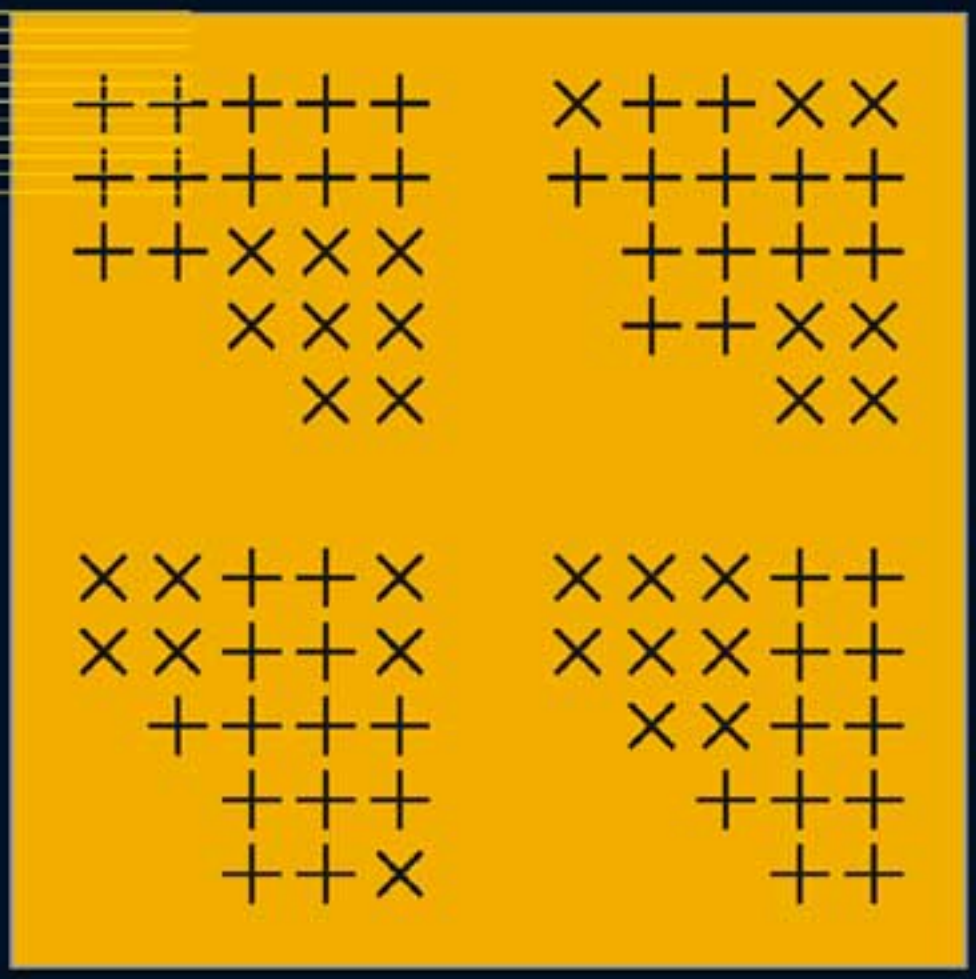
Lecture Notes in Computational Science and Engineering

46



Daniel Kressner

Numerical Methods for General and Structured Eigenvalue Problems

Springer

Lecture Notes
in Computational Science
and Engineering

46

Daniel Kressner

Numerical Methods for General and Structured Eigenvalue Problems

With 32 Figures and 10 Tables

Springer

Daniel Kressner
Institut für Mathematik, MA 4-5
Technische Universität Berlin
10623 Berlin, Germany
email: kressner@math.tu-berlin.de

Library of Congress Control Number: 2005925886

Mathematics Subject Classification (2000): 65-02, 65F15, 65F35, 65Y20, 65F50, 15A18, 93B60

ISSN 1439-7358
ISBN-10 3-540-24546-4 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-24546-9 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media
springeronline.com

Printed in The Netherlands

Typesetting: by the author using a Springer TEX macro package
Cover design: *design & production*, Heidelberg

Printed on acid-free paper SPIN: 11360506 41/TechBooks - 5 4 3 2 1 0

Immer wenn es regnet...

Preface

The purpose of this book is to describe recent developments in solving eigenvalue problems, in particular with respect to the QR and QZ algorithms as well as structured matrices.

Outline

Mathematically speaking, the eigenvalues of a square matrix A are the roots of its characteristic polynomial $\det(A - \lambda I)$. An invariant subspace is a linear subspace that stays invariant under the action of A. In realistic applications, it usually takes a long process of simplifications, linearizations and discretizations before one comes up with the problem of computing the eigenvalues of a matrix. In some cases, the eigenvalues have an intrinsic meaning, e.g., for the expected long-time behavior of a dynamical system; in others they are just meaningless intermediate values of a computational method. The same applies to invariant subspaces, which for example can describe sets of initial states for which a dynamical system produces exponentially decaying states.

Computing eigenvalues has a long history, dating back to at least 1846 when Jacobi [172] wrote his famous paper on solving symmetric eigenvalue problems. Detailed historical accounts of this subject can be found in two papers by Golub and van der Vorst [140, 327].

Chapter 1 of this book is concerned with the QR algorithm, which was introduced by Francis [128] and Kublanovskaya [206] in 1961–1962, partly based on earlier work by Rutishauser [278]. The QR algorithm is a general-purpose, numerically backward stable method for computing all eigenvalues of a non-symmetric matrix. It has undergone only a few modification during the following 40 years, see [348] for a complete overview of the practical QR algorithm as it is currently implemented in LAPACK [10, 17]. An award-winning improvement was made in 2002 when Braman, Byers, and Mathias [62] presented their aggressive early deflation strategy. The combination of this deflation strategy with a tiny-bulge multishift QR algorithm [61, 208] leads to

a variant of the QR algorithm, which can, for sufficiently large matrices, require less than 10% of the computing time needed by the current LAPACK implementation. Similar techniques can also be used to significantly improve the performance of the post-processing step necessary to compute invariant subspaces from the output of the QR algorithm. Besides these algorithmic improvements, Chapter 1 summarizes well-known and also some recent material related to the perturbation analysis of eigenvalues and invariant subspaces; local and global convergence properties of the QR algorithm; and the failure of the large-bulge multishift QR algorithm in finite-precision arithmetic.

The subject of Chapter 2 is the QZ algorithm, a popular method for computing the generalized eigenvalues of a matrix pair (A, B), i.e., the roots of the bivariate polynomial $\det(\beta A - \alpha B)$. The QZ algorithm was developed by Moler and Stewart [248] in 1973. Its probably most notable modification has been the high-performance pipelined QZ algorithm developed by Dackland and Kågström [96]. One topic of Chapter 2 is the use of Householder matrices within the QZ algorithm. The wooly role of infinite eigenvalues is investigated and a tiny-bulge multishift QZ algorithm with aggressive early deflation in the spirit of [61, 208] is described. Numerical experiments illustrate the performance improvements to be gained from these recent developments.

This book is not so much about solving large-scale eigenvalue problems. The practically important aspect of parallelization is completely omitted; we refer to the ScaLAPACK users' guide [49]. Also, methods for computing a few eigenvalues of a large matrix, such as Arnoldi, Lanczos or Jacobi-Davidson methods, are only partially covered. In Chapter 3, we focus on a descendant of the Arnoldi method, the recently introduced Krylov-Schur algorithm by Stewart [307]. Later on, in Chapter 4, it is explained how this algorithm can be adapted to some structured eigenvalue problems in a considerably simple manner. Another subject of Chapter 3 is the balancing of sparse matrices for eigenvalue computations [91].

In many cases, the eigenvalue problem under consideration is known to be structured. Preserving this structure can help preserve induced eigenvalue symmetries in finite-precision arithmetic and may improve the accuracy and efficiency of an eigenvalue computation. Chapter 4 provides an overview of some of the recent developments in the area of structured eigenvalue problems. Particular attention is paid to the concept of structured condition numbers for eigenvalues and invariant subspaces. A detailed treatment of theory, algorithms and applications is given for product, Hamiltonian and skew-Hamiltonian eigenvalue problems, while other structures (skew-symmetric, persymmetric, orthogonal, palindromic) are only briefly discussed.

Appendix B contains an incomplete list of publicly available software for solving general and structured eigenvalue problems. A more complete and regularly updated list can be found at `http://www.cs.umu.se/~kressner/book.php`, the web page of this book.

Prerequisites

Readers of this text need to be familiar with the basic concepts from numerical analysis and linear algebra. Those are covered by any of the text books [103, 141, 304, 305, 354]. Concepts from systems and control theory are occasionally used; either because an algorithm for computing eigenvalues is better understood in a control theoretic setting or such an algorithm can be used for the analysis and design of linear control systems. Knowledge of systems and control theory is not assumed, everything that is needed can be picked up from Appendix A, which contains a brief introduction to this area. Nevertheless, for getting a more complete picture, it might be wise to complement the reading with a state space oriented book on control theory. The monographs [148, 265, 285, 329, 368] are particularly suited for this purpose with respect to content and style of presentation.

Acknowledgments

This book is largely based on my PhD thesis and, once again, I thank all who supported the writing of the thesis, in particular my supervisor Volker Mehrmann and my parents. Turning the thesis into a book would not have been possible without the encouragement and patience of Thanh-Ha Le Thi from Springer in Heidelberg. I have benefited a lot from ongoing joint work and discussions with Ulrike Baur, Peter Benner, Ralph Byers, Heike Faßbender, Michiel Hochstenbach, Bo Kågström, Michael Karow, Emre Mengi, and Françoise Tisseur. Furthermore, I am indebted to Gene Golub, Robert Granat, Nick Higham, Damien Lemonnier, Jörg Liesen, Christian Mehl, Bor Plestenjak, Christian Schröder, Vasile Sima, Valeria Simoncini, Tanja Stykel, Ji-guang Sun, Paul Van Dooren, Krešimir Veselić, David Watkins, and many others for helpful and illuminating discussions. The work on this book was supported by the DFG Research Center MATHEON "Mathematics for key technologies" in Berlin.

Berlin, *Daniel Kressner*
April 2005

Contents

1

The QR Algorithm

Z'n eigenwaarde? Heeft ie een minderwaardigheitscomplex?
—Paul Smit [290]

```
Warning: SCHUR did not converge at index = 4.
```
—MATLAB's response to

```
schur([   0    90     0   300; ...
      -4e+9     0  -300     0; ...
          0  -300     0  4e+9; ...
          0     0   -90     0 ])
```

The QR algorithm is a numerically backward stable method for computing eigenvalues and invariant subspaces of a real or complex matrix. Being developed by Francis [128] and Kublanovskaya [206] in the beginning of the 1960's, the QR algorithm has withstood the test of time and is still the method of choice for small- to medium-sized nonsymmetric matrices. Of course, it has undergone significant improvements since then but the principles remain the same. The purpose of this chapter is to provide an overview of all the ingredients that make the QR algorithm work so well and recent research directions in this area.

Dipping right into the subject, the organization of this chapter is as follows. Section 1.1 is used to introduce the standard eigenvalue problem and the associated notions of invariant subspaces and (real) Schur decompositions. In Section 1.2, we summarize and slightly extend existing perturbation results for eigenvalues and invariant subspaces. The very basic, explicit shifted QR iteration is introduced in the beginning of Section 1.3. In the subsequent subsection, Section 1.3.1, known results on the convergence of the QR iteration are summarized and illustrated. The other subsections are concerned with important implementation details such as preliminary reduction to Hessenberg form, implicit shifting and deflation, which eventually leads to the implicit shifted QR algorithm as it is in use nowadays, see Algorithm 3. In Section 1.3.6, the above-quoted example, for which the QR algorithm fails to converge in a reasonable number of iterations, is explained in more detail. In Section 1.4, we recall balancing and its merits on subsequent eigenvalue computations. Block algorithms, aiming at high performance, are the subject of Section 1.5. First, in Sections 1.5.1 and 1.5.2, the standard block algorithm for reducing a general matrix to Hessenberg form, (implicitly) based on compact WY representations, is described. Deriving a block QR algorithm is a more

subtle issue. In Sections 1.5.3 and 1.5.4, we show the limitations of an approach solely based on increasing the size of bulges chased in the course of a QR iteration. These limitations are avoided if a large number of shifts is distributed over a tightly coupled chain of tiny bulges, yielding the tiny-bulge multishift QR algorithm described in Section 1.5.5. Further performance improvements can be obtained by applying a recently developed so called aggressive early deflation strategy, which is the subject of Section 1.6. To complete the picture, Section 1.7 is concerned with the computation of selected invariant subspaces from a real Schur decomposition. Finally, in Section 1.8, we demonstrate the relevance of recent improvements of the QR algorithm for practical applications by solving a certain linear-quadratic optimal control problem.

Most of the material presented in this chapter is of preparatory value for subsequent chapters but it may also serve as an overview of recent developments related to the QR algorithm.

1.1 The Standard Eigenvalue Problem

The *eigenvalues* of a matrix $A \in \mathbb{R}^{n\times n}$ are the roots of its characteristic polynomial $\det(A - \lambda I)$. The set of all eigenvalues will be denoted by $\lambda(A)$. A nonzero vector $x \in \mathbb{C}^n$ is called an *(right) eigenvector* of A if it satisfies $Ax = \lambda x$ for some eigenvalue $\lambda \in \lambda(A)$. A nonzero vector $y \in \mathbb{C}^n$ is called a *left eigenvector* if it satisfies $y^H A = \lambda y^H$. Spaces spanned by eigenvectors remain invariant under multiplication by A, in the sense that

$$\operatorname{span}\{Ax\} = \operatorname{span}\{\lambda x\} \subseteq \operatorname{span}\{x\}.$$

This concept generalizes to higher-dimensional spaces. A subspace $\mathcal{X} \subset \mathbb{C}^n$ with $A\mathcal{X} \subset \mathcal{X}$ is called a *(right) invariant subspace* of A. Correspondingly, $\mathcal{Y}^H A \subseteq \mathcal{Y}^H$ characterizes a *left invariant subspace* $\mathcal{Y}$. If the columns of X form a basis for an invariant subspace $\mathcal{X}$, then there exists a unique matrix A_{11} satisfying $AX = XA_{11}$. The matrix A_{11} is called the *representation of A with respect to X*. It follows that $\lambda(A_{11}) \subseteq \lambda(A)$ is independent of the choice of basis for $\mathcal{X}$. A nontrivial example is an invariant subspace belonging to a complex conjugate pair of eigenvalues.

Example 1.1. Let $\lambda = \lambda_1 + \mathrm{i}\lambda_2$ with $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}\backslash\{0\}$ be a complex eigenvalue of $A \in \mathbb{R}^{n\times n}$. If $x = x_1 + \mathrm{i}x_2$ is an eigenvector belonging to λ with $x_1, x_2 \in \mathbb{R}^n$, then we find that

$$Ax_1 = \lambda_1 x_1 - \lambda_2 x_2, \quad Ax_2 = \lambda_2 x_1 + \lambda_1 x_2.$$

Note that x_1, x_2 are linearly independent, since otherwise the two above relations imply $\lambda_2 = 0$. This shows that $\operatorname{span}\{x_1, x_2\}$ is a two-dimensional invariant subspace of A admitting the representation

$$A\begin{bmatrix} x_1, x_2 \end{bmatrix} = \begin{bmatrix} x_1, x_2 \end{bmatrix} \begin{bmatrix} \lambda_1 & \lambda_2 \\ -\lambda_2 & \lambda_1 \end{bmatrix}.$$

$\Diamond$

Now, let the columns of the matrices X and $X_\perp$ form orthonormal bases for an invariant subspace $\mathcal{X}$ and its orthogonal complement $\mathcal{X}^\perp$, respectively. Then $U = [X, X_\perp]$ is a unitary matrix and

$$U^H A U = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad \lambda(A) = \lambda(A_{11}) \cup \lambda(A_{22}). \tag{1.1}$$

Such a block triangular decomposition is called *block Schur decomposition* and the matrix $\left[\begin{smallmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{smallmatrix}\right]$ is a *block Schur form* of A. Subsequent application of this decomposition to the blocks A_{11} and A_{22} leads to a triangular decomposition, called *Schur decomposition*. Unfortunately, this decomposition will be complex unless all eigenvalues of A are real. A real alternative is provided by the following well-known theorem, which goes back to Murnaghan and Wintner [252]. It can be proven by successively combining the block decomposition (1.1) with Example 1.1.

Theorem 1.2 (Real Schur decomposition). *Let $A \in \mathbb{R}^{n\times n}$, then there exists an orthogonal matrix Q so that $Q^T A Q = T$ with T in* real Schur form*:*

$$T = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1m} \\ 0 & T_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_{m-1,m} \\ 0 & \cdots & 0 & T_{mm} \end{bmatrix},$$

where all diagonal blocks of T are of order one or two. Scalar blocks contain the real eigenvalues and two-by-two blocks contain the complex conjugate eigenvalue pairs of A.

The whole purpose of the QR algorithm is to compute such a Schur decomposition. Once it has been computed, the eigenvalues of A can be easily obtained from the diagonal blocks of T. Also, the leading k columns of Q span a k-dimensional invariant subspace of A provided that the $(k+1, k)$ entry of T is zero. The representation of A with respect to this basis is given by the leading principal $k \times k$ submatrix of T. Bases for other invariant subspaces can be obtained by reordering the diagonal blocks of T, see Section 1.7.

1.2 Perturbation Analysis

Any numerical method for computing the eigenvalues of a general matrix $A \in \mathbb{R}^{n\times n}$ is affected by rounding errors, which are a consequence of working in finite-precision arithmetic. Another, sometimes less important, source of errors are truncation errors caused by the fact that any eigenvalue computation is necessarily based on iterations. The best we can hope for is that our favorite

algorithm computes the *exact* eigenvalues and invariant subspaces of a *perturbed* matrix $A + E$ where $\|E\|_2 \leq \varepsilon \|A\|_2$ and ε is not much larger than the *unit roundoff* $\mathbf{u}$. Such an algorithm is called numerically *backward stable* and the matrix E is called the *backward error*. Fortunately, almost all algorithms discussed in this book are backward stable. Hence, we can always measure the quality of our results by bounding the effects of small backward errors on the computed quantities. This is commonly called *perturbation analysis* and this section briefly reviews the perturbation analysis for the standard eigenvalue problem. More details can be found, e.g., in the book by Stewart and Sun [308], and a comprehensive report by Sun [317].

1.2.1 Spectral Projectors and Separation

Two quantities play a prominent role in perturbation bounds for eigenvalues and invariant subspaces, the spectral projector P and the separation of two matrices A_{11} and A_{22}, $\operatorname{sep}(A_{11}, A_{22})$.

Suppose we have a block Schur decomposition

$$U^H A U = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}. \tag{1.2}$$

The *spectral projector* belonging to the eigenvalues of $A_{11} \in \mathbb{C}^{k \times k}$ is defined as

$$P = U \begin{bmatrix} I_k & R \\ 0 & 0 \end{bmatrix} U^H, \tag{1.3}$$

where R satisfies the matrix equation

$$A_{11} R - R A_{22} = A_{12}. \tag{1.4}$$

If we partition $U = [X, X_\perp]$ with $X \in \mathbb{C}^{n \times k}$ then P is an oblique projection onto the invariant subspace $\mathcal{X} = \operatorname{range}(X)$. Equation (1.4) is called a *Sylvester equation* and our working assumption will be that it is uniquely solvable.

Lemma 1.3 ([308, Thm. V.1.3]). *The Sylvester equation (1.4) has a unique solution R if and only if A_{11} and A_{22} have no eigenvalues in common, i.e., $\lambda(A_{11}) \cap \lambda(A_{22}) = \emptyset$.*

Proof. Consider the linear operator $\mathbf{T} : \mathbb{C}^{k \times (n-k)} \to \mathbb{C}^{k \times (n-k)}$ defined by

$$\mathbf{T} : R \mapsto A_{11} R - R A_{22}.$$

We will make use of the fact that equation (1.4) is uniquely solvable if and only if $\operatorname{kernel}(\mathbf{T}) = \{0\}$.

Suppose that λ is a common eigenvalue of A_{11} and A_{22}. Let v and w be corresponding left and right eigenvectors of A_{11} and A_{22}, respectively. Then the nonzero matrix vw^H satisfies $\mathbf{T}(vw^H) = 0$.

Conversely, assume there is a matrix $R \in \text{kernel}(\mathbf{T})\backslash\{0\}$. Consider a singular value decomposition $R = V_1 \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} V_2^H$, where $\Sigma \in \mathbb{C}^{l\times l}$ is nonsingular and V_1, V_2 are unitary. If we partition

$$V_1^H A_{11} V_1 = \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix}, \quad V_2^H A_{22} V_2 = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix},$$

where $X_{11}, Y_{11} \in \mathbb{C}^{l\times l}$, then $\mathbf{T}(R) = 0$ implies that the blocks X_{21} and Y_{12} must vanish. Furthermore, $Y_{11} = \Sigma^{-1} X_{11} \Sigma$ showing that the matrices A_{11} and A_{22} have the $l \geq 1$ eigenvalues of X_{11} in common. □

Note that the eigenvalues of $A_{11} = X^H A X$ and $A_{22} = X_\perp^H A X_\perp$ remain invariant under a change of basis for $\mathcal{X}$ and $\mathcal{X}^\perp$, respectively. Hence, we may formulate the unique solvability of the Sylvester equation (1.4) as an intrinsic property of the invariant subspace $\mathcal{X}$.

Definition 1.4. *Let $\mathcal{X}$ be an invariant subspace of A, and let the columns of X and $X_\perp$ form orthonormal bases for $\mathcal{X}$ and $\mathcal{X}^\perp$, respectively. Then $\mathcal{X}$ is called* simple *if*

$$\lambda(X^H A X) \cap \lambda(X_\perp^H A X_\perp) = \emptyset.$$

The spectral projector P defined in (1.3) has a number of useful properties. Its first k columns span the right invariant subspace and its first k rows span the left invariant subspace belonging to $\lambda(A_{11})$. Conversely, if the columns of X and Y form bases for the right and left invariant subspaces, then

$$P = X(Y^H X)^{-1} Y^H. \tag{1.5}$$

The norm of P can be expressed as

$$\|P\|_2 = \sqrt{1 + \|R\|_2^2}, \quad \|P\|_F = \sqrt{k + \|R\|_F^2}. \tag{1.6}$$

In the proof of Lemma 1.3 we have made use of a certain linear map, the *Sylvester operator*

$$\mathbf{T} : R \mapsto A_{11} R - R A_{22}. \tag{1.7}$$

The *separation* of two matrices A_{11} and A_{22}, $\text{sep}(A_{11}, A_{22})$, is defined as the smallest singular value of $\mathbf{T}$:

$$\text{sep}(A_{11}, A_{22}) := \min_{R\neq 0} \frac{\|\mathbf{T}(R)\|_F}{\|R\|_F} = \min_{R\neq 0} \frac{\|A_{11} R - R A_{22}\|_F}{\|R\|_F}. \tag{1.8}$$

If $\mathbf{T}$ is invertible then $\text{sep}(A_{11}, A_{22}) = 1/\|\mathbf{T}^{-1}\|$, where $\|\cdot\|$ is the norm on the space of linear operators $\mathbb{C}^{k\times(n-k)} \to \mathbb{C}^{k\times(n-k)}$ that is induced by the Frobenius norm on $\mathbb{C}^{k\times(n-k)}$. Yet another formulation is obtained by expressing $\mathbf{T}$ in terms of Kronecker products. The *Kronecker product* '$\otimes$' of two matrices $X \in \mathbb{C}^{k\times l}$ and $Y \in \mathbb{C}^{m\times n}$ is the $km \times ln$ matrix

$$X \otimes Y := \begin{bmatrix} x_{11}Y & x_{12}Y & \cdots & x_{1l}Y \\ x_{21}Y & x_{22}Y & \cdots & x_{2l}Y \\ \vdots & \vdots & & \vdots \\ x_{k1}Y & x_{k2}Y & \cdots & x_{kl}Y \end{bmatrix}.$$

The *"vec" operator* stacks the columns of a matrix $Y \in \mathbb{C}^{m\times n}$ into one long vector $\mathrm{vec}(Y) \in \mathbb{C}^{mn}$ in their natural order. The Kronecker product and the vec operator have many useful properties, see [171, Chap. 4]. For our purpose it is sufficient to know that

$$\mathrm{vec}(\mathbf{T}(R)) = K_{\mathbf{T}} \cdot \mathrm{vec}(R), \tag{1.9}$$

where the $(n-k)k \times (n-k)k$ matrix $K_{\mathbf{T}}$ is given by

$$K_{\mathbf{T}} = I_{n-k} \otimes A_{11} - A_{22}^T \otimes I_k.$$

Note that A_{22}^T denotes the complex transpose of A_{22}. Combining (1.8) with (1.9) yields a direct formula for evaluating the separation:

$$\mathrm{sep}(A_{11}, A_{22}) = \sigma_{\min}(K_{\mathbf{T}}) = \sigma_{\min}(I \otimes A_{11} - A_{22}^T \otimes I), \tag{1.10}$$

where $\sigma_{\min}$ denotes the smallest singular value of a matrix. Note that the singular values of the Sylvester operator $\mathbf{T}$ remain the same if the roles of A_{11} and A_{22} in the definition (1.7) are interchanged. In particular,

$$\mathrm{sep}(A_{11}, A_{22}) = \mathrm{sep}(A_{22}, A_{11}).$$

Separation and spectral projectors are not unrelated, for example a direct consequence of (1.6) and the definition of sep is the inequality

$$\|P\|_2 \leq \sqrt{1 + \frac{\|A_{12}\|_F^2}{\mathrm{sep}^2(A_{11}, A_{22})}}, \tag{1.11}$$

see also [308].

1.2.2 Eigenvalues and Eigenvectors

An eigenvalue λ is called *simple* if λ is a simple root of the characteristic polynomial $\det(\lambda I - A)$. We will see that simple eigenvalues and eigenvectors of $A + E$ depend analytically on the entries of E in a neighborhood of $E = 0$. This allows us to expand these quantities in power series in the entries of E, leading to so called *perturbation expansions*. The respective first order terms of these expansions are presented in the following theorem, perturbation expansions of higher order can be found, e.g., in [26, 317].

Theorem 1.5. *Let λ be a simple eigenvalue of $A \in \mathbb{R}^{n\times n}$ with normalized right and left eigenvectors x and y, respectively. Let $E \in \mathcal{B}(0)$ be a perturbation of A, where $\mathcal{B}(0) \subset \mathbb{C}^{n\times n}$ is a sufficiently small open neighborhood of the origin. Then there exist analytic functions $f_\lambda : \mathcal{B}(0) \to \mathbb{C}$ and $f_x : \mathcal{B}(0) \to \mathbb{C}^n$ so that $\lambda = f_\lambda(0)$, $x = f_x(0)$, and $\hat{\lambda} = f_\lambda(E)$ is an eigenvalue of $A + E$ with eigenvector $\hat{x} = f_x(E)$. Moreover $x^H(\hat{x} - x) = 0$, and we have the expansions*

$$\hat{\lambda} = \lambda + \frac{1}{y^H x} y^H E x + \mathcal{O}(\|E\|^2), \tag{1.12}$$

$$\hat{x} = x - X_\perp (X_\perp^H (A - \lambda I) X_\perp)^{-1} X_\perp^H E x + \mathcal{O}(\|E\|^2), \tag{1.13}$$

where the columns of $X_\perp$ form an orthonormal basis for $\operatorname{span}\{x\}^\perp$.

Proof. Let us define the analytic function

$$f(E, \hat{x}, \hat{\lambda}) = \begin{bmatrix} (A + E)\hat{x} - \hat{\lambda}\hat{x} \\ x^H(\hat{x} - x) \end{bmatrix}.$$

If this function vanishes then $\hat{\lambda}$ is an eigenvalue of $A + E$ with the eigenvector $\hat{x}$. The Jacobian of f with respect to $(\hat{x}, \hat{\lambda})$ at $(0, x, \lambda)$ is given by

$$J = \frac{\partial f}{\partial(\hat{x}, \hat{\lambda})}\Big|_{(0,x,\lambda)} = \begin{bmatrix} A - \lambda I & -x \\ x^H & 0 \end{bmatrix}.$$

The fact that λ is simple implies that J is invertible with

$$J^{-1} = \begin{bmatrix} X_\perp (X_\perp^H (A - \lambda I) X_\perp)^{-1} X_\perp^H & x \\ -y^H/(y^H x) & 0 \end{bmatrix}.$$

Hence, the implicit function theorem (see, e.g., [196]) guarantees the existence of functions f_λ and f_x on a sufficiently small open neighborhood of the origin, with the properties stated in the theorem. □

Eigenvalues

By bounding the effects of E in the perturbation expansion (1.12) we get the following perturbation bound for eigenvalues:

$$\begin{aligned} |\hat{\lambda} - \lambda| &= \frac{|y^H E x|}{|y^H x|} + \mathcal{O}(\|E\|^2) \\ &\le \frac{\|E\|_2}{|y^H x|} + \mathcal{O}(\|E\|^2) \\ &= \|P\|_2 \, \|E\|_2 + \mathcal{O}(\|E\|^2), \end{aligned}$$

where $P = (xy^H)/(y^H x)$ is the spectral projector belonging to λ, see (1.5). Note that the utilized upper bound $|y^H E x| \le \|E\|_2$ is attained by $E = \varepsilon y x^H$

for any scalar ε. This shows that the *absolute condition number for a simple eigenvalue* λ can be written as

$$c(\lambda) := \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \sup_{\|E\|_2 \leq \varepsilon} |\hat{\lambda} - \lambda| = \|P\|_2. \tag{1.14}$$

Note that the employed perturbation $E = \varepsilon y x^H$ cannot be chosen to be real unless the eigenvalue λ itself is real. But if A is real then it is reasonable to expect the perturbation E to adhere to this realness and $c(\lambda)$ might not be the appropriate condition number if λ is complex. This fact has found surprisingly little attention in standard text books on numerical linear algebra, which can probably be attributed to the fact that restricting the set of perturbations to be real can only have a limited influence on the condition number.

To see this, let us define the *absolute condition number for a simple eigenvalue* λ *with respect to real perturbations* as follows:

$$c^{\mathbb{R}}(\lambda) := \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \sup_{\substack{\|E\|_F \leq \varepsilon \\ E \in \mathbb{R}^{n \times n}}} |\hat{\lambda} - \lambda|. \tag{1.15}$$

For real λ, we have already seen that one can choose a real rank-one perturbation that attains the supremum in (1.15) with $c^{\mathbb{R}}(\lambda) = c(\lambda) = \|P\|_2$. For complex λ, we clearly have $c^{\mathbb{R}}(\lambda) \leq c(\lambda) = \|P\|_2$ but it is not clear how much $c(\lambda)$ can exceed $c^{\mathbb{R}}(\lambda)$. The following theorem shows that the ratio $c^{\mathbb{R}}(\lambda)/c(\lambda)$ can be bounded from below by $1/\sqrt{2}$.

Theorem 1.6 ([82]). *Let $\lambda \in \mathbb{C}$ be a simple eigenvalue of $A \in \mathbb{R}^{n \times n}$ with normalized right and left eigenvectors $x = x_R + \mathrm{i}x_I$ and $y = y_R + \mathrm{i}y_I$, respectively, where $x_R, x_I, y_R, y_I \in \mathbb{R}^n$. Then the condition number $c^{\mathbb{R}}(\lambda)$ as defined in (1.15) satisfies*

$$c^{\mathbb{R}}(\lambda) = \frac{1}{|y^H x|} \sqrt{\frac{1}{2} + \sqrt{\frac{1}{4}(b^T b - c^T c)^2 + (b^T c)^2}},$$

where $b = x_R \otimes y_R + x_I \otimes y_I$ and $c = x_I \otimes y_R - x_R \otimes y_I$. In particular, we have the inequality

$$c^{\mathbb{R}}(\lambda) \geq c(\lambda)/\sqrt{2}.$$

Proof. The perturbation expansion (1.12) readily implies

$$
\begin{aligned}
c^{\mathbb{R}}(\lambda) &= \lim_{\varepsilon\to 0} \sup\left\{|y^H Ex|/|y^H x| :\ E\in\mathbb{R}^{n\times n}, \|E\|_F \le \varepsilon\right\} \\
&= 1/|y^H x| \cdot \sup\left\{|y^H Ex| :\ E\in\mathbb{R}^{n\times n}, \|E\|_F = 1\right\} \\
&= 1/|y^H x| \cdot \sup_{\substack{E\in\mathbb{R}^{n\times n}\\ \|E\|_F=1}} \left\| \begin{bmatrix} y_R^T E x_R + y_I^T E x_I \\ y_R^T E x_I - y_I^T E x_R \end{bmatrix} \right\|_2 \\
&= 1/|y^H x| \cdot \sup_{\substack{\|\mathrm{vec}(E)\|_2=1\\ E\in\mathbb{R}^{n\times n}}} \left\| \begin{bmatrix} (x_R\otimes y_R)^T \mathrm{vec}(E) + (x_I\otimes y_I)^T \mathrm{vec}(E) \\ (x_I\otimes y_R)^T \mathrm{vec}(E) - (x_R\otimes y_I)^T \mathrm{vec}(E) \end{bmatrix} \right\|_2 \\
&= 1/|y^H x| \cdot \sup_{\substack{\|\mathrm{vec}(E)\|_2=1\\ E\in\mathbb{R}^{n\times n}}} \left\| \begin{bmatrix} (x_R\otimes y_R + x_I\otimes y_I)^T \\ (x_I\otimes y_R - x_R\otimes y_I)^T \end{bmatrix} \mathrm{vec}(E) \right\|_2 . \qquad (1.16)
\end{aligned}
$$

This is a standard linear least-squares problem [48]; the maximum of the second factor is given by the larger singular value of the $n^2 \times 2$ matrix

$$
X = \begin{bmatrix} x_R\otimes y_R + x_I\otimes y_I & x_I\otimes y_R - x_R\otimes y_I \end{bmatrix}. \qquad (1.17)
$$

A vector $\mathrm{vec}(E)$ attaining the supremum in (1.16) is a left singular vector belonging to this singular value. The square of the larger singular value of X is given by the larger root $\theta_\star$ of the polynomial

$$
\det(X^T X - \theta I_2) = \theta^2 - (b^T b + c^T c)\theta + (b^T b)(c^T c) - (b^T c)^2.
$$

Because the eigenvectors x and y are normalized, it can be shown by direct calculation that $b^T b + c^T c = 1$ and $1/4 - (b^T b)(c^T c) = 1/4 \cdot (b^T b - c^T c)^2$. This implies

$$
\theta_\star = \frac{1}{2} + \sqrt{\frac{1}{4}(b^T b - c^T c)^2 + (b^T c)^2},
$$

which concludes the proof. □

For the matrix $A = \left[\begin{smallmatrix} 0 & 1 \\ -1 & 0 \end{smallmatrix}\right]$, we have $c^{\mathbb{R}}(\mathrm{i}) = c^{\mathbb{R}}(-\mathrm{i}) = 1/\sqrt{2}$ and $c(\mathrm{i}) = c(-\mathrm{i}) = 1$, revealing that the bound $c^{\mathbb{R}}(\lambda) \ge c(\lambda)/\sqrt{2}$ can actually be attained. Note that it is the use of the Frobenius norm in the definition (1.15) of $c^{\mathbb{R}}(\lambda)$ that leads to the effect that $c^{\mathbb{R}}(\lambda)$ may become less than the norm of A. A general framework allowing the use of a broad class of norms has been developed by Karow [186] based on the theory of spectral value sets and real μ-functions. Using these results, it can be shown that the bound $c^{\mathbb{R}}(\lambda) \ge c(\lambda)/\sqrt{2}$ remains true if the Frobenius norm in the definition (1.15) of $c^{\mathbb{R}}(\lambda)$ is replaced by the 2-norm [187].

Eigenvectors

Deriving condition numbers for eigenvectors is complicated by the fact that an eigenvector x is not uniquely determined. Measuring the quality of an approximate eigenvector $\hat{x}$ using $\|\hat{x} - x\|_2$ is thus only possible after a suitable

normalization has been applied to x and $\hat{x}$. An alternative is to use $\angle(x, \hat{x})$, the angle between the one-dimensional subspaces spanned by x and $\hat{x}$, see Figure 1.1.

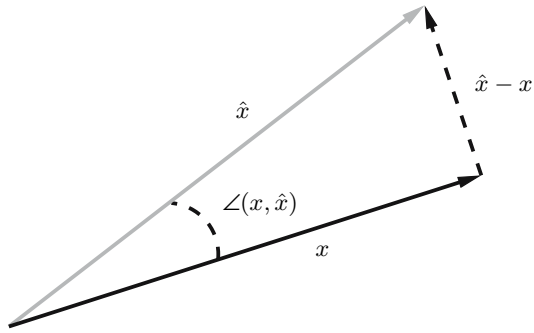


Fig. 1.1. Angle between two vectors.

Corollary 1.7. *Under the assumptions of Theorem 1.5,*

$$\angle(x, \hat{x}) \leq \|(X_\perp^H (A - \lambda I) X_\perp)^{-1}\|_2 \, \|E\|_2 + \mathcal{O}(\|E\|^2).$$

Proof. Using the fact that x is orthogonal to $(\hat{x} - x)$ we have $\tan \angle(x, \hat{x}) = \|\hat{x} - x\|_2$. Expanding arctan yields $\angle(x, \hat{x}) \leq \|\hat{x} - x\|_2 + \mathcal{O}(\|\hat{x} - x\|^3)$, which together with the perturbation expansion (1.13) concludes the proof. □

The *absolute condition number for a simple eigenvector* x can be defined as

$$c(x) := \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \sup_{\|E\|_2 \leq \varepsilon} \angle(x, \hat{x}).$$

If we set $A_{22} = X_\perp^H A X_\perp$ then Corollary 1.7 combined with (1.10) implies

$$c(x) \leq \|(A_{22} - \lambda I)^{-1}\|_2 = \sigma_{\min}^{-1}(A_{22} - \lambda I) = (\operatorname{sep}(\lambda, A_{22}))^{-1}. \tag{1.18}$$

Considering perturbations of the form $E = \varepsilon X_\perp u x^H$, where u is a left singular vector belonging to the smallest singular value of $A_{22} - \lambda I$, it can be shown that the left and right sides of the inequality in (1.18) are actually equal.

1.2.3 Eigenvalue Clusters and Invariant Subspaces

Multiple eigenvalues do not have an expansion of the form (1.12), in fact they may not even be Lipschitz continuous with respect to perturbations of A, as demonstrated by the following example.

Example 1.8 (Bai, Demmel, and McKenney [20]). Let

$$A_\eta = \begin{bmatrix} 0 & 1 & & & 0 \\ & \ddots & \ddots & & \vdots \\ & & \ddots & 1 & \vdots \\ \eta & & & 0 & 0 \\ & & & & 1/2 \end{bmatrix} \in \mathbb{R}^{11\times 11}.$$

For $\eta = 0$, the leading 10-by-10 block is a single Jordan block corresponding to zero eigenvalues. For $\eta \neq 0$, this eigenvalue bifurcates into the ten distinct 10th roots of η. E.g. for $\eta = 10^{-10}$, these bifurcated eigenvalues have absolute value $\eta^{1/10} = 1/10$ showing that they react very sensitively to perturbations of A_0.

On the other hand, if we do not treat the zero eigenvalues of A_0 individually but consider them as a whole cluster of eigenvalues, then the mean of this cluster will be much less sensitive to perturbations. In particular, the mean remains zero no matter which value η takes. ◊

The preceeding example reveals that it can sometimes be important to consider the effect of perturbations on clusters instead of individual eigenvalues. To see this for general matrices, let us consider a block Schur decomposition of the form

$$U^H A U = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad A_{11} \in \mathbb{C}^{k\times k}, \quad A_{22} \in \mathbb{C}^{(n-k)\times(n-k)} \tag{1.19}$$

where the eigenvalues of A_{11} form the eigenvalue cluster of interest. If $\lambda(A_{11})$ only consists of values very close to each other, then the mean of the eigenvalues, $\overline{\lambda(A_{11})} = \operatorname{tr} A_{11}/k$, contains all relevant information. Furthermore, $\overline{\lambda(A_{11})}$ does not suffer from ill-conditioning caused by ill-conditioned eigenvalues of A_{11}. What we need to investigate the sensitivity of $\overline{\lambda(A_{11})}$ is a generalization of the perturbation expansions in Theorem 1.5 to invariant subspaces, see also [313, 317].

Theorem 1.9. *Let A have a block Schur decomposition of the form (1.19) and partition $U = [X, X_\perp]$ so that $\mathcal{X} = \operatorname{range}(X)$ is an invariant subspace belonging to $\lambda(A_{11})$. Let the columns of Y form an orthonormal basis for the corresponding left invariant subspace. Assume $\mathcal{X}$ to be simple and let $E \in \mathcal{B}(0)$ be a perturbation of A, where $\mathcal{B}(0) \subset \mathbb{C}^{n\times n}$ is a sufficiently small open neighborhood of the origin. Then there exist analytic functions $f_{A_{11}} : \mathcal{B}(0) \to \mathbb{C}^{k\times k}$ and $f_X : \mathcal{B}(0) \to \mathbb{C}^{n\times k}$ so that $A_{11} = f_{A_{11}}(0)$, $X = f_X(0)$, and the columns of $\hat{X} = f_X(E)$ span an invariant subspace of $A+E$ corresponding to the representation $\hat{A}_{11} = f_{A_{11}}(E)$. Moreover $X^H(\hat{X} - X) = 0$, and we have the expansions*

$$\hat{A}_{11} = A_{11} + (Y^H X)^{-1} Y^H E X + \mathcal{O}(\|E\|^2), \tag{1.20}$$

$$\hat{X} = X - X_\perp \mathbf{T}^{-1}(X_\perp^H E X) + \mathcal{O}(\|E\|^2), \tag{1.21}$$

with the Sylvester operator $\mathbf{T} : Q \mapsto A_{22}Q - QA_{11}$.

Proof. The theorem is proven by a block version of the proof of Theorem 1.5. In the following, we provide a sketch of the proof and refer the reader to [313] for more details. If

$$f(E, \hat{X}, \hat{A}_{11}) = \begin{bmatrix} (A+E)\hat{X} - \hat{X}\hat{A}_{11} \\ X^H(\hat{X} - X) \end{bmatrix} = 0, \tag{1.22}$$

then range$(\hat{X})$ is an invariant subspace corresponding to the representation $\hat{A}_{11}$. The Jacobian of f with respect to $(\hat{X}, \hat{A}_{11})$ at $(0, X, A_{11})$ can be expressed as a linear matrix operator having the block representation

$$J = \frac{\partial f}{\partial(\hat{X}, \hat{A}_{11})}\Big|_{(0,X,A_{11})} = \begin{bmatrix} \tilde{\mathbf{T}} & -X \\ X^H & 0 \end{bmatrix}$$

with the matrix operator $\tilde{\mathbf{T}} : Z \mapsto AZ - ZA_{11}$. The fact that $\mathcal{X}$ is simple implies the invertibility of the Sylvester operator $\mathbf{T}$ and thus the invertibility of J. In particular, it can be shown that

$$J^{-1} = \begin{bmatrix} X_\perp \mathbf{T}^{-1} X_\perp^H & X \\ -(Y^H X)^{-1} Y^H & 0 \end{bmatrix}.$$

As in the proof of Theorem 1.5, the implicit function theorem guarantees the existence of functions $f_{A_{11}}$ and f_X on a sufficiently small, open neighborhood of the origin, with the properties stated in the theorem. □

We only remark that the implicit equation $f = 0$ in (1.22) can be used to derive Newton and Newton-like methods for computing eigenvectors or invariant subspaces, see, e.g., [102, 264]. Such methods are, however, not treated in this book although they are implicitly present in the QR algorithm [305, p. 418].

Corollary 1.10. *Under the assumptions of Theorem 1.9,*

$$\begin{aligned} \left| \overline{\lambda(\hat{A}_{11})} - \overline{\lambda(A_{11})} \right| &\leq \tfrac{1}{k} \|P\|_2 \, \|E\|_{(1)} + \mathcal{O}(\|E\|^2) \\ &\leq \|P\|_2 \, \|E\|_2 + \mathcal{O}(\|E\|^2), \end{aligned} \tag{1.23}$$

where P is the spectral projector belonging to $\lambda(A_{11})$ and $\|\cdot\|_{(1)}$ denotes the Schatten 1-norm [171] defined as the sum of the singular values.

Proof. The expansion (1.20) yields

$$\begin{aligned} \|\hat{A}_{11} - A_{11}\|_{(1)} &= \|(Y^H X)^{-1} Y^H E X\|_{(1)} + \mathcal{O}(\|E\|^2) \\ &\leq \|(Y^H X)^{-1}\|_2 \, \|E\|_{(1)} + \mathcal{O}(\|E\|^2) \\ &= \|P\|_2 \, \|E\|_{(1)} + \mathcal{O}(\|E\|^2), \end{aligned}$$

where we used (1.5). Combining this inequality with

$$|\operatorname{tr} \hat{A}_{11} - \operatorname{tr} A_{11}| \leq \sum \left|\lambda(\hat{A}_{11} - A_{11})\right| \leq \|\hat{A}_{11} - A_{11}\|_{(1)}$$

concludes the proof. □

Note that the two inequalities in (1.23) are, in first order, equalities for $E = \varepsilon Y X^H$. Hence, the *absolute condition number for the eigenvalue mean* $\bar{\lambda}$ is given by

$$c(\bar{\lambda}) := \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \sup_{\|E\|_2 \leq \varepsilon} \left|\overline{\lambda(\hat{A}_{11})} - \overline{\lambda(A_{11})}\right| = \|P\|_2,$$

which is identical to (1.14) except that the spectral projector P now belongs to a whole cluster of eigenvalues.

In order to obtain condition numbers for invariant subspaces we require a notion of angles or distances between two subspaces.

Definition 1.11. *Let the columns of X and Y form orthonormal bases for the k-dimensional subspaces $\mathcal{X}$ and $\mathcal{Y}$, respectively, and let $\sigma_1 \leq \sigma_2 \leq \cdots \leq \sigma_k$ denote the singular values of $X^H Y$. Then the* canonical angles *between $\mathcal{X}$ and $\mathcal{Y}$ are defined by*

$$\theta_i(\mathcal{X}, \mathcal{Y}) := \arccos \sigma_i, \quad i = 1, \ldots, k.$$

Furthermore, we set $\Theta(\mathcal{X}, \mathcal{Y}) := \operatorname{diag}(\theta_1(\mathcal{X}, \mathcal{Y}), \ldots, \theta_k(\mathcal{X}, \mathcal{Y}))$.

This definition makes sense as the numbers θ_i remain invariant under an orthonormal change of basis for $\mathcal{X}$ or $\mathcal{Y}$, and $\|X^H Y\|_2 \leq 1$ with equality if and only if $\mathcal{X} = \mathcal{Y}$. The largest canonical angle has the geometric characterization

$$\theta_1(\mathcal{X}, \mathcal{Y}) = \max_{\substack{x \in \mathcal{X} \\ x \neq 0}} \min_{\substack{y \in \mathcal{Y} \\ y \neq 0}} \angle(x, y), \tag{1.24}$$

see also Figure 1.2.

It can be shown that any unitarily invariant norm $\|\cdot\|_\gamma$ on $\mathbb{R}^{k \times k}$ defines a unitarily invariant metric d_γ on the space of k-dimensional subspaces via $d_\gamma(\mathcal{X}, \mathcal{Y}) = \|\sin[\Theta(\mathcal{X}, \mathcal{Y})]\|_\gamma$ [308, Sec II.4]. The metric generated by the 2-norm is called the *gap metric* and satisfies

$$d_2(\mathcal{X}, \mathcal{Y}) := \|\sin[\Theta(\mathcal{X}, \mathcal{Y})]\|_2 = \max_{\substack{x \in \mathcal{X} \\ \|x\|_2 = 1}} \min_{y \in \mathcal{Y}} \|x - y\|_2. \tag{1.25}$$

In the case that one of the subspaces is spanned by a non-orthonormal basis, the following lemma provides a useful tool for computing canonical angles.

Lemma 1.12 ([308]). *Let the k-dimensional linear subspaces $\mathcal{X}$ and $\mathcal{Y}$ be spanned by the columns of $[I, 0]^H$, and $[I, Q^H]^H$, respectively. If $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k$ denote the singular values of Q then*

$$\theta_i(\mathcal{X}, \mathcal{Y}) = \arctan \sigma_i, \quad i = 1, \ldots, k.$$

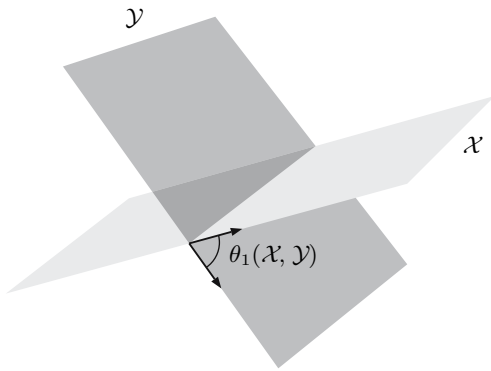


Fig. 1.2. Largest canonical angle between two subspaces.

Proof. The columns of $[I, Q^H]^H (I + Q^H Q)^{-1/2}$ form an orthonormal basis for $\mathcal{Y}$. Consider a singular value decomposition $Q = U\Sigma V^H$ with $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_k)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k$. By Definition 1.11,

$$\cos[\Theta(\mathcal{X}, \mathcal{Y})] = V^H (I + Q^H Q)^{-1/2} V = (I + \Sigma^2)^{1/2}$$

showing that

$$\tan[\Theta(\mathcal{X}, \mathcal{Y})] = (\cos[\Theta(\mathcal{X}, \mathcal{Y})])^{-1} (I - \cos^2[\Theta(\mathcal{X}, \mathcal{Y})])^{-1/2} = \Sigma,$$

which proves the desired result. □

We are now prepared to generalize Corollary 1.7 to invariant subspaces.

Corollary 1.13. *Under the assumptions of Theorem 1.9,*

$$\|\Theta(\mathcal{X}, \hat{\mathcal{X}})\|_F \leq \frac{\|E\|_F}{\operatorname{sep}(A_{11}, A_{22})} + \mathcal{O}(\|E\|^2), \tag{1.26}$$

where $\hat{\mathcal{X}} = \operatorname{range}(\hat{X})$.

Proof. W.l.o.g. we may assume $X = [I, 0]^T$. Since $X^T(\hat{X} - X) = 0$ the matrix $\hat{X}$ must have the form $[I, Q^H]^H$ for some $Q \in \mathbb{C}^{(n-k)\times k}$. Together with the perturbation expansion (1.21) this implies

$$\begin{aligned} \|Q\|_F &= \|\hat{X} - X\|_F = \|X_\perp \mathbf{T}^{-1} (X_\perp^H E X)\|_F + \mathcal{O}(\|E\|^2) \\ &\leq \|\mathbf{T}^{-1}\| \, \|E\|_F + \mathcal{O}(\|E\|^2) = \|E\|_F / \operatorname{sep}(A_{11}, A_{22}) + \mathcal{O}(\|E\|^2). \end{aligned}$$

Inequality (1.26) is proven by applying Lemma 1.12 combined with the expansion $\arctan(z) = z + \mathcal{O}(z^3)$. □

Once again, the derived bound (1.26) is approximately sharp. To see this, let V be a matrix so that $\|V\|_F = 1$ and $\|\mathbf{T}^{-1}(V)\|_F = 1/\operatorname{sep}(A_{11}, A_{22})$. Plugging $E = \varepsilon X_\perp V X^H$ with $\varepsilon > 0$ into the perturbation expansion (1.21) yields

$$\|\Theta(\mathcal{X}, \hat{\mathcal{X}})\|_F = \|\hat{X} - X\|_F + \mathcal{O}(\|\hat{X} - X\|^3) = \varepsilon/\operatorname{sep}(A_{11}, A_{22}) + \mathcal{O}(\varepsilon^2).$$

Hence, we obtain the following *absolute condition number for an invariant subspace* $\mathcal{X}$:

$$c(\mathcal{X}) := \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \sup_{\|E\|_F \leq \varepsilon} \|\Theta(\mathcal{X}, \hat{\mathcal{X}})\|_F = \frac{1}{\operatorname{sep}(A_{11}, A_{22})},$$

see also [299, 301, 308].

On the computation of sep

The separation of two matrices $A_{11} \in \mathbb{C}^{k \times k}$ and $A_{22} \in \mathbb{C}^{(n-k)\times(n-k)}$, $\operatorname{sep}(A_{11}, A_{22})$, equals the smallest singular value of the the $k(n-k) \times k(n-k)$ matrix $K_{\mathbf{T}} = I_{n-k} \otimes A_{11} - A_{22}^T \otimes I_k$. Computing this value using a singular value decomposition of $K_{\mathbf{T}}$ is costly in terms of memory and computational time. A much cheaper estimate of sep can be obtained by applying a norm estimator [164, Ch. 14] to $K_{\mathbf{T}}^{-1}$. This amounts to the solution of a few linear equations $K_{\mathbf{T}} x = c$ and $K_{\mathbf{T}}^T x = d$ for particularly chosen right hand sides c and d or, equivalently, the solution of a few Sylvester equations $A_{11}X - XA_{22} = C$ and $A_{11}^T X - X A_{22}^T = D$. This approach becomes particularly attractive if A_{11} and A_{22} are already in (real) Schur form, see [22, 77, 176, 181].

1.2.4 Global Perturbation Bounds

All the perturbation results derived in the previous two sections are of a local nature; the presence of $\mathcal{O}(\|E\|^2)$ terms in the inequalities makes them difficult to interpret for large perturbations. How large is large depends on the matrix in question. Returning to Example 1.8, we see that already for $\eta = 2^{-10} \approx 10^{-3}$ two eigenvalues of the matrix A_η equal $\lambda = 0.5$ despite the fact that $c(\lambda) = 1$.

To avoid such effects, we must ensure that the perturbation lets no eigenvalue in the considered cluster coalesce with an eigenvalue outside the cluster. We will see that this is guaranteed as long as the perturbation E satisfies the bound

$$\|E\|_F < \frac{\operatorname{sep}(A_{11}, A_{22})}{4\|P\|_2}, \tag{1.27}$$

where $\lambda(A_{11})$ contains the eigenvalues of interest and P is the corresponding spectral projector.

An approach taken by Stewart [301] and extended by Demmel [101], see also [89], can be used to derive exact perturbation bounds which are valid

if (1.27) holds. For this purpose, let the matrix A be close to block Schur form in the sense that the block A_{21} in

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

is considerably small. We look for an invertible matrix of the form $W = \begin{bmatrix} I & 0 \\ -Q & I \end{bmatrix}$ such that

$$W^{-1} \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} W = \begin{bmatrix} A_{11} - A_{12}Q & A_{12} \\ A_{21} + QA_{11} - A_{22}Q - QA_{12}Q & A_{22} + QA_{12} \end{bmatrix}$$

is in block Schur form. This is equivalent to require Q to solve the quadratic matrix equation

$$QA_{11} - A_{22}Q - QA_{12}Q = -A_{21}. \tag{1.28}$$

The existence of such a solution is guaranteed if A_{21} is not too large.

Lemma 1.14. *If $\|A_{12}\|_F \, \|A_{21}\|_F < \operatorname{sep}^2(A_{11}, A_{22})/4$ then there exists a solution Q of the quadratic matrix equation (1.28) with*

$$\|Q\|_F < \frac{2\|A_{21}\|_F}{\operatorname{sep}(A_{11}, A_{22})}. \tag{1.29}$$

Proof. The result follows from a more general theorem by Stewart, see [299, 301] or [308, Thm. 2.11]. The proof is based on constructing an iteration

$$Q_0 \leftarrow 0, \quad Q_{i+1} \leftarrow \mathbf{T}^{-1}(A_{21} - Q_i A_{12} Q_i),$$

with the Sylvester operator $\mathbf{T} : Q \mapsto A_{22}Q - QA_{11}$. It is shown that the iterates satisfy a bound below the right hand side of (1.29) and converge to a solution of (1.28). We will use a similar approach in Section 4.1.1 to derive structured condition numbers for invariant subspaces. □

Having obtained a solution Q of (1.28), an orthonormal basis for an invariant subspace $\hat{\mathcal{X}}$ of A is given by

$$\hat{X} = \begin{bmatrix} I \\ -Q \end{bmatrix} (I + Q^H Q)^{-1/2}, \tag{1.30}$$

and the representation of A with respect to $\hat{X}$ is

$$\hat{A}_{11} = (I + Q^H Q)^{1/2}(A_{11} - A_{12}Q)(I + Q^H Q)^{-1/2}. \tag{1.31}$$

This leads to the following global version of Corollary 1.13.

Theorem 1.15. *Let A have a block Schur decomposition of the form*

$$U^H A U = U \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad A_{11} \in \mathbb{R}^{k \times k}, \quad A_{22} \in \mathbb{R}^{(n-k)\times(n-k)},$$

and assume that the invariant subspace $\mathcal{X}$ spanned by the first k columns of U is simple. Let $E \in \mathbb{R}^{n\times n}$ be a perturbation satisfying (1.27). Then there exists an invariant subspace $\hat{\mathcal{X}}$ of $A+E$ so that

$$\|\tan[\Theta(\mathcal{X},\hat{\mathcal{X}})]\|_F < \eta := \frac{4\|E\|_F}{\operatorname{sep}(A_{11},A_{22}) - 4\|P\|_2\,\|E\|_F}, \tag{1.32}$$

where P is the spectral projector for $\lambda(A_{11})$. Moreover, there exists a representation $\hat{A}_{11}$ of $A+E$ with respect to an orthonormal basis for $\hat{\mathcal{X}}$ so that

$$\|\hat{A}_{11} - A_{11}\|_F < \frac{1-\sqrt{1-\eta^2}}{\sqrt{1-\eta^2}}\|A_{11}\|_F + \frac{\eta}{\sqrt{1-\eta^2}}\|A\|_F.$$

Proof. This theorem is a slightly modified version of a result by Demmel [101, Lemma 7.8]. It differs in the respect that Demmel provides an upper bound on $\|\tan[\Theta(\mathcal{X},\hat{\mathcal{X}})]\|_2$ instead of $\|\tan[\Theta(\mathcal{X},\hat{\mathcal{X}})]\|_F$. The following proof, however, is almost identical to the proof in [101].

Note that we may assume w.l.o.g. that A is already in block Schur form and thus $U = I$. First, we will show that the bound (1.27) implies the assumption of Lemma 1.14 for a certain quadratic matrix equation. For this purpose, let R denote the solution of the Sylvester equation $A_{11}R - RA_{22} = A_{12}$ and apply the similarity transformation $W_R = \left[\begin{smallmatrix} I & -R/\|P\|_2 \\ 0 & I/\|P\|_2 \end{smallmatrix}\right]$ to $A+E$:

$$W_R^{-1}(A+E)W_R = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} + \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix}.$$

As $\|[I, \pm R]\|_2 = \|P\|_2$, it can be directly seen that $\|F_{11}\|_F$, $\|F_{12}\|_F$, $\|F_{21}\|_F$ and $\|F_{22}\|_F$ are bounded from above by $\|P\|_2\|E\|_F$. From the definition of sep it follows that

$$\begin{aligned} \operatorname{sep}(A_{11}+F_{11}, A_{22}+F_{22}) &\geq \operatorname{sep}(A_{11},A_{22}) - \|F_{11}\|_F - \|F_{22}\|_F \\ &\geq \operatorname{sep}(A_{11},A_{22}) - 2\|P\|_2\,\|E\|_F \\ &> 2\|P\|_2\,\|E\|_F \geq 0, \end{aligned} \tag{1.33}$$

where the strict inequality follows from (1.27). This implies

$$\|F_{12}\|_F\|F_{21}\|_F \leq (\|P\|_2\,\|E\|_F)^2 < \operatorname{sep}^2(A_{11}+F_{11}, A_{22}+F_{22})/4,$$

showing that the assumption of Lemma 1.14 is satisfied for the quadratic matrix equation

$$Q(A_{11}+F_{11}) - (A_{22}+F_{22})Q - QF_{12}Q = -F_{21}.$$

Consequently, there exists a solution Q satisfying

$$\begin{aligned} \|Q\|_F &< \frac{2\|F_{21}\|_F}{\operatorname{sep}(A_{11}+F_{11}, A_{22}+F_{22})} \leq \frac{2\|P\|_2\,\|E\|_F}{\operatorname{sep}(A_{11},A_{22}) - 2\|P\|_2\,\|E\|_F} \\ &\leq 4\|P\|_2\,\|E\|_F/\operatorname{sep}(A_{11},A_{22}) < 1. \end{aligned}$$

Thus, $A+E$ has an invariant subspace spanned by the columns of the matrix product $W_R \begin{bmatrix} I \\ -Q \end{bmatrix}$. If we replace Q by $\tilde{Q} = Q(\|P\|_2 \cdot I + RQ)^{-1}$ in the definitions of $\hat{X}$ and $\hat{A}_{11}$ in (1.30) and (1.31), respectively, then the columns of $\hat{X}$ form an orthonormal basis for an invariant subspace $\hat{\mathcal{X}}$ of $A+E$ belonging to the representation $\hat{A}_{11}$. We have

$$\begin{aligned}
\|\tilde{Q}\|_F &\le \|Q\|_F \, \|(\|P\|_2 \cdot I + RQ)^{-1}\|_2 \\
&\le \|Q\|_F / (\|P\|_2 - \|R\|_2 \|Q\|_2) \le \|Q\|_F / (\|P\|_2 - \|R\|_2 \|Q\|_F) \\
&\le \frac{4\|E\|_F}{\operatorname{sep}(A_{11}, A_{22}) - 4\|R\|_2 \, \|E\|_F} \\
&< \frac{4\|E\|_F}{\operatorname{sep}(A_{11}, A_{22}) - 4\|P\|_2 \, \|E\|_F} = \eta,
\end{aligned}$$

which combined with Lemma 1.12 proves (1.32).

To prove the second part of the theorem, let $\tilde{Q} = U\Sigma V^H$ be a singular value decomposition [141] with $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_k)$ and $\sigma_1 \ge \cdots \ge \sigma_k$. Using (1.31), with Q replaced by $\tilde{Q}$, we obtain

$$V^H(\hat{A}_{11} - A_{11})V = (I-\Sigma^2)^{1/2}(V^H A_{11} V - V^H A_{12} U \Sigma)(I-\Sigma^2)^{-1/2} - V^H A_{11} V,$$

which shows

$$\|\hat{A}_{11} - A_{11}\|_F \le \frac{1-\sqrt{1-\sigma_1^2}}{\sqrt{1-\sigma_1^2}} \|A_{11}\|_F + \frac{\sigma_1}{\sqrt{1-\sigma_1^2}} \|A_{12}\|_F.$$

Since $\sigma_1 \le \eta$, this concludes the proof. □

Note that the preceeding proof, in particular (1.33), also reveals that the eigenvalues of A_{11} and A_{22} do not coalesce under perturbations that satisfy (1.27).

1.3 The Basic QR Algorithm

The QR iteration, introduced by Francis [128] and Kublanovskaya [206], generates a sequence of orthogonally similar matrices $A_0 \leftarrow A, A_1, A_2, \ldots$ which, under suitable conditions, converges to a nontrivial block Schur form of A. Its name stems from the QR decomposition that is used in the course of an iteration. The second ingredient of the QR iteration is a real-valued polynomial p_i, the so called *shift polynomial*, which must be chosen before each iteration. The QR decomposition of this polynomial applied to the last iterate A_{i-1} is used to determine the orthogonal similarity transformation that yields the next iterate:

$$p_i(A_{i-1}) = Q_i R_i, \qquad \text{(QR decomposition)} \tag{1.34a}$$

$$A_i \leftarrow Q_i^T A_{i-1} Q_i. \tag{1.34b}$$

1.3.1 Local Convergence

In the following we summarize the convergence analysis by Watkins and Elsner [359] of the QR iteration defined by (1.34). The ith iterate of this sequence can be written as

$$A_i = \hat{Q}_i^T A \hat{Q}_i, \quad \hat{Q}_i := Q_1 Q_2 \cdots Q_i.$$

We have already seen that the matrix A_i has block Schur form

$$\begin{bmatrix} A_{11}^{(i)} & A_{12}^{(i)} \\ 0 & A_{22}^{(i)} \end{bmatrix}, \quad A_{11}^{(i)} \in \mathbb{R}^{k\times k}, \quad A_{11}^{(i)} \in \mathbb{R}^{(n-k)\times(n-k)}, \tag{1.35}$$

if and only if the first k columns of $\hat{Q}_i$ span an invariant subspace of A. Let us assume that this invariant subspace is simple. Then the perturbation analysis developed in the previous section shows that A_i is close to block Schur form (1.35) (i.e., its $(2,1)$ block is of small norm) if and only if the space spanned by the first k columns of $\hat{Q}_i$ is close to an invariant subspace. Hence, we can check the convergence of the QR iteration to block Schur form by investigating the behavior of the subspace sequence defined by

$$\mathcal{S}_i := \operatorname{span}\{\hat{Q}_i e_1, \hat{Q}_i e_2, \ldots, \hat{Q}_i e_k\}.$$

If we define $\mathcal{S}_0 := \operatorname{span}\{e_1, e_2, \ldots, e_k\}$ then

$$\mathcal{S}_i = p_i(A)p_{i-1}(A)\cdots p_1(A)\mathcal{S}_0. \tag{1.36}$$

This relation can be rewritten in the more compact form $\mathcal{S}_i = \hat{p}_i(A)\mathcal{S}_0$, where $\hat{p}_i$ denotes the polynomial product $p_i \cdot p_{i-1} \cdots p_1$.

Theorem 1.16. *Let $A \in \mathbb{R}^{n\times n}$ have a block Schur decomposition*

$$U^H A U = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad A_{11} \in \mathbb{R}^{k\times k}, \quad A_{22} \in \mathbb{R}^{(n-k)\times(n-k)}.$$

Assume that $\lambda(A_{11}) \cap \lambda(A_{22}) = \emptyset$, and let $\mathcal{X}_1$ and $\mathcal{X}_2$ denote the simple invariant subspaces belonging to $\lambda(A_{11})$ and $\lambda(A_{22})$, respectively. Given a sequence of polynomials $p_1, p_2, \ldots$, assume $\hat{p}_i(A_{11})$ to be nonsingular for all $\hat{p}_i = p_i p_{i-1} \cdots p_1$. Let $\mathcal{S}_0$ be any k-dimensional subspace satisfying $\mathcal{S}_0 \cap \mathcal{X}_2 = \emptyset$. Then the gap, see (1.25), between the subspaces $\mathcal{S}_i = \hat{p}_i(A)\mathcal{S}_0$ and the invariant subspace $\mathcal{X}_1$ can be bounded by

$$d_2(\mathcal{S}_i, \mathcal{X}_1) \le C \, \|\hat{p}_i(A_{11})^{-1}\|_2 \, \|\hat{p}_i(A_{22})\|_2, \tag{1.37}$$

where

$$C = \left(\|P\|_2 + \sqrt{\|P\|_2^2 - 1}\right) \frac{d_2(\mathcal{S}_0, \mathcal{X}_1)}{\sqrt{1 - d_2(\mathcal{S}_0, \mathcal{X}_1)^2}}$$

and P is the spectral projector belonging to $\lambda(A_{11})$.

Proof. This result is essentially Theorem 5.1 in [359], but our assumptions are slightly weaker and the presented constant C is potentially smaller.

Let R denote the solution of the Sylvester equation $A_{11}R - RA_{22} = A_{12}$, let $W_R = \begin{bmatrix} I & -R/\|P\|_2 \\ 0 & I/\|P\|_2 \end{bmatrix}$, and apply the similarity transformation UW_R to $\hat{p}_i(A)$:

$$(UW_R)^{-1}\hat{p}_i(A)(UW_R) = \begin{bmatrix} \hat{p}_i(A_{11}) & 0 \\ 0 & \hat{p}_i(A_{22}) \end{bmatrix}.$$

Using two basic results by Watkins and Elsner (Lemma 4.1 and Lemma 4.4 in [359]), it follows that

$$d_2(\mathcal{S}_i, \mathcal{X}_1) \leq \kappa_2(W_R) \frac{d_2(\mathcal{S}_0, \mathcal{X}_1)\|\hat{p}_i(A_{11})^{-1}\|_2 \, \|\hat{p}_i(A_{22})\|_2}{\sqrt{1 - d_2(\mathcal{S}_0, \mathcal{X}_1)^2}},$$

where $\kappa_2(W_R) = \|W_R\|_2 \, \|W_R^{-1}\|_2$. From $\|P\|_2^2 = 1 + \|R\|_2^2$ it can be directly verified that

$$\kappa_2(W_R) = \|P\|_2 + \sqrt{\|P\|_2^2 - 1},$$

which concludes the proof. □

Let us remark that the subspace condition $\mathcal{S}_0 \cap \mathcal{X}_2 = \emptyset$ in the preceeding theorem is rather weak. Later on, we will assume that A is in upper Hessenberg form and $\mathcal{S}_0 = \operatorname{span}\{e_1, e_2, \ldots, e_k\}$. In this case, the subspace condition is satisfied by any k for which the first k subdiagonal entries of A do not vanish.

Apart from a different choice of the initial subspace $\mathcal{S}_0$, the constant C in the upper bound (1.37) cannot be influenced. Thus, in order to obtain (rapid) convergence predicted by this bound we have to choose fortunate shift polynomials that yield small values for $\|\hat{p}_i(A_{11})^{-1}\|_2 \, \|\hat{p}_i(A_{22})\|_2$. We will distinguish two choices, the stationary case $p_i \equiv p$ for some fixed polynomial p, and the instationary case where the polynomials p_i converge to some polynomial $p^\star$ with all roots in $\lambda(A)$.

Stationary shifts

Choosing stationary shift polynomials includes the important special case $p_i(x) = x$, where the iteration (1.34) amounts to the *unshifted QR iteration*:

$$\begin{aligned} A_{i-1} &= Q_i R_i, \qquad \text{(QR decomposition)} \\ A_i &\leftarrow R_i Q_i. \end{aligned}$$

The following example demonstrates that the convergence of this iteration can be rather slow, especially if the eigenvalues of A are not well separated.

Example 1.17. Consider the 10×10 matrix $A = X\Lambda X^{-1}$, where

$$\Lambda = \operatorname{diag}(4, 2, 0.9, 0.8, 0.7, 0.6, 0.59, 0.58, 0.1, 0.09)$$

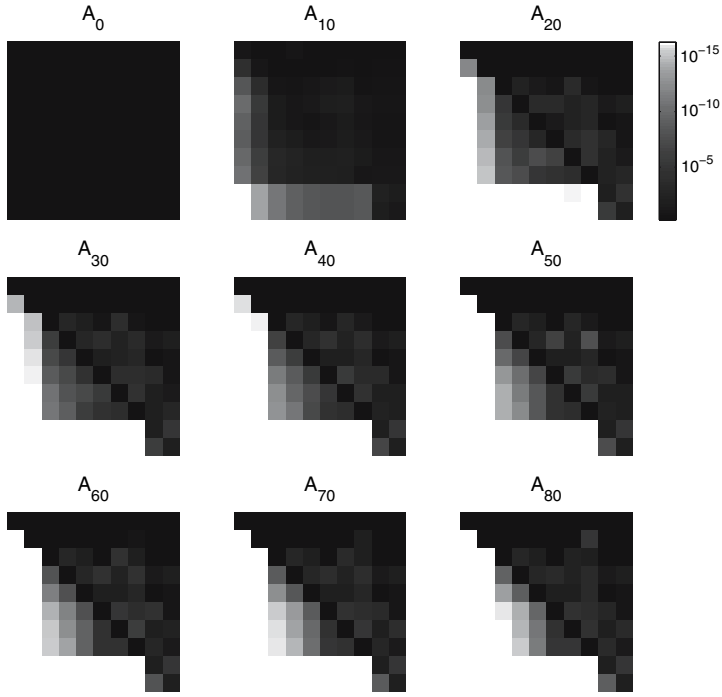


Fig. 1.3. Convergence pattern of the unshifted QR iteration.

and X is a random matrix with condition number $\kappa_2(X) = 10^3$. We applied 80 unshifted QR iterations to $A_0 = A$; the absolute values of the entries in A_i for $i = 0, 10, \ldots, 80$ are displayed in Figure 1.3. First, the eigenvalue cluster $\{0.1, 0.09\}$ converges in the bottom right corner, followed by the individual eigenvalues 4 and 2 in the top left corner. The other eigenvalues converge much slower. Only after $i = 1909$ iterations is the norm of the strictly lower triangular part of A_i less than $\mathbf{u}\,\|A\|_2$. The diagonal entries of A_i approximate the diagonal entries of Λ in descending order. ◊

The observed phenomenon, that the convergence is driven by the separation of eigenvalues, is well explained by the following corollary of Theorem 1.16.

Corollary 1.18 ([359]). *Let $A \in \mathbb{R}^{n\times n}$ and let p be a polynomial. Assume that there exists a partitioning $\lambda(A) = \Lambda_1 \cup \Lambda_2$ such that*

$$\gamma := \frac{\max\{|p(\lambda_2)| : \lambda_2 \in \Lambda_2\}}{\min\{|p(\lambda_1)| : \lambda_1 \in \Lambda_1\}} < 1. \tag{1.38}$$

Let $\mathcal{X}_1$ and $\mathcal{X}_2$ denote the simple invariant subspace belonging to Λ_1 and Λ_2, respectively. Let $\mathcal{S}_0$ be any k-dimensional subspace satisfying $\mathcal{S}_0 \cap \mathcal{X}_2 = \emptyset$.

Then for any $\hat{\gamma} > \gamma$ there exists a constant $\hat{C}$ not depending on $\mathcal{S}_0$ so that the gap between the subspaces $\mathcal{S}_i = p(A)^i \mathcal{S}_0$, $i = 1, 2, \ldots$, and the invariant subspace $\mathcal{X}_1$ can be bounded by

$$d_2(\mathcal{S}_i, \mathcal{X}_1) \leq C\hat{\gamma}^i,$$

where

$$C = \hat{C} \frac{d_2(\mathcal{S}_0, \mathcal{X}_1)}{\sqrt{1 - d_2(\mathcal{S}_0, \mathcal{X}_1)^2}}.$$

Proof. Since the assumptions of Theorem 1.16 are satisfied, there exists a constant $\tilde{C}$ so that

$$d_2(\mathcal{S}_i, \mathcal{X}_1) \leq \tilde{C} \|p(A_{11})^{-i}\|_2 \, \|p(A_{22})^i\|_2, \leq \tilde{C} (\|p(A_{11})^{-1}\|_2 \, \|p(A_{22})\|_2)^i,$$

where $\lambda(A_{11}) = \Lambda_1$ and $\lambda(A_{22}) = \Lambda_2$. If ρ denotes the spectral radius of a matrix then $\gamma = \rho(p(A_{11})^{-1})\rho(p(A_{22}))$ and Lemma A.4 yields for any $\hat{\gamma} > \gamma$ the existence of induced matrix norms $\|\cdot\|_\alpha$ and $\|\cdot\|_\beta$ so that $\hat{\gamma} = \|p(A_{11})^{-1}\|_\alpha \, \|p(A_{22})\|_\beta$. The equivalence of norms on finite-dimensional spaces concludes the proof. □

This corollary predicts only linear convergence for constant shift polynomials, which in fact can be observed in Example 1.17. To achieve quadratic convergence it is necessary to vary p_i in each iteration, based on information contained in A_{i-1}.

Instationary shifts

If the shifts, i.e., the roots of the shift polynomial in each QR iteration, are simple eigenvalues of A, then – under the assumptions of Theorem 1.16 – one iteration of the QR iteration (1.34) yields

$$A_1 = \begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} \\ 0 & A_{22}^{(1)} \end{bmatrix},$$

where the order of $A_{22}^{(1)}$ equals the degree of the shift polynomial p_1. Moreover, the eigenvalues of $A_{22}^{(1)}$ coincide with the roots of p_1 and consequently $p_1(A_{22}^{(1)}) = 0$. This suggests defining the shift polynomial p_i as the characteristic polynomial of $A_{22}^{(i-1)}$, the bottom right $m \times m$ block of A_{i-1} for some fixed integer $m < n$. The roots of such a polynomial p_i are called *Francis shifts*[1]. With this choice, the shifted QR iteration reads as follows:

$$p_i(\lambda) \leftarrow \det(\lambda I_m - A_{22}^{(i-1)}), \tag{1.39a}$$

$$p_i(A_{i-1}) = Q_i R_i, \quad \text{(QR decomposition)} \tag{1.39b}$$

$$A_i \leftarrow Q_i^T A_{i-1} Q_i. \tag{1.39c}$$

[1] It is not known to us who coined the term "Francis shifts". Uses of this term can be found in [103, 305]. Some authors prefer the terms "Wilkinson shifts" or "generalized Rayleigh quotient shifts".

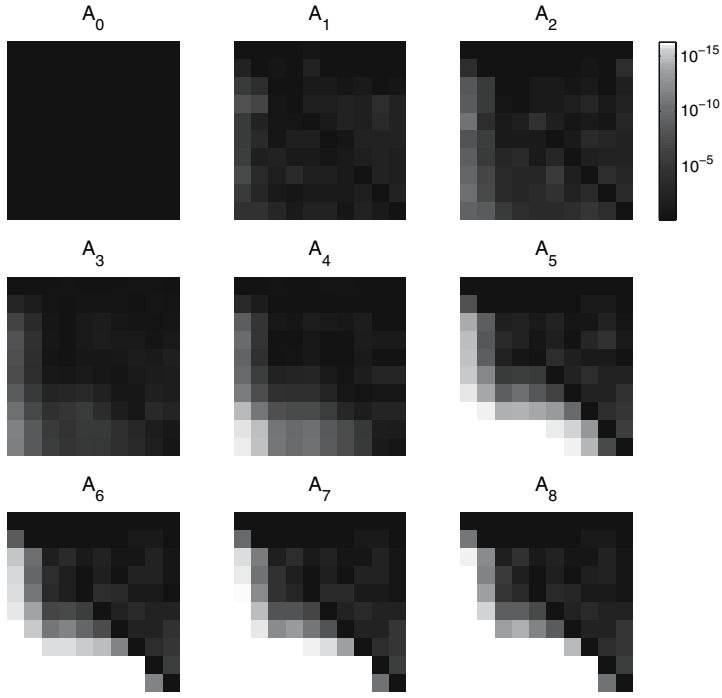


Fig. 1.4. Convergence pattern of the shifted QR iteration with two Francis shifts.

Example 1.19. Let A be the 10×10 matrix defined in Example 1.17. We applied 8 shifted QR iterations of the form (1.39) to $A_0 = A$ with $m = 2$; the absolute values of the entries in A_i for $i = 0, 1, \ldots, 8$ are displayed in Figure 1.4. It can be observed that the 2×2 bottom right block, which contains approximations to the eigenvalue cluster $\{0.59, 0.6\}$, converges rather quickly. Already after six iterations all entries to the left of this block are of absolute value less than $\mathbf{u}\|A\|_2$. Also, the rest of the matrix has made a lot of progress towards convergence. Most notably the eighth diagonal entry of A_8 matches the leading 10 decimal digits of the eigenvalue 0.58. ◊

The rapid convergence of the bottom right 2×2 block exhibited in the preceeding example can be explained by Corollary 1.20 below. Once the shifts

have settled down they are almost stationary shifts to the rest of the matrix explaining the (slower) convergence in this part.

Corollary 1.20 ([359]). *Let $A \in \mathbb{R}^{n\times n}$ and let $\hat{p}_i = p_1 p_2 \cdots p_i$, where the Francis shift polynomials p_i are defined by the sequence (1.39). Assume that the corresponding subspace sequence $\mathcal{S}_i = \hat{p}_i(A)\mathcal{S}_0$ with $\mathcal{S}_0 = \operatorname{span}\{e_1, \ldots, e_{n-m}\}$ converges to some invariant subspace $\mathcal{X}_1$ of A and that all eigenvalues* not *belonging to $\mathcal{X}_1$ are simple. Then this convergence is quadratic.*

Proof. The idea behind the proof is to show that for sufficiently small $\varepsilon = d_2(\mathcal{S}_{i-1}, \mathcal{X}_1)$ the distance of the next iterate, $d_2(\mathcal{S}_i, \mathcal{X}_1)$, is proportional to ε^2. For this purpose, let Λ_1 consist of the eigenvalues belonging to $\mathcal{X}_1$, and let $\mathcal{X}_2$ be the invariant subspace belonging to the remaining eigenvalues $\Lambda_2 = \lambda(A)\backslash\Lambda_1$. For sufficiently small ε we may assume $\mathcal{S}_{i-1} \cap \mathcal{X}_2 = \{0\}$, as $\mathcal{X}_1$ and $\mathcal{X}_2$ are distinct subspaces. The $(i-1)$th iterate of (1.39) takes the form

$$A_{i-1} = \hat{Q}_{i-1}^T A \hat{Q}_{i-1} = \begin{bmatrix} A_{11}^{(i-1)} & A_{12}^{(i-1)} \\ A_{21}^{(i-1)} & A_{22}^{(i-1)} \end{bmatrix}.$$

From $d_2(\mathcal{S}_{i-1}, \mathcal{X}_1) = \varepsilon$, it follows that $\|A_{21}^{(i-1)}\|_2 \leq \sqrt{2}\varepsilon\|A\|_2$ [359, Lemma 6.1]. If c_2 denotes the maximal absolute condition number for any eigenvalue in Λ_2 then for sufficiently small ε we obtain

$$\max\{|p_i(\lambda_2)| : \lambda_2 \in \Lambda_2\} \leq M\varepsilon$$

with $M = c_2(2\|A\|_2)^m$. Since

$$\delta = \min\{|\lambda_2 - \lambda_1| : \lambda_1 \in \Lambda_1, \lambda_2 \in \Lambda_2\} > 0,$$

we know that all roots of p_i have a distance of at least $\delta/2$ to the eigenvalues in Λ_1, provided that ε is chosen sufficiently small. Hence, the quantity γ defined in (1.38) satisfies $\gamma \leq (2/\delta)^m M\varepsilon$. For $\varepsilon < (\delta/2)^m/M$ we can now apply Corollary 1.18 to the ith iteration of (1.39) and obtain some constant $\hat{C}$ so that

$$d_2(\mathcal{S}_i, \mathcal{X}_1) < \sqrt{2}\hat{C}M(2/\delta)^m \frac{\varepsilon^2}{\sqrt{1-\varepsilon^2}} \leq 2\hat{C}M(2/\delta)^m\varepsilon^2,$$

where the latter inequality holds for $\varepsilon \leq 1/\sqrt{2}$. □

1.3.2 Hessenberg Form

A literal implementation of the shifted QR iteration (1.39) is prohibitely expensive; the explicit computation of $p_i(A_{i-1})$ alone requires $\mathcal{O}(mn^3)$ flops. The purpose of this section is to reduce the cost of an overall iteration down to $\mathcal{O}(mn^2)$ flops. First, we recall the well-known result that shifted QR iterations preserve matrices in unreduced Hessenberg form.

Definition 1.21. *A square matrix A is said to be in* upper Hessenberg form *if all its entries below the first subdiagonal are zero. Moreover, such a matrix is called* unreduced[2] *if all its subdiagonal entries are nonzero.*

If one of the subdiagonal entries of the Hessenberg matrix A happens to be zero, one can partition

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix},$$

and consider the Hessenberg matrices A_{11} and A_{22} separately. This process is called deflation and will be discussed in more detail in Section 1.3.4.

Lemma 1.22. *Let $A \in \mathbb{R}^{n\times n}$ be in unreduced Hessenberg form. Let $f : \mathbb{C} \to \mathbb{C}$ be any function analytic on an open neighborhood of $\lambda(A)$ with no zeros in $\lambda(A)$. If $f(A) = QR$ is a QR decomposition then $Q^H AQ$ is again in unreduced Hessenberg form.*

Proof. The proof is based on the fact that A is in unreduced Hessenberg form if and only if the *Krylov matrix*

$$K_n(A, e_1) = \begin{bmatrix} e_1 & Ae_1 & \cdots & A^{n-1}e_1 \end{bmatrix}$$

is an upper triangular, nonsingular matrix. Using the facts that A and $f(A)$ commute and R is invertible we obtain

$$\begin{aligned} K_n(Q^H AQ, e_1) &= Q^H K_n(A, Qe_1) = Q^H K_n(A, f(A)R^{-1}e_1) \\ &= Q^H f(A) K_n(A, R^{-1}e_1) = RK_n(A, R^{-1}e_1) \\ &= \frac{1}{r_{11}} RK_n(A, e_1), \end{aligned}$$

showing that $K_n(Q^H AQ, e_1)$ is upper triangular and nonsingular. □

Reduction to Hessenberg form

If the initial matrix $A_0 = A$ is in upper Hessenberg form then, subtracting possible deflations, shifted QR iterations preserve this form. It remains to show how the initial matrix can be reduced to Hessenberg form. This is usually achieved by applying orthogonal similarity transformations based on Householder matrices to the matrix A.

A *Householder matrix* is a symmetric matrix of the form

$$H(v, \beta) := I - \beta vv^T, \tag{1.40}$$

where $v \in \mathbb{R}^n$ and $\beta \in \mathbb{R}$. It is assumed that either $v = 0$ or $\beta = 2/v^T v$, which ensures that $H(v, \beta)$ is an orthogonal matrix. For a given vector $x \in \mathbb{R}^n$ and

[2] Some authors use the term *proper* instead of unreduced. Strictly speaking, the occasionally used term *irreducible* is misleading as a matrix in unreduced Hessenberg form may be reducible, see also Section 3.3.1.

an integer $j \leq n$ we can always construct a Householder matrix which maps the last $n-j$ elements of x to zero by choosing

$$v = \begin{bmatrix} 0 & 0 \\ 0 & I_{n-j+1} \end{bmatrix} x + \operatorname{sign}(e_j^T x) \left\| \begin{bmatrix} 0 & 0 \\ 0 & I_{n-j+1} \end{bmatrix} x \right\|_2 e_j \tag{1.41}$$

and

$$\beta = \begin{cases} 0 & \text{if } v = 0, \\ 2/v^T v & \text{otherwise.} \end{cases} \tag{1.42}$$

Under this choice of v and β, we identify $H_j(x) \equiv H(v, \beta)$. Note that the multiplication of $H_j(x)$ with a vector y has no effect on the first $j-1$ elements of y.

Let us illustrate the use of Householder matrices for reducing a 5×5 matrix A to Hessenberg form. First, if we apply $H_2(Ae_1)$ from the left to the columns of A then the trailing three entries in the first column of A get annihilated. The first column remains unaffected if the same transformation is applied from the right. This corresponds to the following diagram:

$$A \leftarrow H_2(Ae_1) \cdot A \cdot H_2(Ae_1) = \begin{bmatrix} a & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \end{bmatrix}.$$

Similar diagrams have been used by Wilkinson in his book [364] and by many other authors later on. In such a diagram, a letter denotes a generic non-zero entry and 0 denotes a zero entry. A hat is put on a letter to designate that this entry is being modified during the current transformation. Consequently, an entry denoted by $\hat{0}$ is being annihilated during the current transformation.

Continuing the reduction to Hessenberg form, we can annihilate the trailing two entries of the second column in an analogous manner:

$$A \leftarrow H_3(Ae_2) \cdot A \cdot H_3(Ae_2) = \begin{bmatrix} a & a & \hat{a} & \hat{a} & \hat{a} \\ a & a & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{a} \end{bmatrix}.$$

Finally, the remaining nonzero $(5,3)$ element is addressed:

$$A \leftarrow H_4(Ae_3) \cdot A \cdot H_4(Ae_3) = \begin{bmatrix} a & a & a & \hat{a} & \hat{a} \\ a & a & a & \hat{a} & \hat{a} \\ 0 & a & a & \hat{a} & \hat{a} \\ 0 & 0 & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & \hat{0} & \hat{a} & \hat{a} \end{bmatrix}.$$

For general n, the corresponding procedure is given by Algorithm 1.

Algorithm 1 Reduction to Hessenberg form

Input: A matrix $A \in \mathbb{R}^{n\times n}$.
Output: An orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ such that $H = Q^T AQ$ is in upper Hessenberg form. The matrix A is overwritten by H.

```
Q ← I_n
for j ← 1, ..., n − 2 do
  Q ← Q · H_{j+1}(Ae_j)
  A ← H_{j+1}(Ae_j) · A · H_{j+1}(Ae_j)
end for
```

Several remarks regarding the actual implementation of Algorithm 1 are in order:

1. The Householder matrix $H_{j+1}(Ae_j) \equiv H(v_j, \beta_j)$ is represented by $v_j \in \mathbb{R}^n$ and $\beta_j \in \mathbb{R}$. Both quantities are computed by the LAPACK [10] routine `DLARFG`, which is based on formulas (1.41) and (1.42).
2. The update $A \leftarrow H_{j+1}(Ae_j) \cdot A \cdot H_{j+1}(Ae_j)$ is performed via two rank-one updates by calling LAPACK's `DLARF`. Only those parts of A that will be modified by the update need to be involved. This is the submatrix $A(j+1:n, j:n)$ for the left transformation, and $A(1:n, j+1:n)$ for the right transformation. Here, the *colon notation* $A(i_1:i_2, j_1:j_2)$ is used to designate the submatrix of A defined by rows i_1 through i_2 and columns j_1 through j_2.
3. The leading j entries of each vector v_j are zero and v_j can be scaled so that its $(j+1)$th entry becomes one. Hence, there are only $n-j-1$ nontrivial entries in v_j, which exactly fit in the annihilated part of the jth column of A. The $n-2$ scalars β_j need to be stored in an extra workspace array.

The LAPACK routine `DGEHD2` is such an implementation of Algorithm 1 and requires $\frac{10}{3}n^3 + \mathcal{O}(n^2)$ flops. It does not compute the orthogonal factor Q, there is a separate routine called `DORGHR`, which accumulates Q in reversed order and thus only needs to work on $Q(j+1:n, j+1:n)$ instead of $Q(1:n, j+1:n)$ in each loop. If the unblocked version of `DORGHR` is used (see Section 1.5.1 for a description of the blocked version) then the accumulation of Q requires an additional amount of $\frac{4}{3}n^3$ flops.

1.3.3 Implicit Shifted QR Iteration

If the number of shifts in the shifted QR iteration (1.39) is limited to one then each iteration requires the QR decomposition of an upper Hessenberg matrix. This can be implemented in $\mathcal{O}(n^2)$ flops, see, e.g., [141, Sec. 7.4.2]. A similar algorithm could be constructed to compute the QR decomposition of $p_i(A_{i-1})$ for shift polynomials p_i of higher degree. However, this algorithm would require an extra $n \times n$ workspace array, is difficult to implement in real

arithmetic for complex conjugate shifts, and, even worse, does not guarantee the preservation of Hessenberg forms in finite-precision arithmetic.

The implicit shifted QR iteration, also introduced by Francis [128], avoids these difficulties by making use of the following well-known "uniqueness property" of the Hessenberg reduction.

Theorem 1.23 (Implicit Q theorem). *Let $U = [u_1, \ldots, u_n]$ and $V = [v_1, \ldots, v_n]$ be orthogonal matrices so that both matrices $U^T AU = G$ and $V^T AV = H$ are in upper Hessenberg form and G is unreduced. If $u_1 = v_1$ then there exists a diagonal matrix $D = \mathrm{diag}(1, \pm 1, \ldots, \pm 1)$ so that $V = UD$ and $H = DGD$.*

Now, assume that the last iterate of the shifted QR iteration A_{i-1} is in unreduced upper Hessenberg form and that no root of the shift polynomial p_i is an eigenvalue of A_{i-1}. Let x be the first column of $p_i(A_{i-1})$. Furthermore, assume that Q_i is an orthogonal matrix so that $Q_i^T A_{i-1} Q_i$ is in upper Hessenberg form and that the first column of Q_i is a multiple of x. Then, Lemma 1.22 and Theorem 1.23 imply that $Q_i^T p_i(A_{i-1})$ is upper triangular, and thus $A_i \leftarrow Q_i^T A_{i-1} Q_i$ yields a suitable next iterate for the shifted QR iteration.

