given as described. Then $Q_1 R = (AA^T)^p X_0$ and Q_2 is defined as

$$Q_2 B_2 = (AA^T)(AA^T)^p X_0 R^{-1} - (AA^T)^p X_0 R^{-1} A_1 \tag{3.28}$$

with $A_1 = (R^{-T} X_0^T (AA^T)^p) AA^T (AA^T)^p X_0 R^{-1}$. Clearly

$$((AA^T)^{p+1} X_0 R^{-1} - (AA^T)^p X_0 R^{-1} A_1) B_2^{-1} = Q_2 \tag{3.29}$$

and is an orthonormal basis for span$\{Q_2\}$.

We require a formula for the norms of the images of left singular vectors of A projected into span$\{Q_2\}$. If u_i is a left singular vector of A, then the image u_i in span$\{Q_2\}$ is

$$u_i^T((AA^T)^{p+1} X_0 R^{-1} - (AA^T)^p X_0 R^{-1} A_1) B_2^{-1} = (\sigma_i^{2p+2} u_i^T X_0 R^{-1} - \sigma_i^{2p} u_i^T X_0 R^{-1} A_1) B_2^{-1}, \tag{3.30}$$

which completes the proof □

Now we continue on to prove characterization of the norm of $\|Q_2^T A\|_F^2$.

Theorem 8. *Let $A \in \mathbb{R}^{n\times m}$ be an arbitrary matrix with singular value decomposition $A = U\Sigma V^T$ and singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\text{inf}}$, and X_0 be an orthonormal $n \times b$ matrix. Let $Q_1 R = (AA^T)^p X_0$ and Q_2 be the orthonormal matrices generated by 2 iterations of the block Lanczos algorithm in presented in Algorithm 2 applied to AA^T, with A_1 and B_2 also from the block Lanczos algorithm. Then the squared Frobenius norm $\|A^T Q_2\|_F^2$ is bounded as*

$$\|A^T Q_2\|_F^2 = \sum_{i=1}^{n} (\sigma_i^{2p+4}\hat{u}_i^T R^{-1} - \sigma_i^{2p+2}\hat{u}_i^T R^{-1} A_1) B_2^{-1} \tag{3.31}$$

with

$$\hat{u}_i = X_0^T u_i$$

for singular vectors u_i of A,

$$Q_2 B_2 = (AA^T)^{p+1} X_0 R^{-1} - AA^T X_0 R^{-1} A_1$$

and

$$A_1 = ((AA^T)^p X_0 R^{-1})^T (AA^T)(AA^T)^p X_0 R^{-1}.$$

Proof. Since $\|Q_2^T A\|_F^2 = \mathrm{tr}(A^T Q_2 Q_2^T A)$, then we note that $A = U\Sigma V^T$ and by substitution,

$$\|Q_2^T A\|_F^2 = \mathrm{tr}(A^T Q_2 Q_2^T A) = \mathrm{tr}(V\Sigma U^T Q_2 Q_2^T U \Sigma V^T). \tag{3.32}$$

Since trace is invariant under orthonormal transformations and $Q_2^T U = \hat{U}$,

$$\mathrm{tr}(V\Sigma U^T Q_2 Q_2^T U\Sigma V^T) = \mathrm{tr}(\Sigma U^T Q_2 Q_2^T U \Sigma) = \mathrm{tr}(\Sigma \bar{U}^T \bar{U} \Sigma) = \sum_{i=1}^n \sigma_i^2 \|\bar{u}_i\|_2^2. \tag{3.33}$$

From Lemma 3 we have $\|\bar{u}_i\| = (\sigma_i^{2p+2} \hat{u}_i^T R^{-1} - \sigma_i^{2p} \hat{u}_i^T R^{-1} A_1) B_2^{-1}$. Then

$$\sum_{i=1}^n \sigma_i^2 \|\bar{u}_i\|_2^2 = \sum_{i=1}^n (\sigma_i^{2p+2} \hat{u}_i^T R^{-1} - \sigma_i^{2p} \hat{u}_i^T R^{-1} A_1) B_2^{-1} \tag{3.34}$$

follows, which completes the proof. □

Combining the results from theorems 7 and 8 gives a sufficient condition for the shrink-and-iterate approach to produce a better low-rank approximation than the random projection method.

Proposition 3. *Suppose X_1 and X_2 are random and real orthonormal blocks with n rows and b columns each. Let the random projection method be run on a real input matrix A with start block $\Omega = [X_1\ X_2]$. The low-rank approximation of A restricted to $\mathcal{K}_2(AA^T, (AA^T)^2 X_1)$ will have larger norm than the low-rank approximation in* span$\{AA^T\Omega\}$ *whenever*

$$\begin{aligned} \textstyle\sum_{i=1}^n \sigma^6 \|(\Sigma^2 U^T \Omega)^+ \Omega^T u_i\|_2^2 < \quad & \textstyle\sum_{i=1}^n \sigma^{12} \|(\Sigma^4 U^T X_0)^+ X_0^T u_i\|_2^2 + \\ & \textstyle\sum_{i=1}^n \|(\sigma_i^8 u_i^T X_0 R^{-1} - \sigma_i^6 u^T X_0 R^{-1} A_1) B_2^{-1}\|_2^2 \end{aligned} \tag{3.35}$$

This proposition is a simple combination of Theorems 7 and 8.

3.3.3 Discussion

These new bounds lack the simplicity of the random projection bounds (3.8) or the Krylov subspace eigenvalue estimate bounds (3.9). Immediate insight into the spectral properties that discriminate between cases for which the the shrink-and-iterate approach will outperform the random projections method is difficult. Therefore, we will examine the bounds in (3.35) more closely to derive insight into what spectral properties will lead to a better-performing shrink-and-iterate approach over the random projections method.

Assume that the input matrix is square and positive semi-definite (if not substitute A in the following discussion with AA^T). We note that an equivalent expression to (3.11) is simply

$$\|\hat{A}^{(k)}\|_F^2 \le \sum_{i=1}^{n} \sigma_i^{6p} \|R^{-T} X_0^T u_i\|_2^2. \tag{3.36}$$

This observation leads to the following proposition regarding the relationship between the input matrix A and R in (3.36).

Proposition 4. *Let A be a positive semi-definite real matrix, and X_0 be a real matrix with b linearly independent columns and the product AX_0 defined. Then R^TR is the Cholesky factorization of A^{2p} restricted to the space* span$\{X_0\}$.

This proposition may be derived from observing the QR factorization $QR = A^pX_0$ and

$$R^TR = (QR)^TQR = X_0^TA^{2p}X_0. \tag{3.37}$$

We also remark that since $R^TR = X_0^TA^{2p}X_0$,

- R^TR is the result of the Rayleigh-Ritz procedure using the space span$\{X_0\}$,
- singular values $\sigma_i(R)$ approximate eigenvalues of $\lambda_i(A^p)$,
- singular values $\sigma_i(R^{-1})$ approximate eigenvalues of $\lambda_i(A^{-p})$.

As the norm of low-rank approximations from the shrink-and-iterate approach depends on both A_1 and A_2 from (3.23), we also must reason about tr(A_2). From Lemma 3, we have a formula for the norm of the image of left singular vectors of A in span$\{Q_2\}$, which leads to the following proposition.

Proposition 5. *Let A be a positive semi-definite real matrix, let X_0 be a real matrix with b orthonormal columns and AX_0 defined, let $Q_1R = A^pX_0$ be the QR factorization of A^pX_0, and let A_1 and B_2 be the matrices produced by one step of the block Lanczos process presented in Algorithm 2 run with inputs $A = A$ and $Q_1 = A^pX_0$. Let $A = U\Lambda U^T$ be the spectral decomposition of A and $\hat{u}_i$ be the image of eigenvector i of A in the space* span$\{A^pX_0\}$. *Then the squared norm of the image of eigenvector u_i of A in the space* span$\{Q_2\}$, *where Q_2 is from the block Lanczos algorithm is given by*

$$\|\bar{u}_i\|_2^2 = \|(\sigma_i^2\hat{u}_i^T - \hat{u}_i^T A_1)B_2^{-1}\|_2^2. \tag{3.38}$$

We remark that

- $\hat{u}_i^T\sigma_i^2 - \hat{u}_i^T A_1$ in (3.38) is the residual of the eigenpair (σ_i, u_i) when applied against the low-rank approximation $\hat{A}^{(b)} = A_1$ (b is from Proposition 4) of A in the space span$\{A^pX_0\}$,
- and eigenpairs with both large and small magnitudes may have large residual, and the size of the residual is determined by how closely eigenvalues of A_1 approximate the eigenvalue λ^{2p} *and* how large the image of u_i is in span$\{(A^pX_0\}$.
- If the space span$\{(AA^T)^pX_0\}$ can exclude small singular triplets or eigenpairs well, then they will have small images $\|\hat{u}_i\|$ and then (3.38) will be small regardless of σ_i.

Overall, there is a deep relationship apparent between the convergence of eigenvalues of $X_0^TA^pX_0$ or $X_0A^{2p}X_0$ for increasing p and the approximation properties given in both Theorems 7 and 8. Cases for which the power method converges quickly — when leading eigenvalues of AA^T are well-separated, especially compared to interior and trailing eigenvalues — are the cases for which the shrink-and-iterate approach will outperform the random projection method. When A is Hermitian and positive semi-definite, then the SVD and spectral decomposition coincide, and the above argument holds with AA^T replaced with A.

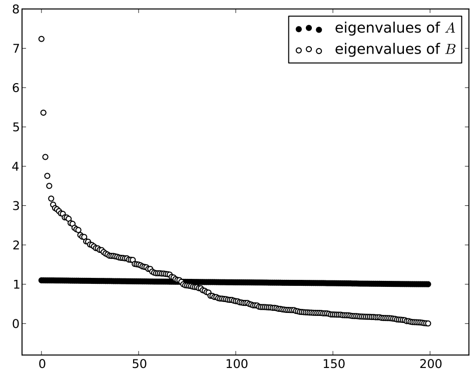

Figure 3.2: Spectra of A and B with uniformly and exponentially distributed eigenvalues.

3.3.4 Convergence example

To better illustrate the relationship between the gaps in the input spectra and the relative performance of shrink-and-iterate and random projections, we present an example. We consider two input matrices, A and B, both diagonal, and where diagonal elements of A are drawn from a uniform distribution $\mathscr{U}(1,1.1)$ and diagonal elements of B are drawn from an exponential distribution with scale parameter $\lambda_P = 1$. Both matrices share the same eigenspace; only the eigenvalues differ. The spectra of both matrices are shown in Figure 3.2. Based on the preceding discussion, we would expect B to have spectral properties amenable to the shrink-and-iterate approach producing a better low-rank approximation while random projections would produce a better low-rank approximation with A. This is indeed the case, and we examine the constituent sub-expressions in (3.35) to show how images of eigenvectors of A and B are changed by the power iteration process.

As the random projection method depends solely on the power parameter p in $A^p X_0$ to produce a subspace that contains the leading eigenvectors of A, we examine the norms of the images $\hat{u}_i$ of eigenvectors u_i of A and B. Since the spectrum of B has much larger gaps between the leading eigenvalues, we expect those eigenvectors associated with leading eigenvalues to have images with larger norms in span$\{BX_0\}$. Figure 3.3 shows the norms of images of eigenvectors in span$\{AX_0\}$ and span$\{BX_0\}$.

Norms of the images of eigenvectors in $\mathscr{K}_2(A, A^2 X_1)$ (with X_1 being composed of the first $b/2$

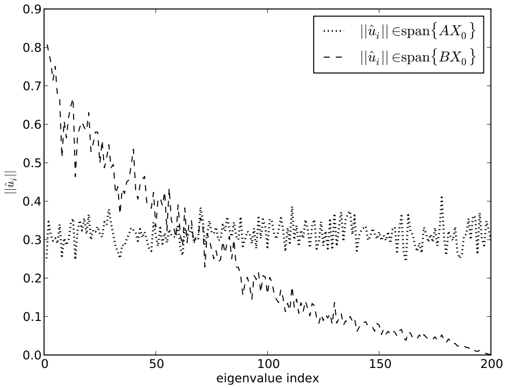

Figure 3.3: Norms of images of eigenvectors in span$\{AX_0\}$ and span$\{BX_0\}$. Eigenvectors with large eigenvalues have larger images in the space of B.

columns of X_1) are more complicated, but also depends largely on the "invariance" of the space $A^{2p+2}X_1R^{-1} - A^{2p}X_1R^{-1}A_1$. When eigenvalues are tightly clustered, then these matrices will have much smaller norms than if leading eigenvalues are well-separated. This may be deduced from the formula for the norm of the image of an eigenvector u_i in the space span$\{A^{p+1}X_1R^{-1} - A^pX_1R^{-1}A_1\}$ from Proposition 5. If λ_i has large magnitude, then the gap $\lambda_i^{p+1} - \lambda_i^p$ will be large compared to all other gaps and the difference (3.29) will be large due to large $\lambda_i^{p+1}u_i^TX_1R^{-1}$. If λ_i is small, then the left-hand term $\lambda_i^{p+1}u_i^TX_1R^{-1}$ of $\lambda_i^{p+1}u_i^TX_1R^{-1} - \lambda_i^pu_i^TX_1R^{-1}A_1$ will be small, but the right-hand term will be larger due to A_1 and the difference will still be large. Thus, leading eigenvalues and trailing eigenvalues both may converge in the space span$\{A^{p+1}X_1R^{-1} - A^pX_1R^{-1}A_1\}$, and the degree of convergence of trailing eigenvectors depends on how much larger the leading gaps $\lambda_i^{p+1}u_i^TX_1R^{-1} - \lambda_i^pu_i^TX_1R^{-1}A_1$ are compared to the trailing ones. Figure 3.4 shows the values of (3.29) for A and B. Figure 3.5 shows the values of (3.29) scaled by eigenvalues, to suggest the low-rank approximation norms of the two matrices. The values translate directly into the errors of low-rank approximation norms. When leading eigenvectors converge to the exclusion of trailing eigenvectors, as is the case with B, then 4 block Lanczos iterations will produce a larger low-rank approximation norm than random projections. If trailing eigenvectors converge at least as fast as the leading eigenvectors, as is the case with A, then random projections produces a better low-rank approximation.

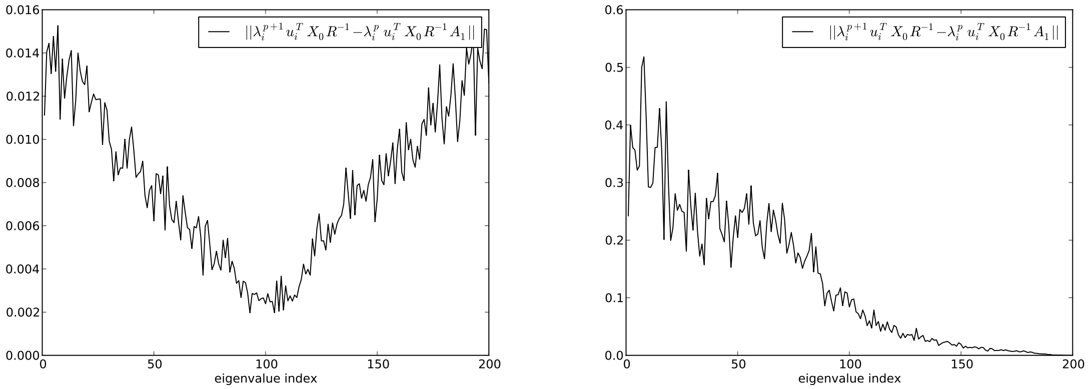

Figure 3.4: Norms of eigenvectors projected against $A^3X_1R^{-1} - A^2X_1R^{-1}A_1$ for A (left) and B (right). Leading eigenvalues with large gaps can overwhelm convergence of smaller eigenvalues for B, but the tighter clustering of eigenvalues in A leads to inclusion of more trailing spectral components.

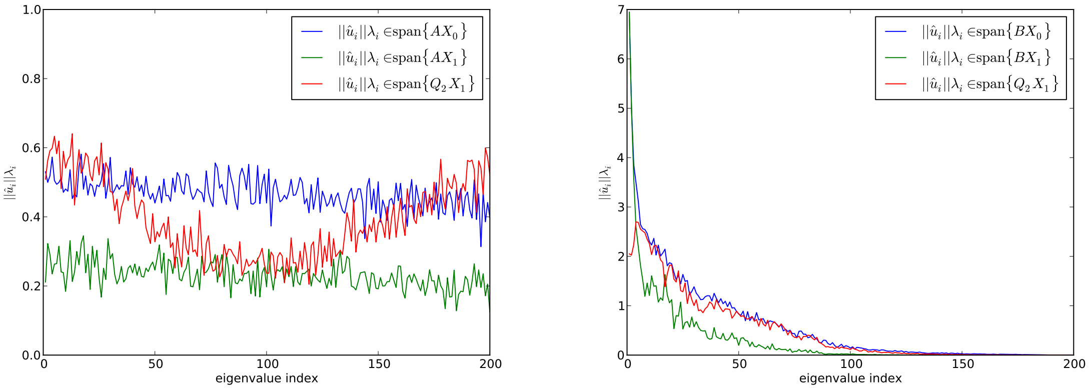

Figure 3.5: Norms of images of eigenvectors scaled by eigenvalues in span$\{AX_0\}$ or span$\{BX_0\}$ and span$\{Q\}$ for A (left) and B (right). B results in a better low-rank approximation norm due to convergence of only leading eigenvalues.

3.4 Stability Analysis

Thus far we have seen both the block Lanczos algorithm and the associated random projection methods, along with convergence analyses of the two. The asymptotic cost of a shrink-and-iterate approach compared to the random projection method allows for more iterations to be performed; however, this analysis neglects the costs of stabilization required to maintain orthogonality of the Lanczos basis as iteration progresses. The costs of stability along with the complications they present to the algorithmic simplicity are a motivation noted for the development of the random projection methods in [36]. The loss of orthogonality witnessed in the Lanczos algorithm is due to round-off error rather than an inherent instability in the Lanczos recurrence. We also note that there are two sources of loss of orthogonality in the Lanczos method: not only can converged eigenvectors re-enter the Lanczos basis, but it is also possible that even the adjacent Lanczos blocks Q_j and Q_{j+1} may not be orthogonal to machine precision due to round-off error. The extra stabilization steps added to Algorithm 4 are intended to address the latter of the two sources of error. Regarding the former source of error, intuitively, we would expect a short Krylov sequence to experience less problematic losses of orthogonality than a longer one. Therefore we may expect the shrink-and-iterate approach to encounter only a limited degree of orthogonality loss if only a few iterations are performed. In such cases, no stabilization would be required to generate a orthonormal Lanczos basis, and the classic block Lanczos method as presented in Algorithm 2 may be used as-is.

These intuitions are indeed proven out by theory. We begin by characterizing the loss of orthogonality over iterations in general, and then focus on the case for adjacent Q_j and Q_{j+1}. Bounds on the loss of orthogonality are presented in [73]. These error analyses are for the single-vector Lanczos algorithm, but may readily be generalized to the block Lanczos algorithm. The results are summarized in our presentation and proof of the following theorem, which is an generalization of the proof in [73] to the block Lanczos method.

Theorem 9. *Let A be a Hermitian matrix. Let Q be the Lanczos basis generated by $i+1$ steps of the block Lanczos routine with block size b in Algorithm 2. Let W be the strictly upper-triangular part of Q^TQ (W is zero on and below the diagonal) and $\mathbf{w}_j = [w_{(j-1)b+1}\ w_{(j-1)b+2}\ \ldots\ w_{(j-1)b+b}]$ the j^{th} block* *of columns of W with $\mathbf{w}_0 = 0$ and w_i represent columns of W. Then*

$$\|\mathbf{w}_{i+1}\| \leq (2\|A\| \max\{\|\mathbf{w}_i\|, \|\mathbf{w}_{i-1}\|\} + O(\epsilon\|A\|_2))\|B_{j+1}^{-1}\| \tag{3.39}$$

where $\|\cdot\|$ is the spectral norm and ϵ is machine epsilon.

Proof. From the Lanczos recurrence, we have

$$Q_{j+1}B_{j+1} = AQ_j - Q_jA_j - Q_{j-1}B_j + F_j \tag{3.40}$$

where F_j is a error term to account for round-off imprecision. Premultiplying both sides by the entire Lanczos basis Q^T gives

$$Q^TQ_{j+1}B_{j+1} = Q^TAQ_j - Q^TQ_jA_j - Q^TQ_{j-1}B_j + Q^TF_j. \tag{3.41}$$

Noting that the matrix form of the whole Lanczos recurrence $AQ = QT + Q_{j+1}B_{j+1}E_j^T + \bar{F}$, where E_j is the "element block" $E_j = [e_{(j-1)b+1}\ e_{(j-1)b+2}\ \ldots\ e_{(j-1)b+b}]$ and $\bar{F}$ is an error term for round-off, we may obtain

$$Q^TQ_{j+1} = (TQ^TQ_j + E_jB_{j+1}Q_{j+1}Q_j + \bar{F}^TQ_j - Q^TQ_jA_jQ^TQ_{j-1}B_j + Q^TF_j)B_{j+1}^{-1}. \tag{3.42}$$

by substitution and back-multiplication with B_{j+1}^{-1}. Let $\bar{\mathbf{w}}_{j+1}$ be the $j+1^{th}$ block of Q^TQ_{j+1}. Then by substitution we have

$$\bar{\mathbf{w}}_{j+1} = (T\mathbf{w}_j - A_j\mathbf{w}_j - \mathbf{w}_{j-1}B_j + E_jB_{j+1}Q_{j+1}Q_j + \bar{F}^TQ_j + Q^TF_j)B_{j+1}^{-1}. \tag{3.43}$$

We apply the matrix norm the inequalities $\|X+Y\| \leq \|X\| + \|Y\|$ and $\|XY\| \leq \|X\|\|Y\|$ to obtain

$$\|\bar{\mathbf{w}}_{j+1}\| \leq (\|T\|\|w_j\| + \|A_j\|\|\mathbf{w}_j\| + \|\mathbf{w}_{j-1}\|\|B_j\| + \|E_jB_{j+1}Q_{j+1}Q_j + \bar{F}^TQ_j + Q^TF_j\|)\|B_{j+1}^{-1}\|. \tag{3.44}$$

Finally, we have the claim from [58] that

$$\|F_j\| \leq \epsilon\|A\| \tag{3.45}$$

and which allows us to bound $O(\epsilon\|A\|) = \|E_jB_{j+1}Q_{j+1}Q_j + \bar{F}^TQ_j + Q^TF_j\|$. As $\|T\| + \|A_j\| \leq 2\|A\|$ and $\|B_j\| \leq \|A\|$, we have

$$\|\mathbf{w}_{j+1}\| \leq \|\bar{\mathbf{w}}_{i+1}\| \leq (2\|A\|\max\{\|\mathbf{w}_i\|, \|\mathbf{w}_{i-1}\|\} + O(\epsilon\|A\|))\|B_{j+1}^{-1}\| \tag{3.46}$$

□

The preceding round-off error analysis also lends insight into the mechanism through which Algorithm 4 reduces orthogonality loss in the Lanczos basis, and leads to the following proposition.

Proposition 6. *Let A be a Hermitian input matrix, and Q_j for $j = 1,\dots,k$ be the Lanczos blocks produced by k iterations of the block Lanczos method in Algorithm 2. Then the orthogonality between adjacent Lanczos blocks is given by*

$$Q_j^TQ_{j+1} = Q_jF_jB_{j+1}^{-1}. \tag{3.47}$$

This proposition is a result of observing that $Q_{j+1}^TQ_j = 0$ in infinitely-precise arithmetic, and premultiplying (3.40) by Q_j:

$$Q_j^TQ_{j+1}B_{j+1} = Q_j^TAQ_j - Q_j^TQ_jA_j - Q_j^TQ_{j-1}B_j + Q_jF_j \tag{3.48}$$

which simplifies to (3.47).

Refinement is intended to reduce $\|Q_j^TF_jB_{j+1}^{-1}\|_F$. The extra stabilization step does not prevent converged eigenvectors from entering the Lanczos basis, rather, it reduces the norm of $Q_j^TQ_{j+1}$ to no more than $\sqrt{\sum_{i=1}^{b}\epsilon^2}$.

3.4.1 Discussion

The resulting bounds on error are very similar to those obtained in [73] and reduce to those bounds when the block size is 1. For single-vector Lanczos, loss of orthogonality is governed by both the largest singular value of A and the smallness of B_{j+1}, which is a scalar value in the single-vector case. The error may grow faster in the block Lanczos routine than single-vector lanczos due to the effects of $\|B_{j+1}^{-1}\|$; it is sufficient that only the smallest singular value of B_{j+1} be small. The Frobenuis norm $\|B_{j+1}\|$ may be large, but if the linear independence of the next Lanczos block $AQ_j - Q_jA_j - Q_{j-1}B_j$ is nearly lost, then B_{j+1} will have a small infimal singular value. These are the cases for which deflation of the Krylov subspace is necessary. However, we may assume that in practice σ_{inf} of B_{j+1} is not negligibly small for small j. This is the analog of an assumption made in [73], and is almost always true in practice, especially when the start block for the Krylov subspace is random. We also remark that the loss of orthogonality for the special case of $\|\mathbf{w}_1\|$ is simply $O(\epsilon)$, as the Lanczos algorithm is equivalent to a full Gram-Schmidt algorithm for $i = 1$.

The immediate result of (3.39) and our assumption that $\|B_{j+1}\|$ is not too small is that we may always generate a Lanczos basis for $\mathscr{K}_i(A, X_0)$ without any appreciable loss of orthogonality for small i. As our compute time constraints provide only for $i \leq 4$, we can safely assume that in practice there will be limited loss of orthogonality beyond $O(\epsilon\|A\|)$. It is also evident that (3.39) does not distinguish between the classic block Lanczos method in Algorithm 2 and the block Lanczos method with refinement in Algorithm 4.

3.5 Conclusion

Random projections have been shown to produce good low-rank approximations without the loss of orthogonality problems that beset the Lanczos algorithms for generating Krylov subspaces for Hermitian inputs. Random projection methods may be interpreted as pseudo-block Krylov subspace methods with a large start block and no true iterations, and the loss of orthogonality in the Lanczos algorithm is proportional to the number of iterations. These two observations suggest that block Krylov subspaces with a large block and only a few iterations may produce low-rank approximations with error at least as small as the random projection method and with good stability without use of reorthogonalization. As compute time costs due to dense linear algebra operations grow quadratically with the number of dimensions used in the random projection method, shrinking the block size and increasing the number of iterations may produce a low-rank approximation with smaller error than the random projection method but still in smaller time. This advantage is most pronounced when the input matrix is sparse compared to its dimension. Additions to the block Lanczos method improve stability without the need for reorthogonalization, and hybrid random projection-Krylov subspace methods may capture the good eigenvalue approximation properties and stability of the random projection method and the better computational complexity of the block Lanczos method.

The error of a low-rank approximation generated by either the random projection method or the block Lanczos method depends on the spectrum of the input matrix A or AA^T. Naturally, the spectrum of the input matrix also determines which of the two approaches produces a better approximation. When the low-rank approximation problem requires eigenpairs or singular triplets with large magnitude, exclusion of the trailing eigenpairs or singular triples is important. However, we have shown that spectra for which the random projection method is able to successfully exclude trailing spectral components are also spectra for which the hybrid random projection-Krylov subspace method will also exclude trailing spectral components. This suggests that input matrices with large, well-separated leading eigenpairs or singular triplets and small, clustered trailing singular triplets are matrices for which the hybrid and shrink-and-iterate approaches will produce smaller low-rank approximation error than the random projection method.

The expressions which give the squared Frobenius norm of low-rank rank approximation error are admittedly not as elegant as the asymptotic bounds due to Kaniel, Paige, Saad or Halko et. al. Further simplification of the bounds may yield simpler expressions that may further clarify the role of the input spectrum. Nevertheless, the observations in the following chapter confirm the insight developed in our discussion comparing the low-rank approximation norms generated by the random projection method and the hybrid method. A useful future effort would further characterize the asymptotic nature of eigenvalue convergence in the hybrid method using bounds similar to Theorem 3; this would allow for comparison of worst-case eigenvalue convergence of the two methods.

Chapter 4

Applications of short block Krylov subspaces

Block Krylov subspaces and random projections have certain similarities. Random projection methods may be considered as quasi-Krylov methods. Though they exhibit good approximation and stability properties, we have shown that the complexity of the random projection method is quadratic in the number of dimensions returned. The relationship between block Krylov subspaces may affect a faster low-rank approximation; by shrinking the block size and performing a limited number of iterations, a block Krylov subspace can produce a low-rank approximation faster than the random projection method, or produce a low-rank approximation with smaller error in equivalent time. It is possible that the Krylov subspace approximation may have both smaller error and be faster to compute than the random projections approximation. Only performing a limited number of iterations may also practically eliminate the need for reorthogonalization in the Krylov subspace generation. The preceding chapter dealt with the theoretical comparison between random projections and the shrink-and-iterate Krylov subspace approach, but implementation details will dictate the realized stability, approximation errors and compute times. In particular, hardware features and programming aspects can have a substantial impact on compute times.

To show both the approximation performance, numerical stability and compute costs of random projections and small numbers of block Lanczos iterations, we performed numerical experiments. These experiments are intended to highlight the convergence differences between random projections and block Lanczos iteration for a variety of practical applications of low-rank matrix approximation. We present low-rank approximation applied to image compression for the facial recognition problem, applied to dimension reduction for information retrieval, applied to graph visualization and analysis, and to mechanical engineering problems. Overall, the results confirm the expectations developed in the previous chapter. When the leading singular values or eigenvalues define the optimal low-rank approximation are large and well-separated, we have observed that both the shrink-and-iterate and hybrid approaches produce low-rank approximations with larger norm than the random projection method. FLOP analyses also reflect the complexity of the two algorithms; the shrink-and-iterate approach with $s = 2$ require FLOPS comparable to the random projection method. The addition of the refinement step in Algorithm 4 adds extra overhead, but the hybrid approach was always faster than random projections. Fewer iterations translate to less loss of orthogonality in block Lanczos iteration, but loss of orthogonality was never larger than 10^{-9} for classic block Lanczos. The addition of a refinement step improved orthgonality by roughly an order of magnitude. Finally, the hybrid approach always had equivalent stability to random projection.

These problems were run on one of two computers, depending on memory requirements. Problems with smaller memory footprints were run on a Apple MacPro workstation running MacOS 10.6.8, with dual quad-core Intel Xeon processors running at 3 Ghz and 8 GB of memory. Software was compiled with GCC 4.2.1 and linked against the native Apple vecLib BLAS and LAPACK. SuperLU 4.3 was used for sparse factorization. Problems with larger memory footprints were run on a Newisys Linux server running RedHat enterprise 2.6, with 8 dual-core AMD Opteron processors running at 1 Ghz and 30 GB of memory. Software was compiled with GCC 4.4.6 and linked against boost uBlas 1.47. Boost uBLAS was only used for sparse matrix-vector products. Other linear algebra operations were linked against ATLAS version 3.8.3.

4.1 Experiments with the Yale face data

The Eigenfaces method [80] is a popular method that uses the truncated eigenvalue decomposition to both reduce dimensionality and clean data. Given a set of images, one may flatten each on into a vector a_i and assemble them into a matrix. After centering by subtraction of the average image $\mu = \sum_{i=1}^{m} a_i/m$, the resulting matrix $A - \mu = U\Sigma V^T$ may be diagonalized. The left singular vectors give the "eigenfaces," and the right singular vectors give the coordinates of each image in the reduced

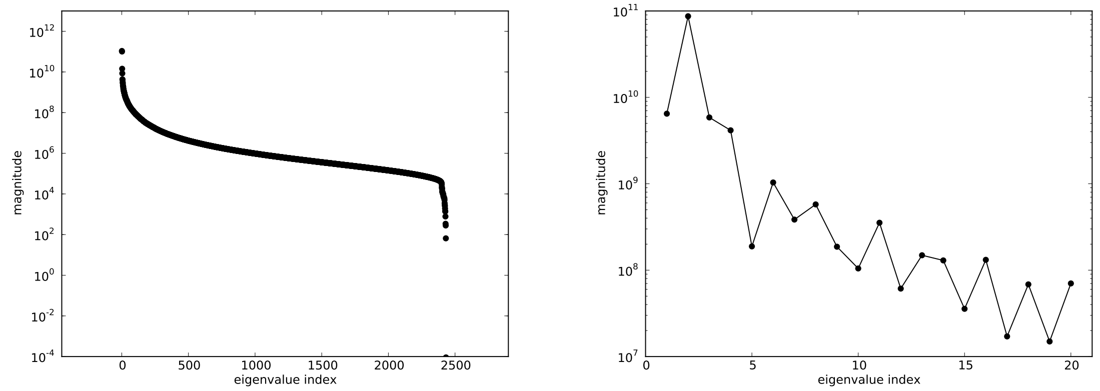

Figure 4.1: Leading 100 eigenvalues of A^TA for the Yale eigenface data (left) and eigenvalue gaps (right).

dimension space.

Chen and Saad studied the application of Krylov subspaces to the eigenface problem in [15] by approximating the SVD reduction. They noted that results of comparable quality may be realized with a Krylov subspace approximation of the SVD; this approximation yields substantial computational savings over the truncated SVD.

Data for these applications tends to be "almost low-rank," like many PCA applications; the spectrum of the centered matrix $A-\mu$ has a few leading singular values that are large, with singular values becoming more clustered towards the interior of the spectrum. We used the Yale extended face database B, with centered images [29]. Figure 4.1 shows the spectrum of $(A-\mu)(A-\mu)^T$. One particular advantage of this data set is that there are sufficiently few images that the entire spectrum of the matrix $B=(A-\mu)(A-\mu)^T$ may be computed directly.

The large local gaps cause the leading two eigenvalues of A to converge rapidly. Therefore, a point of diminishing returns on added dimensions is quickly reached, even with the truncated SVD. Since these two eigenvalues also converge rapidly in a Krylov subspace, the Krylov subspace approximations are close to the SVD approximation.

The substantial gaps between the leading two eigenvalues of the Yale face data and the rest of the spectrum implies good convergence of these two eigenvalues even without the need for power refinement in the random projection method or a large value of p in the hybrid algorithm. Table 4.1 shows the precomputed lower bounds for $\hat{A}^{(k)} \in \mathscr{K}_3(A,X_0)$ and $\hat{A}^{(k)} \in \mathscr{K}_4(A,X_0)$ along with

dimension	4	8	16	32	64
random projections	2.2275e+11	2.3864e+11	2.5255e+11	2.6538e+11	2.7589e+11
$\mathscr{K}_3(A,X_0)$ lower bounds	2.2069e+11	2.3911e+11	2.5335e+11	2.6542e+11	2.7591e+11
$\mathscr{K}_4(A,X_0)$ lower bounds	2.2903e+11	2.4387e+11	2.5649e+11	2.6849e+11	2.7815e+11

Table 4.1: Computed values for random projection and block Lanczos lower bounds for $\mathscr{K}_3(A,X_0)$ and $\mathscr{K}_4(A,X_0)$ for the Yale eigenface data.

the precomputed actual value for $\hat{A}^{(k)} \in \text{span}\{A\Omega\}$ for the random projection methods. The random projection methods will not produce worse low-rank approximations than $\mathscr{K}_3(A,X_0)$ for any dimension, but will never outperform $\mathscr{K}_4(A,X_0)$ for any dimension. The actual values produced are shown in Figure 4.2. Figure 4.2 also shows the low-rank approximation errors for the hybrid approach in Algorithm 5. There is not a substantial difference between $\hat{A}^{(k)} \in \mathscr{K}_3(A,X_0)$ and $\hat{A}^{(k)} \in \mathscr{K}_2(A,AX_0)$. There is a larger difference between $\hat{A}^{(k)}$ restricted to $\mathscr{K}_4(A,X_0)$ and $\hat{A}^{(k)}$ restricted to $\mathscr{K}_2(A,A^2X_0)$; however, both $\hat{A}^{(k)}$ restricted to $\mathscr{K}_4(A,X_0)$ and $\hat{A}^{(k)}$ restricted to $\mathscr{K}_2(A,A^2X_0)$ had larger Frobenius norms than $\hat{A}^{(k)}$ restricted to $\text{span}\{A\Omega\}$.

The large local gaps cause the leading two eigenvalues of A to converge rapidly; these eigenvalues also represent more than 70% of the squared Frobenius norm of A. Figure 4.2 also shows the low-rank errors for the hybrid method with $p = 2$, using both classic block Lanczos and block Lanczos with refinement. There is not a substantial difference between approximations in $\mathscr{K}_3(A,X_0)$ and $\mathscr{K}_2(A,AX_0)$. Both approximations in $\mathscr{K}_4(A,X_0)$ and $\mathscr{K}_2(A,A^2X_0)$ had larger Frobenius norms than the approximation in $\text{span}\{A\Omega\}$. Compared against single-vector Lanczos iteration, the random projection method produces low-rank approximations that are close in norm. Random projections produce approximations with smaller error for small dimensions.

The large leading eigenvalues also imply that B_2 from the block Lanczos algorithm has a large spectral norm. Therefore we expect to see a non-trivial value for the norm of the initial error term F_j in (3.40), and the block Lanczos iteration with refinement will be much more stable than classic block Lanczos even for the first iteration. Each iteration compounds the error accumulated in previous iterations as predicted by at least some multiple of $\|A\|$. We expect to see non-trivial stability improvements in block Lanczos with modified orthgonalization over classic block Lanczos for $\mathscr{K}_3(A,X_0)$ and $\mathscr{K}_4(A,X_0)$. Figure 4.3 shows the loss of orthogonality for random projections, shrink-and-iterate

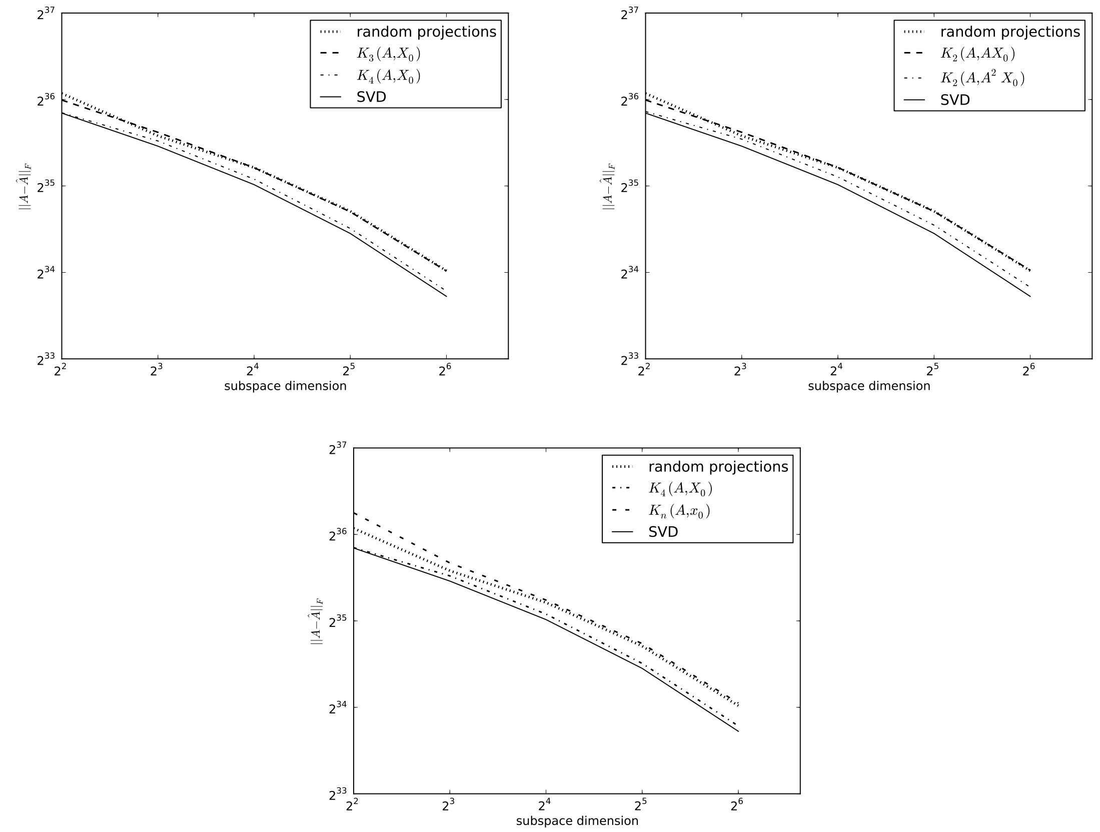

Figure 4.2: Low-rank approximation errors of Yale face data; random projections versus classic Lanczos (top left) and random projections versus the hybrid algorithm with $p = 2$ (top right). Squared Frobenius norm of random projections and block Lanczos iteration compared against single-vector Lanczos (bottom).

