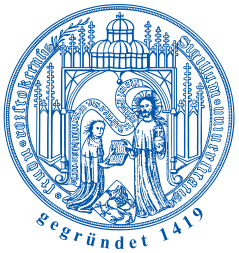
Universität Rostock



Traditio et Innovatio

A convergence analysis for preconditioned gradient type eigensolvers

Bachelorarbeit

Mathematisch-Naturwissenschaftliche Fakultät

Institut für Mathematik

Name:	Torben Sell
Matrikelnummer:	213206395
Betreuer und Gutachter:	Prof. Dr. Klaus Neymeyr
Gutachter:	Dr. Ming Zhou
Abgabedatum:	16.8.2016

Abstract

Symmetric eigenvalue problems can be found in a number of applications. The probably most famous example is the discretization of the 2D Laplace-operator or other partial differential operators. If a closed-meshed grid is used for the discretization, the resulting discretization matrix could be very large - making it impossible to use traditional solving algorithms. As these matrices are usually symmetric, positive definite and sparse, a method using these properties is needed. The use of iterative eigensolvers is self-evident, an implementation that considers the properties of the discretization matrix is presented at the end of this thesis. The main part of this thesis is a convergence analysis for one of the simplest iterative eigensolvers, the preconditioned inverse iteration. The concept of the method is discussed, and so are its improvements.

Contents

1 On the calculation of eigenvalues for large and sparse matrices

1.1 Eigenvalue problems - basic ideas

To calculate eigenvalues and eigenvectors is of central importance in both theoretical and applied mathematics. This section is intended to give an overview on important definitions concerning eigenvalue problems and to introduce a few ideas on how to work with eigenvalues.
We define:

Definition 1.1. *Let $A \in \mathbb{C}^{n\times n}$ be a matrix and $\lambda \in \mathbb{C}$. λ is called* eigenvalue *if there exists a $x \in \mathbb{C}^n \setminus \{0\}$ such that $Ax = \lambda x$.*
In this case we call x an eigenvector *and (λ, x) an* eigenpair.

We notice that an eigenpair represents a vector that is (apart from scaling) invariant under a linear transformation. The eigenvalue is the constant scaling the corresponding eigenvector. Eigenvectors are not unique: If we know an eigenvector x corresponding to the eigenvalue λ, any vector αx with $\alpha \neq 0$ is an eigenvector for the same eigenvalue: $A(\alpha x) = \alpha(Ax) = \alpha(\lambda x) = \lambda(\alpha x)$. This allows us to normalize every eigenvector: We speak of normalized eigenvectors when $\|x\| = 1$ with $\|.\|$ being the euclidean distance, as always in this thesis when not stated otherwise.

Definition 1.2. *A matrix $A \in \mathbb{R}^{n\times n}$ is called* symmetric *if $A = A^T$. A Matrix $B \in \mathbb{C}^{n\times n}$ is called* Hermitian *if $B = \overline{B^T} =: B^*$.*

The reader should note that every symmetric matrix is obviously Hermitian.

Definition 1.3.
A symmetric matrix $A \in \mathbb{R}^{n\times n}$ is called positive definite *if $\forall x \in \mathbb{R}^n \setminus \{0\} : x^T Ax > 0$.*
More generally, a Matrix $B \in \mathbb{C}^{n\times n}$ is called positive definite *if $\forall x \in \mathbb{C}^n \setminus \{0\} : \overline{x^T} Ax > 0$.*

Theorem 1.4. *Every Hermitian matrix $B \in \mathbb{C}^{n\times n}$ has only real eigenvalues. We can find a set of n orthogonal eigenvectors corresponding to the eigenvalues of B.*

Proof. Consider the euclidean inner product (x, y) and its properties. If λ is an eigenvalue

of B and x a corresponding eigenvector, we can deduce

$$\begin{aligned}\lambda(x,x) &= (\lambda x, x)\\ &= (Bx, x)\\ &= (x, B^*x)\\ &= (x, Bx)\\ &= (x, \lambda x)\\ &= \overline{\lambda}(x,x).\end{aligned}$$

As $(x,x) \neq 0$, we can devide by (x,x) and obtain $\lambda = \overline{\lambda}$. Thus, λ is real.
Let us now consider two eigenvectors x and y corresponding to distinct eigenvalues λ and μ. Then

$$\begin{aligned}\lambda(x,y) &= (\lambda x, y)\\ &= (Bx, y)\\ &= (x, B^*y)\\ &= (x, By)\\ &= (x, \mu y)\\ &= \mu(x,y).\end{aligned}$$

Therefore, $(\lambda - \mu)(x,y) = 0$. The eigenvalues are distinct, so $(x,y) = 0$, which means x and y are orthogonal. Each eigenvalue λ of geometric multiplicity k_λ has an eigenspace of dimension k_λ. This eigenspace has an orthonormal basis U_λ, we can choose such bases as sets of eigenvectors. The union of all these sets

$$U = \bigcup_\lambda U_\lambda$$

is the orthogonal set we were looking for. □

The proof uses arguments and definitions not described in detail in this thesis, a complete proof using the spectral theorem can be found in [4].

Corollary 1.5.

(a) Every symmetric matrix $A \in \mathbb{R}^{n\times n}$ has only real eigenvalues. We can find a set of n orthogonal eigenvectors corresponding to the eigenvalues of A.

(b) Every Hermitian matrix can be orthogonally diagonalized.

(c) Every symmetric matrix can be orthogonally diagonalized.

Proof. (a) is clear, for (b) and (c) consider $V = (v_1, v_2, ..., v_n)$, where v_i are the orthogonal eigenvectors.Then $AV = VD$ with a certain diagonal matrix D so that V^*AV and V^TAV are diagonal matrices. □

Theorem 1.6. *For a symmetric matrix A*

(a) A is positive definite and

(b) $\forall \lambda \in \mathbb{R} : \lambda > 0$ if λ is an eigenvalue of A

are equivalent.

Proof. We should first notice that according to theorem 1.4 all eigenvalues of A are real. "$(b) \implies (a)$": Let A be symmetric and all eigenvalues $\lambda_j > 0$. Thus, we have n orthogonal (and therefore linearly independent) eigenvectors, allowing us to write any $x \in \mathbb{R}^n$ as

$$x = \sum_{k=1}^{n} \alpha_k x_k.$$

If $x \neq 0$, at least one $\alpha \neq 0$. Therefore

$$x^T Ax = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j x_i^T A x_j = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j x_i^T \lambda_j x_j = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j \lambda_j x_i^T x_j = \sum_{j=1}^{n} \alpha_j^2 \lambda_j x_j^T x_j,$$

the last equality is due to the orthogonality of the eigenvectors. Each term is non-negative: $\alpha_j^2 \lambda_j \|x_j\|^2 \geq 0$. As at least one $\alpha_j \neq 0$, one term is greater than 0. Therefore $x^T Ax > 0$.
"$(a) \implies (b)$": Let A be symmetric and positive definite. Assuming $\exists k : \lambda_k \leq 0$. We consider the corresponding eigenvector x_k: $x_k^T A x_k = x_k^T \lambda_k x_k = \lambda_k \|x_k\|^2 \leq 0$. Therfore, A can't be positive definite, a contradiction. □

Example 1.7. *The matrix* $A = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}$ *is symmetric and positive definite.*

Proof. A is obviously symmetric and has the eigenvalues $\lambda_1 = 4$ and $\lambda_2 = 2$ with the corresponding eigenvectors $x_1 = (1, 1)^T$ and $x_2 = (-1, 1)^T$. By theorem 1.6 A is positive definite. □

The above mentioned eigenvalue problem can be generalized.

Definition 1.8. *For matrices $A, B \in \mathbb{C}^{n\times n}$ an* eigenpair *(λ, x) is defined as a scalar $\lambda \in \mathbb{C}$ and a vector $x \in \mathbb{C}^n \setminus \{0\}$ obeying*

$$Ax = \lambda Bx. \tag{1.1}$$

The search for such an eigenpair is called the generalized eigenvalue problem. *The pair (A, B) is called a* matrix pencil *or just* pencil.

It is obvious, that the generalized eigenvalue problem becomes a standard eigenvalue problem when choosing $B = I$, where $I \in \mathbb{C}^{n\times n}$ is the identity matrix.
We want to note some properties of the generalized eigenvalue problem: If B is invertible with the inverse matrix B^{-1}, we can multiply 1.1 by B^{-1} from the left and obtain

a standard eigenvalue problem $B^{-1}Ax = \lambda x$. When A and B are symmetric, B^{-1} is symmetric, too, however, $B^{-1}A$ does *not* have to be symmetric. As it is often too complicated to calculate the inverse matrix B^{-1}, as one often wants to retain the symmetry, there are algorithms designed to solve the generalized eigenvalue problem. These algorithms are introduced in [11][Chapter 15] and we will discuss one of them later in this thesis. When A and B are both symmetric and positive definite, there is an important theorem concerning the eigenvalues and eigenvectors. A (more general) version can be found in [11][Chapter 15].

Theorem 1.9. *If $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times n}$ are symmetric and positive definite, there are n eigenvalues λ_1, ..., λ_n in the interval $[-\|B^{-1}A\|, \|B^{-1}A\|]$. To match them, there are n linearly independent eigenvectors v_1, ..., v_n. For different eigenvalues, the corresponding eigenvectors are B-orthogonal, i.e., $(v_i, v_j)_B = 0$ if $\lambda_i \neq \lambda_j$. If $\lambda_i = \lambda_j$, then v_i, v_j may be chosen B-orthogonal.*

It is sometimes useful to consider the inverted form $\mu Ax = Bx$ with $\mu = \frac{1}{\lambda}$, where the smallest eigenvalue λ in the original problem corresponds to the largest eigenvalue μ of the inverted problem. We will use the inverted form to analyse the convergence in section 3.

Even though we will later focus on symmetric and positive definite matrices A and B only, there are very interesting things that can occur with the generalized eigenvalue problem. With not symmetric and positive definite matrices, one might - for example - have infinitely many eigenvalues, one may have less than n eigenvalues. All possible cases and examples can be found at full length in [11].

1.2 Origin of matrices - discretization of the two-dimensional Laplace-operator

We will now look at the origin of a large, sparse, symmetric and positive definite matrix. As an example we present how to discretize the two-dimensional Laplace-operator on $\Omega =]0, 1[^2$.

Definition 1.10. *Let $\Omega \subset \mathbb{R}^n$ be a domain. For $u : \Omega \to \mathbb{R}$ we call*

$$\Delta u(x) = \sum_{k=1}^{n} \frac{\partial^2 u}{\partial x_i^2} \tag{1.2}$$

the Laplace operator.

In some applications it is interesting to find the eigenvalues and eigenfunctions of the Laplace operator. [1]

[1] When building a skyscraper, one wants to ensure that an earthquake is not able to make the skyscraper uncontrollably oscillate. That can happen when the skyscraper has the same eigenfunctions as the

Example 1.11. *We want to solve the PDE*

$$-\Delta u = \lambda u \quad \forall (x,y) \in]0,1[^2 \tag{1.3}$$
$$u(x,0) = u(x,1) = u(0,y) = u(1,y) = 0 \quad \forall x,y \in [0,1], \tag{1.4}$$

using numerical methods. The problem is known as the eigenvalue problem for the Laplace operator*.*

We need discrete derivatives to solve the problem. For a function $f : \mathbb{R} \to \mathbb{R}$ and a small $h > 0$ we can discretize the first derivative $f'(x_j)$ in one direction by one of the three following approximations:

$$\begin{aligned}
&1.: \quad f'(x_j) \approx D^+ x_j := \frac{f(x_{j+1}) - f(x_j)}{h} \\
&2.: \quad f'(x_j) \approx D^- x_j := \frac{f(x_j) - f(x_{j+1})}{h} \\
&3.: \quad f'(x_j) \approx D^0 x_j := \frac{f(x_{j+1}) - f(x_{j-1})}{2h}.
\end{aligned}$$

We can then use the first two options to obtain an approximation for the second derivative:

$$f''(x_j) \approx D^+ D^- x_j := \frac{f(x_{j+1}) - 2f(x_j) + f(x_{j-1})}{h^2}.$$

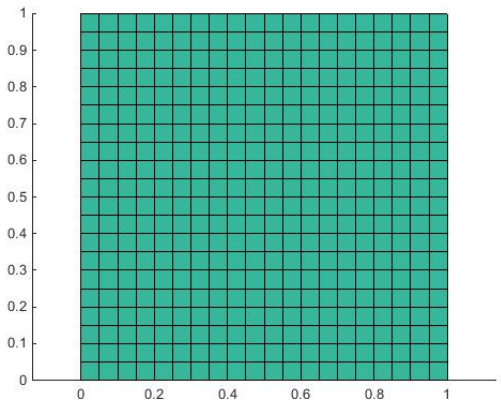


Figure 1.1: A grid on $[0,1]^2$ with $n = 19$, thus $n^2 = 19^2$ inner points

For a finite number of points $(x,y) \in]0,1[^2$ we want to approximate the Laplace operator. In order to do so, we firstly lay a grid over the square $[0,1]^2$. Let $h := \frac{1}{n+1}$, where n is the wanted number of inner points per direction, gaining n^2 inner points in total. Then

earthquake. Therefore, engineers will ensure that the building has no eigenfunction that is similar to vibrations of an earthquake. Similarily, wind can stimulate the resonance frequency of bridges, the most famous example is the Tacoma Narrows Bridge, which collapsed in 1940.

each gridpoint and its value can be described as

$$(i,j)\hat{=}(x_i,y_j) := (hi,hj) \qquad \text{with } i,j \in \{0,..,n+1\}$$
$$u_{i,j} := u(x_i,y_j).$$

The Laplace operator (for the inner points only!) can then be approximated by forming

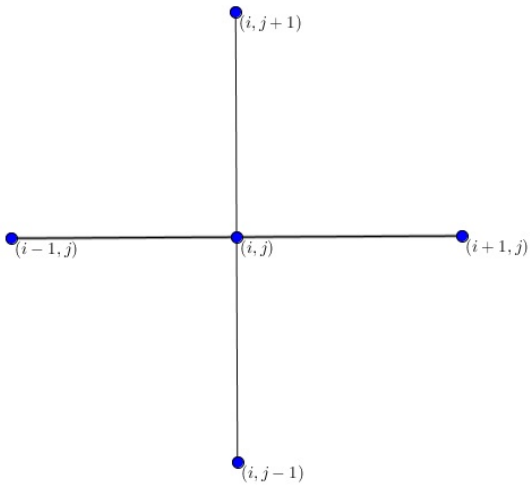


Figure 1.2: 5 point star used for the discretization of the Laplace operator

an equation:

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \tag{1.5}$$
$$\approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} \tag{1.6}$$
$$= \frac{-4u_{i,j} + u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}}{h^2}. \tag{1.7}$$

We now define $k := i + n(j-1) \in \mathbb{R}^{n^2}$ and substitute $u_{i,j}$ by u_k to get rid of the double index. Remember that we have the negative Laplace operator in (1.3), thus we have to multiply equation (1.7) by -1. By doing so, we obtain

$$-\Delta u \approx Au = \lambda u \tag{1.8}$$

with

$$A = \frac{1}{h^2}\begin{pmatrix} B & -I & 0 & 0 & \dots & 0 \\ -I & B & -I & \ddots & & 0 \\ 0 & -I & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -I & 0 \\ \vdots & & \ddots & -I & B & -I \\ 0 & \dots & \dots & 0 & -I & B \end{pmatrix} \in \mathbb{R}^{n^2 \times n^2}, \tag{1.9}$$

where

$$B = \begin{pmatrix} 4 & -1 & 0 & 0 & \dots & 0 \\ -1 & 4 & -1 & \ddots & & 0 \\ 0 & -1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -1 & 0 \\ \vdots & & \ddots & -1 & 4 & -1 \\ 0 & \dots & \dots & 0 & -1 & 4 \end{pmatrix} \in \mathbb{R}^{n \times n} \tag{1.10}$$

and the identity matrix $I \in \mathbb{R}^{n \times n}$.
If we have a solution for the eigenvalue problem (1.8), we can take u as the approximation of an eigenfunction on the (discrete) grid we laid over the square $[0,1]^2$. It is easy to derive a function $\overline{u} : [0,1]^2 \to \mathbb{R}$ with $\overline{u} = u$ on the grid giving a good approximation of the solution to (1.3) by simply defining $\overline{u}(x,y)$ as piecewise linear or bilinear interpolating polynomials. We realize that we have just formulated an eigenvalue problem as in Definition 1.1!

