2009

Iterative Methods for Computing Eigenvalues and Exponentials of Large Matrices

Ping Zhang
University of Kentucky, seraphzhp@gmail.com

ABSTRACT OF DISSERTATION

Ping Zhang

The Graduate School
University of Kentucky
2009

Iterative Methods for Computing Eigenvalues and Exponentials of Large Matrices

ABSTRACT OF DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Arts and Sciences
at the University of Kentucky

By
Ping Zhang
Lexington, Kentucky

Director: Dr. Qiang Ye, Professor of Mathematics
Lexington, Kentucky 2009

ABSTRACT OF DISSERTATION

Iterative Methods for Computing Eigenvalues and Exponentials of Large Matrices

In this dissertation, we study iterative methods for computing eigenvalues and exponentials of large matrices. These types of computational problems arise in a large number of applications, including mathematical models in economics, physical and biological processes. Although numerical methods for computing eigenvalues and matrix exponentials have been well studied in the literature, there is a lack of analysis in inexact iterative methods for eigenvalue computation and certain variants of the Krylov subspace methods for approximating the matrix exponentials. In this work, we proposed an inexact inverse subspace iteration method that generalizes the inexact inverse iteration for computing multiple and clustered eigenvalues of a generalized eigenvalue problem. Compared with other methods, the inexact inverse subspace iteration method is generally more robust. Convergence analysis showed that the linear convergence rate of the exact case is preserved. The second part of the work is to present an inverse Lanczos method to approximate the product of a matrix exponential and a vector. This is proposed to allow use of larger time step in a time-propagation scheme for solving linear initial value problems. Error analysis is given for the inverse Lanczos method, the standard Lanczos method as well as the shift-and-invert Lanczos method. The analysis demonstrates different behaviors of these variants and helps in choosing which variant to use in practice.

KEYWORDS: Krylov subspace, GMRES, Arnoldi, Lanczos, Matrix exponential

Author's signature: Ping Zhang

Date: December 9, 2009

Iterative Methods for Computing Eigenvalues and Exponentials of Large Matrices

By
Ping Zhang

Director of Dissertation:	Qiang Ye
Director of Graduate Studies:	Qiang Ye
Date:	December 9, 2009

DISSERTATION

Ping Zhang

The Graduate School
University of Kentucky
2009

Iterative Methods for Computing Eigenvalues and Exponentials of Large Matrices

DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Arts and Sciences
at the University of Kentucky

By
Ping Zhang
Lexington, Kentucky

Director: Dr. Qiang Ye, Professor of Mathematics
Lexington, Kentucky 2009

ACKNOWLEDGMENTS

The process of completing the research for and the writing of this dissertation has been a long and exciting process, and many people have supported me along the way. First, I would like to thank Dr. Ye for his constant guidance, encouragement, and dedication to both my studies and my research. His amazing discussion through out the past years inspired me a lot of ways. As I look back, I see the many different ways that he has worked to train me to be an independent mathematician. I gave all my gratefulness to him from the bottom of my heart. Next, I would like to thank the other members of my doctoral committee: Dr. Russell Brown, Dr. Larry Harris, Dr. Yuming Zhang (Electrical Engineering), and Dr. Caicheng Lu (Electrical and Computer Engineering). I can not begin to thank my family enough for the constant support and encouragement that they have provided me.

Dedicated to the love of my life, Qian Sun!

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

We are concerned with two numerical problems for large matrices, i.e., computing eigenvalues and exponentials of large matrices by using iterative methods. In particular, we are interested in computing the eigenvalues of a generalized eigenvalue problems by using inexact inverse subspace iteration and computing the product of a matrix exponential with a vector by using the Lanczos algorithm. These types of problems arise in many scientific and engineering applications. The matrices involved are often large and sparse in such applications. Iterative methods, which use matrix-vector multiplications, are well suited for such large scale problems.

The well-known iterative methods for solving eigenvalue problems are the power method (the inverse iteration), the subspace iteration, the Krylov subspace methods and the Jacobi-Davidson algorithm. Traditionally, if the extreme eigenvalues are not well separated or the eigenvalues sought are in the interior of the spectrum, a shift-and-invert transformation (a preconditioning technique) has to be used in combination with these eigenvalue problem solvers. The shift-and-invert transformation requires the solution of a shifted linear systems at each iteration. Owing to the size of the matrices, direct solution of the shifted matrix (i.e., factorization) may not be practical. Alternatively, iterative method (inner iterations) can be used to solve the shift-and-invert equation, which leads to two levels of iterations, called inner-outer iterations. The use of inner-outer iterations (or inexact iterations) has been studied for several methods, such as the Davidson and the Lanczos algorithm [8, 20, 44, 45], the inverse iteration [18, 28, 30], the rational Arnoldi algorithm and truncated RQ iterations [27, 47] and the Jacobi-Davidson method [42].

One of the challenges in implementing inexact iterations is in how the accuracy of the inner iteration affects the convergence characteristic of the outer iteration.

Although the inner-outer iteration technique has been used successfully for several methods [18, 35, 47], the theoretical behaviors are still not well understood for many of them. The first part of this thesis is to study inexact inverse subspace iteration. We develop the inexact inverse subspace iteration method which enjoys similar convergence characteristics as the inexact inverse iteration [18], but can handle multiple and clustered eigenvalues. We present convergence analysis of the inexact inverse subspace iteration method. Our results demonstrate that inexact inverse subspace iteration has very robust convergence characteristic and can withstand very large errors in the inner iterations.

On the other hand, computation of matrix exponentials is a well studied subject. A survey of earlier literature on matrix exponential computation can be found in [33]. Many numerical methods have been developed to compute the matrix exponential, including series methods, Pade approximation, ordinary differential equation methods and matrix decomposition and splitting methods. These methods are primarily for small dense matrices. For large matrices, Krylov subspace approximation has been shown to be very successful in many applications. Saad [17] introduces Krylov subspace approximation method for matrix exponentials and provides a priori and a posterior error estimates. Several other analysis and a combination with shift-and-invert technique have been developed, see [16, 25, 49]. The goal of the second part of the thesis is to present a novel technique for analyzing Lanczos methods for approximating matrix exponentials, which are based on recent results on decay of matrix function of banded matrices. We also propose an inverse Lanczos algorithm, which may be competitive in some situation. A comparative study, both theoretically and numerically, will be carried out for several Lanczos based methods.

The remainder of this thesis is organized as follows.

Chapter 2 reviews the basic background materials which will be used in the later chapters. In particular, section 2.1 introduces the single and multiple vector iteration methods; section 2.2 covers the Krylov subspace methods; section 2.3 presents the Jacobi-Davidson method. Since GMRES method is used in solving the linear system of both the Jacobi-Davidson method and the inexact inverse subspace iteration method in chapter 3, we present it as a separate subsection in section 2.4. Section 2.5 introduces decay bound for matrix functions that are used in Chapter 4.

In Chapter 3, we present an inexact inverse subspace iteration method for computing a few smallest eigenpairs of the generalized eigenvalue problem $Ax = \lambda Bx$. In section 3.1 we first introduce the inexact inverse subspace iteration method. Section 3.2 analyzes the convergence rate of the approximate subspace generated by the inexact inverse subspace iteration algorithm and how the accuracy in the inner iteration influences the convergence rate of the outer iteration. In section 3.3, we discuss the convergence of the block of vectors computed, or the basis of the subspace generated. With a scaling to fix the maximum entry of each column of X_k to be positive, the convergence of the residual of the inexact inverse subspace iteration algorithm has the same rate as the inner iteration threshold. In section 3.4, numerical examples are given to verify the theoretical results and demonstrate competitiveness of the method.

Chapter 4 is devoted to computing the product of a matrix exponential and a vector by using the Lanczos methods. In section 4.1, we start out with the standard Lanczos approximation to matrix exponentials, with a posterior and a priori error bounds. In section 4.2, we propose inverse Lanczos approximation method to approximate $e^{-\tau A}v$ and discuss a prior and a posterior error bounds to illustrate the key factors that influence the error. Section 4.3 is devoted to the shift-and-invert Lanczos approximation. We also give a priori and a posterior error bounds for this method. Numerical examples are given in section 4.4 to verify the theoretical results and compare the performances of each of the Lanczos methods when dealing with a general matrix. In section 4.5 we use all three Lanczos methods to compute the system of ordinary differential equations with a time-dependent forcing term. We compare the performance of three Lanczos methods.

Finally, we present some concluding remarks and future work in Chapter 5.

1.0.1 Notation

Throughout this thesis, we shall use the following notations.

- $\mathbb{R}^{n\times m}$: set of all $n \times m$ matrices with entries in $\mathbb{R}$. $\mathbb{R}$ is the set of real number
- $\mathbb{C}^{n\times m}$: set of all $n \times m$ matrices with entries in $\mathbb{C}$. $\mathbb{C}$ is the set of complex number
- A^H : conjugate transpose of A
- A^T : transpose of A
- I : identity matrix
- e_j : the j^{th} column of the identity matrix.
- $A_{(i:j,k:l)}$: submatrix of A, consisting of the intersections of rows i to j and columns k to l, and when $i : j$ is replaced by :, it means all rows. Similar notations apply for columns
- $\lambda_{\max}(A)$: the largest eigenvalue of matrix A
- $\lambda_{\min}(A)$: the smallest eigenvalue of matrix A
- $\kappa(A)$: spectral condition number of A
- $(u, v) = v^H u$: inner(dot) product
- $\|A\|_2$: 2-norm of matrix A
- $\|x\|_2$: 2-norm of vector x

- $\|A\|_\infty$: infinity-norm of matrix A
- $\|x\|_\infty$: infinity-norm of vector x
- argmin $_y\|f(y)\|_2$: the value of y at which $\|f(y)\|_2$ is minimized
- $\max\{x\}$: the maximal element of vector x

Chapter 2 Preliminaries

In this chapter, we will present some preliminary materials that will be used in the later chapter. Most of them can be found in standard texts, [3, 9, 39, 48]. The outline of this chapter is as follows. In section 2.1, we introduce the classic results on single and multiple vector iterations method for eigenvalue problem. Krylov subspace method is presented in section 2.2. Jacobi-Davidson method and the GMRES method are discussed in section 2.3. Section 2.1, 2.2 and section 2.3 form theoretical foundations for our works in chapter 3. Finally section 2.4 presents some concepts and theoretical results on entry decay of the matrix function.

2.1 Single and multiple vector iterations

2.1.1 Power method

The power method is one of the oldest techniques for finding an eigenvalue and an eigenvector. It generates the sequence of vectors $A^k v_0$ where v_0 is some nonzero initial vector. This sequence of vectors when normalized appropriately converges to a dominant eigenvector, i.e., an eigenvector associated with the eigenvalue of the largest modulus. The scaling of the iteration vectors is necessary in order to prevent over- or underflow. The most commonly used normalization is to ensure that the largest component of the current iterate is equal to one. The standard algorithm is as follows:

```
**A**lgorithm 1. Power Method

Given $x_0$ as the initial guess

For $i = 0, 1, 2, \ldots$ until convergence
```

```
        $y_{i+1} = Ax_i$
        $x_{i+1} = y_{i+1}/\max\{y_{i+1}\}$
        $\mu_{i+1} = \frac{x_{i+1}^T A x_{i+1}}{x_{i+1}^T x_{i+1}}$;
        if $\|y_{i+1} - \mu_{i+1} x_i\| \leq \epsilon |\mu_{i+1}|$;
    End
```

Remark. $\max\{y\}$ is a component of the vector y which has the maximum modulus. By updating $x_{i+1} = y_{i+1}/\max\{y_{i+1}\}$, y_{i+1} is normalized with respect to the $\infty-$norm.

Theorem 2.1.1. *[39] Assume that there is one and only one eigenvalue λ_1 of A of largest modulus and that λ_1 is semi-simple. Then either the initial vector x_0 has no component in the eigenvector corresponding to λ_1 or the sequence of vectors generated by Algorithm 1 converges to an eigenvector associated with λ_1 and μ_{i+1} converges to λ_1 linearly at the rate $\left|\frac{\lambda_2}{\lambda_1}\right|$ where λ_2 is the second largest eigenvalue in modulus.*

This ratio $\left|\frac{\lambda_2}{\lambda_1}\right|$ represents the spectral radius of the linear operator $\frac{1}{\lambda_1}A$ restricted to the subspace that excludes the invariant subspace associated with the dominant eigenvalue. It is a common situation that the eigenvalues λ_1 and λ_2 are very close from one another. As a result convergence may be extremely slow.

2.1.2 Inverse iteration

The inverse iteration method is also called inverse power method. It applies the power method to A^{-1} instead of A. The algorithm is as follows:

```
    **A**lgorithm 2. Inverse Iteration Algorithm

    Given $x_0$ the initial guess

    For $i = 0, 1, \ldots$ until convergence
    $y_{i+1} = (A - \sigma I)^{-1} x_i$
    $x_{i+1} = y_{i+1} / \max\{y_{i+1}\}$
    $\mu_{i+1} = \frac{x_{i+1}^T A x_{i+1}}{x_{i+1}^T x_{i+1}}$;
    if $\|Ax_{i+1} - \mu_{i+1} x_{i+1}\|_2 \le \epsilon |\mu_i|$. Stop the iteration.
End
```

The advantage of inverse iteration combined with shift over the power method is the ability to converge to any desired eigenvalue. By choosing a shift close to a desired eigenvalue, inverse iteration can converge very quickly. This method is particularly effective when we have a good approximation to an eigenvalue and only want to compute this eigenvalue and its corresponding eigenvector. However, inverse iteration does require a factorization of the matrix $A - \sigma I$, making it less attractive when this factorization is expensive. Again, the convergence rate can be close to 1 if the ratio of the first largest eigenvalue and the second largest eigenvalue of $(A - \sigma I)^{-1}$ in magnitude is close to 1.

2.1.3 Subspace iteration

Subspace iteration is also called orthogonal iteration or simultaneous iteration. It is a straightforward block generalization of the power method. The QR factorization is a normalization process that is similar to the normalization used in the power method. This method is used to calculate p eigenvalues of A that are largest in absolute value. The convergence rate depends on $|\lambda_{p+1}/\lambda_p|$ where λ_{p+1} is the $(p+1)^{st}$ largest eigenvalue of A in modulus.

Algorithm 3. Subspace Iteration Algorithm

Given orthonormal Q_0 the initial guess

```
For $k = 0, 1, \ldots$ until convergence
    $V_k = AQ_k$
    $V_k = Q_{k+1}R_{k+1}$
End
```

Although the method is not competitive with other projections methods to be covered in later sections, but it is very reliable; its implementation is comparatively simple and easy to use. Moreover, combined with shift-and-invert enhancement or Chebyshev acceleration, it may be more competitive. So it still is one of the widely used methods in some applications such as structural engineering. For more details about subspace iteration, please refer to [39].

2.2 Krylov Subspace Methods

Krylov subspace methods for the approximation of eigenvalues simply apply the Rayleigh-Ritz method to a Krylov subspace and extract approximations from a subspace of the form

$$K_m(A, v) = \text{ span } \{v, Av, A^2v, \ldots, A^{m-1}v\}.$$

In contrast to subspace iteration, the dimension of the subspace of approximates increases by one at each step of the approximation process. Each vector $u \in K_m(A, v)$ may be represented as $u = p(A)v$ where p is a polynomial of degree less than or equal to $m - 1$. Of course, the dimension of the Krylov space depends on the initial vector v; namely, if v is an eigenvector of A, then the Krylov space $K_m(A, v)$ has dimension one, regardless of the value of m. Furthermore, the dimension of the Krylov subspace $K_m(A, v)$ is always less than or equal to the degree of the minimal polynomial of v with respect to A. It should also be pointed out that the basis $\{v, Av, A^2v, ..., A^{m-1}v\}$ is never used in practice since the vectors become increasingly dependent, as the sequence $\{A^i v\}_{i=0}^{\infty}$ converges to the dominant eigenvector for most choices of v. Instead, the Arnoldi process for non-Hermitian matrices or the Lanczos process for Hermitian matrices is used to develop an orthonormal basis of the Krylov subspace $K_m(A, v)$ with respect to a certain inner product. Once a suitable basis Q_m is constructed and the Rayleigh-Ritz projection A_m is formed, Ritz values and Ritz vectors must be extracted.

2.2.1 The Arnoldi Process

Arnoldi's method is an orthogonal projection method onto K_m for general non-Hermitian matrices. The procedure was introduced in 1951 as a means of reducing a dense matrix into Hessenberg form. Arnoldi introduced this method precisely in this manner and he hinted that the process could give good approximations to some eigenvalues if stopped before completion. It was later discovered that this strategy lead to a good technique for approximating eigenvalues of large sparse matrices. Given an inner product, $\langle \cdot, \cdot \rangle$, the Arnoldi process develops an orthonormal basis $\{v_1, v_2, \ldots, v_m\}$ of the Krylov subspace $K_m(A, x)$. Each vector $v_{j+1}, 1 \leq j \leq m$ is generated by the following recurrence

$$h_{j+1,j} v_{j+1} = A v_j - \sum_{1 \leq k \leq j} h_{kj} v_k$$

where $h_{ij} = \langle v_i, A v_j \rangle$. Letting H_m be the square matrix with entries h_{ij} and V_m be the matrix whose j^{th} column is v_j , we may rewrite the above equation to obtain

$$A V_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^H$$

where e_m is the m^{th} canonical basis vector of $\mathbf{R^m}$. By construction, V_m is orthogonal with respect to the inner product $\langle \cdot, \cdot \rangle$ and H_m is upper Hessenberg.

To see that the columns of V_m produced by the Arnoldi process do indeed form a basis for $K_m(A, v)$ we proceed inductively. Clearly v_1 forms a basis for $K_1(A, v)$. Let us assume that $v_j \neq 0$ for $j : 1 \leq j \leq m$. Suppose now that $V_j = (v_1, \ldots, v_j)$

forms a basis for $K_j(A, v)$, and note that v_{j+1} is defined as a linear combination of the columns of V_j and the vector Av_j. It is easy to see therefore that $\text{span}\ \{v_1, ..., v_{j+1}\} \subseteq K_{j+1}(A, v)$. Now, since the collection $\{v_1, ..., v_{j+1}\}$ is orthonormal with respect to some inner product, then the collection is linearly independent and so it must follow that $\text{span}\ \{v_1, ..., v_{j+1}\} = K_{j+1}(A, v)$.

When $h_{j+1,j} = 0$ for $j : 1 \leq j \leq m$, the Arnoldi process is said to suffer breakdown. Encountering such a breakdown is fortuitous as it implies that the Krylov subspace is $A-$invariant. Ritz values extracted from the Krylov subspace will therefore be exact eigenvalues. Because of the explicit orthogonalization of each new Arnoldi vector against all previous Arnoldi vectors, the Arnoldi process can be computationally expensive. In exact arithmetic, one variant of the algorithm is as follows:

```
**A**lgorithm 4. Arnoldi Algorithm

Choose a vector $v_1$ of norm 1.

For $j = 1, \ldots, m$ compute:

  $h_{ij} = (Av_j, v_i), i = 1, 2, \ldots, j,$

  $w_j = Av_j - \sum_{i=1}^{j} h_{ij} v_i,$

  $h_{j+1,j} = \|w_j\|_2$, if $h_{j+1,j} = 0$ stop

  $v_{j+1} = w_j / h_{j+1,j}.$

  Compute the eigenpairs $(\lambda_j, u_j)$ of $H_j$, and select $r$ Ritz pairs $(\lambda_j, x_j)$,where
  $x_j = V_j u_j$, as approximations to the desired eigenpairs.

End
```

2.2.2 The Lanczos Process

The Hermitian Lanczos algorithm can be viewed as a simplification of Arnoldi's method for the particular case when the matrix is Hermitian. The principle of the method is therefore the same in that it is a projection technique on a Krylov subspace. On the theoretical side there is also much more that can be said on the Lanczos algorithm than there is on Arnoldi's method due to the simplification. Specifically, we have the following theorem.

Theorem 2.2.1. *Assume that Arnoldi's method is applied to a Hermitian matrix A. Then the coefficients h_{ij} generated by the algorithm are real and such that*

$$h_{ij} = 0 \quad \text{for } 1 \leq i < j - 1$$
$$h_{j,j+1} = h_{j+1,j} \quad j = 1, 2, \ldots, m$$

In other words the matrix H_m obtained from the Arnoldi process is real, tridiagonal, and symmetric.

The standard notation used to describe the Lanczos algorithm is obtained by setting

$$\alpha_j = h_{jj}, \quad \beta_j = h_{j-1,j},$$

which leads to the following form of the modified Gram Schmidt variant of Arnoldi's method.

Algorithm 5. Lanczos Algorithm

```
$v_1 = b/\|b\|_2, \beta_1 = 0, v_0 = 0$
For $j = 1$ to $k$
  $z = Av_j - \beta_j v_{j-1}$
  $\alpha_j = v_j^T z$
```

```
        $z = z - \alpha_j v_j$;
        $\beta_{j+1} = \|z\|_2$
        if $\beta_j = 0$ quit
        $v_{j+1} = z/\beta_{j+1}$
        Compute the eigenpairs $(\lambda_j, u_j)$ of $T_j$, and select $r$ Ritz pairs $(\lambda_j, x_j)$,where
        $x_j = V_j u_j$, as approximations to the desired eigenpairs.
    End
```

Proposition 2.2.2. *Denote by V_m the $n \times m$ matrix with column vectors $v_1, \dots, v_m$ and by T_m the $m \times m$ tridiagonal matrix whose nonzero entries are defined by the algorithm. Then the following relations hold:*

$$AV_m = V_m T_m + \beta_{m+1,m} v_{m+1} e_m^T,$$

$$V_m^T A V_m = T_m$$

Throughout this work we will refer to the method of generating a basis of a Krylov subspace by the name Lanczos if the basis may be generated by a short (three-term) recurrence. Reference to the Arnoldi process will be reserved for a method relying on a long recurrence for the generation of a basis of a Krylov subspace. An important and rather surprising property is that the above simple algorithm guarantees, at least in exact arithmetic, that the vectors $v_i, i = 1, 2, \dots,$ are orthogonal. In reality, exact orthogonality of these vectors is only observed at the beginning of the process. Ultimately, the v_i's start losing their global orthogonality very rapidly. There are some ways to recover the orthogonality, like partial or selective orthogonalization,etc. In fact, the process terminates after k steps, if the starting vector has components only in the directions of eigenvector corresponding to k different eigenvalues. In the case of finite termination, we have reduced the matrix A to tridiagonal form with respect to an invariant subspace, and in fact, Lanczos's algorithm was initially viewed as a finite reduction algorithm for A.

2.3 Jacobi-Davidson methods

2.3.1 Jacobi-Davidson method

The Jacobi-Davidson method is a Rayleigh Ritz method for solving the eigenvalue problem of large-scale sparse matrices. It is particularly competitive for seeking interior eigenvalues. There are two stages involved in this method,i.e. subspace extraction and subspace expansion. The subspace extraction stage applies Galerkin approach to extract the Ritz vectors of A. And then computing the orthogonal correction to expend the subspace. Specifically, the two-step procedure is as follows: First, the Galerkin condition is

$$AV_m s - \theta V_m s \perp \{v_1, \ldots, v_m\}$$

which leads to solving the reduce system

$$V_m^T A V_m s - \theta s = 0,$$

where V_m represents the matrix with columns v_1 to v_m which is the orthonormal basis for the subspace. The solution of this reduced system $(\theta_j^m, u_j^m = V_m s_j^m)$ are called the Ritz values and Ritz vectors of A with respect to the subspace spanned by the columns of V_m. These Ritz pairs are approximations for eigenpairs of A. While different from the way in which the Krylov subspace method construct the orthonormal basis vectors, the Jacobi-Davidson method computes the orthogonal correction t for u_j^m so that

$$A(u_j^m + t) = \lambda(u_j^m + t)$$

where $t \perp u_j^m$. If we restrict the operator A to the subspace orthogonal to u_j^m, we get

$$(I - u_j^m(u_j^m)^T)(A - \lambda I)(I - u_j^m(u_j^m)^T)t = -(A - \theta_j^m I)u_j^m.$$

In the above expression, λ is replaced by its approximation θ_j^m since the exact λ is unknown in practical situations. Then we solve the following correction equation only approximately

$$(I - u_j^m(u_j^m)^T)(A - \theta_j^m I)(I - u_j^m(u_j^m)^T)t = -(A - \theta_j^m I)u_j^m.$$

and take $\tilde{t}^m$ for the expansion of the subspace. Notice that $r_j^m \perp \{v_1, \ldots, v_m\}$ and $r_j^m \perp u_j^m$, so the Jacobi-Davidson correction equation represents a consistent linear system. The linear system can be solved with a preconditioner like the generalized minimal residual(GMRES). The preconditioner is also restricted to the subspace orthogonal to u_j^m, which means that we will need to work with

$$\tilde{P} = (I - u_j^m(u_j^m)^T)P(I - u_j^m(u_j^m)^T),$$

and the linear system becomes $\tilde{P}^{-1}\tilde{A}v = z$, where

$$\tilde{A} = (I - u_j^m(u_j^m)^T)(A - \theta_j^m I)(I - u_j^m(u_j^m)^T).$$

Solving the above linear system only takes a handful of more simple operations, yet the acceleration of the algorithm is tremendous. For more details about Jacobi-Davidson method, see the literature [3, 42, 43, 48]. In the following, we state the Jacobi-Davidson algorithm.