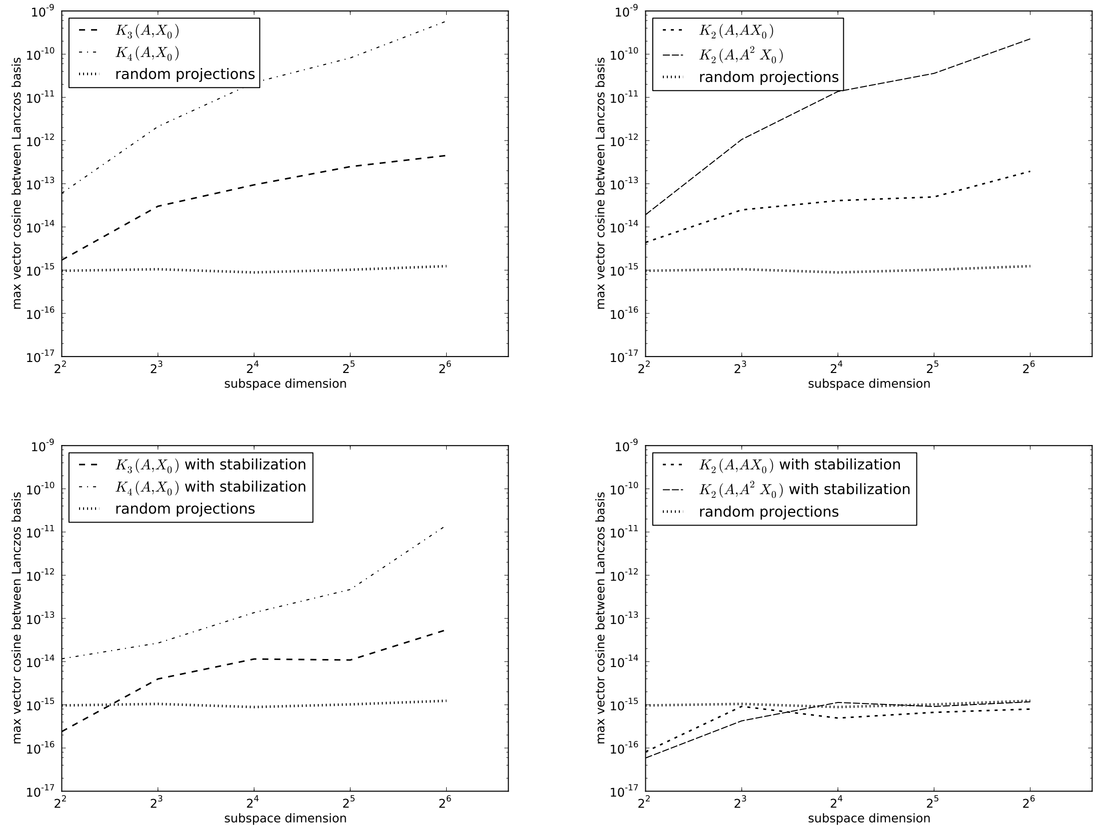

Figure 4.3: Maximum loss of orthogonality in projection basis using Yale face data; random projections versus Lanczos (left) and random projections versus the hybrid approach with refined block Lanczos (right). Top row figures are for classic block Lanczos, and bottom row figures are for block Lanczos with refined orthogonalization.

using classic block Lanczos and block Lanczos with modified orthogonalization for dimensions up to 64. We measure the maximum absolute value of the cosine of the angle between any two basis vectors; this is a more conservative measure than the Frobenius norm used in (3.39). We remark that classic block Lanczos with initial power iteration loses substantial orthgonality even for $\mathscr{K}_2(A, AX_0)$; however, the hybrid method using block Lanczos with modified orthogonalization is as stable as the random projection method when only 2 Lanczos iterations are performed. That the spectral norm $\|A\|$ is large implies that round-off error may render adjacent Lanczos blocks non-orthogonal, even in just the first iteration. This loss of orthogonality is due not to convergence of an eigenvector, but from roundoff error rendering Q_j not orthogonal to $AQ_j - Q_jA_j - Q_jB_{j-1}$. Figure 4.4 shows the Frobenius norm of $Q_j^TQ_{j+1}$ for the first 10 iterations of the block Lanczos method run on A^TA from the Yale

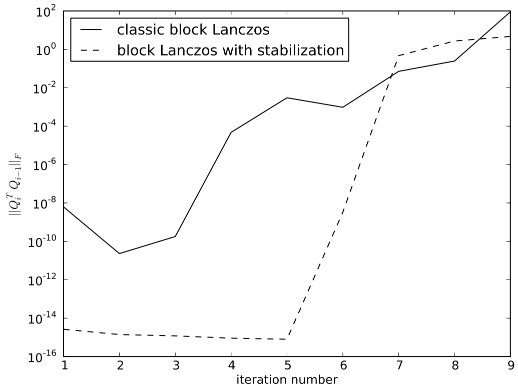

Figure 4.4: Frobenius norm of $Q_j^T Q_{j+1}$ for 10 iterations of block Lanczos on $A^T A$ from the Yale data using a block size of 10.

eigenface data using a block size of 10. It is evident that stabilization does indeed eliminate this error up to iteration 5, but cannot prevent loss of orthogonality due to convergence of eigenvectors at iteration 5.

4.2 Low-rank approximation in Information Retrieval

Information retrieval problems use a truncated SVD as part of Latent Semantic Indexing (LSI). Documents are represented as vectors in a "term space:" dimensions of the vector space correspond to unique terms in the corpus. If the vector $x \in (\mathbb{N} \cup \{0\})^n$ is a document vector, then the i^{th} dimension x_i is the number of times term i occurs in the document. The typical application of LSI involves assembling all document vectors into a matrix A and applying a scaling factor to all terms and documents to amplify the importance of infrequent terms and diminish the importance of frequent ones. The truncated SVD of A gives the "latent space" that separates noise from salient features of the corpus. As a result, low-rank approximation of the term-document matrix may *improve* query results by filtering noise. LSI is useful both for corpus analysis as well as dimension reduction for querying.

To study the approximation of LSI using block Krylov subspace projections and random projections, we study two problems: one using a small term-document matrix from the canonical NPL querying benchmark [78], and a term-document matrix obtained from the UCI Machine Learning

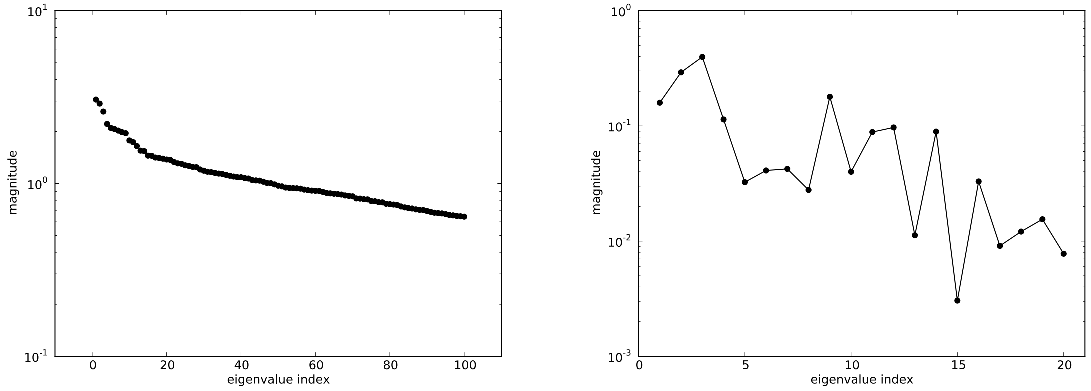

Figure 4.5: Leading 100 eigenvalues of A^TA for the Enron email corpus (left) and eigenvalue gaps (right).

dimension	4	8	16	32	64
random projections	6.2838e+00	1.1118e+01	2.0028e+01	3.5977e+01	6.2632e+01
$\mathscr{K}_3(A,X_0)$ lower bounds	6.6528e+00	1.1914e+01	2.2057e+01	3.7200e+01	6.3089e+01
$\mathscr{K}_3(A,X_0)$ lower bounds	8.3926e+00	1.5099e+01	2.6700e+01	4.4466e+01	7.3493e+01

Table 4.2: Computed values for random projection and block Lanczos lower bounds for $\mathscr{K}_3(A,X_0)$ and $\mathscr{K}_4(A,X_0)$ for Enron email corpus term-document matrix.

Repository's Bag of Words data set [26]. These are intended to show two distinct sides of low-rank approximation as applied to information retrieval problems: the UCI data is large and intended to show compute costs for low-rank matrix approximations costs. The NPL experiments are intended to show the consequences of low-rank approximation when matrix norm is only loosely coupled with application performance.

The NPL is widely cited benchmark for query processing systems; it consists of abstracts from physics technical reports and queries with relevance scores. The term-document matrix measures $7,491 \times 11,429$ with $2,228,087$ nonzero entries. The NPL collection also contains 100 queries with relevance scores. The Bag of Words data collection contains three distinct corpora; we used the Enron email corpus. The data does not include any queries for evaluation. The resulting matrix is $39,861 \times 28,099$ with $3,710,420$ non-zero elements. We rescaled the NPL data with term-frequency and inverse document frequency. The spectrum of the matrix A^TA for the Enron corpus is shown in Figure 4.5.

Figure 4.6 shows the norm of the low-rank approximations to A from the Enron corpus at various

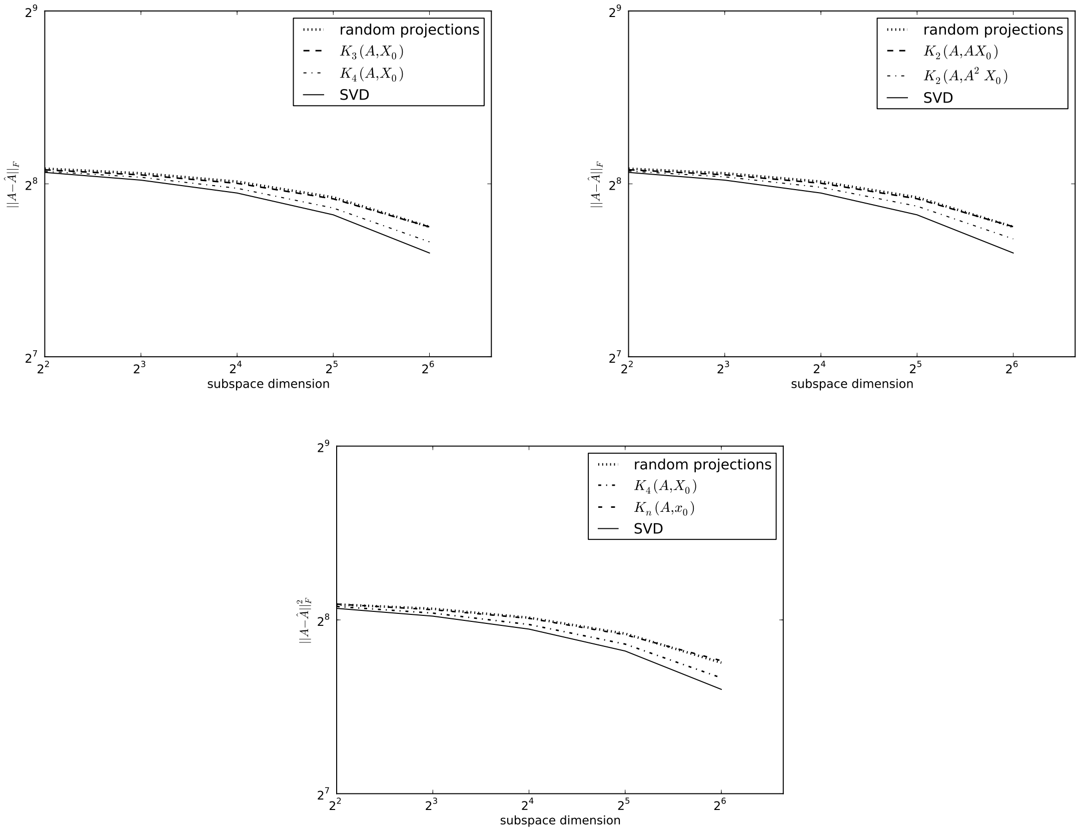

Figure 4.6: Low-rank approximation errors of Enron email corpus; random projections versus classic Lanczos (top left) and random projections versus the hybrid random projection-Krylov subspace method with $p = 2$ (top right). Squared Frobenius norm of random projections and block Lanczos iteration compared against single-vector Lanczos (bottom).

dimensions. Both classic block Lanczos and the hybrid method produce smaller low-rank approximation errors than random projections. The hybrid method does not approximate the SVD as well as in the Yale experiment, this is likely due to the smaller gaps in the leading part of the spectrum. Block Lanczos iteration is more stable with the Enron email corpus term-document matrix than for the Yale eigenface matrix. This better stability may be explained with two interpretations: the smaller spectral norm of the Enron email corpus implies the initial error term F_j in (3.40) will be smaller. Alternately, the smaller leading eigenvalue gaps imply slower convergence of the leading eigenvalues. A direct consequence of this slower convergence is that Lanczos iteration is more stable as shown in the previous chapter. Figure 4.7 contains the maximum cosine angle between any pair of Lanczos vectors, which measures the loss of orthogonality. Compared with the loss of orthogonality witnessed

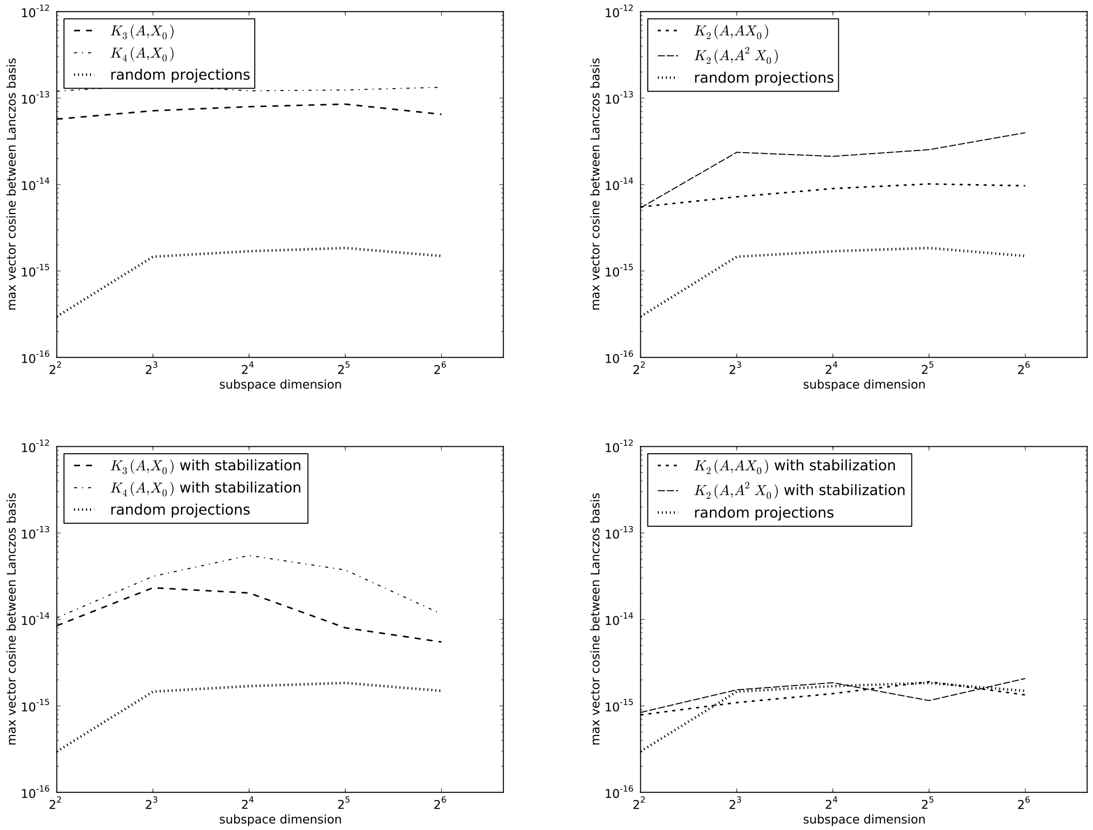

Figure 4.7: Maximum loss of orthogonality in projection basis for Enron email corpus; random projections versus Lanczos (left) and random projections versus the hybrid method using refined block Lanczos (right). Top row figures are for classic block Lanczos, and bottom row figures are for block Lanczos with modified orthogonalization.

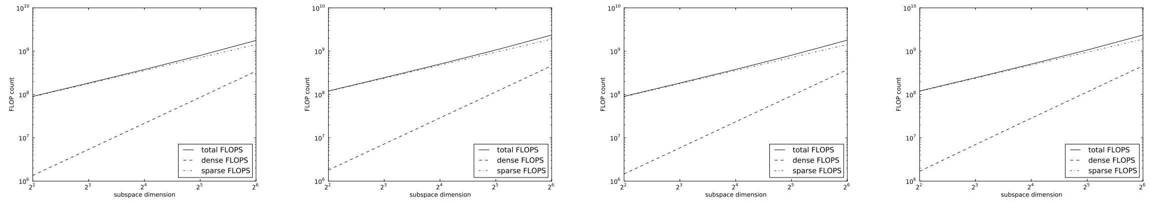

Figure 4.8: FLOP counts for producing low-rank approximations of the Enron email corpus. From left to right the plots show FLOPS for $\mathscr{K}_3(A,X_0)$, $\mathscr{K}_4(A,X_0)$, $\mathscr{K}_2(A,A^2X_0)$, and random projections. FLOPS for sparse matrix operations are considered separately from FLOPS for dense linear algebra operations.

in the Yale eigenface experiment in Figure 4.3, block Lanczos loses less orthogonality when applied to the Enron email corpus. The minimum vector angle cosine is two orders of magnitude smaller for the Enron email corpus than for the Yale eigenface data.

The Enron email corpus is a sufficiently large problem to produce meaningful FLOP comparisons. Based on the preceding theoretical discussion, shrinking the block size by 50% should be expected to reduce the compute time compared to random projections by a factor of four per iteration. Three classic Lanczos iterations should be expected to be 50% faster than random projections, and four classic Lanczos iterations should be expected to be equivalent to random projections. The hybrid method requires only 2 block Lanczos iterations, but will require an extra QR factorization between the initial power iteration and the block Lanczos step. Figure 4.8 shows the FLOP counts for producing a low-rank approximation of the Enron email corpus. As floating-point operations due to dense matrix operations and sparse matrix operations may exhibit different levels of performance, we distinguish between them in our FLOP analysis. All methods exhibit similar growth of FLOP requirements as the dimension of the computed subspace increases. A majority of the expense of all algorithms is sparse matrix operations; the total number of nonzeros in A^TA is 7,420,840, while the subspace dimension is 20,899. The ratio between the FLOP counts shows the expected speedup between the different methods. Figure 4.9 shows the ratios between the total FLOP counts. For all dimensions, the hybrid approach and 4 block Lanczos iterations with a 50% smaller block are roughly no more expensive than random projections, but 3 block Lanczos iterations are roughly 75% of the cost of random projections.

As the asymptotic discussion predicts, four iterations of block Lanczos are roughly equivalent in terms of compute time to the random projection method. Loss of orthogonality may render 4 block

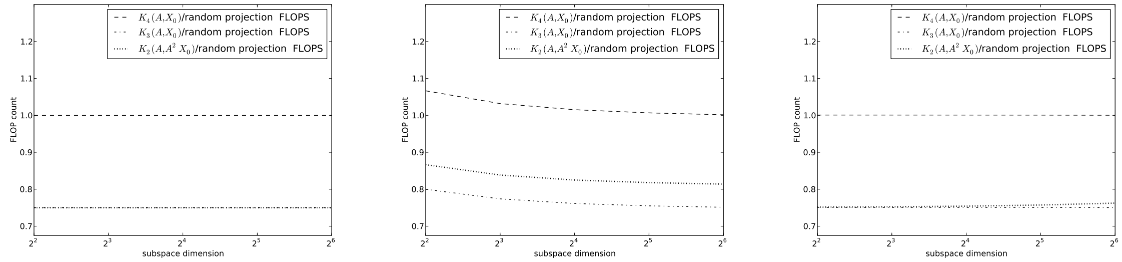

Figure 4.9: Ratios of sparse (left), dense (middle) and total (right) FLOPS for producing a low-rank approximation of the Enron email corpus.

Lanczos iterations not practical. The loss of orthogonality is improved by the addition of a refinement step, but at a cost. Adding a refinement step increases the dense FLOPS by one-third for 4 Lanczos iterations. However, for the Enron collection, as well as the Yale eigenface data, the difference in norm between the hybrid approach with $p = 2$ and four classic block Lanczos iterations is negligible. In these cases, using the hybrid method can manage the increased cost of the refinement step added to Lanczos.

Compute costs and approximation quality are just one facet of the performance of low-rank approximation as applied to information retrieval problems. As noted earlier, relevance scores of information retrieval are not entirely dependent on Frobenius norm optimization. Therefore, we may not necessarily expect that simply producing a better approximation of the SVD will yield improved precision or recall for a query processing system using our low-rank approximations. To illustrate how query processing performance changes with different SVD approximations, we applied low-rank approximation to the NPL collection.

For query ranking, we rescaled the term-document matrix as described previously, and ranked each document by its vector cosine against the query as projected through the latent document space. This reflects the use of the latent space not as a dimension reducer, but as a *query expander* [10]. Note that this method computes the query-document between document d cosine as

$$\cos(a_i, q) = \frac{(P^T a_i)^T P^T q}{\|a_i\| \|P^T q\|} = \frac{a_i (PP^T) q}{\|a_i\| \|P^T q\|} \tag{4.1}$$

for projection matrix P, which is equivalent to the document a_i *not being approximated at all*. Never-

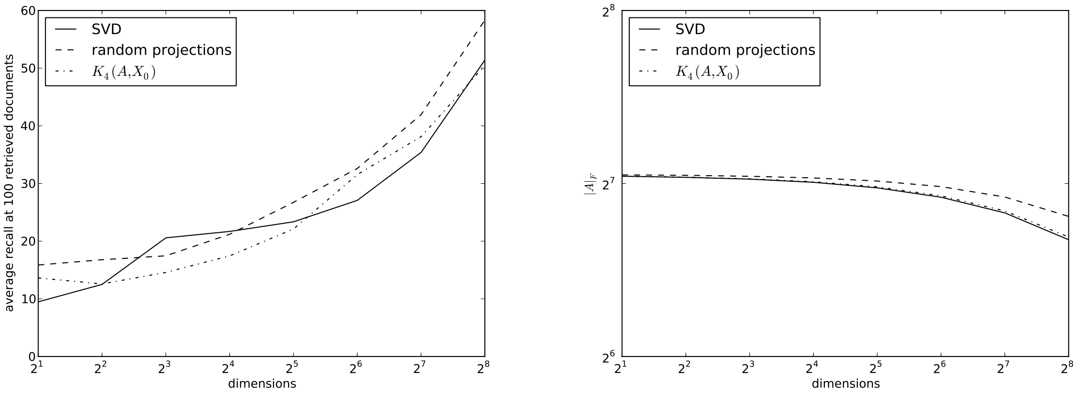

Figure 4.10: Document-level precision at 100 documents (left); this metric reflects the relevant percentage of the first 100 documents. The Frobenius norm error of low rank approximation of the NPL collection term-document matrix (right).

theless, this method produced the best document-level average results. For each query, we computed the document-level recall at 100 document retrieved; this metric measures the number of relevant documents returned in the first 100 search results. As the results varied based on the random start block, we averaged the results over 10 different random start blocks. The document-level recall is shown in Figure 4.10 with the norms of the low-rank term-document matrix approximations. Though the shrink-and-iterate approach using $\mathscr{K}_4(A,X_0)$ produced more relevant query results than the random projection method, the gap between relevance is smaller than the gap in Frobenius norm. Indeed, the Krylov subspace approximation is almost equivalent in matrix norm to the SVD approximation, but lags in query relevance. These observations are consistent with our expectations regarding the decoupled-ness of approximation norm and query performance.

4.3 Experiments with the Colorado road network Laplacian

Low-rank approximation is applicable to various spectral graph problems. One such problem is the computation of the commute-time embedding [70], which may be used for clustering [61, 87] or visualization. Spectral clustering [54], Laplacian embeddings [6], spectral layout algorithms [37, 43] and network analysis methods [53] are further applications that require eigenvectors of the graph Laplacian. The majority of spectral graph problems use the graph Laplacian matrix L and require the *smallest* eigenpairs of L rather than the largest. These eigenpairs may not be computed with

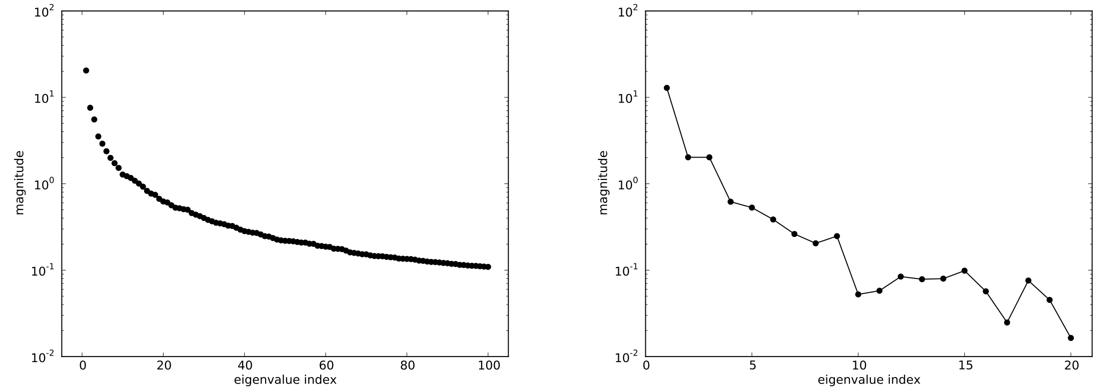

Figure 4.11: Leading 100 eigenvalues of L^+ for the Colorado road network pseudoinverse (left) and eigenvalue gaps (right).

Algorithm 3, as it approximates the leading eigenpairs of L. Krylov subspace methods do provide approximations of these trailing eigenpairs, but the approximations converge slowly when the trailing eigenpairs are tightly clustered. Unfortunately, the trailing eigenpairs of graph Laplacian matrices are typically tightly clustered for non-trivial, large problems. Nevertheless, for certain classes of graphs, shift-and-invert preconditioning is computationally inexpensive. Then the problem of finding the smallest eigenpairs of the graph Laplacian matrix L is replaced by finding the largest eigenpairs of the graph Laplacian psuedoinverse L^+. Moreover, the commute-time embedding is defined as $\Lambda^{1/2}U^T$. The low-rank approximation error $\mathrm{tr}(L^+) - \mathrm{tr}(\hat{L}^{(k)+}) = \|L^+ - \hat{L}^{(k)+}\|_*$ is indicative of squared Frobenius norm error of the approximation of the commute-time embedding.

To study the approximation of Laplacian pseudoinverses, we used the road graph network of Colorado, built from the U.S. Census Bureau's TIGER/Line data [1]. The directly resulting graph contained several connected components; we extracted the largest. The graph also contained numerous subgraphs isomorphic to the path graph, these were converted to a single edge. Two edge weights were available for each edge, we used distance rather than travel time. Figure 4.11 shows the spectrum of the Colorado road graph pseudoinverse. Leading eigenvalues are better-separated than for the Enron document collection. The better leading separation of eigenvalue implies rapid convergence of leading eigenvalues of L^+, even in the worst case predicted by the bounds in Theorem 6. Though the first few leading gaps are large, the eigenvalue magnitudes flatten out more rapidly than

dimension	4	8	16	32	64
random projections	3.1809e+01	4.0320e+01	4.9737e+01	5.8750e+01	6.7062e+01
$\mathscr{K}_3(A,X_0)$ lower bounds	3.0910e+01	4.1032e+01	4.9682e+01	5.8996e+01	6.7459e+01
$\mathscr{K}_4(A,X_0)$ lower bounds	3.5660e+01	4.3742e+01	5.3111e+01	6.2135e+01	7.0121e+01

Table 4.3: Computed values for random projection and block Lanczos lower bounds for $\mathscr{K}_3(A,X_0)$ and $\mathscr{K}_4(A,X_0)$ for the Colorado road network Laplacian pseudoinverse.

for the Yale eigenface data. As eigenvalues flatten out more quickly as for the Yale eigenface data, less convergence may be driven by simply increasing block size. We may expect that random projections to produce approximations that compare less favorably against 3 or 4 block Lanczos iterations than the comparison in Table 4.1. Table 4.3 shows the *a priori* bounds for the random projection method, and for 3 and 4 iterations of the block Lanczos method. The differences between 3 block Lanczos iterations and random projections are less tight than for the Yale eigenface data, and the bounds predict that the crossover point between random projections and $\mathscr{K}_3(A,X_0)$ happens sooner than for the Yale eigenface set. The smaller local gaps lead to a larger difference between the SVD approximation and the approximations from random projections or Krylov subspaces. Figure 4.12 shows the norms of the low-rank approximations generated by random projections and using short block Krylov subspace projections. Overall, the differences between the approximation methods considered in this chapter are similar between the Yale eigenface experiment and this experiment with the Colorado road network. Random projections produce approximations that are close in error to single-vector Krylov subspace projections, while halving the block size and performing three or four Lanczos iterations may produce better low-rank approximations.

The FLOP trends witnessed in the Colorado road network experiments are also similar to those witnessed in the Enron email corpus experiments. Classic block Lanczos with a halved block size is 50% faster than random projections for 3 iterations and roughly equivalent to random projections for 4 iterations. Adding a refinement step to the Lanczos algorithm also causes an increase in cost per iteration that may be offset by using the hybrid approach. Figure 4.13 shows the FLOP counts for producing low-rank approximations of the inverted Colorado road network graph Laplacian. All methods exhibit similar growth of FLOP requirements as the dimension of the computed subspace increases. However, the ratio between the FLOP counts shows the expected speedup between the

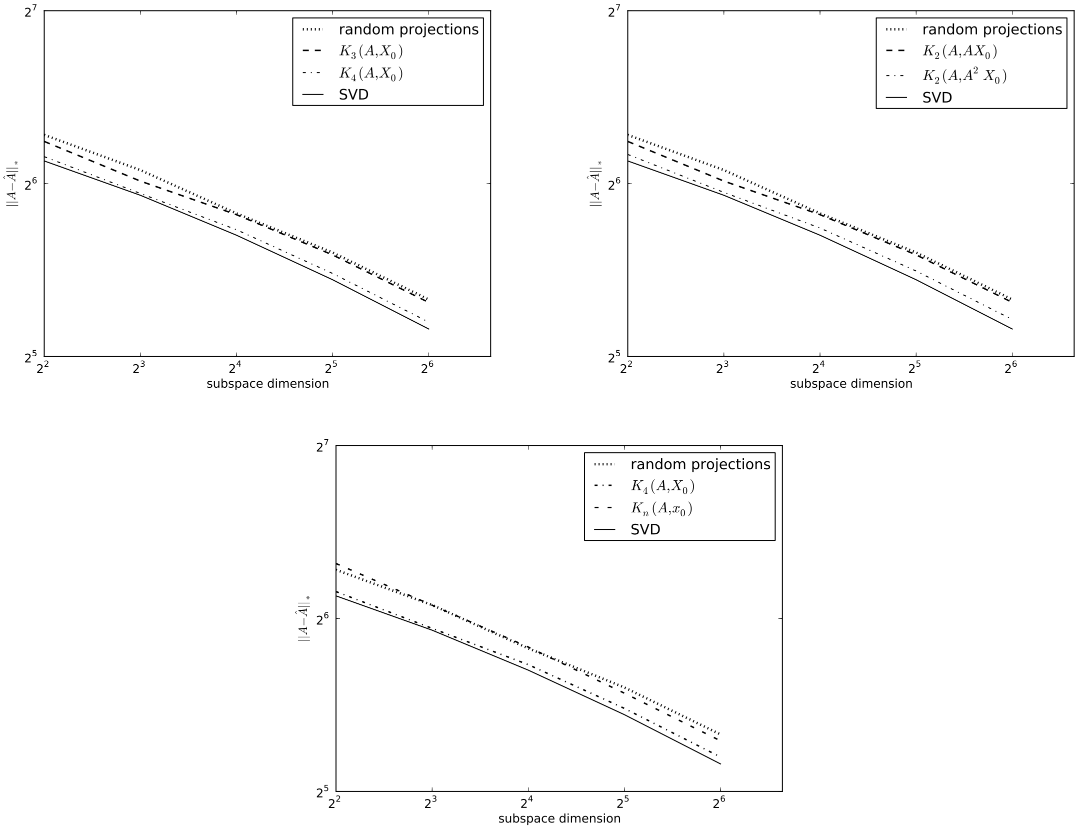

Figure 4.12: Low-rank approximation errors of Colorado road network Laplacian pseudoinverse; random projections versus classic Lanczos (top left) and random projections versus the hybrid approach with $p = 2$ (top right). Squared Frobenius norm of random projections and block Lanczos iteration compared against single-vector Lanczos (bottom).

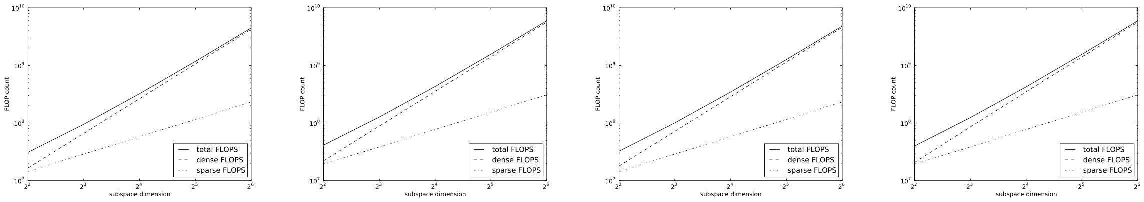

Figure 4.13: FLOP counts for producing low-rank approximations of the inverted Colorado graph network Laplacian. From left to right the plots show FLOPS for $\mathscr{K}_3(A,X_0)$, $\mathscr{K}_4(A,X_0)$, $\mathscr{K}_2(A,A^2X_0)$, and random projections. FLOPS for sparse matrix operations are considered separately from FLOPS for dense linear algebra operations.

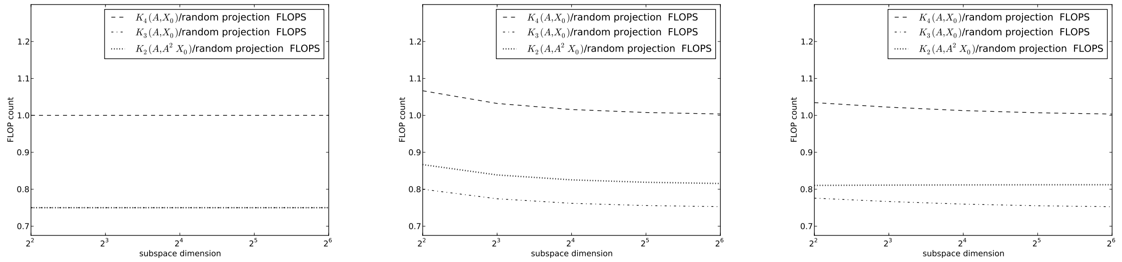

Figure 4.14: Ratios of sparse (left), dense (middle) and total (right) FLOPS for producing a low-rank approximation to the inverted Colorado road network Laplacian.

different methods. Figure 4.14 shows the ratios between the total FLOP counts. Although random projections holds advantages for small subspace dimensions, the shrink-and-iterate and hybrid approaches obtain FLOP parity with random projections at larger dimensions.

The loss of orthogonality for this matrix was also similar to the loss of orthogonality seen in the Yale eigenface experiment; this is shown in Figure 4.15. Without refinement, some non-negligible orthogonality was lost, even in the first iteration. This reduces the effectiveness of shortening the length of the Krylov sequence using p power iterations in the hybrid approach, even though the Krylov sequence length was reduced to 2. However, the addition of the refinement operations to Lanczos iteration improves stability by an order of magnitude for sequences of length greater than 2, and renders the two Lanczos iterations in the hybrid approach just as stable as random projections.

Depending on the application of the commute-time embedding, the error of the low-rank approximation may directly imply the quality of the low-rank approximation. For example, the commute-time distance $d(u,w)$ between two verices u and w in a graph is given in terms of the psuedoinverse of the Laplacian matrix:

$$d(u,w) = e_w^T L^+ e_u \tag{4.2}$$

where e_u is the element vector and assuming an appropriate mapping of dimensions of the vectorspace containing L to vertices in the graph. For the low-rank approximation of the commute-time operator, the nuclear norm approximation error implies the error of the approximated commute-time distance operator. We remark that there are similar distance operators in other domains, such as the Mahalanobis distance operator used in statistics and signal processing.

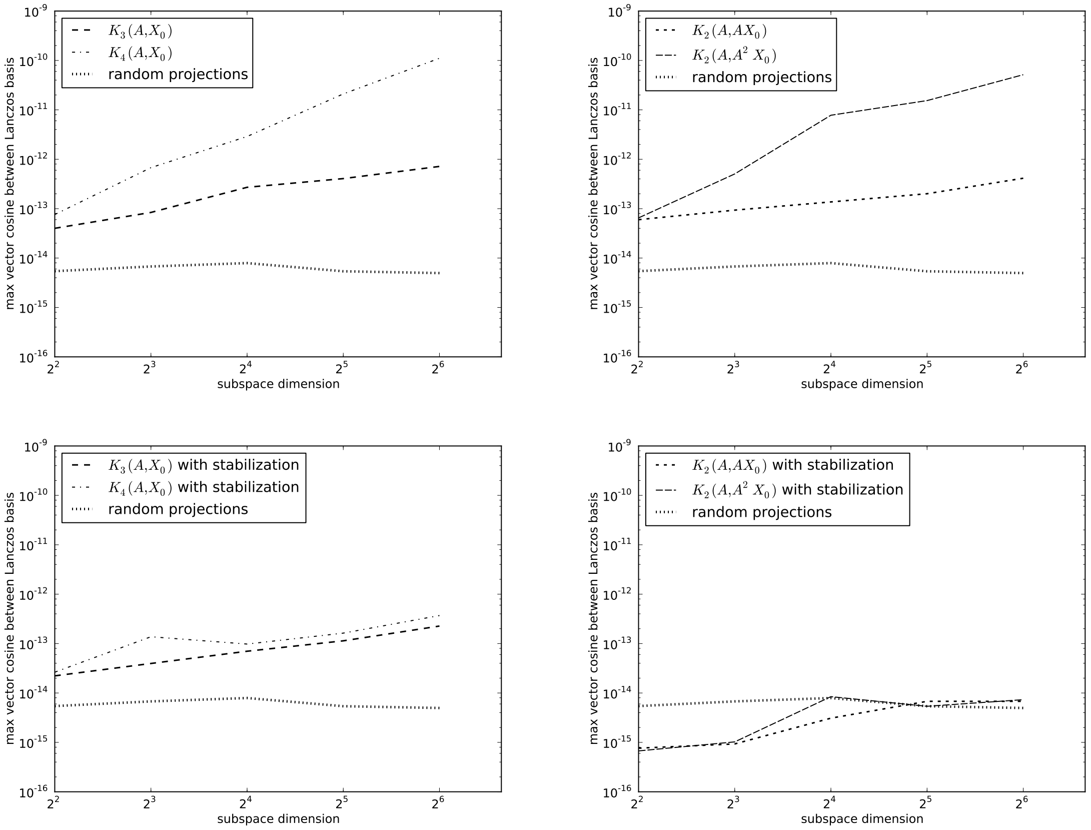

Figure 4.15: Maximum loss of orthogonality in projection basis for the Colorado road network Laplacian; random projections versus classic Lanczos (left) and random projections versus the hybrid random projections-Krylov subspace method using refined block Lanczos (right).

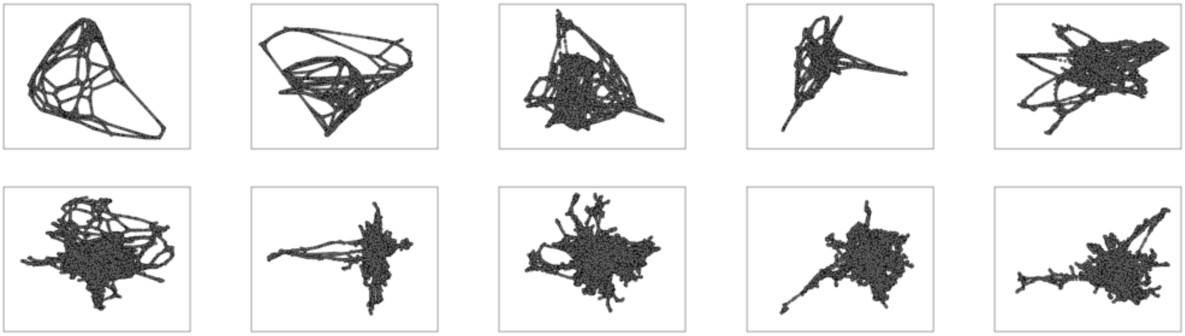

Figure 4.16: Grand tour of first 20 dimensions of commute-time embedding of low-rank approximation of the Colorado Road network graph in $\mathscr{K}_4(L^+, X_0)$.