Algorithm 2 constructs such a matrix Q_i for the shifted QR iteration (1.39) employing Francis shifts. It relies on the facts that the first column of the Householder matrix $H_1(x)$ is a multiple of x and that the orthogonal matrix Q returned by Algorithm 1 has the form $Q = 1 \oplus \tilde{Q}$. Here, '$\oplus$' denotes the direct sum (or block diagonal concatenation) of two matrices.

Algorithm 2 Implicit shifted QR iteration

Input: A matrix $A_{i-1} \in \mathbb{R}^{n \times n}$ with $n \geq 2$ in unreduced upper Hessenberg form, an integer $m \in [2, n]$.

Output: An orthogonal matrix $Q_i \in \mathbb{R}^{n \times n}$ so that $Q_i^T p_i(A_{i-1})$ is upper triangular, where p_i is the Francis shift polynomial of degree m. The matrix A_{i-1} is overwritten by $A_i = Q_i^T A_{i-1} Q_i$.

1. Compute shifts $\sigma_1, \ldots, \sigma_m$ as eigenvalues of $A_{i-1}(n-m+1:n, n-m+1:n)$.
2. Set $x = (A_{i-1} - \sigma_1 I_n)(A_{i-1} - \sigma_2 I_n) \cdots (A_{i-1} - \sigma_m I_n) e_1$.
3. Update $A_{i-1} \leftarrow H_1(x) \cdot A_{i-1} \cdot H_1(x)$.
4. Apply Algorithm 1 to compute an orthogonal matrix Q so that A_{i-1} is reduced to Hessenberg form.
5. Set $Q_i = H_1(x) \cdot Q$.

The shifts $\sigma_1, \ldots, \sigma_m$ in Algorithm 2 can be computed by an auxiliary implementation of the QR algorithm which employs at most two Francis shifts, see for example the LAPACK routine `DLAHQR`. The computation of two Francis shifts, in turn, can be achieved by basic arithmetic operations, although the actual implementation requires some care, see also Remark 1.26 below. The

vector x is always real; it can be computed in real arithmetic by grouping complex conjugate pairs of shifts and using $A_{i-1}^2 - 2\,\mathrm{Re}(\sigma_j)A_{i-1} + |\sigma_j|^2 I_n$ instead of $(A_{i-1} - \sigma_j I_n)(A_{i-1} - \bar{\sigma}_j I_n)$ for such pairs. Making full advantage of the zero structure of x lets its computation require $\mathcal{O}(m^3)$ flops. Alternatively, Dubrulle and Golub [115] have developed an algorithm which computes a multiple of x without explicitly determining the shifts. In Section 1.5.4, the computation of x is connected to the so called pole placement problem.

Step 4 of Algorithm 2 should be based on a special-purpose implementation of Algorithm 1, which exploits the zero structures of the matrix A_{i-1} and the involved Householder matrices $H_{j+1}(A_{i-1}e_j)$, see, e.g., [141, Alg. 7.5.1]. In this case, Step 4 requires $(4m + 3)n^2 + \mathcal{O}(mn)$ flops for reducing A_{i-1} and additionally the same amount of flops for post-multiplying the involved Householder matrices to a given $n \times n$ matrix.

Let us illustrate Algorithm 2 for $n = 6$ and $m = 2$. First, the upper Hessenberg matrix A_{i-1} is overwritten by $H_1(x) \cdot A_{i-1} \cdot H_1(x)$, where $x = (A_{i-1} - \sigma_1 I)(A_{i-1} - \sigma_2 I)e_1$. Only the leading three elements of x are nonzero, this implies that only the first three columns and rows of A_{i-1} are affected:

$$A_{i-1} \leftarrow H_1(x) \cdot A_{i-1} \cdot H_1(x) = \begin{bmatrix} \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{b} & \hat{b} & \hat{b} & \hat{a} & \hat{a} & \hat{a} \\ \hat{b} & \hat{b} & \hat{b} & \hat{a} & \hat{a} & \hat{a} \\ \hat{b} & \hat{b} & \hat{b} & a & a & a \\ 0 & 0 & 0 & a & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix}. \tag{1.43}$$

The 3×3 submatrix $A_{i-1}(2:4, 1:3)$, whose elements are labeled by $\hat{b}$, is called the *bulge*. The subsequent reduction to Hessenberg form can be seen as chasing this bulge down to the bottom right corner along the first subdiagonal of A_{i-1}. This point of view has been emphasized and extended to other QR-like algorithms by Watkins and Elsner [358]. In the first step of Algorithm 1 applied to A_{i-1}, the nonzero elements introduced in the first column of A_{i-1} are annihilated by the Householder matrix $H_2(A_{i-1}e_1)$:

$$A_{i-1} \leftarrow H_2(A_{i-1}e_1) \cdot A_{i-1} \cdot H_2(A_{i-1}e_1) = \begin{bmatrix} a & \hat{a} & \hat{a} & \hat{a} & a & a \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{b} & \hat{b} & \hat{b} & \hat{a} & \hat{a} \\ \hat{0} & \hat{b} & \hat{b} & \hat{b} & \hat{a} & \hat{a} \\ 0 & \hat{b} & \hat{b} & \hat{b} & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix}.$$

Note that the bulge has moved one step downwards. The subsequent application of $H_3(A_{i-1}e_2)$, $H_4(A_{i-1}e_3)$ and $H_5(A_{i-1}e_4)$ pushes the bulge further down to the bottom right corner and, finally, off the corner:

$$A_{i-1} \leftarrow H_3(A_{i-1}e_2) \cdot A_{i-1} \cdot H_3(A_{i-1}e_2) = \begin{bmatrix} a & a & \hat{a} & \hat{a} & \hat{a} & a \\ a & a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{b} & \hat{b} & \hat{b} & \hat{a} \\ 0 & \hat{0} & \hat{b} & \hat{b} & \hat{b} & \hat{a} \\ 0 & 0 & \hat{b} & \hat{b} & \hat{b} & a \end{bmatrix},$$

$$A_{i-1} \leftarrow H_4(A_{i-1}e_3) \cdot A_{i-1} \cdot H_4(A_{i-1}e_3) = \begin{bmatrix} a & a & a & \hat{a} & \hat{a} & \hat{a} \\ a & a & a & \hat{a} & \hat{a} & \hat{a} \\ 0 & a & a & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & \hat{0} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & \hat{0} & \hat{b} & \hat{b} & \hat{b} \end{bmatrix}, \tag{1.44}$$

$$A_{i-1} \leftarrow H_5(A_{i-1}e_4) \cdot A_{i-1} \cdot H_5(A_{i-1}e_4) = \begin{bmatrix} a & a & a & a & \hat{a} & \hat{a} \\ a & a & a & a & \hat{a} & \hat{a} \\ 0 & a & a & a & \hat{a} & \hat{a} \\ 0 & 0 & a & a & \hat{a} & \hat{a} \\ 0 & 0 & 0 & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & 0 & \hat{0} & \hat{a} & \hat{a} \end{bmatrix}.$$

1.3.4 Deflation

Setting a "small" subdiagonal element $a_{k+1,k}$ of a matrix A in upper Hessenberg form to zero makes it a block upper triangular matrix:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad A_{11} \in \mathbb{R}^{k \times k}, \quad A_{22} \in \mathbb{R}^{(n-k)\times(n-k)}.$$

This *deflates* the eigenvalue problem into the two smaller eigenvalue problems associated with the diagonal blocks A_{11} and A_{22}. A subdiagonal entry is considered to be small if it satisfies

$$|a_{k+1,k}| \leq \mathbf{u}\, \|A\|_F. \tag{1.45}$$

This is justified by the fact that the reduction to Hessenberg form as well as QR iterations introduce roundoff errors of order $\mathbf{u}\, \|A\|_F$ anyway.

A less generous deflation criterion compares the magnitude of $|a_{k+1,k}|$ with the magnitudes of its diagonal neighbors:

$$|a_{k+1,k}| \leq \mathbf{u}\, (|a_{k,k}| + |a_{k+1,k+1}|). \tag{1.46}$$

This criterion is used in the LAPACK routines `DHSEQR` and `DLAHQR`. If the right hand side in inequality (1.45) happens to be zero, then $a_{k,k}$ or $a_{k+1,k+1}$ can be replaced by a nonzero element in the same column or row, respectively.

At first sight there is little justification for using the neighbor-wise deflation criterion (1.46) in favor of the norm-wise criterion (1.45). There is no theory saying that the QR algorithm generally produces more exact eigenvalues if (1.46) is used although this effect can occasionally be observed. For graded matrices, Stewart [306] provides some insight why the QR algorithm equipped with the neighbor-wise deflation criterion (1.46) often computes small eigenvalues with high relative accuracy. An instance of such a graded matrix is given in the following example.

Example 1.24. Let

$$A = \begin{bmatrix} 1 & 10^{-03} & 0 & 0 \\ 10^{-03} & 10^{-07} & 10^{-10} & 0 \\ 0 & 10^{-10} & 10^{-14} & 10^{-17} \\ 0 & 0 & 10^{-17} & 10^{-21} \end{bmatrix}.$$

Tabulated below are the exact eigenvalues of A, the computed eigenvalues using deflation criterion (1.45) and, in the last column, the computed eigenvalues using deflation criterion (1.46). In both cases, the implicit shifted QR algorithm with two Francis shifts was used.

Exact eigenvalues	Norm-wise deflation	Neighbor-wise deflation
1.0000009999991000	1.0000009999991001	1.0000009999990999
$-.8999991111128208 \times 10^{-06}$	$-.8999991111128212 \times 10^{-06}$	$-.8999991111128213 \times 10^{-06}$
$.2111111558732113 \times 10^{-13}$	$.2111111085047986 \times 10^{-13}$	$.2111111558732114 \times 10^{-13}$
$-.3736841266803067 \times 10^{-20}$	$0.0999999999999999 \times 10^{-20}$	$-.3736841266803068 \times 10^{-20}$

It can be observed that the two eigenvalues of smallest magnitude are much more accurately computed if the neighbor-wise deflation criterion is used. ◊

Although the deflation of zero subdiagonal elements is a must in order to ensure convergence, it has been shown by Watkins [350] that even tiny subdiagonal elements (as small as $10^{-70} \times \|A\|_2$ in double-precision arithmetic) do not affect the convergence of the QR algorithm.

1.3.5 The Overall Algorithm

Glueing implicit QR iterations and deflation together yields Algorithm 3, the QR algorithm for Hessenberg matrices.

Properly implemented, Algorithm 3 requires approximately $(8 + 6/m)n^3$ flops for computing T under the assumptions that $m \ll n$ and that on average four Francis shifts are necessary to let one eigenvalue converge at the bottom right corner of H. If only the diagonal blocks of T (e.g., for computing eigenvalues) are required then the update of the inactive off-diagonal parts $H(i : l, l + 1 : n)$ and $H(1 : i - 1, i : l)$ can be omitted and the number of flops reduces down to $(\frac{16}{3} + 4/m)n^3$. The accumulation of the orthogonal factor Q requires another $(8 + 6/m)n^3$ flops. These flop counts should not be accepted at face value as the actual number of necessary shifts depends on the matrix in question.

Algorithm 3 Basic QR algorithm

Input: A matrix $H \in \mathbb{R}^{n\times n}$ with $n \geq 2$ in upper Hessenberg form, an integer $m \in [2, n]$.

Output: An orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ such that $T = Q^T HQ$ is a block upper triangular matrix with diagonal blocks of size at most two. The matrix H is overwritten by T. Implicit shifted QR iterations with at most m Francis shifts are used.

$Q \leftarrow I_n$
$i \leftarrow 1, \quad l \leftarrow n$
for $it \leftarrow 0, \ldots, 30n$ **do**
  % The active submatrix is $H(i : l, i : l)$. Search for deflations.
  $k \leftarrow i$
  while $(k < l)$ **and** $(|h_{k+1,k}| > \mathbf{u}(|h_{k,k}| + |h_{k+1,k+1}|))$ **do**
    $k \leftarrow k + 1$
  end while
  if $k < l$ **then**
    % Deflation at position $(k + 1, k)$ found.
    $h_{k+1,k} \leftarrow 0$
    $i \leftarrow k + 1$
    if $i + 1 \geq l$ **then**
      % Block of size at most two has converged.
      $l \leftarrow i - 1, \quad i \leftarrow 1$
      if $l \leq 2$ **then**
        % QR algorithm has converged.
        Exit.
      end if
    end if
  else
    Apply implicit shifted QR iteration, Algorithm 2, with $\min\{m, l - i + 1\}$ shifts to $H(i : l, i : l)$. Let $\tilde{Q}$ denote the corresponding orthogonal matrix.
    Update $H(i : l, l + 1 : n) \leftarrow \tilde{Q}^T H(i : l, l + 1 : n)$.
    Update $H(1 : i - 1, i : l) \leftarrow H(1 : i - 1, i : l)\tilde{Q}$.
    Update $Q(1 : n, i : l) \leftarrow Q(1 : n, i : l)\tilde{Q}$.
  end if
end for
% The QR algorithm did not converge within $30n$ iterations.
Exit and error return.

Example 1.25. We applied Algorithm 3 to randomly generated $n \times n$ Hessenberg matrices with $m \in \{2, 4\}$ and $n \in [10, 50]$. Figure 1.5 displays the ratios between the actually required flops and the estimated $(8 + 6/m)n^3$ flops for computing T. Black curves correspond to matrices generated with the MATLAB [236] command `hess(rand(n))` and gray curves to matrices generated with `triu(rand(n),-1)`. It can be seen that the QR algorithm requires significantly more flops for the second class of matrices. Moreover, the underestimation of the actually needed flops increases from $m = 2$ (factor $1.1 \ldots 1.5$)

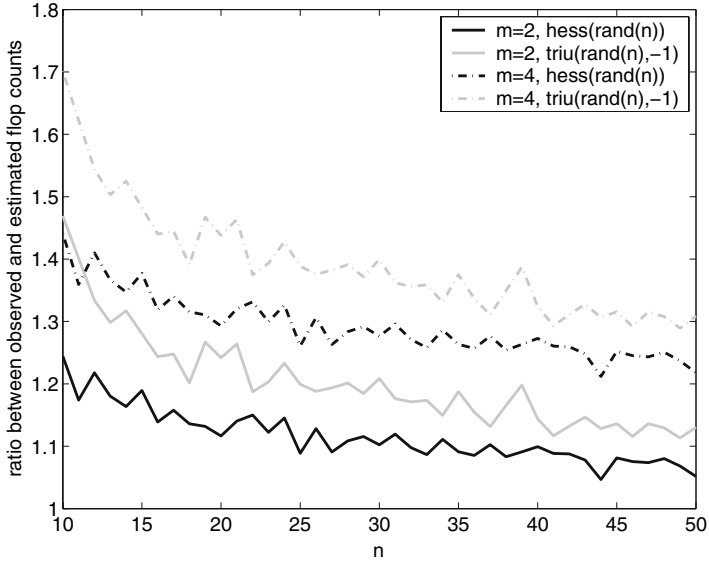


Fig. 1.5. Ratio between observed and estimated flops for reducing a Hessenberg matrix to a block upper triangular matrix using the QR algorithm (Algorithm 3).

to $m = 4$ (factor $1.2 \ldots 1.7$). The latter effect will be explained in more detail in Sections 1.5.3 and 1.5.4. ◊

Remark 1.26. In order to obtain a real Schur form, the 2×2 diagonal blocks of the matrix T returned by Algorithm 3 must be post-processed. Blocks with real eigenvalues need to be transformed to upper triangular form. Blocks with pairs of complex conjugate eigenvalues need to be standardized so that they take the form

$$\begin{bmatrix} a & b \\ -c & a \end{bmatrix}, \quad b, c > 0.$$

The eigenvalues of this matrix are simply given by $a \pm \mathrm{i}\sqrt{bc}$. Both transformations can be achieved by Givens rotations using only basic arithmetic operations. The actual implementation, however, requires some care in order to maintain the backward stability of the QR algorithm. More details can be found in recent work by Sima [286]. See also the LAPACK subroutine `DLANV2`. ◊

Algorithm 3 is implemented in LAPACK as the subroutine `DHSEQR`, which uses only a small number of Francis shifts, typically $m = 6$. The auxiliary subroutine `DLAHQR` employs two shifts; it is called within `DHSEQR` for computing shifts or handling small deflated submatrices.

The QR algorithm is based on orthogonal transformations. This implies that this algorithm, presuming that it converges, is numerically backward

stable, i.e., the computed Schur form is an exact Schur form of a perturbed matrix $H + E$, where $\|E\|_2 \leq \mathcal{O}(\mathbf{u})\|H\|_2$, see [364]. Similar remarks apply to Algorithm 1, the reduction to Hessenberg form, showing that the combination of both algorithms is a backward stable method for computing the eigenvalues of a general real matrix.

1.3.6 Failure of Global Converge

We have seen that the convergence of the QR algorithm is locally quadratic under rather mild assumptions. The global convergence properties of the QR algorithm are much less understood. Because of its relation to the Rayleigh quotient iteration, the QR algorithm with one Francis shift applied to a symmetric matrix converges for almost every matrix [24, 261]. This statement, however, does not hold for nonsymmetric matrices [25]. Nevertheless, there are *very few* examples of practical relevance known, where the QR algorithm does not converge within $30n$ iterations.

A classical example for a matrix, where the QR algorithm fails to converge, is the $n \times n$ cyclic matrix

$$C_n = \begin{bmatrix} 0 & & & 1 \\ 1 & \ddots & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix}. \tag{1.47}$$

This matrix stays invariant under shifted QR iterations that use $m < n$ Francis shifts [233, 364]. The situation can be resolved by replacing the Francis shift polynomial, after 10 and 20 shifted QR iterations have taken place without effecting any deflation, by a so called *exceptional shift polynomial*. Martin, Peters, and Wilkinson [233] proposed the use of the exceptional shift polynomial

$$p_e(x) = x^2 - \frac{3}{2}\beta x + \beta^2, \quad \beta = |h_{l,l-1}| + |h_{l-1,l-2}|,$$

where $H(i : l, i : l)$ is the active submatrix, see Algorithm 3. Note that the EISPACK [291] implementation of the QR algorithm (EISPACK routine `HQR`) uses $p_e(x - h_{ll})$ instead of $p_e(x)$.

Both exceptional shift strategies let the QR algorithm converge for the matrix C_n in (1.47). Still, they can be defeated [23, 100]. One such example is the matrix

$$H(\eta) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & \eta & 0 \\ 0 & -\eta & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad 10^{-6} \geq \eta \geq 2 \times 10^{-14},$$

whose discovery is attributed to Demmel [100, pg. 2]. If the EISPACK subroutine `HQR` is applied to $H(\eta)$ and no restriction is placed on the number of

iterations, then more than 10^4 implicit shifted QR iterations are necessary for the QR algorithm to converge. Day [100] gives a detailed analysis of cases where `HQR` converges extremely slowly, including the matrix $H(\eta)$ above. He also proposed the following strategy, which recovers the quick convergence of the QR algorithm for those cases: For $m = 2$, if the Francis shifts are real, use the shift closest to h_{ll} twice instead of both Francis shifts. Note that a similar shift strategy had already been proposed by Wilkinson [364, pg. 512].

Day's strategy is implemented in the LAPACK subroutine `DLAHQR`. Yet, the following example, which is quoted in the beginning of this chapter, demonstrates that this strategy can also be defeated

Example 1.27. Consider the matrix

$$A = \begin{bmatrix} 0 & 90 & 0 & 300 \\ -4 \times 10^9 & 0 & -300 & 0 \\ 0 & -300 & 0 & 4 \times 10^9 \\ 0 & 0 & -90 & 0 \end{bmatrix}.$$

The LAPACK subroutine `DLAHQR`, which is indirectly called by the MATLAB command `schur`, does not yield any deflation after $30 \times 4 = 120$ iterations. More than 380 iterations are necessary until the first deflation occurs. The quick convergence of the QR algorithm is recovered by balancing A (see next section) before calling `DLAHQR`. Note, however, that this is not always a viable option, e.g., if the user requires A to be exclusively transformed by orthogonal matrices. ◊

An example of practical relevance where the current implementation of the QR algorithm fails to converge can be found in [36]. Shift blurring is, especially for large numbers of shifts, a significant source of slow convergence in finite-precision arithmetic, see Section 1.5.3 below.

1.4 Balancing

Balancing is a preprocessing step, which can have positive effects on the accuracy and performance of numerical methods for computing eigenvalues and therefore has obtained early attention in the development of numerical linear algebra [257, 263]. Balancing usually consists of two stages. In the first stage, the matrix is permuted in order to make it look closer to (block) upper triangular form. The second stage consists of a diagonal similarity transformation (scaling), which aims at equilibrating the row and column norms of the matrix.

1.4.1 Isolating Eigenvalues

In the first stage of the balancing algorithm by Parlett and Reinsch [263], a permutation matrix P is constructed such that $P^T AP$ takes the form

$$P^T AP = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ 0 & A_{22} & A_{23} \\ 0 & 0 & A_{33} \end{bmatrix} = \begin{bmatrix} \searrow & \square & \square \\ 0 & \square & \square \\ 0 & 0 & \searrow \end{bmatrix}, \tag{1.48}$$

i.e., $A_{11} \in \mathbb{R}^{(i_l-1)\times(i_l-1)}$ and $A_{33} \in \mathbb{R}^{(n-i_h)\times(n-i_h)}$ are upper triangular matrices. The obvious benefits are that the so called *isolated eigenvalues* contained in these triangular matrices can be read off without any roundoff error and that the order of the eigenvalue problem is reduced to $i_h - i_l + 1$. The remaining middle block A_{22} is characterized by the property that each column and row contains at least one nonzero off-diagonal element, which implies that no further eigenvalues can be isolated.

The matrix P is composed of elementary permutation matrices

$$P_{ij} = I - e_i e_i^T - e_j e_j^T + e_i e_j^T + e_j e_i^T. \tag{1.49}$$

Such a matrix effects the swap of columns i and j if post-multiplied to a matrix. Correspondingly, the pre-multiplication by P_{ij} swaps rows i and j. Based on these two properties, one can construct a simple algorithm for producing a decomposition of the form (1.48), see Algorithm 4.

The computational work of Algorithm 4 is somewhat difficult to measure in terms of flops, as it does not require any floating point operations. The number of binary comparisons amounts to $\mathcal{O}((i_l+i_h-n)n)$. Experience gained by numerical experiments tells that performance benefits gained from the order reduction of the eigenvalue problem greatly outweigh the execution time needed by Algorithm 4.

1.4.2 Scaling

In the second stage, a diagonal matrix D is constructed so that $B = D^{-1}A_{22}D$ is nearly balanced. Here, A_{22} is the unreduced middle block in the block triangular matrix (1.48). An $m \times m$ matrix B is called *balanced* in a vector norm $\|\cdot\|$ if its row and column norms are equal, i.e., $\|e_j^T B\| = \|Be_j\|$ for $j = 1, \ldots, m$. Under the assumption that A_{22} is irreducible, Osborne [257] showed that if $B = D^{-1}A_{22}D$ is balanced in Euclidean norm, then B has minimal Frobenius norm among all diagonal similarity transformations of A_{22}. Note that Algorithm 4 does not necessarily produce an irreducible matrix A_{22}, an algorithm that guarantees this property will be described in Section 3.3.1.

The iterative procedure proposed by Osborne for computing the scaling matrix D is used in the balancing algorithm by Parlett and Reinsch [263]. Although this algorithm is capable to balance a matrix in any vector norm, the LAPACK implementation `DGEBAL` uses the 1-norm because of efficiency considerations. Grad [142] showed that this algorithm converges and yields a matrix D that balances A_{22} in 1-norm, again under the assumption that A_{22} is irreducible. For more work on balancing, see [90, 91, 117, 156, 185, 289, 310].

Algorithm 4 Permutation to block triangular form [263]

Input: A general matrix $A \in \mathbb{R}^{n\times n}$.

Output: A permutation matrix P such that P^TAP is in block triangular form (1.48) for some integers i_l, i_h. The matrix A is overwritten by P^TAP.

$i_l \leftarrow 1, \quad i_h \leftarrow n, \quad \text{swapped} \leftarrow 1$
$P \leftarrow I_n$
while (swapped $= 1$) **do**
  swapped $\leftarrow 0$
  % Search for row having zero off-diagonal elements in active
  % submatrix $A(i_l : i_h, i_l : i_h)$ and swap with i_hth row.
  $i \leftarrow i_l$
  while $(i \le i_h)$ **and** (swapped $= 0$) **do**
    if $\sum_{\substack{j=i_l \\ j\neq i}}^{i_h} |a_{ij}| = 0$ **then**
      $A \leftarrow P_{i,i_h}^T A P_{i,i_h}, \quad P \leftarrow P P_{i,i_h}$
      $i_h \leftarrow i_h - 1, \quad \text{swapped} \leftarrow 1$
    end if
    $i \leftarrow i + 1$
  end while
  % Search for column having zero off-diagonal elements in active
  % submatrix $A(i_l : i_h, i_l : i_h)$ and swap with i_lth column.
  $j \leftarrow i_l$
  while $(j \le i_h)$ **and** (swapped $= 0$) **do**
    if $\sum_{\substack{i=i_l \\ i\neq j}}^{i_h} |a_{ij}| = 0$ **then**
      $A \leftarrow P_{i_l,j}^T A P_{i_l,j}, \quad P \leftarrow P P_{i_l,j}$
      $i_l \leftarrow i_l + 1, \quad \text{swapped} \leftarrow 1$
    end if
    $j \leftarrow j + 1$
  end while
end while

Algorithm 5 is basically LAPACK's implementation of the Parlett-Reinsch algorithm. To avoid any roundoff errors in this algorithm, the scaling factor β should be a power of the machine base (usually 2). In the LAPACK implementation, $\beta = 2^3 = 8$ is used. If we apply this algorithm to the matrix P^TAP in block triangular form (1.48), then, from the discussion above, we may conclude that the algorithm returns a nearly balanced matrix $D^{-1}A_{22}D$. Algorithm 5 requires at most $4kn^2 + \mathcal{O}(n)$ flops, where $k = \mathcal{O}(1)$ denotes the number of iterations until convergence.

Algorithm 5 restricts the diagonal entries of D to powers of β. This generally yields matrices that are only nearly balanced, as demonstrated by the following example.

Algorithm 5 Balancing [263]

Input: A matrix $A \in \mathbb{R}^{n\times n}$ having the block triangular form (1.48) for integers i_l, i_h. A scaling factor $\beta > 1$.

Output: A diagonal matrix $\tilde{D} = I_{i_l-1} \oplus D \oplus I_{n-i_h}$, with diagonal entries that are powers of β, so that $D^{-1}A_{22}D$ is nearly balanced in 1-norm. The matrix A is overwritten by $\tilde{D}^{-1}A\tilde{D}$.

$\tilde{D} \leftarrow I_n$
converged $\leftarrow 0$
while (converged $= 0$) **do**
  converged $\leftarrow 1$
  for $j \leftarrow i_l, \ldots, i_h$ **do**
    $c \leftarrow \sum_{\substack{i=i_l \\ i\neq j}}^{i_h} |a_{ij}|, \quad r \leftarrow \sum_{\substack{k=i_l \\ k\neq j}}^{i_h} |a_{jk}|, \quad s \leftarrow c + r, \quad \text{scal} \leftarrow 1$
    while $(c < r/\beta)$ **do**
      $c \leftarrow c\beta, \quad r \leftarrow r/\beta, \quad \text{scal} \leftarrow \text{scal} \times \beta$
    end while
    while $c \geq r\beta$ **do**
      $c \leftarrow c/\beta, \quad r \leftarrow r\beta, \quad \text{scal} \leftarrow \text{scal}/\beta$
    end while
    % Scaling is only applied when it results in a non-negligible reduction of
    % the column and row norms.
    if $(c + r) < 0.95 \times s$ **then**
      converged $\leftarrow 0, \quad \tilde{d}_{jj} \leftarrow \text{scal} \times \tilde{d}_{jj}$
      $A(:, j) \leftarrow \text{scal} \times A(:, j), \quad A(j, :) \leftarrow 1/\text{scal} \times A(j, :)$
    end if
  end for
end while

Example 1.28 ([91]). Consider the matrix

$$A = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 2 & 0 & 1/2 & 0 \\ 0 & 2 & 0 & 1/2 \\ 0 & 0 & 2 & 0 \end{bmatrix}$$

This matrix can be balanced by the diagonal matrix $D = \text{diag}(4, 2, 1, 1/2)$. Algorithm 5 with $\beta = 2$, however, returns

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1/2 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

which is an unbalanced matrix. ◊

The two ingredients of balancing, Algorithms 4 and 5, are implemented in the LAPACK routine `DGEBAL`. Note that the information contained in the

permutation matrix P and the scaling matrix $\tilde{D}$ is stored in a vector "scal" of length n as follows. If $j \in [1, i_l - 1] \cup [i_h + 1, n]$, then the permutation $P_{j,\mathrm{scal}(j)}$ has been applied in the course of Algorithm 4. Otherwise, $\mathrm{scal}(j)$ contains $\tilde{d}_{jj}$, the jth diagonal entry of the diagonal matrix $\tilde{D}$ returned by Algorithm 5.

The backward transformation (i.e., multiplication with $(P\tilde{D})^{-1}$) is needed for post-processing computed eigenvectors or invariant subspaces; it is implemented in the LAPACK routine `DGEBAK`.

1.4.3 Merits of Balancing

The following positive effects can be attributed to balancing.

First of all, often the norm of the matrix is decreased which equally decreases the norm of the backward error caused by subsequent orthogonal transformations. Occasionally, however, the norm of the matrix returned by Algorithm 5 is slightly increased. This increase is limited as the returned matrix is nearly balanced in 1-norm, meaning that it is, up to a factor of $\sqrt{n}$, nearly balanced in Euclidean norm.

Second, eigenvalues isolated by Algorithm 4 are read off without any roundoff error. Also, eigenvalues of the scaled block $D^{-1}A_{22}D$ are sometimes more accurately computed, mainly due to the norm decrease. Some numerical experiments illustrating this effect for matrices from the test matrix collection [16] can be found in [90, 91]. It should be emphasized that a norm decrease does not necessarily lead to more accurate eigenvalues, in fact it has been reported that balancing negatively influences the eigenvalue accuracy of certain matrices with unusual scaling. A simple way to come up with such a situation is to reduce a matrix with random entries first to Hessenberg form and balance afterwards [356].

Finally, balancing can have a positive impact on the computational time needed by subsequent methods for computing eigenvalues. Not only the dimension of the eigenvalue problem is reduced to $i_h - i_l + 1$ but also the diagonal scaling of the unreduced block A_{22} may improve the convergence of these methods. For example, balancing greatly improves the efficiency of the QR algorithm applied to the matrix TOLS1090 from the test matrix collection [16], see Section 1.5.5. An extreme case is given in Example 1.27 above, where the QR algorithm converges for the balanced matrix but does not converge for the unbalanced matrix within $30n$ iterations.

1.5 Block Algorithms

The bulge of the computational work of Algorithm 1, which reduces a matrix to Hessenberg form, consists of calls to `DLARF`, a LAPACK routine for applying Householder matrices based on level 2 BLAS operations. Each call to `DLARF` performs only $\mathcal{O}(n^2)$ floating point operations while moving $\mathcal{O}(n^2)$ memory.

Algorithms with such a high communication/computation ratio often perform poorly on modern computers with a deep memory hierarchy, ranging from large and slow to small and fast memory. An instructive discussion on this matter can be found in Demmel's book [103, Sec. 2.6].

The LAPACK routine for reduction to Hessenberg form (`DGEHRD`) attains high efficiency by (implicitly) employing so called WY representations of products of Householder matrices. The application of such representations can be formulated in terms of matrix-matrix multiplications leading to reduced memory traffic, which in turn means better performance. For example, on an average work station, computing the Hessenberg form of a 1000×1000 matrix would require more than three times the time needed if no WY representations were employed.

1.5.1 Compact WY Representation

The following theorem provides the basis for block algorithms for orthogonal decompositions. It shows how to reorganize the application of a product of Householder matrices so that level 3 BLAS can be facilitated.

Theorem 1.29 (Compact WY representation [282]). *Let $k \leq n$ and $Q = H_{j_1}(x_1) \cdot H_{j_2}(x_2) \cdots H_{j_k}(x_k)$ where $H_{j_i}(x_i)$ are Householder matrices of the form (1.40), and $j_i \in [1, n]$, $x_i \in \mathbb{R}^{2n}$. Then there exist an upper triangular matrix $T \in \mathbb{R}^{k \times k}$ and a matrix $W \in \mathbb{R}^{n \times k}$ so that $Q = I_n + WTW^T$.*

Proof. The proof is by induction w.r.t. k. The case $k = 0$ is straightforward. Let Q have a compact WY representation $Q = WTW^T$. Consider for $j = j_{k+1}$ the product

$$\tilde{Q} := Q \cdot H_j(x_j) = Q(I - \beta v v^T).$$

Then a compact WY representation for $\tilde{Q}$ is given by $\tilde{Q} = I_n + \tilde{W}\tilde{T}\tilde{W}^T$, where

$$\tilde{T} = \begin{bmatrix} T & -\beta TW^T v \\ 0 & -\beta \end{bmatrix}, \quad \tilde{W} = \begin{bmatrix} W & v \end{bmatrix}, \tag{1.50}$$

concluding the proof. □

Often, $j_i = i$, in which case a proper implementation of the construction given in the proof of Theorem 1.29 requires $k^2 n - \frac{1}{3}k^3 + \mathcal{O}(k^2)$ flops, see also the LAPACK routine `DLARFT`. Furthermore, an inspection of (1.50) reveals that the upper $k \times k$ block of W is a unit lower triangular matrix. The application of $I_n + WTW^T$ to an $n \times q$ matrix is implemented in the LAPACK routine `DLARFB`. This routine facilitates the Level 3 BLAS operations `DTRMM`, `DGEMM`, and requires about $4knq - k^2 q$ flops.

Remark 1.30. The representation $I_n + WTW^T$ is called compact WY representation because it has a "non-compact" predecessor: $I_n + WY^T$, where $Y = WT^T$ [47]. The obvious disadvantage of using $I_n + WY^T$ is the extra workspace necessary for storing the $n \times k$ matrix Y instead of the potentially much smaller $k \times k$ matrix T. ◊

As an immediate application of compact WY representations, we obtain Algorithm 6, a block algorithm for the computation of the orthogonal factor Q from Algorithm 1, reduction to Hessenberg form. The matrix Q is the product of $n-2$ Householder matrices: $Q = H_2(x_1) \cdot H_3(x_2) \cdots H_{n-1}(x_{n-2})$. For convenience only, we assume that $n-2 = N \cdot n_b$ for an integer N and a given block size n_b. In Algorithm 6, implemented as LAPACK routine `DORGHR`,

Algorithm 6 Block accumulation of Householder matrices

Input: Householder matrices $H_2(x_1), H_3(x_2), \ldots, H_{n-1}(x_{n-2})$. Integers N and n_b so that $n-2 = Nn_b$.

Output: An orthogonal matrix $Q = H_2(x_1) \cdot H_3(x_2) \cdots H_{n-1}(x_{n-2})$.

$Q \leftarrow I_n$
for $p \leftarrow N, N-1, \ldots, 1$ **do**
  Set $s = (p-1)n_b + 1$.
  Apply the construction given in the proof of Theorem 1.29 to compute the compact WY representation

$$H_{s+1}(x_s) \cdot H_{s+2}(x_{s+1}) \cdots H_{s+n_b}(x_{s+n_b-1}) = I_n + WTW^T.$$

  Update $Q(:, s:n) \leftarrow (I_n + WTW^T)Q(:, s:n)$.
end for

$$\frac{1}{2}\frac{n^3}{N} + \frac{1}{6}\frac{n^3}{N^2} + \mathcal{O}(n^2)$$

flops are required to generate the compact WY representations, while

$$\frac{4}{3}n^3 + \frac{3}{2}\frac{n^3}{N} + \frac{1}{6}\frac{n^3}{N^2} + \mathcal{O}(n^2)$$

flops are necessary to apply them to the matrix Q. On the other hand, the unblocked algorithm, based on calls to `DLARF`, requires $\frac{4}{3}n^3 + \mathcal{O}(n^2)$ flops. This shows that blocking is more expensive by roughly a factor of $(1 + 3/(2N))$, as far as flop counts are concerned. The fraction of operations carried out by Level 3 BLAS is attractively high; it tends to 100% when n is increasing while n/N remains constant.

1.5.2 Block Hessenberg Reduction

Developing a block algorithm for reducing a matrix to Hessenberg form, Algorithm 1, is complicated by dependencies between the individual Householder matrices. To resolve these dependencies, Dongarra, Sorensen and Hammarling [111] proposed to maintain a relationship of the form

$$A^{(k)} = \tilde{A}^{(k)} + WX^T + YW^T, \tag{1.51}$$

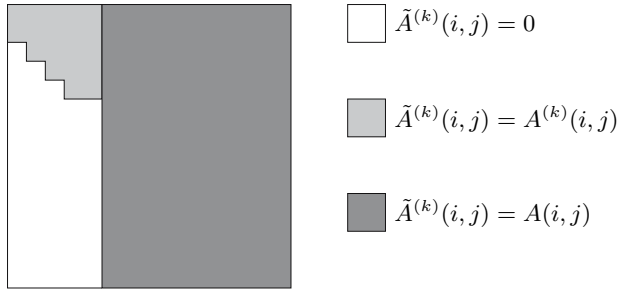


Fig. 1.6. Structure of $\tilde{A}^{(k)}$ for $k = 5, n = 15$. The white and pale-gray parts contain the reduced k-panel, consisting of elements of the matrix $A^{(k)}$. The dark-gray part contains elements of the original matrix A.

where $A^{(k)}$ is the partly reduced matrix obtained after k loops of Algorithm 1 have been applied to A. The matrix $\tilde{A}^{(k)}$ is in the first k columns, the so called *k-panel*, identical to $A^{(k)}$ but in the remaining columns identical to the original matrix A, see also Figure 1.6. In this way, a large part of the updates in Algorithm 1 can be delayed and performed later on, by means of two rank-k updates using level 3 BLAS.

A more economic variant of this idea is implemented in the LAPACK routine `DGEHRD`. Instead of (1.51), a relationship of the form

$$A^{(k)} = (I + WTW^T)^T(\tilde{A}^{(k)} + Y\tilde{W}^T) \tag{1.52}$$

is maintained, where $I + WTW^T$ is the compact WY representation of the first k Householder matrices employed in Algorithm 1. The matrix $\tilde{W}$ equals W with the first k rows set to zero. Algorithm 1.52 produces (1.52) and is implemented in the LAPACK routine `DLAHRD`. After this algorithm has been applied, the matrix $A^{(k)}$ in (1.52) is computed using level 3 BLAS operations. An analogous procedure is applied to columns $k+1, \ldots, 2k$ of $A^{(k)}$, which yields the matrix $A^{(2k)}$ having the first $2k$ columns reduced. Pursuing this process further finally yields a complete Hessenberg form of A.

The described block algorithm for reduction to Hessenberg form as implemented in the LAPACK routine `DGEHRD` requires

$$\frac{10}{3}n^3 + 3\frac{n^3}{N} - \frac{n^3}{N^2} + \mathcal{O}(n^2)$$

flops. This compares favorably with the $\frac{10}{3}n^3 + \mathcal{O}(n^2)$ flops needed by the unblocked algorithm. Unfortunately, the fraction of operations carried out by level 3 BLAS approaches only 70% when n is increasing while n/N remains constant. Thus, it can be expected that the performance of `DGEHRD` is worse than the performance of `DORGHR`, which has a level 3 fraction of approximately 100%. Indeed, this effect can be observed in practice, see Figure 1.7. Recent work by Quintana-Ortí and van de Geijn [266] shows that the level 3 BLAS

Algorithm 7 Panel reduction

Input: A matrix $A \in \mathbb{R}^{n\times n}$ and an integer $k \le n-2$.
Output: Matrices $T \in \mathbb{R}^{k\times k}, W \in \mathbb{R}^{n\times k}$ and $Y \in \mathbb{R}^{n\times k}$ yielding a representation of the form (1.52). The matrix A is overwritten by $\tilde{A}^{(k)}$.

$T \leftarrow [], \quad W \leftarrow [], \quad Y \leftarrow []$
for $j \leftarrow 1, \ldots, k$ **do**
  if $j > 1$ **then**
    % Update jth column of A.
    $A(:,j) \leftarrow A(:,j) + Y\tilde{W}(j,:)^T$
    $A(:,j) \leftarrow A(:,j) + WT^TW^TA(:,j)$
  end if
  Compute jth Householder matrix $H_{j+1}(A(:,j)) \equiv I - \beta vv^T$.
  $A(:,j) \leftarrow (I - \beta vv^T)A(:,j)$
  $x \leftarrow -\beta W^T v$
  $Y \leftarrow [Y, Yx - \beta Av]$
  $T \leftarrow \begin{bmatrix} T & Tx \\ 0 & -\beta \end{bmatrix}$
end for

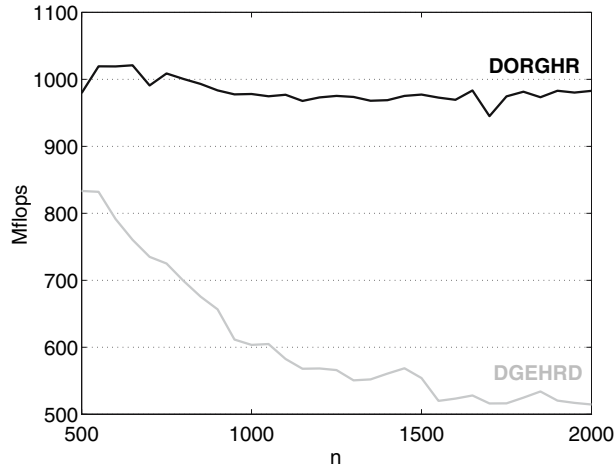


Fig. 1.7. Measured megaflops per second for `DGEHRD` and `DORGHR` applied to random matrices of order $500, 550, \ldots, 2000$.

fraction can be increased to 80% by restricting the update of the jth columns of A and Y in Algorithm 7 to the last $n-j$ entries.

A two-stage approach to Hessenberg reduction, quite similar to the block algorithms for Hessenberg-triangular reduction described in Section 2.5.1, has already been suggested by Bischof and Van Loan [47]. However, this requires roughly 60% more flops. It is thus questionable whether such an approach can achieve higher performance than `DGEHRD`, see also the discussion in Lang's thesis [208, Sec. 2.4.3].

1.5.3 Multishifts and Bulge Pairs

Early attempts to improve the performance of the QR algorithm focused on using shift polynomials of high degree [17], leading to medium-order Householder matrices during the QR iteration and enabling the efficient use of WY representations. This approach, however, has proved disappointing due to the fact that the convergence of such a large-bulge multishift QR algorithm is severely affected by roundoff errors [114, 350, 351]. To explain this phenomenom, Watkins [350, 351] has introduced the notion of bulge pairs, which is in the following briefly described.

Assume that an implicit shifted QR iteration with m shifts, Algorithm 2, is applied to an unreduced $n \times n$ Hessenberg matrix H with $n > m$. Let x be a multiple of the first column of the shift polynomial $p(H) = (H - \sigma_1 I) \cdots (H - \sigma_m I)$. The *initial bulge pair* is the matrix pair (B_0, N), where N is the $(m+1) \times (m+1)$ Jordan block belonging to the eigenvalue zero and

$$B_0 = [x(1:m+1), H(1:m+1, 1:m)] = \begin{bmatrix} x_1 & h_{11} & \cdots & h_{1m} \\ x_2 & h_{21} & \ddots & \vdots \\ \vdots & & \ddots & h_{mm} \\ x_{m+1} & 0 & & h_{m+1,m} \end{bmatrix}.$$

There is a surprisingly simple relationship between the shifts and the eigenvalues of this matrix pair.[3]

Theorem 1.31 ([351]). *The shifts $\sigma_1, \ldots, \sigma_m$ are the finite eigenvalues of the initial bulge pair (B_0, N).*

During the course of a QR iteration, a bulge is created at the top left corner of H and chased down to the bottom right corner along the first subdiagonal of H, see page 29. Let $H^{(j)}$ denote the updated matrix H after the bulge has been chased $j-1$ steps, i.e., $j-1$ loops of Algorithm 1 have been applied after the update $H \leftarrow H_1(x) \cdot H \cdot H_1(x)$. Then the bulge resides in the submatrix

$$B_j = H^{(j)}(j+1 : j+m+1, j : j+m), \tag{1.53}$$

which is exactly the submatrix designated by the entries $\hat{b}$ in (1.43).

Theorem 1.32 ([351]). *The shifts $\sigma_1, \ldots, \sigma_m$ are the finite eigenvalues of the j*th bulge pair (B_j, N).

Note that the definition of the bulge B_j is only possible for $j \leq n - m - 1$, since otherwise (1.53) refers to entries outside of $H^{(j)}$. Such a situation is displayed in (1.44). This issue can be resolved; by adding virtual rows and columns to the matrix $H^{(j)}$, see [351], Theorem 1.32 can be extended to the case $j > n - m - 1$.

[3] For the definition of eigenvalues of matrix pairs, the reader is referred to Section 2.1 in the next chapter.

Theorem 1.32 shows how the shifts are propagated during the QR iteration. In order to achieve quadratic convergence, which usually takes place at the bottom right corner of H, it is essential that the information contained in these shifts is properly propagated. However, several numerical experiments conducted in [351] show that the finite eigenvalues of the bulge pairs (B_j, N) become, as m increases, extremely sensitive to perturbations. Already for $m = 24$, there are some pairs (B_j, N), whose finite eigenvalues are completely swamped with roundoff errors and have no significant digit in common with the intended shifts.

Although no exact relation between the quality of shifts in the bulge pairs and the convergence of the QR algorithm in finite-precision arithmetic was proven in [351], it is intuitively clear that this sensitivity may affect the performance severely. Note that this does not necessarily imply that the QR algorithm does not converge for very large m but the convergence is much slower and must be attributed to linear convergence often taking place at the top left corner of H.

1.5.4 Connection to Pole Placement

Although convincing numerical experiments for the described shift blurring effects are provided in [351], there has been no explanation why bulge pencils are getting so sensitive as m increases. In this section, we connect the computation of x, the first column of the shift polynomial, to the pole placement problem in systems and control theory, see Section A.1.3. This will result in a semi-heuristic explanation for the sensitivity of the initial bulge pair.

Figure 1.8 illustrates this phenomenom for matrices generated with the MATLAB command `triu(rand(300,1))`. As m increases, deflations taking place at the top left corner (signaling linear convergence) start dominating deflations at the bottom right corner (signaling quadratic convergence). The QR algorithm requires about 7.4×10^8 flops for $m = 24$, while it requires only 2.9×10^8 flops for $m = 2$.

First, we note that the unreducedness of A implies

$$x_{m+1} = \prod_{i=1}^{m+1} a_{i+1,i} \neq 0.$$

Furthermore, it can be easily shown that neither the implicitly shifted QR iteration nor the statement of Theorem 1.31 is affected if we replace x by a nonzero scalar multiple thereof. Hence, we may assume without loss of generality that x is normalized such that $x_{m+1} = 1$. By applying a simple equivalence transformations to the initial bulge pair (B_0, N), Theorem 1.31 shows that the shifts $\sigma_1, \ldots, \sigma_m$ are the eigenvalues of the matrix

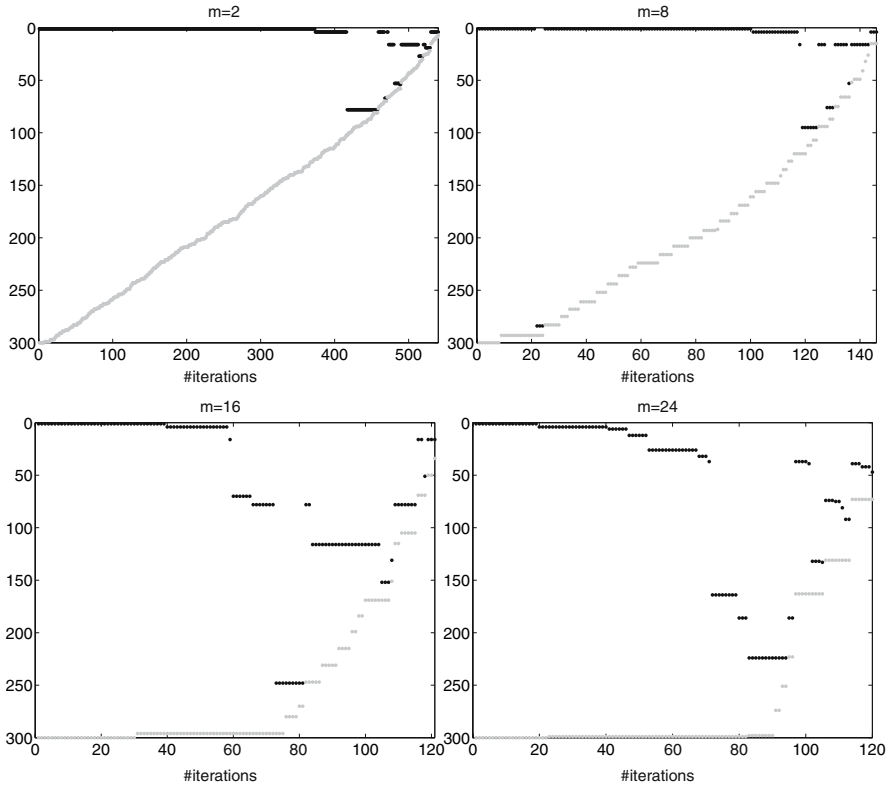


Fig. 1.8. Lower (red dots) and higher (blue dots) indices of the active submatrices that have order larger than m during the implicit shifted QR iteration with m Francis shifts.

$$C = H(1:m,1:m) - h_{m+1,m}x(1:m)e_m^T \tag{1.54}$$
$$= \begin{bmatrix} h_{11} & h_{12} & \dots & h_{1,m-1} & h_{1m} - h_{m+1,m}x_1 \\ h_{21} & h_{22} & \dots & h_{2,m-1} & h_{2m} - h_{m+1,m}x_2 \\ 0 & h_{32} & \dots & h_{3,m-1} & h_{3m} - h_{m+1,m}x_3 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & h_{m,m-1} & h_{mm} - h_{m+1,m}x_m \end{bmatrix}.$$

Next, consider the single-input control system

$$\dot{z}(t) = H(1:m,1:m)^T z(t) - (h_{m+1,m}e_m)u(t) \tag{1.55}$$

with state vector $z(\cdot)$ and input $u(\cdot)$. The linear state feedback $u(t) = x(1:m)^T z(t)$ yields the closed-loop matrix C^T, where C is defined as in (1.54).

Hence, the feedback vector $x(1:m)$ places the poles of the open loop system (1.55) to $\sigma_1, \ldots, \sigma_m$. Since $x(1:m)$ is uniquely defined by this property, we obtain the following connection:

Any pole placement algorithm for single-input systems is a suitable method for computing a multiple of the first column of the shift polynomial; and vice versa.

To some extent, this connection has already been used by Varga for designing a multi-shift pole placement algorithm [334]. A not-so-serious application is the expression of the QL iteration [116], a permuted version of the QR iteration, in three lines of MATLAB code, using functions of the MATLAB Control Toolbox [235]:

```
s = eig(H(1:m,1:m));
x = acker(H(n-m+1:n,n-m+1:n)',H(n-m,n-m+1)*eye(m,1),s);
H = ctrbf(H, [zeros(1,n-m-1) 1 x]', []);
```

An exceedingly more serious consequence is caused by the observation that placing a large number of poles in a single-input problem is often very ill-conditioned [157, 165] in the sense that the poles of the closed loop system are very sensitive to perturbations in the input data. The nature of this ill-conditioning was analyzed in detail by Mehrmann and Xu [243, 244]. Assume that the input data of the pole placement problem ($A = H(1:m, 1:m)^T, b = -h_{m+1,m}e_m$ and $\sigma_1, \ldots, \sigma_m$) are perturbed by sufficiently small perturbations $\triangle A$, $\triangle b$ and $\triangle\sigma_1, \ldots, \triangle\sigma_m$. Set

$$\varepsilon = \max\{\|[\triangle A, \triangle b]\|, |\triangle\sigma_1|, \ldots, |\triangle\sigma_m|\},$$

and let $\hat{f}$ be the feedback vector defined by the perturbed problem. Then, it was shown in [244, Thm. 1.1] that the eigenvalues $\hat{\sigma}_1, \ldots, \hat{\sigma}_m$ of $A + b\hat{f}^T$ satisfy

$$|\hat{\sigma}_i - \sigma_i| \leq \left(1 + \|G\|_2\|G^{-1}\|_2\sqrt{1 + \|\hat{f}\|_2}\right)\varepsilon + \mathcal{O}(\varepsilon^2), \tag{1.56}$$

where G is the eigenvector matrix of the closed loop matrix $C^T = A + bf^T$, normalized such that all columns of G have unit norm. Although (1.56) is only an upper bound, numerical experiments in [243, 244] suggest that (1.56) catches the qualitative behavior of the maximal eigenvalue error rather well.

The presence of the condition number $\|G\|_2\|G^{-1}\|_2$ in (1.56) is particularly worrisome. Often, this term grows rapidly with m, even exponential growth can be proven for some cases [243, Ex. 1]. Indeed, such an effect can be observed in the QR algorithm. For this purpose, we constructed the closed loop matrix C^T, as defined in (1.54), for a matrix A generated by the MATLAB command `hess(rand(250))`. Figure 1.9 displays the condition number of the

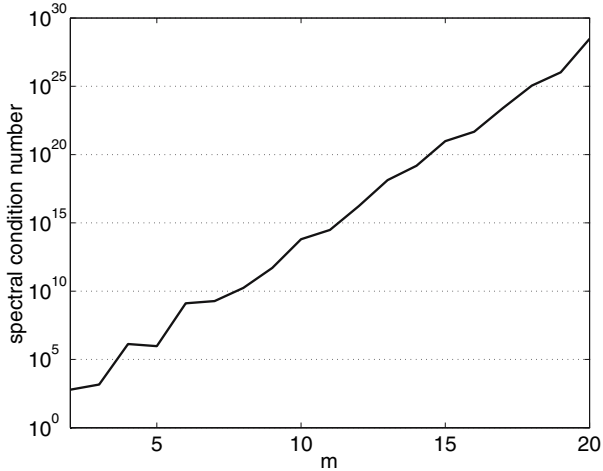


Fig. 1.9. Condition number of the eigenvector matrix G of the closed loop matrix C^T.

corresponding eigenvector matrix G for $m = 2, \ldots, 15$ Francis shifts, clearly exhibiting exponential growth.

The described connection to the notoriously ill-conditioned pole placement problem yields an explanation for the sensitivity of the initial bulge pair. However, it does not explain the sensitivity of the bulge pairs (B_1, N), (B_2, N), ... On the other hand, there is little hope that the shifts, once destroyed because of the sensitivity of $(B_0, \lambda N)$, recover during the bulge chasing process although this event sometimes occurs in practice [351].

1.5.5 Tightly Coupled Tiny Bulges

The trouble with shift blurring has led researchers to develop variants of the implicit shifted QR algorithm that still rely on a large number of simultaneous shifts but chase several tiny bulges instead of one large bulge. This idea has been proposed many times, see [61, 114, 161, 190, 208, 209, 349]. In this section, we describe a slightly generalized variant of an approach proposed independently by Braman, Byers, and Mathias [61] as well as by Lang [208].

In the following, m denotes the number of simultaneous shifts to be used in each QR iteration and n_s denotes the number of shifts contained in each bulge. It is assumed that m is an integer multiple of n_s. To avoid shift blurring phenomena we use tiny values for n_s, say $n_s \in [2, 6]$.

Our algorithm performs an implicit shifted QR iteration with m Francis shifts to a Hessenberg matrix H and consists of three stages, which are described in more detail below. First, a tightly coupled chain of m/n_s bulges is bulge-by-bulge introduced in the top left corner of H. Second, the whole chain is chased down along the subdiagonal until the bottom bulge reaches

the bottom right corner of H. Finally, all bulges are bulge-by-bulge chased off this corner.

Introducing a chain of bulges

Given a set of m Francis shifts, we partition this set into subsets Σ_1, Σ_2, ..., Σ_{m/n_s}. Each Σ_j contains n_s shifts and is closed under complex conjugation. We apply the implicit shifted QR iteration with the shifts contained in Σ_1 and interrupt the bulge chasing process as soon as the bottom right corner of the bulge touches the $(p_h - 1, p_h)$ subdiagonal entry of H, where $p_h = (m/n_s)(n_s + 1) + 1$. Next, the bulge belonging to Σ_2 is introduced and chased so that its bottom right corner is at the $(p_h - n_s - 2, p_h - n_s - 1)$ subdiagonal entry. This process is continued until all m/n_s bulges are stringed like pearls on the subdiagonal of the submatrix $H(1 : p_h, 1 : p_h)$, see Figure 1.10. Note

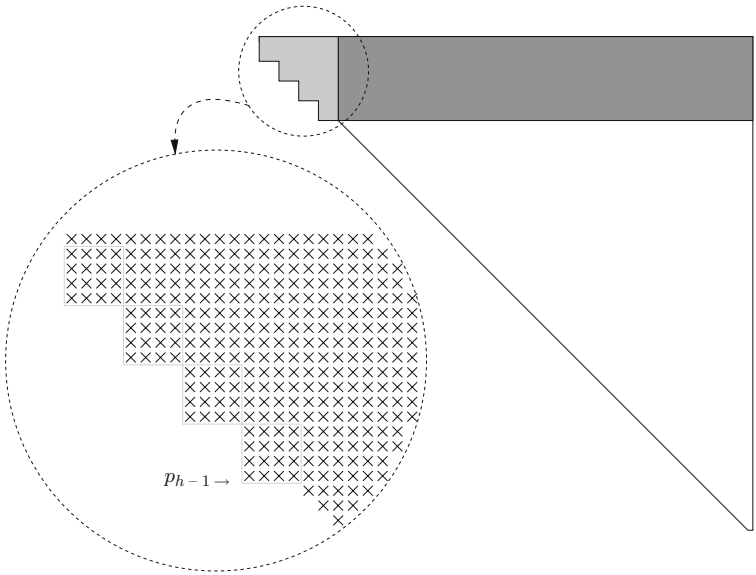


Fig. 1.10. Introducing a chain of $m/n_s = 4$ tightly coupled bulges, each of which contains $n_s = 3$ shifts.

that only this submatrix (painted pale gray in Figure 1.10) must be updated during the bulge chasing process. To update the remaining part of H (painted dark gray), all employed orthogonal transformations are accumulated into a $p_h \times p_h$ matrix U. This enables us to use matrix-matrix multiplications:

$$H(1 : p_h, (p_h + 1) : n) \leftarrow U^T H(1 : p_h, (p_h + 1) : n).$$

Chasing a chain of bulges

Suppose that a chain of bulges resides on the subdiagonal of the submatrix $H(p_l : p_h, p_l : p_h)$, where $p_h = p_l + (n_s + 1)m/n_s$. In the beginning, we have $p_l = 1$ and $p_h = (m/n_s)(n_s + 1) + 1$ but we will now subsequently increase these values by chasing the complete chain. To move the chain to the submatrix $H(p_l + k : p_h + k, p_l + k : p_h + k)$, each individual bulge is chased k steps, as depicted in Figure 1.11. This is done in bottom-to-top order so that

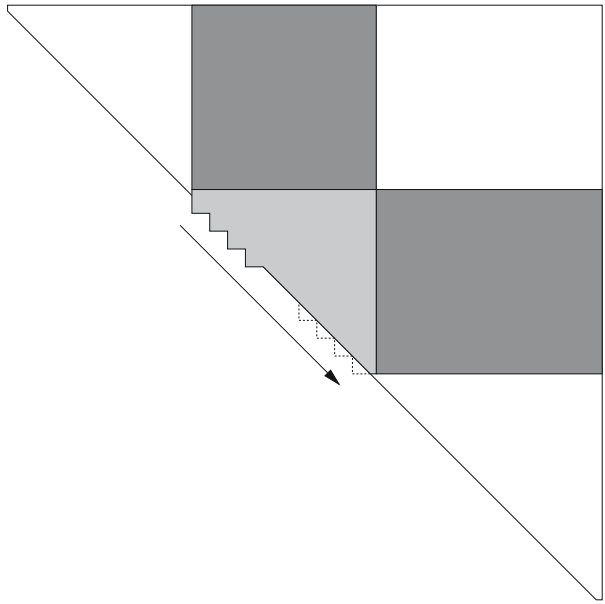


Fig. 1.11. Chasing a chain of $m/n_s = 4$ tightly coupled bulges.

no bulges will cross each other.

Again only a submatrix, namely $H(p_l : p_h + k, p_l : p_h + k)$, must be updated during the bulge chasing process. To update the rest of the matrix, we accumulate all transformations in an orthogonal matrix U of order $((n_s + 1)m/n_s + k + 1)$ and use matrix-matrix multiplications:

$$H(p_l : p_h + k, (p_h + 1) : n) \leftarrow U^T H(p_l : p_h + k, (p_h + 1) : n),$$
$$H(1 : p_l - 1, p_l : p_h + k) \leftarrow H(1 : p_l - 1, p_l : p_h + k)\, U.$$

Note that U has a particular block structure that can be exploited to increase the efficiency of these multiplications:

$$U = \begin{bmatrix} 1 & 0 & 0 \\ 0 & U_{11} & U_{12} \\ 0 & U_{21} & U_{22} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \square & \diagup\!\square \\ 0 & \diagdown\!\square & \square \end{bmatrix}, \tag{1.57}$$

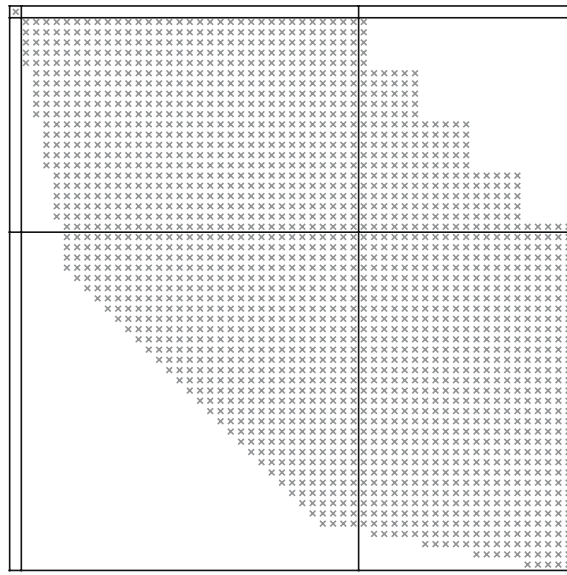


Fig. 1.12. Structure of the transformation matrix U for chasing a chain of $m/n_s = 5$ bulges, each of which contains $n_s = 4$ shifts, $k = 30$ steps.

i.e., the matrices $U_{12} \in \mathbb{R}^{l_1 \times l_1}$ and $U_{21} \in \mathbb{R}^{l_2 \times l_2}$, where $l_1 = (m/n_s)(n_s+1) - n_s$ and $l_2 = k + n_s$, are lower and upper triangular, respectively. There is even more structure present, as illustrated in Figure 1.12. It is, however, difficult to take advantage of this extra banded structure using level 3 BLAS [61].

The rare event of a zero subdiagonal entry between two consecutive bulges during the bulge chasing process is called a "vigilant" deflation [349]. Such a deflation causes a severe loss of information if bulges are chased from above through this zero subdiagonal entry. This can be avoided by reintroducing the bulges in the row in which the zero appears using essentially the same method that has been used for introducing bulges [61]. Note that it is not necessary to take care of vigilant deflations caused by small but not-too-tiny subdiagonal entries [350], see also Section 1.3.4.

Getting rid off a chain of bulges

Once the bottom bulge of the chain has reached the bottom right corner of H, the whole chain is bulge-by-bulge chased off this corner, similarly to the introduction of bulges at the top left corner of H.

Numerical results

For the purpose of illustrating the efficiency gains to be expected from using the described multishift QR algorithm with tightly coupled tiny bulges, let us

consider a Fortran 77 implementation called `MTTQR` based on an implementation available from Byers' web page[4]. Although this routine generally requires more flops than a doubleshift QR algorithm, see [61], it is likely that these extra costs are more than compensated by the fact that `MTTQR` facilitates Level 3 BLAS for a large part of the computation.

Note that the packing density of the bulges is getting higher as n_s, the number of shifts per bulge, increases. For example, a 16×16 principal submatrix of H may either contain 10 shifts distributed over five 3×3 bulges or it may contain 12 shifts distributed over three 5×5 bulges. In either case, essentially the same amount of operations is necessary to chase the chain of bulges from top to bottom. Hence, if shift blurring does not cause problems, using larger values for n_s can improve the efficiency of `MTTQR`.

To verify these statements, we applied `MTTQR` to a subset of real $n \times n$ matrices from the test matrix collection [16], see also Table 1.1. Note that TOLSBAL is the matrix obtained after balancing has been applied to the highly unbalanced matrix TOLS1090. For the parameters m (number of shifts

Matrix name	n	Description
OLM1000	1000	Olmstead model
TUB1000	1000	tubular reactor model
TOLS1090	1090	Tolosa matrix
TOLSBAL	1090	balanced Tolosa matrix
RDB1250	1250	reaction-diffusion Brusselator model, $L = 0.5$
RDB1250L	1250	reaction-diffusion Brusselator model, $L = 1$
BWM2000	2000	Brusselator wave model in chemical reaction
OLM2000	2000	Olmstead model
DW2048	2048	square dielectric waveguide
RDB2048	2048	reaction-diffusion Brusselator model, $L = 0.5$
RDB2048L	2048	reaction-diffusion Brusselator model, $L = 1$
PDE2961	2961	partial differential equation

Table 1.1. Subset of matrices from the test matrix collection [16].

in each iteration) and k (number of steps a chain of bulges is chased before off-diagonal parts are updated), we followed the recommendations given in [61]:

$$m = \begin{cases} 60, & \text{if } 1000 \le n < 2000, \\ 120, & \text{if } 2000 \le n < 2500, \\ 156, & \text{if } 2500 \le n < 3000, \end{cases}$$

and $k = 3/2 \times m - 2$.

From the cpu times displayed in Figure 1.13, we may conclude that `MTTQR` with $n_s = 2$ shifts per bulge requires considerably less time than the LAPACK routine `DHSEQR` for all considered matrices except TOLSBAL and TUB1000.

[4] `http://www.math.ku.edu/~byers/`

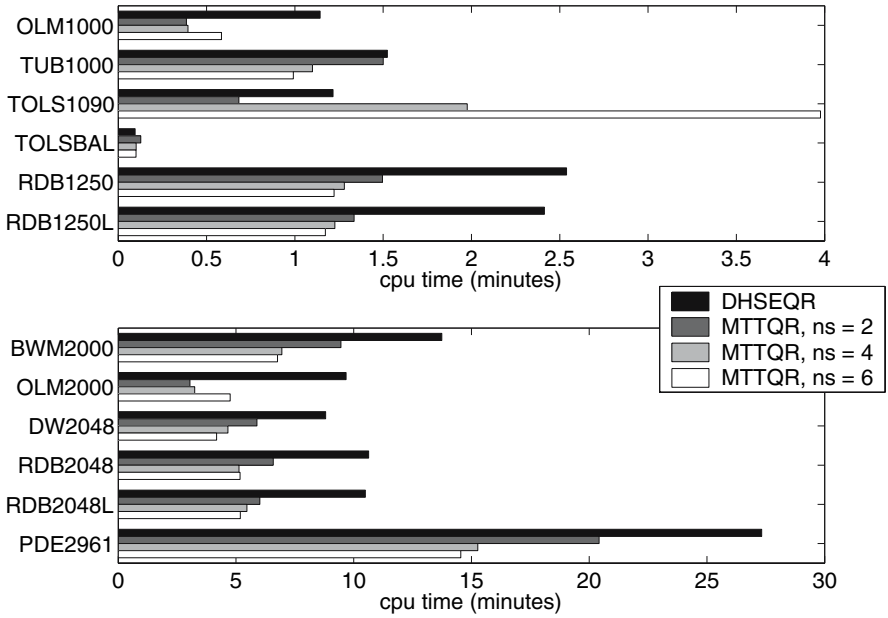


Fig. 1.13. Execution times for `DHSEQR` and `MTTQR`, the tiny-bulge multishift QR algorithm with $n_s \in \{2, 4, 6\}$ shifts per bulge, applied to matrices from the test matrix collection [16].

For TOLSBAL, `MTTQR` consumes 34% more time, which seems to be due to the fact that the QR algorithm converges so quickly that the overhead in `MTTQR` dominates any performance improvements gained by using matrix-matrix multiplications . For TUB1000, we obtain a performance improvement of only 1.5%, which is much less than for the other matrices, where this figure ranges from 25% up to 69%.

Increasing n_s from 2 to 4 often leads to some further speedups. A notable exception is TOLS1090, where `MTTQR` requires 190% more time if $n_s = 4$ instead of $n_s = 2$ is used. We believe that this behavior can be attributed to the poor balancing of this matrix, which seems to amplify shift blurring effects. The highest improvements can be obtained for TUB1000 and BWM2000, where `MTTQR` requires 27% less time if $n_s = 4$ instead of $n_s = 2$ is used.

1.6 Advanced Deflation Techniques

There have been few attempts to modify the deflation criteria described in Section 1.3.4 in order to accelerate convergence of the QR algorithm, see e.g. [4], and by far none of them has been such a success as the aggressive early deflation strategy developed by Braman, Byers, and Mathias [62]. In the following, this approach will be briefly described.

Aggressive Early Deflation

To approach the idea of aggressive early deflation strategy, it is helpful to lift the problem of deflation to a higher level of abstraction. Given a matrix H in unreduced Hessenberg form, we look for a perturbation E so that, after having $H + E$ reduced back to Hessenberg form, the eigenvalue problem is deflated into two or more smaller subproblems. Of course, backward stability should not be sacrificed, thus the norm of the perturbation E must be of order $\mathbf{u}\,\|H\|_F$. In the classical deflation strategies described in Section 1.3.4, the perturbation E is restricted to Hessenberg form. In this case, it is not necessary to reduce $H + E$ back to Hessenberg form and deflation is only possible if some subdiagonal entry of H is sufficiently small. Removing this restriction from E yields potentially for more deflations. The price to be paid is the extra effort for reducing $H + E$. In order to avoid that these additional expenses start dominating the computational time required by (multishift) QR iterations, it is necessary to restrict the perturbations to that part of the matrix where most deflations are expected to happen: in the bottom right corner of H. More precisely, if we partition

$$H = \begin{array}{c} \\ n-w-1 \\ 1 \\ w \end{array} \begin{array}{c} \begin{array}{ccc} n-w-1 & 1 & w \end{array} \\ \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ 0 & H_{32} & H_{33} \end{bmatrix} \end{array}, \tag{1.58}$$

then we only consider perturbations in the so called $w \times (w + 1)$ *deflation window* consisting of $[H_{32}, H_{33}]$.

Hence, our goal is to construct a perturbation of the form

$$E = \begin{array}{c} \\ n-w-1 \\ 1 \\ w \end{array} \begin{array}{c} \begin{array}{ccc} n-w-1 & 1 & w \end{array} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & E_{32} & E_{33} \end{bmatrix} \end{array}, \tag{1.59}$$

so that the reduction of $H + E$ to Hessenberg form results in a deflation. The reduction of $H + E$ requires only $\mathcal{O}(nw^2)$ flops. To see this, let $Q_1 = H_1(H_{32} + E_{32})$ denote a Householder matrix as defined in (1.40). Then, applying Algorithm 1 to $Q_1(H_{33} + E_{33})Q_1$ yields an orthogonal matrix Q_2 so that the matrix

$$\tilde{H} = (I_{n-w} \oplus Q)^T (H + E)(I_{n-w} \oplus Q),$$

with $Q = Q_1 Q_2$, is in Hessenberg form. We call E a *reducing perturbation* if some subdiagonal entry of $\tilde{H}$ is zero, implying that one or more eigenvalues of $\tilde{H}$ are deflated. The following well-known lemma relates reducing perturbations to the concept of controllability of matrix pairs, see Section A.1.2.

Lemma 1.33. *Let the matrices H and E have the form displayed in (1.58) and (1.59), respectively. Then E is a reducing perturbation if and only if the matrix pair $(H_{33} + E_{33}, H_{32} + E_{32})$ is not controllable.*

Proof. Using the reduction given above, let Q be an orthogonal matrix so that $Q^T(H_{32} + E_{32})$ is a multiple of the first unit vector and $Q^T(H_{33} + E_{33})Q$ is an upper Hessenberg matrix. Theorem A.6 implies that $(H_{33} + E_{33}, H_{32} + E_{32})$ is controllable if and only if the Krylov matrix

$$K_n(H_{33} + E_{33}, H_{32} + E_{32}) = QK_n(Q^T(H_{33} + E_{33})Q, Q^T(H_{32} + E_{32}))$$

has full rank. By direct computation, it can be shown that this condition is equivalent to requiring the Hessenberg matrix $Q^T(H_{33} + E_{33})Q$ to be unreduced and the vector $Q^T(H_{32} + E_{32})$ to be nonzero. □

This shows that finding reducing perturbation amounts to finding (small) perturbations that move an controllable system to an uncontrollable system. A number of algorithms have been developed for this purpose, see Section A.4.2. As the perturbations of interest are of very small norm, the heuristic algorithm proposed in [62] (see also Section A.4.2) can be regarded as nearly optimal in the context of the QR algorithm.

In this algorithm, first an orthogonal matrix U is constructed so that $T_{33} = U^T H_{33} U$ is in real Schur form. Then, we partition $T_{33} = \begin{bmatrix} \tilde{T}_{11} & \tilde{T}_{12} \\ 0 & \tilde{T}_{22} \end{bmatrix}$ so that $\tilde{T}_{22}$ is either a real eigenvalue or a 2×2 block containing a pair of complex conjugate eigenvalues of T_{33}. The vector $U^T H_{32} = \begin{bmatrix} \tilde{s}_1 \\ \tilde{s}_2 \end{bmatrix}$ is correspondingly partitioned. Then the matrices $E_{32} = -U \begin{bmatrix} 0 \\ \tilde{s}_2 \end{bmatrix}$ and $E_{33} = 0$ yield a reducing perturbation E of the form (1.59). The Frobenius norm of E is given by $\|\tilde{s}_2\|_2$. We may only make use of this perturbation if $\|\tilde{s}_2\|_2$ is of order $\mathbf{u}\,\|H\|_F$, otherwise the backward stability of the QR algorithm is sacrificed. A more restrictive condition, in the spirit of the neighbor-wise deflation criterion described in Section 1.3.4, is given by

$$\|\tilde{s}_2\|_\infty \leq \mathbf{u} \sqrt{|\det(\tilde{T}_{22})|} \tag{1.60}$$

A variety of other criteria can be found in [62].

If the current ordering of the matrix T_{33} in real Schur form fails to fulfill the criterion (1.60), we may test other possible choices for $\tilde{T}_{22}$ by reordering a different real eigenvalue or complex conjugate pair of eigenvalues to the bottom right corner of T_{33}, see Section 1.7 below. If a reducing perturbation satisfying (1.60) has been found, the described procedure is applied to the unperturbed part $[\tilde{s}_1, \tilde{T}_{11}]$. This process is repeated until no more such reducing perturbation can be found. Eventually, an orthogonal matrix Q is constructed so that

$$(I_{n-w} \oplus Q)^T H (I_{n-w} \oplus Q) = \begin{array}{c} \\ n-w-1 \\ 1 \\ w-d \\ d \end{array} \begin{array}{c} \begin{array}{cccc} n-w-1 & 1 & w-d & d \end{array} \\ \begin{bmatrix} H_{11} & H_{12} & \tilde{H}_{13} & \tilde{H}_{13} \\ H_{21} & H_{22} & \tilde{H}_{23} & \tilde{H}_{24} \\ 0 & s_1 & T_{11} & T_{12} \\ 0 & s_2 & 0 & T_{22} \end{bmatrix} \end{array},$$

where T_{11}, T_{22} are in real Schur form and the entries of the vector s_2 satisfy criteria of the form (1.60). Setting s_2 to zero lets the d eigenvalues contained in T_{22} deflate. The remaining unreduced submatrix can be cheaply returned to Hessenberg form by reducing the $(w-d) \times (w-d+1)$ submatrix $[s_1, T_{11}]$ as described above.

Aggressive early deflation is performed after each multishift QR iteration. It must be combined with the conventional deflation strategy, since aggressive early deflation may fail to detect small subdiagonal elements [61].

Numerical results

The advantage of aggressive early deflation is best illustrated by numerical experiments. Some theoretical justification for the observation that this type of deflation is much more effective than conventional deflation strategies can be found in [62]. We repeated the numerical experiments from Section 1.5.5 by combining aggressive early deflation with the tiny-bulge multishift QR algorithm described therein. Again, this algorithm has been implemented in a Fortran 77 routine, called `ATTQR`. Our choice of parameters for `ATTQR` is based on recommendations given in [62]. The number of shifts in each QR iteration was set to

$$m = \begin{cases} 96, & \text{if } 1000 \leq n < 2000, \\ 120, & \text{if } 2000 \leq n < 2500, \\ 180, & \text{if } 2500 \leq n < 3000, \end{cases}$$

The number of steps a chain of bulges is chased before off-diagonal parts are updated was chosen to be $k = 3/2 \times m - 2$. The size of the deflation window was set to $w = 3/2 \times m$.

The cpu times displayed in Figure 1.14 show that the use of aggressive early deflation results in substantial improvements. The gained savings of computational time (for the case that $n_s = 2$ shifts per bulge are used) range from 14% for the `TOLS1090` matrix up to 74% for the `RDB2048` matrix. Increasing n_s from 2 to 4 leads to some further speedups, except for `TOLS1090` and `OLM2000`. For the other matrices, the gained savings range from 5% up to 24%.

Further Improvements

Having observed the success of aggressive early deflation, it is tempting to ask whether one could use this deflation strategy to deflate eigenvalues at other parts of the matrix. For example, if the shifts do not change very much

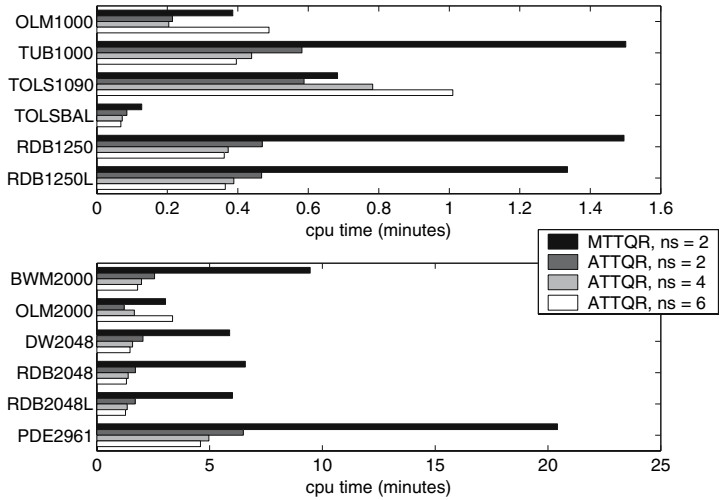


Fig. 1.14. Execution times for `ATTQR`, the tiny-bulge multishift QR algorithm with aggressive early deflation and $n_s \in \{2, 4, 6\}$ shifts per bulge, applied to matrices from the test matrix collection [16].

during a number of QR iterations, then the convergence theory described in Section 1.3.1 predicts some extra linear convergence. This linear convergence does not necessarily manifest itself in deflations in the bottom right corner of the matrix. In fact, the graphs in Figure 1.8 suggest that some deflations may take place at the top left corner. Modifying aggressive early deflation to take care of such deflations is simple; instead of considering the bottom right $w \times (w+1)$ submatrix, one considers the top left $(w+1) \times w$ submatrix of H. However, numerical experiments in [202] demonstrate that the extra speedups gained from this approach are often rather negligible.

A more promising strategy is to search for deflations in the middle of the matrix, which would eventually lead to a recursive QR algorithm. The merits and difficulties of this idea have been investigated in detail by Braman [60].

1.7 Computation of Invariant Subspaces

Let us assume that our favorite variant of the QR algorithm has successfully computed a real Schur decomposition of A:

$$Q^T A Q = T = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1m} \\ 0 & T_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_{m-1,m} \\ 0 & \cdots & 0 & T_{mm} \end{bmatrix}, \tag{1.61}$$

where $T_{ii} \in \mathbb{R}^{n_i \times n_i}$ with $n_i = 1$ if $\lambda(T_{ii}) \subset \mathbb{R}$ and $n_i = 2$ otherwise. For each $j \leq m$, the first $k = \sum_{i=1}^{j} n_i$ columns of Q form an orthonormal basis for an invariant subspace belonging to the eigenvalues $\Lambda_j = \lambda(T_{11}) \cup \cdots \cup \lambda(T_{jj})$. Generally, it is not possible to guarantee a certain order of the diagonal blocks in the real Schur form returned by the QR algorithm. Hence, we need a post-processing step in order to obtain an invariant subspace belonging to a selected cluster of eigenvalues $\Lambda_s \subset \lambda(A)$, which is assumed to be closed under complex conjugation. This section is about computing from a given Schur decomposition (1.61) a reordered Schur decomposition of the form

$$(Q\tilde{Q})^T A(Q\tilde{Q}) = \tilde{T} = \begin{bmatrix} \tilde{T}_{11} & \tilde{T}_{12} & \cdots & \tilde{T}_{1m} \\ 0 & \tilde{T}_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \tilde{T}_{m-1,m} \\ 0 & \cdots & 0 & \tilde{T}_{mm} \end{bmatrix}, \tag{1.62}$$

where $\Lambda_s = \lambda(\tilde{T}_{11}) \cup \cdots \lambda(\tilde{T}_{jj})$ for some $1 \leq j \leq m$.

In the following two subsections we describe the reordering method by Bai and Demmel [18], which is currently implemented in LAPACK. The third subsection describes a more efficient method for eigenvalue reordering, inspired by the tiny-bulge multishift QR algorithm discussed in Section 1.5.5.

1.7.1 Swapping Two Diagonal Blocks

The building block for reordering a given Schur decomposition is the computation of an orthogonal matrix Q so that

$$Q^T \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} Q = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ 0 & \tilde{A}_{22} \end{bmatrix}, \quad \begin{matrix} \lambda(A_{11}) = \lambda(\tilde{A}_{22}), \\ \lambda(A_{22}) = \lambda(\tilde{A}_{11}), \end{matrix} \tag{1.63}$$

where $A_{11} \in \mathbb{R}^{n_1 \times n_1}, A_{22} \in \mathbb{R}^{n_2 \times n_2}$ and $n_1, n_2 \in \{1, 2\}$. This procedure is usually called *swapping* of A_{11} and A_{22}. One should not take this name too literally; it is usually *not* the case that $\tilde{A}_{22}$ is equal to A_{11} or that $\tilde{A}_{11}$ is equal to A_{22}.

Stewart [303] has described an iterative algorithm for producing such a swapping. It first performs an arbitrary QR iteration on $\left[\begin{smallmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{smallmatrix}\right]$ to destroy the reducedness of this matrix. Afterwards, one or more QR iterations with the eigenvalues of A_{11} as shifts are applied, which is likely to quickly converge to a decomposition of the form (1.63). However, numerical examples reported in [18, 110] demonstrate that this algorithm occasionally fails to converge or it produces a block triangular matrix that does not correspond to the intended swapping.

Based on earlier work by other authors [87, 110, 255, 275], Bai and Demmel [18] developed a direct swapping algorithm. Assuming that $\lambda(A_{11}) \cap \lambda(A_{22}) = \emptyset$, it first computes the solution of the Sylvester equation

$$A_{11}X - XA_{22} = \gamma A_{12}, \tag{1.64}$$

where $\gamma \in (0, 1]$ is a scaling factor to prevent from possible overflow in X. This yields the following block diagonal decomposition:

$$\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} = \begin{bmatrix} I_{n_1} & -X \\ 0 & \gamma I_{n_2} \end{bmatrix} \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} I_{n_1} & X/\gamma \\ 0 & I_{n_2}/\gamma \end{bmatrix}.$$

By a QR decomposition, an orthogonal matrix Q is constructed so that

$$Q^T \begin{bmatrix} -X \\ \gamma I_{n_2} \end{bmatrix} = \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad R \in \mathbb{R}^{n_2 \times n_2}.$$

Partition $Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}$ so that $Q_{12} \in \mathbb{R}^{n_1 \times n_1}$, then Q_{12} is invertible and

$$\begin{aligned} Q^T \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} Q &= \begin{bmatrix} \star & R \\ Q_{12}^T & 0 \end{bmatrix} \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} 0 & Q_{12}^{-T} \\ R^{-1} & \star \end{bmatrix} \\ &= \begin{bmatrix} RA_{22}R^{-1} & \star \\ 0 & Q_{12}^T A_{11} Q_{12}^{-T} \end{bmatrix}. \end{aligned}$$

Thus, Q produces the desired swapping.

Let the solution of the Sylvester equation (1.64) be obtained by applying Gaussian elimination with complete pivoting to the Kronecker product formulation of (1.64). Then, in finite-precision arithmetic, the computed factor $\hat{Q}$ satisfies

$$\hat{Q}^T \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \hat{Q} = \begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} \\ \hat{A}_{21} & \hat{A}_{22} \end{bmatrix},$$

where

$$\|\hat{A}_{21}\|_F \leq \frac{\rho \mathbf{u} \, (\|A_{11}\|_F + \|A_{22}\|_F)\|A_{12}\|_F}{[1 + \sigma_{\min}(X)] \operatorname{sep}(A_{11}, A_{22})}$$

and ρ is a small constant of order $\mathcal{O}(1)$ [18]. Hence, a small separation may lead to an error matrix $\hat{A}_{21}$ that cannot be set to zero without sacrificing numerical backward stability. Numerical experiments show that this upper bound is very pessimistic and a small value of $\operatorname{sep}(A_{11}, A_{22})$ only very rarely implies instability. In the LAPACK routine `DLAEXC`, an implementation of the described swapping procedure, the swapping is performed tentatively. If any element in the $(2, 1)$ block entry of $Q^T \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} Q$ exceeds every element of $20\mathbf{u}\,A_{ij}$ in magnitude, then the swapping is rejected. It was argued in [18] that such a rejection may only happen if the eigenvalues of A_{11} and A_{22} are so close that they are numerically indistinguishable.

Bojanczyk and Van Dooren [53] have developed and analyzed an alternative approach to swapping which, instead of solving the Sylvester equation 1.64, relies on the eigenvectors of A.

1.7.2 Reordering

Having the direct swapping algorithm on hand, we can use a bubble sort procedure to reorder a selected set of eigenvalues to the top left corner of a given real Schur form, see Algorithm 8. This algorithm is implemented in the LA-

Algorithm 8 Reordering a real Schur decomposition

Input: A matrix $T \in \mathbb{R}^{n\times n}$ in real Schur form (1.61) with m diagonal blocks, an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ and a simple eigenvalue cluster $\Lambda_s \subset \lambda(T)$, closed under complex conjugation.

Output: A matrix $\tilde{T} \in \mathbb{R}^{n\times n}$ in real Schur form and an orthogonal matrix $\tilde{Q} \in \mathbb{R}^{n\times n}$ so that $\tilde{T} = \tilde{Q}^T T \tilde{Q}$. For some integer j, the set Λ_s is the union of eigenvalues belonging to the j upper-left-most diagonal blocks of $\tilde{T}$. The matrices T and Q are overwritten by $\tilde{T}$ and $Q\tilde{Q}$, respectively.

```
j ← 0
for i ← 1, ..., m do
  if λ(T_ii) ⊂ Λ_s then
    j ← j + 1, select(j) ← i
  end if
end for
top ← 0
for l ← 1, ..., j do
  for i ← select(l), select(l) − 1, ..., top + 1 do
    Swap T_{i−1,i−1} and T_ii by an orthogonal similarity transformation and apply
    this transformation to the rest of the columns and rows of T, and the columns
    of Q.
  end for
  top ← top + 1
end for
```

PACK routine `DTRSEN`, which also provides (estimates of) condition numbers for the eigenvalue cluster Λ_s and the corresponding invariant subspace. If Λ_s contains k eigenvalues, then Algorithm 8 requires $\mathcal{O}(kn^2)$ flops. The exact computational cost depends on the distribution of selected eigenvalues over the block diagonal of T.

1.7.3 Block Algorithm

Each outer loop of Algorithm 8 performs only $\mathcal{O}(\text{select}(l) \cdot n)$ flops while moving $\mathcal{O}(\text{select}(l) \cdot n)$ memory. Even worse, when updating the rows of T in the inner loop, this memory is accessed in a row-by-row fashion. The poor memory reference pattern of Algorithm 8 can be improved using a block algorithm similar to the tiny-bulge multishift QR algorithm, see Section 1.5.5. The basic idea of the block reordering algorithm is best explained by an example.

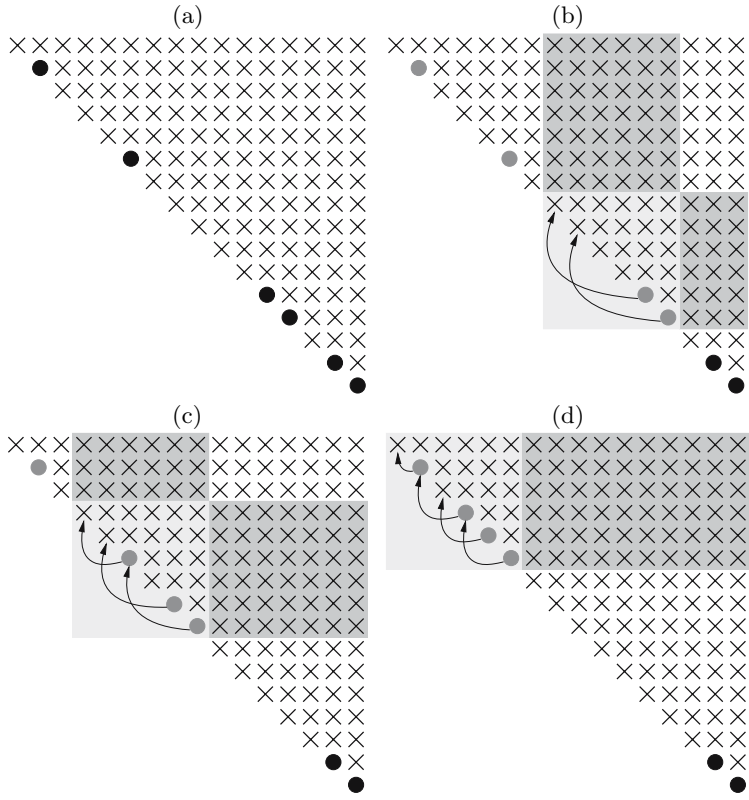


Fig. 1.15.

Let us consider a 16×16 upper triangular matrix T having the eigenvalues at diagonal positions $2, 6, 12, 13, 15$ and 16 selected, see Figure 1.15 (a). We activate the eigenvalues in the $ev = 4$ upper-left-most positions $(2, 6, 12, 13)$, those are the gray disks in Figure 1.15 (b). The active eigenvalues will be reordered to the top in a window-by-window fashion. The first window of order $w = 6$ contains the bottom active eigenvalue in its bottom right corner. This is the light gray area in Figure 1.15 (b). The eigenvalues at positions 12 and 13 are reordered to the top of the window, i.e., positions 8 and 9. The corresponding orthogonal transformations are saved and applied afterwards to the rest of the matrix, this will be discussed in more detail below. The next 6×6 window contains active eigenvalues at positions $6, 8$ and 9, see Figure 1.15 (c). Again, these eigenvalues are reordered to the top of the window. The last window contains all active eigenvalues, which reach their final positions after having been reordered within this window. This process is repeated with the next bunch of at most ev disordered eigenvalues, which in our example are the eigenvalues sitting at positions 15 and 16

We abstain from giving a formal description of the block reordering algorithm, which would be rather technical. Note, however, that the determination of the correct window positions is more complicated in the presence of 2×2 diagonal blocks. For more details, the reader is referred to the report [203] along with the accompanying Fortran 77 implementation `BLKORD` of the delineated block reordering algorithm. `BLKORD` achieves high performance by delaying all orthogonal transformations outside the window, in which the active eigenvalues are reordered. After the reordering within a window has been completed, the transformations are applied to the rest of the matrix, painted dark gray in Figure 1.15. In order to maintain locality of the memory reference pattern, all rows are updated in stripes of n_b columns. It there are sufficiently many transformations, it will be more efficient to accumulate the transformations into a $w \times w$ orthogonal matrix U and use matrix-matrix multiplications to update the rest of the matrix. Assuming that all eigenvalues in the window are real, this matrix U has the following block structure:

$$U = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} = \left[\quad \right], \tag{1.65}$$

i.e., the submatrices $U_{12} \in \mathbb{R}^{(w-m_w)\times(w-m_w)}$ and $U_{21} \in \mathbb{R}^{m_w \times m_w}$ are lower and upper triangular, respectively, where m_w denotes the number of active eigenvalues in the window. If there are pairs of complex conjugate eigenvalues then the submatrices of U have only banded structure, which is more difficult to exploit using level 3 BLAS operations.