Algorithm 6. Jacobi-Davidson Algorithm

```
Start with $t = v_0$, starting guess
For $m = 1, \ldots,$
  For $i = 1, \ldots, m - 1$
```

```
        $t = t - (v_i^T t)v_i$
    End
    $v_m = t/\|t\|_2, v_m^A = Av_m$
    For $i = 1, \ldots, m-1$
        $M_{i,m} = v_i^T v_m^A$
        $M_{m,i} = v_m^T v_i^A$
    End
    $M_{m,m} = v_m^T v_m^A$
    Compute the largest eigenpair $Ms = \theta s$ of the $m$ by $m$ matrix $M$, $(\|s\|_2 = 1)$
    $u = Vs$ with $V = [v_1, \ldots, v_m]$
    $u^A = V^A s$ with $V^A = [v_1^A, \ldots, v_m^A]$
    $r = u^A - \theta u$
    if $(\|r\|_2 \leq \epsilon), \tilde{\lambda} = \theta, \tilde{x} = u$, then stop the iteration.
    Solve $t$ approximately from $(I - uu^T)(A - \theta I)(I - uu^T)t = -r$ with $t \perp u$.
End
```

2.4 GMRES method

The generalized minimal residual method(GMRES), as an extension of minimal residual method(MINRES), is a projection method to solve nonsymmetric linear systems. It generates a sequence of orthogonal vectors, but in the absence of symmetry this can no longer be done with short recurrence; instead, all previously computed vectors in the orthogonal sequence have to be retained. The GMRES algorithm has the property that this residual norm can be computed without the iterate having been formed. Thus, the expensive action of forming the iterate can be postponed until the residual norm is deemed small enough. In exact arithmetic, like any orthogonalizing Krylov subspace method, the GMRES will converge in no more than n steps if no restarts are used, where n is the size of the matrix.

Let the Arnoldi process be

$$AV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T = V_{m+1} \bar{H}_m,$$

where $\bar{H} = \begin{pmatrix} H_m \\ h_{m+1,m} e_m^T \end{pmatrix}$. Any vector x in $x_0 + K_m$ can be written as

$$x = x_0 + V_m y$$

where y is an m-dimensional vector. Defining

$$J(y) = \|b - Ax\|_2 = \|b - A(x_0 + V_m y)\|_2$$

the residual. From the Arnoldi process, we have

$$b - Ax = b - A(x_0 + V_m y) = V_{m+1}(\beta e_1 - \bar{H}_m y)$$

where $\beta = \|r_0\|_2$. Since the column vectors of V_{m+1} are orthonormal, then

$$J(y) = \|b - A(x_0 + V_m y)\|_2 = \|\beta e_1 - \bar{H}_m y\|_2.$$

The GMRES approximation is the unique vector of $x_0 + K_m$ which minimizes $J(y) = \|b - Ax\|_2$. And this approximation can be obtained quite simply as $x_m = x_0 + V_m y_m$ where y_m minimizes the function $J(y) = \|\beta e_1 - \bar{H}_m y\|_2$, i.e.

$$x_m = x_0 + V_m y_m$$

where $y_m = \text{ argmin }_y \|\beta e_1 - \bar{H}_m y\|_2$. The minimizer y_m is inexpensive to compute since it requires the solution of an $(m+1) \times m$ least-squares problem where m is typically small. This gives the following algorithm.

Algorithm 7. GMRES Algorithm

```
Compute $r_0 = b - Ax_0$, $\beta := \|r_0\|_2$, and $v_1 := r_0/\beta$
Set $\bar{H}_m = \{h_{ij}\}_{1 \le i \le m+1} \in \mathbb{R}^{(m+1)\times m}$. $\bar{H}_m = 0$.
For j=1,2,..., m Do:
  Compute $w_j := Av_j$
  For i=1,..., j Do:
    $h_{ij} := (w_j, v_i)$
    $w_j := w_j - h_{ij}v_i$
  EndDo
  $h_{j+1,j} = \|w_j\|_2$. If $h_{j+1,j} = 0$ set $m := j$ and go to the last step
  $v_{j+1} = w_j/h_{j+1,j}$
EndDo
Compute $y_m$ the minimizer of $\|\beta e_1 - \bar{H}_m y\|_2$ and $x_m = x_0 + V_m y_m$.
```

If the algorithm breaks down, the only possibilities are in the Arnoldi loop, when $\hat{v}_{j+1} = 0$, i.e., when $h_{j+1,j} = 0$ at a given step j. In this situation, the algorithm stops because the next Arnoldi vector can not be generated. However, in this situation, the residual vector is zero, i.e., the algorithm will deliver the exact solution at this step. In fact the converse is also true: If the algorithm stops at step j with $b - Ax_j = 0$, then $h_{j+1,j} = 0$.

The GMRES algorithm becomes impractical when m is large because of the growth of memory and computational requirements as m increases. One can use restarting on the Arnoldi orthogonalization. A well known difficulty with the restarted GMRES algorithm is that it can stagnate when the matrix is not positive definite. The full GMRES algorithm is guaranteed to converge in at most n steps, but this would be impractical if there were many steps required for convergence. A preconditioner for the linear system can be used to reduce the number of steps, or a better preconditioner if one is already in use. For more information of GMRES, please refer to Saad [40]. In chapter 3, we will use the GMRES algorithm to solve the linear system equation of the inner iteration of the inexact subspace inner-outer iteration method.

2.5 Matrix functions

In this section we consider the decay of the elements away from the diagonal of certain function of bounded matrices. These results will be used in our error bounds of approximation of matrix exponentials in chapter 3. Several papers already established results on this topic [5, 7, 10, 11]. We will present results of Benzi & Golub [4].

Let P_k be the set of all polynomials with real coefficients and degree less than or equal to k. For a continuous function F on $[-1, 1]$, we consider approximating F by a polynomial $p \in P_k$. The best approximation error is defined as

$$E_k(F) = \inf\{||F - p||_\infty : p \in P_k\}$$

where

$$||F - p||_\infty = \max_{-1 \le x \le 1} |F(x) - p(x)|.$$

Bernstein [6] investigated the asymptotic behavior of the quantity $E_k(F)$ for a function F analytic on a domain which contains the interval $[-1, 1]$. His result states that this error decays to zero exponentially as $k \to \infty$, and shows how to estimate the decay rate.

If F is analytic on a simply connected region of the complex plane containing the interval $[-1, 1]$, there exist ellipses with foci in -1 and 1 such that F is analytic in their interiors. Let $\alpha > 1$ and $\beta > 0$ be the half axes of such an ellipse, $\alpha > \beta$. From the identity

$$\sqrt{\alpha^2 - \beta^2} = 1$$

we find that

$$\alpha - \beta = \frac{1}{\alpha + \beta}$$

and the ellipse is completely specified once the number

$$\chi = \alpha + \beta$$

is known, hence we may denote it by ϵ_χ. Furthermore, note that β is specified once α is, because $\beta = \sqrt{\alpha^2 - 1}$.

Theorem 2.5.1. *Let the function F be analytic in the interior of the ellipse $\epsilon_\chi, \chi > 1$, and continuous on ϵ_χ. In addition, suppose $F(z)$ is real for real z. Then*

$$E_k(F) \leq \frac{2M(\chi)}{\chi^k(\chi - 1)} \tag{2.1}$$

where

$$M(\chi) = \max_{z \in \epsilon_\chi} |F(z)|.$$

It is convenient to introduce now the concept of regularity ellipse of F, as in [31]. It is the ellipse $\epsilon_{\bar{\chi}}$ where

$$\bar{\chi} = \bar{\chi}(F) = \sup\{\chi : F \text{ is analytic in the interior of } \epsilon_\chi\}.$$

Evidently, it is important to study the behavior of the right-hand side of (2.1) as χ varies between 1 and $\bar{\chi}$. In particular, we see that the decay rate may become arbitrarily slow as $\chi \to 1$ (from the right). On the other hand, as χ increases, so does the rate of decay, as long as the quantity $M(\chi)$ remains bounded.

Letting

$$K_0 = \frac{2\chi M(\chi)}{\chi - 1}, \qquad q = \frac{1}{\chi} < 1,$$

we can rewrite the error bound (2.1) as

$$E_k(F) \leq K_0 q^{k+1}.$$

Let m be a nonnegative, even integer. A symmetric matrix $B = (b_{ij})$ is called $m-$banded if

$$b_{ij} = 0 \text{ when } |i-j| > \frac{m}{2}.$$

Consider matrix function $F(B)$, which is defined if $\sigma(B) \subseteq \epsilon_{\bar{\chi}}$. we have the following exponentially decaying bound for the entries of $F(B)$.

Theorem 2.5.2. *Let F be an analytic function in the interior of the ellipse $\varepsilon_\chi, \chi > 1$ and continuous on ε_χ. Let B be symmetric, m-banded, and such that $[-1,1]$ is the smallest interval containing $\sigma(B)$, the spectrum of B. Let $\rho = q^{\frac{2}{m}}, q = \dfrac{1}{\chi}$ and*

$$K = \max\{K_0, \|F(B)\|_2\}$$

with $K_0 = \frac{\chi M(\chi)}{\chi - 1}$ where $M(\chi) = \max_{z \in \varepsilon_\chi} |F(z)|$, Then we have

$$|(F(B))_{ij}| \leq K\rho^{|i-j|}.$$

Remark. As $|i-j|$ increases, matrix entries away from the diagonal exhibit exponentially fast decay.

Chapter 3 Inexact Inverse Subspace Iteration For Generalized Eigenvalue Problems

In this chapter, we are interested in computing a few eigenpairs of the generalized eigenvalue problem

$$Ax = \lambda Bx, \tag{3.1}$$

where A, B are nonsingular $n \times n$ matrices and $\lambda \in \mathbb{C}$ and $x \in \mathbb{C}^n$. The eigenvalues sought may be those in the extreme part of the spectrum or in the interior of spectrum near certain given point. These types of problems arise in many scientific and engineering applications. In such applications the matrices involved are often large and sparse.

Traditionally, shift-and-invert transformations are combined with the classic eigenproblem solvers in order to speed up the convergence. However, due to the inefficiency of the factorization of large and sparse matrices for solving a linear system at each iterative step, iterative methods (inner iteration) may be employed to solve the linear systems inexactly, which leads to inner-outer iterations. But the inner iteration produces only approximate solutions and is effectively equivalent to inexact application of matrix operators. This may affect the convergence behavior (or convergence speed) of the outer iteration as compared with the exact case. Related to this, a challenging problem in implementations lies in how to choose an appropriate stopping threshold for the iterative method (inner iteration) so that the convergence characteristic of the outer iteration can be preserved. This is a problem that has been discussed for several methods, such as inexact inverse iteration [26], inexact Krylov subspace method [20], the rational Arnoldi algorithm [27], inexact Rayleigh Quotient-Type methods [41] and the Jacobi-Davidson method [44]. These work demonstrate that the inner-outer iteration technique may be an effective way for implementing some of these methods for solving the large scale eigenvalue problem.

The classical inverse iteration has been shown to be particularly robust with respect to inexact application of the matrix operator (i.e. approximate solutions of inner linear systems). In [18], it was shown that solving inner linear systems to certain low accuracy is sufficient to recover the convergence characteristic of the outer iteration. This robust convergence characteristic makes the method attractive for the problems where inverting A or a shifted matrix $A - \sigma B$ is difficult. However, the inverse iteration can only handle one simple eigenpair at a time and has slow convergence if the eigenvalue sought is clustered. The subspace iteration (again applied with inverse or shift-and-invert) is a block generalization that can compute several eigenvalues. By simultaneously computing several eigenvalues together, it can handle multiple or clustered eigenvalues. Because of its simplicity, the classical subspace iteration is still widely used in certain applications such as structure engineering. However, its behavior under inexact matrix application is not known.

In this chapter, we develop a block generalization of the inexact inverse iteration method, i.e., the inexact inverse subspace iteration to simultaneously compute several eigenvalues of the generalized eigenvalue problem $Ax = \lambda Bx$. We present a theoretical analysis demonstrating how the accuracy of the solutions in the inner iterations affects the outer iteration in the inexact case. Inexact inverse subspace iteration has also been considered in [32] which is based on different approach using so called tuned preconditioning to deal with inexact solution. The analysis is independent of the choice of the inner iterative solver.

The outline of the chapter is as follows. In Section 3.1, we set the notation and introduce the inexact inverse subspace iteration algorithm. Section 3.2 analyzes convergence of the space to the spectral space sought. In Section 3.3, we further discuss the convergence of the basis vectors of the subspace and their implication with respect to convergence of a residual. Finally in Section 3.4 we present numerical tests to illustrate the theoretical results and provide some comparison to the Jacobi-Davidson algorithm, one of the most effective methods.

3.1 Inexact Inverse Subspace Iteration

We consider solving the generalized eigenvalue problem

$$Ax = \lambda Bx.$$

with the following standard subspace iterations. (A is typically replaced by a shifted matrix $A - \sigma B$ in the shift-and-invert transformation when the eigenvalues near σ are sought.)

```
**A**lgorithm 1. Inverse Subspace Iteration:
Given orthonormal $X_0 \in \mathbb{C}^{n\times p}$;
For $k = 0, 1, 2 \ldots$, until convergence
   $Y_{k+1} = A^{-1}BX_k$;
   $Y_{k+1} = X_{k+1}R_{k+1}$. (QR-factorization)
End
```

Remark 1. Convergence of Algorithm 1 can be tested using the residual $\|Y_{k+1} - X_k(X_k^H Y_{k+1})\|$. For example, if $\|Y_{k+1} - X_k(X_k^H Y_{k+1})\| \leq \epsilon$, where ϵ is the threshold, we stop the iteration.

The inverse subspace iteration above computes eigenvalues of smallest magnitude. When the eigenvalues sought are not well separated, a shift-and-invert transformation can be used as a preconditioning technique for the spectral enhancement of these iterative methods. On the other hand, when the eigenvalues in the interior of the spectrum near some point σ are sought, one can again use the shift-and-invert transformation. But the use of the inverse of the matrix A (or shifted matrix $A - \sigma B$) limits problems to those of moderate size because of the factorization required. For large scale problems, factorization of the matrix may be either impractical or inefficient. Sometimes A is not even explicitly available in some applications. In these cases, an iterative method can be used to solve the linear systems $AY_{k+1} = BX_k$, called inner iterations, while the subspace iteration itself is called outer iteration.

At each step of the outer iteration, when solving for Y_{k+1} in $AY_{k+1} = BX_k$, Y_k can be used as an initial approximation. Then we solve

$$AD_k = BX_k - AY_k \tag{3.2}$$

approximately, i.e. find D_k such that

$$E_k := (BX_k - AY_k) - AD_k$$

is smaller than some threshold, and use D_k to obtain

$$Y_{k+1} = Y_k + D_k.$$

The main purpose of this part of the work is to analyze the convergence characteristic of subspace iteration under inexact solves.

Here, we consider using a threshold ϵ_k, i.e., we shall solve (3.2) such that

$$\|E_k\|_2 < \epsilon_k. \tag{3.3}$$

Obviously, the amount of work required to solve (3.2) is proportional to $\dfrac{\|BX_k - AY_k\|}{\epsilon_k}$. Our analysis later leads to the use of linearly decreasing ϵ_k, i.e., $\epsilon_k = ar^k$ for some positive a and r. However, as we shall see, even though ϵ_k is decreasing, the amount of work does not increase as $\|BX_k - AY_k\|$ will be decreasing as well. We state the inexact inverse subspace iteration as follows.

Algorithm 2. Inexact Inverse Subspace Iteration:

```
Given orthonormal $X_0 \in \mathbb{C}^{n\times p}$; set $Y_0 = 0$
For $k = 0, 1, \ldots$ until convergence
    $Z_k = BX_k - AY_k$;
    Solve $AD_k = Z_k$ such that $E_k = AD_k - Z_k$ satisfies (3.3);
    $Y_{k+1} = Y_k + D_k$;
    $Y_{k+1} = \overline{X}_{k+1}\overline{R}_{k+1}$; (QR factorization)
    For $j = 1, \ldots, p$
        [ ymax,imax ] = max( abs $(\overline{X}_{k+1}(:, j)))$;
        $X_{k+1}(:, j) =$ sign $(\overline{X}_{k+1}($ imax $, j))) * \overline{X}_{k+1}(:, j)$;
        $R_{k+1}(j, :) =$ sign $(\overline{X}_{k+1}($ imax $, j)) * \overline{R}_{k+1}(j, :)$.
    End
End
```

Remark. We have used MATLAB notation in the above algorithm. Namely, the max(v) function finds the maximum absolute value of v and its index. Then the construction of X_{k+1} from $\overline{X}_{k+1}$ in the algorithm is to scale the columns of X_{k+1} so that its maximum entry in absolute value is positive. This will be critical to ensure convergence of the column vectors in X_{k+1} and hence $\|BX_k - AY_k\|$. Without such scaling, columns of X_k converges only in direction.

Remark. Throughout, we shall assume that $\epsilon_k \leq \|B^{-1}\|_2^{-1}$, which will ensure that Y_k constructed has full column rank; see Lemma 3.2.2 below. In this way, QR factorization produces $n \times p$ $\bar{X}_{k+1}$ and hence X_{k+1} with orthonormal columns.

3.2 Convergence Analysis

We now discuss the convergence properties of the inexact inverse subspace algorithm. Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be the eigenvalues of $B^{-1}A$ ordered such that

$$0 < |\lambda_1| \leq \ldots \leq |\lambda_p| < |\lambda_{p+1}| \leq \ldots \leq |\lambda_n|$$

and $v_1, \ldots, v_n$ be the corresponding eigenvectors. Suppose that we are interested in computing the p smallest eigenpairs. Throughout this work, we assume that $B^{-1}A$ is diagonalizable.

Let $V = [v_1, \ldots, v_n], U = (BV)^{-H}$, then

$$U^H A = \Lambda U^H B, \quad AV = BV\Lambda,$$

where

$$\Lambda = \begin{pmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{pmatrix}, \quad \Lambda_1 = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_p \end{pmatrix}, \quad \text{and} \quad \Lambda_2 = \begin{pmatrix} \lambda_{p+1} & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}.$$

Let $U = (U_1, U_2), V = (V_1, V_2)$, where $U_1 \in \mathbb{C}^{n\times p}$, $U_2 \in \mathbb{C}^{n\times(n-p)}$, $V_1 \in \mathbb{C}^{n\times p}$, $V_2 \in \mathbb{C}^{n\times(n-p)}$, then

$$U_i^H A = \Lambda_i U_i^H B, \qquad AV_i = BV_i\Lambda_i.$$

Consider Algorithm 2. Define $X_k^{(i)} = U_i^H B X_k$. Since $U_i^H B V_j = \delta_{ij} I, U_i^H A V_j = \delta_{ij}\Lambda_i$, where δ_{ij} is the Kronecker symbol, then

$$X_k = \sum_{i=1}^{2} V_i X_k^{(i)}.$$

If $X_k^{(1)}$ is invertible, we define

$$t_k := ||X_k^{(2)}(X_k^{(1)})^{-1}||_2$$

Clearly, t_k is a measure of the approximation of the column space of X_k to the column space of V_1. Indeed, the following proposition relates t_k to other measures of subspace approximation.

Proposition 3.2.1. *Assume that $X_k^{(1)}$ is invertible and t_k is defined as above. Then*

$$\frac{t_k}{||V^{-1}||_2} \leq ||X_k(X_k^{(1)})^{-1} - V_1||_2 \leq ||V||_2 t_k \tag{3.4}$$

and

$$\sin \angle(\mathcal{X}_k, \mathcal{V}_1) \leq \|V\|_2 \|R^{-1}\|_2 t_k$$

where $\angle(\mathcal{X}_k, \mathcal{V}_1)$ is the largest canonical angle between $\mathcal{X}_k = \mathcal{R}(X_k)$ and $\mathcal{V}_1 = \mathcal{R}(V_1)$ and $V_1 = WR$ is the QR factorization of V_1.

Proof. From $X_k = V_1 X_k^{(1)} + V_2 X_k^{(2)}$, we have

$$X_k(X_k^{(1)})^{-1} = V_1 + V_2 X_k^{(2)} (X_k^{(1)})^{-1}.$$

Then

$$||X_k(X_k^{(1)})^{-1} - V_1||_2 = ||V_2 X_k^{(2)}(X_k^{(1)})^{-1}||_2 \leq ||V_2||_2 ||X_k^{(2)}(X_k^{(1)})^{-1}||_2 \leq ||V||_2 t_k$$

and

$$||V_2 X_k^{(2)}(X_k^{(1)})^{-1}||_2 = ||V \begin{bmatrix} 0 \\ X_k^{(2)}(X_k^{(1)})^{-1} \end{bmatrix}||_2 \geq \frac{||X_k^{(2)}(X_k^{(1)})^{-1}||_2}{||V^{-1}||_2} = \frac{t_k}{||V^{-1}||_2}.$$

(3.4) is proved.

Let $X_k^{\perp}$ be such that $(X_k, X_k^{\perp})$ is an orthogonal matrix. Then the sine of the largest canonical angle between $\mathcal{X}_k = \mathcal{R}(X_k)$ and $\mathcal{V}_1 = \mathcal{R}(V_1)$ is (see [21] for the definition)

$$\sin \angle(\mathcal{X}_k, \mathcal{V}_1) = \|(X_k^{\perp})^H W\|_2,$$

Then we have

$$\begin{aligned} \sin \angle(\mathcal{X}_k, \mathcal{V}_1) &= \|(X_k^{\perp})^H (W - X_k (X_k^{(1)})^{-1} R^{-1})\|_2 \\ &= \|(X_k^{\perp})^H (V_2 X_k^{(2)} (X_k^{(1)})^{-1}) R^{-1}\|_2 \\ &\leq \|V\|_2 \|R^{-1}\|_2 t_k \end{aligned}$$

□

Next we discuss the convergence of t_k and the condition of ϵ_k for t_k to be bounded.

Lemma 3.2.2. *For Algorithm 2, if $\|E_k\|_2 < \|B^{-1}\|_2^{-1}$, then Y_{k+1} has full column rank.*

Proof. From the algorithm, we know that $AY_{k+1} = (BX_k + E_k)$. Therefore

$$X_k^H B^{-1} AY_{k+1} = I + X_k^H B^{-1} E_k.$$

Since

$$\|X_k^H B^{-1} E_k\|_2 \le \|B^{-1}\|_2 \|E_k\|_2 < 1,$$

$X_k^H B^{-1} AY_{k+1}$ is invertible. Thus Y_{k+1} has full rank. □

From now on, we shall assume that $\epsilon_k \le \|B^{-1}\|_2^{-1}$. Then, all Y_k will have full column rank.

Lemma 3.2.3. *For Algorithm 2, if $X_k^{(1)}$ is invertible, then*

$$\|(X_k^{(1)})^{-1}\|_2 \le \|V\|_2(1 + t_k).$$

Proof. Recognizing that

$$X_k = V_1 X_k^{(1)} + V_2 X_k^{(2)} \text{ and } \|(X_k^{(1)})^{-1}\|_2 = \|X_k(X_k^{(1)})^{-1}\|_2,$$

the conclusion follows directly from the following

$$\|(X_k^{(1)})^{-1}\|_2 = \|V_1 + V_2 X_k^{(2)}(X_k^{(1)})^{-1}\|_2 \le \|V_1\|_2 + t_k\|V_2\|_2 \le (1 + t_k)\|V\|_2.$$

□

Lemma 3.2.4. *Let $\rho = \dfrac{|\lambda_p|}{|\lambda_{p+1}|} < 1$ and $X_k^{(1)}, X_{k+1}^{(1)}$ be non-singular. If $\|V\|_2\|U\|_2(1+t_k)\epsilon_k < 1$, then*

$$t_{k+1} \le \rho t_k + \frac{\rho\|V\|_2\|U\|_2(1 + t_k)^2\epsilon_k}{1 - \|V\|_2\|U\|_2(1 + t_k)\epsilon_k}$$

Proof. From the algorithm we know that $AY_{k+1} = BX_k + E_k$ and $Y_{k+1} = X_{k+1}R_{k+1}$. Since Y_{k+1} has full column rank, R_{k+1} is invertible. Then

$$AX_{k+1} = BX_k R_{k+1}^{-1} + E_k R_{k+1}^{-1}.$$

Multiplying U_i^H on the equation above, we have

$$U_i^H AX_{k+1} = U_i^H BX_k R_{k+1}^{-1} + U_i^H E_k R_{k+1}^{-1}.$$

Utilizing the definition of $X_k^{(i)}$ and the relation $U_i^H A = \Lambda_i U_i^H B$, we have the following two equations:

$$\Lambda_i U_i^H BX_{k+1} = X_k^{(i)} R_{k+1}^{-1} + U_i^H E_k R_{k+1}^{-1},$$

$$X_{k+1}^{(i)} = \Lambda_i^{-1} X_k^{(i)} R_{k+1}^{-1} + \Lambda_i^{-1}\Delta_k^{(i)} \tag{3.5}$$

where $\Delta_k^{(i)} = U_i^H E_k R_{k+1}^{-1}$. We therefore have

$$\begin{aligned}
&X_{k+1}^{(2)}(X_{k+1}^{(1)})^{-1} \\
&= (\Lambda_2^{-1}X_k^{(2)}R_{k+1}^{-1} + \Lambda_2^{-1}\Delta_k^{(2)})(X_{k+1}^{(1)})^{-1} \\
&= \Lambda_2^{-1}X_k^{(2)}R_{k+1}^{-1}(X_{k+1}^{(1)})^{-1} + \Lambda_2^{-1}\Delta_k^{(2)}(X_{k+1}^{(1)})^{-1} \\
&= \Lambda_2^{-1}X_k^{(2)}((X_k^{(1)})^{-1}\Lambda_1)(\Lambda_1^{-1}X_k^{(1)}R_{k+1}^{-1})(X_{k+1}^{(1)})^{-1} + \Lambda_2^{-1}\Delta_k^{(2)}(X_{k+1}^{(1)})^{-1} \\
&= \Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Lambda_1(X_{k+1}^{(1)} - \Lambda_1^{-1}\Delta_k^{(1)})(X_{k+1}^{(1)})^{-1} + \Lambda_2^{-1}\Delta_k^{(2)}(X_{k+1}^{(1)})^{-1} \\
&= \Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Lambda_1 - \Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Delta_k^{(1)}(X_{k+1}^{(1)})^{-1} + \Lambda_2^{-1}\Delta_k^{(2)}(X_{k+1}^{(1)})^{-1} \\
&= \Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Lambda_1 - \left(\Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Delta_k^{(1)} - \Lambda_2^{-1}\Delta_k^{(2)}\right)(X_{k+1}^{(1)})^{-1} \\
&= \Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Lambda_1 \\
&\quad - \left(\Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Delta_k^{(1)} - \Lambda_2^{-1}\Delta_k^{(2)}\right)(\Lambda_1^{-1}X_k^{(1)}R_{k+1}^{-1} + \Lambda_1^{-1}\Delta_k^{(1)})^{-1}
\end{aligned}$$

Since $\Delta_k^{(i)} = U_i^H E_k R_{k+1}^{-1}$ and

$$\begin{aligned}
(\Lambda_1^{-1}X_k^{(1)}R_{k+1}^{-1} + \Lambda_1^{-1}\Delta_k^{(1)})^{-1} &= \left((I + \Delta_k^{(1)}R_{k+1}(X_k^{(1)})^{-1})X_k^{(1)}R_{k+1}^{-1}\right)^{-1}\Lambda_1 \\
&= R_{k+1}(X_k^{(1)})^{-1}(I + \Delta_k^{(1)}R_{k+1}(X_k^{(1)})^{-1})^{-1}\Lambda_1
\end{aligned}$$

Then we further simplify the expression $X_{k+1}^{(2)}(X_{k+1}^{(1)})^{-1}$ to

$$\begin{aligned}
X_{k+1}^{(2)}(X_{k+1}^{(1)})^{-1} &= \Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Lambda_1 \\
&\quad -\Big(\Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}U_1^H E_k R_{k+1}^{-1} - \Lambda_2^{-1}U_2^H E_k R_{k+1}^{-1}\Big) \\
&\quad \times R_{k+1}(X_k^{(1)})^{-1}(I+\Delta_k^{(1)}R_{k+1}(X_k^{(1)})^{-1})^{-1}\Lambda_1 \\
&= \Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Lambda_1 \\
&\quad -\Big(\Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}U_1^H - \Lambda_2^{-1}U_2^H\Big) \\
&\quad \times E_k(X_k^{(1)})^{-1}(I+U_1^H E_k(X_k^{(1)})^{-1})^{-1}\Lambda_1
\end{aligned}$$

Taking 2-norm of the above equation at both sides and using the condition

$$\|V\|_2\|U\|_2(1+t_k)\epsilon_k < 1,$$

we get the upper bound of t_{k+1}.

$$\begin{aligned}
t_{k+1} &= ||X_{k+1}^{(2)}(X_{k+1}^{(1)})^{-1}||_2 \\
&\le \|\Lambda_2^{-1}X_k^{(2)}(X_k^{(1)})^{-1}\Lambda_1\|_2 + \Big(\|\Lambda_2^{-1}\|_2\|X_k^{(2)}(X_k^{(1)})^{-1}\|_2\|U_1^H\|_2 + \|\Lambda_2^{-1}\|_2\|U_2^H\|_2\Big) \\
&\quad \|E_k\|_2\|(X_k^{(1)})^{-1}\|_2\|(I+U_1^H E_k(X_k^{(1)})^{-1})^{-1}\|_2\|\Lambda_1\|_2 \\
&\le \rho t_k + \left(\|\Lambda_2^{-1}\|_2 t_k + \|\Lambda_2^{-1}\|_2\right)\|U\|_2\|E_k\|_2 \\
&\quad \times\|(X_k^{(1)})^{-1}\|_2\|\Lambda_1\|_2\|(I+U_1^H E_k(X_k^{(1)})^{-1}\|_2 \\
&\le \rho t_k + (t_k+1)\|\Lambda_2^{-1}\|_2\frac{\|U\|_2\|E_k\|_2\|(X_k^{(1)})^{-1}\|_2\|\Lambda_1\|_2}{1-\|U\|_2\|E_k\|_2\|(X_k^{(1)})^{-1}\|_2}
\end{aligned}$$

Also by Lemma 3.2.3, we know that $\|(X_k^{(1)})^{-1}\|_2 \le \|V\|_2(1+t_k)$. From this, we finally derive the bound as follows.

$$t_{k+1} \le \rho t_k + \frac{\rho\|V\|_2\|U\|_2(1+t_k)^2\epsilon_k}{1-\|V\|_2\|U\|_2(1+t_k)\epsilon_k}.$$

□

Lemma 3.2.5. *Assume that X_0 is such that $X_0^{(1)}$ is invertible. If*

$$\epsilon_k \le \epsilon := \frac{(1-\rho)t_0}{\|V\|_2\|U\|_2(1+t_0)(\rho+t_0)},$$

for all k, then $t_k \le t_0$.