For other applications of the commute-time embedding, the Frobenius or nuclear norm approximation error does not directly imply quality. We consider one such application of the commute-time embedding: visualization of a graph. For visualization, convergence of the eigenvectors is just as important as the approximation error. This distinction between approximation error and convergence of eigenvectors is important, as the low-rank approximation methods under consideration here may have substantially differing degrees of convergence of eigenvectors, even when low-rank approximations have equivalently low error in the Frobenius or nuclear norm.

To illustrate the difference in low-rank approximations which require eigenvector approximations, we show the grand tour of the first 20 dimensions commute-time embedding of the Colorado road network graph as approximated by random projections, and the Krylov subspace approximation in $\mathscr{K}_4(L^+, X_0)$. Figure 4.16 shows the commute-time embedding generated with 4 iterations of the block Lanczos method with a block size of 10, and 4.17 shows the commute time embedding as generated with the random projection method in Algorithm 3. These may be compared against the low-rank approximation produced with the truncated spectral decomposition in Figure 4.18. It is evident from the plots in Figures 4.16 and 4.17 that the Krylov subspace approximation produces a better visual approximation to the layout produced by the truncated spectral decomposition.

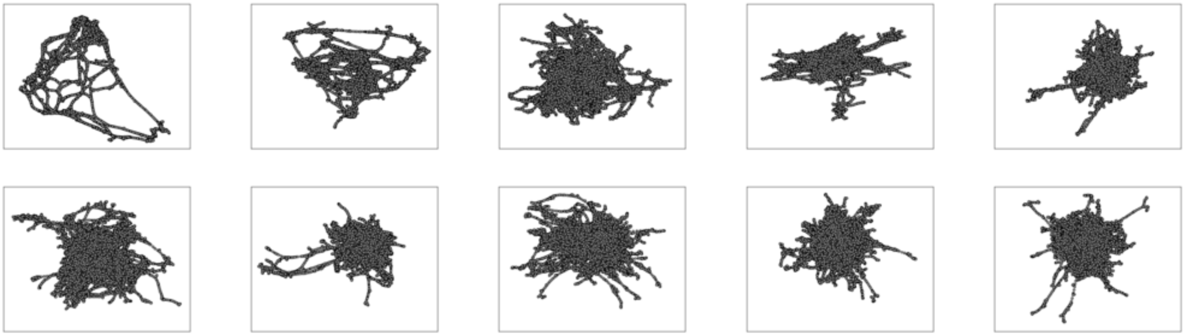

Figure 4.17: Grand tour of first 20 dimensions of commute-time embedding of low-rank approximation of the Colorado Road network graph generated with random projections.

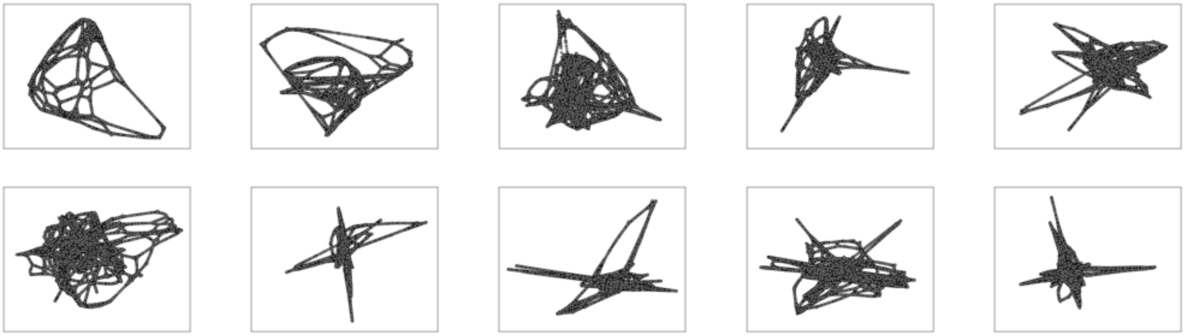

Figure 4.18: Grand tour of first 20 dimensions of commute-time embedding of low-rank approximation of the Colorado Road network graph using the truncated spectral decomposition.

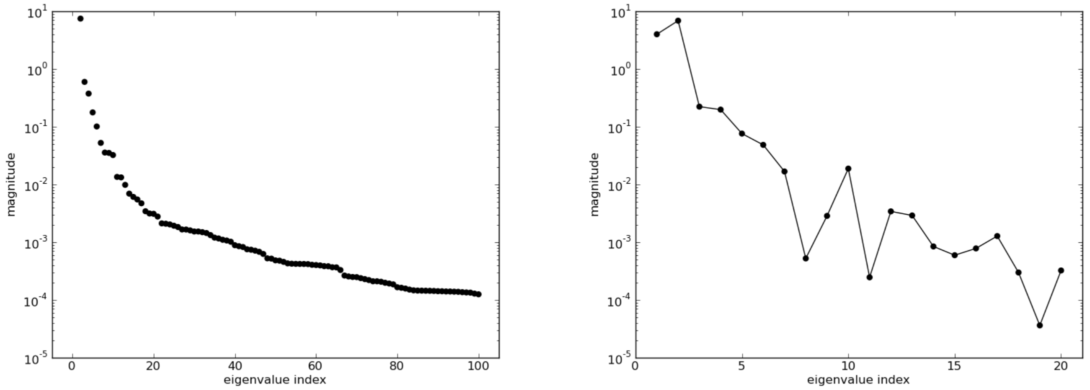

Figure 4.19: Leading 100 eigenvalues (left) and leading 20 gaps (right) of A^{-1} for the crankshaft stiffness matrix problem.

4.4 Stiffness matrix experiments

Thus far, we have considered matrices for which sparse matrix-vector products are all inexpensive relative to the dense linear algebra operations in the random projections or Lanczos algorithms. In many cases, the sparse matrix-vector products are the principal expense in either the Lanczos or random projection method. When these costs are principal, then a new set of tradeoffs are introduced, but the shrink-and-iterate and hybrid approaches still have advantages. These advantages are a direct result of the smaller number of sparse matrix-vector products necessary to produce a low-rank approximation. In fact, the shrink-and-iterate method with three block Lanczos iterations requires 75% fewer sparse matrix-vector products than the random projection method.

To study a case for which sparse matrix-vector products are not inexpensive compared to other dense linear algebra operations, we use the `bmwcra_1` matrix from the University of Florida's sparse matrix collection [18], which is a finite-elements stiffness matrix representing an automotive crankshaft. The matrix is $148,770 \times 148,770$ square and positive definite, with $10,644,002$ nonzero entries. The matrix does admit a sparse Cholesky factorization, but with considerable fill-in; the resulting Cholesky factors have 10,644,002 nonzero elements and consume nearly 4 GB of memory. The leading 100 eigenvalues of the inverted stiffness matrix and leading 20 eigenvalue gaps are shown in Figure 4.19.

Solution of the resulting triangular system is therefore not much less expensive than the matrix-

matrix products and QR factorizations used for low-rank approximation. We would expect this to diminish or eliminate the time advantage of the shrink-and-iterate approach. When the cost of the sparse matrix-vector products necessary to form AQ_i are considered, then the speedup gained by a reduction in block size is limited by the proportion of costs of sparse matrix operations to dense matrix operations, as per Ahmdahl's argument. The cost of each Lanczos iteration is dominated by the sparse matrix-vector products to form AQ_i rather than the operations to form A_i or $Q_{i+1}B_{i+1}$; little speedup can be expected from simply shrinking the block size.

More expensive sparse matrix-vector products compared to dense linear algebra operations products change the compute cost relationship between random projections and block Lanczos iteration; however, it does not render the shrink-and-iterate block Lanczos approach at a significant disadvantage. The compute advantage of Block Lanczos in these cases is that it generates the projection T of A in the working subspace $\mathscr{K}_i(A, X_0)$ as iteration proceeds, as opposed to the random projection method that first must generate a basis for span$\{A\Omega\}$ before generating the projection of A in span$\{A\Omega\}$. Indeed, the random projection method requires just as many sparse matrix-vector products as the shrink-and-iterate approach with 4 iterations, and the shrink-and-iterate approach with 3 iterations requires 75% of the sparse matrix-vector products. The hybrid method may not be advantageous from a compute-time perspective, as the principal driver of cost is sparse matrix-vector products.

The experiments performed with the crankshaft stiffness matrix coincide with the expectations developed with consideration of the sparse matrix-vector products performed in the random projections and block Lanczos methods. The more expensive sparse matrix-vector products do erode the compute advantages of initial power iteration in the hybrid approach, so 2 block Lanczos iterations with 2 initial power iterations are not much better than 4 block Lanczos iterations from a time perspective, though they are better from a loss-of-orthogonality perspective. Figure 4.20 shows the FLOP counts for generating low-rank approximations of the inverted crankshaft matrix for various dimensions, while Figure 4.21 shows the FLOP ratios between the shrink-and-iterate and random projection methods, and hybrid and random projection methods.

The eigenvalue distribution of this problem has large gaps at the extreme of the spectrum; these

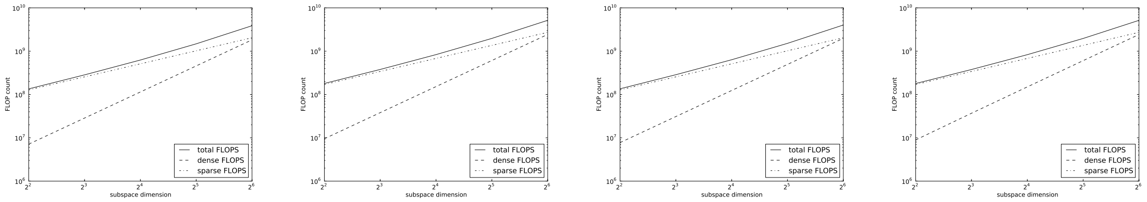

Figure 4.20: FLOP counts for producing low-rank approximations of the inverted crankshaft stiffness matrix. From left to right the plots show FLOPS for $\mathscr{K}_3(A,X_0)$, $\mathscr{K}_4(A,X_0)$, $\mathscr{K}_2(A,A^2X_0)$, and random projections. FLOPS for sparse matrix operations are considered separately from FLOPS for dense linear algebra operations.

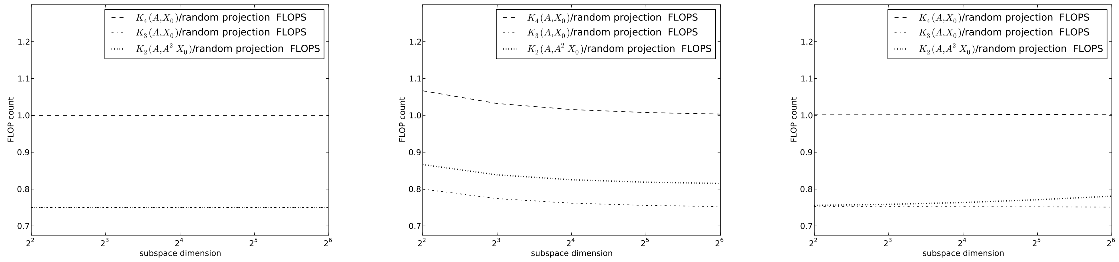

Figure 4.21: Ratios of sparse (left), dense (middle) and total (right) FLOPS for producing a low-rank approximation to the inverted crankshaft stiffness matrix.

gaps are proportionally even larger than for the Yale face database matrix. Like the Yale face data, this distribution places a large fraction of the norm in the leading two eigenvalues; therefore, a method may generate a good low-rank approximation simply by approximating them well. Figure 4.22 shows the approximation errors for low-rank approximations produced by random projections, the shrink-and-iterate method and the hybrid method.

4.5 Conclusion

Low-rank matrix approximation may be accomplished with a truncated SVD or spectral decomposition. However, computation of either decomposition is likely exorbitantly expensive for direct methods, or even for iterative methods when the number of spectral components is large enough. In these cases, approximations of eigenvectors and eigenvalues may produce a low-rank approximation with norm rivaling that of the truncated spectral decomposition, but at far reduced cost. One of several methods may be used to approximate the truncated spectral decomposition.

Recently there has been interest generated in power iteration-like random projection methods for

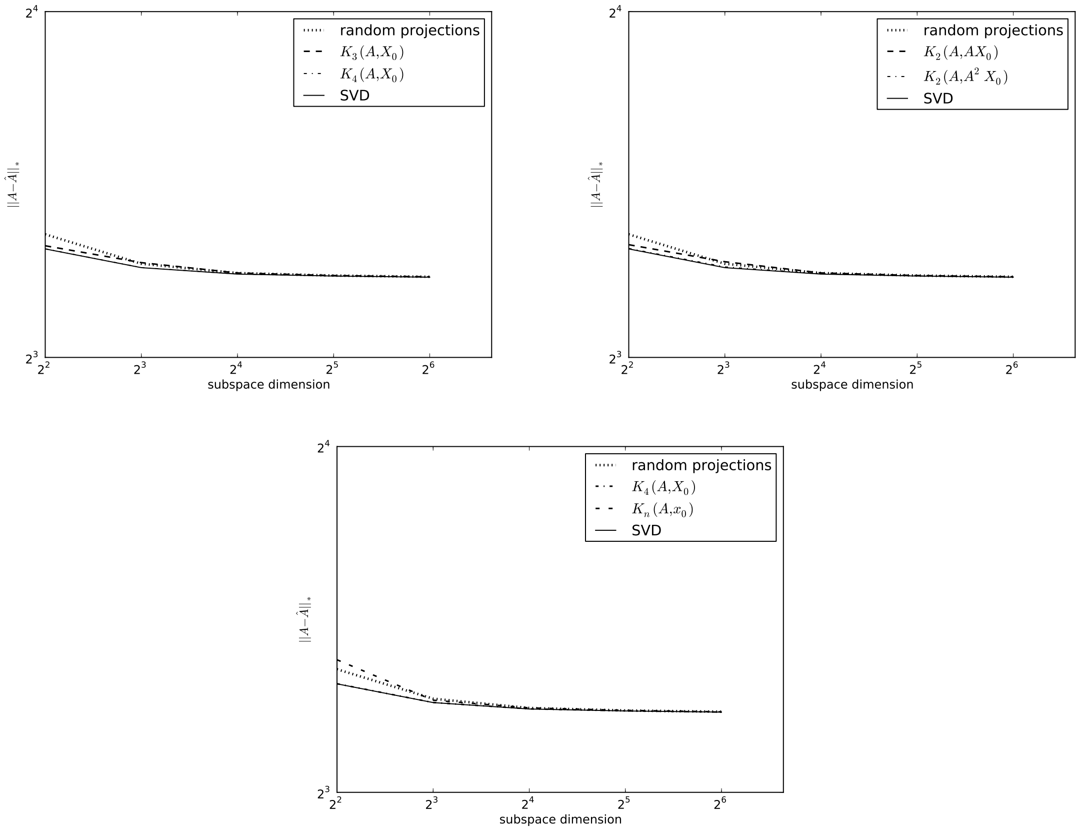

Figure 4.22: Low-rank approximation errors of inverted crankshaft stiffness matrix; random projections versus classic Lanczos (top left) and random projections versus the hybrid method with $p = 2$ (top right). Squared Frobenius norm of random projections and block Lanczos iteration compared against single-vector Lanczos (bottom).

Hermitian eigenproblems and compelling results have been reported. These methods have numerous favorable attributes; they have good eigenvalue approximation properties, they have moderate computational requirements, they lead to simple computational implementations, and they are immune to the loss of orthogonality that afflicts Krylov subspace methods. These approximated eigenvalues and eigenvectors may be used to generate low-rank matrix approximations at substantially reduced cost compared to a truncated SVD or spectral decomposition. Krylov subspace methods are closely related to power iteration, and the success of power iteration combined with random projections suggests hybrid approaches that combine the best elements of each approach.

Though random projections have a variety of favorable attributes, one disadvantage they have is their computational complexity. The computational complexity of random projection methods with power iteration scales quadratically with increased dimension computed. As one wishes to double the dimension of a low-rank matrix approximation, the computational cost of using random projection methods increases four-fold when dense linear algebra operations dominate. When sparse matrix-vector products are inexpensive, then meaningful gains may be realized by block size shrinkage. Alternately, when sparse matrix-vector products are expensive, then the number of sparse matrix-vector products used in random projections becomes a disadvantage compared to Krylov subspace methods. Random projection methods require $p+1$ block vector multiplications against A: p to form the subspace basis and one to project A into that basis. The block Lanczos method produces a projection of A in $\mathcal{K}_i(A, X_0)$ as it iterates; therefore, 3 block Lanczos iterations or one power iteration plus two Block Lanczos iterations will perform fewer sparse matrix-vector products than the random projection method. Use of a block Krylov subspace rather than a random projection space can lead to compute time savings whether sparse matrix-vector products are inexpensive or expensive.

The random projection methods also suggest new ways to cope with the loss of orthogonality experienced in Krylov subspace routines. Low-rank matrix approximation applications of Krylov subspaces do not necessarily require convergence eigenvalues. Loss of orthogonality is directly related to the length of the Krylov sequence, so when convergence of eigenvalues is not important, one may reduce the loss of orthogonality by increasing the block size and reducing the number of iterations. The number of iterations may also be reduced with a hybrid approach that initializes the

Krylov subspace with a start block generated with power iteration. This reduces loss of orthgonality by shortening the length of the Krylov sequence. Further stability may be realized by adding a refinement step to the block Lanczos process. Use of both initial power iteration to minimize the length of the Krylov sequence and a refinement step to reduce round-off error result in the block Lanczos process being as stable as random projections for low-rank matrix approximation. This added stability totally eliminates the need for restarts or selective reorthogonalization to stabilize the block Lanczos method, and thereby render it far less complicated than current practical implementations. As the random projection method uses a harmonic Ritz basis, an interesting future effort would be to compare the performance of the hybrid method against explicit restarts using harmonic Ritz vectors [41] from short Lanczos recurrences.

Several factors may influence compute times, but we focus on a theoretical FLOP count analysis, with a distinction between sparse and dense floating point operations. With respect to dense floating point operations, both the shrink-and-iterate and hybrid approaches will never require more FLOPS than random projections; for 3 iterations, they will require roughly 75% of the dense FLOPS required for random projections. The case is identical for sparse FLOPS, as both the hybrid and shrink can construct a low-rank approximation of A restricted to the Krylov subspace and a basis for the subspace as iteration proceeds. By way of comparison, one feature that hampers the performance of random projection methods is that they must perform at least $p+1$ sparse matrix-vector products to generate an approximation restricted to span$\{A^p\Omega\}$, one to generate an orthonormal basis for the subspace, and another to project the input matrix into the subspace. This feature is a limitation when sparse matrix-vector products are expensive, as is the case for the inverted crankshaft matrix. Yet in these cases, the block Krylov methods developed here also have somewhat of a disadvantage as well, as they must perform at least $1.5k$ sparse matrix-vector products. If the cost of a sparse matrix-vector product is sufficiently large, there will be no difference between the shrink-and-iterate method with 4 iterations and random projections. Likewise, there will not be a substantial difference between the hybrid method operating in $\mathscr{K}_2(A, A^2X_0)$ and the random projection method. Therefore, the biggest advantage for shrink-and-iterate or hybrid methods is when the input matrix has well-separated leading eigenvalues and is sufficiently sparse as to allow for fast sparse matrix-vector products. In these cases, the hybrid approaches allow for both better convergence and possibly faster compute times.

Chapter 5

GrABL: A Greedy Adaptive Block Lanczos method

In the preceding chapters, we have demonstrated short block Krylov subspace methods for low-rank approximation problems; both the “shrink-and-iterate” and hybrid Krylov subspace-random projection methods offer potentially better approximations and smaller computational costs than random projection methods, coupled with reduced loss of orthogonality. The computational time advantages these methods hold over random projections is a direct result of smaller block size; as we mentioned in Chapter 3, shrinking the block size by a factor of s leads to a s^2 speedup per iteration. The “shrink-and-iterate” and hybrid short block Krylov subspace approaches use a fixed shrinkage factor of $s = 2$, and perform 3 or 4 iterations. They therefore have a fixed cost advantage over random projections.

A block Lanczos process that adaptively grows and shrinks block size can leverage large blocks to drive convergence of eigenvalues through exploitation of local gaps, and then deflate the block size to reduce the cost per iteration. An adaptive block method could be better able to manage computational costs than the short block Krylov subspace methods developed in the previous chapters, and obtain an asymptotic advantage over random projections. Additionally, use of a smaller initial block for stationary power iteration, as is used in the hybrid block Krylov-random projections method in Algorithm 5 will result in reduced computational costs due to sparse matrix-vector products. This advantage will be marked in the cases for which sparse matrix-vector products are dominant compute costs, such as the stiffness matrix experiments presented in Section 4.4. Nevertheless, larger degrees of block size shrinkage will necessitate more iterations, which compromises the loss of orthogonality benefits developed for the short block Krylov subspace methods in the previous chapter, which may be addressed with one of the standard partial or selective orthogonalization strategies [57, 74].

Adaptively blocked Krylov subspace methods have been developed previously [5,86]. Rather than using a fixed block size, the algorithm detects clustering of Ritz values as it iterates, and inflates the block size if it determines that the number of clustered Ritz values exceeds the current block size. Notably, the existing adaptive block Krylov subspace methods are intended for accurate computation of eigenvalues. Inflation of the block size is performed to allow for accurate computation of eigenvalues that are tightly clustered. These algorithms do not inflate to accelerate minimization of aggregate singular value error, but are intended to run until convergence of a set of eigenpairs of interest. For the relaxed generic reduction problem, it is tolerable if eigenvalues and eigenvectors have not converged completely, but use of a minimal Krylov subspace introduces a different set of trade-offs not present in the eigenvalue problem. Most importantly, the block size restricts the number of iterations possible, which in turn restricts the convergence of eigenvalues in the Krylov subspace. This restriction changes the approach to inflation and deflation, and dictates when and how they should occur.

We develop GrABL, an adaptive-block size Lanczos algorithm specifically for low-rank approximation problems. GrABL differs from the existing adaptive block Lanczos methods via its inflation method and deflation criterion. Its inflation method is informed by the changes in eigenvalue convergence rates due to increases in block size, and assumes eigenvalue distributions typical of generic low-rank approximation problems and the random projection method. Rather than increasing block sizes when clustered eigenvalues are encountered, it increases block size only when the added block columns increase convergence of the well-separated leading eigenvalues. Rather than deflate when linear independence of any Lanczos block is lost, it deflates immediately after detection of an optimal block size to obtain the best-possible compute advantage for successive iterations.

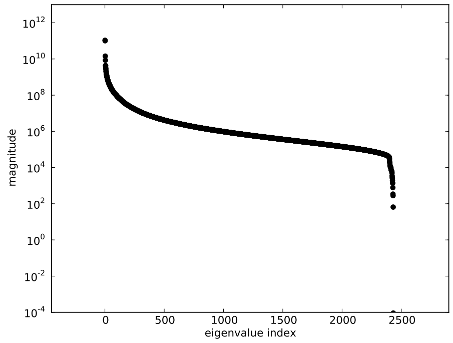

Figure 5.1: Eigenvalues of AA^T for A as extended Yale eigenface database. Images were flattened to vectors.

5.1 Typical spectra of generic dimension reduction problems

Convergence of eigenvalues in Krylov subspaces depends on the distribution of the spectrum; the typical spectrum of generic dimension reduction problems happens to have properties that enable fast convergence of exactly those eigenvalues and eigenvectors necessary for a good dimension reduction. Typically, in a generic dimension reduction problem such as PCA or LSI, it is assumed that there exists a latent, low-dimensional problem that is embedded in a high-dimensional space. The high-dimensional space also has noise added, albeit at a far lower magnitude than the data. The leading eigenvectors are assumed to be a basis for the latent space, and the remaining dimensions are assumed to be noise. The magnitudes of the latent space vectors and the noise-space vectors will reflect the signal to noise ratio of the problem. Convergence of eigenvalues in a Krylov subspace are driven by, among other things, eigenvalue gaps. Therefore, the latent space vectors will converge quickly in a Krylov subspace. In the vast majority of data reduction applications, there is not an abrupt step between the latent space eigenvalues and the noise eigenvalues. Rather, eigenvalue gaps decrease rapidly for the first few gaps, and then decrease more slowly after that. Figure 5.1 shows the spectrum of an eigenface data set, and is characteristic of generic dimension reduction problems. Spectra such as shown in Figure 5.1 are essential to the remainder of this chapter. It is assumed that only a few leading eigenvalues are well-separated from the rest, with tight clustering encountered early in the spectrum. These properties both give good convergence to block methods, but also render them computational over-kill when there are fewer well-separated eigenvalues than number of columns in the start block.

5.2 Classic block Lanczos

If *a priori* knowledge of the spectrum is available to determine an optimal block size, one may simply apply the classic block Lanczos algorithm to produce a minimal Krylov subspace. The classic block Lanczos algorithm is presented in Algorithm 2. We remark that the classic block Lanczos algorithm is elegant in its most abstract instantiation. In exact arithmetic, the Lanczos algorithm guarantees orthogonality between the Lanczos blocks Q_i, but practical implementations include some measures to recover orthogonality lost from finite-precision operations. Provided that the matrix A is sparse, the costs of each Lanczos iteration are dominated by the dense matrix-matrix operations at lines 4, 5 and 7, which require $O(nk^2)$ operations.

5.3 ABLE: adaptively blocked Lanczos for the eigenproblem

It is rare that sufficient knowledge of the spectrum may be known before the fact or is easily deducible from the matrix. This motivates development of block Lanczos methods that can adapt block size as iteration progresses without discarding any previously computed information about the spectrum of A. ABLE [4, 5] is one such algorithm for the non-Hermitian eigenproblem. ABLE uses block size inflation to recover from breakdown and extract clustered eigenvalues. The ABLE algorithm simplified for the Hermitian case is presented in Algorithm 6. It is notable that the problems of breakdown and tightly clustered eigenvalues are not as severe in the Hermitian case. ABLE is intended to solve non-Hermitian eigenproblems and cope with the particular problems they present. However, if one were simply to use ABLE for a low-rank approximation, it is not clear what values should be used for η and θ_k. To alleviate the requirement for knowledge of the spectrum of A to adapt block size, we note that one may monitor convergence of Ritz values in the Krylov subspace to infer the size of the local gaps. The cost of ABLE depends on the block size history, and is $\sum_{i=1}^{r} 4b_i^2 n + 2b_i n_{\mathrm{nnz}}$, where ABLE takes r iterations, and b_i is the block size at iteration i. Of the elements of this expression, $2kn_{\mathrm{nnz}}$ FLOPS are due to sparse operations, with the complement due to dense operations.

Algorithm 6 ABLE: Adaptive block Lanczos for the eigenproblem

```
Require: a priori chosen start vector $Q_1$
$R \leftarrow AQ_1$
for $j = 1 \rightarrow k$ do
    $A_j \leftarrow Q_j^T R$
    $R \leftarrow R - Q_j A_j$
    QR factorize $R = Q_{j+1} B_{j+1}$
    spectral decompose $T_j = U \Theta U^T$
    detect clustering in $\Theta$, if $|\{\theta_j \mid |\theta_j - \theta_k| < \eta\}|$ for a given $\theta_k$ and tolerance $\eta$ is larger than the block size, inflate $Q_j$ with $\hat{Q} \perp Q_i$ for all $i \leq j$
    $R \leftarrow AQ_{j+1}$
    $R \leftarrow R - Q_j B_{j+1}^T$
end for
return $Q, T$
```

5.3.1 Convergence of eigenvalues in block Krylov subspaces

Convergence of eigenvalue approximations for the projection of A into $\mathcal{K}_i(A, X_0)$ is driven both by block size r and Krylov sequence length i; therefore, convergence may be achieved either by increasing i or r. Computational costs are also governed by i and r; the cost to generate a minimal Krylov subspace of dimension k is asymptotic with $kr^2/r = kr$, disregarding the costs of stabilization and the dimension of A. As k is fixed, doubling the block size will double the compute cost to generate the Krylov subspace. The random projection method may have good results, but is also the most expensive method. Naturally, we would like to have eigenvalues converge as quickly as possible given a fixed subspace dimension, subject to minimal computational costs. This introduces a new set of trade-offs between single-vector and block Krylov subspaces. Eigenvalues may converge faster in a block Krylov subspace if spectral properties are favorable, but the larger block size introduces more computational costs compared to a smaller block size coupled with a longer Krylov sequence. Furthermore, the minimal Krylov subspace restriction limits the number of iterations available to a block method more than it limits the number of iterations available to a single vector method. Fewer iterations restrict the convergence of eigenvalues in the subspace, which requires that the improvement in convergence due to local gaps outweigh the loss in convergence due to restrictions on the number of iterations.

The use of a minimal or near-minimal Krylov subspace is a central feature of this effort; limits on convergence determine those spectra for which a block Krylov subspace is advantageous versus a single-vector Krylov subspace. The cases for which a block method is most advantageous are somewhat counter-intuitive for minimal Krylov subspaces. Rather than being most appropriate when eigenvalues are tightly clustered, they are advantageous when they are well-separated. If the leading part of the spectrum of A is sufficiently flat up to desired minimal subspace size k, then block methods cannot achieve tight error bounds without iteration. Conversely, if the leading eigenvalues are all mutually well-separated, then good error bounds are possible through block size alone. Before discussing the convergence of eigenvalues in Krylov subspaces, we recall the error bounds for block Krylov subspace projections due to Underwood [81] in Theorem 3.

We remark that the Underwood theorem may be stated with the substitution of $T^2_{i-1}(\hat{\gamma}_i)$ with

$$T^2_{i-1}(\hat{\gamma}_i) = T^2_{i-1}\left(\frac{1+\gamma_j}{1-\gamma_j}\right) \tag{5.1}$$

with

$$\hat{\gamma}_j = 1 + 2\frac{\lambda_j - \lambda_{r+1}}{\lambda_{r+1} - \lambda_{\inf}} \tag{5.2}$$

though these two formulations are equivalent.

These bounds show what properties of the spectrum of A and the Krylov subspace shrink errors of eigenvalue approximations. Making the denominator in (2.3) large will drive the error $\lambda_j - \lambda_j^{(n)}$ to zero; this is accomplished by either increasing the order of the Chebyshev polynomial or by increasing $\hat{\gamma}_j$. These increases are accomplished by increasing the Krylov sequence length or increasing the block size, respectively. When the leading s eigenvalues are all mutually well-separated, then dramatic improvements may be realized by setting the block size $r = s$. For leading eigenvalues, rapid convergence will also result from lengthening the Krylov sequence, as the Chebyshev polynomial grows rapidly with increasing order $i-1$.

5.3.2 Convergence via block size versus convergence via iterations

The random projection algorithm in Algorithm 3 uses the maximal block size, and performance is good when eigenvalues are well-separated. However, when the leading r eigenvalues are not well-separated, convergence in a short Krylov subspace may be poor. Clearly, if the leading spectrum of A is such that $\lambda_1 = \lambda_i$ for $1 \le i \le r$, then all $\hat{\gamma}_i$ will be equal for all block sizes less than r. We would like to show that if the leading part of the spectrum is "sufficiently flat" — that is, all eigenvalues are tightly clustered and have small local gaps — then the improvement in convergence due to block size is comparatively small. To that end, we prove the following theorem.

Theorem 10. *Suppose $\lambda_1 \ge \lambda_2 \ge \ldots \ge \lambda_{\inf}$. Use $\hat{\gamma}_{i,s}$ for $1+2\frac{\lambda_i-\lambda_{i+s}}{\lambda_{i+s}-\lambda_{\inf}}$. Then the improvement in $\hat{\gamma}_i$ as defined in (2.3) is given by*

$$\frac{\hat{\gamma}_{i,r}}{\hat{\gamma}_{i,r+1}} = \frac{(\lambda_{\inf}-\lambda_{i+r+1})(2\lambda_i-\lambda_{i+r}-\lambda_{\inf})}{(\lambda_{\inf}-\lambda_{i+r})(2\lambda_i-\lambda_{i+r+1}-\lambda_{\inf})}. \tag{5.3}$$

Proof. We have

$$\hat{\gamma}_{i,1} = 1+2\frac{\lambda_i-\lambda_{i+1}}{\lambda_{i+1}-\lambda_{\inf}} \tag{5.4}$$

by its definition in (2.3). Simplifying the ratio $\hat{\gamma}_{i,r}/\hat{\gamma}_{i,r+1}$ immediately yields

$$\frac{\hat{\gamma}_{i,r}}{\hat{\gamma}_{i,r+1}} = \frac{(\lambda_{\inf}-\lambda_{i+r+1})(2\lambda_i-\lambda_{i+r}-\lambda_{\inf})}{(\lambda_{\inf}-\lambda_{i+r})(2\lambda_i-\lambda_{i+r+1}-\lambda_{\inf})} \tag{5.5}$$

which completes the proof. □

An implication of this theorem is that tightly clustered eigenvalues cause minimal returns for eigenvalue convergence due to block size inflation. Such eigenvalue clustering is common in interior eigenvalues in a variety of low-rank approximation problems. Note that $\lambda_{i+r}-\lambda_{i+r+1} < \epsilon$ is a sufficient condition for (5.3) to be close to 1. Conversely, neither λ_{i+r}/λ_i nor $\lambda_{i+r+1}/\lambda_i$ need be small to ensure (5.3) be close to 1.

5.4 Greedy Adaptively-blocked Lanczos

The block Lanczos method has notable advantages over the single-vector Lanczos algorithm; the advantage most relevant to this effort is the more rapid convergence of eigenpairs that are well-separated from the remainder of the spectrum. If the convergence is sufficiently fast, the block method will produce an approximation with smaller error than the single vector method. The block method will also require fewer iterations to produce a minimal Krylov subspace of equal size; fewer iterations imply reduced computational cost and reduced need for stabilization to compensate for round-off error. Increasing block size is not without drawbacks; the computational cost of each Lanczos iteration is proportional to $r^2 i$, where r is the block size. Increasing the block size must balance the increase in convergence against the increase in costs. To minimize compute time, we would like to employ the random projection method to approximate only those eigenvalues that are well-separated such that convergence may be driven by block size, and transition to single-vector Lanczos iteration to generate the remaining basis vectors. By doing so, we should get both the good convergence of the random projection method on the leading eigenvalues that are well-separated, and the fast iteration of single-vector Lanczos. We may also incorporate the power refinement of the random projection method to only the well-separated eigenvalues it will benefit most, and transition to single-vector Lanczos on A without refinement to further maximize approximation performance subject to computational costs. Computational costs are managed both by reducing dense floating-point operations due to block size, and also by only performing stationary power iteration on a block of size $m < k$. Likewise, we can leverage the feature of Lanczos methods that constructs a low-rank approximation as they iterate, rather than first constructing a basis and then projecting the input matrix into it.

5.5 Random projections and orthogonal Lanczos iteration

Though ABLE has mechanisms to adaptively grow (or deflate) its block size at each iteration, we have observed that it is not immediately useful for producing minimal Krylov subspaces that effect good low-rank matrix approximations. Random projections offer good eigenvalue approximation properties with excellent stability, but are expensive. By way of comparison, Lanczos iteration offers a lower computational complexity than the direct eigenvalue approximation method, but with an unavoidable loss of stability with continued iteration. All of these observations suggest a hybrid approach that combines the stability and good eigenvalue approximation of the random projection method in Algorithm 3 and the better computational complexity of classic Lanczos iteration in Algorithm 2.

One of the simplest hybrid approaches would be to directly combine Algorithm 3 and Algorithm 2 such that they operate on orthogonal subspaces. Thus we may compute one low-rank approximation $\hat{A}_1^{(m)} \approx A$ and then compute another approximation $\hat{A}_2^{(k-m)} \approx A - \hat{A}_1^{(m)}$. The second algorithm would benefit from the reduced spectral norm $\|A - \hat{A}_1^{(m)}\|$. This reduction may be substantial when $\hat{A}_1^{(m)}$ approximates the leading eigenpairs of A well.

The resulting hybrid algorithm is presented in Algorithm 7. The algorithm first applies Algorithm 3 to approximate the leading m eigenvalues of A, and then proceeds with deflated single-vector Lanczos iteration as described in Algorithm 8 to obtain maximum speed benefits. Needless to say, the switch-over parameter m determines the cost of the initial random projection phase of the algorithm and the stability of the subsequent Lanczos iterations. In terms of compute time, Algorithm 7 has better computational complexity than the random projection method, but not as good as single vector Lanczos. To generate a k-dimensional subspace and approximation, single-vector Lanczos has complexity $kO(n)$, while the random projection method has complexity $O(nk^2)$. Algorithm 7 uses the random projection method, but to generate an m-dimensional space; the complexity of that operation is $O(nm^2)$. The successive $k-m$ deflated Lanczos iterations jointly require $4knm-2km+2kn_{\mathrm{nnz}}+5kn$ FLOPS. Thus, the speedup is determined by how m compares to k. For example in the extreme case

Algorithm 7 Random projections with orthogonal Lanczos iteration

```
Require: input matrix $A$, switchover parameter $m$, subspace sequence length $k$
generate random sample matrix $X_0$ with $m+1$ columns.
obtain $V$,$\Theta$ from Algorithm 3 with $A$ and $X_0$.
obtain $Q$,$T$,$Z$ from $k-m$ iterations of Algorithm 8 with $A$, $X = [v_1\ v_2\ \ldots\ v_m]$, and start vector $v_{m+1}$.
return $[V\ Q]$, $\begin{bmatrix} \Theta & Z \\ Z^T & T \end{bmatrix}$.
```

Algorithm 8 Deflated single-vector Lanczos

```
Require: input matrix $A$, subspace basis $X$ with $j$ columns, start vector $q_0$, sequence length $k$
$z_1 \leftarrow X^T q_1$
$q_1 \leftarrow q_1 - X z_1$
$w \leftarrow A q_1$
$\alpha_1 \leftarrow q_1^T w$
$w \leftarrow w - q_1 \alpha_1$
$\beta_1 \leftarrow \|w\|$
$q_1 \leftarrow w / \beta_1$
for $i = 2 \rightarrow k$ do
    $z_i \leftarrow X^T q_i$
    $q_i \leftarrow q_i - X z_i$
    $w \leftarrow A q_i$
    $w \leftarrow w - q_i \beta_{i-1}$
    $\alpha_j \leftarrow q_i^T w$
    $w \leftarrow w - q_i \alpha$
    $\beta_i \leftarrow \|w\|$
    $q_{i+1} \leftarrow w / \beta_i$
end for
return $[q_1, \ldots, q_k], \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \alpha_3 & \beta_4 & \\ & & \ddots & \ddots & \ddots \end{bmatrix}, [z_1, \ldots, z_k]$
```

$m = 1$, then the cost is the same as single-vector Lanczos iteration. In general, Algorithm 7 costs $4m^3 + 2m^2 + 4mkn - 2mk + 2mn_{\mathrm{nnz}} - 6mn + 2kn_{\mathrm{nnz}} + 5kn$ FLOPS.

Though Algorithm 7 does effectively combine the random projection and classic Lanczos methods, the key limitation is that the value of the switchover parameter m is left to be determined by the user. To minimize compute costs, one would like to choose m to be as small as possible, but big enough so that the well-separated and large eigenvalues are all approximated by the random projection method. Intuitively, we may say that we would like the random projection method, which has good eigenvalue approximation properties, to approximate those eigenpairs which matter most to approximating A well. Though in actuality, the choice of m is more complicated than that, there is no clear choice of m based on beforehand knowledge of A. To better understand how to choose m well, we consider the relationship between convergence of eigenvalues and block size. With that information, we may choose a value for m that represents the best compromise between compute time and approximation quality.