Numerical Experiments

For the purpose of demonstrating the performance improvements to be expected from using the described block algorithm, we applied `BLKORD` to an upper triangular matrix $T \in \mathbb{R}^{n\times n}$, $n \in \{500, 1000, 1500\}$, generated by the LAPACK routine `DLATME` such that all eigenvalues have moderate condition numbers. The portion of selected eigenvalues was $d \in \{5\%, 25\%, 50\%\}$. To demonstrate that the performance improvements are to some extent independent of the distribution of eigenvalues over the diagonal of T, we considered two different distributions. In the "random" distribution, each eigenvalue is selected with probability d. In the "bottom" distribution, the selected eigenvalues are located in the $dn \times dn$ bottom right submatrix of T.

The parameter ev, the number of active eigenvalues in each block reordering step of `BLKORD`, was chosen to be the optimal value in $\{24, 32, 48, 64, 80\}$. Note that the timings obtained with the (possibly suboptimal) value $ev = 48$ differ at most by 10% from the timings obtained with the optimal value for the matrices under consideration. The window size w was set to $5/2 \times ev$ for no particular good reason, except that this choice matches the window size in the tiny-bulge multishift QR algorithm rather well. The employed orthogonal

transformations were accumulated in a matrix U whenever more than half of the active eigenvalues were located in the lower half of the window. The potential block triangular structure (1.65) of U was not exploited.

			Update of T		Update of $T + Q$	
n	sel.	distr.	DTRSEN	BLKORD	DTRSEN	BLKORD
500	5%	random	0.19	0.05	0.25	0.08
500	5%	bottom	0.24	0.07	0.34	0.13
500	25%	random	0.55	0.16	0.75	0.26
500	25%	bottom	0.86	0.29	1.24	0.50
500	50%	random	0.51	0.20	0.76	0.34
500	50%	bottom	1.06	0.38	1.54	0.63
1000	5%	random	2.23	0.34	2.87	0.62
1000	5%	bottom	2.98	0.50	4.03	0.88
1000	25%	random	6.51	0.95	8.40	1.71
1000	25%	bottom	11.65	1.94	15.83	3.39
1000	50%	random	7.60	1.31	10.08	2.34
1000	50%	bottom	15.56	2.53	21.23	4.49
1500	5%	random	7.92	1.00	9.46	1.77
1500	5%	bottom	11.36	1.61	14.21	2.91
1500	25%	random	25.02	3.01	30.53	5.42
1500	25%	bottom	45.12	6.09	56.64	11.04
1500	50%	random	28.11	4.02	35.93	7.38
1500	50%	bottom	60.47	7.98	75.79	14.64

Table 1.2. Performance results in seconds for unblocked (`DTRSEN`) and blocked (`BLKORD`) reordering of an $n \times n$ matrix in Schur form.

Table 1.2 demonstrates that `BLKORD` performs much better than the LAPACK routine `DTRSEN`; often it requires less than 25% of the time needed by `DTRSEN`. To test the numerical reliability of `BLKORD`, we measured the orthogonality of Q, $\|Q^TQ - I\|_F$, as well as the residual $\|Q^TTQ - \tilde{T}\|_F/\|T\|_F$ and found all values satisfactorily close to $\mathbf{u}$.

1.8 Case Study: Solution of an Optimal Control Problem

In this concluding section, we apply the described algorithms to the optimal control problem for second-order models using Laub's Schur method [211], see also [134, 212]. In this type of control problems, the dynamical system consists of a state equation given by a second-order differential equation

$$M\ddot{z}(t) + L\dot{z}(t) + Kz(t) = Su(t), \quad z(0) = z_0,\ \dot{z}(0) = z_1, \tag{1.66}$$

and an associated output equation

$$y(t) = Nz(t) + P\dot{z}(t), \tag{1.67}$$

where $z(t) \in \mathbb{R}^\ell$, $y(t) \in \mathbb{R}^p$, $u(t) \in \mathbb{R}^m$, $M, L, K \in \mathbb{R}^{\ell\times\ell}$, $S \in \mathbb{R}^{\ell\times m}$, and $N, P \in \mathbb{R}^{p\times\ell}$. Often, M and K are symmetric where M is positive definite, K is positive semidefinite, and L is the sum of a symmetric positive semidefinite and a skew-symmetric matrix. Usually, M is called the *mass matrix*, L is the *Rayleigh matrix* representing damping (the symmetric part) and gyroscopic (the skew-symmetric part) forces, and K is the *stiffness matrix*. Second-order models are often used to model mechanical systems such as large flexible space structures.

Assuming that M is invertible, a first-order description of (1.66)–(1.67) can be obtained by introducing the state vector

$$x(t) = \begin{bmatrix} z(t) \\ \dot{z}(t) \end{bmatrix},$$

which yields a system of the form

$$\dot{x}(t) = \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}L \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ M^{-1}S \end{bmatrix} u(t), \quad x(0) = x_0, \tag{1.68a}$$

$$y(t) = \begin{bmatrix} N & P \end{bmatrix} x(t), \tag{1.68b}$$

where $x_0 = [z_0^T, z_1^T]^T$. This is a standard linear continuous-time system of the form (A.1).

Now, consider the linear quadratic-optimal control problem:

$$\textit{Minimize} \quad J(u(\cdot)) = \frac{1}{2}\int_0^\infty \left(y(t)^T y(t) + u(t)^T u(t)\right)\, \mathrm{d}t \quad \textit{subject to (1.68).}$$

Under some mild assumptions, the solution of this problem is given by a linear feedback law of the form $u(t) = -[0, (M^{-1}S)^T]X_\star x(t)$, see Section A.3. The feedback matrix $X_\star$ can be obtained from the invariant subspace belonging to the $2l$ eigenvalues with negative real part of the $4l \times 4l$ Hamiltonian[5] matrix:

$$H = \left[\begin{array}{cc|cc} 0 & I & 0 & 0 \\ -M^{-1}K & -M^{-1}L & 0 & -(M^{-1}S)(M^{-1}S)^T \\ \hline -N^TN & -N^TP & 0 & (M^{-1}K)^T \\ -P^TN & P^TP & -I & (M^{-1}L)^T \end{array}\right]. \tag{1.69}$$

To compute this so called stable invariant subspace we can proceed in three steps [211]:

1. Compute an orthogonal matrix Q_1 so that $Q_1^T H Q_1$ has Hessenberg form.

[5] For more on Hamiltonian matrices, the reader is referred to Section 4.5.

2. Apply the QR algorithm to compute an orthogonal matrix Q_2 so that

$$T = (Q_1 Q_2)^T H (Q_1 Q_2)$$

has real Schur form.

3. Reorder the diagonal blocks of T by an orthogonal transformation Q_3 so that the first $2l$ diagonal blocks of $Q_3^T T Q_3$ contain eigenvalues having negative real part.

The stable invariant subspace is spanned by the first $2l$ columns of the orthogonal matrix $U = Q_1 Q_2 Q_3$. In the previous sections, we have described and developed various algorithms to improve the performance of Steps 2 and 3. The following example illustrates these improvements.

Example 1.34 ([158, 1, 204]). Let us consider a model of a string consisting of coupled springs, dashpots, and masses as shown in Figure 1.16. The inputs are two forces, one acting on the left end of the string, the other one on the right end. For this problem, the matrices in (1.68) are

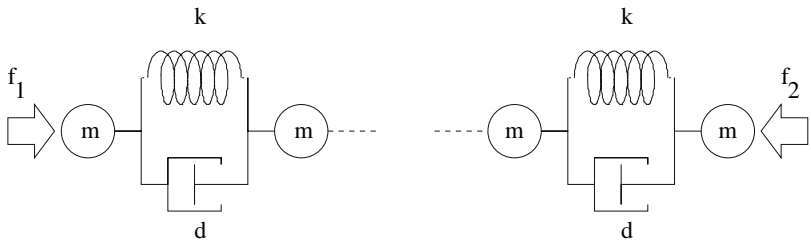


Fig. 1.16. Coupled springs experiment ($k \sim \kappa$, $m \sim \mu$, $d \sim \delta$).

$$M = \mu I_\ell, \qquad L = \delta I_\ell, \qquad N = P = I_\ell,$$

$$K = \kappa \begin{bmatrix} 1 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & \dots \\ 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & \dots & 0 & -1 & 1 \end{bmatrix}, \qquad S = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ \vdots & \vdots \\ 0 & 0 \\ 0 & -1 \end{bmatrix},$$

for some parameters $\delta, \kappa, \mu \in \mathbb{R}$. We have chosen $\ell = 500$, $\delta = 4$, $\kappa = 1$ and $\mu = 4$ for the numerical experiments.

Three different combinations of algorithms were used to perform the three steps listed above. The first combination uses solely LAPACK routines: `DGEHRD` and `DORGHR` for Step 1, `DHSEQR` for Step 2, `DTRSEN` for Step 3. The second combination replaces `DHSEQR` by the tiny-bulge multishift QR algorithm

(ATTQR) with $n_s = 2$ shifts per bulge and aggressive early deflation, see Sections 1.5.5 and 1.6. The third combination uses ATTQR with $n_s = 6$, described in Section 1.5.5, and the block reordering routine BLKORD, see Section 1.7.3. All routine parameters had the same values as in the numerical experiments of Sections 1.6 and 1.7.3.

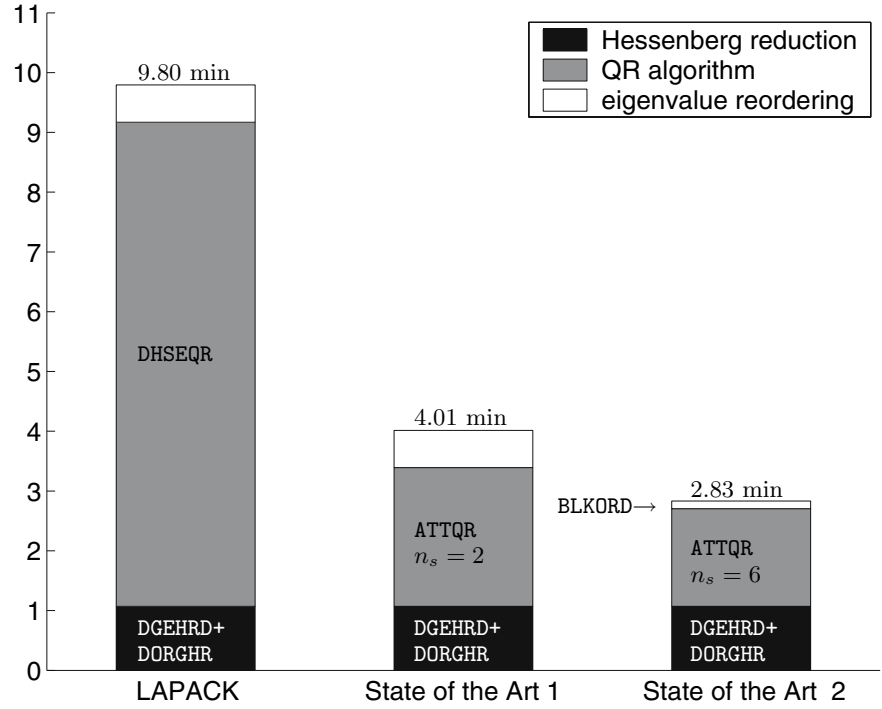


Fig. 1.17. Performance results (in minutes) for the three combinations of algorithms described in Example 1.34.

The obtained performance results are displayed in Figure 1.17. It can be seen that the second combination requires 59% less computational time than LAPACK. The third combination is even faster; it outperforms the first one by 71% and the second one by 29.4%. It is a remarkable fact that the portion of computational time needed by the Hessenberg reduction step increases from 11% to 38%.

2

The QZ Algorithm

There was a young fellow from Trinity
Who tried $\sqrt{\infty}$
But the number of digits
Gave him such figets
That he gave up Math for Divinity
—George Gamow *(quoted in [248])*

The QZ algorithm is a numerically backward stable method for computing generalized eigenvalues and deflating subspaces of regular matrix pairs. It goes back to Moler and Stewart in 1973 [248] and has undergone only a few modifications during the next 25 years, notably through works by Ward [345, 346], Kaufman [189], Dackland and Kågström [96].

The QZ algorithm applied to a matrix pair (A, B) is formally equivalent to applying the QR algorithm to AB^{-1} and $B^{-1}A$ at the same time, provided that B is nonsingular. In finite-precision arithmetic, however, the inversion of the potentially ill-conditioned matrix B may lead to a severe loss of accuracy in the computed eigenvalues and should therefore be avoided. This is the major source of justification for the importance of the QZ algorithm.

This chapter is organized as follows. In Section 2.1, we briefly recall the definition of generalized eigenvalues and deflating subspaces as well as their relation to the generalized Schur decomposition. A more detailed treatment of this subject can be found in, e.g., [308]. Existing perturbation results for generalized eigenvalues and deflating subspaces are summarized in Section 2.2. The explicit shifted QZ iteration, which forms the basis of the QZ algorithm, is introduced in the beginning of Section 2.3. The remainder of that section is concerned with the three ingredients that make the implicit shifted QZ algorithm work: reduction to Hessenberg-triangular form, bulge chasing, deflation of finite and infinite eigenvalues. Particular emphasis is placed on the use of Householder matrices in the bulge chasing process, see Section 2.3.3. Balancing matrix pairs, the subject of Section 2.4, is a subtle issue. We describe two known approaches to balancing and demonstrate their fallibilities. In Section 2.5.1, block algorithms for reducing a matrix pair to Hessenberg-triangular form are investigated, largely based on an algorithm by Dackland and Kågström [96]. Sections 2.5.2 and 2.5.3 deal with the transmission of shifts during an implicit shifted QZ iteration and the behavior of infinite eigenvalues. In Sections 2.5.4 and 2.6, we adapt techniques described in the previous chapter to obtain a tiny-bulge multishift QZ algorithm with aggressive early

deflation. Finally, Section 2.7 is concerned with the computation of selected deflating subspaces from a generalized Schur decomposition.

2.1 The Generalized Eigenvalue Problem

The *(generalized) eigenvalues* of a *matrix pair* $(A, B) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$ are the roots (α, β) of the bivariate polynomial $\det(\beta A - \alpha B)$. Throughout the whole chapter we assume that this polynomial is not identically zero, in which case the corresponding matrix pair is called *regular*. The set of all eigenvalues (α, β) is denoted by $\lambda(A, B)$. By introducing the equivalence classes

$$\langle \alpha, \beta \rangle = \{(\tau\alpha, \tau\beta) : \tau \in \mathbb{C}\backslash\{0\}\}$$

we identify eigenvalues which only differ by a common multiple.

If B is nonsingular then (α, β) is a generalized eigenvalue of (A, B) if and only if α/β is an eigenvalue of AB^{-1}. Hence, in principal we can compute generalized eigenvalues by applying the QR algorithm to the explicitly formed matrix AB^{-1}. As explained in the introduction, the drawback of such an approach is that an ill-conditioned B would unnecessarily spoil the accuracy of the computed eigenvalues. Nevertheless, this connection should be kept in mind; it relates much of the material presented in this chapter to the previous chapter.

A nonzero vector $x \in \mathbb{C}^n$ is called a *(right generalized) eigenvector* of (A, B) if $\beta Ax = \alpha Bx$ for some $(\alpha, \beta) \in \lambda(A, B)$. Correspondingly, a nonzero vector $z \in \mathbb{C}^n$ satisfying $\beta z^H A = \alpha z^H B$ is called a *left (generalized) eigenvector*. Note that Ax and Bx lie in the same direction if x is an eigenvector. A generalization of this idea suggests to call a k-dimensional subspace $\mathcal{X}$ a *(right) deflating subspace* of (A, B) if $A\mathcal{X}$ and $B\mathcal{X}$ are contained in a subspace $\mathcal{Y}$ of dimension k. The regularity of (A, B) implies that such a subspace $\mathcal{Y}$ is uniquely defined; we call $\mathcal{Y}$ a *left deflating subspace* and $(\mathcal{X}, \mathcal{Y})$ a *pair of deflating subspaces*. It is important to remark that $\mathcal{Y}$, despite its name, is generally *not* spanned by left eigenvectors. If the columns of X and Y form bases for $\mathcal{X}$ and $\mathcal{Y}$, respectively, then there exists a uniquely defined matrix pair (A_{11}, B_{11}) satisfying

$$AX = YA_{11}, \quad BX = YB_{11}.$$

This matrix pair is called the *representation of* (A, B) *with respect to* (X, Y). For brevity, we will make use of the conventions $(A, B)\,X \equiv (AX, BX)$ and $Y\,(A_{11}, B_{11}) \equiv (YA_{11}, YB_{11})$. The following example shows how deflating subspaces belonging to complex conjugate pairs of eigenvalues can be constructed.

Example 2.1. Let $(\alpha, \beta) = (\alpha_1 + \mathrm{i}\alpha_2, \beta)$ with $\alpha_1 \in \mathbb{R}$ and $\alpha_2, \beta \in \mathbb{R}\backslash\{0\}$ be an eigenvalue of the regular matrix pair $(A, B) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$. If $x = x_1 + \mathrm{i}x_2$ is an eigenvector belonging to (α, β) with $x_1, x_2 \in \mathbb{R}^n$, then $\beta Ax = \alpha Bx$ implies

$$\beta A x_1 = \alpha_1 B x_1 - \alpha_2 B x_2, \quad \beta A x_2 = \alpha_2 B x_1 + \alpha_1 B x_2.$$

The linear independence of x_1, x_2 as well as the linear independence of

$$y_1 = \frac{1}{\beta} B x_1, \quad y_2 = \frac{1}{\beta} B x_2$$

follow from $\alpha_2 \neq 0$ and $\beta \neq 0$. Hence, $(\mathrm{span}\{x_1, x_2\}, \mathrm{span}\{y_1, y_2\})$ is a pair of two-dimensional deflating subspaces with the real representation

$$(A, B) \begin{bmatrix} x_1, x_2 \end{bmatrix} = \begin{bmatrix} y_1, y_2 \end{bmatrix} \left(\begin{bmatrix} \alpha_1 & \alpha_2 \\ -\alpha_2 & \alpha_1 \end{bmatrix}, \begin{bmatrix} \beta & 0 \\ 0 & \beta \end{bmatrix} \right).$$

$\Diamond$

Not surprisingly, deflating subspaces admit the deflation of a generalized eigenvalue problem into two smaller subproblems, just as invariant subspaces can be used to deflate standard eigenvalue problems. Let $(\mathcal{X}, \mathcal{Y})$ be a pair of deflating subspaces and let the columns of $X, X_\perp, Y, Y_\perp$ form orthonormal bases for $\mathcal{X}, \mathcal{X}^\perp, \mathcal{Y}, \mathcal{Y}^\perp$, respectively. Then $U = [Y, Y_\perp]$ and $V = [X, X_\perp]$ are unitary matrices with

$$U^H (A, B) V = \left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} \right) \tag{2.1}$$

and $\lambda(A, B) = \lambda(A_{11}, B_{11}) \cup \lambda(A_{22}, B_{22})$. This decomposition is called a *generalized block Schur decomposition* of the matrix pair (A, B). We say that any matrix pair with the block triangular structure displayed in (2.1) is in *generalized block Schur form*. A block triangular matrix pair can be subsequently reduced to a triangular matrix pair, giving rise to the so called *generalized Schur decomposition*. This decomposition will be complex unless every eigenvalue belongs to some class $\langle \alpha, \beta \rangle$ with $\alpha, \beta \in \mathbb{R}$. However, similar to the real Schur decomposition of a single matrix, realness can be preserved by allowing two-by-two diagonal blocks in one of the matrices.

Theorem 2.2 (Generalized real Schur decomposition [300]). *Consider a matrix pair $(A, B) \in \mathbb{R}^{n \times n} \times \mathbb{R}^{n \times n}$. Then there exist orthogonal matrices Q and Z so that $Q^T (A, B) Z = (S, T)$, where S is in real Schur form (see Theorem 1.2) while T is upper triangular.*

The purpose of the QZ algorithm is to compute such a generalized real Schur decomposition. This provides almost everything needed to solve the generalized eigenvalue problem. Eigenvalues can be easily computed from the diagonal blocks of T and S, and the leading k columns of Z and Q span pairs of deflating subspaces under the assumption that the $(k+1, k)$ entry of T vanishes. A reordering of the diagonal blocks of S and T can be used to compute other deflating subspaces, see Section 2.7.

2.2 Perturbation Analysis

In this section we investigate the influence of a perturbation (E, F) on the generalized eigenvalues and deflating subspaces of (A, B). The exposition is briefer than for the standard eigenvalue problem as many of the results can be derived by techniques similar to those used in Section 2.2.

2.2.1 Spectral Projectors and Dif

Spectral projectors and the separation of two matrices have played an important role in deriving perturbation results for the standard eigenvalue problem. In the following, we define similar quantities playing this role for the generalized eigenvalue problem. Throughout this section we consider a generalized block Schur decomposition of the form

$$U^H (A, B) V = \left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} \right) \tag{2.2}$$

We partition the unitary matrices as $U = [Y, Y_\perp]$ and $V = [X, X_\perp]$ so that $(\mathcal{X}, \mathcal{Y}) = (\mathrm{span}(X), \mathrm{span}(Y))$ is a pair of deflating subspaces corresponding to the representation $(A_{11}, B_{11}) \in \mathbb{C}^{k\times k} \times \mathbb{C}^{k\times k}$.

The *right* and *left spectral projectors* P_r and P_l belonging to $\lambda(A_{11}, B_{11})$ are defined as

$$P_r = V \begin{bmatrix} I_k & R_r \\ 0 & 0 \end{bmatrix} V^H, \quad P_l = U \begin{bmatrix} I_k & -R_l \\ 0 & 0 \end{bmatrix} U^H, \tag{2.3}$$

where the matrix pair (R_r, R_l) satisfies the following system of matrix equations

$$\begin{aligned} A_{11} R_r - R_l A_{22} &= -A_{12}, \\ B_{11} R_r - R_l B_{22} &= -B_{12}. \end{aligned} \tag{2.4}$$

This system is called a *generalized Sylvester equation* and the corresponding *generalized Sylvester operator* is the linear matrix operator

$$\mathbf{T}_u : (R_r, R_l) \to (A_{11} R_r - R_l A_{22}, B_{11} R_r - R_l B_{22}). \tag{2.5}$$

The invertibility of $\mathbf{T}_u$ is necessary and sufficient for the system of matrix equations (2.4) to be uniquely solvable.

Lemma 2.3 ([300]). *Let (A_{11}, B_{11}) and (A_{22}, B_{22}) be regular matrix pairs and let $\mathbf{T}_u$ be defined as in (2.5). Then $\mathbf{T}_u$ is invertible if and only if*

$$\lambda(A_{11}, B_{11}) \cap \lambda(A_{22}, B_{22}) = \emptyset.$$

Since the eigenvalues of $(A_{11}, B_{11}) = Y^H (A, B) X$ and $(A_{22}, B_{22}) = Y_\perp^H (A, B) X_\perp$ do not depend on a particular choice of bases for $\mathcal{X}$ and $\mathcal{Y}$, the invertibility of $\mathbf{T}_u$ may be assigned as an intrinsic property of deflating subspaces.

Definition 2.4. *Let $(\mathcal{X}, \mathcal{Y})$ be a pair of deflating subspaces for the regular matrix pair (A, B), and let the columns of $X, X_\perp, Y, Y_\perp$ form orthonormal bases for $\mathcal{X}, \mathcal{X}^\perp, \mathcal{Y}, \mathcal{Y}^\perp$, respectively. Then $(\mathcal{X}, \mathcal{Y})$ is called* simple *if*

$$\lambda(Y^H AX, Y^H BX) \cap \lambda(Y_\perp^H AX_\perp, Y_\perp^H BX_\perp) = \emptyset.$$

The *separation of two matrix pairs* (A_{11}, B_{11}) and (A_{22}, B_{22}) is defined as the smallest singular value of $\mathbf{T}_u$:

$$\mathrm{dif}_u[(A_{11}, B_{11}), (A_{22}, B_{22})] := \min_{\substack{(R_r, R_l) \\ \neq (0,0)}} \frac{\|\mathbf{T}_u(R_r, R_l)\|_F}{\|(R_r, R_l)\|_F}, \tag{2.6}$$

where we let $\|(R_r, R_l)\|_F = \left\| \begin{bmatrix} R_r \\ R_l \end{bmatrix} \right\|_F$. While the ordering of arguments does not play a role for the separation of two matrices ($\mathrm{sep}(A_{11}, A_{22}) = \mathrm{sep}(A_{22}, A_{11})$), it generally affects the separation of two matrix pairs. We therefore introduce the quantity

$$\mathrm{dif}_l[(A_{11}, B_{11}), (A_{22}, B_{22})] := \mathrm{dif}_u[(A_{22}, B_{22}), (A_{11}, B_{11})].$$

The associated generalized Sylvester operator is given by

$$\mathbf{T}_l : (Q_r, Q_l) \to (A_{22}Q_r - Q_l A_{11}, B_{22}Q_r - Q_l B_{11}). \tag{2.7}$$

The following example reveals that dif_l and dif_u can differ significantly.

Example 2.5. Let $A_{11} = \begin{bmatrix} 10^5 & 0 \\ 0 & 10^{-5} \end{bmatrix}$, $A_{22} = 1$, $B_{11} = \begin{bmatrix} 10^5 & 10^5 \\ 0 & 10^{-5} \end{bmatrix}$ and $B_{22} = 0$. Then

$$\mathrm{dif}_u[(A_{11}, B_{11}), (A_{22}, B_{22})] = \left\| \begin{bmatrix} A_{11} & -I_2 \\ B_{11} & 0 \end{bmatrix}^{-1} \right\|_2 = 10^{10},$$

while

$$\mathrm{dif}_l[(A_{11}, B_{11}), (A_{22}, B_{22})] = \left\| \begin{bmatrix} -I_2 & A_{11}^T \\ 0 & B_{11}^T \end{bmatrix}^{-1} \right\|_2 = \sqrt{2} \times 10^5.$$

$\Diamond$

Using a Kronecker product formulation, the matrix operator $\mathbf{T}_u$ can be represented as

$$\mathrm{vec}(\mathbf{T}_u(R_r, R_l)) = K_{\mathbf{T}_u} \begin{bmatrix} \mathrm{vec}\,(R_r) \\ \mathrm{vec}\,(R_l) \end{bmatrix},$$

with the $2k(n-k) \times 2k(n-k)$ matrix

$$K_{\mathbf{T}_u} = \begin{bmatrix} I_{n-k} \otimes A_{11} & -A_{22}^T \otimes I_k \\ I_{n-k} \otimes B_{11} & -B_{22}^T \otimes I_k \end{bmatrix}.$$

The definition of dif_u implies

$$\mathrm{dif}_u[(A_{11}, B_{11}), (A_{22}, B_{22})] = \sigma_{\min}(K_{\mathbf{T}_u}).$$

2.2.2 Local Perturbation Bounds

The perturbation theory for generalized eigenvalue problems is comprehensively treated in Chapter VI of the book by Stewart and Sun [308], largely based on works by Stewart [300, 302]. For recent developments in this area the reader is referred to the comprehensive report by Sun [317] and the references therein.

We start with the perturbation expansion of a generalized eigenvalue.

Theorem 2.6 ([223, 317]). *Let the regular matrix pair (A, B) have a simple eigenvalue (α, β) with right and left eigenvectors x and z, respectively, normalized such that $z^H Ax = \alpha$ and $z^H Bx = \beta$. Let $(E, F) \in \mathcal{B}(0)$ be a perturbation of (A, B), where $\mathcal{B}(0) \subset \mathbb{C}^{n\times n} \times \mathbb{C}^{n\times n}$ is a sufficiently small open neighborhood of the origin. Then there exist analytic functions $f_\alpha : \mathcal{B}(0) \to \mathbb{C}$, $f_\beta : \mathcal{B}(0) \to \mathbb{C}$ so that $(\alpha, \beta) = (f_\alpha(0), f_\beta(0))$ and $(\hat\alpha, \hat\beta) := (f_\alpha(E), f_\beta(F))$ is a generalized eigenvalue of the perturbed matrix pair $(A+E, B+F)$. Moreover,*

$$\begin{aligned} \hat\alpha &= \alpha + z^H Ex + O(\|(E, F)\|^2), \\ \hat\beta &= \beta + z^H Fx + O(\|(E, F)\|^2). \end{aligned} \tag{2.8}$$

The eigenvalue (α, β) is a representative of the class $\langle\alpha, \beta\rangle$, which can be interpreted as a one-dimensional subspace spanned by the vector $[\alpha, \beta]$. This suggests using one of the metrics for vector subspaces discussed on page 13 in order to obtain a meaningful measure of the distance between two generalized eigenvalues (α, β) and $(\hat\alpha, \hat\beta)$. In particular, the gap metric in the 2-norm yields the following distance measure [302].

Definition 2.7. *The* chordal distance *between $\langle\alpha, \beta\rangle$ and $\langle\hat\alpha, \hat\beta\rangle$ is defined as*

$$\chi(\langle\alpha, \beta\rangle, \langle\hat\alpha, \hat\beta\rangle) := \frac{|\alpha\hat\beta - \beta\hat\alpha|}{\sqrt{|\alpha|^2 + |\beta|^2}\sqrt{|\hat\alpha|^2 + |\hat\beta|^2}}.$$

Inserting (2.8) into this definition we obtain, after some algebraic manipulations, the perturbation bound

$$\chi(\langle\alpha, \beta\rangle, \langle\hat\alpha, \hat\beta\rangle) \le \frac{\|x\|_2 \, \|z\|_2}{\sqrt{|\alpha|^2 + |\beta|^2}} \, \|(E, F)\|_2 + O(\|(E, F)\|^2).$$

The following theorem is a variation of a result by Sun [315] on the perturbation expansion for pairs of deflating subspaces.

Theorem 2.8. *Let the regular matrix pair (A, B) have a generalized block Schur decomposition of the form (2.2) and partition $U = [Y, Y_\perp]$, $V = [X, X_\perp]$, so that $(\mathcal{X}, \mathcal{Y}) = (\mathrm{span}(X), \mathrm{span}(Y))$ is a pair of deflating subspaces. Assume that $(\mathcal{X}, \mathcal{Y})$ is simple and let $(E, F) \in \mathcal{B}(0)$ be a perturbation of (A, B), where $\mathcal{B}(0) \subset \mathbb{C}^{n\times n} \times \mathbb{C}^{n\times n}$ is a sufficiently small open neighborhood of the origin. Then there exists an analytic function*

$$f_{(X,Y)} : \mathcal{B}(0) \to \mathbb{C}^{n\times k} \times \mathbb{C}^{n\times k}$$

so that $(X,Y) = f_{(X,Y)}(0)$*, and the columns of* $(\hat{X},\hat{Y}) = f_{(X,Y)}(E,F)$ *span a pair of deflating subspaces of* $(A+E, B+F)$*. Moreover,* $X^H(\hat{X}-X) = Y^H(\hat{Y}-Y) = 0$*, and we have the expansion*

$$(\hat{X},\hat{Y}) = (X,Y) + (X_\perp Q_r, Y_\perp Q_l^\star) + \mathcal{O}(\|[E,F]\|^2) \tag{2.9}$$

with $(Q_r,Q_l) = \mathbf{T}_l^{-1}(Y_\perp^H EX, Y_\perp^H FX)$ *and the generalized Sylvester operator* $\mathbf{T}_l$ *as in (2.7).*

Proof. We can apply essentially the same technique that has been used to derive local perturbation bounds for the standard eigenvalue problem, in particular in the proof of Theorem 1.9. If

$$f(E,F,\hat{X},\hat{Y},\hat{A}_{11},\hat{B}_{11}) := \begin{bmatrix} A\hat{X} - \hat{Y}\hat{A}_{11} \\ B\hat{X} - \hat{Y}\hat{B}_{11} \\ X^H(\hat{X}-X) \\ Y^H(\hat{Y}-Y) \end{bmatrix} = 0,$$

then $(\mathrm{span}(\hat{X}), \mathrm{span}(\hat{Y}))$ is a pair of deflating subspaces belonging to $\lambda(\hat{A}_{11}, \hat{B}_{11})$. The Jacobian of f with respect to $(\hat{X},\hat{Y},\hat{A}_{11},\hat{B}_{11})$ at $(0,0,X,Y,A_{11},B_{11})$ is a linear matrix operator having the block representation

$$J = \frac{\partial f}{\partial(\hat{X},\hat{Y},\hat{A}_{11},\hat{B}_{11})}\Big|_{(0,0,X,Y,A_{11},B_{11})} = \begin{bmatrix} & \tilde{\mathbf{T}} & -Y & 0 \\ & & 0 & -Y \\ X^H & 0 & 0 & 0 \\ 0 & Y^H & 0 & 0 \end{bmatrix}$$

with the linear matrix operator $\tilde{\mathbf{T}} : (Z_r, Z_l) :\to (AZ_r - Z_l A_{11}, BZ_r - Z_l B_{11})$. Since $(\mathcal{X},\mathcal{Y})$ is simple, the generalized Sylvester operator $\mathbf{T}_l$ is invertible which in turn implies the invertibility of J. The latter conclusion is shown by verifying that $J^{-1}\circ J = J\circ J^{-1}$ is the identity map for

$$J^{-1} = \begin{bmatrix} & \mathbf{S} & X & 0 \\ & & 0 & Y \\ C_{11} & C_{12} & A_{11} & -A_{11} \\ C_{21} & C_{22} & B_{11} & -B_{11} \end{bmatrix},$$

where

$$\mathbf{S} : (S_1,S_2) \mapsto (X_\perp Q_r, Y_\perp Q_l^\star), \quad (Q_r,Q_l) = \mathbf{T}_l^{-1}(Y_\perp^H S_1, Y_\perp^H S_2).$$

The expressions for the blocks C_{ij} are given by

$$\begin{aligned} (C_{11},C_{12}) &= (\mathbf{T}_l^\star)^{-1}(A_{12},0)\cdot Y_\perp^H - (0,Y^H), \\ (C_{21},C_{22}) &= (\mathbf{T}_l^\star)^{-1}(B_{12},0)\cdot Y_\perp^H - (Y^H,0), \end{aligned}$$

with the generalized Sylvester operator

$$\mathbf{T}_l^\star : (Q_r, Q_l) \to (Q_r A_{22} + Q_l B_{22}, -A_{11} Q_r - B_{11} Q_l),$$

which is, under the given assumption, invertible. The proof is concluded by applying the implicit function theorem. □

Although the proof of Theorem 2.8 also provides perturbation expansions for the representation of $(A+E, B+F)$ with respect to $(\hat{X}, \hat{Y})$, it is difficult to interpret these expansions. See [182] for a discussion on meaningful condition numbers for clusters of generalized eigenvalues.

Corollary 2.9. *Under the assumptions of Theorem 2.8, let*

$$(\hat{\mathcal{X}}, \hat{\mathcal{Y}}) = (\mathrm{span}(\hat{X}), \mathrm{span}(\hat{Y})).$$

Then

$$\|\Theta(\mathcal{X}, \hat{\mathcal{X}})\|_F \leq c(\mathcal{X}, \mathcal{Y}) \, \|(E, F)\|_F + O(\|(E, F)\|^2), \tag{2.10a}$$
$$\|\Theta(\mathcal{Y}, \hat{\mathcal{Y}})\|_F \leq c(\mathcal{X}, \mathcal{Y}) \, \|(E, F)\|_F + O(\|(E, F)\|^2), \tag{2.10b}$$

where $c(\mathcal{X}, \mathcal{Y}) = 1/\operatorname{dif}_l[(A_{11}, B_{11}), (A_{22}, B_{22})]$.

Proof. The proof of this result is virtually identical to the proof of Corollary 1.13, which shows a similar bound for a perturbed invariant subspace. □

It may happen that $\mathcal{X}$ and $\mathcal{Y}$ are not equally sensitive to perturbations. In this case, Corollary 2.9 overestimates the sensitivity of one of the deflating subspaces, see Example 2.10 below. Separating the influence of the operator $\mathbf{T}_l$ on $\mathcal{X}$ and $\mathcal{Y}$ resolves this difficulty. Let

$$c_r^{-1}(\mathcal{X}) := \min_{Q_r \neq 0} \frac{\|\mathbf{T}_l(Q_r, 0)\|_F}{\|Q_r\|_F}, \quad c_l^{-1}(\mathcal{Y}) := \min_{Q_l \neq 0} \frac{\|\mathbf{T}_l(0, Q_l)\|_F}{\|Q_l\|_F}, \tag{2.11}$$

then we can replace $c(\mathcal{X}, \mathcal{Y})$ by the potentially smaller numbers $c_r(\mathcal{X})$ and $c_l(\mathcal{Y})$ in (2.10a) and (2.10b), respectively [315, 317].

Example 2.10 ([317, Ex. 4.2.10]). Let the regular matrix pair $(A, B) \in \mathbb{R}^{4\times4} \times \mathbb{R}^{4\times4}$ be in generalized block Schur form with

$$A_{11} = 10^{-5} \times I_2, \quad B_{11} = \begin{bmatrix} 10^{-4} & 0 \\ 10^{-4} & 10^{-4} \end{bmatrix}, \quad A_{22} = B_{22} = I_2.$$

For the pair of deflating subspaces $(\mathcal{X}, \mathcal{Y}) = (\mathrm{span}([I, 0]^T), \mathrm{span}([I, 0]^T))$ we obtain

$$c(\mathcal{X}, \mathcal{Y}) \approx 2.67 \times 10^4, \quad c_r(\mathcal{X}) = 1.89, \quad c_l(\mathcal{Y}) = 2.67 \times 10^4,$$

showing that $c(\mathcal{X}, \mathcal{Y})$ severely overestimates the sensitivity of the deflating subspace $\mathcal{X}$ alone. ◊

Another point to emphasize is that the derived perturbation bounds may suffer from the fact that the perturbation matrices E and F are equally weighted. This is hard to justify if the norms of E and F differ significantly. For example, if E and F are backward error matrices produced by a backward stable algorithm (such as the QZ algorithm) then typically $\|E\|_F$ is proportional to $\|A\|_F$ while $\|F\|_F$ is proportional to $\|B\|_F$. To balance the effects of perturbations, Sun [317] recommends to introduce weighting factors $\gamma_A, \gamma_B > 0$ and to replace $\|(E,F)\|_F$ by $\|(E/\gamma_A, F/\gamma_B)\|_F$ in the considerations above.

On the computation of dif

Kågström and Poromaa [182, 183] have developed methods for estimating dif_l and dif_r, which are in the spirit of estimators for the separation of two matrices and only require the solution of a few generalized Sylvester equations. Based on contributions of these authors such an estimator is implemented in the LAPACK routine `DTGSEN`. This routine greatly benefits from the fact that all involved coefficient matrices are in or close to triangular form, see also [93, 177, 184] for more details on the efficient solution of such generalized Sylvester equations.

No such estimator has been developed for the individual condition numbers $c_r(\mathcal{X})$ and $c_l(\mathcal{Y})$ defined in (2.11), which is rather surprising in the light of Example 2.10. Also, weighting the perturbation matrices E and F differently, as discussed above, has received little attention in currently implemented estimators.

2.2.3 Global Perturbation Bounds

Similar to Section 1.2.4, it is possible to derive global perturbation bounds for the generalized eigenvalue problem which are valid as long as the generalized eigenvalues belonging to the perturbed pair of deflating subspaces $(\hat{\mathcal{X}}, \hat{\mathcal{Y}})$ do not coalesce with other generalized eigenvalues of $(A+E, B+F)$. Demmel and Kågström [105] showed that this coalescence does not take place if

$$x := \frac{\mathrm{dif}_{\min}\ \|(E,F)\|_F}{\sqrt{\|P_r\|_2^2 + \|P_l\|_2^2 + 2\max\{\|P_r\|_2, \|P_l\|_2\}}} < 1 \tag{2.12}$$

where $\mathrm{dif}_{\min} := \min\{\mathrm{dif}_u[(A_{11}, B_{11}), (A_{22}, B_{22})], \mathrm{dif}_l[(A_{11}, B_{11}), (A_{22}, B_{22})]\}$ and P_r, P_l are the right and left spectral projectors belonging to $\lambda(A_{11}, B_{11})$.

The generalized version of Theorem 1.15 reads as follows.

Theorem 2.11 ([105, Lemma 6]). *Let (A, B) have a generalized block Schur decomposition of the form (2.2) and assume that the pair of deflating subspaces $(\mathcal{X}, \mathcal{Y})$ spanned by the first k columns of V and U, respectively, is simple. If (E, F) is a perturbation that satisfies inequality (2.12) then there exists a pair of deflating subspaces $(\hat{\mathcal{X}}, \hat{\mathcal{Y}})$ of $(A+E, B+F)$ so that*

$$\|\tan[\Theta(\mathcal{X},\hat{\mathcal{X}})]\|_2 < \frac{x}{\|P_r\|_2 - x\sqrt{\|P_r\|_2^2 - 1}},$$
$$\|\tan[\Theta(\mathcal{Y},\hat{\mathcal{Y}})]\|_2 < \frac{x}{\|P_l\|_2 - x\sqrt{\|P_l\|_2^2 - 1}}.$$

2.3 The Basic QZ Algorithm

We now turn to the QZ algorithm by Moler and Stewart [248], which aims to compute a (real) generalized Schur decomposition of a matrix pair (A, B). For simplifying the description of this algorithm, let us temporarily assume that B is invertible. We will later on, in Sections 2.3.4 and 2.5.3, discuss how to deal with a singular matrix B.

The fundamental ingredient of the QZ algorithm is a fairly straight generalization of the QR iteration, the so called QZ iteration [248]. It generates a sequence of orthogonally equivalent matrix pairs $(A_0, B_0) \leftarrow (A, B)$, (A_1, B_1), (A_2, B_2), ..., which, under suitable conditions, converges to a generalized block Schur form of (A, B). The QZ iteration relies on a fortunate choice of shift polynomials p_i and reads as follows:

$$p_i(A_{i-1}B_{i-1}^{-1}) = Q_i R_i, \qquad \text{(QR decomposition)} \tag{2.13a}$$
$$\tilde{B}_i \leftarrow Q_i^T B_{i-1}, \tag{2.13b}$$
$$\tilde{B}_i = B_i Z_i, \qquad \text{(RQ decomposition)} \tag{2.13c}$$
$$A_i \leftarrow Q_i^T A_{i-1} Z_i. \tag{2.13d}$$

It is easy to verify that this iteration is formally equivalent to applying a QR iteration to $A_{i-1}B_{i-1}^{-1}$ (yielding the orthogonal matrix Q_i) as well as to $B_{i-1}^{-1}A_{i-1}$ (yielding the orthogonal matrix Z_i). The advantage of the QZ iteration over the QR iteration is that the explicit inversion of the possibly ill-conditioned matrix B is avoided in the formation of Q and Z. If the QZ iteration converges, it produces the block Schur form of a slightly perturbed matrix pair $(A + E, B + F)$, where $\|E\|_F$ and $\|F\|_F$ are of order $\mathbf{u}\,\|A\|_F$ and $\mathbf{u}\,\|B\|_F$, respectively, which implies numerical backward stability.

The intimate relation between QZ and QR iterations answers many theoretical questions such as the convergence of the QZ iteration. For example, Corollary 1.20 implies under rather mild assumptions quadratic convergence of (2.13) if the employed shifts are the eigenvalues of the bottom right $m \times m$ submatrix of $A_{i-1}B_{i-1}^{-1}$, a choice which is sometimes called *generalized Francis shifts*. It should be mentioned, however, that a direct proof of convergence of the QZ iteration has its own advantages and leads to a quantitatively better description of the convergence behavior, see [360].

2.3.1 Hessenberg-Triangular Form

A matrix pair (A, B) is said to be in (unreduced) *Hessenberg-triangular form* if A is an (unreduced) upper Hessenberg matrix and B is a (nonsingular)

upper triangular matrix. By direct computation, it can be seen that a matrix pair (A, B) with nonsingular upper triangular B is in unreduced Hessenberg-triangular form if and only if AB^{-1} is in unreduced Hessenberg form. Thus, Lemma 1.22 applied to $A_{i-1}B_{i-1}^{-1}$ and $B_{i-1}^{-1}A_{i-1}$ implies that the QZ iteration (2.13) preserves unreduced Hessenberg-triangular forms.

In the following, we show how the initial matrix pair (A, B) of the QZ iteration can be reduced to Hessenberg-triangular form. This amounts to computing orthogonal matrices Q and Z so that

$$Q^T (A, B) Z = \left(\left[\begin{array}{cccc} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{array} \right], \left[\begin{array}{cccc} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{array} \right] \right).$$

The first part, reducing B to upper triangular form, is easy. Simply let $B = QR$ be a QR decomposition, then

$$(A, B) \leftarrow Q^T (A, B) = \left(\left[\begin{array}{cccc} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{array} \right], \left[\begin{array}{cccc} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{array} \right] \right).$$

The more difficult part consists of reducing A to upper Hessenberg form while retaining the upper triangular form of B. Reducing the columns of A by applying a Householder matrix is clearly not an option as it would completely destroy the structure of B. Therefore, we need orthogonal transformations that act on smaller parts of a matrix. Householder matrices of tiny order are a viable option but a simpler alternative is provided by Givens rotations.

An $n \times n$ *Givens rotation matrix* has the form

$$G_{ij}(\theta) = \begin{bmatrix} I_{i-1} & & & & \\ & \cos\theta & & \sin\theta & \\ & & I_{j-i-1} & & \\ & -\sin\theta & & \cos\theta & \\ & & & & I_{n-j} \end{bmatrix}, \tag{2.14}$$

for $i < j$ and some angle $\theta \in [-\pi/2, \pi/2)$. The angle can always be chosen so that the jth component of $G_{ij}(\theta)x$ is zero for a fixed vector $x \in \mathbb{R}^n$. In this case, we identify $G_{ij}(\theta) \equiv G_{ij}(x)$. For $i > j$, we use the notation $G_{ij}(\theta) \equiv G_{ij}(x)$ to identify the Givens rotation which maps the ith component of $G_{ij}(\theta)^T x$ to zero. Givens rotation matrices are clearly orthogonal and they act only on two rows or columns when applied to a matrix from the left or right, respectively. `DLARTG` is the LAPACK routine for constructing and `DROT` is the BLAS for applying a Givens rotation.

In the following, we illustrate how Givens rotations can be used to reduce the first column of the matrix A from bottom to top for $n = 4$. First, $G_{34}(Ae_1)$ is applied from the left. This annihilates the $(4, 1)$ element of A but introduces a nonzero $(4, 3)$ element of B:

$$(A, B) \leftarrow G_{34}(Ae_1) \cdot (A, B) = \left(\begin{bmatrix} a & a & a & a \\ a & a & a & a \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} \end{bmatrix}, \begin{bmatrix} b & b & b & b \\ 0 & b & b & b \\ 0 & 0 & \hat{b} & \hat{b} \\ 0 & 0 & \hat{b} & \hat{b} \end{bmatrix} \right).$$

This nonzero element is immediately annihilated by applying $G_{43}(e_4^T B)$ from the right:

$$(A,B) \leftarrow (A,B) \cdot G_{43}(e_4^T B) = \left(\begin{bmatrix} a & a & \hat{a} & \hat{a} \\ a & a & \hat{a} & \hat{a} \\ a & a & \hat{a} & \hat{a} \\ 0 & a & \hat{a} & \hat{a} \end{bmatrix}, \begin{bmatrix} b & b & \hat{b} & \hat{b} \\ 0 & b & \hat{b} & \hat{b} \\ 0 & 0 & \hat{b} & \hat{b} \\ 0 & 0 & \hat{0} & \hat{b} \end{bmatrix} \right).$$

A similar process is used to annihilate the $(3,1)$ element of A:

$$(A,B) \leftarrow G_{23}(Ae_1) \cdot (A,B) = \left(\begin{bmatrix} a & a & a & a \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} \\ 0 & a & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b \\ 0 & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & b \end{bmatrix} \right),$$

$$(A,B) \leftarrow (A,B) \cdot G_{32}(e_3^T B) = \left(\begin{bmatrix} a & \hat{a} & \hat{a} & a \\ a & \hat{a} & \hat{a} & a \\ 0 & \hat{a} & \hat{a} & a \\ 0 & \hat{a} & \hat{a} & a \end{bmatrix}, \begin{bmatrix} b & \hat{b} & \hat{b} & b \\ 0 & \hat{b} & \hat{b} & b \\ 0 & \hat{0} & \hat{b} & b \\ 0 & 0 & 0 & b \end{bmatrix} \right).$$

We can apply an analogous procedure to the second column of A, reducing (A,B) to the desired form.

Generalizing the described procedure to larger n yields Algorithm 9. This

Algorithm 9 Reduction to Hessenberg-triangular form [248]

Input: A general matrix $A \in \mathbb{R}^{n \times n}$ and an upper triangular matrix $B \in \mathbb{R}^{n \times n}$.
Output: Orthogonal matrices $Q, Z \in \mathbb{R}^{n \times n}$. The matrices A and B are overwritten with the upper Hessenberg matrix $Q^T AZ$ and the upper triangular matrix $Q^T BZ$, respectively.

```
Q ← I_n,   Z ← I_n
for j ← 1, ..., n − 2 do
  for i ← n − 1, n − 2, ..., j + 1 do
    G ← G_{i,i+1}(Ae_j)
    A ← GA,   B ← GB,   Q ← QG^T
    G ← G_{i+1,i}(e_{i+1}^T B)
    A ← AG,   B ← BG,   Z ← ZG^T
  end for
end for
```

algorithm, implemented in the LAPACK routine `DGGHRD`, requires $8n^3 + \mathcal{O}(n^2)$ flops for reducing A and B. The accumulation of the orthogonal factors Q and Z adds another $3n^3 + \mathcal{O}(n^2)$ flops for each factor. The preliminary reduction of B to triangular form takes $2/3 \cdot n^3 + \mathcal{O}(n^2)$ flops plus $2n^3 + \mathcal{O}(n^2)$ for updating A and $4/3 \cdot n^3 + \mathcal{O}(n^2)$ flops for computing the corresponding orthogonal factor.

2.3.2 Implicit Shifted QZ Iteration

By the implicit Q theorem, see Theorem 1.23, a QR iteration applied to a matrix $A_{i-1}B_{i-1}^{-1}$ in unreduced Hessenberg form is equivalent to reducing the matrix

$$H_1(x) \cdot A_{i-1}B_{i-1}^{-1} \cdot H_1(x),$$

to Hessenberg form, where $x = p_i(A_{i-1}B_{i-1}^{-1})\, e_1$ denotes the first column of the selected shift polynomial and $H_1(x)$ is the Householder matrix that maps x to a scalar multiple of e_1. Recall that an important premise is that the orthogonal factor Q used for the Hessenberg reduction takes the form $Q = 1 \oplus \tilde{Q}$.

It follows that a QZ iteration applied to a matrix pair (A_{i-1}, B_{i-1}) in unreduced Hessenberg-triangular form is equivalent to reducing the matrix pair

$$(\tilde{A}_{i-1}, \tilde{B}_{i-1}) := H_1(x) \cdot (A_{i-1}, B_{i-1})$$

to Hessenberg-triangular form, provided that this reduction is carefully implemented. A careful implementation returns a left orthogonal factor Q of the form $Q = 1 \oplus \tilde{Q}$. This can be achieved by employing an RQ instead of a QR decomposition for the preliminary reduction of $\tilde{B}_{i-1}$ to triangular form.

Alternatively, one can use a sequence of Givens rotations to map x, the first column of the shift polynomial, to a multiple of e_1 and propagate this sequence through B_{i-1}. Let us illustrate this idea for $n = 6$ and $m = 2$. First, a sequence of two Givens rotations $G_{23}(\theta_1), G_{12}(\theta_2)$ is constructed so that $G_{12}(\theta_1) \cdot G_{23}(\theta_2) \cdot x$ is mapped to a scalar multiple of e_1, where

$$x = (A_{i-1}B_{i-1}^{-1} - \sigma_1 I)(A_{i-1}B_{i-1}^{-1} - \sigma_2 I)e_1$$

and it is assumed that the set of shifts $\{\sigma_1, \sigma_2\}$ is closed under complex conjugation. This sequence is applied from the left to (A_{i-1}, B_{i-1}),

$$(A_{i-1}, B_{i-1}) \leftarrow G_{12}(\theta_1) \cdot G_{23}(\theta_2) \cdot (A_{i-1}, B_{i-1}),$$

which corresponds to the following Wilkinson diagram:

$$\left(\begin{bmatrix} \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & a & a & a & a \\ 0 & 0 & 0 & a & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & b & b & b \\ 0 & 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

Secondly, a sequence of two Givens rotations $G_{32}(\theta_4), G_{21}(\theta_3)$ is constructed so that the matrix B_{i-1} in

$$(A_{i-1}, B_{i-1}) \leftarrow (A_{i-1}, B_{i-1}) \cdot G_{32}(\theta_4) \cdot G_{21}(\theta_3)$$

is reduced to upper triangular form. This corresponds to the following diagram:

$$\left(\begin{bmatrix} \hat{a} & \hat{a} & \hat{a} & a & a & a \\ \hat{b}_a & \hat{b}_a & \hat{b}_a & a & a & a \\ \hat{b}_a & \hat{b}_a & \hat{b}_a & a & a & a \\ \hat{b}_a & \hat{b}_a & \hat{b}_a & a & a & a \\ 0 & 0 & 0 & a & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} \hat{b} & \hat{b} & \hat{b} & b & b & b \\ \hat{0} & \hat{b}_b & \hat{b}_b & b & b & b \\ 0 & \hat{0} & \hat{b}_b & b & b & b \\ 0 & 0 & 0 & b & b & b \\ 0 & 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right). \tag{2.15}$$

We have used the symbols b_a and b_b to designate elements of the so called bulge pair, see Section 2.5.2 below. Finally, the matrix pair (A_{i-1}, B_{i-1}) is returned to Hessenberg-triangular form using Algorithm 9. Overall, we obtain Algorithm 10 for performing an implicit shifted QZ iteration.

Algorithm 10 Implicit shifted QZ iteration

Input: A matrix pair $(A_{i-1}, B_{i-1}) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$ with $n \geq 2$ in unreduced Hessenberg-triangular form, an integer $m \in [2, n]$.

Output: Orthogonal matrices $Q_i, Z_i \in \mathbb{R}^{n\times n}$ such that

$$(A_i, B_i) = Q_i^T \, (A_{i-1}, B_{i-1}) \, Z_i$$

is a suitable next iterate of the QZ iteration (2.13), where the employed shifts are the eigenvalues of the bottom right $m \times m$ submatrix of $A_{i-1}B_{i-1}^{-1}$. The matrix pair (A_{i-1}, B_{i-1}) is overwritten by (A_i, B_i).

1. Compute $(\alpha_1, \beta_1), \ldots, (\alpha_m, \beta_m)$ as generalized eigenvalues of the matrix pair
$$(A_{i-1}(n-m+1:n, n-m+1:n), B_{i-1}(n-m+1:n, n-m+1:n)).$$
2. Set $x = (\beta_1 A_{i-1}B_{i-1}^{-1} - \alpha_1 I_n) \cdots (\beta_m A_{i-1}B_{i-1}^{-1} - \alpha_m I_n)e_1$.
3. Construct a sequence of Givens rotations
$$\tilde{Q} = G_{12}(\theta_1) \cdots G_{m-1,m}(\theta_{m-1}) \cdot G_{m,m+1}(\theta_m)$$
such that $\tilde{Q}x$ is a scalar multiple of e_1.
4. Update $(A_{i-1}, B_{i-1}) \leftarrow \tilde{Q} \, (A_{i-1}, B_{i-1})$.
5. Construct a sequence of Givens rotations
$$\tilde{Z} = G_{m+1,m}(\theta_{m+m}) \cdot G_{m,m-1}(\theta_{m+m-1}) \cdots G_{21}(\theta_{m+1})$$
such that $B_{i-1}\tilde{Z}$ is upper triangular.
6. Update $(A_{i-1}, B_{i-1}) \leftarrow (A_{i-1}, B_{i-1}) \, \tilde{Z}$.
7. Apply Algorithm 9 to compute orthogonal matrices Q and Z so that $(A_{i-1}, B_{i-1}) \leftarrow Q^T \, (A_{i-1}, B_{i-1}) \, Z$ is reduced to Hessenberg-triangular form.
8. Set $Q_i = \tilde{Q}^T Q$, $Z_i = \tilde{Z} Z$.

The remarks concerning the implementation of the implicit shifted QR iteration, Algorithm 2, apply likewise to Algorithm 10. In particular, Step 7 should be based on a special-purpose implementation of Algorithm 9, which

exploits the zero structure of the matrix A_{i-1}. In this case, Step 7 requires about $12mn^2$ flops for updating A_{i-1} and B_{i-1}. About $6mn^2$ flops are additionally needed for updating each of the orthogonal factors Q and Z. The costs for the other steps of Algorithm 10 are negligible provided that $m \ll n$.

Let us illustrate Algorithm 10 for $n = 6$ and $m = 2$. After Step 6 the matrix pair (A_{i-1}, B_{i-1}) takes the form displayed in (2.15), and the bulge pair resides in rows $2, \ldots, 4$ and columns $1, \ldots, 3$. The subsequent reduction to Hessenberg-triangular form can be seen as chasing this bulge pair down to the bottom right corner along the first subdiagonals of A_{i-1} and B_{i-1}. As for the QR iteration, this point of view has been emphasized by Watkins and Elsner [360]. Applying the first outer loop of Algorithm 9 to the matrix pair (A_{i-1}, B_{i-1}) amounts to moving the bulge pair one step downwards:

$$(A_{i-1}, B_{i-1}) \leftarrow \left(\begin{bmatrix} a & \hat{a} & \hat{a} & \hat{a} & a & a \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{b}_a & \hat{b}_a & \hat{b}_a & \hat{a} & \hat{a} \\ \hat{0} & \hat{b}_a & \hat{b}_a & \hat{b}_a & \hat{a} & \hat{a} \\ 0 & \hat{b}_a & \hat{b}_a & \hat{b}_a & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} b & \hat{b} & \hat{b} & \hat{b} & b & b \\ 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & \hat{0} & \hat{b}_b & \hat{b}_b & \hat{b} & \hat{b} \\ 0 & 0 & 0 & \hat{0} & \hat{b}_b & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

Each further execution of an outer loop of Algorithm 9 pushes the bulge pair further downwards until it vanishes at the bottom right corner:

$$(A_{i-1}, B_{i-1}) \leftarrow \left(\begin{bmatrix} a & a & \hat{a} & \hat{a} & \hat{a} & a \\ a & a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{b}_a & \hat{b}_a & \hat{b}_a & \hat{a} \\ 0 & \hat{0} & \hat{b}_a & \hat{b}_a & \hat{b}_a & \hat{a} \\ 0 & 0 & \hat{b}_a & \hat{b}_a & \hat{b}_a & a \end{bmatrix}, \begin{bmatrix} b & b & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & b & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & \hat{0} & \hat{b}_b & \hat{b}_b & \hat{b} \\ 0 & 0 & 0 & \hat{0} & \hat{b}_b & \hat{b} \\ 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right),$$

$$(A_{i-1}, B_{i-1}) \leftarrow \left(\begin{bmatrix} a & a & a & \hat{a} & \hat{a} & \hat{a} \\ a & a & a & \hat{a} & \hat{a} & \hat{a} \\ 0 & a & a & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & \hat{0} & \hat{b}_a & \hat{b}_a & \hat{b}_a \\ 0 & 0 & \hat{0} & \hat{b}_a & \hat{b}_a & \hat{b}_a \end{bmatrix}, \begin{bmatrix} b & b & b & \hat{b} & \hat{b} & \hat{b} \\ 0 & b & b & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & b & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & \hat{0} & \hat{b}_b & \hat{b}_b \\ 0 & 0 & 0 & 0 & \hat{0} & \hat{b}_b \end{bmatrix} \right), \tag{2.16}$$

$$(A_{i-1}, B_{i-1}) \leftarrow \left(\begin{bmatrix} a & a & a & a & \hat{a} & \hat{a} \\ a & a & a & a & \hat{a} & \hat{a} \\ 0 & a & a & a & \hat{a} & \hat{a} \\ 0 & 0 & a & a & \hat{a} & \hat{a} \\ 0 & 0 & 0 & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & 0 & \hat{0} & \hat{a} & \hat{a} \end{bmatrix}, \begin{bmatrix} b & b & b & b & \hat{b} & \hat{b} \\ 0 & b & b & b & \hat{b} & \hat{b} \\ 0 & 0 & b & b & \hat{b} & \hat{b} \\ 0 & 0 & 0 & b & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & \hat{0} & \hat{b} \end{bmatrix} \right).$$

2.3.3 On the Use of Householder Matrices

Early attempts

Algorithm 10 differs in one aspect significantly from the originally proposed implicit QZ iteration, described for the case $m = 2$ in [248]. Moler and Stewart suggest to use transformations based on Householder matrices instead of Givens rotations to move bulge pairs downwards. Their idea is well explained by a 5×5 example:

$$(A, B) = \left(\begin{bmatrix} a & a & a & a & a \\ b_a & b_a & b_a & a & a \\ b_a & b_a & b_a & a & a \\ b_a & b_a & b_a & a & a \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & b \\ 0 & b_b & b_b & b & b \\ 0 & 0 & b_b & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right). \tag{2.17}$$

First, a Householder matrix is applied from the left to annihilate elements $(3, 1)$ and $(4, 1)$ of A:

$$(A, B) \leftarrow \left(\begin{bmatrix} a & a & a & a & a \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & b \\ 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right). \tag{2.18}$$

Next, a Householder matrix is applied from the right to annihilate the newly introduced nonzero elements $(4, 2)$ and $(4, 3)$ of B:

$$(A, B) \leftarrow \left(\begin{bmatrix} a & \hat{a} & \hat{a} & \hat{a} & a \\ a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a \end{bmatrix}, \begin{bmatrix} b & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & \hat{0} & \hat{0} & \hat{b} & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right),$$

and, finally, a Householder matrix (or Givens rotation) from the right is used to annihilate the $(3, 2)$ element of B:

$$(A, B) \leftarrow \left(\begin{bmatrix} a & \hat{a} & \hat{a} & a & a \\ a & \hat{a} & \hat{a} & a & a \\ 0 & \hat{b}_a & \hat{b}_a & b_a & a \\ 0 & \hat{b}_a & \hat{b}_a & b_a & a \\ 0 & \hat{b}_a & \hat{b}_a & b_a & a \end{bmatrix}, \begin{bmatrix} b & \hat{b} & \hat{b} & b & b \\ 0 & \hat{b} & \hat{b} & b & b \\ 0 & \hat{0} & \hat{b}_b & b_b & b \\ 0 & 0 & 0 & b_b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right). \tag{2.19}$$

Using this type of transformation, Algorithm 10 requires about $28n^2$ flops instead of $24n^2$ flops for reducing A and B. The cost for accumulating the orthogonal transformations grows from $24n^2$ to $28n^2$ flops as well. Nevertheless,

this combination is used in the EISPACK implementation `QZIT` of the QZ algorithm. Later on, Ward [345] proposed to use single shift iterations if both shifts are real. The resulting *combination shift QZ algorithm* partly avoids the increase of flops observed above.

Opposite Householder matrices

A computationally less expensive alternative has been proposed by Watkins and Elsner [360]. It is based on the following lemma, which shows how a Householder matrix applied from the *right* can be used to annihilate several entries in one *column*.

Lemma 2.12. *Let $B \in \mathbb{R}^{n\times n}$ be an invertible matrix, then the first column of $B \cdot H_1(B^{-1}e_1)$ is a scalar multiple of e_1.*

Proof. We have $H_1(B^{-1}e_1) \cdot (B^{-1}e_1) = \gamma e_1$ for some nonzero scalar γ, which implies $B \cdot H_1(B^{-1}e_1)e_1 = 1/\gamma \cdot e_1$. □

Normally, a Householder matrix that is used for annihilating several entries in one column is applied from the left, which tempts us to call $H_1(B^{-1}e_1)$ an *opposite Householder matrix*. Some authors have raised fears that the use of these opposite Householder matrices could spoil the numerical backward stability in the presence of an ill-conditioned matrix B, see, e.g., [96, pg. 444]. The error analysis given below shows that such fears are unfounded. First, we provide an example, demonstrating that although an ill-conditioned B may severely affect the data representing $H_1(B^{-1}e_1)$, it has almost no effect on the purpose of $H_1(B^{-1}e_1)$, which is the introduction of zero entries.

Example 2.13. Consider the matrix

$$B = \begin{bmatrix} 0 & 1 \\ 10^{-15} & 0 \end{bmatrix},$$

Let us assume that our favorite method for solving $Bx = e_1$ delivers the computed solution $\hat{x} = [\ 1/10,\ 1\ ]^T$. This corresponds to a pleasantly small residual $\|B\hat{x} - e_1\|_2 = 10^{-16}$, but the forward error $\|x - \hat{x}\|_2 = 10^{-1}$ is rather large due to the potential ill-conditioning of B. Then

$$\hat{v} = \begin{bmatrix} \sqrt{1 + 1/100} + 1/10 \\ 1 \end{bmatrix}, \quad \hat{\beta} = \frac{2}{\hat{v}^T\hat{v}}$$

satisfy $(I - \hat{\beta}\hat{v}\hat{v}^T)\hat{x} = -\sqrt{1 + 1/100}e_1$. The $(2,1)$ entry of $B(I - \hat{\beta}\hat{v}\hat{v}^T)$ is approximately given by 10^{-16}, i.e., the heavily perturbed Householder matrix $H_1(\hat{x})$ has satisfactorily met its goal. ◊

A brief error analysis explains the phenomena observed in the preceeding example. For this purpose, assume that $\hat{x}$ is the exact solution of a perturbed system, i.e.,

$$(A+E)\hat{x} = e_1, \quad \|E\|_2 \le c_A \|A\|_2. \tag{2.20}$$

It can be expected that the constant c_A is not much larger than the unit roundoff $\mathbf{u}$ if $\hat{x}$ is computed by a numerically backward stable method. Consider the Householder matrix $H_1(\hat{x}) \equiv I - \tilde{\beta}\tilde{v}\tilde{v}^T$ with $\tilde{v} \in \mathbb{R}^n$ and $\tilde{\beta} \in \mathbb{R}$, implying $(I - \tilde{\beta}\tilde{v}\tilde{v}^T)\hat{x} = \tilde{\gamma}e_1$ for some scalar $\tilde{\gamma}$. The computation of the quantities $\tilde{v}$ and $\tilde{\beta}$ defining $H_1(\hat{x})$ is subject to roundoff errors. Using the standard algorithm [141], the computed quantities $\hat{v}, \hat{\beta}$ satisfy

$$\|\hat{v} - \tilde{v}\|_2 \le c_v \approx 4n\mathbf{u}, \quad |\hat{\beta} - \tilde{\beta}| \le c_\beta \approx n\mathbf{u},$$

see [164, p. 365]. It follows that

$$\begin{aligned} \|A(I - \hat{\beta}\hat{v}\hat{v}^T)e_1 - 1/\hat{\gamma} \cdot e_1\|_2 &\le \|A(I - \tilde{\beta}\tilde{v}\tilde{v}^T)e_1 - 1/\hat{\gamma} \cdot e_1\|_2 \\ &\quad + (c_\beta + 2c_v)\|A\|_2 + \mathcal{O}(\mathbf{u}^2) \\ &\le (c_A + c_\beta + 2c_v)\|A\|_2 + \mathcal{O}(\mathbf{u}^2). \end{aligned} \tag{2.21}$$

This shows that if $\hat{x}$ is computed by a backward stable method, then the last $n-1$ elements in the first column of $A(I - \hat{\beta}\hat{v}\hat{v}^T)$ can be safely set to zero.

For demonstrating how opposite Householder matrices can be used for chasing bulge pairs, let us reconsider the 5×5 example displayed in (2.17). Again, a Householder matrix is applied from the left to annihilate elements $(3,1)$ and $(4,1)$ of A, leading to the Wilkinson diagram displayed in (2.18). The submatrix $B_{22} = B(2:4, 2:4)$ is invertible, since B itself is assumed to be invertible. Next, the opposite Householder matrix $H_1(B_{22}^{-1}e_1)$ is applied from the right to columns $2, \ldots, 4$ of A and B, which yields

$$(A,B) \leftarrow \left(\begin{bmatrix} a & \hat{a} & \hat{a} & \hat{a} & a \\ a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & \hat{b}_a & \hat{b}_a & \hat{b}_a & a \\ 0 & \hat{b}_a & \hat{b}_a & \hat{b}_a & a \\ 0 & \hat{b}_a & \hat{b}_a & \hat{b}_a & a \end{bmatrix}, \begin{bmatrix} b & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & 0 & \hat{b}_b & \hat{b}_b & b \\ 0 & 0 & \hat{b}_b & \hat{b}_b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right). \tag{2.22}$$

Note that – in contrast to (2.19) – there remains an additional nonzero $(4,3)$ element in B, which, however, does not hinder subsequent bulge chasing steps. For general m and n, the implicit shifted QZ iteration based on (opposite) Householder matrices is given by Algorithm 11.

If $m \ll n$, a proper implementation of Algorithm 11 requires about $2(4m+3)n^2$ flops for updating A and B. Moreover, about $(4m+3)n^2$ flops are required for updating each of the orthogonal factors Q and Z. This algorithm is implemented for $m = 2$ in the LAPACK routine `DHGEQZ`. A curiosity of this routine is that it still uses Ward's combination shift strategy despite the fact that two single shift QZ iterations based on Givens rotations require roughly 9% more flops than one double shift iteration based on Householder matrices.

It remains to discuss the method for solving the linear system of equations in order to determine an opposite Householder matrix. The obvious choice

Algorithm 11 = Algorithm 10 based on Householder matrices

Input and Output: See Algorithm 10.

Apply Steps 1 and 2 of Algorithm 10.
$(A_{i-1}, B_{i-1}) \leftarrow H_1(x) \cdot (A_{i-1}, B_{i-1})$
$Q_{i-1} \leftarrow H_1(x), \quad Z_{i-1} \leftarrow H_1(B_{i-1}^{-1} e_1)$
$(A_{i-1}, B_{i-1}) \leftarrow (A_{i-1}, B_{i-1})\, Z_{i-1}$
for $j \leftarrow 1, \ldots, n-1$ **do**
  $\tilde{Q} \leftarrow H_{j+1}(A_{i-1} e_j)$
  $(A_{i-1}, B_{i-1}) \leftarrow \tilde{Q} \cdot (A_{i-1}, B_{i-1})$
  $Q_{i-1} \leftarrow Q_{i-1} \tilde{Q}$
  $\tilde{Z} \leftarrow H_{j+1}(B_{i-1}^{-1} e_{j+1})$
  $(A_{i-1}, B_{i-1}) \leftarrow (A_{i-1}, B_{i-1})\, \tilde{Z}$
  $Z_{i-1} \leftarrow Z_{i-1} \tilde{Z}$
end for

is Gaussian elimination with partial pivoting [141]. Note, however, that the constant c_A, which bounds the backward error in (2.20), can be proportional to 2^n if this method is used [164], which in turn sacrifices the numerical backward stability of the QZ algorithm. The famous Wilkinson matrix [364, p. 212] is such an "admittedly highly artificial" example. Examples of practical relevance have been discovered by Wright [367] and Foster [127].

The role of RQ decompositions

Iterative refinement or Gaussian elimination with complete pivoting represent alternatives that avoid the described numerical instability implied by the use of Gaussian elimination with partial pivoting. In this section, we favor RQ decompositions for constructing opposite Householder matrices.

Let $B = RQ$ be an RQ decomposition, i.e., the matrix $R \in \mathbb{R}^{n \times n}$ is upper triangular and $Q \in \mathbb{R}^{n \times n}$ is orthogonal. If B is invertible, then $Q^T e_1$ is a scalar multiple of $B^{-1} e_1$ implying that $H_1(Q^T e_1)$ is an opposite Householder matrix. Even if B is singular, it can be shown that the first column of $B \cdot H_1(Q^T e_1)$ is mapped to a multiple of e_1:

$$B \cdot H_1(Q^T e_1) = R(Q \cdot H_1(Q^T e_1)) = \begin{bmatrix} r_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \begin{bmatrix} \tilde{q}_{11} & 0 \\ 0 & \tilde{Q}_{22} \end{bmatrix} \tag{2.23a}$$

$$= \begin{bmatrix} r_{11}\tilde{q}_{11} & R_{12}\tilde{Q}_{22} \\ 0 & R_{22}\tilde{Q}_{22} \end{bmatrix}. \tag{2.23b}$$

The RQ decomposition enjoys a favorable backward error analysis, the constant c_A in (2.20) can be bounded by roughly $n^2\mathbf{u}$, see, e.g., [164, Thm. 18.4].

Another advantage of using RQ decompositions is that they can be easily updated if multiplied by a Householder matrix from the left, see, e.g., [48]. Algorithm 12, which performs such an update, requires $\mathcal{O}(n^2)$ flops instead of

Algorithm 12 Update of an RQ decomposition

Input: An upper triangular matrix $R \in \mathbb{R}^{n\times n}$, an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$, a Householder matrix $I - \beta vv^T$.

Output: An upper triangular matrix $\tilde{R}$ and an orthogonal matrix $\tilde{Q}$ so that $(I - \beta vv^T)RQ = \tilde{R}\tilde{Q}$. The matrices R and Q are overwritten by $\tilde{R}$ and $\tilde{Q}$, respectively.

1. Construct a sequence of Givens rotations
$$Q_1 = G_{21}(\theta_1) \cdot G_{32}(\theta_2) \cdots G_{n,n-1}(\theta_{n-1})$$
such that $Q_1^T v = \gamma e_n$ for some $\gamma \in \mathbb{R}$.
2. Update $R \leftarrow RQ_1$ and $Q \leftarrow Q_1^T Q$. *% R is in upper Hessenberg form.*
3. Update $R \leftarrow R - \beta\gamma v e_n^T$. *% R is still in upper Hessenberg form.*
4. Construct a sequence of Givens rotations
$$Q_2 = G_{n-1,n}(\theta_{2n-2}) \cdots G_{23}(\theta_{n+1}) \cdot G_{12}(\theta_n)$$
such that RQ_2 is upper triangular.
5. Update $R \leftarrow RQ_2$ and $Q \leftarrow Q_2^T Q$.

$\mathcal{O}(n^3)$ flops that are needed for computing an RQ decomposition from scratch. This decrease of flops may not be relevant in the context of Algorithm 11. The number of shifts per implicit QZ iteration must be kept tiny in order to avoid the shift blurring phenomena observed in Section 1.5.3. This implies that the cost for determining opposite Householder matrices can be expected to be negligible anyway. Nevertheless, Algorithm 12 has useful applications in other parts of the QZ algorithm, see Sections 2.5.1 and 2.6.

2.3.4 Deflation

Setting a "small" subdiagonal element $a_{k+1,k}$ of a matrix pair (A, B) in upper Hessenberg-triangular form reduces A to a block upper triangular matrix:

$$(A, B) = \left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} \right), \quad A_{11}, B_{11} \in \mathbb{R}^{k\times k}.$$

This *deflates* the generalized eigenvalue problem into the two smaller generalized eigenvalue problems associated with the matrix pairs (A_{11}, B_{11}) and (A_{22}, B_{22}). A numerically backward stable criterion for considering a subdiagonal entry to be small is given by

$$|a_{k+1,k}| \le \mathbf{u}\, \|A\|_F. \tag{2.24}$$

All known public implementations of the QZ algorithm employ this deflation criterion.[1] A variation of Example 1.24 can be used to show that the neighbor-

[1] A possible exception is the single-shift complex QZ algorithm implemented in MATLAB before MATLAB's `eig` function was based on LAPACK, see [247].

wise criterion

$$|a_{k+1,k}| \leq \mathbf{u}\,(|a_{k,k}| + |a_{k+1,k+1}|) \tag{2.25}$$

may produce more accurately computed generalized eigenvalues in the presence of graded matrix pairs. It is therefore advisable to use (2.25) instead of (2.24) for deflating generalized eigenvalue problems.

Deflation of infinite eigenvalues

If the kth diagonal entry of the matrix B in a matrix pair (A, B) in upper Hessenberg-triangular form happens to be zero, then there is at least one infinite eigenvalue, i.e., a generalized eigenvalue of the form $(\alpha, 0)$ with $\alpha \neq 0$. This infinite eigenvalue can be deflated at the top left corner of the matrix pair using a procedure described in [248]. Let us illustrate this procedure for $n = 5$ and $k = 3$:

$$(A, B) = \left(\begin{bmatrix} a & a & a & a & a \\ a & a & a & a & a \\ 0 & a & a & a & a \\ 0 & 0 & a & a & a \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & b \\ 0 & b & b & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

First, a Givens rotation is applied to columns 2 and 3 in order to annihilate the $(2, 2)$ element of B:

$$(A, B) \leftarrow \left(\begin{bmatrix} a & \hat{a} & \hat{a} & a & a \\ a & \hat{a} & \hat{a} & a & a \\ 0 & \hat{a} & \hat{a} & a & a \\ 0 & \hat{a} & \hat{a} & a & a \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} b & \hat{b} & \hat{b} & b & b \\ 0 & \hat{0} & \hat{b} & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

Secondly, a Givens rotation acting on rows 3 and 4 is used to annihilate the newly introduced nonzero $(4, 2)$ entry of A:

$$(A, B) \leftarrow \left(\begin{bmatrix} a & a & a & a & a \\ a & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & b \\ 0 & 0 & b & b & b \\ 0 & 0 & 0 & \hat{b} & \hat{b} \\ 0 & 0 & 0 & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

Similarly, a zero entry is introduced in the first diagonal entry of B:

$$(A, B) \leftarrow \left(\begin{bmatrix} \hat{a} & \hat{a} & a & a & a \\ \hat{a} & \hat{a} & a & a & a \\ \hat{a} & \hat{a} & a & a & a \\ 0 & 0 & a & a & a \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} 0 & \hat{b} & b & b & b \\ 0 & 0 & b & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right),$$

$$(A,B) \leftarrow \left(\begin{bmatrix} a & a & a & a & a \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & a & a & a \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} 0 & b & b & b & b \\ 0 & 0 & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

Finally, a Givens rotation acting on rows 1 and 2 can be used to deflate the infinite eigenvalue at top:

$$(A,B) \leftarrow \left(\begin{bmatrix} \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hline \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & a & a & a & a \\ 0 & 0 & a & a & a \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hline 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & b & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

If k, the initial position of the zero diagonal element, is larger than $n/2$ it is less expensive to deflate the infinite eigenvalue at the bottom right corner. This can be achieved by a similar procedure [248]. Some applications, such as the semi-discretized Stokes equation [311], lead to matrix pairs having 50% or more infinite eigenvalues. For such matrix pairs, the performance of the QZ algorithm can be considerably improved by using blocked algorithms for deflating infinite eigenvalues, see [180].

Later on, in Section 2.5.3, we will see that it is not necessary to deflate infinite eigenvalues if $k \in [m+1, n-m]$. Otherwise, if $k \in [1,m] \cup [n-m+1, n]$, a zero and even a small value for b_{kk} can have a negative effect on the convergence of the QZ iteration [189, 353]. It is thus advisable to set b_{kk} to zero and deflate an infinite eigenvalue if b_{kk} is sufficiently small. For testing smallness of b_{kk} we may, similar to (2.24)–(2.25), either use the norm-wise criterion

$$|b_{kk}| \leq \mathbf{u}\, \|B\|_F,$$

as implemented in the LAPACK routine `DHGEQZ`, or the more restrictive neighbor-wise criterion

$$|b_{kk}| \leq \mathbf{u}\, (|b_{k-1,k}| + |b_{k,k+1}|).$$

Note, however, that no matter which criterion is used, the QZ algorithm may utterly fail to correctly identify infinite eigenvalues in finite-precision arithmetic, especially if the index of the matrix pair, see [133], is larger than one [248]. The limerick quoted in the beginning of this chapter alludes to this effect.

Example 2.14. Consider the matrix pair

$$(A,B) = Q^T \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \right) Z,$$

where Q and Z are orthogonal matrices generated by the MATLAB code `rand('state',13); [Q,R] = qr(rand(n)); [Z,R] = qr(rand(n));`. The MATLAB function `qz`, which calls the LAPACK implementation of the QZ algorithm, applied to the matrix pair (A, B) returns an upper triangular matrix pair (S, T) with diagonal elements $(1.00 + 2.43 \times 10^{-8}\mathrm{i}, -0.50 - 0.87\mathrm{i}, -0.50 + 0.87\mathrm{i})$ and $(3.72 \times 10^{-6}, 3.72 \times 10^{-6}, 3.72 \times 10^{-6})$, respectively. ◊

The matrix pair in the preceeding example has index 3 and the diagonal elements of the matrix T, which should all be zero in exact arithmetic, are perturbed to entries of order $\sqrt[3]{\mathbf{u}}$. As a rule of thumb, if a matrix pair (A, B) has index k then at least some of the infinite eigenvalues are likely to show up as entries of order $\sqrt[k]{\mathbf{u}}\,\|B\|_2$ on the diagonal of T [210, 250]. In extreme cases, even otherwise well-conditioned eigenvalues can be affected by perturbations from these defective infinite eigenvalues, e.g., they may coalesce or appear in clusters of eigenvalues corresponding to computed infinite as well as finite eigenvalues. Unfortunately, many applications, such as multibody systems and electrical circuits, lead to matrix pairs with index at least two, see, e.g., [63, 150, 267]. For all these applications, the QZ algorithm is of limited use for identifying infinite eigenvalues.

In principle, the only reliable and robust way to identify infinite eigenvalues is to apply a preprocessing step with a staircase-type algorithm, such as GUPTRI [106, 107]. In some cases, infinite eigenvalues can be cheaply and reliably deflated by exploiting the structure of A and B, see, e.g. [15, 234, 311, 321].

2.3.5 The Overall Algorithm

Glueing implicit QZ iterations and deflation together yields Algorithm 13, the QZ algorithm for Hessenberg-triangular matrix pairs.

As for the QR algorithm, the computational cost of this algorithm depends on the matrix pair in question. Assuming that on average four shifts are necessary to let one eigenvalue converge, Algorithm 13 needs about twice the number of flops required by the basic QR algorithm applied to an $n \times n$ matrix.

Remark 2.15. Similar to the QR algorithm, see Remark 1.26, post-processing must be applied in order to guarantee that Algorithm 13 returns a real generalized Schur form. Based on work by Van Loan [330, 331, 248], the LAPACK routine `DLAGV2` transforms the 2×2 diagonal blocks of the matrix pair (S, T) returned by Algorithm 13 to the form

$$\left(\begin{bmatrix} s_{ii} & s_{i,i+1} \\ 0 & s_{i+1,i+1} \end{bmatrix}, \begin{bmatrix} t_{ii} & t_{i,i+1} \\ 0 & t_{i+1,i+1} \end{bmatrix} \right),$$

in the case of real eigenvalues, or to the form

$$\left(\begin{bmatrix} s_{ii} & s_{i,i+1} \\ 0 & s_{i+1,i+1} \end{bmatrix}, \begin{bmatrix} t_{ii} & 0 \\ 0 & t_{i+1,i+1} \end{bmatrix} \right), \quad t_{ii} \geq t_{i+1,i+1} > 0, ,$$

Algorithm 13 Basic QZ algorithm

Input: A matrix pair $(A, B) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$ with $n \geq 2$ in Hessenberg-triangular form, an integer $m \in [2, n]$.

Output: Orthogonal matrices $Q, Z \in \mathbb{R}^{n\times n}$ such that the matrix pair $(S, T) = Q^T (A, B) Z$ is in generalized real Schur form. The matrices A and B are overwritten by S and T, respectively. Implicit shifted QZ iterations with at most m generalized Francis shifts are used.

$Q \leftarrow I_n, \quad Z \leftarrow I_n, \quad i \leftarrow 1, \quad l \leftarrow n$
for $it \leftarrow 0, \ldots, 30n$ **do**
  % The active matrix pair is $(A(i:l, i:l), B(i:l, i:l))$.
  % Search for deflations in B.
  $k \leftarrow i, \quad \text{tol} \leftarrow 0$
  while $(k < l)$ **and** $(|b_{kk}| > \mathbf{u}\,|b_{k,k+1}| + \text{tol})$ **do**
    $k \leftarrow k + 1, \quad \text{tol} \leftarrow \mathbf{u}\,|b_{k-1,k}|$
  end while
  if $|b_{ll}| > \mathbf{u}\,|b_{l-1,l}|$ **then**
    $k \leftarrow k + 1$
  end if
  if $k \leq l$ **then**
    % Deflation of infinite eigenvalue at position (k, k) *found.*
    $b_{kk} \leftarrow 0$
    Deflate infinite eigenvalue as described in Section 2.3.4.
  end if
  % Search for deflations in A.
  $k \leftarrow i$
  while $(k < l)$ **and** $(|a_{k+1,k}| > \mathbf{u}\,(|a_{k,k}| + |a_{k+1,k+1}|))$ **do**
    $k \leftarrow k + 1$
  end while
  if $k < l$ **then**
    % Deflation at position $(k + 1, k)$ *found.*
    $a_{k+1,k} \leftarrow 0, \quad i \leftarrow k + 1$
    if $i + 1 \geq l$ **then**
      % Block of size at most two has converged.
      $l \leftarrow i - 1, \quad i \leftarrow 1$
      if $l \leq 2$ **then**
        % QZ algorithm has converged.
        Exit.
      end if
    end if
  else
    Apply Algorithm 11 with $\min\{m, l-i+1\}$ shifts to $(A(i:l, i:l), B(i:l, i:l))$.
    Let $\tilde{Q}, \tilde{Z}$ denote the returned orthogonal transformation matrices.
    $(A(i:l, l+1:n), B(i:l, l+1:n)) \leftarrow \tilde{Q}^T\,(A(i:l, l+1:n), B(i:l, l+1:n))$
    $(A(1:i-1, i:l), B(1:i-1, i:l)) \leftarrow (A(1:i-1, i:l), B(1:i-1, i:l))\,\tilde{Z}$.
    $Q(1:n, i:l) \leftarrow Q(1:n, i:l)\tilde{Q}, \quad Z(1:n, i:l) \leftarrow Z(1:n, i:l)\tilde{Z}$.
  end if
end for
% The QZ algorithm did not converge within $30n$ *iterations.*
Exit and error return.

in the case of complex eigenvalues. ◊

Algorithm 13 is implemented in LAPACK as subroutine `DHGEQZ`, which uses either a complex conjugate pair of shifts or a single real shift. The auxiliary subroutine `DLAG2` is used to compute eigenvalues of 2×2 generalized eigenvalue problems in a stable fashion. Algorithm 13 inherits the (rare) possibility of global convergence failures from the QR algorithm. Exceptional shift strategies similar to those described in Section 1.3.6, have been developed for the QZ algorithm and are partly incorporated in `DHGEQZ`.

2.4 Balancing

Balancing a matrix pair is a preprocessing step which can have positive effects on the accuracy and efficiency of subsequent methods for computing generalized eigenvalues and deflating subspaces [218, 346]. As in the matrix case, see Section 1.4, balancing consists of two stages. In the first stage, the matrix pair is permuted in order to make it look closer to a (block) upper triangular matrix pair. The second stage consists of a diagonal equivalence transformation (scaling), which aims at reducing the sensitivity of the generalized eigenvalues.

2.4.1 Isolating Eigenvalues

In the first stage of the balancing algorithm by Ward [346], permutation matrices P_R and P_C are constructed so that $P_R^T(A,B)P_C$ takes the form

$$P_R^T(A,B)P_C = \left(\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ 0 & A_{22} & A_{23} \\ 0 & 0 & A_{33} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ 0 & B_{22} & B_{23} \\ 0 & 0 & B_{33} \end{bmatrix} \right), \tag{2.26}$$

where $A_{11}, B_{11} \in \mathbb{R}^{(i_l-1)\times(i_l-1)}$ and $A_{33}, B_{33} \in \mathbb{R}^{(n-i_h)\times(n-i_h)}$ are upper triangular matrices. The *isolated generalized eigenvalues* contained in the corresponding triangular matrix pairs can be read off without any roundoff error. Consequently, the order of the generalized eigenvalue problem is reduced to $i_h - i_l + 1$. The remaining matrix pair (A_{22}, B_{22}) in the middle is characterized by the property that each column and row of $|A_{22}|+|B_{22}|$ contains at least two nonzero entries. (Here, $|X|$ denotes the matrix that is obtained by replacing the entries of a matrix X by their absolute values.) Elementary permutation matrices can be used to produce the decomposition (2.26) in a similar fashion as in Algorithm 4.

2.4.2 Scaling

In the second stage, an equivalence transformation involving diagonal matrices D_R and D_C is applied to the unreduced matrix pair (A_{22}, B_{22}) in (2.26). For convenience, let us assume $(A,B) = (A_{22}, B_{22})$.

Ward [346] proposed to choose nonsingular diagonal matrices D_R and D_C so that the nonzero elements of $D_R A D_C$ and $D_R B D_C$ are as nearly equal in magnitude as possible. This problem can be formulated as a linear least-squares problem:

$$\sum_{a_{ij}\neq 0} (\log|a_{ij}|^2 + \rho_i + \gamma_j)^2 + \sum_{b_{ij}\neq 0} (\log|b_{ij}|^2 + \rho_i + \gamma_j)^2 = \min, \tag{2.27}$$

where ρ_i and γ_j denote the logarithms of the (positive) diagonal entries of D_R and D_C, respectively. By applying a conjugate gradient method, the minimization problem (2.27) can be iteratively and approximately solved within $\mathcal{O}(n^2)$ flops, see [346] for more details.

This balancing strategy together with the permutation algorithm outlined above is implemented in the LAPACK routine `DGGBAL`. The information contained in the permutation matrices P_R, P_C and the scaling matrices D_R, D_C is stored in two vectors "lscal" and "rscal" of length n in the same way as in the routine `DGEBAL`, see Page 38.

Ward's algorithm must be applied with care. In fact, it is simple to construct examples where a balancing strategy based on (2.27) severely deteriorates the eigenvalue sensitivities.

Example 2.16. Consider the matrix pair

$$(A,B) = \left(\begin{bmatrix} 1 & 10^{-15} & 1 \\ 10^{-15} & 2 & 1 \\ 1 & 1 & 3 \end{bmatrix}, \begin{bmatrix} 3 & 10^{-15} & 1 \\ 10^{-15} & 2 & 4 \\ 1 & 4 & 1 \end{bmatrix} \right).$$

The LAPACK routine `DGGBAL` applied to this matrix pair produces the balanced pair

$$D_R(A,B)D_C = \left(\begin{bmatrix} 10^{6} & 10^{-9} & 10 \\ 10^{-9} & 2\times 10^{6} & 10 \\ 10 & 10 & 3\times 10^{-4} \end{bmatrix}, \begin{bmatrix} 3\times 10^{6} & 10^{-9} & 10 \\ 10^{-9} & 2\times 10^{6} & 40 \\ 10 & 40 & 10^{-4} \end{bmatrix} \right).$$

Let $(\hat{\alpha}_i, \hat{\beta}_i)$ and $(\check{\alpha}_i, \check{\beta}_i)$ denote the generalized eigenvalues computed by the QZ algorithm applied to the matrix pairs (A,B) and $D_R(A,B)D_C$, respectively. The following table displays the chordal distances between these computed eigenvalues and the exact eigenvalues (α_i, β_i) of (A,B).

