SOLVING THE SYMMETRIC DEFINITE GENERALIZED EIGENVALUE PROBLEM

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Science and Engineering

December 2000

By
Philip Ieuan Davies
Department of Mathematics

Contents

List of Tables

List of Figures

Abstract

We consider some theoretical and numerical aspects of the generalized eigenvalue problem

$$Ax = \lambda Bx.$$

where $A, B \in \mathbb{C}^{n\times n}$. A common form of the problem is the symmetric positive definite case, where A and B are both real and symmetric and at least one of them is positive definite. This problem has many applications in science and engineering. We consider the difficulties associated with the computation of the solution. Ideally a method would be backward stable, efficient and able to exploit the special structure of the problem. Several methods for solving this problem are described, each of which meets at least one of these criteria. Of particular interest is the Cholesky method which converts the generalized eigenvalue problem, via a Cholesky factorization of the symmetric positive definite B, to a standard eigenvalue problem $Cy = \lambda y$ while maintaining the symmetry of the problem. Although this method exploits the structure of the problem it is potentially unstable. Provided that a stable eigensolver is used to solve the eigenvalue problem for C, standard error analysis says that the computed eigenvalues are exact for $A + \Delta A$ and $B + \Delta B$ with $\max(\|\Delta A\|_2/\|A\|_2, \|\Delta B\|_2/\|B\|_2)$ bounded by a multiple of $\kappa_2(B)u$, where u is the unit roundoff and $\kappa_2(B) = \|B\|_2\|B^{-1}\|_2$ is the condition number of B. We consider the use of the Jacobi method and the symmetric QR method as our eigensolver, and give a detailed error analysis for each

method that yields backward error bounds potentially much smaller than $\kappa_2(B)u$. We also highlight the need for pivoting in the Cholesky factorization and the need for applying the symmetric QR method on matrices that are graded downwards. We illustrate the sharpness of these new bounds with numerical tests.

In cases of instability, we consider the use of iterative refinement based on Newton's method to produce eigenpairs with small backward errors. We show that provided that the initial approximate eigenpair is "close enough" to the exact eigenpair we can expect Newton's method in floating point arithmetic to converge to a backward stable eigenpair.

We also consider the numerical generation of correlation matrices and their factors. These matrices—symmetric positive semidefinite matrices with unit diagonal—are important in statistics and numerical linear algebra. We present an algorithm of Bendel and Mickey and give improved formulae for the computation of the rotations and prove that the resulting algorithm is numerically stable. We also show how to modify the algorithm to generate a rectangular matrix with columns of unit 2-norm.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institution of learning.

Copyright

Publications

The material in Section 3.3 and some of Chapters 4 and 6 are based on the technical report "Analysis of the Cholesky Method with Iterative Refinement for Solving the Symmetric Definite Generalized Eigenproblem" (with Nicholas J. Higham and Françoise Tisseur), Numerical Analysis Report 360, Manchester Centre for Computational Mathematics, June 2000. This work has been submitted for publication in SIAM Journal on Matrix Analysis and Applications.

The material in Chapter 5 is based on the technical report "Numerically Stable Generation of Correlation Matrices and their Factors" (with Nicholas J. Higham), Numerical Analysis Report 354, Manchester Centre for Computational Mathematics, November 1999. This work is to appear in BIT 40:4, pp. 640–651, 2000.

Acknowledgements

I would like to thank my supervisor Professor Nick Higham for his technical expertise and support throughout this project and for leading the blind through the minefields of numerical linear algebra.

I would also like to thank Professor Sven Hammarling of NAG, Ltd. (Oxford) who has visited Manchester numerous times to offer advice and for providing the initial impetus for this project. Many thanks also to Dr. Françoise Tisseur for her help and collaboration on various sections of this thesis.

Special thanks also go to my family and friends for all the fun times.

Finally, thanks to the Engineering and Physical Sciences Research Council and NAG, Ltd. (Oxford) for providing financial support through a CASE Ph.D. Studentship.

Cheers to you all.

Chapter 1

Introduction

1.1 Overview

In this thesis we consider some theoretical and numerical aspects of the generalized eigenvalue problem

$$Ax = \lambda Bx$$

where $A, B \in \mathbb{C}^{n \times n}$. Of particular interest are numerical methods for solving the symmetric definite generalized eigenvalue problem where A and B are real and symmetric and at least one of them is positive definite.

We start by outlining the thesis, while the rest of Chapter 1 explains some of the necessary theoretical background and applications of the problem. This material has been adapted from various sources including Datta [9], Golub and Van Loan [26], Higham [31], Parlett [47], Saad [49], Stewart [52], [53], Stewart and Sun [55], Watkins [61] and Wilkinson [63].

In Chapter 2 we discuss two popular methods for solving the symmetric eigenvalue problem $Ax = \lambda x$, Jacobi's method and the symmetric QR method. These methods are backward stable and can be used to help solve the symmetric definite generalized eigenvalue problem.

In Chapter 3 we move on to consider numerical methods for the symmetric definite generalized eigenvalue problem. Ideally these methods would be able to exploit the special structure of the problem, efficient and backward stable. A backward stable algorithm is one that produces solutions to the generalized eigenvalue problem that are exact for the perturbed matrix pair $(A + \Delta A, B + \Delta B)$ where ΔA and ΔB are small. The term "small" is normally context dependent. We are particular interested in normwise backward stable algorithms where $\|\Delta A\| \leq c_1 u \|A\|$ and $\|\Delta B\| \leq c_2 u \|B\|$ where u is the unit roundoff and c_1 and c_2 are small constants. Unfortunately no such algorithm exists that satisfies all three desirable properties, but we discuss three methods which satisfy at least one of these criteria. This remains an important open problem in numerical linear algebra and is of particular interest to numerical software library producers such as NAG, Ltd. (Oxford), who seek to provide accurate, reliable and robust library code for use in industry and research. They provided the initial impetus to this research and have supported this project through a CASE Ph.D. Studentship.

Firstly, we discuss the QZ algorithm, which is a generalization of the QR algorithm for non Hermitian matrices. While this method is backward stable it does not exploit the structure of the problem.

Next we consider the popular Cholesky method which converts the generalized eigenvalue problem, via a Cholesky factorization of the symmetric positive definite B, to a standard eigenvalue problem $Cy = \lambda y$ while maintaining the symmetry of the problem. Although this method exploits the structure of the problem it is potentially unstable when B is ill-conditioned with respect to inversion. Standard error analysis says that, providing a backward stable eigensolver is used to solve the eigenvalue problem for C, the computed eigenvalues are exact for $A + \Delta A$ and $B + \Delta B$ where $\max(\|\Delta A\|_2/\|A\|_2, \|\Delta B\|_2/\|B\|_2)$ is bounded by a multiple of $\kappa_2(B)u$, where $\kappa_2(B) = \|B\|_2\|B^{-1}\|_2$ is the condition number of B.

We consider the use of the Jacobi and the symmetric QR method as our eigensolver, and give a detailed error analysis for both the Cholesky–Jacobi and the Cholesky–QR methods that yields backward error bounds potentially much smaller than $\kappa_2(B)u$. We also highlight the need for pivoting in the Cholesky factorization and the need for applying the symmetric QR method on matrices that are graded downwards.

Finally we discuss a method of Chandrasekaran which exploits the structure of the problem and is backward stable but, unfortunately has a potentially high operation count.

In Chapter 4 we discuss a method for repairing any instability that occurs when applying the Cholesky method. We consider the use of iterative refinement based on Newton's method to produce eigenpairs with small backward errors. We show that provided that the initial approximate eigenpair is "close enough" to the exact eigenpair we can expect Newton's method in floating point arithmetic to converge to a backward stable eigenpair.

In Chapter 5 we consider the numerical generation of correlation matrices—symmetric positive semidefinite matrices with unit diagonal—and their factors. We present an algorithm of Bendel and Mickey for computing correlations matrices with specified eigenvalues using a finite sequence of Givens rotations. We give improved formulae for the computation of the rotations and prove that the resulting algorithm is numerically stable. We show by example that the formulae originally proposed, which are used in certain existing Fortran implementations, can lead to serious instability. We also show how to modify the algorithm to generate a rectangular matrix with columns of unit 2-norm. Such a matrix represents a correlation matrix in factored form.

Finally in Chapter 6 we present several numerical examples to illustrate the sharpness of our backward error bounds and identify examples for which the

Cholesky–Jacobi and the Cholesky–QR methods can fail. We also illustrate the benefits of iterative refinement. We finish off by summing up the conclusions of this thesis and highlighting areas of potential future work.

1.2 Basic Definitions and Norms

1.2.1 Basic Definitions

The following standard terminology is used throughout the thesis:

- $A \in \mathbb{C}^{n\times n}$ is HERMITIAN if $A = A^*$ where A^* is the conjugate transpose of A. In the real case $A \in \mathbb{R}^{n\times n}$ is SYMMETRIC if $A = A^T$.
- $A \in \mathbb{C}^{n\times n}$ is HERMITIAN POSITIVE (SEMI) DEFINITE if $A = A^*$ and $x^*Ax > (\geq) 0$ for all nonzero $x \in \mathbb{C}^n$.
- $A \in \mathbb{C}^{n\times n}$ is UNITARY if $A^*A = I$. In the real case $A \in \mathbb{R}^{n\times n}$ is ORTHOGONAL if $A^TA = I$.
- $A \in \mathbb{C}^{n\times n}$ is UPPER (LOWER) TRIANGULAR if $a_{ij} = 0$ for $i > j$ $(i < j)$.
- $A \in \mathbb{C}^{n\times n}$ is UPPER (LOWER) HESSENBERG if $a_{ij} = 0$ for $i > j+1$ $(i < j-1)$.
- $A \in \mathbb{C}^{n\times n}$ is UPPER (LOWER) QUASI-TRIANGULAR if A is upper (lower) Hessenberg and no two consecutive elements on the sub (super) diagonal are nonzero.
- A QR factorization of $A \in \mathbb{R}^{m\times n}$, where $m \geq n$, is given by

$$A = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1,$$

  where $Q \in \mathbb{R}^{m\times m}$ is orthogonal and $R_1 \in \mathbb{R}^{m\times n}$ is upper triangular. It can be computed in several ways, for example by use of Householder transformations, Givens rotations and by the Gram–Schmidt method. The QR

factorization is unique if A has full rank and we require R to have positive diagonal elements.

1.2.2 Vector Norms

We introduce the concept of vector and matrix norms as these provide a convenient scalar measure of size which are used in perturbation and rounding error analyses. A vector norm is a function $\|\cdot\| : \mathbb{C}^n \to \mathbb{R}$ that satisfies the following conditions:

1. $\|x\| > 0$ for all nonzero $x \in \mathbb{C}^n$.
2. $\|\alpha x\| = |\alpha|\|x\|$ where $\alpha \in \mathbb{C}$, $x \in \mathbb{C}^n$.
3. The triangle inequality holds, that is $\|x + y\| \leq \|x\| + \|y\|$ for all $x, y \in \mathbb{C}^n$.

A useful class of vector norms are the Hölder p-norms which are defined by

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}, \quad p \geq 1.$$

The three most useful p-norms are the 1, 2 and ∞ norms:

$$\begin{aligned} \|x\|_1 &= \sum_{i=1}^{n} |x_i|, \\ \|x\|_2 &= \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2} = (x^* x)^{1/2}, \\ \|x\|_\infty &= \max_{1 \leq i \leq n} |x_i|. \end{aligned}$$

The 2-norm, also called the EUCLIDEAN norm, has the useful property that it is invariant under unitary transformations. For unitary Q we have $Q^*Q = I$ and therefore

$$\|Qx\|_2^2 = x^* Q^* Q x = x^* x = \|x\|_2^2.$$

An important property of p-norms is the Hölder inequality

$$|x^* y| \leq \|x\|_p \|y\|_q, \quad \frac{1}{p} + \frac{1}{q} = 1. \tag{1.1}$$

We have equality in (1.1) when $p, q > 1$ if the vectors $(|x_i|^p)$ and $(|y_i|^q)$ are linearly dependent and $x_i y_i$ lie on the same ray in the complex plane for all i. For $p = 1, q = \infty$ equality can be easily obtained, an example being when x and y are multiples of the vector $(1, \ldots, 1)^T$. A special case of the Hölder inequality is the Cauchy-Schwarz inequality

$$|x^* y| \leq \|x\|_2 \|y\|_2.$$

All vector p-norms are equivalent. This means that there exists constants α and β such that

$$\alpha \|x\|_p \leq \|x\|_q \leq \beta \|x\|_p.$$

For $q < p$ it has been shown [25] that

$$\|x\|_p \leq \|x\|_q \leq n^{\left(\frac{1}{q} - \frac{1}{p}\right)} \|x\|_p.$$

This yields useful results about the 1, 2 and ∞ norms:

$$\begin{aligned} \|x\|_2 \leq &\|x\|_1 \leq \sqrt{n} \|x\|_2, \\ \|x\|_\infty \leq &\|x\|_2 \leq \sqrt{n} \|x\|_\infty, \\ \|x\|_\infty \leq &\|x\|_1 \leq n \|x\|_\infty. \end{aligned}$$

Another important vector norm is the dual norm. For an arbitrary vector norm $\|\cdot\| : \mathbb{C}^n \to \mathbb{R}$ the dual norm is defined by

$$\|x\|^D = \max_{z \neq 0} \frac{|z^* x|}{\|z\|}. \tag{1.2}$$

If follows from the Hölder inequality (1.1) that the dual of the p-norm is the q-norm where $\frac{1}{p} + \frac{1}{q} = 1$. The vector z is a vector dual to y if

$$z^* y = \|z\|^D \|y\| = 1.$$

The existence of such a vector is a consequence of the duality theorem [34, Cor. 5.5.15]. We note for later reference that

$$\|xy^*\|_{\alpha,\beta} = \|x\|_\beta \|y\|_\alpha^D.$$

This is easily verified from the definition of the mixed subordinate matrix norm in (1.4) and the dual norm in (1.2).

1.2.3 Matrix Norms

A matrix norm is a function $\|\cdot\| : \mathbb{C}^{m\times n} \to \mathbb{R}$ that satisfies the following conditions:

1. $\|A\| > 0$ for all nonzero $A \in \mathbb{C}^{m\times n}$.
2. $\|\alpha A\| = |\alpha|\|A\|$ where $\alpha \in \mathbb{C}$, $A \in \mathbb{C}^{m\times n}$.
3. The triangle inequality holds, that is $\|A + B\| \leq \|A\| + \|B\|$ for all $A, B \in \mathbb{C}^{m\times n}$.

The most commonly used matrix norms are the Frobenius norm which is defined by

$$\|A\|_F = \left(\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2\right)^{1/2} = (\text{trace}(A^*A))^{1/2}$$

and the subordinate p-norms. Given a vector norm, the corresponding subordinate matrix norm is given by

$$\|A\| = \sup_{x\neq 0} \frac{\|Ax\|}{\|x\|}.$$

The subordinate matrix p-norms are therefore defined using the Hölder p-norms. Again the three most useful subordinate matrix norms are the 1, 2 and the ∞ norms which can be given by:

$$\begin{aligned}
\|A\|_1 &= \max_{1\leq j\leq n} \sum_{i=1}^{m} |a_{ij}|, \quad \text{“max column sum”},\\
\|A\|_2 &= (\rho(A^*A))^{1/2} = \sigma_{\max}(A), \quad \text{spectral norm},\\
\|A\|_\infty &= \max_{1\leq i\leq m} \sum_{j=1}^{n} |a_{ij}|, \quad \text{“max row sum”},
\end{aligned}$$

where the spectral radius

$$\rho(B) = \max\{|\lambda| : \det(B - \lambda I) = 0\}$$

and $\sigma_{\max}(A)$ is the largest singular value of A. The matrix 2-norm and the Frobenius norm are unitarily invariant norms. This means that for unitary matrices U and V we have $\|UAV\|_2 = \|A\|_2$ and $\|UAV\|_F = \|A\|_F$. A norm is consistent if

$$\|AB\| \leq \|A\|\|B\| \tag{1.3}$$

whenever the product AB is defined. For any subordinate matrix norm and the Frobenius norm, the inequality (1.3) holds and therefore they are all consistent norms. An example of a norm that is not consistent is the "max norm" $\|A\| = \max_{i,j} |a_{ij}|$. The best bound obtainable for $A \in \mathbb{C}^{m\times n}$ and $B \in \mathbb{C}^{n\times p}$ is $\|AB\| \leq n\|A\|\|B\|$ where equality can be obtained for $a_{ij} = b_{ij} = 1$.

For the matrix p-norm it can shown that $\|A^*\|_p = \|A\|_q$ if $\frac{1}{p} + \frac{1}{q} = 1$. We can also show that

$$\|A\|_p \leq \|A\|_1^{1/p}\|A\|_\infty^{1-1/p}$$

which includes the useful inequality $\|A\|_2 \leq \sqrt{\|A\|_1\|A\|_\infty}$. Other useful inequalities involving the ubiquitous 1, 2 and ∞ norms are

$$\begin{aligned}
\frac{1}{\sqrt{n}}\|A\|_\infty \leq &\|A\|_2 \leq \sqrt{m}\|A\|_\infty, \\
\frac{1}{\sqrt{m}}\|A\|_1 \leq &\|A\|_2 \leq \sqrt{n}\|A\|_1, \\
\|A\|_2 \leq &\|A\|_F \leq \sqrt{n}\|A\|_2.
\end{aligned}$$

The subordinate matrix norm can be generalized by allowing different norms on the input and output space:

$$\|A\|_{\alpha,\beta} = \max_{x\neq 0} \frac{\|Ax\|_\beta}{\|x\|_\alpha}. \tag{1.4}$$

The choice of $\alpha = 1$ and $\beta = \infty$ gives the "max norm". The result in (1.3) does not hold for the mixed subordinate matrix norm. Instead, for any third vector

norm $\|\cdot\|_\gamma$, we have

$$\|AB\|_{\alpha,\beta} \leq \|A\|_{\gamma,\beta}\|B\|_{\alpha,\gamma}.$$

An important quantity in numerical analysis is the matrix condition number of a nonsingular $A \in \mathbb{C}^{n\times n}$, which is defined by

$$\kappa_{\alpha,\beta}(A) := \lim_{\epsilon\to 0} \sup_{\|\Delta A\|_{\alpha,\beta}\leq\epsilon\|A\|_{\alpha,\beta}} \left(\frac{\|(A+\Delta A)^{-1} - A^{-1}\|_{\beta,\alpha}}{\epsilon\|A^{-1}\|_{\beta,\alpha}} \right).$$

An explicit formula can be given for the condition number:

$$\kappa_{\alpha,\beta}(A) = \|A\|_{\alpha,\beta}\|A^{-1}\|_{\beta,\alpha}. \tag{1.5}$$

One of the reasons the matrix condition number is an important tool in numerical linear algebra is that it can measure the sensitivity of a matrix to perturbations in the data. For example the relative distance of a nonsingular $A \in \mathbb{C}^{n\times n}$ to singularity can be given by

$$\mathrm{dist}_{\alpha,\beta}(A) := \min\left(\frac{\|\Delta A\|_{\alpha,\beta}}{\|A\|_{\alpha,\beta}} : A+\Delta A \text{ singular} \right) = \kappa_{\alpha,\beta}(A)^{-1}. \tag{1.6}$$

See Higham [31] for proofs of the results in (1.5) and (1.6).

1.2.4 Model of Floating Point Arithmetic

Throughout this thesis we use the standard model for floating point arithmetic

$$fl(x \operatorname{op} y) = (x \operatorname{op} y)(1+\delta_1) = \frac{x \operatorname{op} y}{1+\delta_2}, \quad |\delta_1|, |\delta_2| \leq u, \quad \operatorname{op} = +, -, *, /.$$
$$fl(\sqrt{x}) = \sqrt{x}(1+\delta), \quad |\delta| \leq u,$$

where u is the unit roundoff. We will make use of the following lemma [31].

Lemma 1.2.1 *If $|\delta_i| \leq u$ and $\rho_i = \pm 1$ for $i = 1\colon n$, and $nu < 1$, then*

$$\prod_{i=1}^{n}(1+\delta_i)^{\rho_i} = 1+\theta_n, \quad \textit{where} \quad |\theta_n| \leq \frac{nu}{1-nu} =: \gamma_n.$$

We define

$$\tilde{\gamma}_k = \frac{pku}{1 - pku},$$

where p denotes a small integer constant whose exact value is unimportant. We will also write $\tilde{\theta}_k$ to denote a quantity satisfying $|\tilde{\theta}_k| \leq \tilde{\gamma}_k$. Computed quantities are denoted with a hat.

1.3 The Standard Eigenvalue Problem

The standard eigenvalue problem arises when we consider systems of differential equations in areas such as vibration analysis, stability theory and quantum mechanics. For example, consider the following system of n first-order, linear, ordinary differential equations in n unknown functions $x_1(t), \ldots, x_n(t)$:

$$\begin{aligned}
\tfrac{dx_1}{dt} &= a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n, \\
\tfrac{dx_2}{dt} &= a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n, \\
&\vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \\
\tfrac{dx_n}{dt} &= a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n.
\end{aligned}$$

This system is normally expressed as

$$\frac{dx}{dt} = Ax \tag{1.7}$$

where $x = (x_1, \ldots, x_n)^T$, $dx/dt = (dx_1/dt, \ldots, dx_n/dt)^T$ and $A = (a_{ij})$ is the matrix of coefficients. To solve this system we consider solutions of the simple form

$$x(t) = e^{\lambda t}v \tag{1.8}$$

where $e^{\lambda t}$ is a time dependent scalar and v is a nonzero vector independent of t. By substituting (1.8) into (1.7) and we obtain

$$\lambda e^{\lambda t}v = e^{\lambda t}Av.$$

Since $e^{\lambda t}$ is nonzero we obtain the standard eigenvalue problem

$$Av = \lambda v.$$

We now make a formal definition of the problem.

Definition 1.3.1 *Let $A \in \mathbb{C}^{n\times n}$, $x \in \mathbb{C}^n$ be nonzero and λ be a complex scalar. The pair (λ, x) is called an* EIGENPAIR *of A if*

$$Ax = \lambda x. \tag{1.9}$$

The vector x is an EIGENVECTOR *and the scalar λ is an* EIGENVALUE *of the matrix A. The set of eigenvalues is called the* SPECTRUM *of A and is denoted by $\Lambda(A)$.*

Notice that if (λ, x) is an eigenpair of A then so is (λ, cx) where c is any nonzero constant. It is therefore common to consider normalized eigenvectors with $\|x\|_2 = 1$. The eigenpair described in Definition 1.3.1 is sometimes called a right eigenvector as x is multiplied on the right hand side. A left eigenvector (λ, y) is similarly defined except that this eigenpair satisfies

$$y^*A = \lambda y^*.$$

If we consider the solution in (1.8) we see that v must be an eigenvector of A and λ must be the corresponding eigenvalue. For every eigenpair of A a solution of the form (1.8) exists. Therefore if A has n linearly independent eigenvectors $v_1, \ldots, v_n$ with corresponding eigenvalues $\lambda_1, \ldots, \lambda_n$ then

$$x(t) = c_1 e^{\lambda_1 t} v_1 + \cdots + c_n e^{\lambda_n t} v_n$$

is a solution of (1.7) where $c_1, \ldots, c_n$ are arbitrary constants.

1.3.1 Properties of the Eigenvalue Problem

We will first look at the existence and uniqueness of solutions of the standard eigenvalue problem. By rewriting (1.9) as a linear system

$$(A - \lambda I)x = 0$$

we can show the following:

- $x \in N(A-\lambda I)$ where $N(A-\lambda I)$ denotes the null space of the matrix $A-\lambda I$.
- λ is an eigenvalue of A if and only if $N(A - \lambda I) \neq \{0\}$.
- λ is an eigenvalue of A if and only if

$$\det(A - \lambda I) = 0. \tag{1.10}$$

The left-hand side of (1.10) is known as the characteristic polynomial of A.

It is not hard to see that (1.10) is a polynomial in λ with degree n. This means that (1.10) has n complex roots and therefore A has n eigenvalues, some of which may be repeated. We can write (1.10) in factored form as

$$\det(A - \lambda I) = (\lambda - \lambda_1)^{m_1} \dots (\lambda - \lambda_k)^{m_k}$$

where each λ_i is distinct and $m_1 + \cdots + m_k = n$. The eigenvalues are the scalars λ_i and they are said to have ALGEBRAIC MULTIPLICITY m_i. This differs from the GEOMETRIC MULTIPLICITY of an eigenvalue λ_i which is the dimension of the null space $N(A - \lambda_i I)$. The algebraic and geometric multiplicity of an eigenvalue are often the same but can be different. For example,

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

has the characteristic polynomial $\lambda^2 = 0$ and therefore $\lambda_1 = \lambda_2 = 0$ and the eigenvalue 0 has algebraic multiplicity 2. However the null space $N(A - \lambda_1 I)$ is

spanned by $(1,0)^T$ only and therefore the eigenvalue 0 has geometric multiplicity 1. If a matrix has an eigenvalue whose algebraic multiplicity is greater than its geometric multiplicity then the matrix and the eigenvalue is said to be DEFECTIVE. Defective matrices have less than n linearly independent eigenvectors.

There are certain kinds of matrices for which the eigenpairs are easily found:

- Let e_i denote the ith column of the identity matrix. For a diagonal matrix $A = \operatorname{diag}(a_{11}, \ldots, a_{nn})$ it is easy to see that (a_{ii}, e_i) for $i = 1\colon n$ is an eigenpair of A.

- Let A be a block triangular matrix partitioned in the form

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1m} \\ & A_{22} & & A_{2m} \\ & & \ddots & \vdots \\ & & & A_{mm} \end{bmatrix},$$

  where all A_{ii} are square. Then

$$\Lambda(A) = \bigcup_{i=1}^{m} \Lambda(A_{ii}).$$

  This is due to the fact that

$$\det(A - \lambda I) = \prod_{i=1}^{m} \det(A_{ii} - \lambda I)$$

  which is zero if and only if $\det(A_{ii} - \lambda I) = 0$ for some i. If A is triangular and partitioned in the form

$$A = \begin{array}{c} \\ k-1 \\ 1 \\ n-k \end{array} \begin{array}{c} \begin{array}{ccc} k-1 & 1 & n-k \end{array} \\ \begin{bmatrix} A_{11} & a_{1k} & A_{1,k+1} \\ & a_{kk} & a_{k,k+1}^T \\ & & A_{k+1,k+1} \end{bmatrix} \end{array}$$

  then it can be seen that an eigenpair of A is

$$(\lambda, x) = \left(a_{kk}, [x_1^T, x_2, 0]^T\right)$$

where

$$x_1 = (\lambda I - A_{11})^{-1} a_{1k}, \quad x_2 = 1$$

if $\lambda I - A_{11}$ is nonsingular. If $\lambda I - A_{11}$ is singular then x_1 is a null vector of this matrix and $x_2 = 0$. These eigenvectors can be found using a back substitution algorithm.

1.3.2 Similarity Transformations

A key strategy in solving eigenvalue problems is to transform the problem to one which is easier to solve, for example diagonal or triangular form as we have seen that these matrices have easily obtainable eigenpairs. The tools we use are called similarity transformations.

Theorem 1.3.2 *Let $A \in \mathbb{C}^{n\times n}$ and let $Y \in \mathbb{C}^{n\times n}$ be nonsingular. Then the matrices A and $B = YAY^{-1}$ are said to be* SIMILAR. *This means that if (λ, x) is an eigenpair of A then (λ, Yx) is an eigenpair of B. The transformation $B = YAY^{-1}$ is known as a* SIMILARITY TRANSFORMATION.

Proof. If (λ, x) is an eigenpair of A then

$$B(Yx) = YAY^{-1}Yx = YAx = \lambda(Yx)$$

and therefore (λ, Yx) is eigenpair of B. □

An important class of similarity transformations are unitary transformations. This is because they are easy to invert as $U^{-1} = U^*$ and perturbations in A are not magnified in B. For example if we apply a similarity transformation to the perturbed matrix $A + E$ we have

$$B + F = Y(A + E)Y^{-1}$$

where $\|E\|_2\kappa_2(Y)^{-1} \leq \|F\|_2 \leq \|E\|_2\kappa_2(Y)$. For unitary matrices we have $\kappa_2(Y) = 1$ and $\|E\|_2 = \|F\|_2$.

If we can find a similarity transformation such that B is diagonal or triangular then we can obtain all eigenpairs of A. We introduce the following theorems to show that this is possible for certain A.

Theorem 1.3.3 *Let $A \in \mathbb{C}^{n\times n}$ have a complete set of linearly independent eigenvectors $x_1, \ldots, x_n$ with corresponding eigenvalues $\lambda_1, \ldots, \lambda_n$ and define the matrices*

$$X = [x_1, \ldots, x_n], \quad \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n).$$

Then

$$X^{-1}AX = \Lambda. \tag{1.11}$$

Proof. All we need to do is combine all n solutions of (1.9) into the matrix equation $AX = X\Lambda$ and because there are n linearly independent eigenvectors X is nonsingular and can be inverted. □

Theorem 1.3.4 (Schur) *Given $A \in \mathbb{C}^{n\times n}$ then there exists a unitary matrix $U \in \mathbb{C}^{n\times n}$ such that*

$$U^*AU = T \tag{1.12}$$

is an upper triangular matrix .

Proof. We prove this by induction on n. For the base case $n = 1$ the result is trivial. For $A \in \mathbb{C}^{k\times k}$ let x_1 be a normalized eigenvector with corresponding eigenvalue λ_1. Let $X_2 \in \mathbb{C}^{k\times(k-1)}$ denote a matrix such that $U_1 = [x_1 \; X_2]$ is unitary. Then

$$\begin{aligned} U_1^*AU_1 &= \begin{bmatrix} x_1^* \\ X_2^* \end{bmatrix} A \begin{bmatrix} x_1 & X_2 \end{bmatrix} \\ &= \begin{bmatrix} x_1^*Ax_1 & x_1^*AX_2 \\ X_2^*Ax_1 & X_2^*AX_2 \end{bmatrix}. \end{aligned}$$

Since $Ax_1 = \lambda_1 x_1$ and $X_2^*x_1 = 0$ we have

$$U_1^*AU_1 = \begin{bmatrix} \lambda_1 & x_1^*AX_2 \\ 0 & X_2^*AX_2 \end{bmatrix}.$$

Using our inductive hypothesis there exists a unitary matrix $\widetilde{U}_2 \in \mathbb{C}^{(k-1)\times(k-1)}$ such that $\widetilde{U}_2^*(X_2^*AX_2)\widetilde{U}_2$ is triangular. If we define $U_2 = \text{diag}(1, \widetilde{U}_2)$ then we have

$$U_2^*U_1^*AU_1U_2 = \begin{bmatrix} \lambda_1 & x_1^*AX_2\widetilde{U}_2 \\ 0 & \widetilde{U}_2^*X_2^*AX_2\widetilde{U}_2 \end{bmatrix}$$

which is upper triangular. □

Neither of these proofs are constructive so they do not suggest methods for finding these factorizations. The proof of Theorem 1.3.4 does suggest a way of reducing the problem to a $(n-1)\times(n-1)$ problem if we know an eigenvector of A. This process is known as DEFLATION.

The eigenvectors in (1.11) and the Schur vectors in (1.12) bear an interesting relation. Let

$$X = UR$$

be the QR decomposition of the matrix of eigenvectors X. Then (1.11) is equivalent to

$$U^*AU = R\Lambda R^{-1}.$$

As the right hand side is upper triangular, this is the Schur form of A. Hence the Schur vectors are the Q-factor of the matrix of eigenvectors.

1.3.3 The Hermitian Eigenvalue Problem

One common form of the eigenvalue problem is when A is Hermitian. Many of the relevant characteristics of Hermitian matrices can be obtained from the Schur form in Theorem 1.3.4. If we consider (1.12) and take the conjugate transpose we see that

$$T^* = (U^*AU)^* = U^*A^*U = U^*AU = T. \tag{1.13}$$

From this we can see that T is also Hermitian and is therefore both upper and lower triangular and hence diagonal. As U also diagonalizes A we can apply

Theorem 1.3.3 to show that the columns of U are the eigenvectors and therefore (t_{ii}, u_i) are the eigenpairs of A. Also from (1.13) we see that $t_{ii} = \bar{t}_{ii}$, which means t_{ii} must have zero imaginary part and therefore all the eigenvalues of Hermitian matrices are real.

Theorem 1.3.5 (The Spectral Theorem) *Given a Hermitian $A \in \mathbb{C}^{n\times n}$ there exists a unitary matrix $U \in \mathbb{C}^{n\times n}$ and a diagonal matrix $\Lambda \in \mathbb{R}^{n\times n}$ such that*

$$U^*AU = \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n).$$

This is known as the SPECTRAL DECOMPOSITION.

1.4 The Generalized Eigenvalue Problem

The generalized eigenvalue problem arises in many applications such as structural dynamics, electrical networks and quantum chemistry. The most commonly solved problems are those associated with the vibration analysis of large structures. These structures have distributed properties, such as mass and stiffness, and are said to have an infinite number of degrees of freedom as the system is not fully described until the motion at all points is known. It is infeasible to work with an infinite number of degrees of freedom, therefore these systems are expressed by discrete mathematical models using finite element methods or finite difference methods. These methods regard a complex structure as an assemblage of discrete elements involving a workable number of degrees of freedom. Of particular interest are the natural frequencies and modes of vibration of the structure. The design of these structures require consideration of this behaviour. These are found by solving a generalized eigenvalue problem,

$$Kx = \lambda Mx \tag{1.14}$$

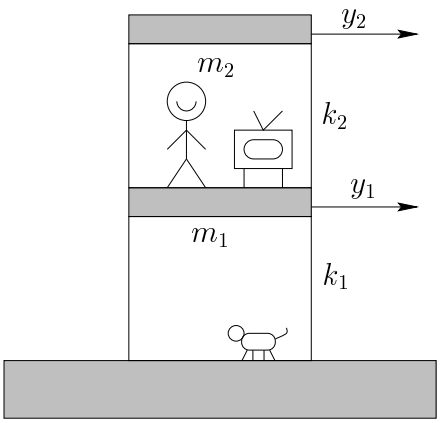


Figure 1.1: Model of a two storey building.

where K and M are known as the stiffness and the mass matrix respectively. Usually the matrices K and M are Hermitian and at least one is positive definite. We now consider free and forced vibration using 2×2 examples.

Example of free vibration

Free vibration occurs when a structure oscillates due to the forces inherent in the system when unaffected by outside forces. The structure will vibrate at one or more of its natural frequencies which depend on the mass and stiffness distribution. Consider the model of a two storey building shown in Figure 1.1. The building has a floor and roof, represented by masses m_1 and m_2 respectively, which have a horizontal motion, represented by y_1 and y_2, which is caused by the deformation of the columns. The constants k_1 and k_2 represent the stiffness of the supporting columns. The equations of motion can be written as

$$\begin{aligned} m_1\ddot{y}_1 + (k_1 + k_2)y_1 - k_2 y_2 &= 0, \\ m_2\ddot{y}_2 - k_2 y_1 + k_2 y_2 &= 0. \end{aligned} \tag{1.15}$$

The equations in (1.15) can be written in matrix form as

$$M\ddot{y} + Ky = 0 \tag{1.16}$$

where $\ddot{y} = (\ddot{y}_1, \ddot{y}_2)^T$, $y = (y_1, y_2)^T$ and the mass and stiffness matrices are given by

$$M = \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix}, \quad K = \begin{bmatrix} k_1 + k_2 & -k_2 \\ -k_2 & k_2 \end{bmatrix}.$$

We consider solutions of the form

$$y(t) = xe^{i\omega t} \tag{1.17}$$

where $e^{i\omega t}$ is a time dependent complex scalar and x is a nonzero vector independent of t. By substituting (1.17) into (1.16) we get

$$e^{i\omega t}Kx - \omega^2 e^{i\omega t}Mx = 0.$$

Since $e^{i\omega t}$ is nonzero we get the generalized eigenvalue problem $Kx = \lambda Mx$ where $\lambda = \omega^2$. We now make a formal definition of the generalized eigenvalue problem similar to Definition 1.3.1.

