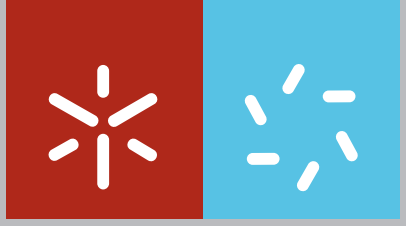
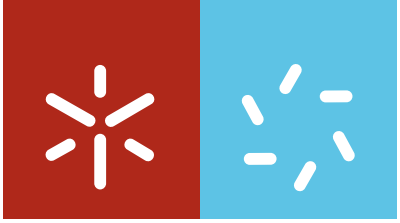


Universidade do Minho
Escola de Ciências

Carla Maria Alves Ferreira

The Unsymmetric Tridiagonal Eigenvalue Problem

Janeiro de 2007



Universidade do Minho
Escola de Ciências

Carla Maria Alves Ferreira

The Unsymmetric Tridiagonal Eigenvalue Problem

Tese de Doutoramento em Ciências
Área de Conhecimento - Matemática

Trabalho efectuado sob a orientação de
Emeritus Professor Beresford Neill Parlett
Professor Doutor Rui Manuel Silva Ralha

Janeiro de 2007

Acknowledgments

The author would like to express her extreme gratitude to Professor Beresford Parlett for sharing his precious time and for making the work of this thesis so rewarding. His expertise, vision and clarity of thought greatly improved this work, as well as its presentation. Our discussions, over many afternoons, have proven enormously valuable in the development of this work and in shaping new ideas for future investigation.

The author is also grateful to Professor Rui Ralha for introducing her to the field of numerical linear algebra and for all the opportunities of research he has offered her, especially the initial encouragement for a long-term visit to the University of California at Berkeley. His suggestions and comments were always important.

The author is also thankful to many friends in the Mathematics Department of the University of Minho, particularly the ones with whom she had the chance to work in the last two academic years.

The author would like to thank to Professor Lisa Santos, Chair of the Mathematics Department, for having been extremely helpful with all the formalities that have made possible the completion of this thesis.

The author could never forget to express many thanks to her mother for her unconditional love and emotional support in every single stage of her journey. Also many thanks to her brothers for their constant advice and understanding.

The author also thanks all her friends who have made life in Berkeley an enriching experience.

Finally, the author is appreciative to the Center of Mathematics of the University of Minho for the financial support for this research.

The Unsymmetric Tridiagonal Eigenvalue Problem

Abstract

The development of satisfactory methods for reducing an unsymmetric matrix to tridiagonal form has been greatly hampered by the fact that there is not an accepted good algorithm for exploiting this form. Nevertheless, recently, promising elimination techniques for achieving a stable reduction to this form have been developed. But the standard QR algorithm destroys it immediately. Our work aims to fill this gap in the armoury of software tools for the matrix eigenvalue problem and so encourage the refinement of methods to reduce a matrix to tridiagonal form.

The *progressive* q*uotient* d*ifference* algorithm with s*hifts* (qds) was presented by Rutishauser as early as 1954. It is equivalent to the shifted LR algorithm written in a special notation for tridiagonals. The much more recent d*ifferential* qds (dqds) is a sophisticated variant of qds. The first contribution of this thesis is a new algorithm, 3dqds, that consists of three dqds steps performed implicitly and such that real arithmetic is maintained in the presence of complex eigenvalues. One advantage of our algorithm over the Hessenberg QR algorithm is that it preserves the tridiagonal form and thus reduces both storage and time. We present some accuracy results comparing a MATLAB implementation of 3dqds with the function eig of that software. These preliminary results suggest the robustness of 3dqds algorithm.

In contrast to the symmetric case, unsymmetric matrices can have a mixture of eigenvalues, some robust in the face of perturbations while others extremely sensitive. We present several condition numbers, some new, that take advantage of tridiagonal form. Ideally an algorithm should report these numbers along with each computed eigenvalue.

On the theoretical side, we present a rigorous proof of a surprising result. It is well known that the greater the ratio of adjacent eigenvalues, the faster LR converges. Nevertheless, in exact arithmetic, LR still converges even when all the eigenvalues are equal and the Jordan form is one big block.

Cálculo de Valores Próprios de Matrizes Tridiagonais Não Simétricas

Resumo

O desenvolvimento de métodos satisfatórios para reduzir uma matriz não simétrica à forma tridiagonal tem sido fortemente travado pelo facto de que não existe um bom algoritmo aceite para explorar esta forma. Contudo, recentemente, promissoras técnicas de eliminação para realizar esta redução de maneira estável foram desenvolvidas. Mas o algoritmo QR standard destrói a forma tridiagonal imediatamente. O nosso trabalho pretende colmatar esta lacuna no conjunto das ferramentas computacionais para o problema de cálculo de valores próprios e assim encorajar o aperfeiçoamento de métodos para redução de uma matriz à forma tridiagonal.

O algoritmo qds (*progressive* q*uotient* d*ifference* with s*hifts*) foi introduzido por Rutishauser e remonta a 1954. É equivalente à versão *shifted* do algoritmo LR escrita numa notação especial para matrizes tridiagonais. O muito mais recente algoritmo dqds (d*ifferential* qds) é uma versão sofisticada do qds. Uma contribuição desta tese é um novo algoritmo, 3dqds, que consiste em três passos do dqds realizados implicitamente e tal que a aritmética real é mantida na presença de valores próprios complexos. Uma vantagem do nosso algoritmo em relação ao Hessenberg QR é que preserva a forma tridiagonal e assim reduz a necessidade de espaço em memória e o tempo de execução. Apresentamos alguns resultados numéricos comparando uma implementaçao do 3dqds em MATLAB com a função eig daquele *software*. Estes resultados preliminares sugerem a robustez do algoritmo 3dqds.

Em contraste com o caso simétrico, matrizes não simétricas podem ter um misto de valores próprios, alguns resistentes em face de perturbações enquanto outros extremamente sensíveis. Apresentamos vários números de condição, alguns novos, que tiram partido da forma tridiagonal. Idealmente, um algoritmo deve fazer acompanhar com estes números cada valor próprio calculado.

Do ponto de vista teórico, apresentamos também uma demonstração rigorosa de um resultado surpreendente. É bem conhecido que quanto maior for a razão entre valores próprios adjacentes, mais rapidamente o algoritmo LR converge. No entanto, em aritmética exacta, o algoritmo LR também converge mesmo quando todos os valores próprios são iguais e a forma de Jordan é um único bloco.

Contents

Introduction

A great deal of effort in science and engineering goes into eigenvalue computations. The symmetric case is well studied and there are good methods available. The unsymmetric case is intrinsically harder. The MATLAB [1] system [33] lets the user compute eigenvalues with one line of code, but this system, although wonderful for developing new numerical methods is too inefficient for day to day work in design and manufacturing. In addition, it cannot deal with the really large matrices, with order greater than 10000, that occur more and more often in applications.

So, we begin with the assumption that good algorithms are needed for real unsymmetric square matrices. Such algorithms have to be iterative in nature. When all eigenvalues are wanted then the preferred methods employ a sequence of similarity transformations which preserve the eigenvalues and gradually change the matrix to upper triangular form.

The reader might object at this point and say that the obvious strategy is to find the characteristic polynomial in a finite number of steps and then find the zeros of the polynomial. This approach died in the 1950's when it was appreciated that the coefficients of the characteristic polynomial are a too compact representation of the matrix eigenvalues; the eigenvalues are extremely sensitive to any uncertainty in the coefficients.

The next most compact pratical representation of a matrix is a tridiagonal form (all entries (i, j) are zero unless $|i - j| \leq 1$). All matrices can be reduced to such a form in a finite number of steps but the reduction is much easier in the symmetric case than in the unsymmetric one.

[1] MATLAB, shortcut for "Matrix Laboratory", is a commercial program sold by The Mathworks, Inc.

Now any iterative method will be much more efficient if the matrix is tridiagonal and if this form is preserved at each step. For example, MATLAB does not preserve this form and that is the reason why it is not suitable for large matrices.

Now comes the question: is the tridiagonal form, in the unsymmetric case, also too sensitive, just as the companion matrix that gives the characteristic polynomial? If it is too sensitive, then, in general, there are two avenues of escape. Either we can try to determine classes of tridiagonals that do determine their eigenvalues to adequate accuracy or we can accept that the Hessenberg form is as far as we should go in reducing a full matrix to a more compact form.

No one really knows the answer to this question. Some work has been done in defining suitable measures of sensitivity, called condition numbers, but no one has studied them carefully. The formulae for the sensitivity of polynomial zeros as functions of the coefficients showed immediately that the condition numbers were going to be huge as soon as the degree goes into the hundreds. For tridiagonals the situation is not so clear. We explore this question in more detail in our study.

The d*ifferential* q*uotient* d*ifference* algorithm with s*hifts* (dqds) was introduced by Fernando and Parlett in 1994 [16] to compute singular values of bidiagonal matrices to high relative accuracy but it may also be used to compute eigenvalues of tridiagonal matrices. In this thesis, based on previous work of Z. Wu [64], we propose a new algorithm for finding all the eigenvalues of a real unsymmetric tridiagonal matrix. This new algorithm, *triple* dqds (3dqds), incorporates three dqds steps implicitly. The motivation for 3dqds is to keep real arithmetic in the presence of complex shifts, efficiency and some elegance. One advantage of 3dqds over the standard Hessenberg QR (used by MATLAB) is this property: it preserves the tridiagonal form. The preliminary numerical results show that 3dqds is a vital tool in the context of eigenvalue problems.

It is possible that perturbations to the tridiagonal entries have just as much effect as perturbing entries far from the diagonal. That is where the more refined condition numbers we are also presenting come in, particularly the ones based on the derivative of an eigenvalue

with respect to various matrix entries.

The reader will find that there is a great deal of technical detail in this thesis. The algorithms that we will present are quite complicated and it was necessary to get all the details correct in order for the programs to work properly.

Next we are going to sketch an outline of this thesis.

In Chapter 1 we introduce notation, give some background and explain the importance of the unsymmetric tridiagonal eigenvalue problem.

Chapter 2 describes different representations for tridiagonal matrices beginning with the LU factorization. Then, new relative condition numbers for measuring the sensitivity of eigenvalues of tridiagonals are presented, to see if tridiagonal form plays a big role in a perturbation study.

In Chapter 3 we describe the LR algorithm, giving emphasis to the implicit double shifted version on an Hessenberg matrix. Then we explain the qds algorithm and show the relation to shifted LR on a tridiagonal matrix with superdiagonal entries of 1's.

Chapter 4 deals with convergence results for basic LR algorithm. The central part of this chapter is the proof of a new convergence result of basic LR algorithm on a real unreduced tridiagonal matrix with a one-point spectrum. The Jordan form is one big Jordan block.

Chapter 5 shows all the details of the derivation of a first version of the triple dqds - an algorithm that performs implicitly three steps of simple dqds keeping real arithmetic in the presence of complex shifts.

Chapter 6 explores the connection between dqds and the Gram-Schmidt orthogonalization process to reveal new results about triple dqds. These results lead to the more elegant and more efficient final version 3dqds.

Chapter 7 gives a preliminary numerical comparison between 3dqds and existing MATLAB's function `eig`. Although the set of test matrices is small, the numerical results permit us to conclude that 3dqds is a competitive algorithm.

Finally, in Chapter 8 we present a summary of our work and briefly set up some plans for future work.

Chapter 1

Setting the scene

This chapter describes notation, introduces definitions and some basic results, discusses canonical forms and perturbation theory in the context of eigenvalue problems. Then devotes attention to the unsymmetric tridiagonal eigenvalue problem describing the connection to the general unsymmetric eigenvalue problem and to Bessel matrices.

1.1 The eigenvalue problem

The eigenvalues of matrices or linear operators play a part in a very large number of applications, both theoretical and practical. In Chatelin [5, Chapter 3] we can find examples from diverse disciplines that show the extent of applications: they range from mathematics to chemistry, and to the dynamics of structures, even on economics. The determination of matrix eigenvalues is generally called the *eigenvalue problem* and it is a central topic in numerical linear algebra. From Golub [22] we quote what follows.

> The eigenvalue problem for square matrices A, that is the determination of nontrivial solutions of $Ax = \lambda x$, is inherently nonlinear and this leads to many computational problems. Computation of the eigenvalues λ via the characteristic equation
>
> $$\det(A - \lambda I) = 0$$

is, except for very special cases, not an option since the coefficients of the characteristic equation cannot be computed from determinant evaluations in a numerical stable way. And even if the characteristic equation could be determined accurately, then the computation of its roots, in finite precision, may be highly unstable since small perturbations in the coefficients may lead to large perturbations of the roots.

The numerical computation of the associated eigenvectors and generalized eigenvectors is even more delicate, in particular when eigenvectors of A make small angles with each other. In the limiting case, when the matrix is defective, A can be reduced to the Jordan canonical form, but arbitrary small perturbations in A may yield a nondefective matrix. This leads to many challenging numerical questions, which give rise to the central problem: how can we compute eigenvalues and eigenvectors in an efficient manner and how accurate are they? [...]

A method that is of great significance and serves as the basis for many algorithms is the Power iteration. It is still in use, but most frequentely as (implicit) part of more efficient techniques, e.g., krylov methods, inverse iteration, QR-method.

What becomes clear is that all these methods are of an iterative nature, and this is necessarily the case, since if there were a method of computing the eigenvalues of an nth order matrix in a finite number of computations, depending only on n, then this would be in contradiction with the fundamental theorem of Abel-Ruffini (and also a well-known result in Galois theory) that no such algorithm exists for the computation of the roots of a general polynomial of degree greater than 4. Hence, an algorithm for a matrix with a general structure (that is, not a diagonal matrix or a triangular matrix or alike) is necessarily iterative and the problem is to identify iterative algorithms which have a fast rate of convergence and lead to accurate results.

1.2 Notation, definitions and basic results

We describe the notation used hereafter and briefly set up well-known definitions and basic facts needed in this thesis.

The vector space of all real $m \times n$ matrices is denoted by $\mathbb{R}^{m\times n}$ and the vector space of real column n-vectors by $\mathbb{R}^n$. Similarly, $\mathbb{C}^{m\times n}$ denotes the vector space of $m \times n$ matrices with complex entries and $\mathbb{C}^n$ the vector space of complex column n-vectors.

Generally, we will have

capital letters	A, H, Δ	for matrices
(double) subscripted lower case letters	$a_{ij}, h_{ij}, \delta_{ij}$	for matrix elements
boldfaced lower case letters	$\boldsymbol{x}, \boldsymbol{c}, \boldsymbol{h}$	for column vectors
subscripted lower case letters	x_k, c_k, h_k	for vector elements
lower case Greek letters	$\alpha, \beta, \gamma, \theta$	for scalars

We may also denote a matrix $A = (a_{ij}) \in \mathbb{C}^{m\times n}$ by its columns, that is, we may choose to say $A = \begin{bmatrix} \boldsymbol{a}_1 & \boldsymbol{a}_2 & \cdots & \boldsymbol{a}_n \end{bmatrix}$. For instance, the $n \times n$ identity matrix will be denoted by I_n and we have $I_n = \begin{bmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 & \cdots & \boldsymbol{e}_n \end{bmatrix}$.

We denote the transpose of A by A^T and, if A is complex, the conjugate transpose by A^*, not A^H. A square matrix A is *symmetric* when $A = A^T$, otherwise is said to be *unsymmetric* or *nonsymmetric*. If $A = A^*$ then A is an *hermitian* matrix.

A square matrix A is *upper triangular (lower triangular)* if $a_{ij} = 0$ when $i > j$ ($i < j$). And A is said to be *diagonal* if $a_{ij} = 0$, $i \neq j$. A diagonal matrix will be written $\mathrm{diag}(a_{11}, a_{22}, \ldots, a_{nn})$ and, for $\boldsymbol{d} = (d_i) \in \mathbb{C}^n$, we define $\mathrm{diag}(\boldsymbol{d}) := \mathrm{diag}(d_1, \ldots, d_n)$.

Other definitions and notation will be introduced when needed.

Precise proofs of the results that will be presented in the following sections can be found in [51, Chapter 6], [34, Chapter 3], [21, Chapter 7], [11, Chapter 4] or [45, Chapter 12].

1.2.1 Eigenvalues

The notions of eigenvalue and eigenvector do not depend on length, angle or inner product.

Definition 1.2.1 *Let $A \in \mathbb{C}^{n\times n}$ and $\boldsymbol{x} \in \mathbb{C}^n$. Then $\boldsymbol{x}$ is a (right) eigenvector of A for the eigenvalue λ if $\boldsymbol{x}$ satisfies*

$$A\boldsymbol{x} = \lambda\boldsymbol{x}, \quad \boldsymbol{x} \neq \mathbf{0}. \tag{1.1}$$

Each nonzero multiple of $\boldsymbol{x}$ is also an eigenvector for λ and $\lambda - \mu$ is an eigenvalue of $A - \mu I$ with eigenvector $\boldsymbol{x}$, for every scalar μ.

Fact 1.2.1 *Eigenvectors corresponding to distinct eigenvalues are linearly independent.*

By the theory of linear equations, (1.1) has a non-zero solution $\boldsymbol{x}$ if and only if the *characteristic polynomial* of A, χ, defined by

$$\chi(\lambda) \equiv \chi_A(\lambda) \equiv \det(A - \lambda I),$$

verifies $\chi(\lambda) = 0$.

The roots of χ are then the eigenvalues[1] of A and are also called the *characteristic values*, *characteristic roots* or *latent roots* of A.

Polynomial χ is of degree n and its leading term is $(-1)^n\lambda^n$. So A can at most have n eigenvalues, some of which may be repeated.

Let $\lambda_1, \lambda_2, \ldots, \lambda_m$ be the distinct eigenvalues of A. Then χ can be represented as

$$\chi(\lambda) = (\lambda_1 - \lambda)^{n_1} \cdots (\lambda_m - \lambda)^{n_n},$$

where the n_i are positive integers with $\sum_{i=1}^{m} n_i = n$. The number n_i is the *algebraic multiplicity* of λ_i, $i = 1, \ldots, m$. If $n_i = 1$, then λ_i is called *simple*.

Since the characteristic polynomial of a real matrix has real coefficients, the complex eigenvalues of a real matrix occur in conjugate pairs.

[1] In German the word *eigen* means *characteristic* or *special*.

The set of the eigenvalues $\lambda_i, i = 1, \ldots, m$, in the complex plane, constitutes the *spectrum* of A and it will be denoted by $\text{spec}(A)$. We call

$$\rho(A) \equiv \max\{|\lambda| : \lambda \in \text{spec}(A)\}$$

the *spectral radius* of A.

Fact 1.2.2 $\text{spec}(A^*) = \{\bar{\lambda} : \lambda \in \text{spec}(A)\}$.

This result states that if λ is an eigenvalue of A, there is a nonzero vector $\boldsymbol{y}$ such that $A^*\boldsymbol{y} = \boldsymbol{y}\bar{\lambda}$, or, equivalently, $\boldsymbol{y}^*A = \lambda\boldsymbol{y}^*$. We say that $\boldsymbol{y}^*$ is a *row eigenvector* of A. Commonly, $\boldsymbol{y}$ is called a *left eigenvector* of A. This means that $\bar{\boldsymbol{y}}$ is a right eigenvector of A^T.

Fact 1.2.3 *For right and left eigenvectors $\boldsymbol{x}$ and $\boldsymbol{y}$ associated with a simple eigenvalue λ, $\boldsymbol{y}^*\boldsymbol{x} \neq 0$.*

Fact 1.2.4 *If λ and μ are two distinct eigenvalues with right and left eigenvectors $\boldsymbol{x}_\lambda$ and $\boldsymbol{y}_\mu$, respectively, then $\boldsymbol{y}_\mu^*\boldsymbol{x}_\lambda = 0$.*

For an eigenvalue λ with right and left eigenvectors $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively, $\{\lambda, \boldsymbol{x}\}$ is called an *eigenpair* and $\{\lambda, \boldsymbol{x}, \boldsymbol{y}^*\}$ an *eigentriple*.

1.2.2 Canonical forms

Most of the computational methods involve reducing a matrix into simpler or even *canonical* forms, from which it is easy to compute its eigenvalues and eigenvectors. These transformations are called *similarity transformations*. The two most common canonical forms are the *Jordan form* and the *Schur form*. For historical reasons we will also refer to the Frobenius canonical form. The *Schur canonical form* is the one that is more useful for pratical use.

Fact 1.2.5 *Let $B = SAS^{-1}$. Then A and B have the same eigenvalues and x is a right eigenvector of A if and only if Sx is a right eigenvector of B.*

The mapping $A \mapsto SAS^{-1}$ is a *similarity transformation* of A. If S is *unitary* (*orthogonal* in the real case), that is, $S^{-1} = S^*$ ($S^{-1} = S^T$), we say that the transformation is a *unitary (orthogonal) similarity transformation*. Matrices A and $B = SAS^{-1}$ are called *similar* matrices.

To motivate the Jordan and Schur forms, let us just recall that for a diagonal or a triangular matrix the eigenvalues are easy to compute: they are simply its diagonal entries.

Below we will see that a matrix in Jordan or Schur form is triangular.

Another particularly important form is the *quasi-triangular* form, which is a special case of the *block triangular* form. A square matrix A is *block upper triangular* if it can be partitioned in the form

$$A = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1l} \\ 0 & A_{22} & \dots & A_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A_{ll} \end{bmatrix}, \tag{1.2}$$

where each *diagonal block* A_{ii} is square. If each diagonal block is of order at most two, then A is said to be in *quasi-triangular* form. Because $\det(A - \lambda I) = \prod_{i=1}^{l} \det(A_{ii} - \lambda I)$ we have

$$\operatorname{spec}(A) = \bigcup_{i=1}^{l} \operatorname{spec}(A_{ii}).$$

Next theorem introduces *Jordan canonical form* and it is followed by some related facts that we will need in our convergence proofs in Chapter 4.

Theorem 1.2.1 JORDAN CANONICAL FORM. *Given $A \in \mathbb{C}^{n\times n}$, there exists an invertible matrix X such that $X^{-1}AX = J$, where J is in Jordan canonical form. This means that J is block diagonal, with $J = \operatorname{diag}(J_{n_1}(\lambda_1), J_{n_2}(\lambda_2), \dots, J_{n_k}(\lambda_k))$ and*

$$J_{n_i}(\lambda_i) = \begin{bmatrix} \lambda_i & 1 & & 0 \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ 0 & & & \lambda_i \end{bmatrix}_{n_i \times n_i}.$$

J is unique, up to permutations of its diagonal blocks.

There is a good deal of terminology associated with Jordan canonical form. First, each block $J_{n_i}(\lambda_i)$ is called a *Jordan block* with eigenvalue λ_i. The determinants

$$\det(\lambda I_{n_i} - J_{n_i}) = (\lambda - \lambda_i)^{n_i}$$

are called the *elementary divisors* of A. If $n_i = 1$ the elementary divisor is called *linear*.

If all the elementary divisors are linear, so that J is diagonal, A is said to be *diagonalizable* or *nondefective*; otherwise A is called *defective*.

If we denote matrix X, that reduces A to Jordan form, as

$$X = \begin{bmatrix} \boldsymbol{x}_1^{(1)} & \dots & \boldsymbol{x}_{n_1}^{(1)} & \boldsymbol{x}_1^{(2)} & \dots & \boldsymbol{x}_{n_2}^{(2)} & \dots & \boldsymbol{x}_1^{(k)} & \dots & \boldsymbol{x}_{n_k}^{(k)} \end{bmatrix}$$

then, for each Jordan block $J_{n_i}(\lambda_i)$, the corresponding vectors $\boldsymbol{x}_1^{(i)}, \dots, \boldsymbol{x}_{n_i}^{(i)}$, $i = 1, \dots, k$, satisfy

$$\begin{aligned} A\boldsymbol{x}_1^{(i)} &= \lambda_i \boldsymbol{x}_1^{(i)} \\ A\boldsymbol{x}_{j+1}^{(i)} &= \lambda_i \boldsymbol{x}_{j+1}^{(i)} + \boldsymbol{x}_j^{(i)}, \qquad j = 1, \dots, n_i - 1. \end{aligned}$$

Vectors $x_{j+1}^{(i)}$, $j = 1, \dots, n_i - 1$ are called *generalized eigenvectors* or *principal vectors* of *grade* $j + 1$, verifying

$$(A - \lambda_i I)^{j+1} \boldsymbol{x}_{j+1}^{(i)} = 0, \qquad (A - \lambda_i I)^j \boldsymbol{x}_{j+1}^{(i)} = \boldsymbol{x}_1^{(i)} \neq \boldsymbol{0}.$$

Fact 1.2.6 *If all the eigenvalues of a matrix A are distinct, then A is diagonalizable.*

A defective matrix does not have n linearly independent eigenvectors.

Fact 1.2.7 *A Jordan block has one right eigenvector* $\boldsymbol{e}_1 = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T$ *and one left eigenvector* $\boldsymbol{e}_n = \begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix}^T$.

The number of linearly independent eigenvectors associated with an eigenvalue λ is called the *geometric multiplicity* of λ and does not exceed its algebraic multiplicity. It corresponds to the number of Jordan blocks associated with λ.

A matrix is defective if and only if the geometric multiplicity of at least one of its eigenvalues is less than the algebraic multiplicity - in this case, commonly, we call these eigenvalues briefly as *multiple* eigenvalues.

A matrix A is *nonderogatory* if every eigenvalue has geometric multiplicity 1. This means that different Jordan blocks J_{n_i} correspond to distinct λ_i.

The (i, j) *minor* of a square matrix A is defined as the determinant of the submatrix obtained by removing the i^{th} row and the j^{th} column of A. The k^{th} *leading principal minor* is the determinant of the first k rows and k columns of A.

The *Companion* matrix of a *monic* polynomial

$$p(\lambda) \equiv \lambda^n + a_{n-1}\lambda^{n-1} + \ldots + a_1\lambda + a_0$$

is the matrix

$$C_p := \begin{bmatrix} 0 & & & & -a_0 \\ 1 & 0 & & & -a_1 \\ & \ddots & \ddots & & \vdots \\ & & \ddots & 0 & -a_{n-2} \\ & & & 1 & -a_{n-1} \end{bmatrix}.$$

Since $p(\lambda) = (-1)^n \det(C_p - \lambda I) = \det(\lambda I - C_p)$, the zeros of the polynomial p are the eigenvalues of C_p. It is easy to see that C_p is a nonderogatory matrix: for all λ, the $(1, n)$ minor of $C_p - \lambda I$ is always 1; consequently $\operatorname{rank}(C_p - \lambda I) \geq n - 1$ and the dimension of the null space of $C_p - \lambda I$ is either 0 (when λ is not an eigenvalue) or 1 (when λ is an eigenvalue); this means that there is only one linearly independent eigenvector to each eigenvalue.

Fact 1.2.8 *A matrix A is similar to the companion matrix of its characteristic polynomial if and only if A is nonderogatory.*

A $m \times n$ matrix $R = (r_{ij}(x))$ whose elements r_{ij} are rational functions, that is, ratios of polynomials, is a *rational* matrix.

Theorem 1.2.2 FROBENIUS, OR RATIONAL, CANONICAL FORM *Given $A \in \mathbb{C}^{n\times n}$, there exists an invertible matrix P such that $P^{-1}AP = F$, where F is in Frobenius canonical form. This means that F is the direct sum of the companion matrices of the elementary divisors of $\lambda I_n - A$.*

If $A \in \mathbb{R}^{n\times n}$, the Frobenius canonical form of A depends on which field ($\mathbb{R}$ or $\mathbb{C}$) over which it is considered.

This form does not say much about eigenvalues but it is, by construction, the matrix with fewest nonzero entries that can be achieved by rational operations on the data, but it is too condensed. The Jordan canonical form tells us all we want to know about eigenvalues but can not be computed stably in general. So, it is used in theory but is very hard to compute because it is very unstable in the face of uncertainty. Thus, these two forms are not used in eigenvalue computations and most of the algorithms will aim to compute the Schur form instead.

Theorem 1.2.3 SCHUR CANONICAL FORM. *Given $A \in \mathbb{C}^{n\times n}$, there exists a unitary matrix Q and an upper triangular matrix T such that $Q^*AQ = T$.*

A matrix A is said to be *normal* if $AA^* = A^*A$.

Fact 1.2.9 *A matrix A is normal if and only if there exists a unitary matrix Q such that $Q^*AQ = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$.*

As said before, a real matrix A can have complex eigenvalues and, therefore, there is not always a real triangular matrix with the same eigenvalues as A. So, we must either use complex numbers or sacrifice the triangular canonical form. Because it will be cheaper to compute, we prefer a canonical form that uses only real numbers and we will settle for a quasi-triangular form.

Theorem 1.2.4 REAL SCHUR CANONICAL FORM. *If $A \in \mathbb{R}^{n\times n}$, there exists a real orthogonal matrix V such that $V^TAV = T$ is real quasi-upper triangular. Its eigenvalues are the eigenvalues of its diagonal blocks. The 1×1 blocks corresponds to real eigenvalues and the 2×2 blocks to complex conjugate pairs of eigenvalues.*

1.2.3 Perturbation theory

In what follows we will discuss how changes in the entries of a matrix A affect the spectrum. If λ is a simple eigenvalue of A, for a given matrix δA, we can identify an eigenvalue $\lambda + \delta\lambda$ of the perturbed $A + \delta A$ *corresponding* to λ: the closest one to λ.

So, we want to relate the size of the matrix perturbation to the size of the eigenvalue change. The norms most widely used in matrix computations are the $1-$norm, the Frobenius norm and the ∞-norm. Unlike the $2-$norm (an example of a $p-$norm and also of great importance) the former matrix norms are easy to calculate. And for a consistent matrix norm $\|.\|$ we have, for any matrix A of order n,

$$\rho(A) \leq \|A\| .$$

For a real matrix A and a matrix norm $\|.\|$, the quantity

$$\mathrm{cond}(A) \equiv \|A\| \left\|A^{-1}\right\|$$

defines the *condition number* of the matrix A. Note that $\mathrm{cond}(A) = \mathrm{cond}(\tau A)$ for any scalar $\tau \neq 0$, that is, cond is scaling invariant by multiplication.

Next theorem relates eigenvalue condition numbers to the condition number of the matrix of all eigenvectors.

Theorem 1.2.5 BAUER-FIKE (CLASSICAL VERSION). *If μ is an eigenvalue of $A + \delta A \in \mathbb{C}^{n\times n}$ and $X^{-1}AX = D = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, then*

$$\min_{\lambda\in \mathrm{spec}(A)} |\lambda - \mu| \leq \mathrm{cond}_p(X) \left\|\delta A\right\|_p$$

where $\|.\|_p$ denotes any of the p-norms.

Extreme eigenvalue sensitivity for a matrix A cannot occur if A is normal. On the other hand, nonnormality does not necessarily imply eigenvalue sensitivity. Indeed, a nonnormal matrix can have a mixture of well-conditioned and ill-conditioned eigenvalues. For this reason, it is beneficial to refine the perturbation theory so that it is applicable to individual eigenvalues and not the spectrum as a whole.

Theorem 1.2.6 *Let λ be a simple eigenvalue of A with right eigenvalue $\boldsymbol{x}$ and left eigenvalue $\boldsymbol{y}$, normalized so that $\|\boldsymbol{x}\|_2 = \|\boldsymbol{y}\|_2 = 1$. Let $\lambda + \delta\lambda$ be the corresponding eigenvalue of $A + \delta A$. Then*

$$|\delta\lambda| \leq \frac{\|\delta A\|}{|\boldsymbol{y}^*\boldsymbol{x}|} + \mathcal{O}(\|\delta A\|)^2 = \frac{\|\delta A\|}{\cos\big(\theta(\boldsymbol{y},\boldsymbol{x})\big)} + \mathcal{O}(\|\delta A\|)^2$$

where $\theta(\boldsymbol{y},\boldsymbol{x})$ is the acute angle between y and x. In other words, $\kappa_\lambda \equiv \dfrac{1}{|\boldsymbol{y}^*\boldsymbol{x}|}$ is the condition number of the eigenvalue λ. It is known as Wilkinson's condition number.

If the eigenvectors are not normalized the condition number κ_λ is given by

$$\kappa_\lambda = \frac{\|\boldsymbol{x}\|\,\|\boldsymbol{y}\|}{|\boldsymbol{y}^*\boldsymbol{x}|}.$$

Each eigenvalue has its own condition number. Some eigenvalues can be well-conditioned and some can be ill-conditioned. A big condition number means sensitivity; ill-conditioned eigenvalues are hard to compute accurately because they are not well-defined.

The right and left eigenvectors of a Jordan Block are orthogonal, and so Wilkinson's condition number of its eigenvalue λ will be $\kappa_\lambda = \infty$. At the other extreme are the normal matrices which have condition number $\kappa_\lambda = 1$. So, the symmetric eigenvalue problem is always a well-conditioned problem.

Theorem 1.2.6 is useful only for sufficiently small $\|\delta A\|$. Next theorem increases the condition number by a factor on n but is true for any modification δA.

Theorem 1.2.7 BAUER-FIKE. *Let A have all simple eigenvalues, i.e., be diagonalizable. Call these eigenvalues λ_i, $i = 1, \ldots, n$, and let $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ be the right and left eigenvectors for λ_i, respectively, normalized so that $\|\boldsymbol{x}_i\|_2 = \|\boldsymbol{y}_i\|_2 = 1$. Then the eigenvalues of $A + \delta A$ lie in disks B_i, where B_i has center λ_i and radius $\dfrac{n}{|\boldsymbol{y}_i^* \boldsymbol{x}_i|} \|\delta A\|_2$.*

More generally, we can say that

$$\forall \tilde{\lambda} \in \operatorname{spec}(A + \delta A): \quad \min_i |\tilde{\lambda} - \lambda_i| \leq n \max_i \frac{1}{|\boldsymbol{y}_i^* \boldsymbol{x}_i|} \|\delta A\|_2 .$$

The proof of this theorem uses a useful inclusion result for eigenvalues, the *Gerschgorin* theorem, which we write below.

Theorem 1.2.8 GERSCHGORIN. *For a matrix $A = (a_{ij}) \in \mathbb{C}^{n \times n}$ it holds that*

$$\operatorname{spec}(A) \subset \bigcup_{i=1}^{n} \mathcal{D}_i \quad \textit{with} \quad \mathcal{D}_i := \left\{ \lambda \in \mathbb{C}: \ |\lambda - a_{ii}| \leq \sum_{j \neq i} |a_{ij}| \right\}, \ \ i = 1, \ldots, n.$$

Disks $\mathcal{D}_i$ are called Gerschgorin disks of the matrix A.

The proof of the theorem shows not only that each eigenvalue of A must lie in a Gerschgorin disk, but also that if the ith component of an eigenvector is maximal, then the corresponding eigenvalue must lie in the ith disk.

Theorem 1.2.9 *If k Gerschgorin disks of the matrix A are disjoint from the other disks, then exactly k eigenvalues of A lie in the union of the k disks.*

Since a multiple eigenvalue has infinite Wilkinson's condition number, being close to a matrix which has multiple eigenvalues implies ill-conditioning. The bigger is the condition number of an eigenvalue, the closer is the matrix to one with a multiple eigenvalue.

Theorem 1.2.10 *Let λ be a simple eigenvalue of A, with unit right and left eigenvectors $\boldsymbol{x}$ and $\boldsymbol{y}$ and condition number $c = \dfrac{1}{|\boldsymbol{y}^* \boldsymbol{x}|}$. Then there is a δA such that $A + \delta A$ has a multiple eigenvalue at λ and*

$$\frac{\|\delta A\|_2}{\|A\|_2} \leq \frac{1}{\sqrt{c^2 - 1}}.$$

When $c \gg 1$, that is the eigenvalue is ill-conditioned, then the upper bound on the distance is $\dfrac{1}{\sqrt{c^2-1}} \approx \dfrac{1}{c}$, the reciprocal of the condition number.

Finally, we relate the condition numbers of the eigenvalues to the smallest possible *condition number* $\mathrm{cond}(S)$ of any S that diagonalizes A.

Theorem 1.2.11 *Let A be diagonalizable with eigenvalues λ_i and right and left eigenvectors $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$, respectively, normalized so that $\|\boldsymbol{x}_i\|_2 = \|\boldsymbol{y}_i\| = 1$. Let us suppose that S satisfies $S^{-1}AS = \Delta = diag(\lambda_1, \ldots, \lambda_n)$. Then*

$$\max_i \frac{1}{|\boldsymbol{y}_i^* \boldsymbol{x}_i|} \leq \|S\|_2 \left\|S^{-1}\right\|_2 .$$

If we choose $S = \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_n \end{bmatrix}$ then

$$\|S\|_2 \left\|S^{-1}\right\|_2 \leq n \max_i \frac{1}{|\boldsymbol{y}_i^* \boldsymbol{x}_i|},$$

that is the condition number of S is within a factor of n of its smallest value.

1.2.4 Relative errors and model of arithmetic

Let $\hat{\sigma}$ be an approximation to a real number σ. The most useful measures of the accuracy of $\hat{\sigma}$ is the *absolute error*

$$E_{\mathrm{abs}}(\hat{\sigma}) = |\sigma - \hat{\sigma}|,$$

and its *relative error*

$$E_{\mathrm{rel}}(\hat{\sigma}) = \frac{|\sigma - \hat{\sigma}|}{|\sigma|}, \qquad \sigma \neq 0.$$

Writing

$$\hat{\sigma} = \sigma(1 + \rho),$$

an equivalent definition of relative error will be $E_{\mathrm{rel}}(\hat{\sigma}) = |\rho|$. Some authors omit the absolute values from these definitions.

In scientific computation, where answers to problems can vary enormously in magnitude, it is usually the relative error that is of interest, because it is scale independent: scaling $\sigma \to \alpha\sigma$ and $\hat{\sigma} \to \alpha\hat{\sigma}$ leaves $E_{\mathrm{rel}}(\hat{\sigma})$ unchanged.

Relative error is connected with the notion of correct significant digits. But while the number of correct significant digits provides a useful way in which to think about the accuracy of an approximation, the relative error is a more precise measure.

Floating point arithmetic

The *maximum relative representation error* in a floating point arithmetic with p digits and base β is $\frac{1}{2}\beta^{1-p}$. When the true value of a computation $a \odot b$ (where $\odot$ is one of the four binary operations $+$, $-$, $*$ and $/$) cannot be represented exactly as a floating point number, it must be approximated by a nearby floating point number. We denote this approximation by $fl(a \odot b)$. The difference $(a \odot b) - fl(a \odot b)$ is called the *roundoff error*. If $a \odot b$ is within the exponent range (otherwise we get *overflow* or *underflow*), then we assume the model of arithmetic

$$fl(a \odot b) = (a \odot b)(1 + \delta), \qquad |\delta| \leq \varepsilon. \tag{1.3}$$

The quantity ε is called variously *unit roundoff*, *machine precision* or *macheps*. If we round as accurately as possible, ε is equal to the maximum relative representation error $\frac{1}{2}\beta^{1-p}$.

The terms accuracy and precision are often confused or used interchangeable, but it is worth making a distinction between them. *Accuracy* refers to the absolute or relative error of an approximate quantity. *Precision* is the accuracy with which the basic arithmetic operations $+, -, *, /$ are performed and for floating point arithmetic is measured by the unit roundoff (1.3).

The IEEE *standard for binary arithmetic* is now common. It includes two kinds of floating point numbers: *single precision* (32 bits long) and *double precision* (64 bits long) that allow 24 and 53 p bits of precision, respectively.

Normally the "big oh" notation $\mathcal{O}$ implies a limiting process. However, in this thesis $\mathcal{O}$ will be a synonym for "of the order of magnitude of" and the usage will be clear from the context. For example, $\mathcal{O}(\varepsilon)$ will mean "of the order of magnitude of machine precision".

1.3 The unsymmetric tridiagonal eigenvalue problem

Tridiagonal matrices have received a great deal of attention since the 1950's. In the symmetric case, every matrix can be stably reduced to a similar tridiagonal matrix by a finite number of elementary orthogonal similarity transformations. In the unsymmetric case, reduction to tridiagonal form is also possible but we have to use non-orthogonal similarities transformations and the reduction may not be stable.

But why should we be interested in tridiagonal matrices? There is a need for eigenvalue methods capable of exploiting and respecting the elegant structure of an unsymmetric tridiagonal matrix so that the development of methods for tridiagonal reduction could be encouraged.

Unsymmetric tridiagonal matrices arise naturally as a result of the execution of the two-sided Lanczos algorithm (see, for instance, Parlett [39]). When applied to a matrix, the two-sided Lanczos algorithm builds a $k \times k$ unsymmetric tridiagonal matrix at the end of the Lanczos step k. This is a candidate method for the reduction of a nonsymmetric matrix to tridiagonal form but in practice it is confined to large sparse matrices. Other methods to perform this task have been proposed in Geist [20], Dongarra [13] and, more recently, in Sidje [50].

Tridiagonals also appear as primary data. For example, they are related to orthogonal polynomials because there is a remarkable three-term recurrence relation among these polynomials. In particular, there are the special *Bessel* polynomials, which also satisfy a three-term recurrence relation. Tridiagonals also arise in other numerical methods such as exponential interpolation [1].

The eigenvalues and eigenvectors of a real nonsymmetric matrix A traditionally have been computed by first reducing A to Hessenberg form H and then computing the eigendecomposition of H by the QR method. But the approach of beginning by reducing the initial matrix to nonsymmetric tridiagonal form is attractive because finding eigenvalues of a tridiagonal matrix is much faster than for a Hessenberg matrix. So, the development of an algorithm for the nonsymmetric tridiagonal eigenvalue problem is therefore a major

topic of research.

In contrast to the symmetric eigenproblem, that is always well-conditioned, the unsymmetric eigenproblem can be effectively very ill-posed. However, there is also a need for a careful study of the sensitivity of eigenvalues of tridiagonals to perturbations.

1.3.1 Orthogonal polynomials

From [38, Chapter 7]. Important in applied mathematics are real functions ϕ, ψ, ... of one real variable and, in particular, the set $\mathcal{P}_n$ of polynomials of degree not exceeding n. We shall not consider the general integral inner products

$$(\phi, \psi) \equiv \int_a^b \omega(x)\phi(x)\psi(x)\, dx$$

but go straight to the discrete case

$$(\phi, \psi) \equiv \sum_{i=1}^{n} \omega_i \phi(\xi_i)\psi(\xi_i). \tag{1.4}$$

To each set of n distinct real numbers $\{\xi_1, \ldots, \xi_n\}$ and possible weights $\{\omega_1, \ldots, \omega_n : \ \omega_i > 0\}$ there corresponds one, and only one, inner product function as defined by (1.4).

Polynomials are rather special functions and for each inner product there is a unique family of monic orthogonal polynomials $\{\phi_0, \phi_1, \ldots, \phi_{n-1}\}$; that is, ϕ_j has degree j, leading coefficient 1 and $(\phi_j, \phi_k) = 0$ for $j \neq k$. This family is the distinguished basis of the inner product space $\mathcal{P}_{n-1}$ enriched with the given inner product.

Tridiagonal matrices come into the picture because there is a remarkable three-term recurrence relation among the $\phi's$; for $j = 1, 2, \ldots, n-1$ and $\beta_0 = 0$,

$$\phi_j(\xi) = (\xi - \alpha_j)\phi_{j-1}(\xi) - \beta_{j-1}^2 \phi_{j-2}(\xi).$$

Once such a relationship has been guessed, it is straightforward to verify what the α's and β's must be

$$\begin{aligned} \alpha_{j+1} &= (\eta\phi_j, \phi_j)/(\phi_j, \phi_j), \qquad j = 0, \ldots, n-1 \\ \beta_j^2 &= (\eta\phi_{j-1}, \phi_j)/(\phi_{j-1}, \phi_{j-1}), \qquad j = 1, \ldots, n-1, \end{aligned}$$

where η denotes the identity function $\eta(\xi) \equiv \xi$. These numbers may be put into an unreduced symmetric tridiagonal matrix T in the obvious way,

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{n-2} & \alpha_{n-1} & \beta_{n-1} \\ & & & \beta_{n-1} & \alpha_n \end{bmatrix},$$

and then, for $j = 1, \ldots, n$, we have

$$\phi_j(\xi) = \det(\xi I_j - T_j)$$

where I_j denotes the identity matrix or order j and T_j denotes the j^{th} *principal submatrix* of T, that is, T_j is the submatrix obtained with the the first j rows and j columns of T. Thus the $\xi's$ and the $\omega's$ determine the unique T.

The question we pose now is how to determine the ξ's and the $\omega's$ from a given T. In other words, which inner products make the ϕ_j, $j = 0, 1, \ldots, n$, mutually orthogonal?

Theorem 1.3.1 *Let $T = S\Lambda S^*$ be the spectral decomposition of an unreduced symmetric T with $S = \begin{bmatrix} \boldsymbol{s}_1 & \boldsymbol{s}_2 & \cdots & \boldsymbol{s}_n \end{bmatrix}$ and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Then the associated inner product of the form (1.4) is given by*

$$\xi_i = \lambda_i, \qquad \omega_i = \gamma s_{i1}^2, \quad i = 1, \ldots, n,$$

for any positive γ. And $\gamma = \sum_1^n \omega_i$.

It is customary to take $\gamma = 1$.

The question "how much of this result extends to the unsymmetric case?" has been studied under the tittle "formal orthogonal polynomials", but that is outside the scope of this thesis. See [26].

1.3.2 The ordinary and generalized Bessel polynomials

The definition of the generalized Bessel polynomials (GBP) is given in terms of the associated family of differential equations

$$z^2 u'' + (az + b)u' - n(n + a - 1)u = 0, \qquad n \in \mathbb{N}, \;\; a, b \in \mathbb{C}. \tag{1.5}$$

It is known [44] that nontrivial polynomial solutions of equation (1.5) exist for every $n \in \mathbb{N}$ and $a, b \in \mathbb{C}$. If $b \neq 0$, there exists a unique polynomial solution $u_n^{(a,b)}$ that satisfies the condition $u_n^{(a,b)}(0) = 1$. The polynomials $u_n^{(a,b)}$ defined in this way are precisely the *generalized Bessel polynomials*. They can be represented in the form

$$u_n^{(a,b)}(z) = \sum_{k=0}^{n} c_{n,k}^{(a,b)} z^k, \tag{1.6}$$

where

$$c_{n,k}^{(a,b)} := \binom{n}{k} \frac{(n + k + a - 2)^{[k]}}{b^k}, \;\; k = 0, 1, \ldots, n, \tag{1.7}$$

with

$$(x)^{[0]} := 1, \;\; (x)^{[k]} := x(x - 1) \cdots (x - k + 1), \quad k \in \mathbb{N}.$$

Moreover, it can be shown that they satisfy the following recurrence relations,

$$(n + a - 1)(2n + a - 2)u_{n+1}^{(a,b)}(z) = \left[(2n + a)(2n + a - 2)\frac{z}{b} + a - 2\right] . \\ .(2n + a - 1)u_n^{(a,b)}(z) + n(2n + a)u_{n-1}^{(a,b)}(z) \tag{1.8}$$

$$z^2(2n + a - 2)u_n^{(a,b)\prime}(z) = [n(2n + a - 2)z - bn]\, u_n^{(a,b)}(z) + bnu_{n-1}^{(a,b)}(z). \tag{1.9}$$

An important result is that for every $n \in \mathbb{N}$, $a \in \mathbb{C}$ and $b \in \mathbb{C}\backslash\{0\}$, the GBP $u_n^{(a,b)}$ has only simple zeros.

Formulas (1.7) and (1.8) show that the constant b is a scaling factor and almost all authors assume $b = 2$; and the case $a \in \mathbb{R}$ is the most investigated one in literature. Taking $a = b = 2$ in (1.5) and (1.6) leads to the particular case of the *ordinary Bessel polynomials*.

Let $z_{ni}^{(a,b)}$ be the zeros of the GBP's $u_n^{(a,b)}$ and consider the problem of computing these zeros.

Matrices whose eigenvalues are the zeros $z_{ni}^{(a,b)}$ can be derived either from the coefficients $c_{n,k}^{(a,b)}$ of the polynomials $u_n^{(a,b)}$ in (1.6), or from the three-term recurrence relation (1.8). The former procedure leads to the so-called Companion matrices which are Hessenberg matrices. The latter procedure is the classical one, usually adopted to compute the zeros of orthogonal polynomials. The matrix one gets this way is tridiagonal and if denoted by $B_n^{(a,b)}$, we have

$$B_n^{(a,b)} = \begin{bmatrix} \alpha_1^{(a,b)} & \gamma_1^{(a,b)} & & & \\ \beta_1^{(a,b)} & \alpha_2^{(a,b)} & \gamma_2^{(a,b)} & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{n-2}^{(a,b)} & \alpha_{n-1}^{(a,b)} & \gamma_{n-1}^{(a,b)} \\ & & & \beta_{n-1}^{(a,b)} & \alpha_n^{(a,b)} \end{bmatrix},$$

where

$$\alpha_1^{(a,b)} := -\frac{a}{b}, \quad \gamma_1^{(a,b)} := -\alpha_1^{(a,b)}, \quad \beta_1^{(a,b)} := \frac{\alpha_1^{(a,b)}}{a+1},$$

and

$$\begin{aligned} \alpha_j^{(a,b)} &:= -b\frac{a-2}{(2j+a-2)(2j+a-4)}, \quad j = 2, \ldots, n, \\ \gamma_j^{(a,b)} &:= b\frac{j+a-2}{(2j+a-2)(2j+a-3)}, \\ \beta_j^{(a,b)} &:= -b\frac{j}{(2j+a-1)(2j+a-2)}, \quad j = 2, \ldots, n-1. \end{aligned}$$

Several methods can be adopted to calculate the eigenvalues of $B_n^{(a,b)}$.

Chapter 2

Representations and measures of sensitivity

This chapter devotes attention to the unsymmetric tridiagonal eigenvalue problem beginning with various representations for tridiagonal matrices. Our first contribution is to provide new measures of eigenvalue sensitivity that exploit the tridiagonal form.

2.1 LU factorization

From Higham [27, Chapter 9]. Much insight into *Gaussian Elimination* (GE) is obtained by expressing it in matrix notation. The strategy of GE is to reduce the initial matrix $A^1 := A$ to an upper triangular matrix using elementary row operations. Let $A^k = (a_{ij}^k)$, $i, j = 1, \ldots, n$, denote the matrix obtained in the k^{th} stage of GE. The purpose of the kth stage of the elimination is to annihilate the elements below the diagonal in the kth column of A^k. This is accomplished by the operations

$$a_{ij}^{k+1} = a_{ij}^k - m_{ik} a_{kj}^k, \quad i = k+1, \ldots, n, \; j = k+1, \ldots, n$$

where $m_{ik} = a_{ik}^k / a_{kk}^k$, $i = k+1, \ldots, n$, are the so-called multipliers. At the end of the $(n-1)$st stage A^{n-1} is upper triangular.

We can write

$$A^{k+1} = M_k A^k := \begin{bmatrix} I_{k-1} & & & & \\ & 1 & & & \\ & -m_{k+1,k} & 1 & & \\ & -m_{k+2,k} & & \ddots & \\ & \vdots & & & \ddots \\ & -m_{n,k} & & & 1 \end{bmatrix} A^k.$$

The matrix M_k can be expressed compactly as $M_k = I - \boldsymbol{m}_k \boldsymbol{e}_k^T$, where $\boldsymbol{e}_k$ is the kth unit vector and $\boldsymbol{m}_k = \begin{bmatrix} 0 & \dots & 0 & m_{k+1,k} & \dots & m_{nk} \end{bmatrix}^T$. We have $\boldsymbol{e}_i^T \boldsymbol{m}_k = 0$ for $i \leq k$. To invert M_k, just flip the signs of the multipliers: $M_k^{-1} = I + \boldsymbol{m}_k \boldsymbol{e}_k$. Overall,

$$M_{n-1} M_{n-2} \dots M_1 A = A^n =: U,$$

and so

$$\begin{aligned} A =& M_1^{-1} M_2^{-1} \dots M_{n-1}^{-1} U \\ =& (I + \boldsymbol{m}_1 \boldsymbol{e}_1^T)(I + \boldsymbol{m}_2 \boldsymbol{e}_2^T) \dots (I + \boldsymbol{m}_{n-1} \boldsymbol{e}_{n-1}^T) U \\ =& \left(I + \sum_{i=1}^{n-1} \boldsymbol{m}_i \boldsymbol{e}_i^T \right) U \\ =& \begin{bmatrix} 1 & & & & \\ m_{2,1} & 1 & & & \\ m_{3,1} & m_{3,2} & \ddots & & \\ \vdots & \vdots & & \ddots & \\ m_{n,1} & m_{n,2} & \dots & m_{n,n-1} & 1 \end{bmatrix} U =: LU. \end{aligned}$$

The conclusion is that GE computes an LU factorization of A : $A = LU$, where L is unit lower triangular and U is upper triangular.

There are two problems with the method described. First, there is a break-down with division by zero if $a_{kk}^k = 0$. Second, if we are working in finite precision and some multiplier

m_{ik} is large, there is a possible loss of significance: in the subtraction $a_{ij}^k - m_{ik}a_{kj}^k$, low-order digits of a_{ij}^{k+1} could be lost. These observations motivate the strategy of *partial pivoting*.

Let A_k denote the k^{th} *leading principal submatrix* of A, that is A_k is the submatrix of the first k rows and first k columns of A.

Theorem 2.1.1 *There exists a unique LU factorization of $A \in \mathbb{R}^{n\times n}$ if and only if A_k is nonsingular for $k = 1, \ldots, n-1$. If A_k is singular for some $1 \le k \le n-1$ the factorization may exist, but if so it is not unique.*

2.1.1 LU and LDU factorizations of a tridiagonal

A $n \times n$ matrix $H = (h_{ij})$ is said to be *upper Hessenberg (lower Hessenberg)* if $h_{ij} = 0$ when $i > j+1$ ($i < j-1$). It is *unreduced*, or *irreducible*, if $h_{i,i-1} \neq 0$ ($h_{i-1,i} \neq 0$), $i = 2, \ldots, n$. And H is said to be *tridiagonal* if it is both upper and lower Hessenberg, that is, $h_{ij} = 0$ when $|i-j| > 1$. A tridiagonal matrix is *unreduced* if $h_{i,i-1}h_{i-1,i} \neq 0$, $i = 2, \ldots, n$.

Let C be a real nonsymmetric tridiagonal matrix,

$$C = \begin{bmatrix} a_1 & c_1 & & & \\ b_1 & a_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2} & a_{n-1} & c_{n-1} \\ & & & b_{n-1} & a_n \end{bmatrix} \in \mathbb{R}^{n\times n}. \tag{2.1}$$

Whenever convenient we will adopte the notation

$$C = \text{tridiag}(\boldsymbol{b}, \boldsymbol{a}, \boldsymbol{c}),$$

where $\boldsymbol{a} = (a_k)$, $k = 1, \ldots, n$, and $\boldsymbol{b} = (b_k)$, $\boldsymbol{c} = (c_k)$, $k = 1, \ldots n-1$.

Assume that C has an LU factorization $C = LU$. Next we will consider details of this factorization as well as the LDU factorization.

LU factorization

A matrix $B = (b_{ij})$ is said to be *upper bidiagonal (lower bidiagonal)* if $b_{ij} = 0$ for $i \neq j$ and $i \neq j-1$ ($i \neq j+1$).

Assuming there are no break-downs, matrices L and U of the factorization of C are lower bidiagonal and upper bidiagonal, respectively, and will be given by

$$L = \begin{bmatrix} 1 & & & & & \\ l_1 & 1 & & & & \\ & l_2 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & l_{n-2} & 1 & \\ & & & & l_{n-1} & 1 \end{bmatrix}, \qquad U = \begin{bmatrix} u_1 & c_1 & & & & \\ & u_2 & c_2 & & & \\ & & u_3 & c_3 & & \\ & & & \ddots & \ddots & \\ & & & & u_{n-1} & c_{n-1} \\ & & & & & u_n \end{bmatrix}. \quad (2.2)$$

GE for computing L and U is described by the recurrence relations

$$u_1 = a_1, \qquad \left.\begin{array}{l} l_{i-1} = b_{i-1}/u_{i-1} \\ u_i = a_i - l_{i-1}c_{i-1} \end{array}\right\} \; i = 2, \ldots, n. \quad (2.3)$$

For $A = (a_{ij})$, let $|A|$ denote the matrix whose entries are $|a_{ij}|$. An error analysis says that, in matrix terms, the computed factors $\tilde{L}$ and $\tilde{U}$ satisfy

$$A = \tilde{L}\tilde{U} + \delta A, \;\; |\delta A| \leq \varepsilon |\tilde{L}||\tilde{U}|,$$

where ε is the unit roundoff. See Higham [27, p.174]

LDU factorization

Assume that $u_i \neq 0,\ i = 1, \ldots, n-1$. Then the factorization $C = LDU$ is immediate: in (2.2) leave L unchanged, define $D := \mathrm{diag}(d_1, \ldots, d_n)$ with $\boldsymbol{d} = (u_i)$, and redefine

$$U = \begin{bmatrix} 1 & \frac{c_1}{u_1} & & & & \\ & 1 & \frac{c_2}{u_2} & & & \\ & & 1 & \frac{c_3}{u_3} & & \\ & & & \ddots & \ddots & \\ & & & & 1 & \frac{c_{n-1}}{u_{n-1}} \\ & & & & & 1 \end{bmatrix} \tag{2.4}$$

If C is a symmetric matrix we must have $U = L^T$ and thus $C = LDL^T$.

2.2 Representations of tridiagonals

One of the powerful ideas in matrix computations is factorization. And the power of methods come from representations, especially in the tridiagonal case. Naturally, we can represent a tridiagonal matrix by exhibiting explicitly its entries. Alternatively, we may decompose the matrix into a product of certain factors and give the entries of these factors.

2.2.1 Normalization

Consider a tridiagonal matrix $C = \mathrm{tridiag}(\boldsymbol{b}, \boldsymbol{a}, \boldsymbol{c})$ as given in (2.1). If for some k, c_k (or b_k) is zero then C can be split into two tridiagonal matrices, say C_1 of order k and C_2 or order $n-k$. The eigenvalues and eigenvectors of C can be recovered from those of C_1 and C_2 and in this way the computation may be reduced to finding the eigenvalues and eigenvectors of smaller tridiagonal submatrices which are unreduced. So, there is no loss of generality in confining attention to the unreduced case, that is, when $b_k c_k \neq 0$, $k = 1, \ldots, n-1$.

If C is unreduced, the nullity (dimension of the null space) of $C - \lambda I$ can not exceed 1. Notice that the *minor* of the $(1, n)$ element of $C - \lambda I$ is $b_1 b_2 \dots b_{n-1} \neq 0$ and, consequently, $\text{rank}(C - \lambda I) \geq n - 1$. So, the dimension of the null space of $C - \lambda I$ is either 0 (when λ is not an eigenvalue) or 1 (when λ is an eigenvalue).

Fact 2.2.1 *Unreduced tridiagonal matrices are nonderogatory.*

It is known that the eigenvalues are determined by the diagonal elements and the products of the off-diagonal elements.

In the eigenvalue computation context there is no loss of generality in supposing that tridiagonals are normalized so that all entries in positions $(i, i+1)$ are 1. So we may assume that $c_k = 1$, $k = 1, \dots, n-1$. In fact,

Lemma 2.2.1 *Any tridiagonal matrix C that does not split (unreduced) is diagonally similar to a form with 1's above the diagonal.*

Proof. If we consider the diagonal matrix

$$D = \text{diag}(1, c_1, c_1 c_2, \dots, c_1 \cdots c_{n-1}),$$

then the similar matrix DTD^{-1} has the form

$$\begin{bmatrix} a_1 & 1 & & & \\ b_1 c_1 & a_2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2} c_{n-2} & a_{n-1} & 1 \\ & & & b_{n-1} c_{n-1} & a_n \end{bmatrix}. \square$$

Such tridiagonals will be denoted by J. Lets say we will always have

$$J = \begin{bmatrix} a_1 & 1 & & & \\ b_1 & a_2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2} & a_{n-1} & 1 \\ & & & b_{n-1} & a_n \end{bmatrix}. \tag{2.5}$$

This is called the qd form.