5.5.1 Choosing m automatically

Our criterion for adaptive inflation of the Krylov subspace leverages our assumed spectral distribution. We suppose that leading eigenvalue gaps are large in proportion to their eigenvalues initially and rapidly become proportionally smaller approaching the interior. This is typical of eigenvalue distributions encountered in generic dimension reduction tasks. Figure 5.2 shows a theoretical but representative eigenvalue distribution. Suppose we are to generate a 40-dimensional minimal Krylov subspace. As noted previously, extending the block size beyond 8 will not lead to improved convergence, especially for the leading 8 eigenvalues. However, if a block size of 8 is used, the bounds from Halko et al. suggest that the approximations obtained by the random projection algorithm will be good. Subsequent Lanczos iterations using a block size of 8 operating on $A - \hat{A}^{(8)}$ as defined previously will not yield much of an advantage over single vector Lanczos due to the block size, as the gaps between t' and t are proportionally small. Ideally, we would like to perform the random projection algorithm with a block size of 8, and then deflate the block size to 1 for the remaining iterations. In order to do this automatically, we require a method to detect the point at which eigenvalue gaps

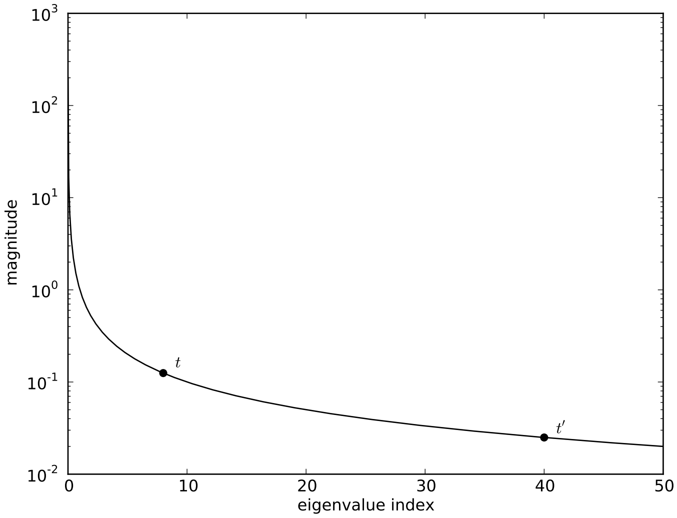

Figure 5.2: Typical eigenvalue distribution. Extension of block size beyond t' does not impart many advantages for the leading 8 eigenvalues.

become proportionally small, assuming an eigenvalue distribution with a shape similar to the distribution in Figure 5.2. Error bounds for eigenpairs in the Krylov subspace allow one to reason about local eigengaps, albeit indirectly and in the presence of "noise" from convergence due to the random start block. In Figure 5.2, the ideal block size is on the point at which local gaps become small, this is also the point at which additional dimensions will result in dramatically decreasing convergence of the principal eigenvalue in the random projection method. More formally, we recall that the bounds for the principal eigenvalue λ_1 are given by

$$0 \le \lambda_1 - \lambda_1^{(n)} \le (\lambda_1 - \lambda_{\text{inf}}) \frac{\tan^2 \theta}{T_{n-1}^2(\hat{\gamma}_1)} \tag{5.6}$$

with

$$\hat{\gamma}_1 = 1 + 2 \frac{\lambda_1 - \lambda_{r+1}}{\lambda_{r+1} - \lambda_{\text{inf}}}. \tag{5.7}$$

The effect of these gaps on the value of $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$ from the above eigenvalue distribution is shown in Figure 5.3. This value represents the change in worst-case bounds on eigenvalue approximation error due to block size increases. A large value of $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$ indicates that block size increases produce only a small additional convergence of the eigenvalue λ_1. The actual value of $\lambda_1^{(k)}$ is shown

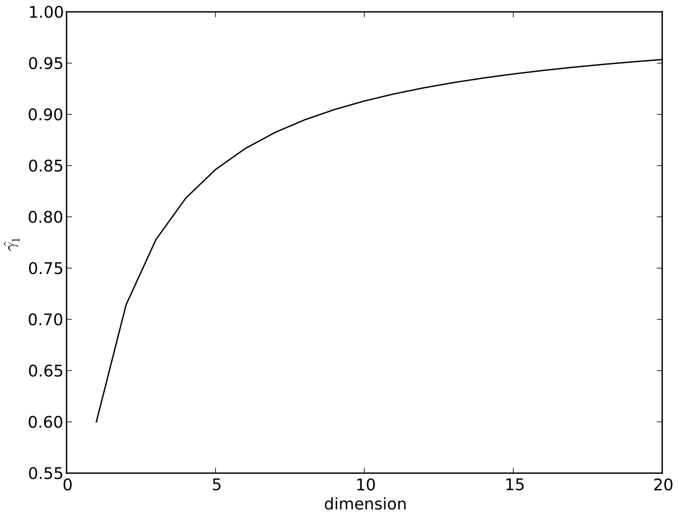

Figure 5.3: Values of $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$ for eigenvalue distribution given in Figure 5.2.

in Figure 5.4. The actual convergence in $\lambda^{(k)}$ follows the trend of the worst-case bounds predicted by $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$; when local gaps $\lambda_1 - \lambda_{r+1}$ fail to produce an increase in $\hat{\gamma}_{1,r}$, then the convergence of the eigenvalue estimate stagnates. This suggests that block size increases over some point — 3 in this example — will not substantially add to the convergence of the principal eigenvalue. Therefore, compute costs benefits may be realized by using a smaller block size than k to produce $k > 3$ dimensions.

The above discussion lacks some rigor; however, it is intended to motivate the following formal discussion. We have already established with Theorem 10 that eigenvalue distributions that have properties similar to those shown in Figure 5.2 — large leading local gaps that decrease rapidly towards the interior of the spectrum — will encounter eigenvalue stagnation if only block size growth is used to drive convergence. Our approach will then track the convergence of eigenvalues with block size inflation, using the following proposition.

Proposition 7. *Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{\inf}$ be real, positive eigenvalues and r and i be natural numbers. We have $\lambda_1 - \lambda_{r+1} \geq \lambda_i - \lambda_{r+1}$ for any i and*

$$\hat{\gamma}_1 = 1 + 2\frac{\lambda_1 - \lambda_{r+1}}{\lambda_{r+1} - \lambda_{\inf}} \geq 1 + 2\frac{\lambda_i - \lambda_{r+1}}{\lambda_{r+1} - \lambda_{\inf}} = \hat{\gamma}_i. \tag{5.8}$$

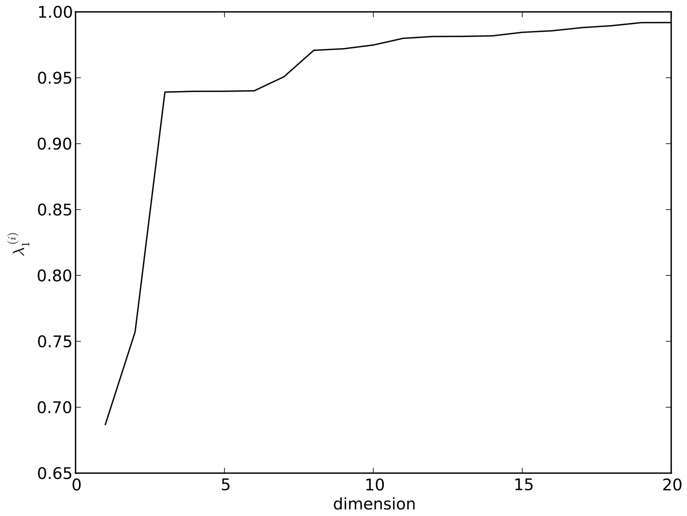

Figure 5.4: Values of $\lambda_1^{(k)}$ from random projections at various dimensions for eigenvalue distribution given in Figure 5.2.

Therefore, the bounds on λ_1 are proportionally no looser than the bounds for any other eigenvalue, and stagnation of the leading eigenvalue implies stagnation of the interior eigenvalues. This allows us to track only the convergence of λ_1.

As the error for approximation of the principal eigenvalue $\lambda_1 - \lambda_1^{(n)}$ is smaller than the errors for more interior eigenvalues, it is sufficient to simply track the convergence of λ_1 and monitor for stagnation. This allows for adaptive generation of the initial start block in Algorithm 7; the initial start block on line 1 may be generated iteratively. As each dimension is added, the principal eigenvalue $\lambda_1^{(j)}$ of approximation $\hat{A}^{(j)}$ may be compared to those from the last few previous iterations. Once the increase in the magnitude of $\lambda_1^{(j)}$ stagnates, then we assume that $\hat{\gamma}_1$ is not increasing much. An iterative inflation procedure is described in Algorithm 9. This algorithm iteratively builds a basis for span$\{AX_0\}$ for a random X_0, while tracking the convergence of the principal eigenvalue. Each iteration generates a new random vector, multiplies it against A, and orthogonalizes against the previous subspace via the Gram-Schmidt process on lines 6 and 7. Lines 12 and 13 represent measuring the cosine angle between the length-m vector e whose entries are all 1 and the last m approximations of λ_1. Once this angle becomes sufficiently small, inflation terminates. This approach may be characterized as greedy, as it will stop inflation at the first stagnation. Non-monotonicity of local gaps may

Algorithm 9 Greedy adaptive block inflation

```
Require: input matrix $A$, stagnation tolerance $\rho$, stagnation window $w$
$j \leftarrow 1$
$\Delta \leftarrow 0$
while $\Delta \le \rho$ do
    generate random vector $x$
    $q_j \leftarrow Ax$
    $q_j \leftarrow q_j - Q_{j-1}Q_{j-1}^T q_j$
    $q_j \leftarrow q_j / \|q_j\|$
    $\hat{A}^{(j)} \leftarrow Q_j^T A Q_j$
    spectral decompose $V\Theta V^T = \hat{A}^{(j)}$
    $\theta_1^{(j)} \leftarrow \theta_1$
    if $j > w$ then
        $a \leftarrow (\theta^{(j-w)2} + \theta^{(j-w+1)2} + \ldots + \theta^{(j)2})$
        $\Delta \leftarrow (\theta^{(j-w)} + \theta^{(j-w+1)} + \ldots + \theta^{(j)})/w^2 a$
    end if
    $j \leftarrow j + 1$
end while
return $\hat{A}^{(j)}, V$
```

result in temporary stagnation followed by a resumption in convergence. Nevertheless, this greedy behavior is advantageous when used to automatically choose the block size m for Algorithm 7; a smaller m results in less cost in the deflated Lanczos iteration.

5.6 The GrABL algorithm

Algorithm 9 provides for an alternate inflation strategy from that used in ABLE, and provides a method for choosing the initial block size m in Algorithm 7 automatically. With this new adaptive inflation algorithm, we develop an adaptively blocked Lanczos algorithm. The algorithm uses the adaptive inflation method to track convergence of the leading eigenvalue to infer the size of the local gaps. Though the adaptive inflation method in Algorithm 9 is greedy, the greediness is advantageous with respect to compute times; the cost of deflated Lanczos iteration depends on the dimension of the deflation subspace. If the deflation subspace has m dimensions, then each Gram-Schmidt step required to orthogonalize a new search direction requires nm FLOPS.

We also note that, assuming an eigenvalue distribution as in Figure 5.2, inflation will terminate with a block size m such that the first m eigenvalues of A are much larger and therefore constitute a larger proportion of the Frobenius norm of A than others. Therefore, it is beneficial to approximate them as well as possible. Better approximations may be attained by incorporation of a power refinement approach as in the direct eigenvalue method in Algorithm 3. Then deflated single-vector Lanczos iteration generates the remaining basis vectors.

The algorithm requires a tolerance for inflation cutoff, a tolerance for deflation, and a tracking parameter w that determines the number of iterations over which $\theta_1^{(i)}$ must be nonincreasing before inflation stops; these are exactly the parameters used by Algorithm 9. After line 1, GrABL has an orthonormal matrix of Ritz vectors V with m columns, with $\text{span}\{V\} = \text{span}\{AX_0\}$ for a random m-column matrix X_0. If GrABL is passed a power refinement parameter, then V is replaced with a new matrix of m Ritz vectors. Then deflated single-vector Lanczos proceeds using the infimal Ritz vector deflated by the preceding $m-1$ Ritz vectors.

The optional refinement step at line 3 allows for further improvement of the initial subspace. This also reflects the hybrid approach between the power-iteration eigenvalue approximation methods

Algorithm 10 Greedy-adaptive Lanczos algorithm with random projection inflation

```
Require: inflation cutoff tolerance $\rho$, convergence window $w$ and dimension $k$; optional refinement parameter $p$
obtain $\hat{A}^{(m)}, V$ from Algorithm 9 run with $w = w$ and $\rho = \rho$, where $V$ has $m$ columns.
if $p > 1$ then
    QR factorize $QR = A^{p-1}V$
    $\hat{A} \leftarrow V^T A V$
    spectral decompose $\hat{A} = V \Theta V^T$
end if
$X \leftarrow [v_1 \ldots v_{m-1}]$
Perform $k - m + 1$ single-vector Lanczos iterations on $A, v_m$ deflated by $X$ using Algorithm 8, get $Q, T, Z$.
$Q = [X, Q]$
$T = \begin{bmatrix} \hat{A} & Z \\ Z^T & T \end{bmatrix}$
return $Q, T$
```

in [36] and traditional Krylov subspace methods. It is this refinement factor that allows GrABL to obtain an advantage over alternate methods such as single-vector Lanczos iteration; we have observed that GrABL with a refinement factor of $p > 1$ produces better low-rank approximations than ordinary Lanczos iteration, while still requiring less computational resources than the direct eigenvalue approximation method in Algorithm 3.

Traditionally, deflation is performed to resolve breakdowns due to loss or near-loss of linear independence in columns of $A^i X_0$ for some i. Those columns that have lost linear independence are removed from the iteration block, and each successive iteration is orthogonalized against them to maintain the Lanczos recurrence. For GrABL, deflation is used somewhat differently, and different vectors may be used. We are deflating not against Lanczos vectors, but against Ritz vectors. When deflation with Ritz vectors is used at iteration i, maintaining orthogonality of the Lanczos basis requires orthogonalization against *all* $i-1$ previous Lanczos blocks rather than only those that have been deflated. When deflation is performed at the first iteration, the Ritz and Lanczos bases coincide. We may simply discard the trailing $m-1$ Ritz vectors, and proceed with single-vector iteration with Ritz vector m deflated against the leading $m-1$ Ritz vectors at cost equal to deflation against Lanczos vectors.

Deflation may result in some improvement in the stability of future Lanczos iterations. We have a relationship between the spectral norm of the input matrix A and the loss of orthogonality in the Lanczos basis [74]. In our observations, we have witnessed some improvements in stability due to deflation, but the improvements were not large or consistent enough to warrant the abandonment of reorthogonalization measures.

As noted previously, the random projection algorithm requires compute effort asymptotic with $O(nk^2)$ to generate a k-dimensional near-minimal Krylov subspace when the size and density of the input matrix is fixed. GrABL has a compute resource advantage over the random projection algorithm as it minimizes the cost per iteration by minimizing the block size as early as possible. Its computational cost is equal to that of Algorithm 7, but the parameter m is minimized via greedy termination of the block size inflation process. Moreover, like Algorithm 7, immediate deflation to single-vector Lanczos iteration minimizes the cost of each iteration of Algorithm 8.

5.7 Conclusion

Random projections may be effective eigenvalue approximation methods which are also robust to loss of orthogonality. Though complete convergence to eigenvectors may be slow in general, random projections still produce good approximations to eigenvalues with only a few stationary iterations. They may be used for finding low-rank matrix approximations when exact eigenvalues are not required. Random projection methods share similarities with block Krylov subspace methods, which have, in single-vector form, also been used for relaxed alternatives to the truncated SVD for low-rank matrix approximation. Random projection methods do have two disadvantages with respect to computational costs: the dense linear algebra operations require FLOPS that scale quadratcially with the dimension of the subspace. The method also requires $(2p+1)k$ sparse matrix-vector products to form a k-dimensional subspace; this is problematic when sparse matrix-vector products are not inexpensive. By performing random projections with stationary iteration only on those m eigenpairs or singular triplets that contribute most to the Frobenius norm of the approximation, computational costs due to both sparse matrix multiplication and dense matrix operations can be reduced. Generating the remaining $k-m$ subspace dimensions with single-vector Lanczos iteration can further reduce compute costs. In this case, only $2mp+k+2m$ sparse matrix-vector multiplications are necessary, and the cost for dense matrix operations is no longer asymptotically quadratic with subspace size k.

Though the FLOP cost analysis of the random projection is somewhat straightforward, it is not trivial to determine the optimal m for changeover from random projections to single-vector Lanczos iteration. We have presented a method that chooses m greedily, and based it on an assumption that eigenvalue clustering becomes tight in a somewhat uniform way. Though clustering of eigenvalues does imply that improvements in eigenvalue convergence solely due to increasing block size will stagnate, the greedy approach may prematurely deflate the working subspace. Though we argue that eager deflation is beneficial from a computational cost standpoint, smaller approximation norms may be realized by prolonged block inflation. It would be useful for future efforts to characterize the convergence of eigenvalue approximations in a hybrid subspace so that worst-case eigenvalue convergence in GrABL may be compared to other Krylov subspace methods.

Chapter 6

Applications of GrABL

To illustrate the performance of GrABL, we present results from numerical experiments. These experiments mirror those presented in Chapter 4, and are intended to enable and engender comparison between the short block Krylov subspaces methods and GrABL. As in the experiments presented previously in Chapter 4, the focii of the experiments may be broken into two categories: experiments to demonstrate the norm and stability of low-rank matrix approximations generated with the methods under consideration, and the actual performance of data analysis methods — for example LSI — when using low-rank approximations generated by the competing low-rank approximation methods. The comparison of low-rank approximation error is relevant, as it clearly shows the differences between the low-rank approximation methods. However, some data analysis methods that use low-rank matrix approximations are not tightly coupled with the Frobenius or nuclear norm; a strictly smaller low-rank approximation error does not necessarily imply improved performance. For example, the effects of PCA on a "downstream" classifier may actually improve performance due to filtering. The projection acts as a filter, separating noise from signal. The performance of the "downstream" consumer of the low-rank approximation is intended to directly show the impact of the different low-rank matrix approximation techniques on methods consuming low-rank matrix approximations.

To study the approximation error, we generated low-rank approximations of a selection of the same data used in the experiments demonstrating the performance of short block Krylov subspaces: the extended Yale eigenface data [45,49], an inverted graph Laplacian matrix representing the road network of Colorado derived from the US Census Bureau's tiger-line GIS data [1], and a term-document matrix from the UCI machince learning repository's bag-of-words collection [26]. In each experiment, we generate a k-dimensional approximation to A, compare the results of the random projection algorithm in 3, GrABL with $\rho = 10^{-3}$ and window parameter $m = 3$, ABLE, and single-vector Lanczos. Runs of Algorithm 7 with $m = k/2$ were also performed to provide comparison against the greedy block size deflation of GrABL. Start and inflation vectors were drawn from a uniform distribution. Runs with no reorthogonalization and with full reorthogonalization were performed with GrABL and Lanczos; only the full reorthogonalization results were used for comparing low-rank approximations. The runs performed without any reorthogonalization are intended to compare the stability of GrABL and single-vector Lanczos iteration. Again, we compare the approximation error $\|A - \hat{A}^{(k)}\|$ for each method, and analyze the magnitudes of the Ritz values to draw conclusions regarding the differing performances of the algorithms. We consider the inflation and deflation histories of the adaptive algorithms. Due to adaptive inflation, ABLE may produce excess Ritz pairs; the Krylov subspace was truncated using the top k Ritz pairs. The experiments were performed on one of the same computers as the experiments in Chapter 4: a MacPro with 2 quad-core Intel Xeon processors running at 3 Ghz. The computer had 8 GB or ram, and was running Mac OS X 10.6.8. The experiments were run using numpy linked against Apple's vecLib tuned LAPACK and BLAS implementations.

To study the results in applications that use low-rank matrix approximations generated by the methods considered herein, we used the NPL information retrieval problem [71] and the visualization of the Colorado road network Laplacian in the commute-time embedding [70].

6.1 Experiments with the Yale face data

Recalling the eigenfaces experiments in Section 4.1, the eigenface method [80] is an application of PCA to the facial recognition problem. Given a set of training images, PCA is used to produce a small subspace that contains the features needed for face classification. As in the experiments evaluating the "shrink-and-iterate" and hybrid methods, we flattened each image into a vector a_i, stacked them into a matrix $[a_1\ a_2\ \ldots] = A'$ and set $A = A' - \mu$ where μ is the arithmetic mean of all images. This

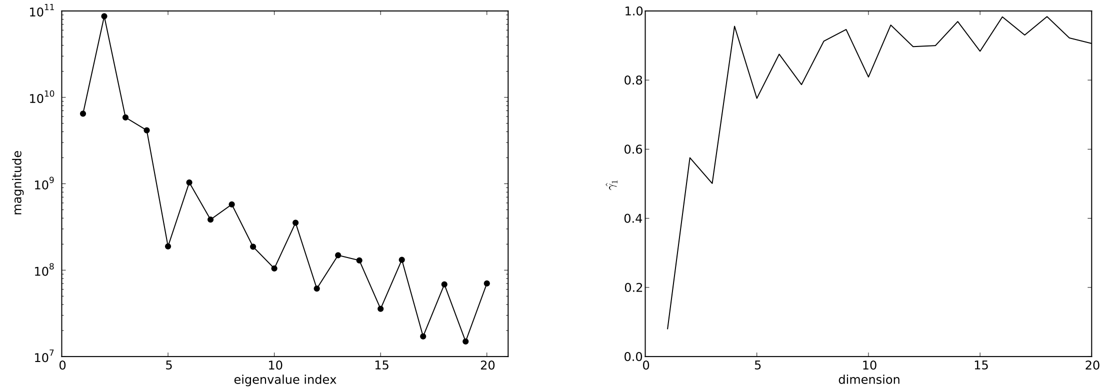

Figure 6.1: Eigenvalue gaps for A^TA for the Yale face data (left), and $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$ (right).

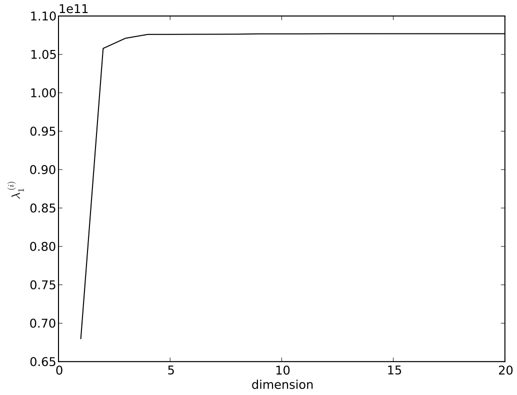

Figure 6.2: Values for $\lambda_1^{(n)}$ for the Yale face data.

corresponds to a centered data matrix as would be used for principal components analysis. We then approximated the Gram (or covariance) matrix A^TA. Eigenvalues of A^TA were presented in a log-scale plot in Figure 5.1. The eigenvalue gaps and values for $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$ are shown in Figure 6.1.

The leading four eigenvalues of A^TA are well-separated and the leading two are separated from the rest by nearly an order of magnitude. When run with $\rho = 10^{-3}$ and deflation window $m = 3$, GrABL chose a block size of 4. Figure 6.2 shows the convergence of $\lambda_1^{(n)}$ from Algorithm 3, which follows the decreasing values for $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$. Therefore, for reduction to 4 or fewer dimensions, GrABL is equivalent to the random projection algorithm. Beyond 4 dimensions, GrABL produces approximations with Frobenius norm close to the random projection method. Figure 6.3 shows the Frobenius

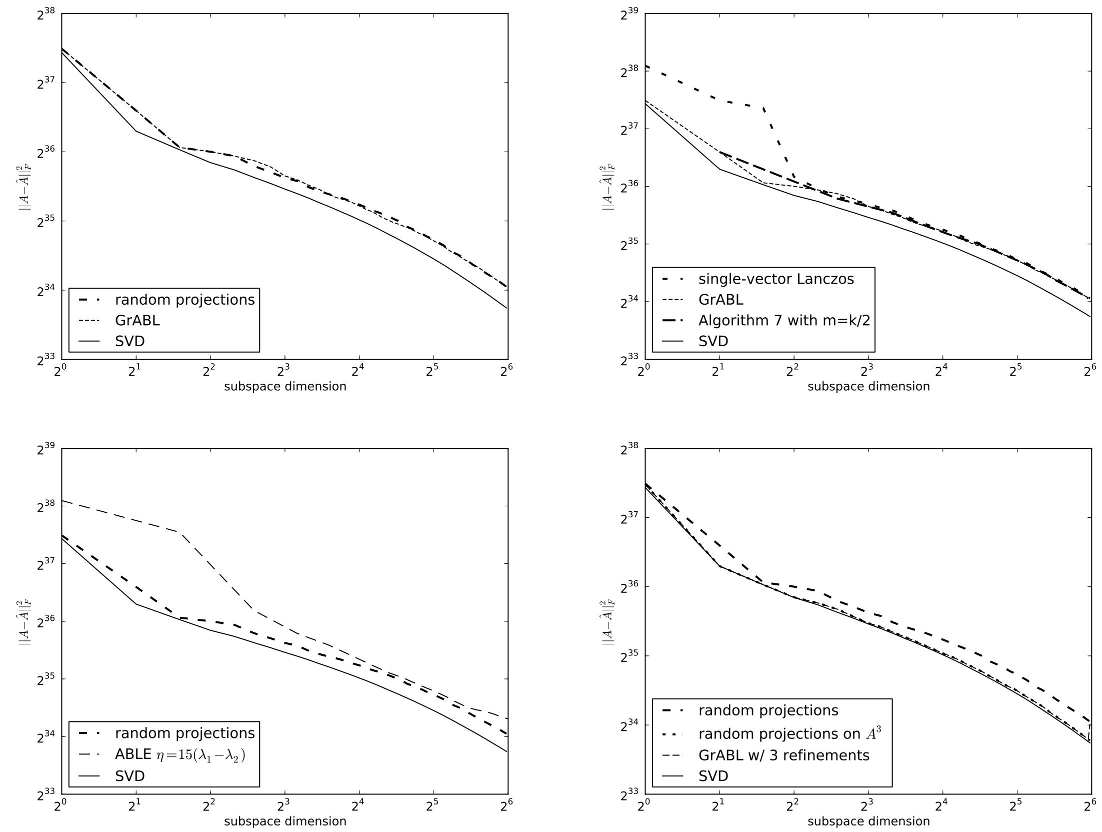

Figure 6.3: Low-rank approximation errors of Yale data; GrABL v. random projections (top left); GrABL v. single-vector and Algorithm 7 with $m = k/2$ (top right), ABLE v. random projections (bottom left) and GrABL v. random projections with refinements (bottom right).

norm of the low-rank approximation $\hat{A}^{(k)}$ for various dimensions k. Using a refinement factor of 3 improves approximation quality further. Though the block size chosen by GrABL was smaller than $k/2$ for large dimensions, Algorithm 7 with $m = k/2$ did not produce substantially smaller approximation errors. Due to the eigenvalue distribution, diminishing returns are rapidly encountered, even when exact eigenpairs are used for projection. Though this might appear to suggest that only the first 5 eigenvectors are needed for the dimension reduction, the resulting face images are substantially improved by adding successive dimensions to the projection. We observe that GrABL exhibits low-rank approximation error comparable to the random projection method for all dimensions, and that single-vector Lanczos lags behind both GrABL and the random projection method for small dimensions before achieving parity for higher dimensions.

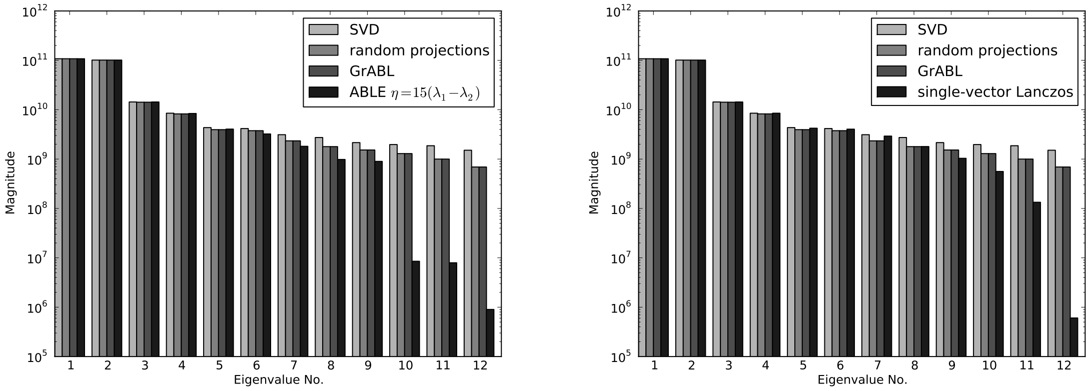

Figure 6.4: Ritz values from minimal 12-dimensional subspaces for GrABL, Lanczos, ABLE and random projections.

To further show sources of the error lag between single vector Lanczos and GrABL/random projection for small dimensions, we show the Ritz values used for the 12-dimensional approximation in Figure 6.4. All algorithms approximate the leading eigenvalues of A^TA comparably well; however, single vector Lanczos has substantial error in its approximation of the tenth eigenvalue. GrABL approximates this eigenvalue better. The poorer approximation of the trailing eigenvalues is a trend evident in Lanczos approximations over all dimensions. ABLE approximates the 5 leading eigenvalues well, but has significant gaps for the trailing ones.

ABLE was run with $\eta = 15(\lambda_1 - \lambda_2)$ in order to include the first 2 eigenvalues. This value of η was chosen to produce a block size of 4, equivalent to the block size found by GrABL. ABLE produces the largest approximation error of all algorithms. This performance may be attributed to the inflation strategy of ABLE; ABLE simply uses random vectors to inflate the Krylov subspace. Random vectors initially will have a small cosine angle between leading eigenvectors of A. Table 6.1 shows the block size history of ABLE and GrABL for the first 5 iterations; block size does not change after the fourth iteration for either algorithm. There is a noticeable lag in approximation error for ABLE at 4 dimensions, which is when it inflates the Krylov subspace. ABLE is not intended to be a minimal Krylov subspace method, so this feature of the algorithm is ordinarily not an issue. For minimal Krylov subspaces, a different inflation strategy or different inflation vectors may produce better results.

Iteration	1	2	3	4	5
ABLE	1	2	3	3	3
GrABL	4	1	1	1	1

Table 6.1: Block size history of ABLE and GrABL run on the Yale data for the first 5 iterations.

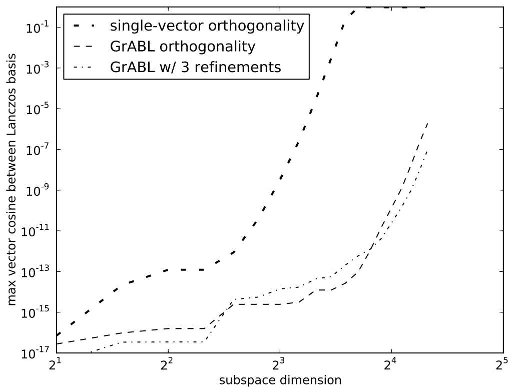

Figure 6.5: Maximum vector cosine between Lanczos basis vectors for GrABL and single-vector Lanczos when run without reorthogonalization.

The random projection method produces good approximations when operating on A^3; it closes much of the gap between the random projection method on A and the spectral approximation. With refinement, GrABL produces near-parity with the random projection approximation operating on A^3 for smaller dimensions. GrABL also performs far fewer passes over the data; which is significant when matrix multiplications are not trivial. This better approximation both decreases the error of the approximation and decreases loss of orthogonality in GrABL.

Loss of orthogonality is a difficulty for all Lanczos algorithms. We ran both single vector Lanczos and GrABL without any reorthogonalization to study the loss of orthogonality. Figure 6.5 shows the maximum vector cosine between Lanczos basis vectors up to 20 dimensions. GrABL demonstrates slower loss of orthogonality, but still loses orthogonality eventually. Power iteration refinement provided a marginal improvement in stability. ABLE has similar stability to single-vector Lanczos initially, but loses orthogonality after the first inflation, likely due to convergence of the principal eigenpair.

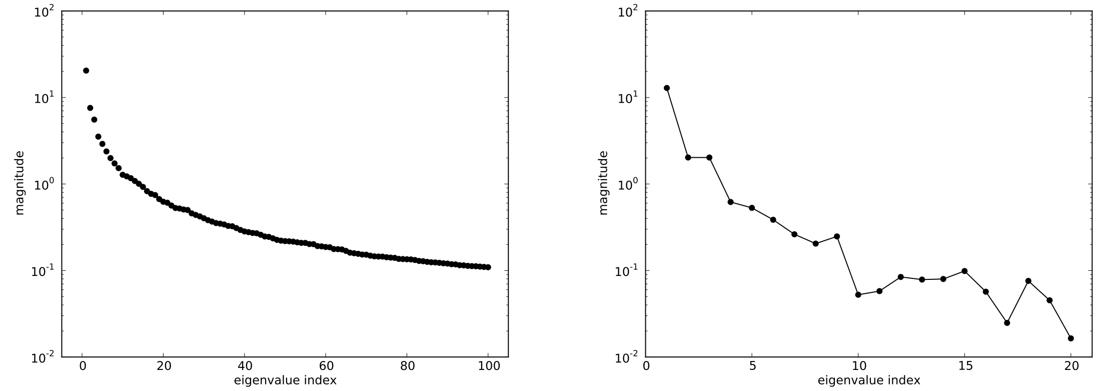

Figure 6.6: Leading 100 eigenvalues of the inverted graph Laplacian matrix for the Colorado road network (left) and eigenvalue gaps (right).

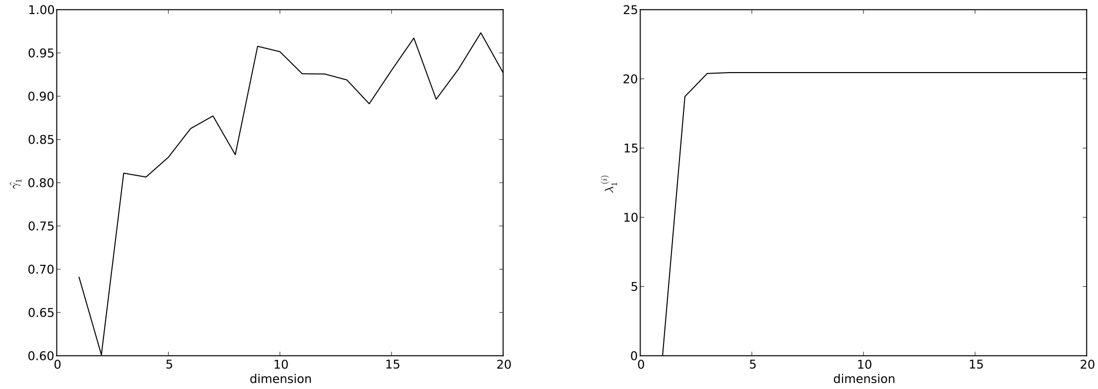

Figure 6.7: Values of $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$ (left) and $\lambda_1^{((n))}$ (right) for the inverted Colorado road network Laplacian.

6.2 Experiments with the Colorado road network Laplacian

We also performed experiments on the road network graph from the state of Colorado, motivations for study of graph Laplacian problems are detailed in Section 4.3. We study the road network of Colorado, as obtained from the US Census Bureau's TIGER-Line GIS data [1]. We used edge weights representing distance between vertices and replaced all 2-degree vertices with an edge. We then extracted the largest connected component. Again, the resulting matrix is $345,025 \times 345,025$. The leading 100 eigenvalues of the reciprocated Laplacian matrix L^+ are shown in Figure 6.6.

The leading eigenvectors of L^+ are well-separated, but the rate of decay in gaps is not as sharp as for the Yale face data. The slower rate of decay of eigenvalues in the Laplacian influences the

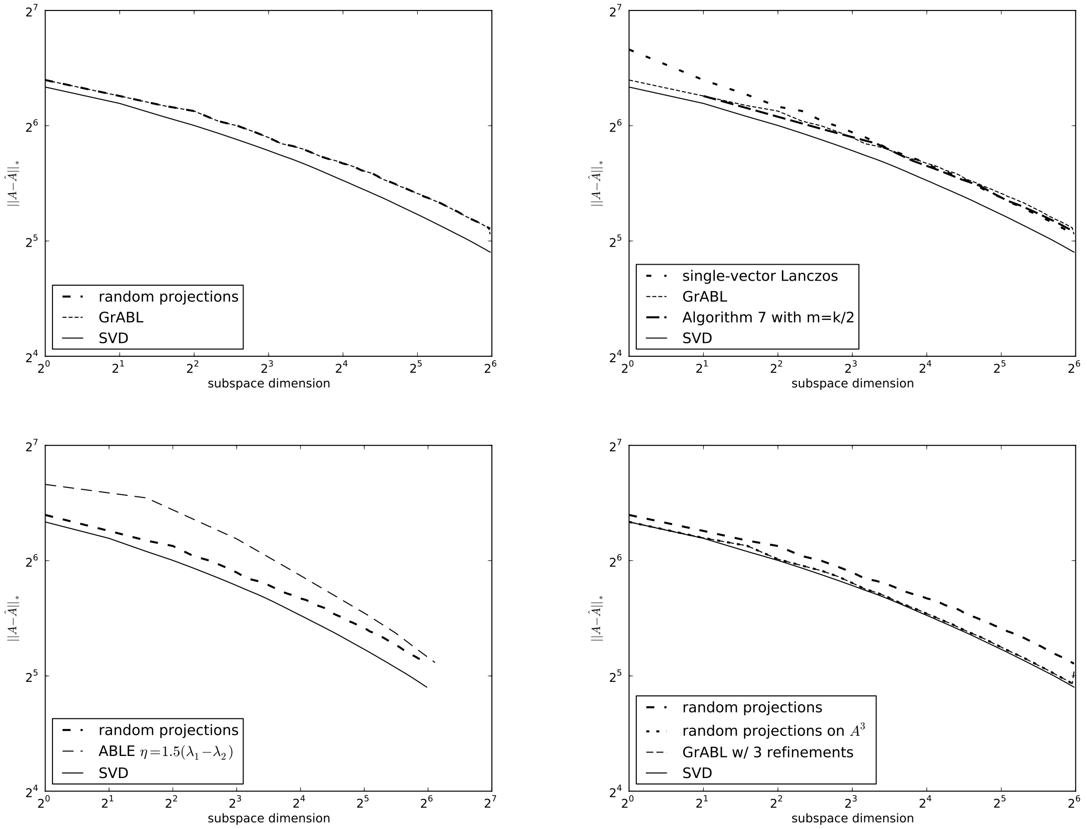

Figure 6.8: Low-rank approximation errors of Colorado graph Laplacian; GrABL v. random projections (top left); GrABL v. single-vector (top right), ABLE v. random projections (bottom left) and GrABL v. random projections with refinements (bottom right).

convergence of eigenvalues in Krylov subspaces, and the relative merits of GrABL to the random projection algorithm to single-vector Lanczos. The low-rank approximations of L^+ are shown in Figure 6.8. Though these approximations also show diminishing returns with added dimensions, the more slowly decaying gaps lead to a larger initial block size. With $\rho = 10^{-3}$ and $m = 3$, GrABL chose a block size of 8. ABLE was run with $\eta = 1.5(\lambda_1 - \lambda_2)$, again to arrive at a block size of 8, as was found by GrABL. Note that Algorithm 7 did not produce significantly smaller errors than GrABL at any dimension, even though it used $m = k/2$.

The overall trends between GrABL, the random projection method and single-vector Lanczos are similar to the trends observed in the Yale face data experiments; GrABL produces approximations with similar norm to the random projection method, while single-vector Lanczos lags initially but ob-

Iteration	1	2	3	4	5	6	7
ABLE	1	2	5	7	8	10	12
GrABL	8	1	1	1	1	1	1

Table 6.2: Block size history of ABLE and GrABL for the first 7 iterations on the Colorado road network.

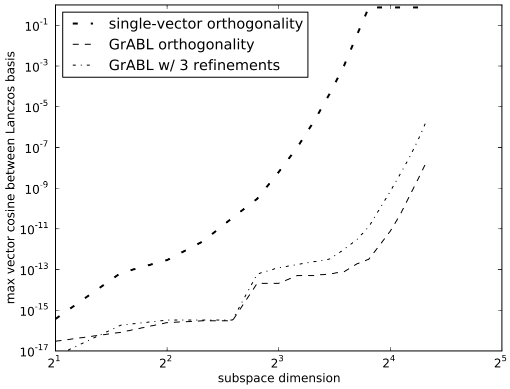

Figure 6.9: Maximum vector cosine between Lanczos basis vectors for GrABL and single-vector Lanczos when run without reorthogonalization.

tains equivalent performance at higher dimensions. The difference in approximation error between the spectral approximation and the random projection approximation is larger, but is closed by random projections operating on L^{+3}. Again, GrABL with $p = 3$ performs nearly as well as the random projection method operating on L^{+3}, but the advantage of random projections on L^{+3} over GrABL is larger than in the Yale experiment. ABLE again produces the largest approximation error. Table 6.2 shows the block size histories of ABLE and GrABL for the first 7 iterations. ABLE only required 7 iterations, and produced 1 excess Lanczos vector. Note that if a subspace between 50 and 80 were required, ABLE's block size of 29 will produce a large number of excess basis vectors that would be discarded.

Orthogonality is also lost rapidly for this graph Laplacian, though not as quickly as for the Yale data. Figure 6.9 shows the maximum vector cosine for Lanczos basis vectors. Adding refinement to GrABL does not result in an improvement in stability in this case.