Definition 1.4.1 *Let $A, B \in \mathbb{C}^{n\times n}$, $x \in \mathbb{C}^n$ be nonzero and λ be a complex scalar. The pair (λ, x) is called an* EIGENPAIR *of (A, B) if*

$$Ax = \lambda Bx.$$

The vector x is an EIGENVECTOR *and the scalar λ is an* EIGENVALUE *of the matrix pair (A, B).*

This definition only considers finite eigenvalues. The matrix pair (A, B) can have infinite eigenvalues when B is singular. This case is discussed in Section 1.4.1.

As in the standard eigenvalue problem we can see that if (λ, x) is an eigenpair of (A, B) then so is (λ, cx) where c is any nonzero constant. Therefore we refer to x as a normalized eigenvector when $\|x\|_2 = 1$. The eigenpair described in Definition 1.4.1 is sometimes called a right eigenpair as the vector x is multiplied

on the right hand side. A left eigenpair (λ, y) is similarly defined except that this eigenpair satisfies $y^*A = \lambda y^*B$.

If we consider the solution in (1.17) we see that x must be an eigenvector of (K, M) and ω^2 must be the corresponding eigenvalue. Notice that if $xe^{i\omega t}$ is a solution of (1.16) then so is $xe^{-i\omega t}$. For every eigenpair of (K, M) a solution of the form (1.17) exists as well as any linear combination of these solutions. Therefore

$$\begin{aligned} x(t) &= (c_1 e^{i\omega_1 t} + d_1 e^{-i\omega_1 t})x_1 + (c_2 e^{i\omega_2 t} + d_2 e^{-i\omega_2 t})x_2 \\ &= (c_3 \cos\omega_1 t + d_3 \sin\omega_2 t)x_1 + (c_4 \cos\omega_2 t + d_4 \sin\omega_2 t)x_2 \end{aligned}$$

is a solution of (1.16) where c_i, d_i for $i = 1, \ldots, 4$ are constants dependent on the initial conditions.

By taking $m = m_1 = m_2$ and $k = k_1 = k_2$ and solving the generalized eigenvalue problem we can show that

$$\omega_1^2 = \frac{k}{m}(0.3820), \quad \omega_2^2 = \frac{k}{m}(2.6180)$$

and the corresponding eigenvectors are

$$x_1 = \begin{bmatrix} 0.5257 \\ 0.8507 \end{bmatrix}, \quad x_2 = \begin{bmatrix} -0.8507 \\ 0.5257 \end{bmatrix}.$$

Figure 1.2 shows the vibration of the building at its two natural frequencies.

Example of forced vibration

Forced vibration occurs when a structure oscillates upon application of external forces. When the external forces are oscillatory in nature then the structure vibrates at this frequency. Consider the system of two spring masses shown in Figure 1.3 where the two masses are represented by m_1 and m_2 and each spring constant is k. The displacement of the masses is represented by y_1 and y_2. The system is excited by the external harmonic force $F_1 \sin\omega t$. The equations of motion for this system are

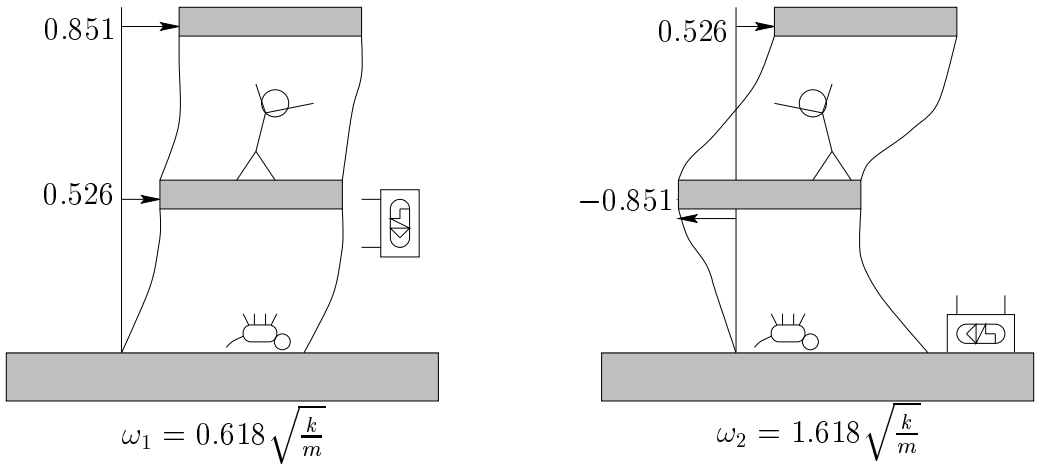


Figure 1.2: Diagrams showing the free vibration of a model building at its two natural frequencies.

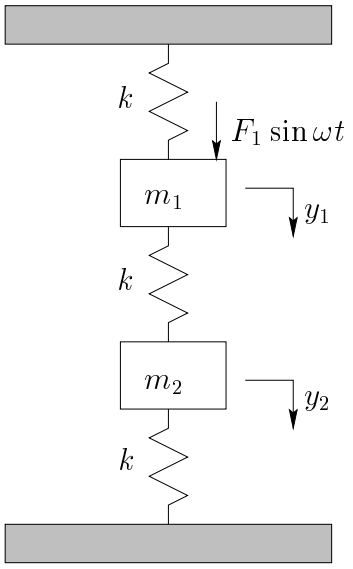


Figure 1.3: Diagrams showing forced harmonic vibration.

$$M\ddot{y} + Ky = \begin{bmatrix} F_1 \\ 0 \end{bmatrix} \sin \omega t \tag{1.18}$$

and the mass and stiffness matrix are given by

$$M = \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix}, \quad K = \begin{bmatrix} 2k & -k \\ -k & 2k \end{bmatrix}.$$

We consider solutions of the form

$$y(t) = x \sin \omega t \tag{1.19}$$

where x is a nonzero vector which does not depend on t. By substituting (1.19) into (1.18) we get

$$(K - \omega^2 M)x = \begin{bmatrix} F_1 \\ 0 \end{bmatrix}$$

whose solution $x = (x_1, x_2)^T$ can be given by

$$x_1 = \frac{(2k - m_2\omega^2)F_1}{\det(K - \omega^2 M)}, \quad x_2 = \frac{kF_1}{\det(K - \omega^2 M)}. \tag{1.20}$$

The amplitude of the solutions in (1.20) can become arbitrarily large when $\det(K - \omega^2 M)$ is close to zero. We can show that

$$\det(K - \omega^2 M) = m_1 m_2 (\omega_1^2 - \omega^2)(\omega_2^2 - \omega^2) \tag{1.21}$$

where ω_1^2 and ω_2^2 are the eigenvalues of (K, M) and the natural frequencies of the system. We can see from (1.21) that the determinant will be close to zero when the frequency of the forced harmonic force is close to one of the natural frequencies of the system. This can cause dangerously large oscillations. This phenomenon is known as resonance and is of great concern to engineers. The collapse of the Tacoma bridge in the state of Washington USA and the Broughton suspension bridge in Manchester in 1831 have been attributed to this phenomenon. In the case of the Tacoma bridge the external force was thought to be the wind although this view has been challenged recently [9]. In the case of the Broughton suspension bridge the external force was thought to be caused by soldiers marching in cadence

over the bridge. Reports about this bridge are hazy at best. Various sources on the internet report that it was the 60th Rifle Corps marching over the bridge, while another says it was the 6th Rifle Corps. Another source says that the bridge probably overloaded. Urban myth or not, this account gave rise to the order "Break step" when marching men across bridges to stop this phenomenon from happening.

The two examples shown ignore the presence of damping in a dynamical system, which is the energy dissipation caused by friction and any other resistances. If these values were considered a first derivative term $C\dot{y}$ would be included in (1.16) and (1.18). Inclusion of this term would lead to a quadratic eigenvalue problem. A good source for information on this problem, which isn't discussed in this thesis, is Lancaster [37]. When these damping values are small they have very little effect on the natural frequencies of the system and are not included in their estimation. The damping effects become important when they limit the amplitude of the vibrations when resonance occurs. They also reduce the amplitude of vibrations with time.

1.4.1 Properties of the Generalized Eigenvalue Problem

For the generalized eigenvalue problem

$$(A - \lambda B)x = 0$$

it is immediate that

- x is a null vector of the matrix $(A - \lambda B)$.
- λ is an eigenvalue of (A, B) if and only if

$$\det(A - \lambda B) = 0. \tag{1.22}$$

Unlike for the standard eigenvalue problem the characteristic equation defined in (1.22) is not necessarily a polynomial with degree n. When B is singular with rank r, then $\det(A - \lambda B)$ is a polynomial with maximum degree r.

When B is nonsingular (1.14) is equivalent to the following standard eigenvalue problems:

- $B^{-1}Ax = \lambda x$.
- $AB^{-1}y = \lambda y$ where $y = Bx$.

Notice that AB^{-1} and $B^{-1}A$ are similar matrices as $AB^{-1} = B(B^{-1}A)B^{-1}$ and therefore they have the same eigenvalues and characteristic equations. Although converting the problem into a standard eigenvalue problem of the form $B^{-1}Ax = \lambda x$ or $AB^{-1}y = \lambda y$ is a possibility, there are reasons for not doing this in practice. Firstly if B is ill conditioned then the eigenvalues of the computed AB^{-1} (or $B^{-1}A$) can be very far from the eigenvalues of (A, B). Also in certain applications A and B can be Hermitian or sparse and it is desirable to preserve these properties. Typically AB^{-1} (or $B^{-1}A$) is neither Hermitian or sparse.

When B is singular we get some differences from the standard eigenvalue problem. Consider the matrices

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

Notice that B is singular with rank 1. The characteristic polynomial for the matrix pair (A, B) is $\lambda - 1 = 0$. It seems that (A, B) has only one eigenvalue $\lambda_1 = 1$ but the matrix pair (A, B) has a second eigenvalue $\lambda_2 = \infty$. We can see this by rewriting the generalized eigenvalue form in (α, β) form:

$$\beta Ax = \alpha Bx, \quad \lambda = \alpha/\beta.$$

This form allows us to think of the generalized eigenvalue as a pair (α, β). This has advantages over using λ as we can interpret infinite eigenvalues as $(\alpha, \beta) = (c, 0)$

where c is a nonzero constant. This case is similar to the case when we have zero eigenvalues of the pair (A, B). This can be represented by $(\alpha, \beta) = (0, c)$. Therefore infinite eigenvalues can be seen as zero eigenvalues of the matrix pair (B, A). If we consider $\det(\beta A - \alpha B) = \beta(\beta - \alpha) = 0$ we can see that the roots are $\beta = 0$ and $\alpha = \beta$ which gives us the eigenvalues 1 and ∞.

Using this convention it can be seen that most generalized eigenvalue problems have n eigenvalues. However there exist examples where (A, B) can have an infinite number of eigenvalues. Consider

$$A = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

Notice that A and B are both singular and that they share the same null vector, that is $Ae_1 = Be_1 = 0$. Therefore for any λ, then $Ae_1 = \lambda Be_1$ and every $\lambda \in \mathbb{C}$ is an eigenvalue of (A, B). Also notice that the characteristic polynomial $\det(A - \lambda B)$ is identically zero for all λ. We now make a formal definition to distinguish these two types of problem.

Definition 1.4.2 *A matrix pair (A, B) is said to be* SINGULAR *if $\det(A - \lambda B)$ is identically zero for all λ. If it is not identically zero then the matrix pair is said to be* REGULAR.

There are certain kinds of matrix pairs where the eigenpairs are easy to obtain.

- Let (A, B) be a regular pair where both A and B are diagonal. Then it is easy to see that $(a_{ii}/b_{ii}, e_i)$ for $i = 1{:}n$ are the eigenpairs of (A, B).

- Let (A, B) be a regular pair where both A and B are block triangular and they are partitioned in the form

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1m} \\ & A_{22} & & A_{2m} \\ & & \ddots & \vdots \\ & & & A_{mm} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1m} \\ & B_{22} & & B_{2m} \\ & & \ddots & \vdots \\ & & & B_{mm} \end{bmatrix},$$

where A_{ii} and B_{ii} are both square matrices of the same dimensions. Then

$$\Lambda(A,B) = \bigcup_{i=1}^{m} \Lambda(A_{ii}, B_{ii}).$$

This is due to the fact that

$$\det(A - \lambda B) = \prod_{i=1}^{m} \det(A_{ii} - \lambda B_{ii}),$$

which is zero if and only if $\det(A_{ii} - \lambda B_{ii}) = 0$ for some i. If A and B are triangular where A (and B similarly) is partitioned in the form

$$A = \begin{array}{c} \\ k-1 \\ 1 \\ n-k \end{array} \begin{array}{c} \begin{array}{ccc} k-1 & 1 & n-k \end{array} \\ \begin{bmatrix} A_{11} & a_{1k} & A_{1,k+1} \\ & a_{kk} & a_{k,k+1}^T \\ & & A_{k+1,k+1} \end{bmatrix} \end{array}$$

then

$$(\lambda, x) = \left(a_{kk}/b_{kk}, [x_1^T, x_2, 0]^T\right)$$

where

$$x_1 = (\lambda B_{11} - A_{11})^{-1} a_{1k}, \quad x_2 = 1$$

if $\lambda B_{11} - A_{11}$ is nonsingular. If $\lambda B_{11} - A_{11}$ is singular then x_1 is a null vector of this matrix and $x_2 = 0$.

1.4.2 Equivalence Transformations

We saw for the standard eigenvalue problem that we could use similarity transformations to transform the problem to one which is easier to solve. These transformations preserved the eigenvalues and changed the eigenvectors in a simple way. The transformations used for the generalized eigenvalue problem are called equivalence transformations.

Theorem 1.4.3 *Let (A, B) be a matrix pair and let $U, V \in \mathbb{C}^{n\times n}$ be nonsingular. Then the matrix pairs (A, B) and (U^*AV, U^*BV) are said to be* EQUIVALENT. *This means that if (λ, x) is an eigenpair of (A, B) then $(\lambda, V^{-1}x)$ is an eigenpair of (U^*AV, U^*BV). The transformation of $\widetilde{A} = U^*AV$ and $\widetilde{B} = U^*BV$ is known as an* EQUIVALENCE TRANSFORMATION.

Proof. Similar to the proof of Theorem 1.3.2 □

As with similarity transformations we would like to use unitary transformations. We can generalize Theorem 1.3.4 to show we can use unitary equivalence transformations to form a triangular matrix pair with the same eigenvalues as (A, B).

Theorem 1.4.4 (Generalized Schur Form) *If (A, B) is a regular pair, then there exist unitary matrices $U, V \in \mathbb{C}^{n\times n}$ such that*

$$U^*AV = T, \quad U^*BV = S \tag{1.23}$$

are triangular matrices.

Proof. The proof is by induction on n. For the base case $n = 1$ the result is trivial. For $A, B \in \mathbb{C}^{k\times k}$ let y_1 be a normalized eigenvector of (A, B) and let $Y_2 \in \mathbb{C}^{k\times(k-1)}$ denote a matrix such that $V_1 = [y_1\ Y_2]$ is unitary. As (A, B) is regular then at least one of Ay_1 and By_1 is nonzero. If Ay_1 is nonzero then let $x_1 = Ay_1/\|Ay_1\|_2$ and let $X_2 \in \mathbb{C}^{k\times(k-1)}$ denote a matrix such that $U_1 = [x_1\ X_2]$ is unitary. Then

$$\begin{aligned} U_1^*AV_1 &= \begin{bmatrix} x_1^* \\ X_2^* \end{bmatrix} A \begin{bmatrix} y_1 & Y_2 \end{bmatrix} \\ &= \begin{bmatrix} x_1^*Ay_1 & x_1^*AY_2 \\ X_2^*Ay_1 & X_2^*AY_2 \end{bmatrix}. \end{aligned}$$

Since U_1 is unitary then $X_2^*Ay_1 = 0$ by construction and therefore

$$U_1^*AV_1 = \begin{bmatrix} \alpha_{11} & x_1^*AY_2 \\ 0 & X_2^*AY_2 \end{bmatrix}.$$

Similarly we have

$$U_1^* B V_1 = \begin{bmatrix} x_1^* B y_1 & x_1^* B Y_2 \\ X_2^* B y_1 & X_2^* B Y_2 \end{bmatrix}.$$

As y_1 is an eigenvector then By_1 is either proportional to Ay_1 or is the zero vector. Therefore we have

$$U_1^* B V_1 = \begin{bmatrix} \beta_{11} & x_1^* B Y_2 \\ 0 & X_2^* B Y_2 \end{bmatrix}.$$

Using our inductive hypothesis there exist unitary matrices $\widetilde{U}_2, \widetilde{V}_2 \in \mathbb{C}^{(k-1)\times(k-1)}$ such that both $\widetilde{U}_2^* X_2^* A Y_2 \widetilde{V}_2$ and $\widetilde{U}_2^* X_2^* B Y_2 \widetilde{V}_2$ are triangular. If we define $U_2 = \mathrm{diag}(1, \widetilde{U}_2)$ and $V_2 = \mathrm{diag}(1, \widetilde{V}_2)$ then we have

$$U_2^* U_1^* A V_1 V_2 = \begin{bmatrix} \alpha_{11} & x_1^* A Y_2 \widetilde{V}_2 \\ 0 & \widetilde{U}_2 X_2^* A Y_2 \widetilde{V}_2 \end{bmatrix},$$

$$U_2^* U_1^* B V_1 V_2 = \begin{bmatrix} \beta_{11} & x_1^* B Y_2 \widetilde{V}_2 \\ 0 & \widetilde{U}_2 X_2^* B Y_2 \widetilde{V}_2 \end{bmatrix},$$

which are both upper triangular. □

1.4.3 The Hermitian Positive Definite Generalized Eigenvalue Problem

A natural generalization of the Hermitian eigenvalue problem would be to have Hermitian A and B. Unfortunately this is not enough to guarantee the nice properties that occur with the Hermitian eigenvalue problem such as real eigenvalues and linearly independent eigenvectors. For example if we take

$$A = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

then the eigenvalue of (A, B) are $\lambda = \pm i$. If we also consider the fact that any matrix can be written as $C = B^{-1}A$ or AB^{-1} where both A and B are Hermitian then pathological examples can be generated where, for example, C is defective. An additional condition is required to ensure nice properties. A common form

of the generalized eigenvalue problem, which occurs in many applications, is the Hermitian positive definite generalized problem. This is when both A and B are Hermitian and at least one of the matrices, B say, is positive definite.

When reducing the pair (A, B) to a more desirable form we would like to preserve the Hermitian nature of the problem, but in general equivalent matrices UAV and UBV are not Hermitian. Instead we use a special kind of equivalence transformations called CONGRUENCE transformations. For nonsingular X we have

$$\widetilde{A} = X^*AX, \quad \widetilde{B} = X^*BX$$

where $\widetilde{A}$ and $\widetilde{B}$ are Hermitian. Using these transformations we can obtain an analogue of Theorem 1.3.5.

Theorem 1.4.5 *Let (A, B) be a Hermitian matrix pair where B is positive definite. Then there exists a nonsingular matrix X such that*

$$X^*AX = \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n),$$
$$X^*BX = I_n.$$

The eigenvalues of (A, B) are the real values $\lambda_1, \ldots, \lambda_n$ and the corresponding eigenvectors are the columns of the matrix X.

Proof. As B is positive definite it has a Hermitian positive definite square root $B^{1/2}$. If we apply the congruence transformation

$$A_1 = B^{-1/2}AB^{-1/2}, \quad B_1 = B^{-1/2}BB^{-1/2} = I_n$$

the problem will have been reduced to a Hermitian eigenvalue problem, $A_1y = \lambda y$ where $y = B^{1/2}x$. If we apply Theorem 1.3.5 then we can see that there exists an unitary matrix Q that diagonalizes A_1. This gives us

$$A_2 = Q^*A_1Q = \Lambda, \quad B_2 = Q^*I_nQ = I_n,$$

where Λ is a diagonal matrix whose elements are real. By setting $X = B^{-1/2}Q$ we see that X^*AX is diagonal and $X^*BX = I_n$. Using Theorem 1.4.3 we see that the eigenvalues of (A, B) are the diagonal elements of Λ. As the eigenvalues of (Λ, I_n) are the columns of the identity matrix e_i, then the eigenvalues of (A, B) are the columns of the matrix X. □

The proof of Theorem 1.4.5 illustrates a way in which the Hermitian positive definite generalized eigenvalue problem can be reduced to a Hermitian eigenvalue problem. As B is positive definite then it can be factorized in the form $B = SS^*$ where S is nonsingular. The generalized eigenvalue problem is then equivalent to the problem

$$(S^{-1}AS^{-*})y = \lambda y \tag{1.24}$$

where $y = S^*x$. In the proof of Theorem 1.4.5, S was the Hermitian positive definite square root $B^{1/2}$ but other, possibly less expensive factors can be formed. Peters and Wilkinson [48] note that probably the first procedure of this kind, used by G. H. Golub on the Illiac at the University of Illinois, was based on finding the spectral decomposition of B using Jacobi's method (see Section 2.1). As B is positive definite it has positive eigenvalues. Therefore

$$B = Q\,\mathrm{diag}(\mu_i)Q^* = (QD)(QD)^*, \quad \text{where } D = \mathrm{diag}(\mu_i^{1/2})$$

and hence we may take $S = QD$. The resulting Hermitian eigenvalue problem was also solved using Jacobi's method.

However, there are more economical ways of forming the factorization of B. The most popular is to form the Cholesky factorization, where S is a lower triangular matrix. This forms the basis of the Cholesky method which is described in Section 3.2.

Although the condition that B is positive definite occurs frequently in practice, we can extend Theorem 1.4.5 to include definite pairs.

Definition 1.4.6 *The Hermitian pair (A, B) is a* DEFINITE *pair if*

$$\gamma(A,B) = \min_{\substack{x\in\mathbb{C}^n \\ \|x\|_2=1}} |x^*(A+iB)x| = \min_{\substack{x\in\mathbb{C}^n \\ \|x\|_2=1}} \sqrt{(x^*Ax)^2 + (x^*Bx)^2} > 0.$$

The quantity γ is known as the Crawford number.

Definite pairs can be transformed into a pair where B is positive definite using the following theorem.

Theorem 1.4.7 *Let (A, B) be a Hermitian definite pair and for $\theta \in \mathbb{R}$ let*

$$A_\theta = A\cos\theta + B\sin\theta,$$
$$B_\theta = -A\sin\theta + B\cos\theta.$$

Then there exists a $\theta \in [0, 2\pi)$ such that B_θ is positive definite and

$$\gamma(A,B) = \lambda_{\min}(B_\theta).$$

Proof. See Stewart and Sun [55]. □

The eigenvalues of (A, B) and (A_θ, B_θ), represented by (α, β) and $(\alpha_\theta, \beta_\theta)$ are related by

$$\begin{bmatrix} \beta \\ \alpha \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \beta_\theta \\ \alpha_\theta \end{bmatrix}.$$

The eigenvectors of (A, B) and (A_θ, B_θ) are the same. If follows that definite pairs have real eigenvalues and are simultaneously diagonalizable.

Chapter 2

Numerical Methods for the Symmetric Eigenvalue Problem

We discuss two popular methods for solving the symmetric eigenvalue problem, Jacobi's method and the symmetric QR algorithm.

2.1 Jacobi's Method

This is one of the oldest numerical methods for the eigenvalue problem, dating back to a paper by Jacobi [36] in 1846. Given a symmetric matrix $A = A_0$, Jacobi's method produces a sequence of orthogonally similar matrices $A_0, A_1, A_2, \ldots$ using the iteration $A_{i+1} = J_i^T A_i J_i$ where J_i is orthogonal and is known as a Jacobi rotation. Providing the Jacobi rotations are chosen correctly A_i will converge to a diagonal matrix as $i \to \infty$. Each Jacobi rotation is designed such that for an index pair (p, q) where $1 \leq p < q \leq n$, $A_{i+1} = J_i^T A_i J_i$ has zeros in the (p, q) and

(q, p) positions. The Jacobi rotation has the form

$$J_i = J(p, q, \theta) = \begin{bmatrix} I_{p-1} & & & & \\ & c & & -s & \\ & & I_{q-p-1} & & \\ & s & & c & \\ & & & & I_{n-q} \end{bmatrix}, \tag{2.1}$$

where $c = \cos\theta$ and $s = \sin\theta$. To determine θ we only have to consider the elements in the (p, q)-plane where the 2×2 restriction matrix is given by

$$\begin{bmatrix} \tilde{a}_{pp} & \tilde{a}_{pq} \\ \tilde{a}_{pq} & \tilde{a}_{qq} \end{bmatrix} = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}^T \begin{bmatrix} a_{pp} & a_{pq} \\ a_{pq} & a_{qq} \end{bmatrix} \begin{bmatrix} c & -s \\ s & c \end{bmatrix}. \tag{2.2}$$

Multiplying out the right hand side of (2.2) and setting the off-diagonal elements of the left hand side to zero we have

$$\begin{bmatrix} \tilde{a}_{pp} & 0 \\ 0 & \tilde{a}_{qq} \end{bmatrix} = \begin{bmatrix} a_{pp}c^2 - 2sca_{pq} + a_{qq}s^2 & (c^2 - s^2)a_{pq} + (a_{pp} - a_{qq})sc \\ (c^2 - s^2)a_{pq} + (a_{pp} - a_{qq})sc & a_{pp}s^2 + 2sca_{pq} + a_{qq}c^2 \end{bmatrix}.$$

This new matrix will be diagonal if

$$(c^2 - s^2)a_{pq} + (a_{pp} - a_{qq})sc = 0 \tag{2.3}$$

or

$$\tau := \frac{a_{qq} - a_{pp}}{2a_{pq}} = \frac{c^2 - s^2}{2sc}. \tag{2.4}$$

Substituting (2.4) into (2.3) we obtain a quadratic equation in $t = \tan\theta$

$$t^2 + 2\tau t - 1 = 0,$$

whose solutions are

$$t = -\tau \pm \sqrt{1 + \tau^2}.$$

The smaller root in magnitude is chosen,

$$t = -\tau + \text{sign}(\tau)\sqrt{1 + \tau^2} = \frac{\text{sign}(\tau)}{(|\tau| + \sqrt{1 + \tau^2})}, \tag{2.5}$$

where sign(x) is defined by

$$\text{sign}(x) = \begin{cases} 1 & \text{if } x \geq 0, \\ -1 & \text{if } x < 0. \end{cases}$$

Then

$$c = \frac{1}{\sqrt{1+t^2}}, \quad s = tc. \tag{2.6}$$

Choosing t to be the smaller of the two roots makes $|\theta| \leq \pi/4$ and therefore c is positive. There is another solution, $\pi/2+\theta$, but the smaller angle ensures convergence of the Algorithms 2.1.1 and 2.1.3. After further trigonometric manipulation we find that

$$\begin{bmatrix} \tilde{a}_{pp} & 0 \\ 0 & \tilde{a}_{qq} \end{bmatrix} = \begin{bmatrix} a_{pp} - a_{pq}t & 0 \\ 0 & a_{qq} + a_{pq}t \end{bmatrix}. \tag{2.7}$$

When overwriting A with $J(p,q,\theta)^T A J(p,q,\theta)$ it is noticeable that only the rows and columns, p and q change, with $\tilde{a}_{pq}$ and $\tilde{a}_{qp}$ becoming zero. We now examine how this helps reduce the Frobenius norm of the off-diagonal elements,

$$\text{off}\,(A) = \sqrt{\sum_{p=1}^{n}\sum_{q\neq p} a_{pq}^2} = \sqrt{\|A\|_F^2 - \sum_{i=1}^{n} a_{ii}^2}. \tag{2.8}$$

If $\widetilde{A} = J(p,q,\theta)^T A J(p,q,\theta)$, then

$$\text{off}(\widetilde{A})^2 = \|J^T A J\|_F^2 - \sum_{i=1}^{n} \tilde{a}_{ii}^2 = \|A\|_F^2 - \sum_{i=1}^{n} \tilde{a}_{ii}^2,$$

since $J(p,q,\theta)$ is orthogonal. Then

$$\begin{aligned} \text{off}(\widetilde{A})^2 &= \text{off}\,(A)^2 + \sum_{i=1}^{n} a_{ii}^2 - \sum_{i=1}^{n} \tilde{a}_{ii}^2 \\ &= \text{off}\,(A)^2 + (a_{pp}^2 + a_{qq}^2 - \tilde{a}_{pp}^2 - \tilde{a}_{qq}^2), \end{aligned}$$

as only rows and columns p, q are changed. By taking the Frobenius norm in (2.2) we obtain

$$a_{pp}^2 + a_{qq}^2 + 2a_{pq}^2 = \tilde{a}_{pp}^2 + \tilde{a}_{qq}^2.$$

Therefore,

$$\mathrm{off}(\widetilde{A})^2 = \mathrm{off}\,(A)^2 - 2a_{pq}^2\ , \tag{2.9}$$

showing that A is brought closer to diagonal form with each Jacobi iteration.

Algorithm 2.1.1 (Basic Jacobi Algorithm) *Given a symmetric matrix A and a suitable strategy to choose the index pair (p,q) at each iteration, this algorithm computes the spectral decomposition $A = Q\Lambda Q^T$.*

$Q = I_n$

repeat

  Choose index pair (p,q)

  if $|a_{pq}|$ is not too small

    $\tau = (a_{qq} - a_{pp})/(2a_{pq})$

    $t = \mathrm{sign}(\tau)/(|\tau| + \sqrt{1+\tau^2})$

    $c = 1/\sqrt{1+t^2}$, $s = tc$

    $A([p\ q],:) = \left[\begin{smallmatrix} c & -s \\ s & c \end{smallmatrix}\right]^T A([p\ q],:)$

    $A(:,[p\ q]) = A(:,[p\ q])\left[\begin{smallmatrix} c & -s \\ s & c \end{smallmatrix}\right]$

    $Q(:,[p\ q]) = Q(:,[p\ q])\left[\begin{smallmatrix} c & -s \\ s & c \end{smallmatrix}\right]$

  end if

until A is sufficiently diagonal

$\Lambda = A$

We still need a suitable strategy for choosing (p,q). In view of (2.9) it may be a good idea to choose (p,q) such that a_{pq}^2 is maximal so that the reduction of $\mathrm{off}(A)$ is maximized for each iteration. This is the basis of the classical Jacobi method.

Lemma 2.1.2 *After one iteration of the classical Jacobi method we have*

$$\mathrm{off}(A_1) \le \sqrt{1 - \frac{1}{N}}\ \mathrm{off}(A_0),$$

where $N = n(n-1)/2$ is half the number of off-diagonal elements of A. After k iterations we have

$$\text{off}(A_k) \leq \left(1 - \frac{1}{N}\right)^{k/2} \text{off}(A_0).$$

Proof. Since a_{pq} is the largest off-diagonal entry we have $\text{off}(A_0)^2 \leq N(a_{pq}^2 + a_{qp}^2)$ and therefore

$$2a_{pq}^2 \geq \frac{1}{N}\text{off}(A_0)^2.$$

Substituting this into (2.9) we get the result. □

This suggests that the classical Jacobi method converges linearly. However, the asymptotic convergence rate has been shown to be quadratic [50]. This means that for large enough k

$$\text{off}(A_{k+N}) = O(\text{off}(A_k)^2).$$

In practice the classical Jacobi method is not used. This is due to the fact that each Jacobi update, computing $A_{i+1} = J_i^T A_i J_i$, involves $O(n)$ flops whereas the search for the optimal (p, q) involves $O(n^2)$. Therefore for large n the search time would dominate. Instead of this we fix the sequence of (p, q) beforehand. A reasonable sequence is to choose (p, q) to zero out each off-diagonal element in a row-by-row fashion as follows

$$(p, q) = (1, 2) \ldots (1, n), (2, 3) \ldots (2, n) \ldots (n-1, n). \tag{2.10}$$

This ordering scheme is known as cyclic-by-row and gives the following procedure.

Algorithm 2.1.3 (Cyclic Jacobi Algorithm) *Given a symmetric matrix A this algorithm computes the spectral decomposition $A = Q\Lambda Q^T$. A sequence of orthogonally similar matrices $A_0, A_1, \ldots$ is generated that zeros the off-diagonal elements in a row wise fashion until A_k is diagonal.*

$Q = I_n$

repeat

for $p = 1{:}\, n-1$

for $q = i+1{:}\, n$

Compute Jacobi update, $A = J_i^T A J_i$ as in Algorithm 2.1.1

end for

end for

until A is sufficiently diagonal

$\Lambda = A$

The cyclic Jacobi method is also asymptotically quadratically convergent [62]. However it is considerably faster than the classical Jacobi method as it does not require a search of the off-diagonal elements at each step.

2.1.1 Error Analysis

The Jacobi methods are backward stable algorithms. This means that the computed diagonal matrix $A_k = \text{diag}(\widehat{\lambda}_i)$ satisfies

$$Q^T(A+E)Q = \text{diag}(\widehat{\lambda}_i)$$

where Q is some orthogonal matrix and

$$\|E\|_2 \leq c_n u \|A\|_2$$

where u is the unit roundoff and c_n is a slow growing polynomial in n. This is easy to show and is a consequence of using only orthogonal transformations (it is also a specific case of the analysis of Section 3.3 where we take $L = D = I_n$). To examine the forward error we consider the following lemma [26, Corollary 8.1.6], [55, pp 203].

Lemma 2.1.4 *If $A, A+E \in \mathbb{R}^{n\times n}$ are symmetric matrices, then*

$$|\lambda_k(A+E) - \lambda_k(A)| \leq \|E\|_2$$

for $k = 1, \ldots, n$.

Using this lemma we can show that

$$|\widehat{\lambda}_i - \lambda_i| \leq c_n u \|A\|_2 \leq c_n u \max_i |\lambda_i|. \tag{2.11}$$

The large eigenvalues of A (those $\widehat{\lambda}_j$ that are close to $\|A\|_2$ in magnitude) will therefore be computed to high relative accuracy while the small eigenvalues may have very few correct digits. When A is positive definite, a more refined componentwise error analysis by Demmel and Veselić [10] shows that

$$\frac{|\widehat{\lambda}_i - \lambda_i|}{|\lambda_i|} \leq c_n u \kappa_2(D^{-1}AD^{-1}) \tag{2.12}$$

where $D = \mathrm{diag}(\sqrt{a_{11}}, \ldots, \sqrt{a_{nn}})$. This can be a much smaller bound than that in (2.11) if the elements of D vary widely in magnitude. If we write

$$A = DHD, \qquad D = \mathrm{diag}(A)^{1/2}, \tag{2.13}$$

so that H has unit diagonal, a result of van der Sluis [60] states that

$$\kappa_2(H) \leq n \min_{F \text{ diagonal}} \kappa_2(FAF). \tag{2.14}$$

Thus H comes within a factor n of minimizing the 2-norm condition number over all two-sided diagonal scalings of A. The bound (2.12) is valid provided we terminate the algorithm when all off-diagonal elements of A_k satisfy

$$|a_{ij}| \leq u\sqrt{a_{ii}a_{jj}}.$$

These high accuracy properties mean that Jacobi's method is interesting. This method has a larger operation count than other methods. Each sweep takes approximately $3n^3$ flops ($6n^3$ flops if eigenvectors are required) and Jacobi's method often takes 5-10 sweeps to converge. However, flops are only a guide to the speed of an algorithm and Jacobi's method is particularly suitable for computation in a parallel environment which would increase the efficiency of the algorithm considerably.