It should be noticed that we can write (1.8) in a nice way using the Kronecker product:

$$A = I \otimes C + C \otimes I, \tag{1.11}$$

where $C = \frac{1}{h^2}\text{tridiag}(-1, 2, -1)$.

We also want to look at the properties of the matrix A, which we will later use to accelerate convergence by choosing an appropriate method.

- The matrix A is obviously symmetric.
- Theorem 1.4 shows that all eigenvalues must be real, furthermore it can be shown that A is positive definite.
- For practical purposes, it is usually not enough to use only 19^2 inner points as in figure 1.1, but rather 1000^2 or more. This causes A to be *large* - meaning that we can not simply use standard methods, such as the QR method, since the factorization and saving of the matrices would consume enormous space and time.
- Finally we can also assert A to be sparse - in most lines we will only find five

nonzero entries, for $A \in \mathbb{R}^{n^2 \times n^2}$, this are only roughly $\frac{500}{n^2}$ % of all matrix entries.

1.3 Special features

One problem occuring with large matrices is that it becomes difficult to save the matrices efficiently. Luckily, we asserted that above mentioned A is sparse. This means we can make use of the many zeros in the matrix and only store the nonzero entries. Algorithms for sparse matrices can also be improved by parallel computing. However, this shall not be the focus of this thesis, we ask the reader to read [13][73 ff.] for further information.

Another important fact to mention is that we can use a symmetric and positive definite matrix A to define an inner product by $(x, y)_A := x^T Ay$. This is used for modifications of the later introduced methods, such as the conjugate gradient method.

2 Preconditioned gradient methods for the Rayleigh quotient

In this section we will discuss properties of the Rayleigh quotient and how it can be used to define iterative eigensolvers.

2.1 The Rayleigh quotient

Definition 2.1. *Let $A \in \mathbb{R}^{n\times n}$ and $x \in \mathbb{R}^n \setminus \{0\}$.*
The Rayleigh quotient *is defined as $\mu(x) := \mu_A(x) := \frac{x^T Ax}{x^T x} = \frac{(x,Ax)}{(x,x)}$.*
The generalized Rayleigh quotient *is defined as $\mu(x) := \mu_{A,B}(x) := \frac{x^T Ax}{x^T Bx} = \frac{(x,Ax)}{(x,Bx)}$.*

In this section we will mainly use the first notation, however, the second notation concerning matrix pencils will come in handy in section 3.

Theorem 2.2. *The eigenvalues for a symmetric matrix $A = A^T \in \mathbb{R}^{n\times n}$ are $\lambda_{max} := \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n =: \lambda_{min}$. Then*

$$\forall x \in \mathbb{R}^n \setminus \{0\} : \lambda_{min} \leq \mu_A(x) \leq \lambda_{max}. \tag{2.1}$$

Furthermore, if v_{min} and v_{max} are corresponding eigenvectors to λ_{min} and λ_{max}, respectively, then

$$\lambda_{min} = \mu_A(v_{min}) = \min_{x\neq 0} \mu_A(x) \tag{2.2}$$

$$\lambda_{max} = \mu_A(v_{max}) = \max_{x\neq 0} \mu_A(x). \tag{2.3}$$

Proof. For a proof see [12][Chapter 1]. □

A similar statement holds for the generalized Rayleigh quotient:

Theorem 2.3. *For a symmetric positive definite matrix $A \in \mathbb{R}^{n\times n}$ and a symmetric positive definite matrix $B \in \mathbb{R}^{n\times n}$ the generalized Rayleigh quotient $\mu(x) = \mu_{A,B}(x)$ enjoys the following properties:*

(1) $\mu(\alpha x) = \mu(x)$, where $\alpha \neq 0$,

(2) $\forall x \in \mathbb{R}^n \setminus \{0\} : \lambda_{min} \leq \mu(x) \leq \lambda_{max}$,

(3) $\nabla\mu(x) = \frac{2(Ax-\mu(x)Bx)}{x^TBx}$,

(4) $\mu(x)$ *is stationary, if, and only if,* x *is an eigenvector of the pencil* (A, B).

Proof. (1) $\mu(\alpha x) = \frac{(\alpha x)^T A(\alpha x)}{(\alpha x)^T B(\alpha x)} = \frac{\alpha^2 x^T Ax}{\alpha^2 x^T Bx} = \mu(x)$.

(2) Due to A and B being symmetric and positive definite, we have n B-orthogonal eigenvectors v_1, ..., v_n by theorem 1.9. Thus we can write any x as $x = \sum_{i=1}^n \alpha_i v_i$. We deduce

$$\begin{aligned}
(x, Ax) &= \left(x, A\sum_{i=1}^n \alpha_i v_i\right) = \sum_{i=1}^n (x, \alpha_i A v_i) = \sum_{i=1}^n (x, \alpha_i \lambda_i B v_i) \\
&= \sum_{i=1}^n \lambda_i (x, \alpha_i B v_i) = \sum_{i=1}^n \lambda_i \left(\sum_{j=1}^n \alpha_j v_j, \alpha_i B v_i\right) \\
&\overset{v_iB-orth.}{=} \sum_{i=1}^n \lambda_i (\alpha_i v_i, \alpha_i B v_i) \\
&\geq \lambda_{min} \sum_{i=1}^n (\alpha_i v_i, \alpha_i B v_i) \\
&\overset{v_iB-orth.}{=} \sum_{i=1}^n \lambda_{min} \left(\sum_{j=1}^n \alpha_j v_j, \alpha_i B v_i\right) = \lambda_{min}\left(x, B\sum_{i=1}^n \alpha_i v_i\right) \\
&= \lambda_{min}(x, Bx).
\end{aligned}$$

Dividing by $(x, Bx) > 0$ yields the lower bound for the generalized Rayleigh quotient, the upper bound is derived similarly.

(3) Keeping $\nabla x^T Ax = 2Ax$ in mind, we obtain

$$\begin{aligned}
\nabla\mu(x) &= \nabla\left(\frac{x^TAx}{x^TBx}\right) \\
&= x^TAx\left(\nabla\left(\frac{1}{x^TBx}\right)\right) + \frac{1}{x^TBx}\left(\nabla(x^TAx)\right) \\
&= x^TAx\frac{-2Bx}{(x^TBx)^2} + \frac{1}{x^TBx}2Ax \\
&= \mu(x)\frac{-2Bx}{x^TBx} + \frac{2Ax}{x^TBx} \\
&= \frac{-2\mu(x)Bx + 2Ax}{x^TBx} \\
&= \frac{2(Ax - \mu(x)Bx)}{x^TBx}.
\end{aligned}$$

(4) Let x be an eigenvector of (A, B). Then $Ax = \lambda Bx$, thus $\mu(x) = \frac{\lambda x^TBx}{x^TBx} = \lambda$. Substituting this in the property (3) conducts $\nabla\mu(x) = \frac{2(\lambda Bx - \lambda Bx)}{x^TBx} = 0$. On the

contrary, if $\nabla\mu(x) = 0$, then $Ax = \mu(x)Bx$, and x is an eigenvector corresponding to the eigenvalue $\mu(x)$.

□

2.2 Minimizing the Rayleigh quotient using gradient methods

Theorem 2.3 tells us, that when looking for generalized eigenvectors, we can similarly find stationary points of the generalized Rayleigh quotient. We will use this information in this section to deduce a gradient type eigensolver.

We firstly consider the (preconditioned) Richardson-method to solve systems of linear equations: If an iteration x_n is known, we can obtain the next iteration by

$$x_{n+1} = x_n - \tau_n T(Ax_n - b),$$

where T is the preconditioner and τ_n a scalar. Good choices for T are obviously $T = A^{-1}$, since we will converge in one step with $\tau = 1$, or $T = A^{\dagger}$, a pseudo-inverse, in the case of a singular matrix A. However, iterative methods wouldn't be needed if the inverse is known, but one might guess a good approximation of the inverse, gaining better performance.
We now consider an eigenvalue problem

$$Ax = \lambda x \Leftrightarrow (A - \lambda I)x = 0.$$

If a sufficiently accurate approximation λ^* of the eigenvalue λ is known, we can obtain the eigenvector x by solving this system of linear equations $(A - \lambda^* I)x = 0$[1] or, if preconditioned, $T(A - \lambda^* I)x = 0$. Using the above mentioned Richardson-method, the next iteration is obtained by

$$x_{n+1} = x_n - \tau_n T(A - \lambda^* I)x_n. \tag{2.4}$$

One should keep in mind, that λ^* needs to be suggested in each step. To implement the method, we replace λ^* by the Rayleigh quotient $\mu_A(x)$, hopefully getting better approximations for the eigenvalue in every step. It may not yet be clear, why this method is a gradient method. We will answer this question shortly.
For the generalized eigenvalue problem with symmetric, positive definite matrices $A, B \in \mathbb{R}^{n\times n}$ we are now introducing a method to approach the smallest eigenvalue. Instead of considering the eigenvalue problem $Ax = \lambda Bx$, we consider the *inverted form* $\mu Ax = Bx$ with $\mu = \frac{1}{\lambda}$, which has advantages when presenting the convergence analysis in section 3. Obviously, minimizing λ and maximizing μ are equivalent. Theorem 2.3 shows that maximizing the Rayleigh quotient $\mu_{B,A}(x)$ brings the maximal eigenvalue μ. From here

[1] Since this system is singular, we can actually find a vector $x \neq 0$ that solves this equation.

on, we write $\mu(x) := \mu_{B,A}(x)$, and $\lambda(x) := \mu_{A,B}(x)$, respectively. A manifest idea is to improve the iterate x along the gradient of the Rayleigh quotient $\nabla\lambda(x) = \frac{2(Ax-\lambda(x)Bx)}{x^TBx} = \frac{2}{\mu(x)x^TBx}(\mu(x)Ax - Bx)$. As an increase of $\mu(x)$ is achieved when decreasing $\lambda(x)$, we use the negative gradient and generally modify this by adding a preconditioner $\tilde{T}$:

$$x_{n+1} = x_n + \tau_n \frac{1}{\mu(x_n)} \frac{2}{x_n^TBx_n} \tilde{T}(Bx_n - \mu(x_n)Ax_n).$$

In order to simplify the representation we use the notation $T_n := \tau_n \frac{2}{x_n^TBx_n}\tilde{T}$:

$$x_{n+1} = x_n + \frac{1}{\mu(x_n)} T_n(Bx_n - \mu(x_n)Ax_n). \tag{2.5}$$

As we iterate along the preconditioned gradient of the Rayleigh quotient, it does make sense to speak about *gradient type eigensolvers*. It now becomes clear, why method 2.4 can be seen as a gradient method as well: It is simply a special case of 2.5 with $B = I$, $\lambda^* = \frac{1}{\mu(x)} = \frac{x^TAx}{x^TBx} = \lambda(x)$, and $\tau_n = 1$. It should also be noted, that we call method 2.5 *PINVIT*, standing for *Preconditioned INVerse Iteration*.

As it is very complicated and time consuming to calculate the best scalar τ_n explicitly, simplest methods either contain a fixed step size, e.g. $\tau_n = 1$, or use Armijo-Goldstein conditions. Both possibilities are implemented in function 7.5.

2.3 Modification and acceleration of a gradient method

In this subsection we will discuss possibilities for modificating, accelerating, and generally improving method (2.5).

2.3.1 PINVIT2 and PINVIT3

A question arising is how to find the maximum of the Rayleigh quotient $\mu(x)$ along the gradient, i.e. the maximum in the one-dimensional manifold $U := x + span\{\nabla\lambda(x)\}$. In other words, in each step we need to find $T = T(\tau)$ such that

$$x' = x + \arg\max_{\tau>0} \left(\frac{(x + \frac{1}{\mu}T(Bx - \mu Ax))^TB(x + \frac{1}{\mu}T(Bx - \mu Ax))}{(x + \frac{1}{\mu}T(Bx - \mu Ax))^TA(x + \frac{1}{\mu}T(Bx - \mu Ax))} \right) \frac{1}{\mu}T(Bx - \mu Ax)). \tag{2.6}$$

2

It is a promising idea to maximize the Rayleigh quotient in the two-dimensional subspace $\tilde{U} = span\{x_n, \nabla\lambda(x_n)\}$, which can be easily done by the Rayleigh-Ritz method. Let $y := \alpha x_n + \beta\nabla\lambda(x_n)$ denote a solution to this maximization task. Then, by theorem 2.3, $\tilde{y} = \frac{1}{\alpha}y = x_n + \frac{\beta}{\alpha}\nabla\lambda(x_n)$ has the same Rayleigh quotient as y and therefore satisfies the maximization task in (2.6), since $U \subset \tilde{U}$. This method is known as *PINVIT2*, as it contains a maximization in a two-dimensional subspace. Both an implementation for the standard, and the generalized eigenvalue problem can be found at the end of this thesis.
However, we may increase the dimension of the subspace: *PINVIT3* maximizes the Rayleigh quotient in the three-dimensional subspace $U = span\{x_n, \nabla\lambda(x_n), x_{n-1}\}$. Unfortunately, the task of finding the maximum of (2.6) gets harder with every dimension, outweighing the advantage of adding another dimension to U quickly. Here again, implementations are provided at the end of this thesis. All the above mentioned methods calculate only one eigenpair, while it may be interesting to calculate the k largest eigenvalues. This is achieved by the *LOBPCG* method that is discussed in section 5.