2.2.2 Products of bidiagonals

Most, but not all, such J permit triangular factorization

$$J = LU$$

where L and U are bidiagonal matrices of a special form that will play a leading role. Precise forms are shown below: L is a lower bidiagonal matrix with 1's in positions (i, i) and U is an upper bidiagonal matrix with 1's in positions $(i, i+1)$,

$$L = \begin{bmatrix} 1 & & & & & \\ l_1 & 1 & & & & \\ & l_2 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & l_{n-2} & 1 & \\ & & & & l_{n-1} & 1 \end{bmatrix}, \qquad U = \begin{bmatrix} u_1 & 1 & & & & \\ & u_2 & 1 & & & \\ & & u_3 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & u_{n-1} & 1 \\ & & & & & u_n \end{bmatrix}.$$

The attractive feature here is that, in the $n \times n$ case, because the 1's need not be represented explicitly the factored form of J requires no more storage than J itself: L and U together are defined by $2n-1$ parameters, exactly the same degree of freedom as in J. Later we will argue that the pair L, U is preferable to J itself.

The pair L,U determines two triangular matrices: first

$$J = LU = \begin{bmatrix} u_1 & 1 & & & & \\ l_1u_1 & l_1+u_2 & 1 & & & \\ & l_2u_2 & l_2+u_3 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & l_{n-2}u_{n-2} & l_{n-2}+u_{n-1} & 1 \\ & & & & l_{n-1}u_{n-1} & l_{n-1}+u_n \end{bmatrix},$$

and second

$$J' = UL = \begin{bmatrix} u_1 + l_1 & 1 & & & & \\ u_2 l_1 & u_2 + l_2 & 1 & & & \\ & u_3 l_2 & u_3 + l_3 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & u_{n-1} l_{n-2} & u_{n-1} + l_{n-1} & 1 \\ & & & & u_n l_{n-1} & u_n \end{bmatrix}.$$

Note that both tridiagonals have their superdiagonal entries, that is, entries $(i, i+1)$, equal to 1. Also note that $J' = L^{-1}JL$, that is, J and J' are similar.

The *inertia* of a symmetric matrix A is the triple of integers

$$\mathit{Inertia}(A) \equiv (\nu, \zeta, \pi),$$

where ν is the number of negative eigenvalues of A, ζ is the number of zero eigenvalues of A and π is the number of positive eigenvalues of A.

If X is a nonsingular matrix, we say that $X^T AX$ and A are *congruent*.

If X is orthogonal, then $X^T AX$ are similar and so have the same eigenvalues. When X is only nonsingular, $X^T AX$ will generally not have the same eigenvalues of A, but the next theorem tells us that the two sets of eigenvalues will at least have the same signs.

Theorem 2.2.1 *Sylvester's inertia theorem. Let A be symmetric and X be nonsingular. Then A and $X^T AX$ have the same inertia.*

A proof of this result can be found, for instance, in Demmel [11, p. 202].

In the past most attention has been paid to the *positive case*: $l_i > 0$, $i = 1, \ldots, n-1$, $u_j > 0$, $j = 1, \ldots, n-1$. Remember the following standard results.

Lemma 2.2.2 *If $l_i u_i > 0$, $i = 1, \ldots, n-1$, then $J = LU$ is symmetrizable by a diagonal similarity and the number of positive (negative) u_i is the number of positive (negative) eigenvalues.*

Proof. Consider the diagonal matrix

$$D = \operatorname{diag}\left(1, \sqrt{l_1 u_1}, \sqrt{l_1 u_1 l_2 u_2}, \ldots, \sqrt{l_1 u_1 l_2 u_2 \cdots l_{n-1} u_{n-1}}\right).$$

Then the ith subdiagonal and superdiagonal entries of the tridiagonal matrix $T \equiv D^{-1}JD$ are equal to $\sqrt{l_i u_i}$, $i = 1, \ldots, n-1$.

Now, notice that

$$T = D^{-1}JD = D^{-1}LUD = (D^{-1}LD)(D^{-1}UD) = L'U',$$

where $L' \equiv D^{-1}LD$ and $U' \equiv D^{-1}UD$ is the LU decomposition of T. The symmetry of T permits us to write

$$T = L'D'L'^{T}$$

where $D' = \operatorname{diag}(u_1, u_2, \ldots, u_n)$. So, we have that J and T are similar and T and D' are congruent. By Sylvester's inertia theorem, J and D' have the same inertia. The eigenvalues of D' are the values u_i and, then, the number of positive (negative) u_i is the number of positive (negative) eigenvalues of J. □

Lemma 2.2.3 *If $l_i u_{i+1} > 0$, $i = 1, \ldots, n-1$, then $J' = UL$ is symmetrizable by a diagonal similarity and the number of positive (negative) u_i is the number of positive (negative) eigenvalues.*

The proof of this lemma is entirely analogous to the proof of the previous lemma.

2.2.3 Balancing

Ordinarily, balancing improves the conditioning of the initial matrix, enabling more accurate computation of eigenvectors and eigenvalues. It is an attempt to concentrate any ill-conditioning of the eigenvector matrix into a diagonal scaling.

Let $A = (a_{ij}) \in \mathbb{C}^{n\times n}$ be a $n \times n$ matrix and let $A^T = (a'_{ij})$ be its transpose. We say that A is *balanced* if $\|\boldsymbol{a}_i\|_2 = \|\boldsymbol{a'}_i\|_2$, $i = 1, \ldots, n$, that is, i^{th} column and i^{th} row of A have the same norm.

Lemma 2.2.4 *Any unreduced tridiagonal matrix $C \in \mathbb{C}^{n\times n}$ may be balanced by a diagonal similarity transformation.*

Proof. Note that we now consider complex matrices. Let

$$C = \text{tridiag}(\boldsymbol{b}, \boldsymbol{a}, \boldsymbol{c})$$

be an unreduced tridiagonal matrix so that

$$D \equiv \text{diag}\left(1, \sqrt{\frac{c_1}{b_1}}, \sqrt{\frac{c_1c_2}{b_1b_2}}, \ldots, \sqrt{\frac{c_1c_2\cdots c_{n-1}}{b_1b_2\cdots b_{n-1}}}\right)$$

is defined. Then the similar matrix DCD^{-1} is symmetric,

$$DCD^{-1} = \begin{bmatrix} a_1 & \sqrt{b_1c_1} & & & \\ \sqrt{b_1c_1} & a_2 & \sqrt{b_2c_2} & & \\ & \ddots & \ddots & \ddots & \\ & & \sqrt{b_{n-2}c_{n-2}} & a_{n-1} & \sqrt{b_{n-1}c_{n-1}} \\ & & & \sqrt{b_{n-1}c_{n-1}} & a_n \end{bmatrix},$$

and, thus, balanced. □

In practice, balancing only attempts to make the norm of each row equal to the norm of the corresponding column and, usually, can not turn a real nonsymmetric matrix into a real symmetric matrix.

So, any unreduced tridiagonal matrix C may be symmetrizable. If C is real and $b_ic_i > 0$, $i = 1, \ldots, n$, then DCD^{-1} is also real and, thus, hermitian. In this case, all eigenvalues of C are real. Otherwise, DCD^{-1} will have complex entries and may have complex eigenvalues.

Symmetrizing is not the only way to balance a matrix. If we consider the diagonal similarity defined by

$$D \equiv \text{diag}\left(1, \sqrt{\left|\frac{c_1}{b_1}\right|}, \sqrt{\left|\frac{c_1c_2}{b_1b_2}\right|}, \ldots, \sqrt{\left|\frac{c_1c_2\cdots c_{n-1}}{b_1b_2\cdots b_{n-1}}\right|}\right)$$

the matrix DCD^{-1} will also be balanced. It will be equal to

$$\begin{bmatrix} a_1 & \mathrm{e}^{\mathbf{i}\varphi_1}\sqrt{|b_1c_1|} & & & \\ \mathrm{e}^{\mathbf{i}\theta_1}\sqrt{|b_1c_1|} & a_2 & \mathrm{e}^{\mathbf{i}\varphi_2}\sqrt{|b_2c_2|} & & \\ & \ddots & \ddots & \ddots & \\ & & \mathrm{e}^{\mathbf{i}\theta_{n-2}}\sqrt{|b_{n-2}c_{n-2}|} & a_{n-1} & \mathrm{e}^{\mathbf{i}\varphi_{n-1}}\sqrt{|b_{n-1}c_{n-1}|} \\ & & & \mathrm{e}^{\mathbf{i}\theta_{n-1}}\sqrt{|b_{n-1}c_{n-1}|} & a_n \end{bmatrix}, \tag{2.6}$$

where $b_i = \mathrm{e}^{\mathbf{i}\theta_i}\,|b_i|$ and $c_i = \mathrm{e}^{\mathbf{i}\varphi_i}\,|c_i|$, $i = 1, \ldots, n-1$, with $\theta_i, \varphi_i \in [0, 2\pi[$, and $\mathbf{i}$ is the imaginary unit.

Lemma 2.2.5 *Any unreduced tridiagonal matrix $C \in \mathbb{R}^{n\times n}$ is diagonally similar to a form ΔT where T is symmetric and $\Delta = diag\,(\delta_1, \delta_2, \ldots, \delta_n)$, $\delta_i = \pm 1$.*

Proof. Consider the diagonal similarity shown in (2.6) and let $B = DCD^{-1}$. Since C is real, φ_i and θ_i are either π or 0, for all i. So, $\mathrm{e}^{\mathbf{i}\theta_i} = \mathrm{sign}(b_i)$ and $\mathrm{e}^{\mathbf{i}\varphi_i} = \mathrm{sign}(c_i)$. Let

$$\begin{aligned} \delta_1 = 1, \quad \delta_k &= \delta_{k-1}\,\mathrm{sign}(b_{k-1}c_{k-1}), \quad k = 2, \ldots, n. \\ \gamma_1 = \mathrm{sign}(c_1), \quad \gamma_k &= \delta_k\,\mathrm{sign}(c_k), \quad k = 2, \ldots, n-1. \end{aligned}$$

Notice that $\mathrm{sign}(\delta_k\gamma_k) = \mathrm{sign}(c_k), k = 1, \ldots, n-1$, and $\mathrm{sign}(\delta_k\gamma_{k-1}) = \mathrm{sign}(b_{k-1})$, $k = 2, \ldots, n$.

Now we can write B as $B = \Delta T$ where $\Delta = \operatorname{diag}(\delta_1, \delta_2, \ldots, \delta_n)$, and

$$T = \begin{bmatrix} \delta_1 a_1 & \gamma_1\sqrt{|b_1c_1|} & & & \\ \gamma_1\sqrt{|b_1c_1|} & \delta_2 a_2 & \gamma_2\sqrt{|b_2c_2|} & & \\ & \ddots & \ddots & \ddots & \\ & & \gamma_{n-2}\sqrt{|b_{n-2}c_{n-2}|} & \delta_{n-1}a_{n-1} & \gamma_{n-1}\sqrt{|b_{n-1}c_{n-1}|} \\ & & & \gamma_{n-1}\sqrt{|b_{n-1}c_{n-1}|} & \delta_n a_n \end{bmatrix}.$$

And $C = D^{-1}\Delta T D$. $\square$

Matrix Δ is called a *signature* matrix.

Consider the following 5×5 example.

Example 2.2.1 *We have*

$$\begin{bmatrix} 1 & -2 & & & \\ 3 & 2 & 2 & & \\ & -1 & 3 & -2 & \\ & & 3 & 4 & 2 \\ & & & -1 & 5 \end{bmatrix} = D^{-1} \begin{bmatrix} 1 & & & & \\ & -1 & & & \\ & & 1 & & \\ & & & -1 & \\ & & & & 1 \end{bmatrix} \begin{bmatrix} 1 & -\sqrt{6} & & & \\ -\sqrt{6} & -2 & -\sqrt{2} & & \\ & -\sqrt{2} & 3 & -\sqrt{6} & \\ & & -\sqrt{6} & -4 & -\sqrt{2} \\ & & & -\sqrt{2} & 5 \end{bmatrix} D,$$

where

$$D = \begin{bmatrix} 1 & & & & \\ & \sqrt{\frac{2}{3}} & & & \\ & & \sqrt{\frac{4}{3}} & & \\ & & & \sqrt{\frac{8}{9}} & \\ & & & & \sqrt{\frac{16}{9}} \end{bmatrix}. \diamond$$

2.3 Generalized eigenproblem

A generalized eigenvalue problem is an eigenproblem involving more than one matrix. In this section we will only give the definition and a result relating this problem to the standard one.

The standard eigenvalue problem asks for which scalars λ the matrix $A - \lambda I$ is singular; these scalars are the eigenvalues. This notion generalizes in several important ways.

If A and B are $m \times n$ matrices, $A - \lambda B$ is called a *matrix pencil* or just a *pencil*.

Definition 2.3.1 *If A and B are square matrices and $\det(A - \lambda B)$ is not identically zero, the pencil $A - \lambda B$ is called* regular. *Otherwise it is called* singular. *When $A - \lambda B$ is regular, $p(\lambda) \equiv \det(A - \lambda B)$ is called the characteristic polynomial of $A - \lambda B$ and the eigenvalues of $A - \lambda B$ are defined to be*

1. *the roots of $p(\lambda) = 0$,*
2. *∞ (with multiplicity $n - \deg p$) if $\deg p < n$.*

Matrix pencils arise naturally in many mathematical models of physical systems and the generalized eigenvalue problem is, in principle, more difficult than the standard one. The next fact relates the eigenvalues of a regular pencil $A - \lambda B$ to the eigenvalues of a single matrix.

Fact 2.3.1 *Let $A - \lambda B$ be regular. If B is nonsingular, all eigenvalues of $A - \lambda B$ are finite and the same as the eigenvalues of AB^{-1} or $B^{-1}A$. If B is singular, $A - \lambda B$ has eigenvalue ∞ with multiplicity $n - rank(B)$. If A is nonsingular, the eigenvalues of $A - \lambda B$ are the same as the reciprocals of the eigenvalues of $A^{-1}B$ or BA^{-1}, where a zero eigenvalue of $A^{-1}B$ corresponds to an infinite eigenvalue of $A - \lambda B$.*

For a proof see [11, p. 174].

For a balanced matrix ΔT the standard eigenvalue problem $(\Delta T - \lambda I)\boldsymbol{v} = 0$ is equivalent to the generalized eigenvalue problem

$$(T - \lambda \Delta)\boldsymbol{v} = 0.$$

In fact, we have

$$(\Delta T - \lambda I)\boldsymbol{v} = 0 \Longleftrightarrow \Delta(\Delta T - \lambda I)\boldsymbol{v} = 0 \Longleftrightarrow (T - \lambda\Delta)\boldsymbol{v} = 0.$$

Both T and Δ are real and symmetric and, so, we are in the presence of a symmetric generalized eigenvalue problem. If either T or Δ is positive definite then the eigenvalues must be real. Usually T is not symmetric positive definite. Thus, the case of interest here is when neither T nor Δ is positive definite (both indefinite) and so there may be complex eigenvalues. See Parlett [38, Chapter 15].

2.4 Measures of sensitivity

Nonsymmetric matrices can have a mixture of poorly and well conditioned eigenvalues. Small perturbations in the matrix, such as roundoff errors, can lead to large changes in the eigenvalues.

In this section we present several measures, some new, for the sensitivity of eigenvalues of a tridiagonal matrix. Tridiagonal matrices may be represented as products of bidiagonals in various ways depending on properties such as symmetry and positive definiteness, and different representations lead to different measures of sensitivity. These are relative condition numbers (relative perturbation results) for a simple eigenvalue $\lambda \neq 0$ of products of bidiagonals. The constraint that $\lambda \neq 0$ is always a weakness of a relative approach. When $\lambda = 0$ the relative approach looses its meaning and we go back to the absolute approach.

Recall that for $A = (a_{ij})$, $|A|$ denotes the matrix whose entries are $|a_{ij}|$. No use will be made of norms.

2.4.1 Balanced product of bidiagonals $\mathrm{T}\Delta = \mathrm{LDU}$

As shown above, any unreduced tridiagonal matrix C may be transformed into a balanced form by a diagonal similarity transformation. So, in this section we will restrict our attention to balanced tridiagonal matrices in the form $B = \Delta T$ (specially relevant for the computation

of eigenvectors) and such that symmetric tridiagonal matrix T admits the product

$$T = LDL^T$$

where $L = I + \mathring{L}$,

$$\mathring{L} = \begin{bmatrix} 0 & & & & \\ l_1 & 0 & & & \\ & \ddots & \ddots & & \\ & & l_{n-2} & 0 & \\ & & & l_{n-1} & 0 \end{bmatrix}$$

and $D = \mathrm{diag}(d_1, \ldots, d_n)$.

Since $\Delta^{-1}B\Delta = \Delta^{-1}\Delta T\Delta = T\Delta$, matrices B and $T\Delta$ are similar. Thus we will be interested in the factorization of $T\Delta$ that will be

$$T\Delta = LDU$$

where $U = (I + \mathring{L}^T)\Delta$. As observed before, when T is real, $\Delta = \mathrm{diag}(1, \pm 1, \ldots, \pm 1)$.

In detail, we are just saying that an unreduced tridiagonal matrix C is always diagonally similar to a form $T\Delta$ where T is symmetric,

$$C = D_1^{-1}BD_1 = D_1^{-1}\Delta T D_1 = D_1^{-1}\Delta T\Delta\Delta^{-1}D_1 = (D_1^{-1}\Delta)(T\Delta)(D_1^{-1}\Delta)^{-1}.$$

Now, let us concentrate on the factorization $T\Delta = LDU$ for the real case. The off-diagonal entries of $T\Delta$ are $\pm l_j d_j$. The perturbations will be of the form

$$l_i \to l_i(1 + \alpha_i), \qquad d_i \to d_i(1 + \beta_i),$$

with $|\alpha_i| \leq \eta$, $|\beta_i| \leq \eta$. In matrix terms,

$$L \to L + \delta L, \qquad D \to D + \delta D,$$

with $\delta L = \mathrm{diag}(\alpha_1, \ldots, \alpha_n)\mathring{L}$ and $\delta D = \mathrm{diag}(\beta_1, \ldots, \beta_n)D$. However the α_i and β_i play no role because we majorize such terms by η.

Then, element by element, we write

$$|\delta L| \le \eta|\mathring{L}|, \qquad |\delta D| \le \eta\,|D|\,, \qquad |\delta U| \le \eta|\mathring{L}^T\Delta| = \eta|\mathring{L}^T|.$$

Matrix Δ does not change.

Consider the change in a simple eigenvalue $\lambda \neq 0$ satisfying

$$LDU\boldsymbol{x} = \boldsymbol{x}\lambda, \qquad \boldsymbol{y}^*LDU = \lambda\boldsymbol{y}^*, \qquad \boldsymbol{y}^*\boldsymbol{x} \neq 0. \tag{2.7}$$

The perturbed values satisfy

$$(L+\delta L)(D+\delta D)(U+\delta U)(\boldsymbol{x}+\delta\boldsymbol{x}) = (\boldsymbol{x}+\delta\boldsymbol{x})(\lambda+\delta\lambda). \tag{2.8}$$

If η is small enough then, after multiplying out the factors in (2.8) and using (2.7), we obtain

$$(LD\delta U + L\delta DU + \delta LDU)\boldsymbol{x} + LDU\delta\boldsymbol{x} + \mathcal{O}(\eta^2) = \boldsymbol{x}\delta\lambda + \delta\boldsymbol{x}\lambda + \mathcal{O}(\eta^2). \tag{2.9}$$

Multiply (2.9) on the left by $\boldsymbol{y}^*$ to get

$$\boldsymbol{y}^*\left(LD\delta U + L\delta DU + \delta LDU\right)\boldsymbol{x} + \boldsymbol{y}^*LDU\delta\boldsymbol{x} = \boldsymbol{y}^*\boldsymbol{x}\delta\lambda + \boldsymbol{y}^*\delta\boldsymbol{x}\lambda + \mathcal{O}(\eta^2).$$

Now use (2.7) to see that $\boldsymbol{y}^*LDU\delta\boldsymbol{x} = \boldsymbol{y}^*\delta\boldsymbol{x}\lambda$ so that

$$\delta\lambda\boldsymbol{y}^*\boldsymbol{x} = \boldsymbol{y}^*\left(LD\delta U + L\delta DU + \delta LDU\right)\boldsymbol{x} + \mathcal{O}(\eta^2). \tag{2.10}$$

There are two ways to proceed. First insert δU, δD, and δL and majorize to find

$$|\delta\lambda||\boldsymbol{y}^*\boldsymbol{x}| \le \eta|\boldsymbol{y}|^T\left(|LD||\mathring{L}^T| + |L||D||U| + |\mathring{L}||DU|\right)|\boldsymbol{x}| + \mathcal{O}(\eta^2).$$

Use the fact that $L = I + \mathring{L}$, $U = I + \mathring{L}^T\Delta$ to find

$$|\delta\lambda||\boldsymbol{y}^*\boldsymbol{x}| \le \eta|\boldsymbol{y}|^T\left(|D| + 3|\mathring{L}D\mathring{L}^T| + 2|D\mathring{L}^T| + 2|\mathring{L}D|\right)|\boldsymbol{x}| + \mathcal{O}(\eta^2).$$

Defining $G_1 \equiv |D| + 3|\mathring{L}D\mathring{L}^T| + 2|D\mathring{L}^T| + 2|\mathring{L}D|$ we have

$$\frac{|\delta\lambda|}{|\lambda|} \le \eta\frac{|\boldsymbol{y}|^TG_1|\boldsymbol{x}|}{|\boldsymbol{y}^*\boldsymbol{x}||\lambda|} + \mathcal{O}(\eta^2).$$

This gives the first relative condition number

$$\boxed{\text{relcond}_1(\lambda; LDU) := \frac{|\boldsymbol{y}|^T G_1 |\boldsymbol{x}|}{|\boldsymbol{y}^*\boldsymbol{x}||\lambda|}}$$

and

$$G_1 = \begin{bmatrix} |d_1| & 2|d_1||l_1| & & & \\ 2|d_1||l_1| & |d_2| + 3|d_1||l_1|^2 & 2|d_2||l_2| & & \\ \ddots & \ddots & \ddots & & \\ & 2|d_{n-2}||l_{n-2}| & |d_{n-1}| + 3|d_{n-2}||l_{n-2}|^2 & 2|d_{n-1}||l_{n-1}| & \\ & & 2|d_{n-1}||l_{n-1}| & |d_n| + 3|d_{n-1}||l_{n-1}|^2 \end{bmatrix}.$$

The second approach avoids the presence of $|\lambda|$ in the denominator. The division by $|\lambda|$ will be done implicitly and, thus, more elegantly. So, return to (2.10) and use the relations (the constraint $\lambda \neq 0$ implies that U is invertible)

$$\boldsymbol{y}^* LD = \lambda \boldsymbol{y}^* U^{-1}, \quad \boldsymbol{y}^* L = \lambda \boldsymbol{y}^* U^{-1} D^{-1}, \quad U\boldsymbol{x} = D^{-1}L^{-1}\lambda \boldsymbol{x} \quad \text{and} \quad DU\boldsymbol{x} = L^{-1}\lambda \boldsymbol{x}$$

to find

$$\delta\lambda \boldsymbol{y}^*\boldsymbol{x} = \lambda \boldsymbol{y}^* \left(U^{-1}\delta U + U^{-1}D^{-1}\delta D D^{-1}L^{-1}\lambda + \delta L L^{-1} \right) \boldsymbol{x} + \mathcal{O}(\eta^2). \tag{2.11}$$

In order to majorize each term we will use the fact that $\mathring{L}$ is a nilpotent matrix. We have $\mathring{L}^n = (\mathring{L}^T)^n = O$, and then

$$\left(I + \mathring{L}\right)^{-1} = \sum_{i=0}^{\infty} \left(-\mathring{L}\right)^i = \sum_{i=0}^{n-1} \left(-\mathring{L}\right)^i, \tag{2.12}$$

$$\left(I + \mathring{L}^T\right)^{-1} = \sum_{i=0}^{\infty} \left(-\mathring{L}^T\right)^i = \sum_{i=0}^{n-1} \left(-\mathring{L}^T\right)^i. \tag{2.13}$$

Thus

$$\left|U^{-1}\right| = \left| \left[\left(I + \mathring{L}^T\right) \Delta \right]^{-1} \right| = \left| \left(I + \mathring{L}^T\right)^{-1} \right| \leq \sum_{i=0}^{n-1} |\mathring{L}^T|^i = \left(I - |\mathring{L}^T|\right)^{-1} \tag{2.14}$$

and

$$\left|L^{-1}\right| = \left|(I+\mathring{L})^{-1}\right| \leq \sum_{i=0}^{n-1} |\mathring{L}|^i = \left(I - |\mathring{L}|\right)^{-1} \tag{2.15}$$

Since $|\delta U| \leq \eta |\mathring{L}|^T$,

$$\left|U^{-1}\delta U\right| \leq \eta |\mathring{L}|^T \left(I - |\mathring{L}|^T\right)^{-1}, \tag{2.16}$$

and, since $|\delta L| \leq \eta |\mathring{L}|$,

$$\left|\delta L L^{-1}\right| \leq \eta |\mathring{L}| \left(I - |\mathring{L}|\right)^{-1}. \tag{2.17}$$

Finally, since $|\delta D| \leq \eta\,|D|$, we have

$$\begin{aligned}\left|U^{-1}D^{-1}\delta D D^{-1}L^{-1}\right| &\leq \eta \left(I - |\mathring{L}|^T\right)^{-1} \left|D^{-1}\right| |D| \left|D^{-1}\right| \left(I - |\mathring{L}|\right)^{-1} \\ &= \eta \left(I - |\mathring{L}|^T\right)^{-1} |D|^{-1} \left(I - |\mathring{L}|\right)^{-1}. \end{aligned} \tag{2.18}$$

Insert these three values, (2.16), (2.17) and (2.18), in (2.11) to find

$$\begin{aligned}\left|\frac{\delta\lambda}{\lambda}\right| \leq \frac{\eta|\boldsymbol{y}|^T}{|\boldsymbol{y}^*\boldsymbol{x}|} \Big[&|\mathring{L}|^T \left(I - |\mathring{L}|^T\right)^{-1} + \left(I - |\mathring{L}|^T\right)^{-1} |D|^{-1} \left(I - |\mathring{L}|\right)^{-1} |\lambda| + \\ &+ |\mathring{L}| \left(I - |\mathring{L}|\right)^{-1} \Big] |\boldsymbol{x}| + \mathcal{O}(\eta^2). \end{aligned} \tag{2.19}$$

It is easy to see that $|\mathring{L}|^T$ and $\left(I - |\mathring{L}|^T\right)^{-1}$ comute, so we can write

$$\begin{aligned}\left|\frac{\delta\lambda}{\lambda}\right| \leq \frac{\eta|\boldsymbol{y}|^T}{|\boldsymbol{y}^*\boldsymbol{x}|} \left(I - |\mathring{L}|^T\right)^{-1} \Big[&|\mathring{L}|^T \left(I - |\mathring{L}|\right) + |D|^{-1} |\lambda| + \\ &+ \left(I - |\mathring{L}|^T\right) |\mathring{L}| \Big] \left(I - |\mathring{L}|\right)^{-1} |\boldsymbol{x}| + \mathcal{O}(\eta^2). \end{aligned} \tag{2.20}$$

Now, using the fact that

$$|\mathring{L}|^T \left(I - |\mathring{L}|\right) + \left(I - |\mathring{L}|^T\right) |\mathring{L}| = |\mathring{L}|^T + |\mathring{L}| - 2|\mathring{L}|^T |\mathring{L}|$$

and that

$$\left(I - |\mathring{L}|^T\right)^{-1} = \left[\left(I - |\mathring{L}|\right)^{-1}\right]^T,$$

we define

$$G_2 \equiv |D|^{-1} |\lambda| - 2|\mathring{L}|^T |\mathring{L}| + |\mathring{L}|^T + |\mathring{L}|.$$

Note that G_2 is scaling invariant because it is a sum of scaling invariant matrices. This is important because the relative error is scaling invariant and, then, the formulae should also be scaling invariant. When we multiply the matrix T by a nonzero scalar α, the eigenvalue changes to $\alpha\lambda$ and the factor D to αD,

$$\alpha T = L(\alpha D)L^T.$$

Factor L does not change neither does $|D|^{-1}|\lambda|$, i.e., $|\alpha D|^{-1}|\alpha\lambda| = |D|^{-1}|\lambda|$.

So (2.20) can be written as

$$\frac{|\delta\lambda|}{|\lambda|} \leq \eta \frac{|\boldsymbol{y}|^T \left[\left(I - |\mathring{L}|\right)^{-1}\right]^T G_2 \left(I - |\mathring{L}|\right)^{-1} |\boldsymbol{x}|}{|\boldsymbol{y}^*\boldsymbol{x}|} + \mathcal{O}(\eta^2).$$

Our second relative condition number is

$$\boxed{\text{relcond}_2(\lambda; LDU) := \frac{v^T G_2 w}{|\boldsymbol{y}^*\boldsymbol{x}|}}$$

where v and w are the solutions of the lower bidiagonal systems

$$\left(I - |\mathring{L}|\right) v = |\boldsymbol{y}| \qquad \text{and} \qquad \left(I - |\mathring{L}|\right) w = |\boldsymbol{x}|$$

and

$$G_2 = \begin{bmatrix} |\lambda||d_1|^{-1} - 2l_1^2 & |l_1| & & & \\ |l_1| & |\lambda||d_2|^{-1} - 2l_2^2 & |l_2| & & \\ & \ddots & \ddots & \ddots & \\ & & |l_{n-2}| & |\lambda||d_{n-1}|^{-1} - 2l_{n-1}^2 & |l_{n-1}| \\ & & & |l_{n-1}| & |\lambda||d_n|^{-1} \end{bmatrix}.$$

Both condition numbers are computable when $\boldsymbol{x}$ and $\boldsymbol{y}$ are known. The former requires only the use of a tridiagonal matrix while the second also requires the solution of two bidiagonal systems with no divisions.

2.4.2 Product of bidiagonals J = LU

Consider a simple eigenvalue of a tridiagonal matrix J represented by LU (relevant for the computation of eigenvalues) where

$$L = I + \mathring{L} \qquad \text{and} \qquad U = N + \text{diag}(\boldsymbol{u}) \tag{2.21}$$

with $\boldsymbol{u} = (u_1, u_2, \ldots, u_n)^T$,

$$\mathring{L} = \begin{bmatrix} 0 & & & & \\ l_1 & 0 & & & \\ & \ddots & \ddots & & \\ & & l_{n-2} & 0 & \\ & & & l_{n-1} & 0 \end{bmatrix} \qquad \text{and} \qquad N = \begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ & & & & 0 \end{bmatrix}.$$

Both N and $\mathring{L}$ are nilpotent matrices, $N^n = \mathring{L}^n = O$.

Without loss of generality we assume that $u_i \neq 0$, $i = 1, 2, \ldots, n-1$. We also assume that $\lambda \neq 0$.

A simple eigenvalue λ has right and left eigenvectors satisfying

$$LU\boldsymbol{x} = \boldsymbol{x}\lambda, \qquad \boldsymbol{y}^* LU = \lambda \boldsymbol{y}^*, \qquad \boldsymbol{y}^* \boldsymbol{x} \neq 0. \tag{2.22}$$

There is no need to normalize $\boldsymbol{x}$ and $\boldsymbol{y}$.

The perturbations of interest are relative:

$$l_i \to l_i(1 + \alpha_i), \qquad u_i \to u_i(1 + \beta_i),$$

with $|\alpha_i| \leq \eta$, $|\beta_i| \leq \eta$. Early in the analysis we will majorize the perturbations $|\alpha_i|$ and $|\beta_i|$ by $\eta << 1$.

In terms of matrices, the perturbations are

$$L \to L + \delta L, \qquad U \to U + \delta U,$$

with $\delta L = \text{diag}(\alpha_1, \ldots, \alpha_n)\mathring{L}$ and $\delta U = \text{diag}(\beta_1, \ldots, \beta_n)\text{diag}(\boldsymbol{u})$.

Then, element by element, we write

$$|\delta L| = |\delta \mathring{L}| \leq \eta |\mathring{L}|, \qquad |\delta U| = |\delta \mathrm{diag}(\boldsymbol{u})| \leq \eta |\mathrm{diag}(\boldsymbol{u})|. \tag{2.23}$$

We are now ready to make a first order analysis in terms of η. The perturbed values satisfy

$$(L + \delta L)(U + \delta U)(\boldsymbol{x} + \delta \boldsymbol{x}) = (\boldsymbol{x} + \delta \boldsymbol{x})(\lambda + \delta \lambda). \tag{2.24}$$

If η is small enough then, after multiplying out the factors in (2.24) and using (2.22), we obtain

$$L\delta U \boldsymbol{x} + \delta L U \boldsymbol{x} + L U \delta \boldsymbol{x} + \mathcal{O}(\eta^2) = \boldsymbol{x}\delta\lambda + \delta \boldsymbol{x}\lambda + \mathcal{O}(\eta^2). \tag{2.25}$$

Premultiply by $\boldsymbol{y}^*$ to get

$$\boldsymbol{y}^* L\delta U \boldsymbol{x} + \boldsymbol{y}^* \delta L U \boldsymbol{x} + \boldsymbol{y}^* L U \delta \boldsymbol{x} = \boldsymbol{y}^* \boldsymbol{x}\delta\lambda + \boldsymbol{y}^* \delta \boldsymbol{x}\lambda + \mathcal{O}(\eta^2).$$

Now use (2.22) to find

$$\delta\lambda \boldsymbol{y}^* \boldsymbol{x} = \boldsymbol{y}^* \left(L\delta U + \delta L U\right) \boldsymbol{x} + \mathcal{O}(\eta^2). \tag{2.26}$$

There are two ways to proceed. To first order, after using (2.23),

$$|\delta\lambda||\boldsymbol{y}^* \boldsymbol{x}| \leq \eta |\boldsymbol{y}|^T \left(|L||\mathrm{diag}(\boldsymbol{u})| + |\mathring{L}||U|\right) |\boldsymbol{x}| + \mathcal{O}(\eta^2). \tag{2.27}$$

Use the fact that $L = I + \mathring{L}$, $U = N + \mathrm{diag}(\boldsymbol{u})$ to find

$$|\delta\lambda||\boldsymbol{y}^* \boldsymbol{x}| \leq \eta |\boldsymbol{y}|^T \left(|\mathrm{diag}(\boldsymbol{u})| + 2|\mathring{L}||\mathrm{diag}(\boldsymbol{u})| + |\mathring{L}|N\right) |\boldsymbol{x}| + \mathcal{O}(\eta^2).$$

Defining $M_1 \equiv |\mathrm{diag}(\boldsymbol{u})| + 2|\mathring{L}||\mathrm{diag}(\boldsymbol{u})| + |\mathring{L}|N$ we have

$$\frac{|\delta\lambda|}{|\lambda|} \leq \eta \frac{|\boldsymbol{y}|^T M_1 |\boldsymbol{x}|}{|\boldsymbol{y}^* \boldsymbol{x}||\lambda|} + \mathcal{O}(\eta^2).$$

This gives the first relative condition number for the form $J = LU$,

$$\boxed{\mathrm{relcond}_1(\lambda; LU) := \frac{|\boldsymbol{y}|^T M_1 |\boldsymbol{x}|}{|\boldsymbol{y}^* \boldsymbol{x}||\lambda|}}$$

and

$$M_1 = \begin{bmatrix} |u_1| & & & \\ 2|l_1||u_1| & |l_1|+|u_2| & & \\ \ddots & \ddots & \ddots & \\ & 2|l_{n-2}||u_{n-2}| & |l_{n-2}|+|u_{n-1}| & \\ & & 2|l_{n-1}||u_{n-1}| & |l_{n-1}|+|u_n| \end{bmatrix}.$$

The second approach tries to bring λ into the right side. Return to (2.26) and, from (2.22), use the relations

$$U\boldsymbol{x} = L^{-1}\boldsymbol{x}\lambda \qquad \text{and} \qquad \boldsymbol{y}^*L = \lambda\boldsymbol{y}^*U^{-1}$$

to find

$$\delta\lambda\boldsymbol{y}^*\boldsymbol{x} = \lambda\boldsymbol{y}^*\left(U^{-1}\delta U + \delta L L^{-1}\right)\boldsymbol{x} + \mathcal{O}(\eta^2). \tag{2.28}$$

After using (2.23), we have

$$|\delta\lambda||\boldsymbol{y}^*\boldsymbol{x}| \leq \eta|\lambda||\boldsymbol{y}|^T\left(\left|U^{-1}\right||\mathrm{diag}(\boldsymbol{u})| + |\mathring{L}|\left|L^{-1}\right|\right)|\boldsymbol{x}| + \mathcal{O}(\eta^2). \tag{2.29}$$

In order to majorize the term $|\mathring{L}|\left|L^{-1}\right|$ we will use the fact that $\mathring{L}$ is a nilpotent matrix. We have $\mathring{L}^n = (\mathring{L}^T)^n = O$, and then

$$L^{-1} = \left(I + \mathring{L}\right)^{-1} = \sum_{i=0}^{n-1}\left(-\mathring{L}\right)^i. \tag{2.30}$$

Note that the supports of the powers of $\mathring{L}$ are disjoint. Hence, term by term,

$$\left|L^{-1}\right| \leq \sum_{i=0}^{n-1}|\mathring{L}|^i = \left(I - |\mathring{L}|\right)^{-1}. \tag{2.31}$$

Then

$$|\mathring{L}|\left|L^{-1}\right| \leq |\mathring{L}|\left(I - |\mathring{L}|\right)^{-1} = \left(I - |\mathring{L}|\right)^{-1} - I. \tag{2.32}$$

To majorize $\left|U^{-1}\right|\,|\mathrm{diag}(\boldsymbol{u})| = \left|U^{-1}\mathrm{diag}(\boldsymbol{u})\right|$, note first that $U^{-1}\mathrm{diag}(\boldsymbol{u}) = \left(\mathrm{diag}(\boldsymbol{u})^{-1}U\right)^{-1}$. Hence

$$\mathrm{diag}(\boldsymbol{u})^{-1}U = \begin{bmatrix} 1 & u_1^{-1} & & & \\ & 1 & u_2^{-1} & & \\ & & \ddots & \ddots & \\ & & & 1 & u_{n-1}^{-1} \\ & & & & 1 \end{bmatrix}.$$

It is convenient to define the nilpotent matrix

$$\mathring{U} = \begin{bmatrix} 0 & u_1^{-1} & & & \\ & 0 & u_2^{-1} & & \\ & & \ddots & \ddots & \\ & & & 0 & u_{n-1}^{-1} \\ & & & & 0 \end{bmatrix}.$$

So, $\mathring{U}^n = O$ and $\mathrm{diag}(\boldsymbol{u})^{-1}U = I + \mathring{U}$. There is no addition or subtraction of scalars in expanding the inverse of $I + \mathring{U}$ and from that it follows that

$$\left(\mathrm{diag}(\boldsymbol{u})^{-1}U\right)^{-1} = \left(I + \mathring{U}\right)^{-1} = \sum_{i=0}^{n-1} \left(-\mathring{U}\right)^i. \tag{2.33}$$

And, term by term,

$$\left|U^{-1}\mathrm{diag}(\boldsymbol{u})\right| \le \sum_{i=0}^{n-1} |\mathring{U}|^i = \left(I - |\mathring{U}|\right)^{-1}. \tag{2.34}$$

Now insert majorations (2.34) and (2.32) into (2.29) to find

$$\left|\frac{\delta\lambda}{\lambda}\right| |\boldsymbol{y}^*\boldsymbol{x}| \le \eta|\boldsymbol{y}|^T \left\{ \left(I - |\mathring{U}|\right)^{-1} + \left[\left(I - |\mathring{L}|\right)^{-1} - I\right] \right\} |\boldsymbol{x}| + \mathcal{O}(\eta^2). \tag{2.35}$$

But matrix $\left(I - |\mathring{L}|\right)^{-1} - I$ is not invertible.

We define

$$M_2 \equiv \left(I - |\mathring{U}|\right)^{-1} + \left[\left(I - |\mathring{L}|\right)^{-1} - I\right].$$

So (2.35) can be written as

$$\frac{|\delta\lambda|}{|\lambda|} \leq \eta \frac{|\boldsymbol{y}|^T M_2 |\boldsymbol{x}|}{|\boldsymbol{y}^* \boldsymbol{x}|} + \mathcal{O}(\eta^2).$$

Our second condition number is

$$\boxed{\text{relcond}_2(\lambda; LU) := \frac{|\boldsymbol{y}|^T M_2 |\boldsymbol{x}|}{|\boldsymbol{y}^* \boldsymbol{x}|}}$$

where

$$M_2 = \begin{bmatrix} 1 & |u_1|^{-1} & |u_1 u_2|^{-1} & |u_1 u_2 u_3|^{-1} & \dots & \cdot & |u_1 u_2 \dots u_{n-1}|^{-1} \\ |l_1| & 1 & |u_2|^{-1} & |u_2 u_3|^{-1} & \dots & \cdot & |u_2 \dots u_{n-1}|^{-1} \\ |l_1 l_2| & |l_2| & 1 & |u_3|^{-1} & \dots & \cdot & |u_3 \dots u_{n-1}|^{-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ |l_1 \dots l_{n-2}| & |l_2 \dots l_{n-2}| & |l_3 \dots l_{n-2}| & \cdot & \dots & 1 & |u_{n-1}|^{-1} \\ |l_1 \dots l_{n-1}| & |l_2 \dots l_{n-1}| & |l_3 \dots l_{n-1}| & \cdot & \dots & |l_{n-1}| & 1 \end{bmatrix}.$$

We do not need this matrix explicitly. Instead we can solve two bidiagonal systems. Going back to (2.35) we can write

$$\begin{aligned} \frac{|\delta\lambda|}{|\lambda|} &\leq \eta \frac{|\boldsymbol{y}|^T \left(I - |\mathring{U}|\right)^{-1} |\boldsymbol{x}| + |\boldsymbol{y}|^T \left[\left(I - |\mathring{L}|\right)^{-1} - I\right] |\boldsymbol{x}|}{|\boldsymbol{y}^* \boldsymbol{x}|} + \mathcal{O}(\eta^2) \\ &= \eta \frac{|\boldsymbol{y}|^T (v + w)}{|\boldsymbol{y}^* \boldsymbol{x}|} + \mathcal{O}(\eta^2) \end{aligned}$$

where v is the solution of the system

$$\left(I - |\mathring{U}|\right) v = |\boldsymbol{x}|$$

and w satisfies

$$\left[\left(I - |\mathring{L}|\right)^{-1} - I\right] |\boldsymbol{x}| = w \iff \left(I - |\mathring{L}|\right) (w + |\boldsymbol{x}|) = |\boldsymbol{x}|.$$

If we solve the system

$$\left(I - |\mathring{L}|\right) z = |\boldsymbol{x}|$$

then $w = z - |\boldsymbol{x}|$.

We may be bothered by the fact that $\mathring{U}$ is not scaling invariant. This defect is a consequence of the normalization in the representation LU that keeps the superdiagonal entries at the value 1. In other words the eigenvectors $\boldsymbol{x}$ and $\boldsymbol{y}$ have to change along with L and U. In more detail, the matrix $10LU$ has representation

$$D^{-1}(10LU)D = (I + 10\mathring{L})(10\text{diag}(\boldsymbol{u}) + N)$$

where $D = \text{diag}(1, 10^{-1}, 10^{-2}, \ldots, 10^{1-n})$.

The qd representation is not scaling invariant. The factors L and U will change if we multiply the matrix J by a scalar; and the condition number will change too. We need a diagonal scaling.

2.4.3 Derivatives from $\Delta\mathrm{T} = \Delta\mathrm{L}\Omega\mathrm{L}^{\mathrm{T}}$

In this section, from Parlett [42], we will describe sensitivity in terms of the generalized singular value decomposition. We will present formulas that state the relative sensitivity of the eigenvalues of $\Delta L\Omega L^T$ to L's entries.

Any real tridiagonal is diagonally similar to ΔT, where ΔT is real symmetric and $\Delta = \text{diag}(\delta_1, \delta_2, \ldots, \delta_n)$, $\delta_i = \pm 1$. This similarity transformation is an instance of balancing a nonsymmetric matrix. We assume that the eigenvalues λ_i are distinct but we allow them to be complex. Thus

$$\begin{aligned} &T\boldsymbol{s}_i = \Delta\boldsymbol{s}_i\lambda_i, \quad i = 1, \ldots, n \\ &S = \Big[\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n\Big], \quad \text{possibly complex.} \end{aligned} \tag{2.36}$$

Suppose such matrix T may be written as

$$T = L\Omega L^T, \tag{2.37}$$

where L is lower bidiagonal and Ω is diagonal.

If T is positive definite then $\Omega = I$ and $T = LL^T$ is the Cholesky factorization of T. If T is indefinite and has the factorization

$$T = L_1 D L_1^T,$$

with $D = \mathrm{diag}(d_1, \ldots, d_n)$ and

$$L_1 = \begin{bmatrix} 1 & & & & \\ l_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & l_{n-2} & 1 & \\ & & & l_{n-1} & 1 \end{bmatrix},$$

then, defining

$$D_1 := \mathrm{diag}(\sqrt{|d_1|}, \ldots, \sqrt{|d_n|}) \quad \text{and} \quad \Omega := \mathrm{diag}(\omega_1, \ldots, \omega_n), \;\; \omega_i = \mathrm{sign}(d_i),$$

we can obtain the factorization (2.37). We will have

$$T = L_1 D_1 \Omega D_1 L_1^T = L \Omega L^T \tag{2.38}$$

with $L \equiv L_1 D_1$. This is the closest factorization to the Cholesky factorization that we can get.

A standard normalization for the eigenvectors is

$$S^T \Delta S = I. \tag{2.39}$$

Let $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ and denote by $\Lambda^{1/2}$ the principal square root of Λ. That is, if $\lambda_i = \rho_i \, \mathrm{e}^{\mathbf{i} 2\theta_i}$ then $\lambda_i^{1/2} = \rho_i^{1/2} \, \mathrm{e}^{\mathbf{i}\theta_i}$, $-\frac{\pi}{2} < \theta_i \leq \frac{\pi}{2}$ and

$$\Lambda^{1/2} = \mathrm{diag}(\lambda_1^{1/2}, \ldots, \lambda_n^{1/2}).$$

As in the symmetric case there is an indefinite singular value decomposition of L,

$$L = \Delta S \Lambda^{1/2} P^T \tag{2.40}$$

defining P which is complex in general. This is a generalization of the familiar singular value decomposition $U\Sigma V^T$ of L.

Note first that from (2.36), we have

$$TS = \Delta S\Lambda \Leftrightarrow (\Delta S)^{-1}T = \Lambda S^{-1} \tag{2.41}$$

Also note that

$$\begin{aligned} P^T\Omega P =& \Lambda^{-1/2}(\Delta S)^{-1}L\Omega L^T(\Delta S)^{-T}\Lambda^{-1/2}, \qquad \text{by (2.40)},\\ =& \Lambda^{-1/2}(\Delta S)^{-1}T(\Delta S)^{-T}\Lambda^{-1/2}, \qquad \text{by (2.38)},\\ =& \Lambda^{-1/2}\left[\Lambda S^{-1}(\Delta S)^{-T}\right]\Lambda^{-1/2}, \qquad \text{by (2.41)}\\ =& I, \qquad \text{by (2.39)}. \end{aligned}$$

So, P is defined by (2.40) as

$$P = L^T(\Delta S)^{-T}\Lambda^{-1/2}.$$

With Δ and Ω fixed we will study how Λ depends on L. Write

$$L = \begin{bmatrix} \mathfrak{a}_1 & & & & \\ \mathfrak{b}_1 & \mathfrak{a}_2 & & & \\ & \ddots & \ddots & & \\ & & \mathfrak{b}_{n-2} & \mathfrak{a}_{n-1} & \\ & & & \mathfrak{b}_{n-1} & \mathfrak{a}_n \end{bmatrix}$$

and let $P := \left[\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n\right]$. So, $P^T\Omega P = I$ and, from (2.40),

$$L(\Omega P) = \Delta S\Lambda^{1/2}P^T\Omega P = \Delta S\Lambda^{1/2}.$$

Also, using (2.39),

$$S^TL = (\Delta S)^{-1}L = \Lambda^{1/2}P^T,$$

Consider a typical *singular triple* $(\lambda, \boldsymbol{s}, \boldsymbol{p}^T)$

$$L(\Omega \boldsymbol{p}) = (\Delta \boldsymbol{s})\lambda^{1/2} \tag{2.42}$$
$$\boldsymbol{s}^T L = \lambda^{1/2} \boldsymbol{p}^T.$$

Recall that we use to note $\boldsymbol{p} = (p_k)$ and $\boldsymbol{s} = (s_k)$, $k = 1, \ldots, n$.

We can now state the relative sensitivity of the eigenvalues of $\Delta T = \Delta L \Omega L^T$ to L's entries. The expressions may be complex.

Theorem 2.4.1 *Let* $\Delta = \mathrm{diag}(\delta_1, \delta_2, \ldots, \delta_n)$, $\delta_i = \pm 1$, *and* $\Omega = diag(\omega_1, \ldots, \omega_n)$, $\omega_i = \pm 1$. *If* ΔT *has distinct eigenvalues and (2.37) holds then, for* $\lambda \neq 0$,

$$\frac{1}{2}\frac{\mathfrak{a}_j}{\lambda}\frac{\partial \lambda}{\partial \mathfrak{a}_j} = \sum_{k=1}^{j} \delta_k s_k^2 - \sum_{m=1}^{j-1} \omega_m p_m^2 = \sum_{m=j}^{n} \omega_m p_m^2 - \sum_{k=j+1}^{n} \delta_k s_k^2, \tag{2.43}$$

$$\frac{1}{2}\frac{\mathfrak{b}_j}{\lambda}\frac{\partial \lambda}{\partial \mathfrak{b}_j} = \sum_{m=1}^{j} \left[\omega_m p_m^2 - \delta_m s_m^2\right] = \sum_{m=j+1}^{n} \left[\delta_m s_m^2 - \omega_m p_m^2\right]. \tag{2.44}$$

For a proof see [42, p. 430].

2.4.4 Derivatives from $\Delta \mathrm{T} = \Delta \mathrm{L} \mathrm{D} \mathrm{L}^{\mathrm{T}}$

From Z. Wu [64]. In this section we will give theoretical evidence that eigenvalues may be determined to high relative accuracy by the bidiagonals. Consider a tridiagonal matrix written in the balanced form ΔT, where T is symmetric and $\Delta = \mathrm{diag}(\delta_1, \delta_2, \ldots, \delta_n)$ with $\delta_i = \pm 1$. Consider a typical eigenpair $(\lambda, \boldsymbol{v})$ of ΔT with the normalization $\boldsymbol{v}^* \boldsymbol{v} = 1$. That is,

$$\Delta T \boldsymbol{v} = \lambda \boldsymbol{v} \Longleftrightarrow (T - \lambda \Delta)\boldsymbol{v} = 0 \tag{2.45}$$

Lemma 2.4.1 *If* $\Im \lambda \neq 0$ *then* $\boldsymbol{v}^* \Delta \boldsymbol{v} = \boldsymbol{v}^* T \boldsymbol{v} = 0$

Proof. Since $(T - \lambda \Delta)\boldsymbol{v} = 0$, we have

$$\boldsymbol{v}^* (T - \lambda \Delta)\boldsymbol{v} = 0. \tag{2.46}$$

Also, since $\boldsymbol{v}^*T\boldsymbol{v}$ is a scalar and T is real and symmetric, we have

$$\overline{\boldsymbol{v}^*T\boldsymbol{v}} = \boldsymbol{v}^TT\overline{\boldsymbol{v}} = \left(\boldsymbol{v}^TT\overline{\boldsymbol{v}}\right)^T = \boldsymbol{v}^*T\boldsymbol{v}$$

Using the same arguments, we have that $\boldsymbol{v}^*\Delta\boldsymbol{v}$ is also real. So, from (2.46),

$$\lambda\boldsymbol{v}^*\Delta\boldsymbol{v} = \boldsymbol{v}^*T\boldsymbol{v},$$

that is, $\lambda\boldsymbol{v}^*\Delta\boldsymbol{v}$ is also real. Let $\lambda = a + b\mathbf{i}$, where $a = \Re\lambda$ and $b = \Im\lambda$. Then

$$\lambda\boldsymbol{v}^*\Delta\boldsymbol{v} = a\boldsymbol{v}^*\Delta\boldsymbol{v} + (b\boldsymbol{v}^*\Delta\boldsymbol{v})\mathbf{i}$$

and, therefore, if $b \neq 0$, we must have $\boldsymbol{v}^*\Delta\boldsymbol{v} = 0$. □

First consider the two following basic facts.

Fact 2.4.1 *Let* $V = \left[\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n\right]$ *be the matrix of eigenvectors of* ΔT*. If* $\lambda_i \neq \lambda_j$*,* $i \neq j$*, then* V *is invertible.*

The proof of this fact rests on the invertibility of the Vandermonde matrix (eigenvectors corresponding to distinct eigenvalues are linearly independent).

Fact 2.4.2 *If* $\lambda_i \neq \lambda_j$ *for* $i \neq j$*, then* $\boldsymbol{v}_i^T\Delta\boldsymbol{v}_j = 0$*.*

Proof. From (2.45), suppose that $\lambda_i \neq \lambda_j$ and vectors $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ satisfy

$$T\boldsymbol{v}_j = \Delta\boldsymbol{v}_j\lambda_j \qquad \text{and} \qquad T\boldsymbol{v}_i = \Delta\boldsymbol{v}_i\lambda_i.$$

Then

$$\boldsymbol{v}_i^TT\boldsymbol{v}_j = \boldsymbol{v}_i^T\Delta\boldsymbol{v}_j\lambda_j \qquad \text{and} \qquad \boldsymbol{v}_j^TT\boldsymbol{v}_i = \boldsymbol{v}_j^T\Delta\boldsymbol{v}_i\lambda_i. \tag{2.47}$$

Note that, since we are dealing with scalars and T is symmetric,

$$\boldsymbol{v}_i^TT\boldsymbol{v}_j = \left(\boldsymbol{v}_i^TT\boldsymbol{v}_j\right)^T = \boldsymbol{v}_j^TT\boldsymbol{v}_i \qquad \text{and} \qquad \boldsymbol{v}_i^T\Delta\boldsymbol{v}_j = \left(\boldsymbol{v}_i^T\Delta\boldsymbol{v}_j\right)^T = \boldsymbol{v}_j^T\Delta\boldsymbol{v}_i.$$

Now, subtracting in (2.47) , we get

$$0 = \boldsymbol{v}_i^T\Delta\boldsymbol{v}_j(\lambda_j - \lambda_i)$$

and, since $\lambda_i \neq \lambda_j$, we obtain that

$$\boldsymbol{v}_i^T \Delta \boldsymbol{v}_j = 0. \ \square$$

Lemma 2.4.2 *Let $\boldsymbol{v}_i$, $i = 1, \ldots, n$ be eigenvectors of ΔT. Assume that $\lambda_i \neq \lambda_j$ for $i \neq j$. Then $\boldsymbol{v}_i^T \Delta \boldsymbol{v}_i \neq 0$, $i = 1, \ldots, n$.*

Proof. Using both facts above, we have that $V^T \Delta V$ is a diagonal matrix with diagonal entries $\boldsymbol{v}_i^T \Delta \boldsymbol{v}_i$, $i = 1, \ldots, n$. Thus

$$\det(V^T \Delta V) = \prod_{i=1}^{n} \boldsymbol{v}_i^T \Delta \boldsymbol{v}_i.$$

Since both V and Δ are invertible, we must have $\det(V^T \Delta V) \neq 0$ and, therefore,

$$\boldsymbol{v}_i^T \Delta \boldsymbol{v}_i \neq 0, \qquad i = 1, \ldots, n. \ \square$$

Theorem 2.4.2 *Assume that T admits the factorization $T = LDL^T$, where L is unit bidiagonal with subdiagonal elements $l_1, l_2, \ldots, l_n$ and $D = diag(d_1, d_2, \ldots, d_n)$. Suppose that all the eigenvalues of ΔT are distinct. Then for $\lambda \neq 0$ and corresponding eigenvector $\boldsymbol{v} = (v_j)$ we have, for $k = 1, \ldots, n$,*

$$\frac{d_k}{\lambda} \frac{\partial \lambda}{\partial d_k} = \frac{\left(\boldsymbol{v}^T L \boldsymbol{e}_k\right) \left(\boldsymbol{e}_k^T L^{-1} \Delta \boldsymbol{v}\right)}{\boldsymbol{v}^T \Delta \boldsymbol{v}}$$

and, for $k = 1, \ldots, n-1$,

$$\frac{1}{\lambda} \frac{\partial \lambda}{\partial l_k} = \frac{2 v_{k+1} \left(\boldsymbol{e}_k^T L^{-1} \Delta \boldsymbol{v}\right)}{\boldsymbol{v}^T \Delta \boldsymbol{v}},$$

And

$$\boldsymbol{e}_k^T L^{-1} \Delta \boldsymbol{v} = \delta_k v_k - \delta_{k-1} v_{k-1} l_{k-1} + (-1)^2 \delta_{k-2} v_{k-2} l_{k-1} l_{k-2} + \quad \ldots + (-1)^{k-1} \delta_1 v_1 l_{k-1} l_{k-2} \ldots l_1,$$
$$k = 1, \ldots, n,$$

$$\boldsymbol{v}^T L \boldsymbol{e}_k = v_k + v_{k+1} l_k, \qquad k = 1, \ldots, n-1,$$

$$\boldsymbol{v}^T L \boldsymbol{e}_n = v_n.$$

Proof. Differentiate

$$T\boldsymbol{v} = \lambda\Delta\boldsymbol{v} \tag{2.48}$$

with respect to d_k to obtain

$$\frac{\partial}{\partial d_k}(T\boldsymbol{v}) = \frac{\partial}{\partial d_k}(\lambda\Delta\boldsymbol{v}).$$

Since we have $T = LDL^T$ it follows that

$$\frac{\partial(LDL^T)}{\partial d_k}\boldsymbol{v} + T\frac{\partial \boldsymbol{v}}{\partial d_k} = \frac{\partial\lambda}{\partial d_k}\Delta\boldsymbol{v} + \lambda\frac{\partial(\Delta\boldsymbol{v})}{\partial d_k}. \tag{2.49}$$

Since $\dfrac{\partial D}{\partial d_k} = \boldsymbol{e}_k\boldsymbol{e}_k^T$, $\dfrac{\partial L}{\partial d_k} = \dfrac{\partial L^T}{\partial d_k} = O$ and $\lambda\dfrac{\partial(\Delta\boldsymbol{v})}{\partial d_k} = \lambda\Delta\dfrac{\partial\boldsymbol{v}}{\partial d_k}$, from (2.49) we get

$$L\boldsymbol{e}_k\boldsymbol{e}_k^TL^T\boldsymbol{v} + (T-\lambda\Delta)\frac{\partial\boldsymbol{v}}{\partial d_k} = \frac{\partial\lambda}{\partial d_k}\Delta\boldsymbol{v}.$$

Now, multiplying by $\boldsymbol{v}^T$ on the left,

$$\boldsymbol{v}^TL\boldsymbol{e}_k\boldsymbol{e}_k^TL^T\boldsymbol{v} + \boldsymbol{v}^T(T-\lambda\Delta)\frac{\partial\boldsymbol{v}}{\partial d_k} = \frac{\partial\lambda}{\partial d_k}\boldsymbol{v}^T\Delta\boldsymbol{v}.$$

From this we obtain

$$\frac{\partial\lambda}{\partial d_k} = \frac{\left(\boldsymbol{v}^TL\boldsymbol{e}_k\right)\left(\boldsymbol{e}_k^TL^T\boldsymbol{v}\right)}{\boldsymbol{v}^T\Delta\boldsymbol{v}},$$

because $\boldsymbol{v}^T(T-\lambda\Delta) = [(T-\lambda\Delta)\boldsymbol{v}]^T = 0$, since T and Δ are symmetric, and $\boldsymbol{v}^T\Delta\boldsymbol{v} \neq 0$ by lemma (2.4.2). And multiplying both sides by d_k,

$$d_k\frac{\partial\lambda}{\partial d_k} = \frac{\left(\boldsymbol{v}^TL\boldsymbol{e}_k\right)\left(d_k\boldsymbol{e}_k^TL^T\boldsymbol{v}\right)}{\boldsymbol{v}^T\Delta\boldsymbol{v}}.$$

To make λ appear in the denominator, note that, using (2.48) we can write

$$d_k\boldsymbol{e}_k^TL^T\boldsymbol{v} = \boldsymbol{e}_k^TDL^T\boldsymbol{v} = \boldsymbol{e}_k^TL^{-1}LDL^T\boldsymbol{v} = \boldsymbol{e}_k^TL^{-1}T\boldsymbol{v} = \boldsymbol{e}_kL^{-1}\Delta\boldsymbol{v}\lambda. \tag{2.50}$$

Then when $\lambda \neq 0$,

$$\frac{d_k}{\lambda}\frac{\partial\lambda}{\partial d_k} = \frac{\left(\boldsymbol{v}^TL\boldsymbol{e}_k\right)\left(\boldsymbol{e}_k^TL^{-1}\Delta\boldsymbol{v}\right)}{\boldsymbol{v}^T\Delta\boldsymbol{v}}.$$

The second relation is obtained similarly. Differentiating (2.48) with respect to l_k we obtain

$$\frac{\partial(LDL^T)}{\partial l_k}\boldsymbol{v} + T\frac{\partial \boldsymbol{v}}{\partial l_k} = \frac{\partial \lambda}{\partial l_k}\Delta\boldsymbol{v} + \lambda\frac{\partial(\Delta\boldsymbol{v})}{\partial l_k}. \tag{2.51}$$

Since $\dfrac{\partial D}{\partial l_k} = O$, $\dfrac{\partial L}{\partial l_k} = \boldsymbol{e}_{k+1}\boldsymbol{e}_k^T$, $\dfrac{\partial L^T}{\partial l_k} = \boldsymbol{e}_k\boldsymbol{e}_{k+1}^T$ and $\lambda\dfrac{\partial(\Delta\boldsymbol{v})}{\partial d_k} = \lambda\Delta\dfrac{\partial \boldsymbol{v}}{\partial d_k}$, it follows that

$$\boldsymbol{e}_{k+1}\boldsymbol{e}_k^T DL^T\boldsymbol{v} + LD\boldsymbol{e}_k\boldsymbol{e}_{k+1}^T\boldsymbol{v} + (T - \lambda\Delta)\frac{\partial \boldsymbol{v}}{\partial l_k} = \frac{\partial \lambda}{\partial l_k}\Delta\boldsymbol{v}.$$

Again, pre-multiplying by $\boldsymbol{v}^T$ and using the same arguments as before,

$$\begin{aligned}\frac{\partial \lambda}{\partial l_k} &= \frac{\boldsymbol{v}^T\boldsymbol{e}_{k+1}\left(\boldsymbol{e}_k^T DL^T\boldsymbol{v}\right) + \left(\boldsymbol{v}^T LD\boldsymbol{e}_k\right)\boldsymbol{e}_{k+1}^T\boldsymbol{v}}{\boldsymbol{v}^T\Delta\boldsymbol{v}} \\ &= \frac{v_{k+1}\left(\boldsymbol{e}_k^T DL^T\boldsymbol{v}\right) + \left(\boldsymbol{e}_k^T DL^T\boldsymbol{v}\right)^T v_{k+1}}{\boldsymbol{v}^T\Delta\boldsymbol{v}}\end{aligned}$$

Using again (2.50) it follows

$$\frac{1}{\lambda}\frac{\partial \lambda}{\partial l_k} = \frac{2v_{k+1}\left(\boldsymbol{e}_k^T L^{-1}\Delta\boldsymbol{v}\right)}{\boldsymbol{v}^T\Delta\boldsymbol{v}}.$$

Finally, the result

$$\boldsymbol{e}_k^T L^{-1}\Delta\boldsymbol{v} = \delta_k v_k - \delta_{k-1}v_{k-1}l_{k-1} + (-1)^2\delta_{k-2}v_{k-2}l_{k-1}l_{k-2} + \quad \ldots + (-1)^{k-1}\delta_1 v_1 l_{k-1}l_{k-2}\ldots l_1,$$
$$k = 1, \ldots, n,$$

$$\boldsymbol{v}^T L\boldsymbol{e}_k = v_k + v_{k+1}l_k, \qquad k = 1, \ldots, n-1,$$
$$\boldsymbol{v}^T L\boldsymbol{e}_n = v_n.$$

follows from direct calculations. □

Notice that d_k, like λ, is proportional to $\|T\|$ while l_k is not. That is, if we multiply T by $\mu \neq 0$, then $L(\mu D)L^T = \mu T$. Consequently, it is natural to consider relative changes to d_k but absolute changes to l_k. In particular, when $k = 1$,

$$\frac{d_1}{\lambda}\frac{\partial \lambda}{\partial d_1} = \frac{(v_1 + v_2 l_1)\delta_1 v_1}{\boldsymbol{v}^T\Delta\boldsymbol{v}}$$
$$\frac{1}{\lambda}\frac{\partial \lambda}{\partial l_1} = \frac{2v_2\delta_1 v_1}{\boldsymbol{v}^T\Delta\boldsymbol{v}}$$

Theorem (2.4.2) permits us to define the following condition numbers

$$\boxed{\text{relcond}(\lambda, d_k; \Delta LDL^T) := \frac{\left|\boldsymbol{v}^T L \boldsymbol{e}_k\right| \left|\boldsymbol{e}_k^T L^{-1} \Delta \boldsymbol{v}\right|}{|\boldsymbol{v}^T \Delta \boldsymbol{v}|}}$$

and

$$\boxed{\text{relcond}(\lambda, l_k; \Delta LDL^T) := \frac{2|v_{k+1}| \left|\boldsymbol{e}_k^T L^{-1} \Delta \boldsymbol{v}\right|}{|\boldsymbol{v}^T \Delta \boldsymbol{v}|}}$$

A comparative study of all these measures of sensitivity we just presented with the more general ones would be useful and is part of the next stage of our study. Classical Bauer-Fike theorem states one condition number for all the eigenvalues and must certainly be too crude. Wilkinson's condition number gives one condition number,

$$\frac{\|\boldsymbol{y}^*\| \, \|\boldsymbol{x}\|}{|\lambda| |\boldsymbol{y}^* \boldsymbol{x}|},$$

for each eigenvalue but must also be pessimistic. It is still too general. The condition numbers we gave are more refined because they exploit the tridiagonal form. The hope is that they will be realistic estimates of sensitivity, not just bounds.