2.2 The Symmetric QR Algorithm

The symmetric QR algorithm is a two part procedure. The first part involves finding an orthogonal Q such that $Q^TAQ = T$ is tridiagonal. The next stage is to perform a series of tridiagonal QR iterations on the matrix T until it has been reduced to diagonal form. This second part requires only $O(n^2)$ flops. The reduction of a matrix to tridiagonal form costs $\frac{4}{3}n^3$ flops ($\frac{8}{3}n^3$ flops if eigenvectors are required) so this part of the procedure will dominate the operation count when n is large. This algorithm is used in MATLAB's `eig` command [42], when the input argument A is symmetric, and in the LAPACK routines `xSYEV` for dense matrices and `xSTEV` for tridiagonal matrices [1].

2.2.1 Reduction to Tridiagonal Form

The reduction to tridiagonal form is done with a series of Householder matrices. A Householder matrix has the form

$$H = I_n - \beta v v^T, \quad \beta = 2/(v^T v) \tag{2.15}$$

where the nonzero vector $v \in \mathbb{R}^n$ is known as the Householder vector. It is not hard to see that H is both symmetric and orthogonal. If $x, y \in \mathbb{R}^n$ are distinct vectors such that $\|x\|_2 = \|y\|_2$ then there exists a unique Householder matrix H, where the Householder vector is $v = x - y$, such that $Hx = y$.

We partition A in the form

$$A = \begin{bmatrix} \alpha_{11} & a^T \\ a & A_{22} \end{bmatrix}$$

and then determine a Householder matrix such that $Ha = \gamma e_1$ where e_1 is the first column of the identity matrix. By taking

$$v = a \pm \|a\|_2 e_1 \tag{2.16}$$

we can see that

$$\begin{aligned}\beta = 2/(v^T v) &= 1/\left(\|a\|_2^2 \pm a_1\|a\|_2\right), \\ vv^T a &= (a \pm \|a\|_2 e_1)(a \pm \|a\|_2 e_1)^T a \\ &= \left(\|a\|_2^2 \pm a_1\|a\|_2\right)(a \pm \|a\|_2 e_1)\end{aligned}$$

where a_1 is the first element in the vector a. Using this we get

$$Ha = \left(I - \beta vv^T\right) a = \mp\|a\|_2 e_1.$$

For the choice of sign in (2.16) we notice that choosing a minus means that Ha is a positive multiple of e_1. However, this could leads to dangerous cancellation, if we used the formula in (2.16), when a is close to a positive multiple of e_1. Instead we can calculate the first element of the vector v using

$$v_1 = a_1 - \|a\|_2 = \frac{a_1^2 - \|a\|_2^2}{a_1 + \|a\|_2} = \frac{-(a_2^2 + \cdots + a_n^2)}{a_1 + \|a\|_2}.$$

This formula, suggested by Parlett [46], does not suffer from any cancellation problems when $a_1 > 0$. Overall we have

$$\operatorname{diag}(1, H) A \operatorname{diag}(1, H) = \begin{bmatrix} \alpha_{11} & \|a\|_2 e_1^T \\ \|a\|_2 e_1 & HA_{22}H \end{bmatrix}. \tag{2.17}$$

For the computation of $HA_{22}H$ we can take advantage of symmetry. Using the formula for a Householder matrix we have

$$HA_{22}H = (I - \beta vv^T)A_{22}(I - \beta vv^T).$$

If we take $p = \beta A_{22} v$ and then

$$w = p - \frac{\beta p^T v}{2} v$$

we get

$$HA_{22}H = A_{22} - vw^T - wv^T.$$

After one Householder transformation the matrix in (2.17) has the form

$$\operatorname{diag}(1,H)A\operatorname{diag}(1,H) = \left[\begin{array}{c|cccc} \alpha_{11} & + & 0 & 0 & 0 \\ \hline + & + & \oplus & \oplus & \oplus \\ 0 & \oplus & + & + & + \\ 0 & \oplus & + & + & + \\ 0 & \oplus & + & + & + \end{array}\right].$$

The reduction continues by recursively reducing $HA_{22}H$ to tridiagonal form, which involves finding Householder matrices such that $H(\oplus,\cdots,\oplus)^T = \gamma e_1$:

$$\left[\begin{array}{c|cccc} + & + & 0 & 0 & 0 \\ \hline + & + & \oplus & \oplus & \oplus \\ 0 & \oplus & + & + & + \\ 0 & \oplus & + & + & + \\ 0 & \oplus & + & + & + \end{array}\right] \overset{H_2AH_2}{\longrightarrow} \left[\begin{array}{cc|ccc} + & + & 0 & 0 & 0 \\ + & + & + & 0 & 0 \\ \hline 0 & + & + & \oplus & \oplus \\ 0 & 0 & \oplus & + & + \\ 0 & 0 & \oplus & + & + \end{array}\right]$$

$$\overset{H_3AH_3}{\longrightarrow} \left[\begin{array}{ccccc} + & + & 0 & 0 & 0 \\ + & + & + & 0 & 0 \\ 0 & + & + & + & 0 \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{array}\right].$$

Algorithm 2.2.1 (Tridiagonal Reduction using Householder Matrices) *Given a symmetric matrix A this algorithm computes $T = Q^TAQ$ where T is tridiagonal and $Q = H_1 \ldots H_{n-2}$ is the product of $n-2$ Householder matrices.*

$Q = I_n$

for $k = i\colon n-2$

Calculate the Householder vector

$\sigma = A(k+2\colon n,k)^TA(k+2\colon n,k)$

$v = A(k+1\colon n,k); v(1) = 1$

if $\sigma = 0$

$\beta = 0$

else

$\mu = \sqrt{a_{k+1,k}^2 + \sigma}$

if $a_{k+1,k} \leq 0$

$v(1) = a_{k+1,k} - \mu$

else

$v(1) = -\sigma/(a_{k+1,k} + \mu)$

end if

$\beta = 2v(1)^2/(\sigma + v(1)^2)$

$v = v/v(1)$

end if

Calculate the update $A = \text{diag}(I_k, H)A\,\text{diag}(I_k, H)$

$p = \beta A(k+1\colon n, k+1\colon n)v$

$w = p - \frac{\beta p^T v}{2} v$

$a_{k+1,k} = \|A(k+1\colon n, k)\|_2$; $a_{k,k+1} = a_{k+1,k}$

$A(k+2\colon n, k) = 0$; $A(k, k+2\colon n) = 0$

$A(k+1\colon n, k+1\colon n) = A(k+1\colon n, k+1\colon n) - vw^T - wv^T$

$Q = Q\,\text{diag}(I_k, H)$

end for

$T = A$

This is the most popular method for tridiagonal reduction. We can also reduce A to tridiagonal form using Givens rotations and this will allow for easier analysis (and possibly different stability properties) in Section 3.4. A Givens rotations has the same form as a Jacobi rotation in (2.1) except $c = \cos\theta$ and $s = \sin\theta$ are chosen such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

The Givens tridiagonal reduction works by computing Givens rotations to zero out elements in the following order:

$$\begin{bmatrix} + & + & x^1 & x^2 & x^3 \\ + & + & + & x^4 & x^5 \\ x^1 & + & + & + & x^6 \\ x^2 & x^4 & + & + & + \\ x^3 & x^5 & x^6 & + & + \end{bmatrix}.$$

If the Givens rotations are chosen in the (i, j)-plane sequence

$$(i, j) = (2, 3), (2, 4), \ldots, (2, n), (3, 4), \ldots, (3, n), \ldots, (n-2, n)$$

then none of the zeros introduced during the reduction will be affected by subsequent rotations.

Algorithm 2.2.2 (Tridiagonal Reduction using Givens Rotations) *Given a symmetric matrix A this algorithm computes $T = Q^T AQ$ where T is tridiagonal and Q is the product of Givens rotations.*

$Q = I_n$
for $i = 1{:}\, n-2$
  for $j = i+2{:}\, n$
    Compute Givens rotation such that $\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a_{i+1,i} \\ a_{ji} \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}$
    if $a_{ji} = 0$
      $c = 1, s = 0$
    else
      if $|a_{ji}| > |a_{i+1,i}|$
        $\tau = -a_{i+1,i}/a_{ji}$
        $s = 1/\sqrt{1+\tau^2}, c = s\tau$
      else

$\tau = -a_{ji}/a_{i+1,i}$

$c = 1/\sqrt{1+\tau^2}, s = c\tau$

end

end

$A([i+1\,j],:) = \left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]^T A([i+1\,j],:)$

$A(:,[i+1\,j]) = A(:,[i+1\,j])\left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]$

$Q(:,[i+1\,j]) = Q(:,[i+1\,j])\left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]$

end

end

$T = A$

2.2.2 Tridiagonal QR Iteration

The tridiagonal QR iteration computes a sequence $T_0, T_1, \ldots$ of orthogonally similar tridiagonal matrices starting with $T = T_0$ such that T_i converges to diagonal form. We can assume T is unreduced, that is T has no zero subdiagonal elements, as otherwise we could split the eigenvalue problem into two smaller unreduced ones. If for some k we have $t_{k,k+1} = t_{k+1,k} = 0$ then

$$T - \lambda I = \begin{array}{c} \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{bmatrix} T_{11} - \lambda I & 0 \\ 0 & T_{22} - \lambda I \end{bmatrix} \end{array}.$$

The basic form of the iteration is to compute the QR factorization of the matrix $T_k - \mu I$ where μ is a shift. Then

$$T_{k+1} = RQ + \mu I = Q^T T_k Q.$$

This is called a shifted QR step. The updated matrix T_{k+1} will be tridiagonal as Q will be upper Hessenberg and R will be upper triangular. This means $RQ + \mu I$

will be upper Hessenberg and as it is also symmetric, it will be lower Hessenberg and therefore tridiagonal.

The purpose of shifts is to help T to converge to diagonal form. A perfect shift would be to choose $\mu \in \lambda(T)$. Then the matrix $T - \mu I$ will have rank less than n and the QR factorization of $T - \mu I$ will have an upper triangular R where $r_{nn} = 0$. Therefore RQ will have the form

$$\begin{bmatrix} + & + & 0 & 0 & 0 \\ + & + & + & 0 & 0 \\ 0 & + & + & + & 0 \\ 0 & 0 & + & + & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the last column of $T_{k+1} = RQ + \mu I$ will equal μe_n. We would then be able to reduce the problem to an $(n-1) \times (n-1)$ problem and then iterate recursively until T is diagonal. If we use a good approximate eigenvalue for μ then we suspect the $(n, n-1)$ element of T will be small. We denote the kth iterate

$$T_k = \begin{bmatrix} a_1 & b_1 & & \\ b_1 & \ddots & \ddots & \\ & \ddots & \ddots & b_{n-1} \\ & & b_{n-1} & a_n \end{bmatrix}. \tag{2.18}$$

The simplest choice of shift is $\mu = a_n$. This is known as the single shift QR iteration. It is cubically convergent for almost all matrices but examples exist where this iteration does not converge. To get global convergence we use the eigenvalue of

$$\begin{bmatrix} a_{n-1} & b_{n-1} \\ b_{n-1} & a_n \end{bmatrix}$$

which is closest to a_n. This is known as the Wilkinson shift and is given by

$$\mu = a_n + d - \text{sign}(d)\sqrt{d^2 + b_{n-1}^2} \tag{2.19}$$

where $d = (a_{n-1} - a_n)/2$. Using this shift the iteration is globally, and at least linearly, convergent. It is asymptotically cubically convergent for almost all matrices.

Instead of applying the shifted QR step explicitly by computing

$$\begin{aligned} Q_k R_k &= T_k - \mu I, \\ T_{k+1} &= R_k Q_k + \mu I, \end{aligned} \tag{2.20}$$

we can form T_{k+1} without forming $T_k - \mu I$. This has advantages when the shift is a lot larger than some of the diagonal elements of T_k. We use the following theorem [26].

Theorem 2.2.3 (Implicit Q Theorem) *Let $Q = [q_1, \ldots, q_n]$ and $V = [v_1, \ldots, v_n]$ be two orthogonal matrices such that both $Q^T A Q = S$ and $V^T A V = T$ are upper Hessenberg where $A \in \mathbb{R}^{n \times n}$. Let k denote the smallest index for which $s_{k+1,k} = 0$, with the convention that $k = n$ if S is unreduced. If $q_1 = v_1$ then*

$$v_i = \pm q_i, \quad |t_{i,i-1}| = |s_{i,i-1}|$$

for $i = 2\colon k$. Also if $k < n$ then $t_{k+1,k} = 0$.

Proof. If we define the orthogonal matrix $W = V^T Q = [w_1, \ldots, w_n]$, we can see that

$$TW = WS$$

and that the $(i-1)$th column of each side of this equation can be given

$$T w_{i-1} = \sum_{j=1}^{i} s_{j,i-1} w_j.$$

Rearranging this we get

$$s_{i,i-1} w_i = T w_{i-1} - \sum_{j=1}^{i-1} s_{j,i-1} w_j. \tag{2.21}$$

Since $q_1 = v_1$ then $w_1 = e_1$ it follows from (2.21) and the fact that T is upper Hessenberg that $[w_1, \ldots, w_k]$ is upper triangular. As W is also orthogonal

$$[w_1, \ldots, w_k] = \text{diag}(\pm 1, \ldots, \pm 1)$$

and therefore $q_i = \pm v_i$ for $i = 2\colon k$. Using this we get

$$|s_{i,i-1}| = |q_i^T A q_{i-1}| = |v_i^T A v_{i-1}| = |t_{i,i-1}|.$$

If $k < n$

$$\begin{aligned} t_{k+1,k} &= e_{k+1}^T T e_k = e_{k+1}^T T W e_k \\ &= e_{k+1}^T W S e_k = e_{k+1}^T \sum_{i=1}^{k} s_{ik} W e_i \\ &= \sum_{i=1}^{k} s_{ik} e_{k+1}^T e_i = 0. \end{aligned}$$

□

When A is symmetric $Q^T A Q$ and $V^T A V$ are tridiagonal matrices. The basic idea of the theorem is that to compute the step (2.20) all we have to do is to compute an orthogonal matrix G_1 such that its first column is equal to that of Q_k. We then compute a second orthogonal matrix $\widetilde{G}$ such that $\widetilde{G} G_1^T A G_1 \widetilde{G}$ is tridiagonal and $G_1 \widetilde{G}$ has the same first column as Q_k .

The procedure starts by computing a Givens rotation, $G_1 = G(1, 2, \theta)$ where c and s satisfy

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a_1 - \mu \\ b_1 \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

If we apply this transformation to T we have

$$T \leftarrow G_1^T T G_1 = \begin{bmatrix} + & + & \oplus & 0 & 0 \\ + & + & + & 0 & 0 \\ \oplus & + & + & + & 0 \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}.$$

We can now chase the unwanted nonzero element $\oplus$ out of the matrix $G_1^T T G_1$ by computing Givens rotations to zero it out. We compute $G_i = G(i, i+1, \theta)$ to zero out $\oplus$ but each time we do this another unwanted nonzero element appears in the $(i, i+2)$ position. We continue until this element falls off the end of the matrix:

$$\xrightarrow{G_1} \begin{bmatrix} + & + & \oplus & 0 & 0 \\ + & + & + & 0 & 0 \\ \oplus & + & + & + & 0 \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix} \xrightarrow{G_2} \begin{bmatrix} + & + & 0 & 0 & 0 \\ + & + & + & \oplus & 0 \\ 0 & + & + & + & 0 \\ 0 & \oplus & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}$$

$$\xrightarrow{G_3} \begin{bmatrix} + & + & 0 & 0 & 0 \\ + & + & + & 0 & 0 \\ 0 & + & + & + & \oplus \\ 0 & 0 & + & + & + \\ 0 & 0 & \oplus & + & + \end{bmatrix} \xrightarrow{G_4} \begin{bmatrix} + & + & 0 & 0 & 0 \\ + & + & + & 0 & 0 \\ 0 & + & + & + & 0 \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}.$$

If we accumulate the Givens rotation $\widetilde{Q} = G_1 \cdots G_{n-1}$, then we have the updated tridiagonal matrix $\widetilde{T}_{k+1} = \widetilde{Q}^T T_k \widetilde{Q}$. By invoking the Theorem 2.2.3 we can show that $\widetilde{Q}$ and $\widetilde{T}_{k+1}$ are equivalent to Q and T_{k+1} in (2.20) up to signs. The first column of $\widetilde{Q}$ is $\widetilde{q}_1 = G_1 \ldots G_{n-1} e_1 = G_1 e_1$ as the first columns of $G_2, \ldots, G_{n-1}$ are e_1. As G_1 is computed to zero the first subdiagonal element of $T_k - sI$ we have $\widetilde{q}_1 = q_1$ and therefore by Theorem 2.2.3 T_{k+1} and $\widetilde{T}_{k+1}$ are the same up to signs.

Algorithm 2.2.4 (Implicit QR Step with Wilkinson Shift) *Given an unreduced tridiagonal T this algorithm computes the updated matrix $Q^T T Q$ where $Q = G_1 \ldots G_{n-1}$ is the product of Givens rotations such that $Q^T(T - \mu I)$ is upper triangular. The scalar μ is called the Wilkinson shift and is the eigenvalue of T's lower 2×2 principal submatrix that is closest to a_n.*

$Q = I_n$

$d = (t_{n-1,n-1} - t_{nn})/2$

$\mu = t_{nn} - \frac{t_{n,n-1}^2}{d + \text{sign}(d)\sqrt{d^2 + t_{n,n-1}^2}}$

$x = t_{11} - \mu \ ; \ y = t_{21}$

for $k = 1\!:\!n-1$

Find Givens rotation G_k such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

$T([k\ k+1],:) = \left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]^T T([k\ k+1],:)$

$T(:,[k\ k+1]) = T(:,[k\ k+1])\left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]$

$Q(:,[k\ k+1]) = Q(:,[k\ k+1])\left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]$

if $k < n-1$

$x = t_{k+1,k} \ ; \ z = t_{k+2,k}$

end if

end for

Using this algorithm we can describe the full symmetric tridiagonal QR iteration. Before applying the QR step we need to monitor the subdiagonal of T in order to bring about decoupling whenever possible.

Algorithm 2.2.5 (Tridiagonal QR Iteration) *Given a symmetric tridiagonal matrix T this algorithm computes the spectral decomposition $T = Q\Lambda Q^T$. It uses the implicit symmetric QR step with Wilkinson shift.*

$Q = I_n$

while T is not diagonal

Set b_i to zero if $|b_i| \leq tol(|a_i| + |a_{i+1}|)$

Partition T such that

$$T = \begin{array}{c} \\ p \\ n-p-q \\ q \end{array} \begin{array}{c} \begin{array}{ccc} p & n-p-q & q \end{array} \\ \begin{bmatrix} T_{11} & 0 & 0 \\ 0 & T_{22} & 0 \\ 0 & 0 & T_{33} \end{bmatrix} \end{array}$$

where T_{22} is unreduced.

Apply Algorithm 2.2.4 to T_{22}.

end while

$\Lambda = T$

Both parts of the symmetric QR algorithm are backward stable. Therefore the computed diagonal matrix $A_k = \text{diag}(\widehat{\lambda}_i)$ satisfies

$$Q^T(A + E)Q = \text{diag}(\widehat{\lambda}_i)$$

where Q is some orthogonal matrix and

$$\|E\|_2 \leq c_n u \|A\|_2.$$

This means that the computed eigenvalues satisfy (2.11) and therefore the large eigenvalues of A will be computed to high relative accuracy while the small eigenvalues may have very few correct digits. For positive definite A we do not have a result similar to (2.12) that we have for Jacobi's method.

The reason why this method is so attractive is that when implemented efficiently it is currently the fastest algorithm for finding all eigenvalues of tridiagonal matrices, taking $O(n^2)$ flops. However if eigenvectors are required the QR iteration takes approximately $6n^3$ flops, if we consider that it takes on average 2 QR steps per eigenvalue to converge. This is only the fastest method for finding all eigenvalues and eigenvectors for small matrices, up to $n = 25$. For dense matrices the overall flop count for finding all eigenvalues and eigenvectors is $8\frac{2}{3}n^3 + O(n^2)$ flops. For larger matrices the symmetric QR algorithm can be outperformed

by bisection algorithms with inverse iteration and divide-and-conquer algorithms (see Demmel [12] for a performance comparison of these algorithms). The bisection algorithm with inverse iteration can fail to produce orthogonal eigenvectors when A has close eigenvalues. When this happens, reorthogonalization techniques such as Gram–Schmidt are needed to compute numerically orthogonal eigenvectors. These techniques are expensive requiring $O(n^3)$ flops. However Parlett and Dhillon [15] have proposed a new $O(n^2)$ flops algorithm, based on bisection, for computing all eigenvalues and eigenvectors of a tridiagonal matrix that does not require reorthogonalization techniques. This algorithm should be part of the next LAPACK release.

Chapter 3

Numerical Methods for the Symmetric Definite Generalized Eigenvalue Problem

3.1 The QZ Algorithm

The QZ algorithm [44] is a generalization of the QR algorithm for nonsymmetric matrices. Like the QR algorithm [47] it is a two part procedure. The first part of the QZ algorithm computes orthogonal matrices Q_1 and Z_1 such that Q_1AZ_1 is upper Hessenberg and Q_1BZ_1 is upper triangular. This is a generalization of the Householder reduction of a nonsymmetric matrix to upper Hessenberg form, which is the first part of the QR algorithm. The second stage computes orthogonal matrices Q_2 and Z_2 such that $Q_2Q_1AZ_1Z_2$ is quasi-triangular while $Q_2Q_1BZ_1Z_2$ remains upper triangular. A generalization of the Francis implicit double shift QR algorithm [23] is used. Therefore the original problem has been reduced to 1×1 and 2×2 subproblems. All that remains to do is to solve these subproblems and then calculate the eigenvectors. This algorithm is the method

of choice for solving the generalized eigenvalue problem when all eigenvalues are required and all we know about A and B are that they are dense. It is used in MATLAB's `qz` command [42] and in the LAPACK routine `xHGEQZ` [1].

3.1.1 Reduction to Hessenberg-Triangular Form

We start by computing the QR factorization of B. This is done by using a series of Householder matrices to zero out the subdiagonal elements of B. Let B be denoted by

$$B = \begin{bmatrix} \oplus & + & + & + & + \\ \oplus & + & + & + & + \\ \oplus & + & + & + & + \\ \oplus & + & + & + & + \\ \oplus & + & + & + & + \end{bmatrix}.$$

We find the Householder matrix H_1, similar to the procedure outlined in (2.15) and (2.16), such that $H_1(\oplus, \ldots, \oplus)^T = \gamma e_1$. Therefore

$$H_1 B = \begin{bmatrix} + & + & + & + & + \\ 0 & \oplus & + & + & + \\ 0 & \oplus & + & + & + \\ 0 & \oplus & + & + & + \\ 0 & \oplus & + & + & + \end{bmatrix}.$$

If this process is repeated using the vector containing the numbers marked with the $\oplus$ in $H_1 B$, the second column can be made into upper triangular form. After $(n-1)$ steps B will be upper triangular. To preserve the eigenvalues of (A, B) we have to perform the same transformations to A. Denoting the product of the Householder matrices by $Q = \text{diag}(I_{n-2}, H_{n-1}) \ldots H_1$ we have

$$A \to QA, \quad B \to QB.$$

The next step is to reduce A to upper-Hessenberg form while preserving the upper triangular form of B. This is done using a series of Givens rotations. We start by introducing a zero at the $(n,1)$ position of A. This is done by choosing c and s such that,

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a_{n-1,1} \\ a_{n,1} \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

Using this rotation, $Q_{n-1,n} = Q(n-1,n,\theta)$, we have

$$A \leftarrow Q_{n-1,n}^T A = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \end{bmatrix}, \; B \leftarrow Q_{n-1,n}^T B = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & \oplus & + \end{bmatrix}.$$

An unwanted element has been introduced to B. By postmultiplying by an appropriate Givens rotation we can restore the triangular form of B without disturbing the zero that has been introduced in A. We choose the Givens rotation to satisfy

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} b_{n,n-1} \\ b_{nn} \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix}.$$

Using this rotation, $Z_{n-1,n} = Z(n-1,n,\theta)$, only the $(n-1)$th and nth columns of A and B are affected and so we have

$$A \leftarrow AZ_{n-1,n} = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \end{bmatrix}, \; B \leftarrow BZ_{n-1,n} = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}.$$

We continue this technique to introduce zeros into A in the following order:

$$\begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ x^3 & + & + & + & + \\ x^2 & x^5 & + & + & + \\ x^1 & x^4 & x^6 & + & + \end{bmatrix}.$$

Each time an element of A is reduced to zero an unwanted element appears in B. By postmultiplying by a suitable Givens rotation we can eliminate this element without disturbing any of the zeros introduced into A. The whole algorithm for Hessenberg-triangular reduction is now given.

Algorithm 3.1.1 (Hessenberg-Triangular Reduction) *Given $A, B \in \mathbb{R}^{n\times n}$ this algorithm computes an upper Hessenberg matrix QAZ and an upper triangular matrix QBZ where Q and Z are orthogonal matrices.*

Compute QR factorization of B

for $i = 1\colon n-1$

Choose Householder matrix H_i such that $H_iB(i\colon n, i) = \gamma e_1$

$A = \text{diag}(I_{i-1}, H_i)A$

$B = \text{diag}(I_{i-1}, H_i)B$

end for

Reduce A to upper Hessenberg form while preserving triangular form of A

for $i = 1\colon n-2$

for $j = n\colon -1\colon i+2$

Choose Givens rotation such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a_{j-1,i} \\ a_{ji} \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

$A([j-1\ j], :) = \left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]^T A([j-1\ j], :)$

$B([j-1\ j], :) = \left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]^T B([j-1\ j], :)$

Choose Givens rotation such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} b_{j,j-1} \\ b_{jj} \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix}.$$

$$A(:,[j-1\ j]) = A(:,[j-1\ j])\begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

$$B(:,[j-1\ j]) = B(:,[j-1\ j])\begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

end for

end for

3.1.2 Zero-Chasing

It can be assumed without loss of generality that A is an unreduced upper-Hessenberg matrix as if for some k we have $a_{k+1,k} = 0$ then the problem can be split into two smaller unreduced problems:

$$A - \lambda B = \begin{array}{c} \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{bmatrix} A_{11} - \lambda B_{11} & A_{12} - \lambda B_{12} \\ 0 & A_{22} - \lambda B_{22} \end{bmatrix} \end{array}.$$

Also we can assume B is nonsingular as if for some k we have $b_{kk} = 0$ we can employ a zero-chasing technique and reduce the problem to a smaller $(n-1)\times(n-1)$ problem that involves a nonsingular B. We illustrate this with the following example:

$$A = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}.$$

We apply Givens rotations to chase the zero down the diagonal and maintain the upper Hessenberg shape of A. We start by zeroing the $(k+1,k+1)$th element of

B and then postmultiply by a suitable Givens rotation to eliminate the unwanted element introduced to A:

$$A \leftarrow Q_{34}^T A = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & \oplus & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B \leftarrow Q_{34}^T B = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}.$$

$$A \leftarrow A Z_{23} = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B \leftarrow B Z_{23} = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}.$$

If we continue this procedure we have

$$A \leftarrow Q_{45}^T A = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & \oplus & + & + \end{bmatrix}, \; B \leftarrow Q_{45}^T B = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

$$A \leftarrow A Z_{34} = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B \leftarrow B Z_{34} = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & 0 & + \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

We then apply a final rotation to zero $a_{n,n-1}$:

$$A \leftarrow AZ_{45} = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B \leftarrow BZ_{45} = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

From the diagram we can see that we have isolated the infinite eigenvalue of (A, B) and we are left with an unreduced $(n-1)\times(n-1)$ problem where $B(1\colon n-1, 1\colon n-1)$ is nonsingular.

3.1.3 The QZ Step

The QZ step involves the formation of the equivalent matrix pair $(\widetilde{A}, \widetilde{B}) = (Q^T AZ, Q^T BZ)$ where Q and Z are orthogonal matrices, $\widetilde{A}$ and $\widetilde{B}$ are upper Hessenberg and upper triangular respectively and $\widetilde{A}\widetilde{B}^{-1}$ is essentially the same matrix that would result if the Francis QR step is explicitly applied to $C = AB^{-1}$. We can see that

$$\widetilde{A}\widetilde{B}^{-1} = Q^T AB^{-1}Q$$

and therefore Q is essentially the orthogonal matrix that performs the QR step and Z is only used to maintain the structure of A and B. The Francis double shift QR step involves applying two single shift QR steps in succession using the eigenvalues a and b of

$$\bar{C} = \begin{bmatrix} c_{n-1,n-1} & c_{n-1,n} \\ c_{n,n-1} & c_{nn} \end{bmatrix}$$

as the shifts. The reason for this is that if the eigenvalues are complex then c_{nn} would tend to be a poor approximate eigenvalue. If two QR steps are applied using a and b we have

$$C - aI = U_1 R_1,$$

$$\begin{aligned} C_1 &= R_1U_1 + aI, \\ C_1 - bI &= U_2R_2, \\ C_2 &= R_2U_2 + bI. \end{aligned}$$

By manipulating these equations we get

$$M := (C - aI)(C - bI) = (U_1U_2)(R_2R_1). \tag{3.1}$$

If we rewrite (3.1)

$$\begin{aligned} M &= C^2 - (a+b)C + abI \\ &= C^2 - \text{trace}(\bar{C})C + \det(\bar{C})I \end{aligned}$$

we can see that M is a real matrix as $\text{trace}(\bar{C})$ and $\det(\bar{C})$ are both real. This means that (3.1) is the QR factorization of a real matrix where $Q = U_1U_2$ is orthogonal. As the Francis double shift basically involves forming Q^TCQ we can write an explicit version of the Francis double shift QR step as:

1. Explicitly form $M = (C - aI)(C - bI)$.
2. Compute the QR factorization of $M = QR$.
3. Compute $C_2 = Q^TCQ$.

The formation of M involves $O(n^3)$ flops. Fortunately we can implement this double shift strategy implicitly by invoking Theorem 2.2.3 and reduce the flop count to $O(n^2)$.

1. Compute m_1, the first column of M.
2. Determine a Householder matrix H_0 such that $H_0m_1 = \gamma e_1$.
3. Compute a sequence of Householder matrices $H_1, \ldots, H_{n-2}$ such that if $\widetilde{Q} = H_0 \ldots H_{n-2}$ then $\widetilde{Q}^TC\widetilde{Q}$ is upper Hessenberg and $\widetilde{q}_1 = q_1$.

The QZ step starts by computing the first column of $M = (C - aI)(C - bI)$ where $C = AB^{-1}$ and a and b are the eigenvalues of C's lower 2×2 submatrix. These eigenvalues are also the zeros of the 2×2 problem

$$\det(\bar{A} - \lambda \bar{B}) = 0$$

where

$$\bar{A} = \begin{bmatrix} a_{n-1,n-1} & a_{n-1,n} \\ a_{n,n-1} & a_{nn} \end{bmatrix}, \quad \bar{B} = \begin{bmatrix} b_{n-1,n-1} & b_{n-1,n} \\ 0 & b_{nn} \end{bmatrix}.$$

As we only require the first column of M we don't have to form C in its entirety. We can take advantage of the structure of A and B to show that

$$\begin{bmatrix} c_1 & c_2 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ 0 & b_{22} \end{bmatrix}^{-1} \tag{3.2}$$

where (c_1, c_2) and (a_1, a_2) denote the first two columns of C and A respectively. Also as C is upper Hessenberg then the first column of M will have the form $m_1 = (x, y, z, 0, \ldots, 0)^T$ where

$$\begin{aligned} x &= (c_{11} - a)(c_{11} - b) + c_{12}c_{21}, \\ y &= c_{21}(c_{11} - b) + c_{21}(c_{22} - a), \\ z &= c_{21}c_{32}. \end{aligned} \tag{3.3}$$

We then determine a Householder matrix H_0 such that $H_0 m_1 = \gamma e_1$. This matrix will have the form

$$H_0 = \begin{array}{c} \\ 3 \\ n-3 \end{array} \begin{array}{c} \begin{array}{cc} 3 & n-3 \end{array} \\ \begin{bmatrix} \widetilde{H}_0 & \\ & I_{n-3} \end{bmatrix} \end{array}.$$

Premultiplying by this matrix gives

$$A \leftarrow H_0 A = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ \oplus & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B \leftarrow H_0 B = \begin{bmatrix} + & + & + & + & + \\ \oplus & + & + & + & + \\ \oplus^1 & \oplus^1 & +^1 & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}.$$

We now work to restore the Hessenberg-triangular form by chasing the unwanted elements down the diagonal. We start by computing a Householder matrix $Z_1 = \mathrm{diag}(\widetilde{Z}_1, I_{n-3})$ such that $(\oplus^1, \oplus^1, +^1)\widetilde{Z}_1 = \gamma e_3^T$:

$$A \leftarrow AZ_1 = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ \oplus & + & + & + & + \\ \oplus & \oplus & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B \leftarrow BZ_1 = \begin{bmatrix} + & + & + & + & + \\ \oplus^2 & +^2 & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}.$$

Another Householder matrix, $Z_2 = \mathrm{diag}(\widetilde{Z}_2, I_{n-2})$ such that $(\oplus^2, +^2)\widetilde{Z}_2 = \gamma e_2^T$, is applied to return B to triangular form:

$$A \leftarrow AZ_2 = \begin{bmatrix} + & + & + & + & + \\ +^1 & + & + & + & + \\ \oplus^1 & + & + & + & + \\ \oplus^1 & \oplus & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B \leftarrow BZ_2 = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & 0 & + & + & + \\ 0 & 0 & 0 & + & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}.$$

We then reduce the first column of A to upper Hessenberg form using a Householder matrix $H_1 = \mathrm{diag}(1, \widetilde{H}_1, I_{n-4})$ such that $\widetilde{H}_1(+^1, \oplus^1, \oplus^1)^T = \gamma e_1$. Premultiplying we get

$$A \leftarrow H_1A = \begin{bmatrix} + & + & + & + & + \\ + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & \oplus & + & + & + \\ 0 & 0 & 0 & + & + \end{bmatrix}, \; B \leftarrow H_1B = \begin{bmatrix} + & + & + & + & + \\ 0 & + & + & + & + \\ 0 & \oplus^2 & +^2 & + & + \\ 0 & \oplus^1 & \oplus^1 & +^1 & + \\ 0 & 0 & 0 & 0 & + \end{bmatrix}.$$

If we continue this procedure recursively the matrix pair (A, B) will return to Hessenberg-triangular form. Notice that $Q^T = H_0 \ldots H_{n-2}$ has the same first column as H_0. Because of the way H_0 was determined we can apply Theorem 2.2.3

to show that the updated matrix pair (Q^TAZ, Q^TBZ) form

$$Q^TAZ(Q^TBZ)^{-1} = Q^TAB^{-1}Q$$

which is essentially the matrix we would have obtained if we had applied the Francis QR step to the matrix AB^{-1}.