[2]To find the maximum, we define $y := \frac{1}{\mu}T(Bx - \mu Ax)$ and use

$$\begin{aligned}\frac{d}{d\tau}\left((x+\tau y)^T B(x+\tau y)\right) &\overset{B=B^T}{=} \frac{d}{d\tau}\left(\tau^2 y^T By + 2\tau x^T By + x^T Bx\right)\\ &= 2\tau y^T By + 2x^T By\end{aligned} \tag{2.7}$$

to derive

$$\begin{aligned}f'(\tau) &= \frac{d}{d\tau}\left(\frac{(x+\frac{\tau}{\mu}T(Bx-\mu Ax))^T B(x+\frac{\tau}{\mu}T(Bx-\mu Ax))}{(x+\frac{\tau}{\mu}T(Bx-\mu Ax))^T A(x+\frac{\tau}{\mu}T(Bx-\mu Ax))}\right)\\ &= \frac{d}{d\tau}\left(\frac{(x+\tau y)^T B(x+\tau y)}{(x+\tau y)^T A(x+\tau y)}\right)\\ &\overset{(2.7)}{=} \left(\frac{(2\tau y^T By + 2x^T By)[(x+\tau y)^T A(x+\tau y)] - [(x+\tau y)^T B(x+\tau y)](2\tau y^T Ay + 2x^T Ay)}{[(x+\tau y)^T A(x+\tau y)]^2}\right).\end{aligned}$$

Thus, we have a stationary point of $f(\tau)$, when the numerator of its derivative equals 0.

$$\begin{aligned}0 &= f'(\tau)\\ &= \left(2\tau y^T By + 2x^T By\right)\left[(x+\tau y)^T A(x+\tau y)\right] - \left[(x+\tau y)^T B(x+\tau y)\right]\left(2\tau y^T Ay + 2x^T Ay\right)\\ &= 2\tau^2\left((y^T By)(x^T Ay) - (x^T Bx)(x^T By)\right) + 2\tau\left((y^T By)(x^T Ax) - (x^T Bx)(y^T Ay)\right)\\ &\quad + 2\left((x^T By)(x^T Ax) - (x^T Bx)(x^T Ay)\right),\end{aligned}$$

which is true for

$$\begin{aligned}\tau = &\frac{(x^T Bx)(y^T Ay) - (y^T By)(x^T Ax)}{2}\\ &\pm\left[\left(\frac{(x^T Bx)(y^T Ay) - (y^T By)(x^T Ax)}{2}\right)^2 + (x^T Bx)(x^T Ay) - (x^T By)(x^T Ax)\right]^{\frac{1}{2}}\end{aligned}$$

by the quadratic formula.

2.3.2 Preconditioners

When speaking of modificating a gradient method one should certainly look at the preconditioner T. T does not necessarily have to be symmetric and positive definite as one would assume in the first place. As shown in [5], it is enough for T to fulfill

$$s_{max}\left(I - A^{1/2}TA^{1/2}\right) \leq \gamma < 1 \tag{2.8}$$

for a given $\gamma \in [0, 1[$ and s_{max} the largest singular value of $I - A^{1/2}TA^{1/2}$. For sparse matrices A, preconditioners are often calculated by performing an incomplete Cholesky factorization or an incomplete LU factorization of A, see [2] or [6] for a good overview over different preconditioners. [3] considers sparse inverse preconditioners only, however these preconditioners are very useful when tackling large problems, all implementations presented at the end of this thesis use a sparse preconditioner T.

2.3.3 The Rayleigh-Ritz method

We now look at how to find the maximum in each step for the PINVIT2 method. We notice that - according to theorem 2.3 - the Rayleigh quotient is maximized at an eigenvector corresponding to the largest eigenvalue in the subspace. This eigenvalue can be found using the Rayleigh-Ritz method, which is explained in [11][Chapter 11], and we waive a description and explanation in this thesis, however implementations for different subspaces can be found at the end of this thesis, see 7.0.2.

3 Convergence Analysis

This is the main part of this thesis, where we discuss the convergence of the above mentioned method (2.5). The argumentation follows that in [1]. From here on we're dropping the indices in (2.5) and will consider one step of the method:

$$x' = x + \frac{1}{\mu(x)} T(Bx - \mu(x)Ax). \tag{3.1}$$

Our aim is to prove the following theorem:

Theorem 3.1 (Convergence Rate Bound)**.** *If $\mu_{i+1} < \mu(x) < \mu_i$ and T satisfies $s_{max}\left(I - A^{1/2}TA^{1/2}\right) \leq \gamma < 1$ (see 2.8) for a given $\gamma \in [0, 1[$, then for x' in* (3.1) *one of the following statements is true:*

$$\mu(x') \geq \mu_i, \tag{3.2}$$

$$0 < \frac{\mu_i - \mu(x')}{\mu(x') - \mu_{i+1}} \leq \sigma^2 \frac{\mu_i - \mu(x)}{\mu(x) - \mu_{i+1}}, \quad \sigma := \gamma + (1-\gamma)\frac{\mu_{i+1}}{\mu_i}. \tag{3.3}$$

The remaining part of this section is split as follows:
Firstly, we will show that we can reduce the inverted general eigenvalue problem $\mu Ax = Bx$ to the standard eigenvalue problem $\mu x = Bx$ by choosing $A = I$ without loss of generality.
Secondly, we will only consider the case $\gamma = 0$, before approaching the general case $0 < \gamma < 1$ thereafter.

3.1 Simplifying the problem

As promised, we will show that we can choose $A = I$ without loss of generality. While doing this, we will use an inner product $(x, y)_A$ defined by $(x, y)_A := x^T Ay$ for different matrices, according to the index used. The reader should note, that this is actually an inner product if A is symmetric and positive definite. The corresponding norm is $||x||_A := (x, x)^{1/2} = (x^T Ax)^{1/2}$.
For a symmetric and positive definite matrix A the root $A^{1/2}$ (and its inverse $A^{-1/2}$) is well defined, so that we can transform our problem as follows:
$x'_A := A^{1/2}x'$, $x_A := A^{1/2}x$, $B_A := A^{-1/2}BA^{-1/2}$, $T_A := A^{1/2}TA^{1/2}$, and $r_A := B_A x_A - \kappa x_A$, where $\kappa := \mu(x)$.
Furthermore, we define a closed ball $\mathcal{B}_A := \{y \in \mathbb{R}^n : ||B_A x_A - y||^2 \leq \gamma^2 ||r_A||^2\}$.

Theorem 3.2. *If T satisfies* (2.8)*, than for x' in* (3.1) *it holds that*

$$\kappa x'_A = B_A x_A - (I - T_A)(B_A x_A - \kappa x_A) \in \mathcal{B}_A. \tag{3.4}$$

Proof. We multiply (3.1), $x' = x + \frac{1}{\mu(x)} T(Bx - \mu(x)Ax)$ by $\mu(x)A^{1/2}$ from the left and obtain

$$\begin{aligned}
\mu(x)A^{1/2}x' &= \mu(x)A^{1/2}x + \frac{\mu(x)A^{1/2}}{\mu(x)} T(Bx - \mu(x)Ax) \\
&= \mu(x)A^{1/2}x + A^{1/2}T(Bx - \mu(x)Ax) \\
&= \kappa(A^{1/2}x) + A^{1/2}TBx - \kappa A^{1/2}TAx \\
&= \kappa x_A + A^{1/2}TA^{1/2}A^{-1/2}BA^{-1/2}A^{1/2}x - \kappa A^{1/2}TA^{1/2}A^{1/2}x \\
&= \kappa x_A + (A^{1/2}TA^{1/2})(A^{-1/2}BA^{-1/2})(A^{1/2}x) - \kappa(A^{1/2}TA^{1/2})(A^{1/2}x) \\
&= \kappa x_A + T_A B_A x_A - \kappa T_A x_A \\
&= \kappa x_A + T_A B_A x_A - \kappa T_A x_A - B_A x_A + B_A x_A \\
&= -(I - T_A)(B_A x_A - \kappa x_A) + B_A x_A \\
&= -(I - T_A)r_A + B_A x_A \quad \text{(the equation postulated!)} \\
&\overset{\mu(x)A^{1/2}x' = \kappa x'_A}{\Longleftrightarrow} B_A x_A - \kappa x'_A = (I - T_A)r_A.
\end{aligned}$$

We now have to show that $\kappa x'_A \in \mathcal{B}_A$. Applying the norm to the last equation and rewriting (2.8) to $s_{max}(I - T_A) \leq \gamma < 1$, we conclude

$$\begin{aligned}
||B_A x_A - \kappa x'_A||^2 &= ||(I - T_A)r_A||^2 \\
&\leq ||I - T_A||^2 ||r_A||^2 \\
&\overset{2.8}{\leq} \gamma^2 ||r_A||^2.
\end{aligned}$$

Thus, $\kappa x'_A \in \mathcal{B}_A$. □

As promised, we will now transform the problem, using the transformations used in theorem 3.2. From now on we will use the following notations:

$$\begin{aligned}
T &= T_A \\
B &= B_A \\
x &= x_A \\
x' &= x'_A \\
r &= r_A \\
\mu(x) &= \frac{x^T Bx}{x^T x}.
\end{aligned}$$

Method (3.1) now becomes

$$\mu(x)x' = Bx - (I - T)(Bx - \mu(x)x), \tag{3.5}$$

and the preconditioner T has to satisfy the condition

$$||I - T|| \leq \gamma, \tag{3.6}$$

which is the transformed version of condition (2.8).
Lastly, we look at the closed ball $\mathcal{B} = \mathcal{B}_A$, which now became

$$\mathcal{B} = \{y : ||Bx - y|| \leq \gamma||r||\}.$$

Theorem 3.2 showed that $\mu(x)x' \in \mathcal{B}$, theorem 2.3[(1)] showed that $\mu(\mu(x)x') = \mu(x')$ (choose $\alpha = \mu(x)$). Therefore, $\mu(x') \in \{\mu(y) : y \in \mathcal{B}\}$!
Our main goal is still to find a bound for $\mu(x')$. If we minimize $\mu(.)$ in $\mathcal{B}$, we have found a lower estimate for $\mu(x')$, as this would be the worst case: $\min_{x \in \mathcal{B}} \mu(x) \leq \mu(x')$. The transformation lead to $A = I$ and hasn't changed anything regarding the convergence bound. Without any loss of generality, we can set $A = I$.
From here on we can focus on the problem of finding the largest eigenvalue of the eigenvalue problem $Bx = \mu x$. We can visualize our gradient method as follows. From a starting point x_0 we will use the Rayleigh quotient to approximate the largest eigenvalue. We will find the next iterate x_1 by increasing the Rayleigh quotient along the gradient of the Rayleigh quotient in x_0. In x_1 we will once again use the Rayleigh quotient to approximate the largest eigenvalue and repeat the process as often as needed.

3.2 Special case: $\gamma = 0$ and $T = I$

If we choose $\gamma = 0$, we realize that condition (3.6) requires $T = I$, simplifying method (3.5) to $\mu(x)x' = Bx$. We reformulate theorem 3.1 for this special case:

Theorem 3.3 (Convergence Rate Bound - $\gamma = 0$)**.** *If $\mu_{i+1} < \mu(x) < \mu_i$, then for x' in $\mu(x)x' = Bx$ one of the following statements is true:*

$$\mu(x') \geq \mu_i, \tag{3.7}$$

$$0 < \frac{\mu_i - \mu(x')}{\mu(x') - \mu_{i+1}} \leq \sigma^2 \frac{\mu_i - \mu(x)}{\mu(x) - \mu_{i+1}}, \quad \sigma := \frac{\mu_{i+1}}{\mu_i}. \tag{3.8}$$

Before coming to the actual proof, we want to give an outline on how to proof this theorem. If we think of the gradient method, we will have to ask what the worst possible case concerning the improvement of the Rayleigh quotient of the iterations could be. The proof therefore tries to find a minimum for $\mu(x')$ when $\mu(x)$ is given. We will find that for an extremal $\mu(x')$, x is a linear combination of two eigenvectors, which allows us to derive that inequality (3.8) indeed holds.

Proof of theorem 3.3. If $\mu(x) \in]\mu_{i+1}, \mu_i[$ is given, we can define $\kappa := \mu(x) \in]\mu_{i+1}, \mu_i[$ and minimize $f(y) := \mu(By)$ only for those y satisfying $\mu(y) = \kappa$. Since $\kappa = \mu(y) = \frac{y^T By}{y^T y} \Leftrightarrow \kappa y^T y - y^T By = 0$, we can consider $h(y) := \kappa y^T y - y^T By = 0$ as a constraint to the task of minimizing $f(y) = \mu(By)$. This extremal problem can be solved using the method of Lagrange multipliers:
We introduce an auxiliary function $\mathcal{L}(y, a) := f(y) + ah(y)$ with a constant a, the Lagrange multiplier. At a stationary point of $\mathcal{L}$, we have

$$\nabla_y \mathcal{L} = \nabla_y f(y) + a\nabla_y h(y) = 0, \tag{3.9}$$
$$\nabla_a \mathcal{L} = h(y) = 0. \tag{3.10}$$

We rewrite equation (3.9):

$$\begin{aligned}
\nabla_y \mathcal{L} &= \nabla_y f(y) + a\nabla_y h(y) \\
&= \nabla_y \left(\mu(By)\right) + a\nabla_y \left(\kappa y^T y - y^T By\right) \\
&\overset{\text{2.3, chain rule}}{=} B\frac{2(B^2 y - \mu(By)By)}{y^T BBy} + a(2\kappa y - 2By) \\
&= \frac{2B^3 y - 2\mu(By)B^2 y}{||By||^2} - 2aBy + 2a\kappa y \\
&= 0.
\end{aligned}$$