The inverted graph Laplacian induces the commute-time embedding. For certain classes of graphs, the commute-time embedding reveals interesting and meaningful structure. The Colorado road network graph is such a graph for which the commute-time embedding is useful. The commute

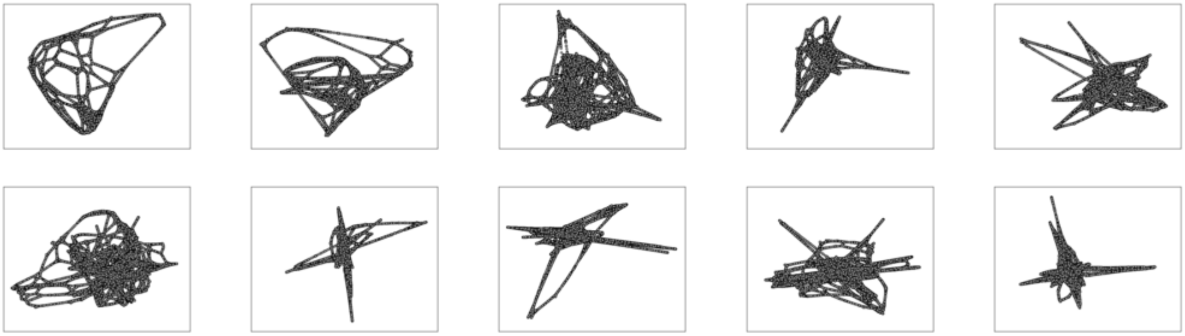

Figure 6.10: Grand tour of first 20 dimensions of commute-time embedding of Colorado Road network graph.

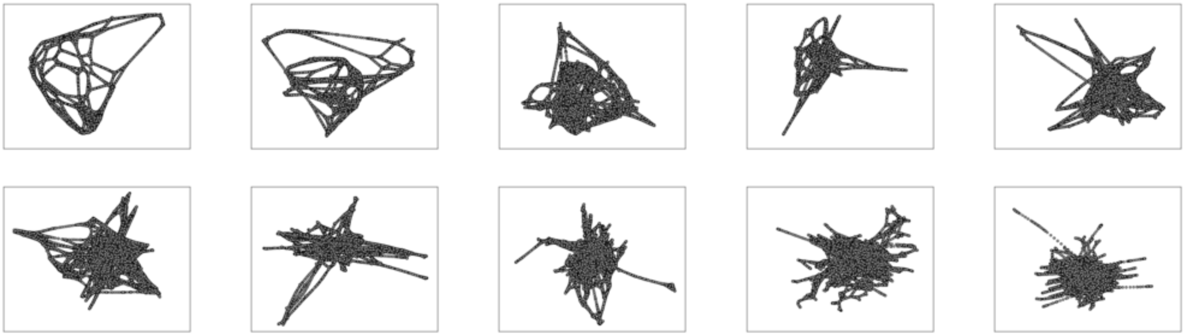

Figure 6.11: Grand tour of first 20 dimensions of commute-time embedding of GrABL-produced low-rank approximation of the Colorado Road network graph using a refinement factor of 2.

time embedding is defined in terms of the pseudoinverse L^+ of the graph Laplacian L with spectral decomposition

$$U \Lambda U^T = L^+. \tag{6.1}$$

Columns of $U\Lambda^{1/2}$ give coordinates of vertices of the graph in the commute-time embedding. Good approximations of eigenvalues *and* eigenvectors of L^+ are necessary for good approximations of the commute-time embedding for visualization purposes. We therefore used modest power refinement factors for the algorithms under consideration; for GrABL and random projections, we used a refinement factor of $p = 2$. Figure 6.10 shows the grand tour of the first 20 dimensions of the commute time embeddings of L^+ using eigenvectors, Figure 6.11 shows the grand tour of the commute-time embedding of the GrABL-produced low-rank approximation and Figure 6.12 shows the commute-time embedding of the random projection-produced low-rank approximation. All 20 dimensions of

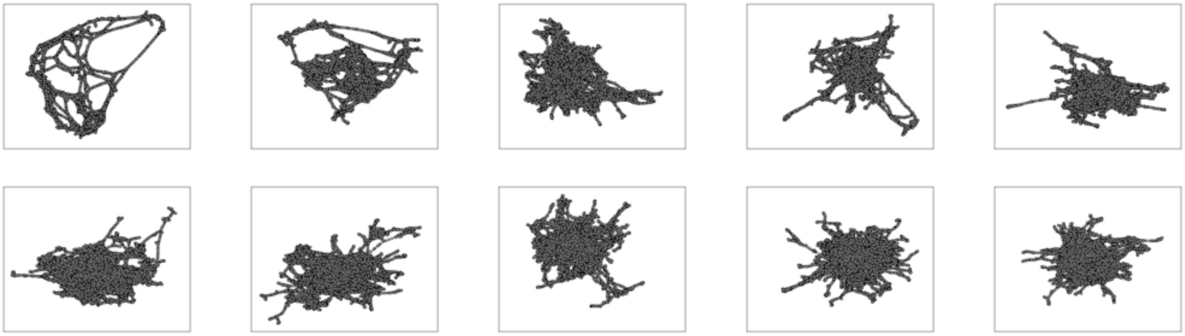

Figure 6.12: Grand tour of first 20 dimensions of commute-time embedding of random projections-produced low-rank approximation of the Colorado Road network graph.

the commute-time embedding in Figure 6.10 exhibit definite structure. All of GrABL, static-block Lanczos and random projections approximate the leading 10 dimensions without any noticeable visual error. The remaining 10 dimensions are also approximated with some accuracy, though the random projection method approximates the trailing 10 dimensions somewhat better than GrABL. Nevertheless, GrABL is intended to be less costly than random projections with refinement.

As this inverted graph Laplacian matrix is sparse, even in its factored form, the extra overhead for refinement is smaller than for the dense Yale face data. We accounted for partial reorthogonalization in single-vector Lanczos iteration in ABLE and GrABL, assuming that $\lceil i^{1/2} \rceil$ Ritz vectors have converged at iteration i. The FLOP counts for generating a subspace of up to 64 dimensions is shown in Figure 6.13. We present both the total FLOP counts, and the FLOPS due to sparse and dense operations. These approximations correspond the norms of the approximations in the bottom-right plot of Figure 6.8. Some overhead is evident for GrABL with refinements at low dimensions; however, this overhead is eventually overwhelmed by the superior computational complexity of GrABL. GrABL only has to perform random projections on a block size of 8. For larger dimensions, the advantage of GrABL over random projections is evident. Note that slope of the lines in the GrABL chart in Figure 6.13 changes after deflation occurs at a block size of 8; this is due to the decreased cost of each successive iteration from block size deflation. The may be contrasted with the constant slope of the lines in the comparable FLOP charts in Figure 4.13.

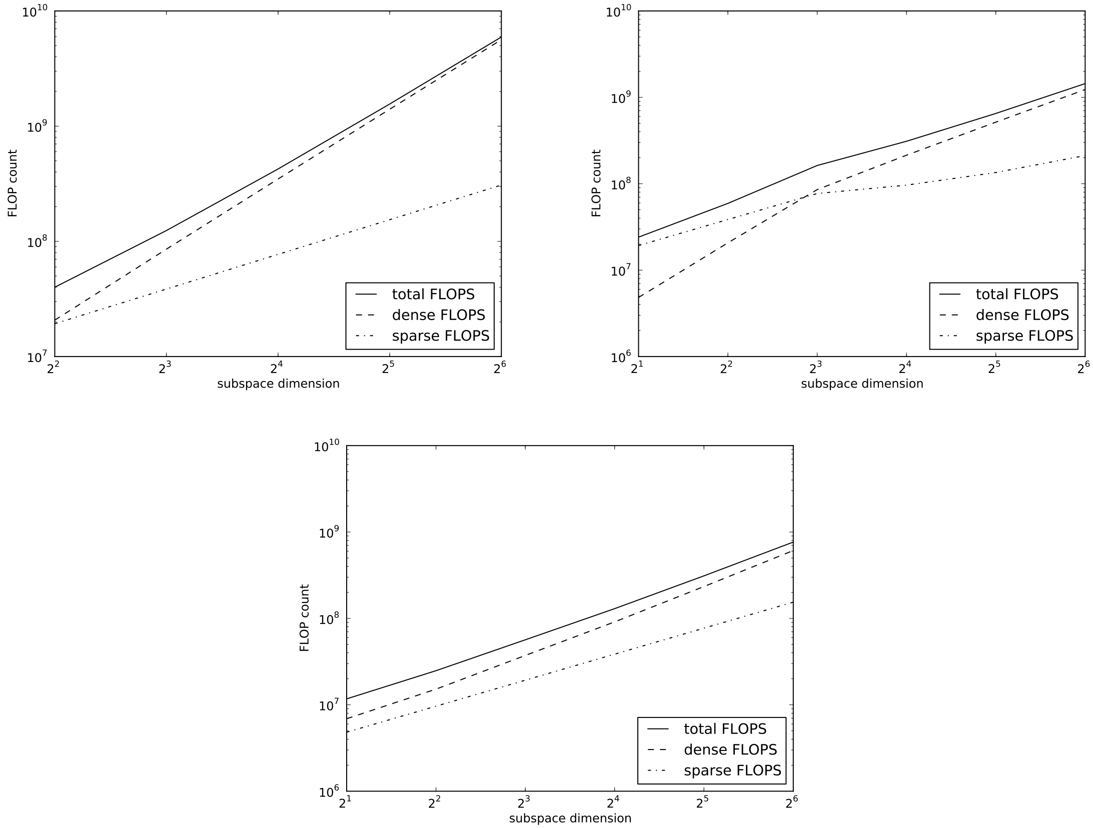

Figure 6.13: FLOP counts for generating a low-rank approximation of $\hat{L}^{+(k)}$ with random projections (top left), GrABL with $p = 3$ (top right) and ordinary single-vector Lanczos for various dimensions.

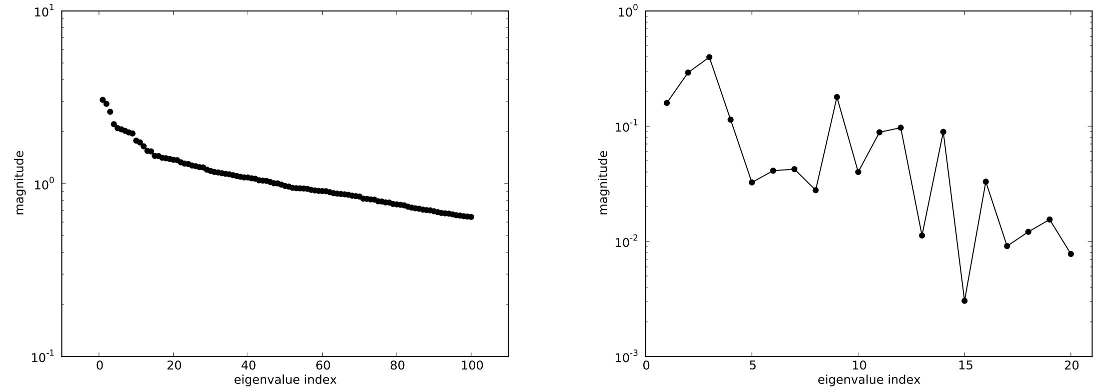

Figure 6.14: Leading 100 eigenvalues of A^TA for the Enron email corpus (left) and eigenvalue gaps (right).

Iteration	1	2	3	4	5	6	7	8	9	10	11
ABLE	1	1	2	2	3	4	4	6	7	8	9
GrABL	5	1	1	1	1	1	1	1	1	1	1

Table 6.3: Block size history of ABLE and GrABL for the first 11 iterations on the Enron corpus.

6.3 Experiments with the Bag of Words term-document matrix

The truncated SVD is used in information retrieval applications as part of LSI. As in the experiments presented in Section 4.2, we used the Enron email corpus which contained 39,861 documents and 28,099 unique terms. The algorithms in consideration herein were applied to the matrix A^TA. The spectrum of A^TA is shown in Figure 6.14. The matrix A^TA has a flatter spectrum than either the eigenface data or the inverted Colorado road network graph Laplacian. The eigenvalue gaps influence the rate of convergence of eigenvalue approximations. The trend apparent between the Yale face data and Colorado road network Laplacian is continued in this experiment; smaller local gaps lead to a smaller advantage of maximal block sizes. Figure 6.15 shows improvement in worst-case bounds due to block size inflation and the actual convergence of $\lambda_1^{(k)}$.

The low-rank approximations are shown in Figure 6.16. The smaller local gaps imply a decreased initial convergence advantage to block size. In contrast to the Yale face data experiment, GrABL and the random projection method hold no particular advantage over single-vector Lanczos, even at small dimensions. GrABL's error advantage over single-vector Lanczos is only slight. ABLE produces larger approximation errors than all other methods, as was the case in the preceding ex-

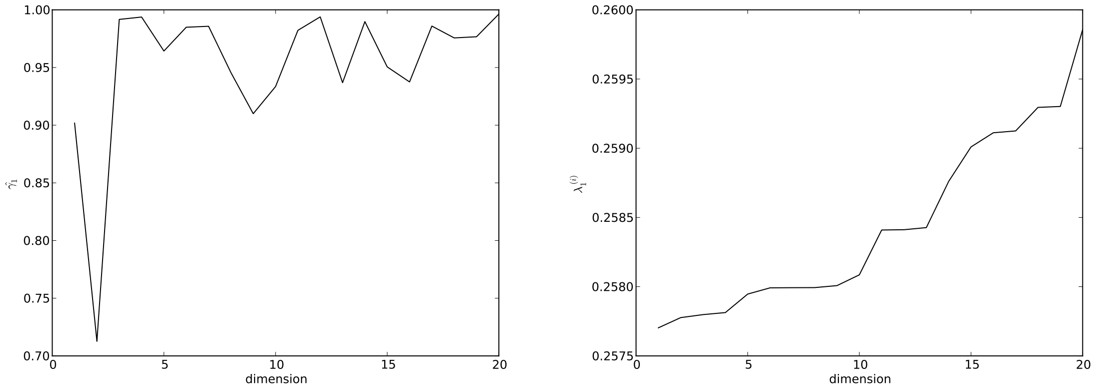

Figure 6.15: Values of $\hat{\gamma}_{1,r}/\hat{\gamma}_{1,r+1}$ (left) and $\lambda_1^{((n))}$ (right) for the Enron email corpus.

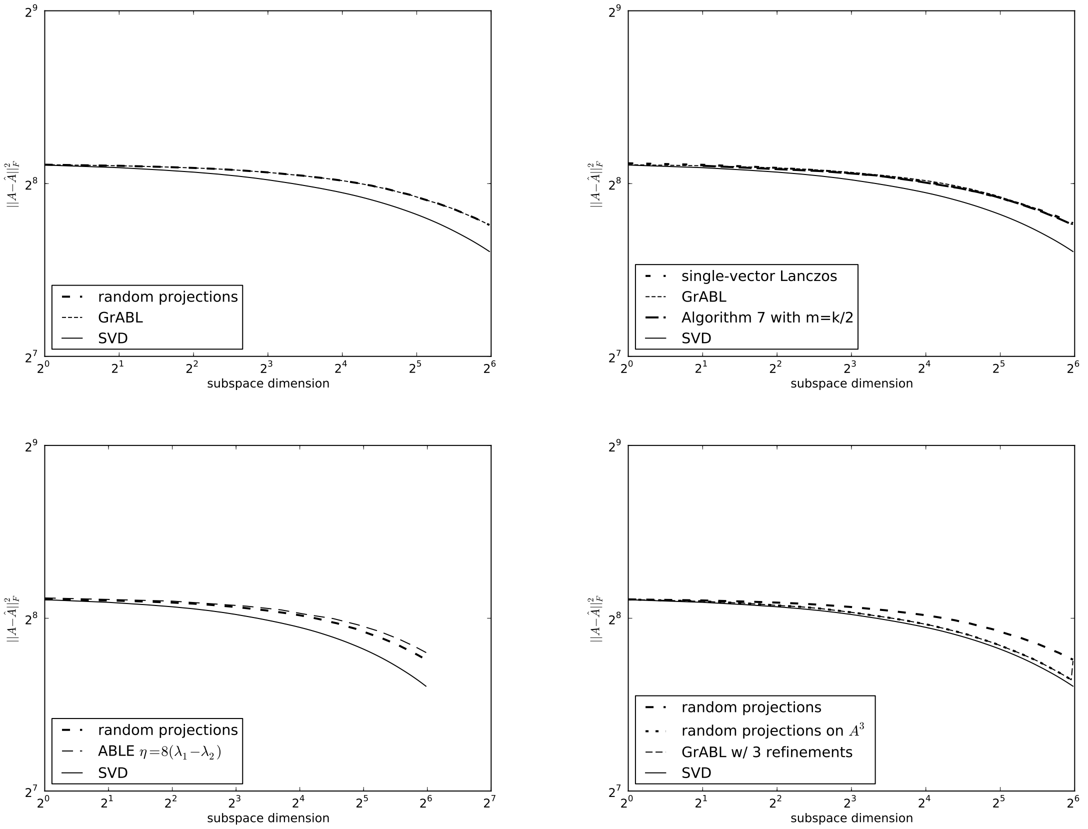

Figure 6.16: Low-rank approximation errors of Enron document corpus; GrABL v. random projections (top left); GrABL v. single-vector (top right), ABLE v. random projections (bottom left) and GrABL v. random projections with refinements (bottom right).

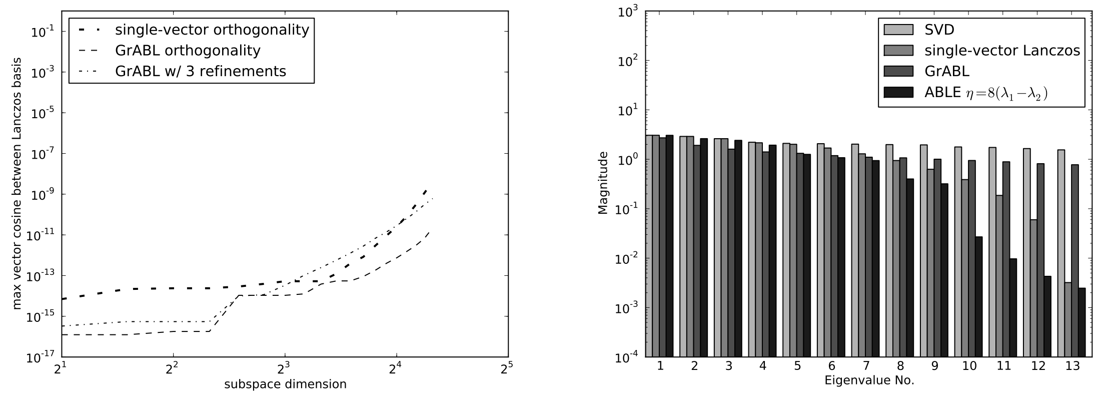

Figure 6.17: Maximum vector cosine between Lanczos basis vectors for GrABL, ABLE and single-vector Lanczos when run without reorthogonalization (left) and Ritz values from minimal 13-dimensional subspaces for GrABL, Lanczos and ABLE (right).

periments. Likewise, Algorithm 7 with $m = k/2$ did not produce smaller approximation errors than GrABL, despite using a larger block size. Application of random projections to $(A^T A)^3$ improved the performance of the random projection method substantially as did the analogous refinements applied to GrABL. ABLE was run with $\eta = 8(\lambda_1 - \lambda_2)$ to produce a block size history most comparable to GrABL. The block size histories of ABLE and GrABL are shown in Table 6.3.

The smaller local gaps lead to slower convergence of eigenvalues, but this also implies more stable iteration. Figure 6.17 shows the loss of orthogonality over iterations for single-vector Lanczos, ABLE and GrABL with eigenvalue estimates from the 13-dimension subspace. Though its trailing eigenvalue estimates have smaller magnitude than GrABL, ABLE does approximate the leading eigenvalue slightly better than single-vector Lanczos at 10 dimensions; the residual is less than 1.1×10^{-5} compared to 6.4×10^{-5} for single-vector Lanczos. The better approximation of the leading eigenvalue implies faster loss of orthogonality. GrABL with refinements also has somewhat faster loss orthogonality than single-vector Lanczos.

The enron corpus problem is also sufficiently large to produce meaningful FLOP comparisons. As in the experiments with the Colorado road network, partial reorthogonalization was used to maintain orthogonality of the Lanczos basis. The FLOP counts for generating a low-rank approximation of the term-document matrix with eigenvalue approximations is shown in Figure 6.18. The plot compares the same norm case as is shown in the lower-right plot in Figure 6.16. The superior computational

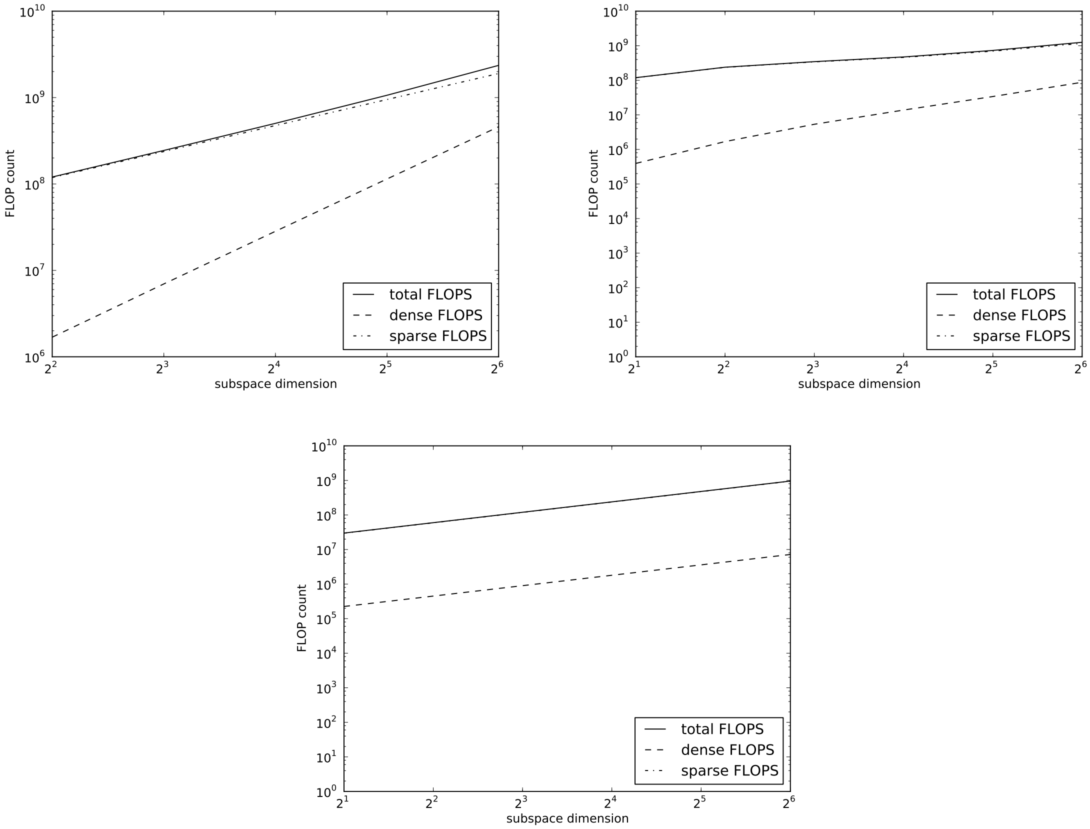

Figure 6.18: FLOP counts for generating low-rank approximations of the term-document Gram matrix $A^T A$ for the enron email corpus. The random projection method (top right), GrABL with $p = 3$ (top right) and single-vector Lanczos (bottom) are compared.

complexity of GrABL is also evident in the scaling results of this experiment, and again, there is a reduction in the slope of the line for FLOPS for generating a subspace of fixed dimension.

6.4 Information retrieval with NPL data

To study the effects of application of SVD approximation to an information retrieval task, we consider the NPL data, the same problem we studied in Section 4.2. The resulting term-document matrix has 7,491 columns and 11,429 rows. We applied term frequency and inverse document frequency scaling to the entries in the term-document matrix, and generated low-rank approximations $\hat{A}^{(k)}$ for $k = 2^i$ with $2 \le i \le 8$. Each query was applied against the low-rank approximation, and documents were scored for relevance using vector angle cosines. We computed document-level averages for each query at 100; the document-level average is simply the average percentage of relevant documents retrieved in the first 100 query results. Figure 6.19 shows the query performance for the random projection, GrABL approximations and LSI, and the Frobenius norms of the low-rank approximations for all three methods.

There are multiple methodologies for application of LSI-like dimension reduction to querying problems. As in the previous experiments with the NPL data, we treat the subspace projection as a query expander, and the cosine formula is

$$\cos(a_i, \mathrm{q}) = \frac{\langle a_i, V^T \mathrm{q}\rangle}{\|a_i\| \|VV^T \mathrm{q}\|}. \tag{6.2}$$

Blom and Ruhe observed that treatment of Krylov subspace methods as query expanders rather than dimension reducers produced better precision and recall in [10]. We witnessed the same behavior, and show results evaluating performance using query expansion cosines. From Figure 6.19, it is evident that random projections produce better query results than GrABL at lower dimensions. In fact, random projections outperform LSI as well. This may be understood in terms of the different subspaces being used as query expanders. Since extremal eigenvalues converge quickly in a Krylov subspace, a minimal Krylov subspace will include both latent feature vectors that represent “signal” from the leading part of the spectrum and will also include latent feature vectors representing “noise”

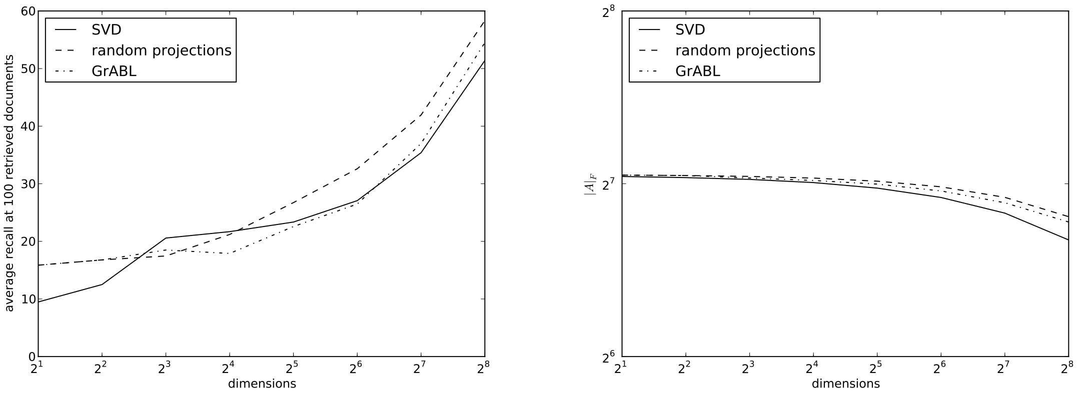

Figure 6.19: Document-level averages for random projection and GrABL approximations of LSI (left) and low-rank approximation norm errors (right) for the NPL collection.

from the trailing part of the spectrum. Random projections are good at excluding trailing parts of the spectrum. The inclusion of trailing spectral features may be remedied by use of a longer, non-minimal Krylov subspace followed by Ritz vector truncation. Use of subspaces as query expanders implies that there is an optimal subspace for query expansion which may be distinct from the norm-optimal projection space for some k; simply projecting the query into the Frobenius norm-optimal rank k projection may not be query-expansion optimal for some k. Simply producing a better approximation of the truncated SVD for rank k may not translate into more relevant search results. Therefore, the relationship between approximation error and query precision is not as clear as it is for applications for which there is a close relationship between Frobenius norm and realized error, such as for the previous road network visualization problem.

6.5 Discussion

For all three data sets, GrABL produces low-rank approximations that approach the accuracy of the random projection method. For some dimensions, GrABL produces slightly *better* approximations than the random projection method. Single-vector Lanczos method lags behind for small dimensions, but does provide equivalent approximations for higher dimensions. Both single-vector Lanczos and GrABL will require stabilization at some point in its iteration to prevent loss of orthogonality of the Lanczos basis. The random projection method never lost any orthogonality in any experiment.

Though loss of orthogonality is an issue, selective reorthogonalization may be applied to stabilize the Lanczos process [57] with costs less than full reorthogonalization.

The approximation errors from GrABL-produced $\hat{A}^{(k)}$s are attributable to good eigenvalue approximations. GrABL excludes Ritz pairs that approximate trailing eigenpairs, as does the random projection algorithm. This quality of GrABL may be accredited to the start block. GrABL incorporates the random projection algorithm in its initial inflation stage; therefore, it always begins with a block of Ritz vectors that have small angles with trailing eigenvectors of A. This start block also excludes null eigenvectors of A. GrABL may have better stability than single-vector Lanczos in some but not all cases. The cause for better stability is not clear, but may be due to the shortening of the Krylov sequence induced by the start block that is later deflated. Adding power iteration refinements to GrABL may or may not improve stability; for the Yale face experiment, stability was improved with refinements, but for the Colorado road graph Laplacian, stability was equivalent. For the Bag of Words data, adding refinements worsened orthogonality loss for some dimensions. ABLE also showed varying stability histories; it lost orthogonality quickly in the Yale experiment, was more stable at large dimensions than all others for the Colorado experiment, but lost orthogonality quickly in the Enron corpus experiment. Block size history is influential; if many of the dimensions ABLE produces are due to inflation rather than iteration, then ABLE will be have better stability but worse approximation. Inflation may help to postpone loss of orthogonality, but cannot prevent convergence of the lead eigenpair from corrupting the orthogonality of the Lanczos basis.

We have observed that the power parameter p in GrABL is important for obtaining low-rank approximation errors that are smaller than random projections. A key advantage of GrABL is that it only performs extra sparse matrix-vector products due to power iteration for large leading eigenvalues; thereby computational costs are reduced when sparse matrix-vector products are expensive. Adding extra power iteration can improve the performance of GrABL substantially. When the spectrum is favorable, as with the Yale face data, GrABL with refinements performs nearly as well as the random projection algorithm, but has far lower computational complexity and requires fewer passes through the data to perform extra power iteration. However, the experiments with the Colorado road network graph and Enron email corpus show that when the eigengaps of the spectrum shrink more slowly, then the error difference between GrABL with refinements and the comparable random projection method on A^p grows faster. Nevertheless, adding refinements always produced better approximations than single-vector Lanczos, random projections on A or GrABL without refinements.

Choosing the value of η may appear to have been done somewhat arbitrarily; however, it was chosen to produce a comparison against GrABL that had the most equivalence. The lead eigengap was scaled in an attempt to include those eigenvalues the are to the left of point t' in Figure 5.2. ABLE is sensitive to the choice of η; small variation in η resulted in different inflation behavior. For example, using the Yale face data and $\eta = 14$, ABLE did not inflate at all to produce a 50-dimensional Krylov subspace, rendering it identical to single-vector Lanczos. Using $\eta = 17$ caused ABLE to inflate as much as possible at every iteration but the last to produce a 50-dimensional subspace. Similar sensitivity was observed with the Colorado road network Laplacian and Enron email corpus experiments.

In terms of compute-time, GrABL enjoys a theoretical advantage over the random projection method due to smaller block sizes. For example, to compute a 20-dimensional subspace on the Yale face data, random projections would require costs proportional to $400n$, where n is the size of the covariance matrix. GrABL would require $(16+16)n$: 16 for the initial block size of 4 and a further 16 for the single-vector iterations. These theoretic advantages may not be completely realized, as block methods may better exploit locality and parallelism. The unavoidable costs of stabilization are also not considered. Nevertheless, one may expect GrABL to be meaningfully faster than the random projection.

Several factors influence the choice of algorithm between single-vector Lanczos, GrABL and the random projection method. We focus on the case in which sparse matrix-vector products are not prohibitively expensive due to the sparsity of A. In these cases, GrABL may use refinement without cost penalties, and obtain a non-trivial cost advantage over the random projection method, while maintaining an approximation and possible stability advantage over single-vector Lanczos. Nevertheless, when sparse matrix-vector products are the dominant cost, GrABL enjoys an advantage over random projections with the same power parameter due to the smaller number of sparse matrix-vector products. The approximation advantage over single-vector Lanczos diminishes when many single-vector

Lanczos iterations are performed after deflation. The approximations obtained with GrABL using refinement are closest to the random projection method when there are only a few iterations after deflation. These two observations suggest that GrABL is well suited for reduction to intermediate dimensions between t' and t in Figure 5.2.

6.6 Conclusion

GrABL is intended to offer a middle ground between the random projection algorithm and minimal Krylov subspace methods. The random projection algorithm produces good eigenvalue approximations, but at a premium expense; smaller block sizes reduce the cost per iteration, but at the expense of admitting Ritz pairs that approximate trailing eigenpairs into the subspace. The overarching motivation behind GrABL is to increase block size until it is no longer advantageous, and deflate aggressively thereafter to minimize the cost per iteration. GrABL also is intended to provide adaptive block size capability similar to ABLE, but without the requirement for beforehand knowledge of the spectrum of the input matrix. With appropriate deflation, GrABL has an overall lower FLOP count than the random projection algorithm.

GrABL is specialized for low-rank matrix approximations and related generic dimension reduction problems for which the optimal truncated eigenvalue decomposition is prohibitively expensive. Problems with many tightly clustered leading eigenvalues or a flat spectrum are not appropriate for GrABL. Yet many low-rank approximation and generic dimension reduction problems typically have a small number of well-separated leading eigenvalues. In these cases, considerable advantages may are realizable from employing block Krylov subspaces rather than single-vector Krylov subspaces to approximate the well-separated eigenvalues. Maximal block sizes used in the random projection algorithm generally give satisfactory results, though their computational costs are high due to costs quadratic in the block size. Use of a smaller block size would mitigate these costs, though the best small block size may not be known *a priori*. When matrix-vector products are inexpensive, GrABL can produce low-rank approximations that are competitive with the random projection method but less expensive to generate, while still being higher quality than a minimal Krylov subspace generated with a single-vector Lanczos approach.

Chapter 7

Inner iteration for minimal Krylov subspaces

Minimal Krylov subspaces offer compute time advantages over eigenvector spaces for a wide variety of low-rank approximation and dimension reduction tasks, such as PCA, POD and LSI. All of these techniques use a truncated spectral decomposition of a positive-definite Gram matrix to effect a reduction in dimensions. Minimal Krylov subspace projections are well-suited as an alternative to the truncated spectral decomposition for these problems, as the eigenvalues and eigenvectors used in the truncated spectral decomposition are also those eigenvalues and eigenvectors that converge first in a Krylov subspace. These eigenvalues tend to be sufficiently separated from their neighbors as to converge with satisfactory speed in a Krylov subspace. However, satisfactory convergence does not necessarily imply that convergence is complete in any minimal Krylov subspace, or that the eigenvalue approximations given by a Krylov subspace are sufficiently converged as to produce a satisfactory low-rank matrix approximation. Even with leading eigenvalue problems, utilization of some acceleration methods may produce better low-rank approximation results than a simple minimal Krylov subspace. Simple minimal Krylov subspaces may be attractive in some cases due to their storage attributes; only k Lanczos basis vectors need be stored. In fact, in some cases, it is not possible to store any basis larger than that for a minimal Krylov subspace. We focus on these cases, and finding ways to produce better Krylov subspaces when storage for basis vectors is constrained.

As minimal Krylov subspaces represent the computationally least expensive Krylov subspace possible, an acceleration method applied to them should also be proportionally inexpensive; it would be useless to apply an expensive eigenvalue acceleration method to a minimal Krylov subspace and thereby ruin its computational advantages. Shift-and-invert preconditioning methods do exist for Krylov subspace methods for the eigenproblem; however, they are computational over-kill for the types of leading eigenvalue problems typically encountered in the aforementioned analysis methods. Shift-and-invert requires factorization of the input matrix, which is expensive in general. The straightforward alternative of simply extending the Krylov subspace likewise introduces additional computational costs beyond the minimal Krylov subspace both in terms of time costs and storage costs. Indeed, there may be cases for which there may simply not be sufficient memory to compute a non-minimal Krylov subspace.

In contrast to shift-and-invert preconditioning, inner iteration may be comparatively inexpensive. Though not as powerful as shift-and-invert preconditioning, inner power iteration is also not as costly. Rather than requiring the inversion of the input matrix, either implicitly or explicitly, inner power iteration only requires extra matrix-vector products. When the input matrix is sparse, these matrix-vector products may be inexpensive compared to the dense linear algebra operations needed to form the Krylov subspace and maintain orthogonality of its basis vectors. Inner power iteration transforms the Krylov subspace from $\mathscr{K}_i(A, x_0)$ to $\mathscr{K}_i(A^p, x_0)$. Local gaps $\lambda_i - \lambda_{i+1}$ between eigenvalues are expanded, resulting in faster convergence of leading eigenvalues, especially when they are already well-separated. An alternate interpretation is that $\mathscr{K}_{i/p}(A^p, x_0)$ is an approximation to the longer subspace $\mathscr{K}_i(A, x_0)$. Both interpretations lend insight into the improved performance of Krylov subspace projections with inner iteration for generation of minimal Krylov subspaces.

7.1 Background

We consider applications of Krylov subspaces as approximations to a truncated eigenspace for low-rank approximation problems. The low-rank approximation problem may range from PCA or PCA-like applications to regularization of ill-posed problems. These classes of problems are optimally solved with projection into the truncated eigenspace; the truncated k-dimensional eigenvector projection not only produces the rank-k approximation with largest Frobenius norm, but also has filtering properties as well. For example, PCA not only finds a Frobenius norm-optimal approximation, but also appears to separate global structure from local deviations or noise. Thus, leading eigenvectors used for a PCA approximation correspond to global components, and the trailing eigenvectors excluded from a PCA approximation represent noisy, local features. Substitution of a minimal Krylov subspace for a truncated eigenspace results in approximations with smaller Frobenius norm, and introduction of noisy components into the approximation. Minimal Krylov subspaces will produce eigenvalue approximations that have not converged to any particular eigenvalue. Some important eigenvalues responsible for a large proportion of the Frobenius norm may not be completely converged in a minimal Krylov subspace, even when the important eigenvalues are well-separated. The most straightforward illustration of the problems caused by non-convergence of eigenvalues in a spectral decomposition is low-rank matrix approximation for regularization of a positive definite linear system. In this application of a truncated spectral decomposition, the linear system $Ax = b$ is replaced by a k-rank approximation $\hat{A}^{(k)}x = b$, in which all eigenvalues of A less than ϵ are set to 0. The goal of regularization is to limit the norm of the solution $\|x\|$ when such large-normed solutions represent physically unlikely solutions. This matrix regularization problem is solved with the truncated spectral decomposition with $\hat{A}^{(k)} = U_k \Lambda_k U_k^T$, with $\Lambda_k = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_k)$ where $\lambda_i \geq \epsilon$ for $i \leq k$ and $U_k = [u_1\ u_2\ \ldots\ u_k]$. The removal of small eigenvalues has a substantial impact on how well the low-rank approximation regularizes the problem. The exclusion of eigenvalues less than ϵ assures that there is no b such that $\|\hat{A}^{(k)\ -1}b\| > \epsilon^{-1}$. Replacement of the eigenspace spanned by columns of U_k with $\mathcal{K}_k(A, b)$ will result in some eigenvalues of $\hat{A}^{(k)}$ possibly having magnitude smaller than ϵ, and the resulting violation of the guarantee $\|\hat{A}^{(k)\ -1}b\| \leq \epsilon^{-1}$.

We recall that minimal Krylov subspaces have a particular advantage owed to their minimally-short nature. No other Krylov subspace is less expensive to produce. Therefore, they represent a compute-cost lower bounds on eigenspace approximation with Krylov subspaces. Choice of an acceleration method must be informed by the additional compute costs introduced. When non-convergence of spectral components is a problem, it may be solved with a non-minimal Krylov subspace followed by Ritz vector truncation. Memory constraints may not permit generation of a non-minimal Krylov subspace; storage of more than k Lanczos vectors may simply not be possible. For example, latent semantic analysis may require hundreds of dimensions; storing 100 dimensions of even a corpus with 10^6 documents or terms will require on the order 10^8 bytes, roughly 762 MB if double-precision floating point numbers are used. This motivates investigation of methods to accelerate convergence of the leading spectral components required for low-rank approximation and exclusion of the trailing spectral components representing noise. Methods to target a neighborhood of the input matrix spectrum have been previously studied, but present unwarranted expense for the minimal Krylov subspace problem when leading spectral components are required. Use of shift-and-invert preconditioning to accelerate convergence of leading eigenvalues imposes compute costs that are generally prohibitive; the input matrix must be inverted. Even when the input matrix is sparse and matrix inversion is not cubic in complexity, the structure of the input matrix may lead to non-trivial fill-in or inversion costs. Therefore, we must choose an acceleration method that is computationally inexpensive, especially relative to minimal Krylov subspaces.