Algorithm 3.1.2 (The QZ Step) *Given an unreduced upper Hessenberg matrix $A \in \mathbb{R}^{n\times n}$ and a nonsingular upper triangular matrix $B \in \mathbb{R}^{n\times n}$ this algorithm computes updated upper Hessenberg and upper triangular matrices Q^TAZ and Q^TBZ where Q and Z are orthogonal and the matrix $Q^TAZ(Q^TBZ)^{-1} = Q^TAB^{-1}Q$ is essentially the matrix we would have obtained if we had applied the Francis QR step to the matrix AB^{-1}.*

Compute the first column m_1 of $(C - aI)(C - bI)$ where $C = AB^{-1}$, a and b are the eigenvalues of C's lower 2×2 submatrix using (3.2) and (3.3).

Find Householder H_0 matrix such that $H_0m_1 = \gamma e_1$.

$A = H_0A, B = H_0B$

for $k = 1{:}\, n - 2$

Find 3×3 Householder matrix Z_{k1} such that

$$[b_{k+2,k}, b_{k+2,k+1}, b_{k+2,k+2}]Z_{k1} = \gamma e_3^T$$

$A(:, [k : k + 2]) = A(:, [k : k + 2])Z_{k1}$

$B(:, [k : k + 2]) = B(:, [k : k + 2])Z_{k1}$

Find 2×2 Householder matrix Z_{k2} such that

$$[b_{k+1,k}, b_{k+1,k+1}]Z_{k2} = \gamma e_2^T$$

$A(:, [k\ k + 1]) = A(:, [k\ k + 1])Z_{k2}$

$B(:, [k\ k + 1]) = B(:, [k\ k + 1])Z_{k2}$

if $k < n - 2$

Find 3×3 Householder matrix Q_k such that

$$Q_k[a_{k+1,k}a_{k+2,k}a_{k+3,k}]^T = \gamma e_1$$

$A([k+1:k+3],:) = Q_k A([k+1:k+3],:)$

$B([k+1:k+3],:) = Q_k B([k+1:k+3],:)$

elseif $k = n-2$

Find 2×2 Householder matrix Q_k such that

$Q_k[a_{k+1,k} a_{k+2,k}]^T = \gamma e_1$

$A([k+1\ k+2],:) = Q_k A([k+1\ k+2],:)$

$B([k+1\ k+2],:) = Q_k B([k+1\ k+2],:)$

end if

end for

We can now use sequences of these steps to produce an algorithm to reduce A to quasi-triangular form and maintain the upper triangular from of B. We have to monitor A's subdiagonal and B diagonal in case we can reduce the order of the problem by decoupling.

Algorithm 3.1.3 (The QZ Algorithm) *Given $A, B \in \mathbb{R}^{n\times n}$, the following algorithm computes orthogonally updated matrices $Q^TAZ = T$ and $Q^TBZ = S$ such that T is upper quasi-triangular and S is upper triangular.*

Compute $A \leftarrow Q^TAZ$ and $B \leftarrow Q^TBZ$ where Q^TAZ is upper Hessenberg

and Q^TBZ is upper triangular using Algorithm 3.1.1

until A is quasi-triangular

Set $a_{i,i-1}$ to zero if $|a_{i,i-1}| \leq tol(|a_{i-1,i-1}| + |a_{ii}|)$.

Partition A such that

$$A = \begin{array}{c} \\ p \\ n-p-q \\ q \end{array} \begin{array}{c} \begin{array}{ccc} p & n-p-q & q \end{array} \\ \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ 0 & A_{22} & A_{23} \\ 0 & 0 & A_{33} \end{bmatrix} \end{array}$$

where A_{22} is unreduced upper Hessenberg.

Partition B similarly

$$B = \begin{array}{c} \\ p \\ n-p-q \\ q \end{array} \begin{array}{c} \begin{array}{ccc} p & n-p-q & q \end{array} \\ \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ 0 & B_{22} & B_{23} \\ 0 & 0 & B_{33} \end{bmatrix} \end{array}.$$

if B_{22} is singular

Apply zero-chasing technique to zero $a_{n-q,n-q-1}$.

else

Apply Algorithm 3.1.2 to A_{22} and B_{22}.

For completeness we have

$A_{12} \leftarrow A_{12}Z, B_{12} \leftarrow B_{12}Z$

$A_{23} \leftarrow Q^T A_{23}, B_{23} \leftarrow Q^T B_{23}$

where Q and Z are the product of the transformations used in Algorithm 3.1.2.

end if

end until

$T = A$; $S = B$

The problem has now been reduced into smaller 1×1 and 2×2 subproblems. The eigenvalues of the 1×1 problems are just the ratio of the diagonal entries of A and B. The eigenvalues of the 2×2 problems can be calculated as the roots of the quadratic equation $\det(A' - \lambda B') = 0$ where

$$A' = \begin{bmatrix} a_{kk} & a_{k,k+1} \\ a_{k+1,k} & a_{k+1,k+1} \end{bmatrix}, \quad B' = \begin{bmatrix} b_{kk} & b_{k,k+1} \\ 0 & b_{k+1,k+1} \end{bmatrix}.$$

Two reasons are given for not using this quadratic but instead reducing A further to upper triangular form. When A and B are real the calculation of eigenvectors is greatly helped if the eigenvalues are expressed as the ratios of the diagonal entries of A and B. Also, if for some k, a_{kk} and b_{kk} are both small then the eigenvalue

$\lambda_k = a_{kk}/b_{kk}$ is ill-conditioned. Therefore more information is contained in the ratio of the diagonal entries.

If B' is singular we can apply the zero-chasing technique to reduce $a_{k+1,k}$ to zero. For nonsingular B' we form the matrix $E = A' - \lambda B'$ where λ is an eigenvalue of (A', B'). Notice E is now singular, if we postmultiply by a matrix Z to reduce either e_{11} or e_{21} to zero the other element should also be reduced to zero. We now premultiply by a matrix Q such that either QAZ or QBZ is upper triangular. Since the first column of QEZ is zero then both QAZ and QBZ must be upper triangular.

The QZ algorithm is a backward stable algorithm, that is, it computes upper quasi-triangular $\widehat{S}$ and upper triangular $\widehat{T}$ that satisfy

$$Q^T(A+E)Z = \widehat{S}, \quad Q^T(B+F)Z = \widehat{T}$$

where Q and Z are orthogonal and

$$\|E\|_2 \leq c_n u \|A\|, \quad \|F\|_2 \leq c_n u \|B\|.$$

The drawback to using this algorithm is that it does not take into account the special structure of the symmetric definite generalized eigenvalue problem. The equivalence transformations destroy the symmetry of the problem and therefore the algorithm does not guarantee to calculate real eigenvalues. Because the algorithm does not exploit the structure it can be expensive. If we consider that on average two QZ iterations per eigenvalue are needed then the algorithm takes $30n^3$ flops. If Q and Z are required then an additional $16n^3$ flops and $20n^3$ flops are required respectively.

3.1.4 Computing the Eigenvectors

Once an approximate eigenvalue λ of the matrix pair (A, B) has been found, we can calculate the corresponding eigenvector using inverse iteration. Starting with

an initial vector $x^{(0)}$ such that $\|x^{(0)}\|_2 = 1$:

for $i = 1, 2, \ldots$

Solve for $x^{(i)}$, $(A - \lambda B)x^{(i)} = Bx^{(i-1)}$

Normalize $x^{(i)}$, $x^{(i)} = x^{(i)}/\|x^{(i)}\|_2$

$\lambda^{(i)} = x^{(i)^*} A x^{(i)} / x^{(i)^*} B x^{(i)}$

until convergence

This is just the power method applied to $(A-\lambda B)^{-1}$. To analyze the behaviour of the iteration we assume that (A, B) has a basis of eigenvectors $\{x_1, \ldots, x_n\}$ with corresponding eigenvalues $\lambda_1, \ldots, \lambda_n$. The initial vector can be written as

$$x^{(0)} = \sum_{i=1}^{n} \beta_i x_i;$$

then $x^{(k)}$ is a vector given by

$$(A - \lambda B)^{-k} x^{(0)} = \sum_{i=1}^{n} \frac{\beta_i}{(\lambda_i - \lambda)^k} x_i.$$

If λ is much closer to λ_j than to the other eigenvalues then $x^{(k)}$ will be dominated by the direction of x_j provided $\beta_j \neq 0$.

Inverse iteration can be used in conjunction with the Hessenberg-triangular form, $(\widetilde{A}, \widetilde{B}) = (Q_0 A Z_0, Q_0 B Z_0)$, that is formed during the QZ algorithm. If inverse iteration is applied to $(\widetilde{A} - \lambda \widetilde{B})^{-1}$ then the computed eigenvector y_i of $(\widetilde{A}, \widetilde{B})$ is related to the eigenvector of (A, B) by $x_i = Z_0 y_i$. As $(\widetilde{A} - \lambda \widetilde{B})$ is upper Hessenberg we can solve the linear system in $O(n^2)$ flops.

In theory the matrix $A - \lambda B$ is singular. However, because λ is an approximate eigenvalue, the matrix will not be singular but just ill-conditioned. This may suggest that there is large error in the solution of $(A - \lambda B)x^{(i)} = Bx^{(i-1)}$. However, it can be shown that the error lies almost entirely in the direction of the required eigenvector [26]. This is why typically only one iteration is required when λ is an eigenvalue calculated with the QZ algorithm.

3.2 Cholesky Method

The Cholesky method, apparently first suggested by Wilkinson [63, pp. 337-340], is a method that reduces the symmetric definite generalized eigenvalue problem to an ordinary eigenvalue problem while maintaining the symmetry of the problem. It begins by computing the Cholesky factorization of the positive definite matrix B, optionally with complete pivoting

$$\Pi^T B \Pi = L D^2 L^T. \tag{3.4}$$

where Π is a permutation matrix, L is a unit lower triangular matrix and D^2 is a positive diagonal matrix. The problem is then reduced to the form

$$Cy \equiv D^{-1} L^{-1} \Pi^T A \Pi L^{-T} D^{-1} y = \lambda y \tag{3.5}$$

where $y = DL^T \Pi^T x$. Any method for solving the symmetric eigenvalue problem can be applied to C such as the symmetric QR or Jacobi method (See Chapter 2). In LAPACK's `xSYGV` driver [1], the symmetric definite generalized eigenvalue problem is solved by applying the symmetric QR algorithm to (3.5). MATLAB 6's `eig` function [42] does likewise when it is given a symmetric definite generalized eigenvalue problem.

3.2.1 Cholesky Factorization

One of the advantages of a symmetric positive definite matrix is that its LU factorization can be converted into a Cholesky factorization: $A = R^T R$, where R is upper triangular with positive diagonal elements. We now give a proof of this fact.

Theorem 3.2.1 (Cholesky Factorization) *If $A \in \mathbb{R}^{n \times n}$ is symmetric positive definite then there exists a unique upper triangular matrix R with positive diagonal elements such that $A = R^T R$.*

Proof. We prove this by induction on n. For the base case $n = 1$ this is obvious. If we assume that it is true for $n-1$ then the leading principal submatrix of $A \in \mathbb{R}^{n\times n}$, $A_{n-1} = A(1\colon n-1, 1\colon n-1)$ is positive definite and has a unique Cholesky factorization $A_{n-1} = R_{n-1}^T R_{n-1}$. Overall we have

$$A = \begin{bmatrix} A_{n-1} & c \\ c^T & \alpha \end{bmatrix} = \begin{bmatrix} R_{n-1}^T & 0 \\ r^T & \beta \end{bmatrix} \begin{bmatrix} R_{n-1} & r \\ 0 & \beta \end{bmatrix} =: R^T R$$

if

$$R_{n-1}^T r = c \tag{3.6}$$

$$r^T r + \beta^2 = \alpha \tag{3.7}$$

As R_{n-1} is nonsingular, equation (3.6) has a unique solution for r. If we consider

$$0 < \det(A) = \det(R^T)\det(R) = \det(R_{n-1})^2\beta^2$$

then $\beta^2 > 0$ and there exists a unique $\beta > 0$ which is a solution to (3.7). ∎

The proof of Theorem 3.2.1 is constructive and shows how we can compute the Cholesky factorization one column at a time. Alternatively, if we consider the equations for working out the (i,j) elements in $A = R^T R$ we get

$$a_{ij} = \sum_{k=1}^{i} r_{ki} r_{kj}, \quad j \geq i.$$

By solving these equations in the order

$$(1,1),(1,2),(2,2),(1,3),\ldots,(3,3),\ldots(n,n)$$

we obtain the following algorithm:

Algorithm 3.2.2 (Cholesky Factorization) *Given a symmetric positive definite matrix $A \in \mathbb{R}^{n\times n}$ this algorithm computes the Cholesky factorization $A = R^T R$.*

for $j = 1{:}\,n$

for $i = 1{:}\,j-1$

$r_{ij} = (a_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj})/r_{ii}$

end

$r_{jj} = (a_{jj} - \sum_{k=1}^{j-1} r_{kj}^2)^{1/2}$

end

The Cholesky factorization takes $n^3/3$ flops which is half the cost of LU factorization.

3.2.2 Complete Pivoting

We can also form a Cholesky factorization with complete pivoting of B as in (3.4). This permutation reorders the data while maintaining the symmetry of B. Note that this operation cannot move off-diagonal elements onto the diagonal of B and is therefore a reordering of the diagonal of B. At the jth step of Algorithm 3.2.2 we permute the largest diagonal entry of $B(j{:}\,n, j{:}\,n)$ into the position b_{jj}. In this way we can compute the factorization (3.4):

Algorithm 3.2.3 (Cholesky Factorization with Complete Pivoting) *Given a symmetric positive definite matrix $A \in \mathbb{R}^{n\times n}$ this algorithm computes the Cholesky factorization with complete pivoting, $\Pi^T A \Pi = R^T R$.*

for $j = 1{:}\,n$

Find $\max |a_{pp}|$ in $A(j{:}\,n, j{:}\,n)$.

$A \leftarrow \Pi^T A \Pi$ where Π is a permutation matrix that swaps the

p and j columns and rows

for $i = 1{:}\,j-1$

$r_{ij} = (a_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj})/r_{ii}$

end

$r_{jj} = (a_{jj} - \sum_{k=1}^{j-1} r_{kj}^2)^{1/2}$

end

If we define $D^2 = \mathrm{diag}(r_{ii}^2)$ then the Cholesky factorization becomes $\Pi^T B \Pi = R^T R = L D^2 L^T$ where L is unit lower triangular. If we are using complete pivoting in the Cholesky factorization then $|l_{ij}| \leq 1$ for $i > j$ and

$$d_1^2 \geq \cdots \geq d_n^2 > 0. \tag{3.8}$$

Hence [31, Thm. 8.13]

$$\kappa_p(L) = \|L\|_p \|L^{-1}\|_p \leq n 2^{n-1}, \qquad p = 1, 2, \infty \tag{3.9}$$

with approximate equality achieved for L^T the Kahan matrix [31, p. 161]. Thus the pivoting tends to keep L well-conditioned and therefore concentrate any ill conditioning of B in D^2.

3.3 Error Analysis of Cholesky–Jacobi Method

The complete Cholesky–Jacobi algorithm, where we use Jacobi's method to solve the symmetric eigenvalue problem in (3.5), is summarized as follows:

Algorithm 3.3.1 (Cholesky–Jacobi Method) *Given $A, B \in \mathbb{R}^{n \times n}$ with A symmetric and B symmetric positive definite, this algorithm calculates the eigenvalues λ_i and corresponding eigenvectors x_i of the pair (A, B).*

1. Compute the Cholesky factorization with complete pivoting $\Pi^T B \Pi = L D^2 L^T$.
   Form $H = D^{-1} L^{-1} \Pi^T A \Pi L^{-T} D^{-1}$ by solving triangular systems.
   $X = \Pi L^{-T} D^{-1}$
2. % Jacobi's method
   done_rot = true

while done_rot = true

done_rot = false

for $i = 1{:}\,n$

for $j = i + 1{:}\,n$

(∗) if $|h_{ij}| > u\sqrt{|h_{ii}h_{jj}|}$

done_rot = true

Form $Q_{ij} \equiv Q_k([i\ j],[i\ j])$ using (2.4), (2.5) and (2.6).

$\text{ind} = [1{:}\,i-1, i+1{:}\,j-1, j+1{:}\,n]$

$H([i\ j], \text{ind}) = Q_{ij}^T H([i\ j], \text{ind})$

$H(\text{ind}, [i\ j]) = H(\text{ind}, [i\ j])Q_{ij}$

$H([i\ j],[i\ j]) = H(\text{ind}, [i\ j])Q_{ij}$

$H([i\ j],[i\ j]) = \begin{bmatrix} \widetilde{h}_{ii} & 0 \\ 0 & \widetilde{h}_{jj} \end{bmatrix}$ using (2.7)

$X(:, [i\ j]) = X(:, [i\ j])Q_{ij}$

end

end

end

end

$\lambda_i = h_{ii}$, $x_i = X(:, i)$, $i = 1{:}\,n$

The test (∗) for whether to apply a rotation is adapted from the one used for Jacobi's method for a symmetric positive definite matrix [10]—we have added absolute values inside the square root since h_{ii} and h_{jj} can be negative. This test is too stringent in general and can cause the algorithm not to converge, but we have found it generally works well and so we used it in our experiments in order to achieve the best possible numerical behaviour.

First, we make a simple but important observation about numerical stability. Assume that the Cholesky factorization is computed exactly and set $\Pi = I$, without loss of generality. We compute $\widehat{C} = C + \Delta C_1$ where, at best, ΔC_1

satisfies a bound of the form

$$|\Delta C_1| \leq c_n u |D^{-1}||L^{-1}||A||L^{-T}||D^{-1}|,$$

where c_n is a constant and u is the unit roundoff. Here, $|A| = (|a_{ij}|)$. Solution of the eigenproblem for $\widehat{C}$ can be assumed to yield the exact eigensystem of $\widehat{C}+\Delta C_2$, for some ΔC_2. Therefore the computed eigensystem is the exact eigensystem of

$$C + \Delta C_1 + \Delta C_2 = D^{-1}L^{-1}\big(A + \Delta A\big)L^{-T}D^{-1}, \qquad \Delta A = LD(\Delta C_1 + \Delta C_2)DL^T,$$

and

$$\begin{aligned} |\Delta A| &\leq |L||D|\big(c_n u |D^{-1}||L^{-1}||A||L^{-T}||D^{-1}| + |\Delta C_2|\big)|D||L^T| \\ &\leq c_n u |L||L^{-1}||A||L^{-T}||L^T| + |L||D||\Delta C_2||D||L^T|. \end{aligned} \tag{3.10}$$

The first term in (3.10) is bounded independently of $\kappa(B)$ due to (3.9). The second term will have the same property provided that ΔC_2 satisfies a bound of the form

$$|\Delta C_2| \leq |D^{-1}| f(|A|, |L^{-1}|, u)|D^{-1}|,$$

where f is a matrix function depending on $|A|$, $|L^{-1}|$ and u, but not $|D^{-1}|$.

If nothing more is known about ΔC_2 than that $\|\Delta C_2\| \leq c_n u \|C\|$ (corresponding to using a normwise backward stable eigensolver for C) then the best bound we can obtain in terms of the original data is of the form

$$\|\Delta A\| \leq g(n) u \kappa(B) \|A\|. \tag{3.11}$$

However, this analysis shows that there is hope for obtaining a bound without the factor $\kappa(B)$ if the eigensolver for C respects the scaling of C when D is ill conditioned.

Now we give an error analysis for the Cholesky–Jacobi method with the aim of obtaining an error bound better than (3.11). We consider first the second part of Algorithm 3.3.1, beginning with the construction of the Jacobi rotation.

Lemma 3.3.2 *Let a Jacobi rotation Q_k be constructed using (2.4), (2.5) and (2.6) so that $Q_k^T H_k Q_k$ has zeros in the (i,j) and (j,i) positions. The computed $\widehat{c}$, $\widehat{s}$ and $\widehat{t}$ satisfy*

$$\widehat{c} = c(1+\tilde{\theta}_1), \quad \widehat{s} = s(1+\tilde{\theta}_1'), \quad \widehat{t} = t(1+\tilde{\theta}_1''),$$

where c, s and t are the exact values for H_k.

Proof. Straightforward. □

In most of the rest of our analysis we will assume that the computed $\widehat{c}$, $\widehat{s}$ and $\widehat{t}$ are exact. It is easily checked that, in view of Lemma 3.3.2, this simplification does not affect the bounds.

Lemma 3.3.3 *If one step of Jacobi's method is performed in the (i,j) plane on the matrix H_m then the computed $\widehat{H}_{m+1}$ satisfies*

$$\widehat{H}_{m+1} = Q_m^T \left(H_m + \Delta H_m\right) Q_m,$$

where the elements of ΔH_m are bounded componentwise by

$$\left.\begin{aligned} |\Delta h_{ik}| &\le \tilde{\gamma}_1 \left(|h_{ik}| + 2|sc||h_{jk}|\right) \\ |\Delta h_{jk}| &\le \tilde{\gamma}_1 \left(|h_{jk}| + 2|sc||h_{ik}|\right) \end{aligned}\right\} \quad k \neq i,j$$

and

$$\begin{aligned} |\Delta h_{ii}| &\le \tilde{\gamma}_1 \left(c^2|h_{ii}| + |s/c||h_{ij}| + s^2|h_{jj}|\right), \\ |\Delta h_{ij}|, |\Delta h_{ji}| &\le \tilde{\gamma}_1 \left(|sc||h_{ii}| + 2s^2|h_{ij}| + |sc||h_{jj}|\right), \\ |\Delta h_{jj}| &\le \tilde{\gamma}_1 \left(s^2|h_{ii}| + |s/c||h_{ij}| + c^2|h_{jj}|\right). \end{aligned}$$

Proof. For the duration of the proof let $Q_m := Q_m([i\ j],[i\ j])$. Writing $H_m = (h_{ij})$ and $\widehat{H}_{m+1} = (\widehat{h}_{ij})$ and using a standard result for matrix–vector multiplication [31, Sec. 3.5], we have, for $k \neq i,j$,

$$\begin{bmatrix} \widehat{h}_{ik} \\ \widehat{h}_{jk} \end{bmatrix} = fl\left(Q_m^T \begin{bmatrix} h_{ik} \\ h_{jk} \end{bmatrix}\right),$$

$$= (Q_m + \Delta Q_m)^T \begin{bmatrix} h_{ik} \\ h_{jk} \end{bmatrix}, \quad |\Delta Q_m| \leq \tilde{\gamma}_1 |Q_m|,$$
$$=: Q_m^T \left(\begin{bmatrix} h_{ik} \\ h_{jk} \end{bmatrix} + \begin{bmatrix} \Delta h_{ik} \\ \Delta h_{jk} \end{bmatrix} \right).$$

Then

$$\begin{bmatrix} |\Delta h_{ik}| \\ |\Delta h_{jk}| \end{bmatrix} \leq |Q_m||\Delta Q_m^T| \begin{bmatrix} |h_{ik}| \\ |h_{jk}| \end{bmatrix}$$
$$\leq \tilde{\gamma}_1 |Q_m||Q_m^T| \begin{bmatrix} |h_{ik}| \\ |h_{jk}| \end{bmatrix}$$
$$= \tilde{\gamma}_1 \begin{bmatrix} 1 & 2|sc| \\ 2|sc| & 1 \end{bmatrix} \begin{bmatrix} |h_{ik}| \\ |h_{jk}| \end{bmatrix},$$

which gives the first two bounds. We calculate the elements at the intersection of rows and columns i and j using

$$\widehat{h}_{ii} = fl(h_{ii} - h_{ij}t) = (1 + \tilde{\theta}_1)h_{ii} - (1 + \tilde{\theta}_1)h_{ij}t,$$
$$\widehat{h}_{jj} = fl(h_{jj} + h_{ij}t) = (1 + \tilde{\theta}_1)h_{jj} + (1 + \tilde{\theta}_1)h_{ij}t,$$

and by setting $\widehat{h}_{ij}$ and $\widehat{h}_{ji}$ to zero. The backward perturbations Δh_{ii}, Δh_{ij} and Δh_{jj} satisfy

$$Q_m^T \left(\begin{bmatrix} h_{ii} & h_{ij} \\ h_{ij} & h_{jj} \end{bmatrix} + \begin{bmatrix} \Delta h_{ii} & \Delta h_{ij} \\ \Delta h_{ij} & \Delta h_{jj} \end{bmatrix} \right) Q_m = \begin{bmatrix} \widehat{h}_{ii} & 0 \\ 0 & \widehat{h}_{jj} \end{bmatrix},$$

which can be expressed as

$$\begin{bmatrix} \Delta h_{ii} & \Delta h_{ij} \\ \Delta h_{ij} & \Delta h_{jj} \end{bmatrix} = Q_m \begin{bmatrix} \widehat{h}_{ii} & 0 \\ 0 & \widehat{h}_{jj} \end{bmatrix} Q_m^T - \begin{bmatrix} h_{ii} & h_{ij} \\ h_{ij} & h_{jj} \end{bmatrix}$$
$$= \begin{bmatrix} c^2\widehat{h}_{ii} + s^2\widehat{h}_{jj} & -sc\widehat{h}_{ii} + sc\widehat{h}_{jj} \\ -sc\widehat{h}_{ii} + sc\widehat{h}_{jj} & s^2\widehat{h}_{ii} + c^2\widehat{h}_{jj} \end{bmatrix} - \begin{bmatrix} h_{ii} & h_{ij} \\ h_{ij} & h_{jj} \end{bmatrix}.$$

Substituting in for $\widehat{h}_{ii}$ and $\widehat{h}_{jj}$ and using suitable Jacobi rotation identities, we then take absolute values to obtain the second group of inequalities. (Note that $\Delta h_{ij} = \Delta h_{ji} = 0$ if c and s are exact, so by bounding Δh_{ij} and Δh_{ji} in this way we are allowing for inexact c and s.) □

In the next lemma we show that in the first rotation of Jacobi's method in Algorithm 3.3.1 a factor D^{-1} can be scaled out of the backward error, leaving a term that we can bound. We make use of the identity

$$sc = \frac{h_{ij}}{\sqrt{4h_{ij}^2 + (h_{ii} - h_{jj})^2}}, \tag{3.12}$$

which comes from manipulating the equations defining a Jacobi rotation and solving for $sc = \frac{1}{2}\sin 2\theta$ in terms of $\tan 2\theta$. In this result, $A_0 \equiv L^{-1}\Pi^T A \Pi L^{-T}$ in (3.5).

Lemma 3.3.4 *Given a symmetric A_0 and a positive diagonal matrix $D_0 = \mathrm{diag}(d_i^2)$, suppose we perform one step of Jacobi's method in the (i,j) plane on $H_0 = D_0^{-1}A_0D_0^{-1}$, obtaining $H_1 = Q_0^T H_0 Q_0$. Then*

$$\widehat{H}_1 = fl(Q_0^T \widehat{H}_0 Q_0) = Q_0^T D_0^{-1}(A_0 + \Delta A_0)D_0^{-1}Q_0, \tag{3.13}$$

where

$$\|\Delta A_0\|_2 \le \tilde{\gamma}_n(1 + 2\omega_0)\|A_0\|_2, \tag{3.14}$$

with

$$\omega_0 = |sc| \max(\rho, 1/\rho), \qquad \rho = d_i/d_j.$$

Proof. We start by forming the matrix $H_0 = (h_{ij})$. Since we are given the squared diagonal elements d_i^2 we have

$$\begin{aligned}
\widehat{h}_{ij} &= fl\left(a_{ij}/\sqrt{d_i^2 d_j^2}\right) \\
&= (1+\theta_3)a_{ij}/(d_i d_j) = (1+\theta_3)h_{ij} \\
&=: \widehat{a}_{ij}/(d_i d_j).
\end{aligned}$$

Thus these initial errors can be thrown onto A_0: $\widehat{H}_0 = D_0^{-1}(A_0 + \Delta_1)D_0^{-1}$ where $|\Delta_1| \le \gamma_3|A_0|$. The errors in applying one step of Jacobi's method to $\widehat{H}_0$ can be expressed as a backward perturbation ΔH_0 to $\widehat{H}_0$ using Lemma 3.3.3. The

corresponding perturbation of $\widehat{A}_0 = A_0 + \Delta_1$ is $\Delta A_0 = D_0 \Delta H_0 D_0$, so we simply scale the componentwise perturbation bounds of Lemma 3.3.3. We find

$$\left.\begin{aligned} |\Delta a_{ik}| &\le \tilde{\gamma}_1 \left(|\widehat{a}_{ik}| + 2|sc||\widehat{a}_{jk}|\rho \right) \\ |\Delta a_{jk}| &\le \tilde{\gamma}_1 \left(|\widehat{a}_{jk}| + 2|sc||\widehat{a}_{ik}|/\rho \right) \end{aligned}\right\} \quad k \neq i, j,$$

$$|\Delta a_{ii}| \le \tilde{\gamma}_1 \left(c^2 |\widehat{a}_{ii}| + |s/c||\widehat{a}_{ij}|\rho + s^2 |\widehat{a}_{jj}|\rho^2 \right), \tag{3.15}$$

$$|\Delta a_{ij,ji}| \le \tilde{\gamma}_1 \left(|sc||\widehat{a}_{ii}|/\rho + 2s^2 |\widehat{a}_{ij}| + |sc||\widehat{a}_{jj}|\rho \right),$$

$$|\Delta a_{jj}| \le \tilde{\gamma}_1 \left(s^2 |\widehat{a}_{ii}|/\rho^2 + |s/c||\widehat{a}_{ij}|/\rho + c^2 |\widehat{a}_{jj}| \right). \tag{3.16}$$

We now work to remove the potentially large ρ^2 and $1/\rho^2$ terms. We can rewrite (3.12) as

$$sc = \frac{\frac{a_{ij}}{d_i d_j}}{\sqrt{4\frac{a_{ij}^2}{d_i^2 d_j^2} + \left(\frac{a_{ij}}{d_i^2} - \frac{a_{jj}}{d_j^2} \right)^2}} = \frac{\rho a_{ij}}{\sqrt{\left(a_{ii} - \rho^2 a_{jj} \right)^2 + 4\rho^2 a_{ij}^2}}. \tag{3.17}$$

Further manipulation yields

$$|a_{jj}|\rho^2 \le |a_{ii}| + \sqrt{\frac{a_{ij}^2 \rho^2}{(sc)^2} - 4a_{ij}^2 \rho^2} = |a_{ii}| + \rho |a_{ij}| \sqrt{\frac{1}{(sc)^2} - 4}.$$

Therefore

$$s^2 |a_{jj}|\rho^2 \le s^2 |a_{ii}| + \rho |a_{ij}| \sqrt{t^2 - 4s^4}. \tag{3.18}$$

A similar manipulation of (3.12) (or a symmetry argument) gives

$$s^2 |a_{ii}|/\rho^2 \le s^2 |a_{jj}| + \frac{|a_{ij}|}{\rho} \sqrt{t^2 - 4s^4}. \tag{3.19}$$

Since $\widehat{a}_{ij} = a_{ij}(1+\theta_3)$ there is no harm in replacing a_{ij} by $\widehat{a}_{ij}$ in (3.18) and (3.19). Since $\theta \in [-\pi/4, \pi/4]$ we have

$$\sqrt{t^2 - 4s^4} + |s/c| = 2|sc|, \tag{3.20}$$

and hence (3.15) and (3.16) may be bounded by

$$|\Delta a_{ii}| \le \tilde{\gamma}_1 \left(|\widehat{a}_{ii}| + 2|sc||\widehat{a}_{ij}|\rho \right),$$

$$|\Delta a_{jj}| \le \tilde{\gamma}_1 \left(|\widehat{a}_{jj}| + 2|sc||\widehat{a}_{ij}|/\rho \right).$$

Using these componentwise bounds we obtain the overall bound given in (3.14). □

Lemma 3.3.4 shows that the Jacobi rotation results in a small backward perturbation to A_0 provided that ω_0 is of order 1. We see from (3.17) that in normal circumstances sc is proportional to $\min(\rho, 1/\rho)$, which keeps ω_0 small. However, in special situations ω_0 can be large, for example when $|a_{ii} - \rho^2 a_{jj}| \ll \rho|a_{ij}|$ with ρ large, which requires that $|a_{jj}|$ be much smaller than $|a_{ij}|$ and B be ill conditioned.

By combining Lemma 3.3.4 with subsequent applications of Lemma 3.3.3 we find that after m steps of Jacobi's method on $H_0 = D_0^{-1} A_0 D_0^{-1}$ we have

$$\widehat{H}_m = Q_{m-1}^T \dots Q_0^T (H_0 + \Delta_0) Q_0 \dots Q_{m-1},$$

where

$$\begin{aligned}\Delta_0 &= \Delta H_0 + \sum_{k=1}^{m-1} Q_0 \dots Q_{k-1} \Delta H_k Q_{k-1}^T \dots Q_0^T \\ &= D_0^{-1} \Big(\Delta A_0 + \sum_{k=1}^{m-1} D_0 Q_0 \dots Q_{k-1} \Delta H_k Q_{k-1}^T \dots Q_0^T D_0 \Big) D_0^{-1}. \end{aligned} \tag{3.21}$$

The ΔH_k are bounded as in Lemma 3.3.3. We would like to bound the term in parentheses by a multiple of $u\|A_0\|_2$, but simply taking norms leads to an unsatisfactory $\kappa(D_0^2)$ factor. To obtain a better bound we introduce, purely for theoretical purposes, a scaling to $\widehat{H}_k$ at each stage of the iteration. For an arbitrary nonsingular diagonal D_k we write

$$\begin{aligned} &\|D_0 Q_0 \dots Q_{k-1} \Delta H_k Q_{k-1}^T \dots Q_0^T D_0\|_2 \\ &= \|D_0 Q_0 \dots Q_{k-1} D_k^{-1} \cdot D_k \Delta H_k D_k \cdot D_k^{-1} Q_{k-1}^T \dots Q_0^T D_0\|_2 \\ &\le \min_{D_k \text{ diag}} \left(\|D_0 Q_0 \dots Q_{k-1} D_k^{-1}\|_2^2 \|D_k \Delta H_k D_k\|_2 \right) \\ &= \min_{D_k \text{ diag}} \left(\|N_k^{-T}\|_2^2 \|D_k \Delta H_k D_k\|_2 \right), \end{aligned} \tag{3.22}$$

where

$$N_k = D_0^{-1} Q_0 \dots Q_{k-1} D_k. \tag{3.23}$$

Define

$$A_k := N_k^T A_0 N_k = D_k H_k D_k. \tag{3.24}$$

By applying Lemma 3.3.4 to a rotation on H_k, we can see that

$$\|D_k \Delta H_k D_k\|_2 \leq \tilde{\gamma}_n(1 + 2\omega_k)\|A_k\|_2, \tag{3.25}$$

where

$$\omega_k = |s_k c_k| \max(\rho_k, 1/\rho_k), \qquad \rho_k = d_i^{(k)}/d_j^{(k)},$$

with a subscript k denoting quantities on the kth step and where $D_k = \mathrm{diag}(d_i^{(k)})$. One way to proceed is to choose D_k to minimize $\kappa_2(M_{k-1})$, where

$$M_{k-1} = D_{k-1}^{-1} Q_{k-1} D_k. \tag{3.26}$$

Notice that

$$N_k = M_0 \ldots M_{k-1}. \tag{3.27}$$

This idea is based on an algorithm of Anjos, Hammarling and Paige [2] that avoids explicitly inverting any of the D_k and uses transformation matrices of the form in (3.26) to diagonalize A while retaining the diagonal form of D_0. The algorithm computes the congruence transformations

$$A_{k+1} = M_k^T A_k M_k, \quad D_{k+1}^2 = M_k^T D_k^2 M_k,$$

where D_k is diagonal for all k and A_k tends to diagonal form as $k \to \infty$. The difference between our approach and that in [2] is that we form $H_0 = D_0^{-1} A_0 D_0^{-1}$ and use D_k in the analysis to obtain stronger error bounds, whereas in [2], in an effort to apply only well conditioned similarity transformations, H_0 is never formed but M_k is computed and applied in the algorithm (and no error analysis is given in [2]).