We multiply with $\frac{||By||^2}{2}$ and obtain

$$0 = B^3 y - \mu(By)B^2 y - a||By||^2 By + a||By||^2 \kappa y.$$

The substitution $c := a||By||^2$ yields

$$0 = B^3 y - \mu(By)B^2 y - cBy + c\kappa y. \tag{3.11}$$

We will now show that any y solving equation (3.11) is a linear combination of at most two eigenvectors.
To obtain this result, we firstly note that $c\kappa$ is positive: $\kappa > 0$ is due to it being larger than an eigenvalue of B and B being symmetric and positive definite, see theorem 1.6. To show that $c > 0$, we consider

$$\mu(By) = \frac{(By, B(By))}{(By, By)} \tag{3.12}$$
$$\begin{aligned}
\Leftrightarrow 0 &= (By, B(By)) - \mu(By)(By, By) \\
&= y^T B^3 y - \mu(By) y^T B^2 y \\
&= y^T (B^3 y - \mu(By)B^2 y) \qquad (3.13) \\
&= (y, B^3 y - \mu(By)B^2 y) \\
&= (B^3 y - \mu(By)B^2 y, y)
\end{aligned}$$

and further

$$\begin{aligned}
c||By-\kappa y||^2 &= c(By-\kappa y, By-\kappa y)\\
&= (cBy-c\kappa y, By-\kappa y)\\
&\overset{(3.11)}{=} (B^3y-\mu(By)B^2y, By-\kappa y)\\
&= (B^3y-\mu(By)B^2y, By)-\kappa(B^3y-\mu(By)B^2y, y)\\
&\overset{(3.13)}{=} (B^3y-\mu(By)B^2y, By)-0\\
&\overset{(3.13)}{=} (B^3y-\mu(By)B^2y, By)-\mu(By)(B^3y-\mu(By)B^2y, y)\\
&= (B^3y-\mu(By)B^2y, By-\mu(By)y)\\
&= y^TB^4y-2\mu(By)y^TB^3y+\mu(By)^2y^TB^2y\\
&= (B^2y-\mu(By)By, B^2y-\mu(By)By)\\
&= ||B^2y-\mu(By)By||^2,
\end{aligned}$$

stating $c=\frac{||B^2y-\mu(By)By||^2}{||By-\kappa y||^2}>0$ as promised.
The right-hand side of equation (3.11) is therefore a polynomial with both leading and last coefficient being positive. We write

$$p(B)y = B^3y-\mu(By)B^2y-cBy+c\kappa y. \tag{3.14}$$

The linear independence of the eigenvectors yields that y can be rewritten as $y=\sum_{i=1}^{m} w_i$, where w_i is the projection of y on every eigenspace. As the eigenspaces are orthogonal, and either w_i is an eigenvector or $w_i=0$, inserting $y=\sum_{i=1}^{m} w_i$ in 3.14 yields

$$\begin{aligned}
p(B)y &= p(B)\sum_{i=1}^{m} w_i\\
&= B^3\sum_{i=1}^{m} w_i-\mu(B\sum_{i=1}^{m} w_i)B^2\sum_{i=1}^{m} w_i-cB\sum_{i=1}^{m} w_i+c\kappa\sum_{i=1}^{m} w_i\\
&= \sum_{i=1}^{m}(B^3-\mu(\sum_{i=1}^{m} Bw_i)B^2-cB+c\kappa)w_i\\
&= \sum_{i=1}^{m}(\mu_i^3-\mu(\sum_{i=1}^{m}\mu_iw_i)\mu_i^2-c\mu_i+c\kappa)w_i\\
&\overset{w_i orth}{=} \sum_{i=1}^{m}(\mu_i^3-\mu(By)\mu_i^2-c\mu_i+c\kappa)w_i\\
&= \sum_{i=1}^{m} p(\mu_i)w_i.
\end{aligned}$$

Since the w_i are orthogonal, $p(\mu_i)w_i=0 \quad \forall i$, showing either $p(\mu_i)=0$ or $w_i=0$. As $p(.)$ is a polynomial of third degree, it has maximally three positive roots. However, maximally two changes of signs in the coefficients occur, thus only two roots can be

positive according to Descartes' rule of signs ([7]). Therefore, only for two indices k and l we have $p(\mu_k) = p(\mu_l) = 0$, allowing $w_k, w_l \neq 0$. This means, that y is a linear combination of at most two eigenvectors v_k and v_l!

We will use this result to finally prove theorem 3.3.
As the behaviour of x is invariant under scaling, we can assume $x = v_k + \alpha v_l$ with two normalized eigenvectors v_k, v_l without loss of generality. This x minimizes $f(y) = \mu(By)$. Remember that $\mu(x') = \mu(Bx)$. We have a look at the Rayleigh quotients $\mu(x) = \mu(v_k + \alpha v_l)$, and $\mu(x') = \mu(Bx) = \mu(B(v_k + \alpha v_l))$.

$$\begin{aligned}
\mu(x) &= \mu(v_k + \alpha v_l) \\
&= \frac{(v_k + \alpha v_l)^T B(v_k + \alpha v_l)}{(v_k + \alpha v_l)^T (v_k + \alpha v_l)} \\
&= \frac{v_k^T B v_k + \alpha v_l^T B v_k + \alpha^2 v_l^T B v_l + \alpha v_k^T B v_l}{v_k^T v_k + \alpha v_l^T v_k + \alpha^2 v_l^T v_l + \alpha v_k^T v_l} \\
&= \frac{\mu_k v_k^T v_k + \alpha\mu_k v_l^T v_k + \alpha^2 \mu_l v_l^T v_l + \alpha\mu_l v_k^T v_l}{\|v_k\|^2 + \alpha v_l^T v_k + \alpha^2 \|v_l\|^2 + \alpha v_k^T v_l} \\
&= \frac{\mu_k \|v_k\|^2 + \alpha\mu_k v_l^T v_k + \alpha^2 \mu_l \|v_l\|^2 + \alpha\mu_l v_k^T v_l}{\|v_k\|^2 + \alpha v_l^T v_k + \alpha^2 \|v_l\|^2 + \alpha v_k^T v_l} \\
&= \frac{\mu_k + \alpha^2 \mu_l}{1 + \alpha^2} \qquad (3.15) \\
\Leftrightarrow \alpha^2 &= \frac{\mu_k - \mu(x)}{\mu(x) - \mu_l} \qquad (3.16)
\end{aligned}$$

$$\begin{aligned}
\mu(x') &= \mu(B(v_k + \alpha v_l)) \\
&= \mu(Bv_k + \alpha B v_l) \\
&= \mu(\mu_k v_k + \alpha\mu_l v_l) \\
&= \ldots \quad \text{as above} \qquad (3.17) \\
&= \frac{\mu_k + \frac{\mu_l^2}{\mu_k^2}\alpha^2 \mu_l}{1 + \frac{\mu_l^2}{\mu_k^2}\alpha^2} \\
\Leftrightarrow \frac{\mu_l^2}{\mu_k^2}\alpha^2 &= \frac{\mu_k - \mu(Bx)}{\mu(Bx) - \mu_l} = \frac{\mu_k - \mu(x')}{\mu(x') - \mu_l}. \qquad (3.18)
\end{aligned}$$

We put equations (3.16) and (3.18) together and obtain

$$\frac{\mu_k - \mu(x')}{\mu(x') - \mu_l} \overset{3.18}{=} \frac{\mu_l^2}{\mu_k^2}\alpha^2 \overset{3.16}{=} \frac{\mu_l^2}{\mu_k^2}\frac{\mu_k - \mu(x)}{\mu(x) - \mu_l} = \sigma^2 \frac{\mu_k - \mu(x)}{\mu(x) - \mu_l}, \qquad (3.19)$$

where $\sigma = \frac{\mu_l}{\mu_k}$. This is almost the equality we were looking for. If $\mu_l < \mu_k$, κ can be chosen in $]\mu_l, \mu_k[$, which implies that there exists an i such that

$$\mu_l \leq \mu_{i+1} < \mu(x) = \kappa < \mu_i \leq \mu_k. \qquad (3.20)$$

Note that $\mu_k < \mu_l$ doesn't bring anything new, simply consider $x = v_l + \beta v_k$, leading to the same results.
With $\mu_l < \mu_k$, it follows from 3.19 that $\mu(x') > \mu(x)$. Therefore, either $\mu(x') \geq \mu_i$ or $\mu(x) < \mu(x') < \mu_i$ is true. Now we have a look at a few fractions. Generally, it holds for $0 < p \leq q$ and any $\delta \geq 0$ that

$$\begin{aligned} & p\delta \leq q\delta \\ \Leftrightarrow\ & pq + p\delta \leq pq + q\delta \\ \Leftrightarrow\ & p(q+\delta) \leq q(p+\delta) \\ \Leftrightarrow\ & \frac{p}{q} \leq \frac{p+\delta}{q+\delta}. \end{aligned} \tag{3.21}$$

Without any constraint, we can consider $\mu_k = \mu_i + \delta_1$ and $\mu_{i+1} = \mu_l + \delta_2$, bringing us the inequalities[1]

$$\frac{\mu_i - \mu(x')}{\mu_i - \mu(x)} \leq \frac{\mu_i + \delta_1 - \mu(x')}{\mu_i + \delta_1 - \mu(x)} = \frac{\mu_k - \mu(x')}{\mu_k - \mu(x)}, \tag{3.22}$$

$$\frac{\mu(x) - \mu_{i+1}}{\mu(x') - \mu_{i+1}} \leq \frac{\mu(x) - \mu_{i+1} + \delta_2}{\mu(x') - \mu_{i+1} + \delta_2} = \frac{\mu(x) - \mu_l}{\mu(x') - \mu_l}. \tag{3.23}$$

Finally, we obtain

$$\begin{aligned} \frac{\mu_i - \mu(x')}{\mu(x') - \mu_{i+1}} \frac{\mu(x) - \mu_{i+1}}{\mu_i - \mu(x)} &= \frac{\mu_i - \mu(x')}{\mu_i - \mu(x)} \frac{\mu(x) - \mu_{i+1}}{\mu(x') - \mu_{i+1}} \\ &\overset{(3.22),(3.23)}{\leq} \frac{\mu_k - \mu(x')}{\mu_k - \mu(x)} \frac{\mu(x) - \mu_l}{\mu(x') - \mu_l} \\ &= \frac{\mu_k - \mu(x')}{\mu(x') - \mu_l} \frac{\mu(x) - \mu_l}{\mu_k - \mu(x)} \\ &\overset{(3.19)}{=} \sigma^2 = \left(\frac{\mu_l}{\mu_k}\right)^2 \leq \left(\frac{\mu_{i+1}}{\mu_i}\right)^2. \end{aligned}$$

Bringing $\frac{\mu(x)-\mu_{i+1}}{\mu_i-\mu(x)}$ on the right-hand side of the equation results in inequality (3.8). □

3.3 The convergence analysis for the preconditioned gradient method: $0 < \gamma < 1$

We will now generalize the ideas used in the last section to proof the general theorem 3.1. We will find that when using a preconditioner, the method of Lagrange multipliers cannot be directly used, since we have inequalities as a restriction instead of an equality as above, hence we will use the Karush-Kuhn-Tucker theory to find a minimum. The rest

[1] Note that in both cases, the numerator is greater than the denominator due to $\mu_{i+1} < \mu(x) < \mu(x') < \mu_i$.

is similar: We will show that in an extremal point, x is again a linear combination of at most two eigenvectors. This will once more allow us to derive the inequality postulated in the theorem. The reader will realize that this proof is quite long and uses two lemmas that we will prove only afterwards to hopefully make the proof easier to overview and understand.

Proof of theorem 3.1. We define the residual vector as $r := Bx - \mu(x)x$, which is non-zero if x is not an eigenvector. Note that we will consider preconditioned gradient method given by

$$\mu(x)x' = Bx - (I - T)r. \tag{3.24}$$

Depending on the preconditioner T, we have different solutions for the next iterate x'. Generally, any vector in the closed ball $\mathcal{B} = \{y : ||Bx - y|| \leq \gamma\|r\|\}$ is possible: The new iterate is somewhere near Bx, the center of the ball, and due to $\|I - T\| \leq \gamma$, we can't go further away from the center as $\gamma\|r\|$, the radius of the ball $\mathcal{B}$. We will now show $\mu(x') > \mu(x)$, which (with an additional argument) proves the left-hand side of the inequality (3.3), before tackling the more complicated right-hand side.
We have

$$\begin{aligned}
&\|Bx - y\|^2 \leq \gamma^2\|r\|^2 \overset{\gamma\leq 1}{<} \|r\|^2 = \|Bx - \mu(x)x\|^2 \\
\Leftrightarrow\ & x^TB^2x + y^Ty - 2x^TBy < x^TB^2x + \mu(x)^2x^Tx - 2\mu(x)x^TBx \\
\Leftrightarrow\ & y^Ty < 2x^TBy + \mu(x)^2x^Tx - 2\mu(x)x^TBx \\
\Leftrightarrow\ & \|y\|^2 < 2(x,y)_B + \mu(x)\frac{x^TBx}{x^Tx}x^Tx - 2\mu(x)x^TBx \\
&\qquad = 2(x,y)_B - \mu(x)x^TBx \\
&\qquad = \frac{y^TBy - y^TBy + 2\mu(x)x^TBy - \mu(x)^2x^TBx}{\mu(x)} \\
&\qquad = \frac{\|y\|_B^2 - \|y - \mu(x)x\|_B^2}{\mu(x)} \\
&\qquad \leq \frac{\|y\|_B^2}{\mu(x)} \\
\Leftrightarrow\ & \mu(x) < \frac{\|y\|_B^2}{\|y\|^2} = \frac{y^TBy}{y^Ty} = \mu(y).
\end{aligned}$$

As in the last section, we have $\mu(x') = \mu(\mu(x)x')$ and $\mu(x)x' \in \mathcal{B}$ by definition of $\mathcal{B}$. Thus, we obtain $\mu(x') > \mu(x) > \mu_{i+1}$. The left-hand side of (3.3) holds when $\mu(x') < \mu_i$, otherwise inequality (3.2) holds. The harder part to prove is the right-hand side of the equation. We will therefore minimize the Rayleigh quotient in the ball $\mathcal{B}$, in order to find the worst-case scenario for the behaviour of $\mu(x')$. We do this by using the Karush-Kuhn-Tucker-Theory, which is similar to the method of Lagrange multipliers, but allows inequalities in the constraints. We will use the following lemma, the proof is postponed and can be found in the next subsection.

Lemma 3.4. *For $\gamma \in]0,1[$ and κ not an eigenvalue of B, may $\{x^*, y^*\}$ be a solution to the constrained minimization problem*

$$\begin{aligned} &\textit{minimize } f(x,y) = \mu(y), x \neq 0 \\ &\textit{with constraints } g(x,y) = ||Bx - y||^2 - \gamma^2 ||Bx - \kappa x||^2 \leq 0, \textit{ and} \\ &h(x,y) = \kappa(x,x) - (x, Bx) = 0. \end{aligned}$$

If x^* is not an eigenvector of B, then it holds that both x^* and y^* belong to a two-dimensional invariant subspace of B corresponding to two distinct eigenvalues, and $\sin \angle(Bx^*, y^*) = \gamma \sin \angle(Bx^*, x^*)$, where $\angle(u,v) = \arccos\left(\frac{(u,v)}{\|u\|\|v\|}\right)$.

As in the exactly preconditioned case, we now derive a convergence rate bound for a two-dimensional B-invariant subspace before tackling the general case. We will again formulate a lemma and give a proof later to retain the overview.