Chapter 3

LR and dqds

The focus of this chapter is the description of LR and qds algorithms showing the connection between them. The dqds version demands a little more of arithmetic effort than qds but has compensating advantages - it enjoys the property of mixed high relative stability.

3.1 LR algorithm

For more than forty years the standard algorithm for calculating eigenvalues has been the QR algorithm of Francis [18]. But historically the LR algorithm preceded the QR algorithm. The whole field had its genesis in 1957 with Rutishauser's quotient-difference algorithm [46, 47], which Rutishauser then generalized to the LR algorithm [48]. The QR algorithm followed shortly thereafter. Surprisingly, the quotient-difference algorithm has had a recent revival. In 1994, Fernando and Parlett [16] introduced new versions for finding singular values of bidiagonal matrices and eigenvalues of symmetric tridiagonal matrices but it may also be used in the unsymmetric tridiagonal case.

Contrary to QR, the LR algorithm preserves bandwith and thus it is one of the most efficient methods for calculating all the eigenvalues of a nonsymmetric tridiagonal matrix. Most of the improvements that have been incorporated into QR over the years such as implicit double-shift iterations, deflation, splitting, and arbitrary shifts, can also be used in

the context of LR iteration.

LR algorithm

The LR transformation gives a reduction of a general matrix to triangular form by means of non-unitary similarity transformations. It is based on the LU decomposition of a matrix. Under mild conditions a square matrix admits triangular factorization, $A = LR$, where L is unit lower-triangular and R is upper triangular.

The method bases essentially on the fact that by starting with the given matrix $A =: A_1$, the infinite sequence of similar matrices $A_1, A_2, A_3, \ldots$, generated by

LR algorithm

$$\begin{aligned} &A_1 = A \\ &\textbf{for } i = 1, 2, \ldots \\ &\quad \text{Factor } A_i = L_i R_i \qquad \text{(LU factorization)} \\ &\quad A_{i+1} = R_i L_i \\ &\textbf{end} \end{aligned} \tag{3.1}$$

converges to a triangular or diagonal matrix A_{inf}, under reasonable assumptions.

In words, we decompose A_i and multiply the factors in reverse order to obtain a matrix A_{i+1} that is similar to A. This process is repeated until convergence occurs. In fact,

$$A_{i+1} = R_i L_i = L_i^{-1}(L_i R_i) L_i = L_i^{-1} A_i L_i$$

and then, by induction, A_{i+1} is similar to A_1.

If A is symmetric and positive definite, and if the decomposition of A_i into $L_i R_i$ is such that L_i is lower triangular and matrix R_i is the transpose of L_i for any i (*Cholesky decomposition*), then $\lim_{i \to \infty} A_i$ exists and is diagonal. A proof of this is given in Rutishauser [48].

But not every matrix has an LU decomposition and difficulties arise when we are close to a matrix that has no LU decomposition. The LR algorithm may not be stable. Stability

can be improved markedly by the introduction of pivoting but this destroys the bandwith structure. Wilkinson [60] uses the LR method with row and column interchanges to produce an algorithm which is stable and requires no more arithmetic than the basic LR. In practice, this variation seems both fast and accurate but no convergence proof has been published yet nor is it apparent how to adapt the technique to give, economically, the complex eigenvalues of real matrices.

3.2 QR algorithm

A standard approach in computing the eigenvalues of a general square matrix is to reduce the matrix first to Hessenberg form by a sequence of orthogonal transformations, and then to determine the eigenvalues of the Hessenberg matrix through an iterative process known as the QR algorithm. The reduction to Hessenberg form requires $\mathcal{O}(n^3)$ operations, where n is the order of the matrix, and the subsequent iterative phase also requires $\mathcal{O}(n^3)$ operations. The function `eig` of MATLAB uses this scheme to compute all of the eigenvalues and eigenvectors of a general matrix.

If the original matrix is symmetric, then the symmetry can be preserved in the initial reduction, so the reduced matrix is tridiagonal. Although the reduction to tridiagonal form still requires $\mathcal{O}(n^3)$ operations, the subsequent iterations preserve the tridiagonal form and, hence, are much less expensive, so that the total cost of the iterative phase is reduced to $\mathcal{O}(n^2)$ operations.

The attractively low operation count obtained when iterating with a tridiagonal matrix suggests that the tridiagonal form would be extremely beneficial in the nonsymmetric case as well. Such an approach presents two difficulties, however. First, QR iteration does not preserve the structure of a nonsymmetric tridiagonal matrix. This problem can be overcome by using LR iteration, which preserves the tridiagonal form. Second, it is difficult to reduce a nonsymmetric matrix to tridiagonal form by similarity in a numerically stable manner. But promising methods to improve the stability of this reduction have been developed recently. See Sidje [50]. See also Dongarra [13] and Geist [20].

QR algorithm

As implemented in the late 1950's and early 1960's, LR proved insufficiently reliable and was displaced by the QR algorithm in the mid 1960's. The QR algorithm, developed by Francis [18], is closely related to the LR algorithm. Based on the use of unitary transformations, the QR algorithm avoids the possible instability of the original LR transformation but it is slower. However, in many respects, this has proved to be the most effective of known methods for the solution of the general algebraic eigenvalue problem. In comparison with current reliable methods, the QR technique is fast and highly satisfactory.

QR algorithm

$A_1 = A$
for $i = 1, 2, \ldots$
  Factor $A_i = Q_i R_i$ (QR factorization)
  $A_{i+1} = R_i Q_i$
end

Orthogonal similarities give stability to the QR algorithm and preserve symmetry as well as the Hessenberg form; thus, the symmetric tridiagonal form is also preserved.

Lemma 3.2.1 *Let $A = QR$ be the* QR *decomposition of A and*

$$\hat{A} = Q^T A Q$$

be its QR *transformation. If A is tridiagonal, symmetric and non-singular, then $\hat{A}$ is also tridiagonal, symmetric and non-singular.*

The QR algorithm does not preserve the tridiagonal form in the non-symmetric case. If A is non-symmetric tridiagonal, then $\hat{A}$ will be upper Hessenberg. But it never breaks down.

Convergence of the basic QR algorithm may be too poor for pratical computations. Convergence rates depend on the ratios of eigenvalues and to speed up convergence, shifts of origin should be incorporated.

QR algorithm with a shift

$A_1 = A$
for $i = 1, 2, \ldots$
  Choose a shift σ_i
  Factor $A_i - \sigma_i I = Q_i R_i$ (QR factorization)
  $A_{i+1} = Q_i R_i + \sigma_i I$
end

Ideally we want to shift by the eigenvalues. If σ_i is an exact eigenvalue of A_i, then the QR iteration converges in one step. See Demmel [11, p.162]. So, shifts σ_i close to eigenvalues hasten convergence. Note that near convergence to a real eigenvalue, the last diagonal element of $A_i = (a_{kj}^{(i)})$ is close to that eigenvalue, so $\sigma_i = a_{nn}^{(i)}$ is a good choice for a shift.

3.3 Shifted LR algorithm

So, in order to accelerate convergence (and also to be able to treat indefinite symmetric matrices) the simple decomposition-recombination procedure (3.1) of LR algorithm must be modified in the following way:

LR algorithm with a shift

$A_1 = A$
for $i = 1, 2, \ldots$
  Choose a shift σ_i
  Factor $A_i - \sigma_i I = L_i R_i$ (LU factorization)
  $A_{i+1} = R_i L_i + \sigma_i I$
end

Notice that matrices A_i and A_{i+1} continue to be similar,

$$A_{i+1} = R_i L_i + \sigma_i I = L_i^{-1}(L_i R_i + \sigma_i I)L_i = L_i^{-1} A_i L_i, \tag{3.2}$$

and we call the transformation *shift restoring*.

By proper choice of σ_i, the last diagonal element $a_{nn}^{(i)}$ of A_i will, in general, converge to the smallest eigenvalue of A. If σ_i is chosen to be close to an eigenvalue of A_i then $A_i - \sigma_i I$ has an eigenvalue close to zero.

There are no simple expressions for convergence rates in shifted LR algorithm because it depends on all shifts σ_i's.

Lemma 3.3.1 *If $A - \sigma I$ is singular and $A - \sigma I = LR$ is the* LU *decomposition of $A - \sigma I$, then the transformation $\hat{A} = L^{-1}AL$ deflates in one step.*

Proof. Let A be of order n. Since $A - \sigma I$ is singular, if $L = (l_{ij})$ and $R = (r_{ij})$ are the factors of its LU factorization, then, by theorem 2.1.1 (page 27), we must have $r_{nn} = 0$. Therefore,

$$L^{-1}AL = L^{-1}(LR + \sigma I)L = RL + \sigma I.$$

But $\boldsymbol{e}_n^T R = \mathbf{0}^T$ and then

$$\boldsymbol{e}_n^T(RL + \sigma I) = \mathbf{0}^T + \sigma \boldsymbol{e}_n^T = \sigma \boldsymbol{e}_n^T. \ \square$$

To say that $\hat{A} = L^{-1}AL$ deflates in one step, means that the first eigenvalue σ is obtained and the remaining eigenvalues of A are the eigenvalues of the submatrix that results from discarding the last row and the last column of $\hat{A}$. This reduction of the order of the eigenproblem is called *deflation*.

In the general case, with LR algorithm there is still much work to be done in finding clever shift strategies that approach an eigenvalue in a stable way. The development of reliable shift strategies is an open domain.

3.4 Implicit double shifted LR algorithm

When $A =: A_1$ is a real unsymmetric matrix and has complex eigenvalues, to speed up the convergence to a complex eigenvalue it is necessary to choose a shift close to an eigenvalue and hence to choose a complex shift. If on iteration i we take a complex shift σ_i, then the resulting matrix $A_{i+1} = R_i L_i + \sigma_i I$ will be complex. This means that all arithmetic has to be complex, increasing the cost by a factor of about 4. Since complex eigenvalues of real matrices occur in complex conjugate pairs, we can shift by σ_i and $\overline{\sigma_i}$ at the same time. It turns out that this will permit us to maintain real arithmetic. This *double shift* technique was developed by Francis for the QR algorithm: it uses real arithmetic and converges to real Schur canonical form. For the details on the *implicit double shifted* QR algorithm see Demmel [10, pp.170-172]. In this section we will describe a similar technique for the LR algorithm.

Implicit double shifted LR algorithm

We will see in detail how to combine two consecutive iterations of LR algorithm choosing successive complex shifts σ and $\overline{\sigma}$ such that the result after this double shift is again real.

Considering the first two steps, the result of shifting by σ and $\overline{\sigma}$ in succession are

$$\begin{aligned} A_1 - \sigma I =& L_1 R_1 \\ A_2 =& R_1 L_1 + \sigma I \quad \text{so} \quad A_2 = L_1^{-1} A_1 L_1 \qquad (3.3) \\ A_2 - \overline{\sigma} I =& L_2 R_2 \\ A_3 =& R_2 L_2 + \overline{\sigma} I \quad \text{so} \quad A_3 = L_2^{-1} A_2 L_2 = L_2^{-1} L_1^{-1} A_1 L_1 L_2 \qquad (3.4) \end{aligned}$$

The real part of σ will be denoted by $\Re\sigma$ and the imaginary part by $\Im\sigma$.

Lemma 3.4.1 *Consider the application of two steps of* LR *algorithm with successive shifts* σ *and* $\overline{\sigma}$*. Then*

$$(L_1 L_2)(R_2 R_1) = A_1^2 - 2(\Re\sigma) A_1 + |\sigma|^2 I.$$

Proof. Since $L_2R_2 = A_2 - \overline{\sigma}I$ and $L_1R_1 = A_1 - \sigma I$, we get

$$\begin{aligned} L_1L_2R_2R_1 &= L_1(A_2 - \overline{\sigma}I)R_1 \\ &= L_1(L_1^{-1}A_1L_1 - \overline{\sigma}I)R_1 \\ &= (A_1 - \overline{\sigma}I)L_1R_1 \\ &= (A_1 - \overline{\sigma}I)(A_1 - \sigma I) \\ &= A_1^2 - 2(\Re\sigma)A_1 + |\sigma|^2 I. \ \square \end{aligned}$$

Define $M \equiv A_1^2 - 2(\Re\sigma)A_1 + |\sigma|^2 I$. Thus $(L_1L_2)(R_2R_1)$ is the LU decomposition of the real matrix M and so (L_1L_2) and (R_2R_1) are both real. This means that

$$A_3 = (L_1L_2)^{-1}A_1(L_1L_2)$$

is also real.

Moreover, the first column of L_1L_2 is proportional to the first column of M:

$$\begin{aligned} M\boldsymbol{e}_1 &= (L_1L_2R_2R_1)\boldsymbol{e}_1 \\ &= r(L_1L_2)\boldsymbol{e}_1, \end{aligned}$$

where r is the element in position $(1,1)$ of upper triangular matrix R_2R_1 (remember that $\boldsymbol{e}_1$ is the first unit vector).

Our implementation of double shifted LR iteration will depend on the following theorem, known as the *Implicit L Theorem*. We will suppose that a reduction of the initial matrix A to Hessenberg form has been performed. Thus, we will be considering double shifted LR on an Hessenberg matrix which is also a form that is preserved by this algorithm.

Theorem 3.4.1 IMPLICIT L THEOREM *Suppose that L is unit lower triangular and $L^{-1}AL = H$ is unreduced upper Hessenberg. Then columns 2 through n of L are determined uniquely by the first column of L.*

Proof. Let $\boldsymbol{l}_j$ and $\boldsymbol{h}_j$, $1 \leq j \leq n$, be the columns of unit lower triangular matrix $L = (l_{ij})_{1\leq i,j\leq n}$ and upper unreduced Hessenberg matrix $H = (h_{ij})_{1\leq i,j\leq n}$, respectively.

Thus,

$$L = \begin{bmatrix} \boldsymbol{l}_1 & \boldsymbol{l}_2 & \boldsymbol{l}_3 & \cdots & \boldsymbol{l}_n \end{bmatrix} \qquad \text{and} \qquad H = \begin{bmatrix} \boldsymbol{h}_1 & \boldsymbol{h}_2 & \boldsymbol{h}_3 & \cdots & \boldsymbol{h}_n \end{bmatrix}$$
$$= \begin{bmatrix} 1 & & & & \\ l_{21} & 1 & & & \\ l_{31} & l_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{n,n-1} & 1 \end{bmatrix} \qquad = \begin{bmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1n} \\ h_{21} & h_{22} & h_{23} & \cdots & h_{2n} \\ & h_{32} & h_{33} & \cdots & h_{3n} \\ & & \ddots & \ddots & \vdots \\ & & & h_{n,n-1} & h_{nn} \end{bmatrix}.$$

Equating entries on each side of the equation

$$AL = LH$$

shows, for the first column, that

$$A\boldsymbol{l}_1 = L\boldsymbol{h}_1 = \boldsymbol{l}_1 h_{11} + \boldsymbol{l}_2 h_{21}. \tag{3.5}$$

Then, looking at the first and second elements of $A\boldsymbol{l}_1$, we get

$$\boldsymbol{e}_1^T A\boldsymbol{l}_1 = h_{11} \qquad \text{and} \qquad \boldsymbol{e}_2^T A\boldsymbol{l}_1 = l_{21}h_{11} + h_{21}.$$

So, given a matrix A and the first column $\boldsymbol{l}_1$ of L, elements h_{11} and h_{21} are uniquely determined. That is, the first column $\boldsymbol{h}_1$ of H is uniquely determined by $\boldsymbol{l}_1$.

Now we can use (3.5) to obtain the second column of L,

$$\boldsymbol{l}_{i2} = \left(A\boldsymbol{l}_1 - \boldsymbol{l}_1 h_{11}\right)_{i2} / h_{21}, \qquad i = 3, \ldots, n$$

since, by assumption, $h_{21} \neq 0$.

To obtain columns $\boldsymbol{h}_j$ and $\boldsymbol{l}_{j+1}$, $2 \leq j \leq n-1$, notice that

$$A\boldsymbol{l}_j = L\boldsymbol{h}_j = \sum_{i=1}^{j} \boldsymbol{l}_i h_{ij} + \boldsymbol{l}_{j+1} h_{j+1,j}. \tag{3.6}$$

Then, the first $j+1$ elements of $A\boldsymbol{l}_j$ are

$$\begin{aligned}
\boldsymbol{e}_1^T A\boldsymbol{l}_j &= h_{1j} \\
\boldsymbol{e}_2^T A\boldsymbol{l}_j &= h_{1j}l_{21} + h_{2j} \\
&\vdots \\
\boldsymbol{e}_j^T A\boldsymbol{l}_j &= h_{1j}l_{j1} + h_{2j}l_{j2} + \cdots + h_{jj} \\
\boldsymbol{e}_{j+1}^T A\boldsymbol{l}_j &= h_{1j}l_{j+1,1} + h_{2j}l_{j+1,2} + \cdots + h_{jj}l_{j+1,j} + h_{j+1,j}.
\end{aligned} \tag{3.7}$$

Given columns $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_j$, these equations determine, in turn, elements $h_{1j}, \ldots, h_{j+1,j}$, that is, column $\boldsymbol{h}_j$ of H. And, since $h_{j+1,j} \neq 0$, from (3.6) we get column $\boldsymbol{l}_{j+1}$,

$$\boldsymbol{l}_{i,j+1} = \frac{1}{h_{j+1,j}} \left(A\boldsymbol{l}_j - \sum_{i=1}^{j} \boldsymbol{l}_i h_{ij} \right)_{i,j+1}, \qquad i = j+2, \ldots, n.$$

Note that if $j = n-1$ we have $\boldsymbol{l}_{j+1} = \boldsymbol{e}_n$ and there is no need for any calculation.

Finally, for $j = n$, we obtain the last column of H, $\boldsymbol{h}_n = \begin{bmatrix} h_{1n} \ldots h_{nn} \end{bmatrix}^T$, the same way, but using only the first j equations in (3.7). $\square$

The implicit L theorem implies that in the double shifted LR algorithm, to compute A_3 in (3.4) from A_1 we will only need to:

1. compute the first column of L_1L_2 which is proportional to the first column of M and so can be gotten just by normalizing this column vector.

2. compute other columns of L_1L_2 implicitly using the implicit L theorem through elementary transformations.

Next we will describe a technique called *bulge chasing* that, after performing 1, allow us to achieve goal 2.

3.4.1 Bulge chasing

The technique of bulge chasing is justified by the implicit L theorem. This theorem plays the same role as the *Implicit Q Theorem* [10, p.168] in the implementation of the implicit double shifted QR algorithm.

When $A = (a_{ij})_{1\leq i,j\leq n}$ is upper Hessenberg, then the first column of M is

$$\begin{bmatrix} a_{11}^2 + a_{12}a_{21} - 2(\Re\sigma)a_{11} + |\sigma|^2 \\ a_{21}(a_{11} + a_{22} - 2(\Re\sigma)) \\ a_{21}a_{32} \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{3.8}$$

We have

$$A_3 = \mathbf{L}^{-1}A_1\mathbf{L} \tag{3.9}$$

where $\mathbf{L} \equiv L_1L_2$ and the first column of $\mathbf{L}$ is proportional to (3.8). We will present a 6×6 example of bulge chasing to show how we obtain A_3 and $\mathbf{L}$ from A_1.

Example 3.4.1

If we consider $n = 6$, the transformation of A_1 to A_3 occurs accordingly to the following steps.

1. Let $\mathbf{l}_1 = \begin{bmatrix} 1 & * & * & 0 & 0 & 0 \end{bmatrix}^T$ be the first column of $\mathbf{L}$, obtained just by normalizing (3.8). Consider the elementary matrix $\mathbf{L}_1 = I + \mathbf{l}_1'e_1^T$, where $\mathbf{l}_1' = \begin{bmatrix} 0 & * & * & 0 & 0 & 0 \end{bmatrix}^T$, that is $\mathbf{l}_1$ with 0 in first position instead of 1. So, since $\mathbf{L}_1^{-1} = I - \mathbf{l}_1'e_1^T$,

$$\mathbf{L}_1^{-1}A_1 = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ + & x & x & x & x & x \\ & & x & x & x & x \\ & & & x & x & x \\ & & & & x & x \end{bmatrix} \quad \text{and} \quad A^{(1)} \equiv \mathbf{L}_1^{-1}A_1\mathbf{L}_1 = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ + & x & x & x & x & x \\ + & & x & x & x & x \\ & & & x & x & x \\ & & & & x & x \end{bmatrix}.$$

We see that there is a 2×1 bulge, indicated by plus signs.

Next we will apply a sequence of elementary similarity transformations such that each transformation pushes the bulge one row down and one column to the right. Finally the bulge is gotten rid of to restore upper Hessenberg form.

2. Form an elementary matriz (Gauss matriz) $\mathbf{L}_2^{-1}$ which affects only rows 3 and 4 of $\mathbf{L}_2^{-1}A^{(1)}$, zeroing out entries $(3,1)$ and $(4,1)$ of $A^{(1)}$. So, $\mathbf{L}_2 = I + \mathbf{l}_2' e_2^T$ with $\mathbf{l}_2' = \begin{bmatrix} 0 & 0 & * & * & 0 & 0 \end{bmatrix}^T$, $\mathbf{L}_2^{-1} = I - \mathbf{l}_2' e_2^T$ and

$$\mathbf{L}_2^{-1}A^{(1)} = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ & x & x & x & x & x \\ & + & x & x & x & x \\ & & & x & x & x \\ & & & & x & x \end{bmatrix} \text{ and } A^{(2)} \equiv \mathbf{L}_2^{-1}A^{(1)}\mathbf{L}_2 = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ & x & x & x & x & x \\ & + & x & x & x & x \\ & + & & x & x & x \\ & & & & x & x \end{bmatrix}.$$

3. Form an elementary matriz $\mathbf{L}_3^{-1}$ which affects only rows 4 and 5 of $\mathbf{L}_3^{-1}A^{(2)}$, zeroing out entries $(4,2)$ and $(5,2)$ of $A^{(2)}$. We will have $\mathbf{L}_3 = I + \mathbf{l}_3' e_3^T$ with $\mathbf{l}_3' = \begin{bmatrix} 0 & 0 & 0 & * & * & 0 \end{bmatrix}^T$, $\mathbf{L}_3^{-1} = I - \mathbf{l}_3' e_3^T$ and

$$\mathbf{L}_3^{-1}A^{(2)} = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ & x & x & x & x & x \\ & & x & x & x & x \\ & & + & x & x & x \\ & & & & x & x \end{bmatrix} \text{ and } A^{(3)} \equiv \mathbf{L}_3^{-1}A^{(2)}\mathbf{L}_3 = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ & x & x & x & x & x \\ & & x & x & x & x \\ & & + & x & x & x \\ & & + & & x & x \end{bmatrix}.$$

4. Form an elementary matriz $\mathbf{L}_4^{-1}$ which affects only rows 5 and 6 of $\mathbf{L}_4^{-1}A^{(3)}$, zeroing out entries $(5,3)$ and $(6,3)$ of $A^{(3)}$. We will have $\mathbf{L}_4 = I + \mathbf{l}_4' e_4^T$ with

$\mathbf{l}_4' = \begin{bmatrix} 0 & 0 & 0 & 0 & * & * \end{bmatrix}^T$, $\mathbf{L}_4^{-1} = I - \mathbf{l}_4' \boldsymbol{e}_4^T$ and

$$A^{(4)} \equiv \mathbf{L}_4^{-1} A^{(3)} \mathbf{L}_4 = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ & x & x & x & x & x \\ & & x & x & x & x \\ & & & x & x & x \\ & & & + & x & x \end{bmatrix}.$$

5. Finally to zero out entry $(6,4)$ of $A^{(4)}$ we form $\mathbf{L}_5 = I + \mathbf{l}_5' \boldsymbol{e}_5^T$ with $\mathbf{l}_5' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & * \end{bmatrix}^T$. We have $\mathbf{L}_5^{-1} = I - \mathbf{l}_5' \boldsymbol{e_5}^T$,

$$A^{(5)} \equiv \mathbf{L}_5^{-1} A^{(4)} \mathbf{L}_5 = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ & x & x & x & x & x \\ & & x & x & x & x \\ & & & x & x & x \\ & & & & x & x \end{bmatrix}$$

and we are back to upper Hessenberg form.

In the end we have

$$A^{(5)} = (\mathbf{L}_1 \mathbf{L}_2 \mathbf{L}_3 \mathbf{L}_4 \mathbf{L}_5)^{-1} A_1 \mathbf{L}_1 \mathbf{L}_2 \mathbf{L}_3 \mathbf{L}_4 \mathbf{L}_5.$$

In exact arithmetic the implicit L theorem ensures that

$$A_3 = A^{(5)} \quad \text{and} \quad \mathbf{L} = L_1 L_2 = \mathbf{L}_1 \mathbf{L}_2 \mathbf{L}_3 \mathbf{L}_4 \mathbf{L}_5. \diamond$$

The rest of the this chapter is concerned with qd algorithms and its relation to LR algorithm.

3.5 The qd algorithms

Consider a tridiagonal matrix normalized into J form as presented earlier in the previous chapter (page 30), that is, such that all superdiagonal entries are equal to 1. When LR is rewritten in the LU representation for a J matrix one obtains the (progressive) qd algorithm. The letters q and d stand for *quotient* and *difference*. When we incorporate shifts we have the qds algorithm.

The differential qd algorithm (dqd) is a variant of qd that requires a little more of arithmetic effort than qd itself. Shifted version is named dqds. They are new to the scene of matrix computations and one feature that makes them attractive is that they seem to be more accurate than their rivals (see Parlett [40]). In particular, in the positive case, all eigenvalues can be found to high relative accuracy as long as the shifts preserve positivity.

3.5.1 Stationary qd algorithms

Triangular factors change in a complicated way under translation. Given L and U of the factorization $J = LU$ as presented in section 2.2.2 (page 31),

$$L = \begin{bmatrix} 1 & & & & & \\ l_1 & 1 & & & & \\ & l_2 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & l_{n-2} & 1 & \\ & & & & l_{n-1} & 1 \end{bmatrix}, \qquad U = \begin{bmatrix} u_1 & 1 & & & & \\ & u_2 & 1 & & & \\ & & u_3 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & u_{n-1} & 1 \\ & & & & & u_n \end{bmatrix},$$

the task here is to compute $\bar{L}$ and $\bar{U}$ such that

$$J - \sigma I = LU - \sigma I = \bar{L}\bar{U}$$

for a given suitable shift σ. Equating entries on each side shows that

$$\begin{aligned} l_i + u_{i+1} - \sigma = \ & \bar{l}_i + \bar{u}_{i+1}, \quad && i = 0, \ldots, n-1, \ \ l_0 = 0, \ \ \bar{l}_0 = 0, \\ l_i u_i = \ & \bar{l}_i \bar{u}_i, && i = 1, \ldots, n-1. \end{aligned}$$

These relations yield the so-called *stationary* qd algorithm with shift:

$$\begin{aligned}
\text{stqds}(\sigma): \quad & \bar{u}_1 = u_1 - \sigma \\
& \textbf{for } i = 1, \ldots, n-1 \\
& \quad \bar{l}_i = l_i u_i / \bar{u}_i \\
& \quad \bar{u}_{i+1} = l_i + u_{i+1} - \sigma - \bar{l}_i \\
& \textbf{end for}.
\end{aligned}$$

Naturally it fails if $\bar{u}_i = 0$ for some $i < n$.

An alternative algorithm for $\bar{L}$ and $\bar{U}$ involves more arithmetic effort and an auxiliary variable but has some advantages in accuracy for finite-precision arithmetic. To derive this algorithm we define a variable t_i by

$$t_{i+1} \equiv \bar{u}_{i+1} - u_{i+1} = l_i - \bar{l}_i - \sigma.$$

Observe that

$$\begin{aligned}
t_{i+1} = & \quad l_i - l_i u_i / \bar{u}_i - \sigma \\
= & \quad l_i (\bar{u}_i - u_i) / \bar{u}_i - \sigma \\
= & \quad t_i l_i / \bar{u}_i - \sigma.
\end{aligned}$$

The associated algorithm is called the *differential* form of the stationary qds algorithm. We will name it as dstqds.

$$\begin{aligned}
\text{dstqds}(\sigma): \quad & t_1 = -\sigma \\
& \textbf{for } i = 1, \ldots, n-1 \\
& \quad \bar{u}_i = u_i + t_i \\
& \quad \bar{l}_i = u_i (l_i / \bar{u}_i) \\
& \quad t_{i+1} = t_i (l_i / \bar{u}_i) - \sigma \\
& \textbf{end for} \\
& \bar{u}_n = u_n + t_n.
\end{aligned}$$

In practice the t-values may be written over each other in a single variable t. If the common subexpression $l_i/\bar{u}_i$ is recognized then only one division is needed. Thus the dstqds exchanges a subtraction for a multiplication, so the extra cost is not excessive.

3.5.2 Progressive qd algorithms

Recall that the product $UL = J'$ has the superdiagonal entries also equal to 1. This section seeks the triangular factorization of $J' - \sigma I$, not $J - \sigma I$:

$$J' - \sigma I = UL - \sigma I = \hat{L}\hat{U}$$

for a suitable shift σ. Equating entries on each side of the defining equation gives:

$$\begin{aligned} u_{i+1} + l_{i+1} - \sigma = &\quad \hat{l}_i + \hat{u}_{i+1}, \qquad i = 0, \ldots, n-1, \;\; \hat{l}_0 = 0, \;\; l_n = 0, \\ l_i u_{i+1} = &\quad \hat{l}_i \hat{u}_i, \qquad\qquad i = 1, \ldots, n-1. \end{aligned}$$

These relations give the so-called *progressive* qd algorithm with shift which we will call qds.

$$\begin{aligned} \mathrm{qds}(\sigma): \quad & \hat{u}_1 = u_1 + l_1 - \sigma; \\ & \textbf{for } i = 1, \ldots, n-1 \\ & \quad \hat{l}_i = l_i u_{i+1}/\hat{u}_i \\ & \quad \hat{u}_{i+1} = u_{i+1} + l_{i+1} - \sigma - \hat{l}_i \\ & \textbf{end for}. \end{aligned}$$

The algorithm qds fails when $\hat{u}_i = 0$ for some $i < n$. When $\sigma = 0$ we write simply qd, not qds.

There is an alternative implementation of qds that is slightly slower than qds but has compensating advantages. Lets define an auxiliary variable

$$d_{i+1} \equiv \hat{u}_{i+1} - l_{i+1} = u_{i+1} - \hat{l}_i - \sigma.$$

Observe that

$$\begin{aligned} d_{i+1} = &\quad u_{i+1} - l_i u_{i+1}/\hat{u}_i - \sigma \\ = &\quad u_{i+1}(\hat{u}_i - l_i)/\hat{u}_i - \sigma \\ = &\quad d_i u_{i+1}/\hat{u}_i - \sigma. \end{aligned}$$

Rutishauser seems to have discovered the unshifted version some years after discovering qd but he did not make much use of it. He called it the *differential* qd algorithm (dqd) and the new shifted version will be called dqds.

$$
\begin{aligned}
\text{dqds}(\sigma): \quad & d_1 = u_1 - \sigma \\
& \textbf{for } i = 1, \ldots, n-1 \\
& \quad \hat{u}_i = d_i + l_i \\
& \quad \hat{l}_i = l_i(u_{i+1}/\hat{u}_i) \\
& \quad d_{i+1} = d_i(u_{i+1}/\hat{u}_i) - \sigma \\
& \textbf{end for} \\
& \hat{u}_n = d_n.
\end{aligned}
$$

By definition, dqd=dqds(0). In practice each d_{i+1} may be written over its predecessor in a single variable d.

In the positive case ($l_i > 0,\ i = 1, \ldots, n-1;\ u_i > 0,\ i = 1, \ldots, n$) dqd requires no subtractions and enjoys very high relative stability. In the symmetric case, dqds, even with the current simple shift strategies, achieves good accuracy in all eigenvalues and is faster then QR. See Fernando and Parlett [16]. In fact, this algorithm finds the singular values of a bidiagonal matrix in $\mathcal{O}(n^2)$ time, but as accurately and rather more efficiently than QR algorithm. It is the sequential algorithm of choice for singular values and is implemented in LAPACK [1] [2].

In what concerns to an error analysis of dqds, it was proved in [16] that dqds yields *mixed high relative stability*. Given matrices L and D and shift σ, suppose that the dqds algorithm in finite precision produces representable output $\tilde{L}$ and $\tilde{D}$. We introduce ideal matrices $\hat{L}$, $\hat{D}$, $\breve{L}$ and $\breve{D}$ such that $\breve{L}$ and $\breve{D}$ is the output of dqds acting on $\hat{L}$ and $\hat{D}$ in exact arithmetic. Moreover $\hat{L}$ and $\hat{D}$ are small relative perturbations of L and D, and $\breve{L}$

[1] LAPACK is a library of Fortran 77 subroutines for solving the most common problems in numerical linear algebra.

and $\breve{D}$ are small relative perturbations of $\tilde{L}$ and $\tilde{D}$. See figure bellow. This property is called *mixed stability* in [9] but note that the perturbations are relative ones. And dqds enjoys high mixed relative stability even with element growth.

$$
\begin{array}{ccc}
L, U & \xrightarrow[\text{computed}]{\text{dqds}} & \tilde{L}, \tilde{U} \\
\text{change each } l_k \text{ by 1 ulp, } u_k \text{ by 3 ulps} \;\Big\downarrow & & \Big\uparrow\; \text{change each } \breve{l}_k,\ \breve{u}_k \text{ by 2 ulps} \\
\hat{L}, \hat{U} & \xrightarrow[\text{exact}]{\text{dqds}} & \breve{L}, \breve{U}
\end{array}
$$

Figure 3.1: Effects of roundoff for dqds

The diagram shows that, in the absence of division by zero, underflow or overflow, the diagram commutes and, for all k, $\hat{l}_k$ and $\hat{u}_k$ differ from l_k and u_k by 3 and 1 ulps [2], respectively, and $\tilde{l}_k$ and $\tilde{u}_k$ differ from $\breve{l}_k$ and $\breve{u}_k$, respectively, by 2 ulps, at most.

3.6 Relation of LR algorithm for J matrices to qds

The LR transform of J is J' and the LR transform of J' is the matrix J'' defined in two steps by

$$J' = L'U', \quad J'' = U'L'.$$

In fact, the first step of the LR algorithm

$$J = LU, \qquad J' = UL,$$

defines J' and the second step defines J'',

$$J' = L'U', \qquad J'' = U'L'.$$

[2] One ulp, "unit in last place", of the normalized floating point number $y = \pm\beta^e \times .d_1 d_2 \ldots d_t$ is $\mathrm{ulp}(y) = \beta^e \times .00\ldots1 = \beta^{e-t}$. It is the right unit for discussing relative errors since it avoids reference to the magnitude of the numbers involved.

Now qd applied to L and U yields L' and U' and so defines J'' implicitly. There is no need to form J' or J''.

When shifts are employed the situation is a little more complicated. It is necessary to look at two successive steps with shifts σ_1 and σ_2.

In shifted LR we have

$$\begin{aligned} J_1 - \sigma_1 I &= L_1 U_1 \\ J_2 &= U_1 L_1 + \sigma_1 I \\ J_2 - \sigma_2 I &= L_2 U_2 \\ J_3 &= U_2 L_2 + \sigma_2 I. \end{aligned}$$

In other words, the shifts are restored so that J_1, J_2, J_3 are similar. Note that

$$\begin{aligned} J_2 &= U_1 L_1 + \sigma_1 I \\ &= L_1^{-1} L_1 U_1 L_1 + \sigma_1 I \\ &= L_1^{-1} \left(J_1 - \sigma_1 I \right) L_1 + \sigma_1 I \\ &= L_1^{-1} J_1 L_1, \end{aligned}$$

and

$$J_3 = L_2^{-1} J_2 L_2.$$

However, if J_2 is not to be formed one cannot explicitly add σ_1 back to the diagonal. On the other hand,

$$J_2 - \sigma_2 I = U_1 L_1 - (\sigma_2 - \sigma_1) I = L_2 U_2.$$

In general, we have

$$\begin{aligned} J_{i+1} - \sigma_{i+1} I &= U_i L_i + \sigma_i I - \sigma_{i+1} I \\ &= U_i L_i - (\sigma_{i+1} - \sigma_i) I = L_{i+1} U_{i+1}. \end{aligned}$$

Thus, to find L_2 and U_2 from L_1 and U_1 it is only necessary to apply $\mathsf{qds}(\sigma_2 - \sigma_1)$. In other words, to get qds equivalent to LR with shifts $\{\sigma_i\}_{i=1}^{\infty}$ it is necessary to use the differences $(\sigma_i - \sigma_{i-1})$ with qds.

In LR the shifts should converge to an eigenvalue of the original J or J'. In qds the shifts should converge to 0 and $u_n \to 0$, $l_{n-1} \to 0$ too and all shifts must be accumulated. See diagram in Figure 3.2.

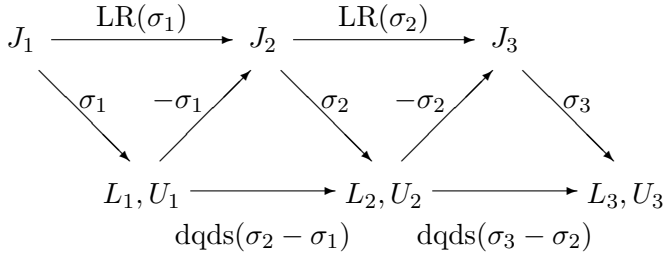


Figure 3.2: Relation of LR to qds

In the positive case both LR and qds are accurate and efficient.

In practice, the LR algorithm for J matrices avoids explicit calculation of the L's and U's and the transformation $J_i \to J_{i+1}$ is effected via a sequence of elementary similarity transformations. For comparison purposes, two implementations of LR, the explicit and the implicit shifted versions, can be found in [40, pp.469-471].

We end this section summarizing some advantages and disadvantages of the factored form LU.

Advantages of the factored form

1. L, U determines the entries of J to greater than working-precision accuracy because the addition and multiplication of l's and u's is implicit. Thus, for instance, the (i, i) entry of J is given by $l_{i-1} + u_i$ implicitly but $fl(l_{i-1} + u_i)$ explicitly.

2. The mapping $L, U \to J$ is naturally parallel; for example, if $\boldsymbol{l} = (l_i)$ and $\boldsymbol{u} = (u_i)$, then $\boldsymbol{l} * \boldsymbol{u}$ gives the off-diagonal entries of J. In contrast, the mapping $J \to L, U$, that is, Gaussian elimination, is intrinsically sequential.

3. Singularity of J is detectable by inspection when L and U are given, but only by calculation from J. So, LU reveals singularity, J does not.

4. First experiments with the measures of sensitivity presented in Chapter 7 show that LU defines the eigenvalues better than J does (usually).

5. Solution of $Jx = b$ takes half the time when L and U are available.

Disadvantages of the factored form

The mapping $J \mapsto L, U$ is not everywhere defined. Even when the factorization exists it can happen that $\|L\|$ and $\|U\|$ greatly exceed $\|J\|$. This is very bad for applying the LR algorithm but harmless when eigenvectors are to be calculated. So, some care is needed to consider the goal before stigmatizing a process as unstable. Moreover, in the eigenvalue context we are free to replace J by $J - \sigma I = LU$ for some suitable chosen shift σ that gives acceptable L and U. And there is a lot that can be done in improving existing shift strategies so that element growth can be monitored.

Chapter 4

Convergence results for LR

In this chapter we establish a new convergence result for the basic LR algorithm on a real unreduced tridiagonal matrix with a one-point spectrum - the Jordan form is only one big Jordan block. First we show the classical convergence results and summarize eigenvector properties of real unreduced unsymmetric tridiagonal matrices.

4.1 Classical results for the convergence of LR algorithm

The k^{th} iteration of the basic LR method is based on the LU decomposition $A_k = L_k U_k$ and on the multiplication of the factors L_k and U_k in reverse order to get the matrix A_{k+1}. The sequence of matrices $A =: A_1, A_2, A_3, \ldots$ is then generated by

LR algorithm

$A_1 = A$
for $i = 1, 2, \ldots$
  Factor $A_i = L_i R_i$ (LU factorization)
  $A_{i+1} = R_i L_i$
end

It is easy to see that A_{k+1} is similar to A_k,

$$A_{k+1} = R_k L_k = L_k^{-1}(L_k R_k)L_k, \tag{4.1}$$

and then, by an inductive argument, A_{k+1} is similar to A_1.

Lemma 4.1.1 *Let $A_1 := A$ and $\{A_i\}_{i=1}^{\infty}$ be the sequence of matrices generated by LR algorithm, $i = 1, 2, \ldots$. Then*

$$A_{i+1} = (L_1 L_2 \ldots L_i)^{-1} A_1 (L_1 L_2 \ldots L_i), \qquad i = 1, 2, \ldots \ldots$$

Proof. Use (4.1) repeatedly. □

Lemma 4.1.2 *Let matrices $\mathcal{L}_i$ and $\mathcal{U}_i$ be defined by*

$$\mathcal{L}_i \equiv L_1 L_2 \ldots L_i \qquad \text{and} \qquad \mathcal{U}_i \equiv R_i R_{i-1} \ldots R_1.$$

Then $\mathcal{L}_i \mathcal{U}_i$ is the LU *decomposition of A_1^i.*

Proof. From lemma 4.1.1 we have

$$L_1 L_2 \ldots L_{i-1} A_i = A_1 L_1 L_2 \ldots L_{i-1}. \tag{4.2}$$

Notice that matrices $\mathcal{L}_i$ and $\mathcal{U}_i$ are unit-lower triangular and upper-triangular, respectively. Consider the product $\mathcal{L}_i \mathcal{U}_i$. For $i = 2, 3, \ldots$, we have

$$\begin{aligned}
\mathcal{L}_i \mathcal{U}_i &= L_1 L_2 \ldots L_{i-1}(L_i R_i) R_{i-1} \ldots R_1 \\
&= L_1 L_2 \ldots L_{i-1} A_i R_{i-1} \ldots R_1 \\
&= A_1 L_1 L_2 \ldots L_{i-1} R_{i-1} \ldots R_1 \qquad \text{(from (4.2))} \\
&= A_1 \mathcal{L}_{i-1} \mathcal{U}_{i-1}.
\end{aligned}$$

Repeated application of this result shows that

$$\mathcal{L}_i \mathcal{U}_i = A_1^i, \tag{4.3}$$

so that $\mathcal{L}_i \mathcal{U}_i$ gives the LU decomposition of A_1^i. □

Thus, we have

$$A_{i+1} = {\mathcal{L}_i}^{-1} A_1 \mathcal{L}_i \qquad \text{and} \qquad \mathcal{L}_i \mathcal{U}_i = A_1^i,$$

that is, i steps of LR applied to A_1 are equivalent to a similarity given by a factorization of A_1^i.

4.1.1 Convergence of LR algorithm in the simplest case

We now restrict ourselves to the case when A_1 is nonsingular and has eigenvalues λ_i, $i = 1, \ldots, n$, of distinct modulus, so that it necessarily has linear elementary divisors. We may write

$$A_1 = X \operatorname{diag}(\lambda_1, \ldots, \lambda_n) X^{-1} =: XDY, \tag{4.4}$$

where the columns of X are the right eigenvectors of A_1 and the rows of Y are the row eigenvectors of A_1.

Suppose we order the eigenvalues λ_i, $i = 1, \ldots, n$, of A_1 so that they satisfy

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|. \tag{4.5}$$

Then, under certain restrictions, we have

$$L_i \to I \qquad \text{and} \qquad R_i \to A_i \to \begin{bmatrix} \lambda_1 & * & * & * \\ & \lambda_2 & * & * \\ & & \ddots & \vdots \\ & & & \lambda_n \end{bmatrix} \qquad \text{as} \qquad i \to \infty.$$

Theorem 4.1.1 *If $A = X \operatorname{diag}(\lambda_1, \ldots, \lambda_n) X^{-1}$ and*

1. *$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$,*

2. *the leading principal minors of X and X^{-1} are nonzero,*

then

$$\lim A_i = A_\infty$$

exists and is upper triangular with the i^{th} diagonal entry of A_∞ equal to λ_i.

Before writing the proof of this result, observe two simple properties of triangular decomposition. The triangular decomposition of a nonsingular matrix X, if it exists, is unique. Further, if

$$X = I + F$$

then, if $\|F\|$ is sufficiently small, the triangular decomposition of X exists, and if

$$X = I + F = LU$$

then

$$L, U \to I \qquad \text{as} \qquad \|F\| \to 0.$$

This can be verified if we think in terms of Gaussian elimination.

The proof of theorem 4.1.1 is rather technical (see Wilkinson [59] and [60, pp.487-492]) and it was first given by Rutishauser [49]. It is based on the fact that if matrix X of right eigenvectors of A admits triangular factorization

$$X = L_X U_X$$

then

$$\mathcal{L}_i \to L_X \qquad \text{and} \qquad A_i = \mathcal{L}_{i-1}^{-1} A_1 \mathcal{L}_{i-1} \to L_X^{-1} A_1 L_X = U_X X^{-1} A X U_X^{-1}$$
$$= U_X \operatorname{diag}(\lambda_1, \ldots, \lambda_n) U_X^{-1},$$

showing that the limiting A_i is upper-triangular with diagonal elements $\lambda_1, \ldots, \lambda_n$.

Proof. From (4.3) we have

$$\mathcal{L}_i \mathcal{U}_i = A_1^i = X D^i Y \tag{4.6}$$

and hence $\mathcal{L}_i \mathcal{U}_i$ is the LU decomposition of $X D^i Y$.

Since we are assuming that the leading principal minors of X and Y are non-zero, then X and Y have LU decompositions. We write

$$X = L_X U_X \qquad \text{and} \qquad Y = L_Y U_Y,$$

and attempt to construct the LU decomposition of XD^iY. We have

$$XD^iY = XD^iY = XD^iL_YU_Y = X(D^iL_YD^{-i})(D^iU_Y). \tag{4.7}$$

If we write

$$D^iL_YD^{-i} = I + F_i, \tag{4.8}$$

with $F_i = \left(f_{kj}^{(i)}\right)$ and $L_Y = (l_{kj})$, we have

$$f_{kj}^{(i)} = \begin{cases} l_{kj}\left(\frac{\lambda_k}{\lambda_j}\right)^i, & k > j \\ 0, & k \leq j \end{cases}. \tag{4.9}$$

From relations (4.5) and (4.9) we see that $F_i \to O$ as $i \to \infty$. Hence

$$\begin{aligned} XD^iY = X(I+F_i)D^iU_Y &= L_XU_X(I+F_i)D^iU_Y \\ &= L_X\left(I + U_XF_iU_X^{-1}\right)U_XD^iU_Y \\ &= L_X(I+G_i)U_XD^iU_Y \end{aligned}$$

where $G_i \to O$ as $i \to \infty$. Thus, for sufficiently large i, $I + G_i$ has an LU decomposition which we may write in the form

$$I + G_i = \left(I + L_G^{(i)}\right)\left(I + U_G^{(i)}\right)$$

where $L_G^{(i)}, U_G^{(i)} \to O$ as $i \to \infty$.

Finally from (4.6) we have

$$\mathcal{L}_i\mathcal{U}_i = L_X\left(I + L_G^{(i)}\right)\left(I + U_G^{(i)}\right)U_XD^iU_Y,$$

which, from the uniqueness of the LU decomposition, gives

$$\mathcal{L}_i = L_X\left(I + L_G^{(i)}\right) \to L_X \qquad \text{as} \qquad i \to \infty.$$

Hence, since $A_{i+1} = {\mathcal{L}_i}^{-1} A_1 \mathcal{L}_i$, we have

$$A_{i+1} = {\mathcal{L}_i}^{-1} X D X^{-1} \mathcal{L}_i \to L_X^{-1} L_X U_X D U_X^{-1} L_X^{-1} L_X = U_X D U_X^{-1}. \tag{4.10}$$

The matrix $U_X D U_X^{-1}$ is upper-triangular and has diagonal elements $\lambda_1, \ldots, \lambda_n$ in this order. □

According to [59, p.80], we should note that it was assumed that A_i has a triangular decomposition at all stages. This is not assured by the non-vanishing of the principal minors of X and Y and hence even when these conditions are satisfied, the LR method can break down. Such a failure corresponds to the non-existence of a decomposition of $I + G_i$. Since $I + G_i \to O$ this cannot happen at a late stage in the process.

If we remove the condition that Y has non-vanishing principal minors, a phenomenon called *disorder of latent roots* occurs - A_{i+1} tends to an upper triangular matrix having as its diagonal the eigenvalues but no longer in monotonic decreasing order. The eigenvalues are therefore disordered in the limiting matrix and this phenomenon happens to be unstable in practice.

The speed of convergence of the LR algorithm is determined essentially by the speed at which the elements of F_i tend to zero and hence on the quantities $\dfrac{\lambda_k}{\lambda_j}$, $k > j$. If $\left|\dfrac{\lambda_k}{\lambda_j}\right|$ is close to 1, that is, if the separation of the eigenvalues is poor, convergence may be slow. So, the speed of convergence of A_i to upper-triangular form depends, particularly, on the ratios $\left|\dfrac{\lambda_{k+1}}{\lambda_k}\right|$ which are the highest.

If we let $L_i = (l_{kj}^{(i)})$, from the relation $L_i = \mathcal{L}_{i-1}^{-1} \mathcal{L}_i$ it can be proved that

$$l_{kj}^{(i)} = \mathcal{O}\left(\frac{\lambda_k}{\lambda_j}\right)^i \qquad \text{as} \qquad i \to \infty \qquad (k > j).$$

So if we have $\lambda_n \approx 0$ then

$$l_{n,j}^{(i)} = \mathcal{O}\left(\frac{\lambda_n}{\lambda_j}\right)^i, \qquad j < n.$$

converges quickly to 0, that is, the last line of L_i converges quickly to $\boldsymbol{e}_n^T$. Thus $A_{i+1} = R_i L_i$ will have the last line converging quickly to $\lambda_n \boldsymbol{e}_n^T \approx \mathbf{0}$. This explains why it is more

efficient to apply LR algorithm to $A-\sigma I$ with σ close λ_n and how shifts of origin accelerate convergence.

Note that in establishing this result of convergence we made the assumption that all the eigenvalues are of different magnitude and in this case A can not be a real matrix with complex conjugate eigenvalues.

4.1.2 Eigenvalues of coincident absolute value

If $\left|\frac{\lambda_k}{\lambda_j}\right| = 1$ it appears at first sight that we no longer have convergence. But this is not the case. If the $|\lambda_i|$ are not distinct then, loosely speaking, A_i may be said to converge to block triangular form. See Parlett [41]. More precisely, suppose

$$|\lambda_1| = |\lambda_2| = \ldots = |\lambda_p| > |\lambda_{p+1}|.$$

Strictly speaking, A_∞ may not exist but, as $i \to \infty$, A_∞ becomes reduced. The elements in the first p columns may not converge but the characteristic polynomial of the leading principal submatrix of order p does converge to the monic polynomial with roots $\lambda_1, \ldots, \lambda_p$. The complementary submatrix has eigenvalues which converge to $\lambda_{p+1}, \ldots, \lambda_n$. If any of these eigenvalues have equal modulus then this submatrix will become reduced also. Thus, as $i \to \infty$, the submatrix blocks which become isolated along the diagonal correspond to groups of eigenvalues of equal modulus.

In the important case of real matrices with complex conjugate pairs, A_i may be expected to have along the diagonal, for large i, isolated 2×2 real submatrix which yield the eigenvalues very conveniently.

Theorem 4.1.2 *Let $A =: A_1$ be a diagonalizable real square matrix. Suppose that the* LU *factorization of A_i exists at every step $i = 1, 2, \ldots$ of* LR *algorithm. Then A_i tends to a block upper triangular form as $i \to \infty$. The diagonal blocks, say X_1^i, X_2^i, ..., X_p^i, need not converge but the eigenvalues of X_j^i, $j = 1, \ldots, p$, converge to the set of eigenvalues of j^{th} largest magnitude. In particular, if the eigenvalues have distinct magnitude, except for*

complex conjugate pairs, then the blocks X_j^i are either 1×1 for real eigenvalues, or 2×2, for a complex conjugate pair.

In [59] Wilkinson separates the proof of this result into two parts: equal eigenvalues having linear elementary divisors and unequal eigenvalues of equal modulus. That is, it is still assumed that the matrix A is diagonalizable.

For the case of non-linear divisors it is given in the last section of [59] a simple counter-example that shows immediately that the LR algorithm does not necessarily give convergence to an upper-triangular matrix. The matrix

$$A = \begin{bmatrix} a & 0 & 0 \\ 1 & a & 0 \\ 0 & 1 & a \end{bmatrix}, \quad a \in \mathbb{R},$$

is LR invariant, that is, $A_i = A$ for all i.

4.2 Eigenvector properties of an unreduced tridiagonal

Eigenvector matrices of real unreduced symmetric tridiagonal matrices have several attractive properties and have been studied widely in the literature. See Parlett [38, Chapter 7]. The eigenvalues are real and distinct and key properties are

- the first and last entries cannot vanish; there are very elegant formulae for the squares of entries of normalized eigenvectors.
- When the off-diagonal entries are all of the same sign, the eigenvector for the rightmost (largest) eigenvalue has no sign changes, for the second largest eigenvalue, the eigenvector has one sign change, and so on. The eigenvector for the leftmost (smallest) eigenvalue has the maximal number of sign changes, namely $n-1$ for a $n \times n$ matrix. See Gantmakher and Krein [19] and Fiedler [17].

Our interest is in the real unsymmetric case and we expect the matrix spectrum to have a mixture of real and complex eigenvalues. Of the properties above only the first extends to

the case of our interest. The proof is identical to the symmetric case and will be omitted. A new difficulty in our case is that the eigenvalues need not be simple, so the Jordan form may not be diagonal. In such cases the eigenvector matrix must be filled out with the so-called generalized eigenvectors with the property that

$$(C - \lambda I)^j v = 0, \;\; (C - \lambda I)^{j-1} v \neq 0.$$

We say v is an eigenvector of grade j (see page 10).

In what follows we shall present some properties of eigenvector matrices that are sufficient to guarantee convergence of the basic LR algorithm without invoking the extra hypotheses needed by Rutishauser and Wilkinson for the general case. To the best of our knowledge these results are new.

Our convergence theory for LR algorithm requires that certain matrices X permit triangular factorization $X = LDU$ or $X = L(DU)$. This property plays a prominent role in Linear Systems Theory.

We will say that X is *strongly* (or *completely*) *regular* with the meaning that X and all its leading principal submatrices are invertible. We shall use the terms "strongly regular" and "permits LU" interchangeably. To be precise, we note that a singular matrix may permit triangular factorization but in our work all the matrices of interest will be invertible.

We proceed from the easier cases to the more difficult in stages.

Most of our results extend directly to complex unreduced tridiagonal matrices but we focus on real matrices for simplicity and because it is the most frequent case in applications.