Proof. We prove $t_k \le t_0$ by induction. Supposing $X_k^{(1)}$ is nonsingular and $t_k \le t_0$ is true for some k, we show that $X_{k+1}^{(1)}$ is nonsingular and $t_{k+1} \le t_0$. First note that from $\epsilon_k \le \epsilon$, we have

$$\|V\|_2\|U\|_2(1+t_k)\epsilon_k \le \|V\|_2\|U\|_2(1+t_0)\epsilon = \frac{(1-\rho)t_0}{\rho+t_0} < 1.$$

We discuss in two cases.

Case I: If $X_{k+1}^{(1)}$ is nonsingular, then by Lemma 3.2.4, we have

$$\begin{aligned} t_{k+1} &\le \rho t_k + \frac{\rho\|V\|_2\|U\|_2(1+t_k)^2\epsilon_k}{1-\|V\|_2\|U\|_2(1+t_k)\epsilon_k} \\ &\le \rho t_0 + \frac{\rho\|V\|_2\|U\|_2(1+t_0)^2\epsilon}{1-\|V\|_2\|U\|_2(1+t_0)\epsilon} \\ &= \rho t_0 + \frac{\rho(1+t_0)\dfrac{(1-\rho)t_0}{\rho+t_0}}{1-\dfrac{(1-\rho)t_0}{\rho+t_0}} = t_0 \end{aligned}$$

Case II: If $X_{k+1}^{(1)}$ is singular, then let

$$\tilde{Y}_{k+1} = Y_{k+1} + \delta V_1 R_{k+1} + \mu V_2 \begin{bmatrix} I \\ 0 \end{bmatrix} R_{k+1}$$

where $Y_{k+1} = X_{k+1}R_{k+1}$ and $\delta, \mu > 0$ are two parameters. Then we have

$$A\tilde{Y}_{k+1} = BX_k + E_k + \delta AV_1R_{k+1} + \mu AV_2 \begin{bmatrix} I \\ 0 \end{bmatrix} R_{k+1} = BX_k + \tilde{E}_k$$

where $\tilde{E}_k = E_k + \delta AV_1R_{k+1} + \mu AV_2 \begin{bmatrix} I \\ 0 \end{bmatrix} R_{k+1}$. Since $\|E_k\|_2 < \epsilon_k$, we have $\|\tilde{E}_k\|_2 < \epsilon_k$ for sufficiently small δ and μ. Let $\tilde{Y}_{k+1} = \tilde{X}_{k+1}\tilde{R}_{k+1}$ be the QR factorization and let $\tilde{X}_{k+1} = V_1\tilde{X}_{k+1}^{(1)} + V_2\tilde{X}_{k+1}^{(2)}$. Then $\tilde{X}_{k+1}$ satisfies the same condition that X_{k+1} does and the bound on t_{k+1} applies to $\tilde{t}_{k+1} := \|\tilde{X}_{k+1}^{(2)}(\tilde{X}_{k+1}^{(1)})^{-1}\|_2$

as well. It follows from

$$\begin{aligned}\tilde{Y}_{k+1} &= Y_{k+1} + \delta V_1 R_{k+1} + \mu V_2 \begin{bmatrix} I \\ 0 \end{bmatrix} R_{k+1} \\ &= [V_1(X_{k+1}^{(1)} + \delta I) + V_2(X_{k+1}^{(2)} + \mu \begin{bmatrix} I \\ 0 \end{bmatrix})]R_{k+1},\end{aligned}$$

that

$$\tilde{X}_{k+1}^{(1)} = (X_{k+1}^{(1)} + \delta I)R_{k+1}\tilde{R}_{k+1}^{-1},$$

and $\tilde{X}_{k+1}^{(2)} = (X_{k+1}^{(2)} + \mu \begin{bmatrix} I \\ 0 \end{bmatrix})R_{k+1}\tilde{R}_{k+1}^{-1}$. So $\tilde{X}_{k+1}^{(1)}$ is nonsingular for sufficiently small $\delta > 0$. Then, by case I, we have

$$\tilde{t}_{k+1} = \|\tilde{X}_{k+1}^{(2)} \left(\tilde{X}_{k+1}^{(1)}\right)^{-1} \|_2 \le t_0$$

for all sufficiently small $\delta > 0$ and $\mu \ge 0$. However,

$$\tilde{X}_{k+1}^{(2)} \left(\tilde{X}_{k+1}^{(1)}\right)^{-1} = (X_{k+1}^{(2)} + \mu \begin{bmatrix} I \\ 0 \end{bmatrix})(X_{k+1}^{(1)} + \delta I)^{-1}$$

is unbounded as $\delta \to 0$, because if $X_{k+1}^{(2)}(X_{k+1}^{(1)} + \delta I)^{-1}$ is unbounded, then $\tilde{t}_{k+1}$ is unbounded by setting $\mu = 0$; and if $X_{k+1}^{(2)}(X_{k+1}^{(1)} + \delta I)^{-1}$ is bounded, then $\tilde{t}_{k+1}$ is unbounded by setting $\mu > 0$. We have obtained a contradiction. Therefore $X_{k+1}^{(1)}$ is nonsingular and hence $t_{k+1} \le t_0$. The proof is complete.

□

We now prove our main result on convergence of t_k. We are interested in the case that ϵ_k is a linearly decreasing sequence.

Theorem 3.2.6. *Assume that X_0 is such that $X_0^{(1)}$ is invertible. Let $\epsilon_k = a\gamma^k$ with $\gamma < 1$ and*

$$a \le \frac{(1-\rho)t_0}{\|V\|_2\|U\|_2(1+t_0)(\rho+t_0)}.$$

Then we have

$$t_k \le \begin{cases} \rho^k t_0 + aC\dfrac{\gamma^k - \rho^k}{\gamma - \rho}, & \text{if } \gamma \ne \rho, \\ \rho^k t_0 + aCk\rho^{k-1} & \text{if } \gamma = \rho, \end{cases}$$

where $C = \|V\|_2 \|U\|_2 (1 + t_0)(\rho + t_0)$.

Proof. Since

$$\epsilon_k \le \frac{(1-\rho)t_0}{\|V\|_2\|U\|_2(1+t_0)(\rho+t_0)},$$

we have $t_k \le t_0$ by Lemma 3.2.3. Then,

$$\|V\|_2\|U\|_2(1+t_k)\epsilon_k \le \frac{(1-\rho)t_0}{\rho+t_0} < 1.$$

It follows from Lemma 3.2.4 that $t_{k+1} \le \rho t_k + C_k \epsilon_k$ where

$$\begin{aligned} C_k &:= \frac{\rho\|V\|_2\|U\|_2(1+t_k)^2}{1-\|V\|_2\|U\|_2(1+t_k)\epsilon_k} \\ &\le \frac{\rho\|V\|_2\|U\|_2(1+t_0)^2}{1-\dfrac{(1-\rho)t_0}{\rho+t_0}} \\ &\le \|V\|_2\|U\|_2(1+t_0)(\rho+t_0) = C. \end{aligned}$$

Therefore, $t_{k+1} \le \rho t_k + aC\gamma^k$. Solving this inequality, the theorem is proved.

$\square$

The conclusion of the above theorem is that the subspace spanned by X_k, $\mathcal{R}(X_k)$, converges to the spectral subspace $\mathcal{R}(V_1)$ linearly at the rate of $\max\{\rho, \gamma\}$. The condition on a is to ensure convergence and is clearly not a necessary condition.

An interesting fact is that there is no gain in convergence rate if we choose $\gamma < \rho$, so we shall focus on the case $\gamma > \rho$. The following corollary gives a more precise bound for the constant C and hence for t_k at the convergence stage.

Corollary 3.2.7. *Let* $1 > \gamma > \rho$ *and* $\epsilon_k = a\gamma^k$. *Assume that* a *is chosen such that* $t_k \to 0$. *Then for sufficiently large* k_0,

$$C_{k_0} = \frac{\rho\|V\|_2\|U\|_2(1+t_{k_0})^2}{1-\|V\|_2\|U\|_2(1+t_{k_0})\epsilon_{k_0}} \sim \rho\|V\|_2\|U\|_2$$

and $\limsup \dfrac{t_k}{a\gamma^{k-1}} \leq \rho(1-\rho/\gamma)^{-1}\|V\|_2\|U\|_2$. *Furthermore, for* $k \geq k_0$,

$$t_k \leq \rho^{k-k_0}t_{k_0} + a\gamma^{k_0}C_{k_0}\frac{\gamma^{k-k_0}-\rho^{k-k_0}}{\gamma-\rho} \sim a\gamma^{k-1}\min\{k-k_0,(1-\rho/\gamma)^{-1}\}\rho\|V\|_2\|U\|_2$$

Proof. It is obvious that, $\lim_{k_0\to\infty} C_{k_0} = \rho\|V\|_2\|U\|_2$. Apply the main theorem to t_k starting from $k = k_0$, we obtain

$$\begin{aligned} t_k &\leq \rho^{k-k_0}t_{k_0} + \frac{\gamma^{k-k_0}-\rho^{k-k_0}}{\gamma-\rho}a\gamma^{k_0}C_{k_0} \\ &\leq \rho^{k-k_0}t_{k_0} + \gamma^{k-k_0-1}\min\{k-k_0,(1-\rho/\gamma)^{-1}\}aC_{k_0}\gamma^{k_0} \\ &\sim a\gamma^{k-1}\min\{k-k_0,(1-\rho/\gamma)^{-1}\}\rho\|V\|_2\|U\|_2 \end{aligned}$$

Taking $k \to \infty$ first and then $k_0 \to \infty$ in the first inequality, we obtain the bound for $\limsup \dfrac{t_k}{a\gamma^{k-1}}$.

□

3.3 Convergence of Basis Vectors and Asymptotic Analysis of Computational Work

In this section, we further analyze convergence of columns of X_k, or the basis of the subspace generated. In general, the subspace iteration does not necessarily lead to convergence of the columns of X_k. However, with the scaling we have introduced, which fixes the maximum entry of each column to be positive, the columns do converge. In the context of inexact version, this is very important because it ensures the convergence of residual $AY_k - BX_k$ to 0. Indeed, we shall show that $AY_k - BX_k$ converges to 0 at the same rate as ϵ_k, so that

$$\frac{\epsilon_k}{\|AY_k - BX_k\|_2}$$

stays near constant asymptotically. Thus the number of iteration (or the amount of work) required to solve the inner system (3.2) is nearly constant.

Below, we will use MATLAB-like notation $X_{(i:j,k:l)}$ to denote the submatrix of X, consisting of the intersections of rows i to j and columns k to l, and when $i : j$ is replaced by :, it means all rows, similarly for columns.

First we observe that if we apply Algorithm 2 with the initial block vector $X_{0_{(:,1:i)}}$ (i.e. the first i columns of X_0) with $i < p$, it is easy to check that $X_{k_{(:,1:i)}}$ will be a sequence of the block vectors generated. This is a property known as simultaneous iterations. Let

$$X_{k_{(:,1:i)}} = V_{(:,1:i)} X_{k,1}^{(i)} + V_{(:,i+1:n)} X_{k,2}^{(i)}. \tag{3.6}$$

Suppose that the QR factorization of V_1 is

$$V_1 = WR$$

where we assume that $W = [w_{ij}]$ satisfies that the largest element of each column is unique and positive. Let j_i be the index of the largest (in absolute value) element of the i^{th} column of W. Define

$$gap_i = \min_{j \neq j_i} \left(w_{j_i i} - |w_{ji}| \right), \quad 1 \leq i \leq p$$

and

$$t_k^{(i)} = \| X_{k,2}^{(i)} (X_{k,1}^{(i)})^{-1} \|_2.$$

Using Theorem 3.2.6, under appropriate assumptions there, $t_k^{(i)}$ converges to 0. Therefore $\mathcal{R}(X_{k_{(:,1:i)}})$ converges to $\mathcal{R}(V_{(:,1:i)})$. We shall further prove convergence of each column of X_k.

Lemma 3.3.1. *Let $x = v + e$, where x, v, e are vectors and v has a unique largest (in absolute value) element. Suppose the largest element of $v = (v_1, v_2, \ldots, v_n)^H$ is v_k and $v_k > 0$. Define*

$$gap = \min_{j \neq k} \{ |v_k| - |v_j| \}.$$

If $\|e\|_\infty \leq \frac{1}{2}$ *gap , then we have*

$$|\max(x) - \max(v)| \leq \|e\|_\infty \tag{3.7}$$

and

$$\left\|\frac{x}{\max(x)} - \frac{v}{\max(v)}\right\|_\infty \leq \frac{2\|e\|_\infty}{\|v\|_\infty}. \tag{3.8}$$

Proof. Since

$$e_j - e_k \leq 2\|e\|_\infty \leq v_k - |v_j| \leq v_k - v_j$$

and

$$-e_j - e_k \leq 2\|e\|_\infty \leq v_k - |v_j| \leq v_k + v_j,$$

then we have

$$|v_j + e_j| \leq v_k + e_k, \text{ for any } j.$$

So, $\max(v+e) = v_k + e_k$, i.e. $\max(x) = \max(v) + e_k$, therefore we have proved inequality (3.7), i.e.,

$$|\max(x) - \max(v)| = |e_k| \leq \|e\|_\infty.$$

Next, we have

$$\frac{x}{\max(x)} - \frac{v}{\max(v)} = (\max(v) - \max(x))\frac{x}{\max(v)\max(x)} + \frac{e}{\max(v)}.$$

Taking the infinity-norm and using (3.7), we have

$$\begin{aligned}\left\|\frac{x}{\max(x)} - \frac{v}{\max(v)}\right\|_\infty &\leq \frac{|(\max(v) - \max(x))|}{|\max(v)|}\left\|\frac{x}{\max(x)}\right\|_\infty + \left\|\frac{e}{\max(v)}\right\|_\infty \\ &\leq \frac{\|e\|_\infty}{|\max(v)|} + \frac{\|e\|_\infty}{|\max(v)|} \\ &= \frac{2\|e\|_\infty}{\|v\|_\infty}\end{aligned}$$

□

Lemma 3.3.2. *Given k and $1 \leq j \leq p$, let R, gap_i and $X_{k,1}^{(j)}$ be defined as in (3.6). Let $(X_{k,1}^{(j)})^{-1}(R_{(1:j,1:j)})^{-1} = [\alpha_{lm}^{(j)}]$, where $\alpha_{lm}^{(j)}$ is dependent on the iteration number k. Assume that*

$$\max(t_k^{(1)}, t_k^{(2)}, \ldots, t_k^{(j)}) < \frac{\min\{\frac{1}{2}gap_j, 1\}}{\Delta_j}, \quad 1 \leq j \leq p \tag{3.9}$$

where

$$\Delta_1 = \|V_{(:,2:n)}\|_2 |(R_{(1,1)})^{-1}|, \tag{3.10}$$

and

$$\Delta_j = j\|V_{(:,j+1:n)}\|_2 \|(R_{(1:j,1:j)})^{-1}\|_2 + \sum_{1\leq i\leq j-1} \frac{\Delta_i}{1-\min\{\frac{1}{2}gap_i, 1\}}, \quad 2 \leq j \leq p. \tag{3.11}$$

then we have

$$\|\alpha_{jj}^{(j)} X_{k_{(:,j)}} - W_{(:,j)}\|_2 \leq \Delta_j \max(t_k^{(1)}, t_k^{(2)}, \ldots, t_k^{(j)}), \quad 1 \leq j \leq p.$$

Proof. We will prove this statement by induction. For $j = 1$, we decompose

$$X_{k_{(:,1)}}(X_{k,1}^{(1)})^{-1}(R_{(1,1)})^{-1} = W_{(:,1)} + V_{(:,2:n)} X_{k,2}^{(1)} (X_{k,1}^{(1)})^{-1} R_{(1,1)}^{-1}. \tag{3.12}$$

So we have

$$\|\alpha_{11}^{(1)} X_{k_{(:,1)}} - W_{(:,1)}\|_2 \quad \leq \quad \|V_{(:,2:n)}\|_2 |(R_{(1,1)})^{-1}| t_k^{(1)} = \Delta_1 t_k^{(1)}.$$

Suppose for some $m \leq p-1$ that

$$\|\alpha_{jj}^{(j)} X_{k_{(:,j)}} - W_{(:,j)}\|_2 \leq \Delta_j \max(t_k^{(1)}, t_k^{(2)}, \ldots, t_k^{(j)}), \quad 1 \leq j \leq m-1$$

is true, we prove that it is also true for $j = m$. In this case

$$X_{k_{(:,1:m)}}(X_{k,1}^{(m)})^{-1} = W_{(:,1:m)} R_{(1:m,1:m)} + V_{(:,m+1:n)} X_{k,2}^{(m)} (X_{k,1}^{(m)})^{-1}.$$

or

$$X_{k_{(:,1:m)}}(X_{k,1}^{(m)})^{-1} R_{(1:m,1:m)}^{-1} = W_{(:,1:m)} + V_{(:,m+1:n)} X_{k,2}^{(m)} (X_{k,1}^{(m)})^{-1} R_{(1:m,1:m)}^{-1}.$$

Therefore

$$\alpha_{mm}^{(m)} X_{k_{(:,m)}} = W_{(:,m)} + V_{(:,m+1:n)} X_{k,2}^{(m)} (X_{k,1}^{(m)})^{-1} (R_{(1:m,1:m)})^{-1} e_m - \sum_{1 \le i \le m-1} \alpha_{im}^{(m)} X_{k_{(:,i)}}. \tag{3.13}$$

Rearranging (3.13) and taking 2-norm of the resulted expression, we then have

$$\begin{aligned} \|\alpha_{mm}^{(m)} X_{k_{(:,m)}} - W_{(:,m)}\|_2 \quad &\le \quad \|V_{(:,m+1:n)}\|_2 \|(R_{(1:m,1:m)})^{-1}\|_2 t_k^{(m)} \\ &\quad + \|\sum_{1 \le i \le m-1} \alpha_{im} X_{k_{(:,i)}}\|_2 \end{aligned} \tag{3.14}$$

Next, we bound $|\alpha_{im}|$. Multiplying $X_{k_{(:,j)}}^H$ on (3.13) for $1 \le j \le m-1$, we have

$$\alpha_{jm}^{(m)} = X_{k_{(:,j)}}^H W_{(:,m)} + X_{k_{(:,j)}}^H V_{(:,m+1:n)} X_{k,2}^{(m)} (X_{k,1}^{(m)})^{-1} (R_{(1:m,1:m)})^{-1} e_m$$

Therefore,

$$\begin{aligned} |\alpha_{jm}^{(m)}| \quad &\le \quad \|X_{k_{(:,j)}}^H W_{(:,m)}\|_2 + \|X_{k_{(:,j)}}^H V_{(:,m+1:n)} X_{k,2}^{(m)} (X_{k,1}^{(m)})^{-1} (R_{(1:m,1:m)})^{-1} e_m\|_2 \\ &\le \quad \left\| \frac{1}{\alpha_{jj}^{(j)}} W_{(:,m)}^H \left(\alpha_{jj}^{(j)} X_{k_{(:,j)}} - W_{(:,j)} \right) \right\|_2 + \|V_{(:,m+1:n)}\|_2 \|(R_{(1:m,1:m)})^{-1}\|_2 t_k^{(m)} \\ &\le \quad \left| \frac{1}{\alpha_{jj}^{(j)}} \right| \|\alpha_{jj}^{(j)} X_{k_{(:,j)}} - W_{(:,j)}\|_2 + \|V_{(:,m+1:n)}\|_2 \|(R_{(1:m,1:m)})^{-1}\|_2 t_k^{(m)} \end{aligned}$$

By the induction assumption, we have that

$$1 - |\alpha_{jj}^{(j)}| \le \|\alpha_{jj}^{(j)} X_{k_{(:,j)}} - W_{(:,j)}\| \le \Delta_j \max(t_k^{(1)}, \ldots, t_k^{(j)}).$$

Using the assumption (3.9), we get the bound for $|\alpha_{jm}^{(m)}|, 1 \le j \le m-1$

$$\begin{aligned} |\alpha_{jm}^{(m)}| \quad &\le \quad \frac{\Delta_j}{1 - \Delta_j \max(t_k^{(1)}, \ldots, t_k^{(j)})} \max(t_k^{(1)}, \ldots, t_k^{(j)}) \\ &\quad + \|V_{(:,m+1:n)}\|_2 \|(R_{(1:m,1:m)})^{-1}\|_2 t_k^{(m)} \\ &\le \quad \frac{\Delta_j}{1 - \min\{\frac{1}{2} gap_j, 1\}} \max(t_k^{(1)}, \ldots, t_k^{(j)}) \\ &\quad + \|V_{(:,m+1:n)}\|_2 \|(R_{(1:m,1:m)})^{-1}\|_2 t_k^{(m)} \end{aligned}$$

Thus, using these bounds on (3.14), we have

$$\begin{aligned}
&\|\alpha_{mm}^{(m)} X_{k_{(:,m)}} - W_{(:,m)}\|_2 \\
&\leq \|V_{(:,m+1:n)}\|_2 \|(R_{(1:m,1:m)})^{-1}\|_2 t_k^{(m)} \\
&\quad + \sum_{1\leq i\leq m-1} \frac{\Delta_j}{1-\min\{\frac{1}{2}gap_j, 1\}} \max(t_k^{(1)}, \ldots, t_k^{(j)}) \\
&\quad + \sum_{1\leq i\leq m-1} \|V_{(:,m+1:n)}\|_2 \|(R_{(1:m,1:m)})^{-1}\|_2 t_k^{(m)} \\
&\leq m\|V_{(:,m+1:n)}\|_2 \|(R_{(1:m,1:m)})^{-1}\|_2 t_k^{(m)} \\
&\quad + \sum_{1\leq i\leq m-1} \frac{\Delta_j}{1-\min\{\frac{1}{2}gap_j, 1\}} \max(t_k^{(1)}, \ldots, t_k^{(j)}) \\
&\leq \Delta_m \max(t_k^{(1)}, t_k^{(2)}, \ldots, t_k^{(m)})
\end{aligned}$$

This completes the induction proof. □

Lemma 3.3.3. *Under the assumptions and notations of Lemma 3.3.2, we have*

$$\left\| \frac{X_{k_{(:,j)}}}{\max(X_{k_{(:,j)}})} - \frac{W_{(:,j)}}{\max(W_{(:,j)})} \right\|_2 \leq 2n\Delta_j \max(t_k^{(1)}, \ldots, t_k^{(j)})$$