Now we discuss the choice of D_k, drawing on analysis from [2]. Since Q_{k-1} is a rotation in the (i, j) plane, we choose D_k to be identical to D_{k-1} in all but

the ith and jth diagonal entries. Thus M_{k-1} is the identity matrix except in the (i,j) plane, in which

$$M_{ij} = M([i\ j],[i\ j]) = \begin{bmatrix} d_i^{-1} & 0 \\ 0 & d_j^{-1} \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} \widetilde{d}_i & 0 \\ 0 & \widetilde{d}_j \end{bmatrix},$$

where we are writing

$$D_{k-1} = \mathrm{diag}(d_i), \quad D_k = \mathrm{diag}(\widetilde{d}_i).$$

We now choose D_k to minimize the 2-norm condition number $\kappa_2(M_{ij})$. It can be shown that for any 2×2 matrix, G say,

$$\kappa_2(G) = \sigma_1(G)/\sigma_2(G) = \left(\phi^2 + \sqrt{\phi^4 - 4\delta^2}\right)/2\delta,$$

where $\phi = \|G\|_F$, $\delta = |\det(G)|$ and $\sigma_1(G) \geq \sigma_2(G)$ are the singular values of G. Using $\kappa_F(G) = \phi^2/\delta$, we obtain

$$\kappa_2(G) = \left(\kappa_F(G) + \sqrt{\kappa_F(G)^2 - 4}\right)/2,$$

so clearly $\kappa_2(G)$ has its minimum when $\kappa_F(G)$ does. Therefore it is only necessary to analyze $\kappa_F(M_{ij})$ in order to find the minimum of $\kappa_2(M_{ij})$. For M_{ij} we have

$$\begin{aligned} \phi^2 &= s^2\left((\widetilde{d}_i/d_j)^2 + (\widetilde{d}_j/d_i)^2\right) + c^2\left((\widetilde{d}_j/d_j)^2 + (\widetilde{d}_i/d_i)^2\right), \\ \delta &= \det(D_{k-1}^{-1})\det(D_k) = (\widetilde{d}_i\widetilde{d}_j)/(d_i d_j). \end{aligned}$$

Setting $\xi = \widetilde{d}_i/\widetilde{d}_j$ we have

$$\kappa_F(M_{ij}) = \phi^2/\delta = \left(c^2(\rho^2 + \xi^2) + s^2(\rho^2\xi^2 + 1)\right)/(\rho\xi).$$

This is an equation with only one unknown, ξ. The minimum of $\kappa_F(M_{ij})$ over ξ occurs at

$$\xi_{\mathrm{opt}}^2 = \left(s^2 + \rho^2 c^2\right) / \left(c^2 + \rho^2 s^2\right),$$

which gives the values

$$\begin{aligned} \kappa_F(M_{ij})_{\min} &= 2\sqrt{1 + s^2c^2\left(\rho - \rho^{-1}\right)^2}, \\ \kappa_2(M_{ij})_{\min} &= |sc(\rho - \rho^{-1})| + \sqrt{1 + s^2c^2\left(\rho - \rho^{-1}\right)^2}. \end{aligned} \tag{3.28}$$

Knowing the ratio $\widetilde{d}_i/\widetilde{d}_j$ that minimizes $\kappa_2(M_0)$, we now have to choose $\widetilde{d}_j$ and then set $\widetilde{d}_i = \widetilde{d}_j \xi_{\text{opt}}$. We set $\|D_k\|_F = \|D_{k-1}\|_F$, or more simply

$$d_i^2 + d_j^2 = \widetilde{d}_i^2 + \widetilde{d}_j^2 = \left(\xi_{\text{opt}}^2 + 1\right) \widetilde{d}_j^2. \tag{3.29}$$

This yields the values

$$\begin{aligned} \widetilde{d}_i^2 &= c^2 d_i^2 + s^2 d_j^2, \\ \widetilde{d}_j^2 &= c^2 d_j^2 + s^2 d_i^2, \end{aligned} \tag{3.30}$$

and the matrix

$$M_{ij} = \begin{bmatrix} c\sqrt{c^2 + s^2/\rho^2} & s\sqrt{s^2 + c^2/\rho^2} \\ -s\sqrt{s^2 + c^2\rho^2} & c\sqrt{c^2 + s^2\rho^2} \end{bmatrix}. \tag{3.31}$$

Clearly,

$$\min(d_i^2, d_j^2) \leq \widetilde{d}_k^2 \leq \max(d_i^2, d_j^2), \qquad k = i, j. \tag{3.32}$$

We note for later reference that a direct calculation reveals

$$\|M_{ij}^{-1}\|_F = \sqrt{2}. \tag{3.33}$$

It is also interesting to note that M_{ij} has columns of equal 2-norm. This is not surprising in view of a result of van der Sluis [60], which states that scaling the columns of an $n \times n$ matrix to have equal 2-norms produces a matrix with 2-norm condition number within a factor $\sqrt{n}$ of the minimum over all column scalings.

To complete our analysis we need to bound $\|A_k\|_2$ and $\|N_i^{-1}\|_2$.

3.3.1 Growth of A_m

We now bound $\|A_m\|_2$, which appears in the bound (3.25). We consider the growth over one step from $A_m = (a_{ij})$ to $A_{m+1} = (\tilde{a}_{ij}) = M_m^T A_m M_m$, as measured by $\phi_m = \max_{i,j} |\tilde{a}_{ij}| / \max_{i,j} |a_{ij}|$. By rewriting (2.7) in terms of A_k, and using (3.24) and (3.30), we can show that

$$|\tilde{a}_{ii}| \leq c^2 |a_{ii}| + s^2 |a_{ii}|/\rho^2 + |a_{ij}| \left(\frac{|s^3|}{c\rho} + |sc|\rho \right), \tag{3.34}$$

$$|\tilde{a}_{jj}| \leq c^2 |a_{jj}| + s^2 |a_{jj}|\rho^2 + |a_{ij}| \left(\frac{|sc|}{\rho} + \frac{|s^3|}{c}\rho \right). \tag{3.35}$$

We would like to bound these two elements linearly in terms of $\max(\rho, 1/\rho)$ (recall that ρ can be greater than or less than 1). The troublesome terms in the bounds are $s^2|a_{jj}|\rho^2$ and $s^2|a_{ii}|/\rho^2$. Upon substitution of (3.18) and (3.19) in (3.34) and (3.35) we obtain bounds linear in ρ and $1/\rho$:

$$|\tilde{a}_{ii}| \leq c^2|a_{ii}| + s^2|a_{jj}| + |a_{ij}| \left(\left(\sqrt{t^2 - 4s^4} + \frac{|s^3|}{c} \right) \frac{1}{\rho} + |sc|\rho \right),$$
$$|\tilde{a}_{jj}| \leq c^2|a_{jj}| + s^2|a_{ii}| + |a_{ij}| \left(\left(\sqrt{t^2 - 4s^4} + \frac{|s^3|}{c} \right) \rho + \frac{|sc|}{\rho} \right).$$

Using (3.20) we find that $\sqrt{t^2 - 4s^4} + |s^3|/|c| = |sc|$, and so

$$|\tilde{a}_{ii}| \leq c^2|a_{ii}| + s^2|a_{jj}| + |a_{ij}||sc| \left(\rho + 1/\rho\right), \tag{3.36}$$
$$|\tilde{a}_{jj}| \leq c^2|a_{jj}| + s^2|a_{ii}| + |a_{ij}||sc| \left(\rho + 1/\rho\right). \tag{3.37}$$

For the other affected elements in rows and columns i and j we have, for $k \neq i, j$,

$$\tilde{a}_{ik} = \tilde{a}_{ki} = a_{ik}c\sqrt{c^2 + s^2/\rho^2} - a_{jk}s\sqrt{s^2 + c^2\rho^2},$$
$$\tilde{a}_{jk} = \tilde{a}_{kj} = a_{ik}s\sqrt{s^2 + c^2/\rho^2} + a_{jk}c\sqrt{c^2 + s^2\rho^2}.$$

These elements can be bounded by

$$|\tilde{a}_{ik}| \leq |a_{ik}| \left(c^2 + |sc|/\rho\right) + |a_{jk}| \left(s^2 + |sc|\rho\right), \tag{3.38}$$
$$|\tilde{a}_{jk}| \leq |a_{ik}| \left(s^2 + |sc|/\rho\right) + |a_{jk}| \left(c^2 + |sc|\rho\right). \tag{3.39}$$

The bounds (3.36)–(3.39) can all be written in the form

$$|\tilde{a}_{pq}| \leq \max_{r,s} |a_{rs}| \big(1 + |sc|(\rho + 1/\rho)\big)$$

and so the growth of A_m over one step is bounded by

$$\phi_m \leq 1 + |sc|(\rho + 1/\rho) \leq 1 + 2|sc| \max(\rho, 1/\rho) = 1 + 2\omega_m.$$

The overall growth bound is

$$\pi_m := \frac{\|A_m\|_2}{\|A_0\|_2} \leq \sqrt{n} \prod_{i=0}^{m-1} \phi_i. \tag{3.40}$$

3.3.2 Bounding $\|N_i^{-1}\|_2$

Our final task is to bound

$$\mu_i := \|N_i^{-1}\|_2 = \|D_i^{-1}Q_{i-1}^T \dots Q_0^T D_0\|_2$$

(see (3.23)). We describe two different bounds. In view of (3.32),

$$\|D_{i+1}^{-1}\|_2 \le \|D_i^{-1}\|_2 \le \cdots \le \|D_0^{-1}\|_2.$$

Thus, since $D_0 = D$, where B has the Cholesky factorization (3.4),

$$\mu_i^2 \le \kappa_2(D)^2 \le \kappa_2(L)\kappa_2(B).$$

However, the point of our analysis is to avoid a $\kappa_2(B)$ term in the bounds. As an alternative way of bounding μ_i we note that, from (3.27),

$$N_i^{-1} = M_{i-1}^{-1} \dots M_0^{-1}.$$

For the row cyclic ordering in (2.10) the congruence transformations can be re-ordered into $2n-3$ groups of up to $\lceil n/2 \rceil$ disjoint transformations $M_{j+1}, \dots, M_{j+p}$ such that, using (3.33),

$$\|M_{j+p}^{-1} \dots M_{j+1}^{-1}\|_2 \le \sqrt{2}.$$

For example a sweep of a 6×6 matrix can be divided into 9 groups of disjoint rotations:

$$\begin{bmatrix} - & 1 & 2 & 3 & 4 & 5 \\ & - & 3 & 4 & 5 & 6 \\ & & - & 5 & 6 & 7 \\ & & & - & 7 & 8 \\ & & & & - & 9 \\ & & & & & - \end{bmatrix}.$$

Here, an integer k in position (i, j) denotes that the (i, j) element is eliminated on the kth step by a rotation in the (i, j) plane, and all rotations on the kth step

are disjoint. Hence we can bound μ_i by

$$\mu_i \leq (\sqrt{2})^{2n-3} = 2^{n-3/2}.$$

Although exponential in n, this bound is independent of $\kappa_2(B)$.

3.3.3 Summary

Our backward error analysis shows that, upon convergence after m Jacobi rotations, Algorithm 3.3.1 has computed a diagonal Λ such that

$$X^T(A + \Delta A)X = \Lambda, \quad X^T(B + \Delta B)X = I,$$

for some nonsingular X, where

$$\|\Delta A\|_2 \leq \tilde{\gamma}_{n^2}\|A\|_2\left(\kappa_2(L)^2 + \sum_{k=0}^{m-1}\mu_k^2\big(1 + 2\omega_k\big)\pi_k\right),$$
$$\|\Delta B\|_2 \leq \tilde{\gamma}_{n^2}\|B\|_2.$$

The term involving $\kappa_2(L)$ takes account of errors in the first stage of Algorithm 3.3.1 and follows from standard error analysis [31, Chap. 10] of Cholesky factorization and the solution of triangular systems. Because of the complete pivoting, $\kappa(L)$ is bounded as in (3.9), and in practice it is usually small. Even when $\kappa(L)$ is large, its full effect tends not to be felt on the backward error, since triangular systems are typically solved to higher accuracy than the bounds suggest [31, Chap. 8].

We do not have a bound better than exponential in n for the term μ_i^2, but this term has been less than 10 in virtually all our numerical tests. We showed in Section 3.3.1 that the growth factor $\pi_k = \|A_k\|_2/\|A_0\|_2$ in (3.40) is certainly bounded by $\pi_k \leq \sqrt{n}\prod_{i=0}^{k-1}(1 + 2\omega_i)$. The term

$$\omega_k = |s_k c_k| \max(\rho_k, 1/\rho_k) \leq |s_k c_k|\kappa_2(D) \leq |s_k c_k|\kappa_2(L)\kappa_2(B)^{1/2}$$

is the most important quantity in our analysis. A large value of ω_k, for some k, is the main indicator of instability in Algorithm 3.3.1.

We stress that our error bounds do not depend on the ordering (3.8), as should be expected since the Jacobi method is insensitive to the ordering of the diagonal of D. The purpose of pivoting in the Cholesky factorization is to keep L well conditioned and thereby to concentrate any ill conditioning of B into D.

The conclusion from the error analysis is that Algorithm 3.3.1 has much better stability properties than the bound (3.11) suggests. When $\kappa_2(B)$ is large it is usually the case that small values of $|s_k c_k|$ cancel any large values of $\max(\rho_k, 1/\rho_k)$ (see the discussion following Lemma 3.3.4), and that π_k is also small, with a resulting small backward error bound.

3.4 Error Analysis of the Cholesky–QR Method

We now consider the Cholesky–GQR method where we use the symmetric QR method, with Givens tridiagonalization, to solve the symmetric eigenproblem in (3.5). We note that this is different to the more popular Cholesky–HQR method which uses the symmetric QR method, with Householder tridiagonalization, for which a full analysis is not given in this thesis.

Algorithm 3.4.1 (Cholesky–GQR Method) *Given $A, B \in \mathbb{R}^{n\times n}$ with A symmetric and B symmetric positive definite, this algorithm calculates the eigenvalues λ_i and corresponding eigenvectors x_i of the pair (A, B).*

1. Compute the Cholesky factorization with complete pivoting $\Pi^T B \Pi = LD^2L^T$.
   Form $H = D^{-1}L^{-1}\Pi^T A \Pi L^{-T} D^{-1}$ by solving triangular systems.
   $X = \Pi L^{-T} D^{-1}$
2. % Symmetric QR algorithm with Wilkinson shift
   Use Algorithm 2.2.2 (Givens method) to form $H = Q_1^T T Q_1$,

where T is tridiagonal.

Use Algorithm 2.2.5 to form $T = Q_2^T D Q_2$.

$X = XQ_1Q_2$

$\lambda_i = d_{ii}$, $x_i = X(:, i)$

We consider the second part of Algorithm 3.4.1, beginning with the calculation of the Givens rotation, which has a different construction to that of the Jacobi rotation in Lemma 3.3.2.

Lemma 3.4.2 *[31, Lemma 18.6] If $x, y \in \mathbb{R}$, let Q_k be a Givens rotation such that*

$$Q_k^T \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} z \\ 0 \end{bmatrix}.$$

If $c = \cos\theta$ and $s = \sin\theta$ are computed according to

$$s = -\frac{y}{\sqrt{x^2 + y^2}}, \quad c = \frac{x}{\sqrt{x^2 + y^2}},$$

then the computed $\widehat{c}$ and $\widehat{s}$ satisfy

$$\widehat{c} = c(1 + \theta_4), \quad \widehat{s} = s(1 + \theta_4'),$$

where c and s are exact values for $[x\ y]^T$.

Proof. Straightforward. □

As in the analysis of the Cholesky–Jacobi method we will assume that the computed $\widehat{c}$ and $\widehat{s}$ are exact. It is easily checked that, in view of Lemma 3.4.2, this simplification does not affect the bounds.

Lemma 3.4.3 *If a Givens similarity transformation is applied to the matrix H_m in the (i, j)-plane then the computed $\widehat{H}_{m+1}$ satisfies*

$$\widehat{H}_{m+1} = Q_m^T \left(H_m + \Delta H_m\right) Q_m,$$

where the elements of ΔH_m can be bounded componentwise for $k \neq i, j$ by

$$|\Delta h_{ik}| \leq \tilde{\gamma}_1 \left(|h_{ik}| + 2|sc||h_{jk}| \right),$$
$$|\Delta h_{jk}| \leq \tilde{\gamma}_1 \left(|h_{jk}| + 2|sc||h_{ik}| \right).$$

For $k = i, j$,

$$|\Delta h_{ii}| \leq \tilde{\gamma}_1 \left(|h_{ii}| + 4|sc||h_{ij}| + 4|sc|^2|h_{jj}| \right),$$
$$|\Delta h_{ij}|, |\Delta h_{ji}| \leq \tilde{\gamma}_1 \left((1 + 4|sc|^2)|h_{ij}| + 2|sc|(|h_{ii}| + |h_{jj}|) \right),$$
$$|\Delta h_{jj}| \leq \tilde{\gamma}_1 \left(|h_{jj}| + 4|sc||h_{ij}| + 4|sc|^2|h_{ii}| \right).$$

Proof. For $k \neq i, j$ we follow the proof of Lemma 3.3.3. For $k = i, j$ we do not set elements to zero or have simplified formulae such as (2.7), as the elements that form the Givens rotation do not depend on the elements of H_m's 2×2 (i, j)-submatrix. Instead we have

$$\widehat{h}_{ii} = fl(c^2 h_{ii} + 2sch_{ij} + s^2 h_{jj}),$$
$$\widehat{h}_{ij} = fl((c^2 - s^2)h_{ij} + sc(h_{jj} - h_{ii})),$$
$$\widehat{h}_{jj} = fl(s^2 h_{ii} - 2sch_{ij} + s^2 h_{jj}).$$

The backward perturbations Δh_{ii}, Δh_{ij} and Δh_{jj} satisfy

$$Q_m^T \left(\begin{bmatrix} h_{ii} & h_{ij} \\ h_{ij} & h_{jj} \end{bmatrix} + \begin{bmatrix} \Delta h_{ii} & \Delta h_{ij} \\ \Delta h_{ij} & \Delta h_{jj} \end{bmatrix} \right) Q_m = \begin{bmatrix} \widehat{h}_{ii} & \widehat{h}_{ij} \\ \widehat{h}_{ji} & \widehat{h}_{jj} \end{bmatrix}.$$

Using this we follow the same path as in the second part of the proof of Lemma 3.3.3 to yield the results. □

Notice the difference in the 2×2 (i, j)-submatrix of ΔH_m in Lemma 3.3.3 and Lemma 3.4.3. This will mean that we will not be able to reduce potentially large ρ^2 and $1/\rho^2$ terms to linear terms in the following lemma.

Lemma 3.4.4 *If a Givens similarity transformation is applied to the matrix $H_m = D_m^{-1} A_m D_m^{-1}$ in the (i, j)-plane, then the computed $\widehat{H}_{m+1}$ satisfies*

$$\widehat{H}_{m+1} = Q_m^T D_m^{-1} \left(A_m + \Delta A_m \right) D_m^{-1} Q_m,$$

where the elements of ΔA_m can be bounded componentwise for $k \neq i, j$ by

$$|\Delta a_{ik}| \leq \tilde{\gamma}_1 \left(|a_{ik}| + 2|sc||a_{jk}|\rho \right),$$
$$|\Delta a_{jk}| \leq \tilde{\gamma}_1 \left(|a_{jk}| + 2|sc||a_{ik}|/\rho \right),$$

where $\rho = d_i/d_j$. For $k = i, j$,

$$\begin{aligned}
|\Delta a_{ii}| &\leq \tilde{\gamma}_1 \left(|a_{ii}| + 4|sc||a_{ij}|\rho + 4|sc|^2|a_{jj}|\rho^2 \right) \\
&\leq (1 + 2|sc|\rho)^2 \|A\|_2, \\
|\Delta a_{ij}|, |\Delta a_{ji}| &\leq \tilde{\gamma}_1 \left((1 + 4|sc|^2)|a_{ij}| + 2|sc|(|a_{ii}|/\rho + |a_{jj}|\rho) \right) \\
&\leq (1 + 2|sc|\rho)(1 + 2|sc|/\rho)\|A\|_2, \\
|\Delta a_{jj}| &\leq \tilde{\gamma}_1 \left(|a_{jj}| + 4|sc||a_{ij}|/\rho + 4|sc|^2|a_{ii}|/\rho^2 \right) \\
&\leq (1 + 2|sc|/\rho)^2 \|A\|_2.
\end{aligned}$$

Proof. Straightforward scaling of the bounds in Lemma 3.4.3. □

Lemma 3.4.4 shows that the Givens rotation results in a small backward perturbation of A_m provided $\omega = |sc| \max(\rho, 1/\rho)$ is of order 1. In the Cholesky–GQR method most Givens rotation are constructed to satisfy

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} h_{ik} \\ h_{jk} \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

where $H_m = D_m^{-1} A_m D_m^{-1}$ and $H_m = (h_{ij})$ and $A_m = (a_{ij})$. It is easy to show that

$$s = \frac{-a_{jk}\rho}{\sqrt{a_{ik}^2 + (a_{jk}\rho)^2}}, \quad c = \frac{a_{ik}}{\sqrt{a_{ik}^2 + (a_{jk}\rho)^2}} \tag{3.41}$$

which gives

$$|sc| = \frac{|a_{ik}a_{jk}|\rho}{a_{ik}^2 + (a_{jk}\rho)^2}.$$

If we examine the two dangerous terms in Lemma 3.4.4 we see that

$$|sc|\rho = |x| \left(\frac{\rho^2}{x^2 + \rho^2} \right) = \frac{\rho^2}{|x|} \left(\frac{x^2}{x^2 + \rho^2} \right),$$
$$|sc|/\rho = \frac{1}{|x|} \left(\frac{x^2}{x^2 + \rho^2} \right) = \frac{|x|}{\rho^2} \left(\frac{\rho^2}{x^2 + \rho^2} \right),$$

where $|x| = |a_{ik}/a_{jk}|$. Using this we can show that

$$\begin{aligned} |sc|\rho &\leq \min\left(\rho^2/|x|, |x|\right), \\ |sc|/\rho &\leq \min\left(|x|/\rho^2, 1/|x|\right). \end{aligned} \tag{3.42}$$

These bounds in (3.42) are within a factor of 2 of the actual values and illustrate more clearly how $|sc|\rho$ or $|sc|/\rho$ can be large. If $|a_{ik}|$ and $|a_{jk}|$ are similarly sized then $|x| \approx 1$ and $\omega = |sc| \max(\rho, 1/\rho)$ will be of order 1 regardless of how ill-conditioned D_m is (or how large ρ or $1/\rho$ is). However large values of ω can happen when ρ or $1/\rho$ is large when the size of $|a_{ik}|$ and $|a_{jk}|$ become dissimilar and $|x|$ or $1/|x|$ are large.

We also have to consider the use of shifts in the first rotation of Algorithm 2.2.4. In these instances the Givens rotations are constructed such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} h_{11} - \mu \\ h_{12} \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

Again we can show that

$$|sc| = \frac{|(a_{11} - \mu d_1^2)a_{21}|\rho}{(a_{11} - \mu d_1^2)^2 + (a_{21}\rho)^2}.$$

Using this we can show that

$$\begin{aligned} |sc|\rho &\leq \min\left(\rho^2/|y|, |y|\right), \\ |sc|/\rho &\leq \min\left(|y|/\rho^2, 1/|y|\right), \end{aligned} \tag{3.43}$$

where $y = (a_{11} - \mu d_1^2)/a_{21}$. The size of the shift is important because if $|y|$ is of order 1 then $\omega = |sc| \max(\rho, 1/\rho)$ will be of order 1 regardless of how ill-conditioned D_m is (or how large ρ or $1/\rho$ is).

By combining Lemma 3.4.4 with subsequent applications of Lemma 3.4.3 we find that after m Givens rotations on $H_0 = D_0^{-1} A_0 D_0^{-1}$ we have

$$\widehat{H}_m = Q_{m-1}^T \ldots Q_0^T (H_0 + \Delta_0) Q_0 \ldots Q_{m-1},$$

where Δ_0 is the same as in (3.21). The ΔH_k are bounded as in Lemma 3.4.3. Again we introduce a scaling to $\widehat{H}_k$ at each stage of the iteration so we can bound

the terms of Δ_0 using (3.22), (3.23) and (3.24). By applying Lemma 3.4.4 to a rotation on H_k, we can see that

$$\|D_k \Delta H_k D_k\|_2 \leq \tilde{\gamma}_n (1 + 2\omega_k)^2 \|A_k\|_2, \tag{3.44}$$

where

$$\omega_k = |s_k c_k| \max(\rho_k, 1/\rho_k), \qquad \rho_k = d_i^{(k)}/d_j^{(k)},$$

with a subscript k denoting quantities on the kth step and where $D_k = \mathrm{diag}(d_i^{(k)})$. Again we use the idea based on the algorithm of Anjos, Hammarling and Paige [2] to try to bound $\|\Delta_0\|_2$. We recall that the AHP algorithm is mathematically identical to the Cholesky-Jacobi method and has similar stability properties. The analogous algorithm for the Cholesky–GQR algorithm (Algorithm 3.4.1) is the MDR Algorithm [5]. The MDR Algorithm uses transformation matrices of the form in (3.26) (the transformation matrices used in the MDR Algorithm are a scaled version of those in (3.31)) to diagonalize A while retaining the diagonal form of D_0. The algorithm computes the congruence transformations

$$A_{k+1} = M_k^T A_k M_k, \quad D_{k+1}^2 = M_k^T D_k^2 M_k,$$

where D_k is diagonal for all k and A_k tends to diagonal form as $k \to \infty$. The difference between our approach and that in [5] is that we form $H_0 = D_0^{-1} A_0 D_0^{-1}$ and use D_k in the analysis to obtain stronger error bounds, whereas in [5], in an effort to apply only well conditioned similarity transformations, H_0 is never formed but M_k is computed and applied in the algorithm (and no error analysis is given in [5]).

3.4.1 Growth of A_m and Bounding $\|N_i^{-1}\|$

We now bound $\|A_m\|_2$, which appears in the bound (3.44). If we consider the growth over one step from $A_m = (a_{ij})$ to $A_{m+1} = (\tilde{a}_{ij}) = M_m^T A_m M_m$, as measured

by $\phi_m = \max_{i,j} |\tilde{a}_{ij}| / \max_{i,j} |a_{ij}|$ then the best we can say is

$$\phi_m \leq \|M_m\|_2^2$$

as the Givens rotations generated at each stage do not depend on any of the elements of $A_{ij} = A([i\ j],[i\ j])$. As $A_k = N_k^T A_0 N_k$ where $N_k = M_0 \dots M_{k-1}$ then the overall growth bound is

$$\frac{\|A_m\|_2}{\|A_0\|_2} \leq \|N_m\|_2^2 \leq \prod_{i=0}^{m-1} \|M_i\|_2^2. \tag{3.45}$$

As $N^{-1} = M_{k-1}^{-1} \dots M_0^{-1}$ we bound this by

$$\|N^{-1}\|_2 \leq \prod_{i=0}^{k-1} \|M_i^{-1}\|_2.$$

Therefore together we have

$$\pi_m^2 := \|N^{-1}\|_2^2 \frac{\|A_m\|_2}{\|A_0\|_2} \leq \prod_{i=0}^{m-1} \kappa_2(M_i)^2 \tag{3.46}$$

where the condition number of M_i is given by (3.28) which depends on $|sc|\rho$ or $|sc|/\rho$.

3.4.2 Summary

Our backward error analysis shows that, upon convergence after m Givens rotations (including those used in the tridiagonalization), Algorithm 3.4.1 has computed a diagonal Λ such that

$$X^T(A + \Delta A)X = \Lambda, \quad X^T(B + \Delta B)X = I,$$

for some nonsingular X, where

$$\|\Delta A\|_2 \leq \tilde{\gamma}_{n^2} \|A\|_2 \left(\kappa_2(L)^2 + \sum_{k=0}^{m-1} \big(1 + 2\omega_k\big)^2 \pi_k^2 \right),$$
$$\|\Delta B\|_2 \leq \tilde{\gamma}_{n^2} \|B\|_2.$$

The term involving $\kappa_2(L)$, which also appears in the analysis of the Cholesky–Jacobi method, takes account of errors in the first stage of Algorithm 3.4.1 . Again the term

$$\omega_k = |s_k c_k| \max(\rho_k, 1/\rho_k) \leq |s_k c_k| \kappa_2(D) \leq |s_k c_k| \kappa_2(L) \kappa_2(B)^{1/2}$$

is the most important quantity in our analysis and a large value of ω_k, for some k, is the main indicator of instability in Algorithm 3.3.1. The noticeable difference between the two analyses is that when ω_k is large for the Cholesky–Jacobi method its effect on the backward error is linear, but when ω_k is large for the Cholesky–GQR method its effect is squared. As these two methods perform different operations these ω_k are different for each algorithm. Examples are shown in Chapter 6 for which the Cholesky–Jacobi method has large values of ω_k while the Cholesky–GQR method has small values and is backward stable, and vice versa.

3.4.3 Graded Matrices

First we consider the matrix $H_0 = D_0^{-1} A_0 D_0^{-1}$ which is graded upwards. This is where the diagonal elements of D_0 are ordered such that $d_1 \geq \cdots \geq d_n$ and the elements of A_0 are generally all of the same order. When forming the tridiagonal reduction of H_0, the Givens rotations, whose formulae is given in (3.41), should have $s = \sin\theta$ larger than $c = \cos\theta$ when ρ is large. Using (3.30) we see that our scaling will change such that

$$\widetilde{d}_i^2 = d_j^2 \frac{a_{ik}^2 + a_{jk}^2}{(a_{ik}/\rho)^2 + a_{jk}^2} \approx d_j^2,$$
$$\widetilde{d}_j^2 = d_i^2 \frac{(a_{ik}/\rho)^2 + (a_{jk}\rho)^2}{a_{ik}^2 + (a_{jk}\rho)^2} \approx d_i^2,$$

Overall, the tridiagonal reduction will give $H_T = \widetilde{D}^{-1} T \widetilde{D}^{-1}$ where T is a tridiagonal matrix whose nonzero elements should be similarly sized, $\widetilde{d}_1 = d_1$ and the

remaining elements of $\widetilde{D}$ are more than likely ordered $\widetilde{d}_2 \leq \cdots \leq \widetilde{d}_n$. Consider

$$H_0 = D_0^{-1} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 2 & 3 & 3 \\ 1 & 2 & 3 & 4 \end{bmatrix} D_0^{-1}$$

where $D_0^2 = \text{diag}(1, 10^{-5}, 10^{-10}, 10^{-15})$. If we compute the tridiagonal reduction then we get

$$H_T \approx \widetilde{D}^{-1} \begin{bmatrix} 1 & 1 & & \\ 1 & 12 & 3 & \\ & 3 & 3 & 1.7 \\ & & 1.7 & 2 \end{bmatrix} \widetilde{D}^{-1}$$

where $\widetilde{D}^2 \approx \text{diag}(1, 3 \times 10^{-15}, 3 \times 10^{-10}, 10^{-6})$.

If we now compute the implicit QR step on H_T a large value of ω_k can occur. The first step is to calculate the shift μ which is the eigenvalue of H_T's lower 2×2 principal submatrix that is closest to h_{nn}. This submatrix has elements that range in size from 10^5 to 10^{10}. Direct calculation shows that $\mu \approx 10^5$ and therefore

$$|y| = |(a_{11} - \mu d_1^2)/a_{21}| \approx 10^5.$$

Using this in (3.43) we can show that

$$|sc|\rho \leq \min(10^{15}/|y|, |y|) \approx 10^5$$

which leads to large backward error and large growth of A_k.

We now consider the graded matrix $H_0 = D_0^{-1} A_0 D_0^{-1}$ where the diagonal elements of D_0 are ordered such that $d_1 \leq \cdots \leq d_n$ and the elements of A_0 are all of the same size. When ρ is small then s should be small compared with c. Using (3.30) we see that our scaling will not change much, that is $\widetilde{d}_i^2 \approx d_i^2$ and $\widetilde{d}_j^2 \approx d_j^2$. Overall, the tridiagonal reduction will give $H_T = \widetilde{D}^{-1} T \widetilde{D}^{-1}$ where T is a

tridiagonal matrix of order 1, $\widetilde{d}_1 = d_1$ and the remaining elements of $\widetilde{D}$ are more than likely ordered $\widetilde{d}_2 \leq \cdots \leq \widetilde{d}_n$. Consider a permuted version of the previous example

$$H_0 = D_0^{-1} \begin{bmatrix} 4 & 3 & 2 & 1 \\ 3 & 3 & 2 & 1 \\ 2 & 2 & 2 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} D_0^{-1}$$

where $D_0^2 = \mathrm{diag}(10^{-15}, 10^{-10}, 10^{-5}, 1)$. If we compute the tridiagonal reduction then we get

$$H_T \approx \widetilde{D}^{-1} \begin{bmatrix} 4 & 3.7 & & & \\ 3.7 & 4.7 & 6.2 \times 10^{-6} & & \\ & 6.2 \times 10^{-6} & 0.8 & 1.2 \times 10^{-11} & \\ & & 1.2 \times 10^{-11} & 0.5 & \end{bmatrix} \widetilde{D}^{-1}$$

where $\widetilde{D}^2 \approx \mathrm{diag}(10^{-15}, 1.6 \times 10^{-10}, 1.3 \times 10^{-5}, 1)$. Here there is less of a problem with the implicit QR step as the shift μ will tend to be small. Direct calculation shows that $\mu \approx 0.5$ and therefore, unless there is any unfortunate cancellation, we have

$$|y| = |(a_{11} - \mu d_1^2)/a_{21}| \approx |a_{11}/a_{21}|.$$

Assuming the elements of A are of similar size then $1/|y|$ will be order 1 and therefore $|sc|/\rho$ will be small.

In conclusion, we have observed that the tridiagonal reduction will tend to approximately reverse the scaling in the elements $d_2, \ldots, d_n$ of $H_0 = D_0^{-1} A_0 D_0^{-1}$ when H_0 is graded upwards. This reduction can be done in a stable manner provided the elements of A_0 are of the same order but problems may occur when applying the shifted QR Givens rotation, as observed. This does not happen when $H_0 = D_0^{-1} A_0 D_0^{-1}$ is graded downwards as the tridiagonal reduction does not affect the scaling and the shifted QR Givens rotation can usually be applied

stably. Therefore it is preferable to apply the symmetric QR algorithm to matrices which are graded downwards.