Consider an unreduced real tridiagonal matrix $C = \text{tridiag}(\boldsymbol{b}, \boldsymbol{a}, \boldsymbol{c})$,

$$C = \begin{bmatrix} a_1 & c_1 & & & \\ b_1 & a_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2} & a_{n-1} & c_{n-1} \\ & & & b_{n-1} & a_n \end{bmatrix} \in \mathbb{R}^{n \times n}$$

with $b_i c_i \neq 0$, $i = 1, \ldots, n-1$.

Define monic polynomials $p_0, p_1, \ldots, p_n$ by

$$p_0(\tau) = 1, \;\; p_j(\tau) := \det(\tau I_j - C_j), \;\; j = 1, \ldots, n,$$

where I_j represents the $j \times j$ identity matrix and C_j the j^{th} leading principal submatrix of C.

4.2.1 All eigenvalues distinct

Suppose all eigenvalues of C are distinct and let the spectrum be

$$\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}.$$

The following matrix plays a key role in our results,

$$P = P_C = [p_{i-1}(\lambda_j)]_{i,j=1}^{n}.$$

The notation means that the (i, j) element of P is equal to $p_{i-1}(\lambda_j)$. So the j^{th} column of P is given by the column vector

$$\boldsymbol{p}(\lambda_j) := \begin{bmatrix} p_0(\lambda_j) & p_1(\lambda_j) & \ldots & p_{n-1}(\lambda_j) \end{bmatrix}^T, \qquad j = 1, \ldots, n,$$

that is,

$$P = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ p_1(\lambda_1) & p_1(\lambda_2) & \ldots & p_1(\lambda_n) \\ \vdots & \vdots & & \vdots \\ p_{n-1}(\lambda_1) & p_{n-1}(\lambda_2) & \ldots & p_{n-1}(\lambda_n) \end{bmatrix}. \tag{4.11}$$

P and P^T are called *polynomial Vandermond* matrices. The standard Vandermond matrix V is defined by

$$V = V_\Lambda = \left[\lambda_i^{j-1}\right]_{i,j=1}^{n},$$

that is,

$$V = \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \ldots & \lambda_1^{n-1} \\ 1 & \lambda_2 & \lambda_2^2 & \ldots & \lambda_2^{n-1} \\ \vdots & \vdots & & \vdots & \\ 1 & \lambda_n & \lambda_n^2 & \ldots & \lambda_n^{n-1} \end{bmatrix}. \tag{4.12}$$

The valuable property of V is that

$$\det(V) = \prod_{i>j}(\lambda_i - \lambda_j) \tag{4.13}$$

where the product extends over all pairs (i, j) with $n \geq i > j \geq 1$; $n(n-1)/2$ terms in all.

When the λ_i are distinct then V is strongly regular because each leading principal submatrix of V is also a Vandermond built from a subset of the eigenvalues.

Lemma 4.2.1 *If all the eigenvalues λ_i, $i = 1, \ldots, n$, are distinct then V is strongly regular.*

Moreover ΠV is strongly regular for any permutation matrix Π because no particular ordering of the eigenvalues was specified in the definition.

Lemma 4.2.2 $\det(P) = \det\left(V^T\right) = \det(V)$.

Proof. P may be reduced to V^T by elementary row operations that leave the determinant unchanged. For example, if $p_1(\tau) = \tau - k$ then add k times row 1 to row 2 of P and the second row becomes $\begin{bmatrix}\lambda_1 & \ldots & \lambda_n\end{bmatrix} = \boldsymbol{e}_2^T V^T$. And so on. $\square$

With the aid of $P = \begin{bmatrix}\boldsymbol{p}(\lambda_1) & \boldsymbol{p}(\lambda_2) & \cdots & \boldsymbol{p}(\lambda_n)\end{bmatrix}$ we can find simple forms for the column and row eigenvectors of C.

We will denote

$$D_b := \operatorname{diag}(1, b_1, b_1b_2, b_1b_2b_3, \ldots, b_1b_2\cdots b_{n-1}),$$
$$D_c := \operatorname{diag}(1, c_1, c_1c_2, c_1c_2c_3, \ldots, c_1c_2\cdots c_{n-1}).$$

Lemma 4.2.3 *With the notation given above*

$$C\left(D_c^{-1}P\right) = \left(D_c^{-1}P\right)\Lambda, \qquad \left(P^T D_b^{-1}\right)C = \Lambda\left(P^T D_b^{-1}\right).$$

Proof. The result is a reformulation of the celebrated three term recurrence (3TR) associated with C:

$$p_{j+1}(\tau) = (\tau - a_{j+1})p_j(\tau) - b_jc_jp_{j-1}(\tau), \qquad j = 1, 2, \ldots, n-1 \tag{4.14}$$
$$p_1(\tau) = (\tau - a_1) = (\tau - a_1)p_0(\tau), \text{ since } \qquad p_0(\tau) = 1.$$

If we define $p_{-1}(\tau) = 0$ then we may use the 3TR for $j = 0$.

Rewrite the 3TR in the form

$$b_j c_j p_{j-1}(\tau) + (a_{j+1} - \tau)p_j(\tau) + p_{j+1}(\tau) = 0, \qquad j = 0, \ldots, n-1.$$

The key step is to divide through by $c_1 \cdots c_j$ $(\neq 0)$ and rearrange coefficients,

$$b_j \frac{p_{j-1}(\tau)}{c_0 c_1 \cdots c_{j-1}} + (a_{j+1} - \tau)\frac{p_j(\tau)}{c_0 c_1 \cdots c_j} + c_{j+1}\frac{p_{j+1}(\tau)}{c_1 \cdots c_{j+1}} = 0, \qquad j = 0, 1, \ldots, n-1,$$

where $b_0 = 0$, $b_n = c_n = 1$ and $c_0 = 1$. This set of equations may be written as

$$(C - \tau I)D_c^{-1}\boldsymbol{p}(\tau) = -\boldsymbol{e}_n \frac{p_n(\tau)}{c_1 \cdots c_{n-1}c_n}.$$

Thus, since for each $\tau = \lambda_k$ we have

$$p_n(\lambda_k) = 0,$$

it follows that

$$(C - \lambda_k I)D_c^{-1}\boldsymbol{p}(\lambda_k) = -\boldsymbol{e}_n \frac{p_n(\lambda_k)}{c_1 \cdots c_{n-1}c_n} = \mathbf{0} \qquad , k = 1, \ldots, n. \tag{4.15}$$

Thus, in matrix terms,

$$CD_c^{-1}P = D_c^{-1}P\Lambda.$$

Note that $C^T = D_c D_b^{-1} C D_c^{-1} D_b$ and then

$$C^T D_b^{-1} P = D_c D_b^{-1}\left(CD_c^{-1}P\right) = D_c D_b^{-1}\left(D_c^{-1}P\Lambda\right) = D_b^{-1}P\Lambda. \ \square$$

These row and column eigenvectors are not scaled properly to be inverses of each other. Since the row eigenvectors for λ_j annihilates all the column eigenvectors for different eigenvalues we may define a special diagonal matrix $\Delta = \Delta_C$ by

$$\left(P^T D_b^{-1}\right)\left(D_c^{-1}P\right) = \Delta := \mathrm{diag}(\delta_1, \ldots, \delta_n),$$
$$\delta_j = \boldsymbol{p}(\lambda_j)^T \left(D_b D_c\right)^{-1} \boldsymbol{p}(\lambda_j) \neq 0.$$

Note that, if $\delta_j = 0$ then the row eigenvector for λ_j annihilates all the column eigenvectors and would be $\mathbf{0}^T$ which contradicts the definition of an eigenvector.

The matrix Δ^{-1} may be attached to either $D_c^{-1}P$ or $D_b^{-1}P$ or shared between them. In general, Δ will be indefinite. We will have

$$\left(P^T D_b^{-1}\right)\left(D_c^{-1}P\Delta^{-1}\right) = \left(D_c^{-1}P\Delta^{-1}\right)\left(P^T D_b^{-1}\right) = I$$

and

$$\left(\Delta^{-1}P^T D_b^{-1}\right)\left(D_c^{-1}P\right) = \left(D_c^{-1}P\right)\left(\Delta^{-1}P^T D_b^{-1}\right) = I.$$

Theorem 4.2.1 *The spectral decomposition of C with simple eigenvalues may be written*

$$C = \left(D_c^{-1}P\right)\Lambda\left(\Delta^{-1}P^T D_b^{-1}\right) = \left(D_c^{-1}P\Delta^{-1}\right)\Lambda\left(P^T D_b^{-1}\right).$$

Theorem 4.2.2 *When C has simple eigenvalues then both column and row eigenvector matrices $D_c^{-1}P\Delta^{-1}$ and $P^T D_b^{-1}$, respectively, are strongly regular.*

The proof needs only the following easy results.

Lemma 4.2.4 *For any invertible diagonal matrices D' and D'', M is strongly regular if, and only if, $D'MD''$ is strongly regular.*

Proof. Let $(D'MD'')_j$ be the j^{th} leading principal submatrix of $D'MD''$. We have

$$\det\left((D'MD'')_j\right) = \det(D'_j)\det(M_j)\det(D''_j) \neq 0, \qquad j = 1, \ldots, n,$$

since all the determinants are nonzero. □

Lemma 4.2.5 *When all the eigenvalues λ_i, $i = 1, \ldots, n$, are distinct, both P and P^T are strongly regular.*

Proof. Use lemmas 4.2.1 and 4.2.2 □

Next we consider the opposite extreme, a maximal Jordan block of C.

4.2.2 The one-point spectrum

Suppose now that C's spectrum consists of a single nonzero point λ and such that its Jordan form is

$$J = \lambda I + N$$

where N is the nilpotent matrix

$$N = \begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ & & & & 0 \end{bmatrix}.$$

From the previous section we know that the relation (4.15) is verified only for λ,

$$(C - \lambda I) D_c^{-1} \boldsymbol{p}(\lambda) = \mathbf{0}.$$

So, the only column eigenvector of C is $D_c^{-1}\boldsymbol{p}(\lambda)$ and its single row eigenvector is $\boldsymbol{p}(\lambda)^T D_b^{-1}$.

The most elegant way to find eigenvectors of higher grade is to differentiate the 3TR as many times as is necessary. Thus, from (4.14),

$$p'_{j+1}(\tau) = (\tau - a_{j+1}) p'_j(\tau) + p_j(\tau) - b_j c_j p'_{j-1}(\tau), \qquad j = 1, 2, \ldots, n-1$$

and rearranging the terms we have

$$b_j c_j p'_{j-1}(\tau) + (a_{j+1} - \tau) p'_j(\tau) + p'_{j+1}(\tau) = p_j(\tau). \tag{4.16}$$

Then, dividing by $c_1 \cdots c_j$,

$$b_j \frac{p'_{j-1}(\tau)}{c_0 c_1 \cdots c_{j-1}} + (a_{j+1} - \tau) \frac{p'_j(\tau)}{c_0 c_1 \cdots c_j} + c_{j+1} \frac{p'_{j+1}(\tau)}{c_1 \cdots c_{j+1}} = \frac{p_j(\tau)}{c_0 c_1 \cdots c_j}, \qquad j = 0, 1, \ldots, n-1,$$

where we define $p'_{-1}(\tau) = 0$, $b_n = c_n = 1$ and $c_0 = 1$. In matrix terms

$$(C - \tau I) D_c^{-1} \boldsymbol{p}'(\tau) = D_c^{-1} \big(\boldsymbol{p}(\tau) - \boldsymbol{e}_n p'_n(\tau) \big).$$

Hence, when $\tau = \lambda$,

$$(C - \lambda I)D_c^{-1}\boldsymbol{p}'(\lambda) = D_c^{-1}\boldsymbol{p}(\lambda), \qquad \text{since } p_n'(\lambda) = 0,$$

and

$$(C - \lambda I)^2 D_c^{-1}\boldsymbol{p}'(\lambda) = (C - \lambda I)D_c^{-1}\boldsymbol{p}(\lambda) = \mathbf{0},$$

since $p_n(\lambda) = (\tau - \lambda)^n$ and λ is a multiple zero of p_n.

To obtain the next vector differentiate (4.16) again,

$$b_j c_j p_{j-1}''(\tau) + (a_{j+1} - \tau)p_j''(\tau) + 1.p_{j+1}''(\tau) = 2p_j'(\tau).$$

In order to keep the superdiagonal entries in the Jordan form at the value 1 we divide through by 2 to obtain

$$(C - \tau I)\frac{1}{2}D_c^{-1}\boldsymbol{p}''(\tau) = D_c^{-1}\left(\boldsymbol{p}'(\tau) - \boldsymbol{e}_n\frac{1}{2}p_n''(\tau)\right).$$

Putting $\tau = \lambda$ and since $p''(\lambda) = 0$,

$$(C - \lambda I)\frac{1}{2}D_c^{-1}\boldsymbol{p}''(\lambda) = D_c^{-1}\boldsymbol{p}'(\lambda),$$

and

$$(C - \lambda I)^3\frac{1}{2}D_c^{-1}\boldsymbol{p}''(\lambda) = (C - \lambda I)^2 D_c^{-1}\boldsymbol{p}'(\lambda) = \mathbf{0}.$$

It may be verified that the appropriate definition of P in the confluent case is the unit lower triangular matrix

$$P_\lambda = P = \left[\boldsymbol{p}(\lambda) \quad \boldsymbol{p}'(\lambda) \quad \tfrac{1}{2!}\boldsymbol{p}''(\lambda) \quad \dots \quad \tfrac{1}{(n-1)!}\boldsymbol{p}^{(n-1)}(\lambda)\right]$$

and so

$$(C - \lambda I)D_c^{-1}P = D_c^{-1}PN. \tag{4.17}$$

The next lemma summarizes this result.

Lemma 4.2.6 *If C has one-point spectrum λ and D_c and P are as defined above, then*

$$C\left(D_c^{-1}P\right) = \left(D_c^{-1}P\right)(N + \lambda I) = \left(D_c^{-1}P\right)J.$$

So, the matrix of generalized right eigenvectors of C is $D_c^{-1}P$. To find the row eigenvectors for C we first use

$$C^T = D_c D_b^{-1} C D_c^{-1} D_b$$

to find, analogously to (4.17),

$$\left(C^T - \lambda I\right) D_b^{-1} P = D_b^{-1} P N.$$

Now, transposing this equation we get

$$P^T D_b^{-1} C = \left(\lambda I + N^T\right) P^T D_b^{-1} = \mathit{I}(\lambda I + N)\,\mathit{I} P^T D_b^{-1}, \tag{4.18}$$

where

$$\mathit{I} = \begin{bmatrix} & & & & 1 \\ & & & 1 & \\ & & \iddots & & \\ & 1 & & & \\ 1 & & & & \end{bmatrix}$$

is the *reversal* or *anti-diagonal* matrix (all entries (i,j) are zero except when $i+j = n+1$). We used the fact that $\mathit{I}\,\mathit{I} = I$ and $N^T = \mathit{I} N \mathit{I}$. Thus, pre-multiplying (4.18) by I, we find that

$$\left(\mathit{I} P^T D_b^{-1}\right) C = (\lambda I + N)\left(\mathit{I} P^T D_b^{-1}\right).$$

So, $\mathit{I} P^T D_b^{-1}$ is the matrix of generalized row eigenvectors of C.

Summarizing,

Lemma 4.2.7 *If C has one-point spectrum λ and D_b and P are as defined above, then*

$$\left(\mathit{I} P^T D_b^{-1}\right) C = J\left(\mathit{I} P^T D_b^{-1}\right).$$

Recall that $D_c^{-1}P$ is lower triangular and $P^T D_b^{-1}$ is upper triangular. Nevertheless, it is not true that the product $\left(\not{I} P^T D_b^{-1}\right)\left(D_c^{-1}P\right)$ is diagonal, as was the case for simple eigenvalues. The reason is subtle: for a Jordan block, the eigenvectors of grade higher than 1 are not uniquely defined. The phrase "the Jordan basis" that can be found in some text books is incorrect; it is not unique.

Consider the equation above in (4.17),

$$CD_c^{-1}P = D_c^{-1}P(\lambda I + N).$$

Post-multiply by any invertible matrix $\varphi(N)$, φ a polynomial, that comutes with $\lambda I + N$ to find that

$$CD_c^{-1}P\varphi(N) = D_c^{-1}P(\lambda I + N)\varphi(N) = D_c^{-1}P\varphi(N)(\lambda I + N).$$

Thus $D_c^{-1}P$ is only unique up to post-multiplication by any invertible polynomial $\varphi(N)$. And there is no loss in normalizing φ to satisfy $\varphi(O) = I$.

Chosen

$$\left(\not{I} P^T D_b^{-1}\right)\left(D_c^{-1}P\right) = \varphi(N),$$

we have proved

Theorem 4.2.3 *If C has one-point spectrum λ and P, D_b, D_c are as defined above then*

$$C = D_c^{-1}P\varphi(N)^{-1}(\lambda I + N)\,\not{I} P^T D_b^{-1},$$

for some polynomial φ with $\varphi(O) = I$.

The example that follows exhibits this feature.

Before, recall that a square matrix A is *Toeplitz* when the entries of A are constant down the diagonals parallel to the main diagonal and is *Hankel* when the entries of A are constant along the diagonals perpendicular to the main diagonal.

Z. S. Liu [31] devised an algorithm to obtain one-point spectrum unreduced tridiagonal matrices of arbitrary dimension $n \times n$. These matrices, that we will call *Liu's matrices*, have only one eigenvalue, zero with algebraic multiplicity n and geometric multiplicity 1. The Jordan form consists of one big Jordan block. We will represent Liu's matrices as

$$Liu_n = \text{tridiag}(\mathbf{1}^n, \boldsymbol{\alpha}^n, \boldsymbol{\gamma}^n)$$

where $\mathbf{1}^n$ always stands for a vector of 1's of length $n-1$. For $n = 6$ we have $\boldsymbol{\alpha}^6 = \begin{bmatrix} 0 & 0 & -1 & 1 & 0 & 0 \end{bmatrix}$ and $\boldsymbol{\gamma}^6 = \begin{bmatrix} -1 & 1 & -1 & 1 & -1 \end{bmatrix}$.

Example 4.2.1

Consider the transpose of Liu_6 matrix

$$C = \begin{bmatrix} 0 & 1 & & & & \\ -1 & 0 & 1 & & & \\ & 1 & -1 & 1 & & \\ & & -1 & 1 & 1 & \\ & & & 1 & 0 & 1 \\ & & & & -1 & 0 \end{bmatrix}.$$

We have

$$\begin{aligned} p_0(\tau) &= 1, \\ p_1(\tau) &= \tau, \\ p_2(\tau) &= \tau^2 + 1, \\ p_3(\lambda) &= (\tau+1)p_2(\tau) - p_1(\tau) = \tau^3 + \tau^2 + 1, \\ p_4(\tau) &= (\tau-1)p_3(\tau) + p_2(\tau) = \tau^4 + \tau, \\ p_5(\tau) &= \tau p_4(\tau) - p_3(\tau) = \tau^5 - \tau^3 - 1, \\ p_6(\tau) &= \tau p_5(\tau) + p_4(\tau) = \tau^6. \end{aligned}$$

Then

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}, \qquad D_b = \operatorname{diag}(1,-1,-1,1,1,-1), \qquad D_c = I.$$

Now, we have

$$P^T D_b^{-1} P = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & -1 \end{bmatrix} = \mathit{I}\mathcal{U}^{-1},$$

defining $\mathcal{U}$.

Thus,

$$U = \left(\mathit{I}P^T D_b^{-1} P\right)^{-1} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

that is, $\mathcal{U} = I - N^3 + N^5$ is a polynomial in N and thus commutes with $\lambda I + N$ which is also a polynomial in N.

Thus $\mathit{I}P^T D_b^{-1} P = \mathcal{U}^{-1} = I + N^3 - N^5 = \varphi(N)$ is unit upper triangular and Toeplitz. So,

$$C = P\mathcal{U}\,(0I + N)\,\mathit{I}P^T D_b^{-1}, \qquad I = \left(\mathit{I}P^T D_b^{-1}\right)(P\mathcal{U}).$$

Note that $P\mathcal{U}$ is in LU form and $P^T D_b^{-1}$ is upper triangular. It will be shown that $C + \sigma I$, $\sigma \neq 0$, can admit the basic LR algorithm with no breakdowns and will converge to

$$\mathcal{U}(\sigma I + N)\mathcal{U}^{-1} = \sigma I + N. \ \diamond$$

4.2.3 The general case

In general, unreduced C will have some simple eigenvalues and some multiple ones. The unreduced property implies that C is nonderogatory, meaning that there is only one Jordan block, and thus one eigenvector, for each eigenvalue.

It follows directly from the previous sections that the matrix P has a column $\boldsymbol{p}$ for each simple eigenvalue λ and a block of columns $\left[\boldsymbol{p}(\lambda) \quad \boldsymbol{p}'(\lambda) \quad \frac{1}{2!}\boldsymbol{p}''(\lambda) \quad \dots \quad \frac{1}{m!}\boldsymbol{p}^{(m)}(\lambda)\right]$ if λ has multiplicity $m+1$. The only constraint on ordering of columns is that each block must be treated as a whole.

As shown in the section on one-point spectrum matrices, the order of the row eigenvectors must be reversed within each block.

We state without proof.

Theorem 4.2.4 *Let J be the (upper) Jordan form of unreduced matrix C. Then the spectral decomposition may be written*

$$C = D_c^{-1} P \mathcal{U} J \mathcal{W} P^T D_b^{-1}$$

where $\mathcal{U}$ is a unit upper triangular matrix that commutes with J and $\mathcal{W}$ is a symmetric permutation matrix that is a direct sum of reversal matrices I conforming to the block structure of J. $\mathcal{W}$ also commutes with J. P, and therefore P^T, are strongly regular.

4.3 Convergence of basic LR algorithm on an unreduced tridiagonal

We will show that the assumptions required by Wilkinson in the general case to ensure the convergence of LR algorithm are no longer needed on an unreduced tridiagonal matrix.

First we recall the essencial facts from the beginning of this chapter. Set $C_1 = C$ and for $k = 1, 2, \ldots$ define

$$\begin{aligned} C_k &= \mathrm{L}_k R_k, \qquad \mathrm{L}_k \text{ being unit lower triangular} \\ C_{k+1} &= R_k \mathrm{L}_k. \end{aligned}$$

The LU factorization of C^k is

$$\mathcal{L}_k \mathcal{U}_k = C^k$$

with $\mathcal{L}_k = \mathrm{L}_1 \mathrm{L}_2 \ldots \mathrm{L}_k$ and $\mathcal{U}_k = R_k R_{i-1} \ldots R_1$. And then

$$C_{k+1} = {\mathcal{L}_k}^{-1} C \mathcal{L}_k.$$

The L factor of a matrix M will be denoted by $\mathcal{L}(M)$, provided that M is strongly regular. In this new notation we will write

$$C_{k+1} = \mathcal{L}(C^k)^{-1} C \mathcal{L}(C^k). \tag{4.19}$$

4.3.1 Eigenvalues of distinct absolute value

The result in this section is not entirely new but helps to understand the new case.

Let C be a nonsingular unreduced tridiagonal matrix. Without loss of generality, we may write

$$\begin{aligned} C &= X \Lambda X^{-1} \\ \Lambda &= \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n), \qquad |\lambda_i| > |\lambda_{i+1}|, \qquad \text{for all } i. \end{aligned}$$

With the notation of previous section,

$$X = D_c^{-1} P \Delta^{-1}, \qquad X^{-1} = P^T D_b^{-1}$$

where $P = P_\Lambda$ is the polynomial Vandermond matrix given in (4.11). Since P is strongly regular (and also X and X^{-1} by theorem 4.2.2) we may write

$$P = LDU,$$

where L is unit lower triangular, D is diagonal and U is unit upper triangular. Then we can manipulate C^k into LU form as follows

$$\begin{aligned} C^k &= X\Lambda^k X^{-1} \\ &= D_c^{-1} P \Delta^{-1} \Lambda^k P^T D_b^{-1} \\ &= D_c^{-1} L D U \Delta^{-1} \Lambda^k U^T D L^T D_b^{-1} \\ &= D_c^{-1} L D U \Delta^{-1} (\Lambda^k U^T \Lambda^{-k}) \Lambda^k D L^T D_b^{-1} \\ &= (D_c^{-1} L D_c)(D_c^{-1} D U \Delta^{-1})(I + E_k)(\Lambda^k D L^T D_b^{-1}) \\ &= (D_c^{-1} L D_c)(I + F_k)(D_c^{-1} D U \Delta^{-1})(\Lambda^k D L^T D_b^{-1}) \end{aligned} \tag{4.20}$$

where

$$F_k = \left(D_c^{-1} D U \Delta^{-1}\right) E_k \left(D_c^{-1} D U \Delta^{-1}\right)^{-1}.$$

Notice that

$$\begin{aligned} X &= \left(D_c^{-1} L D_c\right)\left(D_c^{-1} D U \Delta^{-1}\right) =: L_X U_X \\ Y &= U^T \left(D L^T D_b^{-1}\right) =: L_Y U_Y \end{aligned} \tag{4.21}$$

are the LU factorizations of X and Y, respectively.

Also, we had written

$$\Lambda^k U^T \Lambda^{-k} = I + E_k.$$

So, E_k is strictly lower triangular and, if we let $U = (u_{ij})$ and $E_k = \left(e_{ij}^{(k)}\right)$, we have

$$e_{ij}^{(k)} = \begin{cases} u_{ji}\left(\frac{\lambda_i}{\lambda_j}\right)^k, & i > j \\ 0, & i \le j \end{cases}. \tag{4.22}$$

Thus the $(j+m, j)$ entry of E_k is $u_{j,j+m}\left(\frac{\lambda_{j+m}}{\lambda_j}\right)^k$ and tends to zero as k tends to infinity, since $|\lambda_{j+m}| < |\lambda_j|$. Hence

$$\|F_k\| \le \operatorname{cond}\left(D_c^{-1} D U \Delta^{-1}\right) \|E_k\| \to 0 \qquad \text{as} \qquad k \to \infty.$$

It seems likely that $I + F_k$ is strongly regular for all k but it certainly holds for large enough k, say

$$I + F_k = L_k U_k \qquad \text{with} \qquad L_k, U_k \to I \qquad \text{as} \qquad k \to \infty.$$

Thus, for large enough k, from (4.20),

$$C^k = \left(D_c^{-1} L D_c\right) \left(L_k U_k\right) \left(D_c^{-1} D U \Delta^{-1}\right) \left(\Lambda^k D L^T D_b^{-1}\right),$$

and then

$$\mathcal{L}(C^k) = D_c^{-1} L D_c L_k$$

since $U_k \left(D_c^{-1} D U \Delta^{-1}\right) \left(\Lambda^k D L^T D_b^{-1}\right)$ is upper triangular. Thus

$$\mathcal{L}(C^k) \to D_c^{-1} L D_c = L_X \qquad \text{as} \qquad k \to \infty.$$

Finally, from (4.19),

$$C_{k+1} = \mathcal{L}(C^k)^{-1} C \mathcal{L}(C^k),$$

and thus

$$\begin{aligned} C_{k+1} &\to \left(D_c^{-1} L D_c\right)^{-1} X \Lambda X^{-1} \left(D_c^{-1} L D_c\right) \\ &= L_X^{-1} L_X U_X \Lambda U_X^{-1} L_X^{-1} L_X \\ &= U_X \Lambda U_X^{-1} \qquad \text{as} \qquad k \to \infty, \end{aligned}$$

showing that C_{k+1} converges to an upper triangular matrix with diagonal elements equal to $\lambda_1, \ldots, \lambda_n$.

Recalling that the tridiagonal form is preserved by the LR algorithm, we have proved

Theorem 4.3.1 *Let C be a nonsingular unreduced tridiagonal matrix with distinct eigenvalues λ_i, $i = 1, \ldots, n$. Given the notation above,*

$$\lim C_k = C_\infty$$

exists and is upper bidiagonal with the i^{th} diagonal entry of C_∞ equal to λ_i.

We should note that we are assuming that C_k permits triangular decomposition at all stages of the LR algorithm.

4.3.2 One-point spectrum

Recall that the Vandermond matrix P for this case is unit lower triangular,

$$P = \left[\boldsymbol{p}(\lambda) \quad \boldsymbol{p}'(\lambda) \quad \tfrac{1}{2!}\boldsymbol{p}''(\lambda) \quad \dots \quad \tfrac{1}{(n-1)!}\boldsymbol{p}^{(n-1)}(\lambda)\right].$$

From theorem 4.2.3,

$$C = X(\lambda I + N)X^{-1} \tag{4.23}$$

with

$$X = D_c^{-1} P \varphi(N)^{-1} \qquad \text{and} \qquad X^{-1} = \mathcal{I} P^T D_b^{-1}.$$

Then

$$C^k = D_c^{-1} P \varphi(N)^{-1} (\lambda I + N)^k \mathcal{I} P^T D_b^{-1}.$$

Note that P^T is unit upper triangular. Next we invoke the following lemma.

Lemma 4.3.1 *For all $k \geq n$, $(\lambda I + N)^k \mathcal{I}$ for $\lambda \neq 0$ is strongly regular and thus admits triangular factorization, say*

$$(\lambda I + N)^k \mathcal{I} = L_k U_k = L_k D_k L_k^T,$$

and, as $k \to \infty$, $L_k = I + E_k$, $E_k \to O$. The rate of convergence is low, $\mathcal{O}(1/k)$.

The proof of this lemma will be given later.

Now, we can factor C^k, $k \geq n$:

$$\begin{aligned} C^k &= D_c^{-1} P D_c D_c^{-1} \varphi(N)^{-1} (I + E_k) U_k P^T D_b^{-1} \\ &= D_c^{-1} P D_c (I + F_k) D_c^{-1} \varphi(N)^{-1} U_k P^T D_b^{-1} \end{aligned}$$

with

$$F_k = (\varphi(N) D_c)^{-1} E_k (\varphi(N) D_c) \to O \qquad \text{as} \qquad k \to \infty,$$

since

$$\|F_k\| \leq \operatorname{cond}(\varphi(N) D_c) \|E_k\| \to 0 \qquad \text{as} \qquad k \to \infty.$$

Thus,

$$\mathcal{L}(C^k) = D_c^{-1} P D_c \mathcal{L}(I + F_k) \to D_c^{-1} P D_c \qquad \text{as} \qquad k \to \infty,$$

since $D_c^{-1}\varphi(N)^{-1}U_k P^T D_b^{-1}$ is upper triangular ($\varphi(N)^{-1}$ is upper triangular and Toeplitz).

Finally, notice that the LU factorization of X is

$$X = \left(D_c^{-1} P D_c\right)\left(D_c^{-1}\varphi(N)^{-1}\right) := L_X U_X$$

and then

$$\begin{aligned}
C_{k+1} &= \mathcal{L}(C^k)^{-1} C \mathcal{L}(C^k) \\
&= \mathcal{L}(C^k)^{-1} X(\lambda I + N) X^{-1} \mathcal{L}(C^k) \qquad \text{(by 4.23)} \\
&\to \left(D_c^{-1} P D_c\right)^{-1} X(\lambda I + N) X^{-1} \left(D_c^{-1} P D_c\right) \\
&= L_X^{-1} L_X U_X (\lambda I + N) U_X^{-1} L_X^{-1} L_X \\
&= U_X (\lambda I + N) U_X^{-1} \\
&= D_c^{-1}\varphi(N)^{-1}(\lambda I + N)\varphi(N) D_c \\
&= D_c^{-1}(\lambda I + N) D_c,
\end{aligned}$$

since $\varphi(N)$ commutes with $(\lambda I + N)$. Notice that $D_c^{-1}(\lambda I + N)D_c$ is upper bidiagonal with diagonal entries equal to λ.

We have just proved

Theorem 4.3.2 *Let C be a nonsingular unreduced tridiagonal matrix that permits triangular factorization and has a one-point spectrum λ. Given the notation above, the basic* LR *algorithm applied to C produces a sequence of matrices C_k that converges (in exact arithmetic) to*

$$D_c^{-1}(\lambda I + N) D_c$$

with D_c defined above lemma 4.2.3 (page 91).

Proof of lemma 4.3.1:

The proof rests on the form of powers of Jordan blocks. These are Toeplitz matrices and involve binomial coefficients. We have (see [29, p.138])

$$(\lambda I + N)^k = \sum_{i=0}^{k} \binom{k}{i} \lambda^{k-i} N^i.$$

If J is $n \times n$ then all terms $N^i = O$ for $i \geq n$. Then, for $k \geq n$,

$$(\lambda I + N)^k = \sum_{i=0}^{n-1} \binom{k}{i} \lambda^{k-i} N^i.$$

So $(\lambda I + N)^k I$ is the Hankel matrix

$$\begin{bmatrix}
\binom{k}{n-1}\lambda^{k-(n-1)} & \binom{k}{n-2}\lambda^{k-(n-2)} & \binom{k}{n-3}\lambda^{k-(n-3)} & \dots & \binom{k}{2}\lambda^{k-2} & k\lambda^{k-1} & \lambda^k \\
\binom{k}{n-2}\lambda^{k-(n-2)} & \binom{k}{n-3}\lambda^{k-(n-3)} & \dots & \binom{k}{2}\lambda^{k-2} & k\lambda^{k-1} & \lambda^k & 0 \\
\binom{k}{n-3}\lambda^{k-(n-3)} & \dots & \binom{k}{2}\lambda^{k-2} & k\lambda^{k-1} & \lambda^k & 0 & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\binom{k}{2}\lambda^{k-2} & k\lambda^{k-1} & \lambda^k & 0 & 0 & \dots & 0 \\
k\lambda^{k-1} & \lambda^k & 0 & 0 & \dots & 0 & 0 \\
\lambda^k & 0 & 0 & \dots & 0 & 0 & 0
\end{bmatrix}.$$

As long as $\lambda \neq 0$, it can be factored out of all the powers and is just a scalar converging either to 0 or to ∞ attached to U_k but it does not alter L_k (if $A = LU$ then $\alpha A = L(\alpha U)$). Thus, let Δ_λ be the matrix defined as

$$\Delta_\lambda := \mathrm{diag}(1, \lambda, \lambda^2, \dots, \lambda^{n-1})$$

and factor out λ^{k-n+1} to get

$$(\lambda I + N)^k I = \lambda^{k-n+1} \Delta_\lambda \widetilde{H}_k \Delta_\lambda$$

where

$$\widetilde{H}_k := \begin{bmatrix} \binom{k}{n-1} & \binom{k}{n-2} & \binom{k}{n-3} & \dots & \binom{k}{2} & k & 1 \\ \binom{k}{n-2} & \binom{k}{n-3} & \dots & \binom{k}{2} & k & 1 & 0 \\ \binom{k}{n-3} & \dots & \binom{k}{2} & k & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \binom{k}{2} & k & 1 & 0 & 0 & \dots & 0 \\ k & 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \end{bmatrix}.$$

So, factoring out λ^{k-n+1} and with a diagonal scaling we are free of the powers of λ and will concentrate on $\widetilde{H}_k$.

For $i + j \leq n + 1$, as a function of k, the (i, j) entry of $\widetilde{H}_k$ is a polynomial of degree $n + 1 - i - j$. Since $\widetilde{H}_k$ is strongly regular (which will be proved below), it has an LU factorization. If we say $\widetilde{H}_k = \widetilde{L}_k \widetilde{U}_k$, it may be verified that

$$\widetilde{L}_k = \begin{bmatrix} 1 & & & & & & \\ \mathcal{O}\left(\frac{1}{k}\right) & 1 & & & & & \\ \mathcal{O}\left(\frac{1}{k^2}\right) & \mathcal{O}\left(\frac{1}{k}\right) & 1 & & & & \\ \dots & \dots & \dots & \dots & & & \\ \mathcal{O}\left(\frac{1}{k^{n-3}}\right) & \dots & \mathcal{O}\left(\frac{1}{k^2}\right) & \mathcal{O}\left(\frac{1}{k}\right) & 1 & & \\ \mathcal{O}\left(\frac{1}{k^{n-2}}\right) & \mathcal{O}\left(\frac{1}{k^{n-3}}\right) & \dots & \mathcal{O}\left(\frac{1}{k^2}\right) & \mathcal{O}\left(\frac{1}{k}\right) & 1 & \\ \mathcal{O}\left(\frac{1}{k^{n-1}}\right) & \mathcal{O}\left(\frac{1}{k^{n-2}}\right) & \mathcal{O}\left(\frac{1}{k^{n-3}}\right) & \dots & \mathcal{O}\left(\frac{1}{k^2}\right) & \mathcal{O}\left(\frac{1}{k}\right) & 1 \end{bmatrix}.$$

So, for $p \geq 1$, the $(j + p, j)$ entry of $\widetilde{L}_k$ is $\mathcal{O}\big((\frac{1}{k})^p\big)$ and thus

$$\widetilde{L}_k \to I + E_k, \;\; E_k \to O \qquad \text{as} \qquad k \to \infty. \tag{4.24}$$

But the convergence is very slow, governed by the terms of $\mathcal{O}\left(\frac{1}{k}\right)$.

Since $\widetilde{H}_k$ is symmetric we can write $L_k U_k = L_k D_k L_k^T$. The diagonal matrix D_k is also a function of k but it does not converge to a finite matrix. Nevertheless, for each k,

$(\lambda I + N)^k \mathcal{I}$ has an LU factorization. We have

$$\begin{aligned}(\lambda I + N)^k \mathcal{I} &= \lambda^{k-n+1}\Delta_\lambda \widetilde{H}_k \Delta_\lambda \\ &= \lambda^{k-n+1}\Delta_\lambda \widetilde{L}_k \widetilde{U}_k \Delta_\lambda \\ &= \Delta_\lambda \widetilde{L}_k \Delta_\lambda^{-1} \lambda^{k-n+1} \Delta_\lambda \widetilde{U}_k \Delta_\lambda \\ &= L_k U_k\end{aligned}$$

with $L_k \equiv \Delta_\lambda \widetilde{L}_k \Delta_\lambda^{-1}$ and $U_k \equiv \lambda^{k-n+1}\Delta_\lambda \widetilde{U}_k \Delta_\lambda$. Since (4.24) occur we will also have

$$L_k \to I + G_k, \;\; G_k \to O \qquad \text{as} \qquad k \to \infty,$$

with $G_k = \Delta_\lambda E_k \Delta_\lambda^{-1}$. $\square$

In what follows we will prove that the matrix $\widetilde{H}_k$ is strongly regular.

Lemma 4.3.2 *$\widetilde{H}_k$ is strongly regular.*

Proof: As noticed before, as a function of k, the (i,j) entry of $\widetilde{H}_k$, $i + j \leq n + 1$, is a polynomial of degree $n + 1 - (i + j)$. It is given by

$$\begin{aligned}\binom{k}{n+1-i-j} &= \frac{k(k-1)\cdots\big(k-(n-i-j)\big)}{(n+1-i-j)!} \\ &= \frac{(k)_{n+1-i-j}}{(n+1-i-j)!}\end{aligned}$$

where $(k)_{m+1}$ is the *Pochammer symbol* defined as

$$(k)_0 := 1, \qquad (k)_{m+1} := k(k-1)\cdots(k-m), \;\; m \in \mathbb{N}_0.$$

So, in each entry (i,j), $i + j \leq n + 1$, the dominant term is

$$\frac{k^{n+1-i-j}}{(n+1-i-j)!}$$

and thus $\widetilde{H}_k = \widehat{H}_k$ plus lower order terms with

$$\widehat{H}_k := \begin{bmatrix} \frac{k^{n-1}}{(n-1)!} & \frac{k^{n-2}}{(n-2)!} & \frac{k^{n-3}}{(n-3)!} & \dots & \frac{k^2}{2!} & k & 1 \\ \frac{k^{n-2}}{(n-2)!} & \frac{k^{n-3}}{(n-3)!} & \dots & \frac{k^2}{2!} & k & 1 & 0 \\ \frac{k^{n-3}}{(n-3)!} & \dots & \frac{k^2}{2!} & k & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{k^2}{2!} & k & 1 & 0 & 0 & \dots & 0 \\ k & 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \end{bmatrix}.$$

In order to eliminate k from $\widehat{H}_k$ we define

$$\Delta_k := \mathrm{diag}(1, k, k^2, \dots, k^{n-1})$$

and we have

$$\mathit{I}\,\Delta_k\,\mathit{I} = \mathrm{diag}(k^{n-1}, \dots, k^2, k, 1).$$

Now it can be verified that

$$\widehat{H}_k = \Delta_k^{-1} F_n^{(n-1)}\,\mathit{I}\,\Delta_k\,\mathit{I}$$

with $F_n^{(n-1)}$ given by

$$F_n^{(n-1)} = \begin{bmatrix} \frac{1}{(n-1)!} & \frac{1}{(n-2)!} & \frac{1}{(n-3)!} & \dots & \frac{1}{2!} & 1 & 1 \\ \frac{1}{(n-2)!} & \frac{1}{(n-3)!} & \dots & \frac{1}{2!} & 1 & 1 & 0 \\ \frac{1}{(n-3)!} & \dots & \frac{1}{2!} & 1 & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{1}{2!} & 1 & 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \end{bmatrix}.$$

To prove that $\widetilde{H}_k$ is completely regular it suffices to prove that $F_n^{(n-1)}$ has this property (see lemma 4.2.4, page 93). First observe that $F_n^{(n-1)}$ is nonsingular since $\det\left(F_n^{(n-1)}\right) = (-1)^{n-1}$. Now we still have to show that all the l^{th} leading principal

minors of $F_n^{(n-1)}$ are nonzero, $l = 1, \ldots, n-1$. This follows immediately from the next lemma. $\square$

Recall that the factorial of a positive integer is defined as

$$m! = m(m-1)(m-2)\cdots 2.1, \qquad m \in \mathbb{N}$$

By convention $0! = 1$. We will say that for a negative integer m, $m! = 0$. Also we define the double factorial symbol (!!) as follows

$$\begin{aligned} m!! &= m(m-1)!(m-2)!\cdots 2!1!, \qquad m \in \mathbb{N} \\ 0!! &= 1. \end{aligned}$$

This definition is not universal.

Define the $j \times j$ Hankel matrix

$$F_j^{(n-1)} := \begin{bmatrix} \frac{1}{(n-1)!} & \frac{1}{(n-2)!} & \frac{1}{(n-3)!} & \cdots & \frac{1}{(n-j)!} \\ \frac{1}{(n-2)!} & \frac{1}{(n-3)!} & \cdots & \cdots & \frac{1}{(n-j-1)!} \\ \frac{1}{(n-3)!} & \cdots & \cdots & \cdots & \frac{1}{(n-j-2)!} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \frac{1}{(n-j)!} & \frac{1}{(n-j-1)!} & \frac{1}{(n-j-2)!} & \cdots & \frac{1}{(n-2j+1)!} \end{bmatrix}.$$

The following lemma gives us a useful result to calculate the determinant of $F_j^{(n-1)}$. It seems likely that this result may be already known but we couldn't find it in a search of the literature.

Lemma 4.3.3 *Let $j < n$. Then*

$$\det\left(F_j^{(n-1)}\right) = (-1)^{j(j-1)/2} \frac{(j-1)!!(n-j-1)!!}{(n-1)!!}.$$

Proof. There are two cases to consider: first, $j \leq \frac{n+1}{2}$ and, second, $n > j > \frac{n+1}{2}$.

Case 1: $j \leq \frac{n+1}{2}$

Factor out $\frac{1}{(n-l)!}$ from column l, $l = 1, 2, \ldots, j$, to obtain the new integer matrix

$$\widetilde{F}_j^{(n-1)} := \begin{bmatrix} 1 & 1 & \ldots & 1 \\ n-1 & n-2 & \ldots & n-j \\ (n-1)(n-2) & (n-2)(n-3) & \ldots & (n-j)(n-j-1) \\ \vdots & \vdots & \vdots & \vdots \\ (n-1)\cdots(n-j+1) & (n-2)\cdots(n-j) & \ldots & (n-j)\cdots(n-2j+2) \end{bmatrix}$$

with

$$\det\left(F_j^{(n-1)}\right) = \frac{1}{(n-1)!}\frac{1}{(n-2)!}\cdots\frac{1}{(n-j)!}\det\left(\widetilde{F}_j^{(n-1)}\right).$$

By subtracting suitable multiples of higher rows from lower rows we are left with the transpose of the standard Vandermond Matrix (see (4.12), page 90)

$$V_{\{n-1,n-2,\ldots,n-j\}} =: V_j.$$

In fact, if we use the Pochammer symbol defined above, it is clear that $\widetilde{F}_j^{(n-1)}$ is a polynomial Vandermond matrix,

$$\widetilde{F}_j^{(n-1)} = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ (n-1)_1 & (n-2)_1 & \ldots & (n-j)_1 \\ (n-1)_2 & (n-2)_2 & \ldots & (n-j)_2 \\ \vdots & \vdots & \vdots & \vdots \\ (n-1)_{j-1} & (n-2)_{j-1} & \ldots & (n-j)_{j-1} \end{bmatrix}, \tag{4.25}$$

and the determinant of any polynomial Vandermond matrix is just the determinant of the standard Vandermond matrix (see lemma 4.2.2, page 91).

To be more precise, if we denote the k^{th} row of $\widetilde{F}_j^{(n-1)}$ by Row_l, $l = 1, \ldots, j$, the row operations we have to perform are

$$\text{Row}_k \leftarrow \text{Row}_k + (k-2)\,\text{Row}_{k-1} + (k-3)(n-1)\,\text{Row}_{k-2} + \ldots + 1.(n-1)^{k-3}\,\text{Row}_2,$$

for $k = n, n-1, \ldots, 3$ (note that we start from the bottom). We will end with the leading (monic) terms

$$\begin{bmatrix} 1 & 1 & \ldots & 1 \\ n-1 & n-2 & \ldots & n-j \\ (n-1)^2 & (n-2)^2 & \ldots & (n-j)^2 \\ \vdots & \vdots & \vdots & \vdots \\ (n-1)^{j-1} & (n-2)^{j-1} & \ldots & (n-j)^{j-1} \end{bmatrix}. \tag{4.26}$$

Hence,

$$\det\left(F_j^{(n-1)}\right) = \frac{1}{(n-1)!}\frac{1}{(n-2)!}\cdots\frac{1}{(n-j)!}\det(V_j).$$

But, from (4.13), page 91, we know that

$$\begin{aligned} \det(V_j) &= \prod_{j\geq k>i\geq 1}\big((n-k)-(n-i)\big) \\ &= \prod_{j\geq k>i\geq 1}(i-k) \\ &= (-1)^{j(j-1)/2}\prod_{j\geq k>i\geq 1}(k-i) \\ &= (-1)^{j(j-1)/2}(j-1)!(j-2)!\cdots 2!1!. \end{aligned}$$

Notice that $j(j-1)/2$ is the number of terms in the product $\prod_{j\geq k>i\geq 1}(i-k)$.

Finally, we can write

$$\begin{aligned} \det\left(F_j^{(n-1)}\right) &= (-1)^{j(j-1)/2}\frac{(j-1)!}{(n-1)!}\frac{(j-2)!}{(n-2)!}\cdots\frac{0!}{(n-j)!} \\ &= (-1)^{j(j-1)/2}\frac{(j-1)!!(n-j-1)!!}{(n-1)!!}, \qquad n\geq 2j-1. \end{aligned} \tag{4.27}$$

We have established the result for $n \geq 2j-1$, that is, for $j \leq \dfrac{n+1}{2}$.

Case 2: Now consider the case $n > j > \dfrac{n+1}{2}$. In this case we will have

$$F_j^{(n-1)} = \begin{bmatrix} \frac{1}{(n-1)!} & \frac{1}{(n-2)!} & \frac{1}{(n-3)!} & \dots & \dots & \dots & \dots & \dots & \dots & \frac{1}{(n-j)!} \\ \frac{1}{(n-2)!} & \frac{1}{(n-3)!} & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \frac{1}{(n-j-1)!} \\ \frac{1}{(n-3)!} & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \frac{1}{(n-j-2)!} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & 1 & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & 1 & 1 & 0 & \dots & 0 \\ \frac{1}{(n-j)!} & \frac{1}{(n-j-1)!} & \frac{1}{(n-j-2)!} & \dots & 1 & 1 & 0 & \dots & 0 & 0 \end{bmatrix}.$$

Again, if we factor out $\frac{1}{(n-l)!}$ for column l, $l = 1, 2, \dots, j$, we are left with the polynomial Vandermond matrix with the Pochammer symbol as in (4.25). Its just the case that some values of the Pochammer symbol are equal to zero. So, some zeros appear in the lower right corner of $\widetilde{F}_j^{(n-1)}$ but it is still valid that

$$\det\left(\widetilde{F}_j^{(n-1)}\right) = \det(V_j)$$

and, then, formula (4.27) also applies to the case $n > j > \dfrac{n+1}{2}$.

Observe that from the symmetry of the formula (4.27) it follows, up to the sign,

$$\det\left(F_j^{(n-1)}\right) = \det\left(F_{n-j}^{(n-1)}\right). \ \square$$

Chapter 5

Triple dqds algorithm

In this chapter we describe the derivation of a first version of the triple dqds - an algorithm that performs implicitly three steps of simple dqds keeping real arithmetic in the presence of complex shifts. A preliminary version was developed by Z. Wu [64]. We start by describing the connection to the implicit double shifted LR algorithm and then go into the details of a practical implementation. The relation between dqds and the Gram-Schmidt orthogonalization process can be used to establish new results about triple dqds. We defer this study to the next chapter. For a discussion of the relation of LR to dqds see Fernando and Parlett [16].

5.1 Triple dqds algorithm

The essential point of triple dqds algorithm is the conversion of the implicit double shifted LR algorithm (see page 65) into a dqds format. There are two main reasons to pursue this goal - first, there is both theoretical and practical evidence that the eigenvalues are usually better defined by the entries of L and U rather than by the entries of the product $J = LU$, and, second, dqds produces with no cost the quantities d_i, $i = 1, 2, \ldots, n$, (see page 75) that may be used in a more efficient shift strategy than the one used by the LR algorithm.

Let L_1 and U_1 be real and have the form presented in section 3.5.1 (page 72),

$$L_1 = \begin{bmatrix} 1 & & & & & \\ l_1 & 1 & & & & \\ & l_2 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & l_{n-2} & 1 & \\ & & & & l_{n-1} & 1 \end{bmatrix}, \qquad U_1 = \begin{bmatrix} u_1 & 1 & & & & \\ & u_2 & 1 & & & \\ & & u_3 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & u_{n-1} & 1 \\ & & & & & u_n \end{bmatrix}.$$

Consider that we apply three dqds steps to L_1 and R_1. The results of shifting by σ_1, σ_2 and σ_3 in succession are

$$U_1L_1 - \sigma_1 I = L_2U_2 \tag{5.1}$$

$$U_2L_2 - \sigma_2 I = L_3U_3 \tag{5.2}$$

$$U_3L_3 - \sigma_3 I = L_4U_4 \tag{5.3}$$

The following lemma relates L_4 and U_4 to L_1 and U_1.

Lemma 5.1.1 *Consider that matrices L_4 and U_4 result from the application of three* dqds *steps with successive shifts σ_1, σ_2 and σ_3. Then*

$$L_4U_4 = \mathcal{L}^{-1}U_1L_1\mathcal{L} - (\sigma_1 + \sigma_2 + \sigma_3)I$$

where $\mathcal{L} = L_2L_3$.

Proof. We have

$$\begin{aligned} L_4U_4 =& U_3L_3 - \sigma_3 I \\ =& L_3^{-1}(L_3U_3)L_3 - \sigma_3 I \\ =& L_3^{-1}(U_2L_2 - \sigma_2 I)L_3 - \sigma_3 I & (5.4) \\ =& L_3^{-1}\big(L_2^{-1}(U_1L_1 - \sigma_1 I)L_2 - \sigma_2\big)L_3 - \sigma_3 I & (5.5) \\ =& (L_2L_3)^{-1}U_1L_1(L_2L_3) - (\sigma_1 + \sigma_2 + \sigma_3)I \\ =& \mathcal{L}^{-1}U_1L_1\mathcal{L} - (\sigma_1 + \sigma_2 + \sigma_3)I, \end{aligned}$$

where $\mathcal{L} \equiv L_2L_3$. □

Observe that

$$\mathcal{L}^{-1}U_1L_1\mathcal{L} - (\sigma_1+\sigma_2+\sigma_3)I = \mathcal{L}^{-1}\big(U_1L_1 - (\sigma_1+\sigma_2+\sigma_3)I\big)\mathcal{L}$$

to realize that lemma 5.1.1 just says that matrices L_4U_4 and $U_1L_1 - (\sigma_1+\sigma_2+\sigma_3)I$ are similar.

Now consider the matrix $\mathcal{U} \equiv U_3U_2$. Next lemma tells us about the LU decomposition $\mathcal{LU}$.

Lemma 5.1.2 *Let $\mathcal{L} = L_2L_3$ and $\mathcal{U} = U_3U_2$. Then*

$$\mathcal{LU} = (U_1L_1)^2 - (2\sigma_1+\sigma_2)U_1L_1 + (\sigma_1\sigma_2+\sigma_1^2)I$$

Proof. We have

$$\begin{aligned}
\mathcal{LU} =& (L_2L_3)(U_3U_2)\\
=& L_2(L_3U_3)U_2\\
=& L_2(U_2L_2 - \sigma_2 I)U_2\\
=& (L_2U_2)^2 - \sigma_2 L_2U_2\\
=& (U_1L_1 - \sigma_1 I)^2 - \sigma_2(U_1L_1 - \sigma_1 I)\\
=& (U_1L_1)^2 - (2\sigma_1+\sigma_2)U_1L_1 + (\sigma_1\sigma_2+\sigma_1^2)I. \square
\end{aligned}$$

Next we will show how to do a *triple shift* in order to keep real arithmetic in the presence of complex eigenvalues. That is, we will see how to combine three consecutive dqds iterations with complex shifts σ_1, σ_2 and σ_3 such that the resulting matrices L_4 and U_4 after this triple shift will be again real.

Matrices $\mathcal{L} = L_2L_3$ and $\mathcal{U} = U_3U_2$ will have real entries if we choose

$$\sigma_2 = -2(\Im\sigma_1)\mathbf{i},$$

where $\Im\sigma_1$ denotes the imaginary part of σ_1 and $\mathbf{i}$ is the imaginary unit. This is the only nontrivial complex solution to

$$\begin{cases} 2\sigma_1+\sigma_2 \in \mathbb{R} \\ \sigma_1\sigma_2+\sigma_1^2 \in \mathbb{R} \end{cases}.$$

It is easy to verify that this choice for σ_2 is such that

$$2\sigma_1 + \sigma_2 = 2(\Re\sigma_1) \quad \text{and} \quad \sigma_1\sigma_2 + \sigma_1^2 = |\sigma_1|^2,$$

where $\Re\sigma_1$ denotes the real part of σ_1. Thus, according to lemma 5.1.2 above,

$$\mathcal{LU} = (U_1L_1)^2 - 2(\Re\sigma_1)U_1L_1 + |\sigma_1|^2 I,$$

which is the same matrix that appears in the double shifted LR algorithm if we consider $A_1 \equiv U_1L_1$ (see lemma 3.4.1, page 66).

By lemma 5.1.1, the accumulated shift is then

$$\sigma_1 + \sigma_2 + \sigma_3 = \overline{\sigma}_1 + \sigma_3$$

and if we choose

$$\sigma_3 = a + (\Im\sigma_1)\mathbf{i}, \quad a \in \mathbb{R},$$

the arithmetic will be retained real.

Finally, notice that the choice

$$\sigma_3 = -\overline{\sigma}_1$$

ensures that the transformation from L_1 and U_1 to L_4 and U_4 is a restoring shift transformation since $\sigma_1 + \sigma_2 + \sigma_3 = 0$.

What we have just showed is summarized in the following lemma.

Lemma 5.1.3 *Performing three steps of* dqds *algorithm with successive shifts* $\sigma_1 \in \mathbb{C}$, $\sigma_2 = -2(\Im\sigma_1)\mathbf{i}$ *and* $\sigma_3 = \overline{\sigma}_1$ *retains real arithmetic and the shifts are restored. We will have*

$$\mathcal{LU} = (U_1L_1)^2 - 2(\Re\sigma_1)U_1L_1 + |\sigma_1|^2 I$$
$$L_4U_4 = \mathcal{L}^{-1}U_1L_1\mathcal{L}$$

where $\mathcal{L} = L_2L_3$ *and* $\mathcal{U} = U_3U_2$.

Now the three dqds steps (5.1), (5.2) and (5.3) become

$$U_1L_1 - \sigma_1 I = L_2U_2 \tag{5.6}$$

$$U_2L_2 - \big(- 2(\Im\sigma_1)\mathbf{i}I\big) = L_3U_3 \tag{5.7}$$

$$U_3L_3 - (-\overline{\sigma}_1 I) = L_4U_4 \tag{5.8}$$

and can be rewritten in a more revealing way:

$$\begin{aligned} U_1L_1 - \sigma_1 I &= L_2U_2 \\ (U_2L_2 + \sigma_1 I) - \overline{\sigma}_1 I &= L_3U_3 \\ (U_3L_3 + \overline{\sigma}_1 I) - 0I &= L_4U_4. \end{aligned}$$

Notice that if we only applied the two dqds steps (5.6) and (5.7), the transformation from L_1 and U_1 to L_3 and U_3 would not be a restoring shift transformation, L_3 and U_3 would be complex as well as the accumulated shift $\sigma_1 + \sigma_2 = \sigma_1 + (-2(\Im\sigma_1)\mathbf{i}) = \overline{\sigma}_1$. The third step (5.8) is therefore needed to come back to real factors L_4 and U_4, restoring the shift and getting L_4U_4 similar to U_1L_1.

Another way of analyzing the need of a triple shift is as follows. If we consider only the two steps (5.6) and (5.7) we will have

$$\begin{gathered} L_3U_3 = L_2^{-1}(U_1L_1)L_2 - \overline{\sigma}_1 I \\ U_3L_3 = L_3^{-1}L_2^{-1}(U_1L_1)L_2L_3 - \overline{\sigma}_1 I = \mathcal{L}^{-1}(U_1L_1)\mathcal{L} - \overline{\sigma}_1 I \\ \mathcal{L}\mathcal{U} = (U_1L_1)^2 - 2(\Re\sigma_1)U_1L_1 + |\sigma_1|^2 I, \end{gathered}$$

where $\mathcal{L} = L_2L_3$ and $\mathcal{U} = U_3U_2$. Thus L_1, U_1, $\mathcal{L}$ and $\mathcal{U}$ are real and we can write

$$\mathcal{L}^{-1}(U_1L_1)\mathcal{L} = U_3L_3 + \overline{\sigma}_1 I,$$

which means that $U_3L_3 + \overline{\sigma}_1$ is also real. So, we must have to do the third step (5.8) to get the factors L_4 and U_4 of the LU factorization

$$U_3L_3 + \overline{\sigma}_1 = L_4U_4.$$

The goal now will be to avoid complex factors L_2, U_2, L_3 and U_3 and go straight from L_1 and U_1 to L_4 and U_4, performing the three steps implicitly - *implicit triple dqds* algorithm.

5.2 Connection to the implicit double shifted LR algorithm

This section aims to relate the double shifted LR algorithm to the triple shift dqds algorithm and use the implicit version of the first to derive the implicit version of the second.

If we performed the products U_iL_i, $i = 1, 2, 3$, and added back the shifts $\sigma = \sigma_1$ and $\overline{\sigma} = \overline{\sigma}_1$, the three dqds steps (5.6), (5.7) and (5.8) would correspond to a basic iteration (zero shift) followed by a double shifted iteration (shifts σ and $\overline{\sigma}$) of the LR algorithm with an extra LU factorization in the end. In more detail,

LR algorithm	dqds
$\begin{cases} J_1 = L_1U_1 \\ J_2 = U_1L_1 \end{cases}$	
$\begin{cases} J_2 - \sigma I = L_2U_2 \\ J_3 = U_2L_2 + \sigma I \end{cases}$	$U_1L_1 - \sigma I = L_2U_2$
$\begin{cases} J_3 - \overline{\sigma} I = L_3U_3 \\ J_4 = U_3L_3 + \overline{\sigma} I \end{cases}$	$(U_2L_2 + \sigma I) - \overline{\sigma} I = L_3U_3$
$\begin{cases} J_4 = L_4U_4 \\ \dots \end{cases}$	$(U_3L_3 + \overline{\sigma} I) - 0I = L_4U_4$

Figure 5.1 below also exhibits this relation - LR algorithm goes with matrices J_i and triple dqds goes with the factors L_i and U_i - and shows clearly the need of three dqds steps in order to go from real L_1 and U_1 to real L_4 and U_4.