α_i/β_i	$\chi(\langle\hat{\alpha}_i,\hat{\beta}_i\rangle,\langle\alpha_i,\beta_i\rangle)$	$\chi(\langle\check{\alpha}_i,\check{\beta}_i\rangle,\langle\alpha_i,\beta_i\rangle)$
0.60644158364840	1.7×10^{-17}	3.1×10^{-8}
−0.41974660144673	7.2×10^{-17}	1.7×10^{-7}
0.26785047234378	7.5×10^{-17}	2.6×10^{-8}

It can be observed that balancing has led to a dramatic loss of accuracy for all eigenvalues. ◊

Lemonnier and Van Dooren [218] recently proposed a balancing strategy for matrix pairs which is closer in spirit to the Parlett-Reinsch algorithm for balancing matrices, see also Algorithm 5. It produces diagonal matrices D_R and D_C so that every row of $D_R[A, B]$ and every column of $\begin{bmatrix} A \\ B \end{bmatrix} D_C$ has 2-norm nearly equal to one. Note that, since A and B are real matrices, this condition equals the condition that the 2-norm of every row and every column of the complex matrix $D_R(A + \mathrm{i}B)D_C$ is nearly equal to one. This is a well-studied problem in linear algebra, which amounts, if the 2-norm is replaced by the 1-norm, to the line sum scaling problem, see [254, 274] and the references therein.

The algorithm proposed in [218] first computes a diagonal matrix D_R so that every row of $D_R[A, B]$ has 2-norm equal to one and performs the update $(A, B) \leftarrow D_R(A, B)$. Next, it computes a diagonal matrix D_C so that every column of $\begin{bmatrix} A \\ B \end{bmatrix} D_C$ has 2-norm equal to one and performs the update $(A, B) \leftarrow (A, B)D_C$. This procedure is repeatedly applied until convergence occurs. For the numerical experiments reported in [218], convergence occurs quickly (typically after two or three iterations) if the diagonal entries of D_R and D_C are rounded to powers of two. The experiments also suggest that this algorithm is capable to yield substantial improvements to the accuracy of computed eigenvalues. However, the following example reveals that there is still some potential for further improvements, see also [231].

Example 2.17. Consider the matrices

$$A = 10^{-6} \times \begin{bmatrix} 1 & 2 & 0 \\ 2 & 2 & 2 \\ 0 & 1 & 1 \end{bmatrix}, \quad D = \mathrm{diag}(2^{-17}, 1, 1).$$

The eigenvalues of A are $\{-10^{-6}, 10^{-6}, 4 \times 10^{-6}\}$. If the algorithm by Lemonnier and Van Dooren [218] is applied to the matrix pair $(\tilde{A}, I_3) := (D^{-1}AD, I_3)$, then the returned diagonal scaling matrices are given by $D_R = D_C = I_3$, i.e., no action is taken. The eigenvalues computed by the QZ algorithm applied to $(\tilde{A}, I_3)$ are perturbed by relative errors of order 10^{-11}. On the other hand, if the Parlett-Reinsch algorithm is applied to $\tilde{A}$, then the original matrix A is recovered and the QZ algorithm applied to (A, I_3) computes all eigenvalues to full accuracy. ◊

2.5 Block Algorithms

For a long time, developing block algorithms for the ingredients of the QZ algorithm had been an underappreciated aspect. In the following, we outline some rather recent developments in the area.

2.5.1 Reduction to Hessenberg-Triangular Form

The reduction to Hessenberg-triangular form as implemented in Algorithm 9 is solely based on Givens rotations. This is necessary to avoid excessive fill-in in the triangular B factor. Unfortunately, it also implies that Algorithm 9 performs poorly for larger matrix pencils on modern computer architectures. Dackland and Kågström [96] proposed an alternative by approaching the Hessenberg-triangular form in two stages. In Stage 1, the matrix A is reduced to block upper Hessenberg form using a fairly straightforward block version of Algorithm 9. Stage 2 consists of chasing the unwanted subdiagonal elements to the bottom right corner of the matrix A. Similar ideas have been used for reducing a general matrix to Hessenberg form or a symmetric matrix to tridiagonal form [47, 46, 208].

Stage 1

For describing the reduction to block Hessenberg-triangular form it helps to partition A and B into blocks A_{ij} and B_{ij} of block size n_b. The implicit assumption that n is an integer multiple n_b is for notational convenience only. Let us illustrate the block partitioning for $n = 6n_b$:

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} & A_{26} \\ A_{31} & A_{32} & A_{33} & A_{34} & A_{35} & A_{36} \\ A_{41} & A_{42} & A_{43} & A_{44} & A_{45} & A_{46} \\ A_{51} & A_{52} & A_{53} & A_{54} & A_{55} & A_{56} \\ A_{61} & A_{62} & A_{63} & A_{64} & A_{65} & A_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} & B_{13} & B_{14} & B_{15} & B_{16} \\ 0 & B_{22} & B_{23} & B_{24} & B_{25} & B_{26} \\ 0 & 0 & B_{33} & B_{34} & B_{35} & B_{36} \\ 0 & 0 & 0 & B_{44} & B_{45} & B_{46} \\ 0 & 0 & 0 & 0 & B_{55} & B_{56} \\ 0 & 0 & 0 & 0 & 0 & B_{66} \end{bmatrix} \right).$$

We assume B to be upper triangular, which implies that each of its diagonal blocks B_{ii} is also upper triangular. Our goal is to reduce the matrix pair (A, B) to block upper Hessenberg form

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} & A_{26} \\ 0 & A_{32} & A_{33} & A_{34} & A_{35} & A_{36} \\ 0 & 0 & A_{43} & A_{44} & A_{45} & A_{46} \\ 0 & 0 & 0 & A_{54} & A_{55} & A_{56} \\ 0 & 0 & 0 & 0 & A_{65} & A_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} & B_{13} & B_{14} & B_{15} & B_{16} \\ 0 & B_{22} & B_{23} & B_{24} & B_{25} & B_{26} \\ 0 & 0 & B_{33} & B_{34} & B_{35} & B_{36} \\ 0 & 0 & 0 & B_{44} & B_{45} & B_{46} \\ 0 & 0 & 0 & 0 & B_{55} & B_{56} \\ 0 & 0 & 0 & 0 & 0 & B_{66} \end{bmatrix} \right),$$

where all subdiagonal blocks $A_{i+1,i}$ and all diagonal blocks B_{ii} are upper triangular.

For this purpose, we first apply an orthogonal transformation to $p \geq 2$ bottom blocks of the first block column of A in order to annihilate the last $p-1$ blocks. For example if $p = 3$, this amounts to computing a QR decomposition

$$\begin{bmatrix} A_{41} \\ A_{51} \\ A_{61} \end{bmatrix} = GR = (I + VTV^T) \begin{bmatrix} \hat{A}_{41} \\ 0 \\ 0 \end{bmatrix},$$

where $I + VTV^T$ with $T \in \mathbb{R}^{n_b \times n_b}, V \in \mathbb{R}^{pn_b \times n_b}$ is the compact WY representation of the orthogonal factor G. Applying $(I+VTV^T)^T$ to the last three rows of A and B yields

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} & A_{26} \\ A_{31} & A_{32} & A_{33} & A_{34} & A_{35} & A_{36} \\ \hat{A}_{41} & \hat{A}_{42} & \hat{A}_{43} & \hat{A}_{44} & \hat{A}_{45} & \hat{A}_{46} \\ \hat{0} & \hat{A}_{52} & \hat{A}_{53} & \hat{A}_{54} & \hat{A}_{55} & \hat{A}_{56} \\ \hat{0} & \hat{A}_{62} & \hat{A}_{63} & \hat{A}_{64} & \hat{A}_{65} & \hat{A}_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} & B_{13} & B_{14} & B_{15} & B_{16} \\ 0 & B_{22} & B_{23} & B_{24} & B_{25} & B_{26} \\ 0 & 0 & B_{33} & B_{34} & B_{35} & B_{36} \\ 0 & 0 & 0 & \hat{B}_{44} & \hat{B}_{45} & \hat{B}_{46} \\ 0 & 0 & 0 & \hat{B}_{54} & \hat{B}_{55} & \hat{B}_{56} \\ 0 & 0 & 0 & \hat{B}_{64} & \hat{B}_{65} & \hat{B}_{66} \end{bmatrix} \right).$$

One possibility to annihilate the fill-in in the matrix B consists of computing a complete RQ decomposition

$$\begin{bmatrix} B_{44} & B_{45} & B_{46} \\ B_{54} & B_{55} & B_{56} \\ B_{64} & B_{65} & B_{66} \end{bmatrix} = \tilde{R}\tilde{G}, \tag{2.28}$$

where R is upper triangular and $\tilde{G}$ is orthogonal. A disadvantage of this approach is that one cannot use a slim compact WY representation for applying the factor $\tilde{G}$ afterwards, which becomes rather expensive for larger p. One can avoid this disadvantage by using a technique similar to opposite Householder matrices. Let

$$\tilde{G}^T \begin{bmatrix} I_{n_b} \\ 0 \\ 0 \end{bmatrix} = \check{G} \begin{bmatrix} \tilde{G}_1 \\ 0 \\ 0 \end{bmatrix}$$

be a QR decomposition of the first block column of $\tilde{G}^T$. Then the orthogonal factor $\check{G}$ has a compact WY representation $\check{G} = I + \check{V}\check{T}\check{V}^T$ for some $\check{T} \in \mathbb{R}^{n_b \times n_b}, \check{V} \in \mathbb{R}^{p \cdot n_b \times n_b}$. Applying $I + \check{V}\check{T}\check{V}^T$ to the last three columns of A and B from the right produces the form

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & \hat{A}_{14} & \hat{A}_{15} & \hat{A}_{16} \\ A_{21} & A_{22} & A_{23} & \hat{A}_{24} & \hat{A}_{25} & \hat{A}_{26} \\ A_{31} & A_{32} & A_{33} & \hat{A}_{34} & \hat{A}_{35} & \hat{A}_{36} \\ A_{41} & A_{42} & A_{43} & \hat{A}_{44} & \hat{A}_{45} & \hat{A}_{46} \\ 0 & A_{52} & A_{53} & \hat{A}_{54} & \hat{A}_{55} & \hat{A}_{56} \\ 0 & A_{62} & A_{63} & \hat{A}_{64} & \hat{A}_{65} & \hat{A}_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} & B_{13} & \hat{B}_{14} & \hat{B}_{15} & \hat{B}_{16} \\ 0 & B_{22} & B_{23} & \hat{B}_{24} & \hat{B}_{25} & \hat{B}_{26} \\ 0 & 0 & B_{33} & \hat{B}_{34} & \hat{B}_{35} & \hat{B}_{36} \\ 0 & 0 & 0 & \hat{B}_{44} & \hat{B}_{45} & \hat{B}_{46} \\ 0 & 0 & 0 & \hat{0} & \hat{B}_{55} & \hat{B}_{56} \\ 0 & 0 & 0 & \hat{0} & \hat{B}_{65} & \hat{B}_{66} \end{bmatrix} \right),$$

where $\hat{B}_{44}$ is an upper triangular matrix. In the presence of roundoff errors the norm of the newly created zero blocks in B is of order $\mathbf{u}\,\|\tilde{R}\|_F$. These facts can be proven along the line of argument used in Section 2.3.2.

The blocks A_{31} and A_{41} are annihilated in a similar fashion:

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ \hat{A}_{21} & \hat{A}_{22} & \hat{A}_{23} & \hat{A}_{24} & \hat{A}_{25} & \hat{A}_{26} \\ \hat{0} & \hat{A}_{32} & \hat{A}_{33} & \hat{A}_{34} & \hat{A}_{35} & \hat{A}_{36} \\ \hat{0} & \hat{A}_{42} & \hat{A}_{43} & \hat{A}_{44} & \hat{A}_{45} & \hat{A}_{46} \\ 0 & A_{52} & A_{53} & A_{54} & A_{55} & A_{56} \\ 0 & A_{62} & A_{63} & A_{64} & A_{65} & A_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} & B_{13} & B_{14} & B_{15} & B_{16} \\ 0 & \hat{B}_{22} & \hat{B}_{23} & \hat{B}_{24} & \hat{B}_{25} & \hat{B}_{26} \\ 0 & \hat{B}_{32} & \hat{B}_{33} & \hat{B}_{34} & \hat{B}_{35} & \hat{B}_{36} \\ 0 & \hat{B}_{42} & \hat{B}_{43} & \hat{B}_{44} & \hat{B}_{45} & \hat{B}_{46} \\ 0 & 0 & 0 & 0 & B_{55} & B_{56} \\ 0 & 0 & 0 & 0 & B_{65} & B_{66} \end{bmatrix}\right),$$

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & \hat{A}_{22} & \hat{A}_{23} & \hat{A}_{24} & A_{25} & A_{26} \\ 0 & \hat{A}_{32} & \hat{A}_{33} & \hat{A}_{34} & A_{35} & A_{36} \\ 0 & \hat{A}_{42} & \hat{A}_{43} & \hat{A}_{44} & A_{45} & A_{46} \\ 0 & \hat{A}_{52} & \hat{A}_{53} & \hat{A}_{54} & A_{55} & A_{56} \\ 0 & \hat{A}_{62} & \hat{A}_{63} & \hat{A}_{64} & A_{65} & A_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & \hat{B}_{12} & \hat{B}_{13} & \hat{B}_{14} & B_{15} & B_{16} \\ 0 & \hat{B}_{22} & \hat{B}_{23} & \hat{B}_{24} & B_{25} & B_{26} \\ 0 & \hat{0} & \hat{B}_{33} & \hat{B}_{34} & B_{35} & B_{36} \\ 0 & \hat{0} & \hat{B}_{43} & \hat{B}_{44} & B_{45} & B_{46} \\ 0 & 0 & 0 & 0 & B_{55} & B_{56} \\ 0 & 0 & 0 & 0 & B_{65} & B_{66} \end{bmatrix}\right).$$

The reduction of the second block column proceeds analogously. First, only the blocks A_{62} and B_{65} are annihilated and second, the blocks A_{42}, A_{52} and B_{43}, B_{53}. Algorithm 14 describes the complete procedure schematically for general n, n_b, and p. A proper implementation of this algorithm should be

Algorithm 14 Reduction to block Hessenberg-triangular form

Input: A general matrix $A \in \mathbb{R}^{n\times n}$ and an upper triangular matrix $B \in \mathbb{R}^{n\times n}$. An integer n_b being the block size and an integer $p \geq 2$ being the number of block rows/columns to be transformed in each step of the algorithm.

Output: Orthogonal matrices $Q, Z \in \mathbb{R}^{n\times n}$. The matrices A and B are overwritten with the block upper Hessenberg matrix $Q^T AZ$ and the upper triangular matrix $Q^T BZ$, respectively.

$Q \leftarrow I_n, \quad Z \leftarrow I_n$
$l \leftarrow n - n_b - [(n - 2n_b - 1) \bmod ((p-1)n_b)]$
for $j \leftarrow 1, n_b + 1, 2n_b + 1 \ldots, n - 3n_b + 1$ **do**
  for $i \leftarrow l, l - (p-1)n_b, l - 2(p-1)n_b, \ldots, j + n_b$ **do**
    $m \leftarrow \min\{n - i + 1, pn_b\}$
    Compute QR decomposition $A(i : i + m - 1, j : j + k - 1) = GR$.
    $(A(i : i+m-1, :), B(i : i+m-1, :)) \leftarrow G^T\,(A(i : i+m-1, :), B(i : i+m-1, :))$
    $Q_{:,i:i+m-1} \leftarrow Q_{:,i:i+m-1}G$
    Compute RQ decomposition $B(i : i + m - 1, i : i + m - 1) = \tilde{R}\tilde{G}$.
    Compute QR decomposition $\tilde{G}^T_{1:n_b,:} = \check{G}\check{R}$.
    $(A(:, i : i+m-1), B(:, i : i+m-1)) \leftarrow (A(:, i : i+m-1), B(:, i : i+m-1))\,\check{G}$
    $Z(:, i : i + m - 1) \leftarrow Z(:, i : i + m - 1)\check{G}$
  end for
  $l \leftarrow l + k$
  if $n_b + l > n$ **then**
    $l \leftarrow l - (p-1)n_b$
  end if
end for

Alg. 14		w/o compt. of Q and Z				with compt. of Q and Z			
n	n_b	$p=2$	$p=4$	$p=6$	$p=8$	$p=2$	$p=4$	$p=6$	$p=8$
500	8	0.15	0.10	0.09	0.09	0.26	0.16	0.15	0.14
500	16	0.11	0.08	0.08	0.09	0.19	0.13	0.12	0.13
500	32	0.09	0.07	0.08	0.09	0.14	0.11	0.11	0.12
500	64	0.07	0.07	0.07	0.08	0.11	0.09	0.10	0.11
1000	8	1.39	0.84	0.75	0.71	2.27	1.39	1.21	1.15
1000	16	0.99	0.65	0.62	0.68	1.69	1.07	1.00	1.07
1000	32	0.77	0.58	0.57	0.63	1.28	0.92	0.88	0.93
1000	64	0.69	0.55	0.54	0.67	1.10	0.83	0.82	0.92
1000	128	0.59	0.54	0.56	0.74	0.90	0.76	0.77	0.95
2000	8	14.35	8.64	7.56	7.20	22.46	13.53	11.85	11.28
2000	16	9.45	6.08	5.72	5.70	16.18	10.21	9.21	9.16
2000	32	7.09	4.93	4.55	4.80	13.33	8.34	7.58	7.88
2000	64	6.39	4.42	4.30	5.01	11.71	7.52	6.97	7.61
2000	128	5.68	4.55	4.46	6.13	9.52	7.08	6.70	7.52

Table 2.1. Performance results in minutes for the reduction of a matrix pencil to block Hessenberg form using Algorithm 14.

based on compact WY representations as explained above.

Table 2.1 contains timings that have been obtained from a Fortran implementation of Algorithm 14 applied to random matrix pairs. It does not include the preliminary reduction of B to triangular form. The most remarkable conclusion we can draw from these figures is that larger p often have a positive effect on the performance of Algorithm 14, with greater benefit for moderate block sizes n_b. This is particularly useful considering the fact that the computational cost of the second stage grows with n_b.

Algorithm 14 is almost identical with the original algorithm proposed by Dackland and Kågström [96, Sec. 2.3]. The only difference is that the latter algorithm directly uses the RQ decomposition (2.28) for annihilating blocks in B, which makes it computationally more expensive for $p > 2$.

In some cases, the structure of a matrix pair admits a much cheaper reduction to block Hessenberg-triangular form.

Example 2.18. Consider the following matrix pair of block Hankel matrices

$$(A,B) = \left(\begin{bmatrix} H_1 & H_2 & \cdots & H_N \\ H_2 & H_3 & \cdots & H_{N+1} \\ \vdots & \vdots & \ddots & \vdots \\ H_N & H_{N+1} & \cdots & H_{2N-1} \end{bmatrix}, \begin{bmatrix} H_0 & H_1 & \cdots & H_{N-1} \\ H_1 & H_2 & \cdots & H_N \\ \vdots & \vdots & \ddots & \vdots \\ H_{N-1} & H_N & \cdots & H_{2N-2} \end{bmatrix} \right),$$

where $H_0, H_1, \ldots, H_{2N-1} \in \mathbb{R}^{n_b \times n_b}$. Such matrix pairs play a role in algorithms for reconstructing polygonal shapes from moments, see [137]. Also, $B^{-1}A$ is the companion matrix of a matrix polynomial [136].

The matrix pair (A, B) can be reduced to block Hessenberg-triangular form by applying a QR decomposition to the matrix B augmented by the last n_b columns of A. Then the resulting upper trapezoidal factor $R \in \mathbb{R}^{Nn_b \times (N+1)n_b}$ contains in its first Nn_b columns the upper triangular matrix $Q^T B$ and in its last Nn_b columns the block upper Hessenberg matrix $Q^T A$, where $Q \in \mathbb{R}^{Nn_b \times Nn_b}$ is the orthogonal factor obtained from the QR decomposition. $\Diamond$

Stage 2

In Stage 2, the unwanted $n_b - 1$ subdiagonals of the matrix A in block Hessenberg form are annihilated while the triangular structure of B is preserved. The basic algorithm that applies here is a variant of the QZ iteration, Algorithm 13, with the major difference that bulges are chased along the n_bth instead of the first subdiagonal. Let us illustrate the first few steps for $n_b = 3$ and $n = 8$:

$$\left(\begin{bmatrix} a & a & a & a & a & a & a & a \\ a & a & a & a & a & a & a & a \\ a & a & a & a & a & a & a & a \\ a & a & a & a & a & a & a & a \\ 0 & a & a & a & a & a & a & a \\ 0 & 0 & a & a & a & a & a & a \\ 0 & 0 & 0 & a & a & a & a & a \\ 0 & 0 & 0 & 0 & a & a & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & b & b & b & b \\ 0 & b & b & b & b & b & b & b \\ 0 & 0 & b & b & b & b & b & b \\ 0 & 0 & 0 & b & b & b & b & b \\ 0 & 0 & 0 & 0 & b & b & b & b \\ 0 & 0 & 0 & 0 & 0 & b & b & b \\ 0 & 0 & 0 & 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

First, a Householder matrix of order 3 is constructed, which, when applied to rows $2, \ldots, 4$, annihilates elements $(3, 1)$ and $(4, 1)$ of A:

$$\left(\begin{bmatrix} a & a & a & a & a & a & a & a \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & a & a & a & a & a & a & a \\ 0 & 0 & a & a & a & a & a & a \\ 0 & 0 & 0 & a & a & a & a & a \\ 0 & 0 & 0 & 0 & a & a & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & b & b & b & b \\ 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & b & b & b & b \\ 0 & 0 & 0 & 0 & 0 & b & b & b \\ 0 & 0 & 0 & 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

The introduced nonzero at positions $(3, 2)$ and $(4, 2)$ in B are annihilated by an opposite Householder matrix acting on columns $2, \ldots, 4$:

$$\left(\left[\begin{array}{c|ccc|cccc} a & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ a & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ \hline 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ \hline 0 & 0 & 0 & 0 & a & a & a & a \end{array}\right], \begin{bmatrix} b & \hat{b} & \hat{b} & \hat{b} & b & b & b & b \\ 0 & \hat{b} & \hat{b} & \hat{b} & b & b & b & b \\ 0 & \hat{0} & \hat{b} & \hat{b} & b & b & b & b \\ 0 & \hat{0} & \hat{b} & \hat{b} & b & b & b & b \\ 0 & 0 & 0 & 0 & b & b & b & b \\ 0 & 0 & 0 & 0 & 0 & b & b & b \\ 0 & 0 & 0 & 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

The bulge resides on the third subdiagonal of A in rows $5,\dots,7$ and columns $2,\dots,4$. It is chased by a Householder matrix applied to rows $5,\dots,7$, followed by an opposite Householder matrix applied to columns $2,\dots,4$ in order to annihilate the unwanted nonzeros in B:

$$\left(\begin{bmatrix} a & a & a & a & a & a & a & a \\ a & a & a & a & a & a & a & a \\ 0 & a & a & a & a & a & a & a \\ 0 & a & a & a & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & 0 & 0 & a & a & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & b & b & b & b \\ 0 & b & b & b & b & b & b & b \\ 0 & 0 & b & b & b & b & b & b \\ 0 & 0 & b & b & b & b & b & b \\ 0 & 0 & 0 & 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b \end{bmatrix}\right),$$

$$\left(\begin{bmatrix} a & a & a & a & \hat{a} & \hat{a} & \hat{a} & a \\ a & a & a & a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & a & a & a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & a & a & a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & a & a & a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & 0 & a & a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & 0 & a & a & \hat{a} & \hat{a} & \hat{a} & a \\ 0 & 0 & 0 & 0 & \hat{a} & \hat{a} & \hat{a} & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & b & b & b & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & 0 & b & b & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & 0 & b & b & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & 0 & 0 & 0 & \hat{b} & \hat{b} & \hat{b} & b \\ 0 & 0 & 0 & 0 & \hat{0} & \hat{b} & \hat{b} & b \\ 0 & 0 & 0 & 0 & \hat{0} & \hat{b} & \hat{b} & b \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b \end{bmatrix}\right).$$

This process is repeatedly applied to annihilate the unwanted subdiagonal entries in columns $2,\dots,7$ of A. The original algorithm proposed by Dackland and Kågström [96] is based on Givens rotations instead of (opposite) Householder matrices. High efficiency is attained by delaying the update of parts which are far off from the diagonal. Extensive numerical experiments in [96] demonstrate that this algorithm combined with Stage 1 outperforms the LAPACK routine `DGGHRD` by a factor 2–3. The use of (opposite) Householder matrices results in some further albeit minor speedup [202].

2.5.2 Multishifts and Bulge Pairs

In Sections 2.3.2 and 2.3.3, we have seen that an implicit shifted QZ iteration can be interpreted as chasing a pair of bulges from the top left corner to the bottom right corner of a matrix pair [360]. Analogous to the QR iteration, see Section 1.5.3, Watkins [353] has shown that the indented shifts of a QZ iteration are the finite eigenvalues of these bulge pairs.

To explain this in more detail, suppose that an implicit shifted QZ iteration with m shifts, Algorithm 11, is applied to a matrix pair $(H,T) \in \mathbb{R}^{n\times n}\times\mathbb{R}^{n\times n}$, where $n > m$. Here, we assume that H is an unreduced Hessenberg matrix and that T is an upper triangular matrix. We do not assume that the complete matrix T is nonsingular; it is sufficient to require the submatrix used for the shift computation ($T(n-m+1:n, n-m+1:n)$) and the submatrix involved

in the introduction of the bulge pair $(T(1:m,1:m))$ to be nonsingular. Let x be a multiple of the first column of the shift polynomial

$$p(HT^{-1}) = (\beta_1 HT^{-1} - \alpha_1 I)\cdots(\beta_m HT^{-1} - \alpha_m I).$$

Note that only the nonsingularity of the mth leading principal submatrix of T is required for the proper definition of x [360].

The *initial bulge pair* is the matrix pair $\left(B_0^{(H)}, B_0^{(T)}\right)$, where

$$B_0^{(H)} = [x(1:m+1), H(1:m+1,1:m)] = \begin{bmatrix} x_1 & h_{11} & \cdots & h_{1m} \\ x_2 & h_{21} & \ddots & \vdots \\ \vdots & & \ddots & h_{mm} \\ x_{m+1} & 0 & & h_{m+1,m} \end{bmatrix},$$

$$B_0^{(T)} = [0, T(1:m+1:1:m)] = \begin{bmatrix} 0 & t_{11} & \cdots & t_{1m} \\ 0 & 0 & \ddots & \vdots \\ \vdots & & \ddots & t_{mm} \\ 0 & 0 & \cdots & 0 \end{bmatrix}.$$

Theorem 2.19 ([353]). *If the mth leading principal submatrix of T is nonsingular then the shifts $(\alpha_1,\beta_1),\ldots,(\alpha_m,\beta_m)$ are the finite eigenvalues of the initial bulge pair $\left(B_0^{(H)}, B_0^{(T)}\right)$ provided that $\beta_1 \neq 0,\ldots,\beta_m \neq 0$.*

Analogous to Section 1.5.4, the computation of x can be related to the pole assignment problem for linear descriptor systems [97].

During the course of a QZ iteration, a bulge pair is created at the top left corners of (H,T) and chased down to the bottom right corners along the first subdiagonals. Let $\left(H^{(j)}, T^{(j)}\right)$ denote the updated matrix pair (H,T) obtained after the bulge pair has been chased $(j-1)$ steps. Then, the *jth bulge pair* $\left(B_j^{(H)}, B_j^{(T)}\right)$ is given by the submatrices

$$\begin{aligned} B_j^{(H)} &= H^{(j)}(j+1:j+m+1, j:j+m), \\ B_j^{(T)} &= T^{(j)}(j+1:j+m+1, j:j+m), \end{aligned}$$

which correspond to the submatrices designated by the entries $\hat{b}_a$ and $\hat{b}_b$ in (2.22).

Theorem 2.20 ([353]). *If the mth leading principal submatrix of T is nonsingular, then the shifts $(\alpha_1,\beta_1),\ldots,(\alpha_m,\beta_m)$ are the finite eigenvalues of the jth bulge pair $\left(B_j^{(H)}, B_j^{(T)}\right)$ provided that $\beta_1 \neq 0,\ldots,\beta_m \neq 0$.*

Theorem 2.20 shows how the shifts are propagated during an implicit shifted QZ iteration. Of course, QZ iterations are not immune to the shift blurring effects described in Sections 1.5.3 and 1.5.4. Only modest values for m, say $m \leq 6$, can be used without losing quadratic convergence in finite-precision arithmetic.

2.5.3 Deflation of Infinite Eigenvalues Revisited

Once again, it should be emphasized that Theorem 2.20 only requires the assumptions that the intended shifts are finite and that the mth leading principal submatrix of T is nonsingular. Zero diagonal entries in other parts of T do not affect the information contained in the bulge pairs and do consequently not affect the convergence of the QZ iteration. What happens to such a zero diagonal entry if a bulge pair passes through it? This question has been addressed by Ward [344, 345] for $m \leq 2$, and by Watkins for general m [353]. The answer is that the zero diagonal entry moves m positions upwards along the diagonal.

To see this for Algorithm 11, the QZ iteration based on (opposite) Householder matrices, let us partition the $(m+1) \times (m+2)$ submatrices of $H^{(j)}$ and $T^{(j)}$, which contain the jth bulge pair in the leading $m+1$ columns, as follows:

$$H^{(j)}(j+1:j+m+1, j:j+m+1) = \begin{array}{c} \\ 1 \\ m-1 \\ 1 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & m & 1 \end{array} \\ \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \end{array},$$

$$T^{(j)}(j+1:j+m+1, j:j+m+1) = \begin{array}{c} \\ 1 \\ m-1 \\ 1 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & m & 1 \end{array} \\ \begin{bmatrix} 0 & B_{12} & B_{13} \\ 0 & B_{22} & B_{23} \\ 0 & 0 & 0 \end{bmatrix} \end{array}.$$

Here, the $(j+m+1)$th diagonal entry of $T^{(j)}$ is zero and we are interested in proving that this zero hops to the first entry of B_{12} after the bulge has been chased downwards.

The following assertions hold. The unreducedness of H implies that A_{31}, the tip of the bulge, must be nonzero [353]. The $m \times m$ matrix $\begin{bmatrix} B_{12} \\ B_{22} \end{bmatrix}$ must be nonsingular, otherwise the jth bulge pair contains less than m finite eigenvalues, which contradicts Theorem 2.20.

If the bulge pair is chased downwards, then first a Householder matrix $I - \beta v v^T$ is constructed such that the vector $\left[A_{11}^T, A_{21}^T, A_{31}^T\right]^T$ is mapped to a multiple of the unit vector. Let us partition $v = \left[v_1^T, v_2^T, v_3^T\right]^T$, where $v_1, v_3 \in \mathbb{R}$ and $v_2 \in \mathbb{R}^{m-1}$. Then $A_{31} \neq 0$ implies $v_1 \neq 0$, $v_2 \neq 0$ and $\beta \neq 0$, see (1.41)–(1.42). Applying the Householder matrix $I - \beta v v^T$ from the left to $T^{(j)}(j+1 : j+m+1, j : j+m+1)$ yields the following matrix:

$$\begin{bmatrix} 0 & \tilde{B}_{12} & \tilde{B}_{13} \\ 0 & \tilde{B}_{22} & \tilde{B}_{23} \\ 0 & \tilde{B}_{32} & \tilde{B}_{33} \end{bmatrix} = \begin{bmatrix} 0 & B_{12} - v_1 w^T & B_{13} - v_1 z^T \\ 0 & B_{22} - v_2 w^T & B_{23} - v_2 z^T \\ 0 & -v_3 w^T & -v_3 z^T \end{bmatrix},$$

where $w^T = \beta(v_1 B_{12} + v_2^T B_{22})$ and $z^T = \beta(v_1 B_{13} + v_2^T B_{23})$. Note that the matrix $\begin{bmatrix} \tilde{B}_{22} \\ \tilde{B}_{32} \end{bmatrix}$ must be invertible. Otherwise, there exists a vector $\begin{bmatrix} a \\ b \end{bmatrix} \neq 0$ so

that $\left[\tilde{B}_{22}^T, \tilde{B}_{32}^T\right] \left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right] = 0$, which implies

$$B_{22}^T a - \underbrace{(v_2^T a + v_3 b)}_{=:\gamma} w = 0 \;\Rightarrow\; B_{22}^T(a - \beta\gamma v_2) - (\beta\gamma v_1) B_{12}^T = 0.$$

Thus, either $\gamma = 0$ implying that $a \neq 0$ and $B_{22}^T a = 0$, or $\gamma \neq 0$ implying that $B_{12}^T = 1/(\beta\gamma v_1) \cdot B_{22}^T(a - \beta\gamma v_2)$. In both cases, there is a contradiction to the assertion that $\left[\begin{smallmatrix} B_{12} \\ B_{22} \end{smallmatrix}\right]$ is a nonsingular matrix.

In the next step of the bulge chasing process, an opposite Householder matrix is constructed such that the first column of the $(m+1) \times (m+1)$ matrix

$$\tilde{B} = \begin{bmatrix} \tilde{B}_{12} & \tilde{B}_{13} \\ \tilde{B}_{22} & \tilde{B}_{23} \\ \tilde{B}_{32} & \tilde{B}_{33} \end{bmatrix}$$

is mapped to a multiple of the first unit vector. If we apply an RQ decomposition to $\tilde{B}$, then the first column of the R-factor is zero due to the facts that $\tilde{B}$ is a singular matrix and that the last m rows of $\tilde{B}$ have full rank [48]. Hence, the opposite Householder matrix maps the first column of $\tilde{B}$ to zero, see (2.23). This proves the desired result that the zero entry which initially resides at the $(j+m+1)$th diagonal position of $T^{(j)}$ hops to the $(j+1)$th position.

In principle, this result implies that we could just let the infinite eigenvalues be pushed upwards by QZ iterations and deflate them at top left corner of the matrix pair. In practice, such an approach makes it more difficult to keep track of identified infinite eigenvalues in finite-precision arithmetic, in particular for matrix pairs of higher index.

2.5.4 Tightly Coupled Tiny Bulge Pairs

As for the QR algorithm, shift blurring effects in the QZ algorithm can be avoided by using a tightly coupled chain of tiny bulge pairs instead of one large bulge during a QZ iteration. Such an approach has been developed in [180] based on preliminary results in [3]. For the purpose of describing this approach, let m denote the number of simultaneous shifts to be used in each QZ iteration and let n_s denote the number of shifts contained in each bulge pair. It is assumed that m is an integer multiple of n_s. We assume n_s to be tiny, e.g., $n_s \in [2, 6]$.

Introducing a chain of bulge pairs

The tiny-bulge multishift QZ algorithm begins with introducing m/n_s bulge pairs in the top left corner of the matrix pair (H, T) in Hessenberg-triangular form. Every bulge pair contains a set of n_s shifts. For a proper propagation of shifts, it is necessary that the $((m/n_s)(n_s+1)-1)$th leading principal

submatrix of T is nonsingular. The first bulge pair is introduced by applying an implicit QZ iteration with n_s shifts and interrupting the bulge chasing process as soon as the bottom right corner of the bulge in H touches the $(p_h - 1, p_h)$ subdiagonal entry of H, where $p_h = (m/n_s)(n_s + 1) + 1$. The next bulge pair is chased until the bottom right corner of the bulge in H touches the $(p_h - n_s - 2, p_h - n_s - 1)$ subdiagonal entry. This process is continued until all m/n_s bulge pairs are introduced, see Figure 2.1. Note that only

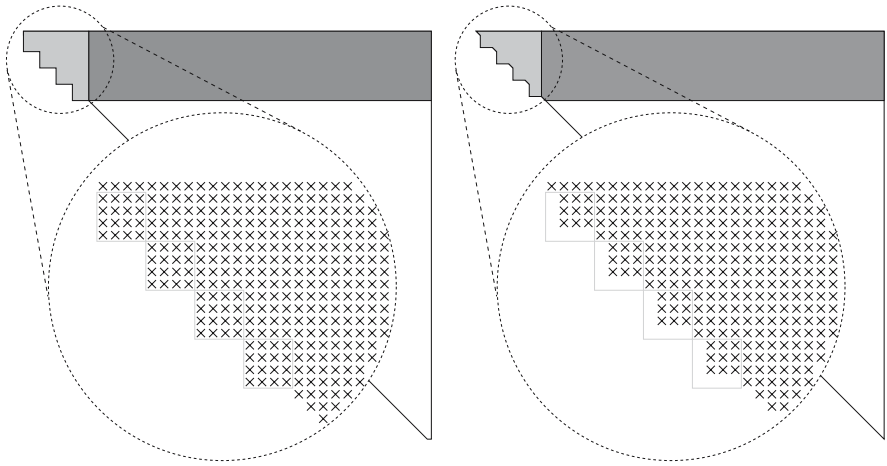


Fig. 2.1. Introducing a chain of $m/n_s = 4$ tightly coupled bulge pairs, each of which contains $n_s = 3$ shifts.

the submatrices painted pale gray in Figure 2.1 must be updated during the bulge chasing process. To update the remaining parts (painted dark gray), all orthogonal transformations from the left are accumulated into a $p_h \times p_h$ matrix U and applied in terms of matrix-matrix multiplications:

$$H(1 : p_h, (p_h + 1) : n) \leftarrow U^T H(1 : p_h, (p_h + 1) : n),$$
$$T(1 : p_h, (p_h + 1) : n) \leftarrow U^T T(1 : p_h, (p_h + 1) : n).$$

Chasing a chain of bulge pairs

In each step of the tiny-bulge multishift QZ algorithm, the chain of bulge pairs, which resides in columns/rows $p_l : p_h$ of (H, T), is chased k steps downwards. This is done in a bulge-by-bulge and bottom-to-top fashion, as described in Section 1.5.5. One such step is illustrated in Figure 2.2. Again, only the principal submatrices painted pale gray in Figure 2.2 must be updated during the bulge chasing process. All transformations from the left and from the right are accumulated in orthogonal matrices U and V, respectively. Then,

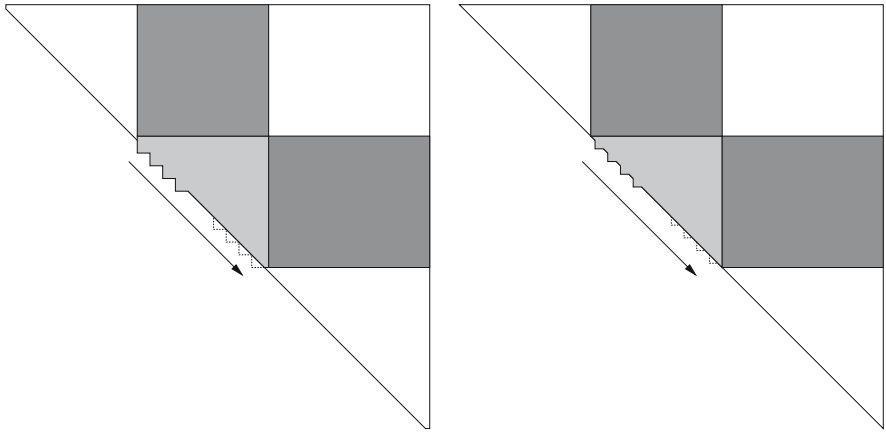


Fig. 2.2. Chasing a chain of $m/n_s = 4$ tightly coupled bulge pairs.

matrix-matrix multiplications can be used to update the rest of the matrix pair (painted dark gray in Figure 2.2):

$$\begin{aligned} H(p_l : p_h + k, (p_h + 1) : n) &\leftarrow U^T H(p_l : p_h + k, (p_h + 1) : n), \\ T(p_l : p_h + k, (p_h + 1) : n) &\leftarrow U^T T(p_l : p_h + k, (p_h + 1) : n), \\ H(1 : p_l - 1, p_l : p_h + k) &\leftarrow H(1 : p_l - 1, p_l : p_h + k)\, V. \\ T(1 : p_l - 1, p_l : p_h + k) &\leftarrow T(1 : p_l - 1, p_l : p_h + k)\, V. \end{aligned}$$

Note that both matrices, U and V, have the following block structure:

$$\begin{array}{c} \\ 1 \\ l_1 \\ l_2 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & l_2 & l_1 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & \square & \triangleright \\ 0 & \triangleright & \square \end{bmatrix} \end{array},$$

where $l_1 = (m/n_s)(n_s + 1) - n_s$ and $l_2 = k + n_s$. Exploiting this structure can have a positive effect on the efficiency of matrix-matrix multiplications involving U or V.

As for the tiny-bulge multishift QR algorithm, we have to be aware of vigilant deflations, i.e., zero or tiny subdiagonal elements in H. In order to preserve the information contained in the bulge pairs, the chain of bulge pairs must be reintroduced in the row in which the zero appears. Fortunately, we have not to be wary of zero or tiny subdiagonal elements in T, since the bulge pairs are properly passed through infinite eigenvalues, see Section 2.5.3.

Getting rid off a chain of bulge pairs

Once the bottom bulge pair of the chain has reached the bottom right corners of the matrix pair, the whole chain is bulge-by-bulge chased off this corner, similarly to the introduction of bulge pairs.

Numerical results

To illustrate the achievable efficiency improvements, the described tiny-bulge multishift QZ algorithm has been implemented in a Fortran 77 routine called `MTTQZ`. We applied `MTTQZ` to randomly generated matrix pairs of order $400, 450, \ldots, 1800$. The matrix pairs were obtained by reducing full matrix pairs, with entries uniformly distributed in the interval $[0, 1]$, to Hessenberg-triangular form. The parameters m (number of shifts in each iteration) and k (number of steps a chain of bulge pairs is chased before off-diagonal parts are updated) were set to

$$m = \begin{cases} 48, \text{ if } n < 1000, \\ 60, \text{ if } 1000 \le n < 2000, \end{cases}$$

and $k = 3/2 \cdot m - 2$. For comparison, we applied the LAPACK routine `DHGEQZ` and the routine `KDHGEQZ`, an implementation of the pipelined QZ algorithm by Dackland and Kågström [96]. In the latter routine, we used the block size $n_b = 48$ which was found to be nearly optimal. The obtained cpu times relative to the cpu time needed by `DHGEQZ` are displayed in Figure 2.3. It can be seen that `MTTQZ` requires up to 80% less cpu time than `DHGEQZ` for sufficiently large matrix pairs. The improvement upon `KDHGEQZ` is much less significant. In fact, `KDHGEQZ` outperforms `MTTQZ` for matrix pairs of order smaller than 500.

2.6 Aggressive Early Deflation

Aggressive early deflation, see Section 1.6, can be adapted to the QZ algorithm in a considerably straightforward manner [3, 180]. Consider a matrix pair (A, B) in Hessenberg-triangular form, partitioned as follows:

$$A = \begin{array}{c} \\ n-w-1 \\ 1 \\ w \end{array} \begin{array}{c} \begin{array}{ccc} n-w-1 & 1 & w \end{array} \\ \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ 0 & A_{32} & A_{33} \end{bmatrix} \end{array}, \quad B = \begin{array}{c} \\ n-w-1 \\ 1 \\ w \end{array} \begin{array}{c} \begin{array}{ccc} n-w-1 & 1 & w \end{array} \\ \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ 0 & B_{22} & B_{23} \\ 0 & 0 & B_{33} \end{bmatrix} \end{array}.$$

Our goal is to construct a correspondingly partitioned perturbation of the form

$$E = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & E_{32} & 0 \end{bmatrix} \tag{2.29}$$

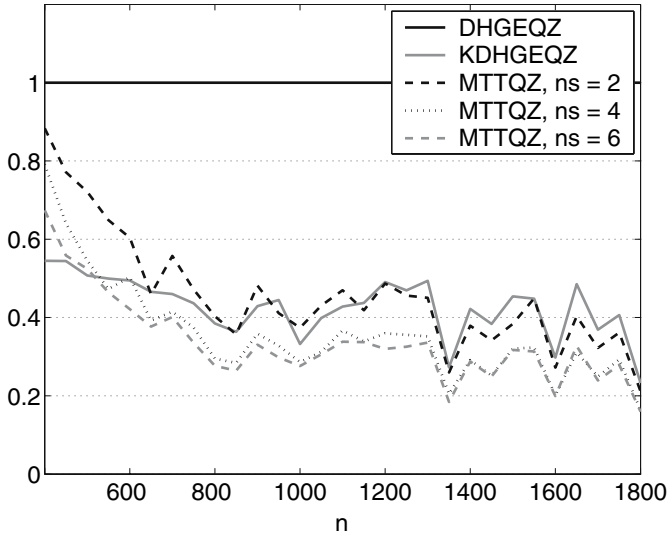


Fig. 2.3. Performance of the tiny-bulge multishift QZ algorithm (`MTTQZ`) relative to the LAPACK routine `DHGEQZ`.

so that the reduction of $(A + E, B)$ to Hessenberg-triangular form results in a deflation, in which case E is called a *reducing perturbation*.

This can be achieved by computing (reordered) generalized Schur decompositions of the matrix pair (A_{33}, B_{33}). First, orthogonal matrices U, V are constructed so that $(S_{33}, T_{33}) = U^T(A_{33}, B_{33})V$ is in real generalized Schur form. Then, we partition

$$(S_{33}, T_{33}) = \left(\begin{bmatrix} \tilde{S}_{11} & \tilde{S}_{12} \\ 0 & \tilde{S}_{22} \end{bmatrix}, \begin{bmatrix} \tilde{T}_{11} & \tilde{T}_{12} \\ 0 & \tilde{T}_{22} \end{bmatrix} \right),$$

so that $(\tilde{S}_{22}, \tilde{T}_{22})$ is either a real generalized eigenvalue or a 2×2 matrix pair containing a complex conjugate pair of generalized eigenvalues. The vector $U^T A_{32} = \begin{bmatrix} \tilde{s}_1 \\ \tilde{s}_2 \end{bmatrix}$ is correspondingly partitioned. Then the matrix $E_{32} = -U \begin{bmatrix} 0 \\ \tilde{s}_2 \end{bmatrix}$ corresponds to a reducing perturbation E of the form (2.29). The Frobenius norm of E is given by $\|\tilde{s}_2\|_2$. We make use of this perturbation if $\tilde{s}_2$ satisfies

$$\|\tilde{s}_2\|_\infty \leq \sqrt{|\det(\tilde{S}_{22})|}. \tag{2.30}$$

This choice preserves the backward stability of the QZ algorithm. A variety of other criteria can be developed in accordance to the criteria presented in [62].

If the current ordering of the matrix pair (S_{33}, T_{33}) fails to fulfill the criterion (2.30), we may test other possible choices for $(\tilde{S}_{22}, \tilde{T}_{22})$ by reordering

the generalized Schur form, see Section 2.7. If it happens, however, that a reducing perturbation satisfying (2.30) has been found, the described procedure is repeatedly applied to the unperturbed part consisting of $\tilde{s}_1$ and $(\tilde{S}_{11}, \tilde{T}_{11})$. Eventually, orthogonal matrices Q and Z are constructed so that

$$(I_{n-w} \oplus Q)^T A (I_{n-w} \oplus Z) = \begin{array}{c} \\ n-w-1 \\ 1 \\ w-d \\ d \end{array} \begin{array}{c} \begin{array}{cccc} n-w-1 & 1 & w-d & d \end{array} \\ \begin{bmatrix} A_{11} & A_{12} & \tilde{A}_{13} & \tilde{A}_{13} \\ A_{21} & A_{22} & \tilde{A}_{23} & \tilde{A}_{24} \\ 0 & s_1 & S_{11} & S_{12} \\ 0 & s_2 & 0 & S_{22} \end{bmatrix} \end{array},$$

$$(I_{n-w} \oplus Q)^T B (I_{n-w} \oplus Z) = \begin{array}{c} \\ n-w-1 \\ 1 \\ w-d \\ d \end{array} \begin{array}{c} \begin{array}{cccc} n-w-1 & 1 & w-d & d \end{array} \\ \begin{bmatrix} B_{11} & B_{12} & \tilde{B}_{13} & \tilde{B}_{13} \\ 0 & B_{22} & \tilde{B}_{23} & \tilde{B}_{24} \\ 0 & 0 & T_{11} & T_{12} \\ 0 & 0 & 0 & T_{22} \end{bmatrix} \end{array},$$

where (S_{11}, T_{11}), (S_{22}, T_{22}) are in real generalized Schur form and the entries of the vector s_2 satisfy criteria of the form (1.60). Setting s_2 to zero amounts to deflating the d generalized eigenvalues contained in (S_{22}, T_{22}). The remaining unreduced matrix pair can be cheaply returned to Hessenberg-triangular form as follows. First, the Householder matrix $H_1(s_1) = I - \beta v v^T$ is applied from the left to $[s_1, S_{11}]$ and T_{11}. Next, the matrix $T_{11} - \beta T v v^T$ is reduced to triangular form using Algorithm 12 (update of RQ decomposition). The matrix S_{11} is updated with the transformations resulting from Algorithm 12. Finally, applying Algorithm 9, reduction to Hessenberg-triangular form, to the updated matrix pair (S_{11}, T_{11}) returns the whole matrix pair (A, B) to Hessenberg-triangular form.

Aggressive early deflation is performed after each multishift QZ iteration. It must be combined with the conventional deflation strategy described in Section 2.3.4.

Numerical results

Aggressive early deflation combined with the tiny-bulge multishift QZ algorithm has been implemented in a Fortran 77 routine called `ATTQZ`. We have applied `ATTQZ` to the random matrix pairs used for the numerical results presented in Section 2.5.4. Also, the employed routine parameters were identical to those used in Section 2.5.4. The size of the deflation window was chosen to be $w = 3m/2$, where m denotes the number of simultaneous shifts used in each iteration.

We have also combined the pipelined QZ algorithm [96] with aggressive early deflation and called this routine `ADHGEQZ`. Here, the number of simultaneous shifts is at most two. Our observation was that the performance of this method is very sensitive to the size of the deflation window. We therefore

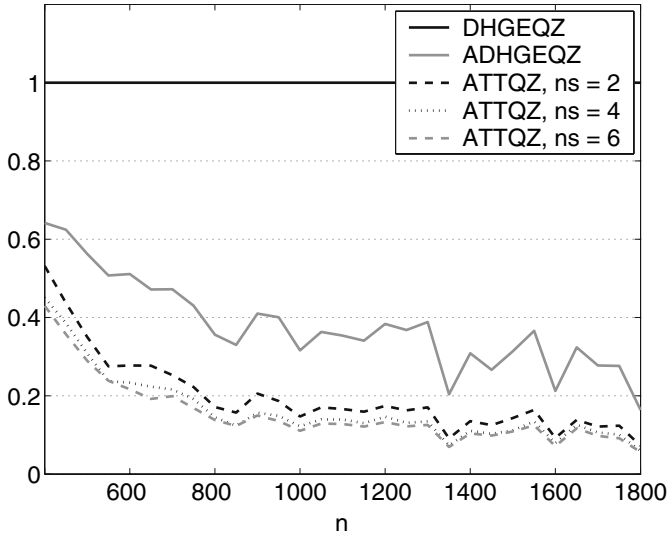


Fig. 2.4. Performance of the tiny-bulge multishift QZ algorithm with aggressive early deflation (`ATTQZ`) relative to the LAPACK routine `DHGEQZ`.

selected an "optimal" window size from the set $\{10, 20, 30, 40, 50\}$ for each matrix pair.

The cpu times displayed in Figure 2.4 show that the use of aggressive early deflation leads to substantial improvements not only in comparison with the LAPACK routine `DHGEQZ` but also in comparison with the routine `MTTQZ`, see Figure 2.3. Also, aggressive early deflation has a greater effect on the performance of the tiny-bulge multishift QZ algorithm than on the performance of the pipelined QZ algorithm. Numerical experiments involving practically relevant test matrix pairs essentially confirm these findings although the picture is less incisive due to the fact that some applications, especially from mechanics, lead to very badly scaled matrix pairs [180].

2.7 Computation of Deflating Subspaces

Suppose our favorite variant of the QZ algorithm has successfully computed a real generalized Schur decomposition of $Q^T(A, B)Z = (S, T)$ with

$$(S,T) = \left(\begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1m} \\ 0 & S_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & S_{m-1,m} \\ 0 & \cdots & 0 & S_{mm} \end{bmatrix}, \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1m} \\ 0 & T_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_{m-1,m} \\ 0 & \cdots & 0 & T_{mm} \end{bmatrix} \right) \tag{2.31}$$

where $S_{ii}, T_{ii} \in \mathbb{R}^{n_i \times n_i}$ with $n_i = 1$ if $\lambda(S_{ii}, T_{ii}) \subset \mathbb{R}$ and $n_i = 2$ otherwise. For each $j \leq m$, the first $k = \sum_{i=1}^{j} n_i$ columns of Q and Z form orthonormal bases for left and right deflating subspaces belonging to the eigenvalues $\Lambda_j = \lambda(S_{11}, T_{11}) \cup \cdots \cup \lambda(S_{jj}, T_{jj})$.

As for the QR algorithm, it is generally impossible to guarantee a certain order of the diagonal blocks in the generalized Schur form returned by the QZ algorithm. Post-processing must be applied to obtain deflating subspaces belonging to a specific cluster of eigenvalues $\Lambda_s \subset \lambda(A, B)$ closed under complex conjugation. From a given decomposition (1.61) a reordered generalized Schur decomposition is computed such that $(Q\tilde{Q})^T(A, B)(Z\tilde{Z}) = (\tilde{S}, \tilde{T})$ with

$$(\tilde{S}, \tilde{T}) = \left(\begin{bmatrix} \tilde{S}_{11} & \tilde{S}_{12} & \cdots & \tilde{S}_{1m} \\ 0 & \tilde{S}_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \tilde{S}_{m-1,m} \\ 0 & \cdots & 0 & \tilde{S}_{mm} \end{bmatrix}, \begin{bmatrix} \tilde{T}_{11} & \tilde{T}_{12} & \cdots & \tilde{T}_{1m} \\ 0 & \tilde{T}_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \tilde{T}_{m-1,m} \\ 0 & \cdots & 0 & \tilde{T}_{mm} \end{bmatrix} \right), \tag{2.32}$$

where $\Lambda_s = \lambda(\tilde{S}_{11}, \tilde{T}_{11}) \cup \cdots \lambda(\tilde{S}_{jj}, \tilde{T}_{jj})$ for some $1 \leq j \leq m$.

Van Dooren [328], Kågström [179], as well as Kågström and Poromaa [182] have developed algorithms that achieve such a reordering. The building block of these algorithms is the computation of orthogonal matrices Q and Z such that

$$Q^T \left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} \right) Z = \left(\begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ 0 & \tilde{A}_{22} \end{bmatrix}, \begin{bmatrix} \tilde{B}_{11} & \tilde{B}_{12} \\ 0 & \tilde{B}_{22} \end{bmatrix} \right) \tag{2.33}$$

and

$$\lambda(A_{11}, B_{11}) = \lambda(\tilde{A}_{22}, \tilde{B}_{22}), \quad \lambda(A_{22}, B_{22}) = \lambda(\tilde{A}_{11}, \tilde{B}_{11}),$$

where $A_{11}, B_{11} \in \mathbb{R}^{n_1 \times n_1}$ and $B_{22}, A_{22} \in \mathbb{R}^{n_2 \times n_2}$ with $n_1, n_2 \in \{1, 2\}$.

The algorithm proposed in [328] for performing the swapping (2.33) is in the spirit of Stewart's algorithm for reordering standard Schur forms [303] and employs QZ iterations with perfect shifts. It also inherits the limitations of Stewart's approach, i.e., failure of convergence or convergence to a block triangular matrix pair that does not correspond to the intended swapping. The algorithm proposed in [179, 182], which will be described in the following, is in the spirit of the work by Bai and Demmel [18] and employs generalized Sylvester equations.

Assuming that $\lambda(A_{11}, B_{11}) \cap \lambda(A_{22}, B_{22}) = \emptyset$, we first compute the solution of the generalized Sylvester equation

$$\begin{aligned} A_{11}X - YA_{22} &= \gamma A_{12}, \\ B_{11}X - YB_{22} &= \gamma B_{12}, \end{aligned} \tag{2.34}$$

where $\gamma \in (0, 1]$ is a scaling factor to prevent overflow in X and Y. This yields the following block diagonal decompositions:

$$\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} = \begin{bmatrix} I_{n_1} & -Y \\ 0 & \gamma I_{n_2} \end{bmatrix} \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} I_{n_1} & X/\gamma \\ 0 & I_{n_2}/\gamma \end{bmatrix},$$
$$\begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} = \begin{bmatrix} I_{n_1} & -Y \\ 0 & \gamma I_{n_2} \end{bmatrix} \begin{bmatrix} B_{11} & 0 \\ 0 & B_{22} \end{bmatrix} \begin{bmatrix} I_{n_1} & X/\gamma \\ 0 & I_{n_2}/\gamma \end{bmatrix}.$$

By QR decompositions, orthogonal matrices Q and Z are constructed such that

$$Q^T \begin{bmatrix} -Y \\ \gamma I_{n_2} \end{bmatrix} = \begin{bmatrix} R_Y \\ 0 \end{bmatrix}, \quad Z^T \begin{bmatrix} -X \\ \gamma I_{n_2} \end{bmatrix} = \begin{bmatrix} R_X \\ 0 \end{bmatrix},$$

with upper triangular matrices $R_X, R_Y \in \mathbb{R}^{n_2 \times n_2}$. If we partition $Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}$ and $Z = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{bmatrix}$ so that $Q_{12}, Z_{12} \in \mathbb{R}^{n_1 \times n_1}$ then both Q_{12} and Z_{12} are invertible. Moreover,

$$\begin{aligned} Q^T \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} Z &= \begin{bmatrix} \star & R_Y \\ Q_{12}^T & 0 \end{bmatrix} \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} 0 & Z_{12}^{-T} \\ R_X^{-1} & \star \end{bmatrix} \\ &= \begin{bmatrix} R_Y A_{22} R_X^{-1} & \star \\ 0 & Q_{12}^T A_{11} Z_{12}^{-T} \end{bmatrix}, \\ Q^T \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} Z &= \begin{bmatrix} \star & R_Y \\ Q_{12}^T & 0 \end{bmatrix} \begin{bmatrix} B_{11} & 0 \\ 0 & B_{22} \end{bmatrix} \begin{bmatrix} 0 & Z_{12}^{-T} \\ R_X^{-1} & \star \end{bmatrix} \\ &= \begin{bmatrix} R_Y B_{22} R_X^{-1} & \star \\ 0 & Q_{12}^T B_{11} Z_{12}^{-T} \end{bmatrix}. \end{aligned}$$

Hence, the matrices Q and Z produce the desired swapping (2.33).

To consider the finite-precision aspects of the described procedure, let $\hat{X}$ and $\hat{Y}$ denote the *computed* solutions of the generalized Sylvester equation (2.34). Then

$$\hat{Q}^T \left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} \right) \hat{Z} = \left(\begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} \\ \hat{A}_{21} & \hat{A}_{22} \end{bmatrix}, \begin{bmatrix} \hat{B}_{11} & \hat{B}_{12} \\ \hat{B}_{21} & \hat{B}_{22} \end{bmatrix} \right),$$

where

$$\begin{aligned} \|\hat{A}_{21}\|_2 &\leq \frac{\|A_{11}X - YA_{22} - \gamma A_{12}\|_F}{(1+\sigma_{\min}(X))^{1/2}(1+\sigma_{\min}(Y))^{1/2}}, \\ \|\hat{B}_{21}\|_2 &\leq \frac{\|B_{11}X - YB_{22} - \gamma B_{12}\|_F}{(1+\sigma_{\min}(X))^{1/2}(1+\sigma_{\min}(Y))^{1/2}}, \end{aligned}$$

see [179]. Let us assume that $\hat{X}$ and $\hat{Y}$ have been obtained from a backward stable solver for linear systems applied to the Kronecker product formulation of (2.34), see also Section 2.2.1. Then the residuals $\|A_{11}X - YA_{22} - \gamma A_{12}\|_F$ and $\|B_{11}X - BA_{22} - \gamma B_{12}\|_F$ can be approximately bounded by $\mathbf{u}(\|A\|_F + \|B\|_F)(\|X\|_2 + \|Y\|_2 + \gamma)$. A sufficiently large value for $\mathrm{dif}_u[(A_{11}, B_{11}), (A_{22}, B_{22})]$ yields moderate norms for X and Y which in turn implies that $\hat{A}_{21}$ and $\hat{B}_{21}$ can be safely set to zero. A small value for dif_u may lead to error matrices

$\hat{A}_{21}, \hat{B}_{21}$ that cannot be set to zero without sacrificing numerical backward stability. Numerical experiments reported in [182, 183] demonstrate that this effect rarely occurs in practice. In the LAPACK routine `DTGEX2`, an implementation of the described swapping procedure, the swapping is performed tentatively and is rejected if any of the entries in $\hat{A}_{21}$ or $\hat{B}_{21}$ is too large.

Along the lines of Algorithm 8, the LAPACK routine `DTGEXC` applies `DTGEX2` in a bubble sort fashion to compute a reordered generalized Schur decomposition of the form (2.32). A block variant of `DTGEXC` can be derived using the same techniques as in Section 1.7.3, see also [203].

3

The Krylov-Schur Algorithm

So far, all discussed algorithms for computing eigenvalues and invariant subspaces are based on orthogonal transformations of the matrix, making them suitable only for small to medium-sized eigenvalue problems. Nevertheless, these algorithms play an important role in methods designed for the eigenvalue computation of large and possibly sparse matrices, such as Arnoldi, Lanczos or Jacobi-Davidson methods; see, e.g., the eigenvalue templates book [19] for a comprehensive overview.

In this short chapter, we focus on a descendant of the Arnoldi method, the recently introduced Krylov-Schur algorithm by Stewart [307]. This algorithm belongs to the class of Krylov subspace methods. It generates a sequence of subspaces containing approximations to a desired subset of eigenvectors and eigenvalues of a matrix A. These approximations are extracted by applying the QR algorithm to the projection of A onto the subspaces. If the approximations are not accurate enough then a reordering of the Schur decomposition computed in the previous step can be used to restart the whole process. We have chosen this algorithm mainly because of its close relationship to the other algorithms described in this book. Furthermore, as will be explained in the next chapter, the Krylov-Schur algorithm admits a rather simple extension to some structured eigenvalue problems such as product and (skew-)Hamiltonian eigenvalue problems.

This chapter is organized as follows. In the next section below, we introduce the notions of Krylov subspaces as well as their relation to Arnoldi decompositions and the Arnoldi method. Section 3.2 is concerned with some two important ingedients aiming to make the Arnoldi method more efficient and more reliable: restarts and deflations. Finally in Section 3.3, we discuss balancing strategies for sparse matrices.

3.1 Basic Tools

The purpose of this section is to briefly summarize some well-known material related to Krylov subspaces, Arnoldi decompositions and the Arnoldi method. A more comprehensive treatment of this subject can be found, e.g., in Saad's book [281].

3.1.1 Krylov Subspaces

First, let us define a Krylov subspace and recall some of its elementary properties.

Definition 3.1. *Let $A \in \mathbb{R}^{n\times n}$ and $u \in \mathbb{R}^n$ with $u \neq 0$, then*

$$K_k(A,u) = \begin{bmatrix} u, & Au, & A^2u, & \ldots, & A^{k-1}u \end{bmatrix}$$

is called the kth Krylov matrix *associated with A and u. The corresponding subspace*

$$\mathcal{K}_k(A,u) = \operatorname{span}\{u, Au, A^2u, \ldots, A^{k-1}u\}$$

is called the kth Krylov subspace *associated with A and u.*

Lemma 3.2. *Let $A \in \mathbb{R}^{n\times n}$ and $u \in \mathbb{R}^n$ with $u \neq 0$, then*

1. $\mathcal{K}_k(A,u) = \{p(A)u \mid p \text{ is a polynomial of degree at most } k-1\}$,
2. $\mathcal{K}_k(A,u) \subseteq \mathcal{K}_{k+1}(A,u)$,
3. $A\mathcal{K}_k(A,u) \subseteq \mathcal{K}_{k+1}(A,u)$,
4. $\mathcal{K}_l(A,u) = \mathcal{K}_{l+1}(A,u)$ *implies* $\mathcal{K}_l(A,u) = \mathcal{K}_k(A,u)$ *for all* $k \geq l$.

The last part of Lemma 3.2 covers an exceptional situation. The relation $\mathcal{K}_{l+1}(A,u) = \mathcal{K}_l(A,u)$ also implies, because of $A\mathcal{K}_l(A,u) \subseteq \mathcal{K}_{l+1}(A,u)$, that $\mathcal{K}_l(A,u)$ is an invariant subspace of A. If, for example, A is an upper Hessenberg matrix and $u = e_1$, then the smallest l for which $\mathcal{K}_l(A,u) = \mathcal{K}_{l+1}(A,u)$ may happen is either $l = n$, if A is unreduced, or the smallest l for which the $(l+1,l)$ subdiagonal entry of A is zero.

Although hitting an exact invariant subspace by Krylov subspaces of order lower than n is a rare event, it is often true that already Krylov subspaces of low order contain good approximations to eigenvectors or invariant subspace belonging to extremal eigenvalues.

Example 3.3 ([305, Ex. 4.3.1]). Let $A = \operatorname{diag}(1, 0.95, 0.95^2, \ldots,)$ and let u be a random vector of length n. The solid black line in Figure 3.1 shows the tangent of the angle between e_1, the eigenvector belonging to the dominant eigenvalue 1, and $\mathcal{K}_k(A,u)$ for $k = 1, \ldots, 25$. The dash-dotted line shows the tangent of the largest canonical angle between $\operatorname{span}\{e_1, e_2\}$, the invariant subspace belonging to $\{1, 0.95\}$, and $\mathcal{K}_k(A,u)$, while the dotted line shows the corresponding figures for $\operatorname{span}\{e_1, e_2, e_3\}$. In contrast, the tangents of the angle between e_1 and the vectors $A^{k-1}u$, which are generated by the power method, are displayed by the gray curve. ◊

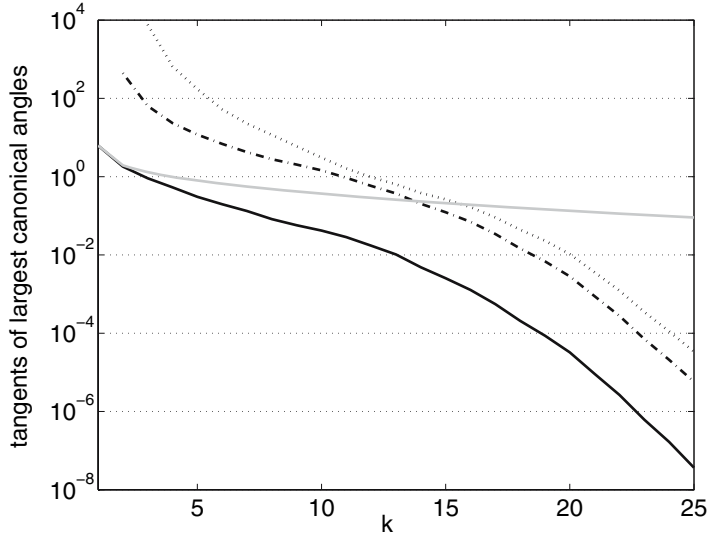


Fig. 3.1. Approximations of eigenvectors and invariant subspaces by Krylov subspaces.

The approximation properties of Krylov subspaces are a well-studied but not completely understood subject of research, which in its full generality goes beyond the scope of this treatment. Only exemplarily, we provide one of the simplest results in this area for symmetric matrices. A more detailed discussion can be found in Parlett's book [261] and the references therein.

Theorem 3.4 ([305, Thm. 4.3.8]). *Let* $A \in \mathbb{R}^{n\times n}$ *be* symmetric *and let* $x_1, \ldots, x_n$ *be an orthonormal set of eigenvectors belonging to the eigenvalues* $\lambda_1 > \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_n$. *Let* $\eta = (\lambda_1 - \lambda_2)/(\lambda_2 - \lambda_n)$, *then*

$$\tan\theta_1(x_1, \mathcal{K}_k(A,u)) \leq \frac{\tan\angle(x_1,u)}{(1+2\sqrt{\eta+\eta^2})^{k-1} + (1+2\sqrt{\eta+\eta^2})^{1-k}}. \tag{3.1}$$

For large k, the bound (3.1) simplifies to

$$\tan\theta_1(x_1, \mathcal{K}_k(A,u)) \lesssim \frac{\tan\angle(x_1,u)}{(1+2\sqrt{\eta+\eta^2})^{k-1}}.$$

In other words, adding a vector to the Krylov subspace reduces the angle between x_1 and $\mathcal{K}_k(A,u)$ by a factor of $(1+2\sqrt{\eta+\eta^2})$. Furthermore, not only the eigenvector x_1 is approximated; there are similar bounds for eigenvectors belonging to smaller eigenvalues. For example, let $\eta_2 = (\lambda_2 - \lambda_3)/(\lambda_3 - \lambda_n)$, then

$$\tan\theta_1(x_2, \mathcal{K}_k(A,u)) \leq \frac{\lambda_n - \lambda_1}{\lambda_2 - \lambda_1} \frac{\tan\angle(x_2,u)}{(1+2\sqrt{\eta_2+\eta_2^2})^{k-2} + (1+2\sqrt{\eta_2+\eta_2^2})^{2-k}}.$$

The convergence theory for nonsymmetric matrices is complicated by the facts that the geometry of eigenvectors or invariant subspaces is more involved and there may not be a complete set of eigenvectors. Existing results indicate that, under suitable conditions, Krylov subspaces tend to approximate eigenvectors belonging to eigenvalues at the boundary of the convex hull of the spectrum, see e.g. [279]. For more recent work in this area, see [27, 28].

3.1.2 The Arnoldi Method

An explicit Krylov basis of the form

$$K_k(A, u) = \begin{bmatrix} u, Au, A^2u, \ldots, A^{k-1}u \end{bmatrix}$$

is not suitable for numerical computations. As k increases, the vectors $A^k u$ almost always converge to an eigenvector belonging to the dominant eigenvalue. This implies that the columns of $K_k(A, u)$ become more and more linearly dependent. Often, the condition number of the matrix $K_k(A, u)$ grows exponentially with k. Hence, a large part of the information contained in $K_k(A, u)$ is getting corrupted by roundoff errors.

The Arnoldi Decomposition

To avoid the described effects, one should choose a basis of a better nature, for example an orthonormal basis. However, explicitly orthogonalizing the column vectors of $K_k(A, u)$ is no remedy; once constructed, the basis $K_k(A, u)$ has already suffered loss of information in finite-precision arithmetic. To avoid this, we need a way to implicitly construct an orthonormal basis for $\mathcal{K}_k(A, u)$. The Arnoldi method provides such a way and the following theorem provides its basic idea. Recall that a Hessenberg matrix is said to be unreduced if all its subdiagonal entries are nonzero, see also Definition 1.21.

Theorem 3.5. *Let the columns of*

$$U_{k+1} = \begin{bmatrix} u_1, u_2, \ldots, u_{k+1} \end{bmatrix} \in \mathbb{R}^{n \times (k+1)}$$

form an orthonormal basis for $\mathcal{K}_{k+1}(A, u_1)$. Then there exists a $(k+1) \times k$ unreduced upper Hessenberg matrix $\hat{H}_k$ so that

$$AU_k = U_{k+1}\hat{H}_k. \tag{3.2}$$

Conversely, a matrix U_{k+1} with orthonormal columns satisfies a relation of the form (3.2) only if the columns of U_{k+1} form a basis for $\mathcal{K}_{k+1}(A, u_1)$.

Proof. This statement is well known, see e.g. [305, Thm. 5.1.1]. The following proof is provided as it reveals some useful relationships between Krylov matrices, QR and Arnoldi decompositions.

Let us partition

$$\left[\, K_k(A, u_1),\, A^k u_1 \,\right] = \left[\, U_k,\, u_{k+1} \,\right] \begin{bmatrix} R_k & r_{k+1} \\ 0 & r_{k+1,k+1}, \end{bmatrix},$$

then $U_k R_k$ is a QR decomposition of $K_k(A, u_1)$. Setting $S_k = R_k^{-1}$, we obtain

$$\begin{aligned} AU_k &= AK_k(A, u_1)S_k = K_{k+1}(A, u_1) \begin{bmatrix} 0 \\ S_k \end{bmatrix} \\ &= U_{k+1} R_{k+1} \begin{bmatrix} 0 \\ S_k \end{bmatrix} = U_{k+1} \hat{H}_k, \end{aligned}$$

where

$$\hat{H}_k = R_{k+1} \begin{bmatrix} 0 \\ S_k \end{bmatrix}.$$

The $(k+1) \times k$ matrix $\hat{H}_k$ is obviously in upper Hessenberg form. $\hat{H}_k$ is unreduced because R_{k+1} is invertible and the ith subdiagonal entry of $\hat{H}_k$ is given by $r_{i+1,i+1} s_{ii} = r_{i+1,i+1}/r_{ii}$.

This proves one direction of the theorem, for the other direction let us assume that there exists a matrix U_{k+1} with orthonormal columns so that (3.2) with an unreduced Hessenberg matrix $\hat{H}_k$ is satisfied. For $k = 1$, we have $Au_1 = h_{11}u_1 + h_{21}u_2$. The fact that h_{21} does not vanish implies that the vector u_2 is a linear combination of u_1 and Au_1. Therefore, the columns of $[u_1, u_2]$ form an orthonormal basis for $K_2(A, u_1)$. For general k, we proceed by induction over k; let the columns U_k form an orthonormal basis for $K_k(A, u_1)$. Partition

$$\hat{H}_k = \begin{bmatrix} \hat{H}_{k-1} & h_k \\ 0 & h_{k+1,k} \end{bmatrix},$$

then (3.2) implies that

$$Au_k = U_k h_k + h_{k+1,k} u_{k+1}.$$

Again, $h_{k+1,k} \neq 0$ implies that u_{k+1}, as a linear combination of Au_k and the columns of U_k, is an element of $\mathcal{K}_{k+1}(A, u_k)$. Hence, the columns of U_{k+1} form an orthonormal basis for $K_{k+1}(A, u_1)$. □

This theorem justifies the following definition.

Definition 3.6. *Let the columns of $U_{k+1} = [U_k, u_{k+1}] \in \mathbb{R}^{n \times (k+1)}$ form an orthonormal basis. If there exists an (unreduced) Hessenberg matrix $\hat{H}_k \in \mathbb{R}^{(k+1) \times k}$ so that*

$$AU_k = U_{k+1} \hat{H}_k, \tag{3.3}$$

then (3.3) is called an (unreduced) Arnoldi decomposition of order k.

By a suitable partition of $\hat{H}_k$, we can rewrite (3.3) as

$$AU_k = [\, U_k,\ u_{k+1} \,] \begin{bmatrix} H_k \\ h_{k+1,k} e_k^T \end{bmatrix} = U_k H_k + h_{k+1,k} u_{k+1} e_k^T. \tag{3.4}$$

This shows that U_k satisfies the "invariant subspace relation" $AU_k = U_k H_k$ except for a rank-one perturbation.

We have seen that U_k may contain good approximations to eigenvectors and invariant subspaces. These approximations and the associated eigenvalues can be obtained from the Ritz vectors and Ritz values defined below, which amounts to the so called *Rayleigh-Ritz method*.

Definition 3.7. *Let $A \in \mathbb{R}^{n\times n}$ and let the columns of $U_k \in \mathbb{R}^{n\times k}$ be orthonormal. The $k \times k$ matrix $H_k = U_k^T A U_k$ is called the* Rayleigh quotient*, an eigenvalue λ of H_k is called a* Ritz value*, and if w is an eigenvector of H_k belonging to λ, then $U_k w$ is called a* Ritz vector *belonging to λ.*

Given an Arnoldi decomposition of the form (3.4), Ritz values and vectors correspond to exact eigenvalues and eigenvectors of the perturbed matrix

$$A - h_{k+1,k} u_{k+1} u_k^T.$$

Assuming fairly well-conditioned eigenvalues and eigenvectors, the perturbation analysis of the standard eigenvalue problem, see Section 1.2, shows that a small value of $\|h_{k+1,k} u_{k+1} u_k^T\|_2 = |h_{k+1,k}|$ implies that all Ritz values and vectors are close to some eigenvalues and vectors of A. Note that an *individual* Ritz vector $U_k w$ belonging to a Ritz value λ with $\|w\|_2 = 1$ may correspond to a backward error much smaller than $|h_{k+1,k}|$:

$$\left(A - (h_{k+1,k} w_k) u_{k+1} (U_k w)^T\right) (U_k w) = \lambda (U_k w),$$

where w_k denotes the kth component of w. It can be shown that if the angle between a simple eigenvector x and $\mathcal{K}_k(A, u_1)$ converges to zero, then there exists at least one Ritz vector $U_k w$ so that the backward error $\|(h_{k+1,k} w_k) u_{k+1} (U_k w)^T\|_2 = |h_{k+1,k} w_k|$ converges to zero, see [305, Sec. 4.4.2].

The Basic Algorithm

The Arnoldi decomposition (3.2) almost immediately leads to an algorithm for its computation, see Algorithm 15. Several remarks concerning the implementation of this algorithm are necessary:

1. Algorithm 15 can be implemented such that exactly one $n \times n$ matrix-vector multiplication is needed per step. The computational cost of Algorithm 15 is dominated by these matrix-vector multiplications as long as $k \ll n$, making it competitive with the power method in this respect. However, the need for storing the $n \times (k+1)$ matrix U_{k+1} makes Algorithm 15 more expensive than the power method in terms of memory requirements.
2. The algorithm breaks down as soon as it encounters $h_{j+1,j} = 0$. In this case, the columns U_j span an invariant subspace and the eigenvalues of H_j are exact eigenvalues of A. If those are not the desired eigenvalues, one can restart the iteration with a random, unit vector orthogonal to U_j.

Algorithm 15 Arnoldi method

Input: A matrix $A \in \mathbb{R}^{n\times n}$, a starting vector $u_1 \in \mathbb{R}^n$ with $\|u_1\|_2 = 1$, and an integer $k \leq n$.

Output: A matrix $U_{k+1} = [u_1, \ldots, u_{k+1}] \in \mathbb{R}^{n\times(k+1)}$ with orthonormal columns and an upper Hessenberg matrix $\hat{H}_k \in \mathbb{R}^{(k+1)\times k}$, defining an Arnoldi decomposition (3.2) of kth order.

$\hat{H}_0 \leftarrow []$
for $j \leftarrow 1, 2, \ldots, k$ **do**
  $h_j \leftarrow U_j^T A u_j$
  $v \leftarrow A u_j - U_j h_j$
  $h_{j+1,j} \leftarrow \|v\|_2$
  $u_{j+1} \leftarrow v/h_{j+1,j}$
  $\hat{H}_j \leftarrow \begin{bmatrix} \hat{H}_{j-1} & h_j \\ 0 & h_{j+1,j} \end{bmatrix}$
end for

3. The algorithm can be seen as an implicit implementation of the Gram-Schmidt orthonormalization process. As such, it inherits the numerical instabilities of Gram-Schmidt, i.e., the orthogonality of the columns of U_k can be severely affected in the presence of roundoff errors, see [48, 261]. This can be avoided by reorthogonalizing the computed vector u_{k+1} against the columns of U_k if it is not sufficiently orthogonal, see [98, 261]. Another stable alternative is based on Householder matrices [341].
4. If not the extremal eigenvalues but eigenvalues close to a given value σ are desired, then Algorithm 15 will more quickly yield approximations to these eigenvalues if A is (implicitly) replaced by $(A - \sigma I)^{-1}$. This amounts to the so called *shift-and-invert Arnoldi method* and requires the solution of a linear system $(A - \sigma I)y = u_k$ in each iteration. An overview of algorithms for solving such linear systems, involving large and sparse matrices, can be found in [21]. Detailed analyses on how accurately these linear systems must be solved in order to achieve a certain level of accuracy in the eigenvalues can be found in [288, 287] and the references therein. One of the most important observations is that the accuracy of the linear system solvers can be considerably relaxed as soon as the eigenvalues of interest start converging.

3.2 Restarting and the Krylov-Schur Algorithm

One of the drawbacks of Algorithm 15 is its need for saving the $n \times (k+1)$ matrix U_{k+1}. Depending on the speed of convergence, this matrix may exceed the available memory long before the desired eigenvalues are sufficiently well approximated by some Ritz values. Also, the cost for computing the Ritz values of the $k \times k$ matrix H_k grows cubically with k.

3.2.1 Restarting an Arnoldi Decomposition

A way out of this dilemma has been suggested by Saad [280] based on earlier work by Manteuffel [232] for the iterative solution of linear systems. Given an Arnoldi decomposition of order m,

$$AU_m = [\ U_m,\ u_{m+1}\ ] \begin{bmatrix} H_m \\ h_{m+1,m} e_m^T \end{bmatrix}, \tag{3.5}$$

Saad proposed to choose a so called *filter polynomial* ψ, based on information contained in $\lambda(H_m)$, and to restart the Arnoldi method with the new starting vector $\tilde{u}_1 = \psi(A)u_1/\|\psi(A)u_1\|_2$. If the roots of ψ approximate eigenvalues of A, this has the effect that components of u_1 in the direction of eigenvectors belonging to these eigenvalues are damped. If we decompose $\lambda(H_m) = \Omega_w \cup \Omega_u$, where the sets Ω_w and Ω_u contain Ritz values approximating "wanted" and "unwanted" eigenvalues, respectively, then a reasonable choice for ψ is

$$\psi(z) = \prod_{\lambda \in \Omega_u} (z - \lambda).$$

The overview paper [295] summarizes other reasonable choices for ψ including those based on Chebyshev polynomials.

The described technique is commonly called *explicit restarting* and can be implemented so that no extra matrix-vector multiplications are required. A drawback of explicit restarting is that only information associated with the Ritz values is exploited; any other information contained in the Arnoldi decomposition is discarded. Sorensen [294] developed an *implicit restarting* technique, which is capable to preserve some of the information contained in $\mathcal{K}_m(A, u_1)$ as well. Implicit restarting transforms and truncates an Arnoldi decomposition (3.5) of order m to an unreduced Arnoldi decomposition of order $k < m$,

$$A\tilde{U}_k = [\ \tilde{U}_k,\ \tilde{u}_{k+1}\ ] \begin{bmatrix} \tilde{H}_k \\ \tilde{h}_{k+1,k} e_k^T \end{bmatrix}, \tag{3.6}$$

so that the columns of U_k form an orthonormal basis for the Krylov subspace $\mathcal{K}_k(A, \psi(A)u_1)$. Note that this is essentially the same decomposition that would have been produced by explicit restarting followed by k steps of the Arnoldi method. Implicit restarting can be accomplished via the application of $m - k$ implicit shifted QR iterations to H_m, where the shifts are the zeros of the $(m - k)$-degree filter polynomial ψ. For more details, the reader is referred to [294] and to the documentation of the software package ARPACK [216], which provides a Fortran 77 implementation of the implicitly restarted Arnoldi algorithm. Moreover, ARPACK facilitates similar techniques, developed by Lehoucq and Sorensen [215], for locking converged and wanted Ritz values as well as purging converged but unwanted Ritz values. In effect, the corresponding Ritz vectors are decoupled from the active part of the computation.

3.2.2 The Krylov Decomposition

Stewart [307] argued that the implicitly restarted Arnoldi algorithm may suffer from the forward instability of implicit QR iterations. This forward instability, explained in detail by Parlett and Le [262], is well demonstrated by the following example.

Example 3.8 ([262, Ex.2.1]). Consider an Arnoldi decomposition

$$A[e_1, \ldots, e_6] = [e_1, \ldots, e_6]H_6 + h_{7,6}e_7e_6^T$$

having the Hessenberg factor

$$H_6 = \begin{bmatrix} 6683.3333 & 14899.672 & 0 & 0 & 0 & 0 \\ 14899.672 & 33336.632 & 34.640987 & 0 & 0 & 0 \\ 0 & 34.640987 & 20.028014 & 11.832164 & 0 & 0 \\ 0 & 0 & 11.832164 & 20.001858 & 10.141851 & 0 \\ 0 & 0 & 0 & 10.141851 & 20.002287 & 7.5592896 \\ 0 & 0 & 0 & 0 & 7.5592896 & 20.002859 \end{bmatrix}.$$

One eigenvalue of H_6 is $\lambda = 40000.0003739678$. Assume that one wants to remove this Ritz value from the Arnoldi decomposition. This corresponds to the application of an implicit QR iteration with the shift λ, which theoretically leads to a deflated eigenvalue at the bottom right corner of H_6. In finite-precision arithmetic, however, the transformed matrix $\hat{H}_6 = \mathrm{fl}(Q^T H_6 Q)$ ($\mathrm{fl}(\cdot)$ denotes the value of the argument obtained in finite-precision arithmic) is far from having this property, its last subdiagonal entry is given by ≈ -226.21.

Moreover, the implicitly restarted Arnoldi algorithm with filter polynomial $\psi(z) = (z-\lambda)$ yields a truncated basis $\tilde{U}_5$ given by the first five columns of the matrix $[e_1, \ldots, e_6]Q$. Theoretically, the Ritz vector x belonging to λ should have no components in the space spanned by the columns of $\tilde{U}_5$, i.e.,

$$\theta_1(\mathrm{span}(x), \mathrm{span}(\tilde{U}_5)^\perp) = 0$$

Again, this relationship is violated in finite precision arithmetic, the computed matrix $\hat{U}_5$ satisfies

$$\theta_1(\mathrm{span}(x), \mathrm{span}(\hat{U}_5)^\perp) \approx 0.0024.$$

Hence, the forward instability of the QR iteration causes components of the unwanted Ritz vector x to persist in the computation. ◊

It should be emphasized that the described effect does not limit the attainable accuracy of eigenvalues computed by the implicitly restarted Arnoldi algorithm, but it may have an effect on the convergence. Moreover, as locking and purging use similar techniques, great care must be taken to implement these operations in a numerically reliable fashion.

An elegant solution to all these difficulties was proposed by Stewart [307]. It consists of relaxing the definition of an Arnoldi decomposition and using the eigenvalue reordering techniques described in Section 1.7 for the restarting, locking and purging operations. The relaxed definition reads as follows.

Definition 3.9. *Let the columns of* $U_{k+1} = [U_k, u_{k+1}] \in \mathbb{R}^{n\times(k+1)}$ *form an orthonormal basis. A* Krylov decomposition of order k *has the form*

$$AU_k = U_k B_k + u_{k+1} b_{k+1}^T, \tag{3.7}$$

or equivalently

$$AU_k = U_{k+1}\hat{B}_k, \quad \text{with } \hat{B}_k = \begin{bmatrix} B_k \\ b_{k+1}^T \end{bmatrix}.$$

Note that there is no restriction on the matrix $\hat{B}_k$ unlike in the Arnoldi decomposition, where this factor is required to be an upper Hessenberg matrix. Nevertheless, any Krylov decomposition is equivalent to an Arnoldi decomposition in the following sense.

Lemma 3.10 ([307]). *Let* $AU_k = U_{k+1}\hat{B}_k$ *be a Krylov decomposition of order* k*. Then there exists an Arnoldi decomposition* $A\tilde{U}_k = \tilde{U}_{k+1}\hat{H}_k$ *of order* k *so that* $\operatorname{span}(U_{k+1}) = \operatorname{span}(\tilde{U}_{k+1})$*.*

Proof. The following proof provides a construction for computing an Arnoldi decomposition corresponding to a given Krylov decomposition.

Partition $\hat{B}_k = \begin{bmatrix} B_k \\ b_{k+1}^T \end{bmatrix}$, let F denote the flip matrix and set $Z = F \cdot H_1(Fb_{k+1})$, which yields $b_{k+1}^T ZF = h_{k+1,k} e_k^T$ (recall that $H_1(\cdot)$ designates a Householder matrix, see Section 1.3.2). Use Algorithm 1 to construct an orthogonal matrix Q so that $Q^T(Z^T B_k^T Z)^T Q$ is in upper Hessenberg form. This implies that the matrix $H_k = (ZQF)^T B_k (ZQF)$ is also in upper Hessenberg form. Since Q takes the form $1 \oplus \tilde{Q}$, we still have $b_{k+1}^T(ZQF) = h_{k+1,k} e_k^T$. Setting $\tilde{U}_k = U_k(ZQF)$ and $\tilde{U}_{k+1} = [\tilde{U}_k, u_{k+1}]$ reveals $\operatorname{span}(\tilde{U}_{k+1}) = \operatorname{span}(U_{k+1})$ and

$$A\tilde{U}_k = \tilde{U}_{k+1} \begin{bmatrix} H_k \\ h_{k+1,k} e_k^T \end{bmatrix},$$

which concludes the proof. □

Particularly useful are Krylov decompositions having the factor B_k in real Schur form. Such a decomposition will be called *Krylov-Schur decomposition.*

3.2.3 Restarting a Krylov Decomposition

Given a Krylov decomposition of order m,

$$AU_m = U_{m+1} \begin{bmatrix} B_m \\ b_{m+1}^T \end{bmatrix}, \tag{3.8}$$

implicit restarting proceeds as follows. First, we apply Hessenberg reduction and the QR algorithm to compute an orthogonal matrix Q_1 so that $T_m = Q_1^T B_m Q_1$ has real Schur form. As already described in Section 3.2.1, the

eigenvalues of T_m are decomposed in a subset Ω_w containing $k < m$ "wanted" Ritz values and a subset Ω_u containing $m - k$ "unwanted" Ritz values. To preserve realness, we assume that Ω_w as well as Ω_u are closed under complex conjugation. Secondly, the Ritz values contained in Ω_w are reordered to the top of T_m. This yields another orthogonal matrix Q_2 so that

$$AU_mQ_1Q_2 = [U_mQ_1Q_2, u_{m+1}] \left[\begin{array}{cc} T_w & \star \\ 0 & T_u \\ \hline b_w^T & \star \end{array}\right], \quad \lambda(T_w) = \Omega_w, \ \lambda(T_u) = \Omega_u.$$

Finally, this Krylov-Schur decomposition is truncated, i.e., if we let $\tilde{U}_k$ contain the first $k < m$ columns of $U_mQ_1Q_2$ and set $\tilde{u}_{k+1} = u_{m+1}$, then we get the following Krylov decomposition of order k:

$$A\tilde{U}_k = [\tilde{U}_k, \tilde{u}_{k+1}] \left[\begin{array}{c} T_w \\ b_w^T \end{array}\right]. \tag{3.9}$$

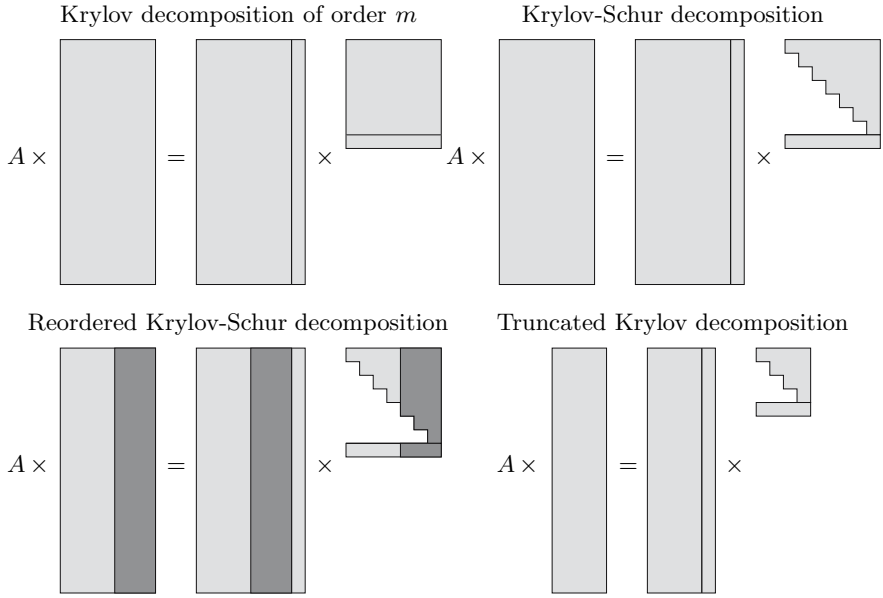


Fig. 3.2. Restarting a Krylov decomposition.