Lemma 3.5. *Let x and y belong to a two-dimensional invariant subspace of B corresponding to the eigenvalues $\mu_k > \mu_l$ and satisfy*

$$\sin\left(\arccos\left(\frac{(Bx,y)}{\|Bx\|\|y\|}\right)\right) = \gamma \sin\left(\arccos\left(\frac{(Bx,x)}{\|Bx\|\|x\|}\right)\right)$$

(see lemma 3.4), where x is not an eigenvector.
Then it holds that

$$\frac{\mu_k - \mu(y)}{\mu(y) - \mu_l} \frac{\mu(x) - \mu_l}{\mu_k - \mu(x)} \leq \left(\gamma + (1-\gamma)\frac{\mu_l}{\mu_k}\right)^2. \tag{3.25}$$

We now have the framework to nifty finish the proof of the convergence rate bound theorem. We have

$$\mu_{i+1} < \mu(x) < \mu(x') < \mu_i \tag{3.26}$$

by the first steps of the proof. Since $\|Bx - y\| \leq \gamma \|Bx - \kappa x\| \Leftrightarrow g(x) \leq 0$ in Lemma 3.4, this lemma showed that there exists a y satisfying

$$\mu(y) \leq \mu(x'), \tag{3.27}$$

and bound (3.25) holds with

$$\mu_l < \mu(x) < \mu(y) < \mu_k. \text{ Further we have} \tag{3.28}$$

$$\mu_l \leq \mu_{i+1} < \mu_i \leq \mu_k. \tag{3.29}$$

Since we assumed $\mu_{i+1} < \mu(x) < \mu(x') < \mu_i$, we can put this and inequalities (3.28),

(3.29) together and gain

$$\mu_l \leq \mu_{i+1} < \mu(x) < \mu(y) \leq \mu(x') < \mu_i \leq \mu_k. \tag{3.30}$$

We use the argumentation of (3.21), and $\mu_i = \mu_{i+1} + \delta$ to show

$$\begin{aligned} & \frac{\mu_{i+1} - \mu(y)}{\mu(x') - \mu_{i+1}} \leq \frac{\mu_{i+1} + \delta - \mu(y)}{\mu(x') - \mu_{i+1} - \delta} = \frac{\mu_i - \mu(y)}{\mu(x') - \mu_i} \\ \Leftrightarrow\ & \frac{\mu(x') - \mu_i}{\mu(x') - \mu_{i+1}} \geq \frac{\mu_i - \mu(y)}{\mu_{i+1} - \mu(y)} \\ \overset{\text{multiply by } -1}{\Leftrightarrow}\ & \frac{\mu_i - \mu(x')}{\mu(x') - \mu_{i+1}} \leq \frac{\mu_i - \mu(y)}{\mu(y) - \mu_{i+1}}, \end{aligned} \tag{3.31}$$

where we have to note $\mu(x') - \mu_i < 0$ according to (3.30), which changes the relation in the first step. Putting everything together we obtain

$$\begin{aligned} \frac{\mu_i - \mu(x')}{\mu(x') - \mu_{i+1}} \frac{\mu(x) - \mu_{i+1}}{\mu_i - \mu(x)} &\overset{(3.31)}{\leq} \frac{\mu_i - \mu(y)}{\mu(y) - \mu_{i+1}} \frac{\mu(x) - \mu_{i+1}}{\mu_i - \mu(x)} \\ &= \frac{\mu_i - \mu(y)}{\mu_i - \mu(x)} \frac{\mu(x) - \mu_{i+1}}{\mu(y) - \mu_{i+1}} \\ &\overset{cf.(3.21)}{\leq} \frac{\mu_k - \mu(y)}{\mu_k - \mu(x)} \frac{\mu(x) - \mu_l}{\mu(y) - \mu_l} \\ &= \frac{\mu_k - \mu(y)}{\mu(y) - \mu_l} \frac{\mu(x) - \mu_l}{\mu_k - \mu(x)} \\ &\overset{(3.25)}{\leq} \left(\gamma + (1-\gamma)\frac{\mu_l}{\mu_k}\right)^2 \\ &\leq \left(\gamma + (1-\gamma)\frac{\mu_{i+1}}{\mu_i}\right)^2, \end{aligned}$$

the last inequality is due to $\frac{\mu_l}{\mu_k} < \frac{\mu_{i+1}}{\mu_i}$, which follows from (3.29). □

3.4 Amendments to the Convergence Rate Bound

We will now provide the proofs for lemma 3.4 and 3.5.

Lemma 3.4, proof. We firstly assume $y = 0$. As we know that the residual $Bx - \kappa x = Bx - \mu(x)x$ is orthogonal to x, we obtain $\|Bx - \kappa x\| \leq \|Bx\|$, and thus $\|Bx\|^2 - \gamma^2\|Bx - \kappa x\|^2 \overset{\gamma<1}{>} \|Bx\|^2 - \|Bx - \kappa x\|^2 \geq 0$. This is a contradiction to the first constraint $g(x, y) \leq 0$, it follows that $y \neq 0$.
The existence of a minimum follows, when we consider that the smooth function $f(x, y)$ certainly has a minimum on the compact set M, where $\|x\| = 1$, and that any (nonzero) multiple of this minimum on the compact set is also a minimum on $\mathbb{R}^n \setminus \{0\}$: Let

$\{x^*, y^*\}$ be a solution with $\|x^*\| = 1$, then $\{\alpha x^*, \alpha y^*\}$ fulfills $g(\alpha x^*, \alpha y^*) = \alpha^2(\|Bx^* - y^*\|^2 - \gamma^2\|Bx^* - \kappa x^*\|^2) = \alpha^2 g(x^*, y^*) \leq 0$ and $h(\alpha x^*, \alpha y^*) = \alpha^2(\kappa(x^*, x^*) - (x^*, Bx^*)) = \alpha^2 h(x^*, y^*) = 0$. Furthermore, $\mu(\alpha y^*) = \mu(y^*)$ by theorem 2.3.
We now aim to prove the regularity in any stationary point $\{x^*, y^*\}$. We have to prove linear independency of the gradients of $g(x, y)$ and $h(x, y)$. As

$$\nabla g(x, y) = \begin{pmatrix} 2(B^2x - By - \gamma^2(B - \kappa I)(Bx - \kappa x)) \\ 2(y - Bx) \end{pmatrix},$$

and

$$\nabla h(x, y) = \begin{pmatrix} 2(\kappa x - Bx) \\ 0 \end{pmatrix}.$$

We may assume linear dependency, which lets us deduce that $y - Bx = 0$, ergo $y = Bx$. Inserting this in the x-derivatives produces the equation

$$\begin{aligned} -2\gamma^2(B - \kappa I)(Bx - \kappa x) &\overset{y=Bx}{=} 2(B^2x - By - \gamma^2(B - \kappa I)(Bx - \kappa x)) \\ &= \frac{\partial g}{\partial x} \\ &\overset{\nabla g = \beta \nabla h}{=} \beta \frac{\partial h}{\partial x} \\ &= 2\beta(\kappa x - Bx) \\ &= -2\beta(Bx - \kappa x). \end{aligned}$$

Thus, $Bx - \kappa x$ is an eigenvector corresponding to the eigenvalue $\frac{\beta}{\gamma^2}$ of the matrix $B - \kappa I$. But this means, that x is an eigenvalue of B, in contradiction to the assumption made in the lemma:

$$\begin{aligned} (B - \kappa I)(Bx - \kappa x) &= \frac{\beta}{\gamma^2}(Bx - \kappa x) \\ \Leftrightarrow (B - \kappa I)^2 x &= \frac{\beta}{\gamma^2}(B - \kappa I)x \\ \Leftrightarrow (B - \kappa I)x &= \frac{\beta}{\gamma^2}x \\ \Leftrightarrow Bx &= \left(\frac{\beta}{\gamma^2} + \kappa\right)x. \end{aligned}$$

We note that ∇g and ∇h are linearly independent. Furthermore, f, g, and h are smooth, hence the Karush-Kuhn-Tucker conditions are valid, therefore for each stationary point $\{x^*, y^*\}$ there exist constants a and b (similar to the Lagrange multipliers in the case where $\gamma = 0$) that fulfill

$$\nabla f(x^*, y^*) + a\nabla g(x^*, y^*) + b\nabla h(x^*, y^*) = 0.$$

Our aim is now to use this equation to obtain a third degree polynomial as we did in

the "'$\gamma = 0$"'-case. We drop the subscript * from now on, thus x refers to x^*, and y to y^*, respectively.
The x-derivatives bring the equation

$$2a(B^2x - By - \gamma^2(B - \kappa I)(Bx - \kappa x)) - 2b(Bx - \kappa x) = 0, \tag{3.32}$$

and the y-derivatives bring

$$\frac{2(By - \mu(y)y)}{(y, y)} + 2a(y - Bx) = 0, \tag{3.33}$$

respectively, where we used theorem 2.3 and the derivatives calculated before.
The Karush-Kuhn-Tucker theory implies that $ag(x, y) = 0$, so either $a = 0$, or $g(x, y) = 0$ must hold. If y was an eigenvector, we have three cases to consider:

- $a \neq 0$, then $y = Bx$ by equation (3.33), thus x would be an eigenvector as shown above, opposing the lemma assumption.
- $a = 0$ and $b \neq 0$, hence by equation (3.32) $||Bx - \kappa x|| = 0$, and x would once again be an eigenvector.
- $a = 0$ and $b = 0$. In this case, $f(x, y) = \mu(y)$ would be a global minimum, with y corresponding to the smallest eigenvalue by theorem 2.3.

We assume the more interesting case, in which y is not an eigenvector. Equation (3.33) then yields $a \neq 0$, so $g(x, y) = 0$, which we write as

$$||Bx - y|| = \gamma||Bx - \kappa x||. \tag{3.34}$$

We rewrite equation (3.32)

$$\begin{aligned}
&2a(B^2x - By - \gamma^2(B - \kappa I)(Bx - \kappa x)) - 2b(Bx - \kappa x) = 0 \\
\Leftrightarrow\ &B^2x - By - \gamma^2(B - \kappa I)(Bx - \kappa x) - \frac{b}{a}(Bx - \kappa x) = 0 \\
\Leftrightarrow\ &B^2x - By - \gamma^2 B(Bx - \kappa x) - (\frac{b}{a} - \gamma^2\kappa)(Bx - \kappa x) = 0 \\
\Leftrightarrow\ &B(Bx - y - \gamma^2(Bx - \kappa x)) = (\frac{b}{a} - \gamma^2\kappa)(Bx - \kappa x),
\end{aligned}$$

which (after substituting $c := \frac{b}{a} - \gamma^2\kappa$) yields

$$B(Bx - y - \gamma^2(Bx - \kappa x)) = c(Bx - \kappa x). \tag{3.35}$$

Let's have a look at equation (3.33). We divide by 2 and apply the inner product with

y, gaining

$$\begin{aligned}
&\left(\frac{By}{\|y\|^2}, y\right) - \left(\frac{\mu(y)y}{\|y\|^2}, y\right) + a(y - Bx, y) = 0 \\
\Leftrightarrow\ &\frac{1}{\|y\|^2}(y, By) - \frac{1}{\|y\|^2}\mu(y)(y, y) + a(y - Bx, y) = 0 \\
\Leftrightarrow\ &\frac{1}{\|y\|^2}(y, By) - \frac{1}{\|y\|^2}\frac{(y, By)}{(y, y)}(y, y) + a(y - Bx, y) = 0 \\
&\Leftrightarrow a(y - Bx, y) = 0,
\end{aligned}$$

which is equivalent to

$$(Bx - y, y) = 0, \tag{3.36}$$

since $a \neq 0$. Applying the inner product with $B^{-1}(Bx - \kappa x)$ to equation (3.35) gives

$$\begin{aligned}
c\|Bx - \kappa x\|_{B^{-1}}^2 &= (c(Bx - \kappa x), B^{-1}(Bx - \kappa x)) \\
&\overset{(3.35)}{=} (B(Bx - y - \gamma^2(Bx - \kappa x)), B^{-1}(Bx - \kappa x)) \\
&= (Bx - y - \gamma^2(Bx - \kappa x), Bx - \kappa x) \\
&= (Bx - y, Bx - \kappa x) - \gamma^2\|Bx - \kappa x\|^2 \\
&\overset{(3.34)}{=} (Bx - y, Bx - \kappa x) - \|Bx - y\|^2 \\
&= (Bx - y, Bx - \kappa x) - (Bx - y, Bx - y) \\
&\overset{(3.36)}{=} (Bx - y, -\kappa x) \\
&= -\kappa(Bx - y, x),
\end{aligned} \tag{3.37}$$

using the linearity of the inner product especially in the second last step. We do also rewrite equation (3.33) using $d := a\|y\|^2 - \mu(y)$:

$$\begin{aligned}
&\frac{2(By - \mu(y)y)}{(y, y)} + 2a(y - Bx) = 0 \\
\Leftrightarrow\ &By - \mu(y)y + a\|y\|^2(y - Bx) = 0 \\
&\Leftrightarrow By - \mu(y)Bx = \mu(y)y - \mu(y)Bx + a\|y\|^2(Bx - y) \\
&\Leftrightarrow By - \mu(y)Bx = (a\|y\|^2 - \mu(y))(Bx - y) \\
&\Leftrightarrow By - \mu(y)Bx = d(Bx - y) \\
&\Leftrightarrow B(y - \mu(y)x) = d(Bx - y).
\end{aligned}$$

Taking the inner product of the last equation[2] with $y - \mu(y)x$ results in

$$\begin{aligned}\|y - \mu(y)x\|_B^2 &= d(Bx - y, y - \mu(y)x)\\ &\overset{(3.36)}{=} -d\mu(y)(Bx - y, x),\end{aligned}$$

showing $-d\mu(y)(Bx - y, x) \geq 0$ due to the norm on the left-hand side. This yields

$$\begin{aligned}0 \leq -d\mu(y)(Bx - y, x) &\overset{3.37}{=} \frac{d\mu(y)c}{\kappa}\|Bx - \kappa x\|_{B^{-1}}^2\\ \overset{\mu(y),\kappa\geq 0}{\Leftrightarrow}\ 0 \leq -d\kappa(Bx - y, x) &= dc\|Bx - \kappa x\|_{B^{-1}}^2.\end{aligned}$$

Thus, $cd \geq 0$.
We multilpy (3.35) by B and obtain

$$\begin{aligned}&B^2(Bx - y - \gamma^2(Bx - \kappa x)) = cB(Bx - \kappa x)\\ \Leftrightarrow\ &B^3x - B^2y - \gamma^2B^3x + \gamma^2\kappa B^2x - cB^2x + c\kappa Bx = 0\\ \Leftrightarrow\ &(1 - \gamma^2)B^3x - B^2y + (\gamma^2\kappa - c)B^2x + c\kappa Bx = 0.\end{aligned}$$

We multiply this polynomial by $d + \mu(y)$ and substitute according to equation (3.38), obtaining

$$\begin{aligned}&(1 - \gamma^2)B^2(B + dI)y - B^2y + (\gamma^2\kappa - c)B(B + dI)y + c\kappa(B + dI)y = 0\\ \Leftrightarrow\ &(1 - \gamma^2)B^3y + (d(1 - \gamma^2) - 1 + \gamma^2\kappa - c)B^2y + (d\gamma^2\kappa + c\kappa - cd)By + dc\kappa y = 0.\end{aligned}$$

We notice that $dc\kappa \geq 0$, since $\kappa > 0$ and $dc \geq 0$, and $1 - \gamma^2 > 0$, since $\gamma < 1$. As in the "$\gamma = 0$"-case, this polynomial has maximally two positive roots by Descartes' rule of signs ([7]). The argument is the same as before and we obtain that y is a linear combination of at most two (normalized) eigenvectors v_k, v_l. The same must hold for x, since $d + \mu(y) = a\|y\|^2 \neq 0$. Otherwise, equation (3.38) would yield

$$\begin{aligned}a\|y\|^2Bx &= (B + dI)y\\ \Leftrightarrow Bx &= c_kv_k + c_lv_l\\ \Leftrightarrow \alpha_k\mu_kv_k + \alpha_l\mu_lv_l + \alpha_m\mu_mv_m &= c_kv_k + c_lv_l,\end{aligned}$$

with constants c_k, c_l, α_k, α_l and α_m. The linear independency of the eigenvectors yields $\mu_m = 0$, a contradiction. Therefore, $x, y \in span\{v_k, v_l\}$.