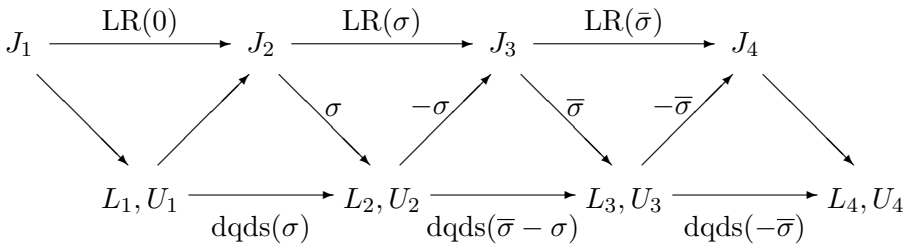


Figure 5.1: Double shift LR and three steps of dqds

Recall from lemma 3.4.1 (page 65) that after a double shifted LR iteration applied to J_2 with shifts σ and $\overline{\sigma}$ we will have

$$\mathbf{LU} = J_2^2 - 2(\Re\sigma)J_2 + |\sigma|^2 I \equiv M$$

and

$$J_4 = \mathbf{L}^{-1} J_2 \mathbf{L}$$

where $\mathbf{L} = L_2 L_3$ is unit lower triangular and $\mathbf{U} = U_3 U_2$ is upper triangular. This is exactly what lemma 5.1.3 tells for the triple dqds algorithm but focusing on the factors of the LU decompositions $J_2 = U_1 L_1$ and $J_4 = L_4 U_4$,

$$\mathcal{LU} = (U_1 L_1)^2 - 2(\Re\sigma)U_1 L_1 + |\sigma|^2 I$$
$$L_4 U_4 = \mathcal{L}^{-1} U_1 L_1 \mathcal{U}$$

with $\mathcal{L} = \mathbf{L}$ and $\mathcal{U} = \mathbf{U}$.

To obtain matrix J_4, the implicit double shifted LR algorithm uses the technique of bulge chasing described in section 3.4.1 (page 68) going directly from tridiagonal matrix J_2 to J_4. This is justified by the implicit L theorem that says that matrix $\mathbf{L}$ is uniquely determined by its first column which we can compute because it is proportional to the first column of M.

Using the same notation of example 3.4.1, $\mathbf{L}$ will be given as a product of elementary or

Gauss matrices L_i, $i = 1, \ldots, n-1$,

$$\mathsf{L} = \mathsf{L}_1 \mathsf{L}_2 \cdots \mathsf{L}_{n-1},$$

where $\mathsf{L}_i = I + \mathsf{l}'_i e_i^T$ with $\mathsf{l}'_i = \begin{bmatrix} 0 & \ldots & 0 & * & * & 0 & \ldots & 0 \end{bmatrix}^T$ (the non-zero elements $*$ appear in entries $i+1$ and $i+2$) and J_4 will be the result of successive similarity transformations given by L_i,

$$\begin{aligned} J_4 &= \mathsf{L}_{n-1}^{-1} \cdots \mathsf{L}_1^{-1} J_2 \mathsf{L}_1 \cdots \mathsf{L}_{n-1} \\ &= (\mathsf{L}_1 \cdots \mathsf{L}_{n-1})^{-1} J_2 \mathsf{L}_1 \cdots \mathsf{L}_{n-1}. \end{aligned}$$

So, L_1 is computed and the first similarity transformation $\mathsf{L}_1^{-1} J_2 \mathsf{L}_1$ spoils the tridiagonal form of J_2. Then the bulge is pushed down and to the right to restore the tridiagonal form using the transformations given by L_i, $i = 2, \ldots, n-1$. In the end we have J_4.

Analogously, to obtain L_4 and U_4, the implicit triple dqds algorithm will construct $\mathcal{L}$ as a product of elementary matrices $\mathcal{L}_i$, $i = 1, \ldots, n-1$, obtained through a similar process of bulge chasing, such that the factors L_4 and U_4 will result from

$$L_4 U_4 = \left(\mathcal{L}_{n-1}^{-1} \ldots \mathcal{L}_2^{-1} \mathcal{L}_1^{-1} U_1\right) \left(L_1 \mathcal{L}_1 \mathcal{L}_2 \ldots \mathcal{L}_{n-1}\right).$$

But things will be done in such a way that the products $U_1 L_1$ and $L_4 U_4$ are not computed explicitly. We will get L_4 and U_4 from L_1 and U_1 without computing any products of the form LU or UL explicitly.

We can now ask: "Do we have $\mathsf{L}_i = \mathcal{L}_i$, $i = 1, \ldots, n-1$?" Implicit L theorem guarantees that $\mathsf{L} = \mathcal{L}$, because the first column of both matrices is the same and there is no more choice. And the expression for the product (see section 2.1, page 25)

$$\mathsf{L}_1 \mathsf{L}_2 \cdots \mathsf{L}_{n-1} = \mathsf{L} = \mathcal{L} = \mathcal{L}_1 \mathcal{L}_2 \ldots \mathcal{L}_{n-1}$$

ensures that

$$\mathsf{L}_i = \mathcal{L}_i, \qquad i = 1, \ldots, n-1.$$

In summary, starting with the factors L_1 and U_1 and the shift σ, if we

- normalize 1st column of $(U_1L_1)^2 - 2(\Re\sigma_1)U_1L_1 + |\sigma_1|^2 I$ (equal to first column of $\mathcal{L}$) and compute $\mathcal{L}_1$
- spoil the bidiagonal form with $\underbrace{\mathcal{L}_1^{-1}U_1}\,\underbrace{L_1\mathcal{L}_1}$ and
- apply bulge chasing to get L' and U'

$$L'U' = \underbrace{\mathcal{L}_{n-1}^{-1}\ldots\mathcal{L}_2^{-1}\mathcal{L}_1^{-1}U_1}\,\underbrace{L_1\mathcal{L}_1\mathcal{L}_2\ldots\mathcal{L}_{n-1}},$$

we will obtain implicitly the same result of performing the three dqds steps (5.6), (5.7) and (5.8), that is, we will get $L' = L_4$ and $U' = U_4$.

We could have started thinking only in performing the two dqds steps (5.6) and (5.7), that is a *double dqds* algorithm. If we decided to use bulge chasing, we would't get L_3 and U_3 but L_4 and U_4, ending up realizing that we were effectively performing a triple dqds iteration.

5.3 Derivation of triple dqds

The role of the implicit triple dqds algorithm will be to construct matrices $\mathcal{L}_i$, $i = 1, \ldots, n$, and the unique matrix X such that

$$\begin{aligned} L_4 =& \mathcal{L}_n^{-1}\mathcal{L}_{n-1}^{-1}\ldots\mathcal{L}_2^{-1}\mathcal{L}_1^{-1}U_1X^{-1} \\ U_4 =& XL_1\mathcal{L}_1\mathcal{L}_2\ldots\mathcal{L}_{n-1}\mathcal{L}_n \end{aligned}$$

(we will have $\mathcal{L}_n = I$). Matrix X will be best written as

$$X = X_nX_2\cdots X_1$$

and each X_i, $i = 1, \ldots, n$, will be the product of two matrices

$$X_i \equiv Y_iZ_i.$$

Conceptually, for each step of the implicit triple dqds algorithm we will start with $F := U_1$ and $G := L_1$ and, at each *minor* step i, $i = 1, \ldots, n$, matrices Z_i, $\mathcal{L}_i$ and Y_i are chosen to chase the bulges. After n minor steps, F will be transformed into L_4 and G into U_4. So, we will be transforming an upper bidiagonal U_1 into a lower bidiagonal L_4 and a lower bidiagonal L_1 into an upper bidiagonal U_4. Details will be given in the next section.

Consider matrices L_1 and U_1 as follows

$$L_1 = \begin{bmatrix} 1 & & & & & \\ l_1 & 1 & & & & \\ & l_2 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & l_{n-2} & 1 & \\ & & & & l_{n-1} & 1 \end{bmatrix} \text{ and } U_1 = \begin{bmatrix} u_1 & 1 & & & & \\ & u_2 & 1 & & & \\ & & u_3 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & u_{n-1} & 1 \\ & & & & & u_n \end{bmatrix}. \tag{5.9}$$

The product U_1L_1 is

$$U_1L_1 = \begin{bmatrix} u_1 + l_1 & 1 & & & & \\ u_2l_1 & u_2 + l_2 & 1 & & & \\ & u_3l_2 & u_3 + l_3 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & u_{n-1}l_{n-2} & u_{n-1} + l_{n-1} & 1 \\ & & & & u_nl_{n-1} & u_n \end{bmatrix}$$

and the first column of $(U_1L_1)^2 - 2(\Re\sigma)U_1L_1 + |\sigma|^2 I$ is

$$\begin{bmatrix} (u_1 + l_1)^2 + u_2l_1 - 2(\Re\sigma)(u_1 + l_1) + |\sigma|^2 \\ u_2l_1(u_1 + l_1 + u_2 + l_2 - 2(\Re\sigma)) \\ u_2l_1u_3l_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{5.10}$$

We will start by presenting a 6×6 example to illustrate the general pattern of matrices Z_i, $\mathcal{L}_i$ and Y_i. If we work through the example in detail, we will understand the algorithm.

Example 5.3.1

For a 6×6 example consider

$$L_1 = \begin{bmatrix} 1 & & & & & \\ l_1 & 1 & & & & \\ & l_2 & 1 & & & \\ & & l_3 & 1 & & \\ & & & l_4 & 1 & \\ & & & & l_5 & 1 \end{bmatrix} \quad \text{and} \quad U_1 = \begin{bmatrix} u_1 & 1 & & & & \\ & u_2 & 1 & & & \\ & & u_3 & 1 & & \\ & & & u_4 & 1 & \\ & & & & u_5 & 1 \\ & & & & & u_6 \end{bmatrix}.$$

We seek

$$L_4 := \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & & \\ & & & \hat{l}_4 & 1 & \\ & & & & \hat{l}_5 & 1 \end{bmatrix} \quad \text{and} \quad U_4 := \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \hat{u}_4 & 1 & \\ & & & & \hat{u}_5 & 1 \\ & & & & & \hat{u}_6 \end{bmatrix}.$$

Initially, we define

$$F := U_1 \qquad \text{and} \qquad G := L_1$$

and

$$F^{(1)} := F \qquad \text{and} \qquad G^{(1)} := G.$$

Step 1

1.a) Define Z_1^{-1} and obtain

$$\underbrace{FZ_1^{-1}}\underbrace{Z_1 G} = \underbrace{U_1 Z_1^{-1}}\underbrace{Z_1 L_1}.$$

Matrix Z_1^{-1} is chosen to zero out entry $(1,2)$ of F and put 1 into $(1,1)$ entry. Observe that

$$\begin{bmatrix} u_1 & 1 \\ 0 & u_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & u_2 \end{bmatrix} \begin{bmatrix} u_1 & 1 \\ 0 & 1 \end{bmatrix}.$$

Then we have

$$Z_1^{-1} = \begin{bmatrix} \frac{1}{u_1} & -\frac{1}{u_1} & & & & & \\ 0 & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix} \quad \text{and} \quad Z_1 = \begin{bmatrix} u_1 & 1 & & & & & \\ 0 & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix}.$$

Thus

$$FZ_1^{-1} = \begin{bmatrix} 1 & 0 & & & & & \\ & u_2 & 1 & & & & \\ & & u_3 & 1 & & & \\ & & & u_4 & 1 & & \\ & & & & u_5 & 1 & \\ & & & & & u_6 & \end{bmatrix} \text{ and } Z_1G = \begin{bmatrix} l_1+u_1 & 1 & & & & & \\ l_1 & 1 & & & & & \\ & l_2 & 1 & & & & \\ & & l_3 & 1 & & & \\ & & & l_4 & 1 & & \\ & & & & l_5 & 1 & \end{bmatrix}.$$

Observe that entries $(1,1)$ of F and $(1,2)$ of G became 1.

Let

$$F \longleftarrow FZ_1^{-1} \qquad \text{and} \qquad G \longleftarrow Z_1G.$$

1.b) Define elementary matrix $\mathcal{L}_1$ to get

$$\underbrace{\mathcal{L}_1^{-1}F}\,\underbrace{G\mathcal{L}_1} = \underbrace{\mathcal{L}_1^{-1}U_1Z_1^{-1}}\,\underbrace{Z_1L_1\mathcal{L}_1}.$$

Let $\boldsymbol{l}_1 = \begin{bmatrix} 1 & * & * & 0 & 0 & 0 \end{bmatrix}^T$ be the first column of $\mathcal{L}_1$, obtained by normalizing (5.10) for $n = 6$. We will have $\mathcal{L}_1 = I + \boldsymbol{l}_1'\boldsymbol{e}_1^T$, where $\boldsymbol{l}_1' = \begin{bmatrix} 0 & * & * & 0 & 0 & 0 \end{bmatrix}^T$, and $\mathcal{L}_1^{-1} = I - \boldsymbol{l}_1'\boldsymbol{e}_1^T$. Thus,

$$\mathcal{L}_1^{-1}F = \begin{bmatrix} 1 & 0 & & & & \\ * & u_2 & 1 & & & \\ + & & u_3 & 1 & & \\ & & & u_4 & 1 & \\ & & & & u_5 & 1 \\ & & & & & u_6 \end{bmatrix} \qquad \text{and} \qquad G\mathcal{L}_1 = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ + & 1 & & & & \\ + & l_2 & 1 & & & \\ + & & l_3 & 1 & & \\ & & & l_4 & 1 & \\ & & & & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow \mathcal{L}_1^{-1}F \qquad \text{and} \qquad G \longleftarrow G\mathcal{L}_1.$$

Note that the preliminary transform Z_1 ensured that, at this point, column 2 of F is unchanged. The first row of F and first row of G are in final form, but first column of F and first column of G have bulges that are indicated by plus signs. Next, the 3×1 bulge in G will be chased.

1.c) Define elementary matrix Y_1 to obtain

$$\underbrace{FY_1^{-1}}\underbrace{Y_1G} = \underbrace{\mathcal{L}_1^{-1}U_1Z_1^{-1}Y_1^{-1}}\underbrace{Y_1Z_1L_1\mathcal{L}_1}.$$

Matrix Y_1 affects only rows 2, 3 and 4 of Y_1G, zeroing out entries $(2,1)$, $(3,1)$ and $(4,1)$ of G. So, $Y_1^{-1} = I + \boldsymbol{y}_1'\boldsymbol{e}_1^T$ with $\boldsymbol{y}_1' = \begin{bmatrix} 0 & * & * & * & 0 & 0 \end{bmatrix}^T$, $Y_1 = I - \boldsymbol{y}_1'\boldsymbol{e}_1^T$ and

$$FY_1^{-1} = \begin{bmatrix} 1 & 0 & & & & \\ \hat{l}_1 & u_2 & 1 & & & \\ + & & u_3 & 1 & & \\ + & & & u_4 & 1 & \\ & & & & u_5 & 1 \\ & & & & & u_6 \end{bmatrix} \qquad \text{and} \qquad Y_1G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & * & & & & \\ & + & 1 & & & \\ & + & l_3 & 1 & & \\ & & & l_4 & 1 & \\ & & & & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow FY_1^{-1} \qquad \text{and} \qquad G \longleftarrow Y_1 G$$

and define

$$F^{(2)} := F = \mathcal{L}_1^{-1} U_1 Z_1^{-1} Y_1^{-1}$$
$$G^{(2)} := G = Y_1 Z_1 L_1 \mathcal{L}_1.$$

This completes the first step. The first column of G is in final form but the first column of F is not yet in final form - there exists a 2×1 bulge.

The pattern shown above is carried down the matrix by later transformations Z_i, $\mathcal{L}_i$ and Y_i, $i = 2, \ldots, 6$, forcing the bulges down by one row and one column at each step. This way in the end F will be transformed into the lower bidiagonal matrix L_4 and G into the upper bidiagonal matrix U_4.

Step 2

2.a) Define Z_2^{-1} and get

$$\underbrace{F Z_2^{-1}}\,\underbrace{Z_2 G} = \underbrace{F^{(2)} Z_2^{-1}}\,\underbrace{Z_2 G^{(2)}}.$$

Matrix Z_2^{-1} will zero out entry $(2,3)$ of F and place 1 into entry $(2,2)$. Matrix Z_2 will turn entry $(2,3)$ of G into 1. We have

$$Z_2^{-1} = \begin{bmatrix} 1 & & & & & & \\ & \frac{1}{u_2} & -\frac{1}{u_2} & & & & \\ & 0 & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix}, \qquad Z_2 = \begin{bmatrix} 1 & & & & & & \\ & u_2 & 1 & & & & \\ & 0 & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix}$$

and

$$FZ_2^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & 0 & & & \\ + & & u_3 & 1 & & \\ + & & & u_4 & 1 & \\ & & & & u_5 & 1 \\ & & & & & u_6 \end{bmatrix}, \qquad Z_2G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & * & 1 & & & \\ & + & 1 & & & \\ & + & l_3 & 1 & & \\ & & & l_4 & 1 & \\ & & & & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow FZ_2^{-1} \qquad \text{and} \qquad G \longleftarrow Z_2G.$$

2.b) Define elementary matrix $\mathcal{L}_2$ to obtain

$$\underbrace{\mathcal{L}_2^{-1}F}\,\underbrace{G\mathcal{L}_2} = \underbrace{\mathcal{L}_2^{-1}F^{(2)}Z_2^{-1}}\,\underbrace{Z_2G^{(2)}\mathcal{L}_2}.$$

Matrix $\mathcal{L}_2$ will affect only rows 3 and 4 of $\mathcal{L}_2^{-1}F$, zeroing out entries $(3,1)$ and $(4,1)$ of F. So, $\mathcal{L}_2 = I + \boldsymbol{l}_2'\boldsymbol{e}_2^T$, where $\boldsymbol{l}_2' = \begin{bmatrix} 0 & 0 & * & * & 0 & 0 \end{bmatrix}^T$, and $\mathcal{L}_2^{-1} = I - \boldsymbol{l}_2'\boldsymbol{e}_2^T$. Thus,

$$\mathcal{L}_2^{-1}F = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & 0 & & & \\ & * & u_3 & 1 & & \\ & + & & u_4 & 1 & \\ & & & & u_5 & 1 \\ & & & & & u_6 \end{bmatrix} \qquad \text{and} \qquad G\mathcal{L}_2 = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & + & 1 & & & \\ & + & l_3 & 1 & & \\ & + & & l_4 & 1 & \\ & & & & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow \mathcal{L}_2^{-1}F \qquad \text{and} \qquad G \longleftarrow G\mathcal{L}_2.$$

Bulge 2×1 in F was chased and second row of F and second row of G are now in final form.

2.c) Define matrix Y_2 to get

$$\underbrace{FY_2^{-1}}\underbrace{Y_2G} = \underbrace{\mathcal{L}_2^{-1}F^{(2)}Z_2^{-1}Y_2^{-1}}\underbrace{Y_2Z_2G^{(2)}\mathcal{L}_2}.$$

Matrix Y_2 affects only rows 3, 4 and 5 of Y_2G, zeroing out entries $(3,2)$, $(4,2)$ and $(5,2)$ of G. So, $Y_2^{-1} = I + \boldsymbol{y}_2'\boldsymbol{e}_2^T$ with $\boldsymbol{y}_2' = \begin{bmatrix}0 & 0 & * & * & * & 0\end{bmatrix}^T$, $Y_2 = I - \boldsymbol{y}_2'\boldsymbol{e}_2^T$ and

$$FY_2^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & 0 & & & \\ & \hat{l}_2 & u_3 & 1 & & \\ & + & & u_4 & 1 & \\ & + & & & u_5 & 1 \\ & & & & & u_6 \end{bmatrix} \quad \text{and} \quad Y_2G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & * & & & \\ & & + & 1 & & \\ & & + & l_4 & 1 & \\ & & & & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow FY_2^{-1} \qquad \text{and} \qquad G \longleftarrow Y_2G$$

and define

$$F^{(3)} := F = \mathcal{L}_2^{-1}F^{(2)}Z_2^{-1}Y_2^{-1}$$
$$G^{(3)} := G = Y_2Z_2G^{(2)}\mathcal{L}_2.$$

The second step is complete. The second column of G is in final form but the second column of F is not yet in final form.

Step 3

3.a) Define Z_3^{-1} and get

$$\underbrace{FZ_3^{-1}}\underbrace{Z_3G} = \underbrace{F^{(3)}Z_3^{-1}}\underbrace{Z_3G^{(3)}}.$$

Matrix Z_3^{-1} will zero out entry $(3,4)$ of F and place 1 into entry $(3,3)$. Z_3 will turn into 1 entry $(3,4)$ of G. We have

$$Z_3^{-1} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & \frac{1}{u_3} & -\frac{1}{u_3} & & \\ & & 0 & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}, \qquad Z_3 = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & u_3 & 1 & & \\ & & 0 & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}$$

and

$$FZ_3^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & 0 & & \\ & + & & u_4 & 1 & \\ & + & & & u_5 & 1 \\ & & & & & u_6 \end{bmatrix}, \qquad Z_3G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & * & 1 & & \\ & & + & 1 & & \\ & & + & l_4 & 1 & \\ & & & & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow FZ_3^{-1} \qquad \text{and} \qquad G \longleftarrow Z_3G.$$

3.b) Define elementary matrix $\mathcal{L}_3$ to obtain

$$\underbrace{\mathcal{L}_3^{-1}F}\underbrace{G\mathcal{L}_3} = \underbrace{\mathcal{L}_3^{-1}F^{(3)}Z_3^{-1}}\underbrace{Z_3G^{(3)}\mathcal{L}_3}.$$

Matrix $\mathcal{L}_3$ will affect only rows 4 and 5 of $\mathcal{L}_3^{-1}F$, zeroing out entries $(4,2)$ and $(5,2)$ of F. So, $\mathcal{L}_3 = I + \boldsymbol{l}_3'\boldsymbol{e}_3^T$, where $\boldsymbol{l}_3' = \begin{bmatrix} 0 & 0 & 0 & * & * & 0 \end{bmatrix}^T$, and $\mathcal{L}_3^{-1} = I - \boldsymbol{l}_3'\boldsymbol{e}_3^T$. Thus,

$$\mathcal{L}_3^{-1}F = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & 0 & & \\ & & * & u_4 & 1 & \\ & & + & & u_5 & 1 \\ & & & & & u_6 \end{bmatrix} \qquad \text{and} \qquad G\mathcal{L}_3 = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & + & 1 & & \\ & & + & l_4 & 1 & \\ & & + & & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow \mathcal{L}_3^{-1}F \qquad \text{and} \qquad G \longleftarrow G\mathcal{L}_3.$$

Third row of F and third row of G are in final form.

3.c) Define elementary matrix Y_3 to produce

$$\underbrace{FY_3^{-1}}\underbrace{Y_3G} = \underbrace{\mathcal{L}_3^{-1}F^{(3)}Z_3^{-1}Y_3^{-1}}\underbrace{Y_3Z_3G^{(3)}\mathcal{L}_3}.$$

Matrix Y_3 affects only rows 4, 5 and 6 of Y_3G, zeroing out entries $(4,3)$, $(5,3)$ and $(6,3)$ of G. So, $Y_3^{-1} = I + \boldsymbol{y}_3'\boldsymbol{e}_3^T$ with $\boldsymbol{y}_3' = \begin{bmatrix} 0 & 0 & 0 & * & * & * \end{bmatrix}^T$, $Y_3 = I - \boldsymbol{y}_3'\boldsymbol{e}_3^T$ and

$$FY_3^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & 0 & & \\ & & \hat{l}_3 & u_4 & 1 & \\ & & + & & u_5 & 1 \\ & & + & & & u_6 \end{bmatrix} \qquad \text{and} \qquad Y_3G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & * & & \\ & & & + & 1 & \\ & & & + & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow FY_3^{-1} \qquad \text{and} \qquad G \longleftarrow Y_3G$$

and define

$$F^{(4)} := F = \mathcal{L}_3^{-1} F^{(3)} Z_3^{-1} Y_3^{-1}$$
$$G^{(4)} := G = Y_3 Z_3 G^{(3)} \mathcal{L}_3.$$

The third step is complete. The third column of G is in final form but the third column of F is not yet in final form.

Step 4

4.a) Define Z_4^{-1} and get

$$\underbrace{F Z_4^{-1}}\,\underbrace{Z_4 G} = \underbrace{F^{(4)} Z_4^{-1}}\,\underbrace{Z_4 G^{(4)}}.$$

Matrix Z_4^{-1} will zero out entry $(4,5)$ of F and place 1 into entry $(4,4)$. Z_4 will turn into 1 entry $(4,5)$ of G. We have

$$Z_4^{-1} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & \frac{1}{u_4} & -\frac{1}{u_4} & \\ & & & 0 & 1 & \\ & & & & & 1 \end{bmatrix}, \qquad Z_4 = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & u_4 & 1 & \\ & & & 0 & 1 & \\ & & & & & 1 \end{bmatrix}$$

and

$$F Z_4^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & 0 & \\ & & + & & u_5 & 1 \\ & & + & & & u_6 \end{bmatrix}, \qquad Z_4 G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & * & 1 & \\ & & & + & 1 & \\ & & & + & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow F Z_4^{-1} \qquad \text{and} \qquad G \longleftarrow Z_4 G.$$

4.b) Apply elementary matrices $\mathcal{L}_4$ and $\mathcal{L}_4^{-1}$,

$$\underbrace{\mathcal{L}_4^{-1}F}\,\underbrace{G\mathcal{L}_4} = \underbrace{\mathcal{L}_4^{-1}F^{(4)}Z_4^{-1}}\,\underbrace{Z_4G^{(4)}\mathcal{L}_4}.$$

Matrix $\mathcal{L}_4$ will affect only rows 5 and 6 of $\mathcal{L}_4^{-1}F$, zeroing out entries $(5,3)$ and $(6,3)$ of F. So, $\mathcal{L}_4 = I + \boldsymbol{l}_4'\boldsymbol{e}_4^T$, where $\boldsymbol{l}_4' = \begin{bmatrix}0 & 0 & 0 & 0 & * & *\end{bmatrix}^T$, and $\mathcal{L}_4^{-1} = I - \boldsymbol{l}_4'\boldsymbol{e}_4^T$. Thus,

$$\mathcal{L}_4^{-1}F = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & 0 & \\ & & & * & u_5 & 1 \\ & & & + & & u_6 \end{bmatrix} \quad \text{and} \quad G\mathcal{L}_4 = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \hat{u}_4 & 1 & \\ & & & + & 1 & \\ & & & + & l_5 & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow \mathcal{L}_4^{-1}F \qquad \text{and} \qquad G \longleftarrow G\mathcal{L}_4.$$

Fourth row of F and fourth row of G are in final form.

4.c) Apply Y_4 and Y_4^{-1},

$$\underbrace{FY_4^{-1}}\,\underbrace{Y_4G} = \underbrace{\mathcal{L}_4^{-1}F^{(4)}Z_4^{-1}Y_4^{-1}}\,\underbrace{Y_4Z_4G^{(4)}\mathcal{L}_4}.$$

Matrix Y_4 affects only rows 5 and 6 of Y_4G, zeroing out entries $(5,4)$ and $(6,4)$ of G. So, $Y_4^{-1} = I + \boldsymbol{y}_4'\boldsymbol{e}_4^T$ with $\boldsymbol{y}_4' = \begin{bmatrix}0 & 0 & 0 & 0 & * & *\end{bmatrix}^T$, $Y_4 = I - \boldsymbol{y}_4'\boldsymbol{e}_4^T$ and

$$FY_4^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & 0 & \\ & & & \hat{l}_4 & u_5 & 1 \\ & & & + & & u_6 \end{bmatrix} \quad \text{and} \quad Y_4G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \hat{u}_4 & 1 & \\ & & & & * & \\ & & & & + & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow FY_4^{-1} \qquad \text{and} \qquad G \longleftarrow Y_4G$$

and define

$$F^{(5)} := F = \mathcal{L}_4^{-1}F^{(4)}Z_4^{-1}Y_4^{-1}$$
$$G^{(5)} := G = Y_4Z_4G^{(4)}\mathcal{L}_4.$$

The fourth step is complete.

Step 5

5.a) Define Z_5^{-1} and get

$$\underbrace{FZ_5^{-1}}\underbrace{Z_5G} = \underbrace{F^{(5)}Z_5^{-1}}\underbrace{Z_5G^{(5)}}.$$

Matrix Z_5^{-1} will zero out entry $(5,6)$ of F and place 1 into entry $(5,5)$. Z_5 will turn into 1 entry $(5,6)$ of G. We have

$$Z_5^{-1} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & \frac{1}{u_5} & -\frac{1}{u_5} \\ & & & & 0 & 1 \end{bmatrix}, \qquad Z_5 = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & u_5 & 1 \\ & & & & 0 & 1 \end{bmatrix}$$

and

$$FZ_5^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & & \\ & & & \hat{l}_4 & 1 & 0 \\ & & & + & & u_6 \end{bmatrix}, \qquad Z_5G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \hat{u}_4 & 1 & \\ & & & & * & 1 \\ & & & & + & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow FZ_5^{-1} \qquad \text{and} \qquad G \longleftarrow Z_5G.$$

5.b) Apply matrices $\mathcal{L}_5$ and $\mathcal{L}_5^{-1}$ to obtain

$$\underbrace{\mathcal{L}_5^{-1}F}\,\underbrace{G\mathcal{L}_5} = \underbrace{\mathcal{L}_5^{-1}F^{(5)}Z_5^{-1}}\,\underbrace{Z_5G^{(5)}\mathcal{L}_5}.$$

Matrix $\mathcal{L}_5$ will affect only rows 6 of $\mathcal{L}_5^{-1}F$, zeroing out entries $(6,4)$ of F. So, $\mathcal{L}_5 = I + \boldsymbol{l}_5'\boldsymbol{e}_5^T$, where $\boldsymbol{l}_5' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & * \end{bmatrix}^T$, and $\mathcal{L}_5^{-1} = I - \boldsymbol{l}_5'\boldsymbol{e}_5^T$. Thus,

$$\mathcal{L}_5^{-1}F = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & & \\ & & & \hat{l}_4 & 1 & 0 \\ & & & & * & u_6 \end{bmatrix} \qquad \text{and} \qquad G\mathcal{L}_5 = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \hat{u}_4 & 1 & \\ & & & & \hat{u}_5 & 1 \\ & & & & + & 1 \end{bmatrix}.$$

Let

$$F \longleftarrow \mathcal{L}_5^{-1}F \qquad \text{and} \qquad G \longleftarrow G\mathcal{L}_5.$$

Fifth row of F and fifth row of G are in final form.

5.c) Apply matrices Y_5 and Y_5^{-1} to obtain

$$\underbrace{FY_5^{-1}}\underbrace{Y_5G} = \underbrace{\mathcal{L}_5^{-1}F^{(5)}Z_5^{-1}Y_5^{-1}}\underbrace{Y_5Z_5G^{(5)}\mathcal{L}_5}.$$

Matrix Y_5 affects only rows 6 of Y_5G, zeroing out entries $(6,5)$ of G. So, $Y_5^{-1} = I + \boldsymbol{y}_5'\boldsymbol{e}_5^T$ with $\boldsymbol{y}_5' = \begin{bmatrix}0 & 0 & 0 & 0 & 0 & *\end{bmatrix}^T$, $Y_5 = I - \boldsymbol{y}_5'\boldsymbol{e}_5^T$ and

$$FY_5^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & & \\ & & & \hat{l}_4 & 1 & 0 \\ & & & & \hat{l}_5 & u_6 \end{bmatrix} \quad \text{and} \quad Y_5G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \hat{u}_4 & 1 & \\ & & & & \hat{u}_5 & 1 \\ & & & & & * \end{bmatrix}.$$

Let

$$F \longleftarrow FY_5^{-1} \qquad \text{and} \qquad G \longleftarrow Y_5G$$

and define

$$F^{(6)} := F = \mathcal{L}_5^{-1}F^{(5)}Z_5^{-1}Y_5^{-1}$$
$$G^{(6)} := G = Y_5Z_5G^{(5)}\mathcal{L}_5.$$

The fifth step is complete and the two bidiagonals are found. There is only need to place 1 in entry $(6,6)$ of F.

Step 6

6.a) Apply Z_6 and Z_6^{-1},

$$\underbrace{FZ_6^{-1}}\underbrace{Z_6G} = \underbrace{F^{(6)}Z_6^{-1}}\underbrace{Z_6G^{(6)}}.$$

Matrix Z_6^{-1} place 1 in entry $(6,6)$ of F. We have

$$Z_6^{-1} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & \frac{1}{u_6} \end{bmatrix}, \qquad Z_6 = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & u_6 \end{bmatrix}$$

and

$$FZ_6^{-1} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & & \\ & & & \hat{l}_4 & 1 & \\ & & & & \hat{l}_5 & 1 \end{bmatrix}, \qquad Z_6G = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \hat{u}_4 & 1 & \\ & & & & \hat{u}_5 & 1 \\ & & & & & * \end{bmatrix}.$$

Let

$$F \longleftarrow FZ_6^{-1} \qquad \text{and} \qquad G \longleftarrow Z_6G.$$

Matrices F and G are already in final form.

6.b) Let $\mathcal{L}_6 = I$ and

$$\underbrace{\mathcal{L}_6^{-1}F}\,\underbrace{G\mathcal{L}_6} = \underbrace{\mathcal{L}_6^{-1}F^{(6)}Z_6^{-1}}\,\underbrace{Z_6G^{(6)}\mathcal{L}_6}.$$

Thus,

$$\mathcal{L}_6^{-1}F = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \hat{l}_3 & 1 & & \\ & & & \hat{l}_4 & 1 & \\ & & & & \hat{l}_5 & 1 \end{bmatrix} \qquad \text{and} \qquad G\mathcal{L}_6 = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \hat{u}_4 & 1 & \\ & & & & \hat{u}_5 & 1 \\ & & & & & \hat{u}_6 \end{bmatrix}.$$

Let

$$F \longleftarrow \mathcal{L}_6^{-1} F \qquad \text{and} \qquad G \longleftarrow G\mathcal{L}_6.$$

6.c) Let $Y_6 = I$ and finally

$$\underbrace{FY_6^{-1}}\,\underbrace{Y_6 G} = \underbrace{\mathcal{L}_6^{-1} F^{(6)} Z_6^{-1} Y_6^{-1}}\,\underbrace{Y_6 Z_6 G^{(6)} \mathcal{L}_6}$$

and

$$F \longleftarrow FY_6^{-1} \qquad \text{and} \qquad G \longleftarrow Y_6 G.$$

In the end we have

$$L_4 = F \;\text{ and }\; U_4 = G. \;\diamond$$

For general matrices L_1 and U_1 as in (5.9), we will have

$$\begin{aligned} L_4 U_4 &= \underbrace{\mathcal{L}_n^{-1} \cdots \mathcal{L}_1^{-1} U_1 Z_1^{-1} Y_1^{-1} \cdots Z_n^{-1} Y_n^{-1}}\,\underbrace{Y_n Z_n \cdots Y_1 Z_1 L_1 \mathcal{L}_1 \cdots \mathcal{L}_n} \\ &= \underbrace{\mathcal{L}_n^{-1} \cdots \mathcal{L}_1^{-1} U_1 X_1^{-1} \cdots X_n^{-1}}\,\underbrace{X_n \cdots X_1 L_1 \mathcal{L}_1 \cdots \mathcal{L}_n} \\ &= \underbrace{\mathcal{L}^{-1} U_1 X^{-1}}\,\underbrace{X L_1 \mathcal{L}} \end{aligned}$$

where

$$\mathcal{L} \equiv \mathcal{L}_1 \cdots \mathcal{L}_n \quad \text{and} \quad X \equiv X_n \cdots X_1$$

with $X_i = Y_i Z_i$, $i = 1, \ldots, n$.

And for a complex shift σ, the triple dqds algorithm will be called tridqds and is given by:

tridqds(σ) :

$F = U_1;\ G = L_1$

$F = FZ_1^{-1};\ G = Z_1 G$

$F = \mathcal{L}_1^{-1} F;\ G = G\mathcal{L}_1$ [form $\mathcal{L}_1$ using (5.10)]

$F = FY_1^{-1};\ G = Y_1 G$

for $i = 2, \ldots, n-3$

$F = FZ_i^{-1};\ G = Z_i G$

$F = \mathcal{L}_i^{-1} F;\ G = G\mathcal{L}_i$

$F = FY_i^{-1};\ G = Y_i G$ [Z_i with one, $\mathcal{L}_i$ with two and Y_i with three off-diagonal entries]

end for

% step n-2

$F = FZ_{n-2}^{-1};\ G = Z_{n-2} G$

$F = \mathcal{L}_{n-2}^{-1} F;\ G = G\mathcal{L}_{n-2}$

$F = FY_{n-2}^{-1};\ G = Y_{n-2} G$ [Y_{n-2} with two off-diagonal entries]

% step n-1

$F = FZ_{n-1}^{-1};\ G = Z_{n-1} G$

$F = \mathcal{L}_{n-1}^{-1} F;\ G = G\mathcal{L}_{n-1}$

$F = FY_{n-1}^{-1};\ G = Y_{n-1} G$ [Y_{n-1} and $\mathcal{L}_{n-1}$ with one off-diagonal entry]

% step n

$\mathcal{L}_n = I;\ Y_n = I$

$F = FZ_n^{-1};\ G = Z_n G$ [Z_n diagonal]

$L_4 = F;\ F_4 = G$

5.3.1 Details of tridqds

In this section we will go into the details of the tridqds algorithm described in the previous section. Consider matrices L_1 and U_1 as defined in (5.9) and let us say that matrices L_4 and U_4 have the same shape with subdiagonal entries $\hat{l}_1, \ldots, \hat{l}_{n-1}$ and diagonal entries $\hat{u}_1, \ldots, \hat{u}_n$, respectively.

For each iteration of tridqds, in the beginning of a minor step i, $i = 2, \ldots, n-2$, the *active* parts of F and G are

$$F = \begin{bmatrix} \ddots & & & & & & \\ \ddots & 1 & & & & & \\ & \hat{l}_{i-1} & u_i & 1 & & & \\ & + & & u_{i+1} & 1 & & \\ & + & & & u_{i+2} & \ddots & \\ & & & & & \ddots & \end{bmatrix} \text{ and } G = \begin{bmatrix} \ddots & \ddots & & & & & \\ & \hat{u}_{i-1} & 1 & & & & \\ & & * & & & & \\ & & + & 1 & & & \\ & & + & l_{i+1} & 1 & & \\ & & & & \ddots & \ddots & \end{bmatrix}. \quad (5.11)$$

So, F starts to be U_1 and is transformed into L_4 and G is given initially as L_1 and is transformed into U_4.

For $i = 2, \ldots, n-3$, a general minor step i consists in what follows.

Typical minor step i

i.a) Define Z_i and let

$$F \longleftarrow F Z_i^{-1} \qquad \text{and} \qquad G \longleftarrow Z_i G.$$

Matrix Z_i^{-1} will zero out entry $(i, i+1)$ of F and place 1 into entry (i, i). Z_i turns into 1 entry $(i, i+1)$ of G.

$$Z_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ & 1 & & & & \\ & & \frac{1}{u_i} & -\frac{1}{u_i} & & \\ & & 0 & 1 & & \\ & & & & 1 & \\ & & & & & \ddots \end{bmatrix}, \qquad Z_i = \begin{bmatrix} \ddots & & & & & \\ & 1 & & & & \\ & & u_i & 1 & & \\ & & 0 & 1 & & \\ & & & & 1 & \\ & & & & & \ddots \end{bmatrix}$$

and

$$FZ_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ \ddots & 1 & & & & \\ & \hat{l}_{i-1} & 1 & 0 & & \\ & + & & u_{i+1} & 1 & \\ & + & & & u_{i+2} & \ddots \\ & & & & & \ddots \end{bmatrix}, \quad Z_iG = \begin{bmatrix} \ddots & \ddots & & & & \\ & \hat{u}_{i-1} & 1 & & & \\ & & * & 1 & & \\ & & + & 1 & & \\ & & + & l_{i+1} & 1 & \\ & & & & \ddots & \ddots \end{bmatrix}.$$

i.b) Define elementary matrix $\mathcal{L}_i$ and let

$$F \longleftarrow \mathcal{L}_i^{-1}F \qquad \text{and} \qquad G \longleftarrow G\mathcal{L}_i.$$

$\mathcal{L}_i$ will affect only rows $i+1$ and $i+2$ of $\mathcal{L}_i^{-1}F$, zeroing out entries $(i+1, i-1)$ and $(i+2, i-1)$ of F;

$\mathcal{L}_i = I + \boldsymbol{l}_i'\boldsymbol{e}_i^T$, where $\boldsymbol{l}_i' = \begin{bmatrix} 0 \dots 0 & * & * & 0 \dots 0 \end{bmatrix}^T$, $\mathcal{L}_i^{-1} = I - \boldsymbol{l}_i'\boldsymbol{e}_i^T$ and

$$\mathcal{L}_i^{-1}F = \begin{bmatrix} \ddots & & & & & \\ \ddots & 1 & & & & \\ & \hat{l}_{i-1} & 1 & & & \\ & & * & u_{i+1} & 1 & \\ & & + & & u_{i+2} & 1 \\ & & & & & \ddots \end{bmatrix}, \quad G\mathcal{L}_i = \begin{bmatrix} \ddots & \ddots & & & & \\ & \hat{u}_i & 1 & & & \\ & + & 1 & & & \\ & + & l_{i+1} & 1 & & \\ & + & & l_{i+2} & 1 & \\ & & & & \ddots & \ddots \end{bmatrix}.$$

i.c) Define elementary matrix Y_i and let

$$F \longleftarrow FY_i^{-1} \qquad \text{and} \qquad G \longleftarrow Y_iG.$$

Y_i affects only rows $i+1$, $i+2$ and $i+3$ of Y_iG, zeroing out entries $(i+1, i)$, $(i+2, i)$ and $(i+3, i)$ of G;

$Y_i^{-1} = I + \boldsymbol{y}_i'\boldsymbol{e}_i^T$ with $\boldsymbol{y}_i' = \begin{bmatrix} 0 \dots 0 & * & * & * & 0 \dots 0 \end{bmatrix}^T$, $Y_i = I - \boldsymbol{y}_i'\boldsymbol{e}_i^T$ and

$$FY_i^{-1} = \begin{bmatrix} \ddots & & & & & & \\ \ddots & 1 & & & & & \\ & \hat{l}_i & u_{i+1} & 1 & & & \\ & + & & u_{i+2} & 1 & & \\ & + & & & u_{i+3} & \ddots & \\ & & & & & \ddots & \end{bmatrix}, \quad Y_iG = \begin{bmatrix} \ddots & \ddots & & & & & \\ & \hat{u}_i & 1 & & & & \\ & & * & & & & \\ & & + & 1 & & & \\ & & + & l_{i+2} & 1 & & \\ & & & & \ddots & \ddots & \end{bmatrix}.$$

Compare with (5.11) to see that we are back to the same pattern from which we began: the bulges have been chased one column to the right and one row to the bottom. Turning to a practical implementation, next we will see how the calculations involved in this process can be organized.

For each minor step i, $i = 2, \ldots, n-3$, consider F and G as in (5.11). Denote the 2×1 bulge in F, indicated with plus signs, by $\begin{bmatrix} x_l & y_l \end{bmatrix}^T$. And denote the entries (i,i), $(i+1,i)$ and $(i+2,i)$ in G, indicated with $*, +, +$, by $\begin{bmatrix} x_r & y_r & z_r \end{bmatrix}^T$. Subscripts l and r derive from "left" and "right", respectively. This way we have

$$F = \begin{bmatrix} \ddots & & & & & & \\ \ddots & 1 & & & & & \\ & \hat{l}_{i-1} & u_i & 1 & & & \\ & x_l & & u_{i+1} & 1 & & \\ & y_l & & & u_{i+2} & \ddots & \\ & & & & & \ddots & \end{bmatrix} \text{ and } G = \begin{bmatrix} \ddots & \ddots & & & & \\ & \hat{u}_{i-1} & 1 & & & \\ & & x_r & & & \\ & & y_r & 1 & & \\ & & z_r & l_{i+1} & 1 & \\ & & & & \ddots & \ddots \end{bmatrix} \tag{5.12}$$

and the typical minor step i can be accomplished using only these auxiliary variables. The details will be shown in the following pages.

Typical minor step i

i.a) • The inverse Z_i^{-1} and Z_i

$$Z_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ & 1 & & & & \\ & & \frac{1}{u_i} & -\frac{1}{u_i} & & \\ & & 0 & 1 & & \\ & & & & 1 & \\ & & & & & \ddots \end{bmatrix}, \qquad Z_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ & 1 & & & & \\ & & u_i & 1 & & \\ & & 0 & 1 & & \\ & & & & 1 & \\ & & & & & \ddots \end{bmatrix}$$

• The effect of Z_i^{-1}

$$FZ_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ \ddots & 1 & & & & \\ & \hat{l}_{i-1} & 1 & 0 & & \\ & x_l & & u_{i+1} & 1 & \\ & y_l & & & u_{i+2} & \ddots \\ & & & & & \ddots \end{bmatrix}$$

• The effect of Z_i

$$Z_iG = \begin{bmatrix} \ddots & \ddots & & & & \\ & \hat{u}_{i-1} & 1 & & & \\ & & x_r & 1 & & \\ & & y_r & 1 & & \\ & & z_r & l_{i+1} & 1 & \\ & & & & \ddots & \ddots \end{bmatrix}$$

where

$$x_r \longleftarrow x_r * u_i + y_r$$

i.b) • The inverse $\mathcal{L}_i^{-1}$

$$\mathcal{L}_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & x_l & 1 & & \\ & & y_l & & 1 & \\ & & & & & \ddots \end{bmatrix}$$

where

$$x_l \longleftarrow -x_l/\hat{l}_{i-1}$$
$$y_l \longleftarrow -y_l/\hat{l}_{i-1}$$

• The effect of $\mathcal{L}_i^{-1}$

$$\mathcal{L}_i^{-1}F = \begin{bmatrix} \ddots & & & & & \\ \ddots & 1 & & & & \\ & \hat{l}_{i-1} & 1 & & & \\ & & x_l & u_{i+1} & 1 & \\ & & y_l & & u_{i+2} & \ddots \\ & & & & & \ddots \end{bmatrix}$$

• The effect of $\mathcal{L}_i$

$$G\mathcal{L}_i = \begin{bmatrix} \ddots & \ddots & & & & \\ & \hat{u}_i & 1 & & & \\ & x_r & 1 & & & \\ & y_r & l_{i+1} & 1 & & \\ & z_r & & l_{i+2} & 1 & \\ & & & & \ddots & \ddots \end{bmatrix}$$

where

$$\hat{u}_i \longleftarrow x_r - x_l$$
$$x_r \longleftarrow y_r - x_l$$
$$y_r \longleftarrow z_r - y_l - x_l * l_{i+1}$$
$$z_r \longleftarrow -y_l * l_{i+2}$$

i.c) • The inverse Y_i^{-1}

$$Y_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ & 1 & & & & \\ & x_r & 1 & & & \\ & y_r & & 1 & & \\ & z_r & & & 1 & \\ & & & & & \ddots \end{bmatrix}$$

where

$$x_r \longleftarrow x_r/\hat{u}_i$$
$$y_r \longleftarrow y_r/\hat{u}_i$$
$$z_r \longleftarrow z_r/\hat{u}_i$$

• The effect of Y_i^{-1}

$$FY_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ \ddots & 1 & & & & \\ & \hat{l}_i & u_{i+1} & 1 & & \\ & x_l & & u_{i+2} & 1 & \\ & y_l & & & u_{i+3} & \ddots \\ & & & & & \ddots \end{bmatrix}$$

where

$$\hat{l}_i \longleftarrow x_l + y_r + x_r * u_{i+1}$$
$$x_l \longleftarrow y_l + z_r + y_r * u_{i+2}$$
$$y_l \longleftarrow z_r * u_{i+3}$$

- The effect of Y_i

$$Y_iG = \begin{bmatrix} \ddots & \ddots & & & & \\ & \hat{u}_i & 1 & & & \\ & & x_r & & & \\ & & y_r & 1 & & \\ & & z_r & l_{i+2} & 1 & \\ & & & & \ddots & \ddots \end{bmatrix}$$

where

$$\begin{aligned} x_r &\longleftarrow 1 - x_r \\ y_r &\longleftarrow l_{i+1} - y_r \\ z_r &\longleftarrow -z_r \end{aligned}$$

Now the details of the last three minor steps.

Minor step n-2

[n-2].a) • The inverse Z_{n-2}^{-1} and Z_{n-2}

$$Z_{n-2}^{-1} = \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & \frac{1}{u_{n-2}} & -\frac{1}{u_{n-2}} \\ & & 0 & 1 \\ & & & & 1 \end{bmatrix}, \qquad Z_{n-2} = \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & u_{n-2} & 1 \\ & & 0 & 1 \\ & & & & 1 \end{bmatrix}$$

- The effect of Z_{n-2}^{-1}

$$FZ_{n-2}^{-1} = \begin{bmatrix} \ddots & & & & \\ \ddots & 1 & & & \\ & \hat{l}_{n-3} & 1 & 0 & \\ & x_l & & u_{n-1} & 1 \\ & y_l & & & u_n \end{bmatrix}$$

- The effect of Z_{n-2}

$$Z_{n-2}G = \begin{bmatrix} \ddots & \ddots & & & \\ & \hat{u}_{n-3} & 1 & & \\ & & x_r & 1 & \\ & & y_r & 1 & \\ & & z_r & l_{n-1} & 1 \end{bmatrix}$$

where

$$x_r \longleftarrow x_r * u_{n-2} + y_r$$

[n-2].b)

- The inverse $\mathcal{L}_{n-2}^{-1}$

$$\mathcal{L}_{n-2}^{-1} = \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & 1 & \\ & & x_l & 1 \\ & & y_l & & 1 \end{bmatrix}$$

where

$$x_l \longleftarrow -x_l/\hat{l}_{n-3}$$
$$y_l \longleftarrow -y_l/\hat{l}_{n-3}$$

- The effect of $\mathcal{L}_{n-2}^{-1}$

$$\mathcal{L}_{n-2}^{-1}F = \begin{bmatrix} \ddots & & & & \\ \ddots & 1 & & & \\ & \hat{l}_{n-3} & 1 & & \\ & & x_l & u_{n-1} & 1 \\ & & y_l & & u_n \end{bmatrix}$$

- The effect of $\mathcal{L}_{n-2}$

$$G\mathcal{L}_{n-2} = \begin{bmatrix} \ddots & \ddots & & & \\ & \hat{u}_{n-3} & 1 & & \\ & & \hat{u}_{n-2} & 1 & \\ & & x_r & 1 & \\ & & y_r & l_{n-1} & 1 \end{bmatrix}$$

where

$$\begin{aligned} \hat{u}_{n-2} &\longleftarrow x_r - x_l \\ x_r &\longleftarrow y_r - x_l \\ y_r &\longleftarrow z_r - y_l - x_l * l_{n-1} \end{aligned}$$

[n-2].c) - The inverse Y_{n-2}^{-1}

$$Y_{n-2}^{-1} = \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & 1 & \\ & & x_r & 1 \\ & & y_r & & 1 \end{bmatrix}$$

where

$$\begin{aligned} x_r &\longleftarrow x_r / \hat{u}_{n-2} \\ y_r &\longleftarrow y_r / \hat{u}_{n-2} \end{aligned}$$

- The effect of Y_{n-2}^{-1}

$$FY_{n-2}^{-1} = \begin{bmatrix} \ddots & \ddots & & & \\ & \hat{l}_{n-3} & 1 & & \\ & & \hat{l}_{n-2} & u_{n-1} & 1 \\ & & x_l & & u_n \end{bmatrix}$$

where

$$\hat{l}_{n-2} \longleftarrow x_l + y_r + x_r * u_{n-1}$$

$$x_l \longleftarrow y_l + y_r * u_n$$

- The effect of Y_{n-2}

$$Y_{n-2}G = \begin{bmatrix} \ddots & \ddots & & & \\ & \hat{u}_{n-3} & 1 & & \\ & & \hat{u}_{n-2} & 1 & \\ & & & x_r & \\ & & & y_r & 1 \end{bmatrix}$$

where

$$x_r \longleftarrow 1 - x_r$$

$$y_r \longleftarrow l_{n-1} - y_r$$

Minor step n-1

[n-1].a)

- The inverse Z_{n-1}^{-1} and Z_{n-1}

$$Z_{n-1}^{-1} = \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & \frac{1}{u_{n-1}} & -\frac{1}{u_{n-1}} \\ & & 0 & 1 \end{bmatrix}, \qquad Z_{n-1} = \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & u_{n-1} & 1 \\ & & 0 & 1 \end{bmatrix}$$

- The effect of Z_{n-1}^{-1}

$$FZ_{n-1}^{-1} = \begin{bmatrix} \ddots & & & \\ \ddots & 1 & & \\ & \hat{l}_{n-2} & 1 & 0 \\ & x_l & & u_n \end{bmatrix}$$

- The effect of Z_{n-1}

$$Z_{n-1}G = \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & x_r & 1 \\ & & y_r & 1 \end{bmatrix}$$

where

$$x_r \longleftarrow x_r * u_{n-1} + y_r$$

[n-1].b)

- The inverse $\mathcal{L}_{n-1}^{-1}$

$$\mathcal{L}_{n-1}^{-1} = \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & 1 & \\ & & x_l & 1 \end{bmatrix}$$

where

$$x_l \longleftarrow -x_l/\hat{l}_{n-2}$$

- The effect of $\mathcal{L}_{n-1}^{-1}$

$$\mathcal{L}_{n-1}^{-1}F = \begin{bmatrix} \ddots & & & \\ \ddots & 1 & & \\ & \hat{l}_{n-2} & 1 & \\ & & x_l & u_n \end{bmatrix}$$

- The effect of $\mathcal{L}_{n-1}$

$$G\mathcal{L}_{n-1} = \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & \hat{u}_{n-1} & 1 \\ & & x_r & 1 \end{bmatrix}$$

where

$$\hat{u}_{n-1} \longleftarrow x_r - x_l$$

$$x_r \longleftarrow y_r - x_l$$

[n-1].c)

- The inverse Y_{n-1}^{-1}

$$Y_{n-1}^{-1} = \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & 1 & \\ & & x_r & 1 \end{bmatrix}$$

where

$$x_r \longleftarrow x_r / \hat{u}_{n-1}$$

- The effect of Y_{n-1}^{-1}

$$FY_{n-1}^{-1} = \begin{bmatrix} \ddots & & & \\ \ddots & 1 & & \\ & \hat{l}_{n-2} & 1 & \\ & & \hat{l}_{n-1} & u_n \end{bmatrix}$$

where

$$\hat{l}_{n-1} \longleftarrow x_l + x_r * u_n$$

- The effect of Y_{n-1}

$$Y_{n-1}G = \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & \hat{u}_{n-1} & 1 \\ & & & x_r \end{bmatrix}$$

where

$$x_r \longleftarrow 1 - x_r$$

Minor step n

n.a) • The inverse Z_n^{-1} and Z_n

$$Z_n^{-1} = \begin{bmatrix} \ddots & & & \\ & \ddots & & \\ & & 1 & \\ & & & \frac{1}{u_n} \end{bmatrix}, \qquad Z_n = \begin{bmatrix} \ddots & & & \\ & \ddots & & \\ & & 1 & \\ & & & u_n \end{bmatrix}$$

• The effect of Z_n^{-1}

$$FZ_n^{-1} = \begin{bmatrix} \ddots & & & \\ \ddots & 1 & & \\ & \hat{l}_{n-2} & 1 & \\ & & \hat{l}_{n-1} & 1 \end{bmatrix}$$

• The effect of Z_n

$$Z_n G = \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & \hat{u}_{n-1} & 1 \\ & & & x_r \end{bmatrix}$$

where

$$x_r \longleftarrow x_r * u_n$$

n.b) • The inverse $\mathcal{L}_n^{-1} = I$

• The effect of $\mathcal{L}_n$

$$G\mathcal{L}_n = \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & \hat{u}_{n-1} & 1 \\ & & & \hat{u}_n \end{bmatrix}$$

where

$$\hat{u}_n \longleftarrow x_r$$

n.c) • The inverse $Y_n^{-1} = I$

Now the last detail that is missing is how to initialize auxiliary variables x_l, y_l and x_r, y_r and z_r, that is, how to organize the first step of the tridqds algorithm.