Proof. Because $W_{(:,j)}$ satisfies the assumption of Lemma 3.3.1 and

$$\|\alpha_{jj}^{(j)} X_{k_{(:,j)}} - W_{(:,j)}\|_\infty \leq \|\alpha_{jj}^{(j)} X_{k_{(:,j)}} - W_{(:,j)}\|_2 \leq \Delta_j \max(t_k^{(1)}, \ldots, t_k^{(j)}) \leq \frac{1}{2}gap_j$$

we apply Lemma 3.3.1 to

$$\begin{aligned}
\left\| \frac{X_{k_{(:,j)}}}{\max(X_{k_{(:,j)}})} - \frac{W_{(:,j)}}{\max(W_{(:,j)})} \right\|_\infty &= \left\| \frac{\alpha_{jj}^{(j)} X_{k_{(:,j)}}}{\max(\alpha_{jj}^{(j)} X_{k_{(:,j)}})} - \frac{W_{(:,j)}}{\max(W_{(:,j)})} \right\|_\infty \\
&\leq \frac{2\left(\|\alpha_{jj}^{(j)} X_{k_{(:,j)}} - W_{(:,j)}\|_\infty\right)}{\|W_{(:,p)}\|_\infty}
\end{aligned}$$

Then the conclusion follows

$$\left\| \frac{X_{k_{(:,j)}}}{\max(X_{k_{(:,j)}})} - \frac{W_{(:,j)}}{\max(W_{(:,j)})} \right\|_2 \leq 2n\Delta_j \max(t_k^{(1)}, \ldots, t_k^{(j)}).$$

□

Lemma 3.3.4. *Under the assumptions and notations of Lemma 3.3.2, we have*

$$\|X_k - X_{k-1}\|_2 \leq 4n\sqrt{np}M\left(\max(t_k^{(1)}, \ldots, t_k^{(p)}) + \max(t_{k-1}^{(1)}, \ldots, t_{k-1}^{(p)})\right)$$

where

$$M = \max_{1 \leq j \leq p} \Delta_j. \tag{3.15}$$

Proof. Let

$$\eta_k = \frac{X_{k_{(:,j)}}}{\max(X_{k_{(:,j)}})} - \frac{X_{k-1_{(:,j)}}}{\max(X_{k-1_{(:,j)}})}$$

Applying Lemma 3.3.3, we have

$$\begin{aligned}
\|\eta_k\|_2 &= \left\| \frac{X_{k_{(:,j)}}}{\max(X_{k_{(:,j)}})} - \frac{W_{(:,j)}}{\max(W_{(:,j)})} + \frac{W_{(:,j)}}{\max(W_{(:,j)})} - \frac{X_{k-1_{(:,j)}}}{\max(X_{k-1_{(:,j)}})} \right\|_2 \\
&\leq \left\| \frac{X_{k_{(:,j)}}}{\max(X_{k_{(:,j)}})} - \frac{W_{(:,j)}}{\max(W_{(:,j)})} \right\|_2 + \left\| \frac{W_{(:,j)}}{\max(W_{(:,j)})} - \frac{X_{k-1_{(:,j)}}}{\max(X_{k-1_{(:,j)}})} \right\|_2 \\
&\leq 2n\Delta_j \left(\max(t_k^{(1)}, \ldots, t_k^{(j)}) + \max(t_{k-1}^{(1)}, \ldots, t_{k-1}^{(j)})\right).
\end{aligned}$$

On the other hand, using $\|X_{k-1_{(:,j)}}\|_2 = \|X_{k_{(:,j)}}\|_2 = 1$, we get

$$\frac{1}{\max(X_{k-1_{(:,j)}})} - \|\eta_k\|_2 \leq \frac{1}{\max(X_{k_{(:,j)}})} \leq \frac{1}{\max(X_{k-1_{(:,j)}})} + \|\eta_k\|_2,$$

i.e.,

$$\left| \frac{1}{\max(X_{k_{(:,j)}})} - \frac{1}{\max(X_{k-1_{(:,j)}})} \right| \leq \|\eta_k\|_2.$$

Therefore,

$$\begin{aligned}
&\|X_{k_{(:,j)}} - X_{k-1_{(:,j)}}\|_2 \\
&\leq \left\| \left(\frac{X_{k_{(:,j)}}}{\max(X_{k_{(:,j)}})} - \frac{X_{k-1_{(:,j)}}}{\max(X_{k-1_{(:,j)}})} \right) \max(X_{k_{(:,j)}}) \right\|_2 \\
&\quad + \left\| \left(\frac{\max(X_{k_{(:,j)}})}{\max(X_{k-1_{(:,j)}})} - 1 \right) X_{k-1_{(:,j)}} \right\|_2 \\
&\leq \|\eta_k\|_2 \, |\max(X_{k_{(:,j)}})| + |\max(X_{k_{(:,j)}})| \left\| \frac{1}{\max(X_{k-1_{(:,j)}})} - \frac{1}{\max(X_{k_{(:,j)}})} \right\|_2 \\
&\leq \|\eta_k\|_2 \|X_{k_{(:,j)}}\|_\infty + \|X_{k_{(:,j)}}\|_\infty \|\eta_k\|_2 \\
&\leq 2\|\eta_k\|_2
\end{aligned}$$

So,

$$\begin{aligned}
\|X_k - X_{k-1}\|_2 &\le \sqrt{p}\|X_k - X_{k-1}\|_1 \\
&= \sqrt{p} \max_{1\le j\le p} \|X_{k_{(:,j)}} - X_{k-1_{(:,j)}}\|_1 \\
&\le \sqrt{np} \max_{1\le j\le p} \|X_{k_{(:,j)}} - X_{k-1_{(:,j)}}\|_2 \\
&\le 4n\sqrt{np}\left(\max_{1\le j\le p}(\Delta_j)\right)\left(\max(t_k^{(1)},\dots,t_k^{(p)}) + \max(t_{k-1}^{(1)},\dots,t_{k-1}^{(p)})\right) \\
&= 4n\sqrt{np}M\left(\max(t_k^{(1)},\dots,t_k^{(p)}) + \max(t_{k-1}^{(1)},\dots,t_{k-1}^{(p)})\right)
\end{aligned}$$

□

Lemma 3.3.5. *Under the assumptions and notations of Lemma 3.3.2, if*

$$\lim_{k\to\infty} \max\{t_k^{(1)},\dots,t_k^{(p)}\} = 0,$$

then $\lim R_k = R\Lambda_1^{-1}R^{-1}$.

Proof. From $\max\{t_k^{(1)},\dots,t_k^{(p)}\} \to 0$ and Lemma 3.3.2, we obtain $\alpha_{jj}^{(j)}X_{k_{(:,j)}} - W_{(:,j)} \to 0, 1 \le j \le p$. Then $\max\{\alpha_{jj}^{(j)}X_{k_{(:,j)}}\} \to \max\{W_{(:,j)}\}$. Because both $\max\{X_{k_{(:,j)}}\}$ and $\max\{W_{(:,j)}\}$ are positive, the sign of $\alpha_{jj}^{(j)}$ must be positive for sufficiently large k. Also, from $|\alpha_{jj}^{(j)}| = \|W_{(:,j)} + \alpha_{jj}^{(j)}X_{k_{(:,j)}} - W_{(:,j)}\|_2$, we have

$$1 - \|\alpha_{jj}^{(j)}X_{k_{(:,j)}} - W_{(:,j)}\|_2 \le |\alpha_{jj}^{(j)}| \le 1 + \|\alpha_{jj}^{(j)}X_{k_{(:,j)}} - W_{(:,j)}\|_2.$$

Taking the limit, we have $\lim|\alpha_{jj}^{(j)}| = 1$ and hence $\lim \alpha_{jj}^{(j)} = 1$. Therefore $\lim X_k = W$.

In addition, from the inexact inverse subspace iteration algorithm, we have

$$AY_k = AX_kR_k = BX_{k-1} + E_{k-1}$$

or

$$R_k = (X_k^HA^HAX_k)^{-1}X_k^HA^H(BX_{k-1} + E_{k-1}).$$

Thus

$$\lim R_k = (W^HA^HAW)^{-1}W^HA^HBW$$

Substituting W by V_1R^{-1} and using the relation $AV_1 = BV_1\Lambda_1$, we finally prove our result

$$\lim R_k = R\Lambda_1^{-1}R^{-1}.$$

$\square$

In what follows, set the inner residual reduction ratio $\delta_k = \dfrac{\epsilon_k}{\|Z_k\|_2}$ where $Z_k = AY_k - BX_k$.

Theorem 3.3.6. *Let $\Delta_j, 1 \le j \le p$ be defined as (3.10) and (3.11) in Lemma 3.3.2 and let M be defined as (3.15) in Lemma 3.3.4. Define*

$$t_0^{(j)} = \|X_{0,2}^{(j)}(X_{0,1}^{(j)})^{-1}\|_2$$

and

$$\rho = \max_{1\le j\le p}\{\rho^{(j)}\} \quad \textit{where} \quad \rho^{(j)} = |\lambda_j/\lambda_{j+1}|, \quad 1 \le j \le p.$$

If $\rho < \gamma < 1$ and $\epsilon_k = a\gamma^k$ with

$$a \le \min_{1\le j\le p}\{\frac{(1-\rho^{(j)})t_0^{(j)}}{\|V\|_2\|U\|_2(1+t_0^{(j)})(\rho^{(j)}+t_0^{(j)})}\}$$

Then

$$E_1\gamma \le \liminf \delta_k \le \limsup \delta_k \le E_2\gamma$$

where

$$E_1^{-1} = 4n\sqrt{np}M\|B\|_2\left(1+\frac{1}{\gamma}\right)\left(1-\frac{\rho}{\gamma}\right)^{-1}\rho\|V\|_2\|U\|_2 + 1$$

and

$$E_2^{-1} = \frac{sep(R\Lambda_1^{-1}R^{-1}, \Lambda_2^{-1})}{\|(AV)^{-1}\|_2\|V\|_2}\liminf\frac{t_k^{(p)}}{\epsilon_{k-1}}$$

Proof. Under the assumptions, we have by Theorem 3.2.6, $\max(t_k^{(1)}, \ldots, t_k^{(p)}) \to 0$.

$$\begin{aligned}
\|Z_k\|_2 &= \|AX_kR_k - BX_k\|_2 \\
&= \|AV_1X_k^{(1)}R_k + AV_2X_k^{(2)}R_k - AV_1\Lambda_1^{-1}X_k^{(1)} - AV_2\Lambda_2^{-1}X_k^{(2)}\|_2 \\
&= \|AV_1(X_k^{(1)}R_k - \Lambda_1^{-1}X_k^{(1)}) + AV_2(X_k^{(2)}R_k - \Lambda_2^{-1}X_k^{(2)})\|_2 \\
&= \left\| A(V_1, V_2) \begin{bmatrix} X_k^{(1)}R_k - \Lambda_1^{-1}X_k^{(1)} \\ X_k^{(2)}R_k - \Lambda_2^{-1}X_k^{(2)} \end{bmatrix} \right\|_2 \\
&\geq \left\| \begin{bmatrix} X_k^{(1)}R_k - \Lambda_1^{-1}X_k^{(1)} \\ X_k^{(2)}R_k - \Lambda_2^{-1}X_k^{(2)} \end{bmatrix} \right\|_2 / \|(AV)^{-1}\|_2 \\
&\geq sep(R_k, \Lambda_2^{-1})\|X_k^{(2)}\|_2 / \|(AV)^{-1}\|_2 \\
&\geq sep(R_k, \Lambda_2^{-1}) \cdot t_k^{(p)} / \|(X_k^{(1)})^{-1}\|_2 \|(AV)^{-1}\|_2 \\
&\geq \frac{t_k^{(p)}}{1 + t_k^{(p)}} \cdot \frac{sep(R_k, \Lambda_2^{-1})}{\|(AV)^{-1}\|_2 \|V\|_2}
\end{aligned}$$

where we have used Lemma 3.2.3 for the bound of $\|(X_k^{(1)})^{-1}\|_2$ and

$$sep(R_k, \Lambda_2^{-1}) = \inf_{\|P\|_F=1} \|PR_k - \Lambda_2^{-1}P\|_2 \leq \frac{\|X_k^{(2)}R_k - \Lambda_2^{-1}X_k^{(2)}\|_2}{\|X_k^{(2)}\|_2}.$$

By Lemma (3.3.5), thus we have

$$\liminf \frac{\|Z_k\|}{\epsilon_{k-1}} \geq \frac{sep(R\Lambda_1^{-1}R^{-1}, \Lambda_2^{-1})}{\|(AV)^{-1}\|_2 \|V\|_2} \liminf \frac{t_k^{(p)}}{\epsilon_{k-1}}.$$

On the other hand, as $\max\{t_k^{(1)}, \ldots, t_k^{(j)}\}$ approaches zero, the assumption of Lemma 3.3.4 is satisfied for sufficiently large k, where we note that $\alpha_{jj}^{(j)}$ is a function of k. We use the conclusion of Lemma 3.3.4 and get the upper bound of $\|Z_k\|_2$ as follows.

$$\begin{aligned}
\|Z_k\|_2 &= \|AX_kR_k - BX_k\|_2 \\
&= \|BX_{k-1} + E_{k-1} - BX_k\|_2 \\
&= \|B(X_{k-1} - X_k) + E_{k-1}\|_2 \\
&\leq \|B\|_2\|X_{k-1} - X_k\|_2 + \|E_{k-1}\|_2 \\
&\leq 4n\sqrt{np}M\|B\|_2 \Big((\max(t_k^{(1)}, \ldots, t_k^{(p)}) + \max(t_{k-1}^{(1)}, \ldots, t_{k-1}^{(p)}) \Big) + \epsilon_{k-1}
\end{aligned}$$

Thus,

$$\begin{aligned}\limsup \frac{\|Z_k\|_2}{\epsilon_{k-1}} &\leq 4n\sqrt{np}M\|B\|_2 \limsup \frac{\max(t_k^{(1)}, \ldots, t_k^{(p)})}{\epsilon_{k-1}} \\ &\quad + 4n\sqrt{np}M\|B\|_2 \limsup \frac{\max(t_{k-1}^{(1)}, \ldots, t_{k-1}^{(p)})}{\epsilon_{k-1}} + 1\end{aligned}$$

Then, apply the Corollary 3.2.7 to simplify the bound, i.e.,

$$\begin{aligned}\limsup \frac{\|Z_k\|_2}{\epsilon_{k-1}} &\leq 4n\sqrt{np}M\|B\|_2 \left(1+\frac{1}{\gamma}\right)\left(1-\frac{\max\limits_{1\leq j\leq p}\{\rho^{(j)}\}}{\gamma}\right)^{-1} \\ &\quad \times \max_{1\leq j\leq p}\{\rho^{(j)}\}\|V\|_2\|U\|_2 + 1.\end{aligned}$$

Then let

$$E_1^{-1} = 4n\sqrt{np}M\|B\|_2 \left(1+\frac{1}{\gamma}\right)\left(1-\frac{\rho}{\gamma}\right)^{-1} \rho\|V\|_2\|U\|_2 + 1$$

and

$$E_2^{-1} = \frac{sep(R\Lambda_1^{-1}R^{-1}, \Lambda_2^{-1})}{\|(AV)^{-1}\|_2\|V\|_2} \liminf \frac{t_k^{(p)}}{\epsilon_{k-1}},$$

then we have proved that $E_1\gamma \leq \liminf \delta_k \leq \limsup \delta_k \leq E_2\gamma$. □

The results above show that, at the convergence stage, δ_k is bounded below and, if $\liminf \frac{t_k^{(p)}}{\epsilon_{k-1}}$ is not 0, it is also bounded above. Hence δ_k is asymptotically near a constant. Note that we have that $\frac{t_k^{(p)}}{\epsilon_{k-1}}$ is bounded by Corollary 3.2.7, but it is possible that $t_k^{(p)}$ converges to 0 faster than ϵ_{k-1}. Obviously, this is a welcome but unlikely situation in general.

3.4 Numerical examples

In this section we present some numerical experiments aimed at illustrating the convergence behavior and the effectiveness of inexact inverse subspace iteration algorithm. We performed several tests involving different types of matrices. The purposes of these experiments are to verify the convergence results. Also we conduct numerical comparison with the Jacobi-Davidson method. In general, the Jacobi-Davidson method is a much faster convergent method but it may fail to converge in some difficult problems where good initial vectors are not found. The subspace iteration is generally more robust and our results demonstrate this for matrices with very closely clustered eigenvalues.

We examine convergence of the following residual in our numerical experiment:

$$\|Z_k\|_2 = \|AX_kR_k - BX_k\|_2.$$

In the numerical results listed in the tables below, "iter" denotes the number of outer iterations; "MV" is the total number of matrix-vector products; "CPU" is the cpu time; and "residual" is the residual norm $\|Z_k\|_2$.

Example 1. *The matrix A is the finite-difference discretization (center difference) on a 32×32 grid of the following eigenvalue problem of the convection diffusion operator:*

$$-\Delta u + 5u_x + 5u_y = \lambda u \ \textit{on} \ (0,1)^2,$$

with the homogeneous Dirichlet boundary condition. B is the matrix of the same dimension as A and is of the form

$$B = \begin{pmatrix} \frac{1}{1^2} & & & & & \\ & \frac{1}{2^2} & & & & \\ & & \frac{10}{3^2} & & & \\ & & & \frac{1}{4^2} & & \\ & & & & \ddots & \\ & & & & & \frac{1}{n^2} \end{pmatrix}.$$

We compute the three smallest eigenvalues with random initial vectors.

In this case, $\rho^{(1)} = \lambda_1/\lambda_2 \approx 0.6655, \rho^{(2)} = \lambda_2/\lambda_3 \approx 0.2262, \rho^{(3)} = \lambda_3/\lambda_4 \approx 0.3756$, thus

$$\max(0.6655, 0.2262, 0.3756) = 0.6655.$$

We consider the convergence behavior of the outer iteration under different parametric values of γ. In Figure 3.1, we present the convergence history of the residual $||Z_k||_\infty$ and the threshold $\epsilon_k = \gamma^k$ (in dotted lines) for various values of γ. The residuals are plotted in the solid, dash-dotted, and dashed lines while ϵ_k is plotted in the dotted lines from the top down for $\gamma = 0.6655, 0.3756$, and 0.2262 respectively in the left figure and for $\gamma = 0.95, 0.85$ and 0.7 respectively in the right figure. On both, the residual for the exact inverse subspace iteration is plotted in the $\times$ mark. In this example, the dashed-dotted line and the dashed lines overlap with the "$\times$" line in the left figure. The numerical results showed that the convergence rate is close to $\max\{\gamma, \max_{1\le j\le p}\{\rho^{(j)}\}\}$.

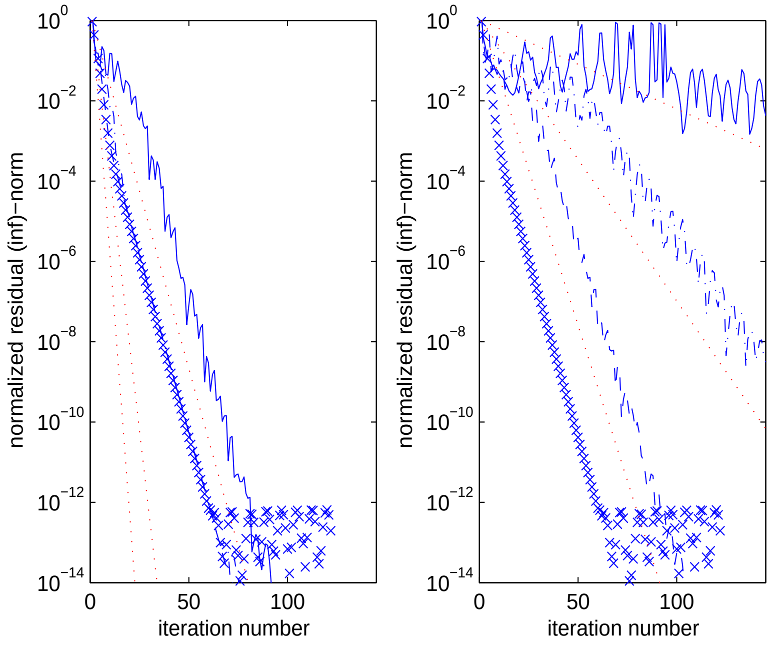

Figure 3.1: Residual convergence history of IIS for computing 3 eigenvalues.

In the next three examples, we compare the inexact inverse subspace iteration with the Jacobi-Davidson algorithm. We see that the Jacobi-Davidson method does not perform well for these examples where the eigenvalues sought are severely clustered, while the inexact inverse subspace iteration finds eigenpair approximations with no difficulties.

Example 2. *The matrix A is derived from the finite-difference discretization of the partial differential equation*

$$-\Delta u + 20u_x - 30u_y = \lambda u,$$

on $[0,1] \times [0,1]$ with homogeneous Dirichlet boundary conditions. We discretize the system by using 32 mesh points in each direction. We use $B = 10 * speye(1024)$. We compare our algorithm with JD method for computing 3 eigenvalues close to $\tau = 0.7952$. Then $\rho = \max_{1\le i\le 3} \{\rho^{(i)}\} \approx 0.52$ and $tol = 1e-8$. The γ in our algorithm is respectively $\rho, 0.4, 0.3, 0.2$ in the left figure and $0.85, 0.75, 0.65, \rho$ in the right figure for computing three eigenpairs. For each of the methods, we list the performance statistics in Table 3.1. Convergence of residual is plotted in Figure 3.2 for Jacobi-Davidson method and in Figure 3.3 for inexact inverse subspace iteration (IIS) .

Algorithm	iter	residual	CPU (secs)	MV
JD	8000	1e-003	356.26	24629
IIS, $\gamma = 0.85$	147	1e-009	85.754	6924
IIS, $\gamma = 0.75$	84	1e-009	79.955	7088
IIS, $\gamma = 0.65$	57	1e-009	62.93	5832
IIS, $\gamma = 0.52$	38	1e-009	57.543	5428
IIS, $\gamma = 0.4$	33	1e-009	67.848	6264
IIS, $\gamma = 0.3$	33	1e-009	85.333	8132
IIS, $\gamma = 0.2$	31	1e-009	96.288	8726

Table 3.1: A comparison between JD and IIS for computing 3 eigenvalue

We see from Figure 3.2 that Jacobi-Davidson method does not converge to the specified tolerance $1e-8$ in this case. Figure 3.3 shows that inexact inverse subspace iteration converges quickly for all the choices of γ. And the convergence rate is close to 0.52 which is $\rho = \max\{\rho^{(i)}\}, 1 \le i \le 3$ when $\gamma \le \rho$. The convergence rate is close to γ when $\gamma \ge \rho$.

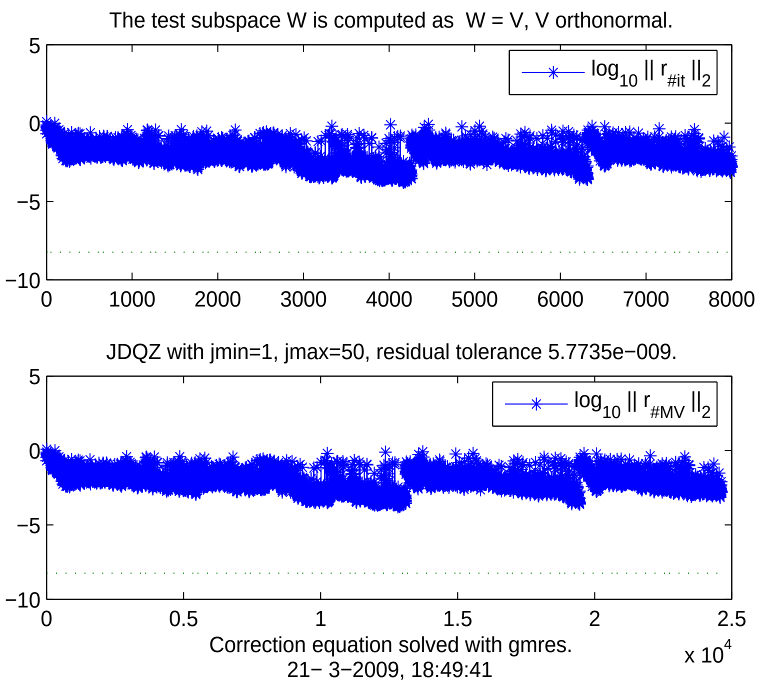

Figure 3.2: Residual convergence of Jacobi-davidson for computing 3 eigenvalues.

Example 3. *The matrix A in this example is the matrix add20 from the Matrix Market collection [19]. The matrix $B = 10 * speye(2395)$. We compare our algorithm with JD method for computing 4 eigenvalues closest to $\tau = 0.09665$. In this case the tolerance is $tol = 1e-8$. Again, let $\rho = \max\{\rho^{(i)}\} \approx 0.881, 1 \le i \le 4$. The γ in our algorithm is respectively $\gamma = \rho, 0.9, 0.95$ in the left of Figure 3.4 and $\gamma = \rho, 0.7, 0.5$ in the right figure. For each of the methods, we list the performance statistics in Table 3.2. Convergence of residual is plotted in Figure 3.4 for inexact inverse subspace iteration(IIS).*

Algorithm	iter	residual	CPU (secs)	MV
JD	1200	7.4-007	4274.3	197219
IIS, $\gamma = 0.95$	638	9.22e-009	3792.363	69080
IIS, $\gamma = 0.9$	311	8.90e-009	2917.255	49082
IIS, $\gamma = 0.881$	258	9.85e-009	2868.024	43886
IIS, $\gamma = 0.7$	175	9.92e-009	2212.431	38508
IIS, $\gamma = 0.5$	183	9.99e-009	2435.793	44314

Table 3.2: A comparison between JD and IIS for computing 4 eigenvalues of add20

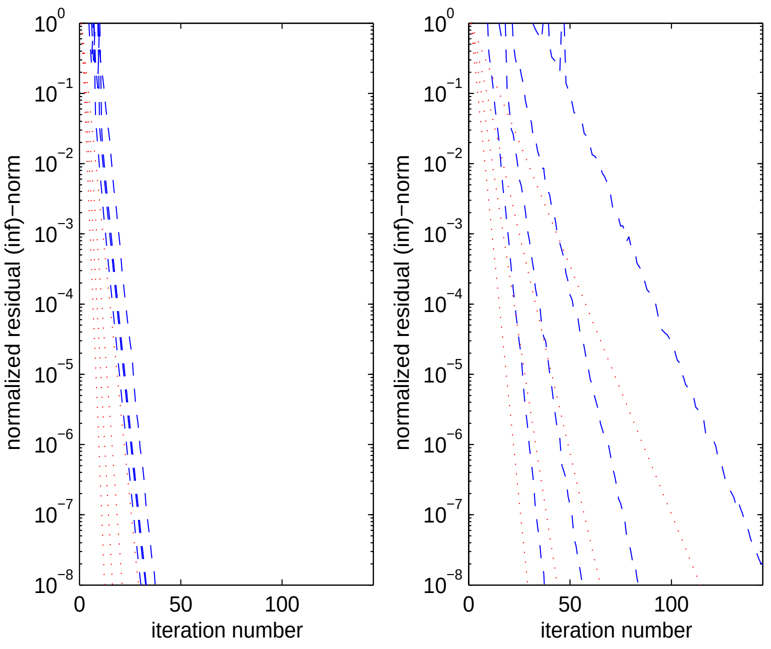

Figure 3.3: Residual convergence of IIS for computing 3 eigenvalues.