The described process is depicted in Figure 3.2. It can be shown that the transition from the extended Krylov decomposition (3.8) to the reduced Krylov decomposition (3.9) is formally equivalent to an implicit restart of the corresponding Arnoldi decomposition with filter polynomial $\psi(z) = \prod_{\lambda \in \lambda(T_u)} (z - \lambda)$, see [307].

After being truncated, the Krylov decomposition is again expanded to a decomposition of order m using Algorithm 16. The remarks concerning the

Algorithm 16 Expanding a Krylov decomposition

Input: A Krylov decomposition of order k: $AU_k = U_{k+1}\hat{B}_k$. An integer $m > k$.
Output: A Krylov decomposition of order m: $AU_m = U_{m+1}\hat{B}_m$.

for $j \leftarrow k+1, 2, \ldots, m$ **do**
  $h_j \leftarrow U_j^T A u_j$
  $v \leftarrow Au_j - U_j h_j$
  $h_{j+1,j} \leftarrow \|v\|_2$
  $u_{j+1} \leftarrow v/h_{j+1,j}$
  $\hat{B}_j \leftarrow \begin{bmatrix} \hat{B}_{j-1} & h_j \\ 0 & h_{j+1,j} \end{bmatrix}$
end for

proper implementation of the Arnoldi method, Algorithm 15, apply likewise to Algorithm 16. The returned factor $\hat{B}_m$ has the following structure:

$$\hat{B}_m = \begin{array}{c} \\ k \\ m-k \\ 1 \end{array} \begin{array}{c} \begin{array}{cc} k & m-k \end{array} \\ \begin{bmatrix} \square & \square \\ \star - \star & \diagdown\!\square \\ & \star \end{bmatrix} \end{array}.$$

If restarting is repeatedly applied to this Krylov decomposition of order m, then the first step consists of reducing the upper $m \times m$ part of $\hat{B}_m$ to Hessenberg form. Note that this reduction can be restricted to the left upper $(k+1) \times (k+1)$ part by applying Algorithm 1 to the matrix $F \cdot \hat{B}_m(k+1 : k+1)^T \cdot F$, where once again F denotes the flip matrix, similar to the construction used in the proof of Lemma 3.10.

3.2.4 Deflating a Krylov Decomposition

The process of restarting and expanding Krylov decompositions is repeated until convergence occurs. In ARPACK [216, Sec. 4.6], a Ritz value λ of an order m Arnoldi decomposition (3.5) is regarded as converged if the associated Ritz vector $U_m w$ ($\|w\|_2 = 1$) satisfies

$$\|A(U_m w) - \lambda(U_m w)\|_2 = |h_{m+1,m} e_m^T w| \leq \max\{\mathbf{u}\,\|H_m\|_F, \texttt{tol} \times |\lambda|\}, \quad (3.10)$$

where `tol` is a chosen user tolerance. A direct extension of this criterion to an order m Krylov decomposition (3.8) is given by

$$\|A(U_m w) - \lambda(U_m w)\|_2 = |b_{m+1}^T w| \leq \max\{\mathbf{u}\,\|B_m\|_F, \texttt{tol} \times |\lambda|\}. \quad (3.11)$$

Both criteria guarantee a small backward error for λ, since λ is an eigenvalue of the slightly perturbed matrix $(A + E)$ with $E = -(b_{m+1}^T w)u_{m+1}(U_m w)^T$. Similarly, a matrix Q_d containing an orthonormal basis for the space spanned by d Ritz vectors $U_m w_1, \ldots, U_m w_d$ is regarded as converged to a basis of an invariant subspace if it satisfies

$$\begin{aligned} \|AQ_d - Q_d(Q_d^T AQ_d)\|_F &= \|b_{m+1}^T Q_d\|_2 \\ &\leq \max\{\mathbf{u}\,\|B_m\|_F, \mathtt{tol} \times \|Q_d^T AQ_d\|_F\|\}. \end{aligned} \tag{3.12}$$

It should be noted that the individual convergence of the Ritz vectors $U_m w_1$, $\ldots, U_m w_d$ does not necessarily imply (nearby) convergence of the orthonormal basis Q_d, at least as long as the corresponding eigenvalues are not particularly well conditioned.

If a Ritz value $\lambda \in \mathbb{R}$ has converged to a wanted eigenvalue and the corresponding Ritz vector is $U_m w$, then the components of this vector should no longer participate in the subsequent search for other eigenvectors. This is the aim of deflation, which proceeds as follows [307]. Given a Krylov decomposition of order m, the factor B_m is reduced to a reordered Schur form $\tilde{B}_m = \left[\begin{smallmatrix} \lambda & \star \\ 0 & \tilde{B}_{m-1} \end{smallmatrix}\right]$. The Krylov decomposition is partitioned as

$$A[\tilde{u}_1, \tilde{U}_{m-1}] = [\tilde{u}_1, \tilde{U}_m] \left[\begin{array}{cc} \lambda & \star \\ 0 & \tilde{B}_{m-1} \\ \hline \tilde{b}_1 & \tilde{b}_{m-1}^T \end{array}\right],$$

where it is assumed that $|\tilde{b}_1| = |b_{m+1}^T w|$ satisfies inequality (3.11), i.e., the Ritz value λ can be regarded as converged and $\tilde{b}_1$ can be safely set to zero. Although newly produced vectors in the Krylov basis must still be orthogonalized against $\tilde{u}_1$, the restarting algorithm can be restricted to the subdecomposition $A\tilde{U}_{m-1} = \tilde{U}_m \left[\begin{smallmatrix} \tilde{B}_{m-1} \\ \tilde{b}_{m-1}^T \end{smallmatrix}\right]$ for the remainder of the computation. This fact can be exploited to improve the performance of the Krylov-Schur algorithm to some extent.

The Krylov-Schur algorithm also gives rise to a more restrictive convergence criterion based on Schur vectors instead of Ritz vectors. Assume that $d < m$ Ritz values have been deflated in a Krylov decomposition of order m:

$$A[Q_d, U_{m-d}] = [Q_d, U_{m-d+1}] \left[\begin{array}{cc} T_d & \star \\ 0 & B_{m-d} \\ \hline 0 & b_{m-d}^T \end{array}\right], \tag{3.13}$$

where $T_d \in \mathbb{R}^{d \times d}$ is a quasi-upper triangular matrix containing the already deflated Ritz values. To deflate another Ritz value, B_{m-d} is reduced to real Schur form:

$$A[Q_d, \tilde{u}_1, \tilde{U}_{m-d-1}] = [Q_d, \tilde{u}_1, \tilde{U}_{m-d}] \left[\begin{array}{ccc} T_d & \star & \star \\ 0 & \lambda & \star \\ 0 & 0 & \tilde{B}_{m-d-1} \\ \hline 0 & \tilde{b}_1 & \tilde{b}_{m-d-1}^T \end{array}\right]. \tag{3.14}$$

The Ritz value $\lambda \in \mathbb{R}$ is regarded as converged if the scalar $\tilde{b}_1$ satisfies:

$$|\tilde{b}_1| \leq \max\{\mathbf{u}\,\|B_{m-d}\|_F, \mathtt{tol} \times |\lambda|\} \tag{3.15}$$

Let $\tilde{Q}$ be an orthogonal matrix such that $\tilde{Q}^T \begin{bmatrix} T_d & \star \\ 0 & \lambda \end{bmatrix} \tilde{Q} = \begin{bmatrix} \lambda & \star \\ 0 & \tilde{T}_d \end{bmatrix}$, then $v = [Q_d, \tilde{u}_1]\tilde{Q}e_1$ is a Ritz vector belonging to λ, which satisfies

$$\|Av - \lambda v\|_2 = |[0, \tilde{b}_1]\tilde{Q}e_1| \leq |\tilde{b}_1|.$$

Thus, the Ritz value λ has also converged in the sense of (3.11). If the selected Ritz value λ does not satisfy (3.15), we may test any other real Ritz value of $\tilde{B}_{m-d-1}$ by reordering the Schur form $\begin{bmatrix} \lambda & \star \\ 0 & \tilde{B}_{m-d-1} \end{bmatrix}$ in (3.14). It has been pointed out by Byers [81] that this deflation strategy, originally suggested by Stewart [307], can be considered as a variant of the aggressive early deflation strategy described in Section 1.6.

So far, we have only considered the deflation of real eigenvalues. A complex conjugate pair of eigenvalues can be deflated in a similar fashion, by reordering the corresponding two-by-two block to the top left corner of B_{m-d}.

Using the more restrictive criterion (3.15) instead of (3.11) has the advantage that the orthonormal basis spanned by the deflated Ritz vectors nearly satisfies the convergence criterion (3.12). To see this, assume that the d locked Ritz values in (3.13) have been deflated based upon a criterion of the form (3.15). Then there exists a vector $\tilde{b}_d \in \mathbb{R}^d$, where each component satisfies an inequality of type (3.15), so that $\|AQ_d - Q_dT_d\|_F = \|\tilde{b}_d\|_2$. Thus,

$$\|AQ_d - Q_dT_d\|_F \leq \sqrt{d}\max\{\mathbf{u}, \mathtt{tol}\} \cdot \|A\|_F.$$

Both algorithms, the implicitly restarted Arnoldi algorithm and the described Krylov-Schur algorithm, have their pros and cons. While the former algorithm can perform restarts with an arbitrarily chosen filter polynomial, the latter algorithm is a more reliable machinery for performing deflations and particular types of restarts. Of course, both algorithms can be combined, e.g., by using Arnoldi for restarts and Krylov-Schur for deflations.

3.3 Balancing Sparse Matrices

In Section 1.4, we have described a two-stage algorithm for balancing general, dense matrices, which can have positive effects on the accuracy and efficiency of subsequent eigenvalue computations. Unfortunately, these algorithms are not suitable for balancing large and sparse matrices, especially if the entries of the matrix under consideration are only implicitly given, e.g., via the action of the matrix on a vector. To resolve this issue, Chen and Demmel [91] have developed a two-stage balancing algorithm, particularly suited for large and sparse matrices, and with similar positive effects.

3.3.1 Irreducible Forms

The first stage of the balancing algorithm consists of reducing a sparse matrix to irreducible form. A matrix $A \in \mathbb{R}^{n\times n}$ is called *reducible* if there exists a permutation matrix $P \in \mathbb{R}^{n\times n}$ so that

$$P^T AP = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \tag{3.16}$$

where A_{11} and A_{22} are square matrices of order not less than one. If no such permutation exists, then A is called *irreducible*. The matrices A_{11} and A_{22} can be further reduced until A is permuted to block upper triangular form with irreducible diagonal blocks:

$$P^T AP = \begin{bmatrix} A_1 & \star & \cdots & \star \\ 0 & A_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \star \\ 0 & \cdots & 0 & A_r \end{bmatrix}, \quad A_i \in \mathbb{R}^{n_i\times n_i}, \quad n_i \geq 1. \tag{3.17}$$

Constructing this final *irreducible form* (a pre-stage of the so called Frobenius normal form [44]) is equivalent to finding the strongly connected components of the incidence graph of A and numbering them in their topological order. To explain this in more detail, let us briefly introduce the necessary graph theoretic tools [193, 339].

Definition 3.11. *The* incidence graph of a matrix $A \in \mathbb{R}^{n\times n}$*, denoted by* $\mathcal{G}_A(V, E)$*, is a directed graph with vertex and edge sets*

$$V = \{v_1, \ldots, v_n\}, \quad E = \{(v_i, v_j) : \ a_{ij} \neq 0\},$$

respectively.

Definition 3.12. *A directed graph is called* strongly connected *if for any pair of vertices v and w there are paths from v to w as well as from w to v. The* strongly connected components *of a directed graph are its maximal strongly connected subgraphs.*

It is one of the fundamental relations between matrices and graphs that the strongly connected components of the incidence graph of a matrix in irreducible form (3.17) are the subgraphs belonging to the vertex sets of the diagonal blocks, i.e.,

$$V_i = \{v_{k_{i-1}+1}, \ldots, v_{k_i}\}, \quad i = 1, \ldots, r, \quad k_i = \sum_{j=1}^{i} n_j,$$

see e.g. [193]. Moreover, there is no edge from any vertex in V_j to any vertex in V_i if $i < j$. This relation, denoted by $V_i \preceq V_j$, defines a partial order on the set of strongly connected components, the so called *topological order*.

We can use these connections to graph theory to reduce a given matrix $A \in \mathbb{R}^{n\times n}$ to irreducible form. First, Tarjan's algorithm [319] is applied to the incidence graph $G_A(V, E)$ of A in order to find the r strongly connected components of $G_A(V, E)$. These components can be written as

$$V_i = \{v_{l_{k_{i-1}+1}}, \ldots, v_{l_{k_i}}\}, \quad i = 1, \ldots, r, \quad k_i = \sum_{j=1}^{i} n_j,$$

for some integers $l_1, \ldots, l_n \in [1, n]$ and integers $n_1, \ldots, n_r$ satisfying $\sum n_i = n$. W.l.o.g., we may assume $V_i \preceq V_j$ for $i < j$. Next, consider a permutation p which maps l_j to j for $j \in [1, n]$. Then, the corresponding permutation matrix $P = [p_{ij}]$ with $p_{ij} = 1$ if and only if $i = p(j)$ produces a matrix $P^T AP$ having irreducible form.

For numerical experiments comparing the benefits and cost of this approach with the permutation algorithm for dense matrices, Algorithm 4, see [91]. Permuting a matrix A to irreducible form has complexity $\mathcal{O}(n + nz)$, where nz denotes the number of nonzeros in A. It was observed in [91] that Algorithm 4 requires more computational time if the density nz/n^2 is less than a certain ratio γ. The exact value of γ depends on the sparsity pattern and the implementation, but a typical observation was $\gamma \approx 1/2$. Note that a matrix can only be permuted if its entries are explicitly accessible.

3.3.2 Krylov-Based Balancing

Algorithm 5, the Parlett-Reinsch algorithm for balancing a general matrix, requires the calculation of row and column norms, implying that matrix entries must be given explicitly. This requirement is sometimes not satisfied, a large and sparse matrix might only be defined through its action on a vector. For these cases, only balancing algorithms which are solely based on a few matrix-vector multiplications, and possibly matrix-transpose-vector multiplications, can be used. Such algorithms were developed in [91], leading to the so called *Krylov-based balancing*, although it should be mentioned that none of the proposed algorithms exploits a complete Krylov subspace.

One such algorithm, KrylovAtz, is built on the following fact.

Lemma 3.13. *Let $A \in \mathbb{R}^{n\times n}$ be an irreducible matrix with non-negative entries and spectral radius $\rho(A)$. Let x and y be the normalized right and left* Perron vectors *of A, i.e., $Ax = \rho(A)x$ and $A^T y = \rho(A)y$ with $\|x\|_2 = \|y\|_2 = 1$. If*

$$D = \operatorname{diag}(\sqrt{x_1/y_1}, \sqrt{x_2/y_2}, \ldots, \sqrt{x_n/y_n}), \tag{3.18}$$

then $\|D^{-1}AD\|_2 = \rho(A)$.

Proof. See, e.g., [91]. □

The scaling employed in Lemma 3.13 achieves minimal 2-norm among all equivalence transformations of A as $\|X^{-1}AX\|_2 \geq \rho(A)$ for any nonsingular matrix X. Also, the right and left Perron vectors of $D^{-1}AD$ equal $D^{-1}x = Dy$, thus the condition number of the spectral radius becomes minimal. If A contains negative entries, then we can apply Lemma 3.13 to $|A| := [|a_{ij}|]_{i,j=1}^n$ to construct a (possibly suboptimal) diagonal scaling matrix D. It was observed in [91] that this choice of scaling improves the accuracy of the computed eigenvalues for almost all considered examples. Nevertheless, it is not clear how to predict the potential gain in accuracy.

It remains to compute the Perron vectors x and y of $|A|$. In principle, one could apply the power method to $|A|$ and $|A^T|$ to approximate these vectors. However, if A is not explicitly defined then also the action of $|A|$ and $|A^T|$ on a vector must be approximated by matrix-vector products which only involve the original matrix A and its transpose. A statistically motivated procedure based on products with a random vector z, where the entries z_i equal 1 or -1 with probability $1/2$, was presented in [91]. It makes use of the fact that multiplying A by z approximates one step of the power method applied to $|A|$ with starting vector $[1, 1, \ldots, 1]^T$, see Algorithm 17.

Algorithm 17 KrylovAtz

Input: An irreducible matrix $A \in \mathbb{R}^{n\times n}$.
Output: A diagonal matrix D so that $D^{-1}AD$ is nearly balanced in the sense of Lemma 3.13.

$D \leftarrow I_n$
for $k = 1, 2, \ldots$ **do**
  $z \leftarrow$ vector of length n with random ± 1 entries
  $p \leftarrow D^{-1}(A(Dz)), \quad r \leftarrow D(A^T(D^{-1}z))$
  for $i = 1, \ldots, n$ **do**
    if $(p_i \neq 0)$ AND $(r_i \neq 0)$ **then**
      $d_{ii} \leftarrow d_{ii} \cdot \sqrt{|p_i|/|r_i|}$
    end if
  end for
end for

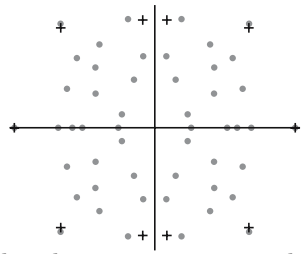
Remark 3.14. Based on the experimental results presented in [91] it was proposed to replace the conditions $p_i \neq 0$ $r_i \neq 0$ in the inner loop of Algorithm 17 by $|p_i| > \delta\|A\|_F$ and $|r_i| > \delta\|A\|_F$ for some $\delta > 0$. Although there is little theoretical justification for adding such a *cutoff value* δ it turns out that the choice $\delta = 10^{-8}$ often results in smaller norms for the scaled matrices. $\Diamond$

Algorithm 17 can be applied to the diagonal blocks of an irreducible form (3.17). If this form is not available, Algorithm 17 can be applied to the complete matrix A with less theoretical justification but often to an equal norm-reducing effect. For numerical experiments, see [91, 36].

Chen and Demmel [91] also proposed a one-sided variant of Algorithm 17, called KrylovAz, which does not require matrix-vector-multiplications involving A^T. Note, however, that the observed numerical results of KrylovAz are in some cases significantly worse than those of KrylovAtz.

4

Structured Eigenvalue Problems



This chapter is concerned with computing eigenvalues and invariant subspaces of a structured matrix. In the scope of this book, an $n \times n$ matrix A is considered to be *structured* if its n^2 entries depend on *less* than n^2 parameters.

Diagonal, upper triangular, sparse matrices belong to the more ubiquitous structures in (numerical) linear algebra. However, these structures will be of less interest in this chapter; none of them induces any particular eigenvalue properties. Instead, we focus on block cyclic, Hamiltonian, orthogonal, symplectic and similar structures, which all induce a distinctive and practically relevant symmetry in the spectrum of the matrix.

For example, consider a Hamiltonian matrix

$$H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}, \quad G = G^T, \quad Q = Q^T, \tag{4.1}$$

where A, G and Q are real square matrices of the same dimension. If λ is an eigenvalue of H then also $-\lambda, \bar{\lambda}, -\bar{\lambda}$ are eigenvalues of H. In other words, $\lambda(H)$ is symmetric with respect to the real *and* the imaginary axis. If one attempts to compute the eigenvalues of H by one of the methods from the previous chapters, such as the QR or the Krylov-Schur algorithm, then the computed eigenvalues will lose the symmetry with respect to the imaginary axis due to the influence of roundoff errors. This makes it difficult to identify eigenvalues of H that have negative or zero real part. However, a proper identification of these eigenvalues is often crucial in applications. For example, deciding whether a certain Hamiltonian matrix has purely imaginary eigenvalues is the most critical step in algorithms for computing the stability radius of a matrix [79, 167] or the H_∞ norm of a linear time-invariant system [59, 65].

Preserving the structure of a matrix helps preserve such eigenvalue symmetries. Other important aspects are accuracy and efficiency.

So far, we have considered methods that are numerically backward stable, i.e., the computed eigenvalues are the exact eigenvalues of a slightly perturbed matrix $A + E$. The concept of *strong backward stability* extends this concept

by requiring the perturbed matrix $A + E$ to have the same structure as the original matrix A [66]. It follows that a strongly backward stable method preserves induced eigenvalue symmetries. But, even better, since the set of admissible perturbations is now restricted to those that preserve the structure of A, the worst-case effect of E on the eigenvalue accuracy may be much smaller than the worst-case effect of general, unstructured perturbations. This question can be answered by structured condition numbers, which are the subject of Section 4.1.1. If the structured condition number of an eigenvalue is significantly smaller than the standard condition number then a strongly backward stable eigenvalue solver is likely to attain much higher accuracy for this eigenvalue than a general-purpose method. It must be said, however, that there is often no or only a slight difference between the standard and structured eigenvalue condition numbers.

Checking whether an eigenvalue/eigenvector pair has been computed in a backward stable manner simply amounts to checking the norm of the residual, a fact that has been used to develop meaningful stopping criteria for Krylov subspace methods. Depending on the structure under consideration, checking strong backward stability can be much more complicated; this issue is discussed in Section 4.1.2.

The presence of structure in a matrix gives rise to the hope that its eigenvalues and invariant subspaces can be computed more efficiently. This is explained in more detail in Section 4.1.3. It turns out that the actually attainable reduction in computational time is rather modest for some of the structures considered in this chapter.

Starting from Section 4.2, this chapter aims to provide an overview on theory, algorithms and applications for a number of popular matrix structures. Close attention is paid to product and (skew-)Hamiltonian eigenvalue problems. This choice is clearly biased towards the author's research but it can also be attributed to the fact that these structures admit structured Schur decompositions, which makes them more accessible than many others. Finally, Section 4.6 summarizes existing results for some other classes of matrices including symmetric, skew-symmetric, persymmetric and orthogonal matrices. Moreover, we consider palindromic matrix pairs to exemplify that some of the techniques developed in this chapter can be extended to generalized eigenvalue problems.

4.1 General Concepts

The diversity of structures is too high to provide a general framework that covers all aspects of structured eigenvalue problems. The aim of this section is to present those concepts that admit a rather general presentation, such as the notions of structured condition number and structured backward error.[1]

[1] If one considers only Lie algebras, Jordan algebras and automorphism groups associated with bilinear and sesquilinear forms then Mackey, Mackey, and Tisseur

4.1.1 Structured Condition Number

The condition numbers $c(\lambda)$ and $c(\mathcal{X})$ introduced in Section 1.2 provide first-order bounds on the worst-case effect of a perturbation E on an eigenvalue λ and an invariant subspace $\mathcal{X}$ of a matrix A. These measures are most appropriate if the only information available on E is that its norm is below a certain perturbation threshold ε. In the context of structured eigenvalue problems, however, there is often more information available. If, for example, λ and $\mathcal{X}$ have been computed by a strongly backward stable method it is reasonable to assume that $A + E$ has the same structure as A. Under such circumstances, $c(\lambda)$ and $c(\mathcal{X})$ may severely overestimate the actual worst-case effect of E. To obtain a more appropriate measure one can resort to *structured condition numbers*, which aim to provide first-order bounds on the worst-case effect of all perturbations E which additionally preserve the structure of A. An extreme example is provided by

$$A = \left[\begin{array}{cc|cc} 0 & -1 & 2 & 0 \\ 1 & 0 & 0 & 2 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{array}\right], \quad \mathcal{X} = \operatorname{span}\left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \right\}. \tag{4.2}$$

While $c(\mathcal{X}) = \infty$, we will see that the structured condition number is given by $c^{\mathcal{M}}(\mathcal{X}) = 1/2$, if the set of admissible perturbations is restricted to matrices of the form $E = \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix}$ with $E_{ij} = \begin{bmatrix} \beta_{ij} & \gamma_{ij} \\ -\gamma_{ij} & \beta_{ij} \end{bmatrix}$ for parameters $\beta_{ij}, \gamma_{ij} \in \mathbb{R}$.

Eigenvalues

Theorem 1.5 shows that a simple eigenvalue λ depends analytically on the entries of the matrix A in a sufficiently small open neighborhood $\mathcal{B}(A)$. To be more specific, there exists a uniquely defined analytic function $f_\lambda : \mathcal{B}(A) \to \mathbb{C}$ so that $\lambda = f_\lambda(A)$ and $\hat{\lambda} = f_\lambda(A + E)$ is an eigenvalue of $A + E$ for every $A + E \in \mathcal{B}(A)$. Moreover, one has the expansion

$$\hat{\lambda} = \lambda + \frac{1}{|y^H x|} y^H E x + \mathcal{O}(\|E\|^2), \tag{4.3}$$

where x and y are right and left eigenvectors belonging to λ normalized such that $\|x\|_2 = \|y\|_2 = 1$. This expansion forms the basis for our further considerations.

In analogy to (1.14) we establish the following notion of a structured condition number with respect to a structure described by a set $\mathcal{M} \subseteq \mathbb{C}^{n \times n}$.

[228, 229, 230] have shown that many more aspects admit such a unified treatment.

Definition 4.1. *Let λ be a simple eigenvalue of a matrix $A \in \mathbb{C}^{n\times n}$ and let $\hat{\lambda} = f_\lambda(A+E)$ denote the corresponding eigenvalue of the perturbed matrix $A+E$ with $A+E \in \mathcal{M}$ for some $\mathcal{M} \subseteq \mathbb{C}^{n\times n}$ and $\|E\|$ sufficiently small. Then the absolute* structured condition number for λ with respect to $\mathcal{M}$ *is defined as*

$$c^{\mathcal{M}}(\lambda) := \lim_{\varepsilon\to 0} \frac{1}{\varepsilon} \sup\left\{ |\hat{\lambda} - \lambda| : \ A+E \in \mathcal{M}, \|E\| \le \varepsilon \right\}, \tag{4.4}$$

where $\|\cdot\|$ denotes some matrix norm.

Remark 4.2. To obtain computable expressions for $c^{\mathcal{M}}(\lambda)$ it is often convenient to use the Frobenius norm in the definition (4.4) of $c^{\mathcal{M}}(\lambda)$. Such a choice will be denoted by $c_F^{\mathcal{M}}(\lambda)$. To compare structured and unstructured condition numbers, the matrix 2-norm is usually preferred, which will be denoted by $c_2^{\mathcal{M}}(\lambda)$. ◊

As the supremum in (4.4) is taken over the subset $\mathcal{M}$ instead of the whole matrix space $\mathbb{C}^{n\times n}$, we immediately have the relation $c_2^{\mathcal{M}}(\lambda) \le c(\lambda)$. Much of the work on structured condition numbers is concerned with the question by how far can $c_2^{\mathcal{M}}(\lambda)$ fall below $c(\lambda)$.

The expansion (4.3) can be employed to obtain the alternative expression

$$c^{\mathcal{M}}(\lambda) = \frac{1}{|y^H x|} \lim_{\varepsilon\to 0} \frac{1}{\varepsilon} \sup\left\{ |y^H E x| : \ A+E \in \mathcal{M}, \|E\| \le \varepsilon \right\}. \tag{4.5}$$

Linear structures

Depending on the nature of $\mathcal{M}$ and the employed matrix norm, the supremum in (4.5) might be very difficult to compute. It was shown by Higham and Higham [163] that this task simplifies considerably if we assume $\mathcal{M}$ to be a linear matrix subspace of either $\mathbb{R}^{n\times n}$ or $\mathbb{C}^{n\times n}$. One consequence is that the supremum is always attained at the boundary, i.e.,

$$\begin{aligned} c^{\mathcal{M}}(\lambda) &= \frac{1}{|y^H x|} \lim_{\varepsilon\to 0} \frac{1}{\varepsilon} \sup\left\{ |y^H E x| : \ A+E \in \mathcal{M}, \|E\| = \varepsilon \right\} \\ &= \frac{1}{|y^H x|} \sup\left\{ |y^H E x| : \ E \in \mathcal{M}, \|E\| = 1 \right\}. \end{aligned} \tag{4.6}$$

Another important consequence is that this formula leads to a computable expression for $c_F^{\mathcal{M}}(\lambda)$. To see this, let us introduce the notion of pattern matrices [99, 162, 163, 323]. As $\mathcal{M}$ is a linear matrix space, there is an $n^2 \times m$ matrix M such that for every $E \in \mathcal{M}$ there exists a uniquely defined parameter vector p with

$$\mathrm{vec}(E) = Mp, \quad \|E\|_F = \|p\|_2. \tag{4.7}$$

In other words, the columns of M form an orthonormal basis for the linear space spanned by all $\mathrm{vec}(E)$ with $E \in \mathcal{M}$. Any such matrix M is called a *pattern matrix* for $\mathcal{M}$. Together with (4.6), the relationships (4.7) yield

$$c_F^{\mathcal{M}}(\lambda) = \frac{1}{|y^H x|} \sup \left\{ |(\bar{x} \otimes y)^H M p| : \ \|p\|_2 = 1, \ p \in \mathbb{K}^m, \right\}, \tag{4.8}$$

where $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$ and $\bar{x}$ denotes the complex conjugate of x.

When $\mathbb{K} = \mathbb{C}$ is a complex matrix space the supremum in (4.8) is taken over all $p \in \mathbb{C}^m$ and consequently

$$c_F^{\mathcal{M}}(\lambda) = \frac{1}{|y^H x|} \|(\bar{x} \otimes y)^H M\|_2. \tag{4.9}$$

Also, if M is real and λ is a real eigenvalue, we have $c_F^{\mathcal{M}}(\lambda) = \|(x \otimes y)^T M\|_2 / |y^T x|$ since x, y can be chosen to be real and the supremum in (4.8) is taken over all $p \in \mathbb{R}^m$.

Example 4.3. Consider the set $\mathcal{M} \equiv$ Hamil of all $2n \times 2n$ Hamiltonian matrices of the form (4.1). It is considerably simple to find a pattern matrix M_{Hamil} satisfying the conditions in (4.7). One suitable choice is given by

$$M_{\text{Hamil}} = \begin{bmatrix} P_2 \otimes I_n & 0 \\ 0 & P_2 \otimes I_n \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} I_{n^2} & 0 & 0 \\ 0 & M_{\text{symm}} & 0 \\ 0 & 0 & M_{\text{symm}} \\ -\frac{1}{\sqrt{2}} P_{\text{vec}} & 0 & 0 \end{bmatrix},$$

where P_2 is a perfect shuffle matrix of dimension $2n$ (see Section 4.2.3) and $M_{\text{symm}} \in \mathbb{R}^{n^2 \times n(n+1)/2}$ is a pattern matrix for symmetric matrices. The $n^2 \times n^2$ matrix P_{vec} is the vec-permutation matrix [160], characterized by the property $P_{\text{vec}} \cdot \text{vec}(A) = \text{vec}(A^T)$ for all $n \times n$ matrices A. For example, for $n = 2$ we can take

$$M_{\text{symm}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/\sqrt{2} & 0 \\ 0 & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad P_{\text{vec}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

If λ is a real eigenvalue of a Hamiltonian matrix H with normalized right and left eigenvectors x and y, the structured condition number of λ is given by $c_F^{\text{Hamil}}(\lambda) = \|(x \otimes y)^T M_{\text{Hamil}}\|_2 / |y^T x|$. However, as we will see in Section 4.5.2, it is not necessary to compute this expression; there is no difference between the structured and unstructured condition numbers for real eigenvalues of Hamiltonian matrices. ◊

Some complications arise if $\mathbb{K} = \mathbb{R}$ but λ is a complex eigenvalue or M is a complex matrix. In this case, the supremum in (4.8) is also taken over all $p \in \mathbb{R}^m$ but $(\bar{x} \otimes y)^H M$ may be a complex vector. In a similar way as in Section 1.2.2 for the standard eigenvalue condition number we can show

$$\frac{1}{\sqrt{2}|y^H x|} \|(\bar{x} \otimes y)^H M\|_2 \le c_F^{\mathcal{M}}(\lambda) \le \frac{1}{|y^H x|} \|(\bar{x} \otimes y)^H M\|_2, \tag{4.10}$$

see also [146], [277]. To obtain an exact expression for $c_F^{\mathcal{M}}(\lambda)$, let us consider the relation

$$|(\bar{x} \otimes y)^H M p|^2 = \left| \operatorname{Re}\left((\bar{x} \otimes y)^H M\right) p \right|^2 + \left| \operatorname{Im}\left((\bar{x} \otimes y)^H M\right) p \right|^2,$$

which together with (4.8) implies

$$c_F^{\mathcal{M}}(\lambda) = \frac{1}{|y^H x|} \left\| \begin{bmatrix} \operatorname{Re}\left((\bar{x} \otimes y)^H M\right) \\ \operatorname{Im}\left((\bar{x} \otimes y)^H M\right) \end{bmatrix} \right\|_2. \tag{4.11}$$

For a real pattern matrix M, this formula can be rewritten as

$$c_F^{\mathcal{M}}(\lambda) = \frac{1}{|y^H x|} \| [x_R \otimes y_R + x_I \otimes y_I,\ x_R \otimes y_I - x_I \otimes y_R]^T M \|_2, \tag{4.12}$$

where $x = x_R + \imath x_I$ and $y = y_R + \imath y_I$ with $x_R, x_I, y_R, y_I \in \mathbb{R}^n$.

Remark 4.4. It is important to remark that although we have obtained readily computable expressions for structured eigenvalue condition numbers, these formulas tell little about the relationship between unstructured and structured condition numbers. This question must be addressed for each structure individually [146, 163, 188, 256, 277, 323].

$\Diamond$

Nonlinear structures

The task of finding structured eigenvalue condition numbers for nonlinearly structured matrices can be reduced to the linear case if $\mathcal{M}$ is known to be a smooth manifold, see, e.g., [214] for an introduction to manifolds. Given a smooth manifold $\mathcal{M}$, any element $A + E \in \mathcal{M}$ can be approximated in first order by $A + \tilde{E}$ with a suitably chosen $\tilde{E}$ in the tangent space of $\mathcal{M}$ at A. The proof of the following theorem is built on this fact.

Theorem 4.5 ([188]). *Let λ be a simple eigenvalue of the $n \times n$ matrix $A \in \mathcal{M}$, where $\mathcal{M}$ is a smooth real or complex manifold. Then the structured condition number for λ with respect to $\mathcal{M}$ is given by*

$$c^{\mathcal{M}}(\lambda) = \frac{1}{|y^H x|} \sup\left\{ |y^H E x| :\ E \in T_A\mathcal{M}, \|E\| = 1 \right\} = c^{T_A\mathcal{M}}(\lambda), \tag{4.13}$$

where $T_A\mathcal{M}$ is the tangent space of $\mathcal{M}$ at A.

This result shows that the structured condition number of λ with respect to $\mathcal{M}$ boils down to the structured condition number of λ with respect to the linear matrix space $T_A\mathcal{M}$. Hence, we can apply the results from the previous paragraph to obtain computable expressions for $c_F^{\mathcal{M}}(\lambda)$. However, as demonstrated by the following example, it can generally not be expected that the construction of the corresponding pattern matrix M is as simply as in Example 4.3.

Example 4.6. Let $\mathcal{M}$ be the smooth manifold of all $2n \times 2n$ symplectic matrices,

$$\mathcal{M} \equiv \text{symp} = \{S : \ S^T J S = J\}, \quad J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}.$$

The tangent space at $S \in \mathcal{M}$ is given by

$$T_S\mathcal{M} = \{SH : \ JH = -H^T J\} = \{SH : \ H \text{ is Hamiltonian}\}.$$

If M_{Hamil} denotes the pattern matrix for Hamiltonian matrices constructed in Example 4.3 then a pattern matrix M_{symp} for $T_S\mathcal{M}$ can be obtained from a QR decomposition

$$(I \otimes S)M_{\text{Hamil}} = M_{\text{symp}}R,$$

where $M_{\text{symp}} \in \mathbb{R}^{4n^2 \times (3n^2+n)}$ satisfies $M_{\text{symp}}^T M_{\text{symp}} = I$ and R is an upper triangular matrix. By Theorem 4.5, the structured conditon number of a real eigenvalue λ of S is consequently given by $c_F^{\text{symp}}(\lambda) = \|(x \otimes y)^T M_{\text{symp}}\|_2 / |y^T x|$.

Although the described procedure admits the direct calculation of structured eigenvalue condition numbers for symplectic matrices, it should be emphasized that it requires the QR decomposition of a $4n^2 \times (3n^2 + n)$ matrix, which takes $\mathcal{O}(n^6)$ flops and is thus computationally rather expensive.

For a comprehensive treatment of symplectic eigenvalue problems, covering theory, algorithms, and applications, the reader is referred to the book by Faßbender [122]. ◊

Pseudospectra

The *pseudospectrum* of an $n \times n$ matrix A for a perturbation level $\varepsilon \geq 0$ is defined by

$$\Lambda_\varepsilon(A) := \{\lambda \in \mathbb{C} : \ \exists E \in \mathbb{C}^{n \times n}, \ \|E\|_2 \leq \varepsilon, \ \lambda \in \lambda(A + E)\}. \tag{4.14}$$

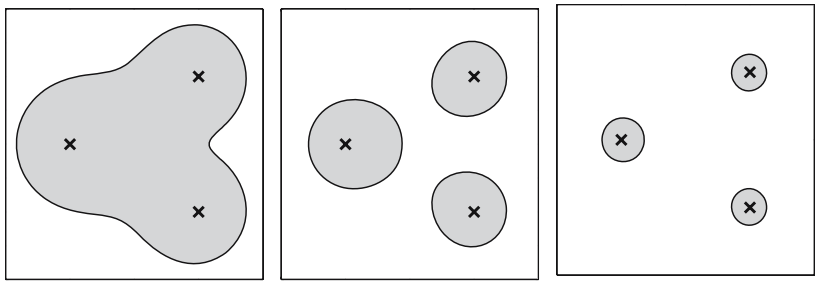
Pseudospectra can be used to characterize the behavior of eigenvalues under finite perturbations, see the recent book by Trefethen and Embree [326] for a comprehensive overview. If all eigenvalues of A are simple then $\Lambda_\varepsilon(A)$ converges for $\varepsilon \to 0$ to discs of radius $c(\lambda) \cdot \varepsilon$ centered around each eigenvalue $\lambda \in \lambda(A)$ [186]. This shows that a simple eigenvalue is equally sensitive in any direction of the complex plane with respect to unstructured perturbations.

The same cannot be said about structured perturbations. For example, if we restrict the perturbation E in (4.14) to be real then the corresponding *real pseudospectrum* converges to ellipses around the eigenvalues [186]. Thus, the eigenvalues can be much less sensitive in some directions of the complex plane. Structured condition numbers, which provide a measure on the most sensitive direction, are not capable to capture such delicacies.

Example 4.7. The unstructured and real pseudospectra of

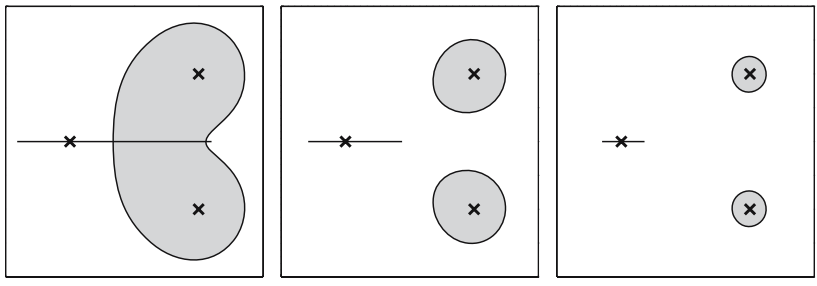
$$A = \begin{bmatrix} 2 & 1 & 2 \\ -1 & 2 & 2 \\ 0 & 0 & 0 \end{bmatrix}. \tag{4.15}$$

are displayed in Figures 4.1 and 4.2, respectively. While the unstructured



$\varepsilon = 0.6$ $\varepsilon = 0.4$ $\varepsilon = 0.2$

Fig. 4.1. Unstructured pseudospectra of the matrix A in (4.15) for different perturbation levels ε.



$\varepsilon = 0.6$ $\varepsilon = 0.4$ $\varepsilon = 0.2$

Fig. 4.2. Real pseudospectra of the matrix A in (4.15) for different perturbation levels ε.

pseudospectrum approaches discs for smaller ε, it can be seen that the real pseudospectra approaches a line, which can be regarded as a degenerate ellipse, for the zero eigenvalue. ◊

Although structured pseudospectra draw a much clearer picture of the behavior of eigenvalues under structured perturbation, it should be mentioned that they are rather complicated and costly to compute, depending on the structure under consideration. More on structured pseudospectra with applications in systems and control theory can be found in the monographs by Karow [186] and Hinrichsen and Pritchard [168].

Invariant Subspaces

As for eigenvalues, a suitable approach to derive the structured condition number for a simple invariant subspace $\mathcal{X}$ is to consider some kind of perturbation expansion for $\mathcal{X}$. Theorem 1.9 shows that an orthonormal basis X of $\mathcal{X}$ changes analytically under perturbations of the matrix A in a sufficiently small open neighborhood $\mathcal{B}(A)$. To recall this result, let the columns of $X_\perp$ form an orthonormal basis for $\mathcal{X}^\perp$ and consider the corresponding block Schur decomposition

$$[X, X_\perp]^H A [X, X_\perp] = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}. \tag{4.16}$$

Then there exists an analytic function $f_X : \mathcal{B}(A) \to \mathbb{C}^{n\times k}$ so that $X = f_X(A)$ and the columns of $\hat{X} = f_X(A+E)$ span an invariant subspace $\hat{\mathcal{X}}$ of $A+E$. Moreover, $X^H(\hat{X} - X) = 0$ and

$$\hat{X} = X - X_\perp \mathbf{T}^{-1}(X_\perp^H E X) + \mathcal{O}(\|E\|_F^2), \tag{4.17}$$

with the Sylvester operator $\mathbf{T} : R \mapsto A_{22}R - RA_{11}$.

Definition 4.8. *Let $\mathcal{X}$ be a simple invariant subspace of a matrix $A \in \mathbb{C}^{n\times n}$ and let $\hat{\mathcal{X}}$ denote the corresponding invariant subspace of the perturbed matrix $A+E$ with $A+E \in \mathcal{M}$ for some $\mathcal{M} \subseteq \mathbb{C}^{n\times n}$ and $\|E\|_F$ sufficiently small. Then the* structured condition number for $\mathcal{X}$ with respect to $\mathcal{M}$ *is defined as*

$$c^{\mathcal{M}}(\mathcal{X}) := \lim_{\varepsilon\to 0} \sup \left\{ \frac{\|\Theta(\mathcal{X}, \hat{\mathcal{X}})\|_F}{\varepsilon} : \ A+E \in \mathcal{M}, \|E\|_F \le \varepsilon \right\}, \tag{4.18}$$

where $\Theta(\mathcal{X}, \hat{\mathcal{X}})$ is the matrix of canonical angles, see Definition 1.11.

Structured condition numbers for eigenvectors and invariant subspaces have been studied in [83, 163, 169, 195, 200, 317]. The (structured) perturbation analysis of quadratic matrix equations is a closely related area, which is comprehensively treated in the book by Konstantinov, Gu, Mehrmann, and Petkov [194], see also [318].

For the sake of simplification, we assume for the rest of this section that $\mathcal{M}$ is a linear matrix space. Along the lines of the proof of Corollary 1.13, the expansion (4.17) implies

$$\begin{aligned} c^{\mathcal{M}}(\mathcal{X}) &= \lim_{\varepsilon\to 0} \frac{1}{\varepsilon} \sup \left\{ \|\mathbf{T}^{-1}(X_\perp^H E X)\|_F : \ A+E \in \mathcal{M}, \|E\|_F \le \varepsilon \right\} \\ &= \lim_{\varepsilon\to 0} \frac{1}{\varepsilon} \sup \left\{ \|\mathbf{T}^{-1}(X_\perp^H E X)\|_F : \ A+E \in \mathcal{M}, \|E\|_F = \varepsilon \right\} \\ &= \sup \left\{ \|\mathbf{T}^{-1}(X_\perp^H E X)\|_F : \ E \in \mathcal{M}, \|E\|_F = 1 \right\}. \end{aligned} \tag{4.19}$$

A pattern matrix approach

Since $\mathcal{M}$ is assumed to be linear, we can again use the pattern matrix approach to develop computable expressions for $c^{\mathcal{M}}(\mathcal{X})$. Let M be a pattern matrix of $\mathcal{M}$ in the sense of (4.7). Moreover, we require the Kronecker product representation of the Sylvester operator $\mathbf{T} : R \mapsto A_{22}R - RA_{11}$ associated with a block Schur decomposition (4.16),

$$K_{\mathbf{T}} = I_{n-k} \otimes A_{11} - A_{22}^T \otimes I_k,$$

see Section 1.2.1. Then

$$\operatorname{vec}(\mathbf{T}^{-1}(X_\perp^H E X)) = K_{\mathbf{T}}^{-1}(X^T \otimes X_\perp^H) \operatorname{vec}(E) = K_{\mathbf{T}}^{-1}(X^T \otimes X_\perp^H)Mp,$$

which together with (4.19) yields the formula

$$c^{\mathcal{M}}(\mathcal{X}) = \sup_{\|p\|_2=1} \|K_{\mathbf{T}}^{-1}(X^T \otimes X_\perp^H)Mp\|_2 = \|K_{\mathbf{T}}^{-1}(X^T \otimes X_\perp^H)M\|_2, \quad (4.20)$$

provided that either p is complex or p, M and X are real.

An orthogonal decomposition approach

Remark 4.4 applies likewise to structured condition numbers for invariant subspaces; although (4.20) yields a directly computable formula for $c^{\mathcal{M}}(\mathcal{X})$, it tells little about the relationship between $c^{\mathcal{M}}(\mathcal{X})$ and the standard condition number $c(\mathcal{X})$.

A sometimes more instructive approach has been introduced in [83] based on previous work on structured condition numbers for invariant subspaces of Hamiltonian matrices [85, 195]. The expression (4.19) reveals that $c^{\mathcal{M}}(\mathcal{X})$ is the norm of the operator $\mathbf{T}^{-1}$ restricted to the linear matrix space

$$\mathcal{N} := \{X_\perp^H E X : \ E \in \mathcal{M}\}.$$

Now, let $\mathcal{L} := \mathbf{T}^{-1}\mathcal{N}$ denote the preimage of $\mathcal{N}$ under $\mathbf{T}$. Then

$$c^{\mathcal{M}}(\mathcal{X}) = \|\mathbf{T}_s^{-1}\|,$$

where $\mathbf{T}_s$ is the restriction of $\mathbf{T}$ to $\mathcal{L} \to \mathcal{N}$, i.e., $\mathbf{T}_s := \mathbf{T}\big|_{\mathcal{L}\to\mathcal{N}}$, and $\|\cdot\|$ denotes the operator norm induced by the Frobenius norm. The operator $\mathbf{T}_s$ can be considered as the part of $\mathbf{T}$ that acts on the linear matrix spaces induced by $\mathcal{M}$.

For many of the structures considered in this chapter, we additionally have the property that the operator $\mathbf{T}^\star : Q \mapsto A_{22}^H Q - Q A_{11}^H$ satisfies $\mathbf{T}^\star : \mathcal{N} \to \mathcal{L}$. Note that $\mathbf{T}^\star$ is the Sylvester operator dual to $\mathbf{T}$:

$$\langle \mathbf{T}(R), Q \rangle = \langle R, \mathbf{T}^\star(Q) \rangle,$$

with the *matrix inner product* $\langle X, Y\rangle = \mathrm{tr}(Y^H X)$. This implies $\mathbf{T} : \mathcal{L}^\perp \to \mathcal{N}^\perp$, where $^\perp$ denotes the orthogonal complement w.r.t. the matrix inner product. Hence, $\mathbf{T}$ decomposes orthogonally into $\mathbf{T}_s$ and $\mathbf{T}_u := \mathbf{T}\big|_{\mathcal{L}^\perp \to \mathcal{N}^\perp}$ and we have

$$c(\mathcal{X}) = \max\{\|\mathbf{T}_s^{-1}\|, \|\mathbf{T}_u^{-1}\|\}. \tag{4.21}$$

This shows that comparing $c(\mathcal{X})$ with $c^{\mathcal{M}}(\mathcal{X})$ is equivalent to comparing $\|\mathbf{T}_u^{-1}\|$ with $\|\mathbf{T}_s^{-1}\|$.

Example 4.9. Consider the embedding of a complex matrix $B + \mathrm{i}C$, with $B, C \in \mathbb{R}^{n\times n}$, into a real $2n \times 2n$ matrix of the form

$$A = \begin{bmatrix} B & C \\ -C & B \end{bmatrix}.$$

Let the columns of $Y + \mathrm{i}Z$ and $Y_\perp + \mathrm{i}Z_\perp$, where $Y, Z \in \mathbb{R}^{n\times k}$ and $Y_\perp, Z_\perp \in \mathbb{R}^{n\times(n-k)}$, form orthonormal bases for an invariant subspace of $B + \mathrm{i}C$ and its orthogonal complement, respectively. Then the columns of $X = \left[\begin{smallmatrix} Y & Z \\ -Z & Y \end{smallmatrix}\right]$ and $X_\perp = \left[\begin{smallmatrix} Y_\perp & Z_\perp \\ -Z_\perp & Y_\perp \end{smallmatrix}\right]$ form orthonormal bases for an invariant subspace $\mathcal{X}$ of A and $\mathcal{X}^\perp$, respectively. This corresponds to the block Schur decomposition

$$[X, X_\perp]^T A [X, X_\perp] =: \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline 0 & A_{22} \end{array}\right] = \left[\begin{array}{cc|cc} B_{11} & C_{11} & B_{12} & C_{12} \\ -C_{11} & B_{11} & -C_{12} & B_{12} \\ \hline 0 & 0 & B_{22} & C_{22} \\ 0 & 0 & -C_{22} & B_{22} \end{array}\right]$$

with the associated Sylvester operator $\mathbf{T} : R \mapsto A_{22}R - RA_{11}$.

If we consider only perturbations having the same structure as A then $\mathcal{M} = \left\{\left[\begin{smallmatrix} F & G \\ -G & F \end{smallmatrix}\right]\right\}$ and

$$\mathcal{N} := X_\perp^T \mathcal{M} X = \left\{\begin{bmatrix} F_{21} & G_{21} \\ -G_{21} & F_{21} \end{bmatrix}\right\}, \quad \mathcal{N}^\perp = \left\{\begin{bmatrix} F_{21} & G_{21} \\ G_{21} & -F_{21} \end{bmatrix}\right\}.$$

Moreover, we have $\mathbf{T} : \mathcal{N} \to \mathcal{N}$ and $\mathbf{T}^\star : \mathcal{N} \to \mathcal{N}$. The restricted operator $\mathbf{T}_s := \mathbf{T}\big|_{\mathcal{N}\to\mathcal{N}}$ becomes singular only if $B_{11} + \mathrm{i}C_{11}$ and $B_{22} + \mathrm{i}C_{22}$ have eigenvalues in common, while $\mathbf{T}_u := \mathbf{T}\big|_{\mathcal{N}^\perp\to\mathcal{N}^\perp}$ becomes singular if $B_{11}+\mathrm{i}C_{11}$ and $B_{22} - \mathrm{i}C_{22}$ have eigenvalues in common. Thus, there are situations in which the unstructured condition number $c(\mathcal{X}) = \max\{\|\mathbf{T}_s^{-1}\|, \|\mathbf{T}_u^{-1}\|\}$ can be significantly larger than the structured condition number $c^{\mathcal{M}}(\mathcal{X}) = \|\mathbf{T}_s^{-1}\|$, e.g., if $\mathrm{i}\gamma$ is nearly an eigenvalue of $B_{11}+\mathrm{i}C_{11}$ while $-\mathrm{i}\gamma$ is nearly an eigenvalue of $B_{22} + \mathrm{i}C_{22}$ for some $\gamma \in \mathbb{R}$. ◊

The introductionary example (4.2) is a special case of Example 4.9, where the unstructured condition number is infinite due to the purely imaginary eigenvalue pair $\pm i$. The results above suggest that the structured condition number is finite and given by

$$c^{\mathcal{M}}(\mathcal{X}) = \inf_{|\beta|^2+|\gamma|^2=1} \left\{ \left\| \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \beta & \gamma \\ -\gamma & \beta \end{bmatrix} - \begin{bmatrix} \beta & \gamma \\ -\gamma & \beta \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \right\|_F \right\}^{-1} = \frac{1}{2}.$$

However, all our arguments so far rest on the perturbation expansion (4.17), which requires the invariant subspace to be simple; a condition that is not satisfied by (4.2). This restriction is removed in the following section by adapting the global perturbation analysis for invariant subspaces developed by Stewart [301], see also Section 1.2.4.

Global perturbation bounds

As in Section 1.2.4, we consider the *perturbed* block Schur decomposition

$$[X, X_\perp]^H (A+E)[X, X_\perp] = \begin{bmatrix} A_{11}+E_{11} & A_{12}+E_{12} \\ E_{21} & A_{22}+E_{22} \end{bmatrix} =: \begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} \\ E_{21} & \hat{A}_{22} \end{bmatrix}. \quad (4.22)$$

In order to obtain a formula for $\hat{X}$, a basis for the perturbed invariant subspace $\hat{\mathcal{X}}$ near $\mathcal{X}$, we look for an invertible matrix of the form $W = \left[\begin{smallmatrix} I & 0 \\ -R & I \end{smallmatrix}\right]$ so that

$$W^{-1} \begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} \\ E_{21} & \hat{A}_{22} \end{bmatrix} W = \begin{bmatrix} \hat{A}_{11} - \hat{A}_{12}R & \hat{A}_{12} \\ E_{21} + R\hat{A}_{11} - \hat{A}_{22}R - R\hat{A}_{12}R & \hat{A}_{22} + R\hat{A}_{12} \end{bmatrix}$$

is in block upper triangular form. This implies that R is a solution of the quadratic matrix equation

$$\hat{A}_{22}R - R\hat{A}_{11} + R\hat{A}_{12}R = E_{21}. \quad (4.23)$$

The following theorem gives conditions for the existence of such a solution if the perturbation E is known to be contained in a linear matrix space $\mathcal{M}$.

Theorem 4.10. *Consider a perturbed block Schur decomposition (4.22) and let* $\hat{\mathbf{T}} : R \mapsto \hat{A}_{22}R - R\hat{A}_{11}$. *Set* $\mathcal{N} = \{E_{21} : E \in \mathcal{M}\}$ *and assume that there exists a linear matrix space* $\mathcal{L}$, *having the same dimension as* $\mathcal{N}$, *such that* $\hat{\mathbf{T}} : \mathcal{L} \to \mathcal{N}$ *and* $R\hat{A}_{12}R \in \mathcal{N}$ *for all* $R \in \mathcal{L}$. *Moreover, assume* $\hat{\mathbf{T}}_s := \hat{\mathbf{T}}\big|_{\mathcal{L}\to\mathcal{N}}$ *to be invertible.*

If $\|\hat{A}_{12}\|_F \, \|\hat{A}_{21}\|_F < 1/(2\|\hat{\mathbf{T}}_s^{-1}\|)^2$ *then there exists a solution* $R \in \mathcal{L}$ *of the quadratic matrix equation (4.23) with*

$$\|R\|_F < 2\|\hat{\mathbf{T}}_s^{-1}\| \, \|E_{21}\|_F.$$

Proof. The result can be proven by constructing an iteration

$$R_0 \leftarrow 0, \quad R_{i+1} \leftarrow \hat{\mathbf{T}}_s^{-1}(E_{21} - R_i\hat{A}_{12}R_i),$$

which is well-defined because $R_i \in \mathcal{L}$ implies $R_i\hat{A}_{12}Q_i \in \mathcal{N}$. This approach is identical to the approach taken in the proof of Lemma 1.14, which was based on techniques developed in [299, 301, 308]. In fact, it can be shown in precisely the same way as in [308, Thm. 2.11] that all iterates R_i satisfy a bound of the form $\|R_i\|_F \le 2\|\hat{\mathbf{T}}_s^{-1}\| \, \|E_{21}\|_F - \delta$ for some $\delta > 0$ and converge to a solution of (4.23). □

Having obtained a solution R of (4.23), a basis for an invariant subspace $\hat{\mathcal{X}}$ of $A+E$ is given by $\hat{X} = \begin{bmatrix} I \\ -R \end{bmatrix}$. Together with Lemma 1.12, this leads to the following global structured perturbation result for invariant subspaces.

Corollary 4.11. *Under the assumptions of Theorem 4.10 there exists an invariant subspace $\hat{\mathcal{X}}$ of $A+E$ so that*

$$\|\tan\Theta(\mathcal{X},\hat{\mathcal{X}})\|_F < 2\|\hat{\mathbf{T}}_s^{-1}\|\,\|E_{21}\|_F. \tag{4.24}$$

The quantity $\|\hat{\mathbf{T}}_s^{-1}\|$ in the bound (4.24) can be related to $\|\mathbf{T}_s^{-1}\|$, the norm of the inverse of the unperturbed Sylvester operator, by using the following lemma.

Lemma 4.12. *Given a perturbed block Schur decomposition of the form (4.22), let $\mathbf{T} : R \mapsto A_{22}R - RA_{11}$ and $\hat{\mathbf{T}} : R \mapsto \hat{A}_{22}R - R\hat{A}_{11}$. Set $\mathcal{N} = \{E_{21} : E \in \mathcal{M}\}$ and assume that there exists a linear matrix space $\mathcal{L}$, having the same dimension as $\mathcal{N}$, such that $\mathbf{T} : \mathcal{L} \to \mathcal{N}$ and $\hat{\mathbf{T}} : \mathcal{L} \to \mathcal{N}$. If $\hat{\mathbf{T}}_s := \hat{\mathbf{T}}\big|_{\mathcal{L}\to\mathcal{N}}$ is invertible and satisfies*

$$\|\hat{\mathbf{T}}_s^{-1}\| < \frac{1}{\|E_{11}\|_F + \|E_{22}\|_F}, \tag{4.25}$$

then $\mathbf{T}_s := \mathbf{T}\big|_{\mathcal{L}\to\mathcal{N}}$ is invertible and satisfies

$$\|\mathbf{T}_s^{-1}\| \le \frac{\|\hat{\mathbf{T}}_s^{-1}\|}{1 - \|\hat{\mathbf{T}}_s^{-1}\|(\|E_{11}\|_F + \|E_{22}\|_F)}. \tag{4.26}$$

Proof. Let $\triangle : R \mapsto E_{22}R - RE_{11}$, then $\triangle : \mathcal{L} \to \mathcal{N}$ and $\mathbf{T}_s = \hat{\mathbf{T}}_s - \triangle$. Considering $\hat{\mathbf{T}}_s^{-1} \circ \triangle$, the composition of $\hat{\mathbf{T}}_s^{-1}$ and $\triangle$, the inequality (4.25) implies

$$\|\hat{\mathbf{T}}_s^{-1} \circ \triangle\| \le \|\hat{\mathbf{T}}_s^{-1}\|\,\|\triangle\| \le \|\hat{\mathbf{T}}_s^{-1}\|(\|E_{11}\|_F + \|E_{22}\|_F) < 1.$$

Thus, the Neumann series

$$\sum_{i=0}^{\infty} (\hat{\mathbf{T}}_s^{-1} \circ \mathbf{T}_s)^i \circ \hat{\mathbf{T}}_s^{-1}$$

converges to $\mathbf{T}_s^{-1}$, which also proves (4.26). ◊

Combining Corollary 4.11 with Lemma 4.12 shows

$$\|\tan\Theta(\mathcal{X},\hat{\mathcal{X}})\|_F \le \alpha \frac{\|\mathbf{T}_s^{-1}\|\,\|E\|_F}{1 - \sqrt{2}\|\mathbf{T}_s^{-1}\|\,\|E\|_F} = \alpha\|\mathbf{T}_s^{-1}\|\,\|E\|_F + \mathcal{O}(\|E\|_F^2).$$

with $\alpha = 2$. The presence of the factor $\alpha = 2$ in this bound is artificial; a slight modification of the proof of Theorem 4.10 shows that α can be made

arbitrarily close to one under the assumption that the perturbation E is sufficiently small. This shows that $c^{\mathcal{M}}(\mathcal{X})$, the structured condition number of $\mathcal{X}$, is bounded from above by $\|\mathbf{T}_s^{-1}\|$, even if the operator $\mathbf{T}$ itself is not invertible. Whether $\|\mathbf{T}_s^{-1}\|$ coincides with the condition number in the case of a singular $\mathbf{T}$ depends on the uniqueness of the matrix space $\mathcal{L}$ in Theorem 4.10, see [83] for more details.

Example 4.13 (Continuation of Example 4.9). If

$$\lambda(B_{11} + \mathrm{i}C_{11}) \cap \lambda(B_{22} + \mathrm{i}C_{22}) = \emptyset$$

but

$$\lambda(B_{11} + \mathrm{i}C_{11}) \cap \lambda(B_{22} - \mathrm{i}C_{22}) \neq \emptyset$$

then $\mathcal{L} = \mathcal{N}$ is the unique choice which yields $\mathbf{T} : \mathcal{L} \to \mathcal{N}$ and an invertible $\mathbf{T}_s = \mathbf{T}\big|_{\mathcal{L}\to\mathcal{N}}$. Thus, $c^{\mathcal{M}}(\mathcal{X}) = \|\mathbf{T}_s^{-1}\| = 1/\operatorname{sep}(B_{11} + \mathrm{i}C_{11}, B_{22} + \mathrm{i}C_{22})$ also holds for singular $\mathbf{T}$. This verifies $c^{\mathcal{M}}(\mathcal{X}) = 1/2$ for the introductionary example (4.2). ◊

4.1.2 Structured Backward Error

Having computed an eigenvalue $\hat{\lambda}$ and an eigenvector $\hat{x}$ with $\|\hat{x}\|_2 = 1$ of a matrix A, it is natural to ask whether the result of this computation corresponds to an exact eigenvalue/eigenvector pair of a slightly perturbed matrix $A + E$. The answer to this question can be used to assess the numerical backward stability of the employed algorithm. Mathematically, the *backward error* is defined by

$$\eta(\hat{\lambda}, \hat{x}) = \min\{\|E\|_F : \ (A + E)\hat{x} = \hat{\lambda}\hat{x}, \ E \in \mathbb{C}^{n\times n}\}.$$

It turns out that this quantity simplifies to the norm of the residual, $\eta(\hat{\lambda}, \hat{x}) = \|A\hat{x} - \hat{\lambda}\hat{x}\|_2$ [141]. If the eigenvector $\hat{x}$ is not available or not of interest, it is more appropriate to consider the backward error for $\hat{\lambda}$ alone,

$$\eta(\hat{\lambda}) = \min\{\|E\|_2 : \ \det(A + E - \hat{\lambda}I) = 0, \ E \in \mathbb{C}^{n\times n}\}. \tag{4.27}$$

By the famous Schmidt-Mirsky theorem, we have $\eta(\hat{\lambda}) = \sigma_{\min}(A - \hat{\lambda}I)$.

The *structured backward error* for a matrix $A \in \mathcal{M}$ with respect to a set $\mathcal{M} \subseteq \mathbb{C}^{n\times n}$ is similarly defined as

$$\eta^{\mathcal{M}}(\hat{\lambda}, \hat{x}) = \inf\{\|E\|_F : \ (A + E)\hat{x} = \hat{\lambda}\hat{x}, \ E \in \mathcal{M}\}. \tag{4.28}$$

Evaluating $\eta^{\mathcal{M}}(\hat{\lambda}, \hat{x})$ can be used to assess the strong numerical backward stability of the numerical method that has been used to obtain $\hat{\lambda}$ and $\hat{x}$. Once again, the pattern matrix approach yields a computable expression for (4.28),

provided that $\mathcal{M}$ is a linear matrix space of $\mathbb{C}^{n\times n}$ [163, 323]. Let $M \in \mathbb{C}^{n^2\times m}$ denote a pattern matrix for $\mathcal{M}$. Then

$$(\hat{\lambda}I - A)\hat{x} - E\hat{x} = r - (\hat{x}^T \otimes I_n)\,\mathrm{vec}(E) = r - (\hat{x}^T \otimes I)Mp,$$

where $r = (A - \hat{\lambda})\hat{x}$. Thus, (4.28) is equivalent to the linear least-squares problem

$$\eta^{\mathcal{M}}(\hat{\lambda}, \hat{x}) = \inf\{\|p\|_2 :\ (\hat{x}^T \otimes I_n)Mp = r,\ p \in \mathbb{C}^m\},$$

which can be solved, e.g., by applying a singular value decomposition to the matrix $(\hat{x}^T \otimes I)M$ [48]. When $\mathcal{M}$ is a linear matrix space of $\mathbb{R}^{n\times n}$ and $\hat{\lambda}, \hat{x}$ are real then p will be automatically real. Complications arise for the combination real $\mathcal{M}$ but complex $\hat{\lambda}$, in which case the real and imaginary parts of the constraint $(A + E)\hat{x} = \hat{\lambda}\hat{x}$ in (4.28) must be considered separately [323].

Remark 4.4 applies likewise to the structured backward error; the fact that $\eta^{\mathcal{M}}(\hat{\lambda}, \hat{x})$ is computable tells nothing about its relationship to $\eta(\hat{\lambda}, \hat{x})$. This question must be addressed for each structure individually, see, e.g., [314]. Moreover, only in exceptional cases is it possible to derive results on the structured backward error of nonlinearly structured eigenvalue problems, see, e.g., [169, 316].

A structured backward error $\eta^{\mathcal{M}}(\hat{\lambda})$ for the eigenvalue alone can be defined similarly as in (4.27). However, even for linear structures it is difficult to evaluate $\eta^{\mathcal{M}}(\hat{\lambda})$, which is closely related to the concept of structured singular values [186, 168]. For some structures, however, such as symmetric and Toeplitz matrices [276, 277], it can be shown that $\eta^{\mathcal{M}}(\hat{\lambda}) = \eta(\hat{\lambda})$.

4.1.3 Algorithms and Efficiency

Structure-preserving algorithms have the potential to solve eigenvalue problems in a more accurate and more efficient way than general-purpose algorithms. An ideal algorithm tailored to the matrix structure would

- be strongly backward stable in the sense of Bunch [66], i.e., the computed eigenvalues and invariant subspaces are the exact eigenvalues and invariant subspaces of matrix with the same structure;
- be reliable, i.e., capable to solve all eigenvalue problems in the considered matrix class; and
- require an equal amount of (or even less) computational work than a competitive general-purpose method.

For some structures – including block cyclic, skew-Hamiltonian and symmetric matrices – such an ideal method is well established, while for others it still remains an open problem to design a method that meets all three requirements satisfactorily.

The fact that a structured matrix depends on less than n^2 parameters gives rise to the hope that an algorithm taking advantage of the structure requires

substantially less effort than a general-purpose algorithm. For example, the QR algorithm applied to a symmetric matrix requires roughly 10% of the flops required by the same algorithm applied to a general matrix [141]. For other structures, such as Hamiltonian matrices, the difference may be less significant, see Section 4.5.8. Moreover, in view of recent progress made in improving the performance of the QR algorithm it may require considerable implementation efforts to turn this reduction of flops into an actual reduction of computational time.

The convergence of the Krylov-Schur algorithm and other iterative methods strongly depends on the properties of the matrix A and the subset of eigenvalues to be computed, which makes the computational cost rather difficult to predict. For methods based on Krylov subspaces, each iteration requires a matrix-vector multiplication. Linear structures almost always admit the more efficient computation of these matrix-vector multiplications, to a varying degree of extent. Some structured matrices, such as skew-Hamiltonian and block cyclic matrices, induce some structure in the Krylov subspace making it possible to reduce the computational effort even further.

4.2 Products of Matrices

In this section, we consider the problem of computing eigenvalues and invariant subspaces of a matrix product

$$\Pi_{\mathcal{A}} = A^{(p)} A^{(p-1)} \cdots A^{(1)}, \tag{4.29}$$

where $A^{(1)}, \ldots, A^{(p)}$ are $n \times n$ matrices. At first sight, such an eigenvalue problem does not match the definition of a structured eigenvalue problem stated in the beginning of this chapter; $\Pi_{\mathcal{A}}$ is an $n \times n$ matrix but depends on the pn^2 entries defining its p factors $A^{(1)}, \ldots, A^{(p)}$. However, any product eigenvalue problem can be equivalently seen as an $pn \times pn$ eigenvalue problem with block cyclic structure. This section is much about using this connection to address several tasks related to (4.29) in a convenient manner.

Probably the best known example for (4.29) is the singular value decomposition of a square matrix A [141]. It is a basic linear algebra fact that the eigenvalues of $A^T A$ or AA^T are the squared singular values of A. Product eigenvalue problems involving up to six factors appear in various areas of systems and control theory, such as model reduction of linear time-invariant (descriptor) systems [12, 249, 312, 325], linear-quadratic optimal control [29, 32], and $\mathcal{H}_\infty$ control [33, 165], see also Appendix A. Other applications, where the number of factors can be arbitrarily large, include queueing network models [57, 309], as well as computational methods for analyzing bifurcations and computing Floquet multipliers of ordinary and partial differential equations [121, 224, 225]. Product eigenvalue problems are used to address a number of computational tasks related to periodic discrete-time systems, see [338]

and the references therein. By applying appropriate sampling and discretization techniques, many physical and chemical processes exhibiting seasonal or periodic behavior can be described by such systems [246]. An interesting example for a periodic discrete-time system, describing a model for the dynamics of nitrogen absorption, distribution, and translocation in citrus trees, can be found in [64]. To obtain the eigenvalues of this system, an eigenvalue problem consisting of 365 factors has to be solved. On the theoretical side, the product eigenvalue problem provides a powerful unifying concept for addressing other structured eigenvalue problems [357].

4.2.1 Structured Decompositions

The following decomposition plays the same central role for the product eigenvalue problem (4.29) that the Schur decomposition plays for the standard eigenvalue problem or the generalized Schur decomposition for the generalized eigenvalue problem.

Theorem 4.14 (Periodic Schur decomposition [52, 159]). *Let $A^{(1)}, \ldots, A^{(p)} \in \mathbb{R}^{n\times n}$, then there exist orthogonal matrices $Q^{(1)}, \ldots, Q^{(p)} \in \mathbb{R}^{n\times n}$ such that*

$$\begin{aligned} T^{(p)} &= Q^{(1)T} A^{(p)} Q^{(p)}, \\ T^{(p-1)} &= Q^{(p)T} A^{(p-1)} Q^{(p-1)}, \\ &\vdots \\ T^{(1)} &= Q^{(2)T} A^{(1)} Q^{(1)}, \end{aligned} \tag{4.30}$$

where $T^{(p)}$ has real Schur form and $T^{(1)}, \ldots, T^{(p-1)}$ are upper triangular matrices.

The periodic Schur decomposition (4.30) can be written in the more compact form

$$T^{(l)} = Q^{(l+1)T} A^{(l)} Q^{(l)}, \quad l = 1, \ldots, p,$$

if we identify $Q^{(p+1)}$ with $Q^{(1)}$. More generally spoken, we will make use of the following convention:

Throughout the entire section we identify $\star^{(l)}$ with $\star^{(l-1 \mod p)+1}$, where $\star$ can be replaced by any symbol.

There is also a complex version of the periodic Schur decomposition (4.30), i.e., there are unitary matrices $Q^{(1)}, \ldots, Q^{(p)}$ so that the matrices

$$T^{(l)} = Q^{(l+1)H} A^{(l)} Q^{(l)}, \quad l = 1, \ldots, p, \tag{4.31}$$

are upper triangular. This implies a Schur decomposition for $\Pi_{\mathcal{A}}$:

$$Q^{(1)H} \Pi_{\mathcal{A}} Q^{(1)} = T^{(p)} T^{(p-l)} \cdots T^{(1)} = \begin{bmatrix} \ddots & \\ & \ddots \end{bmatrix}. \tag{4.32}$$

Hence, if $t_{ii}^{(l)}$ denotes the ith diagonal element of $T^{(l)}$, then the n eigenvalues of Π_A are given by the n products $t_{ii}^{(p)} t_{ii}^{(p-1)} \cdots t_{ii}^{(1)}$, $i = 1, \ldots, n$. By a suitable reordering of the periodic Schur decomposition, see [147, 159] and Section 4.2.4, we can let the eigenvalues of Π_A appear in any desirable order on the diagonals of $T^{(l)}$.

The Schur decomposition (4.32) also implies that the first k columns of $Q^{(1)}$ span an invariant subspace of $\Pi_{\mathcal{A}}$. More generally, it can be shown that if we consider all cyclic permutations

$$\Pi_{\mathcal{A}}^{(l)} = A^{(p+l-1)} A^{(p+l-2)} \cdots A^{(l)}, \quad l = 1, \ldots, p, \tag{4.33}$$

then the first k columns of $Q^{(l)}$ span an invariant subspace of $\Pi_{\mathcal{A}}^{(l)}$ for each $l \in [1, p]$. These invariant subspaces can be related to certain invariant subspaces of the *block cyclic matrix*

$$\mathcal{A} = \begin{bmatrix} 0 & & & A^{(p)} \\ A^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & A^{(p-1)} & 0 \end{bmatrix}. \tag{4.34}$$

To see this, let us partition

$$Q^{(l)} = \begin{matrix} \\ \end{matrix} \overset{\begin{matrix} k & n-k \end{matrix}}{\begin{bmatrix} X^{(l)}, & X_{\perp}^{(l)} \end{bmatrix}}, \quad T^{(l)} = \begin{matrix} k \\ n-k \end{matrix} \overset{\begin{matrix} k & n-k \end{matrix}}{\begin{bmatrix} A_{11}^{(l)} & A_{12}^{(l)} \\ 0 & A_{22}^{(l)} \end{bmatrix}}.$$

By setting

$$X = X^{(1)} \oplus X^{(2)} \oplus \cdots \oplus X^{(p)}, \quad X_{\perp} = X_{\perp}^{(1)} \oplus X_{\perp}^{(2)} \oplus \cdots \oplus X_{\perp}^{(p)}, \tag{4.35}$$

and

$$\mathcal{A}_{ij} = \begin{bmatrix} 0 & & & A_{ij}^{(p)} \\ A_{ij}^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & A_{ij}^{(p-1)} & 0 \end{bmatrix}, \tag{4.36}$$

we obtain a block Schur decomposition for $\mathcal{A}$:

$$\mathcal{A}[X, X_{\perp}] = [X, X_{\perp}] \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ 0 & \mathcal{A}_{22} \end{bmatrix}. \tag{4.37}$$

In particular, $\mathcal{X} = \operatorname{span}(X)$ is an invariant subspace of $\mathcal{A}$ belonging to the eigenvalues of the block cyclic matrix $\mathcal{A}_{11}$. Not every invariant subspace of $\mathcal{A}$ admits a block cyclic representation but in the context of product eigenvalue problems it is sufficient to consider this type of subspace.

Definition 4.15 ([222]). *Let $\mathcal{X}$ be a (pk)-dimensional (left) invariant subspace of a block cyclic matrix $\mathcal{A} \in \mathbb{R}^{pn \times pn}$. If there exist matrices $X^{(1)}$, $X^{(2)}$, $\ldots$, $X^{(p)} \in \mathbb{R}^{n \times k}$ so that*

$$\mathcal{X} = \operatorname{span}(X^{(1)} \oplus X^{(2)} \oplus \cdots \oplus X^{(p)}),$$

then $\mathcal{X}$ is a so called (right) periodic invariant subspace *of $\mathcal{A}$.*

By direct computation, it can be seen that every periodic invariant subspace admits a block cyclic representation. For $k = 1$, this yields the following well-known relationship between the eigenvalues of $\Pi_{\mathcal{A}}$ and $\mathcal{A}$.

Corollary 4.16. *Let λ be an eigenvalue of the matrix product $\Pi_{\mathcal{A}}$ having the form (4.29). Then $\lambda^{1/p}, \omega\lambda^{1/p}, \ldots, \omega^{p-1}\lambda^{1/p}$, where ω is the pth primitive root of unity, are eigenvalues of the block cyclic matrix $\mathcal{A}$ having the form (4.34).*

Proof. By the (complex) periodic Schur decomposition and the construction given above, there exists an invariant subspace of $\mathcal{A}$ block Schur decomposition of the form (4.37) with

$$\mathcal{A}_{11} = \begin{bmatrix} 0 & & & t_{11}^{(p)} \\ t_{11}^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & t_{11}^{(p-1)} & 0 \end{bmatrix}, \quad \lambda = t_{11}^{(p)} t_{11}^{(p-1)} \cdots t_{11}^{(1)}.$$

The result follows by observing $\mathcal{A}_{11}^p = \lambda I_p$. □

Bhatia [45, Sec. VIII.5] calls a p-tuple of the form $\{\alpha, \omega\alpha, \ldots, \omega^{k-1}\alpha\}$ a *p-Carrollian* tuple in honor of the writer and mathematician Lewis Carroll. As any periodic invariant admits a block cyclic representation it does also belong to eigenvalues that form a set of p-Carrollian tuples.

4.2.2 Perturbation Analysis

Benner, Mehrmann and Xu [43] as well as Lin and Sun [221] analyzed the effect of perturbations in the factors of a product eigenvalue problem (4.29). In this section, we show how these perturbation results can be derived via a different approach; by treating the product eigenvalue problem as a structured eigenvalue problem involving the block cyclic matrix $\mathcal{A}$.

As in the perturbation analysis for standard eigenvalue problems, see Section 1.2, we start with a block Schur decomposition

$$\mathcal{A}[X, X_\perp] = [X, X_\perp] \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ 0 & \mathcal{A}_{22} \end{bmatrix}. \tag{4.38}$$

As we will only consider periodic invariant subspaces, we can assume some extra structure: X and $X_\perp$ have block diagonal form (4.35); $\mathcal{A}_{11} \in \mathbb{C}^{pk \times pk}$,

$\mathcal{A}_{12} \in \mathbb{C}^{pk\times p(n-k)}$ and $\mathcal{A}_{22} \in \mathbb{C}^{p(n-k)\times p(n-k)}$ are block cyclic matrices of the form (4.36).

The Sylvester operator associated with (4.38) has a number of useful properties that are summarized in the following lemma.

Lemma 4.17. *Let* $\mathbf{T} : \mathbb{C}^{pk\times p(n-k)} \to \mathbb{C}^{pk\times p(n-k)}$ *be the Sylvester operator defined by* $\mathbf{T} : R \mapsto \mathcal{A}_{11}R - R\mathcal{A}_{22}$, *where* $\mathcal{A}_{11}$ *and* $\mathcal{A}_{22}$ *are block cyclic matrices of the form (4.36). Then the following statements hold:*

1. *Let* $\mathcal{Z}^{(j)}$, $j = 0, \ldots, p-1$, *be the matrix subspaces*

$$\mathcal{Z}^{(j)} := \left\{ \mathcal{C}_k^j \left(Z^{(1)} \oplus Z^{(2)} \oplus \cdots \oplus Z^{(p)} \right) : \ Z^{(l)} \in \mathbb{C}^{pk\times p(n-k)} \right\}, \quad (4.39)$$

where

$$\mathcal{C}_k = \begin{bmatrix} 0 & & & I_k \\ I_k & \ddots & & \\ & \ddots & \ddots & \\ & & I_k & 0 \end{bmatrix}.$$

Then

$$\mathbf{T}\mathcal{Z}^{(0)} \subseteq \mathcal{Z}^{(1)}, \ \ldots, \ \mathbf{T}\mathcal{Z}^{(p-2)} \subseteq \mathcal{Z}^{(p-1)}, \ \mathbf{T}\mathcal{Z}^{(p-1)} \subseteq \mathcal{Z}^{(0)}. \quad (4.40)$$

2. $\mathbf{T}$ *is invertible if and only if* $\lambda(\mathcal{A}_{11}) \cap \lambda(\mathcal{A}_{22}) = \emptyset$, *which is equivalent to the condition* $\lambda\left(A_{11}^{(p)} A_{11}^{(p-1)} \cdots A_{11}^{(1)}\right) \cap \lambda\left(A_{22}^{(p)} A_{22}^{(p-1)} \cdots A_{22}^{(1)}\right) = \emptyset$.
3. *If* $\mathbf{T}$ *is invertible, then*

$$\mathcal{Z}^{(0)} = \mathbf{T}^{-1}\mathcal{Z}^{(1)}, \ \ldots, \ \mathcal{Z}^{(p-2)} = \mathbf{T}^{-1}\mathcal{Z}^{(p-1)}, \ \mathcal{Z}^{(p-1)} = \mathbf{T}^{-1}\mathcal{Z}^{(0)}.$$

Proof. 1. A matrix R is an element of $\mathcal{Z}^{(j)}$ if and only if there exist block diagonal matrices $R_1, R_2 \in \mathcal{Z}^{(0)}$ so that $R = \mathcal{C}_k^j R_1 = R_2 \mathcal{C}_{n-k}^j$. We have

$$\begin{aligned} \mathcal{A}_{11}R &= \left(A_{11}^{(p)} \oplus A_{11}^{(1)} \oplus \cdots A_{11}^{(p-1)}\right) \mathcal{C}_k^{j+1} R_1 \\ &\in \left(A_{11}^{(p)} \oplus A_{11}^{(1)} \oplus \cdots A_{11}^{(p-1)}\right) \mathcal{Z}^{(j+1)} \subseteq \mathcal{Z}^{(j+1)} \end{aligned}$$

and

$$\begin{aligned} R\mathcal{A}_{22} &= R_2 \mathcal{C}_{n-k}^{j+1} \left(A_{22}^{(1)} \oplus A_{22}^{(2)} \oplus \cdots A_{22}^{(p)}\right) \\ &\in \mathcal{Z}^{(j+1)} \left(A_{22}^{(1)} \oplus A_{22}^{(2)} \oplus \cdots A_{22}^{(p)}\right) \subseteq \mathcal{Z}^{(j+1)}, \end{aligned}$$

where, using $\mathcal{C}_k^p = I$ and $\mathcal{C}_{n-k}^p = I$, the subspace $\mathcal{Z}^{(p)}$ can be identified with $\mathcal{Z}^{(0)}$. This shows $\mathbf{T}\mathcal{Z}^{(j)} \subseteq \mathcal{Z}^{(j+1)}$.

2. This statement follows from Lemma 1.3 and

$$\lambda \in \lambda(\mathcal{A}_{ii}) \quad \Leftrightarrow \quad \lambda^p \in \lambda\left(A_{ii}^{(p)} A_{ii}^{(p-1)} \cdots A_{ii}^{(1)}\right), \quad i \in \{1, 2\}.$$

3. The fact that $\{\mathcal{Z}^{(0)}, \mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(p-1)}\}$ forms an orthogonal basis for the matrix space $\mathbb{C}^{pk \times p(n-k)}$ with respect to the inner product $\langle A, B \rangle = \operatorname{tr}(B^H A)$ implies for invertible $\mathbf{T}$ that the set inclusions (4.40) become

$$\mathbf{T}\mathcal{Z}^{(0)} = \mathcal{Z}^{(1)}, \ \ldots, \ \mathbf{T}\mathcal{Z}^{(p-2)} = \mathcal{Z}^{(p-1)}, \ \mathbf{T}\mathcal{Z}^{(p-1)} = \mathcal{Z}^{(0)},$$

which concludes the proof.

□

If $\mathbf{T}$ is invertible then the left invariant subspace belonging to the eigenvalues of $\mathcal{A}_{11}$ is given by

$$\mathcal{Y} = \operatorname{span}\left([X, X_\perp] \cdot [I, \mathbf{T}^{-1}\mathcal{A}_{12}]^H\right).$$

The third part of Lemma 4.17 shows $\mathbf{T}^{-1}\mathcal{A}_{12} \in \mathcal{Z}^{(0)}$, i.e., this matrix is block diagonal. Hence, $\mathcal{Y}$ is a left *periodic* invariant subspace.

Invariant Subspaces

Another consequence of Lemma 4.17 is that the Sylvester operator $\mathbf{T}$ decomposes orthogonally with respect to the structure. If we let $\mathcal{M} \equiv \text{cyc}$ denote the set of all $pn \times pn$ block cyclic matrices and

$$\mathcal{L} := \mathcal{Z}^{(0)}, \quad \mathcal{N} := \mathcal{Z}^{(1)} = \{X_\perp^T E X : \ E \in \text{cyc}\},$$

then the second part of Lemma 4.17 implies $\mathbf{T} : \mathcal{L} \to \mathcal{N}$ and $\mathbf{T} : \mathcal{L}^\perp \to \mathcal{N}^\perp$. By the results in Section 4.1.1, the structured condition number for the invariant subspace $\mathcal{X}$ spanned by the columns of the matrix X in (4.38) satisfies $c^{\text{cyc}}(\mathcal{X}) = \|\mathbf{T}_s^{-1}\|$ with $\mathbf{T}_s := \mathbf{T}\big|_{\mathcal{L} \to \mathcal{N}}$. Moreover, we have for the unstructured condition number

$$c(\mathcal{X}) = \max\{\|\mathbf{T}_s^{-1}\|, \|\mathbf{T}_u^{-1}\|\}$$

with $\mathbf{T}_u := \mathbf{T}\big|_{\mathcal{L}^\perp \to \mathcal{N}^\perp}$.

The following example demonstrates that the quantities $c^{\text{cyc}}(\mathcal{X})$ and $c(\mathcal{X})$ may differ significantly.

Example 4.18. Let $p = 2$, $\mathcal{A}_{11} = \left[\begin{smallmatrix} 0 & 0 \\ 1 & 0 \end{smallmatrix}\right]$ and $\mathcal{A}_{22} = \left[\begin{smallmatrix} 0 & C \\ D & 0 \end{smallmatrix}\right]$, where

$$C = \begin{bmatrix} 10^5 & 10^5 \\ 0 & 10^{-5} \end{bmatrix}, \quad D = \begin{bmatrix} 10^{-5} & 0 \\ 0 & 10^5 \end{bmatrix}.$$

Then the structured condition number is given by

$$c^{\text{cyc}}(\mathcal{X}) = \left\| \begin{bmatrix} C & -I_2 \\ 0 & D \end{bmatrix}^{-1} \right\|_2 = \sqrt{2} \times 10^5,$$

while the unstructured condition number is much higher,

$$c(\mathcal{X}) = \max\left\{c^{\text{cyc}}(X), \left\|\begin{bmatrix} D & -I_2 \\ 0 & C \end{bmatrix}^{-1}\right\|_2\right\} = 10^{10}.$$

$\Diamond$

The quantity $c^{\text{cyc}}(\mathcal{X})$ measures the overall conditioning of $\mathcal{X}$, which can be written as the direct sum of the subspaces $\mathcal{X}^{(1)} = \text{span}\left(X^{(1)}\right), \ \ldots, \ \mathcal{X}^{(p)} = \text{span}\left(X^{(p)}\right)$. Since $\mathcal{X}^{(1)}$ is an invariant subspace of the matrix product $\Pi_{\mathcal{A}}$ it might be of interest to measure the individual conditioning of $\mathcal{X}^{(1)}$. In this case, the appropriate structured condition number is given by

$$c^{\text{cyc}(1)}(\mathcal{X}) := \sup\{\|E_1^T \, \mathbf{T}_{\mathcal{A}}^{-1}(\mathcal{E}_{21})\|_F : \mathcal{E}_{21} \in \mathcal{N}, \ \|\mathcal{E}_{21}\|_F = 1\},$$

where E_1 consists of the first $(n-k)$ columns of the identity matrix $I_{p(n-k)}$, see also [221]. In a similar fashion, one can derive condition numbers for the individual subspaces $\mathcal{X}^{(2)}, \ldots, \mathcal{X}^{(p)}$.

On the computation of $c^{\text{cyc}}(\mathcal{X})$

Computing the structured condition number $c^{\text{cyc}}(\mathcal{X})$ is equivalent to computing the norm of the inverse of the Sylvester operator $\mathbf{T}_s : \mathcal{L} \to \mathcal{N}$ with $\mathbf{T}_s : R \mapsto \mathcal{A}_{22}R - R\mathcal{A}_{11}$. If we let

$$\begin{aligned} R &= (R^{(1)} \oplus \cdots \oplus R^{(p)}), \\ \mathbf{T}_s(R) &= \mathcal{C}_{n-k}(R^{(1)} \oplus \cdots \oplus R^{(p)}), \end{aligned}$$

where $\mathcal{C}_{n-k}$ is defined as in Lemma 4.17, then

$$\begin{aligned} A_{22}^{(p)} R^{(p)} - R^{(1)} A_{11}^{(p)} &= R^{(p)}, \\ A_{22}^{(1)} R^{(1)} - R^{(2)} A_{11}^{(1)} &= R^{(1)}, \\ &\vdots \\ A_{22}^{(p-1)} R^{(p-1)} - R^{(p)} A_{11}^{(p-1)} &= R^{(p-1)}. \end{aligned} \tag{4.41}$$

The system of matrix equations (4.41) is called a *periodic Sylvester equation*. It is equivalent to the linear system of equations

$$K_{\mathbf{T}_s} \begin{bmatrix} \text{vec}(R^{(1)}) \\ \vdots \\ \text{vec}(R^{(p-1)}) \\ \text{vec}(R^{(p)}) \end{bmatrix} = \begin{bmatrix} \text{vec}(R^{(1)}) \\ \vdots \\ \text{vec}(R^{(p-1)}) \\ \text{vec}(R^{(p)}) \end{bmatrix}, \tag{4.42}$$

where $K_{\mathbf{T}_s}$ can be written as the sum of a block diagonal and a block cyclic matrix:

$$K_{\mathbf{T}_s} = \begin{bmatrix} I_k \otimes A_{22}^{(1)} & -A_{11}^{(1)} \otimes I_{n-k} & & \\ & \ddots & \ddots & \\ & & I_k \otimes A_{22}^{(p-1)} & -A_{11}^{(p-1)} \otimes I_{n-k} \\ -A_{11}^{(p)} \otimes I_{n-k} & & & I_k \otimes A_{22}^{(p)} \end{bmatrix}.$$

Thus $\|K_{\mathbf{T}_s}^{-1}\|_2 = c^{\mathrm{cyc}}(\mathcal{X})$, which can be estimated by applying a norm estimator [164, Ch. 14] to $K_{\mathbf{T}_s}^{-1}$. This amounts to the solution of a few linear equations $K_{\mathbf{T}_s} q_1 = r_1$ and $K_{\mathbf{T}_s}^T q_2 = r_2$ for particularly chosen right hand sides r_1 and r_2 or, equivalently, the solution of a few periodic Sylvester equations. Efficient methods for solving (4.41) and periodic matrix equations of similar type can be found in [86, 297, 335].

Eigenvalues

We have seen that to any eigenvalue λ of the product eigenvalue problem (4.29) there exists a p-dimensional periodic invariant subspace $\mathcal{X}$ belonging to the eigenvalues $\lambda^{1/k}$, $\omega\lambda^{1/k}$, ..., $\omega^{k-1}\lambda^{1/k}$ of $\mathcal{A}$, where ω is the pth primitive root of unity. Now, we can choose vectors $x^{(1)}, \ldots, x^{(p)} \in \mathbb{C}^n$ with $\|x^{(1)}\|_2 = \cdots = \|x^{(p)}\|_2 = 1$ so that the columns of $X = x^{(1)} \oplus \cdots \oplus x^{(p)}$ form an orthonormal basis for $\mathcal{X}$. Similarly, there exist vectors $y^{(1)}, \ldots, y^{(p)} \in \mathbb{C}^n$, $\|y^{(1)}\|_2 = \cdots = \|y^{(p)}\|_2 = 1$, so that $\mathcal{Y} = \mathrm{span}(Y)$ with $Y = y^{(1)} \oplus \cdots \oplus y^{(p)}$ is a left periodic invariant subspace belonging to the same set of eigenvalues.