Use of power iteration to expand local gaps and accelerate convergence in a projective eigenvalue method was proposed in [36], but is also comparable to inner iteration used in flexible GMRES [69] for preconditioning linear systems. In flexible GMRES, the preconditioner is allowed to change from iteration to iteration, which allows for an iterative solver to be used as a preconditioner instead of a static approach such as a ILU or SSOR preconditioner. Any iterative method could be used to approximate A^{-1} in flexible GMRES, even GMRES itself. We note that the linear systems preconditioning problem is different from the eigenvalue preconditioning problem; for linear systems, an effective preconditioner is a matrix M^{-1} that approximates A^{-1} and such that the condition number $\kappa(M^{-1}A)$ is small. Inner power iteration is similar to using p steps of conjugate gradient as an inner iteration in GMRES, but the Krylov subspaces generated differ. Inner power iteration produces $\mathcal{K}_i(A^p, x_0)$ whereas flexible GMRES produces $\text{span}\{x_0, q_1(A)x_0, q_2(A)x_0\}$ for polynomials q_i of order ip. When the input matrix is symmetric, inner power iteration with $p = 2$ produces the same Krylov subspace as the normal equations, as both solve the equation $A^T Ax = A^T b$. Though power iteration is a eigenvalue method in its own right, its slow convergence for single vectors renders it ineffective for practical use. Nevertheless, combination of inner power iteration with Lanczos outer iteration produces an eigenvalue approximation method that approximates leading spectral components more successfully than ordinary Lanczos iteration, without the need for expensive and complicated implicit restart methods. First, we review power iteration methods and Krylov subspace methods for the eigenproblem. Analysis of the convergence of eigenvalues in Krylov subspaces indicates the potential effects of inner power iteration; we then present the Lanczos algorithm modified with inner power iteration.

7.2 Inner iteration with Krylov subspace methods for the eigenproblem

Krylov subspaces are deeply related to power iteration; in fact, a Krylov is the very space spanned by the successive products of power iteration. Recall that given a matrix A and a suitable vector x_0, a Krylov subspace is given by

$$\mathcal{K}_i(A, x_0) = \text{span}\{x_0, Ax_0, A^2x_0, \ldots, A^{i-1}x_0\}. \tag{7.1}$$

For minimal Krylov subspaces, substitution of the input matrix A with A^p will increase the leading eigengaps. This substitution improves the quality of the leading eigenvalue approximations in the subspace produced. The direct formation of A^p may be expensive in terms of compute time, and may further lead to loss of sparsity. Instead of explicit formation of A^p as an input matrix to the Lanczos routine, A^p may be formed via an inner iteration. Such an approach is similar to other inner iteration preconditioning methods for Krylov subspaces, which solve a linear system to affect a shift-and-invert preconditioning of A. The resulting algorithm is presented in Algorithm 11.

Unfortunately, adding inner power iteration causes some difficulty for use of the Lanczos method for eigenvalue approximation. Given the input matrix A^p and the output matrices Q and T from the Lanczos algorithm, we have

$$Q^T A^p Q = T \tag{7.2}$$

and a Ritz value $\theta_i = v_i^T A^p v_i$ for some Ritz vector v_i approximates an eigenvalue of A^p from below. However, it is not the case that $\sqrt[p]{\theta_i} = v_i^T A v_i$ due to round-off error. Therefore, once augmented with inner iteration, the Lanczos method cannot be used to approximate eigenvalues of A directly as lower bounds. Instead, the input matrix must be projected into the Krylov subspace after Lanczos iteration to obtain lower eigenvalue approximations. The expense of this second projection is not as severe as it may seem if one compares the approach to Algorithm 3. Both approaches first generate an

Algorithm 11 Lanczos with inner power iteration

Require: *a priori* chosen start vector q_1, power p

```
$r \leftarrow q_1$
**for** $i = 1 \rightarrow p$ **do**
    $r \leftarrow Ar$
**end for**
**for** $j = 1 \rightarrow k$ **do**
    $\alpha_j \leftarrow q_j^T r$
    $r \leftarrow r - q_j \alpha_j$
    $\beta_{j+1} = \|q_{j+1}\|$
    $r \leftarrow q_1$
    **for** $i = 1 \rightarrow p$ **do**
        $r \leftarrow Ar$
    **end for**
    $r \leftarrow r - q_j \beta_{j+1}$
**end for**
**return** $[q_1\ q_2\ \ldots], \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \alpha_3 & \beta_4 & \\ & & \ddots & \ddots & \ddots \end{bmatrix}$
```

orthonormal basis for a k-dimensional space using k sparse matrix-vector products and dense linear algebra operations using no more than $O(nk^2)$ FLOPS and project the matrix onto the orthonormal basis. The complexity of using the basis from k iterations of the Lanczos algorithm is no worse than the random projection method. It is likely that when selective reorthogonalization is used, then the Lanczos method will likely require less than $O(nk^2)$ FLOPS to generate its projection matrix, and will be faster than the random projection method.

7.3 Convergence of eigenvalues

Adjustment of the input matrix spectrum influences convergence of eigenvalues. Often, Krylov subspace projections offer fast convergence of leading eigenvalues; for many low-rank approximation problems, leading eigenvalues are well-separated and converge with satisfactory rapidity. Nevertheless, inner power iteration can accelerate convergence of leading eigenvalues. The mechanism by which inner iteration accelerates convergence is intuitive; it expands the local eigenvalue separation that drives eigenvalue convergence in Krylov subspaces. An alternate intuition is that the Krylov subspace $\mathcal{K}_i(A^p, x_0)$ approximates the larger Krylov subspace $\mathcal{K}_{ip}(A, x_0)$. Both ideas are illuminating, but the latter of the two leads to a more elegant formal result. Indeed, there is a relationship between the optimal minimizing polynomials for $\|q(A)x_0\|$ for A^p and A; this relationship between polynomials yields worst-cases comparison between eigenvalue estimates for $\mathcal{K}_i(A^p, x_0)$ and $\mathcal{K}_i(A, x_0)$. We begin by generalizing the worst-case bounds from [67] to compare the optimal degree-i polynomial to the non-optimal degree ip polynomial. This difference leads directly to a restatement of the Saad bounds [67].

The context of these re-derivation of bounds for Krylov subspace approximations of the form $\mathcal{K}_i(A, X_0)$ lies in our original low-rank approximation task. This context motivates bounds that allow us to compare the convergence of eigenvalues in $\mathcal{K}_{i/p}(A^p, x_0)$ to convergence of eigenvalues in $\mathcal{K}_i(A, x_0)$ so that we may quantify the loss of accuracy when substituting a power iteration-augmented minimal Krylov subspace for a non-minimal one.

Recall Theorem 2 from [67], where the error in eigenvalue approximation is bounded as

$$0 \leq \lambda_i - \lambda_i^{(n)} \leq (\lambda_i - \lambda_{\inf}) \left(\frac{K_i^{(n)} \| \phi_i - \hat{x}_i \|}{T_{n-i}(\hat{\gamma}_i)} \right)^2 \tag{7.3}$$

where

$$K_i^{(n)} = \prod_{\lambda_j^{(n)} \in \sigma_i^{(n)}} \frac{\lambda_i^{(n)} - \lambda_{\inf}}{\lambda_i^{(n)} - \lambda_i} \text{ and } K_1^{(n)} = 1$$

with $\sigma_i^{(n)}$ is the set of the first $i-1$ approximate eigenvalues, $T_j(\cdot)$ is the Chebyshev polynomial of the first kind with order j, and

$$\hat{\gamma}_i = 1 + \frac{2(\lambda_i - \lambda_{i+1})}{\lambda_{i+1} - \lambda_{\inf}}.$$

The form of these bounds is a result of the relationship between the Krylov subspace $\mathcal{K}_i(A, x_0)$ and the set of polynomials of degree no greater than i. In $\mathcal{K}_i(A, x_0)$, one may construct a degree $i-1$ polynomial $q(x)$ that is optimal for minimizing $\|q(A)x_0\|$. Furthermore, the roots of this polynomial $q(x)$ are the approximate eigenvalues of the projection of A into $\mathcal{K}_i(A, x_0)$ [66, 67, 82]. A consequence of this property of minimization of $\|q(A)x_0\|$ is that no other polynomial $f(x)$ of degree $i-1$ results in a smaller $\|f(A)x_0\|$ than $q(x)$. The Chebyshev polynomial in the denominator of (7.3) is a result of this optimality. Thus, we may expect that changes to the structure of the Krylov subspace and resulting optimal polynomial $q(x)$ will result in a change of the Chebyshev polynomial. The Krylov subspace $\mathcal{K}_i(A, x_0)$ admits an optimal polynomial $q(x)$ of degree no greater than $i-1$ for minimization of $\|q(A)x_0\|$ and the Krylov subspace $\mathcal{K}_i(A^p, x_0)$ admits an optimal polynomial $r(x)$ of degree no greater than $i-1$ for minimization of $\|r(A^p)x_0\|$. If $\mathscr{P}^{(i-1)}$ is the set of all polynomials of degree no larger than $i-1$, then we have both $q(x) \in \mathscr{P}^{(i-1)}$ and $r(x) \in \mathscr{P}^{(i-1)}$. There is then a apparent relationship between $q(x)$ and $r(x)$ that we exploit in the following theorem to show how the bounds for (7.3) change with inner power iteration.

Theorem 11. *Consider a Krylov subspace $\mathcal{K}_n(A, X_0)$ with block size r. Let λ_i be an eigenvalue of the matrix A with associated eigenvector ϕ_i with $\|\phi_i\| = 1$. Let p be a natural number. Assume that the vectors $\pi_1(A)\phi_j$ are linearly independent where $\pi_i(A)$ is the orthogonal projection on E_1, the space spanned by the initial start vectors of the Krylov subspace. Let θ_i be the approximation of λ_i^p from the* *projection of A^p into $\mathscr{K}_n(A^p, x_0)$. Then the error of the approximate eigenvalue $\lambda_i^{(n)}$ as generated by the projection of A into $\mathscr{K}_n(A^p, x_0)$ for approximating λ_i is bounded as*

$$0 \le \lambda_i - \lambda_i^{(n)} \le (\lambda_i - \lambda_{\text{inf}}) \left(\frac{K_i^{(n,p)} \tan^2 \theta(\phi_i, x_0)}{T_{n-i}(\delta_i)} \right)^2 \tag{7.4}$$

where

$$K_i^{(n,p)} = \prod_{\theta_j \in \sigma_i^{(n)}} \frac{\sqrt[p]{\theta_j} - \lambda_{\text{inf}}}{\sqrt[p]{\theta_j} - \lambda_i} \text{ and } K_1^{(n)} = 1$$

with $\sigma_i^{(n)}$ is the set of the first $i-1$ approximate eigenvalues of A^p, $T_j(\cdot)$ is the Chebyshev polynomial of the first kind of order j and

$$\delta_i = 1 + \frac{2(\lambda_i^p - \lambda_{i+1}^p)}{\lambda_{i+1}^p - \lambda_{\text{inf}}^p}.$$

Proof. We proceed in the same manner as Theorem 2 in [67]. By the Courant characterization of eigenvalues of symmetric operators, we have

$$\lambda_i^{(n)} = \max_{u \in F_i^{(n)}} \frac{u^T A^p u}{u^T u}. \tag{7.5}$$

Then

$$0 \le \lambda_i - \lambda_i^{(n)} = \min_{u \in F_i^{(n)}} \frac{u^T (\lambda_i I - A^p) u}{u^T u}. \tag{7.6}$$

There is an optimal polynomial $q(x)$ of degree no greater than $n-1$ that minimizes $\|q(A^p)x_0\|$. Then $u = q(A^p)x_0 = \sum_{j=1}^M a_j q(\lambda_j^p)\phi_j$. Consider a transform function operating on polynomials $f(x) = \sum_{j=0}^k a_j x^j$, with

$$m(f)(x) = \sum_{j=1}^{n-1} a_j x^{pj}. \tag{7.7}$$

Then $q(A^p) = m(q)(A)$. Therefore $m(q)(x)$ also minimizes $\|m(q)(A)x_0\|$ and $u = m(q)(A)x_0$. Note that though $q(x) \in \mathscr{P}^{(n-1)}$ but $m(q)(x) \notin \mathscr{P}^{(n-1)}$. The mapping $m(f)(x)$ in fact defines a new set of polynomials $\mathscr{Q}^{(n-1)} = \{m(g) \| g \in \mathscr{P}^{(n-1)}\}$ for any set of polynomials $\mathscr{P}^{(n-1)}$ where all polynomials in $\mathscr{Q}^{(n-1)}$ have degree no greater than $p(n-1)$ and have no nonzero coefficients for any power of x not integer divisible by p. Let $r(x) = m(q)(x)$. Then

$$\frac{u^T(\lambda_i - A)u}{u^T u} = \frac{1}{\|u\|^2}\left(\sum_{j=1}^{i}(\lambda_i - \lambda_j)r(\lambda_j)^2 a_j^2 + \sum_{j=i+1}^{M}(\lambda_i - \lambda_j)r(\lambda_j)^2 a_j^2\right). \tag{7.8}$$

and

$$\frac{u^T(\lambda_i I - A)u}{u^T u} \le (\lambda_i - \lambda_{\inf})\frac{\sum_{j=i}^{M}(\lambda_i - \lambda_j)r(\lambda_j)^2 a_j^2}{\sum_{j=1}^{M} r(\lambda_j)^2 a_j^2}. \tag{7.9}$$

Since $r(x) = \sum_{i=1}^{M} a_i x^{pi} = q(x^p)$, $q(x^p) = 0 \Longrightarrow r(x) = 0$. From Lemma 3 in [67], we have $q(\theta_j) = 0$ for $1 \le j < i$, so $r(\sqrt[p]{\theta_j}) = 0$ for $1 \le j < i$, and $r(x)$ has $n-1$ real roots. Then $r(x)$ may be written as $r(x) = f(x)\prod_{j=1}^{i-1}(x - \sqrt[p]{\theta_j})$ where $f(x) = \sum_{j=0}^{n-i} a_i x^{ip}$ and $f(x) \in \mathscr{Q}^{(n-i)}$. Following the derivations in [67], we have

$$\frac{u^T(\lambda_i I - A)u}{u^T u} \le (\lambda_i - \lambda_{\inf})\sum_{j=i+1}^{M}\frac{(\lambda_j - \sqrt[p]{\theta_1})^2 \ldots (\lambda_j - \sqrt[p]{\theta_{i-1}})^2 f(\lambda_j)^2 a_j^2}{(\lambda_i - \sqrt[p]{\theta_1})^2 \ldots (\lambda_i - \sqrt[p]{\theta_{i-1}})^2 f(\lambda_i)^2 a_i^2}, \tag{7.10}$$

and noting that $\sqrt[p]{\theta_j} \ge \lambda_{\inf}$ for any j leads to

$$\frac{u^T(\lambda_i I - A)u}{u^T u} \le (\lambda_i - \lambda_{\inf})\prod_{j=1}^{i-1}\frac{(\sqrt[p]{\theta_j} - \lambda_{\inf})^2}{(\sqrt[p]{\theta_j} - \lambda_i)^2}\cdot\sum_{j=i+1}^{M}\frac{a_j f(\lambda_j)^2}{a_i f(\lambda_i)^2}. \tag{7.11}$$

As $r(x) = \prod_{j=1}^{n-1}(x - \sqrt[p]{\theta_j})$ is the polynomial in $\mathscr{Q}^{(n-1)} = \left\{\sum_{j=1}^{i-1} a_i x^{pi} | a_i \in \mathbb{R}\right\}$ that minimizes (7.8), it is implied that $f(x)$ minimizes $\sum_{j=i+1}^{M}(a_j^2 f(\lambda_j)^2)/(a_i^2 f(\lambda_i)^2)$. Thus

$$\min_{u \in F^{(n)}}\frac{u^T(\lambda_i I - A)u}{u^T u} \le (\lambda_i - \lambda_{\inf})(K_i^{(n,p)})^2 \cdot \min_{f \in \mathscr{Q}^{(n-i)}}\sum_{j=i+1}^{M}\frac{a_j f(\lambda_j)^2}{a_i f(\lambda_i)^2}. \tag{7.12}$$

We may separate terms in the right-most side of (7.12) and obtain

$$\min_{g \in \mathscr{Q}^{(n-i)}}\sum_{j=i+1}^{M}\frac{a_j f(\lambda_j)^2}{a_i f(\lambda_i)^2} \le \left(\sum_{j=i+1}^{M}\frac{a_j^2}{a_i^2}\right)\min_{f \in \mathscr{Q}^{(n-i)}}\max_{j \ge i+1}\left|\frac{f(\lambda_j)}{f(\lambda_i)}\right|^2. \tag{7.13}$$

Noting that $T_{n-i}(\delta) \in \mathscr{Q}^{(n-i)}$ for $\delta = 1 + 2(x^p - \lambda_{i+1}^p)/(\lambda_{i+1}^p - \lambda_{\inf}^p)$, and $\sum_{j=i+1}^{M} a_j^2/a_i^2 \le \tan^2\theta(\phi_i, x_0)$, we get

$$\left(\sum_{j=i+1}^{M}\frac{a_j^2}{a_i^2}\right)\min_{f \in \mathscr{Q}^{(n-i)}}\max_{j \ge i+1}\left|\frac{f(\lambda_j)}{f(\lambda_i)}\right|^2 \le \tan^2\theta(\phi_i, x_0)\cdot\frac{1}{T_{n-i}^2(\delta_i)}. \tag{7.14}$$

Combining (7.12) and (7.14) yields

$$0 \le \lambda_i - \lambda_i^{(n)} \le (\lambda_i - \lambda_{\text{inf}}) \left(\frac{K_i^{(n,p)} \tan^2 \theta(\phi_i, x_0)}{T_{n-i}^2(\delta_i)} \right)^2 \tag{7.15}$$

which completes the proof. □

Note that the value in (7.4) differs from that obtained by simply substituting λ_i^p into the bounds from [67], which would be

$$0 \le \lambda_i^p - \lambda_i^{(n)} \le (\lambda_i^p - \lambda_{\text{inf}}^p) \left(\frac{K_i^{(n,1)} \tan^2 \theta(\phi_i, x_0)}{T_{n-i}^2(\delta_i)} \right)^2 . \tag{7.16}$$

A notable consequence of this theorem is that eigenvalues of A converge in $\mathcal{K}_i(A^p, x_0)$ at least as fast as eigenvalues of A^p. Since local gaps in the spectrum are expanded by the power p, we may expect δ_i to be larger than γ_i, resulting in faster convergence, especially with successive iterations.

7.4 Examples

To show the effects of inner power iteration on the theoretical asymptotic convergence and *a posteriori* convergence of eigenvalue approximations in Krylov subspaces, we present numerical results. We use two matrices in the experiment: both exhibit the spectra typical of low-rank approximation problems encountered in PCA-like dimension reduction problems. The first problem arises form structural analysis and involves a stiffness matrix. The second experiment is on a much larger matrix derived from a information retrieval problem.

7.4.1 nos5 stiffness matrix

The nos5 stiffness matrix from the Harwell-Boeing matrix collection [20] arises from finite-elements analysis of a building. Buildings may typically be treated as stiff structures with negligible damping; therefore, natural frequencies may be calculated from the stiffness matrix alone. The most important frequencies of interest are those eigenvalues of A with *smallest* magnitude, but if A is easily inverted — for example, with a sparse Cholesky factorization — then A^{-1} may be used, and trailing

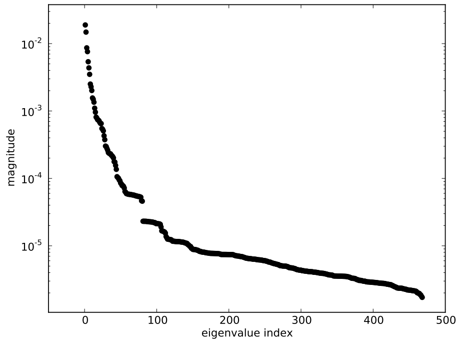

Figure 7.1: Spectrum of the inverted nos5 stiffness matrix.

eigenvalue	1	2	3
$\mathscr{K}_{12}(A,x_0)$	1.53×10^{-10}	7.1×10^{-13}	2.04×10^{-3}
$\mathscr{K}_3(A^4,x_0)$	1.28×10^{-4}	1.4×10^{-2}	2.52×10^{2}
$\mathscr{K}_3(A,x_0)$	1.14×10^{-2}	1.98	3.11×10^{4}

Table 7.1: Bounds for eigenvalue approximations using $\mathscr{K}_{12}(A,x_0)$, $\mathscr{K}_3(A,x_0)$, and $\mathscr{K}_3(A^4,x_0)$ with random x_0 for the inverted nos5 matrix.

eigenvalues of A become leading eigenvalues of A^{-1}.

To study the convergence of eigenvalues of the inverted nos5 matrix, we generated a random start vector x_0, and examine the bounds predicted by Theorem 11 and actual eigenvalue errors for the first 3 eigenvalues of the inverted A in $\mathscr{K}_{12}(A,x_0)$, the minimal $\mathscr{K}_3(A^4,x_0)$ and, for a baseline, the minimal $\mathscr{K}_3(A,x_0)$. The worst-case bounds are presented in Table 7.1, and the errors are presented in Table 7.2. It is important to note that the tightness of the bounds should be considered in terms of relative error; that is, the worst-case error for the leading eigenvalue approximation in $\mathscr{K}_{12}(A,x_0)$ is 1.54×10^{-10}, which is 8.15×10^{-8} percent error.

It is clear that the worst-case bounds are pessimistic in these cases; notably for λ_3. The addition of inner power iteration does not obtain bounds as tight as the non-minimal Krylov subspace, but

eigenvalue	1	2	3
$\mathscr{K}_{12}(A,x_0)$	$-3.47\times10^{-18*}$	$5.2\times10^{-18*}$	7.01×10^{-12}
$\mathscr{K}_3(A^4,x_0)$	1.59×10^{-5}	1.47×10^{-4}	7.75×10^{-3}
$\mathscr{K}_3(A,x_0)$	2.06×10^{-3}	5.78×10^{-3}	8.51×10^{-3}

Table 7.2: Eigenvalue approximation errors using $\mathscr{K}_{20}(A,x_0)$, $\mathscr{K}_5(A,x_0)$, and $\mathscr{K}_5(A^4,x_0)$ with random x_0 for the inverted nos5 matrix. * indicates a value outside of machine precision.

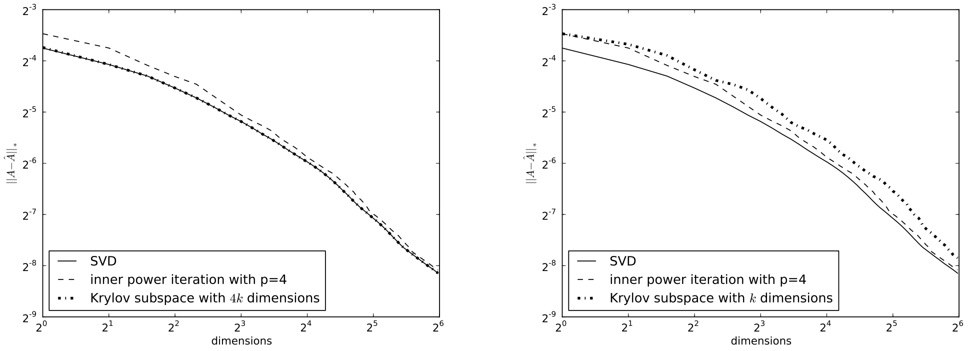

Figure 7.2: Approximation error $\|A - \hat{A}^{(k)}\|_*$ for inverted nos5 stiffness matrix A (left). The approximation error using the baseline minimal Krylov subspace $\mathscr{K}_k(A,x_0)$ is shown on the right.

still represents bounds two orders of magnitude tighter than the minimal Krylov subspace generated without inner iteration. We note that $\mathscr{K}_3(A^4,x_0)$ obtains less than 1% relative error for λ_1 and λ_2. Though $\mathscr{K}_{12}(A,x_0)$ has less than 10^{-9} percent error for any of λ_1, λ_2 or λ_3, this extra tightness is only significant for λ_3, which $\mathscr{K}_3(A^4,x_0)$ approximates poorly. By way of comparison, $\mathscr{K}_3(A,x_0)$ has at least 10% error for all eigenvalue approximations. Therefore, the relaxation to the minimal Krylov subspace $\mathscr{K}_3(A^4,x_0)$ will not sacrifice much accuracy when it is computationally less expensive than the non-minimal Krylov subspace.

The impact of the errors may also be viewed in terms of the low-rank approximation error of A in a Krylov subspace. Figure 7.2 shows the low-rank approximation errors of A in the 3 respective Krylov subspaces, with the approximation generated by the truncated SVD for a baseline comparison. Though the non-minimal Krylov subspace is almost indistinguishable from the truncated SVD approximation, the minimal Krylov subspace with inner power iteration becomes ever closer with increasing dimension. This is significant, as the non-minimal Krylov subspace requires four times as much memory.

7.4.2 Enron email corpus experiment

To study the effects of inner power iteration on convergence of eigenvalues for an LSI application, we use the Enron email corpus as collected from the UCI Machine Learning Repository [26]. The matrix was approximated in the Krylov subspace $\mathscr{K}_i(AA^T,x_0)$ for a random x_0. The first 100 eigenvalues of

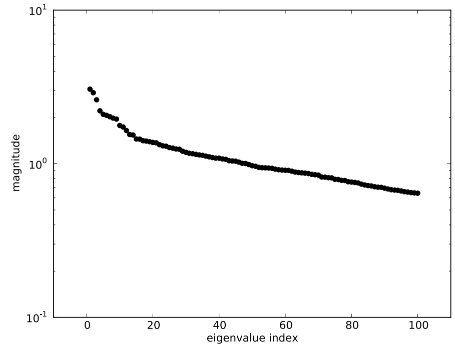

Figure 7.3: Spectrum of the centered covariance matrix of the Enron email data.

eigenvalue	1	2	3	4	5
$\mathscr{K}_{20}(A,x_0)$	1.16×10^{-5}	5.25×10^{-7}	1.81×10^{-5}	7.48	1×10^6
$\mathscr{K}_5(A^4,x_0)$	0.29	3.0	2.93×10^4	2.32×10^6	2.96×10^8
$\mathscr{K}_5(A,x_0)$	12.36	2.54×10^3	9.03×10^7	1.75×10^6	8.21×10^4

Table 7.3: Bounds for eigenvalue approximations using $\mathscr{K}_{20}(A,x_0)$, $\mathscr{K}_5(A,x_0)$, and $\mathscr{K}_5(A^4,x_0)$ with random x_0 for the Enron email data.

the matrix AA^T are shown in Figure 7.3.

We begin by exploring the theoretical bounds for approximation with a non-minimal Krylov subspace without inner iteration to approximation with a minimal one using inner iteration; that is, we compare the rank-k approximation from $\mathscr{K}_{kp}(A,x_0)$ to the approximation from $\mathscr{K}_k(A^p,x_0)$. Note that the length of the non-minimal Krylov subspace grows quickly, even for small p. For comparison, we set $k = 5$ and $p = 4$. The resulting bounds from Theorem 11 are given in Table 7.3. Clearly, the worst-case approximations for $\mathscr{K}_{20}(A,x_0)$ are better than those for $\mathscr{K}_5(A^4,x_0)$. Much of this difference is due to the smaller value obtained from the Chebyshev polynomial in the denominator of (7.4); the order is much larger for a 20-dimensional subspace. Nevertheless, the expansion of local gaps renders the worst-case bounds for the leading eigenvalue are relatively small, and the bounds with inner power iteration are better than the bounds for the minimal Krylov subspace $\mathscr{K}_5(A,x_0)$. The approximation errors are shown in Table 7.4. In this case, the worst-case bounds are rather pessimistic, but the inner power iteration produces better eigenvalue approximations than the baseline Krylov subspace $\mathscr{K}_5(A,x_0)$.

To show the improvement in low-rank approximation using a non-minimal Krylov subspace and

eigenvalue	1	2	3	4	5
$\mathscr{K}_{20}(A,x_0)$	$1.78\times10^{-15*}$	$7.73\times10^{-13*}$	1.67×10^{-11}	2.51×10^{-4}	1.39×10^{-4}
$\mathscr{K}_5(A^4,x_0)$	3.54×10^{-4}	1.66×10^{-2}	3.46×10^{-3}	0.27	1.95
$\mathscr{K}_5(A,x_0)$	2.51×10^{-2}	0.32	1.01	1.75	2.07

Table 7.4: Eigenvalue approximation errors using $\mathscr{K}_{20}(A,x_0)$, $\mathscr{K}_5(A,x_0)$, and $\mathscr{K}_5(A^4,x_0)$ with random x_0 for the Enron email data. * indicates a value outside of machine precision.

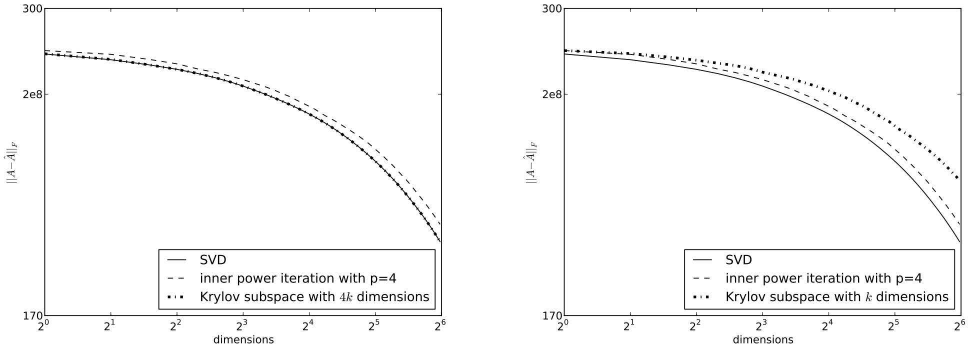

Figure 7.4: Approximation error $\|A-\hat{A}^{(k)}\|_F$ for AA^T using the Enron email data (left). The approximation error using the baseline minimal Krylov subspace $\mathscr{K}_k(A,x_0)$ is shown on the right.

a minimal Krylov subspace augmented with inner power iteration, we generate k-rank approximations using $\mathscr{K}_{kp}(A,x0)$ and $\mathscr{K}_k(A^p,x_0)$ for $p=4$ and $1\le k\le 64$. The approximation error $\|A-\hat{A}^{(k)}\|_F$ is shown in Figure 7.4. The plot includes the error associated with a low-rank approximation generated from the truncated SVD; this serves as a baseline for the Frobenius-norm optimal k-rank approximation. The low-rank approximation using truncated Ritz vector approximation is almost equivalent to the truncated SVD; the inner iteration relaxation lags behind. Nevertheless, the inner iteration Krylov subspace has better eigenvalue approximations than the baseline minimal Krylov subspace when no inner iteration is used.

To show the relative work required to generate the low-rank approximations, we present the theoretical FLOP counts for the Lanczos algorithm to generate both $\mathscr{K}_{kp}(A,x_0)$ and $\mathscr{K}_k(A^p,x_0)$. These costs are presented in Figure 7.5; these costs include partial reorthogonalization costs. We assume that the number of converged Ritz vectors that must be excluded from each new basis vector are asymptotically radical in the number of dimensions of the Krylov subspace. It is further assumed that orthogonality is lost faster with inner power iteration; a conservative estimate of a factor of 4 times more work would be necessary to maintain orthogonality when inner iteration is used. The

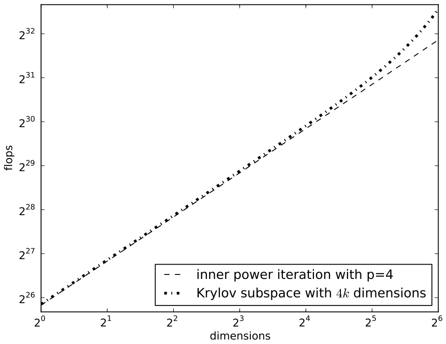

Figure 7.5: Theoretical FLOP count for generating Krylov subspaces using the Lanczos algorithm. It is assumed that inner power iteration will result in 4 times as much work to maintain orthogonality.

input matrix is sparse, but has over 3 million nonzero elements, which is an order of magnitude larger than the number of dimensions of the matrix. Therefore, matrix-vector products dominate in the single-vector Lanczos method. Interestingly, the FLOP counts are roughly equivalent for small dimensions, but the non-minimal Krylov subspace becomes more expensive for larger dimensions. This is due to the costs of maintaining orthogonality; since the cost of maintaining orthogonality is $O(f(x)^3)$, where $f(x)$ is some function of the dimension of the Krylov subspace. As $f(x)$ is assumed to be radical, then the cost of maintaining orthogonality is quadratic in the number of dimensions. As inner power iteration reduces the number of iterations necessary, then less work will be required to exclude already-converged Ritz vectors from the rest of the Krylov subspace. The cost tradeoff is in addition to the storage space tradeoff; storing a basis for $\mathscr{K}_{64}(A^p, x_0)$ requires roughly 19.5 megabytes of storage double precision numbers, but $\mathscr{K}_{256}(A, x_0)$ requires nearly 78 megabytes.

7.5 Conclusion

In some cases, minimal Krylov subspace projections may be used to approximate a truncated singular vectorspace or eigenspace. For many PCA-like applications, the leading eigenpairs needed for the truncated projection converge fairly quickly; however, their convergence may be unsatisfactory due to the minimal length of the Krylov subspace. Non-minimal Krylov subspaces may achieve far better eigenvalue approximations, but the extra storage requirements and iteration costs negate the compute and space advantages of minimal Krylov subspaces.

7. Inner iteration for minimal Krylov subspaces

As local eigenvalue gaps drive convergence in Krylov subspaces, one may add inner power iteration to the classic Lanczos algorithm to change the Krylov subspace from $\mathscr{K}_i(A,x_0)$ to $\mathscr{K}_i(A^p,x_0)$. Thus, one may generate minimal Krylov subspace $\mathscr{K}_k(A^p,x_0) \subset \mathscr{K}_{ip}(A,x_0)$ for the non-minimal Krylov subspace $\mathscr{K}_{ip}(A,x_0)$. The effects of this change may be understood as expansion of local gaps at the leading end of the spectrum, thereby resulting in faster convergence of leading eigenvalues than in $\mathscr{K}_i(A,x_0)$. When the input matrix is sparse in proportion to its dimension, notable compute and space advantages of minimal Krylov subspaces are maintained.

We have only considered simple inner power iteration generating polynomials of the form $q(x) = x^p$. Application of more complicated polynomials of the form $q(x) = a_p x^p + a_{p-1}x^{p-1} + \ldots + a_0$ where not all a_i are zero may lead to even better results and allow more targeting of the spectrum of the input matrix A. The theoretical arguments that show how inner power iteration accelerates the convergence of leading eigenvalues and decelerates the convergence of trailing ones also implies that use of a polynomial $q(x) = x^{1/p}$ will accelerate convergence of trailing eigenvalues at the expense of leading ones. An inner Newton iteration could approximate the product $A^{1/p}x$ for problems that require the smallest eigenvalues of A.

Chapter 8

Shift and invert preconditioning

Some low-rank approximation problems have spectra that cause difficulty for Krylov subspaces; tightly clustered eigenvalues converge slowly. Inner power iterations can improve the quality of minimal Krylov subspace projections for truncated SVD or spectral decomposition approximation; this is due to acceleration of leading eigenpairs. Inner power iteration is inexpensive and appropriate for problems that require approximation of the leading part of the spectrum of the input matrix. Many problems in low-rank approximation require the leading eigenpairs or singular pairs; however, some important problems require the eigenpairs or singular triplets with smallest magnitude rather than the largest magnitude. Inner power iteration is not appropriate for these cases, as it will degrade the quality of the Krylov subspace by suppressing the very eigenpairs or singular triplets the problem requires. One may apply a fractional matrix power $1/p$ to accelerate convergence of small eigenpairs, but this would require Newton iteration to approximate the product $A^{1/p}$ several times over for each iteration. Moreover, the trailing eigenpairs are often so tightly clustered that inner power iteration will not accelerate the convergence of trailing eigenpairs sufficiently.

Shift-and-invert preconditioning [24,51,72] is used in Krylov subspace methods to accelerate convergence of eigenpairs that converge slowly due to tight clustering. Shift-and-invert preconditioning leverages the fact that matrix polynomials in A may be applied to the eigenvalues of A, so that for an arbitrary Hermitian matrix A with spectral decomposition $A = U\Lambda U^T$ and a polynomial $q(x)$

$$q(A) = q(U\Lambda U^T) = Up(\Lambda)U^T. \tag{8.1}$$

This is the same property leveraged by inner power iteration, but instead of using the polynomial $q(x) = x^p$ for some $p \in \mathbb{N}$, shift-and-invert preconditioning uses

$$q(x) = (x-s)^{-1} \tag{8.2}$$

for some fixed shift s. By comparison with inner iteration which only requires extra sparse matrix-vector products, shift-and-invert preconditioning requires an expensive matrix inversion, as the name suggests. When the input matrix A is large and sparse, direct matrix inversion may be accomplished with a sparse LU factorization; this may be expensive in general, but is efficient in some cases. For example, the sparse factorization is inexpensive for matrices arising from circuit simulation and for matrices of some classes of graphs. When the sparse factorization is not expensive, then shift-and-invert preconditioning may be applied to drastically improve the convergence of trailing eigenvalues of the input matrix A. Choosing shifts close to eigenvalues of interest accelerates their convergence. Unfortunately, such a shift is difficult to choose beforehand in many low-rank approximation problems. Choosing shifts adaptively can accelerate convergence; this is the basis for algorithms such as Rayleigh quotient iteration and Jacobi-Davidson [4]. Neither algorithm is suited to low-rank approximation without alteration. Rayleigh quotient iteration may exhibit erratic convergence, and cannot be guaranteed to converge to eigenvalues close to the initial shift s. Jacobi-Davidson converges more regularly, but we have observed that the subspaces it produces may be inferior for low-rank approximation to those produced by Krylov subspace iteration on the shifted and inverted matrix for trailing eigenvalue problems. In order to produce adaptive shifts for Krylov subspaces, we propose a new algorithm that first uses the random projection method in Algorithm 3 to generate a set S of trailing eigenvalue approximations for A, and then forms a matrix polynomial of the form $\prod_{s \in S}(A - sI)^{-1}$ to simultaneously perform all shifts.

The role of the shift parameter s in (8.2) is significant, and allows control over the parts of the spectrum of A that are accelerated. Convergence of eigenvalues in Krylov subspaces is governed by local gaps $\lambda_i - \lambda_{i+1}$; applying a polynomial filter merely changes the gaps to $q(\lambda_i) - q(\lambda_{i+1})$. Therefore, eigenvalues λ_i close to s will have $\lambda_i - s < \epsilon$ for some small ϵ and will have correspondingly large $q(\lambda_i)$. Moreover, as the derivative of $q(x)$ is large for x close to s, even those λ_i that are tightly clustered will become well separated with application of the polynomial filter. Choice of a good shift parameter will result in much-improved convergence relative to a mediocre or poor choice of s. For positive semi-definite problems requiring trailing eigenvalues, it is sufficient to simply set $s = 0$, but without *a priori* information about the spectrum of the input matrix, a better choice is elusive. Multiple shifts may be used to accelerate convergence to the desired part of the spectrum, and the subspaces combined or recycled to avoid loss of any useful information already computed. The rational Krylov method [65] uses multiple shifts to target multiple parts of the spectrum; however, it is intended for non-Hermitian problems and when shifts are known beforehand. For Hermitian problems, simplifications are possible that allow for accelerated convergence and reduced computational costs. Additionally, these simplifications appear to allow for reduced sensitivity to the initial choice of initial shift parameter s, resulting in more robust performance in general.

8.1 Background

Krylov subspace projections are successful at quickly calculating eigenvalue approximations for leading and trailing eigenvalues that are moderately-separated from their neighbors. When eigenvalues are tightly clustered, then convergence requires a great many iterations; this slow convergence can be critical due to practical constraints on the size of the Krylov subspace basis that can be stored in memory. Even when restarts are used to extend the number of iterations, slow convergence of eigenvalues between restarts may be sufficiently poor to cause stagnation. For example, implicitly Lanczos iteration [11, 46] using a Krylov subspace of dimension 100 failed to bring the trailing eigenvalue of the Colorado graph network Laplacian to convergence, even after 24 hours of computation.

Recalling the asymptotic bounds governing convergence of eigenvalues in Krylov subspaces, we have that eigenvalue bounds have error bounded by

$$0 \le \lambda_j - \lambda_j^{(i)} \le (\lambda_j - \lambda_j^{(i)}) \frac{\tan^2 \Theta}{T_{i-1}\left(\frac{1+\gamma}{1-\gamma}\right)^2} \tag{8.3}$$

where

$$\gamma = \frac{\lambda_j - \lambda_{r+1}}{\lambda_{r+1} - \lambda_{\inf}}. \tag{8.4}$$

For the Colorado road network Laplacian, the trailing eigenvalues have small values for γ in (8.3). The resulting values for $T_{i-1}((1+\gamma)/(1-\gamma))$ are small even for large i. For example, we have $\lambda_n = 0.0489$, $\lambda_{n-1} = 0.132$ and $\lambda_1 = 321{,}018.5$. The resulting value of $\gamma = 2.59 \times 10^{-7}$ gives $T_{100}((1+\gamma)/(1-\gamma)) = 1.0052$. Even expanding the block size causes minimal improvement in eigenvalue convergence, for a block size of 50, the value of $T_{100}((1+\gamma)/(1-\gamma)) = 1.297$.