From Table 3.2, we see that Jacobi-Davidson only converges to $7.4e-7$, yet the inexact inverse subspace iteration converges to the tolerance with less iterations and less CPU times. The inexact inverse subspace iteration also uses much less matrix-vector products and presents faster convergence for all the current choices of γ. Figure 3.4 further confirms our analysis of the convergence rate of inexact inverse subspace iteration. The convergence rate is $\max\{\gamma, \rho\}$ with $\rho = \max\{\rho^{(i)}\}, 1 \leq i \leq 4$.

Example 4. *The example we use is the matrix dw2048 from Matrix market collection [19]. The matrix $B = 10 * speye(2048)$. We compare our algorithm with JD method for computing 3 eigenvalues closest to $\tau = 0.09665$. Let $\rho = \max\{\rho^{(i)}\} \approx 0.9187, 1 \leq i \leq 3$. In Figure 3.5, the γ in our algorithm is respectively $\rho, 0.7, 0.4$ in the left figure and $0.95, \rho$ in the right figure. Once again, we list the performance statistics in Table 3.3 for both methods. The convergence of residual for the inexact inverse subspace iteration is plotted in Figure 3.5. Once again, the inexact inverse subspace iteration presents better performance.*

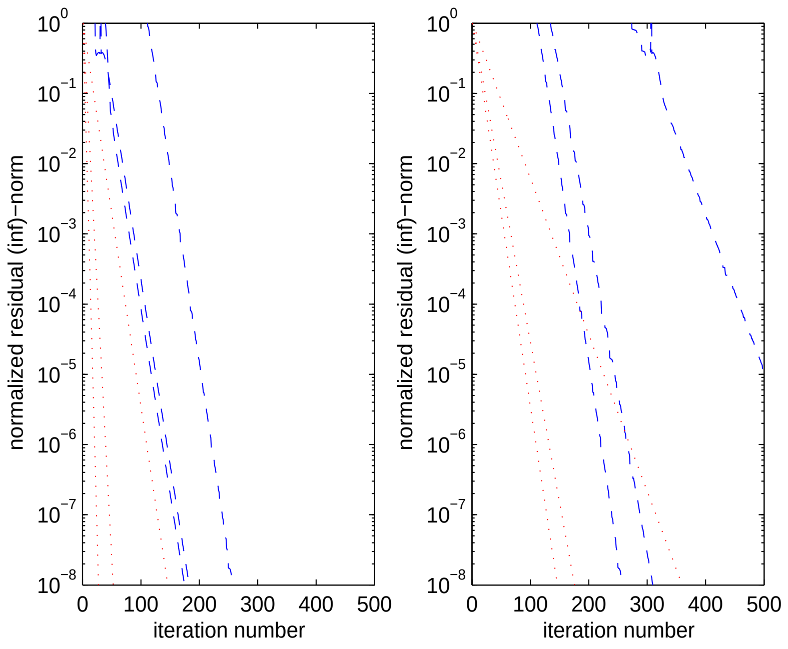

Figure 3.4: Residual convergence of IIS for computing 4 eigenvalues of add20.

Algorithm	iter	residual	CPU (secs)	MV
JD	900	3.5e-005	2778.1 secs	197550
IIS, $\gamma = 0.95$	504	9.57e-009	2047.144	41490
IIS, $\gamma = 0.9187$	306	8.82e-009	1374.5	30000
IIS, $\gamma = 0.7$	243	9.74e-009	1889.698	41318
IIS, $\gamma = 0.4$	205	9.89e-009	2242.435	47766

Table 3.3: A comparison between JD and IIS for computing 3 eigenvalues of dw2048

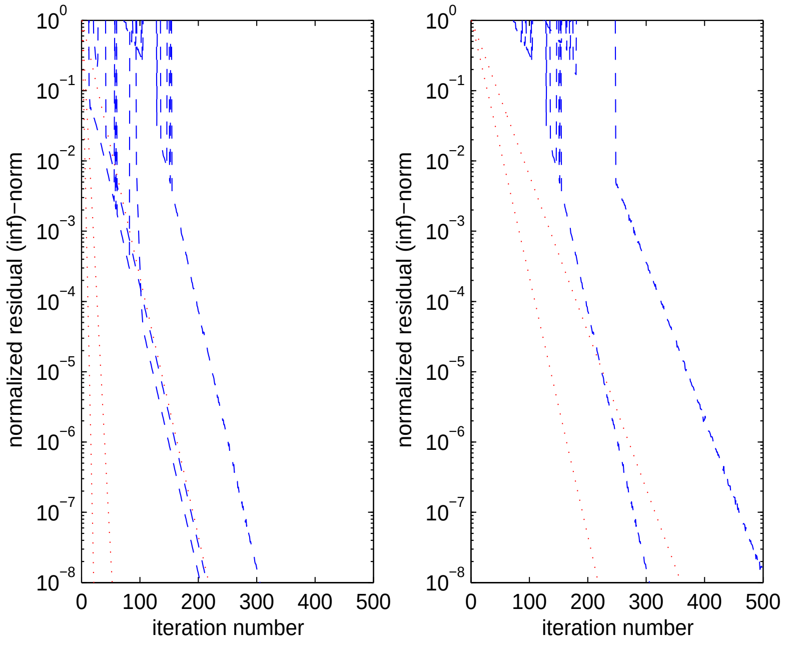

Figure 3.5: Residual convergence of IIS for computing 3 eigenvalues of dw2048.

Chapter 4 Krylov Subspace Method for the Evaluation of Matrix Exponential

This part of the thesis is devoted to the problem of computing the product of a matrix exponential and a vector of the form

$$e^{-\tau A}v \tag{4.1}$$

by using Krylov subspace approximation methods. Here A is a large, sparse and symmetric positive definite matrix, v is a vector and τ is a positive constant. Often, τ is the time step parameter in a finite difference time-stepping method.

The problem (4.1) occurs in many applications. A simple example is the solution of the ordinary differential equations of the form:

$$\frac{dv(t)}{dt} = -Av(t) + r(t), \quad v(0) = v_0, \tag{4.2}$$

or the discretized partial differential equation of the following

$$\begin{cases} \dfrac{\partial u(x,t)}{\partial t} = -Lu(x,t) + r(t), & x \in \Omega \\ u(x,0) = u_0, & x \in \Omega \\ u(x,t) = \sigma(x), & x \in \partial\Omega, t > 0 \end{cases} \tag{4.3}$$

where L is a positive definite self-adjoint differential operator. By discretizing (4.3) with respect to the space variable, the partial differential equation is reduced to the ordinary differential equation of type (4.2). As is well known, the solution of the system (4.2) is

$$v(t) = e^{-tA}v_0 + \int_0^t e^{(s-t)A}r(s)ds. \tag{4.4}$$

The numerical solution for (4.4) by a time-stepping procedure is based on approximation of the formula

$$v(t+\tau) = e^{-\tau A}v(t) + \int_0^\tau e^{-(\tau-\delta)A}r(t+\delta)d\delta. \tag{4.5}$$

The calculation of (4.5) involves the matrix-vector product of form (4.2). There are many other practical applications in which the exponential integrators only involve the evaluation or approximation of the product of the exponential matrix with a vector, such as, functions in statistical methods for spatial date and other complex structures [23], solutions to fractional-in-space partial differential equations [24] and solutions of differential equations [12], etc.

Notice that, in (4.1), if one calculates the exponential matrix explicitly and then find the matrix-vector product, the computation and storage cost would be extremely expensive even though A is sparse. But computing and storing $e^{-\tau A}v$ may be done more efficiently. Similarly to computing $A^{-1}v$, Krylov subspace projection technique is an efficient technique for these types of problems. Krylov subspace projection method to approximate the matrix exponential is based on a combination of the Krylov subspace projection and the computation of matrix-vector product instead of the matrix exponential itself. One first projects the exponential of the large and sparse matrix into a small Krylov subspace, and then uses the transformed exponential of the compression matrix to approximate the original matrix-vector product.

The earliest work on theoretical analysis of Krylov subspace approximation method goes to Saad [38] in 1992 in which a priori and a posteriori error estimates are established. He also points out that the Krylov subspace approximation approach, which has been used with success in several applications, provides a systematic way of defining high order explicit-type schemes for solving systems of ordinary differential equations or time-dependent partial differential equations. Following this work, several different Krylov subspace approximation approaches have been proposed, such as the polynomial methods of calculating general function including matrix exponential function [49], variants of the standard Krylov subspace approximation methods [12, 16, 22], extended Krylov subspace approximation methods [13, 25], and restarted Krylov subspace approximation methods [1, 14].

Despite the success of the Krylov subspace approximation techniques to matrix exponential operators in many applications [36], analysis of standard methods is based on the use of small time step τ. For many problems, larger time steps are desirable. For this purpose, we propose the Krylov subspace method generated by A^{-1}, which allows larger time steps in some cases. It turns out a similar idea has also been considered. In [16], the Krylov subspace method based on Krylov subspace generated by $(I + \sigma A)^{-1}$ is considered with σ chosen to minimize the error bound.

We shall introduce a novel technique for analyzing convergence of all three methods mentioned above. We compare all three Krylov subspace approximation methods by their respective theoretical error bounds. Numerical examples are also given to verify our theoretical bounds. The chapter is closed by using the three Krylov subspace approximation methods to solve the system of ODEs of the form (4.2).

4.1 Standard Lanczos approximation

We recall the definition of the m^{th} Krylov subspace of $A \in \mathbf{R}^{n\times n}$ and $0 \neq v \in \mathbf{R}^n$ given by

$$K_m(A, v) = \text{ span } \{v, Av, \ldots, A^{m-1}v\} = \{q(A)v : q \in \mathcal{P}_{m-1}\}.$$

We consider a sequence of approximations $w_m = q(A)v \in K_m(A, v)$ to $e^{-\tau A}v$ with polynomials $q \in \mathcal{P}_{m-1}$ which in some sense approximate the exponential function. The standard Lanczos approximation is based on the Lanczos decomposition for A and v. Given v, Lanczos process generates V_m, T_m such that

$$AV_m = V_m T_m + \beta_{m+1} v_{m+1} e_m^T, \text{ where } V_m e_1 = v_1. \tag{4.6}$$

Here, the columns of $V_m = [v_1, v_2, \ldots, v_m]$ form an orthonormal basis of $K_m(A, v)$ with $v_1 = v/\|v\|$, T_m is a unreduced tridiagonal matrix, and $e_m \in \mathbf{R}^n$ denotes the m^{th} unit coordinate vector. For more details about the standard Lanczos algorithm, see Section 2.2.2.

The vector $V_m V_m^T e^{-\tau A} v$ is the projection of $e^{-\tau A} v$ on $K_m(A, v)$, which is the closest approximation to $e^{-\tau A} v$ from $K_m(A, v)$. Let $\beta = \|v\|$ and $v = \beta v_1$, then

$$V_m V_m^T e^{-\tau A} v = \beta V_m V_m^T e^{-\tau A} v_1 = \beta V_m V_m^T e^{-\tau A} V_m e_1 \approx \beta V_m e^{-\tau T_m} e_1.$$

$e^{-\tau T_m} e_1$ is used to approximate $V_m^T e^{-\tau A} V_m e_1$. Then the standard Lanczos approximation to $e^{-\tau A} v$ is given by

$$w_m^{SL} := \beta V_m e^{-\tau T_m} e_1. \tag{4.7}$$

The superscript SL represents its correspondence to the standard Lanczos method. We also define the error

$$E_m^{SL}(\tau) = w(\tau) - w_m^{SL}(\tau), \tag{4.8}$$

where $w(\tau)$ is $e^{-\tau A} v$. Saad [38] has proved that the error of the above approximation is related to the norm of the matrix and the dimension of the subspace.

Theorem 4.1.1. *Let A be any matrix and let $\rho = \|A\|$. Then the error of the approximation using Lanczos method satisfies*

$$\|e^{-\tau A} v - \beta V_m e^{-\tau T_m} e_1\| \leq 2\beta \frac{(\tau\rho)^m e^{\tau\rho}}{m!}$$

From the theorem, we can see clearly that the smaller ρ is, the better the approximation. For symmetric positive definite matrix A, ρ will be the largest eigenvalue of A. If $\|A\|$ turns out to be a large number, we will have to use very small τ to reduce the error. Note that in the context of time-stepping (4.5), τ is the step size. This priori error bound may turn out to be pessimistic. Next, we present a posteriori bound.

4.1.1 Behavior of $|e_m^T e^{-\tau T_m} e_1|$

In what follows, we first analyze the behavior of the absolute value of the $(m, 1)$ entry of the function $e^{-\tau T_m}$. In Bezni and Golub [4], it was shown that the function of a bounded matrix has its entries decay away from the diagonal, see section 2.5. Using the decay bounds of Benzi and Golub, then we can get a bound for the $(m, 1)$ entry of the matrix $e^{-\tau T_m}$.

Lemma 4.1.2. *Let T_m be symmetric and tridiagonal. Denote $a = \lambda_{\min}(T_m)$, the smallest eigenvalue of T_m, and $b = \lambda_{\max}(T_m)$, the largest eigenvalue of T_m. For any fixed q such that $0 < q < 1$, we have*

$$|e_m^T e^{-\tau T_m} e_1| \leq \frac{2}{1-q} e^{\frac{\tau(b-a)\left(q+\frac{1}{q}\right)}{4}} q^{m-1}.$$

Proof. Define $f(\lambda) = e^{-\tau\lambda}$ and $F = fo\psi^{-1}$, where $\psi : \mathbb{C} \to \mathbb{C}$ is defined as

$$\psi(\lambda) = \frac{2\lambda - (a+b)}{b-a}.$$

Then $\psi([a, b]) = [-1, 1]$. Define

$$B = \psi(T_m) = \frac{2}{b-a} T_m - \frac{a+b}{b-a} I.$$

Then the spectrum of the symmetric matrix B is contained in $[-1, 1]$. Furthermore, let $\chi = \dfrac{1}{q}$. Define an ellipse ϵ_χ which has $-1, 1$ as its foci, and $\alpha = \dfrac{\chi^2+1}{2\chi}, \beta = \dfrac{\chi^2-1}{2\chi}$ as its semi-major axis and semi-minor axis respectively, $\alpha > \beta > 0, \alpha > 1$. Since f is analytic on $\mathbb{C}$, f is analytic in the interior of the ellipse $\varepsilon_\chi, \chi > 1$, and continuous on ε_χ. Therefore, f satisfies the assumptions of Theorem 2.5.2 in the preliminary. Applying Theorem 2.5.2, we have the decay bound of the $(m, 1)$ entry of the matrix $e^{-\tau T_m}$

$$|e_m^T e^{-\tau T_m} e_1| \leq K q^{m-1}, \quad q = \frac{1}{\chi},$$

where

$$K = \max\{K_0, \|F(B)\|\}, \quad K_0 = \frac{\chi M(\chi)}{\chi - 1}, \quad \text{and } M(\chi) = \max_{z \in \varepsilon_\chi} |F(z)|$$

We now look at the bound for $M(\chi)$. Let $z = x + iy \in \varepsilon_\chi$. Set

$$u = \frac{(b-a)x + a + b}{2}, \quad v = \frac{b-a}{2} y.$$

Then

$$|F(z)| = |e^{-\tau(u+iv)}| = e^{-\tau u}|e^{-i\tau v}| = e^{-\tau u}.$$

We know that $\alpha = \dfrac{\chi^2+1}{2\chi} = \dfrac{1}{2}\left(q + \dfrac{1}{q}\right)$. Therefore ,

$$\begin{aligned} M(\chi) &= \max_{z\in\varepsilon_\chi} |F(z)| \\ &= \max_{-\alpha\le x\le\alpha} e^{\frac{\tau((a-b)x-a-b)}{2}} \\ &= e^{\frac{\tau((b-a)\alpha-a-b)}{2}} \\ &= e^{\frac{\tau\left((b-a)\left(q+\frac{1}{q}\right)-2(a+b)\right)}{4}} \end{aligned}$$

We then get the bound for K_0,

$$K_0 = \frac{2\chi M(\chi)}{\chi-1} < \frac{2}{1-q} e^{\frac{\tau\left((b-a)\left(q+\frac{1}{q}\right)-2(a+b)\right)}{4}} \le \frac{2}{1-q} e^{\frac{\tau(b-a)\left(q+\frac{1}{q}\right)}{4}}.$$

For the expression $\|F(B)\|$, we can bound it as follows:

$$\|F(B)\| = \|e^{-\tau T_m}\| \le e^{-\tau a} \le 1.$$

Since $\dfrac{2}{1-q} e^{\frac{\tau(b-a)\left(q+\frac{1}{q}\right)}{4}} > 1$, we have

$$|e_m^T e^{-\tau T_m} e_1| \; < \; \frac{2}{1-q} e^{\frac{\tau(b-a)\left(q+\frac{1}{q}\right)}{4}} q^{m-1}.$$

□

Remark. We see from the bound that, if $q \to 0$, the coefficient $e^{\frac{\tau(b-a)\left(q+\frac{1}{q}\right)}{4}} \to \infty$. Thus while small q gives a faster decay term q^{m-1}, its coefficient also become larger. If τ is such that $\frac{\tau(b-a)}{4} < 1$, then letting $q = \frac{\tau(b-a)}{4}$, we have

$$|e_m^T e^{-\tau T_m} e_1| < \frac{2}{1-q} e^2 q^{m-1}.$$

In this case, $|e_m^T e^{-\tau T_m} e_1|$ decays at least at the rate of $q = \dfrac{\tau(b-a)}{4}$.

We note that the actual bound in the lemma is pessimistic for practical estimation of the value $|e_m^T e^{-\tau T_m} e_1|$. But it shows the fast decay rate of entry $(m, 1)$ as the projection dimension m increases.

4.1.2 Error Bounds

In this section, we consider approximating $w = e^{-\tau A}v$ by using the standard Lanczos approximation. Let the approximate solution $w_m^{SL}(t)$ be defined as in (4.7). T_m is the projection of A onto the Krylov subspace $K_m(A, v)$. The error $E_m^{SL}(t)$ is defined as in (4.8). We have the following posterior error bound on the standard Lanczos approximation.

Theorem 4.1.3. *The error of the standard Lanczos approximation to the matrix exponential on $K_m(A, v)$ satisfies*

$$\|E_m^{SL}(\tau)\| \leq \tau\beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T e^{-tT_m} e_1|.$$

where $\beta = \|v\|$, T_m and β_{m+1} are generated from the Lanczos process (4.6).

Proof. As we know that $w(t) = e^{-tA}v$ is a solution of

$$w(t)' = -Aw(t), \quad w(0) = v.$$

$w_m^{SL}(t) = \beta V_m e^{-tT_m} e_1$ is the approximate solution. And $w_m^{SL}(t)' = -\beta V_m T_m e^{-tT_m} e_1$. Using (4.6), we have

$$\begin{aligned} w_m^{SL}(t)' &= -\beta(AV_m - \beta_{m+1} v_{m+1} e_m^T) e^{-tT_m} e_1 \\ &= -\beta A V_m e^{-tT_m} e_1 + \beta\beta_{m+1} v_{m+1} e_m^T e^{-tT_m} e_1 \\ &= -A w_m^{SL}(t) + \beta\beta_{m+1}(e_m^T e^{-tT_m} e_1) v_{m+1} \end{aligned}$$

Since $E_m^{SL}(t) = w(t) - w_m^{SL}(t), E_m(0) = 0$ and

$$E_m^{SL}(t)' = -AE_m^{SL}(t) - \beta\beta_{m+1}(e_m^T e^{-tT_m} e_1) v_{m+1}$$

By solving the above ODE, the posterior error is:

$$\begin{aligned} E_m^{SL}(\tau) &= \textstyle\int_0^\tau e^{(t-\tau)A} \left(-\beta\beta_{m+1}(e_m^T e^{-tT_m} e_1) v_{m+1}\right) dt \\ &= -\beta\beta_{m+1} \textstyle\int_0^\tau (e_m^T e^{-tT_m} e_1) e^{(t-\tau)A} v_{m+1} dt \end{aligned}$$

Taking the norm of the error, we have

$$\begin{aligned}\|E_m^{SL}(\tau)\| &\leq \beta|\beta_{m+1}|\max_{0\leq t\leq\tau}|e_m^Te^{-tT_m}e_1|\int_0^\tau\|e^{(t-\tau)A}\|dt\|v_{m+1}\| \\ &\leq \tau\beta|\beta_{m+1}|\max_{0\leq t\leq\tau}|e_m^Te^{-tT_m}e_1|\end{aligned}$$

□

Remark 2. Observe from the above bound that there are four factors that determine the magnitude of the bound, τ, β, $|\beta_{m+1}|$ and $\max_{0\leq t\leq\tau}|e_m^Te^{-\tau T_m}e_1|$. First, the smaller τ is, the smaller the bound of $\|E_m^{SL}(\tau)\|$. Second, the value β which is the norm of the initial vector $w(0) = v$. The smaller the norm of v, the better the bound. Third, the value $|\beta_{m+1}|$ can be bounded in terms of $\|A\|$. The quantity $\max\limits_{0\leq t\leq\tau}\{|e_m^Te^{-tT_m}e_1|\}$ becomes small as m increases. From the bound of $|e_m^Te^{-\tau T_m}e_1|$, we know that increasing the dimension of the Krylov subspace will reduce the value $\max\limits_{0\leq t\leq\tau}\{|e_m^Te^{-tT_m}e_1|\}$, thereby reducing the approximation error.

Remark 3. Notice that, if the eigenvalues of A are large, then $|\beta_{m+1}|$ is large. $\max_{0\leq t\leq\tau}|e_m^Te^{-tT_m}e_1|$ might not be small enough to bring down the error to the desirable level. Increasing the dimension will reduce the error to certain level and then start to stagnate. At the early stage of reducing τ, the value $\max_{0\leq t\leq\tau}|e_m^Te^{-tT_m}e_1|$ may increase, but if τ is small enough, we are able to reduce error to any desirable level.

Corollary 4.1.4. *For any $0 < q < 1$, the error of the standard Lanczos approximation to the matrix exponential on $K_m(A, v)$ satisfies*

$$\|E_m^{SL}(\tau)\| \leq 2\beta\tau\|A\|\frac{2}{1-q}e^{\frac{\tau(\lambda_n-\lambda_1)\left(q+\frac{1}{q}\right)}{4}}q^{m-1}$$

where $\lambda_1 = \lambda_{\min}(A), \lambda_n = \lambda_{\max}(A)$.

Proof. Let T_m be the projection of A onto the Krylov subspace $K_m(A, v)$. By Lemma

4.1.2 and Theorem 4.1.3, we have

$$\begin{aligned}
\|E_m^{SL}(\tau)\| &\leq \tau\beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T e^{-tT_m} e_1| \\
&\leq \tau\beta|\beta_{m+1}| \frac{2}{1-q} \max_{0\leq t\leq\tau} \{e^{\frac{t(\lambda_{\max}(T_m)-\lambda_{\min}(T_m))\left(q+\frac{1}{q}\right)}{4}}\} q^{m-1} \\
&\leq \tau\beta|\beta_{m+1}| \frac{2}{1-q} e^{\frac{\tau(\lambda_{\max}(T_m)-\lambda_{\min}(T_m))\left(q+\frac{1}{q}\right)}{4}} q^{m-1}.
\end{aligned}$$

And noting that $\lambda_{\min}(T_m) \geq \lambda_{\min}(A)$ and $\lambda_{\max}(T_m) \leq \lambda_{\max}(A)$, we have

$$\begin{aligned}
\|E_m^{SL}(\tau)\| &\leq \tau\beta|\beta_{m+1}| \frac{2}{1-q} e^{\frac{\tau(\lambda_{\max}(A)-\lambda_{\min}(A))\left(q+\frac{1}{q}\right)}{4}} q^{m-1} \\
&= \tau\beta|\beta_{m+1}| \frac{2}{1-q} e^{\frac{\tau(\lambda_n-\lambda_1)\left(q+\frac{1}{q}\right)}{4}} q^{m-1}.
\end{aligned}$$

Since $|\beta_{m+1}| = \|\beta_{m+1} v_{m+1} e_m^T\| = \|AV_m - V_m T_m\| \leq 2\|A\|$, then we have

$$\|E_m^{SL}(\tau)\| \leq 2\beta\tau\|A\| \frac{2}{1-q} e^{\frac{\tau(\lambda_n-\lambda_1)\left(q+\frac{1}{q}\right)}{4}} q^{m-1}.$$

$\square$

4.2 Inverse Lanczos approximation

In this section, we propose the inverse Lanczos approximation method. The inverse Lanczos approximation is based on the projection on $K_m(A^{-1}, v)$. Applying Lanczos algorithm to A^{-1} and v, we have

$$A^{-1} V_m = V_m T_m + \beta_{m+1} v_{m+1} e_m^T. \tag{4.9}$$

Similarly, a natural closest approximation to $e^{-\tau A} v$ from $K_m(A^{-1}, v)$ is the vector $V_m V_m^T e^{-\tau A} v$, which is

$$V_m V_m^T e^{-\tau A} v = \beta V_m V_m^T e^{-\tau A} V_m e_1 \approx \beta V_m e^{-\tau T_m^{-1}} e_1,$$

where $\beta = \|v\|$, $v = \beta v_1$ and $v_1 = V_m e_1$. We call

$$w_m^{IL} := \beta V_m e^{-\tau T_m^{-1}} e_1, \tag{4.10}$$

the inverse Lanczos approximation to $e^{-\tau A}v$. We define the error by

$$E_m^{IL} = w(\tau) - w_m^{IL}(\tau). \tag{4.11}$$

where $w(\tau) = e^{-\tau A}v$. A difference between the standard Krylov subspace method and the inverse Krylov subspace method is the spectral transformation. The reciprocals of the smallest eigenvalues of A become the largest eigenvalues of A^{-1}. This transformation plays an important role in the numerical approximation to $e^{-\tau A}v$ which is mainly determined by the lower end of the spectrum. In the following, we will discuss a posterior error estimation for this method. And the analysis shows that if the matrix A has very large eigenvalues, yet the smallest eigenvalue is not too small, say around the magnitude of 1, the Lanczos method with inverse is more effective than the standard Lanczos method.