[2]Note that this equation is equivalent to another equation, that we'll use later:

$$(d + \mu(y))Bx = (B + dI)y \tag{3.38}$$

What is left, is to show the equation with the angles. We have

$$\begin{aligned}\cos^2 \angle(Bx, y) &= \left(\frac{(Bx, y)}{\|Bx\|\|y\|}\right)^2 \\ &\overset{(3.36)}{=} \left(\frac{(y, y)}{\|Bx\|\|y\|}\right)^2 \\ &= \left(\frac{\|y\|^2}{\|Bx\|\|y\|}\right)^2 \\ &= \frac{\|y\|^2}{\|Bx\|^2},\end{aligned}$$

which brings us[3]

$$\begin{aligned}\sin^2 \angle(Bx, y) &= 1 - \cos^2 \angle(Bx, y) \\ &= 1 - \frac{\|y\|^2}{\|Bx\|^2} \\ &= \frac{\|Bx^2\| - \|y\|^2}{\|Bx\|^2} \\ &\overset{(3.36)}{=} \frac{\|Bx - y\|^2}{\|Bx\|^2}.\end{aligned}$$

As the angle-range is $[0, \pi]$, we obtain $\sin \angle(Bx, y) = \frac{\|Bx-y\|}{\|Bx\|}$. Similarly,[4]

$$\begin{aligned}\cos^2 \angle(Bx, x) &\overset{\kappa \geq 0}{=} \cos^2 \angle(Bx, \kappa x) \\ &= \left(\frac{(Bx, \kappa x)}{\|Bx\|\|\kappa x\|}\right)^2 = \left(\frac{\kappa(Bx, x)}{\|Bx\|\|\kappa x\|}\right)^2 \\ &= \left(\frac{\kappa(\kappa x, x)}{\|Bx\|\|\kappa x\|}\right)^2 = \left(\frac{(\kappa x, \kappa x)}{\|Bx\|\|\kappa x\|}\right)^2 \\ &= \left(\frac{\|\kappa x\|^2}{\|Bx\|\|\kappa x\|}\right)^2 \\ &= \frac{\|\kappa x\|^2}{\|Bx\|^2},\end{aligned}$$

[3]The equation (3.36) states that y and $Bx - y$ are orthogonal, thus we can apply Pythagoras' theorem, $\|y\|^2 + \|Bx - y\|^2 = \|Bx - y + y\|^2 (= \|Bx\|^2)$, what we use here.

[4]We use $h(x, y) = 0$, which is equivalent to $(\kappa x, x) = (Bx, x)$ by properties of the inner product, and B. Furthermore, $(\kappa x, \kappa x) = (Bx, \kappa x)$, thus $(Bx - \kappa x, \kappa x) = 0$, allowing us once more to apply Pythagoras' theorem on $Bx - \kappa x$, and κx.

and further

$$\begin{aligned}\sin^2 \angle(Bx, \kappa x) &= 1 - \cos^2 \angle(Bx, \kappa x) \\ &= \frac{\|Bx\|^2 - \|\kappa x\|^2}{\|Bx\|^2} \\ &= \frac{\|Bx - \kappa x\|^2}{\|Bx\|^2}.\end{aligned}$$

This yields

$$\sin \angle(Bx, x) = \frac{\|Bx - \kappa x\|}{\|Bx\|} \overset{(3.34)}{=} \frac{\|Bx - y\|}{\gamma \|Bx\|} = \frac{1}{\gamma} \sin \angle(Bx, y).$$

□

The last piece of the proof for theorem 3.1 missing is the proof for lemma 3.5, which we will now present.

Lemma 3.5, proof. Without any constraint we can represent x, y, and Bx by $u := c(1, \alpha)^T$, $v := d(1, \beta)^T$, and $w := c(\mu_k, \alpha\mu_l)^T$, when $x = cv_k + c\alpha v_l$, et cetera, where $\{v_k, v_l\}$ are the orthonormal eigenvectors corresponding to μ_k and μ_l.
The orthonormality yields

$$\|x\|^2 = c^2\|v_k\|^2 + c^2\alpha\|v_l\|^2 = c^2 + c^2\alpha^2 = \|u\|^2,$$

and similarly $\|y\| = \|v\|$, and $\|Bx\| = \|w\|$. Furthermore,

$$\begin{aligned}(Bx, y) &= (\mu_k c v_k + \mu_l c\alpha v_l, d v_k + d\beta v_l) = \mu_k cd(v_k, v_k) + \mu_l c\alpha d\beta (v_l, v_l) \\ &= \mu_k cd + \mu_l c\alpha d\beta = (w, v).\end{aligned}$$

By the definition of arccos, the angles between the vectors are pairwise equal and we obtain

$$\sin \angle(w, v) = \sin \angle(Bx, y) \overset{Lemma3.4}{=} \gamma \sin \angle(Bx, x) = \gamma \sin \angle(w, u).$$

In a three-dimensional space, it holds that $\sin \angle(a, b) = \frac{\|a \times b\|}{\|a\|\|b\|}$, we use this to rewrite the last equation by adding an artificial dimension:

$$\frac{\left\| \begin{bmatrix} w \\ 0 \end{bmatrix} \times \begin{bmatrix} v \\ 0 \end{bmatrix} \right\|}{\|w\|\|v\|} = \sin \angle(w, v) \overset{Lemma3.4}{=} \gamma \sin \angle(w, u) = \gamma \frac{\left\| \begin{bmatrix} w \\ 0 \end{bmatrix} \times \begin{bmatrix} u \\ 0 \end{bmatrix} \right\|}{\|w\|\|u\|}. \tag{3.39}$$

We consider the norms of the cross products

$$\left\| \begin{bmatrix} w \\ 0 \end{bmatrix} \times \begin{bmatrix} v \\ 0 \end{bmatrix} \right\| = \left\| \begin{pmatrix} 0 \\ 0 \\ cd\mu_k\beta - cd\alpha\mu_l \end{pmatrix} \right\| = |cd||\mu_k\beta - \alpha\mu_l| \text{ and}$$
$$\left\| \begin{bmatrix} w \\ 0 \end{bmatrix} \times \begin{bmatrix} u \\ 0 \end{bmatrix} \right\| = \left\| \begin{pmatrix} 0 \\ 0 \\ c^2\mu_k\alpha - c^2\alpha\mu_l \end{pmatrix} \right\| = c^2|\mu_k\alpha - \alpha\mu_l|,$$

which we substitute in equation (3.39). Solving for γ^2 yields

$$\begin{aligned}
\gamma^2 &= \left(\frac{\left\| \begin{pmatrix} w \\ 0 \end{pmatrix} \times \begin{pmatrix} v \\ 0 \end{pmatrix} \right\| \|w\|\|u\|}{\left\| \begin{pmatrix} w \\ 0 \end{pmatrix} \times \begin{pmatrix} u \\ 0 \end{pmatrix} \right\| \|w\|\|v\|} \right)^2 \\
&= \frac{c^2d^2(\mu_k\beta - \alpha\mu_l)^2\|u\|^2}{c^4(\mu_k\alpha - \alpha\mu_l)^2\|v\|^2} \\
&= \frac{d^2(\mu_k\beta - \alpha\mu_l)^2c^2(1+\alpha^2)}{c^2(\mu_k\alpha - \alpha\mu_l)^2d^2(1+\beta^2)} \\
&= \frac{(\mu_k\beta - \alpha\mu_l)^2(1+\alpha^2)}{(\mu_k\alpha - \alpha\mu_l)^2(1+\beta^2)}.
\end{aligned}$$

Since c and d are arbitrary real numbers, we can set $c = d = 1$ without loss of generality. As we derived equation (3.16), we can now derive a similar equation for x and y.

$$\alpha^2 = \frac{\mu_k - \mu(x)}{\mu(x) - \mu_l} \quad \text{and} \tag{3.40}$$
$$\beta^2 = \frac{\mu_k - \mu(y)}{\mu(y) - \mu_l}. \tag{3.41}$$

We know $\mu(y) \geq \mu(x) > \mu_l$, and therefore $\frac{\mu(y)-\mu_l}{\mu(x)-\mu_l} \geq 1$. This helps us to simplify

$$\frac{1+\alpha^2}{1+\beta^2} = \frac{1 + \frac{\mu_k - \mu(x)}{\mu(x)-\mu_l}}{1 + \frac{\mu_k - \mu(y)}{\mu(y)-\mu_l}} = \frac{(\mu(x) - \mu_l + \mu_k - \mu(x))(\mu(y) - \mu_l)}{(\mu(y) - \mu_l + \mu_k - \mu(y))(\mu(x) - \mu_l)} \geq 1,$$

bringing a lower bound for γ^2:

$$\gamma^2 \geq \frac{(\mu_k\beta - \alpha\mu_l)^2}{(\mu_k\alpha - \alpha\mu_l)^2}.$$

This can be used to state

$$\begin{aligned}
&(\mu_k\alpha - \alpha\mu_l)^2\gamma^2 \geq (\mu_k\beta - \alpha\mu_l)^2 \\
\Leftrightarrow\ &|\mu_k\alpha - \alpha\mu_l|\gamma \geq |\mu_k\beta - \alpha\mu_l| \\
\Leftrightarrow\ &|\mu_k\alpha|\left|1 - \frac{\mu_l}{\mu_k}\right|\gamma \geq |\mu_k\alpha|\left|\frac{\beta}{\alpha} - \frac{\mu_l}{\mu_k}\right|,
\end{aligned}$$

to obtain

$$\Leftrightarrow \left|1 - \frac{\mu_l}{\mu_k}\right|\gamma \geq \left|\frac{\beta}{\alpha} - \frac{\mu_l}{\mu_k}\right|. \tag{3.42}$$

We now have everything we need, we only have to put all the pieces together!

$$\begin{aligned}
\frac{\mu(x) - \mu_l}{\mu_k - \mu(x)}\frac{\mu_k - \mu(y)}{\mu(y) - \mu_l} &\overset{(3.41),(3.40)}{=} \frac{\beta^2}{\alpha^2} \\
&= \left|\frac{\beta}{\alpha}\right|^2 \\
&\leq \left(\left|\frac{\beta}{\alpha} - \frac{\mu_l}{\mu_k}\right| + \left|\frac{\mu_l}{\mu_k}\right|\right)^2 \\
&\overset{(3.42)}{\leq} \left(\gamma\left(1 - \frac{\mu_l}{\mu_k}\right) + \frac{\mu_l}{\mu_k}\right)^2 \\
&= \left(\gamma + (1-\gamma)\frac{\mu_l}{\mu_k}\right)^2,
\end{aligned}$$

where we used $0 < \frac{\mu_l}{\mu_k} < 1$ in the second last inequality, allowing us to drop the $|.|$. □

4 Computational experiments

After all that theory, it's time to look at some practical stuff. This section analyses the MATLAB-code written by the author. It can be found at the end of this thesis in section 7.

4.1 Large, sparse matrices

One of the problems, which turned out to be among the most difficult, was to construct large and sparse matrices A and B. One option used is to firstly define a random vector, which contains n values, that will later become the eigenvalues for a matrix. This vector was then used to define a sparse matrix by the command *sprandsym*. Unfortunately, this command is very time consuming for large n. An alternative used was to simply define a sparse tridiagonal matrix, however the results are very specific and doubtedly can be generalized.

4.2 Preconditioning

The preconditioner that appeared to be most promising from the literature was an inverse approximation by an incomplete cholesky factorization. The calculation of this preconditioner was achieved by the MATLAB command *ichol*. It consumed more time than all other problems combined. For small problems, i.e. $n < 10000$, T was chosen as a perturbation of the inverse, i.e. $T = (1 + \varepsilon)A^{-1}$, since it allowed to experiment with different values for γ. This choice of T was only used in function 7.4 to test the convergence in dependence on γ.

4.3 Calculating the bound γ

The bound γ was calculated in function 7.4. It was achieved by transforming the generalized EVP according to section 3.1, before calculating $\gamma = ||I - T_A||$. An alternative approach would obviously be to calculate the largest singular values of $I - T_A$ by vector iterations, but the author found his way more elegant.

4.4 Testing the theory

As the bound γ was calculated in 7.4, it could be seen that using *ichol* leads to a very good approximate, i.e. $T \approx A^{-1}$, which lead to fast convergence of the PINVIT-methods. The choice of T as a perturbation to A^{-1} resulted in γ being proportional to the parameter ε, precisely $\gamma = \varepsilon$. To test the theory, the largest eigenvalue μ was calculated by the MATLAB command *eigs*. Different test-setups have been considered:

(1) For fixed matrices A and B, and different values of gamma, the steps, until the PINVIT method 7.8 converged (with an error to μ of less than 10^{-8}), were counted. Thus, T was chosen as a perturbation to A^{-1}.

(2) The same has been done for all the methods to see which method converges fastest, where random matrices were calculated and T was calculated by an incomplete cholesky factorization.

(3) For a fixed step size, the relative error between the largest eigenvalues calculated by the MATLAB-command and the different methods has been analyzed.

The first test-setup was executed 100 times. Every time random matrices A and B were calculated, before the eigenvalue problem was solved using five different preconditioners. The medians of the needed number of steps have been calculated for each preconditioner. The results are presented in the following table.