Minor step 1

Initially we let

$$F = \begin{bmatrix} u_1 & 1 & & & \\ & u_2 & 1 & & \\ & & u_3 & \ddots & \\ & & & \ddots & 1 \\ & & & & u_n \end{bmatrix}, \qquad G = \begin{bmatrix} x_r & & & & \\ y_r & 1 & & & \\ z_r & l_2 & 1 & & \\ & & \ddots & \ddots & \\ & & & l_{n-1} & 1 \end{bmatrix}$$

where

$$\begin{aligned} x_r &\longleftarrow 1 \\ y_r &\longleftarrow l_1 \\ z_r &\longleftarrow 0 \end{aligned}$$

1.a) • The inverse Z_1^{-1} and Z_1

$$Z_1^{-1} = \begin{bmatrix} \frac{1}{u_1} & -\frac{1}{u_1} & & & \\ 0 & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}, \qquad Z_1 = \begin{bmatrix} u_1 & 1 & & & \\ 0 & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}$$

- The effect of Z_1^{-1}

$$FZ_1^{-1} = \begin{bmatrix} 1 & 0 & & & \\ & u_2 & 1 & & \\ & & u_3 & \ddots & \\ & & & \ddots & 1 \\ & & & & u_n \end{bmatrix}$$

- The effect of Z_1

$$Z_1G = \begin{bmatrix} x_r & 1 & & & \\ y_r & 1 & & & \\ z_r & l_2 & 1 & & \\ & & \ddots & \ddots & \\ & & & l_{n-1} & 1 \end{bmatrix}$$

where

$$x_r \longleftarrow x_r * u_1 + y_r$$

1.b) - The inverse $\mathcal{L}_1^{-1}$

$$\mathcal{L}_1^{-1} = \begin{bmatrix} 1 & & & & \\ x_l & 1 & & & \\ y_l & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}$$

where

$$\begin{aligned} x_l &\longleftarrow (u_1 + l_1)^2 + u_2 l_1 - 2(\Re\sigma)(u_1 + l_1) + |\sigma|^2 \\ y_l &\longleftarrow -u_2 l_1 u_3 l_2 / x_l \\ x_l &\longleftarrow -u_2 l_1 \big(u_1 + l_1 + u_2 + l_2 - 2(\Re\sigma)\big) / x_l \end{aligned}$$

- The effect of $\mathcal{L}_1^{-1}$

$$\mathcal{L}_1^{-1}F = \begin{bmatrix} 1 & 0 & & & \\ x_l & u_2 & 1 & & \\ y_l & & u_3 & \ddots & \\ & & & \ddots & 1 \\ & & & & u_n \end{bmatrix}$$

- The effect of $\mathcal{L}_1$

$$G\mathcal{L}_1 = \begin{bmatrix} \hat{u}_1 & 1 & & & \\ x_r & 1 & & & \\ y_r & l_2 & 1 & & \\ z_r & & \ddots & \ddots & \\ & & & l_{n-1} & 1 \end{bmatrix}$$

where

$$\begin{aligned} \hat{u}_1 &\longleftarrow x_r - x_l \\ x_r &\longleftarrow y_r - x_l \\ y_r &\longleftarrow z_r - y_l - x_l * l_2 \\ z_r &\longleftarrow -y_l * l_3 \end{aligned}$$

1.c) - The inverse Y_1^{-1}

$$Y_1^{-1} = \begin{bmatrix} 1 & & & & \\ x_r & 1 & & & \\ y_r & & 1 & & \\ z_r & & & \ddots & \\ & & & & 1 \end{bmatrix}$$

where

$$x_r \longleftarrow x_r/\hat{u}_1$$
$$y_r \longleftarrow y_r/\hat{u}_1$$
$$z_r \longleftarrow z_r/\hat{u}_1$$

- The effect of Y_1^{-1}

$$FY_1^{-1} = \begin{bmatrix} 1 & 0 & & & \\ \hat{l}_1 & u_2 & 1 & & \\ x_l & & u_3 & \ddots & \\ y_l & & & \ddots & 1 \\ & & & & u_n \end{bmatrix}$$

where

$$\hat{l}_1 \longleftarrow x_l + y_r + x_r * u_2$$
$$x_l \longleftarrow y_l + z_r + y_r * u_3$$
$$y_l \longleftarrow z_r * u_4$$

- The effect of Y_1

$$Y_1G = \begin{bmatrix} \hat{u}_1 & 1 & & & \\ & x_r & & & \\ & y_r & 1 & & \\ & z_r & l_3 & 1 & \\ & & & \ddots & \ddots \\ & & & & l_{n-1} & 1 \end{bmatrix}$$

where

$$x_r \longleftarrow 1 - x_r$$
$$y_r \longleftarrow l_2 - y_r$$
$$z_r \longleftarrow -z_r$$

It can be verified that minor step 1 could be incorporated in the inner loop, $i = 2, \ldots, n-3$, but that would demand an extra auxiliary variable and it is not worthwhile. The final version of tridqds algorithm is presented in the following pages (comments are included to make it easier to relate to the matrix formulation in page 140).

tridqds(σ) :

$x_r = 1; \;\; y_r = l_1; \;\; z_r = 0$

% the effect of Z_1

$x_r = x_r * u_1 + y_r$

% the inverse $\mathcal{L}_1^{-1}$

$x_l = (u_1 + l_1)^2 + u_2 l_1 - 2(\Re\sigma)(u_1 + l_1) + |\sigma|^2$

$y_l = -u_2 l_1 u_3 l_2 / x_l$

$x_l = -u_2 l_1 (u_1 + l_1 + u_2 + l_2 - 2(\Re\sigma))/x_l$

% the effect of $\mathcal{L}_1$

$\hat{u}_1 = x_r - x_l;$

$x_r = y_r - x_l; \;\; y_r = z_r - y_l - x_l * l_2; \;\; z_r = -y_l * l_3$

% the inverse Y_1^{-1}

$x_r = x_r/\hat{u}_1; \;\; y_r = y_r/\hat{u}_1; \;\; z_r = z_r/\hat{u}_1$

% the effect of Y_1^{-1}

$\hat{l}_1 = x_l + y_r + x_r * u_2$

$x_l = y_l + z_r + y_r * u_3; \;\; y_l = z_r * u_4$

% the effect of Y_1

$x_r = 1 - x_r; \;\; y_r = l_2 - y_r; \;\; z_r = -z_r$

for $i = 2, \ldots, n-3$

% the effect of Z_i

$x_r = x_r * u_i + y_r$

% the inverse $\mathcal{L}_i^{-1}$

$x_l = -x_l/\hat{l}_{i-1}; \;\; y_l = -y_l/\hat{l}_{i-1};$

% the effect of $\mathcal{L}_i$

$\hat{u}_i = x_r - x_l;$

$x_r = y_r - x_l;\ \ y_r = z_r - y_l - x_l * l_{i+1};\ \ z_r = -y_l * l_{i+2}$

% the inverse Y_i^{-1}

$x_r = x_r/\hat{u}_i;\ \ y_r = y_r/\hat{u}_i;\ \ z_r = z_r/\hat{u}_i$

% the effect of Y_i^{-1}

$\hat{l}_i = x_l + y_r + x_r * u_{i+1}$

$x_l = y_l + z_r + y_r * u_{i+2};\ \ y_l = z_r * u_{i+3}$

% the effect of Y_i

$x_r = 1 - x_r;\ \ y_r = l_{i+1} - y_r;\ \ z_r = -z_r$

end for

% step n-2

% the effect of Z_{n-2}

$x_r = x_r * u_{n-2} + y_r$

% the inverse $\mathcal{L}_{n-2}^{-1}$

$x_l = -x_l/\hat{l}_{n-3};\ \ y_l = -y_l/\hat{l}_{n-3};$

% the effect of $\mathcal{L}_{n-2}$

$\hat{u}_{n-2} = x_r - x_l;$

$x_r = y_r - x_l;\ \ y_r = z_r - y_l - x_l * l_{n-1}$

% the inverse Y_{n-2}^{-1}

$x_r = x_r/\hat{u}_{n-2};\ \ y_r = y_r/\hat{u}_{n-2}$

% the effect of Y_{n-2}^{-1}

$\hat{l}_{n-2} = x_l + y_r + x_r * u_{n-1}$

$x_l = y_l + y_r * u_n$

% the effect of Y_{n-2}

$x_r = 1 - x_r;\ \ y_r = l_{n-1} - y_r$

% step n-1

% the effect of Z_{n-1}

$x_r = x_r * u_{n-1} + y_r$

% the inverse $\mathcal{L}_{n-1}^{-1}$

$x_l = -x_l/\hat{l}_{n-2}$

% the effect of $\mathcal{L}_{n-1}$

$\hat{u}_{n-1} = x_r - x_l;$

$x_r = y_r - x_l$

% the inverse Y_{n-1}^{-1}

$x_r = x_r/\hat{u}_{n-1}$

% the effect of Y_{n-1}^{-1}

$\hat{l}_{n-1} = x_l + x_r * u_n$

% the effect of Y_{n-1}

$x_r = 1 - x_r$

% step n

% the effect of Z_n

$x_r = x_r * u_n$

% the inverse $\mathcal{L}_n^{-1} = I$

% the effect of $\mathcal{L}_n$

$\hat{u}_n = x_r;$

% the inverse $Y_n^{-1} = I$

5.3.2 Operation count for tridqds

In this section we will see how three steps of simple dqds algorithm compares with one step of tridqds in what respects to the number of floating point operations required. In order to make easier this comparison we first remember dqds algorithm from section 3.5.2 (page 75).

$$
\begin{aligned}
\mathrm{dqds}(\sigma): \quad & d_1 = u_1 - \sigma \\
& \textbf{for } i = 1, \ldots, n-1 \\
& \quad \hat{u}_i = d_i + l_i \\
& \quad \hat{l}_i = l_i(u_{i+1}/\hat{u}_i) \\
& \quad d_{i+1} = d_i(u_{i+1}/\hat{u}_i) - \sigma \\
& \textbf{end for} \\
& \hat{u}_n = d_n.
\end{aligned}
\tag{5.13}
$$

In practice, each d_{i+1} may be written over its predecessor in a single variable d and, if the common subexpression $u_{i+1}/\hat{u}_i$ is recognized, then only one division is needed if we use an auxiliary variable.

Table 5.1 below shows that the operation count of one step of tridqds is comparable to three steps of dqds (table expresses only the number of floating point operations in the inner loops).

	tridqds	3 dqds steps
Divisions	5	3
Multiplications	6	6
Additions	5	3
Subtractions	6	3
Assignments	16	12
Auxiliary variables	5	2

Table 5.1: Operation count of tridqds and 3 dqds steps

But to make three steps of dqds equivalent to tridqds we have to consider dqds in complex arithmetic and the total cost is raised by a factor of about 4. Thus, in complex arithmetic, three steps of dqds are much more expensive than one step of tridqds.

Chapter 6

New version of triple dqds

In this chapter we will first describe the connection of a generalized Gram-Schmidt process with the LU factorization and explain how dqds may be derived from this relation independently of qd. Then, we show that quantities d_i in dqds provide useful information about the diagonal of $(UL)^{-1}$ and can be incorporated in a shift strategy. For the case of complex eigenvalues we try to generalize these results and exhibit similar ones for tridqds.

A new version of the triple dqds algorithm follows from several results that we establish for the quantities involved in tridqds. This version will be called 3dqds and is more elegant and more efficient than tridqds. Numerical results of an implementation of 3dqds will be shown in the next chapter.

6.1 Gram-Schmidt factors

Finding the LDU factorization of any product BC is equivalent to applying a generalized Gram-Schmidt process to the rows of B and to the columns of C so that $B = LP^*$, $C = QU$, and P^*Q is diagonal. See Parlett [40]. When this Gram-Schmidt process is applied to DUL in an efficient manner one obtains the variant dqd of qd algorithm that, as we have discussed earlier, requires a little more arithmetic effort than qd itself. So, in this section we show that dqd could have been discovered independently of qd.

The Gram-Schmidt process is the standard way of producing an orthonormal set of vectors $\left\{\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_k\right\}$ from a linearly independent set $\left\{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_k\right\}$. The defining property is that $\text{span}(\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_j) = \text{span}(\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_j)$, for each $j = 1, \ldots, k$. The matrix formulation of this process is the QR factorization

$$F = QR$$

where $F = \begin{bmatrix} \boldsymbol{f}_1 & \boldsymbol{f}_2 & \ldots & \boldsymbol{f}_k \end{bmatrix}$, $Q = \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \ldots & \boldsymbol{q}_k \end{bmatrix}$ and R is a $k \times k$ upper triangular matrix.

The generalization of this process to a pair of vector sets $\left\{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_k\right\}$ and $\left\{\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_k\right\}$ is so natural that there can be little objection to keeping the name Gram-Schmidt. The context determines immediately whether one or two sets of vectors are involved. Recall that F^* denotes the conjugate transpose of F.

Theorem 6.1.1 *Let F and G be complex $n \times k$ matrices, $n \geq k$, such that G^*F permits triangular factorization*

$$G^*F = \tilde{L}\tilde{D}\tilde{R},$$

where $\tilde{L}$ and $\tilde{R}$ are unit triangular, lower and upper, respectively, and $\tilde{D}$ is diagonal. Then there exist unique $n \times k$ matrices $\tilde{Q}$ and $\tilde{P}$ such that

$$F = \tilde{Q}\tilde{R}, \qquad G = \tilde{P}\tilde{L}^*, \qquad \tilde{P}^*\tilde{Q} = \tilde{D}.$$

This result can be proved by construction.

We will use X_{ij} to denote the (i, j) element of matrix X.

Note 6.1.1 *When G and F are real and $G = F$, the traditional* QR *factorization is recovered with an unconventional normalization. Suppose that F^TF permits an LU factorization. We must have*

$$F^TF = \tilde{L}\tilde{D}\tilde{L}^T$$

and then, from

$$F = \tilde{Q}\tilde{L}^T = \tilde{P}\tilde{L}^T,$$

we get $\tilde{Q} = \tilde{P}$*, since* $\tilde{L}^T$ *is invertible. Thus*

$$\tilde{P}^T\tilde{Q} = \tilde{Q}^T\tilde{Q} = \tilde{D}.$$

Now,

$$\begin{aligned}\tilde{Q}^T\tilde{Q} = \tilde{D} &\Leftrightarrow \left(\tilde{D}^{-\frac{1}{2}}\right)^T\tilde{Q}^T\tilde{Q}\tilde{D}^{-\frac{1}{2}} = \left(\tilde{D}^{-\frac{1}{2}}\right)^T\tilde{D}\tilde{D}^{-\frac{1}{2}}\\ &\Leftrightarrow \left[\tilde{Q}\tilde{D}^{-\frac{1}{2}}\right]^T\tilde{Q}\tilde{D}^{-\frac{1}{2}} = \left(\tilde{D}^{-\frac{1}{2}}\right)^T\tilde{D}^{\frac{1}{2}}\\ &\Leftrightarrow Q^TQ = I\end{aligned}$$

where $Q \equiv \tilde{Q}\tilde{D}^{-\frac{1}{2}}$*. We have*

$$F = \tilde{Q}\tilde{R} = \tilde{Q}\tilde{D}^{-\frac{1}{2}}\tilde{D}^{\frac{1}{2}}\tilde{R} = QR$$

$R \equiv \tilde{D}^{\frac{1}{2}}\tilde{R} = \tilde{D}^{\frac{1}{2}}\tilde{L}^T$.

Observe that we have assumed that $\tilde{D}$ *is invertible.*

Note 6.1.2 *In practice, when* $n = k$ *and* $\tilde{D}$ *is invertible one can omit* $\tilde{Q}$ *and write*

$$F = (\tilde{P}^*)^{-1}(\tilde{D}\tilde{R}),\ \ G = \tilde{P}\tilde{L}^*$$

and still call it the Gram-Schmidt factorization. The important feature is the uniqueness of $\tilde{Q}$ *and* $\tilde{P}$*.*

The Gram-Schmidt factorization leads directly to the differential qd algorithms. Let us show how.

6.2 Derivation of dqds from Gram-Schmidt

Consider

$$L = \begin{bmatrix} 1 & & & & & \\ l_1 & 1 & & & & \\ & l_2 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & l_{n-2} & 1 & \\ & & & & l_{n-1} & 1 \end{bmatrix}, \quad U = \begin{bmatrix} u_1 & 1 & & & & \\ & u_2 & 1 & & & \\ & & u_3 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & u_{n-1} & 1 \\ & & & & & u_n \end{bmatrix}, \tag{6.1}$$

$$\hat{L} = \begin{bmatrix} 1 & & & & & \\ \hat{l}_1 & 1 & & & & \\ & \hat{l}_2 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & \hat{l}_{n-2} & 1 & \\ & & & & \hat{l}_{n-1} & 1 \end{bmatrix}, \quad \hat{U} = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ & \hat{u}_2 & 1 & & & \\ & & \hat{u}_3 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & \hat{u}_{n-1} & 1 \\ & & & & & \hat{u}_n \end{bmatrix} \tag{6.2}$$

and dqds algorithm shown in (5.13), page 161, with $\sigma = 0$, that is, dqd algorithm.

Corollary 6.2.1 *Let L and U be $n \times n$ bidiagonal matrices as given above. Suppose UL permits factorization*

$$UL = \hat{L}\hat{D}\hat{R} = \hat{L}\hat{U}, \tag{6.3}$$

where $\hat{L}$ and $\hat{R}$ are unit bidiagonal, lower and upper, respectively, and $\hat{D}$ is diagonal. Then there exist unique matrices $\tilde{P}$ and $\tilde{Q}$ such that

$$U = \hat{L}\tilde{P}^T, \quad L = \tilde{Q}\hat{R}, \quad \tilde{P}^T\tilde{Q} = \hat{D}.$$

Proof. In previous theorem just take $F = L$ and $G^* = U$ and use the fact that U and L are real. □

In words, apply Gram-Schmidt to the columns of L and to the rows of U (or columns of U^T), in natural order, to obtain $\hat{L}$ and $\hat{R}$. Then note that $\hat{U} = \hat{D}\hat{R}$. In fact, after Gram-Schmidt, we have

$$UL = \hat{L}\tilde{P}^T\tilde{Q}\hat{R} = \hat{L}\hat{D}\hat{R} = \hat{L}\hat{U}.$$

We certainly have $\tilde{P}^T\tilde{Q} = \hat{D}$ because L and U are invertible and the two QR factorizations are unique.

Also notice that if $\hat{U}_{ii} = 0$, $i < n$, then UL does not permit triangular factorization. However, the theorem allows $\hat{U}_{nn} = 0$. When $\hat{U}_{nn} \neq 0$ then $\hat{U}$ is invertible and so is $\hat{D}$. In this case matrices $\tilde{Q}$ and $\tilde{P}$ are also invertible and, since $\tilde{Q} = \tilde{P}^{-T}\hat{D}$ and $\tilde{Q}^{-1}L = \hat{D}^{-1}\tilde{P}^TL = \hat{R}$, we can rewrite the factorization as

$$UL = U\tilde{P}^{-T}\tilde{P}^TL = \left(U\tilde{P}^{-T}\right)\left[\hat{D}\left(\hat{D}^{-1}\tilde{P}^TL\right)\right] = \hat{L}\hat{D}\hat{R} = \hat{L}\hat{U}.$$

If we define $K := \tilde{P}^T$ we can write

$$\underbrace{UK^{-1}}\underbrace{KL} = \hat{L}\hat{U}. \tag{6.4}$$

The matrix K is hidden when we just write $UL = \hat{L}\hat{U}$. However the identification of K with the Gram-Schmidt process goes only half way in the derivation of the `dqd` algorithm. The nature of Gram-Schmidt process shows that $\tilde{P}$ and $\tilde{Q}$ are upper Hessenberg matrices. We are going to show that they may be written as the product of n simple matrices that are non-orthogonal analogues of plane rotations. That means that L may be changed into $\hat{U}$ and U into $\hat{L}$ by a sequence of simple transformations and neither K, $\tilde{Q}$ nor $\tilde{P}$ need appear explicitly.

A *plane transformer* in plane (i, j), $i \neq j$, is an identity matrix except for the entries (i, i), (i, j), (j, i) and (j, j). The 2×2 submatrix they define must be invertible.

Let us describe the first minor step in mapping $U \longrightarrow \hat{L}$, $L \longrightarrow \hat{U}$. We seek an invertible matrix

$$X = \begin{bmatrix} x & z & & & & \\ -y & w & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix}$$

such that

$$UX^{-1} = \begin{bmatrix} 1 & 0 & & & & \\ \hat{l}_1 & * & 1 & & & \\ & & u_3 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & u_{n-1} & 1 \\ & & & & & u_n \end{bmatrix} \quad \text{and} \quad XL = \begin{bmatrix} \hat{u}_1 & 1 & & & & \\ 0 & 1 & & & & \\ & l_2 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & l_{n-2} & 1 & \\ & & & & l_{n-1} & 1 \end{bmatrix}$$

where * may be anything. Note that

$$X^{-1} = \begin{bmatrix} \frac{w}{det} & -\frac{z}{det} & & & \\ \frac{y}{det} & \frac{x}{det} & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}$$

where $det = xw + yz$. Thus, we are left with the following two equations

$$\begin{bmatrix} x & z \\ -y & w \end{bmatrix} \begin{bmatrix} 1 & 0 \\ l_1 & 1 \end{bmatrix} = \begin{bmatrix} \hat{u}_1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} u_1 & 1 \\ 0 & u_2 \end{bmatrix} \begin{bmatrix} w & -z \\ y & x \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \hat{l}_1 & * \end{bmatrix} .det$$

We must have

$$x + zl_1 = \hat{u}_1 \tag{6.5}$$

$$z = 1 \tag{6.6}$$

$$-y + wl_1 = 0 \tag{6.7}$$

$$w = 1 \tag{6.8}$$

$$u_1 w + y = det \tag{6.9}$$

$$-u_1 z + x = 0 \tag{6.10}$$

$$u_2 y = \hat{l}_1.det \tag{6.11}$$

$$u_2 x = *.det. \tag{6.12}$$

So, $z = w = 1$, equation (6.7) shows that $y = l_1$ and equation (6.10) gives $x = u_1$. Thus $det = u_1 + l_1$ which verifies (6.9). From (6.5),

$$\hat{u}_1 = u_1 + l_1$$

and from (6.11) we learn that

$$u_2 y = u_2 l_1 = \hat{l}_1 det = \hat{l}_1 \hat{u}_1.$$

Finally, and of most interest, from (6.12),

$$* = u_2 x/det = u_2 u_1/det = u_2 u_1/\hat{u}_1.$$

This gives the intermediate quantity u_2 in `dqd` and we see it here as something that gets carried to the next minor step.

If we write $d_1 = u_1$ we obtain the start of the inner loop of `dqd`:

$$\begin{aligned} \hat{u}_1 &= d_1 + l_1 \\ \hat{l}_1 &= l_1(u_2/\hat{u}_1) \\ d_2 &= d_1(u_2/\hat{u}_1). \end{aligned}$$

The typical minor step is similar. Below we show matrices part way through the transformation $U \to \hat{L}$, $L \to \hat{U}$. At minor step i, $i = 1, \ldots, n-1$, we have

$$X_i = \begin{bmatrix} \ddots & & & & & & \\ & 1 & & & & & \\ & & d_i & 1 & & & \\ & & -l_i & 1 & & & \\ & & & & 1 & & \\ & & & & & \ddots & \end{bmatrix}, \quad X_i^{-1} = \begin{bmatrix} \ddots & & & & & & \\ & 1 & & & & & \\ & & 1/\hat{u}_i & -1/\hat{u}_i & & & \\ & & l_i/\hat{u}_i & d_i/\hat{u}_i & & & \\ & & & & & 1 & \\ & & & & & & \ddots \end{bmatrix}. \tag{6.13}$$

Consider that after completion of step i we let

$$U \longleftarrow UX_i^{-1} \text{ and } L \longleftarrow X_i L.$$

So, at the end of minor step i we have

$$UX_i^{-1} = \begin{bmatrix} \ddots & & & & & & \\ \ddots & 1 & 0 & & & & \\ & \hat{l}_i & d_{i+1} & 1 & & & \\ & & & u_{i+2} & 1 & & \\ & & & & u_{i+3} & \ddots & \\ & & & & & \ddots & \end{bmatrix}, \quad X_i L = \begin{bmatrix} \ddots & \ddots & & & & & \\ & \hat{u}_i & 1 & & & & \\ & 0 & 1 & & & & \\ & & l_{i+1} & 1 & & & \\ & & & l_{i+2} & 1 & & \\ & & & & \ddots & \ddots & \end{bmatrix}.$$

At the end of minor step $n-1$,

$$UX_{n-1}^{-1} = \begin{bmatrix} \ddots & & & \\ \ddots & 1 & & \\ & \hat{l}_{n-2} & 1 & 0 \\ & & \hat{l}_{n-1} & d_n \end{bmatrix} \quad \text{and} \quad X_{n-1}L = \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & \hat{u}_{n-1} & 1 \\ & & 0 & 1 \end{bmatrix}.$$

If $d_n \neq 0$ we simply multiply row n on the right by d_n, letting $\hat{u}_n = d_n$, and divide column n on the left by d_n leaving 1 in its place. This is achieved by defining

$$X_n = \begin{bmatrix} \ddots & & & \\ & \ddots & & \\ & & 1 & \\ & & & d_n \end{bmatrix}$$

and finally

$$\hat{L} := \left(U \longleftarrow UX_n^{-1}\right) \text{ and } \hat{U} := (L \longleftarrow X_n L).$$

That is,

$$\hat{L} = UX_1^{-1} \dots X_n^{-1}$$
$$\hat{U} = X_n \dots X_1 L$$

So, recalling matrix K in (6.4) we can write

$$\underbrace{UK^{-1}}\underbrace{KL} = \hat{L}\hat{U}$$

where $K \equiv X_n \dots X_1$.

If $d_n = 0$ we can not define X_n. Observe that

$$\begin{bmatrix} \ddots & & & \\ \ddots & 1 & & \\ & \hat{l}_{n-2} & 1 & \\ & & \hat{l}_{n-1} & d_n \end{bmatrix} \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & \hat{u}_{n-1} & 1 \\ & & & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \ddots & & & \\ \ddots & 1 & & \\ & \hat{l}_{n-2} & 1 & \\ & & \hat{l}_{n-1} & 1 \end{bmatrix} \begin{bmatrix} \ddots & & & \\ & 1 & & \\ & & 1 & 0 \\ & & & d_n \end{bmatrix} \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & \hat{u}_{n-1} & 1 \\ & & & 1 \end{bmatrix}.$$

So, $\hat{L}$ does not change and we let $\hat{u}_n = 1$ and $d_n = 0$. $\hat{U}$ is not invertible but the factorization $\hat{L}\hat{D}\hat{R}$ in (6.3) with $\hat{U} = \hat{D}\hat{R}$ exists.

We have derived the dqd algorithm without reference to qd. Of more significance is the fact that the quantities d_i, $i = 1, \ldots, n$, provide useful information about UL, that qd does not reveal, and so dqd facilitates the choice of a shift.

6.2.1 The meaning of d_i

The meaning of d_i is revealed by the next theorem.

Theorem 6.2.2 *Consider L and U as described in (6.1). If U is invertible and UL permits triangular factorization then the quantities d_i, $i = 1, \ldots, n$, generated by the dqd algorithm applied to L and U satisfy*

$$d_i^{-1} = \left[(UL)^{-1}\right]_{ii}, \quad i = 1, \ldots, n.$$

Note that if U is invertible and $UL = \hat{L}\hat{U}$ then $\hat{U}$ must also be invertible. This means that u_i and $\hat{u}_i$ are all non-zero.

Proof. The algorithm may be considered as transforming L to $\hat{U}$ by premultiplications and U to $\hat{L}$ by inverse multiplications on the right, as described in the previous section. At the end of the $(i-1)^{\text{th}}$ plane transformer X_{i-1}, $i = 2, \ldots, n$, the situation is as indicated below:

$$UK_{i-1}^{-1} = \begin{bmatrix} \ddots & & & & & & \\ \ddots & 1 & 0 & & & & \\ & \hat{l}_{i-1} & d_i & 1 & & & \\ & & & u_{i+1} & 1 & & \\ & & & & u_{i+2} & \ddots & \\ & & & & & \ddots & \end{bmatrix}, \quad K_{i-1}L = \begin{bmatrix} \ddots & \ddots & & & & & \\ & \hat{u}_{i-1} & 1 & & & & \\ & 0 & 1 & & & & \\ & & l_i & 1 & & & \\ & & & l_{i+1} & 1 & & \\ & & & & \ddots & \ddots & \end{bmatrix}.$$

where

$$K_{i-1} \equiv X_{i-1}X_{i-2}\ldots X_1$$

and X_i is the plane transform given in (6.13).

The striking fact is that row i of $K_{i-1}L$ and column i of UK_{i-1}^{-1} are singletons. That is

$$\boldsymbol{e}_i^T K_{i-1} L = \boldsymbol{e}_i^T \quad \text{and} \quad U K_{i-1}^{-1} \boldsymbol{e}_i = \boldsymbol{e}_i d_i, \;\; i = 2, \ldots, n.$$

Rearranging these equations yields, for $i = 2, \ldots, n$,

$$\boldsymbol{e}_i^T K_{i-1} = \boldsymbol{e}_i^T L^{-1}, \qquad K_{i-1}^{-1} \boldsymbol{e}_i = U^{-1} \boldsymbol{e}_i d_i$$

and

$$\begin{aligned} \left[(UL)^{-1}\right]_{ii} &= \left(\boldsymbol{e}_i^T L^{-1}\right)\left(U^{-1}\boldsymbol{e}_i\right) \\ &= \left(\boldsymbol{e}_i^T K_{i-1}\right)\left(K_{i-1}^{-1}\boldsymbol{e}_i d_i^{-1}\right) \\ &= d_i^{-1}. \end{aligned}$$

Now for case $i = 1$. We also have

$$\boldsymbol{e}_1^T L = \boldsymbol{e}_1^T \quad \text{and} \quad U\boldsymbol{e}_1 = \boldsymbol{e}_1 u_1,$$

and, analogously,

$$\left[(UL)^{-1}\right]_{11} = \left(\boldsymbol{e}_1^T L^{-1}\right)\left(U^{-1}\boldsymbol{e}_1\right) = \boldsymbol{e}_1^T u_1^{-1} \boldsymbol{e}_1 = d_1^{-1}. \;\square$$

In the positive case ($l_i > 0,\; u_i > 0$), UL is diagonally similar to a symmetric positive definite matrix (see lemma 2.2.3) and the quantity $\min |d_j|$ gives useful information on the eigenvalue nearest 0.

Corollary 6.2.3 *In the positive case*

$$\frac{\min_i d_i}{n} < \lambda_{\min}(UL) \leq \min_i d_i.$$

Before writing the proof of this corollary we need to remember that if A is a symmetric positive definite matrix then

$$\text{trace}(A) > \lambda_{\max}(A) \quad \text{and} \quad \lambda_{\max}(A) \geq A_{ii}.$$

This is simple to prove: eigenvalues $\lambda_1, \ldots, \lambda_n$ of A are all positive and the first inequality is immediate. Also, A is orthogonally similar to $D = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. We have $A = Q^T D Q$, for an orthogonal matrix Q and then

$$\begin{aligned} A_{ii} &= \left[Q^T D Q\right]_{ii} \\ &= \boldsymbol{e}_i^T Q^T D Q \boldsymbol{e}_i \\ &= \boldsymbol{q}_i^T D \boldsymbol{q}_i \\ &= \lambda_1 q_{1i}^2 + \ldots + \lambda_n q_{ni}^2 \\ &\leq \lambda_{\max}(A) \left\|\boldsymbol{q}_i\right\|_2^2 = \lambda_{\max}(A), \end{aligned}$$

where we let $\boldsymbol{q}_i := \Big[q_{1i}, \ldots, q_{ni}\Big]^T$.

Also recall that for a matrix M diagonally similar to A we have

$$M_{ii} = A_{ii}$$

because a diagonal similarity does not affect the diagonal entries of A.

Now we write the proof of corollary 6.2.3.

Proof. For any matrix $M = (m_{ij})_{1 \leq i,j \leq n}$ that is diagonally similar to a positive definite symmetric matrix we have

$$\max_i m_{ii} \leq \lambda_{\max}(M) < \operatorname{trace}(M).$$

If we let $M = (UL)^{-1}$ we have $\left[(UL)^{-1}\right]_{ii} = d_i^{-1}$ and then

$$\left(\min_i d_i\right)^{-1} = \max_i d_i^{-1} \leq \lambda_{\max}\left((UL)^{-1}\right) < \sum_{i=1}^{n} d_i^{-1} < n \left(\min_i d_i\right)^{-1}.$$

Since $\left(\lambda_{\max}(UL)^{-1}\right) = (\lambda_{\min}(UL))^{-1}$ we can write

$$\left(\min_i d_i\right)^{-1} \leq (\lambda_{\min}(UL))^{-1} < n \left(\min_i d_i\right)^{-1}$$

or, equivalently,

$$\frac{\min_i d_i}{n} < \lambda_{\min}(UL) \leq \min_i d_i. \ \square$$

Even in the general case, as $u_n \to 0$, $\min_i |d_i|$ becomes an increasingly accurate approximation to $|\lambda_{\min}|$.

LR, QR and qd algorithms are only as good as their shift strategies. In practice, one uses qds and dqds, the shifted versions of qd and dqd.

The derivation of dqds(σ) in terms of Gram-Schmidt process is not obvious. Formally if $\left(U - \sigma L^{-1}\right) L$ admits triangular factorization we write

$$UL - \sigma I = \left(U - \sigma L^{-1}\right) L = \hat{L}\hat{D}\hat{R} = \hat{L}\hat{U}$$

and apply Gram-Schmidt to the columns of L and to the rows of $U - \sigma L^{-1}$ to obtain

$$U - \sigma L^{-1} = \hat{L}\tilde{P}^T, \qquad L = \tilde{Q}\hat{R}, \qquad \tilde{P}^T\tilde{Q} = \hat{D}.$$

If $\hat{D}$ is invertible, we can eliminate $\tilde{Q}$:

$$L = \left(\tilde{Q}\hat{D}^{-1}\right)\left(\hat{D}\hat{R}\right) = \tilde{P}^{-T}\hat{U}, \qquad U - \sigma L^{-1} = \hat{L}\tilde{P}^T,$$

that is

$$KL = \hat{U}, \qquad \left(U - \sigma L^{-1}\right) K^{-1} = \hat{L},$$

with $K := \tilde{P}^T$. At first glance the new term $-\sigma L^{-1}$ appears to spoil the derivation of K as a product of plane rotations. However, it is not necessary to know all the terms of L^{-1} but only the $(i+1, i)$ entries immediately below the main diagonal. The change for the unshifted case is small.

If one looks at the two matrices part way through the transformations $L \longrightarrow \hat{U}$, $U - \sigma L^{-1} \longrightarrow \hat{L}$, the singleton column in the second matrix (from theorem 6.2.2) has disappeared and the relation of d_i to $(UL)^{-1}$ will be more complicated.

6.2.2 Complex eigenvalues and dqd

Theorem 6.2.2 gives information about the diagonal entries of $(UL)^{-1}$ that can be used in a choice of a shift strategy. In the presence of a pair of complex conjugate eigenvalues it will

be useful to know not just the diagonal entries but also the entries $(i+1,i)$ and $(i,i+1)$, that is, the diagonal 2×2 blocks of $(UL)^{-1}$.

We will use the notation $A_{i:j,k:l}$ to represent the submatrix of A lying in rows i through j and columns k through l.

Next result gives the entries of the 2×2 submatrix

$$\left[(UL)^{-1}\right]_{i:i+1,i:i+1} \tag{6.14}$$

and then we will analyze when we may have complex eigenvalues.

Theorem 6.2.4 *Consider L and U as described in (6.1). If U is invertible and d_i, $i=1,\ldots,n$, are the intermediate quantities generated by the* dqd *algorithm applied to L and U, then*

$$\left[(UL)^{-1}\right]_{i:i+1,i:i+1} = \begin{bmatrix} \frac{1}{d_i} & -\frac{1}{d_i u_{i+1}} \\ -\frac{l_i}{d_i} & \frac{1}{d_{i+1}} \end{bmatrix}, \quad i=1,\ldots,n-1.$$

Proof. From theorem 6.2.2

$$\frac{1}{d_i} = \left[(UL)^{-1}\right]_{ii}, \;\; i=1,\ldots,n.$$

Also from the proof of theorem 6.2.2, we have

$$\boldsymbol{e}_i^T K_{i-1} = \boldsymbol{e}_i^T L^{-1} \quad \text{and} \quad d_i^{-1} K_{i-1}^{-1} \boldsymbol{e}_i = U^{-1}\boldsymbol{e}_i, \;\; i=2,\ldots,n,$$

where

$$K_{i-1} = X_{i-1}X_{i-2}\ldots X_1, \;\; i=2,\ldots,n-1$$

and X_i is given by (6.13).

Thus,

$$\begin{aligned} \left[(UL)^{-1}\right]_{i,i+1} &= \left(\boldsymbol{e}_i^T L^{-1}\right)\left(U^{-1}\boldsymbol{e}_{i+1}\right) \\ &= \left(\boldsymbol{e}_i^T K_{i-1}\right)\left(d_{i+1}^{-1} K_i^{-1}\boldsymbol{e}_{i+1}\right) \\ &= d_{i+1}^{-1}\boldsymbol{e}_i^T\left(K_{i-1}K_i^{-1}\right)\boldsymbol{e}_{i+1} \\ &= d_{i+1}^{-1}\left(\boldsymbol{e}_i^T X_i^{-1}\boldsymbol{e}_{i+1}\right) \\ &= -\frac{1}{d_{i+1}}\frac{1}{\hat{u}_i}. \end{aligned}$$

In dqd algorithm we have

$$d_{i+1} = d_i(u_{i+1}/\hat{u}_i)$$

and then

$$\left[(UL)^{-1}\right]_{i,i+1} = -\frac{1}{d_i u_{i+1}}.$$

Analogously,

$$\begin{aligned}
\left[(UL)^{-1}\right]_{i+1,i} &= \left(\boldsymbol{e}_{i+1}^T L^{-1}\right)\left(U^{-1}\boldsymbol{e}_i\right) \\
&= \left(\boldsymbol{e}_{i+1}^T K_i\right)\left(d_i^{-1} K_{i-1}^{-1}\boldsymbol{e}_i\right) \\
&= d_i^{-1}\boldsymbol{e}_{i+1}^T\left(K_i K_{i-1}^{-1}\right)\boldsymbol{e}_i \\
&= d_i^{-1}\left(\boldsymbol{e}_{i+1}^T X_i \boldsymbol{e}_i\right) \\
&= -\frac{l_i}{d_i}.
\end{aligned}$$

For $i = 1$, notice that $d_1 = u_1$ and we have

$$\boldsymbol{e}_1^T = L^{-1}\boldsymbol{e}_1^T \quad \text{and} \quad u_1^{-1}\boldsymbol{e}_1 = U^{-1}\boldsymbol{e}_1.$$

So,

$$\begin{aligned}
\left[(UL)^{-1}\right]_{1,2} &= \left(\boldsymbol{e}_1^T L^{-1}\right)\left(U^{-1}\boldsymbol{e}_2\right) \\
&= \boldsymbol{e}_1^T\left(d_2^{-1} K_1^{-1}\boldsymbol{e}_2\right) \\
&= d_2^{-1}\left(\boldsymbol{e}_1^T K_1^{-1}\boldsymbol{e}_2\right) \\
&= d_2^{-1}\left(\boldsymbol{e}_2^T X_1^{-1}\boldsymbol{e}_2\right) \\
&= -\frac{1}{d_2}\frac{1}{\hat{u}_1} = -\frac{1}{d_1 u_2}
\end{aligned}$$

and, finally,

$$\begin{aligned}\left[(UL)^{-1}\right]_{2,1} &= \left(\boldsymbol{e}_2^T L^{-1}\right)\left(U^{-1}\boldsymbol{e}_1\right)\\ &= \left(\boldsymbol{e}_2^T K_1\right)\left(d_1^{-1}\boldsymbol{e}_1\right)\\ &= d_1^{-1}\left(\boldsymbol{e}_2^T K_1 \boldsymbol{e}_1\right)\\ &= d_1^{-1}\left(\boldsymbol{e}_2^T X_1 \boldsymbol{e}_1\right)\\ &= -\frac{l_1}{d_1}.\ \square\end{aligned}$$

Now we are interested in finding the eigenvalues of the 2×2 submatrix of theorem 6.2.4. It can be verified that the discriminant of the characteristic polynomial is

$$\Delta_i \equiv \left(\frac{d_{i+1}-d_i}{d_i d_{i+1}}\right)^2 + \frac{4l_i}{d_i^2 u_{i+1}}. \tag{6.15}$$

Lemma 6.2.1 *If $l_i u_{i+1} > 0$ then the 2×2 matrix in (6.14) has real eigenvalues.*

Proof. According to (6.15), if $l_i u_{i+1} \geq 0$ then $\Delta_i \geq 0$. $\square$

If $l_i u_{i+1} < 0$ we must evaluate Δ_i to know if we are in the presence of complex eigenvalues.

6.3 New version of triple dqds

Consider matrices L and U as given in (6.1) and the application of tridqds algorithm. Initially, we define

$$F := U \qquad \text{and} \qquad G := L$$

and

$$F^{(1)} := F \qquad \text{and} \qquad G^{(1)} := G.$$

At the beginning of each minor step i, $i = 2, \ldots, n$, we have

$$F^{(i)} := \mathcal{L}_{i-1}^{-1} F^{(i-1)} Z_{i-1}^{-1} Y_{i-1}^{-1}$$
$$G^{(i)} := Y_{i-1} Z_{i-1} G^{(i-1)} \mathcal{L}_{i-1}.$$

Therefore

$$F^{(i)} := \left(\mathcal{L}_{i-1}^{-1} \ldots \mathcal{L}_2^{-1} \mathcal{L}_1^{-1}\right) U \left(Z_1^{-1} Y_1^{-1} Z_2^{-1} Y_2^{-1} \ldots Z_{i-1}^{-1} Y_{i-1}^{-1}\right)$$
$$G^{(i)} := (Y_{i-1} Z_{i-1} \ldots Y_2 Z_2 Y_1 Z_1) L (\mathcal{L}_1 \mathcal{L}_2 \ldots \mathcal{L}_{i-1})$$

and

$$F^{(i)} G^{(i)} = (\mathcal{L}_{i-1}^{-1} \ldots \mathcal{L}_2^{-1} \mathcal{L}_1^{-1})(UL)(\mathcal{L}_1 \mathcal{L}_2 \ldots \mathcal{L}_{i-1}). \tag{6.16}$$

As established in (5.12), page 143, at the beginning of minor step $i = 2, \ldots, n-2$, of tridqds algorithm we have

$$F^{(i)} = \begin{bmatrix} \ddots & & & & & \\ \ddots & 1 & & & & \\ & \hat{l}_{i-1} & u_i & 1 & & \\ & f_1^{i-1} & & u_{i+1} & 1 & \\ & f_2^{i-1} & & & u_{i+2} & \ddots \\ & & & & & \ddots \end{bmatrix}, \quad G^{(i)} = \begin{bmatrix} \ddots & \ddots & & & & \\ & \hat{u}_{i-1} & 1 & & & \\ & & d_i & & & \\ & & g_1^i & 1 & & \\ & & g_2^i & l_{i+1} & 1 & \\ & & & & \ddots & \ddots \end{bmatrix}, \tag{6.17}$$

where we defined

$$f_1^{i-1} := x_l \qquad \text{and} \qquad d_i := x_r$$
$$f_2^{i-1} := y_l \qquad\qquad g_1^i := y_r$$
$$g_2^i := z_r.$$

The superscript refers to the column at which the element is positioned.

At the beginning of minor step $n-1$,

$$F^{n-1} = \begin{bmatrix} \ddots & \ddots & & & \\ & \hat{l}_{n-3} & 1 & & \\ & & \hat{l}_{n-2} & u_{n-1} & 1 \\ & & f_1^{n-2} & & u_n \end{bmatrix}, \qquad G^{n-1} = \begin{bmatrix} \ddots & \ddots & & & \\ & \hat{u}_{n-3} & 1 & & \\ & & \hat{u}_{n-2} & 1 & \\ & & & d_{n-1} & \\ & & & g_1^{n-1} & 1 \end{bmatrix}$$

and for minor step n

$$F_n = \begin{bmatrix} \ddots & & & \\ \ddots & 1 & & \\ & \hat{l}_{n-2} & 1 & \\ & & \hat{l}_{n-1} & u_n \end{bmatrix}, \qquad G_n = \begin{bmatrix} \ddots & \ddots & & \\ & \hat{u}_{n-2} & 1 & \\ & & \hat{u}_{n-1} & 1 \\ & & & d_n \end{bmatrix}.$$

Since initially we let

$$F^{(1)} = U \qquad \text{and} \qquad G^{(1)} = L,$$

then

$$d_1 = 1, \qquad g_1^1 = l_1 \qquad \text{and} \qquad g_2^1 = 0.$$

6.3.1 New notation for tridqds

Next we will rewrite tridqds algorithm with the notation introduced in (6.17). First column of $F^{(1)}$ is $u_1 \boldsymbol{e}_1$ and quantities f_1^0 and f_2^0 presented at minor step 1 are just auxiliary variables used for clarity and do not *belong* to $F^{(1)}$.

The first minor step of tridqds.

Minor step 1 :

$$d_1 = 1; \;\; g_1^1 = l_1; \;\; g_2^1 = 0$$

$$d_2 = d_1 * u_1 + g_1^1$$

$$f_1^0 = (u_1 + l_1)^2 + u_2 l_1 - 2(\Re\sigma)(u_1 + l_1) + |\sigma|^2$$
$$f_2^0 = u_2 l_1 u_3 l_2 / f_1^0$$
$$f_1^0 = u_2 l_1 \big(u_1 + l_1 + u_2 + l_2 - 2(\Re\sigma)\big) / f_1^0$$
$$f_1^1 = -f_1^0; \;\; f_2^1 = -f_2^0$$

$$\hat{u}_1 = d_2 - f_1^1;$$
$$d_2 = g_1^1 - f_1^1; \;\; g_1^2 = g_2^1 - f_2^1 - f_1^1 * l_2; \;\; g_2^2 = -f_2^1 * l_3$$

$$d_2 = d_2 / \hat{u}_1; \;\; g_1^2 = g_1^2 / \hat{u}_1; \;\; g_2^2 = g_2^2 / \hat{u}_1$$

$$\hat{l}_1 = f_1^1 + g_1^2 + d_2 * u_2$$
$$f_1^1 = f_2^1 + g_2^2 + g_1^2 * u_3; \;\; f_2^1 = g_2^2 * u_4$$

$$d_2 = 1 - d_2; \;\; g_1^2 = l_2 - g_1^2; \;\; g_2^2 = -g_2^2$$

The inner loop is changed to:

for $i = 2, \ldots, n-3$

$$d_{i+1} = d_i * u_i + g_1^i$$

$$f_1^i = -f_1^{i-1}/\hat{l}_{i-1}; \;\; f_2^i = -f_2^{i-1}/\hat{l}_{i-1};$$

$$\hat{u}_i = d_{i+1} - f_1^i;$$
$$d_{i+1} = g_1^i - f_1^i; \;\; g_1^{i+1} = g_2^i - f_2^i - f_1^i * l_{i+1}; \;\; g_2^{i+1} = -f_2^i * l_{i+2}$$

$$d_{i+1} = d_{i+1}/\hat{u}_i; \;\; g_1^{i+1} = g_1^{i+1}/\hat{u}_i; \;\; g_2^{i+1} = g_2^{i+1}/\hat{u}_i$$

$$\hat{l}_i = f_1^i + g_1^{i+1} + d_{i+1} * u_{i+1}$$
$$f_1^i = f_2^i + g_2^{i+1} + g_1^{i+1} * u_{i+2}; \;\; f_2^i = g_2^{i+1} * u_{i+3}$$

$$d_{i+1} = 1 - d_{i+1}; \;\; g_1^{i+1} = l_{i+1} - g_1^{i+1}; \;\; g_2^{i+1} = -g_2^{i+1}$$

end for

We begin the loop with quantities f_1^{i-1}, f_2^{i-1}, d_i, g_1^i and g_2^i and end with quantities f_1^i, f_2^i, d_{i+1}, g_1^{i+1} and g_2^{i+1}, and all values are preserved. For completeness we will also write down the minor steps $n-2$, $n-1$ and n with this new notation.

Minor step n-2 :

$$d_{n-1} = d_{n-2} * u_{n-2} + g_1^{n-2}$$

$$f_1^{n-2} = -f_1^{n-3}/\hat{l}_{n-3};\;\; f_2^{n-2} = -f_2^{n-3}/\hat{l}_{n-3};$$

$$\hat{u}_{n-2} = d_{n-1} - f_1^{n-2};$$
$$d_{n-1} = g_1^{n-2} - f_1^{n-2};\;\; g_1^{n-1} = g_2^{n-2} - f_2^{n-2} - f_1^{n-2} * l_{n-1}$$

$$d_{n-1} = d_{n-1}/\hat{u}_{n-2};\;\; g_1^{n-1} = g_1^{n-1}/\hat{u}_{n-2}$$

$$\hat{l}_{n-2} = f_1^{n-2} + g_1^{n-1} + d_{n-1} * u_{n-1}$$
$$f_1^{n-2} = f_2^{n-2} + g_1^{n-1} * u_n$$

$$d_{n-1} = 1 - d_{n-1};\;\; g_1^{n-1} = l_{n-1} - g_1^{n-1}$$

Minor step n-1 :

$$d_n = d_{n-1} * u_{n-1} + g_1^{n-1}$$

$$f_1^{n-1} = -f_1^{n-2}/\hat{l}_{n-2}$$

$$\hat{u}_{n-1} = d_n - f_1^{n-1}$$
$$d_n = g_1^{n-1} - f_1^{n-1}$$

$$d_n = d_n/\hat{u}_{n-1}$$

$$\hat{l}_{n-1} = f_1^{n-1} + d_n * u_n$$

$$d_n = 1 - d_n$$

Minor step n :

$$\hat{u}_n = d_n * u_n$$

$$\hat{u}_n = \hat{u}_n;$$

To represent elementary matrix $\mathcal{L}_i$ at minor step i, $i = 1, \ldots, n-2$, we will use the notation

$$\mathcal{L}_i = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & h_1^i & 1 & & & \\ & & h_2^i & & 1 & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}. \tag{6.18}$$

Elements h_1^i and h_2^i are at positions $(i+1, i)$ and $(i+2, i)$, respectively. For $i = n-1$ and $i = n$,

$$\mathcal{L}_{n-1} = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \\ & & & h_1^{n-1} & 1 \end{bmatrix} \qquad \text{and} \qquad \mathcal{L}_n = I.$$

We will write

$$\mathcal{L}_i = I + \boldsymbol{h}_i \boldsymbol{e}_i^T, \qquad i = 1, \ldots, n-1,$$

where

$$\boldsymbol{h}_i = \begin{cases} \begin{bmatrix} 0 & \ldots & 0 & 0 & h_1^i & h_2^i & 0 & \ldots & 0 \end{bmatrix}^T, & i = 1, \ldots, n-2 \\ \\ \begin{bmatrix} 0 & \ldots & 0 & 0 & h_1^{n-1} \end{bmatrix}^T, & i = n-1. \end{cases}$$

It is well known that

$$\mathcal{L}_i^{-1} = I - \boldsymbol{h}_i \boldsymbol{e}_i^T$$

and

$$\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_i = \begin{bmatrix} 1 & & & & & & & \\ h_1^1 & 1 & & & & & & \\ h_2^1 & h_1^2 & \ddots & & & & & \\ & h_2^2 & \ddots & 1 & & & & \\ & & \ddots & h_1^i & 1 & & & \\ & & & h_2^i & & 1 & & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{bmatrix}.$$

In what follows we will need to refer to the columns and to the rows of $\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_i$. For $i = 1, \dots, n$, the columns are

$$(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_i)\boldsymbol{e}_k = \begin{cases} \Big[0 \quad \dots \quad 0 \quad \overbrace{1}^{k^{\text{th}}} \quad h_1^k \quad h_2^k \quad 0 \quad \dots \quad 0\Big]^T, & k \le i \\ \boldsymbol{e}_k, & k > i \end{cases}. \tag{6.19}$$

Note that $(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_{n-1})\boldsymbol{e}_{n-1} = \begin{bmatrix} 0 & \dots & 0 & 1 & h_1^{n-1} \end{bmatrix}^T$ and $(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_n)\boldsymbol{e}_n = \boldsymbol{e}_n$. And for $i = 1, \dots, n$, the rows are given by

$$\boldsymbol{e}_k^T(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_i) = \begin{cases} \Big[0 \quad \dots \quad 0 \quad h_2^{k-2} \quad h_1^{k-1} \quad \overbrace{1}^{k^{\text{th}}} \quad 0 \quad \dots \quad 0\Big], & k \le i+1 \\ \Big[0 \quad \dots \quad 0 \quad h_2^{k-2} \quad 0 \quad 1 \quad 0 \quad \dots \quad 0\Big], & k = i+2 \\ \boldsymbol{e}_k^T, & k > i+2 \end{cases} \tag{6.20}$$

Note that $\boldsymbol{e}_1^T(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_i) = \boldsymbol{e}_1^T$ and $\boldsymbol{e}_2^T(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_i) = \begin{bmatrix} h_1^1 & 1 & 0 & \dots & 0 \end{bmatrix}$.

6.3.2 From tridqds to 3dqds

The next sequence of lemmas will permit us to change the tridqds algorithm into an equivalent but more efficient format. This is a central contribution of the thesis and lead us to the new version, 3dqds, that will be presented in page 201.

Lemma 6.3.1 *Consider tridqds algorithm applied to matrices L and U as given in (6.1). For $\mathcal{L}_1$ we have*

$$h_1^1 = f_1^0 \qquad \textit{and} \qquad h_2^1 = f_2^0.$$

For $\mathcal{L}_i$ at minor step i, $i = 2, \ldots, n-2$, we have

$$h_1^i = f_1^{i-1}/\hat{l}_{i-1} \qquad \textit{and} \qquad h_2^i = f_2^{i-1}/\hat{l}_{i-1}.$$

Finally, for $\mathcal{L}_{n-1}$

$$h_1^{n-1} = f_1^{n-2}/\hat{l}_{n-2}.$$

Proof. In tridqds algorithm the inverse $\mathcal{L}_i^{-1}$, $i = 2, \ldots, n-2$, is chosen to zero out entries $(i+1, i-1)$ and $(i+2, i-1)$ of $F^{(i)}$ using entry $(i, i-1)$ as the pivot. Thus

$$\mathcal{L}_i^{-1} = \begin{bmatrix} \ddots & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & -f_1^{i-1}/\hat{l}_{i-1} & 1 & & \\ & & -f_2^{i-1}/\hat{l}_{i-1} & & 1 & \\ & & & & & \ddots \end{bmatrix}. \tag{6.21}$$

So, entries h_1^i and h_2^i in $\mathcal{L}_i$ are the symmetric of the corresponding entries in $\mathcal{L}_i^{-1}$. $\mathcal{L}_{n-1}$ is chosen to zero out entry $(n, n-2)$ of F^{n-1}. The case $i = 1$ is immediate (see minor step 1 to observe the initialization of f_1^0 and f_2^0). $\square$

In all the results that follow we will consider $n \geq 4$.

Lemma 6.3.2 *Consider* tridqds *algorithm and matrix $F^{(i)}$ at the beginning of minor step i, $i = 2, \ldots, n-1$. Then, for $i = 2$,*

$$f_1^1 = \frac{u_3 l_2}{\hat{u}_1} f_1^0 + \frac{l_3 + u_3 - \hat{u}_1}{\hat{u}_1} f_2^0 + \frac{u_3}{\hat{u}_1} g_2^1 \qquad \textit{and} \qquad f_2^1 = \frac{u_4 l_3}{\hat{u}_1} f_2^0.$$

For $i = 3, \ldots, n-2$,

$$f_1^{i-1} = \frac{u_{i+1} l_i}{\hat{l}_{i-2} \hat{u}_{i-1}} f_1^{i-2} + \frac{l_{i+1} + u_{i+1} - \hat{u}_{i-1}}{\hat{l}_{i-2} \hat{u}_{i-1}} f_2^{i-2} + \frac{u_{i+1}}{\hat{u}_{i-1}} g_2^{i-1}$$

and

$$f_2^{i-1} = \frac{u_{i+2} l_{i+1}}{\hat{l}_{i-2} \hat{u}_{i-1}} f_2^{i-2}.$$

At the beginning of minor step $n-1$,

$$f_1^{n-2} = \frac{u_n l_{n-1}}{\hat{l}_{n-3} \hat{u}_{n-2}} f_1^{n-3} + \frac{u_n - \hat{u}_{n-2}}{\hat{l}_{n-3} \hat{u}_{n-2}} f_2^{n-3} + \frac{u_n}{\hat{u}_{n-2}} g_2^{n-2}.$$

Proof. At the end of minor step 1 we obtain

$$\begin{aligned} f_1^1 &= -f_2^0 + \left[\left(f_2^0 * l_3 \right) / \hat{u}_1 \right] + \left[\left(g_2^1 + f_2^0 + f_1^0 * l_2 \right) / \hat{u}_1 \right] * u_3 \\ &= \frac{u_3 l_2}{\hat{u}_1} f_1^0 + \frac{l_3 + u_3 - \hat{u}_1}{\hat{u}_1} f_2^0 + \frac{u_3}{\hat{u}_1} g_2^1 \end{aligned}$$

and

$$f_2^1 = \left[\left(f_2^0 * l_3 \right) / \hat{u}_1 \right] * u_4 = \frac{u_4 l_3}{\hat{u}_1} f_2^0.$$

The sequence of assignments to compute f_2^i, $i = 2, \ldots, n-3$, is

$$\begin{aligned} f_2^i &\longleftarrow -f_2^{i-1} / \hat{l}_{i-1} \\ g_2^{i+1} &\longleftarrow -f_2^i * l_{i+2} \\ g_2^{i+1} &\longleftarrow g_2^{i+1} / \hat{u}_i \\ f_2^i &\longleftarrow g_2^{i+1} * u_{i+3}. \end{aligned}$$

Doing *forward substitution* we get

$$\begin{aligned} f_2^i &= \left\{ \left[- \left(-f_2^{i-1} / \hat{l}_{i-1} \right) * l_{i+2} \right] / \hat{u}_i \right\} * u_{i+3} \\ &= \frac{f_2^{i-1} l_{i+2}}{\hat{l}_{i-1} \hat{u}_i} * u_{i+3} \\ &= \frac{u_{i+3} l_{i+2}}{\hat{l}_{i-1} \hat{u}_i} f_2^{i-1}. \end{aligned}$$

So, in the beginning of minor step i, $i = 3, \ldots, n-2$, we have

$$f_2^{i-1} = \frac{u_{i+2} l_{i+1}}{\hat{l}_{i-2}\hat{u}_{i-1}} f_2^{i-2}.$$

Analogously, to obtain f_1^i, $i = 2, \ldots, n-3$, the sequence is

$$\begin{aligned}
f_1^i &\longleftarrow -f_1^{i-1}/\hat{l}_{i-1} \\
f_2^i &\longleftarrow -f_2^{i-1}/\hat{l}_{i-1} \\
g_1^{i+1} &\longleftarrow g_2^i - f_2^i - f_1^i * l_{i+1}; \\
g_2^{i+1} &\longleftarrow -f_2^i * l_{i+2} \\
g_1^{i+1} &\longleftarrow g_1^{i+1}/\hat{u}_i; \\
g_2^{i+1} &\longleftarrow g_2^{i+1}/\hat{u}_i \\
f_1^i &\longleftarrow f_2^i + g_2^{i+1} + g_1^{i+1} * u_{i+2}.
\end{aligned}$$

Thus, after minor step i,

$$\begin{aligned}
f_1^i &= \left(-f_2^{i-1}/\hat{l}_{i-1}\right) + \left\{\left[-\left(-f_2^{i-1}/\hat{l}_{i-1}\right) * l_{i+2}\right]/\hat{u}_i\right\} + \\
&\quad + \left\{\left[g_2^i - \left(-f_2^{i-1}/\hat{l}_{i-1}\right) - \left(-f_1^{i-1}/\hat{l}_{i-1}\right) * l_{i+1}\right]/\hat{u}_i\right\} * u_{i+2} \\
&= -\frac{f_2^{i-1}}{\hat{l}_{i-1}} + \frac{f_2^{i-1} l_{i+2}}{\hat{l}_{i-1}\hat{u}_i} + \left(\frac{g_2^i}{\hat{u}_i} + \frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} + \frac{f_1^{i-1} l_{i+1}}{\hat{l}_{i-1}\hat{u}_i}\right) u_{i+2} \\
&= -\frac{1}{\hat{l}_{i-1}} f_2^{i-1} + \frac{l_{i+2}}{\hat{l}_{i-1}\hat{u}_i} f_2^{i-1} + \frac{u_{i+2}}{\hat{u}_i} g_2^i + \frac{u_{i+2}}{\hat{l}_{i-1}\hat{u}_i} f_2^{i-1} + \frac{u_{i+2} l_{i+1}}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} \\
&= \frac{u_{i+2} l_{i+1}}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{l_{i+2} + u_{i+2} - \hat{u}_i}{\hat{l}_{i-1}\hat{u}_i} f_2^{i-1} + \frac{u_{i+2}}{\hat{u}_i} g_2^i.
\end{aligned}$$

In the beginning of minor step i, $i = 3, \ldots, n-2$, we have

$$f_1^{i-1} = \frac{u_{i+1} l_i}{\hat{l}_{i-2}\hat{u}_{i-1}} f_1^{i-2} + \frac{l_{i+1} + u_{i+1} - \hat{u}_{i-1}}{\hat{l}_{i-2}\hat{u}_{i-1}} f_2^{i-2} + \frac{u_{i+1}}{\hat{u}_{i-1}} g_2^{i-1}.$$

At the end of minor step $i = n-2$, or at the beginning of minor step $n-1$,

$$\begin{aligned}
f_1^{n-2} &= \left(-f_2^{n-3}/\hat{l}_{n-3}\right) + \left\{\left[g_2^{n-2} - \left(-f_2^{n-3}/\hat{l}_{n-3}\right) - \left(-f_1^{n-3}/\hat{l}_{n-3}\right) * l_{n-1}\right]/\hat{u}_{n-2}\right\} * u_n \\
&= \frac{u_n l_{n-1}}{\hat{l}_{n-3}\hat{u}_{n-2}} f_1^{n-3} + \frac{u_n - \hat{u}_{n-2}}{\hat{l}_{n-3}\hat{u}_{n-2}} f_2^{n-3} + \frac{u_n}{\hat{u}_{n-2}} g_2^{n-2}. \ \square
\end{aligned}$$

Lemma 6.3.3 *Consider matrix $G^{(i)}$ at the beginning of minor step i, $i = 2, \ldots, n-1$, of* tridqds *algorithm. Then, for $i = 2$,*

$$g_1^2 = l_2 - \left(\frac{l_2 f_1^0}{\hat{u}_1} + \frac{f_2^0}{\hat{u}_1} + \frac{g_2^1}{\hat{u}_1} \right) \qquad \text{and} \qquad g_2^2 = -\frac{l_3}{\hat{u}_1} f_2^0.$$

For $i = 3, \ldots, n-1$,

$$g_1^i = l_i - \left(\frac{l_i}{\hat{l}_{i-2}\hat{u}_{i-1}} f_1^{i-2} + \frac{f_2^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}} + \frac{g_2^{i-1}}{\hat{u}_{i-1}} \right)$$

and for $i = 3, \ldots, n-2$,

$$g_2^i = -\frac{l_{i+1}}{\hat{l}_{i-2}\hat{u}_{i-1}} f_2^{i-2}.$$

Proof. At the beginning of minor step $i = 2$, from minor step 1, we have

$$g_2^2 = -\left[\left(f_2^0 * l_3 \right) / \hat{u}_1 \right] = -\frac{l_3}{\hat{u}_1} f_2^0$$

and

$$g_1^2 = l_2 - \left[\left(g_2^1 + f_2^0 + f_1^0 * l_2 \right) / \hat{u}_1 \right] = l_2 - \left(\frac{l_2 f_1^0}{\hat{u}_1} + \frac{f_2^0}{\hat{u}_1} + \frac{g_2^1}{\hat{u}_1} \right).$$

The sequence of assignments to compute g_2^{i+1}, $i = 2, \ldots, n-3$, is

$$\begin{aligned} f_2^i &\longleftarrow -f_2^{i-1}/\hat{l}_{i-1} \\ g_2^{i+1} &\longleftarrow -f_2^i * l_{i+2} \\ g_2^{i+1} &\longleftarrow g_2^{i+1}/\hat{u}_i \\ g_2^{i+1} &\longleftarrow -g_2^{i+1}. \end{aligned}$$

Again, substituting forwards, we get

$$g_2^{i+1} = -\left\{ \left[-\left(-f_2^{i-1}/\hat{l}_{i-1} \right) * l_{i+2} \right] / \hat{u}_i \right\} = -\frac{l_{i+2}}{\hat{l}_{i-1}\hat{u}_i} f_2^{i-1}.$$

Thus in the beginning of minor step $i = 3, \ldots, n-2$ we have

$$g_2^i = -\frac{l_{i+1}}{\hat{l}_{i-2}\hat{u}_{i-1}} f_2^{i-2}.$$

Now the sequence of assignments to compute g_1^{i+1}, $i = 2, \ldots, n-3$, is

$$\begin{aligned} f_1^i &\longleftarrow -f_1^{i-1}/\hat{l}_{i-1} \\ f_2^i &\longleftarrow -f_2^{i-1}/\hat{l}_{i-1} \\ g_1^{i+1} &\longleftarrow g_2^i - f_2^i - f_1^i * l_{i+1} \\ g_1^{i+1} &\longleftarrow g_1^{i+1}/\hat{u}_i \\ g_1^{i+1} &\longleftarrow l_{i+1} - g_1^{i+1} \end{aligned}$$

and we obtain

$$\begin{aligned} g_1^{i+1} &= l_{i+1} - \left\{ \left[g_2^i - \left(-f_2^{i-1}/\hat{l}_{i-1} \right) - \left(-f_1^{i-1}/\hat{l}_{i-1} \right) * l_{i+1} \right] /\hat{u}_i \right\} \\ &= l_{i+1} - \frac{g_2^i}{\hat{u}_i} - \frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} - \frac{f_1^{i-1}}{\hat{l}_{i-1}\hat{u}_i} l_{i+1} \\ &= l_{i+1} - \left(\frac{l_{i+1}}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} + \frac{g_2^i}{\hat{u}_i} \right). \end{aligned}$$

Thus, in the beginning of minor step i, $i = 3, \ldots, n-2$, we have

$$g_1^i = l_i - \left(\frac{l_i}{\hat{l}_{i-2}\hat{u}_{i-1}} f_1^{i-2} + \frac{f_2^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}} + \frac{g_2^{i-1}}{\hat{u}_{i-1}} \right).$$

Finally, at the beginning of minor step $n-1$,

$$\begin{aligned} g_1^{n-1} &= l_{n-1} - \left\{ \left[g_2^{n-2} - \left(-f_2^{n-3}/\hat{l}_{n-3} \right) - \left(-f_1^{n-3}/\hat{l}_{n-3} \right) * l_{n-1} \right] /\hat{u}_{n-2} \right\} \\ &= l_{n-1} - \left(\frac{l_{n-1}}{\hat{l}_{n-3}\hat{u}_{n-2}} f_1^{n-3} + \frac{f_2^{n-3}}{\hat{l}_{n-3}\hat{u}_{n-2}} + \frac{g_2^{n-2}}{\hat{u}_{n-2}} \right). \quad \square \end{aligned}$$

Lemma 6.3.4 *Let $F^{(i)}$ and $G^{(i)}$ be the matrices obtained at the beginning of minor step i, $i = 2, \ldots, n-1$, when we apply* tridqds *algorithm to L and U. Then*

$$f_2^{i-1} = -u_{i+2} g_2^i,$$

for $i = 2, \ldots, n-2$, and

$$f_2^{n-2} = -f_2^{n-3}/\hat{l}_{n-3}.$$

Proof. From lemmas 6.3.2 and 6.3.3 we know that, for $i = 3, \ldots, n-2$,

$$f_2^{i-1} = \frac{u_{i+2} l_{i+1}}{\hat{l}_{i-2}\hat{u}_{i-1}} f_2^{i-2} \qquad \text{and} \qquad g_2^i = -\frac{l_{i+1}}{\hat{l}_{i-2}\hat{u}_{i-1}} f_2^{i-2}.$$

The result follows immediately.