Before we proceed to a posterior error analysis of the inverse Lanczos approximation method, we first analyze the quantity $|e_m^{-1}T_m^{-1}e^{-\tau T_m^{-1}}e_1|$. Similar to the role that the quantity $|e_m^{-1}e^{-\tau T_m}e_1|$ plays in the standard Lanczos approximation method, the behavior of $|e_m^{-1}T_m^{-1}e^{-\tau T_m^{-1}}e_1|$ determines the posterior error of inverse Lanczos approximation method.

4.2.1 Behavior of $|e_m^{-1}T_m^{-1}e^{-\tau T_m^{-1}}e_1|$

Lemma 4.2.1. *Let T_m be symmetric and tridiagonal. Denote $a = \lambda_{\min}(T_m)$ and $b = \lambda_{\max}(T_m)$. For any fixed q such that $\dfrac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} < q < 1$, where $\kappa = \frac{b}{a}$, we have*

$$|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1| < K_0 q^{m-1},$$

where $K_0 = \dfrac{4}{(1-q)\left(\dfrac{a-b}{2}\left(\dfrac{1}{q}+q\right)+a+b\right)}$.

Proof. Define $\psi : \mathbb{C} \to \mathbb{C}$ as

$$\psi(\lambda) = \frac{2\lambda - (a+b)}{b-a}.$$

then $\psi([a,b]) = [-1,1]$. Define

$$B = \psi(T_m) = \frac{2}{b-a}T_m - \frac{a+b}{b-a}I.$$

Then the spectrum of the symmetric matrix B is contained in $[-1,1]$. Let $\chi = \dfrac{1}{q}$. Let ϵ_χ be the ellipse which has $-1, 1$ as its foci and $\alpha = \dfrac{\chi^2+1}{2\chi}$ and $\beta = \dfrac{\chi^2-1}{2\chi}$ as its semi-major axis and semi-minor axis. Let $f(\lambda) = \lambda^{-1}e^{-\tau\lambda^{-1}}$ and $F = fo\psi^{-1}$. Then

$$F(z) = \left(\frac{(b-a)}{2}z + \frac{a+b}{2}\right)^{-1} e^{-\tau\left(\frac{(b-a)}{2}z + \frac{a+b}{2}\right)^{-1}}.$$

Next we find the regularity ellipse of F. Recall that the regularity ellipse of F, as in [31], is the ellipse $\epsilon_{\bar{\chi}}$ where

$$\bar{\chi} = \bar{\chi}(F) = \sup\{\chi : F \text{ is analytic in the interior of } \epsilon_\chi\}.$$

So in this case the regularity ellipse for this F is $\varepsilon_{\bar{\chi}}$

$$\bar{\chi} = \frac{b+a}{b-a} + \sqrt{(\frac{b+a}{b-a})^2 - 1} = \frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1},$$

and $\kappa = \dfrac{b}{a}$ is the spectral condition number of T_m. For $\dfrac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} < q < 1$, i.e., $1 < \chi < \bar{\chi}$, the function F is analytic inside ε_χ and continuous on ε_χ. According to Theorem 2.5.2, we have

$$|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1| \le K q^{m-1}, \quad q = \frac{1}{\chi},$$

where $K = \max\{\dfrac{\chi M(\chi)}{\chi - 1}, \|F(B)\|_2\}$ and $M(\chi) = \max_{z\in\epsilon_\chi}|F(z)|$.

Next, let us look at the bound for $M(\chi)$. Let $z = x + iy \in \varepsilon_\chi$. Set

$$u = \frac{(b-a)x + a + b}{2}, \quad v = \frac{b-a}{2}y.$$

Then

$$|F(z)| = |(u+iv)^{-1}e^{-\tau(u+iv)^{-1}}| = \frac{1}{\sqrt{u^2+v^2}}|e^{-\tau\left((u-iv)/(u^2+v^2)\right)}|$$

and since $\alpha < \dfrac{b+a}{b-a}$, then $u > 0$. So we have

$$|e^{-\tau\left((u-iv)/(u^2+v^2)\right)}| = e^{-\tau\left(\dfrac{u}{u^2+v^2}\right)} < 1.$$

Therefore,

$$|F(z)| < \frac{1}{\sqrt{u^2+v^2}}.$$

The function $\dfrac{1}{\sqrt{u^2+v^2}}$ attains its maximum on the ellipse ε_χ, where $\chi = \alpha + \beta$, at the point $z = -\alpha$ on the real axis, so that

$$M(\chi) = \max_{z\in\varepsilon_\chi} |F(z)| < \max_{z\in\varepsilon_\chi} \frac{1}{\sqrt{((b-a)(-\alpha)+a+b)^2/4}} = \frac{2}{\dfrac{(a-b)(\chi^2+1)}{2\chi}+a+b}$$

Thus,

$$\frac{2\chi M(\chi)}{\chi - 1} < \frac{4}{(1-q)\left(\dfrac{a-b}{2}(\dfrac{1}{q}+q)+a+b\right)} = K_0$$

and

$$\|F(B)\| = \|T_m^{-1}e^{-\tau T_m^{-1}}\| \leq \frac{1}{a}e^{-\tau\dfrac{1}{b}} \leq \frac{1}{a}$$

Notice that

$$\begin{aligned} K_0 &= \frac{4}{(1-q)\left(\dfrac{a-b}{2}\left(\dfrac{1}{q}+q\right)+a+b\right)} \\ &\geq \frac{4}{(1-q)\left(\dfrac{a-b}{2}\cdot 2+a+b\right)} \\ &= \frac{2}{a(1-q)} \geq \frac{1}{a} \end{aligned}$$

Therefore,

$$|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1| < \max\{K_0, \frac{1}{a}\} q^{m-1} = K_0 q^{m-1}.$$

$\square$

4.2.2 Posterior error analysis

In this section, we present error bounds for approximating $w = e^{-\tau A}v$ by using the inverse Lanczos approximation. Let T_m be the projection of A onto the Krylov subspace $K_m(A^{-1}, v)$. The error E_m^{IL} is defined as in (4.11).

Theorem 4.2.2. *The posterior error of the inverse Lanczos approximation to the matrix exponential on $K_m(A^{-1}, v)$ satisfies*

$$\|E_m^{IL}(\tau)\| \leq \beta|\beta_{m+1}| \max_{0\leq t\leq \tau} |e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1| \|I - e^{-\tau A}\|,$$

where $\beta = \|v\|$, T_m and β_{m+1} are generated from the inverse Lanczos process (4.9).

Proof. We rewrite (4.9) as

$$V_m T_m^{-1} = AV_m + \beta_{m+1} A v_{m+1} e_m^T T_m^{-1},$$

then

$$\begin{aligned} w_m^{IL}(t)' &= -\beta V_m T_m^{-1} e^{-tT_m^{-1}} e_1 \\ &= -\beta(AV_m + \beta_{m+1} A v_{m+1} e_m^T T_m^{-1}) e^{-tT_m^{-1}} e_1 \\ &= -\beta A V_m e^{-tT_m^{-1}} e_1 - \beta\beta_{m+1} A v_{m+1} e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1 \\ &= -A w_m^{IL}(t) - \beta\beta_{m+1}(e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1) A v_{m+1} \end{aligned}$$

Since $E_m^{IL}(t) = w(t) - w_m^{IL}(t)$, $E_m^{IL}(0) = 0$, then

$$E_m^{IL}(t)' = -AE_m^{IL}(t) + \beta\beta_{m+1}(e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1) A v_{m+1}$$

Solve the above ODE on E_m^{IL}, the posterior error is:

$$\begin{aligned} E_m^{IL}(\tau) &= \int_0^\tau e^{(t-\tau)A} \left(\beta\beta_{m+1}(e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1) A v_{m+1}\right) dt \\ &= \beta\beta_{m+1} \int_0^\tau (e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1) e^{(t-\tau)A} A v_{m+1} dt \end{aligned}$$

$$\begin{aligned} \|E_m^{IL}(\tau)\| &\leq \beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1| \|\int_0^\tau e^{(t-\tau)A} A dt\| \|v_{m+1}\| \\ &\leq \beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1| \|I - e^{-\tau A}\| \end{aligned}$$

□

Remark 4. From the bound of $|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1|$, we know that increasing the dimension of the Krylov subspace will reduce the value $|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1|$, thereby reducing the approximation error. For the term $\|I - e^{-\tau A}\|$, the smaller τ is, the closer the matrix $e^{-\tau A}$ to identity matrix, and thereby the smaller the value $\|I - e^{-\tau A}\|$. But the speed of the decay of $\|I - e^{-\tau A}\|$ is fairly slow at the early stage unless the eigenvalues of A are all very small. So whether this term will play a role in reducing the error as τ decreases depends on the magnitude of the eigenvalues of A. The upper bound of $\|I - e^{-\tau A}\|$ is 1.

Remark 5. We also notice that, if A has very small eigenvalues, then the reciprocal of these eigenvalues could be huge, then $|\beta_{m+1}|$ is large. On the other hand, we know that the magnitude of $|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1|$ largely depends on the condition number of T_m. If the smallest eigenvalue of T_m is away from zero, $|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1|$ decays rapidly as m increases. But if the smallest eigenvalues of the original matrix A are small, in this case, the smallest eigenvalues of T_m would be very small, and therefore the expression $|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1|$ would be large. But if the small eigenvalues of A are bounded away from zero , both $|\beta_{m+1}|$ and $|e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1|$ are bounded, and therefore give a good approximation to the matrix exponential and vector product. For cases when A has huge largest eigenvalues and moderate smallest eigenvalues, the Lanczos approximation onto the Krylov subspace $K_m(A^{-1}, v)$ will provide better approximation to $e^{-\tau A}v$ than Lanczos approximation onto Krylov subspace $K_m(A, v)$.

Corollary 4.2.3. *Denote $\lambda_1 = \lambda_{\min}(A)$ and $\lambda_n = \lambda_{\max}(A)$. For any $\dfrac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} < q < 1$, $\kappa = \dfrac{\lambda_n}{\lambda_1}$, the error of the inverse Lanczos approximation to the matrix exponential satisfies*

$$\|E_m^{IL}(\tau)\| \leq \beta K_0 q^{m-1},$$

where $K_0 = \dfrac{8\lambda_n}{(1-q)\left(\dfrac{\lambda_1 - \lambda_n}{2}\left(\dfrac{1}{q} + q\right) + \lambda_1 + \lambda_n\right)}$.

Proof. Let T_m be the projection of A onto the Krylov subspace $K_m(A^{-1}, v)$. Noting that $\lambda_{\min}(T_m) \geq \dfrac{1}{\lambda_{\max}(A)}$ and $\lambda_{\max}(T_m) \leq \dfrac{1}{\lambda_{\min}(A)}$, for $\dfrac{\sqrt{\lambda_{\max}(T_m)/\lambda_{\min}(T_m)} - 1}{\sqrt{\lambda_{\max}(T_m)/\lambda_{\min}(T_m)} + 1} < \dfrac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} < q < 1$, by Lemma 4.2.1 and Theorem 4.2.2, we have

$$\begin{aligned}
\|E_m^{IL}(\tau)\| &\leq \beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1| \|I - e^{-\tau A}\| \\
&\leq \beta|\beta_{m+1}| \frac{4q^{m-1}}{(1-q)\left(\frac{\lambda_{\min}(T_m) - \lambda_{\max}(T_m)}{2}\left(\frac{1}{q} + q\right) + \lambda_{\min}(T_m) + \lambda_{\max}(T_m)\right)} \\
&= \beta|\beta_{m+1}| \frac{4q^{m-1}}{(1-q)\left(\lambda_{\min}(T_m)\left[\frac{1/q+q}{2} + 1\right] - \lambda_{\max}(T_m)\left[\frac{1/q+q}{2} - 1\right]\right)} \\
&\leq \beta|\beta_{m+1}|\lambda_1\lambda_n \frac{4q^{m-1}}{(1-q)\left(\lambda_{\min}(A)\left[\frac{1/q+q}{2} + 1\right] - \lambda_{\max}(A)\left[\frac{1/q+q}{2} - 1\right]\right)} \\
&\leq \beta|\beta_{m+1}|\lambda_1\lambda_n \frac{4q^{m-1}}{(1-q)\left(\lambda_1\left[\frac{1/q+q}{2} + 1\right] - \lambda_n\left[\frac{1/q+q}{2} - 1\right]\right)} \\
&\leq 2\beta\lambda_n \frac{4q^{m-1}}{(1-q)\left(\lambda_1\left[\frac{1/q+q}{2} + 1\right] - \lambda_n\left[\frac{1/q+q}{2} - 1\right]\right)} \\
&= \beta K_0 q^{m-1},
\end{aligned}$$

where $|\beta_{m+1}| \leq \|A^{-1}\| + \|T_m\| \leq 2\|A^{-1}\| = \dfrac{2}{\lambda_1}$ and $\|I - e^{-\tau A}\| \leq 1$. □

4.3 Shift-and-invert Lanczos approximation

As we can see from the analysis of the inverse Lanczos method, the method outperforms the standard Lanczos method when the matrix A has huge largest eigenvalues and moderate smallest eigenvalues. But when the matrix A has very small eigenvalues, the inverse Lanczos method may perform worse. The shift-and-invert Lanczos approximation with appropriate shift might perform better in this case. In [16], it was proposed to use Krylov subspace generated by $(I + \sigma A)^{-1}$ to approximate $e^{-\tau A}v$. A sophisticated technique is introduced to minimize error with respect to σ. Here we propose to use the shift-and-inverse approximation that is similar to the inverse

Lanczos approximation. It is based on the Lanczos method for $(A+\sigma I)^{-1}$ and v,

$$(A+\sigma I)^{-1}V_m = V_mT_m + \beta_{m+1}v_{m+1}e_m^T, \quad \text{for } \sigma > 0. \tag{4.12}$$

The closest approximation to $e^{-\tau A}v$ from $K_m((A+\sigma I)^{-1}, v)$ is the vector $V_mV_m^Te^{-\tau A}v$, which is

$$V_mV_m^Te^{-\tau A}v = \beta V_mV_m^Te^{-\tau A}V_me_1 \approx \beta V_m e^{-\tau(T_m^{-1}-\sigma I)}e_1,$$

where $\beta = \|v\|$, $v = \beta v_1$ and $v_1 = V_me_1$.

The approximation to $e^{-\tau A}v$ defined by

$$w_m^{SIL} := \beta V_m e^{-\tau(T_m^{-1}-\sigma I)}e_1. \tag{4.13}$$

is called the shift-and-invert Lanczos approximation The superscript SIL represents its correspondence to the shift-and-invert Lanczos method. We also define the error

$$E_m^{SIL}(\tau) = w(\tau) - w_m^{SIL}(\tau), \tag{4.14}$$

where $w(\tau) = e^{-\tau A}v$. One would think that the smaller the largest eigenvalue after the shift-and-invert transformation, the better the approximation error, since this will induce smaller $|\beta_{m+1}|$. But this will induce larger $|e_m^TT_m^{-1}e^{-\tau(T_m^{-1}-\sigma I)}e_1|$ which shows up in the bound of the posterior error of the shift-and-invert Lanczos approximation. So a proper shift is crucial for this method to provide better approximation to the solution $e^{-\tau A}v$ which can have moderate $|\beta_{m+1}|$ and a smaller $|e_m^TT_m^{-1}e^{-\tau(T_m^{-1}-\sigma I)}e_1|$. In the following we will first analyze the behavior of the quantity $|e_m^TT_m^{-1}e^{-\tau(T_m^{-1}-\sigma I)}e_1|$, and then presents the posterior analysis for the Lanczos approximation on $K_m((A+\sigma I)^{-1}, v)$.

4.3.1 Behavior of $|e_m^TT_m^{-1}e^{-\tau(T_m^{-1}-\sigma I)}e_1|$

Lemma 4.3.1. *Let T_m be symmetric and tridiagonal. Denote $a = \lambda_{\min}(T_m)$ and $b = \lambda_{\max}(T_m)$. For any fixed q such that $\dfrac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} < q < 1$, $\kappa = \dfrac{b}{a}$, then we have*

$$|e_m^TT_m^{-1}e^{-\tau(T_m^{-1}-\sigma I)}e_1| < K_0q^{m-1},$$

where $K_0 = \dfrac{4e^{\tau\sigma}}{(1-q)\left(\dfrac{a-b}{2}\left(\dfrac{1}{q}+q\right)+a+b\right)}$.

We omit the proof since it is similar to the one in the inverse Lanczos approximation. In the following, we look at the posterior error bound for the shift-and-invert Lanczos method.

4.3.2 Posterior error analysis

We consider the approximation of $w = e^{-\tau A}v$ by using the shift-and-invert Lanczos approximation. Let the approximate solution $w_m^{SIL}(t)$ be defined as in (4.13). T_m is the projection of A onto the Krylov subspace $K_m((A+\sigma I)^{-1}, v)$. The error E_m^{SIL} is defined as in (4.14). Next we present the posterior error analysis.

Theorem 4.3.2. *The posterior error of the Lanczos approximation to the matrix exponential on* $K_m((A+\sigma I)^{-1}, v))$ *is*

$$\|E_m^{SIL}(\tau)\| \leq \beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1|(\|I - e^{-\tau A}\| + \tau|\sigma|),$$

where $\beta = \|v\|$, T_m *and* β_{m+1} *are generated from the Lanczos process (4.12).*

Proof. Since

$$(A+\sigma I)^{-1}V_m = V_m T_m + \beta_{m+1} v_{m+1} e_m^T,$$

we have

$$V_m(T_m^{-1} - \sigma I) = AV_m + \beta_{m+1}(A+\sigma I)v_{m+1}e_m^T T_m^{-1}.$$

Then

$$\begin{aligned} w_m^{SIL}(t)' &= -\beta(AV_m + \beta_{m+1}(A+\sigma I)v_{m+1}e_m^T T_m^{-1})e^{-t(T_m^{-1}-\sigma I)}e_1 \\ &= -\beta AV_m e^{-t(T_m^{-1}-\sigma I)}e_1 - \beta\beta_{m+1}(A+\sigma I)v_{m+1}e_m^T T_m^{-1}e^{-t(T_m^{-1}-\sigma I)}e_1 \\ &= -Aw_m^{SIL}(t) - \beta\beta_{m+1}(e_m^T T_m^{-1}e^{-t(T_m^{-1}-\sigma I)}e_1)(A+\sigma I)v_{m+1} \end{aligned}$$

Since $E_m^{SIL}(t) = w(t) - w_m^{SIL}(t)$, we have

$$E_m^{SIL}(t)' = -AE_m^{SIL}(t) + \beta\beta_{m+1}(e_m^T T_m^{-1}e^{-t(T_m^{-1}-\sigma I)}e_1)(A+\sigma I)v_{m+1}$$

Solving the above ODE with $E_m^{SIL}(0) = 0$, the posterior error is:

$$\begin{aligned} E_m^{SIL}(\tau) &= \textstyle\int_0^\tau e^{(t-\tau)A}(\beta\beta_{m+1}(e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1)(A+\sigma I)v_{m+1})dt \\ &= \beta\beta_{m+1} \textstyle\int_0^\tau (e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1) e^{(t-\tau)A}(A+\sigma I)v_{m+1}dt \end{aligned}$$

Taking norm on both sides of the above expression, we have

$$\begin{aligned} &\|E_m^{SIL}(\tau)\| \\ \leq\ & \beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1| \| \int_0^\tau e^{(t-\tau)A}(A+\sigma I)dt\| \|v_{m+1}\| \\ \leq\ & \beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1|(\|I - e^{-\tau A}\| + \tau|\sigma|) \end{aligned}$$

□

Notice from the bound of this theorem, similar argument as in the inverse Lanczos approximation can be made for the term $\|I - e^{-\tau A}\|$ as τ decreases. The modulus of the $(m, 1)$ entry of matrix function $T_m^{-1} e^{-t(T_m^{-1}-\sigma I)}$, i.e. $|e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1|$ decays away from the diagonal as the dimension increases. Choosing small σ will make $|\beta_{m+1}|$ small, but the term $\max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1|$ will be large due to the small smallest eigenvalue of T_m. This is similar to the analysis of simply inverting the matrix. So the best σ should not be too large and not be too small.

Corollary 4.3.3. *Denote $\lambda_1 = \lambda_{\min}(A)$ and $\lambda_n = \lambda_{\max}(A)$. For any $\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} < q < 1$, $\kappa = \frac{\lambda_n + \sigma}{\lambda_1 + \sigma}$, the error of the shift-and-invert Lanczos approximation to the matrix exponential $e^{-\tau A}v$ on Krylov subspace $K_m((A + \sigma I)^{-1}, v)$ satisfies*

$$\|E_m^{SIL}(\tau)\| \leq \beta K_0 q^{m-1},$$

where $K_0 = \dfrac{8(\lambda_n + \sigma)e^{\tau\sigma}}{(1-q)\left(\frac{\lambda_1 - \lambda_n}{2}\left(\frac{1}{q} + q\right) + \lambda_1 + \lambda_n + 2\sigma\right)}(1 + \tau|\sigma|).$

In the next section, we compare all the three Lanczos approximation methods by using examples and explain the benefit of choosing the right Lanczos method, which in the end allows us to take larger time step in computing solutions in problems like (4.5).

4.4 Numerical Examples

In this section, we carry out numerical tests of Lanzcos based approximation methods. We compare these methods for computing $e^{-\tau A}v$ for various values of τ and dimension m. We define $\varphi(T_m,t) = e_m^T e^{-tT_m} e_1$ for the standard Lanczos approximation, $\varphi(T_m,t) = e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1$ for the inverse Lanczos approximation and $\varphi(T_m,t) = e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1$ for the shift-and-invert Lanczos approximation. As shown in the posterior error bound, we shall examine the quantity $\beta(T_m) = \max\limits_{0\le t\le\tau} |\varphi(T_m,t)|$ for each Lanczos based method. For the purpose of numerical execution, we use $\alpha(T_m)$ to approximate $\beta(T_m)$, where $\alpha(T_m) = \max\limits_{t} |\varphi(T_m,t)|, t = \dfrac{i\tau}{\text{pnum}}, 1 \le i \le \text{ pnum }$, i is integer and *pnum* is the number of equal partition between 0 and τ.

Example 5. *In this example, we compare the standard Lanczos method and the inverse Lanczos method. The matrix is plat362.mtx from Matrix Market [2]. The size of the matrix is* 362 *by* 362. *The largest eigenvalue is approximately* $7.74\text{e}-001$, *the smallest eigenvalue is approximately* $3.55\text{e}-012$. *The number of equal partition we use to calculate* $|\alpha(T_m)|$ *is* $pnum = 1000$. *We use the standard Lanczos method and the inverse Lanczos method to compute* $w = e^{-\tau A}v$ *where* v *is a random vector with* $\|v\| = 1$. *We compare the error of the approximation* $\|E_m\| = \|w - w_m\|$ *for each method where* w *is computed by* $\exp\{-\tau A\}v$ *of MATLAB. In Table 4.1, we list various components of the posterior bounds and error. In Figure 4.1, we give the error history of the standard Lanczos approximation and the inverse Lanczos approximation with* $\tau = 0.1$ *and* m *increasing from* $m = 1$ *to* $m = 20$. *The dash-dotted line in the figure corresponds to the error history of the inverse Lanczos approximation and the solid line corresponds to the error history of the standard Lanczos approximation.*

From Table 4.1, we see that the standard Lanczos algorithm on this matrix performs much better than the inverse Lanczos algorithm. To achieve an error of $4.95e-016$, the standard Lanczos algorithm only needs time step $\tau = 0.1, m = 10$,

Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	$\lVert E_m\rVert$	CPU (secs)
SLanczos	10	0.1	1.63e-001	5.84e-022	4.95e-016	4.04e+002
SLanczos	10	0.001	1.63e-001	1.78e-039	3.55e-016	4.80e+001
ILanczos	10	0.1	5.69e+007	6.69e-007	1.36e-002	3.13e-001
ILanczos	10	0.001	5.69e+007	6.69e-007	1.39e-004	3.28e-001

Table 4.1: A comparison between SL method and IL method, SL performs better

while for the inverse Lanczos method, even when the time step is $\tau = 0.001$, the error only reaches to $1.39e - 004$. The reason is that the largest eigenvalue of A is of order 1 while the smallest eigenvalue is very small, resulting β_{m+1} being small for the standard Lanczos method, but large for the inverse Lanczos method. Also, we notice that the posterior bound $\tau\beta_{m+1}|\alpha(T_m)|$ of the standard Lanczos method is way smaller than the error $\|E_m\|$, this is due to the roundoff error of the machine which results in $\|E_m\|$ as order of 10^{-16} instead of a quantity very close to zero.