Problem size n	γ	Average number of steps needed until convergence
100	0.1	50.5
	0.5	41.7
	0.9	91.7
	0.99	684.7
	0.999	4751.8
1000	0.1	43.9
	0.5	37.6
	0.9	101.4
	0.99	801.2
	0.999	6800.8

For $\gamma \geq 1$, the method didn't converge once when tested. For γ close to 1, the number of steps until convergence needed, grows rapidly. However, for smaller values of γ, the number of steps needed were firstly small and secondly not corresponding to γ, consider that for $\gamma = 0.1$ and $\gamma = 0.5$ roughly the same number of steps were needed. In fact, $\gamma = 0.5$ yielded the fewest steps needed for the γ tested.

The second test-setup was also executed 100 times. Random matrices have been calculated, T was chosen as an incomplete cholesky factorization. The standard PINVIT method, and both PINVIT2 and PINVIT3 were used to calculate the eigenvalues, all using the same preconditioner. The average number of steps needed can be found in the

following chart.

n	Steps PINVIT	Time PINVIT	Steps PINVIT2	Time PINVIT2	Steps PINVIT3	Time PINVIT3
1000	63.23	0.1993 s	18.32	0.0969 s	9.68	0.0548 s
3000	53.97	1.3013 s	35.69	0.9506 s	10.51	0.3085 s

One thing, that has to be mentioned and explained is, that PINVIT2 didn't converge at times. This is due to it "being stuck" at another eigenvector. If at some point, the current iterate x is an eigenvector, μ is the corresponding eigenvalue and thus $gradmu = 0$ in method 7.9. For the chart, only those examples were used where PINVIT2 did converge.

As the others, the third test-setup was executed 100 times for different step sizes, but for the same matrices. The preconditioner was again calculated by an incomplete cholesky factorization. The problem size was chosen as $n = 3000$, the largest eigenvalue was in average 3428. The relative error was calculated, the results are presented in the following chart.

Number of steps	Relative error PINVIT	Relative error PINVIT2	Relative error PINVIT3
10	62.55 %	1.89 %	1.66 %
20	62.64 %	1.75 %	1.63 %
30	62.64 %	1.67 %	1.62 %

It is interesting that the error of PINVIT becomes larger between 10 and 20 steps. This is explainable since we choose a fixed step size, allowing the Rayleigh quotient to become larger in some iterations. However, as shown before, the method still converges, as could be seen before. It is still advisable to use PINVIT2 or PINVIT3 as these methods use an optimal step size.

5 The LOBPCG method

It is often not enough to calculate the largest or smallest eigenvalue only, but one would rather be interested in the k largest eigenvalues. This is achieved by the *LOBPCG* method. Both an implementation and a test-setup are provided at the end of this thesis, see 7.18 and 7.3, respectively. The method is similar to PINVIT3, but calculates and stores more eigenvector and eigenvalue iterates at the same time. They are all thrown into a Rayleigh-Ritz step, where the largest k Ritz-values are approximates for the largest k eigenvalues. For each vector, the Rayleigh quotient and gradients of it are calculated, before using the old iterates $x_{j,n-1}$, the current iterate $x_{j,n}$ and the gradient $\nabla\lambda(x_{j,n})$ to get a new iterate $\tilde{x}_{j,n+1}$ by the Rayleigh-Ritz method. Another step of the Rayleigh-Ritz method is performed on all the new iterates $\tilde{x}_{1,n+1}$, ..., $\tilde{x}_{k,n+1}$. The resulting vectors are the new iterates $x_{j,n+1}$.
It is obvious, that theorem 3.1 holds for the iterate corresponding to the largest Rayleigh quotient calculated in each step. A convergence analysis for the other iterates, which correspond to the second, third, ..., k-th largest eigenvalues is presented in [10].

6 Conclusion and outlook

A convergence analysis for the preconditioned iterative eigensolver PINVIT has been made. Furthermore, the theory was tested and different iterative eigensolvers were compared. It could be seen that iterative eigensolvers implemented using saving techniques for sparse matrices were indeed a lot faster than methods that didn't use those techniques. The comparison of PINVIT, PINVIT2, and PINVIT3 yielded - as expected - that PINVIT needs more steps than PINVIT2 which again needed more steps than PINVIT3. The same result holds for the time needed until convergence in the last section. It has become clear that the preconditioner has a huge impact on both the total time needed to solve a given eigenvalue problem and the steps needed by the iterative eigensolver. Finding the best preconditioner is therefore a key part when working with preconditioned eigensolvers and can be elaborated more than done in this thesis.

A long term aim is to obtain a sharp convergence bound for the LOBPCG method, first steps to this bound were made by the authors of [10]. At the moment, a convergence analysis for PINVIT3 is neither known. However, the convergence analysis for PINVIT2 is structurally similar to that from PINVIT, as shown in this thesis. For more complicated methods, such as the Jacobi-Davidson method ([14]), a convergence analysis is yet to be found.

7 Matlab code

The author of this thesis has implemented different codes for comparing different eigensolvers, and for testing the convergence analysis made in section 3.

Listing 7.1: Main programm to compare different eigensolvers for the standard EVP

```
function comparing_iterative_eigensolvers
clc
tic
%n as a square number (usually the case for grid on squares)
n=1000;

%% Initialization for standard EVP, with a sparse matrix B (as above),
%% for very large n

%Needed: Starting vector x, matrix B, preconditioner T
x=ones(n,1);
for k=1:n/2
    x(2*k)=0;
end
x=x/norm(x);
t1=toc
tic
rc=100*rand(n,1);
B=sprandsym(n,log(n)/n,rc);
t2=toc
tic
I=speye(n);
T=1/2*I;

%% calling different iterative eigensolvers, note that A=eye(n)

% number of steps
s=1000;
t5=toc
tic
mumax=eigs(B,1)
t6=toc
tic
[mu1,x1]=PINVIT(x,B,T,s,n,mumax);
t7=toc
tic
[mu2,x2]=PINVIT2(x,B,T,s,n,mumax);
t8=toc
tic
```

```
[mu3,x3]=PINVIT3(x,B,T,s,n,mumax);
t9=toc

mumax-mu1
mumax-mu2
mumax-mu3

end
```

Listing 7.2: Main programm to compare different eigensolvers for the generalized EVP

```
function comparing_gen_iterative_eigensolvers
clc
tic
%% Initialisation for the second test-setup
% tgenPINVIT=0;
% tgenPINVIT2=0;
% tgenPINVIT3=0;
% step=zeros(3,1);

%% Initialisation for the third test-setup
relerror10PINVIT=0;
relerror10PINVIT2=0;
relerror10PINVIT3=0;
relerror20PINVIT=0;
relerror20PINVIT2=0;
relerror20PINVIT3=0;
relerror30PINVIT=0;
relerror30PINVIT2=0;
relerror30PINVIT3=0;
avgmumax=0;

% problem size n
n=3000;

% Needed: Starting vector x, matrices A and B, preconditioner T
x=ones(n,1);
for k=1:n/2
    x(2*k)=0;
end
x=x/norm(x);

for t=1:25
%% Initialization for generalized EVP, with sparse matrices A and B,
%% for very large n - first possibility
t1=toc
tic
rc=100*rand(n,1);
B=sprandsym(n,log(n)/n,rc);
t2=toc
tic
rc=100*rand(n,1);
A=sprandsym(n,log(n)/n,rc);
t3=toc
```

```
tic
opts.type = 'ict';
L1=ichol(A,opts);
L2=inv(L1);
T=L2'*L2;
t4=toc

%% Initialization for generalized EVP, with sparse matrices A and B,
%% for very large n - second possibility
% tic
% D = sparse(1:n,1:n,2*ones(1,n),n,n);
% E = sparse(2:n,1:n-1,-1*ones(1,n-1),n,n);
% A = E+D+E';
% B = sparse(1:n,1:n,2*ones(1,n),n,n);
% t3=toc
% tic
% opts.type = 'ict';
% L1=ichol(A,opts);
% L2=inv(L1);
% T=L2'*L2;
% t4=toc

%% calling different iterative eigensolvers for the generalized EVP

% number of steps, needed for third test-setup
s=10;

t5=toc
tic
mumax=eigs(B,A,1);
t6=toc
tic
[mu1,x1,k]=genPINVIT(x,A,B,T,s,n,mumax);
t7=toc
tic
[mu2,x2,l]=genPINVIT2(x,A,B,T,s,n,mumax);
t8=toc
tic
[mu3,x3,m]=genPINVIT3(x,A,B,T,s,n,mumax);
t9=toc

%% Values calculated for the second test-setup
% tgenPINVIT=tgenPINVIT+t7;
% tgenPINVIT2=tgenPINVIT2+t8;
% tgenPINVIT3=tgenPINVIT3+t9;
% step(1)=step(1)+k;
% step(2)=step(2)+l;
% step(3)=step(3)+m;

%% first couple of values calculated for the third test-setup
relerror10PINVIT=relerror10PINVIT+(mumax-mu1)/mumax;
relerror10PINVIT2=relerror10PINVIT2+(mumax-mu2)/mumax;
relerror10PINVIT3=relerror10PINVIT3+(mumax-mu3)/mumax;
avgmumax=avgmumax+mumax;
```

```
98
99
100 %% 20 steps, needed for third test-setup
101 s=20;
102 tic
103 mumax=eigs(B,A,1);
104 t6=toc
105 tic
106 [mu1,x1,k]=genPINVIT(x,A,B,T,s,n,mumax);
107 t7=toc
108 tic
109 [mu2,x2,l]=genPINVIT2(x,A,B,T,s,n,mumax);
110 t8=toc
111 tic
112 [mu3,x3,m]=genPINVIT3(x,A,B,T,s,n,mumax);
113 t9=toc
114
115 % second couple of values calculated for the third test-setup
116 relerror20PINVIT=relerror20PINVIT+(mumax-mu1)/mumax;
117 relerror20PINVIT2=relerror20PINVIT2+(mumax-mu2)/mumax;
118 relerror20PINVIT3=relerror20PINVIT3+(mumax-mu3)/mumax;
119
120 %% 30 steps, needed for third test-setup
121 s=30;
122 tic
123 [mu1,x1,k]=genPINVIT(x,A,B,T,s,n,mumax);
124 t7=toc
125 tic
126 [mu2,x2,l]=genPINVIT2(x,A,B,T,s,n,mumax);
127 t8=toc
128 tic
129 [mu3,x3,m]=genPINVIT3(x,A,B,T,s,n,mumax);
130 t9=toc
131
132 % third couple of values calculated for the third test-setup
133 relerror30PINVIT=relerror30PINVIT+(mumax-mu1)/mumax;
134 relerror30PINVIT2=relerror30PINVIT2+(mumax-mu2)/mumax;
135 relerror30PINVIT3=relerror30PINVIT3+(mumax-mu3)/mumax;
136
137
138
139 end
140 %% Values calculated for the second test-setup
141 % tgenPINVIT
142 % tgenPINVIT2
143 % tgenPINVIT3
144 % step=step*1/10
145
146 %% Values calculated for the third test-setup
147 relerror10PINVIT=relerror10PINVIT/25
148 relerror10PINVIT2=relerror10PINVIT2/25
149 relerror10PINVIT3=relerror10PINVIT3/25
150 relerror20PINVIT=relerror20PINVIT/25
151 relerror20PINVIT2=relerror20PINVIT2/25
```

```
relerror20PINVIT3=relerror20PINVIT3/25
relerror30PINVIT=relerror30PINVIT/25
relerror30PINVIT2=relerror30PINVIT2/25
relerror30PINVIT3=relerror30PINVIT3/25
avgmumax=avgmumax/25


end
```

Listing 7.3: Main programm to test the LOBPCG method for the generalized EVP

```
function testing_LOBPCG
clc
tic

% Choose size n and number of eigenvalues to be computed k
n=8000;
k=5;


%% Initialization for generalized EVP, with sparse matrices A and B,
%% for very large n

% Needed: Starting vector x_1, ..., x_k (stored in X),
% matrices A and B, preconditioner T

% S
X=zeros(n,k);
for i=1:k
    j=i;
    while j<=n
        X(j,i)=1;
        j=j+k;
    end
    X(1:n,i)=X(1:n,i)/norm(X(1:n,i));
end
t1=toc

% B
tic
rc=100*rand(n,1);
B=sprandsym(n,log(n)/n,rc);
t2=toc

% A
tic
rc=100*rand(n,1);
A=sprandsym(n,log(n)/n,rc);
t3=toc

% T using incomplete Cholesky factorization
tic
opts.type = 'ict';
L1=ichol(A,opts);
```

```
L2=inv(L1);
T=L2'*L2;
t4=toc

% calculating eigenvalues by MATLAB routine
tic
eigen=eigs(B,A,k);
evmax=max(eigen)
t5=toc

%% Calculating the k largest eigenvalues using the LOBPCG method
s=3; % Step size
tic
[eigenvalues,eigenvectors]=LOBPCG(X,k,A,B,T,s,n,evmax);
t6=toc

tic

for i=1:k
    [a,ai]=max(eigen);
    [b,bi]=max(eigenvalues);
    norm(a-b)
    eigen(ai)=0;
    eigenvalues(bi)=0;
end
t7=toc

end
```

Listing 7.4: Main programm for testing the convergence analysis

```
function convergence_analysis_PINVIT
clc
epsilon(1)=1;
epsilon(2)=5;
epsilon(3)=9;
epsilon(4)=9.9;
epsilon(5)=9.99;
step=zeros(5,1);
% Problem size n
n=5000;

for t=1:100
%% Initialization for generalized EVP
% Needed: Starting vector x, matrices A and B, preconditioner T

x=ones(n,1);
for k=1:n/2
    x(2*k)=0;
end
x=x/norm(x);


%% A and B calculated as random matrices
```

```
tic
rc1=100*rand(n,1);
A=sprandsym(n,100/n,rc1);
rc2=100*rand(n,1);
B=sprandsym(n,100/n,rc2);
t1=toc;

%% Preconditioner calculated using inverse of incomplete cholesky
%% factorization of A
% tic
% opts.type = 'ict';
% L1=ichol(A,opts);
% L2=inv(L1);
% T=L2'*L2;
% t2=toc
%% Preconditioner calculated as perturbation of inv(A)
tic
for m=1:5
epsilon1=epsilon(m)*1e-1;
T=(1+epsilon1)*inv(A);
t2=toc;



%% transformation the problem according to thesis, section 3, to
    calculate
%% gamma
% tic
% Af=sqrtm(full(A));
% T1=Af*T*Af;
% gamma(t)=norm(eye(n)-T1);
% t3=toc;

%% calling iterative eigensolver

% number of steps
s=1000;
tic
mumax=eigs(B,A,1);
t4=toc;
tic
[mu1,x1,k]=genPINVIT(x,A,B,T,s,n,mumax);
t5=toc;
step(m)=step(m)+k;
mumax-mu1;
end
end
step=step*1/100
end
```