For $i = 2$, that is at the end of minor step 1, we have in sequence

$$f_2^1 \longleftarrow g_2^2 * u_4$$
$$g_2^2 \longleftarrow -g_2^2.$$

Thus,

$$g_2^2 = -\frac{f_2^1}{u_4} \Leftrightarrow f_2^1 = -u_4 g_2^2.$$

For $i = n-1$, $f_2^{n-2} = -f_2^{n-3}/\hat{l}_{n-3}$ comes from the last assignment for f_2^{n-2} in step $n-2$. $\square$

Lemma 6.3.5 *Let $F^{(i)}$ and $G^{(i)}$ be the matrices obtained at the beginning of each minor step i, $i = 2, \ldots, n$, when we apply* tridqds *algorithm to L and U. Then*

$$f_1^1 = u_3 \left(l_2 - g_1^2\right) - g_2^2 - f_2^0,$$

for $i = 2, \ldots, n-3$,

$$f_1^i = u_{i+2}(l_{i+1} - g_1^{i+1}) - g_2^{i+1} - \frac{f_2^{i-1}}{\hat{l}_{i-1}}$$

and

$$f_1^{n-2} = u_n \left(l_{n-1} - g_1^{n-1}\right) - \frac{f_2^{n-3}}{\hat{l}_{n-3}}, \qquad f_1^{n-1} = -\frac{f_1^{n-2}}{\hat{l}_{n-2}}.$$

Proof. From lemmas 6.3.2 and 6.3.3 we know that, for $i = 2, \ldots, n-3$,

$$f_1^i = \frac{u_{i+2} l_{i+1}}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{l_{i+2} + u_{i+2} - \hat{u}_i}{\hat{l}_{i-1}\hat{u}_i} f_2^{i-1} + \frac{u_{i+2}}{\hat{u}_i} g_2^i$$

and

$$g_1^{i+1} = l_{i+1} - \left(\frac{l_{i+1}}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} + \frac{g_2^i}{\hat{u}_i}\right).$$

Also for $i = 3, \ldots, n-2$, from lemma 6.3.4,

$$g_2^{i+1} = -\frac{l_{i+2}}{\hat{l}_{i-1}\hat{u}_i} f_2^{i-1}.$$

So,

$$\begin{aligned} f_1^i &= u_{i+2}\left(\frac{l_{i+1}}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} + \frac{g_2^i}{\hat{u}_i}\right) + (l_{i+2} - \hat{u}_i)\frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} \\ &= u_{i+2}(l_{i+1} - g_1^{i+1}) + l_{i+2}\frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} - \frac{f_2^{i-1}}{\hat{l}_{i-1}} \\ &= u_{i+1}(l_i - g_1^i) - g_2^{i+1} - \frac{f_2^{i-1}}{\hat{l}_{i-1}}. \end{aligned}$$

The cases $i = 2$ and $i = n-2$ also derive from lemmas 6.3.2 and 6.3.3. The case $i = n-1$ follows directly from step $n-1$. □

Lemma 6.3.6 *Consider $G^{(i)}$ at the beginning of minor step i, $i = 1, \ldots, n$, of* tridqds *algorithm. Then quantities d_i satisfy*

$$d_1 = 1, \qquad d_2 = 1 - \left(\frac{g_1^1}{\hat{u}_1} + \frac{f_1^0}{\hat{u}_1}\right)$$

and

$$d_i = 1 - \left(\frac{g_1^{i-1}}{\hat{u}_{i-1}} + \frac{f_1^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}}\right), \quad i = 3, \ldots, n.$$

Proof. From minor step 1 the result for d_2 is straightforward,

$$d_2 = 1 - [(g_1^1 + f_1^0)/\hat{u}_1].$$

In minor step i, $i = 2, \ldots, n-1$, we obtain d_{i+1} through the sequence

$$\begin{aligned} f_1^i &\longleftarrow -f_1^{i-1}/\hat{l}_{i-1} \\ d_{i+1} &\longleftarrow g_1^i - f_1^i \\ d_{i+1} &\longleftarrow d_{i+1}/\hat{u}_i \\ d_{i+1} &\longleftarrow 1 - d_{i+1}. \end{aligned}$$

Then,

$$d_{i+1} = 1 - \left\{ \left[g_1^i - \left(-f_1^{i-1}/\hat{l}_{i-1} \right) \right] / \hat{u}_i \right\} = 1 - \left(\frac{g_1^i}{\hat{u}_i} + \frac{f_1^{i-1}}{\hat{l}_{i-1}\hat{u}_i} \right).$$

For $i = 3, \ldots, n$ we have

$$d_i = 1 - \left(\frac{g_1^{i-1}}{\hat{u}_{i-1}} + \frac{f_1^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}} \right). \ \square$$

Lemma 6.3.7 *Consider $\hat{u}_i$, $i = 1, \ldots, n$, obtained when* tridqds *algorithm is applied to L and U. Then*

$$\begin{aligned} \hat{u}_1 &= d_1 u_1 + g_1^1 + f_1^0, \\ \hat{u}_i &= d_i u_i + g_1^i + \frac{f_1^{i-1}}{\hat{l}_{i-1}}, \qquad i = 2, \ldots, n-1, \\ \hat{u}_n &= d_n u_n. \end{aligned}$$

Proof. For $i = 1$ we have

$$\hat{u}_1 = \left(d_1 * u_1 + g_1^1 \right) + f_1^0 = d_1 u_1 + g_1^1 + f_1^0.$$

To obtain $\hat{u}_i$, $i = 2, \ldots, n-1$, we need to compute

$$\begin{aligned} d_{i+1} &\longleftarrow d_i * u_i + g_1^i \\ f_1^i &\longleftarrow -f_1^{i-1}/\hat{l}_{i-1} \\ \hat{u}_i &\longleftarrow d_{i+1} - f_1^i. \end{aligned}$$

Thus,

$$\hat{u}_i = \left(d_i * u_i + g_1^i \right) - \left(-f_1^{i-1}/\hat{l}_{i-1} \right) = d_i u_i + g_1^i + \frac{f_1^{i-1}}{\hat{l}_{i-1}}.$$

In the end

$$\hat{u}_n = d_n * u_n. \ \square$$

The result of the next theorem is quite surprising. It gives for the quantities d_i generated by the triple dqds a relation to the quantities u_i and $\hat{u}_i$ similar to the one that exists in dqd. Recall from (5.13), page 161, that in dqd algorithm we have

$$\begin{aligned} d_1 &= u_1 \\ d_{i+1} &= d_i \left(\frac{u_{i+1}}{\hat{u}_i} \right), \qquad i = 1, \ldots, n-1, \end{aligned}$$

and for tridqds

Theorem 6.3.1 *For quantities d_i, $i = 1, \ldots, n$, generated by tridqds algorithm applied to L and U we have $d_1 = 1$ and*

$$\begin{aligned} d_{i+1} &= d_i \left(\frac{u_i}{\hat{u}_i} \right), \qquad i = 1, \ldots, n-2 \\ d_n &= \frac{u_n}{\hat{u}_n}. \end{aligned}$$

Proof. From lemma 6.3.6 and lemma 6.3.7, for $i = 3, \ldots, n-1$, we have

$$d_i = 1 - \frac{\hat{l}_{i-2} g_1^{i-1} + f_1^{i-2}}{\hat{l}_{i-2} \hat{u}_{i-1}} \qquad \text{and} \qquad \hat{u_i} = d_i u_i + \frac{\hat{l}_{i-1} g_1^i + f_1^{i-1}}{\hat{l}_{i-1}}.$$

Thus

$$\hat{u_i} = d_i u_i + \hat{u_i} \frac{\hat{l}_{i-1} g_1^i + f_1^{i-1}}{\hat{l}_{i-1} \hat{u_i}} = d_i u_i + \hat{u}_i (1 - d_{i+1}), \qquad i = 2, \ldots, n-2.$$

Solving this equation yields

$$\hat{u_i} = d_i u_i + \hat{u}_i - \hat{u}_i d_{i+1} \Leftrightarrow d_{i+1} = d_i \left(\frac{u_i}{\hat{u}_i} \right).$$

For $i = 1$ this result also holds. Using the same lemmas, we have

$$d_2 = 1 - \frac{l_1 + f_1^0}{\hat{u}_1} \qquad \text{and} \qquad \hat{u_1} = u_1 + l_1 + f_1^0$$

since $d_1 = 1$ and $g_1^1 = l_1$. Then

$$d_2 = \frac{\hat{u}_1 - (\hat{u}_1 - u_1)}{\hat{u}_1} = \frac{u_1}{\hat{u}_1} = d_1 \left(\frac{u_1}{\hat{u}_1} \right).$$

For d_n the result is immediate. □

Next lemma show us the expressions for $\hat{l}_i$, $i = 1, \ldots, n-1$.

Lemma 6.3.8 *Consider $\hat{l}_i$, $i = 1, \ldots, n-1$, obtained when* tridqds *algorithm is applied to L and U. Then*

$$\begin{aligned}
\hat{l}_1 &= \frac{l_2 + u_2 - \hat{u}_1}{\hat{u}_1} f_1^0 + \frac{u_2}{\hat{u}_1} g_1^1 + \frac{g_2^1 + f_2^0}{\hat{u}_1}, \\
\hat{l}_i &= \frac{u_{i+1} + l_{i+1} - \hat{u}_i}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{u_{i+1}}{\hat{u}_i} g_1^i + \frac{\hat{l}_{i-1} - u_{i+2}}{\hat{l}_{i-1}\hat{u}_i} g_2^i, \qquad i = 2, \ldots, n-2, \\
\hat{l}_{n-1} &= \frac{u_n - \hat{u}_{n-1}}{\hat{l}_{n-2}\hat{u}_{n-1}} f_1^{n-2} + \frac{u_n}{\hat{u}_{n-1}} g_1^{n-1}.
\end{aligned}$$

Proof. For $\hat{l}_1$ we have

$$\begin{aligned}
\hat{l}_1 &= -f_1^0 + \left[\left(g_2^1 + f_2^0 + f_1^0 * l_2\right)/\hat{u}_1\right] + \left[\left(g_1^1 + f_1^0\right)/\hat{u}_1\right] * u_2 \\
&= \frac{l_2 + u_2 - \hat{u}_1}{\hat{u}_1} f_1^0 + \frac{u_2}{\hat{u}_1} g_1^1 + \frac{g_2^1 + f_2^0}{\hat{u}_1}.
\end{aligned}$$

To calculate $\hat{l}_i$, $i = 2, \ldots, n-2$ we need to follow

$$\begin{aligned}
f_1^i &\longleftarrow -f_1^{i-1}/\hat{l}_{i-1} \\
f_2^i &\longleftarrow -f_2^{i-1}/\hat{l}_{i-1} \\
d_{i+1} &\longleftarrow g_1^i - f_1^i \\
g_1^{i+1} &\longleftarrow g_2^i - f_2^i - f_1^i * l_{i+1} \\
d_{i+1} &\longleftarrow d_{i+1}/\hat{u}_i \\
g_1^{i+1} &\longleftarrow g_1^{i+1}/\hat{u}_i \\
\hat{l}_i &\longleftarrow f_1^i + g_1^{i+1} + d_{i+1} * u_{i+1}.
\end{aligned}$$

This way,

$$\begin{aligned}
\hat{l}_i &= \left(-f_1^{i-1}/\hat{l}_{i-1}\right) + \left\{\left[g_2^i - \left(-f_2^{i-1}/\hat{l}_{i-1}\right) - \left(-f_1^{i-1}/\hat{l}_{i-1}\right) * l_{i+1}\right]/\hat{u}_i\right\} + \\
&\quad + \left\{\left[g_1^i - \left(-f_1^{i-1}/\hat{l}_{i-1}\right)\right]/\hat{u}_i\right\} * u_{i+1} \\
&= -\frac{\hat{u}_i f_1^{i-1}}{\hat{l}_{i-1}\hat{u}_i} + \frac{\hat{l}_{i-1} g_2^i + f_2^{i-1} + l_{i+1} f_1^{i-1}}{\hat{l}_{i-1}\hat{u}_i} + \frac{\hat{l}_{i-1} g_1^i + f_1^{i-1}}{\hat{l}_{i-1}\hat{u}_i} u_{i+1} \\
&= \frac{u_{i+1} + l_{i+1} - \hat{u}_i}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{u_{i+1}}{\hat{u}_i} g_1^i + \frac{f_2^{i-1} + \hat{l}_{i-1} g_2^i}{\hat{l}_{i-1}\hat{u}_i}.
\end{aligned}$$

From lemma 6.3.4 we know that, for $i = 2, \ldots, n-2$,

$$f_2^{i-1} = -u_{i+2} g_2^i.$$

So,

$$\hat{l}_i = \frac{u_{i+1} + l_{i+1} - \hat{u}_i}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{u_{i+1}}{\hat{u}_i} g_1^i + \frac{\hat{l}_{i-1} - u_{i+2}}{\hat{l}_{i-1}\hat{u}_i} g_2^i.$$

To compute $\hat{l}_{n-1}$ we have

$$\begin{aligned} f_1^{n-1} &\longleftarrow -f_1^{n-2}/\hat{l}_{n-2} \\ d_n &\longleftarrow g_1^{n-1} - f_1^{n-1} \\ d_n &\longleftarrow d_n/\hat{u}_{n-1} \\ \hat{l}_{n-1} &\longleftarrow f_1^{n-1} + d_n * u_n. \end{aligned}$$

Thus,

$$\begin{aligned} \hat{l}_{n-1} &= \left(-f_1^{n-2}/\hat{l}_{n-2}\right) + \left\{ \left[g_1^{n-1} - \left(-f_1^{n-2}/\hat{l}_{n-2}\right) \right] /\hat{u}_{n-1} \right\} * u_n \\ &= \frac{u_n - \hat{u}_{n-1}}{\hat{l}_{n-2}\hat{u}_{n-1}} f_1^{n-2} + \frac{u_n}{\hat{u}_{n-1}} g_1^{n-1}. \ \square \end{aligned}$$

But these expressions for $\hat{l}_i$, $i = 1, \ldots, n-1$, can still be simplified.

Lemma 6.3.9 *Consider $\hat{l}_i$, $i = 1, \ldots, n-1$, obtained when* tridqds *algorithm is applied to L and U. Then*

$$\begin{aligned} \hat{l}_1 &= u_2(1 - d_2) + (l_2 - g_1^2) - f_1^0, \\ \hat{l}_i &= u_{i+1}(1 - d_{i+1}) + \left(l_{i+1} - g_1^{i+1}\right) - \frac{f_1^{i-1}}{\hat{l}_{i-1}}, \qquad i = 2, \ldots, n-2, \\ \hat{l}_{n-1} &= u_n(1 - d_n) - \frac{f_1^{n-2}}{\hat{l}_{n-2}}. \end{aligned}$$

Proof. By previous lemma, for $i = 2, \ldots, n-2$,

$$\begin{aligned} \hat{l}_i &= \frac{u_{i+1} + l_{i+1} - \hat{u}_i}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{u_{i+1}}{\hat{u}_i} g_1^i + \frac{\hat{l}_{i-1} - u_{i+2}}{\hat{l}_{i-1}\hat{u}_i} g_2^i \\ &= \frac{u_{i+1}}{\hat{u}_i} \left(\frac{f_1^{i-1}}{\hat{l}_{i-1}} + g_1^i \right) + \frac{l_{i+1} - \hat{u}_i}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{\hat{l}_{i-1} - u_{i+2}}{\hat{l}_{i-1}\hat{u}_i} g_2^i. \end{aligned} \tag{6.22}$$

From lemma 6.3.7 and theorem 6.3.1, we can write

$$\frac{u_{i+1}}{\hat{u}_i}(\hat{u_i} - d_i u_i) = u_{i+1}\left(1 - d_i \frac{u_i}{\hat{u}_i}\right) = u_{i+1}(1 - d_{i+1}). \tag{6.23}$$

Now the last term in (6.22). We have

$$\frac{\hat{l}_{i-1} - u_{i+2}}{\hat{l}_{i-1}\hat{u}_i} g_2^i = \frac{g_2^i}{\hat{u}_i} + \frac{-u_{i+2} g_2^i}{\hat{l}_{i-1}\hat{u}_i} = \frac{g_2^i}{\hat{u}_i} + \frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i},$$

since, from lemma 6.3.4, it holds

$$f_2^{i-1} = -u_{i+2} g_2^i.$$

Then joining the last two terms in (6.22) gives

$$\begin{aligned}
\frac{l_{i+1} - \hat{u}_i}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{\hat{l}_{i-1} - u_{i+2}}{\hat{l}_{i-1}\hat{u}_i} g_2^i &= \frac{l_{i+1}}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} - \frac{f_1^{i-1}}{\hat{l}_{i-1}} + \frac{g_2^i}{\hat{u}_i} + \frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} \\
&= \left(\frac{l_{i+1}}{\hat{l}_{i-1}\hat{u}_i} f_1^{i-1} + \frac{f_2^{i-1}}{\hat{l}_{i-1}\hat{u}_i} + \frac{g_2^i}{\hat{u}_i}\right) - \frac{f_1^{i-1}}{\hat{l}_{i-1}} \\
&= (l_{i+1} - g_1^{i+1}) - \frac{f_1^{i-1}}{\hat{l}_{i-1}},
\end{aligned} \tag{6.24}$$

since, from lemma 6.3.3,

$$g_1^i = l_i - \left(\frac{l_i}{\hat{l}_{i-2}\hat{u}_{i-1}} f_1^{i-2} + \frac{f_2^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}} + \frac{g_2^{i-1}}{\hat{u}_{i-1}}\right).$$

Finally, from (6.22), (6.23) and (6.24), we obtain

$$\hat{l_i} = u_{i+1}(1 - d_{i+1}) + \left(l_{i+1} - g_1^{i+1}\right) - \frac{f_1^{i-1}}{\hat{l}_{i-1}}.$$

For $i = n - 1$,

$$\begin{aligned}
\hat{l}_{n-1} &= \frac{u_n - \hat{u}_{n-1}}{\hat{l}_{n-2}\hat{u}_{n-1}} f_1^{n-2} + \frac{u_n}{\hat{u}_{n-1}} g_1^{n-1} \\
&= \frac{u_n}{\hat{u}_{n-1}}\left(\frac{f_1^{n-2}}{\hat{l}_{n-2}} + g_1^{n-1}\right) - \frac{f_1^{n-2}}{\hat{l}_{n-2}} \\
&= \frac{u_n}{\hat{u}_{n-1}}\left(\hat{u}_{n-1} - d_{n-1} u_{n-1}\right) - \frac{f_1^{n-2}}{\hat{l}_{n-2}} \\
&= u_n(1 - d_n) - \frac{f_1^{n-2}}{\hat{l}_{n-2}}.
\end{aligned}$$

The case $i = 1$:

$$\begin{aligned}
\hat{l}_1 &= \frac{l_2 + u_2 - \hat{u}_1}{\hat{u}_1} f_1^0 + \frac{u_2}{\hat{u}_1} g_1^1 + \frac{g_2^1 + f_2^0}{\hat{u}_1} \\
&= \frac{u_2}{\hat{u}_1} \left(f_1^0 + g_1^1 \right) + \frac{l_2 - \hat{u}_1}{\hat{u}_1} f_1^0 + \frac{g_2^1 + f_2^0}{\hat{u}_1} \\
&= \frac{u_2}{\hat{u}_1} (\hat{u}_1 - d_1 u_1) + \left(\frac{l_2}{\hat{u}_1} f_1^0 + \frac{g_2^1}{\hat{u}_1} + \frac{f_2^0}{\hat{u}_1} \right) - f_1^0 \\
&= u_2(1 - d_2) + \left(l_2 - g_1^2 \right) - f_1^0,
\end{aligned}$$

using lemma 6.3.7, theorem 6.3.1 and lemma 6.3.3. □

Combining all these results we obtain the new version of tridqds algorithm that we will call 3dqds. We introduce a new temporary variable aux.

The changes to the first minor step:

Minor step 1 :

$$\begin{aligned}
& d_1 = 1; \;\; g_1^1 = l_1; \;\; g_2^1 = 0 \\
& f_1^0 = (u_1 + l_1)^2 + u_2 l_1 - 2(\Re\sigma)(u_1 + l_1) + |\sigma|^2 \\
& f_2^0 = u_2 l_1 u_3 l_2 / f_1^0 \\
& f_1^0 = u_2 l_1 \left(u_1 + l_1 + u_2 + l_2 - 2(\Re\sigma) \right) / f_1^0 \\
& \hat{u}_1 = d_1 u_1 + g_1^1 + f_1^0 \\
& d_2 = d_1 u_1 / \hat{u}_1 \\
& aux = \left(l_2 f_1^0 + f_2^0 + g_2^1 \right) / \hat{u}_1 \\
& g_1^2 = l_2 - aux \\
& \hat{l}_1 = u_2(1 - d_2) + aux - f_1^0 \\
& g_2^2 = -l_3 f_2^0 / \hat{u}_1 \\
& f_1^1 = u_3 . aux - g_2^2 - f_2^0 \\
& f_2^1 = -u_4 g_2^1
\end{aligned}$$

The inner loop is changed to:

for $i = 2, \ldots, n-3$

$\hat{u}_i = d_i u_i + g_1^i + f_1^{i-1}/\hat{l}_{i-1}$

$d_{i+1} = d_i u_i / \hat{u}_i$

$aux = (l_{i+1} f_1^{i-1} + f_2^{i-1} + \hat{l}_{i-1} g_2^i)/(\hat{l}_{i-1}\hat{u}_i)$

$g_1^{i+1} = l_{i+1} - aux$

$\hat{l}_i = u_{i+1}(1 - d_{i+1}) + aux - f_1^{i-1}/\hat{l}_{i-1}$

$g_2^{i+1} = -l_{i+2} f_2^{i-1}/(\hat{l}_{i-1}\hat{u}_i)$

$f_1^i = u_{i+2}.aux - g_2^{i+1} - f_2^{i-1}/\hat{l}_{i-1}$

$f_2^i = -u_{i+3} g_2^{i+1}$

end for

The last three steps become:

Minor step n-2 :

$\hat{u}_{n-2} = d_{n-2} u_{n-2} + g_1^{n-2} + f_1^{n-3}/\hat{l}_{n-3}$

$d_{n-1} = d_{n-2} u_{n-2} / \hat{u}_{n-2}$

$aux = (l_{n-1} f_1^{n-3} + f_2^{n-3} + \hat{l}_{n-3} g_2^{n-2})/(\hat{l}_{n-3}\hat{u}_{n-2})$

$g_1^{n-1} = l_{n-1} - aux$

$\hat{l}_{n-2} = u_{n-1}(1 - d_{n-1}) + aux - f_1^{n-3}/\hat{l}_{n-3}$

$f_1^{n-2} = u_n.aux - f_2^{n-3}/\hat{l}_{n-3}$

Minor step n-1 :

$\hat{u}_{n-1} = d_{n-1} u_{n-1} + g_1^{n-1} + f_1^{n-2}/\hat{l}_{n-2}$

$d_n = 1 - (g_1^{n-1} + f_1^{n-2}/\hat{l}_{n-2})/\hat{u}_{n-1}$

$\hat{l}_{n-1} = u_n(1 - d_n) - f_1^{n-2}/\hat{l}_{n-2}$

Minor step n :

$\hat{u}_n = d_n * u_n$

Now, turning into a computacional implementation we will go back again to variables x_l and y_l to play the role of f_1^i and f_2^i, respectively, and to variables y_r and z_r to replace g_1^i and g_2^i, respectively. Variable d_i will substitute x_r. Recognizing common subexpressions, we will introduce some changes with the goal of reducing the number of divisions needed. If we let

$$\bar{l} = \frac{1}{\hat{l}_{i-1}} \qquad \text{and} \qquad \bar{u} = \frac{1}{\hat{u}_i}$$

and initially

$$\begin{aligned} x_l &= f_1^{i-1} & y_r &= g_1^i \\ y_l &= f_2^{i-1} & z_r &= g_2^i \end{aligned}$$

then the inner loop will be

for $i = 2, \ldots, n-3$

$x_l = x_l * \bar{l}$

$y_l = y_l * \bar{l}$

$\hat{u}_i = d_i * u_i + y_r + x_l$

$\bar{u} = 1/\hat{u}_i$

$d_{i+1} = d_i * u_i * \bar{u}$

$aux = (l_{i+1} * x_l + y_l + z_r) * \bar{u}$

$y_r = l_{i+1} - aux$

$\hat{l}_i = u_{i+1} - d_{i+1} * u_{i+1} + aux - x_l$

$z_r = -l_{i+2} * y_l * \bar{u}$

$x_l = u_{i+2} * aux - z_r - y_l$

$y_l = -u_{i+3} * z_r$

$\bar{l} = 1/\hat{l}_i$

end for

So, we incorporated two additional auxiliary variables but we reduced the number of divisions to only 2.

It is still possible to reduce the number or multiplications by noticing that the multiplication $d_i * u_i$ can be performed implicitly. If we define t_i by

$$t_i \equiv d_i * u_i$$

then

$$t_{i+1} = d_{i+1} * u_{i+1} = d_i * u_i * \bar{u} * u_{i+1} = t_i * \bar{u} * u_{i+1}.$$

Thus, $\hat{u}_i$ and $\hat{l}_i$ will be given by

$$\begin{aligned} \hat{u_i} &= t_i + y_r + x_l \\ t_{i+1} &= t_i * \bar{u} * u_{i+1} \\ \hat{l_i} &= u_{i+1} - t_{i+1} + aux - x_l. \end{aligned}$$

It can be verified that there is no need for both variables $\bar{l}$ and $\bar{u}$. The variable aux_1 will play the role of both. The final version of 3dqds algorithm is presented below.

3dqds(σ) :

$t = 1; \;\; y_r = l_1; \;\; z_r = 0;$

$x_l = (u_1 + l_1)^2 + u_2 * l_1 - 2(\Re\sigma) * (u_1 + l_1) + |\sigma|^2$

$y_l = u_2 * l_1 * u_3 * l_2 / x_l$

$x_l = u_2 * l_1 * \big(u_1 + l_1 + u_2 + l_2 - 2(\Re\sigma)\big)/x_l$

$t = t * u_1$

$\hat{u}_1 = t + y_r + x_l$

$aux_1 = 1/\hat{u}_1$

$aux = (l_2 * x_l + y_l + z_r) * aux_1$

$y_r = l_2 - aux$

$t = t * u_2 * aux_1$

$\hat{l}_1 = u_2 - t + aux - x_l$

$z_r = -l_3 * y_l * aux_1$

$x_l = u_3 * aux - z_r - y_l$

$y_l = -u_4 * z_r$

$aux_1 = 1/\hat{l}_1$

for $i = 2, \ldots, n-3$

$x_l = x_l * aux_1$

$y_l = y_l * aux_1$

$\hat{u}_i = t + y_r + x_l$

$aux_1 = 1/\hat{u}_i$

$aux = (l_{i+1} * x_l + y_l + z_r) * aux_1$

$y_r = l_{i+1} - aux$

$t = u_{i+1} * t * aux_1$

$\hat{l}_i = u_{i+1} - t + aux - x_l$

$z_r = -l_{i+2} * y_l * aux_1$

$x_l = u_{i+2} * aux - z_r - y_l$

$y_l = -u_{i+3} * z_r$

$aux_1 = 1/\hat{l}_i$

end for

$x_l = x_l * aux_1$

$y_l = y_l * aux_1$

$\hat{u}_{n-2} = t + y_r + x_l$

$aux_1 = 1/\hat{u}_{n-2}$

$aux = (l_{n-1} * x_l + y_l + z_r) * aux_1$

$y_r = l_{n-1} - aux$

$t = u_{n-1} * t * aux_1$

$\hat{l}_{n-2} = u_{n-1} - t + aux - x_l$

$x_l = u_n * aux - y_l$

$aux_1 = 1/\hat{l}_{n-2}$

$$x_l = x_l * aux_1$$
$$\hat{u}_{n-1} = t + y_r + x_l$$
$$aux_1 = (y_r + x_l)/\hat{u}_{n-1}$$
$$\hat{l}_{n-1} = u_n * aux_1 - x_l$$

$$\hat{u}_n = (1 - aux_1) * u_n$$

6.3.3 Operation count for 3dqds

Table 6.1 below shows that the more elegant version 3dqds of triple dqds algorithm has the advantage of performing less three divisions than the version tridqds. The price to pay for this reduction on the number of divisions is worthwhile considering that in certain machines the cost of a division is much more expensive than the cost of a multiplication.

	tridqds	3dqds
Divisions	5	2
Multiplications	6	10
Additions	5	5
Subtractions	6	5
Assignments	16	12
Auxiliary variables	5	7

Figure 6.1: Operation Count of tridqds and 3dqds

The simplicity of 3dqds is very attractive and preliminary numerical tests reveal its robustness.

6.3.4 Entries of $(UL)^{-1}$

In this section we will present results for the entries of $(UL)^{-1}$ analogous to the results shown for dqd in theorem 6.2.4 (page 176). So, the goal is to obtain expressions for

$$\left[(UL)^{-1}\right]_{i:i+1,i:i+1} = \begin{bmatrix} \left[(UL)^{-1}\right]_{i,i} & \left[(UL)^{-1}\right]_{i,i+1} \\ \left[(UL)^{-1}\right]_{i+1,i} & \left[(UL)^{-1}\right]_{i+1,i+1} \end{bmatrix}, \qquad i = 1, \ldots, n-1,$$

when we apply triple dqds.

The first lemma gives the expression for entries $(i-2,i)$, $(i-1,i)$ and (i,i) of $\left(F^{(i)}G^{(i)}\right)^{-1}$ that will be used to obtain the diagonal of $(UL)^{-1}$. See page 179 for recalling the properties of the matrices $F^{(i)}$ and $G^{(i)}$.

Lemma 6.3.10 *Consider $F^{(i)}$ and $G^{(i)}$ at the beginning of minor step i, $i = 1, \ldots, n$, when we apply tridqds algorithm to L and U. If U is invertible then*

$$\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{ii} = \frac{1}{d_i u_i}, \qquad i = 1, \ldots, n \tag{6.25}$$

$$\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-1,i} = -\frac{1}{d_i u_i \hat{u}_{i-1}}, \qquad i = 2, \ldots, n \tag{6.26}$$

$$\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-2,i} = \frac{1}{d_i u_i \hat{u}_{i-1} \hat{u}_{i-2}}, \qquad i = 3, \ldots, n \tag{6.27}$$

Proof. We start by using the interesting fact that row i of $G^{(i)}$ and column i of $F^{(i)}$ are singletons (see (6.17)). We have, for $i = 1, \ldots, n$,

$$F^{(i)}\boldsymbol{e}_i = u_i \boldsymbol{e}_i \qquad \text{and} \qquad \boldsymbol{e}_i^T G^{(i)} = d_i \boldsymbol{e}_i^T$$

and, equivalently,

$$u_i^{-1}\boldsymbol{e}_i = \left(F^{(i)}\right)^{-1}\boldsymbol{e}_i \qquad \text{and} \qquad d_i^{-1}\boldsymbol{e}_i^T = \boldsymbol{e}_i^T\left(G^{(i)}\right)^{-1}.$$

Therefore,

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{ii} &= \boldsymbol{e}_i^T\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]\boldsymbol{e}_i \\ &= \left[\boldsymbol{e}_i^T\left(G^{(i)}\right)^{-1}\right]\left[\left(F^{(i)}\right)^{-1}\boldsymbol{e}_i\right] \\ &= d_i^{-1}\boldsymbol{e}_i^T u_i^{-1}\boldsymbol{e}_i \\ &= \frac{1}{d_i u_i}.\end{aligned}$$

Analogously, for $i = 2, \ldots, n$, we can write

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-1,i} &= \boldsymbol{e}_{i-1}^T\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]\boldsymbol{e}_i \\ &= \left[\boldsymbol{e}_{i-1}^T\left(G^{(i)}\right)^{-1}\right]\left[\left(F^{(i)}\right)^{-1}\boldsymbol{e}_i\right] \\ &= \left[\boldsymbol{e}_{i-1}^T\left(G^{(i)}\right)^{-1}\right]u_i^{-1}\boldsymbol{e}_i. \qquad (6.28)\end{aligned}$$

But now row $(i-1)$ of $G^{(i)}$ is not a singleton. We have

$$\boldsymbol{e}_{i-1}^T\left(G^{(i)}\right) = \hat{u}_{i-1}\boldsymbol{e}_{i-1}^T + \boldsymbol{e}_i^T.$$

Then

$$\begin{aligned}\boldsymbol{e}_{i-1}^T &= \left(\hat{u}_{i-1}\boldsymbol{e}_{i-1}^T + \boldsymbol{e}_i^T\right)\left(G^{(i)}\right)^{-1} \\ &= \hat{u}_{i-1}\boldsymbol{e}_{i-1}^T\left(G^{(i)}\right)^{-1} + \boldsymbol{e}_i^T\left(G^{(i)}\right)^{-1} \\ &= \hat{u}_{i-1}\boldsymbol{e}_{i-1}^T\left(G^{(i)}\right)^{-1} + d_i^{-1}\boldsymbol{e}_i^T.\end{aligned}$$

And

$$\boldsymbol{e}_{i-1}^T\left(G^{(i)}\right)^{-1} = \left(\boldsymbol{e}_{i-1}^T - d_i^{-1}\boldsymbol{e}_i^T\right)\hat{u}_{i-1}^{-1}. \qquad (6.29)$$

Thus, from (6.28), we get

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-1,i} &= \left[\left(\boldsymbol{e}_{i-1}^T - d_i^{-1}\boldsymbol{e}_i^T\right)\hat{u}_{i-1}^{-1}\right]u_i^{-1}\boldsymbol{e}_i \\ &= -d_i^{-1}\hat{u}_{i-1}^{-1}u_i^{-1} \\ &= -\frac{1}{d_i u_i \hat{u}_{i-1}},\end{aligned}$$

as we wanted to show.

Finally, for $i = 3, \ldots, n$,

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-2,i} &= \boldsymbol{e}_{i-2}^T\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]\boldsymbol{e}_i \\ &= \left[\boldsymbol{e}_{i-2}^T\left(G^{(i)}\right)^{-1}\right]\left[\left(F^{(i)}\right)^{-1}\boldsymbol{e}_i\right] \\ &= \left[\boldsymbol{e}_{i-2}^T\left(G^{(i)}\right)^{-1}\right]u_i^{-1}\boldsymbol{e}_i. \end{aligned} \tag{6.30}$$

We have

$$\boldsymbol{e}_{i-2}^T\left(G^{(i)}\right) = \hat{u}_{i-2}\boldsymbol{e}_{i-2}^T + \boldsymbol{e}_{i-1}^T \tag{6.31}$$

and then

$$\begin{aligned}\boldsymbol{e}_{i-2}^T\left(G^{(i)}\right)^{-1} &= \left(\boldsymbol{e}_{i-2}^T - \boldsymbol{e}_{i-1}^T\left(G^{(i)}\right)^{-1}\right)\hat{u}_{i-2}^{-1} \\ &= \left[\boldsymbol{e}_{i-2}^T - \left(\boldsymbol{e}_{i-1}^T - d_i^{-1}\boldsymbol{e}_i^T\right)\hat{u}_{i-1}^{-1}\right]\hat{u}_{i-2}^{-1}. \end{aligned} \tag{6.32}$$

From (6.30) we get

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-2,i} &= \left[\boldsymbol{e}_{i-2}^T - \left(\boldsymbol{e}_{i-1}^T - d_i^{-1}\boldsymbol{e}_i^T\right)\hat{u}_{i-1}^{-1}\right]\hat{u}_{i-2}^{-1}u_i^{-1}\boldsymbol{e}_i \\ &= d_i^{-1}\hat{u}_{i-1}^{-1}\hat{u}_{i-2}^{-1}u_i^{-1} \\ &= \frac{1}{d_i u_i \hat{u}_{i-1}\hat{u}_{i-2}}. \ \square \end{aligned}$$

The next theorem gives the expression for the diagonal of $(UL)^{-1}$.

In (6.16) we already saw that

$$\begin{aligned}F^{(i)}G^{(i)} &= \left(\mathcal{L}_{i-1}^{-1}\ldots\mathcal{L}_2^{-1}\mathcal{L}_1^{-1}\right)(UL)\left(\mathcal{L}_1\mathcal{L}_2\ldots\mathcal{L}_{i-1}\right) \\ &= \left(\mathcal{L}_1\mathcal{L}_2\ldots\mathcal{L}_{i-1}\right)^{-1}(UL)(\mathcal{L}_1\mathcal{L}_2\ldots\mathcal{L}_{i-1}). \end{aligned}$$

Then

$$UL = (\mathcal{L}_1\mathcal{L}_2\ldots\mathcal{L}_{i-1})F^{(i)}G^{(i)}(\mathcal{L}_1\mathcal{L}_2\ldots\mathcal{L}_{i-1})^{-1}.$$

and

$$(UL)^{-1} = (\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_{i-1})\left(F^{(i)}G^{(i)}\right)^{-1}(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_{i-1})^{-1}. \tag{6.33}$$

Theorem 6.3.2 *Consider L and U as described in (6.1). If U is invertible and* tridqds *algorithm applied to L and U is successful, then the quantities d_i, f_1^{i-1} and f_2^{i-1} generated by the algorithm satisfy*

$$\begin{aligned}
\left[(UL)^{-1}\right]_{1,1} =& \frac{1}{d_1u_1}\\
\left[(UL)^{-1}\right]_{2,2} =& \frac{1}{d_2u_2}\left(1-\frac{f_1^0}{\hat{u}_1}\right)\\
\left[(UL)^{-1}\right]_{3,3} =& \frac{1}{d_3u_3}\left(1-\frac{f_1^1}{\hat{l}_1\hat{u}_2}+\frac{f_2^0}{\hat{u}_2\hat{u}_1}\right)\\
\left[(UL)^{-1}\right]_{i,i} =& \frac{1}{d_iu_i}\left(1-\frac{f_1^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}}+\frac{f_2^{i-3}}{\hat{l}_{i-3}\hat{u}_{i-1}\hat{u}_{i-2}}\right), \qquad i=4,\dots,n.
\end{aligned}$$

Proof. Using (6.19), (6.20) and (6.33), for $i = 4,\dots,n$,

$$\begin{aligned}
\left[(UL)^{-1}\right]_{i,i} &= \boldsymbol{e}_i^T\left[(UL)^{-1}\right]\boldsymbol{e}_i\\
&= \boldsymbol{e}_i^T(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_{i-1})\left(F^{(i)}G^{(i)}\right)^{-1}(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_{i-1})^{-1}\boldsymbol{e}_i\\
&= \left[\boldsymbol{e}_i^T(\mathcal{L}_1\mathcal{L}_2\dots\mathcal{L}_{i-1})\right]\left[\left(F^{(i)}G^{(i)}\right)^{-1}\boldsymbol{e}_i\right]\\
&= \begin{bmatrix}0 & \dots & 0 & h_2^{i-2} & h_1^{i-1} & 1 & 0 & \dots & 0\end{bmatrix}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\boldsymbol{e}_i\right]\\
&= (h_2^{i-2}\boldsymbol{e}_{i-2}^T + h_1^{i-1}\boldsymbol{e}_{i-1}^T + \boldsymbol{e}_i^T)\left[\left(F^{(i)}G^{(i)}\right)^{-1}\boldsymbol{e}_i\right]\\
&= h_2^{i-2}\boldsymbol{e}_{i-2}^T\left(F^{(i)}G^{(i)}\right)^{-1}\boldsymbol{e}_i + h_1^{i-1}\boldsymbol{e}_{i-1}^T\left(F^{(i)}G^{(i)}\right)^{-1}\boldsymbol{e}_i + \boldsymbol{e}_i^T\left(F^{(i)}G^{(i)}\right)^{-1}\boldsymbol{e}_i\\
&= h_2^{i-2}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-2,i} + h_1^{i-1}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-1,i} + \left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i,i}.
\end{aligned}$$

Using lemma 6.3.10 yields

$$\begin{aligned}
\left[(UL)^{-1}\right]_{i,i} &= \frac{h_2^{i-2}}{d_iu_i\hat{u}_{i-1}\hat{u}_{i-2}} - \frac{h_1^{i-1}}{d_iu_i\hat{u}_{i-1}} + \frac{1}{d_iu_i}\\
&= \frac{1}{d_iu_i}\left(1-\frac{h_1^{i-1}}{\hat{u}_{i-1}}+\frac{h_2^{i-2}}{\hat{u}_{i-1}\hat{u}_{i-2}}\right).
\end{aligned} \tag{6.34}$$

Finally, by lemma 6.3.1, we have

$$h_1^{i-1} = f_1^{i-2}/\hat{l}_{i-2} \qquad \text{and} \qquad h_2^{i-2} = f_2^{i-3}/\hat{l}_{i-3},$$

and then

$$\left[(UL)^{-1}\right]_{i,i} = \frac{1}{d_i u_i}\left(1 - \frac{f_1^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}} + \frac{f_2^{i-3}}{\hat{l}_{i-3}\hat{u}_{i-1}\hat{u}_{i-2}}\right).$$

Now let us see the cases $i = 1, 2, 3$. By lemma 6.3.10, we have

$$\left[(UL)^{-1}\right]_{1,1} = \left[\left(F^{(1)}G^{(1)}\right)^{-1}\right]_{11} = \frac{1}{d_1 u_1}.$$

For $i = 2$, using lemmas 6.3.10 and 6.3.1 again,

$$\begin{aligned}
\left[(UL)^{-1}\right]_{2,2} &= \boldsymbol{e}_2^T\left[(UL)^{-1}\right]\boldsymbol{e}_2 \\
&= \boldsymbol{e}_2^T\mathcal{L}_1\left(F^{(2)}G^{(2)}\right)^{-1}\mathcal{L}_1^{-1}\boldsymbol{e}_2 \\
&= \boldsymbol{e}_2^T\mathcal{L}_1\left[\left(F^{(2)}G^{(2)}\right)^{-1}\boldsymbol{e}_2\right] \\
&= \begin{bmatrix} h_1^1 & 1 & 0 & \dots & 0 \end{bmatrix}\left[\left(F^{(2)}G^{(2)}\right)^{-1}\boldsymbol{e}_2\right] \\
&= \left(h_1^1\boldsymbol{e}_1^T + \boldsymbol{e}_2^T\right)\left[\left(F^{(2)}G^{(2)}\right)^{-1}\boldsymbol{e}_2\right] \\
&= h_1^1\boldsymbol{e}_1^T\left(F^{(2)}G^{(2)}\right)^{-1}\boldsymbol{e}_2 + \boldsymbol{e}_2^T\left(F^{(2)}G^{(2)}\right)^{-1}\boldsymbol{e}_2 \\
&= h_1^1\left[\left(F^{(2)}G^{(2)}\right)^{-1}\right]_{1,2} + \left[\left(F^{(2)}G^{(2)}\right)^{-1}\right]_{2,2} \\
&= -\frac{h_1^1}{d_2 u_2 \hat{u}_1} + \frac{1}{d_2 u_2} \\
&= \frac{1}{d_2 u_2}\left(1 - \frac{f_1^0}{\hat{u}_1}\right).
\end{aligned}$$

Finally, for $i = 3$, (6.34) also yields,

$$\left[(UL)^{-1}\right]_{3,3} = \frac{1}{d_3 u_3}\left(1 - \frac{h_1^2}{\hat{u}_2} + \frac{h_2^1}{\hat{u}_2\hat{u}_1}\right),$$

and by lemma 6.3.1 we have

$$h_1^2 = f_1^1/\hat{l}_1 \qquad \text{and} \qquad h_2^1 = f_2^0.$$

So,

$$\left[(UL)^{-1}\right]_{3,3} = \frac{1}{d_3 u_3}\left(1 - \frac{f_1^1}{\hat{l}_1\hat{u}_2} + \frac{f_2^0}{\hat{u}_2\hat{u}_1}\right). \ \square$$

Next lemma will drive us to the off-diagonal elements of $(UL)^{-1}$.

Lemma 6.3.11 *Consider $F^{(i)}$ and $G^{(i)}$ at the beginning of minor step i, $i = 1, \ldots, n$, when we apply* tridqds *algorithm to L and U. If U is invertible then*

$$\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i+1,i} = -\frac{g_1^i}{d_i u_i}, \qquad i = 1, \ldots, n-1$$

$$\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i,i+1} = -\frac{1}{d_i u_i u_{i+1}}, \qquad i = 1, \ldots, n-1$$

$$\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-1,i+1} = \frac{1}{d_i u_i u_{i+1}\hat{u}_{i-1}}, \qquad i = 2, \ldots, n-1$$
$$\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-2,i+1} = -\frac{1}{d_i u_i u_{i+1}\hat{u}_{i-1}\hat{u}_{i-2}}, \qquad i = 3, \ldots, n-1$$

Proof. For $i = 1, \ldots, n-1$, we can write

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i+1,i} &= \boldsymbol{e}_{i+1}^T\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]\boldsymbol{e}_i \\ &= \left[\boldsymbol{e}_{i+1}^T\left(G^{(i)}\right)^{-1}\right]\left[\left(F^{(i)}\right)^{-1}\boldsymbol{e}_i\right] \end{aligned} \tag{6.35}$$

We already know that

$$u_i^{-1}\boldsymbol{e}_i = \left(F^{(i)}\right)^{-1}\boldsymbol{e}_i \qquad \text{and} \qquad d_i^{-1}\boldsymbol{e}_i^T = \boldsymbol{e}_i^T\left(G^{(i)}\right)^{-1},$$

and for row $(i+1)$ of $G^{(i)}$ we have

$$\boldsymbol{e}_{i+1}^T\left(G^{(i)}\right) = g_1^i\boldsymbol{e}_i^T + \boldsymbol{e}_{i+1}^T.$$

Then

$$\begin{aligned}\boldsymbol{e}_{i+1}^T &= \left(g_1^i \boldsymbol{e}_i^T + \boldsymbol{e}_{i+1}^T\right)\left(G^{(i)}\right)^{-1} \\ &= g_1^i \boldsymbol{e}_i^T \left(G^{(i)}\right)^{-1} + \boldsymbol{e}_{i+1}^T \left(G^{(i)}\right)^{-1} \\ &= g_1^i d_i^{-1} \boldsymbol{e}_i^T + \boldsymbol{e}_{i+1}^T \left(G^{(i)}\right)^{-1}.\end{aligned}$$

And

$$\boldsymbol{e}_{i+1}^T \left(G^{(i)}\right)^{-1} = \boldsymbol{e}_{i+1}^T - g_1^i d_i^{-1} \boldsymbol{e}_i^T.$$

Thus, from (6.35) we get

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i+1,i} &= \left(\boldsymbol{e}_{i+1}^T - g_1^i d_i^{-1} \boldsymbol{e}_i^T\right) u_i^{-1} \boldsymbol{e}_i \\ &= -\frac{g_1^i}{d_i u_i}.\end{aligned}$$

For $i = 1, \ldots, n-1$, we have

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i,i+1} &= \boldsymbol{e}_i^T \left[\left(F^{(i)}G^{(i)}\right)^{-1}\right] \boldsymbol{e}_{i+1} \\ &= \left[\boldsymbol{e}_i^T \left(G^{(i)}\right)^{-1}\right]\left[\left(F^{(i)}\right)^{-1} \boldsymbol{e}_{i+1}\right] \\ &= d_i^{-1} \boldsymbol{e}_i^T \left[\left(F^{(i)}\right)^{-1} \boldsymbol{e}_{i+1}\right]. \qquad (6.36)\end{aligned}$$

Column $(i+1)$ of $F^{(i)}$ is

$$\left(F^{(i)}\right) \boldsymbol{e}_{i+1} = \boldsymbol{e}_i + u_{i+1} \boldsymbol{e}_{i+1}.$$

Then

$$\boldsymbol{e}_{i+1} = \left(F^{(i)}\right)^{-1} \boldsymbol{e}_i + u_{i+1} \left(F^{(i)}\right)^{-1} \boldsymbol{e}_{i+1} = u_i^{-1} \boldsymbol{e}_i + u_{i+1} \left(F^{(i)}\right)^{-1} \boldsymbol{e}_{i+1},$$

which gives

$$\left(F^{(i)}\right)^{-1} \boldsymbol{e}_{i+1} = (\boldsymbol{e}_{i+1} - u_i^{-1} \boldsymbol{e}_i) u_{i+1}^{-1}. \qquad (6.37)$$

Thus, from (6.36), we get

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i,i+1} &= d_i^{-1} \boldsymbol{e}_i^T (\boldsymbol{e}_{i+1} - u_i^{-1} \boldsymbol{e}_i) u_{i+1}^{-1} \\ &= -\frac{1}{d_i u_i u_{i+1}}.\end{aligned}$$

For $i = 2, \ldots, n-1$, using (6.29) and (6.37), we have

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-1,i+1} &= \boldsymbol{e}_{i-1}^T\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]\boldsymbol{e}_{i+1}\\ &= \left[\boldsymbol{e}_{i-1}^T\left(G^{(i)}\right)^{-1}\right]\left[\left(F^{(i)}\right)^{-1}\boldsymbol{e}_{i+1}\right]\\ &= \left(\boldsymbol{e}_{i-1}^T - d_i^{-1}\boldsymbol{e}_i^T\right)\hat{u}_{i-1}^{-1}\left(\boldsymbol{e}_{i+1} - u_i^{-1}\boldsymbol{e}_i\right)u_{i+1}^{-1}\\ &= \frac{1}{d_i u_i u_{i+1}\hat{u}_{i-1}}.\end{aligned}$$

Finally, for $i = 3, \ldots, n-1$, using (6.32) and (6.37) gives

$$\begin{aligned}\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-2,i+1} &= \boldsymbol{e}_{i-2}^T\left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]\boldsymbol{e}_{i+1}\\ &= \left[\boldsymbol{e}_{i-2}^T\left(G^{(i)}\right)^{-1}\right]\left[\left(F^{(i)}\right)^{-1}\boldsymbol{e}_{i+1}\right]\\ &= \left(\left[\boldsymbol{e}_{i-2}^T - \left(\boldsymbol{e}_{i-1}^T - d_i^{-1}\boldsymbol{e}_i^T\right)\hat{u}_{i-1}^{-1}\right]\hat{u}_{i-2}^{-1}\right)\left(\boldsymbol{e}_{i+1} - u_i^{-1}\boldsymbol{e}_i\right)u_{i+1}^{-1}\\ &= -\frac{1}{d_i u_i u_{i+1}\hat{u}_{i-1}\hat{u}_{i-2}}.\ \square\end{aligned}$$

Next theorem give the entries $(i, i+1)$ of $(UL)^{-1}$, $i = 1, \ldots, n-1$.

Theorem 6.3.3 *Consider L and U as described in (6.1). If U is invertible and* tridqds *algorithm applied to L and U is successful, then the quantities d_i, f_1^{i-1} and f_2^{i-1} generated by the algorithm satisfy*

$$\begin{aligned}\left[(UL)^{-1}\right]_{1,2} &= -\frac{1}{d_1 u_1 u_2},\\ \left[(UL)^{-1}\right]_{2,3} &= -\frac{1}{d_2 u_2 u_3}\left(1 - \frac{f_1^0}{\hat{u}_1}\right),\\ \left[(UL)^{-1}\right]_{3,4} &= -\frac{1}{d_3 u_3 u_4}\left(1 - \frac{f_1^1}{\hat{l}_1\hat{u}_2} + \frac{f_2^0}{\hat{u}_2\hat{u}_1}\right),\\ \left[(UL)^{-1}\right]_{i,i+1} &= -\frac{1}{d_i u_i u_{i+1}}\left(1 - \frac{f_1^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}} + \frac{f_2^{i-3}}{\hat{l}_{i-3}\hat{u}_{i-1}\hat{u}_{i-2}}\right), \qquad i = 4, \ldots, n-1.\end{aligned}$$

Proof. Using (6.19), (6.20) and (6.33), for $i = 4, \ldots, n$,

$$
\begin{aligned}
&\left[(UL)^{-1}\right]_{i,i+1} = \boldsymbol{e}_i^T \left[(UL)^{-1}\right] \boldsymbol{e}_{i+1} \\
&= \boldsymbol{e}_i^T (\mathcal{L}_1\mathcal{L}_2 \ldots \mathcal{L}_{i-1}) \left(F^{(i)}G^{(i)}\right)^{-1} (\mathcal{L}_1\mathcal{L}_2 \ldots \mathcal{L}_{i-1})^{-1} \boldsymbol{e}_{i+1} \\
&= \left[\boldsymbol{e}_i^T (\mathcal{L}_1\mathcal{L}_2 \ldots \mathcal{L}_{i-1})\right] \left[\left(F^{(i)}G^{(i)}\right)^{-1} \boldsymbol{e}_{i+1}\right] \\
&= \begin{bmatrix} 0 & \ldots & 0 & h_2^{i-2} & h_1^{i-1} & 1 & 0 & \ldots & 0 \end{bmatrix} \left[\left(F^{(i)}G^{(i)}\right)^{-1} \boldsymbol{e}_{i+1}\right] \\
&= (h_2^{i-2}\boldsymbol{e}_{i-2}^T + h_1^{i-1}\boldsymbol{e}_{i-1}^T + \boldsymbol{e}_i^T) \left[\left(F^{(i)}G^{(i)}\right)^{-1} \boldsymbol{e}_{i+1}\right] \\
&= h_2^{i-2}\boldsymbol{e}_{i-2}^T \left(F^{(i)}G^{(i)}\right)^{-1} \boldsymbol{e}_{i+1} + h_1^{i-1}\boldsymbol{e}_{i-1}^T \left(F^{(i)}G^{(i)}\right)^{-1} \boldsymbol{e}_{i+1} + \boldsymbol{e}_i^T \left(F^{(i)}G^{(i)}\right)^{-1} \boldsymbol{e}_{i+1} \\
&= h_2^{i-2} \left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-2,i+1} + h_1^{i-1} \left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i-1,i+1} + \left[\left(F^{(i)}G^{(i)}\right)^{-1}\right]_{i,i+1}.
\end{aligned}
$$

Using lemma 6.3.11 yields

$$
\begin{aligned}
\left[(UL)^{-1}\right]_{i,i+1} &= -\frac{h_2^{i-2}}{d_i u_i u_{i+1} \hat{u}_{i-1} \hat{u}_{i-2}} + \frac{h_1^{i-1}}{d_i u_i u_{i+1} \hat{u}_{i-1}} - \frac{1}{d_i u_i u_{i+1}} \\
&= -\frac{1}{d_i u_i u_{i+1}} \left(1 - \frac{h_1^{i-1}}{\hat{u}_{i-1}} + \frac{h_2^{i-2}}{\hat{u}_{i-1}\hat{u}_{i-2}}\right).
\end{aligned} \tag{6.38}
$$

Finally, by lemma 6.3.1, we have

$$
h_1^{i-1} = f_1^{i-2}/\hat{l}_{i-2} \qquad \text{and} \qquad h_2^{i-2} = f_2^{i-3}/\hat{l}_{i-3},
$$

and then

$$
\left[(UL)^{-1}\right]_{i,i+1} = -\frac{1}{d_i u_i u_{i+1}} \left(1 - \frac{f_1^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}} + \frac{f_2^{i-3}}{\hat{l}_{i-3}\hat{u}_{i-1}\hat{u}_{i-2}}\right).
$$

Now let us see the cases $i = 1, 2, 3$. Using lemma 6.3.11, we have

$$
\left[(UL)^{-1}\right]_{1,2} = \left[\left(F^1 G^{(1)}\right)^{-1}\right]_{1,2} = -\frac{1}{d_1 u_1 u_2}.
$$

For $i = 2$, lemmas 6.3.11 and 6.3.1 give

$$\begin{aligned}
\left[(UL)^{-1}\right]_{2,3} &= \boldsymbol{e}_2^T \left[(UL)^{-1}\right] \boldsymbol{e}_3 \\
&= \boldsymbol{e}_2^T \mathcal{L}_1 \left(F^{(2)} G^{(2)}\right)^{-1} \mathcal{L}_1^{-1} \boldsymbol{e_3} \\
&= \boldsymbol{e}_2^T \mathcal{L}_1 \left[\left(F^{(2)} G^{(2)}\right)^{-1} \boldsymbol{e}_3\right] \\
&= \begin{bmatrix} h_1^1 & 1 & 0 & \dots & 0 \end{bmatrix} \left[\left(F^{(2)} G^{(2)}\right)^{-1} \boldsymbol{e}_3\right] \\
&= \left(h_1^1 \boldsymbol{e}_1^T + \boldsymbol{e}_2^T\right) \left[\left(F^{(2)} G^{(2)}\right)^{-1} \boldsymbol{e}_3\right] \\
&= h_1^1 \boldsymbol{e}_1^T \left(F^{(2)} G^{(2)}\right)^{-1} \boldsymbol{e}_3 + \boldsymbol{e}_2^T \left(F^{(2)} G^{(2)}\right)^{-1} \boldsymbol{e}_3 \\
&= h_1^1 \left[\left(F^{(2)} G^{(2)}\right)^{-1}\right]_{1,3} + \left[\left(F^{(2)} G^{(2)}\right)^{-1}\right]_{2,3} \\
&= \frac{h_1^1}{d_2 u_2 u_3 \hat{u}_1} - \frac{1}{d_2 u_2 u_3} \\
&= -\frac{1}{d_2 u_2 u_3} \left(1 - \frac{f_1^0}{\hat{u}_1}\right).
\end{aligned}$$

Finally, for $i = 3$, (6.38) also yields,

$$\begin{aligned}
\left[(UL)^{-1}\right]_{3,4} &= -\frac{1}{d_3 u_3 u_4} \left(1 - \frac{h_1^2}{\hat{u}_2} + \frac{h_2^1}{\hat{u}_2 \hat{u}_1}\right) \\
&= -\frac{1}{d_3 u_3 u_4} \left(1 - \frac{f_1^1}{\hat{l}_1 \hat{u}_2} + \frac{f_2^0}{\hat{u}_2 \hat{u}_1}\right),
\end{aligned}$$

since by lemma 6.3.1 we have

$$h_1^2 = f_1^1 / \hat{l}_1 \qquad \text{and} \qquad h_2^1 = f_2^0. \ \square$$

Finally the entries $(i+1, i)$ of $(UL)^{-1}$, $i = 1, \dots, n-1$.