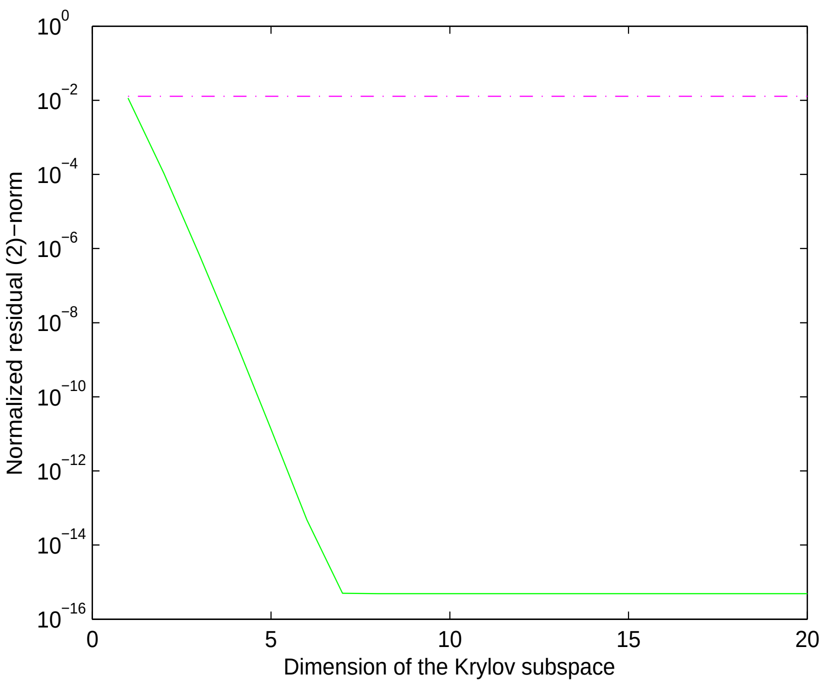

Figure 4.1: Error convergence history of Example 5

From the Figure 4.1, we see that, for fixed $\tau = 0.1$, the error reduces quickly to order of 10^{-16} for the standard Lanczos method as m increases, but the error of the inverse Lanczos approximation stagnates at order of 10^{-2} even with increasing dimension m.

Next we show an example whose largest eigenvalue is large while small eigenvalue is not too small in which applying the inverse Lanczos method performs better than the standard Lanczos method.

Example 6. *The matrix is lund_a.mtx from Matrix Market [2]. The size is* 147 *by* 147*. The largest eigenvalue is approximately* $2.24e + 008$*, the smallest eigenvalue is approximately* $8.00e{+}001$*. The number of equal partition we use to calculate* $|\alpha(T_m)|$ *is* $pnum = 1000$*. We plot the value* $|\varphi(T_m, t)|, t = \dfrac{i\tau}{pnum}, 1 \leq i \leq 100$ *to give a picture of how* $|\varphi(T_m, t)|$ *changes with respect to t for different Lanczos based methods and different* τ*, see Figure 4.3. We use the standard Lanczos approximation method and the inverse Lanczos method to compute* $w = e^{-\tau A}v$ *where* v *is a random vector with* $\|v\| = 1$*. The quantity* w_m *is the approximation solution as introduced in the beginning of each subsection. We compare the error* $\|E_m\| = \|w - w_m\|$ *of the approximation of each Lanczos method where* w *is computed by* $\exp\{-\tau A\}v$ *of MATLAB. Again, we list various components of the posterior bounds and error in Table 4.2. In Figure 4.2, we give the error of the standard Lanczos approximation and the inverse Lanczos approximation with* $\tau = 0.1$ *and* m *increasing from* $m = 1$ *to* $m = 20$*. The solid line in the figure corresponds to the error history of the standard Lanczos method as* m *increases from* $m = 1$ *to* $m = 20$*, the dash-dotted line corresponds to the error history of the inverse Lanczos approximation as* m *increases from* $m = 1$ *to* $m = 20$*.*

In this case, Lanczos algorithm on the inverse matrix performs better than the standard Lanczos approximation method for both $m = 10$ and $m = 20$ with τ not too small because β_{m+1} is large for the standard Lanczos approximation method, while it is small for the inverse Lanczos approximation method. On the other hand, we see from the table that reducing τ is not beneficial for the inverse Lanczos method at all.

Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	$\lVert E_m\rVert$	CPU (secs)
SLanczos	10	0.1	6.28e+007	2.32e-005	1.69e-004	4.37e-001
SLanczos	10	0.001	6.28e+007	2.78e-003	4.66e-001	3.59e-001
ILanczos	10	0.1	2.24e-005	9.33e+000	1.14e-013	4.37e-001
ILanczos	10	0.001	2.24e-005	4.18e+003	5.15e-008	4.06e-001
SLanczos	20	0.1	4.10e+007	1.79e-003	1.47e-004	7.65e-001
SLanczos	20	0.001	4.10e+007	4.24e-003	4.07e-001	5.79e-001
ILanczos	20	0.1	4.76e-006	3.54e-004	1.59e-014	8.75e-001
ILanczos	20	0.001	4.76e-006	4.91e+002	6.69e-012	8.75e-001
SLanczos	20	0.000001	4.10e+007	4.20e-004	4.28e-004	5.00e-001
SLanczos	20	0.00000001	4.10e+007	8.90e-024	4.80e-016	4.53e-001
ILanczos	20	0.000001	4.76e-006	1.78e+005	1.34e-003	8.13e-001
ILanczos	20	0.00000001	4.76e-006	2.03e+005	1.26e-001	7.50e-001

Table 4.2: A comparison between SL method and IL method, IL performs better

For the standard Lanczos method, the error deteriorates as τ decreases and after τ decreases to a certain point, the approximation starts to improve again. In this case with $\tau = 0.00000001$, we get a better approximation with error $\|E_m\| = 4.80e-016$. So for the standard Lanczos method, after reducing τ to a certain point, the error does improve. We also give the errors for different dimensions and fixed $\tau = 0.1$. As one can see from Figure 4.2, increasing the dimension does not decrease the error for the standard Lanczos method, but does reduce the error for the inverse Lanczos method.

We next consider an example where the matrix has extremely large and small eigenvalues. This is a situation that both of the standard Lanczos approximation method and the inverse Lanczos approximation method do not perform well.

Example 7. *The matrix is bcsstm27.mtx from Matrix Market [2]. In order to make it symmetric positive definite, we multiply A by constant* 100*, and then let A=A*A. The size is* 1224 *by* 1224*. The largest eigenvalue is approximately* $1.54e+011$*, the smallest eigenvalue is approximately* $9.91e-007$*. The number of equal partition we use to calculate* $|\alpha(T_m)|$ *is* $pnum = 200000$*. We use both the standard Lanczos approximation method and the inverse Lanczos approximation method to compare the*

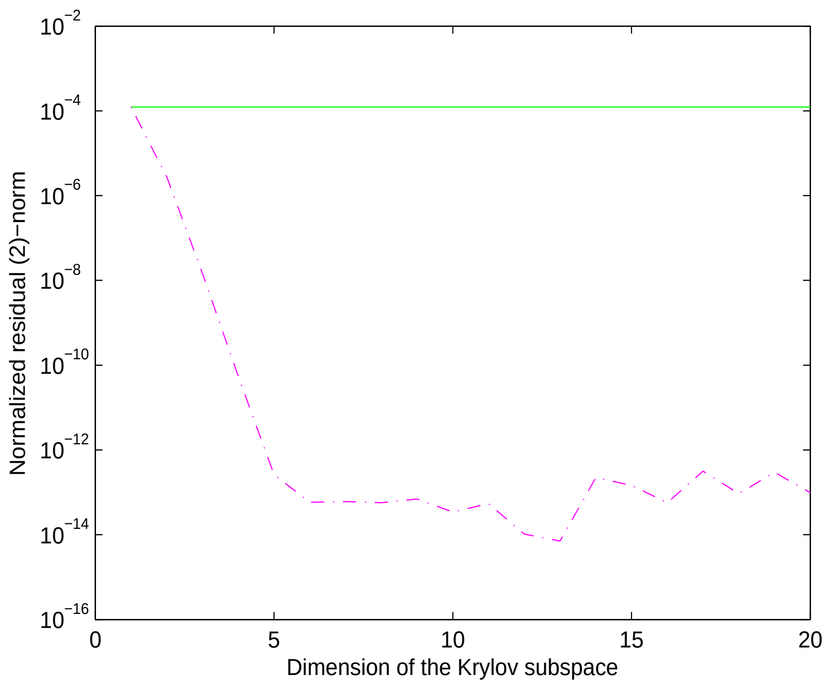

Figure 4.2: Error convergence history of Example 6

error of the approximation and we list various components of the posterior bound of each of these two Lanczos methods, the error and the CPU time in Table 4.3. In Table 4.4, we list various components of the posterior bound of the shift-and-invert Lanczos method, its error and the CPU time. The shift we used is $\sigma = 1$. And we plot the error againt m for all three Lanczos methods in Figure 4.4.

Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	$\lVert E_m\rVert$	CPU (secs)
SLanczos	10	0.1	3.79e+010	1.27e-008	6.12e-001	4.31e+002
SLanczos	10	0.001	3.79e+010	9.96e-003	7.49e-001	8.25e+001
SLanczos	10	0.00000001	3.79e+010	1.10e-002	2.48e-001	6.42e+001
ILanczos	10	0.1	4.26e+002	2.91e-008	7.83e-001	5.75e+001
ILanczos	10	0.001	4.26e+002	2.91e-008	6.28e-001	5.78e+001
ILanczos	10	0.00000001	4.26e+002	2.91e-008	3.54e-001	5.97e+001

Table 4.3: A comparison between SL method and IL method, both perform worse

In this example, we see that both the standard Lanczos approximation method and the inverse approximation method do not provide good approximation. For the

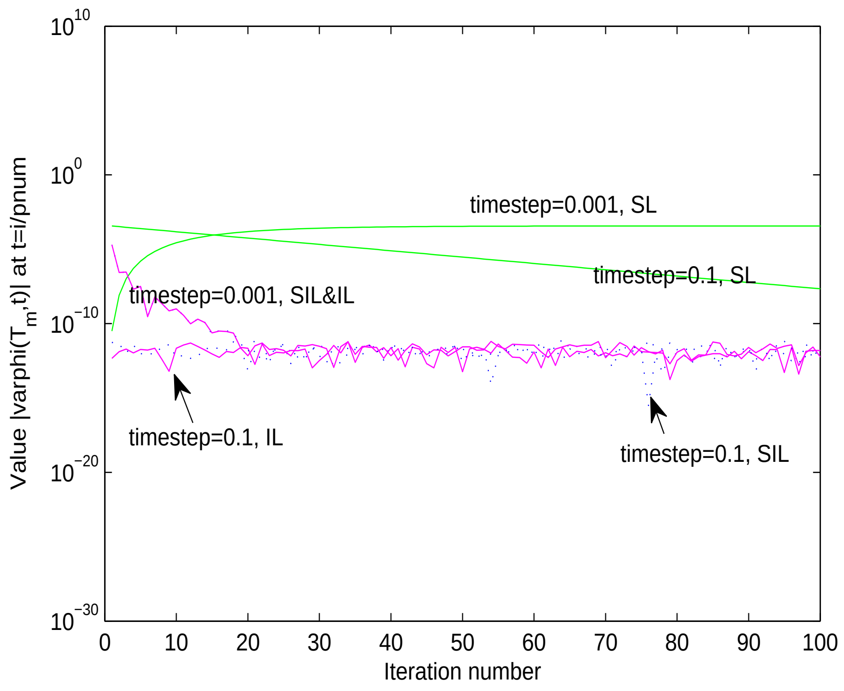

Figure 4.3: Function value $|\varphi(T_m, t)|$ as t changes

standard Lanczos approximation, no matter $\tau = 0.1$, $\tau = 0.001$ and $\tau = 0.00000001$, The error is of order 10^{-1}. This is because the eigenvalues of A are huge, the standard Lanczos method fails to converge even with $\tau = 0.00000001$. For the inverse Lanczos approximation, the smallest eigenvalue of A is very small. Eigenvalues of T_m^{-1} approaches the eigenvalues of A. Thus $e^{-\tau T_m^{-1}}$ is close to identity matrix and then $w_m = \beta V_m e^{-\tau T_m^{-1}} e_1$ is close to v. Therefore, the error of the inverse Lanczos approximation in this case is close to $\|e^{-\tau A}v - v\|$ which is also of order 10^{-1}. In this case, the shift-and-invert Lanczos method achieves a little better approximation due to the characteristics of the spectrum of the transformed matrix. See Table 4.4. We also see in Table 4.4 that small τ will produce larger error for fixed dimension m. And the error gets smaller as the dimension increases from $m = 10$ to $m = 50$. In Figure 4.4, we give the error of each Lanczos approximation against the dimension m. Dimension m changes from 1 to 100. As we can see that the error of the shift-and-invert Lanczos approximation(the lowest line in the graph) decreases as the dimension m increases, and starts to stagnate when the error reaches to order of $1e-006$ to $1e-007$. The standard Lanczos approximation and the inverse Lanczos approximation both fail to give error under the order of 10^{-1}.

Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	$\lVert E_m\rVert$	CPU (secs)
SILanczos	10	0.1	2.17e-001	4.28e+001	2.69e-003	7.77e+001
SILanczos	10	0.01	2.17e-001	4.28e+001	2.49e-002	7.57e+001
SILanczos	10	0.001	2.17e-001	4.28e+001	1.33e-001	7.86e+001
SILanczos	50	0.1	2.43e-001	8.09e-001	3.76e-006	7.33e+000
SILanczos	50	0.01	2.43e-001	2.50e+001	1.01e-003	7.12e+000
SILanczos	50	0.001	2.43e-001	4.34e-001	1.65e-002	6.84e+000

Table 4.4: A case study of shift-and-invert Lanczos method with different τ and m

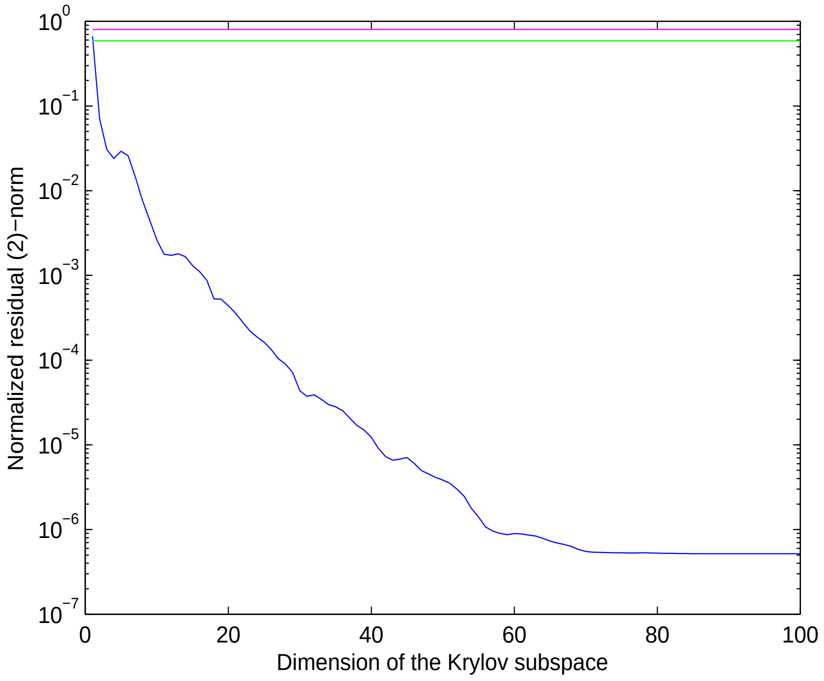

Figure 4.4: Error convergence history of Example 7

Example 8. *The matrix is mhd3200b.mtx from Matrix Market collection [2]. The size is* 3200 *by* 3200*. The largest eigenvalue is approximately* $2.19e+000$ *, the smallest eigenvalue is approximately* $1.37e-013$*. The number of equal partition to calculate*

$|\alpha(T_m)|$ *is* 1000. *We apply all three Lanczos approximation methods to calculate* $e^{-\tau A}v$*, where* v *is a random vector with* $\|v\| = 1$*. In the shift-and-invert transformation, we use shift* $\sigma = 1$*. In Figure 4.5, we plot the error of all three Lanczos approximation methods against* m*. The dotted line represents that the timing step is* 0.001*, the solid line represents that the timing step is* 0.1*. The solid line represents the error history of the inverse Lanczos approximation method, the bottom line represents the standard Lanczos approximation method, the line in between represents the shift-and-invert Lanczos approximation method.* m *changes from* 1 *to* 20*. In Table 4.5, we list various components of the posterior bounds, error and CPU time for all the three Lanczos approximation methods with fixed* $\tau = 0.1$ *and fixed dimension* $m = 50$*.*

Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	$\lVert E_m\rVert$	CPU (secs)
SLanczos	50	0.1	2.59e-002	2.55e-140	3.89e-016	1.37e+003
ILanczos	50	0.1	1.54e+009	8.60e-008	1.07e-002	2.18e+001
SILanczos	50	0.1	1.86e-002	8.60e-008	4.23e-016	3.55e+000

Table 4.5: A comparison among SL, IL and SIL methods, SL and SIL performs better

Table 4.5 once again showed the relation between the error and the components of the posterior bounds for each Lanczos approximation. Among all three Lanczos methods, the standard Lanczos method and the shift-and-invert Lanczos method both converge to order of 10^{-16}. The inverse Lanczos method only converges to order of 10^{-2} in this case because the smallest eigenvalue of the matrix is close to zero which results in huge β_{m+1} for the inverse Lanczos approximation method. For the standard Lanczos method, the quantity $|\alpha(T_m)|$ is very small which leads to a smaller posterior error bound than the real error $\|E_m\| = 3.89e{-}016$. This is because that the quantity $w_m = \beta V_m e^{-\tau T_m} e_1$ is zero matrix in this case. Since the largest eigenvalues of A are huge, so are the eigenvalues of T_m. Therefore, $e^{-\tau T_m}$ is approximately a zero matrix. Therefore, the error is $\|E_m\| = \|w\| = \|e^{-\tau A}v\|$ itself in this case which

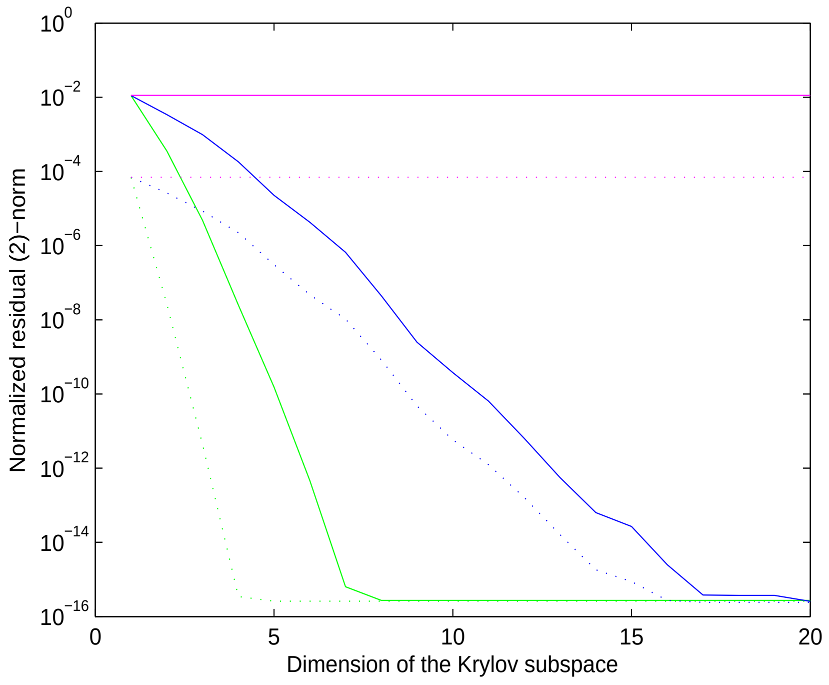

Figure 4.5: Error convergence history of Example 8

is close to $3.89e - 016$. In Figure 4.5, we see that, between the standard Lanczos method and the shift-and-invert Lanczos method, the shift-and-invert method needs larger number of Lanczos iterations to reduce to the same error precision. When timing step τ becomes smaller, the number of Lanczos iterations decreases for both the standard Lanczos method and the shift-and-invert Lanczos method. Both the standard Lanczos method and the shift-and-invert method start to stagnate after the error precision reaches to certain accuracy.

Example 9. *The square matrix is discretized from a differential operator and is of* *form*

$$-\frac{1}{(n+1)^2}\begin{pmatrix} -2 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & & -2 & 1 \\ & & & & 1 & -2 \end{pmatrix}$$

where n is the size of the matrix. The size we used in this case is $n = 1000$. The matrix has very large high eigenvalues, but the smallest eigenvalue is approximately 9.87e+000*. The number of equal partition to calculate $|\alpha(T_m)|$ is $pnum = 5000$. We apply all three Lanczos approximation methods to see the error of the approximation. In the shift-and-invert transformation method, we use shift $\sigma = 1$. The '+' line represents that the timing step is* 0.001*, the solid line represents that the timing step is* 0.1*. See Table 4.6 and Figure 4.6.*

Algorithm	m	τ	β_{m+1}	$\|\alpha(T_m)\|$	$\|E_m\|$	CPU (secs)
SLanczos	50	0.1	9.77e+005	1.09e-003	2.84e-001	2.84e-001
ILanczos	50	0.1	3.06e-005	3.06e+000	2.23e-011	3.27e+001
SILanczos	50	0.1	3.05e-005	3.06e+000	3.69e-011	3.29e+001
SLanczos	50	0.001	9.77e+005	1.09e-003	5.83e-002	1.59e+001
ILanczos	50	0.001	3.06e-005	1.05e+003	2.89e-013	2.99e+001
SILanczos	50	0.001	3.05e-005	1.04e+003	3.93e-013	3.03e+001

Table 4.6: A comparison among SL, IL and SIL methods, IL and SIL performs better

In this case, the standard Lanczos method does not converge. The inverse Lanczos method and the shift-and-invert Lanczos method behave similarly. To reach to the same error precision, the number of Lanczos iterations increases when the timing step τ becomes smaller for both the inverse Lanczos method and the shift-and-invert method. While for the standard Lanczos method, reducing the timing step τ does improve the error precision a little.

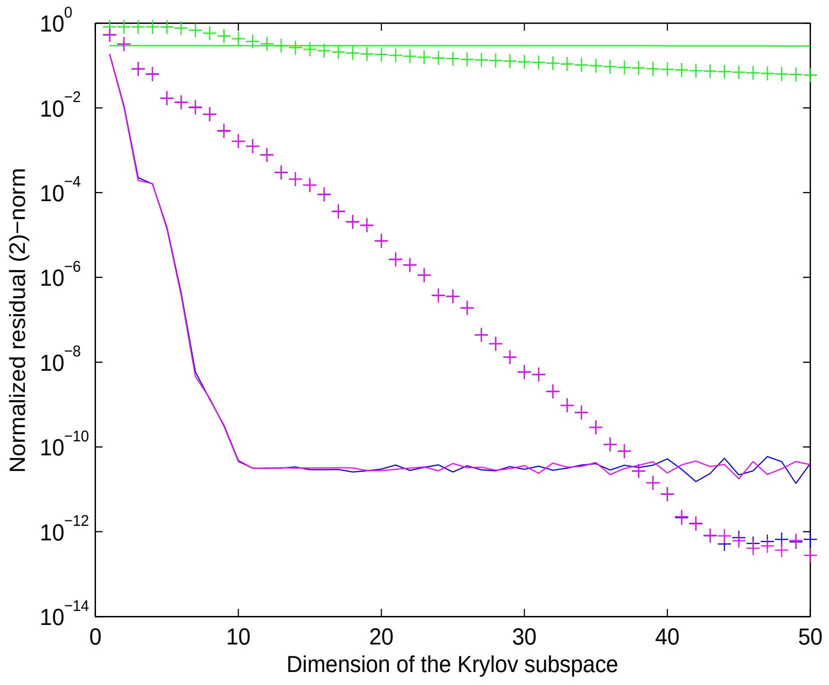

Figure 4.6: Error convergence history of Example 9

Observe from the above two examples, we see that the number of Lanczos iteration increases for Example 9 and decreases for Example 8 as the timing step becomes smaller. This is due to the overall performance of the quantities :

$$|e_m^T e^{-\tau T_m} e_1|, |e_m^T T_m^{-1} e^{-\tau T_m^{-1}} e_1|, |e_m^T T_m^{-1} e^{-\tau (T_m^{-1} - \sigma I)} e_1|,$$

$$\|I - e^{-\tau A}\|, \text{ and } \tau$$

in the posterior error bounds. We give two separate, detailed table for each example with pnum $= 5000$.