7.0.1 Implementations of different preconditoned iterative eigensolvers

Listing 7.5: Simplest iterative method

```
function [mu,x]=PINVIT(x,B,T,s,n,ev)
t=0;
mu=0;
k=1;
%% Iteration (choose one of the loops)
% for k=1:s
while ev-mu>10e-8
    b=B*x;
    mu=(x'*b)/(x'*x);
    gradmu=1/mu*T*(b-mu*x);

    x2=x+gradmu;
%% Using Armijo-Goldstein conditions
%      mu_new=x2'*(B*x2)/(x2'*x2);
%      alpha=1;
%      while (mu>=mu_new)
%         alpha=alpha/2;
%         x2=x+alpha*gradmu;
%         mu_new=(x2'*(B*x2))/(x2'*x2);
%         if (alpha<10e-20)
%             t=1;
%             break
%         end
%      end
%      if(t==1)
%          break
%      end
    x=x2;
    k=k+1;
end
k
end
```

Listing 7.6: PINVIT2, each step using an optimization in a two-dimensional subspace

```
function [mu,x]=PINVIT2(x,B,T,s,n,ev)
I=speye(n);
mu=0;

%% Iteration (choose one of the loops)
% for k=1:s
while ev-mu>10e-4

    b=B*x;
    mu=x'*b/(x'*x);
    gradmu=T*(b-mu*x);
    gradmu=gradmu/norm(gradmu);

    tau=rayleighritz2(n,B,x,gradmu);
```

```
15
16
17      x=x+tau*gradmu;
18      x=x/norm(x);
19  end
20
21  end
```

Listing 7.7: PINVIT3, each step using an optimization in a three-dimensional subspace

```
1   function [mu,x]=PINVIT3(x,B,T,s,n,ev)
2   I=speye(n);
3   mu=0;
4
5   %% first step
6   b=B*x;
7   mu=x'*b/(x'*x);
8   gradmu=T*(b-mu*x);
9   gradmu=gradmu/norm(gradmu);
10  tau=rayleighritz2(n,B,x,gradmu);
11  p=x;
12  x=x+tau*gradmu;
13  x=x/norm(x);
14
15  %% Iteration (choose one of the loops)
16  % for k=1:s
17  while ev-mu>10e-4
18
19      b=B*x;
20      mu=x'*b/(x'*x);
21      gradmu=T*(b-mu*x);
22      gradmu=gradmu/norm(gradmu);
23
24      [tau,gamma]=rayleighritz3(n,B,x,gradmu,p);
25
26      x_old=x;
27      x=x+tau*gradmu+gamma*p;
28      x=x/norm(x);
29      p=T*(b-mu*x_old);
30  end
31
32  end
```

Listing 7.8: Simplest iterative method for the generalized EVP, for which a convergence analysis has been made in this thesis

```
1   function [mu,x,k]=genPINVIT(x,A,B,T,s,n,ev)
2   mu=0;
3   k=0;
4   %% Iteration (choose one of the loops)
5   for k=1:s % for third test-setup
6   % while ev-mu>1e-8
7       a=A*x;
8       b=B*x;
```

```
    mu=(x'*b)/(x'*a);
    gradmu=1/mu*T*(b-mu*a);

    x=x+gradmu;
    k=k+1;
end
k;
end
```

Listing 7.9: PINVIT2 for the generalized EVP, each step using an optimization in a two-dimensional subspace

```
function [mu,x,k]=genPINVIT2(x,A,B,T,s,n,ev)
mu=0;
k=0;
%% Iteration (choose one of the loops)
for k=1:s %for the third test-setup
% while ev-mu>1e-8
    a=A*x;
    b=B*x;
    mu=x'*b/(x'*a);
    gradmu=T*(b-mu*a);

    % Orthonormalize gradmu and x
    gradmu=gradmu-(gradmu'*x)*x;
    gradmu=gradmu/norm(gradmu);

    x=genrayleighritz2(n,A,B,x,gradmu);
  %  x=x/norm(x);
    k=k+1;
end
k;
end
```

Listing 7.10: PINVIT3 for the generalized EVP, each step using an optimization in a three-dimensional subspace

```
function [mu,x,k]=genPINVIT3(x,A,B,T,s,n,ev)
k=1;
%% first step
a=A*x;
b=B*x;
mu=x'*b/(x'*a);
gradmu=T*(b-mu*a);
gradmu=gradmu/norm(gradmu);
p=x;
x=genrayleighritz2(n,A,B,x,gradmu);
x=x/norm(x);

%% Iteration (choose one of the loops)
for k=1:s %for third test-setup
% while ev-mu>1e-8
    a=A*x;
    b=B*x;
```

```
    mu=x'*b/(x'*a);
    gradmu=T*(b-mu*a);
    gradmu=gradmu/norm(gradmu);

    x_old=x;
    x=genrayleighritz3(n,A,B,gradmu,x,p);

    x=x/norm(x);
    p=x_old-(x'*x_old)*x;
    p=p/norm(p);
    k=k+1;
end
k;
end
```

7.0.2 Implementations of the Rayleigh-Ritz method for different subspaces

Listing 7.11: Rayleigh-Ritz-method for two-dimensional subspaces

```
function [ tau] = rayleighritz2(n,B,v1,v2)
S(1:n,1)=v1;
S(1:n,2)=v2;

% calculate Ritz values
[U,D]=eig(S'*B*S);

% calculate corresponding vector
[x,i]=max(diag(D));

u=U(1:2,i);
u=u/u(1);
tau=u(2);
end
```

Listing 7.12: Rayleigh-Ritz-method for three-dimensional subspaces

```
function [tau,gamma] = rayleighritz3(n,B,v1,v2,v3)
S(1:n,1)=v1;
S(1:n,2)=v2;
S(1:n,3)=v3;

% calculate Ritz values
[U,D]=eig(S'*B*S);

% calculate corresponding vector
[x,i]=max(diag(D));

u=U(1:3,i);
u=u/u(1);
tau=u(2);
gamma=u(3);
```

```
end
```

Listing 7.13: generalized Rayleigh-Ritz-method for 2-dimensional subspaces

```
function [ x] = genrayleighritz2(n,A,B,v1,v2)
S(1:n,1)=v1;
S(1:n,2)=v2;

% calculate Ritz values
[U,D]=eig(S'*B*S,S'*A*S);

% calculate corresponding vector
[x,i]=max(diag(D));

x=S*U(1:2,i);

end
```

Listing 7.14: generalized Rayleigh-Ritz-method for 3-dimensional subspaces

```
function [x] = genrayleighritz3(n,A,B,v1,v2,v3)
S(1:n,1)=v1;
S(1:n,2)=v2;
S(1:n,3)=v3;

% calculate Ritz values
[U,D]=eig(S'*B*S,S'*A*S);

% calculate corresponding vector
[x,i]=max(diag(D));

x=S*U(1:3,i);
end
```

Listing 7.15: generalized Rayleigh-Ritz-method for $2k$-dimensional subspaces

```
function [X,P] = genrayleighritz2k(n,A,B,S1,S2,k)
S(1:n,1:k)=S1;
S(1:n,(k+1):(2*k))=S2;

% calculate Ritz values
[U,D]=eig(S'*B*S,S'*A*S);


% calculate corresponding vector
for j=1:k
    [x,i]=max(diag(D));

    D(j,j)=0;
    P(1:n,j)=S1(1:n,i);
    X(1:n,j)=S*U(1:2*k,i);
    X(1:n,j)=X(1:n,j)/norm(X(1:n,j));
end

end
```

Listing 7.16: generalized Rayleigh-Ritz-method for $3k$-dimensional subspaces

```
function [X,P] = genrayleighritz3k(n,A,B,S1,S2,S3,k)
S(1:n,1:k)=S1;
S(1:n,(k+1):(2*k))=S2;
S(1:n,(2*k+1):(3*k))=S3;

% calculate Ritz values
[U,D]=eig(S'*B*S,S'*A*S);

% calculate corresponding vector
for j=1:k
    [x,i]=max(diag(D));

    D(i,i)=0;

    X(1:n,j)=S*U(1:3*k,i);
    X(1:n,j)=X(1:n,j)/norm(X(1:n,j));
    P(1:n,j)=S1*U(1:k,i)+S3*U(2*k+1:3*k,i);
    P(1:n,j)=P(1:n,j)/norm(P(1:n,j));
end

end
```

Listing 7.17: generalized Rayleigh-Ritz-method for k-dimensional subspaces

```
function X = genrayleighritzX(n,A,B,X,k)
S=X;
% calculate Ritz values
[U,D]=eig(S'*B*S,S'*A*S);

% sort vectors
for i=1:k
    [x,j]=max(diag(D));
    X(1:n,i)=S*U(1:k,j);
    D(j,j)=0;
end

end
```

7.0.3 LOBPCG implementations

The implementation of the LOBPCG methods is according to the pseudo code offered in [10].

Listing 7.18: Locally Optimal Block Preconditioned Conjugate Gradient

```
function [mu,X]=LOBPCG(X,k,A,B,T,s,n,ev)
count=1;
%% first step
a=A*X;
b=B*X;
```

```
for j=1:k
    mu(j)=X(1:n,j)'*b(1:n,j)/(X(1:n,j)'*a(1:n,j));
    gradmu(1:n,j)=T*(b(1:n,j)-mu(j)*a(1:n,j));
    gradmu(1:n,j)=gradmu(1:n,j)/norm(gradmu(1:n,j));
end
[X,P] = genrayleighritz2k(n,A,B,gradmu,X,k);

%% Iteration (choose one of the loops)
for t=1:s
% while ev-mu(1)>10e-4
    a=A*X;
    b=B*X;
    for j=1:k
        mu(j)=X(1:n,j)'*b(1:n,j)/(X(1:n,j)'*a(1:n,j));
        gradmu(1:n,j)=T*(b(1:n,j)-mu(j)*a(1:n,j));
        gradmu(1:n,j)=gradmu(1:n,j)/norm(gradmu(1:n,j));
    end
    [X,P]=genrayleighritz3k(n,A,B,gradmu,X,P,k);
    count=count+1;
end
count


end
```

Listing 7.19: Locally Optimal Block Preconditioned Conjugate Gradient, alternative version

```
function [mu,X]=LOBPCG2(X,k,A,B,T,s,n,ev)
count=1;
%% first step
a=A*X;
b=B*X;
for j=1:k
    mu(j)=X(1:n,j)'*b(1:n,j)/(X(1:n,j)'*a(1:n,j));
    gradmu(1:n,j)=T*(b(1:n,j)-mu(j)*a(1:n,j));
    gradmu(1:n,j)=gradmu(1:n,j)/norm(gradmu(1:n,j));
    tau=genrayleighritz2(n,A,B,X(1:n,j),gradmu(1:n,j));
    P(1:n,j)=X(1:n,j);
    X(1:n,j)=X(1:n,j)+tau*gradmu(1:n,j);
    X(1:n,j)=X(1:n,j)/norm(X(1:n,j));
end

%% Iteration (choose one of the loops)
for t=1:s
% while ev-mu(1)>10e-4
    a=A*X;
    b=B*X;
    for j=1:k
        mu(j)=X(1:n,j)'*b(1:n,j)/(X(1:n,j)'*a(1:n,j));
        gradmu(1:n,j)=T*(b(1:n,j)-mu(j)*a(1:n,j));
        gradmu(1:n,j)=gradmu(1:n,j)/norm(gradmu(1:n,j));

```

```
26          [tau,gamma]=genrayleighritz3(n,A,B,gradmu(1:n,j),X(1:n,j),P(1:n
    ,j));
27
28          X(1:n,j)=gradmu(1:n,j)+tau*X(1:n,j)+gamma*P(1:n,j);
29          norm(X(1:n,j))
30          X(1:n,j)=X(1:n,j)/norm(X(1:n,j));
31          P(1:n,j)=gradmu(1:n,j)+gamma*P(1:n,j);
32          P(1:n,j)=P(1:n,j)/norm(P(1:n,j));
33      end
34      X=genrayleighritzX(n,A,B,X,k);
35      for j=1:k
36          X(1:n,j)=X(1:n,j)/norm(X(1:n,j));
37      end
38
39      count=count+1;
40  end
41  count
42
43
44  end
```

Bibliography

[1] Merico E Argentati, Andrew V Knyazev, Klaus Neymeyr, Evgueni E Ovtchinnikov, and Ming Zhou. Convergence theory for preconditioned eigenvalue solvers in a nutshell. *Foundations of Computational Mathematics*, pages 1–15, 2014.

[2] Michele Benzi. Preconditioning techniques for large linear systems: a survey. *Journal of computational Physics*, 182(2):418–477, 2002.

[3] Michele Benzi, Carl D Meyer, and Miroslav Tuma. A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM Journal on Scientific Computing*, 17(5):1135–1149, 1996.

[4] Albrecht Beutelspacher. Lineare algebra. *Vieweg, Braunschweig*, 2003.

[5] Henricus Bouwmeester, Andrew Dougherty, and Andrew V Knyazev. Nonsymmetric preconditioning for conjugate gradient and steepest descent methods. *Procedia Computer Science*, 51:276–285, 2015.

[6] Are Magnus Bruaset. *A survey of preconditioned iterative methods*, volume 328. CRC Press, 1995.

[7] DR Curtiss. Recent extentions of descartes' rule of signs. *Annals of Mathematics*, pages 251–278, 1918.

[8] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013.

[9] Carl Geiger and Christian Kanzow. *Theorie und Numerik restringierter Optimierungsaufgaben*. Springer-Verlag, 2013.

[10] Andrew V Knyazev. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. *SIAM journal on scientific computing*, 23(2):517–541, 2001.

[11] Beresford N Parlett. *The symmetric eigenvalue problem*, volume 7. SIAM, 1980.

[12] Youcef Saad. *Numerical methods for large eigenvalue problems*, volume 158. SIAM, 1992.

[13] Yousef Saad. *Iterative methods for sparse linear systems*. Siam, 2003.

[14] Gerard LG Sleijpen and Henk A Van der Vorst. A jacobi–davidson iteration method for linear eigenvalue problems. *Siam Review*, 42(2):267–293, 2000.

[15] Ming Zhou. *Über Gradientenverfahren zur Lösung von Eigenwertproblemen elliptischer Differentialoperatoren*. PhD thesis, 2012.

Declaration of Authorship

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig angefertigt und ohne fremde Hilfe verfasst habe. Dazu habe ich keine außer den von mir angegebenen Hilfsmitteln und Quellen verwendet und die den benutzten Werken inhaltlich und wörtlich entnommenen Stellen habe ich als solche kenntlich gemacht.

I hereby certify that this thesis has been composed by me and is based on my own work, unless stated otherwise. No other person's work has been used without due acknowledgement in this thesis. All references and verbatim extracts have been quoted, and all sources of information, including graphs and data sets, have been specifically acknowledged.

Rostock, on the 16th August 2016

Torben Sell