The invariant subspace $\mathcal{X}$ has the representation

$$\mathcal{A}X = X\mathcal{A}_{11}, \quad \mathcal{A}_{11} = \begin{bmatrix} 0 & & & \alpha^{(p)} \\ \alpha^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & \alpha^{(p-1)} & 0 \end{bmatrix},$$

for some scalars $\alpha^{(1)}, \ldots, \alpha^{(p)}$. We assume that λ is a simple eigenvalue which implies that $\mathcal{X}$ is a simple invariant subspace. Consider a block cyclic perturbation

$$\mathcal{E} = \begin{bmatrix} 0 & & & E^{(p)} \\ E^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & E^{(p-1)} & 0 \end{bmatrix},$$

then the perturbation expansion (1.20) in Theorem 1.9 proves the existence of a matrix $\hat{\mathcal{A}}_{11}$ which satisfies

$$\hat{\mathcal{A}}_{11} = \mathcal{A}_{11} + (Y^H X)^{-1} Y^H \mathcal{E} X + \mathcal{O}(\|\mathcal{E}\|^2), \tag{4.43}$$

where the eigenvalues of $\hat{\mathcal{A}}_{11}$ are eigenvalues of $\mathcal{A}+\mathcal{E}$. Since $\mathcal{A}_{11}$ and $(Y^HX)^{-1}Y^H\mathcal{E}X$ are block cyclic matrices, the matrix $\hat{\mathcal{A}}_{11}$ can also be assumed to be block cyclic.

The expression (4.43) can be rewritten in terms of the entries $\hat{\alpha}^{(l)}$ of $\hat{\mathcal{A}}_{11}$:

$$\hat{\alpha}^{(l)} = \alpha^{(l)} + \frac{1}{y^{(l+1)H}x^{(l)}} y^{(l+1)H}E^{(l)}x^{(l)} + \mathcal{O}(\|\mathcal{E}\|^2), \quad l = 1, \ldots, p.$$

The structured condition number for each individual $\alpha^{(l)}$ is thus given by $c_2^{\mathrm{cyc}}(\alpha^{(l)}) = c_F^{\mathrm{cyc}}(\alpha^{(l)}) = 1/|y^{(l+1)H}x^{(l)}|$. If we measure the change of all quantities using

$$\triangle\alpha := \Big(\sum_{l=1}^{p} |\hat{\alpha}^{(l)} - \alpha^{(l)}|^2\Big)^{1/2} = \|\hat{\mathcal{A}}_{11} - \mathcal{A}_{11}\|_F,$$

then

$$\triangle\alpha = \|(Y^HX)^{-1}Y^H\mathcal{E}X\|_F + \mathcal{O}(\|\mathcal{E}\|^2) \le \|(Y^HX)^{-1}\|_2\,\|\mathcal{E}\|_F + \mathcal{O}(\|\mathcal{E}\|^2).$$

For the block cyclic perturbations $\mathcal{E} = \varepsilon Y\mathcal{C}_1X^H$ (the matrix $\mathcal{C}_1$ is defined in Lemma 4.17), this inequality is attained in first order. Hence, the structured condition number for $\alpha = [\alpha_1, \ldots, \alpha_p]$ satisfies

$$\begin{aligned} c_F^{\mathrm{cyc}}(\alpha) &:= \lim_{\varepsilon\to 0}\frac{1}{\varepsilon}\sup\{\triangle\alpha :\ \mathcal{E}\in\mathrm{cyc},\ \|\mathcal{E}\|_F \le \varepsilon\} \\ &= \|(Y^HX)^{-1}\|_2 = \max\{c^{\mathrm{cyc}}(\alpha^{(1)}), \ldots, c^{\mathrm{cyc}}(\alpha^{(p)})\}. \end{aligned}$$

Note that $c^{\mathrm{cyc}}(\alpha)$ coincides with the standard condition number for the eigenvalue mean of $\mathcal{A}_{11}$, see Section 1.2.3.

These results can be used to bound the distance between the eigenvalues $\lambda = \prod \alpha^{(l)}$ and $\hat{\lambda} = \prod \hat{\alpha}^{(l)}$ of the matrix products $\Pi_{\mathcal{A}} = A^{(p)}A^{(p-1)}\cdots A^{(1)}$ and

$$\Pi_{\hat{\mathcal{A}}} = \Big(A^{(p)} + E^{(p)}\Big)\Big(A^{(p-1)} + E^{(p-1)}\Big)\cdots\Big(A^{(1)} + E^{(1)}\Big),$$

respectively. We obtain

$$\begin{aligned} |\hat{\lambda} - \lambda| &= \Big|\prod_{l=1}^{p}[(\hat{\alpha}^{(l)} - \alpha^{(l)}) + \alpha^{(l)}] - \prod_{l=1}^{p}\alpha^{(l)}\Big| \\ &= \Big|\sum_{l=1}^{p}\Big[(\hat{\alpha}^{(l)} - \alpha^{(l)})\prod_{k\neq l}\alpha^{(k)}\Big]\Big| + \mathcal{O}(\|\mathcal{E}\|^2) \\ &\le \sum_{l=1}^{p}\Big[|\hat{\alpha}^{(l)} - \alpha^{(l)}|\prod_{k\neq l}|\alpha^{(k)}|\Big] + \mathcal{O}(\|\mathcal{E}\|^2) \\ &\le \sum_{l=1}^{p}\Big[c_2^{\mathrm{cyc}}(\alpha^{(l)})\,\|E^{(l)}\|_2\prod_{k\neq l}|\alpha^{(k)}|\Big] + \mathcal{O}(\|\mathcal{E}\|^2). \end{aligned}$$

4.2.3 The Periodic QR Algorithm

To solve the product eigenvalue problem, one could in principal apply a general-purpose eigenvalue solver to the explicitely formed matrix product $\Pi_{\mathcal{A}}$. For the sake of numerical stability, however, it is important to take the fact that $\Pi_{\mathcal{A}}$ is a matrix product into account and develop numerical methods that work directly on the factors $A^{(1)}, \ldots, A^{(p)}$. This well-appreciated principle has driven the development of many reliable algorithms for solving instances of the product eigenvalue problem, such as the Golub-Reinsch algorithm for the singular value decomposition [138] or the QZ algorithm for the generalized eigenvalue problem. Both methods have in common that they are numerically backward stable with respect to the factors, i.e., the computed eigenvalues and invariant subspaces correspond to a matrix product with slightly perturbed factors.

A method that meets this goal for the matrix product

$$\Pi_{\mathcal{A}} = A^{(p)} A^{(p-1)} \cdots A^{(1)}, \tag{4.44}$$

is the *periodic QR algorithm* [52, 159, 331]. By a computing a periodic Schur decomposition, it achieves a small factor-wise backward error, i.e., the computed eigenvalues are the exact eigenvalues of

$$(A^{(p)} + E^{(p)})(A^{(p-1)} + E^{(p-1)}) \cdots (A^{(1)} + E^{(1)}). \tag{4.45}$$

where each $\|E^{(l)}\|_F$ is of order $\mathbf{u}\,\|A^{(l)}\|_F$. The periodic QR algorithm was developed independently in the beginning of the 1990's by Bojanczyk, Golub, and Van Dooren [52], and Hench and Laub [159]. Already in 1975, Van Loan [331] published an algorithm for the case $p = 2$ comprising the main ideas behind the periodic QR algorithm. Van Loan's paper is based on his PhD thesis [330], which is a valuable source of information, particularly in providing important details concerning the reliable implementation of the periodic QR algorithm.

Although the periodic QR algorithm was to some extent connected to the standard QR algorithm in the works mentioned above, it has been introduced as an independent algorithm. In the following, we show that the periodic QR algorithm naturally arises from the standard QR algorithm by considering a shuffled version of the block cyclic matrix $\mathcal{A}$ associated with (4.44), see also [199].

The Perfect Shuffle

The *perfect shuffle* is a permutation that can be used to turn a block structured matrix into a structured block matrix. This is a common trick in (numerical) linear algebra; it has been used, e.g., to construct algorithms for block Toeplitz matrices [132]. The block structure of our interest is a block cyclic matrix of the form

$$\mathcal{A} = \begin{bmatrix} 0 & & & A^{(p)} \\ A^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & A^{(p-1)} & 0 \end{bmatrix}. \tag{4.46}$$

The perfect shuffle permutation can be obtained as follows. Let

$$z = [\, z^{(1)}, z^{(2)}, \ldots, z^{(p)} \,],$$

where $z^{(l)}$ is a row vector of length n. Imagine that each $z^{(l)}$ represents a deck of n cards. A perfect shuffle stacks exactly one card from each deck, rotationally until all decks are exhausted. The row vector that corresponds to the shuffled deck is given by

$$\tilde{z} = [\, z_1^{(1)}, z_1^{(2)}, \ldots, z_1^{(p)}, z_2^{(1)}, \ldots, z_2^{(p)}, \ldots, z_n^{(1)}, \ldots, z_n^{(p)} \,].$$

There exists a unique permutation matrix $P_p \in \mathbb{R}^{pn \times pn}$ such that $\tilde{z} = zP_p$ (for notational convenience we will drop the subscript and simply write P in the following). Applying this permutation to the block cyclic matrix $\mathcal{A}$ turns it into an $n \times n$ block matrix with cyclic $p \times p$ blocks:

$$\tilde{\mathcal{A}} := P^T \mathcal{A} P = \begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & & \vdots \\ A_{n1} & \cdots & A_{nn} \end{bmatrix}, \; A_{ij} := \begin{bmatrix} 0 & & & a_{ij}^{(p)} \\ a_{ij}^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & a_{ij}^{(p-1)} & 0 \end{bmatrix}. \tag{4.47}$$

Any matrix of the form (4.47) will be called a *cyclic block matrix*. Similarly, applying P to a block diagonal matrix $\mathcal{D}$ yields an $n \times n$ block matrix $\tilde{\mathcal{D}} = P^T \mathcal{D} P$ with diagonal $p \times p$ matrices as entries. We refer to any matrix of the latter form as a *diagonal block matrix*. The following straightforward lemma reveals a useful relation between cyclic and diagonal block matrices.

Lemma 4.19. *Let $\tilde{\mathcal{A}}$ be a cyclic block matrix and let $\tilde{\mathcal{D}}_1$, $\tilde{\mathcal{D}}_2$ be diagonal block matrices. Then $\tilde{\mathcal{D}}_1 \tilde{\mathcal{A}} \tilde{\mathcal{D}}_2$ is again a cyclic block matrix.*

The perfect shuffle version of the periodic Schur form

Using the perfect shuffle, the periodic Schur decomposition can be interpreted as a structured Schur decomposition for cyclic block matrices, see also Figure 4.3.

Corollary 4.20. *Let $\tilde{\mathcal{A}} \in \mathbb{R}^{pn \times pn}$ be a cyclic block matrix of the form (4.47). Then there exists an orthogonal diagonal block matrix $\tilde{Q}$ so that $\tilde{Q}^T \tilde{\mathcal{A}} \tilde{Q}$ is a block upper triangular matrix with $p \times p$ and $2p \times 2p$ diagonal blocks.*

Proof. The result is obtained by applying Theorem 4.14 to the $n \times n$ coefficient matrices $A^{(1)}, \ldots, A^{(p)}$ of the block cyclic matrix $\mathcal{A} = P^T \tilde{\mathcal{A}} P$ and setting

$$\tilde{Q} = P^T (Q^{(1)} \oplus Q^{(2)} \oplus \cdots \oplus Q^{(p)}) P,$$

where P is the perfect shuffle matrix. □

periodic Schur form of a matrix product

$$\begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ 0 & c_{22} & c_{23} & c_{24} \\ 0 & c_{32} & c_{33} & c_{34} \\ 0 & 0 & 0 & c_{44} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ 0 & b_{22} & b_{23} & b_{24} \\ 0 & 0 & b_{33} & b_{34} \\ 0 & 0 & 0 & b_{44} \end{bmatrix} \times \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

block Schur form of corresponding cyclic block matrix

$$\left[\begin{array}{ccc:ccc:ccc:ccc} 0 & 0 & c_{11} & 0 & 0 & c_{12} & 0 & 0 & c_{13} & 0 & 0 & c_{14} \\ a_{11} & 0 & 0 & a_{12} & 0 & 0 & a_{13} & 0 & 0 & a_{14} & 0 & 0 \\ 0 & b_{11} & 0 & 0 & b_{12} & 0 & 0 & b_{13} & 0 & 0 & b_{14} & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & c_{22} & 0 & 0 & c_{23} & 0 & 0 & c_{24} \\ 0 & 0 & 0 & a_{22} & 0 & 0 & a_{23} & 0 & 0 & a_{24} & 0 & 0 \\ 0 & 0 & 0 & 0 & b_{22} & 0 & 0 & b_{23} & 0 & 0 & b_{24} & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & c_{32} & 0 & 0 & c_{33} & 0 & 0 & c_{34} \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{33} & 0 & 0 & a_{34} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{33} & 0 & 0 & b_{34} & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c_{44} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{44} & 0 \end{array}\right]$$

Fig. 4.3. Relation between periodic Schur form and block Schur form of a cyclic block matrix.

Reduction to periodic Hessenberg form

We have seen that reducing a general matrix $A \in \mathbb{R}^{n \times n}$ to Hessenberg form is a preliminary step in the QR algorithm in order to reduce its computational complexity. Algorithm 1 is the basic algorithm for such a reduction based on Householder matrices. Let us recall that a Householder matrix which maps the last $n - j$ elements of a vector $x \in \mathbb{R}^n$ to zero is given by

$$H_j(x) = I - \beta v_j(x) v_j(x)^T,$$

where

$$v_j(x) = \begin{bmatrix} 0 & 0 \\ 0 & I_{n-j+1} \end{bmatrix} x + \operatorname{sign}(e_j^T x) \left\| \begin{bmatrix} 0 & 0 \\ 0 & I_{n-j+1} \end{bmatrix} x \right\|_2 e_j$$

and

$$\beta = \begin{cases} 0 & \text{if } v_j(x) = 0, \\ 2/(v_j(x)^T v_j(x)) & \text{otherwise.} \end{cases}$$

The following theorem shows that Algorithm 1 preserves cyclic block structures.

Theorem 4.21. *If Algorithm 1 is applied to a cyclic block matrix $\tilde{\mathcal{A}} \in \mathbb{R}^{np\times np}$ then an orthogonal diagonal block matrix $\tilde{\mathcal{Q}}$ and a cyclic block matrix $\tilde{\mathcal{Q}}^T\tilde{\mathcal{A}}\tilde{\mathcal{Q}}$ in upper Hessenberg form are returned.*

Proof. Assume that after $(j-1)$ loops of Algorithm 1 the matrix $\tilde{\mathcal{A}}$ has been overwritten by a cyclic block matrix. Then,

$$\tilde{\mathcal{A}}e_j = y \otimes e_{l'}, \quad y = \left[\, a_{1k}^{(l)}, a_{2k}^{(l)}, \ldots, a_{nk}^{(l)} \,\right]^T, \tag{4.48}$$

where $l' = j \bmod p+1$, $l = (j-1) \bmod p+1$ and $k = (j-l)/p+1$. Since

$$v_{j+1}(\tilde{\mathcal{A}}e_j) = v_{j+1}(y \otimes e_{l'}) = \begin{cases} l < p : v_k(y) \otimes e_{l'}, \\ l = p : v_{k+1}(y) \otimes e_{l'}, \end{cases}$$

it follows that $H_{j+1}(\tilde{\mathcal{A}}e_j)$ is a diagonal block matrix. Thus, Lemma 4.19 shows that the j-th loop of Algorithm 1 preserves the cyclic block form of $\tilde{\mathcal{A}}$. The statement about $\tilde{\mathcal{Q}}$ is a consequence of the group property of orthogonal diagonal block matrices. □

Hence, Algorithm 1 applied to $\tilde{\mathcal{A}}$ only operates on the entries $a_{ij}^{(l)}$; it should thus be possible to reformulate this algorithm in terms of operations on the factors $A^{(1)}, \ldots, A^{(p)}$. In the following, we will derive such a reformulation. First note that the proof of Theorem 4.21 also shows that $\tilde{\mathcal{A}} \leftarrow H_{j+1}(\tilde{\mathcal{A}}e_j)^T\tilde{\mathcal{A}}H_{j+1}(\tilde{\mathcal{A}}e_j)$ is equivalent to the updates

$$\begin{cases} l < p : A^{(l+1)} \leftarrow A^{(l+1)} \cdot H_k(A^{(l)}e_k),\ A^{(l)} \leftarrow H_k(A^{(l)}e_k)^T \cdot A^{(l)}, \\ l = p : A^{(1)} \leftarrow A^{(1)} \cdot H_{k+1}(A^{(p)}e_k), \quad A^{(p)} \leftarrow H_{k+1}(A^{(p)}e_k)^T \cdot A^{(p)}, \end{cases}$$

where the quantities k and l are defined as in (4.48). Furthermore, if we set

$$\tilde{\mathcal{Q}} = P^T \mathrm{diag}(Q^{(1)}, Q^{(2)}, \ldots, Q^{(p)})P,$$

then $\tilde{\mathcal{Q}} \leftarrow \tilde{\mathcal{Q}}H_{j+1}(\tilde{\mathcal{A}}e_j)$ equals $Q^{(l+1)} \leftarrow Q^{(l+1)}H_k(A^{(l)}e_k)$ for $l < p$ and $Q^{(1)} \leftarrow Q^{(1)}H_{k+1}(A^{(p)}e_k)$ for $l = p$. Altogether, we can rewrite Algorithm 1 as shown in Algorithm 18.

Note that this algorithm corresponds to the algorithm for reduction to periodic Hessenberg form described in [52]. It requires roughly p times the flops required by Algorithm 1, reduction to standard Hessenberg form, applied to an $n \times n$ matrix. Let us emphasize again that Algorithm 18 performs exactly the same operations as Algorithm 1 applied to $\tilde{\mathcal{A}}$. Hence, also in the presence of roundoff errors both algorithms produce the same result, an entity that is commonly called *numerical equivalence*.

Example 4.22. If $p = 2$ and $A^{(1)} = A^{(2)T}$ then Algorithm 1 reduces the symmetric matrix $\tilde{\mathcal{A}}$ to tridiagonal form. On the other hand, Algorithm 18 returns $Q^{(2)T}A^{(1)}Q^{(1)}$ in bidiagonal form. Hence, as a special case we obtain that bidiagonal reduction (see, e.g, [305, Alg. 3.2]) applied to $A^{(1)}$ is numerically equivalent to tridiagonal reduction applied to $\tilde{\mathcal{A}}$. A similar observation has been made by Paige [259]. ◊

Algorithm 18 Reduction to periodic Hessenberg form

Input: Matrices $A^{(1)}, \ldots, A^{(p)} \in \mathbb{R}^{n \times n}$.

Output: Orthogonal matrices $Q^{(1)}, \ldots, Q^{(p)}$ such that $H^{(l)} = Q^{(l+1)T} A^{(l)} Q^{(l)}$ is upper triangular for $l = 1, \ldots, p-1$ and $H^{(p)} = Q^{(1)T} A^{(p)} Q^{(p)}$ is in upper Hessenberg form. Each matrix $A^{(l)}$ is overwritten by $H^{(l)}$.

$Q^{(1)} \leftarrow I_n, Q^{(2)} \leftarrow I_n, \ldots, Q^{(p)} \leftarrow I_n$
for $k \leftarrow 1, \ldots, n-1$ **do**
  for $l \leftarrow 1, \ldots, p-1$ **do**
    $Q^{(l+1)} \leftarrow Q^{(l+1)} \cdot H_k(A^{(l)} e_k)$
    $A^{(l+1)} \leftarrow A^{(l+1)} \cdot H_k(A^{(l)} e_k)$
    $A^{(l)} \leftarrow H_k(A^{(l)} e_k) A^{(l)}$
  end for
  $Q^{(1)} \leftarrow Q^{(1)} \cdot H_{k+1}(A^{(p)} e_k)$
  $A^{(1)} \leftarrow A^{(1)} \cdot H_{k+1}(A^{(p)} e_k)$
  $A^{(p)} \leftarrow H_{k+1}(A^{(p)} e_k) \cdot A^{(p)}$
end for

periodic Hessenberg form of a matrix product

$$\begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ 0 & c_{32} & c_{33} & c_{34} \\ 0 & 0 & c_{43} & c_{44} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ 0 & b_{22} & b_{23} & b_{24} \\ 0 & 0 & b_{33} & b_{34} \\ 0 & 0 & 0 & b_{44} \end{bmatrix} \times \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

Hessenberg form of corresponding cyclic block matrix

$$\left[\begin{array}{ccc:ccc:ccc:ccc} 0 & 0 & c_{11} & 0 & 0 & c_{12} & 0 & 0 & c_{13} & 0 & 0 & c_{14} \\ a_{11} & 0 & 0 & a_{12} & 0 & 0 & a_{13} & 0 & 0 & a_{14} & 0 & 0 \\ 0 & b_{11} & 0 & 0 & b_{12} & 0 & 0 & b_{13} & 0 & 0 & b_{14} & 0 \\ \hdashline 0 & 0 & c_{21} & 0 & 0 & c_{22} & 0 & 0 & c_{23} & 0 & 0 & c_{24} \\ 0 & 0 & 0 & a_{22} & 0 & 0 & a_{23} & 0 & 0 & a_{24} & 0 & 0 \\ 0 & 0 & 0 & 0 & b_{22} & 0 & 0 & b_{23} & 0 & 0 & b_{24} & 0 \\ \hdashline 0 & 0 & 0 & 0 & 0 & c_{32} & 0 & 0 & c_{33} & 0 & 0 & c_{34} \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{33} & 0 & 0 & a_{34} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{33} & 0 & 0 & b_{34} & 0 \\ \hdashline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c_{43} & 0 & 0 & c_{44} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{44} & 0 \end{array}\right]$$

Fig. 4.4. Relation between periodic Hessenberg form and Hessenberg form of a cyclic block matrix.

Periodic QR Iteration

A QR iteration applied to a general Hessenberg matrix A as described in Algorithm 2 first performs an update of the form

$$A \leftarrow H_1(x) \cdot A \cdot H_1(x),$$

where x is the first column of the shift polynomial belonging to m Francis shifts. This introduces a bulge in the top left corner of A. The second step

of a QR iteration consists of restoring the Hessenberg form of A using Algorithm 1. The following theorem shows that QR iterations preserve cyclic block structures if m is wisely chosen.

Theorem 4.23. *If Algorithm 2 is applied to a cyclic block matrix $\tilde{\mathcal{A}} \in \mathbb{R}^{np\times np}$ in Hessenberg form and the number of Francis shifts is an integer multiple of p, say $m = pt$, then the structure of $\tilde{\mathcal{A}}$ is preserved and an orthogonal diagonal block matrix $\tilde{\mathcal{Q}}$ is returned.*

Proof. The bottom right $m\times m$ submatrix of $\tilde{\mathcal{A}}$ is a cyclic block matrix. Thus, the Francis shifts form a p-Carrollian tuple and can be partitioned into groups $\{\sigma_i^{(1)},\dots,\sigma_i^{(p)}\}$, $i = 1,\dots,t$, where each group contains the pth roots of some $\gamma_i \in \mathbb{C}$. Using the fact that $\Pi_{\mathcal{A}} = A^{(p)}A^{(p-1)}\cdots A^{(1)}$ is the upper left $n \times n$ block of the block diagonal matrix $(P\tilde{\mathcal{A}}P^T)^p$ we obtain

$$\begin{aligned} x &= \prod_{i=1}^{t}\prod_{l=1}^{p}(\tilde{\mathcal{A}} - \sigma_i^{(l)} I_{np})e_1 = \prod_{i=1}^{t}\left(\tilde{\mathcal{A}}^p - \prod_{l=1}^{p}\sigma_i^{(l)} I_{np}\right)e_1 \\ &= P^T \cdot \prod_{i=1}^{t}\left((P\tilde{\mathcal{A}}P^T)^p - \gamma_i I_{np}\right)Pe_1 = \left(\prod_{i=1}^{t}(\Pi_{\mathcal{A}} - \gamma_i I_n)e_1\right)\otimes e_1. \end{aligned}$$

Thus, $H_1(x)$ is a block diagonal matrix, which together with Theorem 4.21 concludes the proof. □

The subdiagonal entries of a cyclic block $\tilde{\mathcal{A}}$ in Hessenberg form consist of the diagonal entries of the triangular matrices $A^{(1)}$, …, $A^{(p-1)}$ and the subdiagonal entries of the Hessenberg matrix $A^{(p)}$. Hence, $\tilde{\mathcal{A}}$ is unreduced if and only if all the triangular factors are nonsingular and the Hessenberg factor is unreduced. Similar to Hessenberg reduction the proof of Theorem 4.23 gives a way to rewrite Algorithm 2, the standard QR iteration, in terms of operations on the factors $A^{(1)},\dots,A^{(p)}$, see Algorithm 19.

Note that Algorithm 19 is a "Householder version" of the periodic QR iteration with t shifts described in [52]. It requires roughly p times the flops required by Algorithm 2 applied to an $n \times n$ matrix with t shifts.

Example 4.24. This is a continuation of Example 4.22. If $A^{(1)}$ and $A^{(2)} = A^{(1)T}$ satisfy the assumptions of Algorithm 19 then $A^{(1)}$ is a bidiagonal matrix with nonzero diagonal and supdiagonal elements. Algorithm 2 applied to the tridiagonal matrix $\tilde{\mathcal{A}}$ performs an implicit shifted symmetric QR iteration, while Algorithm 19 performs a bidiagonal QR iteration [138]. This shows that both QR iterations are numerically equivalent. ◊

Deflation

A deflation occurs when one of the subdiagonal entries of $\tilde{\mathcal{A}}$ becomes sufficiently small. The usual criterion is to declare a subdiagonal entry negligible

Algorithm 19 Periodic QR iteration

Input: Nonsingular upper triangular matrices $A^{(1)}, \ldots, A^{(p-1)} \in \mathbb{R}^{n \times n}$ and a matrix $A^{(p)} \in \mathbb{R}^{n \times n}$ in unreduced upper Hessenberg form, an integer $t \in [2, n]$.

Output: Orthogonal matrices $Q^{(1)}, \ldots, Q^{(p)}$ such that $Q^{(l+1)T} A^{(l)} Q^{(l)}$, $l = 1, \ldots, p$, are the factors of the cyclic block matrix that would have been obtained after one QR iteration with $m = pt$ Francis shifts has been applied to $\tilde{\mathcal{A}}$. Each matrix $A^{(l)}$ is overwritten by $Q^{(l+1)T} A^{(l)} Q^{(l)}$.

1. Compute $\{\gamma_1, \ldots, \gamma_t\}$ as the eigenvalues of $\Pi_A(n-t+1:n, n-t+1:n)$.
2. Set $x = (\Pi_A - \gamma_1 I_n)(\Pi_A - \gamma_2 I_n) \cdots (\Pi_A - \gamma_t I_n) e_1$.
3. Update $A^{(1)} \leftarrow A^{(1)} \cdot H_1(x)$, $\quad A^{(p)} \leftarrow H_1(x) \cdot A^{(p)}$.
4. Apply Algorithm 18 to compute orthogonal matrices $Q^{(1)}, \ldots, Q^{(p)}$ so that $A^{(1)} \leftarrow Q^{(2)T} A^{(1)} Q^{(1)}, \ldots, A^{(p-1)} \leftarrow Q^{(p)T} A^{(p-1)} Q^{(p-1)}$ are upper triangular and $A^{(p)} \leftarrow Q^{(1)T} A^{(p)} Q^{(p)}$ is in Hessenberg form.
5. Update $Q^{(1)} \leftarrow H_1(x) \cdot Q^{(1)}$.

if it is small compared to the neighboring diagonal elements, see Section 1.3.4. However, this is not a very meaningful choice for matrices with zero diagonal elements like $\tilde{\mathcal{A}}$. Considering the action of the Householder matrices in the course of a QR iteration it is advisable to base the criterion on the two closest nonzero elements in the same row and column. Suitable generic criteria for $\tilde{\mathcal{A}}$ are thus given by

$$|a^{(p)}_{j+1,j}| \leq \mathbf{u}(|a^{(p)}_{j,j}| + |a^{(p)}_{j+1,j+1}|), \tag{4.49}$$

$$|a^{(l)}_{jj}| \leq \mathbf{u}(|a^{(l)}_{j-1,j}| + |a^{(l)}_{j,j+1}|), \quad l = 1, \ldots, p-1, \tag{4.50}$$

where an entry on the right hand side of (4.50) is replaced by zero if the index is out of range. Note that inequality (4.50) may only be satisfied if the 2-norm condition number of $A^{(l)}$ is at least $1/(2\mathbf{u})$.

Situation (4.49) is easily handled, setting $a^{(p)}_{j+1,j}$ zero makes $\tilde{\mathcal{A}}$ block upper triangular,

$$\tilde{\mathcal{A}} = \begin{bmatrix} \tilde{\mathcal{A}}_{11} & \tilde{\mathcal{A}}_{12} \\ 0 & \tilde{\mathcal{A}}_{22} \end{bmatrix},$$

where $\tilde{\mathcal{A}}_{11} \in \mathbb{R}^{jp \times jp}$ and $\tilde{\mathcal{A}}_{22} \in \mathbb{R}^{(j-1)p \times (j-1)p}$ are cyclic block matrices.

In contrast, situation (4.50) yields a deflation into two smaller eigenvalue problems which do not carry the structure of $\tilde{\mathcal{A}}$. For illustration, consider the case $p = 3$, $n = 4$ and $a^{(1)}_{22} = 0$ displayed in Figure 4.5. Fortunately, there is an easy way to force deflations at $a^{(3)}_{21}$ and $a^{(3)}_{32}$ so that afterwards the deflation stemming from $a^{(1)}_{22}$ resides in a deflated $p \times p$ cyclic matrix and can thus be ignored. Applying an implicit shifted QR step with p zero shifts introduces the zero $a^{(3)}_{21}$ element. An RQ step is a QR step implicitly applied to $(F^T \tilde{\mathcal{A}} F)^T$, where F is the flip matrix. Hence, an implicitly shifted RQ step with p zero

$$\left[\begin{array}{cccc:cccc:cccc}
0 & 0 & a_{11}^{(3)} & 0 & 0 & a_{12}^{(3)} & 0 & 0 & a_{13}^{(3)} & 0 & 0 & a_{14}^{(3)} \\
a_{11}^{(1)} & 0 & 0 & a_{12}^{(1)} & 0 & 0 & a_{13}^{(1)} & 0 & 0 & a_{14}^{(1)} & 0 & 0 \\
0 & a_{11}^{(2)} & 0 & 0 & a_{12}^{(2)} & 0 & 0 & a_{13}^{(2)} & 0 & 0 & a_{14}^{(2)} & 0 \\
\hdashline
0 & 0 & a_{21}^{(3)} & 0 & 0 & a_{22}^{(3)} & 0 & 0 & a_{23}^{(3)} & 0 & 0 & a_{24}^{(3)} \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & a_{23}^{(1)} & 0 & 0 & a_{24}^{(1)} & 0 & 0 \\
0 & 0 & 0 & 0 & a_{22}^{(2)} & 0 & 0 & a_{23}^{(2)} & 0 & 0 & a_{24}^{(2)} & 0 \\
\hdashline
0 & 0 & 0 & 0 & 0 & a_{32}^{(3)} & 0 & 0 & a_{33}^{(3)} & 0 & 0 & a_{34}^{(3)} \\
0 & 0 & 0 & 0 & 0 & 0 & a_{33}^{(1)} & 0 & 0 & a_{34}^{(1)} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{33}^{(2)} & 0 & 0 & a_{34}^{(2)} & 0 \\
\hdashline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{43}^{(3)} & 0 & 0 & a_{44}^{(3)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{44}^{(1)} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{44}^{(2)} & 0
\end{array}\right]$$

Fig. 4.5. Structure-destroying deflation of a cyclic block matrix.

shifts preserves the structure of $\tilde{\mathcal{A}}$ and gives the zero $a_{32}^{(3)}$ element. It can be shown that this procedure is numerically equivalent to the deflation procedure presented in [52], which is displayed in Algorithm 20. A notable difference is

Algorithm 20 Deflation of singular triangular factors

Input: Upper triangular matrices $A^{(1)}, \ldots, A^{(p-1)} \in \mathbb{R}^{n \times n}$ and a matrix $A^{(p)} \in \mathbb{R}^{n \times n}$ in upper Hessenberg form, an integer $t \in [2, n]$. Integers l and j with $1 \leq l \leq p-1$ and $a_{jj}^{(l)} = 0$.

Output: Orthogonal matrices $Q^{(1)}, \ldots, Q^{(p)}$ such that $A^{(l)} \leftarrow Q^{(l+1)T} A^{(l)} Q^{(l)}$, $l = 1, \ldots, p$, is again in Hessenberg-triangular form with $a_{j,j-1} = 0$ (if $j > 1$) and $a_{j+1,j} = 0$ (if $j < n$).

$Q^{(1)} \leftarrow I_n,\ Q^{(2)} \leftarrow I_n,\ \ldots,\ Q^{(p)} \leftarrow I_n$
% Annihilate first $j-1$ subdiagonal entries of $A^{(p)}$ and propagate Givens rotations.
for $\tilde{l} \leftarrow p, 1, 2, \ldots, p-1$ **do**
  for $i \leftarrow 1, \ldots, j-1$ **do**
    $G_i \leftarrow G_{i,i+1}(A^{(\tilde{l})} e_i)$
    $A^{(\tilde{l})} \leftarrow G_i A^{(\tilde{l})}, \quad A^{(\tilde{l}+1)} \leftarrow A^{(\tilde{l}+1)} G_i^T, \quad Q^{(\tilde{l}+1)} \leftarrow Q^{(\tilde{l}+1)} G_i^T$
  end for
end for
% Annihilate last $n-j$ subdiagonal entries of $A^{(p)}$ and propagate Givens rotations.
for $\tilde{l} \leftarrow p, p-1, p-2, \ldots, 1$ **do**
  for $i \leftarrow n-1, n-2, \ldots, j$ **do**
    $G_i \leftarrow G_{i+1,i}(e_{i+1}^T A^{(\tilde{l})})$
    $A^{(\tilde{l})} \leftarrow A^{(\tilde{l})} G_i, \quad A^{(\tilde{l}-1)} \leftarrow G_i^T A^{(\tilde{l}-1)}, \quad Q^{(\tilde{l})} \leftarrow Q^{(\tilde{l})} G_i$
  end for
end for

that the deflation criteria suggested in [52] are based on the norms of the factors $A^{(l)}$ instead of the magnitudes of nearby entries.

Summary

We have shown that reduction to Hessenberg form as well as QR iterations preserve cyclic block structures. If the factors $A^{(1)}, \ldots, A^{(p-1)}$ are sufficiently well conditioned then the complete QR algorithm is structure-preserving. Otherwise, a special deflation technique, which is not part of the standard QR algorithm, must be used. Overall, this yields an algorithm for solving block cyclic eigenvalue problems (or, equivalently, product eigenvalue problems), which is ideal in the sense of Section 4.1.3.

4.2.4 Computation of Invariant Subspaces

Let us assume that the periodic QR algorithm has computed a (real) periodic Schur decomposition

$$T^{(l)} = Q^{(l+1)T} A^{(l)} Q^{(l)}, \quad l = 1, \ldots, p,$$

where $T^{(p)}$ has real Schur form and $T^{(1)}, \ldots, T^{(p-1)}$ are upper triangular matrices. If the $(k+1, k)$ subdiagonal entry of $T^{(p)}$ is zero, then the first k columns of $Q^{(1)}$ span an invariant subspace of the matrix product $\Pi_{\mathcal{A}} = A^{(p)} A^{(p-1)} \cdots A^{(p-1)}$ belonging to the eigenvalues of its leading $k \times k$ principal submatrix. To obtain invariant subspaces belonging to other eigenvalues, it is necessary to reorder the periodic Schur decomposition.

As this can be done in a bubble-sort fashion as for standard Schur decompositions, see Section 1.7, it is sufficient to develop a swapping algorithm for the periodic Schur decomposition. That is, the computation of orthogonal matrices $Q^{(1)}, \ldots, Q^{(p)}$ such that

$$Q^{(l+1)T} \begin{bmatrix} A_{11}^{(l)} & A_{12}^{(l)} \\ 0 & A_{22}^{(l)} \end{bmatrix} Q^{(l)} = \begin{bmatrix} B_{11}^{(l)} & B_{12}^{(l)} \\ 0 & B_{22}^{(l)} \end{bmatrix}, \quad l = 1, \ldots, p, \tag{4.51}$$

where $A_{11}^{(l)}, B_{22}^{(l)} \in \mathbb{R}^{n_1 \times n_1}$, $A_{22}^{(l)}, B_{11}^{(l)} \in \mathbb{R}^{n_2 \times n_2}$, $n_1, n_2 \in \{1, 2\}$, and

$$\begin{aligned} \lambda(A_{11}^{(p)} A_{11}^{(p-1)} \cdots A_{11}^{(1)}) &= \lambda(B_{22}^{(p)} B_{22}^{(p-1)} \cdots B_{22}^{(1)}), \\ \lambda(A_{22}^{(p)} A_{22}^{(p-1)} \cdots A_{22}^{(1)}) &= \lambda(B_{11}^{(p)} B_{11}^{(p-1)} \cdots B_{11}^{(1)}). \end{aligned}$$

Not surprisingly, swapping blocks in a periodic Schur decomposition can be related to swapping blocks in a standard Schur decomposition. If we partition

$$Q^{(l)} = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} n_2 & n_1 \end{array} \\ \begin{bmatrix} Q_{11}^{(l)} & Q_{12}^{(l)} \\ Q_{21}^{(l)} & Q_{22}^{(l)} \end{bmatrix} \end{array}$$

and set

$$\mathcal{Q} = \begin{bmatrix} Q_{11}^{(1)} \oplus \cdots \oplus Q_{11}^{(p)} & Q_{12}^{(1)} \oplus \cdots \oplus Q_{12}^{(p)} \\ Q_{21}^{(1)} \oplus \cdots \oplus Q_{21}^{(p)} & Q_{22}^{(1)} \oplus \cdots \oplus Q_{22}^{(p)} \end{bmatrix}, \tag{4.52}$$

then (4.51) is equivalent to

$$\mathcal{Q}^T \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ 0 & \mathcal{A}_{22} \end{bmatrix} \mathcal{Q} = \begin{bmatrix} \mathcal{B}_{11} & \mathcal{B}_{12} \\ 0 & \mathcal{B}_{22} \end{bmatrix}, \tag{4.53}$$

where $\mathcal{A}_{ij}$ and $\mathcal{B}_{ij}$ are the block cyclic matrices associated with $A_{ij}^{(l)}$ and $B_{ij}^{(l)}$, respectively, see also (4.36).

In principal, the orthogonal factor $\mathcal{Q}$ in (4.53) can be constructed as described in Section 1.7.1. Assuming that $\lambda(\mathcal{A}_{11}) \cap \lambda(\mathcal{A}_{22}) = \emptyset$, the solution of the Sylvester equation

$$\mathcal{A}_{11}\mathcal{X} - \mathcal{X}\mathcal{A}_{22} = \gamma\mathcal{A}_{12} \tag{4.54}$$

for some scaling factor $\gamma \leq 1$ is computed. Now, $\mathcal{Q}$ can be determined from a QR decomposition

$$\mathcal{Q}^T \begin{bmatrix} -\mathcal{X} \\ \gamma I_{pn_2} \end{bmatrix} = \begin{bmatrix} \mathcal{R} \\ 0 \end{bmatrix}.$$

Since Lemma 4.17 implies that $\mathcal{X}$ is a block diagonal matrix, the factor $\mathcal{Q}$ can always be chosen so that it has the form (4.52).

The finite-precision properties of this algorithm are as follows. Let $\hat{Q}^{(1)}$, $\ldots$, $\hat{Q}^{(p)}$ denote the computed orthogonal factors, then the exact swapping relation (4.51) is perturbed to

$$\hat{Q}^{(l+1)T} \begin{bmatrix} A_{11}^{(l)} & A_{12}^{(l)} \\ 0 & A_{22}^{(l)} \end{bmatrix} \hat{Q}^{(l)} = \begin{bmatrix} \hat{B}_{11}^{(l)} & \hat{B}_{12}^{(l)} \\ \hat{B}_{21}^{(l)} & \hat{B}_{22}^{(l)} \end{bmatrix}, \quad l = 1, \ldots, p.$$

Let us assume that the Sylvester equation (4.54) is solved by applying Gaussian elimination with complete pivoting to its Kronecker product formulation, see also (4.42). Then from the discussion in Section 1.7.1 it can be expected that the subdiagonal blocks $\hat{B}_{21}^{(l)}$ almost always satisfy

$$\|\hat{B}_{21}^{(l)}\|_F \leq c^{(l)}\mathbf{u}(\|\mathcal{A}_{11}\|_F + \|\mathcal{A}_{12}\|_F + \|\mathcal{A}_{22}\|_F) \tag{4.55}$$

for some modest numbers $c^{(l)} \geq 1$. By a preliminary scaling, we may assume $(\|A_{11}^{(l)}\|_F + \|A_{12}^{(l)}\|_F + \|A_{22}^{(l)}\|_F) = \gamma$ for some constant γ independent of l. In this case, the bound (4.55) implies $\|\hat{B}_{21}^{(l)}\|_F \leq \sqrt{p}c^{(l)}\mathbf{u}\gamma$. Note that for $p = 2$, the described procedure is very similar to swapping algorithms for generalized Schur forms, see Section 2.7.

There is much more to be said about reordering periodic Schur decompositions; the recent work by Granat and Kågström [147] turns the outlined procedure into an efficient and reliable algorithm.

4.2.5 The Periodic Krylov-Schur Algorithm

The periodic QR algorithm is suitable only for matrix products of modest dimension, say $n \leq 3000$. To handle larger product eigenvalue problems with possibly sparse factors, it is necessary to resort to methods based on matrix-vector multiplications, such as the Krylov-Schur algorithm described in Chapter 3. Applying this algorithm blindly to a matrix product

$$\Pi_{\mathcal{A}} = A^{(p)} A^{(p-1)} \cdots A^{(1)},$$

would unnecessarily spoil the accuracy of eigenvalues, especially for those of small magnitude. To avoid this effect, this section explains how the Krylov-Schur algorithm can be adapted to yield a numerically backward stable method for product eigenvalue problems [201].

Periodic Arnoldi Decompositions

From the described relationships between product eigenvalue problems and block cyclic matrices we have seen that any structure-preserving method for computing p-Carrollian eigenvalue tuples and periodic invariant subspaces of a $pn \times pn$ block cyclic matrix $\mathcal{A}$ is a suitable method for solving the corresponding product eigenvalue problem.

In the following, we will develop such a structure-preserving method based on the *Krylov subspace*

$$\mathcal{K}_k(\mathcal{A}, u_1) = \operatorname{span}\{u_1, \mathcal{A}u_1, \mathcal{A}^2 u_1, \ldots, \mathcal{A}^{k-1} u_1\}$$

for some starting vector $u_1 \in \mathbb{R}^{pn}$. As we are interested in approximating periodic invariant subspaces, which are spanned by block diagonal matrices, it is reasonable to require that $\mathcal{K}_k(\mathcal{A}, u_1)$ itself admits a block diagonal basis. Indeed, it will be shown that if a starting vector of the form $u_1 = [u_1^{(1)T}, 0, \ldots, 0]^T$, with $u_1^{(1)} \in \mathbb{R}^n$, is used, then one can construct, by a minor modification of the standard Arnoldi method, Algorithm 15, a decomposition of the following type:

$$\mathcal{A}\,(U_k^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) = (U_{k+1}^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)})\,\hat{\mathcal{H}}_k. \tag{4.56}$$

Here, all matrices $U_{k+1}^{(1)} = [u_1^{(1)}, \ldots, u_{k+1}^{(1)}]$ and $U_k^{(l)} = [u_1^{(l)}, \ldots, u_k^{(l)}]$, $l = 2, \ldots, p$, are assumed to have orthonormal columns. Moreover, the factor $\hat{\mathcal{H}}_k$ in (4.56) takes the form

$$\hat{\mathcal{H}}_k = \begin{bmatrix} 0 & & & \hat{H}_k^{(p)} \\ H_k^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & H_k^{(p-1)} & 0 \end{bmatrix} = \begin{bmatrix} & & & \blacksquare \\ & & & \star \\ \blacksquare & & & \\ & \ddots & & \\ & & \blacksquare & \end{bmatrix}, \tag{4.57}$$

i.e., $H_k^{(1)}, \ldots, H_k^{(p-1)} \in \mathbb{R}^{k\times k}$ are upper triangular matrices while the matrix $\hat{H}_k^{(p)} = \begin{bmatrix} H_k^{(p)} \\ h_{k+1,k} e_k^T \end{bmatrix} \in \mathbb{R}^{(k+1)\times k}$ has (rectangular) upper Hessenberg form.

Any decomposition of the form (4.56)–(4.57) will be called *periodic Arnoldi decomposition* of order k. Similar decompositions have played a role in Krylov subspace methods for solving linear systems with block cyclic matrices, see [55, 56, 57, 120, 130]. For $p = 1$ the periodic Arnoldi decomposition is nothing but a standard Arnoldi decomposition.

Lemma 4.25. *Consider a kth-order periodic Arnoldi decomposition of the form (4.56)–(4.57) and assume that the upper triangular matrices* $H_k^{(1)}, \ldots, H_k^{(p-1)}$ *are invertible and that* $\hat{H}_k^{(p)}$ *is in unreduced Hessenberg form. Then the columns of* $(U_{k+1}^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)})$ *form a basis for* $\mathcal{K}_{pk+1}(\mathcal{A}, [u_1^{(1)T}, 0, \ldots, 0]^T)$.

Proof. Permuting the $(k+1)$th row of $\hat{\mathcal{H}}_k$ to the bottom row and applying a perfect shuffle permutation to the other rows and columns turns $\hat{\mathcal{H}}_k$ into a Hessenberg matrix, whose subdiagonal entries are composed of the diagonal entries of $H_k^{(1)}, \ldots, H_k^{(p-1)}$ and the subdiagonal entries of $\hat{H}_k^{(p)}$. This permutation turns (4.56) into a standard (although particularly structured) Arnoldi decomposition. Moreover, the made assumptions guarantee that the constructed Hessenberg matrix is unreduced. Together with Theorem 3.5, this proves the desired result. Note that this proof also shows that any periodic Arnoldi decomposition can be transformed into an equivalent standard Arnoldi decomposition. □

Algorithm 21 produces a periodic Arnoldi decomposition. It is formally and numerically equivalent to the standard Arnoldi method applied to the block cyclic matrix $\mathcal{A}$ with starting vector $u_1 = [u_1^{(1)T}, 0, \ldots, 0]^T$, with the notable difference that the columns of the produced Krylov basis are sorted in a particular order. The remarks concerning the implementation of Algorithm 15 apply likewise to Algorithm 21. Some additional remarks:

1. Algorithm 21 can be implemented such that in each outer loop exactly p matrix-vector multiplications, involving each of the matrices $A^{(1)}, \ldots, A^{(p)}$, are needed. It can be expected that the computational cost of Algorithm 21 is dominated by these matrix-vector multiplications making it comparable to the cost of the standard Arnoldi method applied to the matrix product $\Pi_{\mathcal{A}} = A^{(p)} A^{(p-1)} \cdots A^{(1)}$. (Note that this statement is only true under the assumption that the matrices $A^{(1)}, \ldots, A^{(p)}$ are subsequently applied to a vector whenever the standard Arnoldi method requests a matrix-vector product. If the matrix product or parts of it can be condensed in a cheap manner, the computational cost of the standard Arnoldi method may be considerably lower.)
2. A major drawback of using Algorithm 21 in comparison to the standard Arnoldi method applied to $\Pi_{\mathcal{A}}$ is the increase of memory require-

Algorithm 21 Periodic Arnoldi method

Input: Matrices $A^{(1)}, \ldots, A^{(p)} \in \mathbb{R}^{n \times n}$, a starting vector $u_1^{(1)} \in \mathbb{R}^n$ with $\|u_1^{(1)}\|_2 = 1$, and an integer $k \leq n$.

Output: Matrices $U_{k+1}^{(1)} \in \mathbb{R}^{n \times (k+1)}$, $U_k^{(2)}, \ldots, U_k^{(p)} \in \mathbb{R}^{n \times k}$ having orthonormal columns, upper triangular matrices $H_k^{(1)}, \ldots, H_k^{(p-1)} \in \mathbb{R}^{k \times k}$ and an upper Hessenberg matrix $\hat{H}_k^{(p)} \in \mathbb{R}^{(k+1) \times k}$, defining a kth order periodic Arnoldi decomposition (4.56)–(4.57).

$H_0^{(1)} \leftarrow [], \ \ldots, \ H_0^{(p-1)} \leftarrow [], \ \hat{H}_0^{(p)} \leftarrow []$
for $j \leftarrow 1, 2, \ldots, k$ **do**
  for $l \leftarrow 1, 2, \ldots, p-1$ **do**
    $h_j^{(l)} \leftarrow U_{j-1}^{(l+1)H} A^{(l)} u_j^{(l)}$
    $v \leftarrow A^{(l)} u_j^{(l)} - U_{j-1}^{(l+1)} h_j^{(l)}$
    $h_{jj}^{(l)} \leftarrow \|v\|_2$
    $u_j^{(l+1)} \leftarrow v / h_{jj}^{(l)}$
    $H_j^{(l)} \leftarrow \begin{bmatrix} H_{j-1}^{(l)} & h_j^{(l)} \\ 0 & h_{jj}^{(l)} \end{bmatrix}$
  end for
  $h_j^{(p)} \leftarrow U_j^{(1)H} A^{(p)} u_j^{(p)}$
  $v \leftarrow A^{(p)} u_j^{(p)} - U_j^{(1)} h_j^{(p)}$
  $h_{j+1,j}^{(p)} \leftarrow \|v\|_2$
  $u_{j+1}^{(1)} \leftarrow v / h_{j+1,j}^{(p)}$
  $\hat{H}_j^{(p)} \leftarrow \begin{bmatrix} \hat{H}_{j-1}^{(p)} & h_j^{(p)} \\ 0 & h_{j+1,j}^{(p)} \end{bmatrix}$
end for

ments by roughly a factor of p due to the need for storing each basis $U_{k+1}^{(1)}, U_k^{(2)}, \ldots, U_k^{(p)}$ instead of only one $n \times (k+1)$ basis.

3. Algorithm 21 breaks down as soon as it encounters $h_{jj}^{(l)} = 0$, for $1 \leq l \leq p-1$, or $h_{j+1,j}^{(p)} = 0$. In both cases, an exact periodic invariant subspace has been found and can be deflated. The first case, which can only happen if one of the matrices $A^{(1)}, \ldots, A^{(p-1)}$ is (nearly) singular, requires some attention in order not to destroy the structure of the periodic Arnoldi decomposition, see [201].

Let us emphasize again that Algorithm 21 is equivalent to the standard Arnoldi method. This implies that many results on the Arnoldi method, for instance on its convergence, carry over to Algorithm 21. Moreover, the usual methods for computing Ritz values, vectors and bases, can be applied to extract approximations to eigenvalues and invariant subspaces from the Krylov subspace basis produced by this algorithm.

Periodic Krylov Decompositions

We have seen in Chapter 3 that the applicability of the standard Arnoldi method can be considerably enhanced by employing techniques for restarting and deflating Arnoldi decompositions. In the following, we adapt these techniques by employing an adapted notion of Krylov decompositions.

Definition 4.26. *Let $\mathcal{A}$ be a block cyclic matrix of the form (4.34). The decomposition*

$$\mathcal{A}\,(U_k^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) = (U_{k+1}^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)})\,\hat{\mathcal{B}}_k, \tag{4.58}$$

is called a periodic Krylov decomposition *of order k if all matrices $U_{k+1}^{(1)} = [u_1^{(1)}, \ldots, u_{k+1}^{(1)}]$ and $U_k^{(l)} = [u_1^{(l)}, \ldots, u_k^{(l)}]$, $l = 2, \ldots, p$, have orthonormal columns and if the factor $\hat{\mathcal{B}}_k$ takes the form*

$$\hat{\mathcal{B}}_k = \begin{bmatrix} 0 & & & \hat{B}_k^{(p)} \\ B_k^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & B_k^{(p-1)} & 0 \end{bmatrix} \tag{4.59}$$

for some matrices $B_k^{(1)}, \ldots, B_k^{(p-1)} \in \mathbb{R}^{k\times k}$ and $\hat{B}_k^{(p)} = \begin{bmatrix} B_k^{(p)} \\ b_k^{(p)T} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k}$.

We have seen that any Krylov decomposition is actually equivalent to an Arnoldi decomposition, see Lemma 3.10. The following lemma is an extension of this fact to periodic decompositions.

Lemma 4.27. *Let*

$$\mathcal{A}\,(U_k^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) = (U_{k+1}^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)})\,\hat{\mathcal{B}}_k$$

be a periodic Krylov decomposition. Then there exists a periodic Arnoldi decomposition

$$\mathcal{A}\,(\tilde{U}_k^{(1)} \oplus \tilde{U}_k^{(2)} \cdots \oplus \tilde{U}_k^{(p)}) = (\tilde{U}_{k+1}^{(1)} \oplus \tilde{U}_k^{(2)} \cdots \oplus \tilde{U}_k^{(p)})\,\hat{\mathcal{H}}_k \tag{4.60}$$

so that $\operatorname{span}(\tilde{U}_{k+1}^{(1)}) = \operatorname{span}(U_{k+1}^{(1)})$ and $\operatorname{span}(\tilde{U}_k^{(l)}) = \operatorname{span}(U_k^{(l)})$ for $l = 1, \ldots, p$.

Proof. First, let us partition

$$\hat{B}_{k+1}^{(p)} = \begin{bmatrix} B_k^{(p)} \\ b_{k+1}^{(p)T} \end{bmatrix}, \quad B_k^{(p)} \in \mathbb{C}^{k\times k}, \quad b_{k+1}^{(p)} \in \mathbb{C}^k,$$

and construct an orthogonal matrix Q_0, e.g. a Householder matrix, such that $Q_0^T b_{k+1}^{(p)} = h_{k+1,k} e_k$ for some $h_{k+1,k} \in \mathbb{R}$. Next, we apply Algorithm 22, a row-oriented version of Algorithm 18, to the matrices

Algorithm 22 Row-oriented reduction to periodic Hessenberg form

Input: Matrices $B^{(1)}, \ldots, B^{(p)} \in \mathbb{R}^{k \times k}$.

Output: Orthogonal matrices $Q^{(1)}, \ldots, Q^{(p)}$ such that $H^{(l)} = Q^{(l+1)T} B^{(l)} Q^{(l)}$ is upper triangular for $l = 1, \ldots, p-1$ and $H^{(p)} = Q^{(1)T} B^{(p)} Q^{(p)}$ is in upper Hessenberg form. Each matrix $B^{(l)}$ is overwritten by $H^{(l)}$.

Remark: $\check{H}_j(x)$ denotes a Householder matrix which maps the leading $j-1$ entries of a vector $x \in \mathbb{R}^k$ to zero without affecting its trailing $k-j$ entries.

```
Q^(1) ← I_k, ..., Q^(p) ← I_k
for j ← k, k−1, ..., 2 do
  for l ← p−1, p−2, ..., 1 do
    x ← B^(l)T e_j
    B^(l) ← B^(l) · Ȟ_j(x)
    B^(l−1) ← Ȟ_j(x) · B^(l−1)
    Q^(l) ← Q^(l) · Ȟ_j(x)
  end for
  x ← B^(p)T e_j
  B^(p) ← B^(l) · Ȟ_{j−1}(x)
  B^(p−1) ← Ȟ_{j−1}(x) · B^(p−1)
  Q^(p) ← Q^(p) · Ȟ_{j−1}(x)
end for
```

$$B_k^{(p)} Q_0, \; Q_0^T B_k^{(p-1)}, \; B_k^{(p-2)}, \; \ldots, \; B_k^{(1)}.$$

It is simple to check that the orthogonal factor $Q^{(p)}$ returned by this algorithm satisfies $e_k^T Q^{(p)} = e_k^T$. This implies that setting

$$\tilde{U}_k^{(1)} = U_k^{(1)} Q^{(1)}, \; \ldots, \; \tilde{U}_k^{(p-1)} = U_k^{(p-1)} Q^{(p-1)}, \; \tilde{U}_k^{(p)} = U_k^{(p)} Q_0 Q^{(p)},$$

and $\tilde{U}_{k+1}^{(1)} = [\tilde{U}_k^{(1)}, u_{k+1}^{(1)}]$ yields a periodic Arnoldi decomposition (4.60), where the factor $\hat{\mathcal{H}}_k$ takes the form (4.57) with the coefficient matrices

$$\begin{aligned}
\hat{H}_k^{(1)} &= (Q^{(2)} \oplus 1)^T \hat{B}_k^{(1)} Q_0 Q^{(1)}, \\
H_k^{(2)} &= Q^{(3)T} \hat{B}_k^{(2)} Q^{(2)}, \; \ldots, \; H_k^{(p-1)} = Q^{(p)T} \hat{B}_k^{(p-1)} Q^{(p-1)}. \\
H_k^{(p)} &= (Q_0 Q^{(1)})^T \hat{B}_k^{(p)} Q^{(p)},
\end{aligned}$$

having Hessenberg-triangular form, which completes the proof. □

In analogy to the notation used in Chapter 3, a decomposition of the form (4.58)–(4.59) is called a *periodic Krylov-Schur decomposition* if all coefficient matrices $B_k^{(1)}, \ldots, B_k^{(p)}$ are upper triangular.

Implicit Restarting

Given a periodic Krylov decomposition of order m,

$$\mathcal{A}\,(U_m^{(1)} \oplus U_m^{(2)} \oplus \cdots \oplus U_m^{(p)}) = (U_{m+1}^{(1)} \oplus U_m^{(2)} \oplus \cdots \oplus U_m^{(p)})\,\hat{\mathcal{B}}_m, \tag{4.61}$$

where $\hat{\mathcal{B}}_m$ is a block cyclic matrix of the form (4.59), implicit restarting proceeds as follows. First, orthogonal matrices $Q_1^{(1)}, \ldots, Q_1^{(p)} \in \mathbb{R}^{m\times m}$ are constructed such that $T_m^{(l)} = Q_1^{(l+1)T} H_m^{(l)} Q_1^{(l)}$ is a periodic Schur decomposition. The eigenvalues of the product

$$\Pi_{\mathcal{T}} = T_m^{(p)} T_m^{(p-1)} \cdots T_m^{(1)}$$

correspond to the m Carrollian tuples of Ritz values of the Krylov decomposition (4.61). Some of these eigenvalues may be of interest and approximate desired eigenvalues of $\Pi_{\mathcal{A}}$, but some may not. Therefore, the next step of implicit restarting consists of reordering the $k < m$ wanted eigenvalues to the top left corner of the block triangular matrix $\Pi_{\mathcal{T}}$, using reliable reordering methods as described in Section 4.2.4 and [147]. This yields another set of orthogonal matrices $Q_2^{(1)}, \ldots, Q_2^{(p)} \in \mathbb{R}^{m\times m}$ so that

$$\begin{aligned}&\mathcal{A}\,(U_m^{(1)}Q_1^{(1)}Q_2^{(1)} \oplus U_m^{(2)}Q_1^{(2)}Q_2^{(2)} \oplus \cdots \oplus U_m^{(p)}Q_1^{(p)}Q_2^{(p)})\\ &= ([U_m^{(1)}Q_1^{(1)}Q_2^{(1)}, u_{m+1}^{(1)}] \oplus U_m^{(2)}Q_1^{(2)}Q_2^{(2)} \oplus \cdots \oplus U_m^{(p)}Q_1^{(p)}Q_2^{(p)})\,\hat{\mathcal{T}}_m,\end{aligned}$$

where $\hat{\mathcal{T}}_m$ is a block cyclic matrix of the form (4.59). The coefficients of $\hat{\mathcal{T}}_m$ are given by $T_m^{(l)} = \left[\begin{smallmatrix} T_w^{(l)} & \star \\ 0 & T_u^{(l)} \end{smallmatrix}\right]$, $l = 1, \ldots, p-1$, and

$$\hat{T}_m^{(p)} = \left[\begin{array}{cc} T_w^{(p)} & \star \\ 0 & T_u^{(p)} \\ \hline b_w^{(p)T} & \star \end{array}\right].$$

Here, the $k \times k$ product $T_w^{(p)} T_w^{(p-1)} \cdots T_w^{(1)}$ contains the wanted eigenvalues of Π_T while the $(m-k)\times(m-k)$ product $T_u^{(p)} T_u^{(p-1)} \cdots T_u^{(1)}$ contains the unwanted eigenvalues. Note that these products correspond to the block cyclic matrices

$$\mathcal{T}_w = \begin{bmatrix} 0 & & & T_w^{(p)} \\ T_w^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & T_w^{(p-1)} & 0 \end{bmatrix}, \quad \mathcal{T}_u = \begin{bmatrix} 0 & & & T_u^{(p)} \\ T_u^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & T_u^{(p-1)} & 0 \end{bmatrix}. \tag{4.62}$$

Finally, the constructed periodic Krylov-Schur decomposition is truncated. By letting $\tilde{U}_k^{(l)}$ contain the first k columns of $U_m^{(l)}Q_1^{(l)}Q_2^{(l)}$, $l = 1, \ldots, p$, and setting $\tilde{u}_{k+1}^{(1)} = u_{m+1}^{(1)}$, we obtain the following Krylov-Schur decomposition of order k:

$$\mathcal{A}\,(\tilde{U}_k^{(1)} \oplus \tilde{U}_k^{(2)} \oplus \cdots \oplus \tilde{U}_k^{(p)}) = ([\tilde{U}_k^{(1)}, \tilde{u}_{k+1}^{(1)}] \oplus \tilde{U}_k^{(2)} \oplus \cdots \oplus \tilde{U}_k^{(p)})\,\tilde{\mathcal{T}}_w,$$

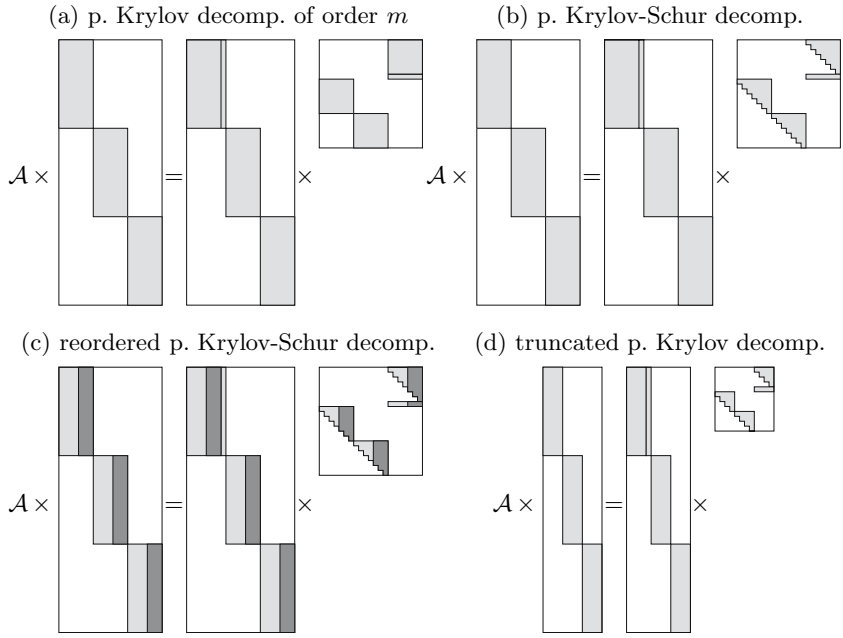


Fig. 4.6. Restarting a periodic Krylov decomposition.

where $\hat{\mathcal{T}}_w$ is almost identical to $\mathcal{T}_w$ in (4.62), with the only difference that the $k\times k$ coefficient matrix $T_w^{(p)}$ is replaced by the $(k+1)\times k$ matrix $\hat{T}_w^{(p)} = \begin{bmatrix} T_w^{(p)} \\ b_w^{(p)T} \end{bmatrix}$.

The described procedure of reduction to periodic Krylov-Schur decomposition, reordering and truncation is depicted in Figure 4.6. Note that the Krylov subspace corresponding to the truncated decomposition is the same subspace that would have been obtained if implicit restarting with the filter polynomial

$$\psi(z) = \prod_{\lambda\in\lambda(\mathcal{T}_u)} (z-\lambda),$$

see Section 3.2.1, had been applied to the standard Arnoldi decomposition corresponding to (4.61).

Deflation

After a periodic Krylov decomposition has been truncated to order k, it can again be expanded to a decomposition of order m by applying a variant of the periodic Arnoldi method, Algorithm 21. This process of truncation and expansion is repeated until convergence occurs. To decide on convergence, we suggest the criterion (3.12), which has been used in Section 3.2.4 for deflating invariant subspaces from standard Krylov decompositions. In order to

preserve the structure of periodic Krylov decompositions, we only deflate periodic invariant subspaces that belong to a p-Carrollian tuple of Ritz values. The deflation procedure is analogous to the procedure described in Section 3.2.4, see also [201].

Numerical Examples

To illustrate the numerical differences between the standard and periodic Krylov-Schur algorithms, let us consider the following simple example:

$$A^{(1)} = A^{(2)} = A^{(3)} = \mathrm{diag}(1, 10^{-1}, 10^{-2}, 10^{-3}, \ldots, 10^{-50}). \tag{4.63}$$

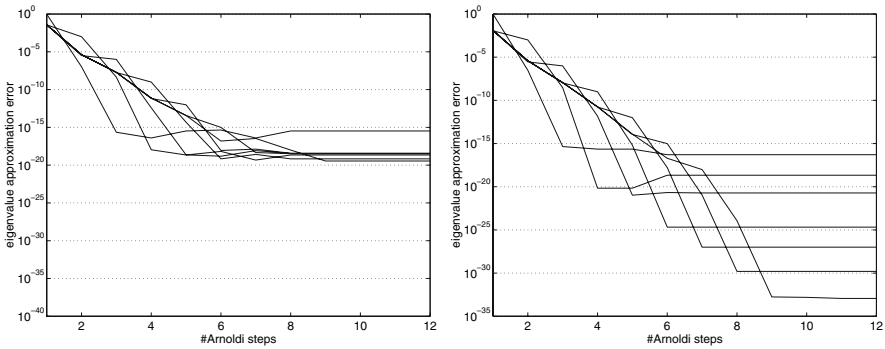


Fig. 4.7. Approximation errors of the 7 largest eigenvalues for the standard (left plot) and the periodic (right plot) Krylov-Schur algorithm applied to the product $A^{(3)}A^{(2)}A^{(1)}$ defined in (4.63).

Figure 4.7 displays the approximation error

$$\min\{|\lambda_i - \sigma_j| : \sigma_j \text{ is a Ritz value}\} \tag{4.64}$$

for each of the 7 largest eigenvalues $1, 10^{-3}, \ldots, 10^{-18}$ versus the number of (periodic) Arnoldi steps. It can be seen that these errors stagnate at a value above 10^{-20} for the Krylov-Schur algorithm, while the periodic Krylov-Schur algorithm is capable to compute much more accurate approximations to the smaller eigenvalues (which correspond to the lower error curves). All computations in this and the following numerical example have been performed using MATLAB implementations of the (periodic) Krylov-Schur algorithm.

Model reduction

The periodic Krylov-Schur algorithm could also be a useful tool in model reduction, see Section A.2. Some popular model reduction techniques, such as

balanced truncation, require the computation of the Hankel singular values, which are the singular values of a product RS^T for square matrices R and S. The underlying LTI system is reduced by maintaining only these parts that belong to non-negligible Hankel singular values. For this purpose, it is important to identify these values correctly.

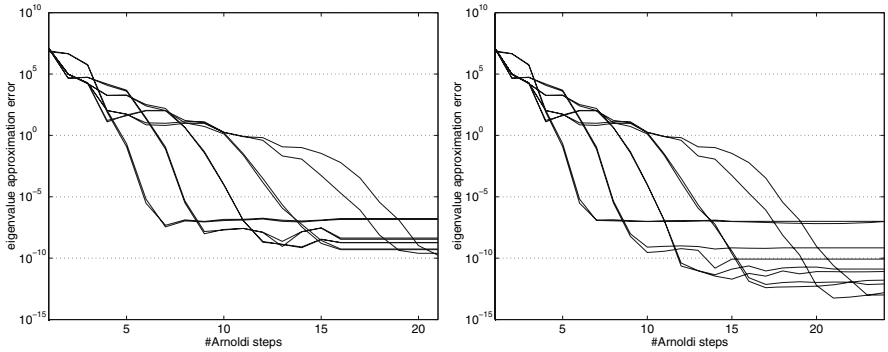


Fig. 4.8. Approximation errors of the 10 largest eigenvalues for the standard (left plot) and the periodic (right plot) Krylov-Schur algorithm applied to the matrix product Π corresponding to a discretized clamped beam model.

Computing the Hankel singular values is equivalent to computing the positive square roots of the eigenvalues of the matrix product $\Pi = SR^T RS^T$. In the following, we consider a practical example to compare the standard Krylov-Schur algorithm with the periodic Krylov-Schur algorithm applied to Π. For this purpose, we have used the discretized model of a clamped beam as described in [13, 88]. The corresponding matrices R and S as well as precomputed Hankel singular values can be obtained from the model reduction benchmark collection [88]. Figure 4.8 displays the eigenvalue approximation errors, see (4.64), for the 10 largest eigenvalue of Π:

$$\{5.7{\cdot}10^6, 4.7{\cdot}10^6, 7.4{\cdot}10^4, 7.1{\cdot}10^4, 2.0{\cdot}10^3, 1.9{\cdot}10^3, 1.1{\cdot}10^2, 13, 9.7\}$$

(only the leading two significant digits are displayed). A restart was applied after 17 Arnoldi steps, truncating the (periodic) Arnoldi decomposition to a (periodic) Krylov-Schur decomposition which maintains the 12 largest Ritz values. The errors in the eigenvalue approximations produced by the Krylov-Schur algorithm stagnate at a level above 10^{-10}. Again, the periodic Krylov-Schur algorithm produces more accurate approximations; the lower error curves in the right plot of Figure 4.8 correspond to the smaller eigenvalues.

4.2.6 Further Notes and References

The product eigenvalue problem can be generalized in many different directions. One generalization is to consider the computation of periodic deflating subspaces and generalized eigenvalues of the block cyclic / diagonal matrix pair $(\mathcal{A}, \mathcal{B})$, where $\mathcal{B}$ is given by $\mathcal{B} = B^{(1)} \oplus \cdots \oplus B^{(p)}$ for some $B^{(l)} \in \mathbb{R}^{n \times n}$, see [52, 159]. If $\mathcal{B}$ is invertible, this corresponds to the computation of eigenvalues and invariant subspaces of the general matrix product

$$A^{(p)}[B^{(p)}]^{-1} A^{(p-1)}[B^{(p-1)}]^{-1} \cdots A^{(1)}[B^{(1)}]^{-1}. \tag{4.65}$$

Perturbation analyses for this type of generalized product eigenvalue problems can be found in [43, 221]. The approach developed in Section 4.2.2 can be generalized to this situation without any difficulties using structured perturbation results for deflating subspaces [83].

If P denotes the perfect shuffle matrix, then reduction to Hessenberg-triangular form as described in Algorithm 9 does not preserve the cyclic / diagonal block structure of $(\tilde{\mathcal{A}}, \tilde{\mathcal{B}}) := P^T(\mathcal{A}, \mathcal{B})P$ as this algorithm employs Givens rotations acting on consecutive rows and columns. A remedy is to use (opposite) Householder matrices of order p or Givens rotations acting on pairs of rows or columns having distance p. Then, reducing $(\tilde{\mathcal{A}}, \tilde{\mathcal{B}})$ to Hessenberg-triangular form preserves its cyclic/diagonal block structure and is numerically equivalent to the algorithms described in [52, 159, 331] for reducing (4.65). Along the lines of the proof of Theorem 4.23, it can be shown that QZ iterations based on Householder matrices, see Algorithm 11, preserve the structure of $(\tilde{\mathcal{A}}, \tilde{\mathcal{B}})$. Taking Hessenberg-triangular reduction and QZ iterations together yields the *periodic QZ algorithm* described in [52, 159].

Even more general products of the form

$$[A^{(p)}]^{s^{(p)}} [A^{(p-1)}]^{s^{(p-1)}} \cdots [A^{(1)}]^{s^{(1)}}, \quad s^{(1)}, \ldots, s^{(p)} \in \{1, -1\}.$$

can be found in [32, 43, 139]. Such products can be related to block cyclic/ diagonal matrix pairs by introducing identities at appropriate places [43].

Another generalization of the product eigenvalue problem is to allow varying dimensions, i.e., $A^{(l)} \in \mathbb{R}^{n^{(l)} \times n^{(l+1)}}$ for some integers $n^{(1)}, \ldots, n^{(p)}$. Varga [336] has developed a finite algorithm reducing this problem to an equivalent product eigenvalue problem involving p square factors of order $n = \min\{n^{(1)}, \ldots, n^{(p)}\}$.

The classification of (generalized) product eigenvalue problems with respect to similarity and equivalence transformations is well-understood in the field of representation theory, see, e.g., [112, 253]. Very readable introductions to this area can be found in papers by Sergeichuk [283, 284]. GUPTRI-like algorithms for (generalized) product eigenvalue problems are described in [284, 337]. For some preliminary work on balancing, see [197].

4.3 Skew-Hamiltonian and Hamiltonian Matrices

This and the following two sections are devoted to two classes of structured matrices, skew-Hamiltonian and Hamiltonian matrices. A *skew-Hamiltonian matrix* has the form

$$W = \begin{bmatrix} A & G \\ Q & A^T \end{bmatrix}, \quad G = -G^T, \quad Q = -Q^T, \tag{4.66}$$

while a *Hamiltonian matrix* reads as

$$H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}, \quad G = G^T, \quad Q = Q^T, \tag{4.67}$$

where A, G and Q are real $n \times n$ matrices. A number of applications from control theory and related areas lead to eigenvalue problems involving such matrices; with a stronger emphasis on Hamiltonian matrices:

- stability radius and H_∞ norm computation [79, 59, 135], see also Section A.4.1;
- linear quadratic optimal control problems and the solution of continuous-time algebraic Riccati equations [29, 240, 285], see also Section A.3;
- the solution of anti-symmetric Riccati equations [298];
- H_∞ control [33];
- passivity preserving model reduction [12, 296];
- quadratic eigenvalue problems [242, 324];
- computation of pseudospectra [75] and the distance to uncontrollability [149, 74], see also Section A.4.2.

An ubiquitous matrix when dealing with skew-Hamiltonian and Hamiltonian eigenvalue problems is the skew-symmetric matrix

$$J_{2n} = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}. \tag{4.68}$$

In the following we will drop the subscript whenever the dimension of J_{2n} is clear from its context. By straightforward algebraic manipulation one can show that a Hamiltonian matrix H is equivalently defined by the property $HJ = (HJ)^T$. Likewise, a matrix W is skew-Hamiltonian if and only if $WJ = -(WJ)^T$. Any matrix $S \in \mathbb{R}^{2n\times 2n}$ satisfying $S^T J S = S J S^T = J$ is called *symplectic*, and since

$$(S^{-1}HS)J = S^{-1}HJS^{-T} = S^{-1}J^T H^T S^{-T} = [(S^{-1}HS)J]^T,$$

we see that symplectic similarity transformations preserve Hamiltonian structures. There are relevant cases, however, where both H and $S^{-1}HS$ are Hamiltonian but S is not a symplectic matrix [129]. In a similar fashion the same can be shown for skew-Hamiltonian matrices.

The remainder of this section is concerned with subsidiary material related to skew-Hamiltonian and Hamiltonian eigenvalue problems. In a first reading, it might be wise to go directly to page 181 (for skew-Hamiltonian matrices) or to page 191 (for Hamiltonian matrices).

4.3.1 Elementary Orthogonal Symplectic Matrices

From a numerical point of view it is desirable that a symplectic matrix $U \in \mathbb{R}^{2n\times 2n}$ to be used in a transformation algorithm is also orthogonal. Such a matrix is called *orthogonal symplectic*. Two types of elementary orthogonal matrices belong to this class. These are $2n \times 2n$ Givens rotation matrices of the type

$$G_{j,n+j}(\theta) = \begin{bmatrix} I_{j-1} & & & & \\ & \cos\theta & & \sin\theta & \\ & & I_{n-1} & & \\ & -\sin\theta & & \cos\theta & \\ & & & & I_{n-j} \end{bmatrix}, \quad 1 \le j \le n,$$

for some angle $\theta \in [-\pi/2, \pi/2)$ and the direct sum of two identical $n \times n$ Householder matrices

$$(H_j \oplus H_j)(v,\beta) = \begin{bmatrix} I_n - \beta vv^T & \\ & I_n - \beta vv^T \end{bmatrix},$$

see also page 25.

A simple combination of these transformations can be used to map an arbitrary vector $x \in \mathbb{R}^{2n}$ into the linear space

$$\mathcal{E}_j = \operatorname{span}\{e_1, \ldots, e_j, e_{n+1}, \ldots, e_{n+j-1}\},$$

where e_i is the ith unit vector of length $2n$. Such mappings form the backbone of virtually all structure-preserving algorithms based on orthogonal symplectic transformations. They can be constructed using Algorithm 23, where it should be noted that the elements $1, \ldots, j-1$ and $n+1, \ldots, n+j-1$ of the vector x remain unaffected.

Algorithm 23 Construction of elementary orthogonal symplectic matrices

Input: A vector $x \in \mathbb{R}^{2n}$ and an index $j \le n$.
Output: Vectors $v, w \in \mathbb{R}^n$ and $\beta, \gamma, \theta \in \mathbb{R}$ so that

$$[(H_j \oplus H_j)(v,\beta) \cdot G_{j,n+j}(\theta) \cdot (H_j \oplus H_j)(w,\gamma)]^T x \in \mathcal{E}_j.$$

1. Determine $v \in \mathbb{R}^n$ and $\beta \in \mathbb{R}$ such that the last $n-j$ elements of $x \leftarrow (H_j \oplus H_j)(v,\beta)x$ are zero.
2. Determine $\theta \in [-\pi/2, \pi/2)$ such that the $(n+j)$th element of $x \leftarrow G_{j,n+j}(\theta)x$ is zero.
3. Determine $w \in \mathbb{R}^n$ and $\gamma \in \mathbb{R}$ such that the $(j+1)$th to the nth elements of $x \leftarrow (H_j \oplus H_j)(w,\gamma)x$ are zero.

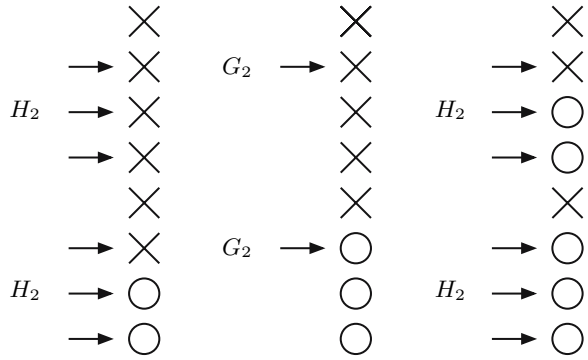


Fig. 4.9. The three steps of Algorithm 23 for $n = 4$ and $j = 2$.

The three steps of Algorithm 23 are illustrated in Figure 4.9. Orthogonal symplectic matrices of the form

$$\begin{aligned} E_j(x) &\equiv E_j(v, w, \beta, \gamma, \theta) \\ &:= (H_j \oplus H_j)(v, \beta) \cdot G_{j,n+j}(\theta) \cdot (H_j \oplus H_j)(w, \gamma), \end{aligned} \tag{4.69}$$

as computed by Algorithm 23 and with $1 \le j \le n$, will be called *elementary*.

Remark 4.28. Let $F = \begin{bmatrix} 0 & I_n \\ I_n & 0 \end{bmatrix}$, then

$$[F \cdot E_j(Fx) \cdot F]^T x \in \operatorname{span}\{e_1, \ldots, e_{j-1}, e_{n+1}, \ldots, e_{n+j}\}.$$

For the sake of brevity we set $E_{n+j}(x) := F \cdot E_j(Fx) \cdot F$, whenever $1 \le j \le n$. ◊

The following lemma shows that every orthogonal symplectic matrix can be factorized into elementary matrices.

Lemma 4.29 ([362, 67]). *Every orthogonal symplectic matrix $U \in \mathbb{R}^{2n \times 2n}$ has the block structure*

$$U = \begin{bmatrix} U_1 & U_2 \\ -U_2 & U_1 \end{bmatrix}, \quad U_1, U_2 \in \mathbb{R}^{n \times n},$$

and can be written as the product of at most n elementary orthogonal symplectic matrices.

4.3.2 The Symplectic QR Decomposition

As a first application of elementary orthogonal symplectic matrices we show how to decompose a general matrix into the product of an orthogonal symplectic and a block triangular matrix.

Lemma 4.30 ([67, 362]). *Let $A \in \mathbb{R}^{2m\times n}$ with $m \geq n$, then there exists an orthogonal symplectic matrix $Q \in \mathbb{R}^{2m\times 2m}$ so that $A = QR$ and*

$$R = \begin{bmatrix} R_{11} \\ R_{21} \end{bmatrix}, \quad R_{11} = \begin{bmatrix} \ddots \\ 0 \end{bmatrix}, \quad R_{21} = \begin{bmatrix} 0 & \ddots \\ \ddots & 0 \\ 0 & \end{bmatrix}, \tag{4.70}$$

that is, the matrix $R_{11} \in \mathbb{R}^{m\times n}$ is upper triangular and $R_{21} \in \mathbb{R}^{m\times n}$ is strictly upper triangular. If $m = n$ and the matrix A contains the first n columns of a $2n \times 2n$ symplectic matrix, then R_{21} is zero.

A decomposition of the form (4.70) is called a *symplectic QR decomposition*, it is useful for computing and refining invariant subspaces of Hamiltonian matrices, see Section 4.5.4 below. Other applications include the symplectic integration of Hamiltonian systems [217, 293]. Algorithm 24 proves the first part of Lemma 4.30 by construction. This algorithm, implemented in the

Algorithm 24 Symplectic QR decomposition

Input: A general matrix $A \in \mathbb{R}^{2m\times n}$ with $m \geq n$.
Output: An orthogonal symplectic matrix $Q \in \mathbb{R}^{2m\times 2m}$; A is overwritten with $R = Q^T A$ having the form (4.70).

$Q \leftarrow I_{2m}$.
for $j \leftarrow 1, \ldots, n$ **do**
  $x \leftarrow Ae_j$
  Apply Algorithm 23 to compute $E_j(x)$.
  $A \leftarrow E_j(x)^T A, \quad Q \leftarrow Q\, E_j(x)$
end for

HAPACK [37] routine `DGESQR`, requires $8(mn^2 - n^3/3) + \mathcal{O}(n^2)$ flops for computing the matrix R, and additionally $\frac{16}{3}n^3 + 16m^2n - 16mn^2 + \mathcal{O}(n^2)$ flops for accumulating the orthogonal symplectic factor Q in reversed order.

The finite-precision properties of Algorithm 24 are as follows. Similarly as for the standard QR decomposition [141] one can show that there exists an orthogonal symplectic matrix V which transforms the computed block upper triangular matrix $\hat{R}$ to a matrix near A, i.e., $V\hat{R} = A + E$, where $\|E\|_2 = \mathcal{O}(\mathbf{u})\|A\|_2$. Moreover, the computed factor $\hat{Q}$ is almost orthogonal in the sense that $\|\hat{Q}^T\hat{Q} - I\|_2 = \mathcal{O}(\mathbf{u})$, and it has the block representation $\hat{Q} = \begin{bmatrix} \hat{Q}_1 & \hat{Q}_2 \\ -\hat{Q}_2 & \hat{Q}_1 \end{bmatrix}$. This implies, together with the following lemma, that $\hat{Q}$ is close to an orthogonal symplectic matrix.

Lemma 4.31. *Let $\hat{Q} = \begin{bmatrix} Q_1 & Q_2 \\ -Q_2 & Q_1 \end{bmatrix}$ be invertible with $Q_1, Q_2 \in \mathbb{R}^{n\times n}$. Then there exist an orthogonal symplectic matrix Q and a symmetric matrix H such that $\hat{Q} = QH$ and*

$$\frac{\|\hat{Q}^T\hat{Q} - I\|_2}{\|\hat{Q}\|_2 + 1} \leq \|\hat{Q} - Q\|_2 \leq \|\hat{Q}^T\hat{Q} - I\|_2. \tag{4.71}$$

Proof. This lemma is shown by proving that $\hat{Q}$ has an orthogonal symplectic singular value decomposition (SVD), i.e., there exist orthogonal symplectic matrices U and V so that $\hat{Q} = U(D \oplus D)V^T$, where D is a diagonal matrix. By the symplectic URV decomposition, see [42] or Section 4.5.5, there are orthogonal symplectic matrices $\tilde{U}$ and $\tilde{V}$ so that

$$\hat{Q} = \tilde{U} \begin{bmatrix} R_{11} & R_{12} \\ 0 & -R_{22}^T \end{bmatrix} \tilde{V}^T =: R,$$

where $R_{11} \in \mathbb{R}^{n \times n}$ is upper triangular and $R_{22} \in \mathbb{R}^{n \times n}$ has upper Hessenberg form. From $J\hat{Q}J^T = \hat{Q}$, it follows that

$$JRJ^T = JU^T\hat{Q}VJ^T = U^TJ\hat{Q}J^TV = U^T\hat{Q}V = R.$$

Thus, R has the same block structure as $\hat{Q}$, which implies $R_{11} = -R_{22}^T$ and $R_{12} = 0$. Now let $R_{11} = U_1DV_1^T$ be an SVD, then the existence of an orthogonal symplectic SVD is shown by setting $U = \tilde{U}(U_1 \oplus U_1)$ and $V = \tilde{V}(V_1 \oplus V_1)$.

The first part of the lemma follows from setting $Q = UV^T$ and $H = VDV^T$. Inequality (4.71) is a well-known result, see, e.g., [164, p.389]. □

4.3.3 An Orthogonal Symplectic WY-like Representation

We have seen that the LAPACK routine for reducing a matrix to Hessenberg form attains high efficiency by (implicitly) employing compact WY representations of the involved orthogonal transformations, see Section 1.5. The following theorem describes a variant of this representation, suitable for elementary orthogonal symplectic matrices as defined in (4.69).

Theorem 4.32 ([198]). *Let $k \leq n$ and $Q = E_{j_1}(x_1) \cdot E_{j_2}(x_2) \cdots E_{j_k}(x_k)$, where the elementary matrices $E_{j_i}(x_i)$ are defined as in (4.69) with $j_i \in [1, n]$ and $x_i \in \mathbb{R}^{2n}$. Then there exist matrices $R \in \mathbb{R}^{3k \times k}, S \in \mathbb{R}^{k \times 3k}, T \in \mathbb{R}^{3k \times 3k}$ and $W \in \mathbb{R}^{n \times 3k}$ so that*

$$Q = \begin{bmatrix} I_n + WTW^T & WRSW^T \\ -WRSW^T & I_n + WTW^T \end{bmatrix}. \tag{4.72}$$

Furthermore, these matrices can be partitioned as

$$R = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix}, \; S = \begin{bmatrix} S_1 \; S_2 \; S_3 \end{bmatrix}, \; T = \begin{bmatrix} T_{11} \; T_{12} \; T_{13} \\ T_{21} \; T_{22} \; T_{23} \\ T_{31} \; T_{32} \; T_{33} \end{bmatrix},$$

where all matrices $R_i, S_l, T_{il} \in \mathbb{R}^{k \times k}$ are upper triangular, and

$$W = \begin{bmatrix} W_1 \; W_2 \; W_3 \end{bmatrix},$$

where $W_1, W_2, W_3 \in \mathbb{R}^{n \times k}$ and W_2 contains in its ith column e_{j_i}, the j_ith column of the $n \times n$ identity matrix.

The HAPACK routine `DLAEST` can be used to construct WY-like representations of the form (4.72) and requires $(4k-2)kn + \frac{19}{3}k^3 + \mathcal{O}(k^2)$ flops. Taking care of the generic structures present in R, S, T and W, the routine `DLAESB` applies the WY-like representation (4.72) to a $2n \times q$ matrix, which requires $16k(n-k)q + \mathcal{O}(kq)$ flops. This routine attains high efficiency by making exclusive use of calls to level 3 BLAS.

4.3.4 Block Symplectic QR Decomposition

Using the results of the previous section we can now easily derive a block oriented version of Algorithm 24 for computing the symplectic QR decomposition of a general $2m \times n$ matrix A.

Let us partition A into block columns

$$A = \left[A_1 \; A_2 \; \ldots \; A_N \right],$$

For convenience only, we will assume that each A_i has n_b columns so that $n = Nn_b$. The following block algorithm for the symplectic QR decomposition goes hand in hand with block algorithms for the standard QR decomposition [47]. The idea is as follows. At the beginning of step p $(1 \le p \le N)$ the matrix A has been overwritten with

$$Q_{j-1} \cdots Q_1 A = \begin{array}{c} \\ (p-1)n_b \\ r \\ (p-1)n_b \\ r \end{array} \begin{array}{c} \begin{array}{ccc} (p-1)n_b & n_b & q \end{array} \\ \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \\ 0 & R_{42} & R_{43} \end{bmatrix} \end{array},$$

where $q = n - pn_b$ and $r = m - (p-1)n_b$. The symplectic QR decomposition of $\begin{bmatrix} R_{22} \\ R_{42} \end{bmatrix}$ is then computed and the resulting orthogonal symplectic factor applied to $\begin{bmatrix} R_{23} \\ R_{43} \end{bmatrix}$. Algorithm 25 is a formal description of this procedure.

In this algorithm, implemented in the HAPACK routine `DGESQB`,

$$6(2mn^2 - n^3)/N + 29n^3/(3N^2) + \mathcal{O}(n^2)$$

flops are required to generate the WY-like representations while

$$8(mn^2 - n^3/3) - (8mn^2 - 19n^3)/N - 49n^3/(3N^2) + \mathcal{O}(n^2)$$

flops are necessary to apply them to the matrix A. On the other hand, Algorithm 24 requires $8(mn^2 - n^3/3) + \mathcal{O}(n^2)$ flops to $Q^T A$. Hence, Algorithm 25 is more expensive by roughly a factor of $(1 + 2.5/N)$, at least when flops are concerned. Basically the same observation holds for the computation of the orthogonal symplectic factor Q. In an efficient implementation, of course, Q would be accumulated in reversed order. See the HAPACK routines `DOSGSB` and `DOSGSQ` for more details.

Algorithm 25 Symplectic QR decomposition (blocked version)

Input: A matrix $A \in \mathbb{R}^{2m \times n}$ with $m \geq n$ and $n = N \cdot n_b$.
Output: An orthogonal symplectic matrix $Q \in \mathbb{R}^{2m \times 2m}$; A is overwritten with $Q^T A$ having the form (4.70). In contrast to Algorithm 24 a block oriented method is used.

$Q \leftarrow I_{2m}$
for $p \leftarrow 1, \ldots, N$ **do**
  Set $s = (p-1)n_b + 1$.
  Apply Algorithm 24 and the construction given in the proof of Theorem 4.32 to compute the WY-like representation (4.72) of an orthogonal symplectic matrix Q_p so that

$$Q_p^T \begin{bmatrix} A(s:m, s:s+n_b-1) \\ A(m+s:2m, s:s+n_b-1) \end{bmatrix}$$

  has the form (4.70).
  Update $\begin{bmatrix} A(s:m,s+n_b:n) \\ A(m+s:2m,s+n_b:n) \end{bmatrix} \leftarrow Q_p^T \begin{bmatrix} A(s:m,s+n_b:n) \\ A(m+s:2m,s+n_b:n) \end{bmatrix}$.
  Update $\begin{bmatrix} Q(:,s:m) & Q(:,m+s:2m) \end{bmatrix} \leftarrow \begin{bmatrix} Q(:,s:m) & Q(:,m+s:2m) \end{bmatrix} Q_p$.
end for

4.4 Skew-Hamiltonian Matrices

Imposing skew-Hamiltonian structure on a matrix W has a number of consequences for the eigenvalues and eigenvectors of W; one is that every eigenvalue has even algebraic multiplicity and hence appears at least twice. An easy way to access all these spectral properties is to observe that for any skew-Hamiltonian matrix W there exists a symplectic matrix S so that

$$S^{-1} W S = \begin{bmatrix} W_{11} & 0 \\ 0 & W_{11}^T \end{bmatrix}. \tag{4.73}$$

This decomposition – among others – will be described in the following subsection.