We see from Table 4.7, as τ decrease, β_{m+1} does not change for each of the Lanczos methods as β_{m+1} of the Lanczos process depends on A and v but independent of τ. However $\alpha(T_m)$ behaves differently. For the standard Lanczos method, $\alpha(T_m)$ decreases as τ decreases; for the inverse Lanczos method and the shift-and-invert Lanczos method, it keeps unchanged as τ decreases. The terms $\|I - e^{-\tau A}\|$ and $\|I - e^{-\tau A}\| + \tau|\sigma|$ both decrease but at different rate. For the standard Lanczos

Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	τ
SLanczos	10	0.1	4.29e-001	3.36e-021	0.1
SLanczos	10	0.01	4.29e-001	3.54e-030	0.01
SLanczos	10	0.001	4.29e-001	1.04e-038	0.001
Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	$\lVert I - e^{-\tau A}\rVert$
ILanczos	10	0.1	8.40e+009	1.17e-009	1.97e-001
ILanczos	10	0.01	8.40e+009	1.17e-009	2.17e-002
ILanczos	10	0.001	8.40e+009	1.17e-009	2.2e-003
Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	$\lVert I - e^{-\tau A}\rVert + \tau\lvert\sigma\rvert$
SILanczos	10	0.1	1.10e-001	5.19e-008	2.97e-001
SILanczos	10	0.01	1.10e-001	5.19e-008	3.17e-002
SILanczos	10	0.001	1.10e-001	5.19e-008	3.2e-003

Table 4.7: Analysis of the number of the Lanczos iteration for Example 8

Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	τ
SLanczos	10	0.1	1.04e+006	1.12e-002	0.1
SLanczos	10	0.01	1.04e+006	1.12e-002	0.01
SLanczos	10	0.001	1.04e+006	1.12e-002	0.001
Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert$	$\lVert I - e^{-\tau A}\rVert$
ILanczos	10	0.1	1.05e-003	4.57e+001	1
ILanczos	10	0.01	1.05e-003	1.54e+003	1
ILanczos	10	0.001	1.05e-003	2.63e+003	1
Algorithm	m	τ	β_{m+1}	$\lvert\alpha(T_m)\rvert\rvert$	$\lVert I - e^{-\tau A}\rVert + \tau\lvert\sigma\rvert$
SILanczos	10	0.1	1.05e-003	4.59e+001	1.1
SILanczos	10	0.01	1.05e-003	1.54e+003	1.01
SILanczos	10	0.001	1.05e-003	2.63e+003	1.001

Table 4.8: Analysis of the number of the Lanczos iteration for Example 9

approximation method, the error bound is

$$\|E_m^{SL}(\tau)\| \leq \tau\beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T e^{-tT_m} e_1|.$$

In this example, $|\alpha(T_m)|$ decreases with decreasing τ. Therefore the method takes smaller number of Lanczos iteration to reach to the same precision as when τ is bigger. For the inverse Lanczos method and the shift-and-invert Lanczos method, $|\alpha(T_m)|$ keep unchanged with decreasing τ, but the quantities $e^{-\tau A}$ is very close to the identity matrix as A has very small eigenvalues. So $\|I - e^{-\tau A}\|$ becomes smaller as τ decreases. Considering the bound for the shift-and-invert Lanczos method

$$\|E_m^{SIL}(\tau)\| \leq \beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-t(T_m^{-1}-\sigma I)} e_1|(\|I - e^{-\tau A}\| + \tau|\sigma|),$$

the overall error are smaller when τ decreases, so it takes less number of Lanczos iterations as τ becomes smaller. For the inverse Lanczos method,

$$\|E_m^{IL}(\tau)\| \leq \beta|\beta_{m+1}| \max_{0\leq t\leq\tau} |e_m^T T_m^{-1} e^{-tT_m^{-1}} e_1|\|I - e^{-\tau A}\|.$$

The quantity $|\alpha(T_m)|$ keeps unchanged, but $\|I - e^{-\tau A}\|$ is dropping with respect to τ, so the overall error drops from 10^{-2} to 10^{-4} from start, even though the error does not change as the dimension increases.

Similar argument can be made for Example 9. The difference is that $|\alpha(T_m)|$ for the inverse Lanczos and shift-and-invert Lanczos methods go to increase as τ becomes smaller. Yet the term $\|I - e^{-\tau A}\|$ and $\|I - e^{-\tau A}\| + \tau|\sigma|$ do not help in achieving better precision. This is because A has many large eigenvalues away from zero in this case. Therefore the number of Lanczos iteration increases as the timing step becomes smaller for the inverse Lanczos method and the shift-and-invert Lanczos method. So larger τ is more beneficial in terms of the convergence speed for the inverse Lanczos method and the shift-and-invert Lanczos method for matrix with large or modest eigenvalues. In order to achieve convergence of the approximation, the standard Lanczos method is the one since the other two might fail to converge by reducing τ.

Finally, we want to emphasize the point that, for a typical matrix, according to the characteristics of its spectrum, we can choose a method which uses larger τ, i.e. less timing-steps, to keep the error bound within a desirable level.

4.5 Solving the system of ODEs with a time-dependent forcing term

In this section, we discuss the numerical solution of the system of ODEs with a time-dependent forcing term by using Krylov subspace method for time-propagation.

As is given out at the beginning of this chapter, the solution of the system of ODEs of the form

$$\frac{dw(t)}{dt} = -Aw(t) + r(t), w(0) = w_0,$$

is

$$w(t+\delta) = e^{-\delta A}w(t) + \int_0^\delta e^{-(\delta-\tau)A}r(t+\tau)d\tau.$$

At each step of the time propagation, we need to evaluate a product of matrix exponential and a vector. We consider using Krylov subspace methods. The objective is to use large time step. The calculation of $w(t+\delta)$ would involve a numerical integration. In general the integration is approximated by a quadrature rule on the whole interval $[0,\delta]$ and the accuracy depends on which quadrature rule one uses. Consider a general quadrature formula of the form,

$$\int_0^\delta e^{-(\delta-\tau)A}r(t+\tau)d\tau \approx \delta \sum_{j=1}^p \mu_j e^{-(\delta-\tau_j)A}r(t+\tau_j)$$

where $\tau_j's$ are the quadrature nodes in the interval $[0,\delta]$. Therefore,

$$w(t+\delta) \approx e^{-\delta A}w(t) + \delta \sum_{j=1}^p \mu_j e^{-(\delta-\tau_j)A}r(t+\tau_j).$$

So the calculation of $w(t+\delta)$ would involve several calculations of the form $e^{-\delta A}v$. Once again, we can use larger timing step by choosing among different Krylov subspace methods. For the simplest trapezoidal rule, we can combine the vectors

$$w(t+\delta) = e^{-\delta A}[w(t) + \frac{\delta}{2}r(t)] + \frac{\delta}{2}r(t+\delta).$$

So only one evaluation of an exponential times a vector is needed at each time step. Another effective quadrature rule is Simpson rule. The approximated $w(t+\delta)$ is

$$w(t+\delta) = e^{-\delta A}\left(w(t) + \frac{\delta}{6}r(t)\right) + \frac{2\delta}{3}e^{-\frac{\delta}{2}A}r\left(t+\frac{\delta}{2}\right) + \frac{\delta}{6}r(t+\delta).$$

In the following, we give an example to illustrate the effectiveness of the approximation by using Simpson rule.

Example 10. *The problem is a simplified version in section 6.2 of [17]. Then the symmetric problem with time-varying forcing term*

$$\frac{\partial u(x,t)}{\partial t} = \Delta u(x,t) + r(x,t),$$

is defined on the unit real line, with homogeneous boundary conditions and initial conditions: $u(x,0) = x(x-1)$*. The exact solution of the above partial differential equation is give by*

$$u(x,t) = \frac{x(x-1)}{1+t}.$$

And the function r *can be solved from the above relationship. We are dealing with* $\gamma = 0$*. Then* $r(t) = \frac{x(1-x)}{(1+t)^2} - \frac{2}{1+t}$*. The grid points we took is* 1000 *in one dimension. The matrix has very large high eigenvalues, but the smallest eigenvalue is approximately* $9.87e+000$*. We apply all three Lanczos approximation methods to see the error of the approximation. In the shift-and-invert Lanczos method, we use shift* $\sigma = 1$*. During the whole experiment, we fix* $m = 10$*.*

In all four Figures, the top curve is the error history for the standard Lanczos approximation which is in the green dash dot line. The red solid line corresponds to the error history of the inverse Lanczos approximation. The blue solid "+" line corresponds to the error history of the shift-and-invert approximation. The inverse Lanczos approximation method and the shift-and-invert Lanczos approximation method have similar performances in this case. They almost overlap with each other and locate at the bottom part of each picture. In Figure 4.7, the inverse Lanczos and the shift-and-invert Lanczos approximation methods converge to order 1 for $\tau = 0.1$; in Figure 4.8, they both converge to order $1e-2$ for $\tau = 0.01$; in Figure 4.9, they both converge to order $1e-3$ for $\tau = 0.001$; in Figure 4.10, they both converge to order $1e-4$ for $\tau = 0.0001$. While for the standard Lanczos approximation method, the error stays around order 1 to $1e-1$. We see that in order to achieve an error precision of $1e-2$, we only need to take the time-step $\tau = 0.01$ for either the inverse Lanczos

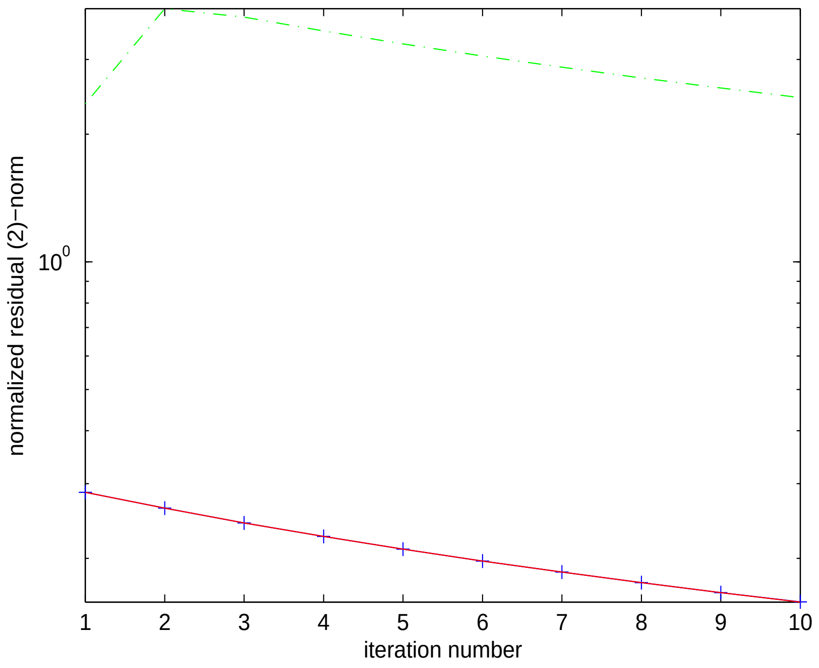

Figure 4.7: Error history for SL, IL and SIL approximation with $\tau = 0.1$

approximation method or the shift-and-invert Lanczos approximation method, while even with time-step $\tau = 0.0001$, the standard Lanczos approximation method still can not achieve the error $1e - 2$. In that case, we have to use much smaller τ to get the error precision $1e - 2$ for the standard Lanczos method, which is very time consuming. Therefore a better choice of the Lanczos methods will allow us to use less time-steps which is very useful for practical purposes.

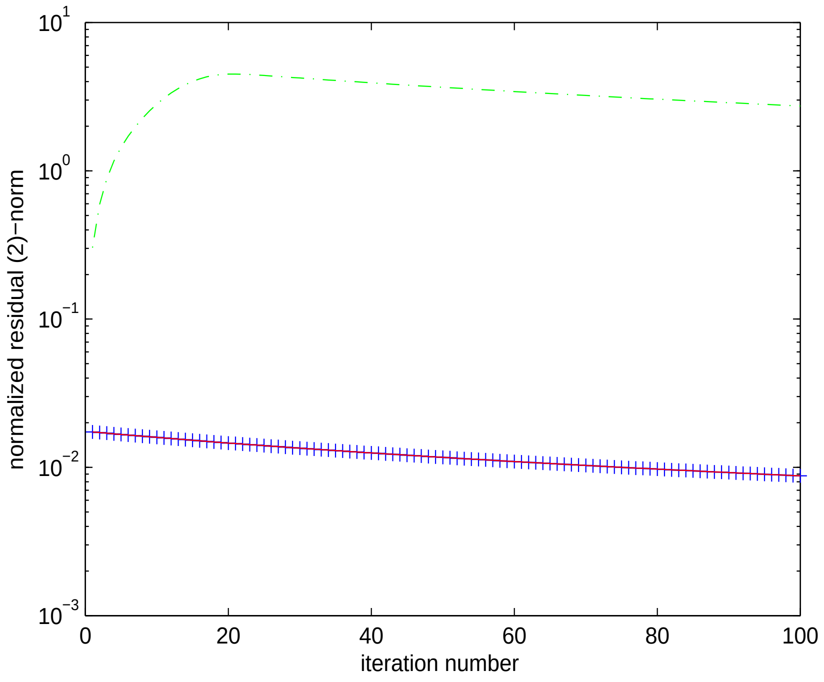

Figure 4.8: Error history for SL, IL and SIL approximation with $\tau = 0.01$

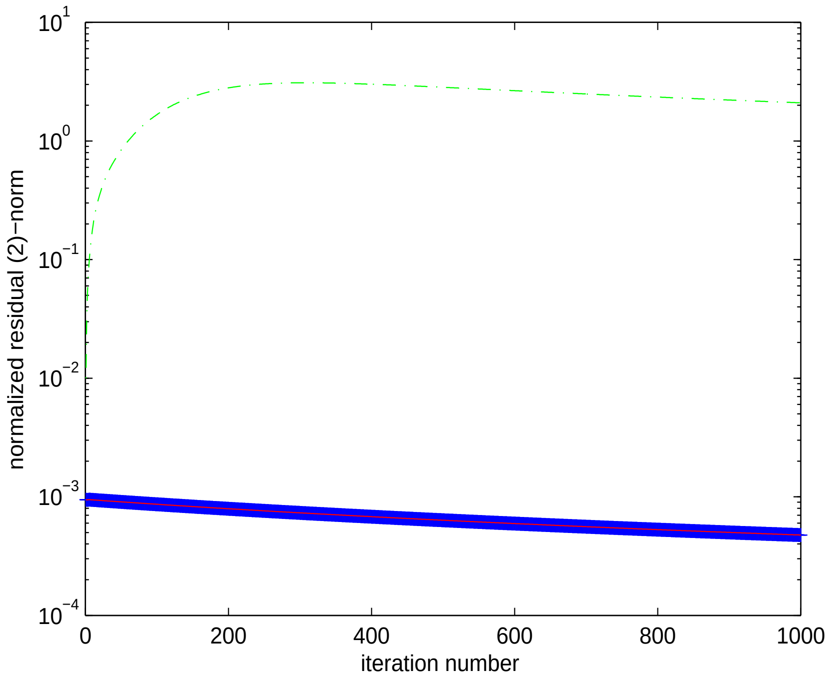

Figure 4.9: Error history for SL, IL and SIL approximation with $\tau = 0.001$

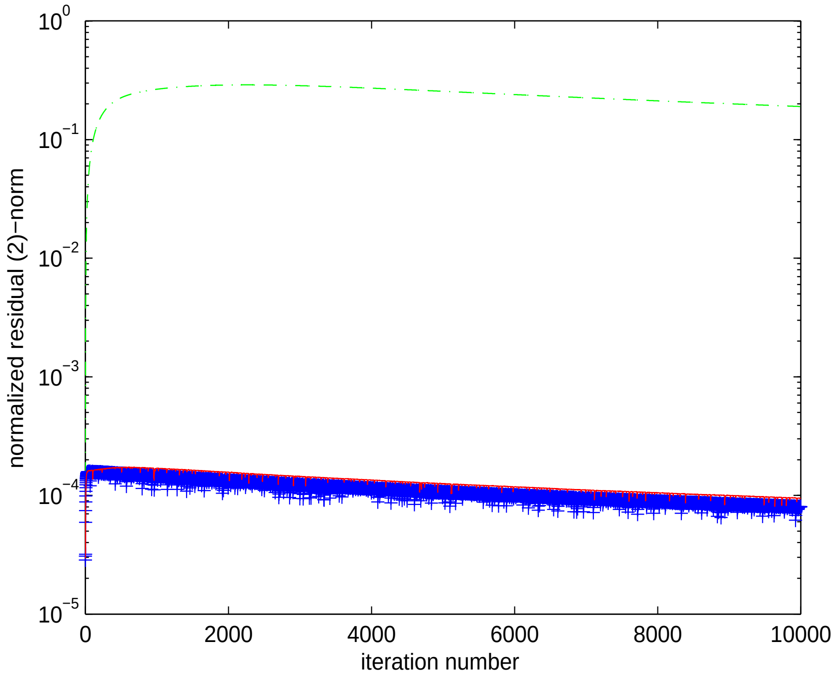

Figure 4.10: Error history for SL, IL and SIL approximation with $\tau = 0.0001$

Chapter 5 Conclusions and further work

In this thesis, we have proposed an inexact inverse subspace iteration method which can solve multiple and clustered eigenvalues of the generalized eigenvalue problems. We showed that the inexact inverse subspace iteration recovers its linear convergence rate of the outer iteration by solving the inner linear systems to a certain accuracy. We have also proposed a Krylov subspace approximation based on Lanczos Algorithm applied to A^{-1} for approximating matrix exponentials. We have developed a novel technique for analyze errors of three Lanczos methods. We have compared these Lanczos approximation methods both theoretically and numerically. In particular, the analysis allows us to choose which method to use based on the spectrum information of the matrix.

There are problems that remain open. First of all, we can apply our analysis to the Extended Krylov Subspace Method(EKSM), a recent work by Druskin and Knizhnerman in [13]. Due to the different construction of the orthonormal basis, the analysis might be quite different from the one that we discussed in chapter 4. Also due to the operation of A^{-1} on the vector, there is a possibility to improve the numerical approximation by switching to inexact linear solver. This has been observed in the shift and invert Krylov subspace method [37]. And the theoretical analysis is also needed to guide the usage of the inexact inner-outer iteration solver. Secondly, it would be useful to present a survey of all the current techniques and provide practical guidance on using different Krylov subspace approximations for approximating $e^{-\tau A}v$.

Bibliography

[1] M. AFANASJEW, M. EIERMANN AND O. G. ERNST, Implementation of a restarted Krylov subspace method for the evaluation of matrix functions, Linear Algebra and its applications, 429(2008), pp:2293-2314.

[2] Z. BAI, D. DAY, J. W. DEMMEL AND J. J. DONGARRA, A test matrix collection for non Hermitian eigenvalue problems (release 1.0). Technical Report. CS-97-355, Department of Computer Science, University of Tennessee, Knoxville, TN, USA, March 1997. Also available online from http://math.nist.gov/MatrixMarket.

[3] Z. BAI, J. DEMMEL, J. DONGARRA, A. RUHE AND H. VAN DER VORST, editors, Templates for the solution of algebraic eigenvalue problems: a practical guide, SIAM, Philadelphia, 2000.

[4] MICHELE BENZI AND GENE H. GOLUB, Bounds for the entries of matrix functions with applications to preconditioning, BIT, 39(1999), pp:417-438.

[5] MICHELE BENZI AND NADER RAZOUK, Decay bounds and $O(n)$ algorithms for approximating functions of sparse matrices, ETNA, 28(2007), pp:16-39.

[6] S. N. BERNSTEIN, Lecons sur les Proprietes Extremales et la Meilleure Approximation des Fonctions Analytiques d'une Variable Reelle, Gauthier-Villars, Paris, 1926.

[7] PETER B. BORWEIN, Rational approximations with real poles to e^{-x} and x^n, J. Approximation theory, 38(1983), pp:279-283.

[8] M. CROUZEIX, B. PHILIPPE AND M. SADKANE, The Davidson method, SIAM J. Sci. Stat. Comput., 15(1994), pp:62-76.

[9] JAMES W. DEMMEL, Applied numerical linear algebra, SIAM, Philadelphia, 1997.

[10] S. DEMKO, W. F. MOSS AND P. W. SMITH, Decay rates for inverses of band matrices, Math. Comp. 43(1984), pp:491-499.

[11] S. Demko, Inverses of banded matrices and local convergence of spline projections, SIAM J. Numer. Anal., 14(1977), pp:616-619.

[12] V. L. DRUSKIN AND L. A. KNIZHNERMAN, Krylov subspace approximations of eigenpairs and matrix functions in exact and computer arithemetic, Numer. Lin. Alg. Appl., 2(1995), pp:205-217.

[13] V. L. DRUSKIN AND L. A. KNIZHNERMAN, Extended Krylov subspaces: approximation of the matrix square root and related functions, SIAM J. Matrix Analysis and Applications, 19(1998), pp:755-771.

[14] MICHAEL EIERMANN AND OLIVER G. ERNST, A restarted Krylov subspace method for the evaluation of matrix functions, SIAM J. Numer. Anal., 44(2006), pp:2481-2504.

[15] J. VAN DEN ESHOF, A. FROMMER, T. LIPPERT, K. SCHILLING, AND H. A. VAN DER VORST, Numerical methods for the QCD overlap operator. sign-function and error bounds, Computer physics communications, 146(2002), pp:203-224.

[16] JASPER VAN DEN ESHOF AND MARLIS HOCHBRUCK, Preconditioning Lanczos approximations to the matrix exponential, SIAM J. on Sci. Computing, 27(2005), pp:1438-1457.

[17] E. GALLOPOULOS AND Y. SAAD, Efficient solution of parabolic equations by Krylov approximation methods, SIAM J. Sci. Statist. Comput., 13(1992), pp:1236-1264.

[18] G. H. GOLUB AND Q. YE, Inexact inverse iteration for genralized eigenvalue problems, BIT, 40(2000), pp:671-684.

[19] G. H. GOLUB AND Q. YE, Inexact preconditioned conjugate gradient method with inner-outer iteration, SIAM J. Sci. Comput., 21 (1999), pp:1305-1320.

[20] G. H. GOLUB, ZHENYUE ZHANG AND HONGYUAN ZHA, Large sparse symmetric eigenvalue problems with homogeneous linear constraints: the Lanczos process with inner-outer iteration, Linear Algebra and its Application, 309(2000), pp:289-306.

[21] G. H. GOLUB AND CH. VAN LOAN, Matrix computations, The Johns Hopkins University Press, the 2nd edition, 1989.

[22] M. HOCHBRUCK AND C. LUBICH, On krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal., 34(1997), pp:1911-1925.

[23] M. ILIC, I. W. TURNER, AND A. N. PETTITT, Bayesian computations and efficient algorithms for computing functions of large, sparse matrices, ANZIAM J. 45(E) (2004), pp:C504-C518.

[24] M. ILIC, I. W. TURNER, AND V. ANH, Numerical solution of the fractional poisson equations using an adaptively preconditioned Lanczos methods, SIAM journal and numerical anaylsis, 2007

[25] L. KNIZHNERMAN AND V. SIMONCINI, A new investigation of the extended Krylov subspace method for matrix function evaluations, Numer. Linear Algebra Appl., 2009.

[26] Y. LAI, K. LIN AND W. LIN, An inexact inverse iteration for large sparse eigenvalue problems, Numer. Linear Alg. Appl. 4 (1997), pp:425-437.

[27] R. LEHOUCQ AND K. MEERBERGEN, Using generalized Cayley transformations within an inexact rational Krylov sequence method, SIAM J. Matrix Anal. Appl. 20(1999), pp:131-148.

[28] FREITAG M. AND SPENCE A., Convergence of inexact inverse iteration with application to preconditioned iterative solves, BIT Numerical Mathematics, 47(2007), pp:27-44.

[29] FREITAG M. AND SPENCE A., Rayleigh quotient iteration and simplified Jacobi-Davidson method with preconditioned iterative solves, Linear Algebra and Its Applications, 2007.

[30] FREITAG M. AND SPENCE A., Convergence theory for inexact inverse iteration applied to the generalised nonsymmetric eigenproblem. Electronic Transactions on Numerical Analysis, 28(2007), pp:40-67.

[31] G. MEINARDUS, Approximation of functions: theory and numerical mehods, Springer-Verlag, New York, 1967.

[32] ROBBE M.,SADKANE M., SPENCE A. Inexact inverse subspace iteration with preconditioning applied to non-Hermitian eigenvalue problems, SIAM J. M. Anal. 2007.

[33] C. MOLER AND C. VAN LOAN, Nineteen dubious ways to compute the exponential of a matrix, SIAM Review, 20(1978), pp:801-836.

[34] C. MOLER AND C. VAN LOAN, Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later, SIAM Review, 45(2003), pp:3-49.

[35] R. MORGAN AND D. SCOTT, Preconditioning the Lanczos algorithm for sparse symmetric eigenvalue problems, SIAM J. Sci. Stat. Comput. 14(1998), pp:585-593.

[36] A. NAUTS AND R. WYATT, New approach to many state quantum dynamics: The recurisive residue generation method, Phys. Rev. Lett., 51(1983), pp:2238-2241.

[37] M. POPOLIZIO, Acceleration techniques for approximationg the matrix exponential, ph.d. thesis online, 2008.

[38] Y. SAAD. Analysis of some Krylov subspace approximations to the matrix exponential operator. SIAM J. Numer. Anal., 29(1992), pp:209-228.

[39] YOUSEF SAAD, Numerical methods for large eigenvalue problems, Manchester University Press, 1993

[40] Y. SAAD, Iterative methods for sparse linear systems, SIAM, 2000.

[41] V. SIMONCINI AND L. ELDEN, Inexact Rayleigh Quotient-Type methods for eigenvalue computations, BIT, 42(2002), pp:159-182.

[42] G. L. G. SLEIJPEN, G. L. BOOTEN, D. R. FOKKEMA AND H. A. VAN DER VORST, Jacobi Davidson type methods for generalized eigenproblems and polynomial eigenproblems. BIT, 36(1996), pp:595-633.

[43] G. L. G. SLEIJPEN, H. A. VAN DER VORST AND E. MEIJERINK, Efficient expansion of subspaces in the Jacobi-Davidson method for standard and generalized eigenproblems, Electron. Trans. Numer. Anal., 7(1998), pp:75-89.

[44] G. SLEIJPEN AND H. VAN DER VORST, A Jacobi-Davidson iteration method for linear eigenvalue problems, SIAM J. Matrix Anal. Appl. 17(1996), pp:401-425.

[45] A. STATHOPOULOS, Y. SAAD AND C. FISHER, Robust preconditioning of large sparse symmetric eigenvalue problems, J. Comp. Appl. Math., 64(1995), pp:197-215.

[46] E. D. STRULER, Improving the convergence of the jacobi-davidson algorithm, Technical Report UIUCDCS-R-2000-2173, 2000.

[47] D. SORENSON AND C. YANG, A truncated RQ iteration for large scale eigenvalue calculations, SIAM J. Matrix Anal. Appl., 19(1998), pp:1045-1073.

[48] HENK A. VAN DER VORST, Computational methods for large eigenvalue problems, Handbook of numerical analysis, North-Holland, Amsterdam, 2002

[49] A. ZAFER, Calculating the matrix exponential of a constant matrix on time scales, Applied Mathematics Letters, 21(2008), pp:612-616.

Vita

- **Personal Information**
  - Born November 3, 1977 in Fushun, Liaoning Province, China
- **Education**
  - 2009 [Expected], Ph.D., University of Kentucky
  - 2007, M.A., University of Kentucky
  - 2002, M.A., Dalian University of Technology
  - 1999, B.A., Liaoning Normal University
- **Scholastic and Professional Honors**
  - AY 2008-2009, Max Steckler Fellowship
  - AY 2008-2009, Summer Research Fellowship
  - AY 2007-2008, Max Steckler Fellowship
  - AY 2005-2008, Van Meter Fellowship