To allow Krylov subspaces to calculate eigenvalues that are tightly clustered, one may apply transformations to the spectrum that leave the eigenvectors unchanged. For example, applying a diagonal shift

$$A' = A - sI \tag{8.5}$$

to generate a new matrix A' leaves eigenvectors unchanged. Likewise matrix inversions share the same eigenspace, since $A^{-1} = U\Lambda^{-1}U^T$ when $A = U\Lambda U^T$. In general, any matrix polynomial in A does not alter the eigenspace of A, so to calculate eigenvalues of A that are prone to slow convergence one may apply a shift and an inversion. Applying shifts is not expensive, but matrix inversion may be costly when the input matrix is large and sparse. When a sparse factorization is tractable, simply inverting the matrix will result in satisfactory convergence of eigenvalues. Following the Colorado road network example with $A' = A^{-1}$, the reciprocated eigenvalues are $\lambda'_1 = \lambda_n^{-1} = 20.454$, $\lambda'_2 = \lambda_{n-1}^{-1} = 7.576$, and $\lambda'_n = \lambda_1^{-1} = 3.115 \times 10^{-6}$, which gives $\gamma = 0.63$ and $T_{10}((1+\gamma)/(1-\gamma)) = 1.22 \times 10^9$. Though this convergence is strong, selection of a shift close to λ'_1 will lead to even faster convergence. With $s = 0.04$, $\lambda'_1 = (\lambda_n - 0.04)^{-1} = 112.51$, $\lambda'_2 = (\lambda_{n-1} - 0.04)^{-1} = 10.87$, and $\lambda'_n = (\lambda_1 - s)^{-1} = 3.115 \times 10^{-6}$, which gives $\gamma = 0.9$ and $T_{10}((1+\gamma)/(1-\gamma)) = 4.48 \times 10^{15}$. We note that though the resulting eigenvalue approximations are for $(A - sI)^{-1}$, eigenvalue approximations for A may be recovered by application of the function

$$f(x) = (x^{-1} + s)^{-1}. \tag{8.6}$$

In some important cases, sparse LU or Cholesky factorization is not expensive. For example, sparse factorizations are tractable for inversion of non-Hermitian matrices arising from circuit simulation; these inversions are used to approximate transfer functions defined in terms of a large, sparse matrix pencil. Some classes of graphs share the same sparsity and planarity properties of circuit graphs that allow for efficient sparse factorizations; road network graphs are one such graph.

The mention of circuit simulation graphs is not simply to demonstrate a class of graphs for which shift and invert methods are feasible due to relatively inexpensive sparse factorizations, but also to motivate methods to generate improved convergence due to good selection of shifts. This notion is not limited to the non-Hermitian eigenproblem; numerous existing methods use adaptive shifts and inversion to accelerate convergence towards chosen eigenvalues, such as Rayleigh quotient iteration or the Jacobi-Davidson method [4]. In the circuit simulation domain, shifts are easy to know *a priori* as they are given by outside constraints on the problem; the shifts are complex frequencies around which the circuit simulation must be accurate. For a low-rank approximation problem in which low-order eigenpairs are needed, it is only known that the eigenvalues are close to zero. For these cases, judicious shifts are not easy to know beforehand.

Without good beforehand knowledge of the neighborhoods containing the desired eigenpairs of A, good shifts for the simple shift and invert method are elusive. One may adapt the shift parameter such that it exploits available information to choose shifts close to where eigenvalues are expected to be. A rational polynomial of the form

$$q(x) = \prod_{s_i \in S} \frac{1}{(x - s_i)} \tag{8.7}$$

for a set of shifts S can produce faster convergence within a neighborhood containing the shifts S. Likewise, an adaptive algorithm could use the already-computed eigenvalue approximations of A to choose shifts that are close to eigenvalues that have not converged satisfactorily.

Adaptive or multiple shifts are not without their drawbacks. Adaptive shifts require explicit orthgonalization to generate an orthonormal basis for the resulting Krylov subspace union $\bigcup_{s_i \in S} \mathcal{K}_i((A - s_i I)^{-1}, x_0)$ for a set of shifts S. This increases the computational complexity of each Lanczos iteration from linear $O(nk)$ to quadratic in the number of dimensions $O(nk^2)$. A further problem is that the restriction $\hat{A}^{(k)}$ of A to $\bigcup_{s_i \in S} \mathcal{K}_i((A - s_i I)^{-1}, x_0)$ is not explicitly computed. Conversely, use of rational polynomials other than the simple shift $q(x) = (x - s)^{-1}$ will render any low-rank approximation of the input matrix A restricted to $\mathcal{K}_i(q(A), x_0)$ to be useless for approximating eigenvalues of A. This would require a re-projection of A into $\mathcal{K}_i(q(A), x_0)$, at a cost of $O(nk^2)$ FLOPS.

Adaptive shifts are also not novel. For example, Rayleigh quotient iteration [4] may be viewed as inverse power iteration with an adaptive shift at each step. The resulting convergence is improved from linear convergence to the smallest eigenpair to cubic. The Jacobi-Davidson [4, 77] method also uses adaptive shifts at each step to likewise accelerate convergence and cope with inexact matrix inversion. Polynomial filtering methods are also not novel in general, having been used for implicitly approximating the truncated SVD [42].

8.2 Adaptive shift methods

Adaptive shift methods for eigenvalue problems may readily be adapted to low-rank approximation problems as an accelerated alternative to a Krylov subspace from a shifted and inverted matrix. The simplest adaptive shift method is Rayleigh quotient iteration, presented in numerous texts, we refer to the description in [4]. Rayleigh quotient iteration is similar to inverse iteration – power iteration on $(A-sI)^{-1}$ – but uses an adaptively-determined shift s rather than a static one. It derives its name from the method used to choose the shift at each iteration; it uses the Rayleigh quotient v^TAv/v^Tv to determine the next shift.

Rayleigh quotient iteration implicitly defines a subspace in terms of its vectors q_i; is sufficient to simply apply an orthonormalization process such as Gram-Schmidt to produce an orthogonal set of vectors. Neglecting the cost of the matrix inversion, the overriding complexity is from the orthonormalization process, and is $O(nk^2)$ for $A \in \mathbb{R}^{n\times n}$. The resulting algorithm is in Algorithm 12. A drawback of Rayleigh quotient iteration is that it has unpredictable behaviors [4]; it may not converge to the eigenvalue closest to the initial shift s, and the inital start vector q_0 is also influential in convergence, and should be close to u_i, where (λ_i, u_i) is the eigenpair towards which convergence is desired. For example, simply selecting $s = 0$ for a positive definite input matrix A may not produce the smallest eigenvalue λ_{inf}. This unpredictable behavior may be corrected by use of the subspace Q in Algorithm 12 to generate a subspace approximation $\hat{A}^{(j)}$ of A in span$\{Q\}$. The shift s may then be chosen to be $\lambda_{\text{inf}}(\hat{A}^{(j)})$. This essentially results in a variant of the Jacobi-Davidson Algorithm [4] with an exact solution. However, the method omits restarts, as it is intended to produce a minimal subspace in the same spirit as minimal Krylov subspaces.

Algorithm 12 Rayleigh quotient iteration and orthonormal subspace generation

```
Require: a priori chosen start vector $q_1$ and initial shift $s$.
$Q \leftarrow q_1/\|q_1\|$
for $j = 1,2,\ldots$ do
  $y \leftarrow (A - sI)^{-1} q_i$
  if $(A - sI)$ is singular then
    return $q_{i+1}, \theta$
  end if
  $\theta \leftarrow \|y\|_2$
  $s \leftarrow \frac{y^T A y}{\theta^2}$
  $q_{i+1} \leftarrow y/\theta$
  $Q \leftarrow [Q \;\; q_{i+1} - QQ^T q_{i+1}/\|q_{i+1} - QQ^T q_{i+1}\|]$
end for
```

These methods exhibit fast convergence, but for generating a subspace they may not be ideal choices due to the need for a matrix inversion at each iteration. Though we have assumed that the sparse factorization of $A - sI$ is tractable, it may still be a non-negligible expense. It is notable the Jacobi-Davidson may tolerate an inexact solution of $(A - sI)x = q_i$, but this still requires an iterative solver, which may be more expensive than the sparse factorization of $A - sI$ in the cases under consideration here. Computational savings may be realized by reducing the number of factorizations needed per subspace dimension generated. This motivates consideration of Krylov subspaces using multiple shifts when sparse matrix factorizations are not expensive.

8.3 Polynomial filter methods

Eigenvalue convergence may be accelerated by using multiple shifts in ways alternate to the multiple shifts in Rayleigh quotient iteration to produce better subspaces at reduced cost. One such method is to use polynomials other than the simple shift and invert polynomial (8.2). Slightly more complicated polynomials may not require much more computation, but could lead to faster convergence of the desired eigenvalues than a simple fixed shift, and have less sensitivity to the choice of shift s as methods using (8.2).

Shift-and-invert methods attain better convergence through expansion of local gaps between eigenvalues. The role of local gaps is expressed in the denominator of 8.3. The effectiveness of a preconditioner is influenced by how much it is able to expand gaps between desired λ_i and λ_{i+1} when they are tightly clustered, and how small it renders the infimum of eigenvalues. For a simple shift we present the following proposition.

Proposition 8. *Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{\text{inf}}$ be real eigenvalues, let s be an arbitrary real number, let $g = \lambda_i - \lambda_{i+1}$ be the eigenvalue gap between λ_i and λ_{i+1} for some natural i. Let the distance between the shift s and λ_i be written as $\delta = \lambda_i - s$. Let $q_1(x) = (x-s)^{-1}$ be a spectral transform polynomial. Then the gap for the transformed eigenvalues is given by*

$$q_1(\lambda_i) - q_1(\lambda_{i+1}) = \frac{1}{\lambda_i - s} - \frac{1}{\lambda_{i+1} - s} = \frac{1}{\lambda_i - s} - \frac{1}{\lambda_i + g - s} = \frac{1}{\delta} - \frac{1}{\delta + g}. \tag{8.8}$$

We have the limit of (8.8) goes to ∞ as $\delta \to 0$, which implies that gaps are expanded proportionally to the closeness of the shift to λ_i. This means that good choice of s will be important to getting good results out of a shift-and-invert method.

To alleviate the importance of the shift s on convergence of eigenvalues, we may use multiple shifts, and form a matrix polynomial of the form $q(x) = \prod_{s_i \in S}(x-s)^{-1}$ for a set of shifts S. This approach allows multiple guesses for the neighborhoods containing desired eigenvalues, and the multiple shifts reinforce each other. For example, consider the case $|S| = 2$ and the following proposition.

Proposition 9. *Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{\text{inf}}$ be real eigenvalues, let s_1 and s_2 be arbitrary real numbers, let $g = \lambda_i - \lambda_{i+1}$ be the eigenvalue gap between λ_i and λ_{i+1} for some natural i. Let the distance between the shift s and λ_i be written as $\delta = \lambda_i - s$. Let $q_2(x) = ((x-s_2)(x-s_2))^{-1}$ be a spectral transform polynomial. Assume without loss of generality that $|\lambda_i - s_1| \leq |\lambda_i - s_2|$. Let $\delta = \lambda_i - s_1$. Let $s_2 = s_1 + c$. Then the* *transformed gap is given by*

$$
\begin{aligned}
q_2(\lambda_i) - q_2(\lambda_{i+1}) &= \frac{1}{(\lambda_i - s_1)(\lambda_i - s_2)} - \frac{1}{(\lambda_{i+1} - s_1)(\lambda_{i+1} - s_2)} \\
&= \frac{1}{(\lambda_i - s_1)(\lambda_i - s_1 - c)} - \frac{1}{(\lambda_i + g - s_1)(\lambda_i + g - s_1 - c)} \qquad (8.9) \\
&= \frac{1}{\delta^2 + \delta c} - \frac{1}{(\delta + g)(\delta + g - c)}.
\end{aligned}
$$

This equation also has a limit of $\pm\infty$ as $\delta \leftarrow 0$, provided that $g \neq 0$. Moreover, the gap expansion is larger when the proportion of the gap g to the distance of the shifts is small, as the denominator in the leading term of term of (8.9) is δ^2 instead of δ. For example consider, if $\lambda_{i+1} = 5.868 \times 10^{-7}$ and $\lambda_i = 1.621 \times 10^{-6}$, then $g = 1.034 \times 10^{-6}$. If $s = s_1 = 1 \times 10^{-8}$ and $s_2 = 1 \times 10^{-6}$, then from (8.8), $q_1(x) = 1.112 \times 10^6$ but from (8.9), $q_2(x) = -5.196 \times 10^{12}$. These are the actual values of eigenvalues from the normalized Colorado road network Laplacian graph, presented in the following results section. By having more shifts, the chance of one being a good guess is improved. We note that a polynomial of the form in (8.7) may transform a positive semi-definite input problem into an indefinite problem. This presents a difficulty, as the width of the interval containing all eigenvalues $\lambda \in [a, b]$ is increased. This implies that the value of gamma from (8.4) may be *increased* due to the larger value of $\lambda_i - \text{inf}$. Replacement of $q(x) = \prod_{s_i \in S}(x - s_i)^{-1}$ with $q(x)^2$ maintains positive semidefiniteness of the input problem and avoids the growth of $\lambda_i - \lambda_{\text{inf}}$ relative to λ_i.

Naturally, blind guesses at locations of eigenvalues may lead to choices that are rather poor. Even when the input matrix is positive semi-definite and small eigenvalues are known to be greater than 0, locating the location of those that are responsible for most of the norm of A^+ is still difficult. Multiple shifts do alleviate the need to precisely locate the neighborhood of the desired small eigenvalues, but do not alleviate the need so much that beforehand guesses may lead to convergence better than simply using $q(x) = x^{-1}$. Rather than simple blind guessing, one may use a preliminary set of iterations to gather information regarding the neighborhoods of desired eigenvalues of A. With information regarding the neighborhoods of eigenvalues of A, one may either perform a thick restart [84]

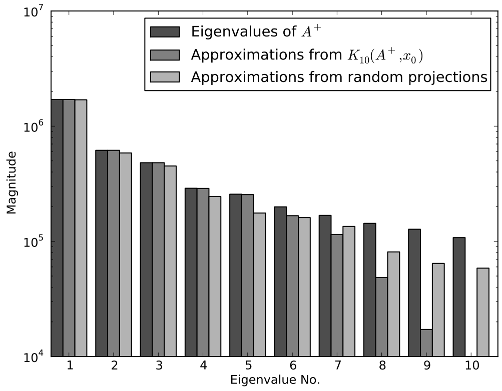

Figure 8.1: Leading 10 eigenvalues and eigenvalue approximations using $\mathscr{K}_{10}(A^+, x_0)$ and random projections. Random projection approximations have more uniform tightness compared to single-vector Krylov subspace approximations.

and save whatever useful spectral information may be gleaned from the preliminary subspace, or simply discard it and perform new iterations. With better information regarding the neighborhoods containing the desired eigenvalues of A, multiple shifts can outperform a single shift more decisively.

We recall that the random projection method in Algorithm 3 and investigated in previous chapters produces good eigenvalue approximations. One characteristic we have observed is that the random projection method using a block size of k is that it produces eigenvalue approximations that have roughly uniform tightness for all k leading eigenvalues. Figure 8.1 illustrates this difference in bounds. Note that the random projection method produces better approximations of eigenvalues 8 through 10. This behavior suggests that random projections or block Krylov subspaces with large blocks be used to generate the preliminary subspace for determining the shifts to use. With this information, we present the algorithm for multiple shift polynomial preconditioning for Krylov subspaces.

A drawback of using polynomials of the form (8.7) is that it is no longer possible to recover the original eigenvalues of the input matrix A from those of the transformed matrix $q(A)$; this necessitates the reprojection of A into the solution subspace on line 3. A function of the form $q(x) = (x - s)^{-1}$ is bijective, so there is always a unique solution y for $q(y) = x$ for any x. Multiple shifts result in polynomials that are not necessarily bijective, so there may be multiple solutions of $q(y)$ for a given x. Therefore the shift must be used to generate a projection basis Q with $\text{span}\{Q\} = \mathscr{K}_k(q(A), x_0)$, and

Algorithm 13 Preconditioned Krylov subspaces with multiple shift polynomials

```
Require: a priori chosen start block $X_0$ with $m$ columns for using $m$ shifts.
compute $U, \Psi$ using random projections (Algorithm 3) on $A^{-1}$ with $X_0$.
compute $Q, T$ using Lanczos iteration (Algorithm 1) on $\prod_{j=1}^{m}(A - \psi_j I)^{-2}$ with $\sum_{i=1}^{m} v_i/k$.
$\hat{A} \leftarrow Q^T A Q$.
spectral decompose $V, \Theta = \hat{A}$
return $QV, \Theta$
```

that basis then used to project the original input matrix A to $Q^T AQ$ for eigenvalue approximation. This requires a dense matrix-matrix product with complexity $O(nk^2)$. We note that this is the same complexity as the complete Gram-Schmidt process necessary for generation of an orthonormal basis in Rayleigh quotient iteration or the Jacobi-Davidson method.

8.4 Numerical results

To demonstrate the improvement realizable through use of multiple shift polynomials over single shift polynomials, we perform experiments with two large graph Laplacian matrices for which the sparse factorization is reasonably inexpensive. We compare the rates of convergence over many dimensions. Multiple shift polynomials are similar to inner iteration preconditioning in that both use polynomials or order 2 or greater; we also compare the convergence of multiple shift polynomials against inner power iteration with a single shift to exclude effects of simply using a higher order polynomial from the effects of using judiciously-chosen multiple shifts. We also consider the convergence of eigenvalues in Jacobi-Davidson spaces for a further point of reference.

We compared the convergence of eigenvalues for the Colorado road network Laplacian matrix, both for the normalized and combinatorial Laplaicians. The leading 64 eigenvalues of both matrices is shown in Figure 8.2. We remark that both matrices exhibit tight clustering of the trailing eigenvalues. This, combined with the relatively large leading eigenvalues, leads to terribly slow convergence of trailing eigenvalues. The values of γ from (2.3) are $\gamma = 5.17 \times 10^{-7}$ for the normalized Laplacian and $\gamma = 2.59 \times 10^{-7}$ for the combinatorial Laplacian. Even for large subspaces, the convergence of these trailing eigenvalues is limited; the denominator term in (2.3) for the normalized Laplacian is

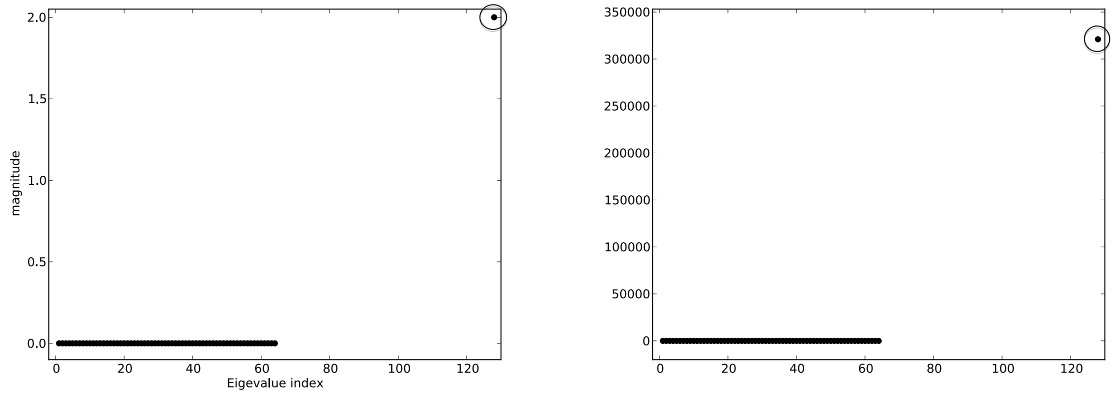

Figure 8.2: Trailing 64 eigenvalues and leading eigenvalue (circled) for normalized Graph Laplacian (left) and combinatorial graph Laplacian matrices (right) for the Colorado road network graph.

less than 5 for a subspace dimension of 1,000. Therefore preconditioning is needed.

These matrices are always positive semi-definite with exactly one null eigenvector, and their structure is planar and sparse. We have observed that the sparse LU-factorization may be computed in the order of seconds using the Super-LU library [47]. These matrices make good candidates for application of multiple polynomial shifts. For comparison of the methods discussed in the preceding section, we first performed single-vector Lanczos iteration to produce a low-rank approximation of L restricted to $\mathscr{K}_i(L^+, x_0)$; this corresponds to simple shift preconditioning for $s = 0$. We also performed baseline runs with Jacobi-Davidson using a deflation tolerance of $\tau = 1 \times 10^{-11}$. For evaluation of multiple-shift polynomials, we ran Algorithm 13 with a multi-shift parameter $m = 4$. We generated subspaces of dimension up to 64, and compared the nuclear norms of the approximation to the approximation generated by the truncated spectral decomposition, which is coincides with the truncated SVD as the input matrix is positive semi-definite. Figure 8.3 shows the low-rank approximation norms for the normalized Laplacian, and Figure 8.4 shows the low-rank approximation norms for the combinatorial Laplacian. The multi-shift Krylov subspace projections produce better low-rank approximations than either Jacobi-Davidson or single-shift Krylov subspaces for $s = 0$ or $s = \sum_{i=1}^{k} \theta_i(\hat{A}^{(k)})/k$ with $\hat{A}^{(k)} \in \text{span}\{AX_0\}$. It is important that the multi-shift Krylov subspaces require multiple solutions of a sparse factored LU matrix per dimension, rather than one solution for the single shift subspaces. This presents an additional computational expense; in this way the

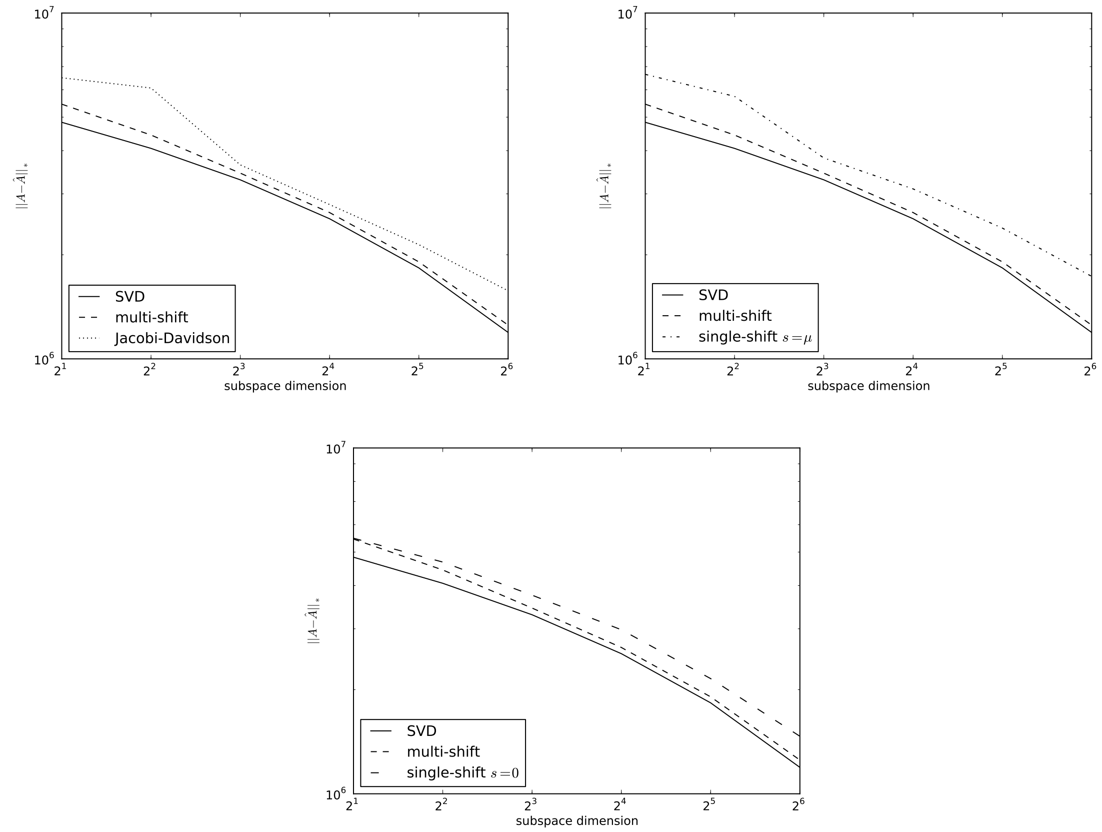

Figure 8.3: Errors of low-rank approximations of normalized Laplacian for Colorado road network graph: Jacobi-Davidson v. multi-shift Krylov (upper-left), single shift with shift from preliminary random projections approximation v. multi-shift (upper-right), and unshifted pseudoinverse v. multi-shift (bottom).

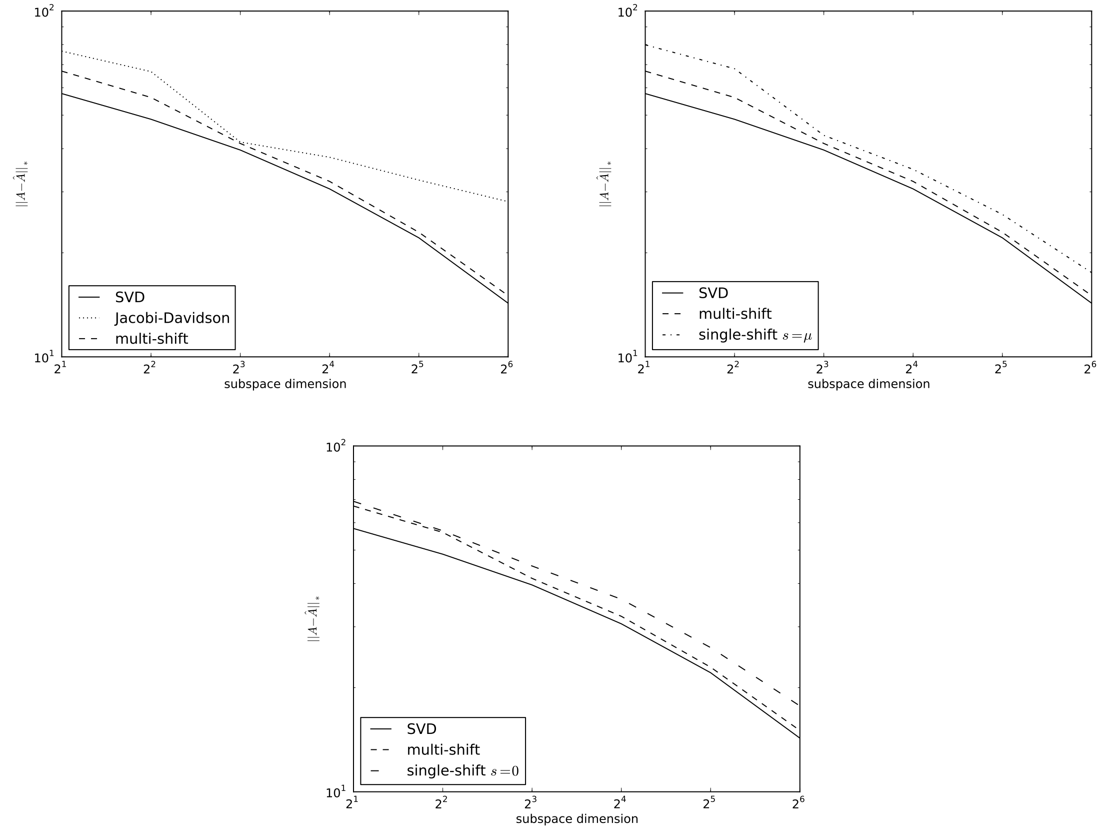

Figure 8.4: Errors of low-rank approximations of combinatorial Laplacian for Colorado road network graph: Jacobi-Davidson v. multi-shift Krylov (upper-left), single shift with shift from preliminary random projections approximation v. multi-shift (upper-right), and unshifted pseudoinverse v. multi-shift (bottom).

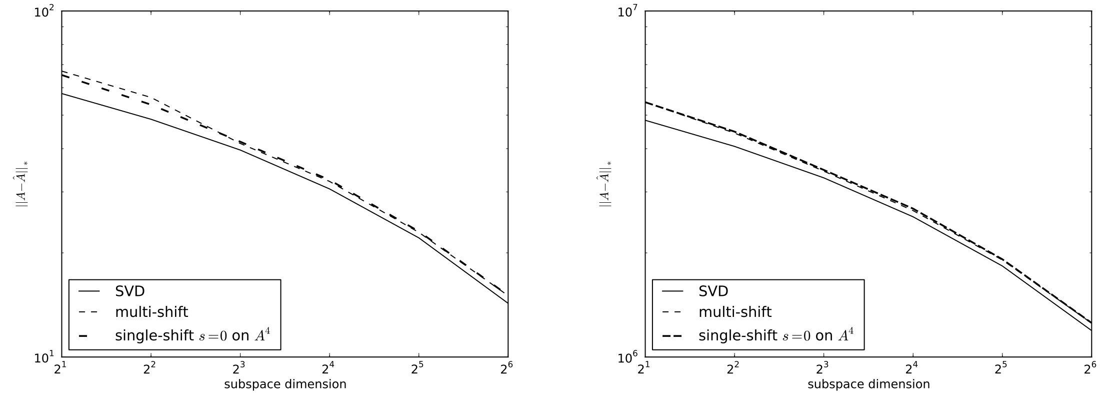

Figure 8.5: Comparison of low-rank approximation norms between inner-iteration preconditioning on A^{-1} with 4 inner iterations and multi-shift preconditioning with a preconditioning factor of $m = 4$. The results for the combinatorial Laplacian of the Colorado road network are on the left and those for the normalized Laplacian are on the left.

method of using multiple shifts may be compared to the inner power iteration method. Multiple shifts will derive some benefit from the multiple shifts, but will also obtain some benefit from the inner-iteration like use of a higher-order matrix polynomial to better drive convergence of leading eigenvalues. For example, one could use the sparse LU factorization combined with inner iterations to form A^{-m} for instead of using Algorithm 13 and m shifts. One may also apply inner iteration preconditioning to the singly-shifted Krylov subspace $\mathscr{K}_i((A-\mu I)^{-1}, x_0)$ for $\mu = \sum_{i=1}^{m} \theta_i(\hat{A}^{(k)})/k$ for an initial random projection eigenvalue approximation. The difference between the two methods depends on how well the multiple shifts accelerate convergence compared to simply using a higher order matrix polynomial. Compared to inner iteration preconditioning, the multiple shift method does not have an advantage as distinct as its advantage over the single-shift and Jacobi-Davidson methods; it does produce only slightly better low-rank approximations, especially for intermediate dimensions. Figure 8.5 shows the comparison between inner power iteration preconditioning and multi-shift iteration. These results suggest that some non-negligible degree of the good approximation generated by the multi-shift methods is due to the inner-iteration effects expanding local gaps in the leading spectrum of A^{-1}. However, for intermediate dimensions, multiple-shift preconditioning still produces slightly better approximations, which may be significant when memory resources are limited.

8.5 Conclusion

Interesting classes of problems require low-rank approximation of a matrix using its eigenpairs with smallest rather than largest dimension. For these matrices, Krylov subspaces or random projections fail, as the latter is intended only to approximate leading eigenvalues, and the former encounters difficulty when the input matrix has tightly-clustered eigenvalues. In these instances, Krylov subspaces can still give good convergence if the input problem can be preconditioned with a shift and matrix inversion. Simple matrix inversions yield good results for positive definite problems, but the convergence can be accelerated by using multiple shifts that are close to eigenvalues of interest.

In comparison with the combination of inner iteration preconditioning with shift-and-invert preconditioning with one shift, multiple shifts have some, albeit limited, advantages. When shift-and-invert with a single shift produces adequate approximations, then addition of inner iteration will improve these to good results, and multiple shifts will lead to a marginal improvement. Multiple shifts will be most appropriate when each solution of the factored LU system is of moderate or greater cost. This would impose a limit on the number of inner iterations due to cost, and the advantages of multiple shifts over inner iteration will be most pronounced.

One unexpected result we witness in these experiments is the comparatively poor approximation properties of the Jacobi-Davidson method for generating low-rank approximations in minimal iterations. In every case, a shift-and-invert transformation applied to the Lanczos process produced a better low-rank approximation error than the Jacobi-Davidson method. This suggests that shift-and-invert Lanczos methods are preferable to Jacobi-Davidson for minimal low-rank approximation when matrix inversions are not expensive, and future efforts could generalize these results to inexact Krylov subspaces.

Chapter 9

Low-rank approximation using Krylov subspaces for fast convolution

Low-rank matrix approximation provides opportunities for dramatic improvements in compute time in some instances. One such instance is convolution, which may be used for signal filtering or model reduction. For example, simply filtering n signals requires n convolutions, or evaluation of k frequencies of the pseudovelocity shock response spectrum [28] on n input signals requires nk convolutions. Low-rank approximation allows for a reduction in work corresponding to the reduction in rank of the input data. The discrete convolution of two functions f and g

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \tag{9.1}$$

may be obtained more simply in the frequency domain as

$$(f * g)[n] = \mathscr{F}^{-1}\{\mathscr{F}\{f\} \cdot \mathscr{F}\{g\}\}[n]. \tag{9.2}$$

As the Fourier transform is itself an orthonormal projection, we may apply basis expansion $\mathscr{F}\{g\} = \sum_{i=1}^{r} u_i c_i$ for r basis vectors u_i in the frequency domain and reconstruct the convolution $(f * g)$ using the convolved basis vectors $(f * \mathscr{F}^{-1}\{u_i\})$. This may drastically reduce the number of convolutions necessary when presented with a large number of input vectors that are low rank. The sole requirement on the basis vectors u_i is that they be linearly independent; one may use singular vectors from a proper orthogonal decomposition [13, 76], an orthogonal basis set from random projections or a basis set for a Krylov subspace. Therefore, this problem type allows evaluation of the random projection method against the hybrid block Krylov subspace method, GrABL and inner power iteration for low-rank approximation.

We consider an application of low-rank approximation to a set of 1-dimensional input acceleration time histories. These time histories were obtained from the response of two 1-DOF linear spring-mass-damper systems to random impulsive loading. These systems were meant to represent simple physical structures and model their response to shock loading. As only two separate systems were used to generate the data, the rank of the output was much lower than the actual number of signals contained. We consider the error in low-rank approximation of the various methods developed previously and its impact on theoretical error bounds on the convolution output. The resulting data matrix has 131,072 columns and 2,000,0000 rows, with 497,197,284 nonzero elements. Solving the eigenproblem on $A^T A$ will result in sparse matrix-vector products that are far more expensive than dense linear algebra operations. Therefore, low-rank approximation methods that minimize sparse matrix-vector products will result in the most meaningful compute cost savings when only the low-rank approximation $\hat{A}^{(k)}$ to A need be computed. When a orthonormal projection matrix Q with $Q^T A = \hat{A}^{(k)}$ is required, dense matrix operations are a significant cost, and the reduction in complexity due to block size shrinkage may be expected to result in compute savings for the shrink-and-iterate, hybrid random projections-block Krylov and GrABL methods.

The methods described here bear similarity to Krylov methods for model reduction and solution of ODEs, though there are distinctions between them. For ODE solution, one may solve

$$\dot{y}(t) = Ay(t) \tag{9.3}$$

using the matrix exponential e^{At} with initial condition $y(0) = y_0$. Krylov subspace projections may be used to approximate the matrix exponential operator e^{At} with a lower-rank $e^{\hat{A}t}$ restricted to a Krylov subspace [38, 68]. Model reduction applications also use subspace projections for approximating solutions to differential equations of the form

$$E\dot{x}(t) = Ax + Bu(t) \tag{9.4}$$

$$y(t) = C^T x(t). \tag{9.5}$$

These may be solved with convolution of the system's transfer function with the input vector as $y(t) = (h * u)(t)$, which is accomplished in the frequency domain as $Y(s) = U(s) \cdot H(s)$ with

$$H(s) = D + C^T (A - Es)^{-1} B. \tag{9.6}$$

Krylov subspace approximations to the matrix in (9.6) representing the transfer function may be used to approximate the system response at a single frequency s or a set of complex frequencies [27,34,35]. Other model reduction methods for nonlinear systems

$$\dot{x}(t) = f(x(t), u(t)) \tag{9.7}$$

are based on proper orthogonal decomposition of the inputs $u(t)$ and outputs $y(t)$ empirically obtained from model runs [2] that may be used to project the high-dimensional state $x(t)$ into a low-dimensional subspace and produce a low-rank approximation $\xi(t)$. In all these cases, the model — be it e^{At}, $H(s)$ or $x(t)$ — is approximated in a low-dimensional subspace. Thus, the complexity of the model is reduced. In this application of low-rank approximation, we are reducing the dimension of the *inputs* rather than the model, which is allowed to remain high-dimensional. It would correspond to approximating y_0 or $U(s)$ for the ODE or model reduction problems rather than the operators that describe the state of the system. The reduction in compute cost comes not from deriving a model approximation that is easier to evaluate, but finding a low-rank space that characterizes all of the possible inputs. The basis vectors of that low-rank input space then may be run through the original, high-dimensional model and then can reconstruct the outputs $y(t)$. This enables reduction of compute costs for cases in which evaluation of the transfer function $H(s)$ may not be expensive, but must be evaluated a great many times.

9.1 Low-rank approximation error and convolution

If one approximates an input signal f with a low-rank approximation $\hat{f}$ such that $f = \hat{f} + e$ for some error e, then the low-rank approximation error will influence the error of operations on $\hat{f}$. In the case of convolution $(f * g)$, the error $\|(f * g) - (\hat{f} * g)\|$ depends on the approximation error e of f and g. The formal results are summarized in the following theorem.

Theorem 12. *Let f and g be two arbitrary vectors in $\mathbb{R}^n$. Let $\hat{f}$ be a low-rank approximation of f with $\hat{f} + e = f$ such that $\hat{f} \perp e$. Let $\epsilon = \|e\|^2$. Then the error in the discrete convolution $\|(f * g) - (\hat{f} * g)\|^2$ is bounded as*

$$\|(f * g) - (\hat{f} * g)\|^2 \leq G_{\max}^2 \epsilon \tag{9.8}$$

where $G_{\max} = \max_i\{|\mathscr{F}\{g\}_i|\}$ is the maximum element of the frequency domain transform of g.

Proof. Since the frequency domain transform is unitary, then $\|x\| = \|\mathscr{F}\{x\}\|$ for any x. Let $F = \mathscr{F}\{f\}$, $\hat{F} = \mathscr{F}\{\hat{f}\}$, $E = \mathscr{F}\{e\}$ and $G = \mathscr{F}\{g\}$. Note that $F = \hat{F} + E$, as a frequency transform may also be expressed as a change of basis.

$$\|(f * g) - (\hat{f} * g)\|^2 = \|(F \cdot G) - (\hat{F} \cdot G)\|^2 = \sum_{i=1}^{n} (F_i G_i - \hat{F}_i G_i)^2. \tag{9.9}$$

Since $E = F - \hat{F}$, then any individual entry E_i of E has

$$E_i^2 = \hat{F}_i^2 + F_i^2 - 2F_i\hat{F}_i. \tag{9.10}$$

Then

$$\sum_{i=1}^{n} (F_i G_i - \hat{F}_i G_i)^2 = \sum_{i=1}^{n} G_i^2 F_i^2 + G_i^2 \hat{F}_i^2 - 2G_i F_i \hat{F}_i = \sum_{i=1}^{n} G_i^2 E_i^2. \tag{9.11}$$

If $G_{\max} = \max_i\{|\mathscr{F}\{g\}_i|\}$, then

$$\sum_{i=1}^{n} G_i^2 E_i^2 \leq G_{\max}^2 \sum_{i=1}^{n} E_i^2 = G_{\max}^2 \|E\|^2 = G_{\max}^2 \epsilon \tag{9.12}$$

as the frequency transform is unitary and $\|e\|^2 = \|E\|^2 = \epsilon$. □

One may note that the Frobenius error of the low-rank approximation of A with $\hat{A}^{(k)}$ is the aggregate $\|e\|^2$ error over all columns of A. Thus $\|A - \hat{A}^{(k)}\|_F^2$ does bound any $\|e\|^2$ from above; when an orthogonal projection is used to produce $\hat{A}^{(k)}$, the errors are orthogonal to the low-rank approximations, and colums a_i of a have approximation error

$$\|a_i - \hat{a}_i\|^2 = \|a_i\|^2 - \|\hat{a}_2\|^2 = \|e_i\|^2. \tag{9.13}$$

Thus we may simply measure the norms of the original data $\|a_i\|$ and their corresponding low-rank approximations to easily calculate $\|e_i\|$. Thus, it is an easy matter to apply the bounds from Theorem 12 *a posteriori* to determine the maximum possible error.