3.5 Chandrasekaran's Algorithm

A deflation method has been proposed by Chandrasekaran [8]. This method starts by computing the eigendecomposition

$$G^T A^{-1} G = V \Phi V^T \tag{3.47}$$

where $B = GG^T$ and $\Phi = \text{diag}(\phi_1, \ldots, \phi_n)$ and the eigenvalues are ordered by decreasing magnitude. Notice that this is the inverse of the matrix $G^{-1}AG^{-T}$ which is formed in the Cholesky method. From (3.47) we have a matrix of eigenvectors of the matrix pair (A, B), $X = A^{-1}GV$ such that

$$X^T A X = \Phi, \quad X^T B X = \Phi^2.$$

The method uses the key observation that even when both A and B are ill-conditioned, the matrix $G^T A^{-1} G$ can be computed to sufficient accuracy such that the eigenpairs, corresponding to the largest eigenvalues of $G^T A^{-1} G$ in magnitude, will be computed with backward error of order u. This means that $(\widehat{x}_i, \widehat{\phi}_i)$ corresponding to the larger $\widehat{\phi}$ satisfy

$$\|(\widehat{\phi}_i A - B)\widehat{x}_i\| \lesssim u\|\widehat{x}_i\| \left(|\widehat{\phi}_i| \|A\| + \|B\| \right).$$

These large eigenvalues of $G^T A^{-1} G$ can then be deflated from A and B and we can then work recursively on the reduced problem. If these eigenvalues are deflated in order of decreasing size in magnitude, Chandrasekaran [8] shows that the sequence of deflating transformations can be applied in a numerically stable manner.

We now look at the deflation process. We have the computed eigendecomposition of $G^T A^{-1} G$ in (3.47) and the matrix of eigenvectors $X = A^{-1}GV$. We now

find a Householder matrix Q_1 such that $Q_1 x_1 = \pm\|x_1\|_2 e_1$ where x_1 is the first column of X. If we apply a similarity transformation using Q_1 we have

$$A \leftarrow Q_1 A Q_1^T = \begin{array}{c} \\ 1 \\ n-1 \end{array} \begin{array}{c} \begin{array}{cc} 1 & n-1 \end{array} \\ \begin{bmatrix} \alpha_1 & a_1^T \\ a_1 & \bar{A}_1 \end{bmatrix} \end{array},$$

$$B \leftarrow Q_1 B Q_1^T = \begin{array}{c} \\ 1 \\ n-1 \end{array} \begin{array}{c} \begin{array}{cc} 1 & n-1 \end{array} \\ \begin{bmatrix} \beta_1 & b_1^T \\ b_1 & \bar{B}_1 \end{bmatrix} \end{array}.$$

If we update the eigenvector matrix we have

$$X \leftarrow Q_1 X = \begin{bmatrix} \pm\|x_1\|_2 & h_1^T \\ 0 & \bar{X}_1 \end{bmatrix}.$$

Because of the way the Householder matrix is formed we see that $Q_1 A Q_1^T y = \lambda Q_1 B Q_1^T y$ where $y = Q_1 x_1 = \|x\|_2 e_1$. This means that the first columns of $Q_1 A Q_1^T$ and $Q_1 B Q_1^T$ are parallel. We now find a Householder matrix $\bar{W}_1$ such that $\bar{W}_1 a_1 = \pm\|a_1\|_2 e_1$. Because $a_1$and b_1 are parallel we also have $\bar{W}_1 b_1 = \pm\|b_1\|_2 e_1$. Defining $W_1 = \mathrm{diag}(1, \bar{W}_1)$ we have

$$X \leftarrow W_1 X = \begin{bmatrix} \pm\|x_1\|_2 & h_1^T \\ 0 & \bar{W}_1 \bar{X}_1 \end{bmatrix},$$

$$A \leftarrow W_1 A W_1^T = \begin{bmatrix} \alpha_1 & \pm\|a_1\|_2 e_1^T \\ \pm\|a_1\|_2 e_1 & \bar{W}_1 \bar{A}_1 \bar{W}_1^T \end{bmatrix},$$

$$B \leftarrow W_1 B W_1^T = \begin{bmatrix} \beta_1 & \pm\|b_1\|_2 e_1^T \\ \pm\|b_1\|_2 e_1 & \bar{W}_1 \bar{B}_1 \bar{W}_1^T \end{bmatrix},$$

where $(\beta_1, \pm\|b_1\|_2)^T = \phi_1(\alpha_1, \pm\|a_1\|_2)^T$. We now find a Gauss transform L_1^{-1} to make the first columns of A and B to be parallel to e_1:

$$L_1^{-1} = \begin{bmatrix} 1 & 0 \\ \rho e_1 & I_{n-1} \end{bmatrix}, \quad \rho = -\frac{\pm\|b_1\|_2}{\beta_1} = -\frac{\pm\|a_1\|_2}{\alpha_1}.$$

Applying this transformation we have

$$X \leftarrow L_1^{-1}X = \begin{bmatrix} \pm\|x_1\|_2 & 0 \\ 0 & X_2 \end{bmatrix},$$

$$A \leftarrow L_1^{-1}AL_1^{-T} = \begin{bmatrix} \alpha_1 & 0 \\ 0 & \bar{W}_1\bar{A}_1\bar{W}_1^T - \frac{\|a_1\|_2^2}{\alpha_1}e_1e_1^T \end{bmatrix},$$

$$B \leftarrow L_1^{-1}BL_1^{-T} = \begin{bmatrix} \beta_1 & 0 \\ 0 & \bar{W}_1\bar{B}_1\bar{W}_1^T - \frac{\|b_1\|_2^2}{\beta_1}e_1e_1^T \end{bmatrix}.$$

The first row of X is obtained by noting that $X^TAX = \Phi$ and $X^TBX = \Phi^2$. We can apply this technique recursively if the calculated eigenpairs are accurate enough. If not we can recompute the eigendecomposition of the Schur complements of A and B and apply the same technique again to the accurate eigenpairs. We now give the algorithm in full.

Algorithm 3.5.1 (Chandrasekaran's Algorithm) *Given $A, B \in \mathbb{R}^{n\times n}$ such that A is symmetric and B is symmetric positive definite the following algorithm calculates the eigenvalues λ_i and corresponding eigenvectors of the pair (A, B).*

$\alpha = \|A\|_2/\|B\|_2$, $Y = I_n$.
Scale matrices A and B such that $\|A\|_2 = \|B\|_2$.
Compute eigendecomposition $B = U\Sigma U^T$.
Compute eigendecomposition $\sqrt{\Sigma}U^TA^{-1}U\sqrt{\Sigma} = V\Phi V^T$ where the
    eigenvalues are ordered from largest to smallest in magnitude.
Compute generalized eigenvectors of (A, B), $X = A^{-1}U\sqrt{\Sigma}V$.
for $i = 1\colon n-1$
    if $\|(\phi_iA - B)x_i\| \leq \epsilon\|x_i\|(|\phi_i|\|A\| + \|B\|)$
        Compute Householder matrix Q_i such that $Q_ix_i = \gamma e_i$.
        $A = Q_iAQ_i^T$, $B = Q_iBQ_i^T$
        $X = Q_iX$

$Y = YQ_i$

if $|\phi_i| \geq 1$

Make ith row and column of A and B exactly parallel.

$A(:,i) = B(:,i)/\lambda_i$, $A(i,:) = A(:,i)^T$

Find Householder matrix W_i such that $W_iB(:,i)$ has

zeros below the $(i+1)$th element.

$A = W_iAW_i^T$, $B = W_iBW_i^T$

$X = W_iX$

$Y = YW_i$

Find an elementary Gauss transform L_i^{-1} such that

$L_i^{-1}B(:,i) = \gamma e_i$

$A = L_i^{-1}AL_i^{-T}$, $B = L_i^{-1}BL_i^{-T}$

$X = L_i^{-1}X$

$Y = YL_i$

else if $|\phi_i| < 1$

Make ith row and column of A and B exactly parallel.

$B(:,i) = \lambda_iA(:,i)$, $B(i,:) = B(:,i)^T$

Find Householder matrix W_i such that $W_iA(:,i)$ has

zeros below the $(i+1)$th element.

$A = W_iAW_i^T$, $B = W_iBW_i^T$

$X = W_iX$

$Y = YW_i$

Find an elementary Gauss transform L_i^{-1} such that

$L_i^{-1}A(:,i) = \gamma e_i$

$A = L_i^{-1}AL_i^{-T}$, $B = L_i^{-1}BL_i^{-T}$

$X = L_i^{-1}X$

$Y = YL_i$

end

else

Recompute eigendecomposition of $B = U\Sigma U^T$.

Recompute eigendecomposition of $\sqrt{\Sigma}U^T A^{-1}U\sqrt{\Sigma} = V\Lambda V^T$

where the eigenvalues are ordered from largest to smallest in magnitude.

Recompute generalized eigenvectors of (A, B), $X = A^{-1}U\sqrt{\Sigma}V$.

end

end

$\lambda_i = \alpha A(i,i)/B(i,i)$, $x_i = Y(:,i)$, $i = 1{:}\,n$

We now discuss some of the implementation issues that were glossed over in Algorithm 3.5.1. Firstly all eigendecompositions are performed using the symmetric QR algorithm. Secondly, for the formation of the matrix $\sqrt{\Sigma}U^T A^{-1}U\sqrt{\Sigma}$ we first compute the eigendecomposition of $A = Z\Delta Z^T$. Then we compute

$$\sqrt{\Sigma}\left(\left(U^T Z^T\right)\Delta^{-1}\left(UZ\right)\right)\sqrt{\Sigma}$$

in the order suggested by the parentheses.

To form $G^T A^{-1}G$ we assume that A is nonsingular. The additive perturbations of the algorithm may cause A to become singular. When A is singular the zero eigenvalues of (A, B) can be deflated from the problem by computing a Householder transform H_s such that $H_s x_i = e_i$. When forming

$$A = H_s A H_s^T, \quad B = H_s B H_s^T$$

the ith row and column will be zero. Using elementary Gauss transforms we can we can reduce the ith row and column of B to γe_i.

The total flop count for this algorithm is $\leq O(n^4)$ but this upper limit has never been observed in practice. This algorithm always runs to completion with

an operation count of $O(n^3)$ flops in practice. Unfortunately, due to the potentially high number of eigendecompositions and deflation steps that can be needed, this algorithm can be a lot slower than the other methods mentioned.

Chapter 4

Iterative Refinement

4.1 Newton's Method

In Section 3.3 and 3.4 we have shown under what circumstances the Cholesky method can produce unstable solutions. We now consider the use of iterative refinement to restore numerical stability. The idea of using iterative refinement to improve numerical stability has been investigated for linear systems by several authors; see [31, Chap. 11] for a survey and [32] for the most recent results. Iterative refinement has previously been used with residuals computed in extended precision to improve the accuracy of approximate solutions to the standard eigenproblem [17], [16], [56]. Tisseur [57] shows how iterative refinement can be used in fixed or extended precision to improve the forward and backward errors of approximate solutions to the generalized eigenvalue problem. The generalized eigenvalue problem can be written as

$$Ax = \lambda Bx, \qquad e_s^T x = 1 \text{ (for some fixed } s)$$

where the second equation is needed because we have one degree of freedom in the choice of eigenvector (if x is an eigenvector then so is αx for some $\alpha \neq 0$) and we do not assume that A and B are symmetric. We can apply Newton's method

to the equivalent nonlinear equation problem

$$F(v) = F\left(\begin{bmatrix} x \\ \lambda \end{bmatrix}\right) = \begin{bmatrix} (A - \lambda B)x \\ \alpha e_s^T x - \alpha \end{bmatrix} : \mathbb{R}^{n+1} \to \mathbb{R}^{n+1} \tag{4.1}$$

where $\alpha = \max(\|A\|, \|B\|)$. Newton's iteration generates a sequence $\{v_k\}$ produced by

$$v_{k+1} = v_k - J(v_k)^{-1} F(v_k), \quad k = 0, 1, \ldots, \tag{4.2}$$

where $J(v_k) = F'(v_k)$ is the Jacobian. This requires solving linear systems whose coefficient matrices are

$$J(v) = J\left(\begin{bmatrix} x \\ \lambda \end{bmatrix}\right) = \begin{bmatrix} A - \lambda B & -Bx \\ \alpha e_s^T & 0 \end{bmatrix}. \tag{4.3}$$

To solve (4.2) we require the Jacobian in (4.3) to be nonsingular. We now consider the possibility of the Jacobian being singular at a zero of F.

Lemma 4.1.1 *[57] Let $v_* = (x_*^T, \lambda_*)^T$, where λ_* is finite, be a zero of F as defined in* (4.1). *Then $J(v_*)$ in* (4.3) *is singular if and only if λ_* is a multiple eigenvalue of (A, B).*

Proof. We start by supposing that $J(v_*)$ is singular. Using the formula [28]

$$\det\left(\begin{bmatrix} M & u \\ v^T & \mu \end{bmatrix}\right) = \mu \det(M) - v^T M^A u, \tag{4.4}$$

where M^A is the adjugate of M we obtain

$$0 = \det(J(v_*)) = e_s^T (A - \lambda_* B)^A B x_*$$

because $\det(A - \lambda_* B) = 0$ as λ_* is an eigenvalue of (A, B). The adjugate of a matrix M is just the transpose of the matrix of cofactors, C, which has the formula

$$C_{ij} = (-1)^{i+j} \det(M_{ij})$$

where M_{ij} is the matrix obtained by removing row i and column j from M. The adjugate has the property that

$$M^A M = \det(M) I$$

so if we define $y^T = e_s^T (A - \lambda_* B)^A$ then

$$y^T (A - \lambda_* B) = e_s^T \det(A - \lambda_* B) I = 0.$$

Therefore y is a left eigenvector of the matrix pair (A, B) corresponding to the eigenvalue λ_* and

$$y^T B x_* = e_s^T (A - \lambda_* B)^A B x_* = 0.$$

If λ_* was a simple eigenvalue with corresponding left and right eigenvectors y and x then $y^T B x_* \neq 0$. Therefore λ_* must be an eigenvalue with multiplicity at least 2.

If λ_* is a multiple eigenvalue then there exists a left eigenvector y such that

$$(y^T \; 0) \begin{bmatrix} A - \lambda_* B & -B x_* \\ e_s^T & 0 \end{bmatrix} = 0$$

and therefore $J(v_*)$ is singular. □

Because we keep one element of v_k constant during the iteration (4.2) we can reduce the problem to one of size n. If we write

$$v_{k+1} = v_k + \Delta v_k = \begin{bmatrix} x_k + \Delta x_k \\ \lambda_k + \Delta \lambda_k \end{bmatrix}$$

then the iteration (4.2) becomes

$$(A - \lambda_k B)\Delta x_k - \Delta \lambda_k B x_k = r_k, \quad e_s^T x_{k+1} = e_s^T x_k = 1, \tag{4.5}$$

where $r_k = \lambda_k B x_k - A x_k$ is the residual. As in [16] we note that $e_s^T \Delta x_k = 0$ and therefore the sth column of $A - \lambda_k B$ does not participate in the iteration. If we define

$$\delta_k = \Delta x_k + \Delta \lambda_k e_s$$

and

$$M_k = (A - \lambda_k B) - ((A - \lambda_k B)e_s + Bx_k)e_s^T \tag{4.6}$$

then we can rewrite the iteration (4.5) as

$$M_k\delta_k = r_k, \quad \lambda_{k+1} = \lambda_k + e_s^T\delta_k, \quad x_{k+1} = x_k + \delta_k - e_s^T\delta_k e_s. \tag{4.7}$$

Notice that the matrix M_k in (4.6) is the matrix $A - \lambda_k B$ where the sth column has been replaced with $-Bx_k$. We now give the algorithm in full, which is a straightforward implementation of (4.7).

Algorithm 4.1.2 *Given general $A, B \in \mathbb{R}^{n\times n}$ and an approximate eigenpair (x, λ) with $\|x\|_\infty = x_s = 1$, this algorithm applies iterative refinement to λ and x:*

repeat until convergence

$r = \lambda Bx - Ax$

Form M: the matrix $A - \lambda B$ with column s replaced by $-Bx$.

Factor $PM = LU$ (LU factorization with partial pivoting)

Solve $M\delta = r$ using the LU factors

$\lambda = \lambda + \delta_s$; $\delta_s = 0$

$x = x + \delta$

end

This algorithm is expensive as each iteration requires $O(n^3)$ flops for the factorization of M. If A and B are symmetric and simultaneously diagonalizable then we can take advantage of the eigendecomposition computed by Algorithm 3.3.1 or 3.4.1, the cost per iteration can be reduced to $O(n^2)$ flops. We assume we have computed a nonsingular matrix X and a diagonal matrix D such that $X^TAX \approx D = \text{diag}(\lambda_1, \ldots, \lambda_n)$ and $X^TBX \approx I_n$. Then

$$\begin{aligned} X^T r_k &= X^T M_k \delta_k \\ &= ((D - \lambda_k I) - X^T((A - \lambda_k B)e_s + Bx_k)e_s^T X)X^{-1}\delta_k. \end{aligned} \tag{4.8}$$

If we define the terms

$$D_{\lambda_k} = D - \lambda_k I, \quad v_k = X^T((A - \lambda_k B)e_s + Bx_k),$$

$$f = X^T e_s, \quad w_k = X^{-1}\delta_k, \quad g_k = X^T r_k$$

then the iteration in (4.8) becomes

$$(D_{\lambda_k} - v_k f^T)w_k = g_k \tag{4.9}$$

The matrix in (4.9) is a rank one modification to a diagonal matrix. As D_{λ_0} is singular we cannot use the Sherman-Morrison-Woodbury formula. One way of solving (4.9) cheaply is to reduce $D_{\lambda_k} - v_k f^T$ to upper Hessenberg form, which can be done in $n-1$ rotations. We define the rotations such that

$$J_1^T \dots J_{n-1}^T v_k = \pm \|v_k\|_2 e_1$$

where J_k is a rotation in the $(k, k+1)$ coordinate plane designed to zero the $(k+1)$th element of v_k. The product

$$J_1^T \dots J_{n-1}^T (D_{\lambda_k} - v_k f^T) = H \pm \|v_k\|_2 e_1 f^T = H_1$$

will be upper Hessenberg. Using a QR factorization of H_1, the solution of (4.9) can be found in $O(n^2)$ flops.

Algorithm 4.1.3 *Given symmetric $A, B \in \mathbb{R}^{n\times n}$, nonsingular $X \in \mathbb{R}^{n\times n}$ and diagonal $D \in \mathbb{R}^{n\times n}$ such that $X^TAX \approx D$ and $X^TBX \approx I$, and an approximate eigenpair (x, λ) with $\|x\|_\infty = x_s = 1$, this algorithm applies iterative refinement to λ and x at a cost of $O(n^2)$ flops per iteration.*

repeat until convergence

$r = \lambda Bx - Ax$

$D_\lambda = D - \lambda I$

$d = -Bx - c_{\lambda s}$, where $c_{\lambda s}$ is the sth column of $A - \lambda B$

$v = X^T d$; $f = X^T e_s$

Compute Givens rotations J_k in the $(k, k+1)$ plane, such that

$Q_1^T v := J_1^T \dots J_{n-1}^T v = \|v\|_2 e_1$

Compute orthogonal Q_2 such that

$T = Q_2^T Q_1^T (D_\lambda + v f^T)$ is upper triangular

$z = Q_2^T Q_1^T X^T r$

Solve $Tw = z$ for w

$\delta = Xw$

$\lambda = \lambda + \delta_s$; $\delta_s = 0$

$x = x + \delta$

end

4.2 Analysis of Newton's Method

In this section we consider a general function $F: \mathbb{R}^n \to \mathbb{R}^n$ and assume that it is continuously differentiable. We will also assume that there exists $v_* \in \mathbb{R}^n$ such that $F(v_*) = 0$ and the Jacobian $J(v)$ is Lipschitz continuous with constant β, that is, for all $\bar{v}, v \in \mathbb{R}^n$

$$\|J(\bar{v}) - J(v)\| \leq \beta \|\bar{v} - v\|.$$

First we consider the change in the forward error caused by a single step of Newton's method in floating point arithmetic. In floating point arithmetic Newton's method can be written as

$$\bar{v} = v - (J + E)^{-1}(r + e) + \epsilon \tag{4.10}$$

where for notational convenience $J = J(v)$, $r = F(v)$, $\bar{v} = \widehat{v}_{k+1}$ and $v = \widehat{v}_k$. The errors are categorized as:

- e is the error in calculating the residual. We assume that the residual can be computed in extended precision $\bar{u} \leq u$ and then rounded back to working precision u. We bound this error by

$$\|e\| \leq u\|r\| + \psi(F, v, u, \bar{u}).$$

- E is the error in forming the Jacobian $J(v)$ and solving the linear system. We bound this error by the function

$$\|E\| \leq u\phi(F, v, n, u)$$

which reflects the instability of the solver and the error made in approximating $J(v)$.

- ϵ is the error made when adding the correction to v. We have

$$\|\epsilon\| \leq u(\|v\| + \|d\|)$$

where

$$d = (J + E)^{-1}(r + e). \tag{4.11}$$

4.2.1 Forward Error

We now consider the change in the forward error caused by one step of the iteration (4.10). We will make use of the following lemma.

Lemma 4.2.1 *[14, Lemma 4.1.12] For any v, $\bar{v} \in \mathbb{R}^n$*

$$\|F(\bar{v}) - F(v) - J(v)(\bar{v} - v)\| \leq \frac{\beta}{2}\|\bar{v} - v\|^2. \tag{4.12}$$

Theorem 4.2.2 *[57, Theorem 2.2] Let $F\colon \mathbb{R}^n \to \mathbb{R}^n$, $v_* \in \mathbb{R}^n$ such that $F(v_*) = 0$ and assume that the Jacobian at the zero $J_* = J(v_*)$ is nonsingular and that*

$$\|J^{-1}E\| \leq \nu < 1. \tag{4.13}$$

Then for all v such that

$$\beta\|J_*^{-1}\|\|v - v_*\| \leq \mu < 1, \tag{4.14}$$

v in (4.10) *is well defined and*

$$\|\bar{v} - v_*\| \leq G\|v - v_*\| + g,$$

where

$$G = \frac{1}{1-\nu}\|J^{-1}E\| + \frac{(1+u)^2}{2(1-\mu)(1-\nu)}\beta\|J_*^{-1}\|\|v - v_*\| + \frac{u(2+u)}{(1-\mu)(1-\nu)}\kappa(J_*) + u$$

and

$$g = \frac{1+u}{(1-\mu)(1-\nu)}\|J_*^{-1}\|\psi(F, v, u, \bar{u}) + u\|v_*\|.$$

Proof. Assumption (4.13) guarantees that $J+E$ is nonsingular and therefore $\bar{v}$ in (4.10) is well defined. Starting with (4.10) we have

$$\begin{aligned}\bar{v} - v_* &= v - v_* - (J+E)^{-1}(r+e) + \epsilon \\ &= (I - (J+E)^{-1}J)(v - v_*) - (J+E)^{-1}(r - J(v - v_*) + e) + \epsilon,\end{aligned}$$

which upon taking norms gives us

$$\begin{aligned}\|\bar{v} - v_*\| \leq \|I - (J+E)^{-1}J\|\|v - v_*\| + \\ \|(J+E)^{-1}\|\left(\|r - J(v - v_*)\| + \|e\|\right) + \|\epsilon\|.\end{aligned} \tag{4.15}$$

We now bound the individual terms in (4.15). Starting with the first term we have

$$I - (J+E)^{-1}J = (J+E)^{-1}E = (I + J^{-1}E)^{-1}J^{-1}E$$

and it follows from (4.13) that

$$\|I - (J+E)^{-1}J\| \leq \frac{1}{1-\nu}\|J^{-1}E\|. \tag{4.16}$$

$(J+E)$ is nonsingular and can be bounded by

$$\|(J+E)^{-1}\| \leq \frac{\|J^{-1}\|}{1 - \|J^{-1}E\|}. \tag{4.17}$$

From the identity

$$J = J_*(I + J_*^{-1}(J - J_*)) \tag{4.18}$$

and using assumption (4.14) and the Lipschitz property of J we have

$$\|J_*^{-1}(J - J_*)\| \leq \beta\|J_*^{-1}\|\|v - v_*\| \leq \mu < 1 \tag{4.19}$$

and therefore J is nonsingular with inverse $J^{-1} = ((I + J_*^{-1}(J - J_*))^{-1}J_*^{-1}$ which can be bounded by

$$\|J^{-1}\| \leq \frac{\|J_*^{-1}\|}{1 - \|J_*^{-1}(J - J_*)\|} \leq \frac{1}{1 - \mu}\|J_*^{-1}\|. \tag{4.20}$$

Substituting this in (4.17) we have

$$\|(J + E)^{-1}\| \leq \frac{1}{(1 - \mu)(1 - \nu)}\|J_*^{-1}\|. \tag{4.21}$$

Using Lemma 4.2.1 we can show that

$$\|r - J(v - v_*)\|, \|r - J_*(v - v_*)\| \leq \frac{\beta}{2}\|v - v_*\|^2. \tag{4.22}$$

Using $\|e\| \leq u\|r\| + \psi(F, v, u, \bar{u})$ and rearranging (4.22) to give

$$\begin{aligned} \|r\| &\leq \|r - J(v - v_*)\| + \|J(v - v_*)\| \\ &\leq \frac{\beta}{2}\|v - v_*\|^2 + \|J_*\|\|v - v_*\| \end{aligned} \tag{4.23}$$

we get

$$\|e\| \leq u\left(\frac{\beta}{2}\|v - v_*\|^2 + \|J_*\|\|v - v_*\|\right) + \psi(F, v, u, \bar{u}). \tag{4.24}$$

Finally we have

$$\|\epsilon\| \leq u(\|v - v_*\| + \|v_*\| + \|d\|)$$

where $d = (J + E)^{-1}(r + e)$ and can be bounded using (4.21) and (4.23):

$$\begin{aligned} \|d\| &\leq \|(J + E)^{-1}\|(\|r\| + \|e\|) \\ &\leq \|(J + E)^{-1}\|((1 + u)\|r\| + \psi(F, v, u, \bar{u})) \\ &\leq \frac{1}{(1 - \mu)(1 - \nu)}\|J_*^{-1}\|\left((1 + u)\left(\frac{\beta}{2}\|v - v_*\| + \|J_*\|\right)\|v - v_*\|\right. \\ &\qquad \left. + \psi(F, v, u, \bar{u})\right). \end{aligned} \tag{4.25}$$

If we now take the inequalities (4.16), (4.21), (4.22), (4.24) and (4.25) and place them in (4.15) and collect like terms we get the G and g mentioned in the theorem. □

In exact arithmetic we have $u = \nu = \psi(F, v, u, \bar{u}) = E = 0$ and Theorem 4.2.2 becomes

$$\|\bar{v} - v_*\| \leq G_E \|v - v_*\|$$

where $G_E = \frac{1}{2}\beta\|J_*^{-1}\|\|v - v_*\|$, which is the quadratic local convergence theorem for Newton's method [14, Theorem 5.2.1] applied to a single step.

If μ, $\nu \leq \frac{1}{8}$ and the Jacobian J_* is not too ill-conditioned, say $u\kappa(J_*) \leq \frac{1}{8}$, then $G \leq \frac{1}{2}$. Therefore the error gets smaller unless $g \gtrsim \|v - v_*\|$. Hence, the best limiting accuracy is

$$\frac{g}{\|v_*\|} = \frac{1+u}{(1-\mu)(1-\nu)} \frac{\|J_*^{-1}\|}{\|v_*\|} \psi(F, v, u, \bar{u}) + u,$$

which depends upon how accurately the residual is computed. The rate of convergence, however, depends on the accuracy of the Jacobian and the stability of the solver, since G depends largely on the size of E. Also G is independent of $\bar{u}$ and therefore the rate of convergence is independent of the precision used to calculate the residual.

Corollary 4.2.3 *Let $F : \mathbb{R}^n \to \mathbb{R}^n$, $v_* \in \mathbb{R}^n$ such that $F(v_*) = 0$ and the Jacobian at the zero $J_* = J(v_*)$ be nonsingular and satisfies*

$$u\kappa(J_*) \leq \frac{1}{8}. \tag{4.26}$$

Assume also that

$$u\|J(\widehat{v}_k)^{-1}\|\phi(F, \widehat{v}_k, n, u) \leq \frac{1}{8} \quad \forall k. \tag{4.27}$$

Then for all v_0 such that

$$\beta\|J_*^{-1}\|\|v_0 - v_*\| \leq \frac{1}{8}. \tag{4.28}$$

Newton's method produces a sequence $\{\widehat{v}_k\}$ *in floating point arithmetic that decreases until the first* k *where*

$$\frac{\|\widehat{v}_{k+1} - v_*\|}{\|v_*\|} \approx \frac{\|J_*^{-1}\|}{\|v_*\|}\psi(F, v, u, \bar{u}) + u. \tag{4.29}$$

Proof. The assumptions (4.26), (4.27) and (4.28) mean that Theorem 4.2.2 can be applied to the first step. They also imply that $G < 1$ and therefore the error contracts unless (4.29) holds. For subsequent iterations we can apply Theorem 4.2.2 with v_0 replaced by $\widehat{v}_k$ in (4.14). □

4.2.2 Residual

We now consider the effect of Newton's method on the residual.

Theorem 4.2.4 *[57, Theorem 2.4] Let* $F\colon \mathbb{R}^n \to \mathbb{R}^n$, $v_* \in \mathbb{R}^n$ *such that* $F(v_*) = 0$ *and assume that the Jacobian at the zero* $J_* = J(v_*)$ *is nonsingular and*

$$\beta\|J_*^{-1}\|\|v - v_*\| \le \mu < 1, \tag{4.30}$$

$$u\|J^{-1}\|\phi(F, v, n, u) \le \nu < 1. \tag{4.31}$$

Let

$$\tau = \beta g\|J_*^{-1}\|,$$

where g *is defined in Theorem* 4.2.2. *Then*

$$\|F(\bar{v})\| \le H\|F(v)\| + h,$$

where

$$H = c_0\left(\mu + \tau + u\kappa(J_*)\right)$$

and

$$h = c_1\left(\mu + \tau + u\kappa(J_*)\right)\psi(F, v, u, \bar{u}) + c_2\left(\mu + \tau + 1\right)u\|J\|\|v\|,$$

where $c_i, i = 0\colon 2$ *are constants of order 1.*

Proof. From (4.31) we have

$$\|J^{-1}E\| \leq u\|J^{-1}\|\phi(F, v, n, u) \leq \nu < 1 \tag{4.32}$$

and thus $\bar{v}$ is well defined. We define $\bar{r} = F(\bar{v})$ and $w = \bar{r} - r - J(\bar{v} - v)$. From (4.10) and (4.11) we have $\bar{v} - v = -d + \epsilon$ and $Jd = r + e - Ed$ which yields

$$\begin{aligned} \bar{r} &= r + J(-d + \epsilon) + w, \\ &= -e + Ed + J\epsilon + w. \end{aligned}$$

By taking norms we get

$$\begin{aligned} \|\bar{r}\| &\leq \|e\| + \|E\|\|d\| + \|J\|\|\epsilon\| + \|w\| \\ &\leq u\|r\| + \psi(F, v, u, \bar{u}) + u\|d\|(\phi(F, v, n, u) + \|J\|) \\ &\qquad + u\|J\|\|v\| + \|w\|. \end{aligned} \tag{4.33}$$

Using (4.18) and (4.19) we have

$$\|J\| \leq (1 + \mu)\|J_*\|. \tag{4.34}$$

From (4.25) and (4.32) we have

$$\begin{aligned} \|d\| &\leq \|(J + E)^{-1}\| \left(\|r\| + \|e\|\right) \\ &\leq \frac{1}{1 - \nu}\|J^{-1}\| \left((1 + u)\|r\| + \psi(F, v, u, \bar{u})\right). \end{aligned} \tag{4.35}$$

Combining (4.20), (4.34) and (4.35) gives

$$u\|d\|(\phi(F, v, n, u) + \|J\|) \leq \frac{k(1 + u)}{1 - \nu}\|r\| + \frac{k}{1 - \nu}\psi(F, v, u, \bar{u}) \tag{4.36}$$

where

$$k = u\|J^{-1}\|\phi(F, v, n, u)\frac{1 + \mu}{1 - \mu}u\kappa(J_*).$$

Lemma 4.2.1 shows that

$$\|w\| \leq \frac{\beta}{2}\|\bar{v} - v\|^2$$

where we bound $\|\bar{v} - v\|$ in two different ways. Firstly using (4.10) and (4.11) we have

$$\|\bar{v} - v\| \leq (1+u)\|d\| + u\|v\|$$

which using (4.34) and (4.35) gives

$$\|\bar{v} - v\| \leq \|J_*^{-1}\| \left(\frac{(1+u)^2}{(1-\mu)(1-\nu)} \|r\| + \frac{(1+u)}{(1-\mu)(1-\nu)} \psi(F, v, u, \bar{u}) \right) + u\|v\|. \tag{4.37}$$

Secondly, using the triangle inequality and Theorem 4.2.2 we get

$$\|\bar{v} - v\| \leq (G+1)\|v - v_*\| + g. \tag{4.38}$$

Combining (4.37) and (4.38) we get

$$\begin{aligned} \|w\| \leq & \frac{(1+u)^2(G+1)}{2(1-\mu)(1-\nu)} \beta \|J_*^{-1}\| \|v - v_*\| \|r\| + \frac{(1+u)^2}{2(1-\mu)(1-\nu)} \beta g \|J_*^{-1}\| \|r\| \\ & + \frac{(1+u)(G+1)}{2(1-\mu)(1-\nu)} \beta \|J_*^{-1}\| \|v - v_*\| \psi(F, v, u, \bar{u}) \\ & + \frac{(1+u)}{2(1-\mu)(1-\nu)} \beta g \|J_*^{-1}\| \psi(F, v, u, \bar{u}) \\ & + \frac{u(G+1)}{2(1-\mu)} \beta \|J_*^{-1}\| \|v - v_*\| \|J\| \|v\| + \frac{\beta}{2} g u \|v\|. \end{aligned} \tag{4.39}$$

Substituting (4.36) and (4.39) into (4.33) yields the values H and h in the theorem. □

Corollary 4.2.5 *Let $F : \mathbb{R}^n \to \mathbb{R}^n$, $v_* \in \mathbb{R}^n$ such that $F(v_*) = 0$ and the Jacobian at the zero $J_* = J(v_*)$ be nonsingular and satisfies*

$$u\kappa(J_*) \leq \frac{1}{8}.$$

Assume also that

$$u\|J(\widehat{v}_k)^{-1}\| \phi(F, \widehat{v}_k, n, u) \leq \frac{1}{8} \quad \forall k$$

and that the limiting accuracy $g \approx \|J_*^{-1}\| \psi(F, \widehat{v}_k, u, \bar{u}) + u\|v_*\|$ satisfies

$$\beta g \|J_*^{-1}\| \leq \frac{1}{8}.$$

Then for all v_0 such that

$$\beta\|J_*^{-1}\|\|v_0 - v_*\| \leq \frac{1}{8}$$

Newton's method produces a sequence $\{F(\widehat{v}_k)\}$ in floating point arithmetic that decreases until the first k where

$$\|F(\widehat{v}_{k+1})\| \approx \psi(F, \widehat{v}_k, u, \bar{u}) + u\|J(\widehat{v}_k)\|\|\widehat{v}_k\|. \tag{4.40}$$

We note that if we have a zero of F in floating point arithmetic $\widehat{v}_* = fl(v_*) = v_* + \Delta v_*$ where $\|\Delta v_*\| \leq u\|v_*\|$, then using Lemma 4.2.1 we have

$$F(\widehat{v}_*) = F(v_* + \Delta v_*) = J(v_*)\Delta v_* + \theta, \quad \text{where } \|\theta\| \leq \frac{\beta}{2}\|\widehat{v}_* - v_*\|^2.$$

Therefore

$$\|F(\widehat{v}_*)\| \leq u\|J(v_*)\|\|v_*\| + \frac{\beta}{2}u^2\|v_*\|^2$$

is the best bound we can hope for the norm of the residual. Notice that the first term is the same as that in the limiting residual in Corollary 4.2.5.