Theorem 6.3.4 *Consider L and U as described in (6.1). If U is invertible and* tridqds *algorithm applied to L and U is successful, then the quantities d_i, f_1^{i-1}, f_2^{i-1} and g_1^i generated*

by the algorithm satisfy

$$\begin{aligned}
\left[(UL)^{-1}\right]_{2,1} &= -\frac{1}{d_1 u_1} g_1^1, \\
\left[(UL)^{-1}\right]_{3,2} &= -\frac{1}{d_2 u_2}\left(g_1^2 + \frac{f_2^0}{\hat{u}_1}\right), \\
\left[(UL)^{-1}\right]_{i+1,i} &= -\frac{1}{d_i u_i}\left(g_1^i + \frac{f_2^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}}\right), \qquad i = 3, \ldots, n-1.
\end{aligned}$$

Proof. Using (6.19), (6.20) and (6.33), for $i = 3, \ldots, n$,

$$\begin{aligned}
\left[(UL)^{-1}\right]_{i+1,i} &= \boldsymbol{e}_{i+1}^T \left[(UL)^{-1}\right] \boldsymbol{e}_i \\
&= \boldsymbol{e}_{i+1}^T (\mathcal{L}_1 \mathcal{L}_2 \ldots \mathcal{L}_{i-1}) \left(F^{(i)} G^{(i)}\right)^{-1} (\mathcal{L}_1 \mathcal{L}_2 \ldots \mathcal{L}_{i-1})^{-1} \boldsymbol{e}_i \qquad (6.39) \\
&= \left[\boldsymbol{e}_{i+1}^T (\mathcal{L}_1 \mathcal{L}_2 \ldots \mathcal{L}_{i-1})\right] \left[\left(F^{(i)} G^{(i)}\right)^{-1} \boldsymbol{e}_i\right] \\
&= \begin{bmatrix} 0 & \ldots & 0 & 0 & h_2^{i-1} & 0 & 1 & \ldots & 0 \end{bmatrix} \left[\left(F^{(i)} G^{(i)}\right)^{-1} \boldsymbol{e}_i\right] \\
&= \left(h_2^{i-1} \boldsymbol{e}_{i-1}^T + \boldsymbol{e}_{i+1}^T\right) \left[\left(F^{(i)} G^{(i)}\right)^{-1} \boldsymbol{e}_i\right] \\
&= h_2^{i-1} \boldsymbol{e}_{i-1}^T \left(F^{(i)} G^{(i)}\right)^{-1} \boldsymbol{e}_i + \boldsymbol{e}_{i+1}^T \left(F^{(i)} G^{(i)}\right)^{-1} \boldsymbol{e}_i \\
&= h_2^{i-1} \left[\left(F^{(i)} G^{(i)}\right)^{-1}\right]_{i-1,i} + \left[\left(F^{(i)} G^{(i)}\right)^{-1}\right]_{i+1,i}. \qquad (6.40)
\end{aligned}$$

Using lemmas 6.3.10, 6.3.11 and 6.3.1,

$$\begin{aligned}
\left[(UL)^{-1}\right]_{i+1,i} &= -\frac{h_2^{i-1}}{d_i u_i \hat{u}_{i-1}} - \frac{g_1^i}{d_i u_i} \\
&= -\frac{1}{d_i u_i}\left(g_1^i + \frac{h_2^{i-1}}{\hat{u}_{i-1}}\right) \\
&= -\frac{1}{d_i u_i}\left(g_1^i + \frac{f_2^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}}\right).
\end{aligned}$$

For $i = 2$, (6.40) also holds,

$$\left[(UL)^{-1}\right]_{3,2} = h_2^1 \left[\left(F^{(i)} G^{(i)}\right)^{-1}\right]_{1,2} + \left[\left(F^{(i)} G^{(i)}\right)^{-1}\right]_{3,2}.$$

Again, using lemmas 6.3.10, 6.3.11 and 6.3.1,

$$\begin{aligned}\left[(UL)^{-1}\right]_{3,2} &= -\frac{h_2^1}{d_2u_2\hat{u}_1} - \frac{g_1^2}{d_2u_2} \\ &= -\frac{1}{d_2u_2}\left(g_1^2 + \frac{h_2^1}{\hat{u}_1}\right) \\ &= -\frac{1}{d_2u_2}\left(g_1^2 + \frac{f_2^0}{\hat{u}_1}\right).\end{aligned}$$

Finally, for $i = 1$, by lemma 6.3.11,

$$\left[(UL)^{-1}\right]_{2,1} = \left[\left(F^1G^{(1)}\right)^{-1}\right]_{2,1} = -\frac{g_1^1}{d_1u_1}. \ \square$$

Finally, theorems 6.3.2, 6.3.3 and 6.3.4 together give the entries for the matrix

$$\left[(UL)^{-1}\right]_{i:i+1,i:i+1} = \begin{bmatrix}\alpha^i & \gamma^i \\ & \\ \beta^i & \alpha^{i+1}\end{bmatrix}, \qquad i = 1, \ldots, n-1,$$

where

$$\begin{aligned}\alpha^i &:= \left[(UL)^{-1}\right]_{i,i}, & \gamma^i &:= \left[(UL)^{-1}\right]_{i,i+1} \\ \beta^i &:= \left[(UL)^{-1}\right]_{i+1,i}, & \alpha^{i+1} &:= \left[(UL)^{-1}\right]_{i+1,i+1}.\end{aligned}$$

Theorem 6.3.5 *Given the notation above, for $i = 4, \ldots, n-1$, we have*

$$\begin{aligned}\alpha^i &= \frac{1-a_i}{d_iu_i}, & \gamma^i &= -\frac{1-a_i}{d_iu_iu_{i+1}} \\ \beta^i &= -\frac{1}{d_iu_i}\left(g_1^i + \frac{f_2^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}}\right), & \alpha^{i+1} &= \frac{1-a_{i+1}}{d_{i+1}u_{i+1}},\end{aligned}$$

where $a_i = \dfrac{f_1^{i-2}}{\hat{l}_{i-2}\hat{u}_{i-1}} + \dfrac{f_2^{i-3}}{\hat{l}_{i-3}\hat{u}_{i-1}\hat{u}_{i-2}}$.

Proof. Use theorems 6.3.4, 6.3.3 and 6.3.2. $\square$

Chapter 7

Implementation details and numerical examples

The implementation of `3dqds` and all our numerical experiments were carried out in MATLAB 7.2.0.232 (R2006a) on a Pentium M 1.6GHz machine under Windows XP. All our computations were done in IEEE standard floating point arithmetic using double precision arithmetic with unit roundoff $\varepsilon = 2^{-53} \approx 1.1 \times 10^{-16}$. We used the advantages of the exception handling feature incorporated in this arithmetic.

In what follows `eigval` refers to our function to obtain the eigenvalues of an unreduced real tridiagonal matrix using `3dqds` and `eig` to MATLAB's function which uses LAPACK [2] routines that implement the QR algorithm on Hessenberg matrices. We report comparative results on carefully chosen tridiagonal matrices for test purposes: Bessel, Toeplitz, Clement, Liu's and symmetric matrices. We also present the first numerical results for a comparative study of some of the condition numbers presented in Chapter 2.

7.1 Implementation details

Next sections describe some of the details we had to consider in our implementation of eigval function. They are concerned with the shift strategy, the stopping criterion for deflation, the tolerance for element growth in 3dqds code, the need to choose a different shift after a failure and the possibility that the initial LU factorization of the J form of the input matrix C does not exist.

7.1.1 Choosing a shift for 3dqds

Consider matrices L and U as described in (6.1), page 166, and remember that the shift information for 3dqds is coded in the first column of $(UL)^2 - 2(\Re\sigma)UL + |\sigma|^2 I$ that, for a shift σ, is given by

$$\begin{bmatrix} (u_1 + l_1)^2 + u_2 l_1 - 2(\Re\sigma)(u_1 + l_1) + |\sigma|^2 \\ u_2 l_1 (u_1 + l_1 + u_2 + l_2 - 2(\Re\sigma)) \\ u_2 l_1 u_3 l_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

So, to completely specify one iteration of 3dqds we need

$$2(\Re\sigma) = \sigma + \overline{\sigma} \quad \text{and} \quad |\sigma|^2 = \sigma\overline{\sigma}.$$

Since 3dqds is a restoring shift transformation, in the case of a real eigenvalue, a reasonable choice of a single shift is the (n, n) element of $U_k L_k$, where L_k and U_k are the factors obtained after the k^{th} iteration. The generalization for double shifting is to use the *Francis shift*, which means that σ and $\bar{\sigma}$ are the eigenvalues of the bottom 2×2 corner of $U_k L_k$,

$$\begin{bmatrix} l_{n-1} + u_{n-1} & 1 \\ u_n l_{n-1} & u_n \end{bmatrix}. \tag{7.1}$$

This will let us converge to either two real eigenvalues in the bottom 2×2 or a single 2×2 block with complex conjugate eigenvalues. When we are close to convergence, we expect

the $(n-1, n-2)$ entry, and possibly the $(n, n-1)$ entry, to be tiny so that the eigenvalues of this 2×2 submatrix are good approximations to eigenvalues of J.

If σ and $\bar{\sigma}$ are the eigenvalues of (7.1) then

$$\sigma + \bar{\sigma} = l_{n-1} + u_{n-1} + u_n =: \text{Sum}$$
$$\sigma\bar{\sigma} = u_n u_{n-1} =: \text{Prod}$$

and Sum and Prod will be the shift information given to 3dqds. When the $(n, n-1)$ entry is negligible, then u_n will be an approximation for a real eigenvalue of J. If not, and if the $(n-1, n-2)$ entry is negligible then

$$\sigma_1 = \frac{1}{2}\left(\text{Sum} + \sqrt{\text{Sum}^2 - 4\,\text{Prod}}\right)$$
$$\sigma_2 = \text{Prod}\,/\sigma_1$$

will be the approximations to the eigenvalues of the 2×2 corner block which may be both real or complex conjugate. Then deflation takes place and the algorithm proceeds with the remaining submatrix until approximations to all eigenvalues are found.

In our present shift strategy, each time deflation occurs, we may choose to use first zero shifts (Sum = 0 and Prod = 0) until l_{n-1} or l_{n-2} is *small*, and only then switch to the Francis shift. In general, this procedure improves convergence.

7.1.2 Criterion for deflation

Since we expect $U_k L_k$ to converge to a quasi-bidiagonal form, we use only tests to decide when l_{n-1} or l_{n-2} is negligible. These tests are performed before a 3dqds transformation takes place (not after) and consist in verifying if

$$|l_{n-1}| < Tol_{Def} \qquad \text{or} \qquad |l_{n-2}| < Tol_{Def}$$

for a given tolerance Tol_{Def}. If we choose $Tol_{Def} = \varepsilon$, it may be too severe and it will take more time to converge; if we relax too much, we will converge faster but we may loose accuracy seriously. So, we must find a compromise between the speed of convergence and

the accuracy we want to attain. By default, in our experiments we used $Tol_{Def} = \varepsilon$ since we were mainly concerned with accuracy. This is the most crucial parameter and needs more study.

7.1.3 Tolerance for element growth

The LU transformation fails when zero pivots do occur. Furthermore, for reasons of numerical stability, we do not accept the result of a particular transformation if, in consequence of the occurrence of pivots of very small size, elements of excessive size do appear in L or U factors. For this reason, we monitor the *element growth* and reject L and U when

$$\max_i |l_i| > Tol_{Growth} \|J\| \qquad \text{or} \qquad \max_i |u_i| > Tol_{Growth} \|J\|$$

for a given tolerance Tol_{Growth}. When this happens, we signal a *breakdown* but it must be emphasized that is not a dramatic situation since we may recover. It is just a transformation whose result is neglected.

If we allow less element growth, we will have more breakdowns and it will take more time to converge. Allowing huge element growth will imply less accuracy. Again, there must be a trade-off between element growth and accuracy. Usually we let $Tol_{Growth} = 1/\sqrt{\varepsilon}$.

The possibility of dealing with divisions by zero in IEEE arithmetic allows 3dqds transformation to be carried out till the very end, even when a pivot is zero or too small. Comparison of the size of the new values of l_i and u_i with $Tol_{Growth} \|J\|$ is only performed at the end of a transformation.

7.1.4 The shift after a failure

After a failure, there is the question of what shift to use next. We don't want to move away from the previous shift too much, just the necessary amount so that the 3dqds transformation does not breakdown. We also have to admit the possibility of a succession of failures.

So, while we do not have a successful transformation we will repeat the following

if Prod $\neq 0$

Sum $= (1+\delta) * \text{Sum}$

Prod $= (1+\delta)^2 * \text{Prod}$

$\delta = 2 * \delta$

else

small $= \sqrt{\varepsilon} * \text{local}$

Sum $= 2 * \text{small}$

Prod $= \text{small}^2$

end if

where "local" is a local norm and, initially, we let $\delta = \sqrt{\varepsilon}$. This procedure forces to define a limited number of failures allowed.

7.1.5 Initial LU factorization

The J form of C may not have an LU factorization. In this case we must choose an initial shift τ so that the factorization

$$J - \tau I = LU$$

is possible and the 3dqds transform may start. Initially, we let $\tau = \theta$, for a *small* θ, and then, if we are still not successful, we increase τ by θ until we have success. In the end τ must be added to the approximations given by eigval. But this choice needs more thought because it seems to affect the accuracy more than we expected.

7.2 Numerical examples

In the figures containing the plots of the computed eigenvalues that are presented in this section, the results obtained with `eig` are represented by a plus sign and the ones provided by eigval are represented with a big dot.

7.2.1 Bessel matrices

Bessel matrices, associated with generalized Bessel polynomials (see section 1.3.2, page 22), are nonsymmetric tridiagonals matrices defined by $B_n^{(a,b)} = \text{tridiag}(\boldsymbol{\beta}^{(a,b)}, \boldsymbol{\alpha}^{(a,b)}, \boldsymbol{\gamma}^{(a,b)})$ with

$$\alpha_1^{(a,b)} = -\frac{a}{b}, \quad \gamma_1^{(a,b)} = -\alpha_1^{(a,b)}, \quad \beta_1^{(a,b)} = \frac{\alpha_1^{(a,b)}}{a+1},$$

and

$$\begin{aligned}
\alpha_j^{(a,b)} &:= -b\frac{a-2}{(2j+a-2)(2j+a-4)}, \quad j = 2, \ldots, n, \\
\gamma_j^{(a,b)} &:= b\frac{j+a-2}{(2j+a-2)(2j+a-3)}, \\
\beta_j^{(a,b)} &:= -b\frac{j}{(2j+a-1)(2j+a-2)}, \quad j = 2, \ldots, n-1.
\end{aligned}$$

All the eigenvalues are simple and below we report three theorems from [44] that contain information about the localization of these eigenvalues. These theorems apply to the cases

$$a \in \mathbb{R}, \quad n > 1 - a, \quad b = 2. \tag{7.2}$$

Theorem 7.2.1 *Let the condition (7.2) be satisfied. Also let $n \geq 1$. Then all the eigenvalues of $B_n^{(a,b)}$ lie in the cardioil region*

$$C(n,a) := \left\{ z = \rho\, \mathrm{e}^{\mathbf{i}\theta} \in \mathbb{C} : \ 0 < \rho < \frac{1-\cos\theta}{n+a-1} \right\} \cup \left\{ \frac{-2}{n+a-1} \right\}.$$

Theorem 7.2.2 *Let the condition (7.2) be satisfied. Also let $n \geq 2$. Then all the eigenvalues of $B_n^{(a,b)}$ belong to the sector*

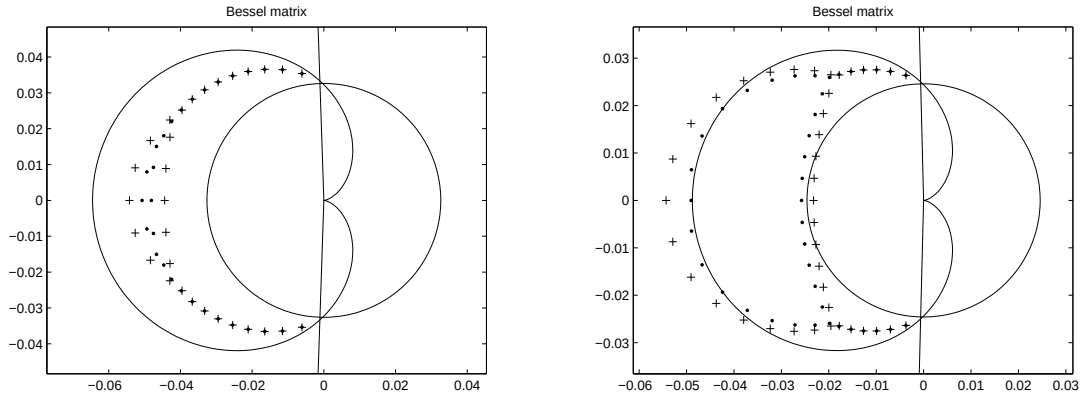
$$S(n,a) := \left\{ z = \rho\, \mathrm{e}^{\mathbf{i}\theta} \in \mathbb{C} : \ |\theta| > \cos^{-1}\left(\frac{-a}{2n+a-2}\right), \ -\pi < \theta \leq \pi \right\}.$$

Theorem 7.2.3 *Let the condition (7.2) be satisfied. Also let $n \geq 1$. Then all the eigenvalues of $B_n^{(a,b)}$ belong to the infinite region*

$$I(n,a) := \left\{ z \in \mathbb{C} : \ |z| > \frac{2}{2n+a-\frac{2}{3}} \right\}.$$

Thus, the eigenvalues of the matrix $B_n^{(a,b)}$ lie in the intersection of the inclusion regions defined by the sets $C(n,a)$, $S(n,a)$ and $I(n,a)$. But these eigenvalues suffer from ill-conditioning that increases with n. Matrices $B_n^{(a,b)}$ are close to defective matrices.

Figure 7.1 shows, along with the inclusion regions defined above, the results obtained with Matlab function `eig` (plus sign) and with eigval (big dot) for the classical case $a = b = 2$, for $n = 30$ and $n = 40$. As n increases, computed eigenvalues even fail to belong to the inclusion regions. But the answers of eigval are usually better.

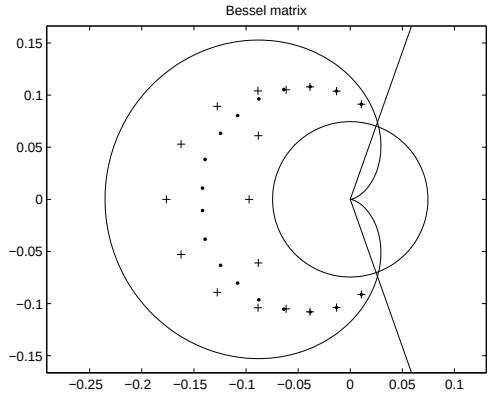


(a) $n = 30$ (b) $n = 40$

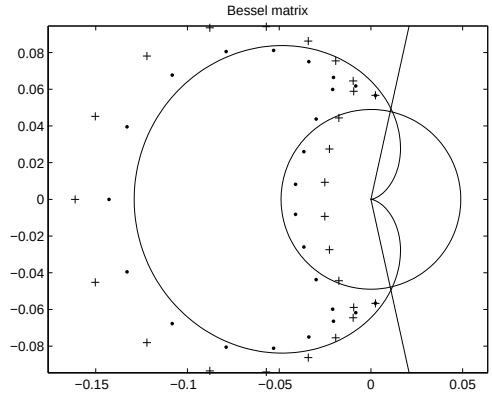
Figure 7.1: *Bessel matrix with* $a = 2$ *and* $b = 2$

In [44] it is mentioned that the ill-conditining seems to reach its maximum when a ranges from -8.5 to -4.5. We also report the cases $a = -8.5$, $b = 2$ (for $n = 18$ and $n = 25$), $a = -4.5$, $b = 2$ (for $n = 20$ and $n = 25$) and $a = 12$, $b = 2$ (for $n = 40$ and $n = 50$). See Figure 7.2. The number of iterations needed for eigval to converge is approximately equal to $2n$.

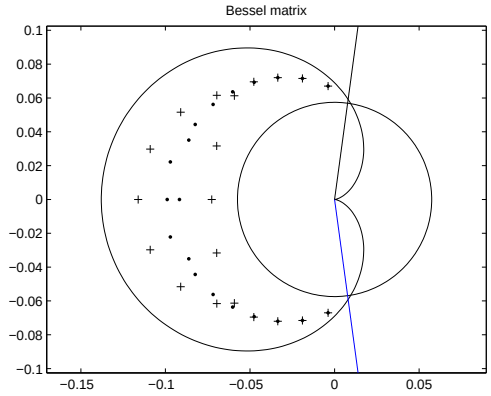
Bessel matrices are notoriously difficult. There is an interesting phenomenon of bifurcation in the eigenvalues computed by both algorithms and the regularity, in spite of the presence of ill-conditioning, is quite interesting. The approximations given by eigval follow the same pattern as the ones given by MATLAB but seem to be better. As n increases the huge deviation from the expected curve is not due to element growth - it rarely occurred.



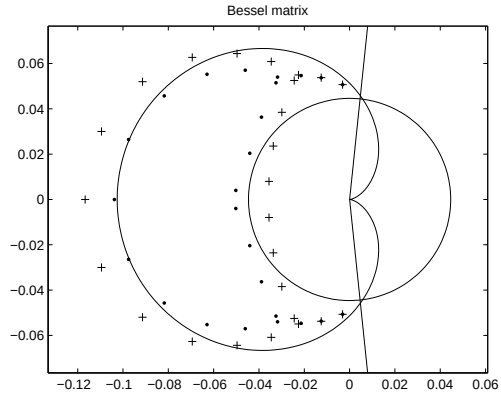
(a) $a = -8.5,\ b = 2;\ n = 18$



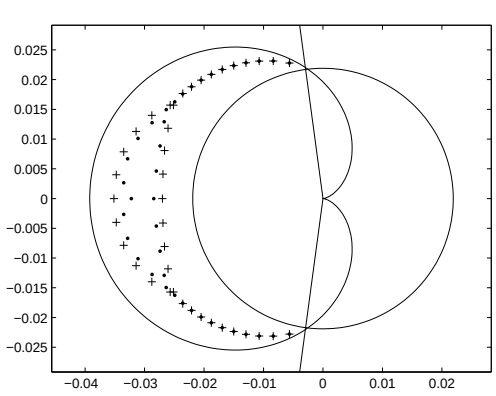
(b) $a = -8.5,\ b = 2;\ n = 25$



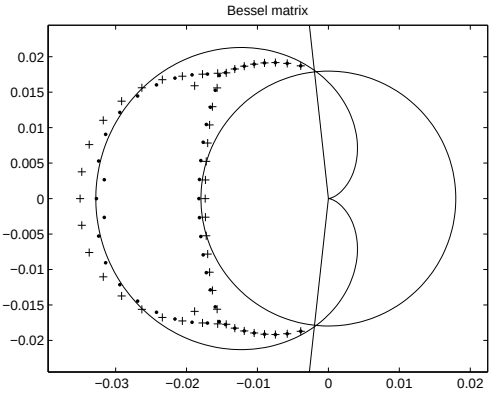
(c) $a = -4.5,\ b = 2;\ n = 20$



(d) $a = -4.5,\ b = 2;\ n = 25$



(e) $a = 12,\ b = 2;\ n = 40$



(f) $a = 12,\ b = 2;\ n = 50$

Figure 7.2: *Bessel matrices with* $a = -8.5, b = 2$*;* $a = -4.5, b = 2$ *and* $a = 12, b = 2$

Condition numbers

For a simple eigenvalue $\lambda \neq 0$ of a given matrix A, redefine now κ_λ as the Wilkinson's relative condition number for λ (see section 1.2.3, page 14), that is,

$$\kappa_\lambda(A) \equiv \frac{\|y^*\| \, \|x\|}{|\lambda||y^*x|}$$

For $J = LU$ the relative condition numbers (see section 2.4.2, page 44)

$$\text{relcond}_1(\lambda; LU) = \frac{|y|^T M_1 |x|}{|y^*x||\lambda|}$$

and

$$\text{relcond}_2(\lambda; LU) = \frac{|y|^T (v + w)}{|y^*x|}$$

give the sensitivity of eigenvalues to perturbations in the entries of the factors L and U.

In Table 7.1 we display the condition number κ_λ for $B_n^{(a,b)}$ with $a = -4.5$, $b = 2$ and $n = 20$, and for the corresponding J form. We also display the condition numbers relcond_1 and relcond_2 for the LU representation of J. It may be seen that, although the eigenvalues are much more sensitive to perturbations in J than to perturbations in $B_n^{(a,b)}$, the situation improves significantly in the L, U representation, that is, these factors define the eigenvalues not only much better than J does but, also, they define better the eigenvalues than $B_n^{(a,b)}$ itself.

The eigenvectors we used to produce these numbers were the eigenvectors delivered by `eig` function of MATLAB.

7.2.2 Clement matrices

The so-called *Clement matrix* (see [4])

$$C_n = \text{tridiag}(\boldsymbol{\beta}, \mathbf{0}, \boldsymbol{\gamma})$$

with $\boldsymbol{\gamma} = (\gamma_j)$, $\gamma_j = j$ and $\boldsymbol{\beta} = (\beta_j)$, $\beta_j = \gamma_{n-j}$, $j = 1, \ldots, n-1$, has exact eigenvalues

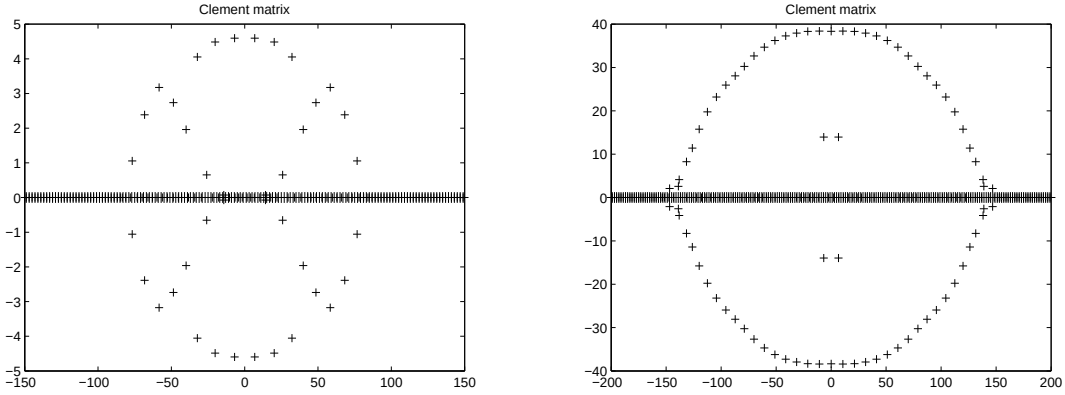
$$\pm\, n-1, n-3, \ldots, 1, \qquad \text{for } n \text{ even},$$

$$\pm\, n-1, n-3, \ldots, 0, \qquad \text{for } n \text{ odd}.$$

λ	$\kappa_\lambda(B)$	$\kappa_\lambda(J)$	$\mathrm{relcond}_1(\lambda; LU)$	$\mathrm{relcond}_2(\lambda; LU)$
$-3.8\,10^{-3}\ -6.7\,10^{-2}\mathbf{i}$	$6\,10^{8}$	$3\,10^{22}$	$3\,10^{5}$	$6\,10^{5}$
$-3.8\,10^{-3}\ +6.7\,10^{-2}\mathbf{i}$	$6\,10^{8}$	$4\,10^{22}$	$5\,10^{5}$	$9\,10^{5}$
$-7.1\,10^{-2}\ -1.6\,10^{-2}\mathbf{i}$	$5\,10^{10}$	$4\,10^{25}$	$5\,10^{5}$	$10\,10^{5}$
$-7.1\,10^{-2}\ +1.6\,10^{-2}\mathbf{i}$	$5\,10^{10}$	$4\,10^{25}$	$3\,10^{6}$	$10\,10^{6}$
$-1.9\,10^{-2}\ -7.2\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$3\,10^{23}$	$3\,10^{6}$	$2\,10^{6}$
$-1.9\,10^{-2}\ +7.2\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$3\,10^{23}$	$9\,10^{8}$	$1\,10^{8}$
$-3.4\,10^{-2}\ -7.2\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$2\,10^{24}$	$8\,10^{8}$	$1\,10^{8}$
$-3.4\,10^{-2}\ +7.2\,10^{-2}\mathbf{i}$	$1\,10^{12}$	$2\,10^{24}$	$3\,10^{10}$	$3\,10^{9}$
$-6.5\,10^{-2}\ -4.6\,10^{-2}\mathbf{i}$	$1\,10^{12}$	$3\,10^{25}$	$3\,10^{10}$	$2\,10^{9}$
$-6.5\,10^{-2}\ +4.6\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$3\,10^{25}$	$10\,10^{10}$	$6\,10^{9}$
$-4.7\,10^{-2}\ -6.9\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$6\,10^{24}$	$10\,10^{10}$	$7\,10^{9}$
$-4.7\,10^{-2}\ +6.9\,10^{-2}\mathbf{i}$	$1\,10^{13}$	$6\,10^{24}$	$5\,10^{11}$	$3\,10^{10}$
$-6.0\,10^{-2}\ -6.6\,10^{-2}\mathbf{i}$	$1\,10^{13}$	$9\,10^{24}$	$4\,10^{11}$	$3\,10^{10}$
$-6.0\,10^{-2}\ +6.6\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$9\,10^{24}$	$1\,10^{12}$	$9\,10^{10}$
$-8.0\,10^{-2}\ -5.9\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$4\,10^{24}$	$1\,10^{12}$	$8\,10^{10}$
$-8.0\,10^{-2}\ +5.9\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$4\,10^{24}$	$3\,10^{12}$	$2\,10^{11}$
$-1.0\,10^{-1}\ -4.3\,10^{-2}\mathbf{i}$	$1\,10^{14}$	$1\,10^{24}$	$2\,10^{12}$	$1\,10^{11}$
$-1.0\,10^{-1}\ +4.3\,10^{-2}\mathbf{i}$	$2\,10^{14}$	$1\,10^{24}$	$3\,10^{12}$	$2\,10^{11}$
$-1.2\,10^{-1}\ -1.6\,10^{-2}\mathbf{i}$	$2\,10^{14}$	$5\,10^{23}$	$2\,10^{12}$	$2\,10^{11}$
$-1.2\,10^{-1}\ +1.6\,10^{-2}\mathbf{i}$	$2\,10^{14}$	$5\,10^{23}$	$2\,10^{12}$	$1\,10^{11}$

Table 7.1: *Relative condition numbers for the eigenvalues of* $B_{20}^{(-4.5,2)}$

Figure 7.3 illustrates the results of both algorithms for the Clement matrices with $n = 150$ and $n = 200$. All the approximations given by eigval are real while the approximations obtained with MATLAB have large imaginary parts.



(a) $n = 150$ (b) $n = 200$

Figure 7.3: *Clement matrix*

Table 7.2 shows the largest and the smallest Wilkinson's relative condition number, $\kappa_{\max} \equiv \max_\lambda \kappa_\lambda(C_n)$ and $\kappa_{\min} \equiv \min_\lambda \kappa_\lambda(C_n)$, and also the largest and smallest of the relative condition number $\text{relcond}_1(\lambda; LU)$, max_{relcond_1} and min_{relcond_1}, respectively.

n	$\kappa_{\max}$	$\kappa_{\min}$	max_{relcond_1}	min_{relcond_1}
150	$3.5\ 10^{20}$	$1.0\ 10^{4}$	$6.4\ 10^{4}$	$2.8\ 10^{1}$
200	$1.2\ 10^{28}$	$2.3\ 10^{11}$	$2.3\ 10^{4}$	$2.3\ 10^{1}$

Table 7.2: *Relative condition numbers for the Clement matrix*

The minimum and maximum relative errors, rel_{min} and rel_{max}, respectively, for $n = 150$ and $n = 200$, as well as for $n = 300$ and $n = 450$, are presented in Table 7.3. When n increases almost all the approximations given by Matlab exhibit significantly high relative errors, while with eigval we can consider that all the approximations produced have satisfactory relative accuracy.

	eigval		eig	
n	rel_{min}	rel_{max}	rel_{min}	rel_{max}
150	$7.8\ 10^{-12}$	$8.1\ 10^{-9}$	$3.2\ 10^{-15}$	$7.8\ 10^{1}$
200	$1.2\ 10^{-11}$	$6.4\ 10^{-9}$	$2.9\ 10^{-16}$	$1.1\ 10^{2}$
300	$3.6\ 10^{-11}$	$1.1\ 10^{-8}$	$6.6\ 10^{-3}$	$1.7\ 10^{2}$
450	$4.5\ 10^{-12}$	$1.8\ 10^{-8}$	$4.5\ 10^{-3}$	$2.6\ 10^{2}$

Table 7.3: *Relative errors for the Clement matrix*

MATLAB function `eig` does not detect that Clement matrices are symmetrizable matrices. Our function eigval gives much better results than `eig`, almost as good as the ones MATLAB delivers if we give as input a symmetric matrix similar to C_n.

7.2.3 Liu's matrices

Z. S. Liu [31] devised an algorithm to obtain one-point spectrum unreduced tridiagonal matrices of arbitrary dimension $n \times n$. These matrices, that we already introduced in Chapter 4 and called *Liu's matrices*, have only one eigenvalue, zero with multiplicity n, and the Jordan form consists of one Jordan block. We represent Liu's matrices as

$$Liu_n = \text{tridiag}(\mathbf{1}^n, \boldsymbol{\alpha}^n, \boldsymbol{\gamma}^n)$$

where $\mathbf{1}^n$ always stands for a vector of 1's of length $n-1$. Notice that the transpose of Liu_n is already in J form.

We considered Liu_n for $n = 6$,

$$Liu_6 = \begin{bmatrix} 0 & -1 & & & & \\ 1 & 0 & 1 & & & \\ & 1 & -1 & -1 & & \\ & & 1 & 1 & 1 & \\ & & & 1 & 0 & -1 \\ & & & & 1 & 0 \end{bmatrix},$$

for $n = 14$,

$$\boldsymbol{\alpha}^{14} = [0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ -1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0]^T$$
$$\boldsymbol{\gamma}^{14} = [-1\ \ 1\ \ 1\ \ -1\ \ 1\ \ -1\ \ -1\ \ -1\ \ 1\ \ -1\ \ 1\ \ 1\ \ -1]^T,$$

and for $n = 28$,

$$\boldsymbol{\alpha}^{28} = [0,0,0,0,0,0,-1,1,0,0,0,0,0,-1,1,0,0,0,0,0,1,-1,0,0,0,0,0,0]^T$$
$$\boldsymbol{\gamma}^{28} = [-1,1,1,-1,1,-1,-1,-1,1,-1,1,1,-1,-1,-1,1,1,-1,1,-1,-1,-1,1,-1,1,1,-1]^T.$$

Now a note on perturbation theory for multiple eigenvalues. Consider the example of perturbing by ϵ the $(n,1)$ entry of an $n \times n$ Jordan block. Let

$$\widetilde{C} = \begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ \epsilon & & & 0 \end{bmatrix}.$$

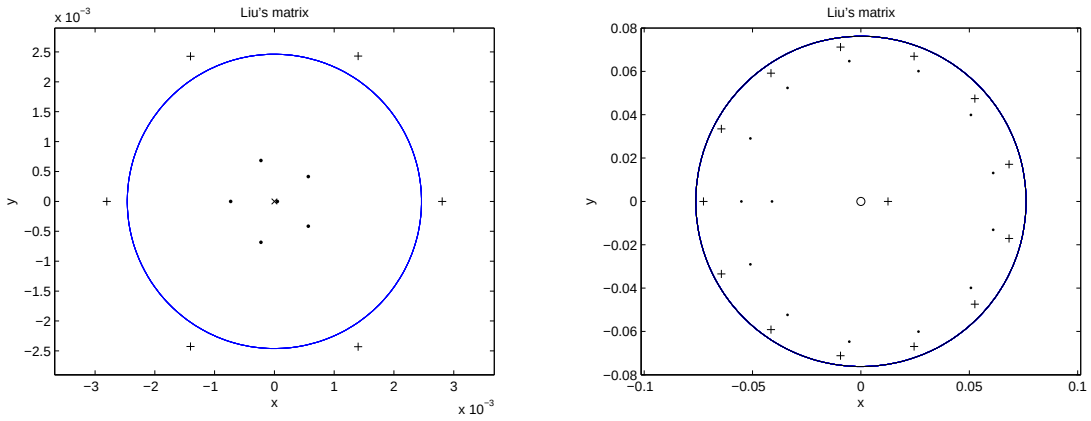
Then the characteristic equation changes from $\lambda^n = 0$ to $\lambda^n - \epsilon = 0$. So the eigenvalues of the perturbed matrix $\widetilde{C}$ are the n possible complex roots of ϵ, $\lambda_k = \sqrt[n]{\epsilon}\, \mathrm{e}^{\mathbf{i}\frac{2k\pi}{n}}$, $k = 0, \ldots, n-1$. The n^{th} root of ϵ grows much faster than any multiple of ϵ for small ϵ. More formally, the condition number of a multiple eigenvalue is infinite because at $\epsilon = 0$, for $n \geq 2$,

$$\frac{d\lambda}{d\epsilon} = \frac{1}{n\epsilon^{1-\frac{1}{n}}} = \infty.$$

For example, if we take $n = 16$ and $\epsilon = 10^{-16}$, then for each eigenvalue λ_k, we have $|\lambda_k| = 0.1$, a change 10^{15} times greater than ϵ. However, having an infinite condition number does not mean that the eigenvalues cannot be computed with any correct digits.

The more apart from the diagonal entries the perturbations occur, the worse the effect on the eigenvalues seems to be. So, we decided to show our numerical results together with the circles $|z| = \sqrt[n]{\varepsilon}$ where ε is the unit roundoff. If the approximations are inside this circle we may consider the results good enough, since they are not worse than the ones that we would get in exact arithmetic for the Jordan block similar to Liu's matrix perturbing by ε the $(1,n)$ entry .

The results we obtained with eigval and `eig` for $n = 6$, $n = 14$ and $n = 28$ are shown in the following figures.



(a) $n = 6$ (b) $n = 14$

Figure 7.4: *Liu's matrices*

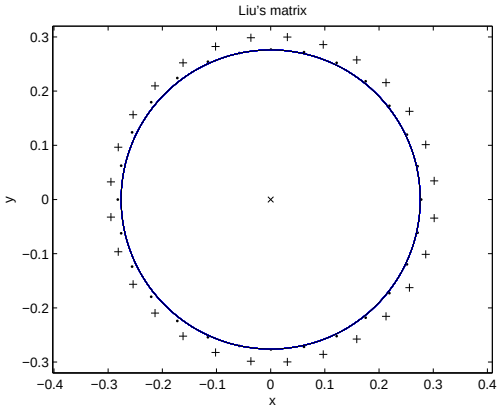


Figure 7.5: *Liu's matrix for* $n = 28$

The accuracy of our approximations is slightly better than the accuracy of those provided by MATLAB and the convergence is surprisingly quick. The number of iterations needed for eigval to converge is less than $2n$.

7.2.4 Toeplitz matrices

A tridiagonal Toeplitz matrix has the form

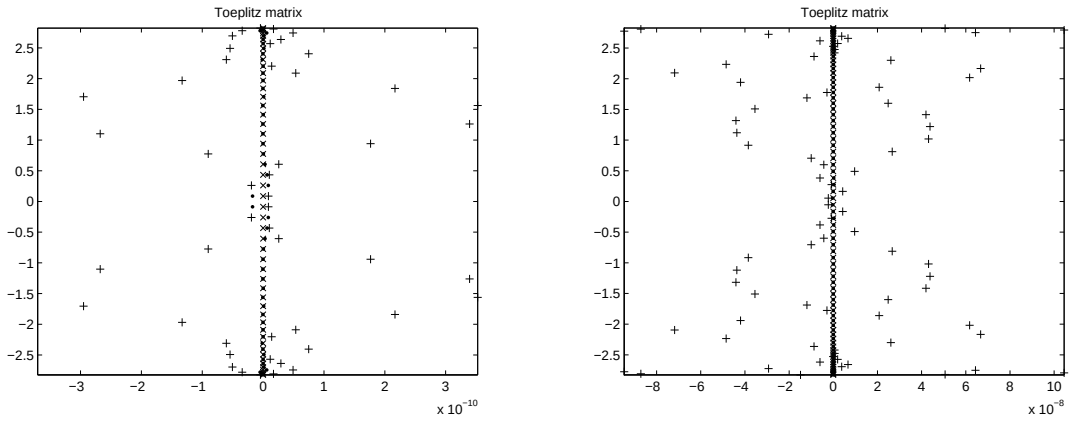
$$T_n = \begin{bmatrix} a & c & & & \\ b & a & c & & \\ & \ddots & \ddots & \ddots & \\ & & b & a & c \\ & & & b & a \end{bmatrix}.$$

Such matrices arise, for example, when discretizing partial differential equations or boundary value problems for ordinary differential equations [Higham, p.522] . The eigenvalues are known explicitly:

$$a + 2(bc)^{1/2} \cos\left(\frac{k\pi}{n+1}\right), \quad k = 1, \ldots, n.$$

If $bc < 0$ then the exact eigenvalues are complex with real part equal to a.

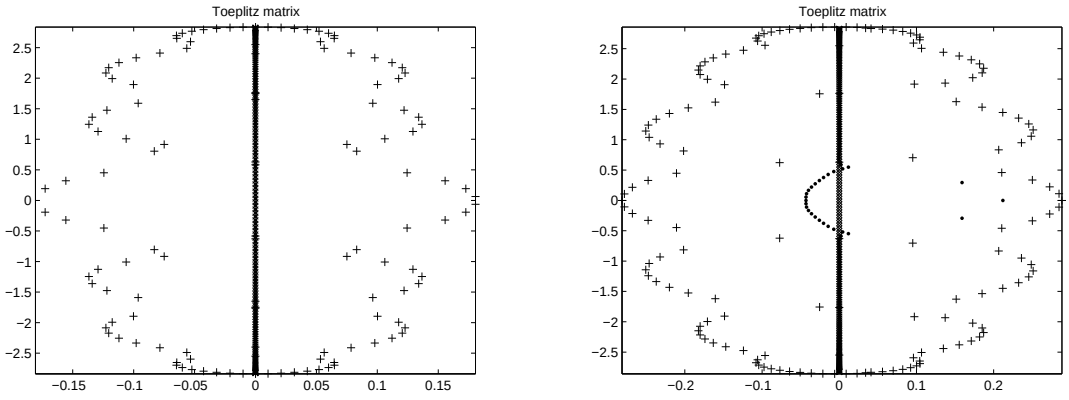
We first show the results for T_{50} and T_{80} with $a = 1$, $b = 2$ and $c = -1$. In Figure 7.6 we also represent the exact solutions with a times sign ($\times$). For an exact eigenvalue λ and an approximation $\hat{\lambda}$ we do not represent λ or $\hat{\lambda}$ but $\lambda - a$ and $\hat{\lambda} - a$.



(a) $n = 50$ (b) $n = 80$

Figure 7.6: *Toeplitz matrix with* $a = 1$, $b = 2$ *and* $c = -1$

We also show the results for $n = 150$ and $n = 200$.



(a) $n = 150$ (b) $n = 200$

Figure 7.7: *Toeplitz matrix with* $a = 1$, $b = 2$ *and* $c = -1$

The accuracy of the approximations given by eigval is better than the accuracy of the approximations delivered by MATLAB. See Table 7.4 for a report on the relative errors for the different values of n.

	eigval		eig	
n	rel_{min}	rel_{max}	rel_{min}	rel_{max}
50	$3.8\ 10^{-14}$	$2.6\ 10^{-11}$	$3.8\ 10^{-12}$	$3.6\ 10^{-10}$
80	$5.4\ 10^{-14}$	$3.5\ 10^{-10}$	$5.0\ 10^{-9}$	$1.2\ 10^{-6}$
150	$1.2\ 10^{-13}$	$4.3\ 10^{-5}$	$3.2\ 10^{-3}$	$1.8\ 10^{-1}$
200	$2.7\ 10^{-13}$	$2.1\ 10^{-1}$	$2.1\ 10^{-3}$	$2.3\ 10^{-1}$

Table 7.4: *Relative errors for the Toeplitz matrix with* $a = 1$, $b = 2$ *and* $c = -1$

7.2.5 Symmetric matrices

Finally, as an example of a symmetric matrix we will just consider the tridiagonal Toeplitz T_n as described in the previous section with $b = c$. For this case the exact eigenvalues are

$$a + 2b\cos\left(\frac{k\pi}{n+1}\right), \quad k = 1, \ldots, n.$$

The behavior of `eig` function of Matlab is better than eigval. In table 7.5 we show the bounds for the relative errors of the approximations obtained with eigval for the case of $a = 5$, $b = 1$ and $c = 1$. The approximations provided by MATLAB have full accuracy.

	eigval	
n	rel_{min}	rel_{max}
50	$4.8\ 10^{-15}$	$5.4\ 10^{-12}$
100	$1.8\ 10^{-15}$	$1.4\ 10^{-11}$
200	$3.0\ 10^{-14}$	$2.1\ 10^{-8}$

Table 7.5: *Relative errors for the Toeplitz matrix with* $a = 5$ *and* $b = c = 1$

Chapter 8

Summary

The unsymmetric eigenvalue problem is, in general, much harder than the real symmetric eigenvalue problem for the basic reason that it is not always well posed. More precisely, the derivative of an eigenvalue with respect to certain matrix entries may be infinite. When executed in computer arithmetic any method finds itself aiming at a target (an eigenvalue) that may change at every step in the process. In contrast the derivatives in the symmetric case are all bounded by 1.

The ideal algorithm should compute each eigenvalue to nearly the accuracy to which it is determined by the data (matrix entries). Ideally the algorithm should report what that attainable accuracy is. The traditional way to do this is to deliver well chosen condition numbers along with the computed eigenvalues. It turns out that computing the condition numbers is just as difficult as computing the eigenvalues themselves.

We focus on the tridiagonal eigenproblem which occurs in its own right (for instance with Bessel polynomials) and also as a condensed form of a real square matrix. The first contribution of this thesis is to consider the sensitivity of the eigenvalues to different representations of an unreduced tridiagonal matrix C:

a) the non-trivial entries of L and D where $T\Delta = D_1^{-1}CD_1 = LDU = LDL^T\Delta$ with Δ a signature matrix and T symmetric;

b) the non-trivial entries of L and U where $J = D_2^{-1}CD_2 = LU$, and the $(i, i+1)$ entries of J are all 1.

These new measures are, in general, smaller than Wilkinson's condition number and are sometimes quite realistic despite being upper bounds. Our measures take advantage of the tridiagonal form and play a fundamental role in assessing attainable accuracy. See Chapter 7 for examples.

A second contribution is purely mathematical. We follow a hint given in Wilkinson's monumental book "*The Algebraic Eigenvalue Problem*" and produce a rigorous proof that the LR algorithm, and our `dqd` algorithm, converge without breakdown to the nonzero eigenvalue of an unreduced tridiagonal matrix with a one-point spectrum. The context is exact arithmetic and the rate of convergence is very slow ($\mathcal{O}(1/k)$ as $k \to \infty$) but the surprise is that the algorithm actually does converge in the presence of a (large) Jordan block.

The main contribution of this thesis is practical: the presentation of a robust and efficient algorithm `3dqds` that computes both real and complex eigenvalues of a real tridiagonal matrix while employing real arithmetic throughout the computation. In general, in our numerical tests, the output is more accurate than MATLAB's procedure `eig` and requires $\mathcal{O}(n^2)$ arithmetic effort instead of MATLAB's $\mathcal{O}(n^3)$.

To the best of our knowledge this is the first robust algorithm that takes advantage of tridiagonal form. However more work needs to be done in tuning some of the parameters. Deep study of a variety of cases is needed. The basic difficulty is easy to state: demand too much accuracy and any procedure will fail (never converge), demand too little accuracy and the unnecessary errors in accepting one eigenvalue can spoil the accuracy in another one.

8.1 Future work

Our test bed is far too small to draw conclusions. We are not yet in a position to claim that eigval function (that uses 3dqds code) always delivers more accurate approximations than `eig` function of MATLAB. The first results we obtained are very good results but we need a a more comprehensive study of test matrices with a variety of distributions.

Finding a general shift strategy that is efficient over a wide range of matrices is very difficult. It is a research topic in itself. In future work we plan to dedicate effort in trying to improve the shift strategy implemented in the present version of our code.

We also plan to develop a more sophisticated deflation strategy that monitors convergence to each eigenvalue (or conjugate pair) to detect when the process has stagnated, i.e., when further steps will not reduce the corresponding off-diagonal entry.

A different aspect of 3dqds is that it converts the given tridiagonal matrix to LU form via the factorization $J - \sigma I = LU$. This initial shift affects the attainable accuracy and needs more study. In this context, the issues of a numerical error analysis need to be considered.

We have now a tool to tackle the tridiagonal eigenvalue problem. It will be of interest to compare the performance of eigval with the possible rival algorithms: Françoise Tisseur's *Ehrlich-Aberth method* [3], the *Extended HR algorithm* of A. Liu [31] (unpublished finite precision implementation) and David Day's complex dqds algorithm [7] (unpublished preliminary version).

The computation of row and column eigenvectors is beyond the scope of this thesis but it is a necessary component to assess the attainable accuracy and to refine initial approximations. Our ultimate goal is to produce a software package that is both more efficient, more accurate and more informative than MATLAB's `eig` procedure.

Bibliography

[1] G. Ammar, W. Dayawansa and C. Martin. *Exponential interpolation: theory and numerical algorithms.* Applied Mathematics and Computation, 41 (3):189-232 (1991). Cited in [30].

[2] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney and D. Sorensen. *LAPACK User's Guide.* Third Edition, SIAM, Philadelphia, 1999. `http://www.netlib.org/lapack/lug/lapack_lug.html`

[3] D. A. Bini, L. Gemignani and F. Tisseur. *The Ehrlich-Aberth method for the nonsymmetric tridiagonal eigenvalue problem.* SIAM J. Matrix Anal. Appl., 27(1):153-175, 2005.

[4] P. A. Clement. *A class of triple-diagonal matrices for test purposes.* SIAM Review, 1 (vol.1), January, 1959.

[5] F. Chatelin. *Eigenvalues of Matrices.* John Wiley and Sons, Inc., 1995.

[6] A. Dax and S. Kaniel. *The ELR method for computing the eigenvalues of a general matrix.* SIAM J. Numer. Anal., 18:597-605, 1981.

[7] D. Day. *The differential qd algorithm for the tridiagonal eigenvalue problem.* In preparation, 1997.

[8] D. Day. *An efficient implementation of the nonsymmetric Lanczos algorithm*. SIAM J. Matrix Analysis and Applications, 3 (vol.18):566-589, 1997.

[9] L. S. DeJong. *Towards a formal definition of numerical stability*. Numerische Mathematic, 28:211-220, 1977.

[10] J. W. Demmel. *Open Problems in Numerical Linear Algebra*. IMA Preprint Series #961, Institute for Mathematics and its Applications, University of Minnesota, Minneapolis, MN, 1992.

[11] J. W. Demmel. *Applied Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, 1997.

[12] J. W. Demmel. *LAPACK Working Note 60, UT CS-93-192*. Parallel Numerical Linear Algebra. Computer Science Division and Mathematics Department, University of California at Berkeley.

[13] J. J. Dongarra, G.A. Geist and C. H. Romine. *Algorithm 710: FORTRAN subroutines for computing the eigenvalues and eigenvectors of a general matrix by reduction to general tridiagonal form*. ACM Trans. Math. Software, 18:392-400, 1992.

[14] M. I. Falcão and M. J. Soares. *Análise numérica, um curso prático com o MATLAB*. Departamento de Matemática da Universidade do Minho, 2004.

[15] K. Fernando. *Accurate ordering of eigenvectors and singular vectors without eigenvalues and singular values*. Linear Algebra and its Applications, 374:1-17, 2003.

[16] K. Fernando and B. Parlett. *Accurate singular values and differential qd algorithms*. Numerische Mathematic, 67:191-229, 1994.

[17] M. Fiedler. *Special matrices and their applications in numerical mathematics*. Martinus Nijhoff Publishers, 1986.

[18] J. G. F. Francis. *The QR transformation - a unitary analogue to the LR transformation, Parts I and II.* Computer Journal, 4:265-272 and 332-245, 1961/62.

[19] F. R. Gantmakher and M. G. Krein. *Oscillation matrices and kernels and small vibrations of mechanical systems.* United States Atomic Energy Commission, Office of Technical Information. Translated from a publication of the State Publishing House for Technical-Theoretical Literature, Moscow-Leningrad, 1950.

[20] G. A. Geist. *Reduction of a general matrix to tridiagonal form.* SIAM J. Matrix Anal. Appl., 12:362-373, 1991.

[21] G. H. Golub and Charles F. Van Loan. *Matrix Computation.* Johns Hopkins University Press, Baltimore and London, 3rd ed., 1996.

[22] G. H. Golub and Henk A. van der Vorst. *Numerical Progress in Eigenvalue Computation in the 20th Century.* Working document, September, 1999.

[23] G. W. Golub and W. Kahan. *Calculating the singular values and pseudoinverse of a matrix.* SIAM J. Numer. Anal. Ser. B 2:205-224, 1965.

[24] R. T. Gregory. *Defective and derogatory matrices.* SIAM Review, 2 (vol.2), April, 1960.

[25] R. T. Gregory and D. L. karney. *A collection of matrices for testing computational algorithms.* John Wiley and Sons, Inc., 1969.

[26] M. H. Gutknecht, D. Boley, S. Elhay, G. Golub. *Nonsymmetric Lanczos and finding orthogonal polynomials associated with indefinite weights.* Numerical Algorithms 1, 21-43, 1991.

[27] N. J. Higham. *Accuracy and Stability of Numerical Algorithms, Second Edition.* Society for Industrial and Applied Mathematics, 2002.

[28] D. J. Higham and N. J. Higham. *MATLAB guide*. Society for Industrial and Applied Mathematics, 2000.

[29] R. A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1996.

[30] E. R. Jessup. *A case against a divide and conquer approach to the non-symmetric eigenvalue problem.* Applied Numerical Mathematics, 12:403-420, 1993.

[31] Z. S. Liu. *On the extended HR algorithm*. Technical Report PAM-564, Center for Pure and Applied Mathematics, University of California, Berkeley, CA, USA, 1992.

[32] H. Lütkepohl. *Handbook of matrices*. John Wiley and Sons, Inc., 1996.

[33] *MATLAB*. The MathWorks, Inc., Natick, Massachusetts, USA. `http://www.mathworks.com`.

[34] M. Marcus and H. Ming. *A survey of matrix theory and matrix inequalities*. Allyn and Bacon, Inc., Boston, USA, 1964.

[35] B. N. Parlett. *The development and use of methods of LR type*. SIAM Rev., 6:275-295, 1964.

[36] B. N. Parlett. *Canonical Decomposition of Hessenberg Matrices*. Mathematics of Computation, 98(vol.21):223-227, 1967.

[37] B. N. Parlett. *Global Convergence of the Basic QR algorithm on Hessenberg Matrices*. Mathematics of Computation, 104(vol.22):803-817, 1968.

[38] B. N. Parlett. *The Symmetric Eigenvalue Problem*. Prentice Hall, Engelwood Cliffs, New Jersey, 1980.

[39] B. N. Parlett. *Reduction to tridiagonal form and minimal realizations*, SIAM Jornal on Matrix Analysis, 13:567-593, 1992.

[40] B. N. Parlett. *The new qd algorithms*. Acta Numerica, 459-491, 1995.

[41] B. N. Parlett. *What Hadamard Missed*. Unpublished Technical Report, 1996.

[42] B. N. Parlett. *Spectral sensitivity of products of bidiagonals*. Linear Algebra and Its Applications, 275-276:417-431, 1998.

[43] B. N. Parlett and Osni A. Marques. *An implementation of the dqds algorithm*. Linear Algebra and Its Applications, 309:217-259, 2000.

[44] L. Pasquini. *Accurate computation of the zeros of the generalized Bessel polynomials*. Numerische Mathematic, 86:507-538, 2000.

[45] R. Plato. *Concise Numerical Mathematics*. Graduate Studies in Mathematics, volume 57. American Mathematical Society.

[46] H. Rutishauser. *Der Quotienten-Differenzen-Algorithmus*. Z. angew. Math. Physik 5:233-251, 1954. Cited in [55].

[47] H. Rutishauser. *Der Quotienten-Differenzen-Algorithmus*. Mitt. Inst. angew. Math. ETH, no.7, Birkhäuser, Basel, 1957. Cited in [55].

[48] H. Rutishauser. *Solution of eigenvalue problems with the LR-transformation*. National Bureau of Standards Applied Mathematics series 49:47-81, 1958.

[49] H. Rutishauser and H.R. Schwarz. *The LR transformation method for symmetric matrices*. Numerische Mathematic, 5:273-289, 1963.

[50] R. B. Sidje and K.Burrage. *QRT: A QR-based tridiagonalization algorithm for nonsymmetric matrices*. SIAM J. Matrix Anal. Appl., 26:878-900, 2005.

[51] G. W. Stewart. *Introduction to Matrix Computations*. Academic Press, Inc. (London) Ltd., 1973.

[52] G. W. Stewart and Ji-guang Sun. *Matrix Perturbation Theory*. Academic Press, Inc., 1990.

[53] L. N. Trefethen. *Three mysteries of Gaussian elimination*. ACM SIGNUM Newsletter, 20:2-5, 1985. Cited in [27].

[54] H. Xu. *The relation between the QR and LR algorithms*. SIAM J. Matrix Anal. Appl., 19(vol.2):551-555, 1998.

[55] D. S. Watkins. *QR-like algorithms - An overview of convergence theory and practice*. Lectures in Applied Mathematics, 32:879-893, 1996.

[56] D. S. Watkins and L. Elsner. *Convergence of algorithms of decomposition type for the eigenvalue problem*. Linear Algebra and Its Applications, 143:19-47, 1991.

[57] J. Wilkinson. *Calculation of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection*. Numerische Mathematic, 4:362-367, 1962.

[58] J. Wilkinson. *Rounding Errors in Algebraic Processes*. Prentice-Hall, Englewood Cliffs, New Jersey, 1963.

[59] J. Wilkinson. *Convergence of the LR, QR, and related algorithms*. Computing Journal, 8:77-84, 1965.

[60] J. Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, 1965.

[61] J. Wilkinson. *Global convergence of tridiagonal QR algorithm with origin shifts*. Linear Algebra and Its Applications, 1:409-420, 1968.

[62] Wolfram Mathematica, Documentation Center. `http://reference.wolfram.com/mathematica/guide/Mathematica.html`

[63] Yao Yang. *Error Analysis of the qds and dqds Algorithms.* Ph.D thesis, University of California, Berkeley, 1994.

[64] Z. Wu. *The Triple dqds Algorithm for Complex Eigenvalues.* Ph.D thesis, University of California, Berkeley, 1996.