4.4.1 Structured Decompositions

By constructing a sequence of elementary orthogonal symplectic transformation matrices we obtain the following structured Hessenberg-like decomposition for skew-Hamiltonian matrices.

Theorem 4.33 (PVL decomposition [333]). *Let $W \in \mathbb{R}^{2n \times 2n}$ be skew-Hamiltonian. Then there exists an orthogonal symplectic matrix U so that $U^T W U$ has* Paige/Van Loan (PVL) *form, i.e.,*

$$U^T W U = \begin{bmatrix} W_{11} & W_{12} \\ 0 & W_{11}^T \end{bmatrix} = \begin{bmatrix} \ddots & \vdots \\ & \ddots \end{bmatrix}, \tag{4.74}$$

where $W_{11} \in \mathbb{R}^{n \times n}$ is an upper Hessenberg matrix.

The PVL decomposition (4.74) is a consequence of Algorithm 26. Let us illus-

Algorithm 26 PVL decomposition of skew-Hamiltonian matrix

Input: A skew-Hamiltonian matrix $W \in \mathbb{R}^{2n\times 2n}$.
Output: An orthogonal symplectic matrix $U \in \mathbb{R}^{2n\times 2n}$; W is overwritten with $U^T W U$ having PVL form (4.74).

$U \leftarrow I_{2n}$
for $j \leftarrow 1, \ldots, n-1$ **do**
  $x \leftarrow We_j$
  Apply Algorithm 23 to compute $E_{j+1}(x)$.
  $W \leftarrow E_{j+1}(x)^T W E_{j+1}(x), \quad U \leftarrow U E_{j+1}(x)$
end for

trate its idea for $n = 4$. First, $E_2(We_1)$ is used to annihilate entries $3, \ldots, 8$ in the first column of W:

$$W \leftarrow E_2(We_1)^T W E_2(We_1) = \left[\begin{array}{cccc|cccc} a & \hat{a} & \hat{a} & \hat{a} & 0 & \hat{g} & \hat{g} & \hat{g} \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & 0 & \hat{g} & \hat{g} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & 0 & \hat{g} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & 0 \\ \hline 0 & \hat{0} & \hat{0} & \hat{0} & a & \hat{a} & \hat{0} & \hat{0} \\ \hat{0} & 0 & \hat{q} & \hat{q} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{q} & 0 & \hat{q} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{q} & \hat{q} & 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \end{array}\right]$$

Columns two and three are reduced by applying $E_3(We_2)$ and $E_4(We_3)$ consecutively:

$$W \leftarrow E_3(We_2)^T W E_3(We_2) = \left[\begin{array}{cccc|cccc} a & a & \hat{a} & \hat{a} & 0 & g & \hat{g} & \hat{g} \\ a & a & \hat{a} & \hat{a} & g & 0 & \hat{g} & \hat{g} \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & 0 & \hat{g} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & 0 \\ \hline 0 & 0 & 0 & 0 & a & a & 0 & 0 \\ 0 & 0 & \hat{0} & \hat{0} & a & a & \hat{a} & \hat{0} \\ 0 & \hat{0} & 0 & \hat{q} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{q} & 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \end{array}\right],$$

$$W \leftarrow E_4(We_3)^T W E_4(We_3) = \left[\begin{array}{cccc|cccc} a & a & a & \hat{a} & 0 & g & g & \hat{g} \\ a & a & a & \hat{a} & g & 0 & g & \hat{g} \\ 0 & a & a & \hat{a} & g & g & 0 & \hat{g} \\ 0 & 0 & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & 0 \\ \hline 0 & 0 & 0 & 0 & a & a & 0 & 0 \\ 0 & 0 & 0 & 0 & a & a & a & \hat{0} \\ 0 & 0 & 0 & \hat{0} & a & a & a & \hat{a} \\ 0 & 0 & \hat{0} & 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \end{array}\right].$$

Algorithm 26 is implemented in the HAPACK routine `DSHPVL`; it requires $\frac{40}{3}n^3 + \mathcal{O}(n^2)$ flops for reducing W and additionally $\frac{16}{3}n^3 + \mathcal{O}(n^2)$ flops for computing the orthogonal symplectic factor U. The HAPACK routine `DSHPVB` employs a blocked version of Algorithm 26 based on the WY-like representation described in Section 4.3.3.

An immediate consequence of the PVL decomposition (4.74) is that every eigenvalue of W has even algebraic multiplicity. The same is true for the geometric multiplicities. To see this we need to eliminate the skew-symmetric off-diagonal block W_{12}, for which we can use solutions of the following singular Sylvester equation.

Proposition 4.34. *The Sylvester equation*

$$W_{11}P - PW_{11}^T = -W_{12} \tag{4.75}$$

is solvable for all skew-symmetric matrices $W_{12} \in \mathbb{R}^{n\times n}$ if and only if $W_{11} \in \mathbb{R}^{n\times n}$ is nonderogatory*, i.e., every eigenvalue of W_{11} has geometric multiplicity one. In this case, any solution P of (4.75) is real and symmetric.*

Proof. The first part of the proposition can be found in [133, 125]. Although the second part is not explicitly stated in [125], it follows from the results obtained in [125]. For the sake of completeness we provide the complete proof.

W.l.o.g., we may assume that W_{11} is in real Jordan canonical form. Partition (4.75) into

$$\begin{bmatrix} J_1 & 0 \\ 0 & J_2 \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} - \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \begin{bmatrix} J_1^T & 0 \\ 0 & J_2^T \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} \\ -B_{12}^T & B_{22} \end{bmatrix}, \tag{4.76}$$

where $J_2 \in \mathbb{R}^{m\times m}$ is a single real Jordan block. The Sylvester equation

$$J_1P_{12} - P_{12}J_2^T = B_{12}$$

is solvable for all right hand sides B_{12} if and only if $\lambda(J_1) \cap \lambda(J_2) = \emptyset$. In this case, the solution is unique and as P_{21}^T satisfies the same equation, it follows that $P_{21} = P_{12}^T$. The $m \times m$ matrix P_{22} satisfies

$$J_2P_{22} - P_{22}J_2^T = B_{22} \tag{4.77}$$

If J_2 is a scalar then every $P_{22} \in \mathbb{R}$ is a solution of (4.77). If $J_2 = \left[\begin{smallmatrix} \alpha & \beta \\ -\beta & \alpha \end{smallmatrix}\right]$ with $\beta \neq 0$ is a two-by-two matrix corresponding to a complex conjugate pair of eigenvalues then (4.77) can be written as

$$\begin{bmatrix} \alpha & \beta \\ -\beta & \alpha \end{bmatrix} P_{22} - P_{22} \begin{bmatrix} \alpha & -\beta \\ \beta & \alpha \end{bmatrix} = \begin{bmatrix} 0 & \gamma \\ -\gamma & 0 \end{bmatrix}.$$

Since $\beta \neq 0$, any solution to this equation has the form

$$P_{22} = \begin{bmatrix} \delta & \eta \\ \eta & \delta - \gamma/\beta \end{bmatrix},$$

for arbitrary $\eta, \delta \in \mathbb{R}$. If J_2 is a general $m \times m$ Jordan block then a solution of (4.77) can be obtained by combining the above results with backward substitution. It remains to prove that any solution P_{22} of (4.77) is symmetric. For this purpose decompose $P_{22} = X + Y$, where X is the symmetric part and Y is the skew-symmetric part of P_{22}. Then Y must satisfy $J_2 Y - Y J_2^T = 0$. If F denotes the *flip matrix*, i.e., F has ones on its anti-diagonal and zeros everywhere else, then this equation implies that YF commutes with J_2, because of $(FJ_2F)^T = J_2$. By a well-known result from linear algebra (see, e.g., [170]) the only matrices that commute with Jordan blocks corresponding to a single real eigenvalue are upper triangular Toeplitz matrices meaning that Y is a Hankel matrix. However, the only skew-symmetric Hankel matrix is $Y = 0$. Similarly, if J_2 corresponds to a complex conjugate pair of eigenvalues, let F_2 be the matrix that is zero except its antidiagonal blocks which are two-by-two identity matrices. Then $J_2(YF_2) - (YF_2)\tilde{J}_2 = 0$, where $\tilde{J}_2$ is identical with J_2 besides that its two-by-two diagonal blocks are transposed. Along the lines of the proof for the scalar case it can be shown that YF_2 is a block upper triangular Toeplitz matrix with symmetric two-by-two diagonal blocks. Thus Y is a skew-symmetric block Hankel matrix with symmetric blocks. Again, the only matrix satisfying these constraints is $Y = 0$.

Thus, we have shown that all solutions of (4.76) have the property that P_{22} is symmetric and $P_{21} = P_{12}^T$, given the assumption that $\lambda(J_1) \cap \lambda(J_2) = \emptyset$ holds. If this condition fails then there exists no solution. The proof is completed by applying an induction argument to P_{11}. □

We now use this proposition to block-diagonalize a skew-Hamiltonian matrix in PVL form (4.74) assuming that W_{11} is nonderogatory. For this purpose let R be a solution of (4.75), then the symmetry of R implies that $\left[\begin{smallmatrix} I & R \\ 0 & I \end{smallmatrix}\right]$ is symplectic. Applying the corresponding symplectic similarity transformation yields the transformed matrix

$$\begin{bmatrix} I & R \\ 0 & I \end{bmatrix}^{-1} \begin{bmatrix} W_{11} & W_{12} \\ 0 & W_{11}^T \end{bmatrix} \begin{bmatrix} I & R \\ 0 & I \end{bmatrix} = \begin{bmatrix} W_{11} & 0 \\ 0 & W_{11}^T \end{bmatrix}. \tag{4.78}$$

This also shows that if x is a right eigenvector belonging to an eigenvalue λ of W then $J\bar{x}$ is a left eigenvector belonging λ.

Note that there is a lot of freedom in the choice of R as equation (4.75) admits infinitely many solutions. From a numerical point of view the matrix R should be chosen so that its norm is as small as possible. The same question arises in the context of structured condition numbers and will be discussed in Section 4.4.2 below.

It should be stressed that assuming W_{11} to be nonderogatory is not necessary and thus, the even geometric multiplicity of eigenvalues also holds in the general case. In fact, Faßbender, Mackey, Mackey and Xu [125] have shown that any skew-Hamiltonian matrix can be reduced to block diagonal form (4.78) using symplectic similarity transformations. The proof, however, is much more involved than the derivation given above.

Another way to go from a skew-Hamiltonian matrix W in PVL form (4.74) to a more condensed form is to reduce W_{11} further to real Schur form. This can be achieved by constructing an orthogonal matrix Q_1 so that $T = Q_1^T W_{11} Q_1$ is in *real Schur form*, see Theorem 1.2. Setting $\tilde{U} = U(Q_1 \oplus Q_1)$, we obtain a *skew-Hamiltonian (real) Schur decomposition* of W:

$$\tilde{U}^T W \tilde{U} = \begin{bmatrix} T & \tilde{G} \\ 0 & T^T \end{bmatrix}, \tag{4.79}$$

where $\tilde{G} = Q_1^T W_{12} Q_1$ is again skew-symmetric.

4.4.2 Perturbation Analysis

In this section we investigate the change of eigenvalues and certain invariant subspaces of a skew-Hamiltonian matrix W under a sufficiently small, skew-Hamiltonian perturbation E. Requiring the perturbation to be structured as well may have a strong positive impact on the sensitivity of the skew-Hamiltonian eigenvalue problem; this is demonstrated by the following example.

Example 4.35. Consider the parameter-dependent matrix

$$W(\varepsilon_1, \varepsilon_2) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ \varepsilon_1 & \varepsilon_2 & 1 & 0 \\ -\varepsilon_2 & 0 & 0 & 2 \end{bmatrix}.$$

The vector $e_1 = [1, 0, 0, 0]^T$ is an eigenvector of $W(0, 0)$ associated with the eigenvalue $\lambda = 1$. No matter how small $\varepsilon_1 > 0$ is, any eigenvector of $W(\varepsilon_1, 0)$ belonging to λ has the completely different form $[0, 0, \alpha, 0]^T$ for some $\alpha \neq 0$. On the other hand, $W(0, \varepsilon_2)$ has an eigenvector $[1, 0, 0, \varepsilon_2]^T$ rather close to e_1. The fundamental difference between $W(\varepsilon_1, 0)$ and $W(0, \varepsilon_2)$ is that the latter is a skew-Hamiltonian matrix while the former is not. ◊

Eigenvalues

We now turn to the structured perturbation analysis for eigenvalues of skew-Hamiltonian matrices. As λ is necessarily a multiple eigenvalue we cannot apply the techniques from Section 4.1.1, which rely on perturbation expansions of eigenvalues. Instead, we must consider the eigenvalue cluster containing all copies of λ and apply the corresponding expansion, see Theorem 1.9.

Assuming that λ has algebraic multiplicity two, there exist two linearly independent eigenvectors x_1 and x_2 belonging to λ. Let $[x_1, x_2] = XR$ be a QR decomposition with unitary $X \in \mathbb{C}^{2n \times 2}$ and upper triangular $R \in \mathbb{C}^{2 \times 2}$, then

$$WX = W[x_1, x_2]R^{-1} = [x_1, x_2]A_{11}R^{-1} = [x_1, x_2]R^{-1}A_{11} = XA_{11},$$

where $A_{11} = \mathrm{diag}(\lambda, \lambda)$. An analogous relation holds for the two eigenvectors $\hat{x}_1, \hat{x}_2$ belonging to the eigenvalue $\hat{\lambda}$ of the perturbed matrix $W + E$. As the spectral norm of $\hat{A}_{11} - A_{11}$ is given by $|\hat{\lambda} - \lambda|$, the expansion (1.20) in Theorem 1.9 implies

$$\begin{aligned} |\hat{\lambda} - \lambda| &= \|(X^T JX)^{-1} X^T JEX\|_2 + \mathcal{O}(\|E\|^2) \\ &\leq \|(X^T JX)^{-1}\|_2 + \mathcal{O}(\|E\|^2), \end{aligned} \tag{4.80}$$

where we also used the fact that the columns of $J\bar{X}$ span the two-dimensional left invariant subspace belonging to λ. (Note that $\bar{X}$ denotes the complex conjugate of X.) This yields the following structured perturbation result for eigenvalues of skew-Hamiltonian matrices.

Corollary 4.36. *Let $W, E \in \mathbb{R}^{2n\times 2n}$ be skew-Hamiltonian matrices. Assume that λ is an eigenvalue of W having multiplicity two. Then there exists an eigenvalue $\hat{\lambda}$ of $W + E$ so that*

$$|\hat{\lambda} - \lambda| \leq \|P\|_2 \, \|E\|_2 + \mathcal{O}(\|E\|^2), \tag{4.81}$$

where P is the spectral projector belonging to the eigenvalue cluster $\{\lambda, \lambda\}$.

In order to prove that $\|P\|_2$ is the appropriate structured condition number we have to show that there exists a skew-Hamiltonian perturbation E such that inequality (4.80) is approximately attained. For real λ we may assume $X \in \mathbb{R}^{2n\times 2}$ and make use of the skew-Hamiltonian perturbation $E = \varepsilon J_{2n}^T X J_2 X^T$. This implies that the *structured condition number for an eigenvalue $\lambda \in \mathbb{R}$ of a skew-Hamiltonian matrix* satisfies

$$c_2^{\mathrm{sHam}}(\lambda) := \lim_{\varepsilon\to 0} \frac{1}{\varepsilon} \sup\{|\hat{\lambda} - \lambda| : \ \|E\|_2 \leq \varepsilon, \, E \text{ is skew-Hamiltonian}\} = \|P\|_2.$$

There is a simple expression for computing $\|P\|_2$ from the eigenvectors belonging to λ.

Lemma 4.37. *Under the assumptions of Corollary 4.36,*

$$\|P\|_2 = \frac{1}{|x_1^T J x_2|} \sqrt{\|x_1\|_2^2 \, \|x_2\|_2^2 - |x_1^H x_2|^2},$$

where $x_1, x_2 \in \mathbb{C}^{2n}$ are two linearly independent eigenvectors belonging to λ.

Proof. See [38]. □

Structured backward errors and condition numbers for eigenvalues of skew-Hamiltonian matrices with additional structure can be found in [323].

Invariant Subspaces

Invariant subspaces of skew-Hamiltonian matrices of interest in applications are usually isotropic.

Definition 4.38. *A subspace $\mathcal{X} \subseteq \mathbb{R}^{2n}$ is called* isotropic *if $\mathcal{X} \perp J_{2n}\mathcal{X}$. A maximal isotropic subspace is called* Lagrangian.

Since $x^T Jx = 0$ for an arbitrary vector x, any eigenvector of W spans an isotropic invariant subspace. Also the first $k \leq n$ columns of the matrix $\tilde{U}$ in a skew-Hamiltonian Schur decomposition (4.79) share this property. Roughly speaking, an invariant subspace $\mathcal{X}$ of W is isotropic if $\mathcal{X}$ is spanned by the first $k \leq n$ columns of a symplectic matrix.

The following lemma investigates the relationship between isotropic invariant subspaces and skew-Hamiltonian Schur decompositions in more detail.

Lemma 4.39. *Let $W \in \mathbb{R}^{2n\times 2n}$ be a skew-Hamiltonian matrix and let $X \in \mathbb{R}^{2n\times k}$ ($k \leq n$) have orthonormal columns. Then the columns of X span an isotropic invariant subspace of W if and only if there exists an orthogonal symplectic matrix $U = [X, Z, J^T X, J^T Z]$ with some $Z \in \mathbb{R}^{2n\times(n-k)}$ so that*

$$U^T W U = \begin{array}{c} \\ k \\ n-k \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cccc} k & n-k & k & n-k \end{array} \\ \begin{bmatrix} A_{11} & A_{12} & G_{11} & G_{12} \\ 0 & A_{22} & -G_{12}^T & G_{22} \\ 0 & 0 & A_{11}^T & 0 \\ 0 & Q_{22} & A_{12}^T & A_{22}^T \end{bmatrix} \end{array}. \tag{4.82}$$

Proof. Assume that the columns of X span an isotropic subspace. Then the symplectic QR decomposition can be used to construct an orthogonal symplectic matrix $U = [X, Z, J^T X, J^T Z]$. Moreover, if the columns of X span an invariant subspace then $[Z, J^T X, J^T Z]^T W X = 0$, completing the proof of (4.82). The other direction is straightforward. □

Necessarily, an isotropic invariant subspace $\mathcal{X}$ is not simple implying $c(\mathcal{X}) = \infty$, i.e., $\mathcal{X}$ is arbitrarily ill-conditioned under general perturbations. This fact is illustrated in Example 4.35. In [200], it is shown that this effect does not occur under skew-Hamiltonian perturbations, provided that $\mathcal{X}$ is semi-simple.

Definition 4.40. *Let the columns of $X \in \mathbb{R}^{2n\times k}$ form an orthogonal basis for an isotropic invariant subspace $\mathcal{X}$ of a skew-Hamiltonian matrix W. Furthermore, choose $Z \in \mathbb{R}^{2n\times(n-k)}$ so that $U = [X, Z, J^T X, J^T Z]$ is orthogonal symplectic and $U^T W U$ has the form (4.82). Then $\mathcal{X}$ is called* semi-simple *if $\lambda(A_{11}) \cap \lambda\left(\begin{bmatrix} A_{22} & G_{22} \\ Q_{22} & A_{22}^T \end{bmatrix}\right) = \emptyset$ and A_{11} is* nonderogatory*, i.e., each eigenvalue of A_{11} has geometric multiplicity one.*

For simplicity, let us restrict ourselves to an n-dimensional isotropic invariant subspace $\mathcal{X} = \operatorname{span}(X)$ with $X \in \mathbb{R}^{2n\times n}$. The corresponding block Schur decomposition takes the form

$$[X, JX]^T W [X, JX] = \begin{bmatrix} W_{11} & W_{12} \\ 0 & W_{11}^T \end{bmatrix},$$

and the associated Sylvester operator is given by

$$\mathbf{T} : R \mapsto W_{11}^T R - R W_{11}.$$

Assuming $\mathcal{X}$ to be semi-simple yields a nonderogatory W_{11}. By Proposition 4.34, this implies that any R which yields a skew-symmetric $\mathbf{T}(R)$ is necessarily a symmetric matrix.

In the following, we attempt to apply the structured perturbation technique for invariant subspaces developed in Section 4.1.1. If $\mathcal{M} \equiv \text{sHam}$ denotes the set of $2n \times 2n$ skew-Hamiltonian matrices then

$$\mathcal{N} = \{(JX)^T E J X : \ E \in \text{sHam}\} = \text{skew},$$

i.e., $\mathcal{N}$ consists of the set of all $n\times n$ skew-symmetric matrices. Note that skew has dimension $n(n-1)/2$ while symm, the set of $n\times n$ symmetric matrices, has dimension $n(n+1)/2$. Thus there is some freedom in the choice of an $n(n-1)/2$-dimensional matrix space $\mathcal{L} \subset \text{symm}$. By the results of Section 4.1.1, any such $\mathcal{L}$ yields a restricted operator $\mathbf{T}_s = \mathbf{T}\big|_{\mathcal{L}\to\mathcal{N}}$ with

$$c^{\text{sHam}}(\mathcal{X}) \le \|\mathbf{T}_s^{-1}\|.$$

The set $\mathcal{L}$ which yields the lowest bound on $c^{\text{sHam}}(\mathcal{X})$ is given by

$$\mathcal{L} = \left\{ R_\star : \ E_{21} \in \text{skew}, \ \|R_\star\|_F = \min\{\|R\|_F : \ \mathbf{T}(R) = E_{21}\} \right\} \tag{4.83}$$

Under this choice of $\mathcal{L}$, it can actually be shown that $c^{\text{sHam}}(\mathcal{X}) = \|\mathbf{T}_s^{-1}\|$ [200].

The derivation of structured condition numbers for lower-dimensional isotropic invariant subspaces is technically more complicated but follows a similar methodology [200]. For eigenvectors, we have the following result.

Lemma 4.41. *Let $x \in \mathbb{R}^n$ with $\|x\|_2 = 1$ be an eigenvector of a skew-Hamiltonian matrix W belonging to an eigenvalue $\lambda \in \mathbb{R}$ of algebraic multiplicity two. Consider a block Schur decomposition of the form (4.82) with $A_{11} = \lambda$ and $X = x$. Then $c^{\text{sHam}} = \sigma_{\min}^{-1}(W_\lambda)$ with*

$$W_\lambda = \begin{bmatrix} A_{22} - \lambda I & G_{22} & -G_{12}^T \\ Q_{22} & A_{22}^T - \lambda I & A_{12}^T \end{bmatrix}.$$

4.4.3 A QR-Based Algorithm

In Section 4.4.1 we used a constructive approach to prove the skew-Hamiltonian Schur decomposition

$$U^T W U = \begin{bmatrix} T & \tilde{G} \\ 0 & T^T \end{bmatrix}, \tag{4.84}$$

where U is orthogonal symplectic and T has real Schur form. Algorithm 27 summarizes this construction. This algorithm is implemented in the HAPACK

Algorithm 27 Skew-Hamiltonian Schur decomposition

Input: A skew-Hamiltonian matrix $W \in \mathbb{R}^{2n\times 2n}$.
Output: An orthogonal symplectic matrix $U \in \mathbb{R}^{2n\times 2n}$; W is overwritten with $U^T W U$ having skew-Hamiltonian Schur form (4.84).

1. Apply Algorithm 26 to compute an orthogonal symplectic matrix U such that $W \leftarrow U^T W U$ has PVL form.
2. Apply the QR algorithm to the $(1,1)$ block W_{11} of W to compute an orthogonal matrix V such that $V^T W_{11} V$ has real Schur form.
3. Update $W \leftarrow (V \oplus V)^T W (Q \oplus Q)$, $U \leftarrow U(V \oplus V)$.

routine `DSHES`; it requires roughly $\frac{62}{3}n^3$ flops if only the eigenvalues are desired, and $\frac{136}{3}n^3$ flops if the skew-Hamiltonian Schur form and the orthogonal symplectic factor U are computed. Note that these numbers are based on the flop estimates for the QR algorithm with two Francis shifts listed on page 31. This compares favorably with the QR algorithm applied to the whole matrix W, which takes $\frac{256}{3}n^3$ and $\frac{640}{3}n^3$ flops, respectively.

The finite-precision properties of Algorithm 27 are as follows. Similarly as for the QR algorithm [364, 322] one can show that there exists an orthogonal symplectic matrix Z which transforms the computed skew-Hamiltonian Schur form $\hat{W} = \begin{bmatrix} \hat{T} & \hat{G} \\ 0 & \hat{T}^T \end{bmatrix}$ to a skew-Hamiltonian matrix near to W, i.e., $Z\hat{W}Z^T = W + E$, where E is skew-Hamiltonian, $\|E\|_2 = \mathcal{O}(\mathbf{u})\|W\|_2$ and $\mathbf{u}$ denotes the unit roundoff. Moreover, the computed factor $\hat{U}$ is almost orthogonal in the sense that $\|\hat{U}^T\hat{U} - I\|_2 = \mathcal{O}(\mathbf{u})$, and it has the block representation $\hat{U} = \begin{bmatrix} \hat{U}_1 & \hat{U}_2 \\ -\hat{U}_2 & \hat{U}_1 \end{bmatrix}$. This implies that $\hat{U}$ is close to an orthogonal symplectic matrix, see Lemma 4.31. To summarize, Algorithm 27 is a strongly backward stable method for computing eigenvalues of skew-Hamiltonian matrices.

4.4.4 Computation of Invariant Subspaces

Once a skew-Hamiltonian Schur decomposition (4.84) has been computed, the eigenvalues can be easily obtained from the diagonal blocks of T. Furthermore, if the $(k+1,k)$ entry of T is zero, then the first k columns of U span an isotropic

invariant subspace $\mathcal{X}$ of W belonging to the eigenvalues of $T(1:k, 1:k)$. Isotropic invariant subspaces belonging to other eigenvalues can be obtained by swapping the diagonal blocks of T as described in Section 1.7.

4.4.5 SHIRA

SHIRA [242] is a structure-preserving variant of the implicitly restarted Arnoldi algorithm suitable for computing a few eigenvalues and the associated isotropic invariant subspaces of a large and possibly sparse skew-Hamiltonian matrix W. It is based on the following fact.

Lemma 4.42 ([242]). *Let $W \in \mathbb{R}^{2n\times 2n}$ be a skew-Hamiltonian matrix and let $u_1 \in \mathbb{R}^{2n}$. Then any Krylov subspace $\mathcal{K}_k(W, u_1)$ is isotropic.*

Proof. For arbitrary integers i and j we have $(W^i)^T J W^j = J W^{i+j}$, which is a skew-symmetric matrix since W^{i+j} is skew-Hamiltonian. Thus

$$K_k(W, u_1)^T J K_k(W, u_1) = 0,$$

with the Krylov matrix $K_k(W, u_1)$, which implies that $\mathcal{K}_k(W, u_1)$ is isotropic. □

Let us now consider an unreduced Arnoldi decomposition (see Section 3.1.2) of order $k < n$:

$$W U_k = [\, U_k,\ u_{k+1} \,] \begin{bmatrix} H_k \\ h_{k+1,k} e_k^T \end{bmatrix}. \tag{4.85}$$

Lemma 4.42 implies that the columns of $[U_{k+1}, JU_{k+1}]$ form an orthonormal basis. Roundoff errors will cloud the situation, as usual. It is thus necessary to modify the Arnoldi method, Algorithm 15, so that a newly produced vector $v = Au_j$ is orthogonalized not only against all previously generated vectors contained in U_j, but also against the columns of JU_j.

In the generic case, every eigenvalue λ of W has two linearly independent eigenvectors x_1, x_2 satisfying $x_1^T J x_2 = 0$. An *isotropic* Krylov subspace can only contain approximations to *one* eigenvector belonging to λ. Consequently, the eigenvalues do not appear in duplicate in the Hessenberg factor H_k. Note that (4.85) can be considered as a truncated PVL decomposition:

$$[\, U_k,\ JU_k \,]^T W [\, U_k,\ JU_k \,] = \begin{bmatrix} H_k & U_k^T W J U_k \\ 0 & H_k^T \end{bmatrix} = \begin{bmatrix} \ddots & \ddots & 0 \\ & \ddots & \ddots \\ & & \ddots \end{bmatrix}.$$

SHIRA inherits the restarting and deflation strategies from the implicitly restarted Arnoldi method, making it very simple to implement. For example, only a few changes to the source code of ARPACK are necessary to enforce isotropy of the Krylov subspace. Numerical experiments in [14, 242] show

that such an implementation of SHIRA is usually more efficient than ARPACK for computing a few eigenvalues of a skew-Hamiltonian matrix due to the fact that ARPACK often computes duplicate eigenvalues. Of course, one can also use Krylov-Schur algorithms for restarting and deflation. Again, the isotropy of Krylov subspaces must be enforced while expanding a Krylov decomposition.

There is another advantage to be gained from enforcing isotropic Krylov subspaces. Assume that the first d Ritz values of the mth order Krylov decomposition

$$W[Q_d, U_{m-d}] = [Q_d, U_{m-d+1}] \left[\begin{array}{cc} T_d & \star \\ \hline 0 & B_{m-d} \\ 0 & b_{m-d}^T \end{array}\right],$$

have been deflated based upon a criterion of the form (3.15). Then the Frobenius norm of the residual $R_d = WQ_d - Q_dT_d$ can be bounded by $\sqrt{d}\max\{\mathbf{u}, \mathtt{tol}\}\,\|W\|_F$. Since $Q_d^T(JQ_d) = 0$, it follows that Q_d is the exact *isotropic* invariant subspace of the slightly perturbed skew-Hamiltonian matrix

$$\hat{W} = W - R_dQ_d^T + J(R_dQ_d^T)^TJ(I - Q_dQ_d^T),$$

see also [323].

4.4.6 Other Algorithms and Extensions

Similarly as the Hessenberg form of a general matrix can be computed by Gauss transformations [141, Sec. 7.4.7] it has been shown by Stefanovski and Trenčevski [298] how non-orthogonal symplectic transformations can be used to compute the PVL form of a skew-Hamiltonian matrix. An unsymmetric Lanczos process for skew-Hamiltonian matrices has been proposed in [355].

A matrix pair (A, B) is called skew-Hamiltonian if $BJA^T = AJB^T$. Skew-Hamiltonian matrix pairs have the same spectral properties as skew-Hamiltonian matrices and can be addressed by a suitable generalization of the PVL decomposition [205].

4.5 Hamiltonian matrices

One of the most remarkable properties of a Hamiltonian matrix

$$H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}, \quad G = G^T, \quad Q = Q^T,$$

is that its eigenvalues always occur in pairs $\{\lambda, -\lambda\}$, if $\lambda \in \mathbb{R} \cup \imath\mathbb{R}$, or in quadruples $\{\lambda, -\lambda, \bar{\lambda}, -\bar{\lambda}\}$, if $\lambda \in \mathbb{C}\backslash(\mathbb{R} \cup \imath\mathbb{R})$. The preservation of these pairings in finite-precision arithmetic is a major benefit from using structure-preserving algorithms for computing the eigenvalues of H.

We will only briefly touch the difficulties that arise when H has eigenvalues on the imaginary axis. Although this case is well-analyzed with respect

to structured decompositions, see [269, 270, 271, 219, 220, 129] and the references given therein, it is still an open research problem to define appropriate structured condition numbers and design satisfactory algorithms for this case.

4.5.1 Structured Decompositions

A major difficulty in developing computational methods for the Hamiltonian eigenvalue problem is that there is so far no $\mathcal{O}(n^3)$ method for computing a useful structured Hessenberg-like form known. Although a slight modification of Algorithm 26 can be used to construct an orthogonal symplectic matrix U so that

$$U^T H U = \begin{bmatrix} \tilde{A} & \tilde{G} \\ \tilde{Q} & -\tilde{A}^T \end{bmatrix} = \begin{bmatrix} \text{⧅} & \square \\ \text{⧄} & \text{⧅} \end{bmatrix},$$

i.e., $\tilde{A}$ has upper Hessenberg form and $\tilde{Q}$ is a diagonal matrix, this Hamiltonian PVL decomposition [260] is of limited use. The Hamiltonian QR algorithm, see Section 4.5.3 below, only preserves this form if the $(2,1)$ block can be written as $Q = \gamma e_n e_n^T$ for some $\gamma \in \mathbb{R}$. In this case, $U^T H U$ is called a *Hamiltonian Hessenberg form*. Byers [76] derived a simple method for reducing H to such a form under the assumption that one of the off-diagonal blocks G or Q in H has tiny rank, i.e., rank 1, 2 or at most 3.

The general case, however, remains elusive. That it might be difficult to find a simple method is indicated by a result in [8], which shows that the first column x of an orthogonal symplectic matrix U that reduces H to Hamiltonian Hessenberg form has to satisfy the nonlinear equations

$$x^T J H^{2i-1} x = 0, \quad i = 1, \ldots, n.$$

This result can even be extended to non-orthogonal symplectic transformations [268].

A Schur-like form for Hamiltonian matrices is given by the following theorem [260, 220].

Theorem 4.43. *Let H be a Hamiltonian matrix and assume that all eigenvalues of H that are on the imaginary axis have even algebraic multiplicity. Then there exists an orthogonal symplectic matrix U so that $U^T H U$ is in* Hamiltonian Schur form, *i.e.,*

$$U^T H U = \begin{bmatrix} T & \tilde{G} \\ 0 & -T^T \end{bmatrix}, \tag{4.86}$$

where $T \in \mathbb{R}^{n \times n}$ has real Schur form.

4.5.2 Perturbation Analysis

An extensive perturbation analysis of (block) Hamiltonian Schur forms for the case that H has no purely imaginary eigenvalues has been presented in [195]. The analysis used therein is based on the technique of splitting operators and Lyapunov majorants. The approach used in this section is somewhat simpler; it is based on the techniques developed in Section 4.1.1.

Eigenvalues

Let λ be a simple eigenvalue of a Hamiltonian matrix H with normalized right and left eigenvectors x and y, respectively. Equation (4.6) yields

$$c_2^{\mathrm{Ham}}(\lambda) = \frac{1}{|y^H x|} \sup \left\{ |y^H E x| : \ E \in \mathrm{Ham}, \ \|E\|_2 = 1 \right\}$$

where Ham denotes the set of $2n \times 2n$ Hamiltonian matrices. If λ is real we may assume x and y to be real and can construct a Householder matrix V which maps x to Jy. Then $E = JV$ is a Hamiltonian matrix that satisfies $\|E\|_2 = 1$ and $|y^T E x| = |y^T y| = 1$. Hence, $c_2^{\mathrm{Ham}}(\lambda) = c(\lambda)$. The same argument can be used to show this equality for a purely imaginary λ [188].

If λ is neither real nor purely imaginary then the structured and unstructured eigenvalue condition numbers may differ. Still, one can show

$$\frac{1}{\sqrt{2}} c(\lambda) \le c_2^{\mathrm{Ham}}(\lambda) \le c(\lambda),$$

see [188, 277].

Invariant Subspaces

Let the columns of $X \in \mathbb{R}^{2n \times k}$ span a simple isotropic invariant subspace $\mathcal{X}$ of H. By the symplectic QR decomposition there exists a matrix $Y \in \mathbb{R}^{2n \times k}$ so that $U = [X, Y, JX, JY]$ is an orthogonal symplectic matrix. Moreover, we have the block Hamiltonian Schur form

$$U^T H U = \begin{array}{c} \\ k \\ n-k \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cccc} k & n-k & k & n-k \end{array} \\ \begin{bmatrix} A_{11} & A_{12} & G_{11} & G_{12} \\ 0 & A_{22} & G_{12}^T & G_{22} \\ 0 & 0 & -A_{11}^T & 0 \\ 0 & Q_{22} & -A_{12}^T & -A_{22}^T \end{bmatrix} \end{array},$$

see also Lemma 4.39. The associated Sylvester operator is given by

$$\mathbf{T} : \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} \mapsto \begin{bmatrix} A_{22} & G_{12}^T & G_{22} \\ 0 & -A_{11}^T & 0 \\ Q_{22} & -A_{12}^T & -A_{22}^T \end{bmatrix} \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} - \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} A_{11}.$$

If we let

$$\mathcal{L} = \left\{ \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} : \; R_1, R_3 \in \mathbb{R}^{(n-k)\times k}, \; R_2 \in \mathbb{R}^{k\times k}, \; R_2 \in \text{sym} \right\}$$

then $\{[Y, JX, JY]^T HX : \; H \in \text{Ham}\} = \mathcal{L}$ and $\mathbf{T} : \mathcal{L} \to \mathcal{L}$. Hence, the *structured condition number* for $\mathcal{X}$ is given by

$$c^{\text{Ham}}(\mathcal{X}) = \|\mathbf{T}_s^{-1}\|.$$

where $\mathbf{T}_s = \mathbf{T}\big|_{\mathcal{L}\to\mathcal{L}}$.

Obviously, $c^{\text{Ham}}(\mathcal{X})$ coincides with the unstructured condition number $c(\mathcal{X})$ if $\mathcal{X}$ is one-dimensional, i.e., $\mathcal{X}$ is spanned by a real eigenvector. A less trivial observation is that the same holds if $\mathcal{X}$ is the *stable invariant subspace*, i.e., the n-dimensional subspace belonging to all eigenvalues in the open left half plane. To see this, first note that in this case $\mathcal{L} = \text{symm}$ and

$$c^{\text{Ham}}(\mathcal{X}) = \sup_{\substack{S\neq 0 \\ S\in\text{symm}}} \frac{\|A_{11}^T S + S A_{11}\|_F}{\|S\|_F} = 1 \Big/ \inf_{\substack{R\neq 0 \\ R\in\text{symm}}} \frac{\|A_{11}R + RA_{11}^T\|_F}{\|R\|_F},$$

where $\lambda(A_{11}) \subset \mathbb{C}^-$. Using a result by Byers and Nash [85], we have

$$\inf_{\substack{R\neq 0 \\ R\in\text{symm}}} \frac{\|A_{11}R + RA_{11}^T\|_F}{\|R\|_F} = \inf_{R\neq 0} \frac{\|A_{11}R + RA_{11}^T\|_F}{\|R\|_F},$$

which indeed shows $c^{\text{Ham}}(\mathcal{X}) = c(\mathcal{X})$ for stable invariant subspaces.

4.5.3 An Explicit Hamiltonian QR Algorithm

Byers' Hamiltonian QR algorithm [76, 352] is a strongly backward stable method for computing the Hamiltonian Schur form of a Hamiltonian matrix H with no purely imaginary eigenvalues. Its only obstacle is that there is no implicit implementation of complexity less than $\mathcal{O}(n^4)$ known, except for the case when a Hamiltonian Hessenberg form exists [76, 78].

One iteration of the Hamiltonian QR algorithm computes the symplectic QR decomposition of the first n columns of the symplectic matrix

$$M = [(H - \sigma_1 I)(H - \sigma_2 I)][(H + \sigma_1 I)(H + \sigma_2 I)]^{-1}, \tag{4.87}$$

where $\{\sigma_1, \sigma_2\}$ is a pair of real or complex conjugate shifts. This yields an orthogonal symplectic matrix U so that

$$U^T M = \begin{bmatrix} \ddots & \square \\ & \ddots \end{bmatrix}. \tag{4.88}$$

The next iterate is obtained by updating $H \leftarrow U^T H U$. Let us partition H as follows:

$$H = \begin{array}{c} \\ 2 \\ n-2 \\ 2 \\ n-2 \end{array} \begin{array}{c} \begin{array}{cccc} 2 & n-2 & 2 & n-2 \end{array} \\ \begin{bmatrix} A_{11} & A_{12} & G_{11} & G_{12} \\ A_{21} & A_{22} & G_{12}^T & G_{22} \\ Q_{11} & Q_{12} & -A_{11}^T & -A_{21}^T \\ Q_{12}^T & Q_{22} & -A_{12}^T & -A_{22}^T \end{bmatrix} \end{array}. \tag{4.89}$$

In Section 1.3.1, we have seen that under rather mild assumptions and a fortunate choice of shifts, it can be shown that the submatrices A_{21}, Q_{11} and Q_{12} converge to zero, i.e., H converges to a block Hamiltonian Schur form. Choosing the shifts σ_1, σ_2 as the eigenvalues of the submatrix $\left[\begin{smallmatrix} A_{11} & G_{11} \\ Q_{11} & -A_{11}^T \end{smallmatrix}\right]$ that have positive real part results in quadratic convergence. If this submatrix happens to have two imaginary eigenvalues, we suggest to choose the one eigenvalue with positive real part twice, and if there are four purely imaginary eigenvalues, then one should perhaps resort to ad hoc shifts.

If the norms of the blocks A_{21}, Q_{11} and Q_{12} become less than $\mathbf{u}\,\|H\|_F$, then we may safely regard them as zero and apply the iteration to the submatrix $\left[\begin{smallmatrix} A_{22} & G_{22} \\ Q_{22} & -A_{22}^T \end{smallmatrix}\right]$. Eventually, this yields a Hamiltonian Schur form of H. Note that the Hamiltonian QR algorithm is not guaranteed to converge if H has eigenvalues on the imaginary axis. Still, one can observe convergence to a block Hamiltonian Schur form, where the unreduced block $\left[\begin{smallmatrix} A_{22} & G_{22} \\ Q_{22} & -A_{22}^T \end{smallmatrix}\right]$ contains all eigenvalues on the imaginary axis.

Remark 4.44. One can avoid the explicit computation of the potentially ill-conditioned matrix M in (4.87) by the following product QR decomposition approach. First, an orthogonal matrix Q_r is computed so that $(H+\sigma_1 I)(H+\sigma_2 I)Q_r^T$ has the block triangular structure displayed in (4.88). This can be achieved by a minor modification of the standard RQ decomposition [42]. Secondly, the orthogonal symplectic matrix U is computed from the symplectic QR decomposition of the first n columns of $(H-\sigma_1 I)(H-\sigma_2 I)Q_r^T$. ◊

4.5.4 Reordering a Hamiltonian Schur Decomposition

If the Hamiltonian QR algorithm has successfully computed a Hamiltonian Schur decomposition,

$$U^T H U = \begin{bmatrix} T & \tilde{G} \\ 0 & -T^T \end{bmatrix} \tag{4.90}$$

then the first n columns of the orthogonal symplectic matrix U span an isotropic subspace belonging to the eigenvalues of T. Many applications require the stable invariant subspace, for this purpose the Schur decomposition (4.90) must be reordered so that T contains all eigenvalues with negative real part.

One way to achieve this is as follows. If there is a block in T which contains a real positive eigenvalue or a pair of complex conjugate eigenvalues with positive real part, then this block is swapped to the bottom right diagonal block T_{mm} of T using the reordering algorithm described in Section 1.7.2. Now, let G_{mm} denote the corresponding block in $\tilde{G}$; it remains to find an orthogonal symplectic matrix U_{mm} so that

$$U_{mm}^T \begin{bmatrix} T_{mm} & G_{mm} \\ 0 & -T_{mm}^T \end{bmatrix} U_{mm} = \begin{bmatrix} \tilde{T}_{mm} & \tilde{G}_{mm} \\ 0 & -\tilde{T}_{mm}^T \end{bmatrix} \tag{4.91}$$

and the eigenvalues of $\tilde{T}_{mm}$ have negative real part. If X is the solution of the Lyapunov equation $T_{mm}X - XT_{mm}^T = G_{mm}$, then X is symmetric and consequently the columns of $[-X, I]^T$ span an isotropic subspace. Thus, there exists a symplectic QR decomposition

$$\begin{bmatrix} -X \\ I \end{bmatrix} = U_{mm} \begin{bmatrix} R \\ 0 \end{bmatrix}.$$

By direct computation, it can be seen that U_{mm} is an orthogonal symplectic matrix which produces a reordering of the form (4.91). As for the swapping algorithm described in Section 1.7.1 it may happen that in some pathological cases, the norm of the $(2,1)$ block in the reordered matrix is larger than $\mathcal{O}(\mathbf{u})\|H\|_F$. In this case, the swap must be rejected in order to guarantee the strong backward stability of the algorithm. A different kind of reordering algorithm, which is based on Hamiltonian QR iterations with perfect shifts, can be found in [76].

Conclusively, we have a method for computing eigenvalues and selected invariant subspaces of Hamiltonian matrices. This method is strongly backward stable and reliable, as long as there are no eigenvalues on the imaginary axis. However, as mentioned in the beginning of this section, in general it requires $\mathcal{O}(n^4)$ flops, making it unattractive for decently large problems.

4.5.5 Algorithms Based on H^2

One of the first $\mathcal{O}(n^3)$ structure-preserving methods for the Hamiltonian eigenvalue problem was developed by Van Loan [333]. It is based on the fact that H^2 is a skew-Hamiltonian matrix, because

$$(H^2J)^T = (HJ)^T H^T = HJH^T = -H(HJ)^T = -H^2J.$$

Thus, one can apply Algorithm 27 to H^2 and take the positive and negative square roots of the computed eigenvalues, which gives the eigenvalues of H. An implicit version of this algorithm, called *square-reduced method* (SQRED), has been implemented in [31]. The main advantage of this approach is that the eigenvalue symmetries of H are fully recovered in finite-precision arithmetic. Also, the computational cost is low when compared to the QR algorithm. The

disadvantage of Van Loan's method is that a loss of accuracy up to half the number of significant digits of the computed eigenvalues of H is possible. An error analysis in [333] shows that for an eigenvalue λ of H the computed $\hat{\lambda}$ satisfies

$$|\hat{\lambda} - \lambda| \lessapprox c(\lambda) \min\{\mathbf{u}\|H\|_2^2/|\lambda|, \sqrt{\mathbf{u}}\|H\|_2\}.$$

This indicates that particularly eigenvalues with $|\lambda| \ll \|H\|_2$ are affected by the $\sqrt{\mathbf{u}}$-effect. Note that a similar effect occurs when one attempts to compute the singular values of a general matrix A from the eigenvalues of $A^T A$, see e.g. [305, Sec. 3.3.2].

An algorithm that is based on the same idea but achieves numerical backward stability by completely avoiding the squaring of H was developed by Benner, Mehrmann and Xu [42]. First, orthogonal symplectic matrices U and V are computed to reduce H to a so called *symplectic URV form*:

$$U^T H V = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix} = \begin{bmatrix} \diagdown & \square \\ 0 & \diagdown \end{bmatrix}, \tag{4.92}$$

i.e., the matrix $R_{21} \in \mathbb{R}^{n\times n}$ is zero, $R_{11} \in \mathbb{R}^{n\times n}$ is upper triangular and $R_{22} \in \mathbb{R}^{n\times n}$ is lower Hessenberg. A simple calculation reveals

$$U^T H^2 U = \begin{bmatrix} -R_{11}R_{22}^T & R_{11}R_{12}^T - R_{12}R_{11}^T \\ 0 & -R_{22}R_{11}^T \end{bmatrix},$$

showing that the eigenvalues of H are the square roots of the eigenvalues of the upper Hessenberg matrix $-R_{11}R_{22}^T$. In a second step, the periodic QR algorithm, see Section 4.2.3, is applied to compute the eigenvalues of this matrix product in a numerically backward stable manner. The positive and negative square roots of these eigenvalues are the eigenvalues of H. Moreover, this yields orthogonal $n \times n$ matrices Q_1 and Q_2 so that $U \leftarrow U(Q_2 \oplus Q_2)$ and $V \leftarrow V(Q_1 \oplus Q_1)$ reduce H to the same form as displayed in (4.92) but with the additional benefit that R_{22}^T is in real Schur form.

Algorithm 28 can be used to compute a symplectic URV decomposition (4.92). This algorithm is implemented in the HAPACK routines `DGESUV` and `DOSGSU`; it requires $\frac{80}{3}n^3 + \mathcal{O}(n^2)$ floating point operations (flops) to reduce H and additionally $\frac{16}{3}n^3 + \mathcal{O}(n^2)$ flops to compute each of the orthogonal symplectic factors U and V. Note that this algorithm does not assume H to be a Hamiltonian matrix, but even if H is Hamiltonian, this structure will be destroyed. The HAPACK routine `DGESUB` employs a blocked version of Algorithm 28 based on the WY-like representation described in Section 4.3.3.

Let us illustrate the first two loops of Algorithm 28 for the reduction of an 8×8 matrix $H = \left[\begin{smallmatrix} A & G \\ Q & B \end{smallmatrix}\right]$. First, the elementary orthogonal symplectic matrix $E_1(He_1)$ is applied from the left to annihilate the entries $(2:8, 1)$ of H:

Algorithm 28 Symplectic URV decomposition

Input: A matrix $H \in \mathbb{R}^{2n\times 2n}$.
Output: Orthogonal symplectic matrices $U, V \in \mathbb{R}^{2n\times 2n}$; H is overwritten with $U^T HV$ having the form (4.92).

$U \leftarrow I_{2n},\ V \leftarrow I_{2n}$.
for $j \leftarrow 1, 2, \ldots, n$ **do**
  Set $x \leftarrow He_j$.
  Apply Algorithm 23 to compute $E_j(x)$.
  Update $H \leftarrow E_j(x)^T H,\ U \leftarrow UE_j(x)$.
  if $j < n$ **then**
    Set $y \leftarrow H^T e_{n+j}$.
    Apply Algorithm 23 to compute $E_{j+1}(y)$.
    Update $H \leftarrow HE_{n+j+1}(y),\ V \leftarrow VE_{n+j+1}(y)$.
  end if
end for

$$H \leftarrow E_1(He_1)^T H = \left[\begin{array}{cccc|cccc} \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hline \hat{0} & \hat{q} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hat{0} & \hat{q} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hat{0} & \hat{q} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hat{0} & \hat{q} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \end{array}\right].$$

The entries $(5, 2:4)$ and $(5, 7:8)$ are annihilated by applying $E_6(H^T e_5)$ from the right:

$$H \leftarrow HE_6(H^T e_5) = \left[\begin{array}{cccc|cccc} a & \hat{a} & \hat{a} & \hat{a} & g & \hat{g} & \hat{g} & \hat{g} \\ 0 & \hat{a} & \hat{a} & \hat{a} & g & \hat{g} & \hat{g} & \hat{g} \\ 0 & \hat{a} & \hat{a} & \hat{a} & g & \hat{g} & \hat{g} & \hat{g} \\ 0 & \hat{a} & \hat{a} & \hat{a} & g & \hat{g} & \hat{g} & \hat{g} \\ \hline 0 & \hat{0} & \hat{0} & \hat{0} & b & \hat{b} & \hat{0} & \hat{0} \\ 0 & \hat{q} & \hat{q} & \hat{q} & b & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{q} & \hat{q} & \hat{q} & b & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{q} & \hat{q} & \hat{q} & b & \hat{b} & \hat{b} & \hat{b} \end{array}\right].$$

Secondly, the second/sixth rows and columns are reduced by applying $E_2(He_2)$ and $E_7(H^T e_6)$ consecutively:

$$H \leftarrow E_2(He_2)^T H = \left[\begin{array}{cccc|cccc} a & a & a & a & g & g & g & g \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hline 0 & 0 & 0 & 0 & b & b & 0 & 0 \\ 0 & \hat{0} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{0} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{0} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \end{array}\right],$$

$$H \leftarrow HE_7(H^T e_6) = \left[\begin{array}{cccc|cccc} a & a & \hat{a} & \hat{a} & g & g & \hat{g} & \hat{g} \\ 0 & a & \hat{a} & \hat{a} & g & g & \hat{g} & \hat{g} \\ 0 & 0 & \hat{a} & \hat{a} & g & g & \hat{g} & \hat{g} \\ 0 & 0 & \hat{a} & \hat{a} & g & g & \hat{g} & \hat{g} \\ \hline 0 & 0 & 0 & 0 & b & b & 0 & 0 \\ 0 & 0 & \hat{0} & \hat{0} & b & b & \hat{b} & \hat{0} \\ 0 & 0 & \hat{q} & \hat{q} & b & b & \hat{b} & \hat{b} \\ 0 & 0 & \hat{q} & \hat{q} & b & b & \hat{b} & \hat{b} \end{array}\right].$$

The whole procedure, symplectic URV decomposition and periodic QR algorithm, is a numerically backward stable method for computing the eigenvalues of a Hamiltonian matrix H. It preserves the eigenvalue symmetries of H in finite-precision arithmetic and requires less computational effort than the QR algorithm. As the periodic QR algorithm inherits the reliability of the standard QR algorithm, this method can be regarded as highly reliable. Its only drawback is that it does not take full advantage of the structure of H. Furthermore, it is not yet clear whether the method is strongly backward stable or not.

4.5.6 Computation of Invariant Subspaces Based on H^2

Note that the approach explained in the previous section only provides the eigenvalues of a Hamiltonian matrix. Invariant subspaces can be obtained by employing the following relationship between the eigenvalues and invariant subspaces of a matrix and an appropriate embedding of this matrix.

Theorem 4.45 ([41]). *Let $A \in \mathbb{R}^{n\times n}$ and define $B = \begin{bmatrix} 0 & A \\ A & 0 \end{bmatrix}$. Then $\lambda(B) = \lambda(A) \cup (-\lambda(A))$; it is assumed that $\lambda(A) \cap \imath\mathbb{R} = \emptyset$. If the columns of the matrix $[Q_1^T, Q_2^T]^T$ span an invariant subspace of B belonging to eigenvalues in the open right half plane, then the columns of $Q_1 - Q_2$ span an invariant subspace of A belonging to eigenvalues in the open left half plane.*

An orthogonal basis for the subspace spanned by the columns of $Q_1 - Q_2$ can be obtained, e.g., from a rank-revealing QR decomposition [141] of $Q_1 - Q_2$. For general matrices it is of course not advisable to use the above result in order to compute invariant subspaces of the matrix A as it would

unnecessarily double the dimension of the problem. But if A is a Hamiltonian matrix then the results from the previous section can be used to compute invariant subspaces of the extended matrix B which in turn yield invariant subspaces of A.

To see this, let $H \in \mathbb{R}^{2n\times 2n}$ be Hamiltonian with $\lambda(H) \cap \imath\mathbb{R} = \emptyset$. Then we apply Algorithm 28 and the periodic QR algorithm to H. From this we obtain orthogonal symplectic matrices $U = \left[\begin{smallmatrix} U_{11} & U_{12} \\ -U_{12} & U_{22} \end{smallmatrix}\right]$ and $V = \left[\begin{smallmatrix} V_{11} & V_{12} \\ -V_{12} & V_{22} \end{smallmatrix}\right]$ such that

$$R := U^T H V = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}$$

where R_{11} is upper triangular and R_{22}^T is in real Schur form.

Then

$$B := \begin{bmatrix} U^T & 0 \\ 0 & V^T \end{bmatrix} \begin{bmatrix} 0 & H \\ H & 0 \end{bmatrix} \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} = \begin{bmatrix} 0 & R \\ (JRJ)^T & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & R_{11} & R_{12} \\ 0 & 0 & 0 & R_{22} \\ -R_{22}^T & R_{12}^T & 0 & 0 \\ 0 & -R_{11}^T & 0 & 0 \end{bmatrix}.$$

Swapping the middle block rows/columns of B corresponds to $P^T B P$ with a certain permutation matrix P, which transforms B to block upper triangular form.

Now let $W = \left[\begin{smallmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{smallmatrix}\right]$ be orthogonal such that

$$W^T \begin{bmatrix} 0 & R_{11} \\ -R_{22}^T & 0 \end{bmatrix} W = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} =: T \tag{4.93}$$

is quasi upper triangular with all eigenvalues of $T_{11} \in \mathbb{R}^{n\times n}$ and $-T_{22} \in \mathbb{R}^{n\times n}$ located in the open right half plane. Note that this is possible as the eigenvalues of $\left[\begin{smallmatrix} 0 & R_{11} \\ -R_{22}^T & 0 \end{smallmatrix}\right]$ are exactly those of H, and $\lambda(H) \cap \imath\mathbb{R} = \emptyset$. Hence,

$$\tilde{B} := \begin{bmatrix} W^T & 0 \\ 0 & W^T \end{bmatrix} P^T B P \begin{bmatrix} W & 0 \\ 0 & W \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & C_{11} & C_{12} \\ 0 & T_{22} & C_{12}^T & C_{22} \\ 0 & 0 & -T_{11}^T & 0 \\ 0 & 0 & -T_{12}^T & -T_{22}^T \end{bmatrix}. \tag{4.94}$$

This implies that the first n columns of $\left[\begin{smallmatrix} U & 0 \\ 0 & V \end{smallmatrix}\right] P \left[\begin{smallmatrix} W & 0 \\ 0 & W \end{smallmatrix}\right]$ span an invariant subspace of $\left[\begin{smallmatrix} 0 & H \\ H & 0 \end{smallmatrix}\right]$ belonging to the eigenvalues of T_{11}. Thus, as a corollary of Theorem 4.45 we obtain that the columns of

$$\begin{bmatrix} U_{11}W_{11} - V_{11}W_{21} \\ -U_{12}W_{11} + V_{12}W_{21} \end{bmatrix} \tag{4.95}$$

span the invariant subspace of H associated with all stable eigenvalues, i.e., the n eigenvalues in the open left half plane. Computing the matrix W in (4.93)

can be implemented efficiently using the underlying structure; for details see [41]. The number of flops needed by the overall algorithm is approximately 60% of the number of flops the standard QR algorithm would require to compute the invariant subspace under consideration [41]. The HAPACK routine `DHASUB` is based on the described algorithm. It should be emphasized that this algorithm might encounter numerical difficulties if H has eigenvalues close to the imaginary axis, i.e., it is *not* guaranteed to be *backward stable* for such cases. Recently, Chu, Liu, and Mehrmann [92] refined the symplectic URV approach further and presented a *strongly* backward stable algorithm, again under the assumption that H has no eigenvalues on the imaginary axis.

The described procedure admits several extensions, also implemented in `DHASUB`. First, assume that not the invariant subspace belonging to all stable eigenvalues but the invariant subspace belonging to $k < n$ selected stable eigenvalues is to be computed. This can be achieved by reordering the corresponding eigenvalues in the quasi-upper triangular matrix product $R_{11}R_{22}^T$ to the top left corner, see [202], and restricting all computations to the first k columns of the matrix in (4.95). By setting $k = 1$, eigenvectors for specified real eigenvalues can be computed. Complex eigenvectors cannot be computed directly, but with $k = 2$ and choosing a pair of conjugate complex eigenvalues, eigenvectors can be obtained from the 2-dimensional basis of the corresponding invariant subspace. Second, invariant subspaces of H belonging to eigenvalues in the open right half plane can be computed by a variant of Theorem 4.45 saying that the columns of $Q_1 + Q_2$ span such an invariant subspace. Finally, if the Hamiltonian matrix H has eigenvalues on or close to the imaginary axis then the numerical rank of the matrix in (4.95) is likely to be less than n. In this case, it is preferable to reorder the eigenvalues of the $4n \times 4n$ matrix $\tilde{B}$ in (4.94) such that all eigenvalues in the upper $2n \times 2n$ block are in the open right half plane. This can be achieved, e.g., by the reordering described in Section 4.5.4. It is thus possible to determine an orthogonal symplectic matrix Z such that

$$Z^T \tilde{B} Z = \begin{bmatrix} T_{11} & \tilde{T}_{12} & C_{11} & \tilde{C}_{12} \\ 0 & \tilde{T}_{22} & \tilde{C}_{12}^T & \tilde{C}_{22} \\ 0 & 0 & -T_{11}^T & 0 \\ 0 & 0 & -\tilde{T}_{12}^T & -\tilde{T}_{22}^T \end{bmatrix},$$

where $\tilde{B}$ is the matrix defined in (4.94) and all the eigenvalues of $\tilde{T}_{22}$ lie in the open right half plane. Now define

$$\tilde{Q} := \begin{bmatrix} \tilde{Q}_{11} & \tilde{Q}_{12} \\ \tilde{Q}_{21} & \tilde{Q}_{22} \end{bmatrix} := \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} P \begin{bmatrix} W & 0 \\ 0 & W \end{bmatrix} Z, \tag{4.96}$$

then by construction the columns of the matrix $[\tilde{Q}_{11}^T, \tilde{Q}_{21}^T]^T$ span the invariant subspace for $\begin{bmatrix} 0 & H \\ H & 0 \end{bmatrix}$ associated with all eigenvalues in the open right half plane. Again by an application of Theorem 4.45, we obtain the invariant

subspace of H associated with all stable eigenvalues using a rank-revealing QR decomposition of $\tilde{Q}_{11} - \tilde{Q}_{21}$.

4.5.7 Symplectic Balancing

We have seen in Section 1.4.3 that balancing can be a beneficial pre-processing step for computing eigenvalues of general matrices. Of course, standard balancing can be applied to a Hamiltonian matrix H as well; this, however, would destroy the structure of H and prevent the subsequent use of structure-preserving methods. Therefore, Benner [30] has developed a special-purpose balancing algorithm that is based on symplectic similarity transformations and thus preserves the structure of H. As general balancing, it consists of two stages, which are described in the following

Isolating Eigenvalues

The first stage consists of permuting H in order to isolate eigenvalues. It is tempting to require the applied permutations to be symplectic. This leads to rather complicated block triangular forms, see [30, Eq. (3.10)], indicating that the group of symplectic permutation matrices is too restrictive to obtain useful classifications for $P^T H P$. Instead, we propose to broaden the range of similarity transformations to $\tilde{P}^T H \tilde{P}$, where $\tilde{P} = DP$ is symplectic, $D = \operatorname{diag}\{\pm 1, \ldots, \pm 1\}$ and P is a permutation matrix. These *symplectic generalized permutation matrices* clearly form a group, which can be generated by the following two classes of elementary matrices:

$$P_{ij}^{(d)} = P_{ij} \oplus P_{ij}, \tag{4.97}$$

where $1 \le i < j \le n$, $P_{ij} \in \mathbb{R}^{n \times n}$ defined as in (1.49), and

$$P_i^{(s)} = I_{2n} - \begin{bmatrix} e_i & e_{i+n} \end{bmatrix} \begin{bmatrix} e_i^T \\ e_{i+n}^T \end{bmatrix} + \begin{bmatrix} e_i & -e_{i+n} \end{bmatrix} \begin{bmatrix} e_{i+n}^T \\ e_i^T \end{bmatrix}, \tag{4.98}$$

where $1 \le i < n$. If H is post-multiplied by $P_{ij}^{(d)}$ then columns $i \leftrightarrow j$ and columns $(n+i) \leftrightarrow (n+j)$ of H are swapped. A post-multiplication by $P_i^{(s)}$ swaps columns $i \leftrightarrow (n+i)$ and scales the ith column by -1. Analogous statements hold for the rows of H if this matrix is pre-multiplied by $P_{ij}^{(d)}$ or $P_i^{(s)}$.

Combinations of these matrices can be used to compute a symplectic generalized permutation matrix $\tilde{P}$ so that

$$\tilde{P}^T H \tilde{P} = \begin{bmatrix} A_{11} & A_{21} & G_{11} & G_{12} \\ 0 & A_{22} & G_{12}^T & G_{22} \\ 0 & 0 & -A_{11}^T & 0 \\ 0 & Q_{22} & -A_{21}^T & -A_{22}^T \end{bmatrix} = \begin{bmatrix} \diagdown & \square & \square & \square \\ 0 & \square & \square & \square \\ 0 & 0 & \diagdown & 0 \\ 0 & \square & \square & \square \end{bmatrix}, \tag{4.99}$$

where $A_{11} \in \mathbb{R}^{i_l \times i_l}$ is an upper triangular matrix. The unreduced Hamiltonian submatrix $\begin{bmatrix} A_{22} & G_{22} \\ Q_{22} & A_{22}^T \end{bmatrix}$ is characterized by the property that all columns have at least one nonzero off-diagonal element. An algorithm that produces the block triangular form (4.99) can be developed along the lines of Algorithm 4 for isolating eigenvalues of general matrices, see [30, 202] for more details.

Scaling

The second stage of symplectic balancing consists of finding a diagonal matrix D so that

$$(D \oplus D^{-1})^{-1} \begin{bmatrix} A_{22} & G_{22} \\ Q_{22} & -A_{22}^T \end{bmatrix} (D \oplus D^{-1}) = \begin{bmatrix} D^{-1}A_{22}D & D^{-1}G_{22}D^{-1} \\ DQ_{22}D & -(D^{-1}A_{22}D)^T \end{bmatrix}$$

is nearly balanced in 1-norm, i.e., the rows and columns of this matrix are nearly equal in 1-norm.

An iterative procedure achieving this aim has been developed in [30], in the spirit of the Parlett-Reinsch algorithm for equilibrating the row and column norms of a general matrix, see Algorithm 5. It converges if there is no restriction on the diagonal entries of D and under the assumption that $\begin{bmatrix} A_{22} & G_{22} \\ Q_{22} & A_{22}^T \end{bmatrix}$ is irreducible. Note that, strictly speaking, this assumption is not satisfied by the submatrices of the block triangular form (4.99). A structure-preserving block triangular form yielding irreducible Hamiltonian submatrices has been presented in [36]. The construction of this form, however, requires graph-theoretic tools that are more suitable for large and sparse matrices.

Algorithm 29 is basically HAPACK's implementation of the symplectic scaling procedure described in [30]. To avoid any roundoff error in this algorithm, the scaling factor β should be a power of the machine base (usually 2). By exploiting the fact that the 1-norm of the ith column {row} of H is equal to the 1-norm of the $(n+i)$th row {column} for $1 \le i \le n$, Algorithm 29 only needs to balance the first n rows and columns of H. It can thus be concluded that it requires about half the number of operations required by the Parlett-Reinsch algorithm applied to H.

Both ingredients of symplectic balancing, permuting and scaling, are implemented in the HAPACK routine `DHABAL`. The information contained in the generalized symplectic permutation matrix $\tilde{P}$ and the symplectic scaling matrix $\tilde{D}$ is stored in a vector "scal" of length n as follows. If $j \in [1, i_l - 1]$, then the permutation $P^{(d)}_{j,\mathrm{scal}(j)}$ (if $\mathrm{scal}(j) \le n$) or the symplectic generalized permutation $P^{(d)}_{i_l,j} P^{(s)}_{i_l}$ (if $\mathrm{scal}(j) > n$) has been applied in the course of permuting H to the form (4.99). Otherwise, $\mathrm{scal}(j)$ contains $\tilde{d}_{jj}$, the jth diagonal entry of the diagonal matrix $\tilde{D}$ returned by Algorithm 29.

The backward transformation, i.e., multiplication with $(P\tilde{D})^{-1}$, is implemented in the HAPACK routine `DHABAK`. A slight modification of the described procedure can be used for balancing skew-Hamiltonian matrices, see [30].

Algorithm 29 Symplectic Scaling

Input: A Hamiltonian matrix $H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix} \in \mathbb{R}^{2n \times 2n}$ having the block triangular form (4.99) for an integer i_l. A scaling factor $\beta \in \mathbb{R}$.

Output: A symplectic diagonal matrix $\tilde{D} = I_{i_l-1} \oplus D \oplus I_{i_l-1} \oplus D^{-1}$, with diagonal entries that are powers of β, so that $D^{-1}HD$ is nearly balanced in 1-norm. The matrix H is overwritten by $\tilde{D}^{-1}H\tilde{D}$.

$\tilde{D} \leftarrow I_n$
converged $\leftarrow 0$
while converged $= 0$ **do**
  converged $\leftarrow 1$
  for $j \leftarrow i_l, \ldots, n$ **do**
    $c \leftarrow \sum_{\substack{i=i_l \\ i \neq j}}^{n} (|a_{ij}| + |q_{ij}|)$, $r \leftarrow \sum_{\substack{k=i_l \\ k \neq j}}^{n} (|a_{jk}| + |g_{jk}|)$, $\delta_q \leftarrow |q_{jj}|$, $\delta_g \leftarrow |g_{jj}|$
    $s \leftarrow c + r$, scal $\leftarrow 1$
    while $((r + \delta_g/\beta)/\beta) \geq ((c + \delta_q\beta)\beta)$ **do**
      $c \leftarrow c\beta$, $r \leftarrow r/\beta$, $\delta_q \leftarrow \delta_q\beta^2$, $\delta_g \leftarrow \delta_g/\beta^2$
      scal $\leftarrow$ scal $\times\, \beta$
    end while
    while $((r + \delta_g\beta)\beta) \leq ((c + \delta_q/\beta)/\beta)$ **do**
      $c \leftarrow c/\beta$, $r \leftarrow r\beta$, $\delta_q \leftarrow \delta_q/\beta^2$, $\delta_g \leftarrow \delta_g\beta^2$
      scal $\leftarrow$ scal$/\beta$
    end while
    % Balance if necessary.
    if scal $\neq 1$ **then**
      converged $\leftarrow 0$, $\tilde{d}_{jj} \leftarrow$ scal $\times\, \tilde{d}_{jj}$, $\tilde{d}_{n+j,n+j} \leftarrow 1/\text{scal} \times \tilde{d}_{n+j,n+j}$
      $A(:, j) \leftarrow$ scal $\times\, A(:, j)$, $A(j, :) \leftarrow 1/\text{scal} \times A(j, :)$
      $G(:, j) \leftarrow 1/\text{scal} \times G(:, j)$, $G(j, :) \leftarrow 1/\text{scal} \times G(j, :)$
      $Q(:, j) \leftarrow$ scal $\times\, Q(:, j)$, $Q(j, :) \leftarrow$ scal $\times\, Q(j, :)$
    end if
  end for
end while

Symplectic balancing a (skew-)Hamiltonian matrix has essentially the same positive effects as balancing a general matrix. Several numerical experiments confirming this statement can be found in [30, 36].

4.5.8 Numerical Experiments

To give a flavor of the numerical behavior of the described algorithms for computing eigenvalues and invariant subspaces of Hamiltonian matrices, we applied them to data from the CAREX benchmark collection by Abels and Benner [1]. This collection is an updated version of [39] and contains several examples of continuous-time algebraic Riccati equations (CAREs) of the form

$$Q + A^T X + XA - XGX = 0.$$

Computing eigenvalues and invariant subspaces of the associated Hamiltonian matrix $H = \begin{bmatrix} A & -G \\ -Q & -A^T \end{bmatrix}$ plays a fundamental role in most algorithms for solving CAREs; see, e.g., Section A.3 and [29, 240, 285].

Accuracy of eigenvalues

For comparing the accuracy of the computed eigenvalues, we considered the QR algorithm with balancing (`QR`), the square-reduced method with symplectic balancing (`SQRED`), as well as the symplectic URV approach described in Section 4.5.5 combined with symplectic balancing (`DHAESU`).

Table 4.1 gives an account on the eigenvalue accuracy for all Hamiltonian matrices from the benchmark collection [1]. The following quantities are displayed:

$$\text{fwd} = \max_i \frac{|\hat{\lambda}_i - \lambda_i|}{\|H\|_2}, \quad \text{bwd} = \max_i \frac{\sigma_{\min}(H - \hat{\lambda}_i I_{2n})}{\|H\|_2},$$

where $\hat{\lambda}_i$ denotes the computed approximation to the exact eigenvalue λ_i of H. The quantity fwd can be considered as the maximal forward error while bwd represents the maximal backward error of the computed eigenvalues. All computations have been performed in MATLAB 6.1, partly using the mex interfaces mentioned in Section B.4.2. The "exact" eigenvalues λ_i have been computed in variable precision arithmetic (64 decimal digits) as provided by the Symbolic Math Toolbox in MATLAB. Note, however, that this toolbox failed to deliver "exact" eigenvalues for Examples 4.2 and 4.4, where it returned with an error message.

In general, the structured perturbation analysis from Section 4.5.2 predicts that structure preservation will not lead to a higher accuracy in the eigenvalues. Though not explained by this analysis, Examples 1.1 and 2.4 of Table 4.1 show that the structure-preserving algorithms may return significantly more accurate eigenvalues. The known possible loss of accuracy of the square-reduced method can be observed in Examples 2.2 and 4.4. It is remarkable that the square-reduced method displays its numerical backward instability only for Example 4.4, where the measured backward error 2.9×10^{-13} is significantly larger than the machine precision. A simple matrix leading to an even larger backward error can be constructed by setting

$$H = U^T \begin{bmatrix} A & 0 \\ 0 & -A^T \end{bmatrix} U, \quad A = \text{diag}(1, 10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}), \tag{4.100}$$

where U is a random orthogonal symplectic matrix obtained from the symplectic QR decomposition of a random matrix, see also [31]. The errors obtained for this matrix are displayed in the last row of Table 4.1.

Ex	QR		SQRED		DHAESU	
1.1	2.8×10^{-16}	(1.0×10^{-08})	0.0	(0.0)	0.0	(0.0)
1.2	7.4×10^{-17}	(6.9×10^{-18})	7.4×10^{-17}	(6.9×10^{-18})	7.1×10^{-17}	(1.0×10^{-16})
1.3	6.6×10^{-16}	(1.3×10^{-15})	8.8×10^{-17}	(1.9×10^{-16})	2.5×10^{-16}	(4.2×10^{-16})
1.4	1.1×10^{-15}	(1.1×10^{-15})	3.3×10^{-16}	(2.3×10^{-16})	1.5×10^{-15}	(1.4×10^{-15})
1.5	2.1×10^{-16}	(2.5×10^{-15})	6.3×10^{-16}	(1.8×10^{-15})	6.7×10^{-17}	(8.0×10^{-16})
1.6	4.1×10^{-20}	(9.6×10^{-20})	2.1×10^{-20}	(2.9×10^{-19})	3.3×10^{-20}	(6.8×10^{-21})
2.1	1.0×10^{-16}	(1.5×10^{-17})	1.0×10^{-16}	(1.5×10^{-17})	1.1×10^{-16}	(6.0×10^{-17})
2.2	2.4×10^{-18}	(1.3×10^{-17})	9.5×10^{-16}	(9.6×10^{-14})	2.4×10^{-18}	(5.9×10^{-18})
2.3	4.5×10^{-19}	(9.8×10^{-19})	1.6×10^{-19}	(6.5×10^{-20})	2.3×10^{-19}	(8.0×10^{-20})
2.4	7.2×10^{-16}	(3.3×10^{-11})	1.9×10^{-16}	(6.4×10^{-10})	2.1×10^{-16}	(2.0×10^{-16})
2.5	7.8×10^{-17}	(2.4×10^{-09})	2.5×10^{-17}	(1.6×10^{-09})	7.8×10^{-17}	(1.9×10^{-09})
2.6	7.2×10^{-16}	(8.7×10^{-16})	1.3×10^{-16}	(5.4×10^{-17})	3.2×10^{-16}	(2.2×10^{-16})
2.7	1.4×10^{-22}	(7.0×10^{-22})	3.7×10^{-21}	(2.7×10^{-20})	1.4×10^{-22}	(9.4×10^{-22})
2.8	3.4×10^{-16}	(5.2×10^{-16})	3.4×10^{-16}	(3.3×10^{-16})	9.2×10^{-17}	(6.3×10^{-17})
2.9	9.1×10^{-23}	(1.8×10^{-23})	5.6×10^{-23}	(1.3×10^{-20})	2.8×10^{-23}	(5.4×10^{-23})
3.1	3.0×10^{-16}	(7.4×10^{-16})	1.3×10^{-16}	(8.4×10^{-16})	1.5×10^{-16}	(6.1×10^{-16})
3.2	2.4×10^{-15}	(2.4×10^{-15})	2.7×10^{-15}	(2.7×10^{-15})	3.4×10^{-15}	(3.4×10^{-15})
4.1	2.3×10^{-15}	(2.4×10^{-15})	8.6×10^{-16}	(9.2×10^{-16})	1.2×10^{-15}	(1.3×10^{-15})
4.2	9.5×10^{-15}		3.1×10^{-15}		4.9×10^{-15}	
4.3	1.2×10^{-15}	(8.4×10^{-15})	9.3×10^{-16}	(7.6×10^{-15})	9.1×10^{-16}	(1.7×10^{-15})
4.4	4.5×10^{-20}		3.4×10^{-16}		6.2×10^{-20}	
(4.100)	5.9×10^{-16}	(1.3×10^{-16})	1.1×10^{-09}	(1.1×10^{-09})	1.6×10^{-16}	(1.3×10^{-16})

Table 4.1. Backward (and forward) errors of the eigenvalues computed by the QR algorithm, the square-reduced method, and the HAPACK routine `DHAESU`.

Accuracy of invariant subspaces

To test the accuracy of the stable invariant subspaces computed by the HAPACK routine `DHASUB`, see Section 4.5.6, we repeated the experiment from the previous section with this routine and measured the relative residual

$$\text{res} = \|H\hat{X} - \hat{X}(\hat{X}^T H\hat{X})\|_F / \|H\|_F,$$

where the columns of $\hat{X}$ form the computed orthonormal basis for the stable invariant subspace. The parameter `METH` in `DHASUB` can be used to control the choice of method for computing the stable invariant subspace. If `METH = 'S'`, an orthonormal basis for this subspace is obtained by applying a QR decomposition to the $2n \times n$ matrix in (4.95). If `METH = 'L'`, this basis is obtained by applying a rank-revealing QR decomposition to the $2n \times 2n$ matrix $\tilde{Q}_{11} - \tilde{Q}_{21}$ from (4.95). The results in Table 4.2 suggest that `METH = 'S'` often behaves like a numerically backward stable method, except for Examples 2.4, 2.6, and 2.9, while the computationally more expensive choice `METH = 'L'` seems to be numerically backward stable in general. Note that Example 4.4 represents a highly unbalanced Hamiltonian matrix, for which the QR algorithm fails to converge [36]. Also, `DHASUB` encounters convergence problems,

Ex 1.1	Ex 1.2	Ex 1.3	Ex 1.4	Ex 1.5	Ex 1.6	Ex 2.1
1.8×10^{-16}	9.3×10^{-17}	3.8×10^{-15}	1.7×10^{-15}	2.8×10^{-16}	2.5×10^{-16}	1.4×10^{-16}
3.2×10^{-16}	3.7×10^{-16}	3.0×10^{-16}	4.4×10^{-16}	4.7×10^{-16}	5.8×10^{-16}	2.9×10^{-16}
Ex 2.2	Ex 2.3	Ex 2.4	Ex 2.5	Ex 2.6	Ex 2.7	Ex 2.8
1.1×10^{-16}	6.1×10^{-17}	4.5×10^{-02}	6.7×10^{-17}	1.6×10^{-04}	1.8×10^{-17}	5.1×10^{-16}
6.6×10^{-16}	2.4×10^{-16}	6.5×10^{-16}	2.5×10^{-16}	5.4×10^{-16}	7.3×10^{-16}	2.9×10^{-16}
Ex 2.9	Ex 3.1	Ex 3.2	Ex 4.1	Ex 4.2	Ex 4.3	Ex 4.4
1.1×10^{-10}	5.0×10^{-16}	4.2×10^{-15}	1.5×10^{-15}	7.8×10^{-16}	4.8×10^{-15}	8.9×10^{-14}
1.0×10^{-15}	7.2×10^{-16}	1.1×10^{-15}	6.1×10^{-16}	9.8×10^{-16}	9.1×10^{-16}	–

Table 4.2. Relative residuals of invariant subspaces computed by `DHASUB` with `METH = 'S'` (upper row) and `METH = 'L'` (lower row).

which could only be avoided when symplectic balancing (`BALANC = 'B'`) was combined with `METH = 'S'`. For all other examples, symplectic balancing had no significant positive effect on the numerical behavior of `DHASUB`.

Computing Times

To compare the required computing times, we have considered the following Fortran 77 implementations:

LAPACK – routines `DGEHRD`, `DORGHR` and `DHSEQR` for computing the eigenvalues via a Schur decomposition, followed by `DTRSEN` for reordering the stable eigenvalues to the top;

HAPACK – routines `DGESUB`, `DOSGSU` and `DHGPQR` for computing the eigenvalues via a symplectic URV/periodic Schur decomposition, followed by `DHASUB` for extracting the stable invariant subspace;

SQRED – routines `DHASRD` and `DHAEVS` from [31] for computing the eigenvalues via the square reduced method.

All parameters of the block algorithms implemented in LAPACK and HAPACK were set to their default values, i.e., the following values for the blocksize n_b, the crossover point n_x and the number of simultaneous shifts n_s were used:

	DGEHRD	DORGHR	DHSEQR	DGESUB	DOSGSU	DHGPQR
n_b	32	32	–	16	16	–
n_x	128	128	50	64	64	50
n_s	–	–	6	–	–	6

The obtained execution times are displayed in Figure 4.10. One may conclude that if only eigenvalues are to be computed then both SQRED and HAPACK require substantially less time than LAPACK. If, however, the stable invariant subspace is of concern then HAPACK requires up to 60% more time than LAPACK.

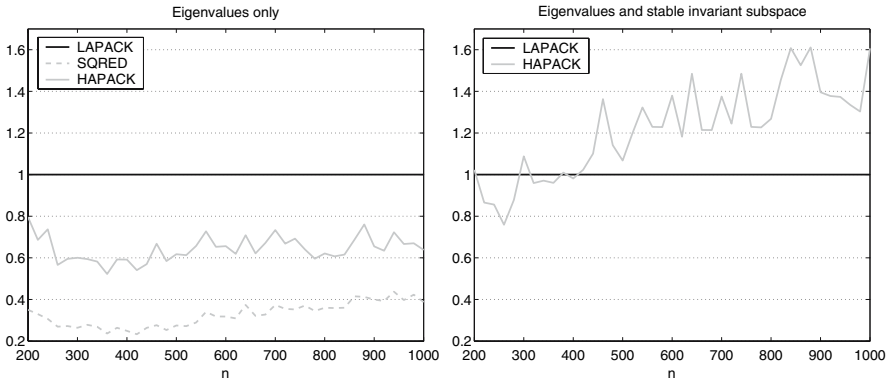


Fig. 4.10. Computational times of SQRED and HAPACK for computing eigenvalues and invariant subspaces of $2n \times 2n$ Hamiltonian matrices ($n = 200, 220, \ldots, 1000$), relative to the computational time of LAPACK.

4.5.9 Other Algorithms and Extensions

There is a vast number of algorithms for solving Hamiltonian eigenvalue problems available. Algorithms based on orthogonal transformations include the Hamiltonian Jacobi algorithm [80, 70], its variants for Hamiltonian matrices that have additional structure [124] and the multishift algorithm [5]. Algorithms based on symplectic but non-orthogonal transformations include the SR algorithm [72, 67, 240] and related methods [73, 268]. A completely different class of algorithms is based on the matrix sign function, see, e.g., [29, 240, 285] and the references therein. Newton-like methods directed towards the computation or refinement of stable invariant subspaces for Hamiltonian matrices can be found in [2, 35, 151, 191, 207, 240, 241].

If H is Hamiltonian then H^2 and $(H - \sigma I)^{-1}(H + \sigma I)^{-1}$ with $\sigma \in \mathbb{R}$ are both skew-Hamiltonian. This makes it possible to use SHIRA, see Section 4.4.5, as a symmetry-preserving (shift-and-invert) Arnoldi method for Hamiltonian matrices, see [242, 202]. There are several variants of symplectic Lanczos processes for Hamiltonian matrices available, see [34, 38, 126, 355].

One suitable generalization of a Hamiltonian matrix is a Hamiltonian/skew-Hamiltonian matrix pair (H, W), i.e., H is a Hamiltonian matrix while W is a skew-Hamiltonian matrix. This matrix pair is equivalent to the symmetric/skew-symmetric matrix pair (JH, JW). Condensed and canonical forms for such matrix pairs have been derived in [84, 237, 238, 320], a perturbation analysis can be found in [58], while structure-preserving algorithms for computing eigenvalues and deflating subspaces of (H, W) have been developed in [32, 239].

Another suitable generalization is to consider a matrix pair (A, B) to be Hamiltonian if $BJA^T = -AJB^T$. A structured eigenvalue solver for this kind of matrix pairs has been proposed in [42].

4.6 A Bouquet of Other Structures

In this section, we peek into some other structures. The treatment is necessarily rather incomplete; a detailed discussion of all the issues related to these structures would give rise to one or more further books.

4.6.1 Symmetric Matrices

Probably the two most fundamental properties of a symmetric matrix A are that every eigenvalue is real and every right eigenvector is also a left eigenvector belonging to the same eigenvalue. Both facts immediately follow from the observation that a Schur decomposition of A always takes the form

$$Q^T A Q = \operatorname{diag}(\lambda_1, \ldots, \lambda_n).$$

It is simple to show that the structured eigenvalue and invariant subspace condition numbers are equal to the corresponding unstructured condition numbers, i.e.,

$$c_2^{\text{symm}}(\lambda) = c(\lambda) = 1$$

and

$$c^{\text{symm}}(\mathcal{X}) = c(\mathcal{X}) = \frac{1}{\min\{|\mu - \lambda| : \ \lambda \in \Lambda_1, \mu \in \Lambda_2\}},$$

where $\mathcal{X}$ is a simple invariant subspace belonging to an eigenvalue subset $\Lambda_1 \subset \lambda(A)$, and $\Lambda_2 = \lambda(A) \backslash \Lambda_1$. Moreover, the QR and the Arnoldi algorithm automatically preserve symmetric matrices.

These facts should not lead to the wrong conclusion that the preservation of symmetric matrices is not important. Algorithms tailored to symmetric matrices (e.g., divide and conquer or Lanczos methods) take much less computational effort and sometimes achieve high relative accuracy in the eigenvalues and – having the right representation of A at hand – even in the eigenvectors. However, these topics are far beyond the scope of this book; we refer to [94, 103, 113, 261] for introductions to this flourishing branch of eigenvalue computation.

4.6.2 Skew-symmetric Matrices

Any eigenvalue of a skew-symmetric matrix $A \in \mathbb{R}^{n \times n}$ is purely imaginary. If n is odd then there is at least one zero eigenvalue. As for symmetric matrices, any right eigenvector is also a left eigenvector belonging to the same eigenvalue (this is true for any normal matrix). The real Schur form of A takes the form

$$Q^T A Q = \begin{bmatrix} 0 & \alpha_1 \\ -\alpha_1 & 0 \end{bmatrix} \oplus \cdots \oplus \begin{bmatrix} 0 & \alpha_k \\ -\alpha_k & 0 \end{bmatrix} \oplus 0 \oplus \cdots \oplus 0.$$

for some real scalars $\alpha_1, \ldots, \alpha_k \neq 0$.

While the structured condition number for a nonzero eigenvalue always satisfies $c_2^{\text{skew}}(\lambda) = c(\lambda) = 1$, we have for a simple zero eigenvalue $c_2^{\text{skew}}(0) = 0$ but $c(0) = 1$ [277]. Again, there is nothing to be gained for an invariant subspace $\mathcal{X}$; it is simple to show $c^{\text{skew}}(\mathcal{X}) = c(\mathcal{X})$.

Skew-symmetric matrices have received much less attention than symmetric matrices, probably due to the fact that skew-symmetry plays a less important role in applications. Again, the QR and the Arnoldi algorithm automatically preserve skew-symmetric matrices. Variants of the QR algorithm tailored to skew-symmetric matrices have been discussed in [178, 347]. Jacobi-like algorithms for skew-symmetric matrices and other Lie algebras have been developed and analyzed in [152, 154, 155, 192, 258, 273, 363].

4.6.3 Persymmetric Matrices

A real $2n \times 2n$ matrix A is called *persymmetric* if it is symmetric about the anti-diagonal. E.g., A takes the following form for $n = 2$:

$$A = \left[\begin{array}{cc|cc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{13} \\ \hline a_{31} & a_{12} & a_{22} & a_{12} \\ a_{41} & a_{31} & a_{21} & a_{11} \end{array}\right].$$

If we *additionally* assume A to be *symmetric* then we can write

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & FA_{11}^T F \end{bmatrix},$$

where F denotes the $n \times n$ flip matrix. Note that A is also a *centrosymmetric* matrix, i.e., $A = FAF$. A practically relevant example of such a matrix is the Gramian of a set of frequency exponentials $\{e^{\pm \mathrm{i}\lambda_k t}\}$, which plays a role in the control of mechanical and electric vibrations [272]. Employing the orthogonal matrix $U = \frac{1}{\sqrt{2}} \begin{bmatrix} I & F \\ -F & I \end{bmatrix}$ we have

$$U^T A U = \begin{bmatrix} A_{11} - A_{12}F & 0 \\ 0 & FA_{11}F + FA_{12} \end{bmatrix}, \tag{4.101}$$

where we used the symmetry of A_{11} and the persymmetry of A_{12}. This is a popular trick when dealing with centrosymmetric matrices [361]. (A similar but technically slightly more complicated trick can be used if A has odd dimension.) Thus,

$$\lambda(A) = \lambda(A_{11} - A_{12}F) \cup \lambda(A_{11} + A_{12}F).$$

Perhaps more importantly, if these two eigenvalue sets are disjoint then any eigenvector belonging to $\lambda(A_{11} - A_{12}F)$ takes the form

$$x = U \begin{bmatrix} \tilde{x} \\ 0 \end{bmatrix} = \begin{bmatrix} \tilde{x} \\ -F\tilde{x} \end{bmatrix}$$

for some $\tilde{x} \in \mathbb{R}^n$. This property of x is sometimes called center-skew-symmetry. Analogously, any eigenvector belonging to $\lambda(A_{11}+A_{12}F)$ is center-symmetric. While the structured and unstructured eigenvalue conditon numbers for symmetric persymmetric matrices are the same [188, 277], there can be a significant difference in the invariant subspace condition numbers. With respect to structured perturbations, the separation between $\lambda(A_{11} - A_{12}F)$ and $\lambda(A_{11} + A_{12}F)$, which can be arbitrarily small, does not play any role for invariant subspaces belonging to eigenvalues from one of the two eigenvalue sets [83]. It can thus be important to retain the symmetry structure of the eigenvectors in finite-precision arithmetic. Krylov subspace methods achieve this goal automatically if the starting vector is chosen to be center-symmetric (or center-skew-symmetric). This property has been used by Voss [340] to construct a structure-preserving Lanczos process. Efficient algorithms for performing matrix-vector products with centrosymmetric matrices have been investigated in [123, 245, 251].

We now briefly consider a persymmetric and *skew-symmetric* matrix A,

$$A = \begin{bmatrix} A_{11} & A_{12} \\ -A_{12}^T & FA_{11}^TF \end{bmatrix}.$$

Again, the orthogonal matrix U can be used to reduce A:

$$U^TAU = \begin{bmatrix} 0 & A_{11}F + A_{12} \\ FA_{11} - A_{12}^T & 0 \end{bmatrix}.$$

Hence, the eigenvalues of A are the positive and negative square roots of the eigenvalues of the matrix product $(A_{11}F + A_{12})(FA_{11} - A_{12}^T)$, see also [272].

Structure-preserving Jacobi algorithms for symmetric persymmetric and skew-symmetric persymmetric matrices have been recently developed by Mackey, Mackey, and Dunlavy [226].

4.6.4 Orthogonal Matrices

All the eigenvalues of a real orthogonal matrix are on the unit circle. Moreover, as orthogonality implies normality, any right eigenvector is also a left eigenvector belonging to the same eigenvalue.

The set of orthogonal matrices $\mathcal{M} \equiv \text{orth} = \{A : \ A^TA = I\}$ forms a smooth manifold, and the tangent space of $\mathcal{M}$ at A is given by

$$T_A\mathcal{M} = \{AH : \ H = -H^T\} = \{AH : \ H \in \text{skew}\}.$$

By Theorem 4.5,

$$\begin{aligned} c_2^{\text{orth}}(\lambda) &= \sup\{|x^HAHx| : \ H \in \text{skew}, \|AH\|_2 = 1\} \\ &= \sup\{|x^HHx| : \ H \in \text{skew}, \|H\|_2 = 1\}. \end{aligned}$$

If $\lambda = \pm 1$ then x can be chosen to be real which implies $x^T H x = 0$ for all x and consequently $c_2^{\text{orth}}(\lambda) = 0$, provided that λ is simple. If λ is complex, we decompose $x = x_R + \imath x_I$ with real vectors x_R, x_I. The fact that $\bar{x}$ is orthogonal to x yields $\|x_R\|_2 = \|x_I\|_2 = 1/\sqrt{2}$ and $x_R^T x_I = 0$. Consequently, the two nonzero singular values of the skew-symmetric matrix $H = 2(x_R x_I^T - x_I x_R^T)$ are both one, and hence $\|H\|_2 = 1$. Moreover, $|x^H H x| = 4(\|x_R\|_2^2 \, \|x_I\|_2^2) = 1$, which shows $c_2^{\text{orth}}(\lambda) \geq 1$ and hence $c_2^{\text{orth}}(\lambda) = 1$. A more general perturbation analysis of orthogonal and unitary eigenvalue problems, based on the Cayley transform, can be found in [51].