9.2 Numerical results

To generate numerical results, we generated data using two 1 degree-of-freedom linear spring-mass-damper systems representing stiff mechanical structures. The systems were subjected to randomly-determined impulsive loading for an initial period of 10 milliseconds, and then subjected to a second 50 millisecond period of impulsive loading at 1 second. The systems were run for 2 seconds, at which point they had reached equilibrium. The sampling rate was 2 MHz. 2^{17} runs were performed; 2^{16} for each system. Sparse random noise was added to the outputs. The resulting data matrix had 2×10^6 columns and 2^{17} rows; the matrix had 497,197,284 nonzero elements. We did not center the columns of the matrix. The leading 64 elements of the spectrum of A^TA is shown in Figure 9.1. We compare the results of the proper orthogonal decomposition against low-rank approximation using random projections, GrABL, the hybrid block Krylov-random projections method with 50% block size, single-vector Lanczos with inner power iteration with $p = 2$ and GrABL with $\rho = 0.001$, a deflation window of 3 and $p = 2$. These parameters resulted in a initial block size of 4. All methods were used to generate low-rank approximations of $k = 2^i$ dimensions for $i = 1, 2, \ldots, 6$. The low-rank approximation error indicates the maximum convolution error, and we present a simple example using a low-pass Butterworth filter. We also consider the compute costs of random projections and the Krylov subspace methods developed previously herein.

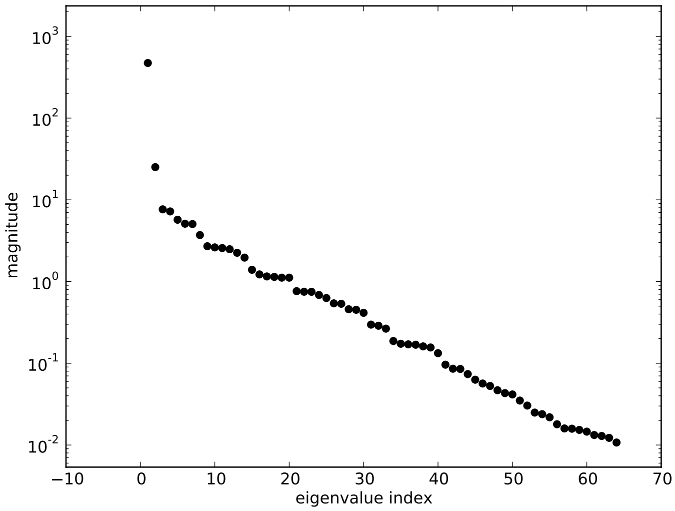

Figure 9.1: Leading 64 eigenvalues of $A^T A$.

We first consider the squared Frobenius norm of the approximation error; this is indicative of the aggregate squared Euclidean error over all columns of the input matrix A. The squared Frobenius norm of the approximation error $\|A-\hat{A}^{(k)}\|_F^2$ for $\hat{A}^{(k)}$ as computed by PCA (equivalent to POD), GrABL with $p = 2$, inner iterations with $p = 2$, random projections and the hybrid random projections-block Krylov method operating in $\mathscr{K}_2(A, A^2 X_0)$ are shown in Figure 9.2. Even though there are over 100,000 signals, it is possible to reconstruct all of them in a 64-dimensional subspace with Frobenius norm error of less than 1% for all methods. The SVD produces the smallest Frobenius norm error by definition, and the hybrid method is the relaxed method with best error at all dimensions. As mentioned previously, Frobenius norm error is combined error over all columns of $A - \hat{A}^{(k)}$. Since columns of A are individual signals, the approximation error of any one individual signal $\|a_i - \hat{a}_i\|^2 = \|e_i\|^2$ is bounded by $\|A-\hat{A}^{(k)}\|_F^2$, but this is likely a pessimistic bound. Equation (9.13) gives a method to easily compute the individual error *a posteriori*. Figure 9.3 shows the maximum individual error magnitude $(\max_i \|e_i\|^2)/\|a_i\|^2$; this is the percent error for the largest individual approximation error.

Theorem 12 shows the relationship between approximation error $\|a_i - \hat{a}_i\|^2$ and the convolved approximation error $\|(a_i * g) - (\hat{a}_i * g)\|^2$ for some appropriate g. We consider two arbitrarily-chosen signals, one from each of the models run to produce columns of A. We compare the approximation

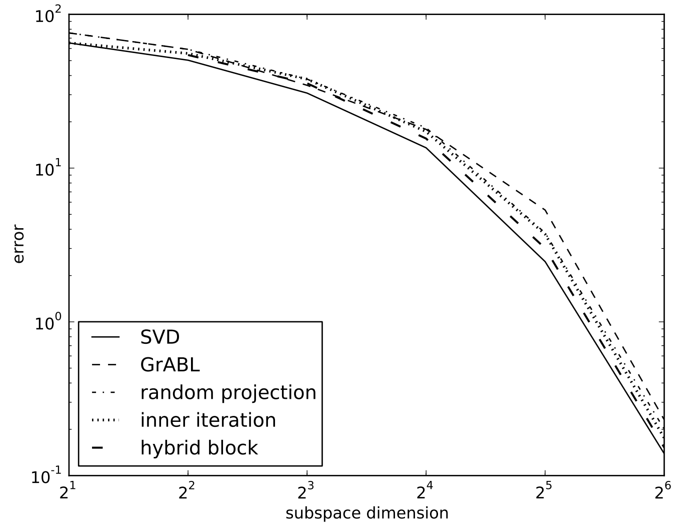

Figure 9.2: Squared Frobenius norm of approximation error of A with $\hat{A}^{(k)}$ for PCA, GrABL, random projections, inner power iterations with $p = 2$, and the hybrid random projections-block Lanczos method.

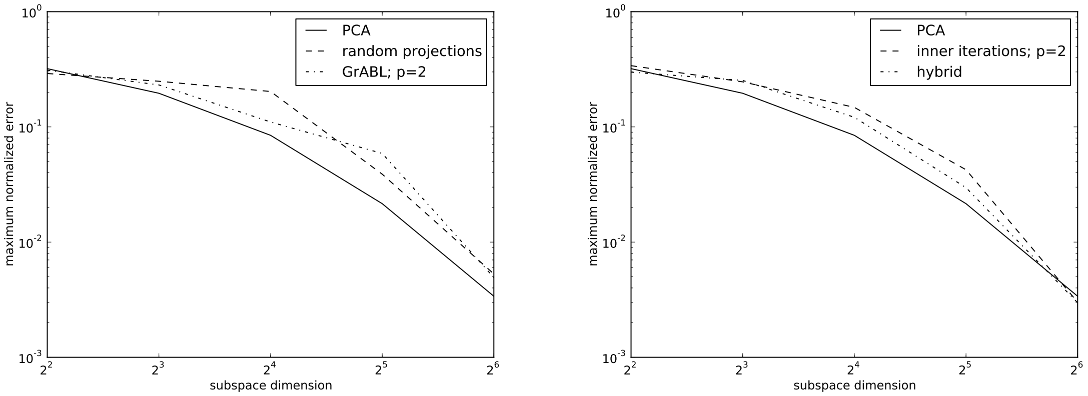

Figure 9.3: Maximum column approximation error as a percent for PCA, GrABL and random projections (left), and PCA, inner power iterations with $p = 2$, and the hybrid random projections-block Lanczos method (right).

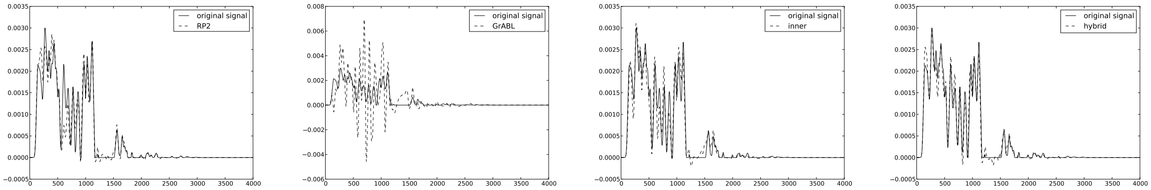

Figure 9.4: Original and approximated example signals from the stiffer of the two models.

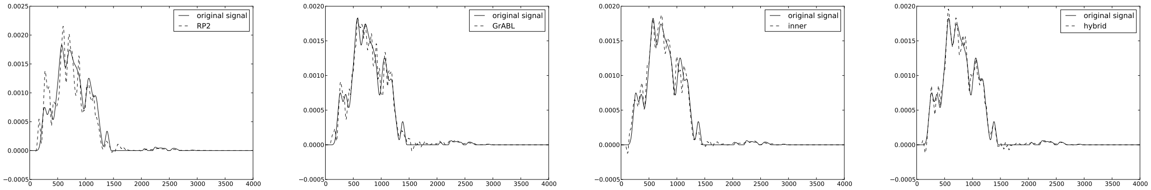

Figure 9.5: Original and approximated example signals from the softer of the two models.

norm $\|a_i - \hat{a}_i\|$ for each signal, as well as the convolved approximation error $\|(a_i * g) - (\hat{a}_i * g)\|^2$ for a second-order low-pass Butterworth filter with a cutoff frequency of 1 kHz. Figure 9.4 shows the original signals from the first model, and Figure 9.5 shows the original signals from the second model. Figures 9.6 and 9.7 show the filtered signals from the first and second model, respectively. Only the first 4000 samples are shown. A key characteristic of Butterworth filters is that they are flat in the passband; the filter used for these examples has a gain of no more than 1 for any frequency. Therefore, Theorem 12 predicts that all filtered approximation errors will be no greater than unfiltered approximation errors. For the shown figures, this indeed holds; the approximation and filtered approximation errors are shown in Table 9.1.

Approximation accuracy is not the only consideration for this example; the resulting matrix has a large number of nonzero elements, especially in comparison to the dimension of the Gram matrix

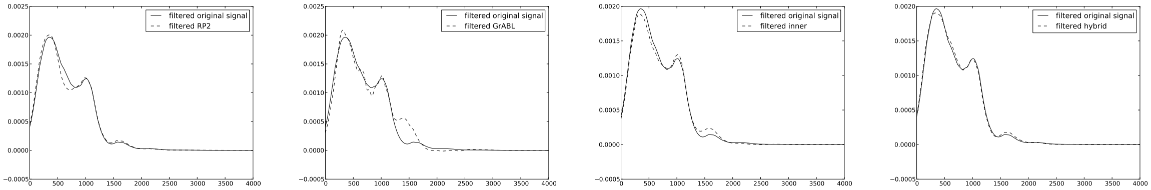

Figure 9.6: Original and approximated example signals from the stiffer model filtered with Butterworth low-pass filter.

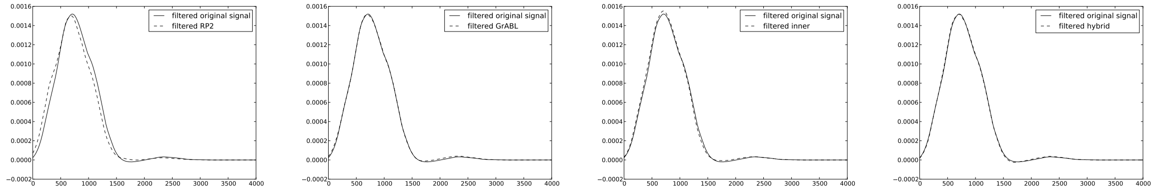

Figure 9.7: Original and approximated example signals from the softer model filtered with Butterworth low-pass filter.

	SVD	random projections	GrABL	inner iterations	hybrid method
model 1 error	1.08×10^{-4}	1.91×10^{-2}	9.32×10^{-2}	1.67×10^{-2}	1.16×10^{-2}
model 1 filtered error	1.33×10^{-5}	5.12×10^{-3}	1.01×10^{-2}	4.1×10^{-3}	2.24×10^{-3}
model 2 error	3.58×10^{-4}	1.34×10^{-2}	8.16×10^{-3}	7.38×10^{-3}	5.23×10^{-3}
model 2 filtered error	1.85×10^{-5}	6.07×10^{-3}	7.11×10^{-4}	1.89×10^{-3}	7.45×10^{-4}

Table 9.1: Convolved and unconvolved errors for the two example signals.

$A^T A$ used as input to the various algorithms here. The matrix A has nearly a half billion nonzero elements, but $A^T A$ has 3 orders of magnitude fewer dimensions. Compute costs for developing the low-rank approximations here are non-trivial, and due to the structure of the problem, sparse matrix-vector products may be expected to be the predominate driver of compute costs. Figure 9.8 shows the theoretical FLOP counts for the various algorithms; Figure 9.9 shows the ratio of FLOPS normalized to FLOPS required by the random projection algorithm. The random projection method is not the most expensive for small dimensions; this is due to the larger number of matrix-vector products in GrABL during its initial refinement phase. However, for larger dimensions, GrABL obtains an advantage. Inner iterations and the hybrid method require fewer dense floating-point operations than the random projection method, but require the same number of sparse operations; for this problem, their compute performance should be expected to be roughly equivalent to the random projection method. Only GrABL may be expected to be faster.

9.3 Discussion and Conclusion

Overall the trends developed elsewhere in previous chapters are also apparent in these observations. The leading singular values of A (and eigenvalues of $A^T A$) are well-separated, and become more clustered to the interior. These are the conditions that predict the hybrid method generating

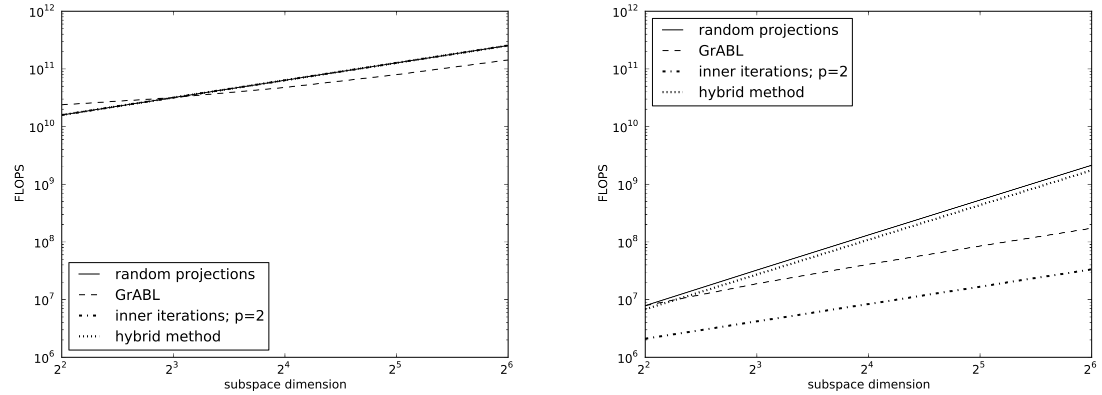

Figure 9.8: Sparse (left) and dense (right) FLOP counts of random projections, GrABL, inner iterations and the hybrid method.

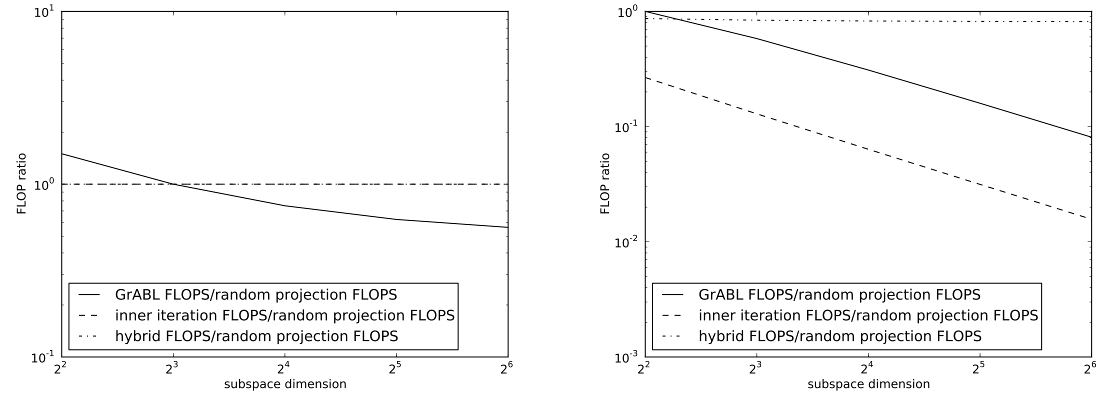

Figure 9.9: Sparse (left) and dense (right) ratios of GrABL, inner iterations and the hybrid method to random projections.

approximations in $\mathscr{K}_2(A, A^2X_0)$ will produce smaller approximation errors than random projections. The random projection method itself may produce smaller approximation errors than a single-vector Krylov subspace method, but at the cost of extra sparse matrix-vector products. When the hybrid method produces smaller errors than random projections, then it is implied that it produces smaller errors than a minimal single-vector Krylov subspace as well.

The rapidly-encountered singular value clustering also implies that GrABL will use a small initial block, and will obtain compute-cost advantages over random projections. Due to the large number of nonzero elements in the matrix, the matrix vector products A^TAx are expensive, and dominate the compute costs. The advantages from block size shrinkage for the shrink-and-iterate, the hybrid random projections-block Krylov method and GrABL are irrelevant. Instead, minimization of matrix-vector multiplications is most important. In such cases; the convergence advantages of methods incorporating some stationary iteration may be erased by simply generating a non-minimal Krylov subspace. However, it is important to note that sparse matrix-vector products are rather amenable to parallelization; whereas some of the dense matrix operations, such as QR factorizations, used in random projections and block Krylov methods, are not as amenable to parallelization. Nevertheless, in instances such as these, GrABL should be expected to present principle compute advantages at larger dimensions. Inner power iteration, shrink-and-iterate and the hybrid method may still produce smaller errors, but compute costs should be expected to be similar to random projections. Finally, the compute costs considered here are only in terms of finding approximations to singular values of A. If $Q^TA^TAQ = \hat{A}^{(k)}$ is the restriction of A^TA to span$\{Q\}$, then estimation of a basis P for the left singular vector space of A may be accomplished with

$$P = AQ\hat{A}^{(k)-1}, \tag{9.14}$$

but this operation requires k additional sparse matrix-vector products. A solution would be to use the Golub-Kahan on A rather than the Lanczos algorithm on A^TA. The asymptotic complexity of this method would be at least as bad as the Lanczos method on AA^T, and the compute cost advantage of the shrink-and-iterate, hybrid and GrABL methods are again relevant.

We have applied random projections, GrABL, single-vector Lanczos and the hybrid random projections/block Krylov subspace methods to the low-rank approximation problem as applied to effect fast signal convolution. All of these methods are intended to be relaxed alternatives to the truncated singular value decomposition, which may require substantial compute time to solve. Substitution of a relaxed alternative to the truncated singular value decomposition will allow for possibly significant compute savings.

There are various methods that may be surrogates for a truncated singular vector space. Single-vector Krylov subspaces have been proposed by several authors [10, 14, 15, 75], but block Krylov subspaces have not been considered. Block methods offer faster convergence of Krylov subspaces to singular vector spaces, and can better exploit local singular value separation. The random projection methods proposed in [36] suggest consideration of block Krylov subspaces. Random projections and single-vector Krylov subspaces both produce compelling results, but the two methods provide distinct advantages. Random projections are successful at excluding singular vectors corresponding to small singular values from its subspace, but require $2k$ matrix multiplications to produce a k-dimensional subspace. Random projections also require $O(nk^2)$ FLOPS to produce a k-dimensional subspace. Single-vector Krylov subspaces only require k matrix multiplications to produce a k-dimensional subspace, but may include singular vectors corresponding to small singular values in its subspace; these vectors are understood as noisy elements of the data. Krylov subspaces also are plagued with loss of orthogonality with continued iteration, and require reorthogonalizarion measures which random projections do not. In terms of cost, single-vector Lanczos methods produce the best low-rank approximations, but random projections may produce the best in terms of error. Composite methods combining some stationary power iteration with Krylov subspace iteration may produce errors that are smaller than single-vector Lanczos and possibly better than random projections, compute times that are no worse than random projections and still require less stabilization than ordinary Lanczos.

Chapter 10

Conclusion

This dissertation has examined acceleration methods for minimal and near-minimal Krylov subspaces for use in generating low-rank matrix approximations. Krylov subspace projections present a relaxed solution from the Frobenius norm-optimal low-rank approximation, but this relaxation may not sacrifice much in terms of approximation error and still yield significant savings in terms of computatiomal time savings. Existing uses of Krylov subspaces for low-rank approximation are intended to present the most compute-time savings over the non-relaxed, optimal solutions. Our methods are intended to produce approximations with smaller error than the least expensive Krylov subspaces examined to date, but are still faster than the fully-converged, norm-optimal truncated SVD or spectral decomposition approximations. Many of the methods examined herein are based on the performance of random projection methods, which are also intended for use as truncated SVD or spectral decomposition alternatives. Low-rank approximations arise for either estimation of the eigenpairs or singular triplets of a matrix, or to reduce the dimension of a input matrix that is inherently low-rank, but is embedded in a high-dimensional space. Such low-rank problems embedded in high-dimensional space arise frequently in engineering, network science, machine learning and many other fields.

Low-rank approximation may be optimally solved with truncated matrix factorizations; the spectral decomposition is truncated when the input matrix is square and positive semi-definite, and the singular value decomposition is truncated when the input matrix is not square or is indefinite. These factorizations produce the best rank-k approximation $\hat{A}^{(k)}$ to the input matrix A in terms of the spectral and Forbenius norm, as the residual is given by

$$\|\hat{A}^{(k)} - A\| \geq \|\hat{A}^{(k)}\| - \|A\| \tag{10.1}$$

with equality when the singular vector spaces of A and $\hat{A}^{(k)}$ coincide. Thus, the minimum value for (10.1) is achieved by $\hat{A}^{(k)} = U_k \Sigma_k V_k^T$ or $\hat{A}^{(k)} = U_k \Lambda_k U_k^T$ where $U_k \Sigma_k V_k^T$ is the truncated SVD or $U_k \Lambda_k U_k^T$ is the spectral decomposition, either using the k eigenpairs or singular triplets with largest magnitude. Regrettably, computing k eigenpairs or singular triplets may be expensive, even when k is small relative to the dimension of A and the computation is performed iteratively. Converged truncated matrix factorizations may be slow to converge for a number of reasons.

Krylov subspaces are an alternative to the use of a eigenvector or singular vector space. Krylov subspaces are defined in terms of the span of a start vector or block vector and a matrix; therefore, they may be generated using only matrix-vector products and an orthonormalziation routine. When the input matrix is Hermitian, then a full Gram-Schmidt process is not required and a Krylov subspace of dimension k may be generated for a $n \times n$ input matrix with asymptotic cost of $O(nk)$. Even a short Krylov subspace will produce good approximations of the leading eigenvalues used in the truncated SVD or spectral decomposition when those eigenvalues are well-separated from the remainder of the spectrum as is typical of a low-rank problem embedded in a high-dimensional space. The fast convergence of precisely those eigenvalues or singular values that matter most combined with the good computational complexity of Krylov subspace generation make their use attractive.

The limited existing work covering Krylov subspaces for low-rank approximation all uses minimal Krylov subspaces. These subspaces are the least expensive to generate, as they only require k matrix multiplications to generate a k-dimensional space. This alternative may be much less expensive than a converged truncated spectral decomposition, especially when k is large, but the difference in error between the Frobenius norm-optimal truncated SVD or spectral decomposition approximation and the Krylov subspace approximation may be non-trivial. One is presented with a choice of an expensive truncated SVD or spectral decomposition with optimal error or an inexpensive Krylov subspace with non-trivial error. With some minor increases in compute time, an approximation with smaller error may be possible.

10. Conclusion

Sources of error in Krylov subspace projections are due to error in eigenvalue estimates, and eigenvalues that are converging to trailing rather than leading eigenvalues. Use of block Krylov subspaces — motivated by random projection methods — combined with Ritz vector truncation of the Krylov subspace both decreases the error of leading eigenvalue approximations and excludes approximations of trailing eigenvalues form the Krylov subspace. Block Krylov subspaces with large start blocks and short Krylov sequences can produce low-rank approximations with lower error than random projections at equivalent computational complexity. Short Krylov sequences also lead to decreased loss of orthogonality, to the point that no reorthgonalization is required. Compute costs are increased by larger block sizes, but adaptive deflation can reduce these to limit the overall computational complexity of subspace generation while minimizing the convergence sacrificed by deflation. Inner power iterations — similar to the power iteration-like refinement used in random projections — also improves convergence in minimal Krylov subspaces by accelerating convergence of leading eigenvalues; this is an alternative to an extended Krylov subspace combined with Ritz value truncation. Finally, for approximation problems that require trailing eigenpairs, shift-and-invert preconditioning may benefit from multiple shifts over inner iteration preconditioning when the number of inner iterations is limited.

We have presented several new Krylov subspace algorithms specialized for the low-rank approximation problem in which some computational resource is constrained, be it memory or compute time. These methods are intended to be alternatives to either minimal single-vector Krylov subspaces proposed previously or the random projection methods. Short block Krylov subspaces with large blocks can produce subspaces that produce low-rank approximation errors smaller than the random projection method, but with compute times that are at worst equivalent and could be up to 25% faster. We have developed an adaptive block Lanczos method that applies random projections to inflate and deflate the Krylov subspace to best exploit local eigenvalue gaps. We have developed new asymptotic bounds that characterize the effects of inner power iteration in a single-vector Krylov subspace for use when limited storage prevents generation of a large Lanczos basis. We have investigated the use of a hybrid random projections-preconditioning approach to select better shifts.

Short block Krylov subspaces have particular advantages, especially when sparse matrix-vector products are inexpensive compared to the dense linear algebra operations used in random projection or Lanczos methods. This is often the case when input matrix is structured and fairly sparse. In these cases, short block Krylov subspaces hold advantages over both the random projection method and single vector Krylov subspaces. The hybrid random projection-block Krylov subspace method may produce smaller approximation errors than random projections, but have equivalent compute time. The hybrid method also requires no reorthogonalization, which is required by single-vector Krylov subspaces methods. Additionally, for large-dimensional subspaces, truncation of a non-minimal Krylov subspace can become the asymptotically dominant compute cost.

When dense matrix operations are the predominant compute costs, reducing block sizes inevitably leads to reduced compute costs. GrABL provides a greedy method to reduce the block size to a minimum, subject to providing best convergence of leading eigenvalues. A reduction of block size also allows for stationary power iteration to be performed only with the initial start block, thereby reducing the number of sparse matrix-vector products. GrABL presents further compute saving over the shrink-and-iterate and hybrid methods, as it will provide compute savings over random projections in all cases. We have observed that when used with a power iteration parameter $p > 1$, GrABL produces smaller low-rank approximation norm than the random projection method, but does so with fewer sparse matrix-vector products or dense linear algebra FLOPS when the subspace dimension is sufficiently large. The compute time advantages will be expanded further when sparse matrix-vector products are expensive.

Stationary power iteration may be integrated as inner iteration in single-vector Lanczos iteration. This accelerates convergence of leading eigenvalues and improved the quality of the subspace for low-rank approximation for many applications. The use of inner power iteration is useful when memory constraints prevent storage of non-minimal Krylov bases. Nevertheless, the gap between a minimal Krylov subspace using p inner power iterations and a non-minimal Krylov subspace that is simply p times longer may be substantial. Though compute costs to truncate the non-minimal subspace may be non-trivial, inner iteration should be limited to cases for which sparse matrix-vector products are cheap and memory is constrained.

Judicious choices of shifts can allow for faster convergence when shift-and-invert preconditioning is used to solve a trailing eigenproblem, as is required for spectral graph problems. Though faster convergence can lead to Krylov subspaces that produce lower-error approximations, using shifts from an initial random projection pass does not appreciably improve approximation over simply using inner power iteration. The extra compute costs for using multiple shifts must be considered. Therefore, inner power iteration is a better choice for accelerating convergence of trailing eigenvalues compared to using multiple shifts.

The specialized Krylov subspace methods developed here will present more choices for low-rank approximation when a truncated SVD or spectral decomposition is optimal. Rather than being limited to the expensive truncated matrix factorization or the inexpensive Krylov subspace or random projections approximation that still has some error, one may apply an acceleration method introduced here to obtain better convergence at modest compute cost, all while remaining within modest memory footprints. The improved stability of short Krylov sequences may allow parallel Krylov subspace methods to become simpler and eliminate some inter-node communication. Future extensions of this effort would refine and simplify the bounds on the errors of block Krylov subspaces to lend further insight into those spectral properties that cause random projections to outperform block Krylov subspaces. Finally, though these acceleration methods were designed specifically for minimal Krylov subspaces, integration of acceleration methods to restarted Krylov subspaces may yield iterative eigenvalue computation routines that converge faster for certain classes of Hermitian matrices.

Bibliography

[1] UA Census 2000. Ua census 2000 tiger/line files. `http://www.census.gov/geo/www/tiger/tigerua/ua_tgr2k.html`, May 2002.

[2] A.C. Antoulas, D.C. Sorensen, and S. Gugercin. A survey of model reduction methods for large-scale systems. *Contemporary mathematics*, 280:193–220, 2001.

[3] J. Baglama and L. Reichel. Augmented implicitly restarted Lanczos bidiagonalization methods. *SIAM Journal on Scientific Computing*, 27(1):19–42, 2006.

[4] Z. Bai. *Templates for the solution of algebraic eigenvalue problems*, volume 11. Society for Industrial Mathematics, 2000.

[5] Z. Bai, D. Day, and Q. Ye. ABLE: An adaptive block Lanczos method for non-Hermitian eigenvalue problems. *SIAM Journal on Matrix Analysis and Applications*, 20:1060, 1999.

[6] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[7] M.W. Berry, Z. Drmac, and E.R. Jessup. Matrices, vector spaces, and information retrieval. *SIAM review*, 41(2):335–362, 1999.

[8] M.W. Berry, S.T. Dumais, and G.W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM review*, 37(4):573–595, 1995.

[9] K. Blom and A. Ruhe. Information retrieval using very short Krylov sequences. *Computational information retrieval*, pages 41–56, 2001.

[10] K. Blom and A. Ruhe. A Krylov subspace method for information retrieval. *SIAM J. Matrix Anal. Appl.*, 26:566–582, February 2005.

[11] D. Calvetti, L. Reichel, and D.C. Sorensen. An implicitly restarted Lanczos method for large symmetric eigenvalue problems. *Electronic Transactions on Numerical Analysis*, 2(1):21, 1994.

[12] A. Chapman and Y. Saad. Deflated and augmented Krylov subspace techniques. *Numerical Linear Algebra with Applications*, 4(1):43–66, 1997.

[13] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current Science*, 78(7):808–817, 2000.

[14] J. Chen, H. Fang, and Y. Saad. Fast approximate k NN graph construction for high dimensional data via recursive Lanczos bisection. *The Journal of Machine Learning Research*, 10:1989–2012, 2009.

[15] J. Chen and Y. Saad. Lanczos vectors versus singular vectors for effective dimension reduction. *IEEE Transactions on Knowledge and Data Engineering*, pages 1091–1103, 2009.

[16] J. Cullum and W.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*, volume 13, pages 505–509. IEEE, 1974.

[17] J.K. Cullum and R.A. Willoughby. *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. 1.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002.

[18] T.A. Davis. University of Florida sparse matrix collection. In *NA Digest*. Citeseer, 1994.

[19] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[20] I.S. Du, R.G. Grimes, and J.G. Lewis. Users' guide for the Harwell-Boeing sparse matrix collection (release i). Technical report, Report RAL-92-086, Atlas Centre, Rutherford Appleton Laboratory, Didcot, Oxon, UK, 1992.

[21] S.T. Dumais. Latent semantic indexing (LSI): TREC-3 report. *NIST special publication*, (500225):219–230, 1995.

[22] L. Eldén. Partial least-squares vs. Lanczos bidiagonalization–i: analysis of a projection method for multiple regression. *Computational Statistics & Data Analysis*, 46(1):11 – 31, 2004.

[23] L. Elden. *Matrix Methods in Data Mining and Pattern Recognition*. Society for Industrial and Applied Mathematics., 2007.

[24] T. Ericsson and A. Ruhe. The spectral transformation lanczos method for the numerical solution of large sparse generalized symmetric eigenvalue problems. *Mathematics of Computation*, 35(152):1251–1268, 1980.

[25] P. Feldmann and R.W. Freund. Efficient linear circuit analysis by Padé approximation via the Lanczos process. In *Proceedings of the conference on European design automation*, page 175. IEEE Computer Society Press, 1994.

[26] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[27] R.W. Freund. Model reduction methods based on Krylov subspaces. *Acta Numerica*, 12(-1):267–319, 2003.

[28] H.A. Gaberson. Pseudo velocity shock spectrum rules for analysis of mechanical shock. *IMAC XXV, Orlando, FL*, 2007.

[29] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.

[30] G. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, 2(2):205–224, 1965.

[31] G.H. Golub, F.T. Luk, and M.L. Overton. A block Lanczos method for computing the singular values and corresponding singular vectors of a matrix. *ACM Transactions on Mathematical Software (TOMS)*, 7(2):149–169, 1981.

[32] G.H. Golub and H.A. Van der Vorst. Eigenvalue computation in the 20th century. *Journal of Computational and Applied Mathematics*, 123(1):35–65, 2000.

[33] R.G. Grimes, J.G. Lewis, and H.D. Simon. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *SIAM Journal on Matrix Analysis and Applications*, 15(1):228–272, 1994.

[34] E.J. Grimme. *Krylov projection methods for model reduction*. PhD thesis, University of Illinois, 1997.

[35] E.J. Grimme, D.C. Sorensen, and P. Dooren. Model reduction of state space systems via an implicitly restarted Lanczos method. *Numerical Algorithms*, 12(1):1–31, 1996.

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

[37] K.M. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 17(3):219–229, 1970.

[38] M. Hochbruck and C. Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 34(5):1911–1925, 1997.

[39] I.T. Jolliffe. *Principal component analysis*. Springer Series in Statistics, 2002.

[40] S. Kaniel. Estimates for some computational techniques in linear algebra. *Mathematics of Computation*, 20(95):369–378, 1966.

[41] E. Kokiopoulou, C. Bekas, and E. Gallopoulos. Computing smallest singular triplets with implicitly restarted Lanczos bidiagonalization. *Appl. Numer. Math.*, 49:39–61, April 2004.

[42] E. Kokiopoulou and Y. Saad. Polynomial filtering in latent semantic indexing for information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 104–111. ACM, 2004.

[43] Y. Koren. On spectral graph drawing. *Computing and Combinatorics*, pages 496–508, 2003.

[44] C. Lanczos. Iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45:255–282, 1950.

[45] K.C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 27(5):684–698, 2005.

[46] R.B. Lehoucq, D.C. Sorensen, and C. Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, volume 6. SIAM, 1998.

[47] X.S. Li. An overview of SuperLU: Algorithms, implementation, and user interface. *ACM Transactions on Mathematical Software (TOMS)*, 31(3):302–325, 2005.

[48] Y.C. Liang, H.P. Lee, S.P. Lim, W.Z. Lin, K.H. Lee, and C.G. Wu. Proper orthogonal decomposition and its applicationsâĂŤpart i: Theory. *Journal of Sound and Vibration*, 252(3):527–544, 2002.

[49] A.M. Martínez and A.C. Kak. PCA versus LDA. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(2):228–233, February 2001.

[50] P.G. Martinsson, A. Szlam, and M. Tygert. Normalized power iterations for the computation of SVD. *Manuscript., Nov*, 2010.

[51] R.B. Morgan and D.S. Scott. Preconditioning the Lanczos algorithm for sparse symmetric eigenvalue problems. *SIAM Journal on Scientific Computing*, 14(3):585–593, 1993.

[52] R.B. Morgan and M. Zeng. Harmonic projection methods for large non-symmetric eigenvalue problems. *Numerical linear algebra with applications*, 5(1):33–55, 1998.

[53] M.E.J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104, 2006.

[54] A. Ng. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14: Proceeding of the 2001 Conference*, pages 849–856.

[55] C.C. Paige. *The computation of eigenvalues and eigenvectors of very large sparse matrices*. University of London, 1971.

[56] C.C. Paige, B.N. Parlett, and H.A. Van der Vorst. Approximate solutions and eigenvalue bounds from Krylov subspaces. *Numerical linear algebra with applications*, 2(2):115–133, 1995.

[57] B.N. Parley and D.S. Scott. The Lanczos algorithm with selective orthogonalization. *Mathematics of computation*, 33(145):217–238, 1979.

[58] B.N. Partlett and S.H. Crandall. The symmetric eigenvalue problem. *Journal of Applied Mechanics*, 48:988, 1981.

[59] K. Pearson. LIII. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559–572, 1901.

[60] A. Pothen, H.D. Simon, and K.P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11:430, 1990.

[61] H. Qiu and E.R. Hancock. Clustering and embedding using commute times. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1873–1890, 2007.

[62] M. Rathinam and L.R. Petzold. A new look at proper orthogonal decomposition. *SIAM Journal on Numerical Analysis*, 41(5):1893–1925, 2003.

[63] C.X. Ren and D.Q. Dai. Bilinear Lanczos components for fast dimensionality reduction and feature extraction. *Pattern Recognition*, 2010.

[64] A. Ruhe. Implementation aspects of band Lanczos algorithms for computation of eigenvalues of large sparse symmetric matrices. *Mathematics of Computation*, pages 680–687, 1979.

[65] A. Ruhe. Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils. *SIAM Journal on Scientific Computing*, 19:1535, 1998.

[66] Y. Saad. *Calcul de valeurs propres de grandes matrices hermitiennes par des techniques de partitionnement*. PhD thesis, Institut National Polytechnique de Grenoble-INPG, 1974.

[67] Y. Saad. On the rates of convergence of the Lanczos and the block-Lanczos methods. *SIAM Journal on Numerical Analysis*, 17(5):687–706, 1980.

[68] Y. Saad. Analysis of some Krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 29(1):209–228, 1992.

[69] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM Journal on Scientific Computing*, 14(2):461–469, 1993.

[70] M. Saerens, F. Fouss, L. Yen, and P. Dupont. The principal components analysis of a graph, and its relationships to spectral clustering. *Machine Learning: ECML 2004*, pages 371–383.

[71] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[72] D.S. Scott. The advantages of inverted operators in Rayleigh-Ritz approximations. *SIAM Journal on Scientific and Statistical Computing*, 3(1):68–75, 1982.

[73] H.D. Simon. Analysis of the symmetric Lanczos algorithm with reorthogonalization methods. *Linear algebra and its applications*, 61:101–131, 1984.

[74] H.D. Simon. The Lanczos algorithm with partial reorthogonalization. *Mathematics of Computation*, 42(165):115–142, 1984.

[75] H.D. Simon and H. Zha. Low-rank matrix approximation using the Lanczos bidiagonalization process with applications. *SIAM Journal on Scientific Computing*, 21(6):2257–2274, 2000.

[76] D. Skillicorn. *Understanding complex datasets: data mining with matrix decompositions*. Chapman & Hall/CRC, 2007.

[77] G.L.G. Sleijpen and H.A. Van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM Review*, pages 267–293, 2000.

[78] K. Sparck Jones and C.A. Webster. Research on relevance weighting 1976-1979. *Computer Laboratory, University of Cambridge (also BL R&D Report 5553)*, 1980.

[79] F. Tsalakanidou, D. Tzovaras, and M.G. Strintzis. Use of depth and colour Eigenfaces for face recognition. *Pattern Recognition Letters*, 24(9):1427–1435, 2003.

[80] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

[81] R.R. Underwood. *An iterative block Lanczos method for the solution of large sparse symmetric eigenproblems.* PhD thesis, Stanford, CA, USA, 1975. AAI7525622.

[82] J.S. Vandergraft. Generalized Rayleigh methods with applications to finding eigenvalues of large matrices. *Linear Algebra Appl.*, 4:353–368, 1971.

[83] H. Wold. *Partial Least Squares*. John Wiley & Sons, Inc., 2004.

[84] K. Wu and H. Simon. Thick-Restart Lanczos method for large symmetric eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 22(2):602–616, May 2000.

[85] T. Yang. Theoretical error bounds on the convergence of the Lanczos and block-Lanczos methods. *Computers & Mathematics with Applications*, 38(9):19–38, 1999.

[86] Q. Ye. An adaptive block Lanczos algorithm. *Numerical Algorithms*, 12(1):97–110, 1996.

[87] L. Yen, D. Vanvyve, F. Wouters, F. Fouss, M. Verleysen, and M. Saerens. Clustering using a random walk based distance measure. In *Proceedings of the 13th Symposium on Artificial Neural Networks (ESANN 2005)*, pages 317–324. Citeseer, 2005.

[88] Y. Zhou and Y. Saad. Block Krylov-Schur method for large symmetric eigenvalue problems. *Numerical Algorithms*, 47(4):341–359, 2008.

1 DEFENSE TECHNICAL
(PDF) INFORMATION CTR
DTIC OCA

2 DIRECTOR
(PDF) US ARMY RESEARCH LAB
RDRL CIO LL
IMAL HRA MAIL & RECORDS MGMT

1 GOVT PRINTG OFC
(PDF) A MALHOTRA

1 DIR US ARMY EVALUATION CTR HQ
(HC) TEAE SV
P A THOMPSON
2202 ABERDEEN BLVD 2ND FL
APG MD 21005-5001

5 DIR USARL
(2 HC RDRL SL
3 PDF) J BEILFUSS (HC)
P TANENBAUM (HC)
RDRL SLB S
M PERRY
RDRL SLB W
A BREUER
RDRL SLE
R FLORES