4.2.3 Applications to the Generalized Eigenvalue Problem

We now apply these results to the generalized eigenvalue problem. The Jacobian $J(v)$ in (4.3) is Lipschitz continuous in $\mathbb{R}^{n+1}$ with constant $\beta = 2\|B\|$ and we can calculate the residual such that

$$\psi(F, v, u, \bar{u}) = \bar{\gamma}_n \left(\|A\| + |\lambda|\|B\|\right)\|x\|.$$

We can also say that λ_* needs to be a simple eigenvalue, by Lemma 4.1.1, such that J is not too ill-conditioned to apply the results of Corollaries 4.2.3 and 4.2.5. We assume that the linear system solver is not too unstable and that (x_0, λ_0) is sufficiently close to the zero (x_*, λ_*). Then Newton's method for the generalized

eigenvalue problem is well defined and the limiting forward error can be bounded by

$$\begin{aligned}\frac{\|(\widehat{x}_k^T, \widehat{\lambda}_k) - (\widehat{x}_*^T, \widehat{\lambda}_*)\|}{\|(\widehat{x}_*^T, \widehat{\lambda}_*)\|} &\lesssim \bar{\gamma}_n \frac{\|J_*^{-1}\|}{\|v_*\|} \left(\|A\| + |\lambda_*|\|B\|\right) \|x_*\| + u \\ &\leq \bar{\gamma}_n \|J_*^{-1}\| \max\left(\|A\|, \|B\|\right) \frac{1 + |\lambda_*|}{\max(1, |\lambda_*|)} + u \\ &\leq 2\bar{\gamma}_n \|J_*^{-1}\| \max\left(\|A\|, \|B\|\right) + u\end{aligned}$$

We have $\|J_*\| \geq \max(\|A\|, \|B\|)$ and from (4.26), $u\kappa(J_*) < 1$. If we use double precision in the calculation of the residual, $\bar{u} = u^2$ we have $\bar{\gamma}_n \approx nu^2$ and therefore the limiting forward error would then be

$$\frac{\|(\widehat{x}_k^T, \widehat{\lambda}_k) - (\widehat{x}_*^T, \widehat{\lambda}_*)\|}{\|(\widehat{x}_*^T, \widehat{\lambda}_*)\|} \lesssim \gamma_n.$$

We now consider the limiting backward error of the Newton iteration. The normwise backward error of an approximate eigenpair $(\widetilde{x}, \widetilde{\lambda})$ of (A, B) is defined by

$$\eta(\widetilde{x}, \widetilde{\lambda}) = \min\left\{\epsilon : (A + \Delta A)\widetilde{x} = \widetilde{\lambda}(B + \Delta B)\widetilde{x}, \quad \|\Delta A\|_{\alpha,\beta} \leq \epsilon\|A\|_{\alpha,\beta}, \right. \\ \left. \|\Delta B\|_{\alpha,\beta} \leq \epsilon\|B\|_{\alpha,\beta}\right\}. \quad (4.41)$$

The following theorem gives an explicit expression for $\eta(\widetilde{x}, \widetilde{\lambda})$ [24], [29].

Theorem 4.2.6 *The normwise backward error $\eta(\widetilde{x}, \widetilde{\lambda})$ is given by*

$$\eta(\widetilde{x}, \widetilde{\lambda}) = \frac{\|r\|_\beta}{(|\widetilde{\lambda}|\,\|B\|_{\alpha,\beta} + \|A\|_{\alpha,\beta})\|\widetilde{x}\|_\alpha}, \quad (4.42)$$

where $r = \widetilde{\lambda}B\widetilde{x} - A\widetilde{x}$.

Proof. It is straightforward to show that (4.42) is a lower bound of (4.41). This bound is attained for the perturbations

$$\Delta A = \frac{\|A\|_{\alpha,\beta}}{(|\widetilde{\lambda}|\,\|B\|_{\alpha,\beta} + \|A\|_{\alpha,\beta})} rz^*, \quad \Delta B = -\operatorname{sign}(\widetilde{\lambda}) \frac{\|B\|_{\alpha,\beta}}{(|\widetilde{\lambda}|\,\|B\|_{\alpha,\beta} + \|A\|_{\alpha,\beta})} rz^*,$$

where for complex α,

$$\operatorname{sign}(\alpha) = \begin{cases} \frac{\bar{\alpha}}{|\alpha|} & \alpha \neq 0, \\ 0 & \alpha = 0. \end{cases}$$

and z is a vector dual to $\widetilde{x}$ with respect to the α norm. □

For Hermitian A and B, we denote by $\eta^H(\widetilde{x}, \widetilde{\lambda})$ the backward error (4.41) with the additional constraint that the perturbations ΔA and ΔB are Hermitian. Clearly $\eta^H(\widetilde{x}, \widetilde{\lambda}) \geq \eta(\widetilde{x}, \widetilde{\lambda})$. However the following theorem shows that when $\widetilde{\lambda}$ is real, requiring the backward perturbations to be Hermitian has no effect on the backward error for the 2-norm.

Theorem 4.2.7 *[29] If A and B are Hermitian and $\widetilde{\lambda}$ is real then for the 2-norm, $\eta_2^H(\widetilde{x}, \widetilde{\lambda}) = \eta_2(\widetilde{x}, \widetilde{\lambda})$.*

Proof. We define the Hermitian matrix H such that

$$H\widetilde{x} := (\Delta A - \widetilde{\lambda}\Delta B)\widetilde{x} = r,$$

where $r = \widetilde{\lambda}B\widetilde{x} - A\widetilde{x}$. We take $H := (\|r\|_2/\|\widetilde{x}\|_2)P$, where P is a suitably chosen Householder matrix (if r is a multiple of x we choose $P = I_n$). We can choose such a P provided $\widetilde{x}^* r$ is real which occurs when $\widetilde{\lambda}$ is real. To satisfy $H = \Delta A - \widetilde{\lambda}\Delta B$ with Hermitian ΔA and ΔB we define

$$\Delta A = \frac{\|A\|_2}{(|\widetilde{\lambda}|\,\|B\|_2 + \|A\|_2)} H, \quad \Delta B = -\operatorname{sign}(\widetilde{\lambda}) \frac{\|B\|_2}{(|\widetilde{\lambda}|\,\|B\|_2 + \|A\|_2)} H. \tag{4.43}$$

Using (4.42) we have

$$\|H\|_2 = \|r\|_2/\|\widetilde{x}\|_2 = \eta_2(\widetilde{x}, \widetilde{\lambda})(|\widetilde{\lambda}|\,\|B\|_2 + \|A\|_2)$$

Substituting this into (4.43) we see that $\eta_2^H(\widetilde{x}, \widetilde{\lambda}) \leq \eta_2(\widetilde{x}, \widetilde{\lambda})$. Since $\eta_2^H(\widetilde{x}, \widetilde{\lambda}) \geq \eta_2(\widetilde{x}, \widetilde{\lambda})$ equality must hold. □

Hence, for the symmetric definite generalized eigenvalue problem it is appropriate to use the general definition (4.41) and the formula (4.42).

Notice that the normwise backward error (4.42) is just a scaled residual. If we consider the limiting residual (4.40) then we can bound the terms by

$$\|\widehat{v}_i\|_\infty \leq 1 + |\widehat{\lambda}_i|, \quad \|J(\widehat{v}_i)\|_\infty \leq (3 + |\widehat{\lambda}_i|)\max(\|A\|_\infty, \|B\|_\infty).$$

If we also bound $(\|A\|_\infty + |\widehat{\lambda}_i|\|B\|_\infty)\|\widehat{x}_i\|_\infty \geq \min(\|A\|, \|B\|)(1 + |\widehat{\lambda}_i|)$ then we can show, using (4.42), that the limiting backward error is bounded by

$$\eta_\infty(\widehat{x}, \widehat{\lambda}) \leq \tilde{\gamma}_n + u(3 + |\lambda|)\max\left(\frac{\|A\|_\infty}{\|B\|_\infty}, \frac{\|B\|_\infty}{\|A\|_\infty}\right). \tag{4.44}$$

This backward error bound is small if λ is of order 1 and the problem is well balanced, that is, $\|A\|_\infty \approx \|B\|_\infty$. If the problem is not well balanced, we can change the GEP to make it so. We can scale the GEP to $(\alpha A)x = (\alpha\lambda)Bx$, where $\alpha = \|B\|_\infty/\|A\|_\infty$ and the backward error now depends on the size of $\bar{\lambda} = \alpha\lambda$. If $|\bar{\lambda}| \leq 1$ a small backward error is ensured, while for $|\bar{\lambda}| \geq 1$ we can consider the problem $Bx = \bar{\mu}\bar{A}x$, for which $|\bar{\mu}| \leq 1$. Practical experience shows that it is not necessary to scale or to reverse the problem—a backward error of order u is obtained as long as the starting vector is good enough for Newton's method to converge.

4.3 Summary

We have shown how to repair instability in the computed eigenpairs when the Cholesky method produces unstable solutions. Using iterative refinement, a backward error of order u is obtained for Algorithms 4.1.2 and 4.1.3, as long as the starting vector is good enough for Newton's method to converge. When the starting vector is not close enough, iterative refinement is not guaranteed to converge and may even converge to the wrong eigenpair. Large values of $\phi(F, \widehat{v}_k, n, u)$, $\kappa(J_*)$ and $\|v_0 - v_*\|$ can prevent some of the conditions of Corollaries 4.2.3 and 4.2.5 from being satisfied. If this is the case, Newton's method in floating point arithmetic is not guaranteed to decrease until it reaches the limiting

residual (4.40). As the instability of the linear system solver, $\phi(F, \widehat{v}_k, n, u)$, is one of the conditions of Corollaries 4.2.3 and 4.2.5, Algorithm 4.1.3 is less likely to converge than Algorithm 4.1.2. Large error in the computed eigenpairs can also cause Newton's method not to converge. Our experience shows that extreme examples, involving ill-conditioned B, that are specific to the Cholesky–Jacobi or the Cholesky–QR method are needed for this to happen (see Example 5 in Chapter 6). When Newton's method fails to converge to the correct eigenpair a warning can alert the user that the method has not converged or if any duplicate eigenpairs are found.

Chapter 5

Numerically Stable Generation of Correlation Matrices

An important class of symmetric positive semidefinite matrices is those with unit diagonal. They arise in statistics as matrices of correlation coefficients and are known as correlation matrices [33], [51, p. 24]. They also play an important role in numerical analysis, because of the approximate optimality property (2.14). This property accounts for the appearance of correlation matrices in several contexts in numerical analysis, three of which we briefly explain.

Cholesky factorization. Standard error analysis shows that when Cholesky factorization is used to solve a symmetric positive definite linear system $Ax = b$ in floating point arithmetic, the relative error $\|x - \widehat{x}\|_2/\|x\|_2$ of the computed solution $\widehat{x}$ is bounded by a multiple of $\kappa_2(A)u$, where u is the unit roundoff. A more refined analysis [11], [31, Sec. 10.1], [64] shows that $\|D(x - \widehat{x})\|_2/\|Dx\|_2$ is bounded by a multiple of $\kappa_2(H)u$, with D and H as defined in (2.13). If A is "artificially ill conditioned", in the sense that H is well conditioned and D has diagonal elements of widely varying magnitude, then the latter bound can guarantee much better relative accuracy in $\widehat{x}$ than the original one. The refined analysis also provides a condition for Cholesky factorization to succeed in floating

point arithmetic (that is, for no square roots of negative quantities or divisions by zero to occur) involving an upper bound on $\kappa_2(H)u$ rather than the more usual $\kappa_2(A)u$.

Preconditioning. The simplest preconditioner for iterative methods for solving a symmetric positive definite linear system $Ax = b$ is the Jacobi preconditioner [3], [27], which uses D in (2.13) to transform to a linear system involving the correlation matrix H. The motivation is the result (2.14), since one of the aims of preconditioning is to reduce the condition number $\kappa_2(A)$. For some matrices arising in PDE applications, D in (2.13) is an optimal scaling: Forsythe and Straus [22] show that $\kappa_2(H)$ is minimal if A is symmetric positive definite with property A (that is, there exists a permutation matrix P such that PAP^T can be expressed as a block 2×2 matrix whose $(1,1)$ and $(2,2)$ blocks are diagonal). Thus, for example, any symmetric positive definite block tridiagonal matrix whose diagonal blocks are identities is optimally scaled. For a summary of more general scaling results of this type see [27, Sec. 10.5].

Jacobi methods. In section 2.1 we noted that when Jacobi's method is applied to a symmetric positive definite matrix A, using a suitable stopping criterion, the computed eigenvalues satisfy relative error bounds proportional to $\kappa_2(H)u$ rather than the quantity $\kappa_2(A)u$ that would be expected for a normwise backward stable method.

For testing theory and algorithms in statistics and these numerical analysis applications, and for simulations in signal processing [33], it is desirable to be able to generate correlation matrices with a specified eigenvalue distribution or 2-norm condition number. One approach, used by Demmel and Veselić [10], is to generate a random symmetric positive definite matrix with specified eigenvalues by any existing technique and then scale it to produce unit diagonal. However, the act of scaling can change the condition number by an arbitrary amount. Certain

special matrices are available with the desired properties. For example, the Kac–Murdock–Szego Toeplitz matrix with (i,j) element $\rho^{|i-j|}$ has unit diagonal and is positive semidefinite if $|\rho| \leq 1$. Choosing ρ to achieve a desired condition number is possible but nontrivial, as the eigenvalues are known only in terms of the roots of a certain nonlinear equation [59]. For $C \in \mathbb{R}^{m\times n}$ with $\|C\|_2 < 1$, the matrix

$$A = \begin{bmatrix} I_m & C \\ C^T & I_n \end{bmatrix}$$

(which has property A and so is optimally scaled) is easily shown to be positive definite with $\kappa_2(A) = (1+\|C\|_2)/(1-\|C\|_2)$, but A has many zero elements so is not necessarily a good test matrix for the Cholesky or Jacobi methods.

We begin, in the next section, by briefly reviewing some necessary background theory.

5.1 Theory

Constraints on the spectrum of a correlation matrix can be deduced from the following standard majorization result [34, Thms. 4.3.26, 4.3.32] (see [38] for a recent geometric proof).

Theorem 5.1.1 *Let $A \in \mathbb{R}^{n\times n}$ be symmetric. A necessary and sufficient condition for A to have eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ and diagonal elements $\alpha_1 \leq \alpha_2 \leq \cdots \leq \alpha_n$ (in any order along the diagonal) is that*

$$\sum_{i=1}^{k} \lambda_i \leq \sum_{i=1}^{k} \alpha_i, \qquad k = 1\colon n, \tag{5.1}$$

with equality for $k = n$.

Note that the requirement of equality in (5.1) for $k = n$ follows trivially from the fact that $\mathrm{trace}(A) = \sum_i \alpha_i = \sum_i \lambda_i$. For a correlation matrix, $\alpha_i \equiv 1$ and

$\lambda_i \geq 0$, and the theorem implies that any set of nonnegative eigenvalues summing to n is possible.

A natural question, which does not seem to have been considered before, is whether it is possible to generate a banded correlation matrix with specified nonnegative spectrum summing to n. Recall that A has bandwidth p if $a_{ij} = 0$ for $|i-j| > p$. The next result gives a negative answer to this question for the narrow bandwidth case that is of most practical interest.

Theorem 5.1.2 *Not every set of nonnegative scalars $\lambda_1, \ldots, \lambda_n$ summing to n can be the eigenvalues of a correlation matrix of bandwidth p if $p < (n-1)/2$.*

Proof. Let $A \in \mathbb{R}^{n\times n}$ be a correlation matrix of bandwidth p. Because A is positive semidefinite with unit diagonal, its off-diagonal elements are bounded in magnitude by 1. The condition on p implies that every row of A has less than $n-1$ nonzero elements. Gershgorin's theorem therefore implies that every eigenvalue λ of A satisfies $\lambda < n$. Thus $\lambda_1 = n$ and $\lambda_2 = \cdots = \lambda_n = 0$ is an example of a set of eigenvalues that is not achievable. □

Our interest is now in transforming a given symmetric positive semidefinite matrix with trace n into a correlation matrix, while preserving the eigenvalues. That such a transformation is possible follows from Theorem 5.1.1 and is also a corollary of the following more general result that applies to nonsymmetric matrices. This result has a long history in which it appears in various forms; see, in particular, [20], [40], [58]. We give a proof, since it suggests an algorithm.

Theorem 5.1.3 *If $A \in \mathbb{C}^{n\times n}$ has trace n then there exists a unitary U such that U^*AU has unit diagonal. If A is real or Hermitian then U can be taken to be real.*

Proof. Since the field of values $F(A) = \{ x^*Ax : x^*x = 1,\ x \in \mathbb{C}^n \}$ is a convex set containing the eigenvalues of A,

$$1 = \mathrm{trace}(A)/n = (\lambda_1 + \cdots + \lambda_n)/n \in F(A).$$

Hence there exists $x \in \mathbb{C}^n$ such that

$$x^*Ax = 1, \qquad x^*x = 1. \tag{5.2}$$

Choose $\widetilde{Q}$ so that $[x\ \widetilde{Q}]$ is unitary and form

$$\begin{bmatrix} x^* \\ \widetilde{Q}^* \end{bmatrix} A \begin{bmatrix} x & \widetilde{Q} \end{bmatrix} = \begin{bmatrix} 1 & x^*A\widetilde{Q} \\ \widetilde{Q}^*Ax & \widetilde{Q}^*A\widetilde{Q} \end{bmatrix}.$$

Taking the trace, we find that the matrix $\widetilde{Q}^*A\widetilde{Q} \in \mathbb{C}^{(n-1)\times(n-1)}$ has trace $n-1$ and so the first part of the result follows by induction, since the case $n = 1$ is trivial. If A is real or Hermitian we have to show that $x = u + iv$ in (5.2) can be taken to be real. We can assume that $u \neq 0$, because if $u = 0$ then $x/i = v$ is a real vector satisfying (5.2). From the real parts of (5.2) we have

$$u^*Au + v^*Av = 1, \qquad u^*u + v^*v = 1.$$

Subtracting these two equations gives

$$u^*(A - I)u = -v^*(A - I)v. \tag{5.3}$$

Defining the real vector $x(t) = u + tv$, we now try to find t such that $x(t)^*Ax(t) = x(t)^*x(t)$. This equation can be written

$$u^*Au + t(u^*Av + v^*Au) + t^2v^*Av = u^*u + t(u^*v + v^*u) + t^2v^*v,$$

or, using (5.3),

$$at^2 + bt - a = 0,$$

where $a = v^*(A - I)v$ and $b = u^*(A - I)v + v^*(A - I)u$. When A is real or Hermitian both a and b are real and we have the real solutions

$$t = \frac{-b \pm \sqrt{b^2 + 4a^2}}{2a},$$

or $t = 0$ if $a = 0$. Since $u \neq 0$, $x(t) = u + tv$ is nonzero for one of the solutions t and so $x(t)/\|x(t)\|_2$ is a real vector satisfying (5.2). □

An algorithm based on the construction in this proof is given by Marsaglia and Olkin [40]. It rests on the ability to construct a vector x satisfying (5.2). Finding x is not straightforward, and an iterative technique is used in [40]. Unfortunately, no upper bound on the number of iterations is available, although the average behaviour is stated to be satisfactory when A is a diagonal matrix of random eigenvalues. We therefore turn our attention in the next section to a method of Bendel and Mickey, which is easy to implement and has a known and satisfactory computational cost.

5.2 The Bendel–Mickey Algorithm

Given a symmetric positive semidefinite matrix $A \in \mathbb{R}^{n\times n}$ with $\operatorname{trace}(A) = n$ the Bendel–Mickey algorithm [4] transforms A into a correlation matrix in $n-1$ steps, each of which consists of applying an orthogonal similarity transformation in an (i,j) coordinate plane to introduce a 1 in the (i,i) position. (This method is outlined by Golub and Van Loan in Problems[1] 8.4.1 and 8.4.2 of [26].) It suffices to describe the first stage. If A does not have unit diagonal then, since $\operatorname{trace}(A) = n$, there exist i and j with $i < j$ such that $a_{ii} < 1 < a_{jj}$ or $a_{ii} > 1 > a_{jj}$. We apply a Givens rotation to obtain

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a_{ii} & a_{ij} \\ a_{ij} & a_{jj} \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix} = \begin{bmatrix} 1 & \tilde{a}_{ij} \\ \tilde{a}_{ij} & \tilde{a}_{jj} \end{bmatrix}, \qquad c^2 + s^2 = 1. \tag{5.4}$$

Equating (1,1) elements yields the equation

$$c^2 a_{ii} - 2sca_{ij} + s^2 a_{jj} = 1. \tag{5.5}$$

In [4], and in the Fortran implementation in [39], this equation is solved by converting it into a quadratic in $\cos 2\theta$ (where $c = \cos\theta$). However, it is better

[1] In the 1989 second edition of [26] these problems are numbered 8.5.3 and 8.5.4.

in terms of efficiency and numerical stability to express (5.5) as a quadratic in $t = s/c$,

$$(a_{jj} - 1)t^2 - 2ta_{ij} + a_{ii} - 1 = 0, \tag{5.6}$$

from which

$$t = \frac{a_{ij} \pm \sqrt{a_{ij}^2 - (a_{ii} - 1)(a_{jj} - 1)}}{a_{jj} - 1}. \tag{5.7}$$

Note that the choice of i and j ensures that $(a_{ii} - 1)(a_{jj} - 1) < 0$ and hence that t is real; moreover, the argument of the square root is evaluated accurately as there is no damaging cancellation. We take the sign to be positive in (5.7) to avoid cancellation. If necessary, the other root can be obtained from the fact that the product of the roots is $(a_{ii} - 1)/(a_{jj} - 1)$. Then we recover

$$c = \frac{1}{\sqrt{1 + t^2}}, \qquad s = ct. \tag{5.8}$$

We could, alternatively, use a Householder reflection, as suggested by Marsaglia and Olkin [40]. However, any 2×2 Householder reflection has the form

$$\begin{bmatrix} -c & s \\ s & c \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} c & -s \\ s & c \end{bmatrix} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix},$$

from which it is easily seen that applying a Householder reflection is equivalent, up to signs, to applying a Givens rotation.

If we start with a diagonal A this approach does not necessarily produce a full matrix (that is, some of the correlations may be zero). To ensure that a full matrix is produced and to add randomness we begin by applying a random orthogonal similarity transformation, for which the natural distribution is the Haar distribution [54].

Algorithm 5.2.1 (Bendel and Mickey) *Given nonnegative scalars $\lambda_1, \ldots, \lambda_n$ summing to n, this algorithm produces a random correlation matrix having eigenvalues $\lambda_1, \ldots, \lambda_n$.*

1. Form a random orthogonal matrix $U \in \mathbb{R}^{n\times n}$ from the Haar distribution.
   Let $A = U\,\mathrm{diag}(\lambda_i)U^T$.
2. while some $a_{ii} \neq 1$

   Find indices i and j with $i < j$ so that $a_{ii} < 1 < a_{jj}$ or $a_{ii} > 1 > a_{jj}$.
   Apply an orthogonal similarity transformation comprising
   a Givens rotation in the (i, j) plane to set $a_{ii} = 1$,
   as described by (5.4), (5.7) and (5.8).

   end

The matrix U in Algorithm 5.2.1 can be formed by the method of Stewart [54] (see also [31, Section 26.3]), which constructs it as a product of $n-1$ random Householder transformations and a diagonal matrix of ± 1s.

Algorithm 5.2.1 has several attractive features. First, it has a high degree of randomness: the first stage produces a matrix with random eigenvectors from a standard distribution, and the indices in the loop and the root of (5.6) can be selected randomly, too, if desired. Second, the algorithm is of known and reasonable cost. The initial formation of A requires $7n^3/3$ flops using Stewart's method. The while loop requires at most $n-1$ iterations so the second stage costs at most $12n^2$ flops, which is negligible compared with the first stage. Finally, software for the first stage is readily available, in MATLAB as `gallery('randsvd',...)`, and in Fortran 77 in directory `lapack/testing/matgen` of the LAPACK distribution (see [13] for documentation), so the algorithm is very easy to implement. Algorithm 5.2.1 is implemented in MATLAB 6 as `gallery('randcorr',...)`.

The statistical distribution of the matrices produced by Algorithm 5.2.1 is not well understood; see Holmes [33] for details of what is known.

Finally, we note that an algorithm similar to Algorithm 5.2.1 for constructing a symmetric matrix with given eigenvalues and diagonal elements satisfying the

majorization condition (5.1) is given by Chan and Li [6], with a Fortran implementation in [7]. As Ikramov [35] has noted, the same algorithm is also given, without reference to [6], by Zha and Zhang [65].

5.3 Error Analysis and Implementation Issues

Now we consider the behaviour of Algorithm 5.2.1 in floating point arithmetic and compare it with the existing version of the Bendel and Mickey algorithm in [4], [39].

Lemma 5.3.1 *Let $\widehat{A}_i$ denote the computed matrix at the end of stage i in Algorithm* 5.2.1*, for $i = 1, 2$. Then*

$$\widehat{A}_1 = Q_1^T(\operatorname{diag}(\lambda_i) + \Delta A_1)Q_1, \qquad \|\Delta A_1\|_F \le \tilde{\gamma}_{n^2}\|\operatorname{diag}(\lambda_i)\|_F,$$
$$\widehat{A}_2 = Q_2^T(\widehat{A}_1 + \Delta A_2)Q_2, \qquad \|\Delta A_2\|_F \le \tilde{\gamma}_n\|\widehat{A}_1\|_F,$$

where Q_1 and Q_2 are orthogonal. Consequently, the output of the algorithm is a matrix $\widehat{A}$ satisfying

$$\widehat{A} = Q^T(\operatorname{diag}(\lambda_i) + \Delta)Q, \qquad \|\Delta\|_F \le \tilde{\gamma}_{n^2}\|\operatorname{diag}(\lambda_i)\|_F,$$

where Q is orthogonal.

Proof. The result for stage 1 follows from standard error analysis of Householder transformations [31, Sec. 18.3]. Similarly, for stage 2 we can adapt standard error analysis of Givens transformations [31, Sec. 18.5]. □

The lemma shows that most of the error in Algorithm 5.2.1 comes, as does most of the work, from stage 1. It follows from standard perturbation theory that the eigenvalues of $\widehat{A}$ differ from the desired eigenvalues λ_i by absolute amounts at most $\|\Delta\|_F \le \sqrt{n}\tilde{\gamma}_{n^2}\max_i \lambda_i$. If $\max(\lambda_i)/\min(\lambda_i) > n^{-1/2}\tilde{\gamma}_{n^2}^{-1}$ then $\widehat{A}$ can be indefinite and so fail to be a correlation matrix. In this situation it is preferable,

if the application allows it, to generate A in factored form, as described in the next section.

The computed $\widehat{A}$ from Algorithm 5.2.1 will not usually have diagonal elements exactly equal to 1, because of rounding errors. The questions arises of whether it is safe to set these elements to 1. Consider the general step in stage 2, which is designed to set a_{ii} to unity as described by (5.4), and suppose that $a_{ii} < 1 < a_{jj}$. Since A is positive definite,

$$a_{ij} \leq \sqrt{a_{ii}a_{jj}} < \sqrt{a_{jj}}.$$

The (i,i) element is overwritten by the result of evaluating the formula

$$x = c^2 a_{ii} - 2sca_{ij} + s^2 a_{jj},$$

or some simple rearrangement of it, where the computed c and s that are used have relative errors of order $\tilde{\gamma}_1$. Standard error analysis yields, for any order of evaluation,

$$\begin{aligned} |x - \widehat{x}| &\leq \tilde{\gamma}_1(c^2|a_{ii}| + 2|sc||a_{ij}| + s^2|a_{jj}|) \\ &\leq \tilde{\gamma}_1(|a_{ii}| + |a_{ij}| + |a_{jj}|) \\ &\leq \tilde{\gamma}_1(1 + \sqrt{|a_{jj}|} + |a_{jj}|) \\ &\leq \tilde{\gamma}_1(1 + \sqrt{n} + n) \\ &\leq \tilde{\gamma}_n. \end{aligned}$$

Since $x = 1$ we conclude that replacing $\widehat{x}$ by 1 corresponds to a backward perturbation in A of order $\tilde{\gamma}_n$, which has no effect on the bounds in Lemma 5.3.1.

To summarize, Algorithm 5.2.1 has essentially perfect backward stability and we can explicitly set the diagonal elements to unity without affecting the stability.

Now we consider the implementation of the Bendel–Mickey algorithm in [4], [39]. Here, the Givens rotation in (5.4) is computed by

$$\alpha = a_{ij}^2 + \tfrac{1}{4}(a_{ii} - a_{jj})^2, \qquad \beta = \tfrac{1}{2}(a_{ii} + a_{jj} - 2)(a_{ii} - a_{jj}),$$

$$\gamma = \tfrac{1}{4}(a_{ii} + a_{jj} - 2)^2 - a_{ij}^2, \qquad \delta = \sqrt{\beta^2 - 4\alpha\gamma},$$
$$\theta = \tfrac{1}{2}\cos^{-1}\left(\tfrac{\delta-\beta}{2\alpha}\right), \qquad c = \cos(\theta), \qquad s = \sin(\theta).$$

It is clear that these formulae can involve serious cancellation and are not a stable way of computing θ. The computed Givens rotation will always be orthogonal to working precision, but when applied to A it will not necessarily accurately set a_{ii} to 1. In the Fortran implementation in [39] this weakness is recognized in so far as the routine checks how close the computed diagonal is to 1 and flags an error if a user-specified tolerance for the difference is exceeded.

We have experimented with the NAG library routine G̃05GBF (Mark 18) [45], which is based on the code from [39]. We called the code from MATLAB using a Mex interface and applied direct search optimization routines from the Test Matrix Toolbox [30]. We readily found examples where the computed diagonal is very far from 1. For example, in one run the vector of eigenvalues and the computed correlation matrix were

```
x =

    0.3844
    1.8365
    0.7791


A =

    1.0000   -0.2568   -0.6458
   -0.2568    0.9379    0.2772
   -0.6458    0.2772    1.0621
```

The eigenvalues of the matrix agree with the specified ones to the working precision ($u \approx 10^{-16}$), as expected, but the $(2,2)$ and $(3,3)$ elements are far from 1. In all such examples the NAG routine returns a nonzero error flag signalling the inaccurate diagonal.

According to the Guide to Available Mathematical Software (`http://gams.nist.gov`), the IMSL STAT/LIBRARY contains a routine rncor that generates a random correlation matrix with specified eigenvalues. The documentation provided on GAMS does not specify the method, but it says that an error flag indicates "Considerable loss of precision occurred in the rotations used to form the correlation matrix. Some of the diagonals of COR differ from 1.0 by more than the machine epsilon", which suggests that this routine is also based on the code from [39].

Our conclusion is that the Fortran code in [39] uses an unstable implementation of the Bendel–Mickey algorithm; this code and those based on it should be modified to obtain the Givens rotation from (5.7) and (5.8) and to explicitly set the diagonal to 1.

5.4 Generating (Cholesky) Factors of Correlation Matrices

Any correlation matrix $A \in \mathbb{R}^{n\times n}$ can be written X^TX, where $X \in \mathbb{R}^{m\times n}$ with $m \geq n$ has columns of unit 2-norm. Two-sided orthogonal transformations $A \leftarrow Q^TAQ$ correspond to one-sided transformations $X \leftarrow XQ$. Using this connection, we can derive an analogue of Algorithm 5.2.1 that constructs a random rectangular matrix with columns of unit 2-norm and specified singular values, provided that the squares of the singular values sum to n. This algorithm is of interest for at least two reasons. First, irrespective of rounding errors, the factor X always represents a positive semidefinite matrix $A = X^TX$ and its elements have half the dynamic range of those of A. For computational convenience, and to reduce storage, X can be taken to be square and upper triangular, that is, as the Cholesky factor of A (up to signs). Second, the one-sided Jacobi method for

computing the singular value decomposition (SVD) has high accuracy properties analogous to those mentioned in Section 2.1 for the two-sided algorithm [10], [41]. In this case the relative error in the computed singular values satisfies a bound proportional to the condition number of the matrix scaled to have columns of unit 2-norm, and this matrix has 2-norm condition number within $\sqrt{n}$ of the smallest obtainable by column scaling. The factor X of a correlation matrix is therefore useful for testing the one-sided Jacobi method.

We denote the jth column of X by x_j.

Algorithm 5.4.1 *Given nonnegative scalars $\sigma_1, \ldots, \sigma_n$ with $\sum_{i=1}^n \sigma_i^2 = n$, and an integer $m \geq n$, this algorithm produces a random matrix $X \in \mathbb{R}^{m\times n}$ having columns of unit 2-norm and singular values $\sigma_1, \ldots, \sigma_n$. Optionally, X may be taken to be upper triangular.*

1. Form $X = U\,\mathrm{diag}(\sigma_i)V^T$, where $U \in \mathbb{R}^{m\times n}$ and $V \in \mathbb{R}^{n\times n}$ are
   random matrices with orthonormal columns from the Haar distribution.
2. while some $\|x_j\|_2 \neq 1$
   Find indices i and j with $i < j$ so that $\|x_i\|_2 < 1 < \|x_j\|_2$
   or $\|x_i\|_2 > 1 > \|x_j\|_2$.
   Form $a_{ii} = x_i^T x_i$, $a_{ij} = x_i^T x_j$, $a_{jj} = x_j^T x_j$.
   Construct a Givens rotation $Q = \left[\begin{smallmatrix} c & s \\ -s & c \end{smallmatrix}\right]$ in the (i,j) plane
   according to (5.4), (5.7) and (5.8) to set $\|x_i\|_2 = 1$
   and transform $X \leftarrow XQ$.
   end
   Optionally, compute the QR factorization $X = \widetilde{Q}R$ and
   replace X by $R(1\colon n, 1\colon n)$.

As for Algorithm 5.2.1, the while loop in Algorithm 5.4.1 is executed at most $n-1$ times. The cost of the algorithm is m^3+n^3 flops for the initial formation of X

(which is done without explicitly forming U and V—see [54], [31, Section 26.3]), plus at most $10n^2$ flops for the second stage (note that the column norms $\|x_j\|_2$ need only be computed once, at the start of the second stage, and can then be updated). The optional QR factorization costs a further $2n^2(m - n/3)$ flops (assuming the use of Householder transformations). Algorithm 5.4.1 is implemented in MATLAB 6 as `gallery('randcolu',...)`.

Algorithm 5.4.1 has numerical stability and accuracy properties analogous to those of Algorithm 5.2.1.

We mention that Drmač [19], [18] constructs matrices of the form generated by Algorithm 5.4.1 by applying a one-sided, apparently infinite, product of Givens rotations to a diagonal matrix of singular values, but he gives no details of how this is done.

Chapter 6

Numerical Results

In this section we give several examples to illustrate the behaviour of Cholesky–Jacobi and the Cholesky–QR methods, the sharpness of our backward error bounds and how these algorithms compare. We also show the need for pivoting in the Cholesky–QR method, and to show the benefits of iterative refinement. All our experiments were carried out in MATLAB 5, in which matrix computations are based on LAPACK; the unit roundoff is $u = 2^{-53} \approx 1.1 \times 10^{-16}$. In Algorithms 4.1.2 and 4.1.3 convergence was declared when $\eta_\infty(\widehat{x}, \widehat{\lambda}) \leq u$.

We start by presenting two examples, including a practical example of a vibration problem in structural engineering, where even though B is ill-conditioned, the two algorithms can produce backward stable solutions.

Example 1. Our first example illustrates how our backward error bounds can correctly predict perfect backward stability of the Cholesky–Jacobi and the Cholesky–GQR methods despite large values of $\kappa_2(B)$. We take

$$A = H - I \in \mathbb{R}^{n\times n}, \quad B = \mathrm{diag}(1, \epsilon, \epsilon^2, \ldots, \epsilon^{n-1})$$

where H is the Hilbert matrix. We also applied the Cholesky–GQR and the Cholesky–HQR methods to the permuted example (PHP^T, PBP^T) where $P = I(:, [n\!:\!-1\!:\!1])$ is a permutation matrix. For $n = 8$ and $\epsilon = 10^{-1}, 10^{-2}, 10^{-3},$

Table 6.1: Terms from error analysis of Cholesky–Jacobi method and backward error for Example 1.

ϵ	$\kappa_2(B)$	$\max \omega_k$	$\max \mu_k^2$	$\max \pi_k$	$\max \eta_2(\widehat{x}, \widehat{\lambda})$
10^{-1}	10^7	7.98e-1	3.33e0	3.12e0	7.27e-17
10^{-2}	10^{14}	1.90e0	4.38e0	7.02e0	3.79e-17
10^{-3}	10^{21}	2.38e0	4.67e0	1.04e1	1.84e-17

Table 6.2: Terms from error analysis of Cholesky–QR method and backward error for Example 1.

ϵ	$\kappa_2(B)$	Method	$\max \omega_k$	$\max \kappa_2(N_k)$	$\max \eta_2(\widehat{x}, \widehat{\lambda})$
10^{-1}	10^7	Givens, not permuted	29.05	595.2	5.81e-14
		H'holder, not permuted			1.89e-11
		Givens, permuted	1.30	12.8	1.51e-15
		H'holder, permuted			1.13e-15
10^{-2}	10^{14}	Givens, not permuted	2.98e2	7.20e3	1.59e-11
		H'holder, not permuted			8.71e-06
		Givens, permuted	3.07	20.3	5.42e-16
		H'holder, permuted			1.21e-15
10^{-3}	10^{21}	Givens, not permuted	2.99e3	7.29e4	5.00e-10
		H'holder, not permuted			1.24e-01
		Givens, permuted	5.44	26.8	2.04e-15
		H'holder, permuted			9.65e-16

Tables 6.1 and 6.2 shows the values of the terms appearing in the error analysis along with the maximum backward error over all the computed eigenpairs. The Cholesky–QR method was only stable when it was applied to the permuted example so the matrix C in (3.5) was graded downwards. This is illustrated in Figure 6.1 where for the Cholesky–GQR method applied to (H, B), C in (3.5) will be graded upwards and we have large values of ω when the tridiagonal QR iteration starts. When the Cholesky–GQR method applied to (PHP^T, PBP^T), C is graded downwards, and Figure 6.1 shows that all values of ω are of order 1 and the algorithm is stable on this example.

Example 2. This example is a structural engineering problem that again illustrates independence of our backward error bounds on $\kappa_2(B)$. We consider a cantilever beam as shown in Figure 6.2(a). We assume that the cantilever is

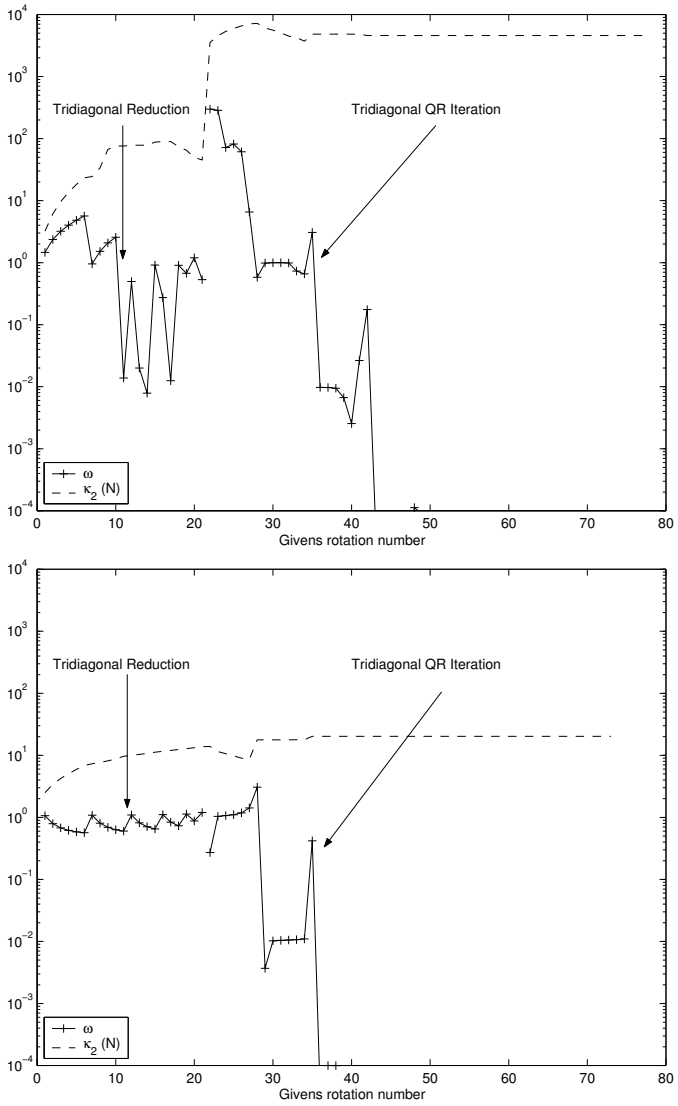
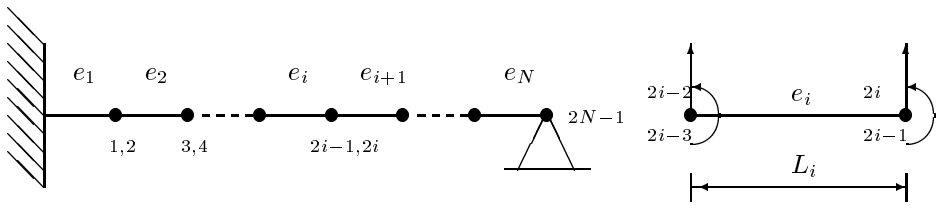


Figure 6.1: Terms from error analysis of Cholesky–GQR method for each Givens rotation for Example 1 where $\epsilon = 10^{-2}$. Top: Givens, not permuted. Bottom: Givens, permuted.



(a) Geometry of supported cantilever beam. (b) Beam finite element.

Figure 6.2: Single span cantilever beam with supported end point.

rigid in its axial direction and that all the deformations are small. The boundary conditions are full-fixity at the base and zero translational displacement at the cantilever end. We also assume that the material properties and cross sections vary along the length of the beam. The equation of motion for the natural vibrations has the form

$$M\ddot{v} + Kv = 0,$$

where M denotes the symmetric positive definite mass inertia matrix and K the symmetric positive definite stiffness matrix. The finite element method leads to the generalized eigenvalue problem

$$K\phi = \lambda M\phi. \tag{6.1}$$

The cantilever is modeled with N finite elements. Each element has 4 degrees of freedom, namely the two beam-end lateral displacements and the two beam-end rotations as shown in Figure 6.2(b). The length of the ith finite element e_i is taken to be L_i and its flexural characteristic to be $(EI)_i$, where E is the modulus of elasticity and I the moment of inertia. The global degrees of freedom are numbered as shown in Figure 6.2(a). If cubic Hermite interpolation polynomials are used to describe displacement along the beam element, then the beam element

stiffness matrix is [43]

$$K_i = \frac{2(EI)_i}{L_i^3}\begin{bmatrix} 6 & 3L_i & -6 & 3L_i \\ 3L_i & 2L_i^2 & -3L_i & L_i^2 \\ -6 & -3L_i & 6 & -3L_i \\ 3L_i & L_i^2 & -3L_i & 2L_i^2 \end{bmatrix}$$

and the beam element consistent mass matrix is

$$M_i = \frac{\overline{m}_i L_i}{420}\begin{bmatrix} 156 & 22L_i & 54 & -13L_i \\ 22L_i & 4L_i^2 & 13L_i & -3L_i^2 \\ 54 & 13L_i & 156 & -22L_i \\ -13L_i & -3L_i^2 & -22L_i & 4L_i^2 \end{bmatrix},$$

where $\overline{m}_i$ is the average mass per unit length for the ith beam. The global stiffness and mass inertia matrices are obtained by assembling the K_i and M_i, $i = 1{:}N$.

For our example, we chose $N = 5$ finite elements leading to 9 degrees of freedom and we varied the parameters e_i, L_i, $(EI)_i$ and $\overline{m}_i$, sometimes applying direct search to maximize the backward error over these variables. The backward errors for the Cholesky–Jacobi and the Cholesky–HQR method with pivoting were always of order u, with our backward error bounds for Algorithm 3.3.1 also of order u. Table 6.3 shows results for two sets of parameters. The second set of results shows again that pivoting can be needed for stability of the Cholesky–QR method.

Example 3. This is an example where the Cholesky–Jacobi method is unstable and there is only one large value of ω_k. With $n = 10$, we take $A \in \mathbb{R}^{n\times n}$ to be a random symmetric matrix and $B = I_n$ and replace the (n, n) entries of each matrix by 10^{-24}. Jacobi rotations not involving the nth plane have $\rho = 1$ and therefore ω_k is small. However, when we first apply a Jacobi rotation in the $(1, n)$ plane we see that $\rho = 10^{12}$ and

$$a_{11} - \rho^2 a_{nn} = a_{11} - 1 \ll \rho a_{1n} = 10^{12} a_{1n}$$

Table 6.3: Result for two instances of the cantilever beam problem.

$\kappa_2(M) = 3.9 \times 10^{10}$, $\kappa_2(L) = 1.8$

	$\max \omega_k$	$\max \mu_k^2$	$\max \pi_k$	$\max \eta_2(\widehat{x}, \widehat{\lambda})$
Cholesky–Jacobi	4.58e0	8.3e0	1.63e0	9.27e-17
Cholesky–QR (no pivoting)				9.38e-17
Cholesky–QR (with pivoting)				5.78e-17

$\kappa_2(M) = 6.7 \times 10^{6}$, $\kappa_2(L) = 2.2$

	$\max \omega_k$	$\max \mu_k^2$	$\max \pi_k$	$\max \eta_2(\widehat{x}, \widehat{\lambda})$
Cholesky–Jacobi	3.86e0	4.18e0	2.45e0	1.78e-16
Cholesky–QR (no pivoting)				3.59e-12
Cholesky–QR (with pivoting)				1.41e-16

and therefore, from (3.17), $sc \approx 1/2$ and $\omega_k \approx 5 \times 10^{11}$. This is the only ill-conditioned M_k transformation as, using our scaling strategy, we set $\widetilde{d}_n^2 = c^2 d_n^2 + s^2 d_1^2 = O(1)$ in (3.30), and afterwards ρ is always approximately 1 for all subsequent rotations. The other key terms from the error bounds are $\max_k \pi_k = 8.4 \times 10^{11}$ and $\max_k \mu_k^2 = 2.0$. The computed eigenvalues consist of a group of 8 of order 1, all with backward errors of order 10^{-5} and two eigenvalues of order 10^{12}, with backward errors of order u. Applying Algorithm 4.1.2 to the eigenvalues with large backward errors we found that backward errors of order u were produced within 3–6 iterations; Algorithm 4.1.3 did not converge for any of the eigenvalues. The Cholesky–GQR and the Cholesky–HQR method were stable on this example. The key terms from the analysis where $\max_k \omega_k = 1.89$ and $\max_k \kappa_2(N_k) = 6.25$ so our error bounds correctly predict the small backward errors.

Example 4. Our next example is adapted from a problem used by Fix and Heiberger [21] and shows that it is possible for the Cholesky–Jacobi method to be stable when the Cholesky–QR method, both with and without a pre-permutation

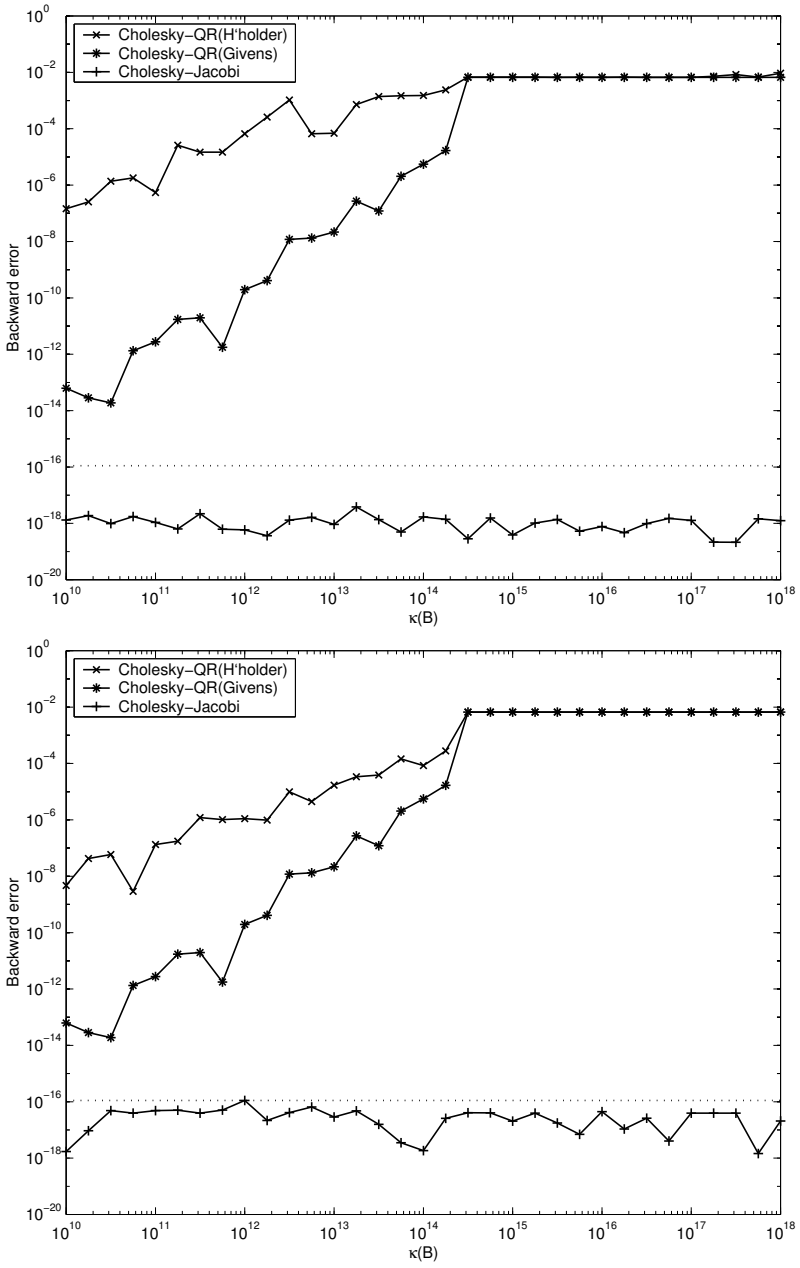


Figure 6.3: Behaviour of the backward error for eigenvalue of smallest modulus of problem (6.2) with $\alpha = 1$, $\delta = 10^{-2}$ and $\beta = 10^{-8}$. Top : not permuted. Bottom: permuted. Dotted line denotes unit roundoff level.

step, using either Householder or Givens tridiagonalization, is unstable. Let

$$A = \begin{bmatrix} 1 & \alpha & \beta & \delta \\ \alpha & 2 & 0 & 0 \\ \beta & 0 & 3 & 0 \\ \delta & 0 & 0 & \epsilon \end{bmatrix}, \quad B = \mathrm{diag}(\epsilon, 1, \epsilon, 1), \qquad \alpha, \delta > 0, \quad 0 < \epsilon < 1. \tag{6.2}$$

We solved the problem for $\alpha = 1$, $\delta = 10^{-2}$, $\beta = 10^{-8}$ and a range of ϵ from 10^{-10} to 10^{-18} by the Cholesky–Jacobi method and the Cholesky–QR method. Figure 6.3 plots the condition number of B against the backward error $\eta_2(\widehat{x}_{\min}, \widehat{\lambda}_{\min})$, for the eigenvalue $\widehat{\lambda}_{\min}$ of minimal modulus. The Cholesky QR method performs unstably for all of the pairs (A, B), while Algorithm 3.3.1 displays excellent stability. For Algorithm 3.3.1 we have $\max_k \omega_k = \max_k \pi_k = 1.0$ and $\max_k \mu_k^2 = 1.71$, so our error bounds predict the small backward errors.

Example 5. This example is one of a form suggested by G. W. Stewart, and is similar to Example 3, that causes difficulties for the Cholesky–Jacobi method, and we use it to compare Algorithms 4.1.2 and 4.1.3. The matrices are

$$\mathrm{diag}(A) = d, \quad a_{ij} = \min(i, j) \text{ for } i \neq j, \quad B = \mathrm{diag}(d), \quad d = [1, \epsilon, \epsilon^2, \ldots, \epsilon^{n-1}],$$

with $0 < \epsilon < 1$. We take $n = 8$ with three choices of ϵ, and concentrate on the three eigenvalues of smallest absolute value. We report in Table 6.4 the backward error $\eta_\infty(\widehat{x}, \widehat{\lambda})$ of the computed eigenpair and the forward error

$$e(\widehat{\lambda}) = \frac{|\lambda - \widehat{\lambda}|}{|\lambda|}$$

of the computed eigenvalue, where the exact λ is obtained using MATLAB's Symbolic Math Toolbox; these statistics are given both before and after refinement, together with the number of iterations required by Algorithms 4.1.2 and 4.1.3, where "$*$" denotes no convergence after 50 iterations and in this case the quantities from the 50th iteration are shown. Table 6.5 shows the size of the terms appearing in the error bounds of Section 3.3.3. The observed instability

Table 6.4: Iterative refinement of eigenpairs of Example 5. For the entry marked †, convergence was not to the eigenvalue indicated in the leftmost column.

	Before refinement		After refinement					
			Algorithm 4.1.2			Algorithm 4.1.3		
λ	$\eta_\infty(\widetilde{x},\widetilde{\lambda})$	$e(\widetilde{\lambda})$	$\eta_\infty(\widehat{x},\widehat{\lambda})$	$e(\widehat{\lambda})$	it	$\eta_\infty(\widehat{x},\widehat{\lambda})$	$e(\widehat{\lambda})$	it
$\epsilon = 2^{-6} \approx 1.6 \times 10^{-2}$								
1.4e0	6e-7	1e-5	8e-18	2e-16	2	6e-18	0	3
−4.6e1	2e-8	1e-7	2e-18	2e-16	2	2e-18	2e-16	2
−8.4e3	4e-11	1e-9	5e-20	0	1	5e-20	2e-16	2
$\epsilon = 2^{-8} \approx 3.9 \times 10^{-3}$								
1.4e0	7e-5	2e-3	7e-18	0	2	1e-17	2e-16	5
−1.8e2	9e-6	2e-4	5e-18	2e-16	2	7e-17	2e-15	8
−1.4e4	2e-9	2e-6	9e-22	0	2	8e-17	8e-14	19
$\epsilon = 2^{-12} \approx 2.4 \times 10^{-4}$								
1.4e0	8e-3	1e0	2e-20	$0^\dagger$	5	5e-3	1e0	$*$
−3.0e3	2e-3	6e2	4e-20	2e-16	8	2e-3	1e0	$*$
−3.5e7	3e-5	3e-1	2e-17	2e-16	3	3e-5	3e-1	$*$

Table 6.5: Terms from error analysis for Example 5.

ϵ	$\kappa_2(B)$	$\max \omega_k$	$\max \mu_k^2$	$\max \pi_k$
2^{-6}	4e12	1.3e5	7.9	1.1e10
2^{-8}	7e16	1.7e7	8.0	8.8e13
2^{-12}	2e25	2.8e11	8.0	5.7e21

corresponds to large ω_k, but μ_k^2 is small, as is usually the case. We see that, as expected from the theory [57], refining with the unstable linear system solver produces the same limiting backward error as when the stable solver is used, but that it can produce slower convergence and is less likely to converge at all, as we saw also in Example 3. Iterative refinement also improves the forward error e. As one entry in the table shows, it is possible for iterative refinement to converge to a different eigenpair than expected when the original approximate eigenpair is sufficiently poor. The Cholesky–HQR method performs stably on this example.

Example 6. Our last example illustrates how ill condition of L can cause instability. Here, $n = 20$, $A = I$ and $B = R^T R$, where R is a Kahan matrix, and $\kappa_2(B) \approx 1/u$, $\kappa_2(L) \approx 6 \times 10^4$. Figure 6.4 plots the eigenvalues on the x-axis

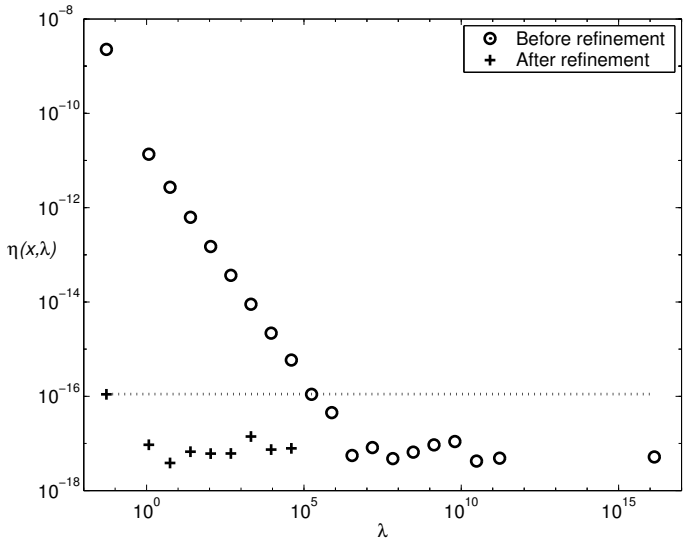


Figure 6.4: Backward errors for Cholesky–HQR method before and after iterative refinement for Kahan matrix example. Dotted line denotes unit roundoff level.

versus the ∞-norm backward errors of the eigenpairs on the y-axis, for eigenpairs both before and after refinement. At most one step of iterative refinement was required. The Cholesky–HQR method was used, with Algorithm 4.1.3; the Cholesky–Jacobi method and 4.1.2 give very similar results. The quantities in the error bounds for the Cholesky–Jacobi method are $\max \omega_k = 0.6$, $\max \mu_k^2 = 285$, $\max \pi_k = 2.4$. As expected, it is the small eigenvalues that have large backward errors initially.

6.1 Conclusions and Future Work

We have shown, through error analysis, that the Cholesky–Jacobi and the Cholesky–GQR methods have better numerical stability properties than the standard backward error bound (3.11) suggests. For problems with an ill conditioned B, these methods can be, and often are, perfectly stable, and numerical experiments show that our bounds can predict the stability well. Our examples have shown that

the popular Cholesky–HQR method also has better numerical stability properties than the standard backward error bound (3.11) suggests. This method seems to be stable when the Cholesky–GQR method is stable but in cases of instability the Cholesky–GQR method seems to outperform the Cholesky–HQR method. A detailed error analysis of the Householder transformations is needed, along the same lines as our analysis of the 2×2 rotation matrices used in the Cholesky–Jacobi and the Cholesky–GQR methods, to fully understood this method. These methods are of practical use as Jacobi's method is easy to code and particularly attractive in a parallel computing environment, while the symmetric QR method is already implemented in LAPACK and MATLAB.

In practice, the Cholesky–QR method appears to perform as well as the Cholesky–Jacobi method, provided that complete pivoting is used in the Cholesky factorization. This reduces the condition of the matrix L and concentrates any ill-conditioning of the matrix B into D^2. As we noted in Section 3.4 this can, to some extent, be explained by the QR method's good performance on graded matrices.

Instability of the Cholesky methods can be cured by iterative refinement, provided it is not too severe, as we have illustrated. Our experience with iterative refinement shows that it is not necessary to scale or to reverse the problem so that (4.44) holds—a backward error of order u is obtained as long as the starting vector is good enough for Newton's method to converge. We have also shown that the unstable linear system solver, Algorithm 4.1.3 produces the same limiting backward error as the stable solver, Algorithm 4.1.2, but that the convergence is slower and it is less likely to converge at all. When Algorithm 4.1.3 can be used safely remains an open problem. When the error in the starting vector becomes too severe, iterative refinement can fail to converge, or converge to the wrong eigenpair. These instances are rare and require extreme examples, such as

Example 5, which involves B such that $\kappa_2(B) = 2e25$ and is specifically designed such that the terms in the error analysis of the Cholesky–Jacobi method are large.

The Cholesky–QR method (without pivoting) is the standard method for solving the symmetric definite generalized eigenvalue problem in LAPACK, MATLAB 6 and the NAG Library, all of which aim to provide exclusively backward stable algorithms. It is clearly desirable for these implementations to incorporate pivoting in the Cholesky factorization, in order to enhance the reliability, and to incorporate the option of iterative refinement of selected eigenpairs, to ameliorate those instances where the Cholesky–QR method behaves unstably.

We have also shown that the popular Bendel and Mickey algorithm for generating random correlation matrices uses numerically unstable formulae, for the generation of the Givens rotations, in existing Fortran implementations. We have given improved formulae for computing the rotations and prove that the resulting algorithm is numerically stable. We have also shown how to modify the algorithm to generate a rectangular matrix with columns or unit 2-norm. Such a matrix represents a correlation matrix in factored form. This form may be preferable, if the application allows it, as the correlation matrix generated by the Bendel and Mickey algorithm can be indefinite if $\max(\lambda_i)/\min(\lambda_i) > n^{-1/2}\tilde{\gamma}_{n^2}^{-1}$.

Bibliography

[1] E. Anderson, Z. Bai, C. H. Bischof, J. W. Demmel, J. J. Dongarra, J. J. Du Croz, A. Greenbaum, A. McKenney, S. Ostrouchov, and D. C. Sorensen. *LAPACK Users' Guide, Release 2.0.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 1995.

[2] M. F. Anjos, S. J. Hammarling, and C. C. Paige. Solving the generalized symmetric eigenvalue problem. Unpublished Manuscript, December 1992.

[3] Richard Barrett, Michael Berry, Tony Chan, James Demmel, June Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Hank van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* SIAM, Philadelphia, 1993. Obtainable from research.att.com:/netlib/linalg using ftp.

[4] R. B. Bendel and M. R. Mickey. Population correlation matrices for sampling experiments. *Commun. Statist–Simul. Comput.*, B7(2):163–182, 1978.

[5] A. Bunse-Gerstner. An algorithm for the symmetric semidefinite generalized eigenvalue problem. *Linear Algebra and Appl.*, 58:43–68, 1984.

[6] N. N. Chan and Kim-Hung Li. Diagonal elements and eigenvalues of real symmetric matrices. *J. Math. Anal. Appl.*, 91:562–566, 1983.

[7] N. N. Chan and Kim-Hung Li. A FORTRAN subroutine for finding a real

symmetric matrix with prescribed diagonal elements and eigenvalues. *The Computer Journal*, 26(2):184–186, May 1983.

[8] S. Chandrasekaran. An efficient and stable algorithm for the symmetric-definite generalized eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 21(4):1202–1228, 2000.

[9] Biswa Nath Datta. *Numerical Linear Algebra and Applications*. Brooks/Cole Publishing Company, CA, USA, 1995.

[10] James Demmel and Krešimir Veselić. Jacobi's method is more accurate than QR. *SIAM J. Matrix Anal. Appl.*, 13(4):1204–1245, 1992.

[11] James W. Demmel. On floating point errors in Cholesky. Technical Report CS-89-87, Department of Computer Science, University of Tennessee, Knoxville, TN, USA, October 1989. LAPACK Working Note 14.

[12] James W. Demmel. *Applied Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1997.

[13] James W. Demmel and A. McKenney. A test matrix generation suite. Preprint MCS-P69-0389, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, USA, March 1989. LAPACK Working Note 9.

[14] J. E. Dennis Jr. and Robert B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1983.

[15] Inderjit S. Dhillon. *A New $O(n^2)$ Algorithm for the symmetric tridiagonal eigenvalue/eigenvector problem*. PhD thesis, University of California, Berkeley, 1997.

[16] J. J. Dongarra, C. B. Moler, and J. H. Wilkinson. Improving the accuracy of computed eigenvalues and eigenvectors. *SIAM J. Numer. Anal.*, 20(1):23–45, 1983.

[17] Jack J. Dongarra. Algorithm 589: SICEDR: A FORTRAN subroutine for improving the accuracy of computed matrix eigenvalues. *ACM Transactions on Mathematical Software*, 8(4):371–375, December 1982.

[18] Zlatko Drmac. Accurate computation of the product-induced singular value decomposition with applications. *SIAM Journal on Numerical Analysis*, 35(5):1969–1994, October 1998.

[19] Zlatko Drmac. A tangent algorithm for computing the generalized singular value decomposition. *SIAM Journal on Numerical Analysis*, 35(5):1804–1832, October 1998.

[20] P. A. Fillmore. On similarity and the diagonal of the matrix. *AMM*, 76:167–169, February 1969.

[21] G. Fix and R. Heiberger. An algorithm for the ill-conditioned generalized eigenvalue problem. *SIAM J. Numer. Anal.*, 9:78–88, 1972.

[22] G. E. Forsythe and E. G. Straus. On best conditioned matrices. *Proc. Amer. Math.*, 6:340–345, 1955.

[23] J. G. F. Francis. The QR transformation: A unitary analogue to the LR transformation, parts I and II. *Comput. J.*, 4:265–272, 332–345, 1961.

[24] Valérie Frayssé and Vincent Toumazou. A note on the normwise perturbation theory for the regular generalized eigenproblem. *Numerical linear algebra with applications*, 5(1):1–10, January/February 1998.

[25] Noel Gastinel. *Linear Numerical Analysis*. Kershaw Publishing, London, 1983. First published in English by Academic Press, London, 1970.

[26] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, Maryland, USA, 2nd edition, 1989.

[27] Anne Greenbaum. *Iterative Methods for Solving Linear Systems*. Frontiers in Applied Mathematics, No. 17. SIAM, Philadelphia, 1997.

[28] H. V. Henderson and S. R. Searle. On deriving the inverse of a sum of matrices. *SIAM Review*, 23(1):53–60, January 1981.

[29] Desmond J. Higham and Nicholas J. Higham. Structured backward error and condition of generalized eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 20(2):493–512, 1998.

[30] Nicholas J. Higham. The Test Matrix Toolbox for MATLAB (version 3.0). Numerical Analysis Report No. 276, Manchester Centre for Computational Mathematics, Manchester, England, September 1995.

[31] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1996.

[32] Nicholas J. Higham. Iterative refinement for linear systems and LAPACK. *IMA J. Numer. Anal.*, 17(4):495–509, 1997.

[33] R. B. Holmes. On random correlation matrices. *SIAM J. Matrix Anal. Appl.*, 12(2):239–272, April 1991.

[34] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, England, 1991.

[35] K. D. Ikramov. A remark on "A note on constructing a symmetric matrix with specified diagonal entries and eigenvalues". *BIT Numerical Mathematics*, 38(4):807–807, December 1998.

[36] C. G. J. Jacobi. Über ein leichtes Verfahren die in der Theorie der Säculärstörungen vorkommenden Gleichungen numerisch aufzulösen. *Journal für die reine und angewandte Mathematik*, 30:51–94, 1846.

[37] P. Lancaster. *Lambda-Matrices and Vibrating Systems*. Pergamon Press, Oxford, 1966.

[38] R. S. Leite, T. R. W. Richa, and C. Tomei. Geometric proofs of some theorems of Schur–Horn type. *Linear Algebra and its Applications*, 286(1–3):149–173, January 1999.

[39] S. P. Lin and R. B. Bendel. Generation of population correlation matrices with specified eigenvalues. *Applied Statistics*, 34(2):193–198, June 1985.

[40] George Marsaglia and Ingram Olkin. Generating correlation matrices. *SIAM J. Sci. Stat. Comput.*, 5(2):470–475, June 1984.

[41] Roy Mathias. Accurate eigensystem computations by Jacobi methods. *SIAM Journal on Matrix Analysis and Applications*, 16(3):977–1003, July 1995.

[42] The MathWork, Inc., Natick, MA, USA. *MATLAB User's Guide*, 1992.

[43] Leonard Meirovitch. *Elemennts of Vibration Analysis*. McGraw-Hill, New York, 2nd edition, 1973.

[44] C. B. Moler and G. W. Stewart. An algorithm for the generalized matrix eigenvalue problems. *SIAM J. Numer. Anal.*, 10(2):241–256, 1973.

[45] NAG Ltd, Oxford. *NAG Fortran Library Manual, Mark 18*, 1997.

[46] B. N. Parlett. Analysis of algorithms for reflections in bisectors. *SIAM Review*, 13:197–208, 1971.

[47] Beresford N. Parlett. *The Symmetric Eigenvalue Problem*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1980.

[48] G. Peters and J. H. Wilkinson. $Ax = \lambda x$ and the generalized eigenproblem. *SIAM J. Numer. Anal.*, 7(4):479–492, 1970.

[49] Y. Saad. *Numerical Methods for Large Eigenvalue Problems: Theory and Algorithms*. John Wiley, New York, 1992.

[50] A. Schonage. On the quadratic convergence of the Jacobi process. *Numer. Math.*, 6:410–412, 1964.

[51] James R. Schott. *Matrix Analysis for Statistics*. Wiley, New York, 1997.

[52] G. W. Stewart. Matrix Algorithm Vol. II Eigensystems. Available from ftp://thales.cs.umd.edu/pub/survey/.

[53] G. W. Stewart. *Introduction to Matrix Computations*. Academic Press, New York, 1973.

[54] G. W. Stewart. The efficient generation of random orthogonal matrices with an application to condition estimators. *SIAM Journal on Numerical Analysis*, 17(3):403–409, June 1980.

[55] G. W. Stewart and Ji-guang Sun. *Matrix Perturbation Theory*. Academic Press, London, England, 1990.

[56] H. J. Symm and J. H. Wilkinson. Realistic error bounds for a simple eigenvalue and its associated eigenvector. *Numerische Mathematik*, 35(2):113–126, September 1980.

[57] F. Tisseur. Newton's method in floating point arithmetic and iterative refinement of generalized eigenvalue problems. Numerical Analysis Report No. 346, Manchester Centre for Computational Mathematics, Manchester, England, August 1999.

[58] Solutions to problem E2741:. Similarity and the diagonal of a matrix. *Amer. Math. Monthly*, 87:62–63, 1980.

[59] William F. Trench. Numerical solution of the eigenvalue problem for Hermitian Toeplitz matrices. *SIAM Journal on Matrix Analysis and Applications*, 10(2):135–146, 1989.

[60] A. van der Sluis. Condition numbers and equilibration of matrices. *Numerische Mathematik*, 14(1):14–23, 1969.

[61] D. S. Watkins. *Fundamentals of Matrix Computations*. John Wiley & Sons, New York, 1991.

[62] J. H. Wilkinson. Note on the quadratic convergence of the cyclic Jacobi process. *Numerische Mathematik*, 4:296–300, 1962.

[63] J. H. Wilkinson. *The Algebraic Eigenvalue Problem*. Oxford University Press, London, 1965.

[64] J. H. Wilkinson. A priori error analysis of algebraic processes. In I. G. Petrovsky, editor, *Proc. International Congress of Mathematicians, Moscow 1966*, pages 629–640. Mir Publishers, Moscow, 1968.

[65] Hongyuan Zha and Zhenyue Zhang. A note on constructing a symmetric matrix with specified diagonal entries and eigenvalues. *BIT Numerical Mathematics*, 35(3):448–451, September 1995.