Orthogonal eigenvalue problems have a number of applications in digital signal processing, see [6, 7] for an overview.

Once again, the QR algorithm automatically preserves orthogonal matrices. To make it work (efficiently), it is important to take the fact that the underlying matrix is orthogonal into account. A careful choice of shifts can lead to cubic convergence or even ensure global convergence [118, 342, 343]. Even better, an orthogonal (or unitary) Hessenberg matrix can be represented by $\mathcal{O}(n)$ so called *Schur parameters* [143, 71]. This makes it possible to implement the QR algorithm very efficiently [144]; for an extension to unitary and orthogonal matrix pairs, see [68]. Gragg and Reichel [145] proposed a divide and conquer approach for unitary Hessenberg matrices based on the methodology of Cuppen's [95] divide and conquer approach for symmetric tridiagonal matrices. Krylov subspace methods for orthogonal and unitary matrices have been developed and analyzed in [50, 69, 173, 174].

4.6.5 Palindromic Matrix Pairs

Finally, we consider *palindromic matrix pairs* $(A, -A^T)$ with $A \in \mathbb{C}^{2n \times 2n}$ to demonstrate how the concept of structured condition numbers can be extended to deflating subspaces. Such matrix pairs arise from certain linearizations of palindromic polynomials [166, 227]. The following result yields a structured Schur form for palindromic matrix pairs.

Lemma 4.46 ([166]). *Let $A \in \mathbb{C}^{2n \times 2n}$, then there exists a unitary matrix $U \in \mathbb{C}^{2n \times 2n}$ such that*

$$U^T A U = \begin{bmatrix} 0 & \cdots & 0 & t_{1,2n} \\ \vdots & \iddots & t_{2,2n-1} & t_{2,2n} \\ 0 & \iddots & \iddots & \vdots \\ t_{2n,1} & t_{2n,2} & \cdots & t_{2n,2n} \end{bmatrix} =: T,$$

i.e., T is anti-triangular.

Note that U^T denotes the complex transpose of U, i.e., $U^T A U$ is *not* similar to A. Nevertheless, $(T, -T^T)$ is equivalent to $(A, -A^T)$ implying that the generalized eigenvalues of $(A, -A^T)$ are given by

$$(t_{1,2n}, -t_{2n,1}), \ldots, (t_{n,n+1}, -t_{n+1,n}), (t_{n+1,n}, -t_{n,n+1}), \ldots, (t_{2n,1}, -t_{1,2n}).$$

It follows immediately that the eigenvalues have the following pairing: (α, β) is an eigenvalue of $(A, -A^T)$ if and only if (β, α) eigenvalue.

In the following, we consider the (right) deflating subspace $\mathcal{X}$ belonging to the eigenvalues $(t_{n+1,n}, -t_{n,n+1}), \ldots, (t_{2n,1}, -t_{1,2n})$. We assume $\mathcal{X}$ to be simple which is equivalent to requiring that none of the eigenvalues has the form (α, α) or $(\alpha, -\alpha)$. Let the columns of X and $X_\perp$ form orthonormal bases for $\mathcal{X}$ and $\mathcal{X}^\perp$, respectively. Then Lemma 4.46 implies a structured generalized block Schur decomposition of the form

$$[X_\perp,\ X]^T(A, -A^T)[X, X_\perp] = \left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, - \begin{bmatrix} A_{22}^T & A_{12}^T \\ 0 & A_{11}^T \end{bmatrix} \right)$$

with $A_{11}, A_{22} \in \mathbb{C}^{n \times n}$. Consider the associated generalized Sylvester operator

$$\mathbf{T}_l : (Q_r, Q_l) \mapsto (A_{22}Q_r + Q_l A_{11}, A_{11}^T Q_r + Q_l A_{22}^T),$$

then the unstructured condition number for $\mathcal{X}$ is determined by $\|\mathbf{T}_l^{-1}\|$, see Section 2.2.2.

Under structured perturbations it is more appropriate to restrict the operator $\mathbf{T}_l$ to the sets

$$\mathcal{N} = \left\{ X^T(E, -E^T)X :\ E \in \mathbb{C}^{2n \times 2n} \right\} = \left\{ (E_{21}, -E_{21}^T) :\ E_{21} \in \mathbb{C}^{n \times n} \right\}$$

and $\mathbf{T}_l^{-1}\mathcal{N} = \mathcal{N}$. Then, similar as in Section 4.1.1, it can be shown that the structured condition number of $\mathcal{X}$ with respect to palindromic perturbations is determined by $\|\mathbf{T}_s^{-1}\|$, where $\mathbf{T}_s := \mathbf{T}\big|_{\mathcal{N} \to \mathcal{N}}$ [83]. Moreover

$$\mathcal{N}^\perp = \left\{ (E_{21}, E_{21}^T) :\ E_{21} \in \mathbb{C}^{n \times n} \right\}$$

and hence $\mathbf{T} : \mathcal{N}^\perp \to \mathcal{N}^\perp$, which implies

$$\|\mathbf{T}^{-1}\| = \max\left\{ \|\mathbf{T}_s^{-1}\|, \|\mathbf{T}_u^{-1}\| \right\}$$

with $\mathbf{T}_u := \mathbf{T}\big|_{\mathcal{N}^\perp \to \mathcal{N}^\perp}$.

The expressions for $\|\mathbf{T}_s^{-1}\|$ and $\|\mathbf{T}_u^{-1}\|$ can be simplified by considering the Sylvester-like operators

$$\mathbf{S}_s : Q \mapsto A_{22}Q + Q^T A_{11}, \quad \mathbf{S}_u : Q \mapsto A_{22}Q - Q^T A_{11}.$$

It is simple to show $\|\mathbf{T}_s^{-1}\| = \|\mathbf{S}_s^{-1}\|/\sqrt{2}$ and $\|\mathbf{T}_u^{-1}\| = \|\mathbf{S}_u^{-1}\|/\sqrt{2}$.

Example 4.47. For $n = 1$, we obtain

$$\|\mathbf{T}_s^{-1}\| = \frac{1}{\sqrt{2}|A_{22} + A_{11}|}, \quad \|\mathbf{T}_u^{-1}\| == \frac{1}{\sqrt{2}|A_{22} - A_{11}|}.$$

Hence, if A_{22}/A_{11} is close to one, the sensitivity of $\mathcal{X}$ with respect to unstructured perturbations can be much larger than with respect to structured perturbations. ◊

The preceeding example suggests that if $(A, -A^T)$ has eigenvalues close to -1 it is important to preserve palindromic structures. There is currently no variant of the QR or QZ algorithm known that fulfills this requirement. However, in [166] it is demonstrated how the Jacobi-like algorithm developed by Mehl [239] for indefinite generalized Hermitian eigenvalue problems can be adapted to palindromic eigenvalue problems.

A

Background in Control Theory

Who controls the past controls the future: who controls the present controls the past. —George Orwell, "1984"

This chapter attempts to give a brief introduction to some concepts of systems and control theory. The presentation is restricted to subjects closely related to this book; either because an algorithm for computing eigenvalues is better understood in a control theoretic setting or such an algorithm can be used for the analysis and design of control systems. Unless otherwise stated, the material presented in this chapter has been compiled from the monographs [148, 265, 285, 329, 368], which should be consulted for proofs and further details.

A.1 Basic Concepts

A continuous-time *linear time-invariant (LTI) system* can be described by a set of matrix differential and algebraic equations

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0, \tag{A.1a}$$

$$y(t) = Cx(t), \tag{A.1b}$$

where $x(t) \in \mathbb{R}^n$ is the vector of *states*, $u(t) \in \mathbb{R}^m$ the vector of *inputs* (or *controls*) and $y(t) \in \mathbb{R}^r$ the vector of *outputs* at time $t \in [0, \infty)$. The system is described by the *state matrix* $A \in \mathbb{R}^{n \times n}$, the *input (control) matrix* $B \in \mathbb{R}^{n \times m}$, and the *output matrix* $C \in \mathbb{R}^{r \times n}$. The two equations (A.1a) and (A.1b) are referred to as *state* and *output equation*, respectively.

In practice, however, the dynamics of a process is described by physical laws that rarely lead to LTI systems. The importance of such systems stems from the fact that many computational tasks related to (A.1) are mathematically well understood and can be solved with reliable numerical algorithms. Often, discretization and linearization techniques are used to construct LTI systems.

Example A.1. Consider the one-dimensional heat equation on a thin wire:

$$\frac{\partial a(t,z)}{\partial t} = \frac{\partial a(t,z)}{\partial z^2}, \quad a(0,z) = a_0(z), \tag{A.2}$$

where $a(t,z)$ denotes the temperature of the wire at time $t \in [0,\infty)$ and location $z \in \Omega := [0,1]$. The function $a_0 : \Omega \to \mathbb{R}$ is the initial temperature distribution. Heating or cooling the ends of the wire corresponds to the boundary conditions

$$a(t,0) = a_l(t), \quad a(t,1) = a_r(t). \tag{A.3}$$

Only the temperature a_m at the center of the wire is measured, i.e.,

$$a_m(t) = a(t,1/2). \tag{A.4}$$

Equations (A.2)–(A.4) constitute one of the simplest examples for the boundary control of a partial differential equation. Let us now partition the wire into N pieces of length $h = 1/N$ and restrict the state variables to the inner end points of these pieces,

$$x(t) = \big[\, a(t,h),\, a(t,2h),\, \ldots,\, a(t,(N-1)h) \,\big]^T .$$

The second derivative in (A.2) is discretized with the central difference quotient,

$$\frac{\partial a(t,z)}{\partial z^2} \approx \frac{a(t,z-h) - 2a(t,z) + a(t,z+h)}{h^2},$$

which yields the approximate state equation

$$\dot{x}(t) = Ax(t) + B \begin{bmatrix} a_l(t) \\ a_r(t) \end{bmatrix}, \tag{A.5}$$

where

$$A = \frac{1}{h^2} \begin{bmatrix} -2 & 1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -2 \end{bmatrix}, \quad B = \frac{1}{h^2} \begin{bmatrix} e_1 & e_{N-1} \end{bmatrix}. \tag{A.6}$$

The approximate output equation is given by

$$a_m(t) \approx y(t) = Cx(t), \tag{A.7}$$

with $C = e^T_{(N-1)/2}$, provided that N is odd. ◊

The tractability of LTI systems owns much to the fact that there is a considerably simple formula describing the state for a given input.

Theorem A.2. *Let the input vector $u : [0,\infty) \to \mathbb{R}^m$ be piecewise continuous. Then the unique solution of the state equation (A.1a) is given by*

$$x(t) = \Phi(u; x_0; t) := e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bu(\tau)\ \mathrm{d}\tau, \tag{A.8}$$

where

$$e^{At} := I_n + At + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + \dots.$$

Plugging (A.8) into the output equation (A.1b) gives

$$y(t) = Ce^{At}x_0 + \int_0^t Ce^{A(t-s)}Bu(s)\ \mathrm{d}s, \tag{A.9}$$

yielding an *input-output map* $u(\cdot) \to y(\cdot)$.

A.1.1 Stability

For the moment, let us consider the homogeneous system

$$\dot{x}(t) = Ax(t), \quad x(0) = x_0. \tag{A.10}$$

Stability is concerned with the long-time behavior of the solution trajectory $x(\cdot)$.

Definition A.3.

1. *The system (A.10) is called* asymptotically stable *if for each x_0 the solution $x(t)$ satisfies $\lim_{t\to\infty} x(t) = 0$.*
2. *If for each x_0 there exists a constant $C > 0$ so that $\|x(t)\| \le C$ for all $t > 0$ then the system is called* stable.
3. *In any other case, the system is called* unstable.

Stability can be completely characterized in terms of the eigenvalues of the state matrix A, which are commonly called the *poles* of the linear system. This can be proven using the following connection between the spectral radius and matrix norms.

Lemma A.4 ([164]). *Let*

$$\rho(A) := \max\{|\lambda| : \lambda \text{ is an eigenvalue of } A\}$$

denote the spectral radius *of A.*

1. *The spectral radius is dominated by any induced matrix norm $\|\cdot\|$, i.e., $\rho(A) \le \|A\|$.*
2. *For each $\epsilon > 0$ and $A \in \mathbb{R}^{n\times n}$ there exists a vector norm $\|\cdot\|$ on $\mathbb{R}^n$ so that the induced matrix norm satisfies $\|A\| \le \rho(A) + \epsilon$.*
3. *Moreover, there is an induced matrix norm satisfying $\|A\| = \rho(A)$ if and only if each eigenvalue λ of A with $|\lambda| = \rho(A)$ has equal algebraic and geometric multiplicities, i.e., λ is semi-simple.*

Applying this lemma to bound the norm of the solution $e^{At}x_0$ yields the following relations.

Theorem A.5.

1. *The system (A.10) is asymptotically stable if and only if* $\lambda(A) \subset \mathbb{C}^-$*, i.e., all eigenvalues of A have negative real part.*
2. *The system (A.10) is stable if and only if* $\lambda(A) \subset \mathbb{C}^- \cup \imath\mathbb{R}$ *and each purely imaginary eigenvalue is semi-simple.*

The definition of stability assumes that the initial condition x_0 may take any value in $\mathbb{R}^n$. If we restrict the possible x_0 to an invariant subspace $\mathcal{U}$ of A then $\mathcal{U}$ is also an invariant subspace of e^{At} and thus $x(t) \in \mathcal{U}$ for all $t \geq 0$. Hence, in this case, the restriction of A restricted to $\mathcal{U}$ rather than A itself truly reflects the stability of such systems. Let us now decompose $\mathbb{R}^n = \mathcal{U}^- \oplus \mathcal{U}^0 \oplus \mathcal{U}^+$, where $\mathcal{U}^-$, $\mathcal{U}^0$ and $\mathcal{U}^+$ are the maximal invariant subspaces of A associated with eigenvalues in $\mathbb{C}^-$, $\imath\mathbb{R}$ and $\mathbb{C}^+$, respectively. Then $x(t) \to 0$ if and only if $x_0 \in \mathcal{U}^-$ and $\|x(t)\| \leq C$ if and only if $x_0 \in \mathcal{U}^- \oplus \mathcal{U}^0$. This is the motivation to call $\mathcal{U}^-$ the *asymptotically stable*, $\mathcal{U}^- \oplus \mathcal{U}^0$ the *stable* and $\mathcal{U}^+$ the *unstable subspace* of (A.10).

A.1.2 Controllability and Observability

Let us now focus on the state equation

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0. \tag{A.11}$$

Such a system is said to be *controllable* if for any initial state $x_0 \in \mathbb{R}^n$ and any prescribed final state $x_f \in \mathbb{R}^n$ there exist a time t_f and a control $u(t)$ so that the solution of (A.11) satisfies $x(t_f) = x_f$. For linear systems it can be assumed w.l.o.g. that $x_f = 0$. Controllability can be completely characterized in terms of algebraic criteria.

Theorem A.6. *The following statements are equivalent:*

1. *the linear system (A.11) is controllable;*
2. $\operatorname{rank}([B, AB, \ldots, A^{n-1}B]) = n$*;*
3. $\operatorname{rank}([A - \lambda I, B]) = n, \ \forall \lambda \in \lambda(A)$*.*

If the system (A.11) is controllable, the *matrix pair* (A, B) is called *controllable*.

Remark A.7. For the single-input case ($m = 1$), the second condition in Theorem A.6 shows that a linear system is controllable if and only if the Krylov subspace $\mathcal{K}_n(A, B)$ equals $\mathbb{R}^n$. This in turn implies the equivalent condition that the Arnoldi method, Algorithm 15, with starting vector $u_1 = B$ does not break down. $\Diamond$

Similar to the notion of stable and unstable subspaces we can partition the space $\mathbb{R}^n$ into a *controllable subspace* $\mathcal{C}$ and an uncontrollable subspace $\mathcal{C}^\perp$. Any initial condition $x_0 \in \mathcal{C}$ is characterized by the fact that there is a control $u(t)$ that drives the solution of (A.11) to zero. For $m = 1$, the relation $\mathcal{K}_n(A, B) = \mathcal{C}$ reveals once again the close relationship between Krylov subspaces and controllability, see also [54].

A matrix pair (C, A) (or an associated linear system) is called *observable* if (A^T, C^T) is controllable. Theorem A.6 applied to (A^T, C^T) yields algebraic criteria for detecting observability.

In some cases, controllability is too much to ask for. If the aim of control is to stabilize a given system it is sufficient that states not belonging to the asymptotically stable subspace $\mathcal{U}^-$ of the homogeneous system $\dot{x}(t) = Ax(t)$ can be driven to zero, i.e., $\mathcal{C} \subseteq \mathcal{U}^0 \cup \mathcal{U}^+$. Any matrix pair (or linear system) satisfying this requirement is called *stabilizable*. Correspondingly, a matrix pair (C, A) (or an associated linear system) is called *detectable* if (A^T, C^T) is stabilizable.

A.1.3 Pole Placement

The control $u(\cdot)$ may be used to influence the behavior of a linear system (A.11). Often, this control is based on information contained in the state vector $x(\cdot)$. A particular choice is the *linear state feedback*

$$u(t) = -Kx(t), \tag{A.12}$$

where $K \in \mathbb{R}^{m \times n}$ is the so called *feedback matrix*. The corresponding *closed-loop system* takes the form

$$\dot{x}(t) = (A - BK)x(t). \tag{A.13}$$

We have already seen that the long-time behavior of a linear system is determined by its poles, i.e., the eigenvalues of its state matrix. It is thus desirable to choose a feedback matrix K so that the eigenvalues of $A - BK$ are moved to a specified region in the complex plane. The following theorem shows that the poles of a controllable system can be allocated to essentially any desired places in the complex plane.

Theorem A.8. *Let Σ denote a set of at most n numbers closed under complex conjugation. For any such Σ, there exists a matrix $K \in \mathbb{R}^{m \times n}$ so that $\lambda(A - BK) = \Sigma$ if and only if the matrix pair (A, B) is controllable.*

In the presence of roundoff errors, pole assignment usually becomes a rather delicate issue, see [329, Sec. 4.1] or the discussion in Section 1.5.4.

A.2 Balanced Truncation Model Reduction

Model reduction aims at approximating an LTI system (A.1) by a reduced LTI system with a similar input-output-behavior but with a much smaller number

$\tilde{n} \ll n$ of states. In the following, we assume (A.1) to be asymptotically stable, i.e., the state matrix A has only eigenvalues in the open left half plane.

Balanced truncation, which is among the most popular model reduction techniques, is closely related to the two Lyapunov equations

$$AP + PA^T = -BB^T, \quad A^TQ + QA = -C^TC, \tag{A.14}$$

see, e.g., [11]. The solutions P and Q, which are unique and symmetric positive definite, are called the *controllability* and *observability Gramians*, respectively. One can also directly compute Cholesky factors R and S with $P = S^TS$ and $Q = R^TR$, see [153, 175].

Balanced truncation proceeds by computing a singular value decomposition

$$RS^T = U\Sigma V^T,$$

where U, V are orthogonal matrices and Σ is a diagonal matrix,

$$\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n),$$

with the so called *Hankel singular values* $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$. Next, the state transformation matrix $T = S^TU\Sigma^{-1/2}$ is applied:

$$\tilde{A} = T^{-1}AT, \quad \tilde{B} = T^{-1}B, \quad \tilde{C} = CT.$$

The transformed system is *balanced*, i.e., the corresponding Gramians $\tilde{P}$ and $\tilde{Q}$ are given by $\tilde{P} = \tilde{Q} = \Sigma$. This balanced system is reduced by maintaining only these parts that belong to non-negligible Hankel singular values. If the Hankel singular value $\sigma_{\tilde{n}+1}$ is considered to be negligible then we partition

$$\tilde{A} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} \tilde{B}_1 \\ \tilde{B}_2 \end{bmatrix}, \quad \tilde{C} = \begin{bmatrix} \tilde{C}_1 & \tilde{C}_2 \end{bmatrix},$$

with $\tilde{A}_{11} \in \mathbb{R}^{\tilde{n}\times\tilde{n}}$, $\tilde{B}_1 \in \mathbb{R}^{\tilde{n}\times m}$ and $\tilde{C}_1 \in \mathbb{R}^{r\times\tilde{n}}$. This gives rise to the reduced system

$$\dot{\tilde{x}}(t) = \tilde{A}_{11}\tilde{x}(t) + \tilde{B}_1u(t), \tag{A.15a}$$

$$y(t) = \tilde{C}_1\tilde{x}(t). \tag{A.15b}$$

It can be shown that the so called H_∞ norm of the difference between the reduced system (A.15) and the original system (A.1) can be bounded from above by twice the sum of the neglected singular values, $2(\sigma_{\tilde{n}+1} + \cdots + \sigma_n)$.

A.3 Linear-Quadratic Optimal Control

The *linear-quadratic optimal control* problem has the following form:

$$\textit{Minimize} \quad J(u(\cdot)) = \frac{1}{2}\int_0^\infty \left(y(t)^T Q y(t) + u(t)^T R u(t)\right)\, \mathrm{d}t \quad \textit{subject to (A.1)},$$

where $Q \in \mathbb{R}^{n\times n}, R \in \mathbb{R}^{m\times m}$ are symmetric matrices, $Q = M^T M$ is positive semidefinite and R is positive definite. Closely related is the continuous-time algebraic Riccati equation (CARE)

$$C^T Q C + XA + A^T X - XGX = 0, \tag{A.16}$$

where $G = BR^{-1}B^T$.

If the matrix pair (A, B) is stabilizable and the matrix pair (A, MB) is detectable, then there exists a unique optimal control $u_\star(\cdot)$ that minimizes $J(u(\cdot))$. This solution can be written as a linear state feedback

$$u_\star(t) = -R^{-1}B^T X_\star x(t),$$

where $X_\star$ is the unique solution of CARE (A.16) that yields an asymptotically stable closed-loop system

$$\dot{x} = (A - GX_\star)x(t).$$

Note that $X_\star$ is symmetric and positive semidefinite.

Solutions of CARE (A.16) can be obtained from certain invariant subspaces of the Hamiltonian matrix

$$H = \begin{bmatrix} A & G \\ -C^T Q C & -A^T \end{bmatrix}.$$

To see this, suppose X is a symmetric solution of (A.16). Then

$$H \begin{bmatrix} I_n & 0 \\ -X & I_n \end{bmatrix} = \begin{bmatrix} I_n & 0 \\ -X & I_n \end{bmatrix} \begin{bmatrix} A - GX & G \\ 0 & -(A - GX)^T \end{bmatrix}.$$

Hence, the columns of $\left[I_n, -X^T\right]^T$ span an invariant subspace of H belonging to the eigenvalues of $A - GX$. This implies that we can solve CAREs by computing invariant subspaces of H. In particular, if we want the solution that stabilizes the closed-loop system, we need the stable invariant subspace. Suppose that a basis of this subspace is given by the columns of $[X_1^T,\ X_2^T]^T$ with $X_1, X_2 \in \mathbb{R}^{n\times n}$ then, under the given assumptions, X_1 is invertible and $X_\star = -X_2 X_1^{-1}$ is the stabilizing solution of (A.16). It should be noted, though, that often the CARE is a detour. In feedback control, the solution of CARE can usually be avoided by working only with invariant subspaces, see [33, 240].

A.4 Distance Problems

Measurement, roundoff, linearization and approximation errors introduce uncertainties in the system matrices defining a linear system (A.1). In order to

guarantee a certain qualitative behavior despite these errors it is of interest to find the smallest perturbation under which a linear system loses a certain desirable property.

A.4.1 Distance to Instability

Finding the norm of the smallest perturbation that makes an asymptotically stable system $\dot{x}(t) = Ax(t)$ unstable amounts to the computation of the *stability radius* of A, which is defined as

$$\gamma(A) := \min\{\|E\|_2 \,:\, \lambda(A+E) \cap \imath\mathbb{R} \neq \emptyset\}.$$

A bisection method for measuring $\gamma(A)$ can be based on the following observation [79]: if $\alpha \geq 0$, then the Hamiltonian matrix

$$H(\alpha) = \begin{bmatrix} A & -\alpha I_n \\ \alpha I_n & -A^T \end{bmatrix}$$

has an eigenvalue on the imaginary axis if and only if $\alpha \geq \gamma(A)$. This suggests a simple bisection algorithm. Start with a lower bound $\beta \geq 0$ and an upper bound $\delta > \gamma(A)$ (an easy-to-compute upper bound is $\|A + A^T\|_F/2$ [332]). Then in each step, set $\alpha := (\beta + \delta)/2$ and compute $\lambda(H(\alpha))$. If there is an eigenvalue on the imaginary axis, choose $\delta = \alpha$, otherwise, set $\beta = \alpha$.

The correct decision whether $H(\alpha)$ has eigenvalues on the imaginary axis is crucial for the success of the bisection method. Byers [79] has shown that if the eigenvalues of $H(\alpha)$ are computed by a strongly backward stable method, then the computed $\gamma(A)$ will be within an $\mathcal{O}(\mathbf{u})\,\|A\|_2$-distance of the exact stability radius. The proof is based on the observation that $-\gamma(A)$ is an eigenvalue of the Hermitian matrix

$$\begin{bmatrix} 0 & -A^T + \imath\omega I \\ -A^T - \imath\omega I & 0 \end{bmatrix},$$

for some $\omega \in \mathbb{R}$, and an application of the Wielandt-Hoffmann theorem [141] to this matrix.

A.4.2 Distance to Uncontrollability

The distance of an controllable matrix pair (A, B) to an uncontrollable one is defined as

$$\rho(A,B) = \min\{\|[E,F]\|_2 \,:\, (A+E, B+F) \text{ is uncontrollable}\}. \qquad \text{(A.17)}$$

The matrix pair $(E, F) = (0, -B)$ is an admissible perturbation, thus the search space can be restricted to perturbations that satisfy $\|[E,F]\|_2 \leq \|B\|_2$, implying that the minimum in (A.17) can actually be attained.

Eising [119] has shown how the multi-parameter optimization problem (A.17) can be reduced to a complex one-parameter optimization problem.

Theorem A.9. *Let $A \in \mathbb{C}^{n\times n}, B \in \mathbb{C}^{n\times m}$ and let $\mu_* \in \mathbb{C}$ be a minimizer of $f(\mu) = \sigma_{\min}([A - \mu I, B])$. Moreover, let u_* and v_* be the corresponding left and right singular vectors. Then with $[E, F] = -f(\mu_\star)u_* v_*^H$ we obtain a perturbation that solves the minimization problem (A.17).*

A number of algorithms for minimizing $\sigma_{\min}([A-\mu I, B])$ have been developed, see [149, 74] for some recent developments in this area, a more comprehensive list of references can be found in [62]. As far as we know, none of these algorithms is capable to produce a reliable estimate of (A.17) within $\mathcal{O}(n^3)$ flops, making them unattractive for certain purposes, such as computing the optimal reducing perturbation for aggressive early deflation, see Section 1.6. Another dissatisfying aspect of Theorem A.9 is that the optimal perturbation $[E, F]$ is generally complex even if the matrices A and B are real.

A cheap alternative has been proposed by Braman, Byers and Mathias [62], which is particularly suitable for small $\rho(A, B)$. In this case, the minimizer $\mu_\star$ nearly makes the matrix $A - \mu_\star I$ singular. Hence, if the eigenvalues of A are sufficiently well conditioned, then $\mu_\star$ will be close to an eigenvalue of A. This suggests to restrict the search for minimizers of $\sigma_{\min}([A - \mu I, B])$ to $\lambda(A)$. Assume that the eigenvalue $\lambda^\star$ of A minimizes $\sigma_{\min}([A - \mu I, B])$ among all $\mu \in \lambda(A)$ and let y be a normalized left eigenvector belonging to $\lambda^\star$. Then the left and right singular vectors $u^\star$ and $v^\star$ used in Theorem A.9 can be approximated by the vectors y and $[0, \tilde{b}^H]^H/\|\tilde{b}\|_2$, where $\tilde{b}^H = y^H B$. The corresponding perturbation that makes (A, B) uncontrollable is given by $[E, F] = -\|\tilde{b}\|_2\,[0, y\tilde{b}^H]$. If A and B are real but $\lambda^\star \in \lambda(A)$ is complex, we can make use of the real perturbation $[E, F] = -[0, Y\tilde{B}]$, where $\tilde{B} = Y^T B$ and the columns of $Y \in \mathbb{R}^{n\times 2}$ form an orthonormal basis for the invariant subspace belonging to $\{\lambda^\star, \bar{\lambda}^\star\}$.

This leads to a simple algorithm for computing an upper bound on $\rho(A, B)$ for real A and B. First, a real Schur decomposition of $A = UTU^T$ is computed. Let us partition $T = \left[\begin{smallmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{smallmatrix}\right]$, where T_{22} is either a real eigenvalue or a 2×2 block containing a pair of complex conjugate eigenvalues, and correspondingly $U^T B = \left[\begin{smallmatrix} \star \\ \tilde{B} \end{smallmatrix}\right]$. Then $\rho(A, B) \le \|\tilde{B}\|_2$. To find the optimal upper bound that can be obtained in this fashion, we must test any other possible T_{22} by reordering other real eigenvalues or complex conjugate pairs of eigenvalues to the bottom right corner of T. If standard reordering techniques as described in Section 1.7 are used, then the described algorithm requires $\mathcal{O}(n^3)$ flops.

B

Software

The history of reliable high quality software for numerical linear algebra started with the book edited by Wilkinson and Reinsch, the Handbook for Automatic Computation*, Vol. 2, Linear Algebra, published in 1971.*
—Gene Golub and Henk A. van der Vorst [140]

B.1 Computational Environment

If not otherwise stated, we have used an IBM Power3 based SMP system for performing the numerical experiments described in this book. The facilitated computer system has four 375 Mhz Power3 Processors and 4 gigabytes of memory. All performance measurements are based on Fortran 77 implementations utilizing BLAS [108, 109, 213] and standard LAPACK [10] routines. We have used the BLAS kernels provided by IBM's machine-specific optimized Fortran library ESSL. All implementations were compiled with the XL Fortran compiler using optimization level 3. Matrices were always stored in an array with leading dimension slightly larger than the number of rows to avoid unnecessary cache conflicts.

B.2 Flop Counts

We count the execution of an elementary operation $+, -, \cdot, /, \sqrt{\cdot}$ as one floating point operation (flop) per execution if the arguments are real numbers. This fairly reflects the amount of computational work required for $+, -$ and $\cdot$; on nowadays processors one execution of these functions usually requires one clock cycle. The same cannot be said about $/$ and $\sqrt{\cdot}$; in our computing environment the division of two random numbers takes 2.1 and the square root of a random number takes 8.3 clock cycles at an average. Consequently, one should count these functions separately. To simplify counting we abstain from doing so, in none of our considered algorithms do divisions and square roots contribute significantly to the computational burden. We do not count operations on integer or boolean variables, except in cases where such operations constitute a major part of the computational work.

Task	Inputs	Flops
$\gamma \leftarrow \alpha + \beta$	$\alpha, \beta \in \mathbb{C}$	2
$\gamma \leftarrow \alpha - \beta$	$\alpha, \beta \in \mathbb{C}$	2
$\gamma \leftarrow \alpha \cdot \beta$	$\alpha, \beta \in \mathbb{C}$	6
$\gamma \leftarrow \alpha \cdot \beta$	$\alpha \in \mathbb{C}, \beta \in \mathbb{R}$	2
$\gamma \leftarrow \alpha / \beta$	$\alpha, \beta \in \mathbb{C}$	9^*
$\gamma \leftarrow \alpha / \beta$	$\alpha \in \mathbb{C}, \beta \in \mathbb{R}$	2

* Using Smith's formula [292].

Table B.1. Flops of elementary functions with complex arguments.

BLAS	Task	Inputs	Flops
`DAXPY`	$y \leftarrow \alpha x + y$	$x, y \in \mathbb{R}^n, \alpha \in \mathbb{R}$	$2n$
`DDOT`	$\alpha \leftarrow x^T y$	$x, y \in \mathbb{R}^n$	$2n - 1^*$
`DNRM2`	$\alpha \leftarrow \|x\|_2$	$x \in \mathbb{R}^n$	$2n - 1^*$
`DROT`	$[\, x,\ y \,] \leftarrow [\, x,\ y \,] \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$	$x, y \in \mathbb{R}^n, c, s \in \mathbb{R}$	$6n$
`DSCAL`	$x \leftarrow \alpha x$	$x \in \mathbb{R}^n, \alpha \in \mathbb{R}$	n
`ZAXPY`	$y \leftarrow \alpha x + y$	$x, y \in \mathbb{C}^n, \alpha \in \mathbb{C}$	$8n$
`ZDOT`	$\alpha \leftarrow x^H y$	$x, y \in \mathbb{C}^n$	$8n - 2^*$
`ZNRM2`	$\alpha \leftarrow \|x\|_2$	$x \in \mathbb{C}^n$	$8n - 2^*$
`ZROT`	$[\, x,\ y \,] \leftarrow [\, x,\ y \,] \begin{bmatrix} c & -\bar{s} \\ s & c \end{bmatrix}$	$x, y \in \mathbb{R}^n, c \in \mathbb{R}, s \in \mathbb{C}$	$18n$
`ZSCAL`	$x \leftarrow \alpha x$	$x \in \mathbb{C}^n, \alpha \in \mathbb{C}$	$6n$

* Depending on the implementation the flop count may be slightly higher to avoid under-/overflow and subtractive cancellation.

Table B.2. Flops of level 1 basic linear algebra subroutines (BLAS).

B.3 Software for Standard and Generalized Eigenvalue Problems

The state-of-the-art production code implementations of the QR and QZ algorithms can be found in the LAPACK software library [10]. These implementations are doomed to get replaced by variants of the QR and QZ algorithms that employ the multishift and aggressive early deflation strategies described Chapters 1 and 2, see [104] for the prospective developments of LAPACK. The MATLAB functions `eig`, `schur`, and `qz` are all based on the corresponding LAPACK routines.

ARPACK [216] is among the most widely used software packages for computing a few eigenvalues and eigenvectors of a large, possibly sparse matrix (pair); it represents a comprehensive implementation of the implicitly restarted Arnoldi algorithm described in Chapter 3. The MATLAB function `eigs` is based on ARPACK.

BLAS	Task	Inputs*	Flops
DGEMV	$y \leftarrow \alpha Ax + \beta y$	$x \in \mathbb{R}^n, y \in \mathbb{R}^m$	$2mn + m + n$
DGER	$A \leftarrow A + \alpha xy^T$	$x \in \mathbb{R}^m, y \in \mathbb{R}^n$	$2mn + n$
DSYMV	$y \leftarrow \alpha Sx + \beta y$	$x \in \mathbb{R}^n, y \in \mathbb{R}^n$	$2n^2 + 4n$
DSYR	$S \leftarrow \alpha S + xx^T$	$x \in \mathbb{R}^n, \alpha \in \mathbb{R}$	$n^2 + 2n$
DSYR2	$S \leftarrow S + \alpha xy^T + \alpha yx^T$	$x, y \in \mathbb{R}^n$	$2n^2 + 4n$
DTRMV	$x \leftarrow Tx$	$x \in \mathbb{R}^n$	n^2
DTRMV	$x \leftarrow Ux$	$x \in \mathbb{R}^n$	$n^2 - n$
DTRSV	$x \leftarrow T^{-1}x$	$x \in \mathbb{R}^n$	n^2
DTRSV	$x \leftarrow U^{-1}x$	$x \in \mathbb{R}^n$	$n^2 - n$
ZGEMV	$y \leftarrow \alpha Ax + \beta y$	$x \in \mathbb{C}^n, y \in \mathbb{C}^m$	$8mn + 6(m + n)$
ZGERC	$A \leftarrow A + \alpha xy^H$	$x \in \mathbb{C}^m, y \in \mathbb{C}^n$	$8mn + 6n$
ZHEMV	$y \leftarrow \alpha Hx + \beta y$	$x \in \mathbb{C}^n, y \in \mathbb{C}^n$	$8n^2 + 16n$
ZHER	$H \leftarrow \alpha H + xx^H$	$x \in \mathbb{C}^n$	$4n^2 + 5n$
ZHER2	$H \leftarrow H + \alpha xy^H + \bar{\alpha} yx^H$	$x, y \in \mathbb{C}^n$	$8n^2 + 19n$
ZTRMV	$x \leftarrow Tx$	$x \in \mathbb{C}^n$	$4n^2 + 2n$
ZTRMV	$x \leftarrow Ux$	$x \in \mathbb{C}^n$	$4n^2 - 4n$
ZTRSV	$x \leftarrow T^{-1}x$	$x \in \mathbb{C}^n$	$4n^2 + 5n$
ZTRSV	$x \leftarrow U^{-1}x$	$x \in \mathbb{C}^n$	$4n^2 - 4n$

* In all subroutines, $A \in \mathbb{K}^{m \times n}$, $H \in \mathbb{C}^{n \times n}$ Hermitian, $S \in \mathbb{R}^{n \times n}$ symmetric, $T \in \mathbb{K}^{n \times n}$ upper triangular, $U \in \mathbb{K}^{n \times n}$ unit upper triangular and $\alpha, \beta \in \mathbb{K}$.

Table B.3. Flops of level 2 BLAS.

BLAS	Task	Inputs*	Flops
DGEMM	$C \leftarrow \alpha AB + \beta C$	$A \in \mathbb{R}^{m \times k}, B \in \mathbb{R}^{k \times n}$	$2kmn + mn$
DSYMM	$C \leftarrow \alpha AB + \beta C$	$A \in \mathbb{R}^{m \times m}, A = A^T, B \in \mathbb{R}^{m \times n}$	$2m^2n + 3mn$
DTRMM	$C \leftarrow \alpha TC$		$m^2n + mn$
DTRMM	$C \leftarrow \alpha UC$		m^2n

* In all subroutines, $C \in \mathbb{K}^{m \times n}$, $H \in \mathbb{C}^{n \times n}$ Hermitian, $S \in \mathbb{R}^{n \times n}$ symmetric, $T \in \mathbb{K}^{m \times m}$ upper triangular, $U \in \mathbb{K}^{m \times m}$ unit upper triangular and $\alpha, \beta \in \mathbb{K}$.

Table B.4. Flops of selected level 3 BLAS.

LAPACK	Task	Flops*
DLARF	Multiplies a Householder matrix $(I - \beta vv^T) \in \mathbb{R}^{m \times m}$ with a general $m \times n$ matrix.	$4mn$
DLARFG	Generates $v \in \mathbb{R}^m$, $\beta \in \mathbb{R}$ defining a Householder matrix $H_1(x) = I - \beta vv^T$ for a given vector $x \in \mathbb{R}^m$.	$3m$
DLARFX	Multiplies a Householder matrix $(I - \beta vv^T) \in \mathbb{R}^{m \times m}$ with a general $m \times n$ matrix. Inline code is used for $m < 11$.	$4mn - n$

* Only high order terms are displayed.

Table B.5. Flops of a few LAPACK subroutines.

ARPACK and LAPACK are both available online from Netlib (`http://www.netlib.org`.

B.4 Software for Structured Eigenvalue Problems

In the following, we list a few software packages for structured eigenvalue problems; a more comprehensive and up-to-date list can be obtained from the web page of this book.

B.4.1 Product Eigenvalue Problems

There exist several implementations of the periodic QR and QZ algorithms for solving real product eigenvalue problems [40, 197, 224]. An implementation of the periodic QZ algorithm for complex periodic eigenvalue problems is available online from `http:/www.math.tu-berlin.de/~kressner/periodic/`.

B.4.2 Hamiltonian and Skew-Hamiltonian Eigenvalue Problems

HAPACK [37] is a collection of Fortran 77 routines aimed at computing eigenvalues and invariant subspaces of skew-Hamiltonian or Hamiltonian matrices. All routines satisfy the SLICOT [40] implementation and documentation standards [366] and most routines come with test routines as well as example input and output data. Also available are a couple of MEX files and MATLAB functions providing user-friendly access to the most important features of HAPACK. In the following, we only give a brief summary of the available routines. Interested users are referred to the HAPACK web page `http://www.tu-chemnitz.de/mathematik/hapack/` for further information.

Driver routines

`DHAESU` Computes the eigenvalues and the symplectic URV/periodic Schur decomposition of a Hamiltonian matrix.

`DHASUB` Computes stable and unstable invariant subspaces of a Hamiltonian matrix from the output of `DHAESU`.

`DSHES` Computes the skew-Hamiltonian Schur decomposition of a skew-Hamiltonian matrix.

`DSHEVX` Computes the eigenvalues and eigenvectors of a skew-Hamiltonian matrix, with preliminary balancing of the matrix, and computes reciprocal condition numbers for the eigenvalues and some eigenvectors.

Computational routines

`DGESQB` Symplectic QR decomposition of a general matrix. Blocked version.

`DGESQR` Symplectic QR decomposition of a general matrix. Unblocked version.

`DGESUB` Symplectic URV decomposition of a general matrix. Blocked version.

`DGESUV` Symplectic URV decomposition of a general matrix. Unblocked version.

`DHABAK` Applies the inverse of a balancing transformation, computed by the routines `DHABAL` or `DSHBAL`.

`DHABAL` Symplectic balancing of a Hamiltonian matrix.

`DHAORD` Reorders the (skew-)Hamiltonian Schur decomposition of a (skew-) Hamiltonian matrix.

`DHAPVB` PVL decomposition of a Hamiltonian matrix. Blocked version.

`DHAPVL` PVL decomposition of a Hamiltonian matrix. Unblocked version.

`DHGPQR` Periodic Schur decomposition of a product of two matrices.

`DOSGPV` Generates the orthogonal symplectic matrix U from a PVL decomposition determined by `DHAPVL` or `DSHPVL`.

`DOSGSB` Generates all or part of the orthogonal symplectic matrix Q from a symplectic QR decomposition determined by `DGESQB` or `DGESQR`. Blocked version.

`DOSGSQ` Generates all or part of the orthogonal symplectic matrix Q from a symplectic QR decomposition determined by `DGEQRB` or `DGEQRS`. Unblocked version.

`DOSGSU` Generates the orthogonal symplectic matrices U and V from a symplectic URV decomposition determined by `DGESUB` or `DGESUV`.

`DOSMPV` Applies the orthogonal symplectic matrix U from a PVL decomposition determined by `DHAPVL` or `DSHPVL` to a general matrix.

`DOSMSB` Applies all or part of the orthogonal symplectic matrix Q from a symplectic QR decomposition determined by `DGESQB` or `DGESQR` to a general matrix. Blocked version.

`DOSMSQ` Applies all or part of the orthogonal symplectic matrix Q from a symplectic QR decomposition determined by `DGESQB` or `DGESQR` to a general matrix. Unblocked version.

`DSHBAL` Symplectic balancing of a skew-Hamiltonian matrix.
`DSHEVC` Eigenvectors of a skew-Hamiltonian matrix in skew-Hamiltonian Schur form.
`DSHPVB` PVL reduction of a skew-Hamiltonian matrix. Blocked version.
`DSHPVL` PVL reduction of a skew-Hamiltonian matrix. Unblocked version.
`DSHSNA` Computes reciprocal condition numbers for the eigenvalues and some eigenvectors of a skew-Hamiltonian matrix in skew-Hamiltonian Schur form.

MATLAB functions for Hamiltonian matrices

`habalance.m` Symplectic scaling to improve eigenvalue accuracy.
`haconv.m` Converts a Hamiltonian matrix between various data representations.
`haeig.m` Eigenvalues of a Hamiltonian matrix.
`hapvl.m` PVL decomposition of a Hamiltonian matrix.
`haschord.m` Reorders Schur form of a Hamiltonian matrix.
`hastab.m` Complete stable/unstable invariant subspaces of a Hamiltonian matrix.
`hasub.m` Selected stable/unstable invariant subspaces of a Hamiltonian matrix.
`haurv.m` Symplectic URV decomposition of a general matrix.
`haurvps.m` Symplectic URV/periodic Schur decomposition of a general matrix.

MATLAB functions for skew-Hamiltonian matrices

`shbalance.m` Symplectic scaling to improve eigenvalue accuracy.
`shcondeig.m` Structured condition numbers for eigenvalues and eigenvectors of a skew-Hamiltonian matrix.
`shconv.m` Converts a skew-Hamiltonian matrix between various data representations.
`sheig.m` Eigenvalues and eigenvectors of a skew-Hamiltonian matrix.
`shpvl.m` PVL decomposition of a skew-Hamiltonian matrix.
`shschord.m` Reorders Schur form of a skew-Hamiltonian matrix.
`shschur.m` Skew-Hamiltonian Schur form of a skew-Hamiltonian matrix.
`shsep.m` Structured condition number for an isotropic invariant subspace of a skew-Hamiltonian matrix.

B.4.3 Other Structures

The ACM Collected Algorithms (CALGO) (`http://www.netlib.org/toms`) is a repository of software associated with papers published in the Transactions on Mathematical Software. It contains the following implementations related to structured eigenvalue problems.

530: Eigenvalues and eigenvectors of real skew-symmetric matrices [347].
730: Eigenvalues and eigenvectors of unitary matrices based on a divide and conquer approach [9].
800: Eigenvalues of Hamiltonian matrices using the square-reduced method [31].

References

1. J. Abels and P. Benner. CAREX - a collection of benchmark examples for continuous-time algebraic Riccati equations (version 2.0). SLICOT working note 1999-14, WGS, 1999. Available online from `http://www.win.tue.nl/niconet/`.
2. P.-A. Absil and P. Van Dooren. Two-sided Grassmann Rayleigh quotient iteration, 2002. Submitted.
3. B. Adlerborn, K. Dackland, and B. Kågström. Parallel and blocked algorithms for reduction of a regular matrix pair to Hessenberg-triangular and generalized Schur forms. In J. Fagerholm et al., editor, *PARA 2002*, LNCS 2367, pages 319–328. Springer-Verlag, 2002.
4. M. Ahues and F. Tisseur. A new deflation criterion for the QR algorithm. LAPACK Working Note 122, 1997.
5. G. S. Ammar, P. Benner, and V. Mehrmann. A multishift algorithm for the numerical solution of algebraic Riccati equations. *Electr. Trans. Num. Anal.*, 1:33–48, 1993.
6. G. S. Ammar, W. B. Gragg, and L. Reichel. On the eigenproblem for orthogonal matrices. In *Proc. IEEE Confrence on Decision and Control*, pages 1963–1966, 1985.
7. G. S. Ammar, W. B. Gragg, and L. Reichel. Direct and inverse unitary eigenproblems in signal processing: An overview. In M. S. Moonen, G. H. Golub, and B. L. R. De Moor, editors, *Linear algebra for large scale and real-time applications (Leuven, 1992)*, volume 232 of *NATO Adv. Sci. Inst. Ser. E Appl. Sci.*, pages 341–343, Dordrecht, 1993. Kluwer Acad. Publ.
8. G. S. Ammar and V. Mehrmann. On Hamiltonian and symplectic Hessenberg forms. *Linear Algebra Appl.*, 149:55–72, 1991.
9. G. S. Ammar, L. Reichel, and D. C. Sorensen. An implementation of a divide and conquer algorithm for the unitary eigenproblem. *ACM Trans. Math. Software*, 18(3):292–307, 1992.
10. E. Anderson, Z. Bai, C. H. Bischof, S. Blackford, J. W. Demmel, J. J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. C. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, third edition, 1999.
11. A. C. Antoulas. *Lectures on the Approximation of Large-Scale Dynamical Systems*. SIAM, Philadelphia, PA, 2004. To appear.

12. A. C. Antoulas and D. C. Sorensen. Approximation of large-scale dynamical systems: an overview. *Int. J. Appl. Math. Comput. Sci.*, 11(5):1093–1121, 2001.
13. A. C. Antoulas, D. C. Sorensen, and S. Gugercin. A survey of model reduction methods for large-scale systems. In *Structured matrices in mathematics, computer science, and engineering, I (Boulder, CO, 1999)*, volume 280 of *Contemp. Math.*, pages 193–219. Amer. Math. Soc., Providence, RI, 2001.
14. T. Apel, V. Mehrmann, and D. S. Watkins. Structured eigenvalue methods for the computation of corner singularities in 3D anisotropic elastic structures. *Comput. Methods Appl. Mech. Engrg*, 191:4459–4473, 2002.
15. P. Arbenz and Z. Drmač. On positive semidefinite matrices with known null space. *SIAM J. Matrix Anal. Appl.*, 24(1):132–149, 2002.
16. Z. Bai, D. Day, J. W. Demmel, and J. J. Dongarra. A test matrix collection for non-Hermitian eigenvalue problems (release 1.0). Technical Report CS-97-355, Department of Computer Science, University of Tennessee, Knoxville, TN, USA, March 1997. Also available online from `http://math.nist.gov/MatrixMarket`.
17. Z. Bai and J. W. Demmel. On a block implementation of the Hessenberg multishift QR iterations. *Internat. J. High Speed Comput.*, 1:97–112, 1989.
18. Z. Bai and J. W. Demmel. On swapping diagonal blocks in real Schur form. *Linear Algebra Appl.*, 186:73–95, 1993.
19. Z. Bai, J. W. Demmel, J. J. Dongarra, A. Ruhe, and H. van der Vorst, editors. *Templates for the Solution of Algebraic Eigenvalue Problems.* Software, Environments, and Tools. SIAM, Philadelphia, PA, 2000.
20. Z. Bai, J. W. Demmel, and A. McKenney. On computing condition numbers for the nonsymmetric eigenproblem. *ACM Trans. Math. Software*, 19(2):202–223, 1993.
21. R. Barrett, M. Berry, T. F. Chan, J. W. Demmel, J. Donato, J. J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* SIAM, Philadelphia, PA, 1994.
22. R. H. Bartels and G. W. Stewart. Algorithm 432: The solution of the matrix equation $AX - BX = C$. *Communications of the ACM*, 8:820–826, 1972.
23. S. Batterson. Convergence of the shifted QR algorithm on 3×3 normal matrices. *Numer. Math.*, 58(4):341–352, 1990.
24. S. Batterson and J. Smillie. The dynamics of Rayleigh quotient iteration. *SIAM J. Numer. Anal.*, 26(3):624–636, 1989.
25. S. Batterson and J. Smillie. Rayleigh quotient iteration for nonsymmetric matrices. *Math. Comp.*, 55:169–178, 1990.
26. H. Baumgärtel. *Endlichdimensionale analytische Störungstheorie.* Akademie-Verlag, Berlin, 1972.
27. C. Beattie, M. Embree, and J. Rossi. Convergence of restarted Krylov subspaces to invariant subspaces. Technical report 01/21, Oxford University Computing Laboratory Numerical Analysis, 2001.
28. C. Beattie, M. Embree, and D. C. Sorensen. Convergence of polynomial restart Krylov methods for eigenvalue computation. Computational and applied mathematics report TR03-08, Rice University, 2003.
29. P. Benner. Computational methods for linear-quadratic optimization. *Supplemento ai Rendiconti del Circolo Matematico di Palermo, Serie II*, No. 58:21–56, 1999.

30. P. Benner. Symplectic balancing of Hamiltonian matrices. *SIAM J. Sci. Comput.*, 22(5):1885–1904, 2000.
31. P. Benner, R. Byers, and E. Barth. Algorithm 800: Fortran 77 subroutines for computing the eigenvalues of Hamiltonian matrices I: The square-reduced method. *ACM Trans. Math. Software*, 26:49–77, 2000.
32. P. Benner, R. Byers, V. Mehrmann, and H. Xu. Numerical computation of deflating subspaces of skew-Hamiltonian/Hamiltonian pencils. *SIAM J. Matrix Anal. Appl.*, 24(1), 2002.
33. P. Benner, R. Byers, V. Mehrmann, and H. Xu. Robust numerical methods for robust control. Technical Report 06-2004, Institut für Mathematik, TU Berlin, 2004.
34. P. Benner and H. Faßbender. An implicitly restarted symplectic Lanczos method for the Hamiltonian eigenvalue problem. *Linear Algebra Appl.*, 263:75–111, 1997.
35. P. Benner and H. Faßbender. A hybrid method for the numerical solution of discrete-time algebraic Riccati equations. *Contemporary Mathematics*, 280:255–269, 2001.
36. P. Benner and D. Kressner. Balancing sparse Hamiltonian eigenproblems, 2003. To appear in *Linear Algebra Appl.*
37. P. Benner and D. Kressner. Fortran 77 subroutines for computing the eigenvalues of Hamiltonian matrices II. Submitted. See also `http://www.tu-chemnitz.de/mathematik/hapack/`, 2004.
38. P. Benner, D. Kressner, and V. Mehrmann. Skew-Hamiltonian and Hamiltonian eigenvalue problems: Theory, algorithms and applications. In Z. Drmač, M. Marušić, and Z. Tutek, editors, *Proceedings of the Conference on Applied Mathematics and Scientific Computing, Brijuni (Croatia), June 23-27, 2003*, pages 3–39. Springer-Verlag, 2005.
39. P. Benner, A. J. Laub, and V. Mehrmann. Benchmarks for the numerical solution of algebraic Riccati equations. *IEEE Control Systems Magazine*, 7(5):18–28, 1997.
40. P. Benner, V. Mehrmann, V. Sima, S. Van Huffel, and A. Varga. SLICOT—a subroutine library in systems and control theory. In *Applied and computational control, signals, and circuits, Vol. 1*, pages 499–539. Birkhäuser Boston, Boston, MA, 1999.
41. P. Benner, V. Mehrmann, and H. Xu. A new method for computing the stable invariant subspace of a real Hamiltonian matrix. *J. Comput. Appl. Math.*, 86:17–43, 1997.
42. P. Benner, V. Mehrmann, and H. Xu. A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils. *Numer. Math.*, 78(3):329–358, 1998.
43. P. Benner, V. Mehrmann, and H. Xu. Perturbation analysis for the eigenvalue problem of a formal product of matrices. *BIT*, 42(1):1–43, 2002.
44. A. Berman and R. J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*, volume 9 of *Classics in Applied Mathematics*. SIAM, Philadelphia, PA, 1994. Revised reprint of the 1979 original.
45. R. Bhatia. *Matrix Analysis*. Springer-Verlag, New York, 1997.
46. C. H. Bischof, B. Lang, and X. Sun. A framework for symmetric band reduction. *ACM Trans. Math. Software*, 26(4):581–601, 2000.
47. C. H. Bischof and C. F. Van Loan. The *WY* representation for products of Householder matrices. *SIAM J. Sci. Statist. Comput.*, 8(1):S2–S13, 1987.

48. Å. Björck. *Numerical Methods for Least Squares Problems.* SIAM, Philadelphia, PA, 1996.
49. L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. W. Demmel, I. Dhillon, J. J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. C. Whaley. *ScaLAPACK Users' Guide.* SIAM, Philadelphia, PA, 1997.
50. B. Bohnhorst. *Beiträge zur numerischen Behandlung des unitären Eigenwertproblems.* PhD thesis, Fakultät für Mathematik, Universität Bielefeld, Bielefeld, Germany, 1993.
51. B. Bohnhorst, A. Bunse-Gerstner, and H. Faßbender. On the perturbation theory for unitary eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 21(3):809–824, 2000.
52. A. Bojanczyk, G. H. Golub, and P. Van Dooren. The periodic Schur decomposition; algorithm and applications. In *Proc. SPIE Conference*, volume 1770, pages 31–42, 1992.
53. A. Bojanczyk and P. Van Dooren. Reordering diagonal blocks in the real Schur form. In M. S. Moonen, G. H. Golub, and B. L. R. De Moor, editors, *Linear algebra for large scale and real-time applications (Leuven, 1992)*, volume 232 of *NATO Adv. Sci. Inst. Ser. E Appl. Sci.*, pages 351–352, Dordrecht, 1993. Kluwer Acad. Publ.
54. D. L. Boley and G. H. Golub. The nonsymmetric Lanczos algorithm and controllability. *Systems Control Lett.*, 16(2):97–105, 1991.
55. W. Bomhof. *Iterative and parallel methods for linear systems, with applications in circuit simulation.* PhD thesis, Universiteit Utrecht, Faculteit der Wiskunde en Informatica, Utrecht, Netherlands, 2001.
56. W. Bomhof and H. van der Vorst. A parallelizable GMRES-type method for p-cyclic matrices, with applications in circuit simulation. In U. Van Rienen, M. Gunther, and D. Hecht, editors, *Scientific computing in electrical engineering: proceedings of the 3rd international workshop, August 20–23, 2000, Warnemünde, Germany*, volume 18 of *LNCSE*, pages 293–300. Springer Verlag, 2001.
57. F. Bonhoure, Y. Dallery, and W. J. Stewart. On the use of periodicity properties for the efficient numerical solution of certain Markov chains. *Numer. Linear Algebra Appl.*, 1(3):265–286, 1994.
58. S. Bora and V. Mehrmann. Linear perturbation theory for structured matrix pencils arising in control theory. Preprint 16-2004, Institut für Mathematik, TU Berlin, 2004.
59. S. Boyd, V. Balakrishnan, and P. Kabamba. A bisection method for computing the $\mathcal{H}_\infty$ norm of a transfer matrix and related problems. *Math. Control, Signals, Sys.*, 2:207–219, 1989.
60. K. Braman. *Toward A Recursive QR Algorithm.* PhD thesis, University of Kansas, 2003.
61. K. Braman, R. Byers, and R. Mathias. The multishift QR algorithm. I. Maintaining well-focused shifts and level 3 performance. *SIAM J. Matrix Anal. Appl.*, 23(4):929–947, 2002.
62. K. Braman, R. Byers, and R. Mathias. The multishift QR algorithm. II. Aggressive early deflation. *SIAM J. Matrix Anal. Appl.*, 23(4):948–973, 2002.
63. K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical solution of initial-value problems in differential-algebraic equations*, volume 14 of *Classics in Applied Mathematics.* SIAM, Philadelphia, PA, 1996. Revised and corrected reprint of the 1989 original.

64. R. Bru, R. Cantó, and B. Ricarte. Modelling nitrogen dynamics in citrus trees. *Mathematical and Computer Modelling*, 38:975–987, 2003.
65. N.A. Bruinsma and M. Steinbuch. A fast algorithm to compute the H_∞-norm of a transfer function matrix. *Sys. Control Lett.*, 14(4):287–293, 1990.
66. J. R. Bunch. The weak and strong stability of algorithms in numerical linear algebra. *Linear Algebra Appl.*, 88/89:49–66, 1987.
67. A. Bunse-Gerstner. Matrix factorizations for symplectic QR-like methods. *Linear Algebra Appl.*, 83:49–77, 1986.
68. A. Bunse-Gerstner and L. Elsner. Schur parameter pencils for the solution of the unitary eigenproblem. *Linear Algebra Appl.*, 154/156:741–778, 1991.
69. A. Bunse-Gerstner and H. Faßbender. Error bounds in the isometric Arnoldi process. *J. Comput. Appl. Math.*, 86(1):53–72, 1997.
70. A. Bunse-Gerstner and H. Faßbender. A Jacobi-like method for solving algebraic Riccati equations on parallel computers. *IEEE Trans. Automat. Control*, 42(8):1071–1084, 1997.
71. A. Bunse-Gerstner and C. He. On a Sturm sequence of polynomials for unitary Hessenberg matrices. *SIAM J. Matrix Anal. Appl.*, 16(4):1043–1055, 1995.
72. A. Bunse-Gerstner and V. Mehrmann. A symplectic QR like algorithm for the solution of the real algebraic Riccati equation. *IEEE Trans. Automat. Control*, 31(12):1104–1113, 1986.
73. A. Bunse-Gerstner, V. Mehrmann, and D. S. Watkins. An SR algorithm for Hamiltonian matrices based on Gaussian elimination. In *XII Symposium on Operations Research (Passau, 1987)*, volume 58 of *Methods Oper. Res.*, pages 339–357. Athenäum/Hain/Hanstein, Königstein, 1989.
74. J. V. Burke, A. S. Lewis, and M. L. Overton. Pseudospectral components and the distance to uncontrollability, 2003. To appear in SIAM J. Matrix Anal. Appl.
75. J. V. Burke, A. S. Lewis, and M. L. Overton. Robust stability and a criss-cross algorithm for pseudospectra. *IMA J. Numer. Anal.*, 23(3):359–375, 2003.
76. R. Byers. *Hamiltonian and Symplectic Algorithms for the Algebraic Riccati Equation*. PhD thesis, Cornell University, Dept. Comp. Sci., Ithaca, NY, 1983.
77. R. Byers. A LINPACK-style condition estimator for the equation $AX-XB^T = C$. *IEEE Trans. Automat. Control*, 29(10):926–928, 1984.
78. R. Byers. A Hamiltonian QR algorithm. *SIAM J. Sci. Statist. Comput.*, 7(1):212–229, 1986.
79. R. Byers. A bisection method for measuring the distance of a stable to unstable matrices. *SIAM J. Sci. Statist. Comput.*, 9:875–881, 1988.
80. R. Byers. A Hamiltonian-Jacobi algorithm. *IEEE Trans. Automat. Control*, 35:566–570, 1990.
81. R. Byers, 2004. Personal communication.
82. R. Byers and D. Kressner. On the condition of a complex eigenvalue under real perturbations. *BIT*, 44(2):209–215, 2004.
83. R. Byers and D. Kressner. Structured condition numbers for invariant subspaces, 2005. In preparation.
84. R. Byers, V. Mehrmann, and H. Xu. A structured staircase algorithm for skew-symmetric/symmetric pencils, 2005. In preparation.
85. R. Byers and S. Nash. On the singular "vectors" of the Lyapunov operator. *SIAM J. Algebraic Discrete Methods*, 8(1):59–66, 1987.

86. R. Byers and N. Rhee. Cyclic Schur and Hessenberg-Schur numerical methods for solving periodic Lyapunov and Sylvester equations. Technical report, Dept. of Mathematics, Univ. of Missouri at Kansas City, 1995.
87. Z. Cao and F. Zhang. Direct methods for ordering eigenvalues of a real matrix (in Chinese). *Chinese Univ. J. Comput. Math.*, 1:27–36, 1981. Cited in [18].
88. Y. Chahlaoui and P. Van Dooren. A collection of benchmark examples for model reduction of linear time invariant dynamical systems. SLICOT working note 2002-2, WGS, 2002.
89. F. Chatelin. *Eigenvalues of matrices*. John Wiley & Sons Ltd., Chichester, 1993.
90. T.-Y. Chen. Balancing sparse matrices for computing eigenvalues. Master's thesis, University of California at Berkeley, 1998.
91. T.-Y. Chen and J. W. Demmel. Balancing sparse matrices for computing eigenvalues. *Linear Algebra Appl.*, 309:261–287, 2000.
92. D. Chu, X. Liu, and V. Mehrmann. A numerically backwards stable method for computing the Hamiltonian Schur form. Preprint 24-2004, Institut für Mathematik, TU Berlin, 2004.
93. K.-W. E. Chu. The solution of the matrix equations $AXB - CXD = E$ and $(YA - DZ, YC - BZ) = (E, F)$. *Linear Algebra Appl.*, 93:93–105, 1987.
94. J. K. Cullum and R. A. Willoughby. *Lanczos algorithms for large symmetric eigenvalue computations. Vol. 1*, volume 41 of *Classics in Applied Mathematics*. SIAM, Philadelphia, PA, 2002. Reprint of the 1985 original.
95. J. J. M. Cuppen. A divide and conquer method for the symmetric tridiagonal eigenproblem. *Numer. Math.*, 36(2):177–195, 1980/81.
96. K. Dackland and B. Kågström. Blocked algorithms and software for reduction of a regular matrix pair to generalized Schur form. *ACM Trans. Math. Software*, 25(4):425–454, 1999.
97. L. Dai. *Singular control systems*, volume 118 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin, 1989.
98. J. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization. *Mathematics of Computation*, 30:772–795, 1976.
99. P. I. Davies. Structured conditioning of matrix functions. *Electron. J. Linear Algebra*, 11:132–161, 2004.
100. D. Day. How the shifted QR algorithm fails to converge and how to fix it. Tech. Report 96–0913, Sandia National Laboratories, Albuquerque, NM, 1996.
101. J. W. Demmel. Computing stable eigendecompositions of matrices. *Linear Algebra Appl.*, 79:163–193, 1986.
102. J. W. Demmel. Three methods for refining estimates of invariant subspaces. *Computing*, 38:43–57, 1987.
103. J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997.
104. J. W. Demmel and J. J. Dongarra. LAPACK 2005 prospectus: Reliable and scalable software for linear algebra computations on high end computers. LAPACK Working Note 165, 2005.
105. J. W. Demmel and B. Kågström. Computing stable eigendecompositions of matrix pencils. *Linear Algebra Appl.*, 88/89:139–186, 1987.
106. J. W. Demmel and B. Kågström. The generalized Schur decomposition of an arbitrary pencil $A - \lambda B$: robust software with error bounds and applications. I. Theory and algorithms. *ACM Trans. Math. Software*, 19(2):160–174, 1993.

107. J. W. Demmel and B. Kågström. The generalized Schur decomposition of an arbitrary pencil $A-\lambda B$: robust software with error bounds and applications. II. Software and applications. *ACM Trans. Math. Software*, 19(2):175–201, 1993.
108. J. J. Dongarra, J. Du Croz, I. S. Duff, and S. Hammarling. A set of level 3 basic linear algebra subprograms. *ACM Trans. Math. Software*, 16:1–17, 1990.
109. J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson. An extended set of fortran basic linear algebra subprograms. *ACM Trans. Math. Software*, 14:1–17, 1988.
110. J. J. Dongarra, S. Hammarling, and J. H. Wilkinson. Numerical considerations in computing invariant subspaces. *SIAM J. Matrix Anal. Appl.*, 13(1):145–161, 1992.
111. J. J. Dongarra, D. C. Sorensen, and S. J. Hammarling. Block reduction of matrices to condensed forms for eigenvalue computations. *J. Comput. Appl. Math.*, 27(1-2):215–227, 1989. Reprinted in *Parallel algorithms for numerical linear algebra*, 215–227, North-Holland, Amsterdam, 1990.
112. P. Donovan and M. R. Freislich. *The representation theory of finite graphs and associated algebras*. Carleton University, Ottawa, Ont., 1973. Carleton Mathematical Lecture Notes, No. 5.
113. Z. Drmač, V. Hari, and I. Slapničar. Advances in Jacobi methods. In *Applied mathematics and scientific computing (Dubrovnik, 2001)*, pages 63–90. Kluwer/Plenum, New York, 2003.
114. A. A. Dubrulle. The multishift QR algorithm – is it worth the trouble? TR 6320-3558, IBM Scientific Center, Palo Alto, CA, 1991.
115. A. A. Dubrulle and G. H. Golub. A multishift QR iteration without computation of the shifts. *Numer. Algorithms*, 7(2-4):173–181, 1994.
116. A. A. Dubrulle, R. S. Martin, and J. H. Wilkinson. The implicit QL algorithm. *Numer. Math.*, 12:377–383, 1968. Also in [365, pp.241–248].
117. B. C. Eaves, A. J. Hoffman, U. G. Rothblum, and H. Schneider. Line-sum-symmetric scalings of square nonnegative matrices. *Math. Programming Stud.*, 25:124–141, 1985.
118. P. J. Eberlein and C. P. Huang. Global convergence of the QR algorithm for unitary matrices with some results for normal matrices. *SIAM J. Numer. Anal.*, 12:97–104, 1975.
119. R. Eising. Between controllable and uncontrollable. *Systems Control Lett.*, 4(5):263–264, 1984.
120. O. G. Ernst. Equivalent iterative methods for p-cyclic matrices. *Numer. Algorithms*, 25(1-4):161–180, 2000.
121. T. F. Fairgrieve and Allan D. Jepson. O. K. Floquet multipliers. *SIAM J. Numer. Anal.*, 28(5):1446–1462, 1991.
122. H. Faßbender. *Symplectic methods for the symplectic eigenproblem*. Kluwer Academic/Plenum Publishers, New York, 2000.
123. H. Faßbender and K. D. Ikramov. Computing matrix-vector products with centrosymmetric and centrohermitian matrices. *Linear Algebra Appl.*, 364:235–241, 2003.
124. H. Faßbender, D. S. Mackey, and N. Mackey. Hamilton and Jacobi come full circle: Jacobi algorithms for structured Hamiltonian eigenproblems. *Linear Algebra Appl.*, 332/334:37–80, 2001.
125. H. Faßbender, D. S. Mackey, N. Mackey, and H. Xu. Hamiltonian square roots of skew-Hamiltonian matrices. *Linear Algebra Appl.*, 287(1-3):125–159, 1999.

126. W. R. Ferng, W.-W. Lin, and C.-S. Wang. The shift-inverted J-Lanczos algorithm for the numerical solutions of large sparse algebraic Riccati equations. *Comput. Math. Appl.*, 33(10):23–40, 1997.
127. L. V. Foster. Gaussian elimination with partial pivoting can fail in practice. *SIAM J. Matrix Anal. Appl.*, 15(4):1354–1362, 1994.
128. J. G. F. Francis. The QR transformation, parts I and II. *Computer Journal*, 4:265–271, 332–345, 1961, 1962.
129. G. Freiling, V. Mehrmann, and H. Xu. Existence, uniqueness, and parametrization of Lagrangian invariant subspaces. *SIAM J. Matrix Anal. Appl.*, 23(4):1045–1069, 2002.
130. R. W. Freund, G. H. Golub, and M. Hochbruck. Krylov subspace methods for non-Hermitian p-cyclic matrices. Technical report, RIACS, NASA Ames Research Center, Moffet Field, CA, 1992. Cited in [131].
131. R. W. Freund, G. H. Golub, and N. M. Nachtigal. Recent advances in Lanczos-based iterative methods for nonsymmetric linear systems. In *Algorithmic trends in computational fluid dynamics (1991)*, ICASE/NASA LaRC Ser., pages 137–162. Springer, New York, 1993.
132. K. A. Gallivan, S. Thirumala, P. Van Dooren, and V. Vermaut. High performance algorithms for Toeplitz and block Toeplitz matrices. *Linear Algebra Appl.*, 241-243:343–388, 1996.
133. F.R. Gantmacher. *The Theory of Matrices.* Chelsea, New York, 1960.
134. J. D. Gardiner. Stabilizing control for second-order models and positive real systems. *AIAA J. Guidance, Dynamics and Control*, 15(1):280–282, 1992.
135. Y. Genin, P. Van Dooren, and V. Vermaut. Convergence of the calculation of H_∞ norms and related questions. In A. Beghi, L. Finesso, and G. Picci, editors, *Proceedings of the Conference on the Mathematical Theory of Networks and Systems, MTNS '98*, pages 429–432, 1998.
136. I. Gohberg, P. Lancaster, and L. Rodman. *Matrix polynomials.* Academic Press Inc. [Harcourt Brace Jovanovich Publishers], New York, 1982. Computer Science and Applied Mathematics.
137. G. H. Golub, P. Milanfar, and J. Varah. A stable numerical method for inverting shape from moments. *SIAM J. Sci. Comput.*, 21(4):1222–1243, 1999/2000.
138. G. H. Golub and C. Reinsch. Singular value decomposition and least squares solution. *Numer. Math.*, 14:403–420, 1970.
139. G. H. Golub, K. Sølna, and P. Van Dooren. Computing the SVD of a general matrix product/quotient. *SIAM J. Matrix Anal. Appl.*, 22(1):1–19, 2000.
140. G. H. Golub and H. van der Vorst. Eigenvalue computation in the 20th century. *J. Comput. Appl. Math.*, 123(1-2):35–65, 2000.
141. G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, MD, third edition, 1996.
142. J. Grad. Matrix balancing. *Comput. J.*, 14:280–284, 1971.
143. W. B. Gragg. Positive definite Toeplitz matrices, the Hessenberg process for isometric operators, and the Gauss quadrature on the unit circle. In *Numerical methods of linear algebra (Russian)*, pages 16–32. Moskov. Gos. Univ., Moscow, 1982.
144. W. B. Gragg. The QR algorithm for unitary Hessenberg matrices. *J. Comput. Appl. Math.*, 16:1–8, 1986.
145. W. B. Gragg and L. Reichel. A divide and conquer method for unitary and orthogonal eigenproblems. *Numer. Math.*, 57(8):695–718, 1990.

146. S. Graillat. Structured condition number and backward error for eigenvalue problems. Research Report RR2005-01, LP2A, University of Perpignan, France, 2005.
147. R. Granat and B. Kågström. Direct eigenvalue reordering in a product of matrices in extended periodic Schur form. Report UMINF-05.05, Department of Computing Science, Umeå University, Umeå, Sweden, 2005.
148. M. Green and D. J. N. Limebeer. *Linear Robust Control*. Prentice-Hall, Englewood Cliffs, NJ, 1995.
149. M. Gu. New methods for estimating the distance to uncontrollability. *SIAM J. Matrix Anal. Appl.*, 21(3):989–1003, 2000.
150. M. Günther and U. Feldmann. The DAE-index in electric circuit simulation. *Math. Comput. Simulation*, 39(5-6):573–582, 1995.
151. C.-H. Guo and P. Lancaster. Analysis and modification of Newton's method for algebraic Riccati equations. *Math. Comp.*, 67:1089–1105, 1998.
152. D. Hacon. Jacobi's method for skew-symmetric matrices. *SIAM J. Matrix Anal. Appl.*, 14(3):619–628, 1993.
153. S. Hammarling. Numerical solution of the stable, nonnegative definite Lyapunov equation. *IMA J. Numer. Anal.*, 2(3):303–323, 1982.
154. V. Hari. On the quadratic convergence of the Paardekooper method. I. *Glas. Mat. Ser. III*, 17(37)(1):183–195, 1982.
155. V. Hari and N. H. Rhee. On the quadratic convergence of the Paardekooper method. II. *Glas. Mat. Ser. III*, 27(47)(2):369–391, 1992.
156. D. J. Hartfiel. Concerning diagonal similarity of irreducible matrices. *Proc. Amer. Math. Soc.*, 30:419–425, 1971.
157. C. He, A. J. Laub, and V. Mehrmann. Placing plenty of poles is pretty preposterous. Preprint SPC 95-17, Forschergruppe 'Scientific Parallel Computing', Fakultät für Mathematik, TU Chemnitz-Zwickau, 1995.
158. J. J. Hench, C. He, V. Kučera, and V. Mehrmann. Dampening controllers via a Riccati equation approach. *IEEE Trans. Automat. Control*, 43(9):1280–1284, 1998.
159. J. J. Hench and A. J. Laub. Numerical solution of the discrete-time periodic Riccati equation. *IEEE Trans. Automat. Control*, 39(6):1197–1210, 1994.
160. H. V. Henderson and S. R. Searle. The vec-permutation matrix, the vec operator and Kronecker products: a review. *Linear and Multilinear Algebra*, 9(4):271–288, 1980/81.
161. G. Henry, D. S. Watkins, and J. J. Dongarra. A parallel implementation of the nonsymmetric QR algorithm for distributed memory architectures. *SIAM J. Sci. Comput.*, 24(1):284–311, 2002.
162. D. J. Higham and N. J. Higham. Backward error and condition of structured linear systems. *SIAM J. Matrix Anal. Appl.*, 13(1):162–175, 1992.
163. D. J. Higham and N. J. Higham. Structured backward error and condition of generalized eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 20(2):493–512, 1999.
164. N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, PA, 1996.
165. N. J. Higham, M. Konstantinov, V. Mehrmann, and P. Petkov. The sensitivity of computational control problems. *IEEE Control Systems Magazine*, 24(1):28–43, 2004.
166. A. Hilliges, C. Mehl, and V. Mehrmann. On the solution of palindromic eigenvalue problems. In *Proceedings of ECCOMAS, Jyväskylä, Finland*, 2004.

167. D. Hinrichsen and A. J. Pritchard. Stability radii of linear systems. *Systems Control Lett.*, 7(1):1–10, 1986.
168. D. Hinrichsen and A. J. Pritchard. *Mathematical Systems Theory I*, volume 48 of *Texts in Applied Mathematics*. Springer-Verlag, 2005.
169. M. E. Hochstenbach and B. Plestenjak. Backward error, condition numbers, and pseudospectra for the multiparameter eigenvalue problem. *Linear Algebra Appl.*, 375:63–81, 2003.
170. R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985.
171. R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, 1991.
172. C. G. J. Jacobi. Über ein leichtes Verfahren die in der Theorie der Säculärstörungen vorkommenden Gleichungen numerisch aufzulösen. *Journal für die reine und angewandte Mathematik*, 30:51–94, 1846. Cited in [327].
173. C. F. Jagels and L. Reichel. The isometric Arnoldi process and an application to iterative solution of large linear systems. In *Iterative methods in linear algebra (Brussels, 1991)*, pages 361–369. North-Holland, Amsterdam, 1992.
174. C. F. Jagels and L. Reichel. A fast minimal residual algorithm for shifted unitary matrices. *Numer. Linear Algebra Appl.*, 1(6):555–570, 1994.
175. I. M. Jaimoukha and E. M. Kasenally. Krylov subspace methods for solving large Lyapunov equations. *SIAM J. Numer. Anal.*, 31:227–251, 1994.
176. I. Jonsson and B. Kågström. Recursive blocked algorithm for solving triangular systems. I. one-sided and coupled Sylvester-type matrix equations. *ACM Trans. Math. Software*, 28(4):392–415, 2002.
177. I. Jonsson and B. Kågström. Recursive blocked algorithm for solving triangular systems. II. Two-sided and generalized Sylvester and Lyapunov matrix equations. *ACM Trans. Math. Software*, 28(4):416–435, 2002.
178. B. Kågström. The QR algorithm to find the eigensystem of a skew-symmetric matrix. Report UMINF-14.71, Institute of Information Processing, University of Umeå, Sweden, 1971.
179. B. Kågström. A direct method for reordering eigenvalues in the generalized real Schur form of a regular matrix pair (A, B). In M. S. Moonen, G. H. Golub, and B. L. R. De Moor, editors, *Linear algebra for large scale and real-time applications (Leuven, 1992)*, volume 232 of *NATO Adv. Sci. Inst. Ser. E Appl. Sci.*, pages 195–218. Kluwer Acad. Publ., Dordrecht, 1993.
180. B. Kågström and D. Kressner. Multishift variants of the QZ algorithm with aggressive early deflation. Report UMINF-05.11, Department of Computing Science, Umeå University, Umeå, Sweden, 2005.
181. B. Kågström and P. Poromaa. Distributed and shared memory block algorithms for the triangular Sylvester equation with sep^{-1} estimators. *SIAM J. Matrix Anal. Appl.*, 13(1):90–101, 1992.
182. B. Kågström and P. Poromaa. Computing eigenspaces with specified eigenvalues of a regular matrix pair (A, B) and condition estimation: theory, algorithms and software. *Numer. Algorithms*, 12(3-4):369–407, 1996.
183. B. Kågström and P. Poromaa. LAPACK-style algorithms and software for solving the generalized Sylvester equation and estimating the separation between regular matrix pairs. *ACM Trans. Math. Software*, 22(1):78–103, 1996.
184. B. Kågström and L. Westin. Generalized Schur methods with condition estimators for solving the generalized Sylvester equation. *IEEE Trans. Automat. Control*, 34(7):745–751, 1989.

185. B. Kalantari, L. Khachiyan, and A. Shokoufandeh. On the complexity of matrix balancing. *SIAM J. Matrix Anal. Appl.*, 18(2):450–463, 1997.
186. M. Karow. *Geometry of Spectral Value Sets*. PhD thesis, Universität Bremen, Fachbereich 3 (Mathematik & Informatik), Bremen, Germany, 2003.
187. M. Karow. Structured Hölder condition numbers for multiple eigenvalues, 2005. In preparation.
188. M. Karow, D. Kressner, and F. Tisseur. Structured eigenvalue condition numbers, 2005. Submitted.
189. L. Kaufman. Some thoughts on the QZ algorithm for solving the generalized eigenvalue problem. *ACM Trans. Math. Software*, 3(1):65–75, 1977.
190. L. Kaufman. A parallel QR algorithm for the symmetric tridiagonal eigenvalue problem. *Journal of Parallel and Distributed Computing*, Vol 3:429–434, 1994.
191. D. L. Kleinman. On an iterative technique for Riccati equation computations. *IEEE Trans. Automat. Control*, AC-13:114–115, 1968.
192. M. Kleinsteuber, U. Helmke, and K. Hüper. Jacobi's algorithm on compact Lie algebras. *SIAM J. Matrix Anal. Appl.*, 26(1):42–69, 2004.
193. D. E. Knuth. *The Art of Computer Programming. Volume 3*. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont., 1973. Sorting and searching, Addison-Wesley Series in Computer Science and Information Processing.
194. M. Konstantinov, D.-W. Gu, V. Mehrmann, and P. Petkov. *Perturbation theory for matrix equations*, volume 9 of *Studies in Computational Mathematics*. North-Holland Publishing Co., Amsterdam, 2003.
195. M. Konstantinov, V. Mehrmann, and P. Petkov. Perturbation analysis of Hamiltonian Schur and block-Schur forms. *SIAM J. Matrix Anal. Appl.*, 23(2):387–424, 2001.
196. S. G. Krantz. *Function Theory of Several Complex Variables*. John Wiley & Sons Inc., New York, 1982.
197. D. Kressner. An efficient and reliable implementation of the periodic QZ algorithm. In *IFAC Workshop on Periodic Control Systems*, 2001.
198. D. Kressner. Block algorithms for orthogonal symplectic factorizations. *BIT*, 43(4):775–790, 2003.
199. D. Kressner. The periodic QR algorithm is a disguised QR algorithm, 2003. To appear in *Linear Algebra Appl.*
200. D. Kressner. Perturbation bounds for isotropic invariant subspaces of skew-Hamiltonian matrices, 2003. To appear in *SIAM J. Matrix Anal. Appl.*
201. D. Kressner. A Krylov-Schur algorithm for matrix products, 2004. Submitted.
202. D. Kressner. *Numerical Methods and Software for General and Structured Eigenvalue Problems*. PhD thesis, TU Berlin, Institut für Mathematik, Berlin, Germany, 2004.
203. D. Kressner. Block algorithms for reordering standard and generalized Schur forms, 2005. In preparation.
204. D. Kressner, V. Mehrmann, and T. Penzl. CTDSX - a collection of benchmark examples for state-space realizations of continuous-time dynamical systems. SLICOT working note 1998-9, WGS, 1998.
205. D. Kressner and E. Mengi. Structure-exploiting methods for computing numerical and pseudospectral radii, 2005. Submitted.
206. V. N. Kublanovskaya. On some algorithms for the solution of the complete eigenvalue problem. *Zhurnal Vychislitelnoi Matematiki i Matematicheskoi Fiziki*, 1:555–570, 1961.

207. P. Lancaster and L. Rodman. *The Algebraic Riccati Equation*. Oxford University Press, Oxford, 1995.
208. B. Lang. Effiziente Orthogonaltransformationen bei der Eigen- und Singulärwertzerlegung. Habilitationsschrift, 1997.
209. B. Lang. Using level 3 BLAS in rotation-based algorithms. *SIAM J. Sci. Comput.*, 19(2):626–634, 1998.
210. H. Langer and B. Najman. Leading coefficients of the eigenvalues of perturbed analytic matrix functions. *Integral Equations Operator Theory*, 16(4):600–604, 1993.
211. A. J. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Trans. Automat. Control*, AC-24:913–921, 1979.
212. A. J. Laub. Invariant subspace methods for the numerical solution of Riccati equations. In S. Bittanti, A. J. Laub, and J. C. Willems, editors, *The Riccati Equation*, pages 163–196. Springer-Verlag, Berlin, 1991.
213. C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh. Basic linear algebra subprograms for fortran usage. *ACM Trans. Math. Software*, 5:308–323, 1979.
214. J. M. Lee. *Introduction to smooth manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2003.
215. R. B. Lehoucq and D. C. Sorensen. Deflation techniques for an implicitly restarted Arnoldi iteration. *SIAM J. Matrix Anal. Appl.*, 17(4):789–821, 1996.
216. R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK users' guide*. SIAM, Philadelphia, PA, 1998. Solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods.
217. B. J. Leimkuhler and E. S. Van Vleck. Orthosymplectic integration of linear Hamiltonian systems. *Numer. Math.*, 77(2):269–282, 1997.
218. D. Lemonnier and P. Van Dooren. Balancing regular matrix pencils, 2004. Submitted to *SIAM J. Matrix Anal. Appl.*
219. W.-W. Lin and T.-C. Ho. On Schur type decompositions for Hamiltonian and symplectic pencils. Technical report, Institute of Applied Mathematics, National Tsing Hua University, Taiwan, 1990.
220. W.-W. Lin, V. Mehrmann, and H. Xu. Canonical forms for Hamiltonian and symplectic matrices and pencils. *Linear Algebra Appl.*, 302/303:469–533, 1999.
221. W.-W. Lin and J.-G. Sun. Perturbation analysis for the eigenproblem of periodic matrix pairs. *Linear Algebra Appl.*, 337:157–187, 2001.
222. W.-W. Lin, P. Van Dooren, and Q.-F. Xu. Periodic invariant subspaces in control. In *Proc. of IFAC Workshop on Periodic Control Systems, Como, Italy*, 2001.
223. X.-G. Liu. *Differential expansion theory of matrix functions and its applications (in Chinese)*. PhD thesis, Computing Center, Academia Sinica, Beijing, 1990. Cited in [317].
224. K. Lust. Continuation and bifurcation analysis of periodic solutions of partial differential equations. In *Continuation methods in fluid dynamics (Aussois, 1998)*, pages 191–202. Vieweg, Braunschweig, 2000.
225. K. Lust. Improved numerical Floquet multipliers. *Internat. J. Bifur. Chaos Appl. Sci. Engrg.*, 11(9):2389–2410, 2001.
226. D. S. Mackey, N. Mackey, and D. Dunlavy. Structure preserving algorithms for perplectic eigenproblems. *Electron. J. Linear Algebra*, 13:10–39, 2005.

227. D. S. Mackey, N. Mackey, C. Mehl, and V. Mehrmann. Palindromic polynomial eigenvalue problems: Good vibrations from good linearizations, 2005. Submitted.
228. D. S. Mackey, N. Mackey, and F. Tisseur. Structured tools for structured matrices. *Electron. J. Linear Algebra*, 10:106–145, 2003.
229. D. S. Mackey, N. Mackey, and F. Tisseur. $\mathbb{G}$-reflectors: analogues of Householder transformations in scalar product spaces. *Linear Algebra Appl.*, 385:187–213, 2004.
230. D. S. Mackey, N. Mackey, and F. Tisseur. Structured factorizations in scalar product spaces. Numerical Analysis Report No. 421, Manchester Centre for Computational Mathematics, 2004. To appear in SIAM J. Matrix Anal. Appl.
231. A. N. Malyshev. Balancing, pseudospectrum and inverse free iteration for matrix pencils, June 2004. Presentation given at the *V International Workshop on Accurate Solution of Eigenvalue Problems*, Hagen, Germany.
232. T. A. Manteuffel. Adaptive procedure for estimating parameters for the nonsymmetric Tchebychev iteration. *Numer. Math.*, 31(2):183–208, 1978.
233. R. S. Martin, G. Peters, and J. H. Wilkinson. The QR algorithm for real Hessenberg matrices. *Numer. Math.*, 14:219–231, 1970. Also in [365, pp.359–371].
234. R. März. Canonical projectors for linear differential algebraic equations. *Comput. Math. Appl.*, 31(4-5):121–135, 1996. Selected topics in numerical methods (Miskolc, 1994).
235. The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, Mass, 01760. *The* MATLAB *Control Toolbox, Version 5*, 2000.
236. The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, Mass, 01760. MATLAB *Version 6.5*, 2002.
237. C. Mehl. Condensed forms for skew-Hamiltonian/Hamiltonian pencils. *SIAM J. Matrix Anal. Appl.*, 21(2):454–476, 1999.
238. C. Mehl. Anti-triangular and anti-m-Hessenberg forms for Hermitian matrices and pencils. *Linear Algebra Appl.*, 317(1-3):143–176, 2000.
239. C. Mehl. Jacobi-like algorithms for the indefinite generalized Hermitian eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 25(4):964–985, 2004.
240. V. Mehrmann. *The Autonomous Linear Quadratic Control Problem, Theory and Numerical Solution.* Number 163 in Lecture Notes in Control and Information Sciences. Springer-Verlag, Heidelberg, 1991.
241. V. Mehrmann and E. Tan. Defect correction methods for the solution of algebraic Riccati equations. *IEEE Trans. Automat. Control*, 33(7):695–698, 1988.
242. V. Mehrmann and D. S. Watkins. Structure-preserving methods for computing eigenpairs of large sparse skew-Hamiltonian/Hamiltonian pencils. *SIAM J. Sci. Comput.*, 22(6):1905–1925, 2000.
243. V. Mehrmann and H. Xu. An analysis of the pole placement problem. I. The single-input case. *Electron. Trans. Numer. Anal.*, 4(Sept.):89–105, 1996.
244. V. Mehrmann and H. Xu. Choosing poles so that the single-input pole placement problem is well conditioned. *SIAM J. Matrix Anal. Appl.*, 19(3):664–681, 1998.
245. A. Melman. Symmetric centrosymmetric matrix-vector multiplication. *Linear Algebra Appl.*, 320(1-3):193–198, 2000.
246. R. A. Meyer and C. S. Burrus. A unified analysis of multirate and periodically time-varying digital filters. *IEEE Trans. Circuits and Systems*, CAS-22:162–168, 1975.

247. C. B. Moler. Cleve's corner: MATLAB incorporates LAPACK, 2000. See `http://www.mathworks.com/company/newsletter/win00/index.shtml`.
248. C. B. Moler and G. W. Stewart. An algorithm for generalized matrix eigenvalue problems. *SIAM J. Numer. Anal.*, 10:241–256, 1973.
249. B. C. Moore. Principal component analysis in linear systems: controllability, observability, and model reduction. *IEEE Trans. Automat. Control*, 26(1):17–32, 1981.
250. J. Moro, J. V. Burke, and M. L. Overton. On the Lidskii-Vishik-Lyusternik perturbation theory for eigenvalues of matrices with arbitrary Jordan structure. *SIAM J. Matrix Anal. Appl.*, 18(4):793–817, 1997.
251. Z. J. Mou. Fast algorithms for computing symmetric/Hermitian matrix-vector products. *Electronic Letters*, 27:1272–1274, 1991.
252. F. D. Murnaghan and A. Wintner. A canonical form for real matrices under orthogonal transformations. *Proc. Natl. Acad. Sci. USA*, 17:417–420, 1931.
253. L. A. Nazarova. Representations of quivers of infinite type. *Izv. Akad. Nauk SSSR Ser. Mat.*, 37:752–791, 1973.
254. A. Nemirovski and U. G. Rothblum. On complexity of matrix scaling. *Linear Algebra Appl.*, 302/303:435–460, 1999. Special issue dedicated to Hans Schneider (Madison, WI, 1998).
255. K. C. Ng and B. N. Parlett. Development of an accurate algorithm for EXP(Bt), Part I, Programs to swap diagonal block, Part II. CPAM-294, Univ. of California, Berkeley, 1988. Cited in [18].
256. S. Noschese and L. Pasquini. Eigenvalue condition numbers: zero-structured versus traditional. Preprint 28, Mathematics Department, University of Rome La Sapienza, Italy, 2004.
257. E. E. Osborne. On preconditioning of matrices. *Journal of the ACM*, 7:338–345, 1960.
258. M. H. C. Paardekooper. An eigenvalue algorithm for skew-symmetric matrices. *Numer. Math.*, 17:189–202, 1971.
259. C. C. Paige. Bidiagonalization of matrices and solutions of the linear equations. *SIAM J. Numer. Anal.*, 11:197–209, 1974.
260. C. C. Paige and C. F. Van Loan. A Schur decomposition for Hamiltonian matrices. *Linear Algebra Appl.*, 41:11–32, 1981.
261. B. N. Parlett. *The Symmetric Eigenvalue Problem*, volume 20 of *Classics in Applied Mathematics*. SIAM, Philadelphia, PA, 1998. Corrected reprint of the 1980 original.
262. B. N. Parlett and J. Le. Forward instability of tridiagonal QR. *SIAM J. Matrix Anal. Appl.*, 14(1):279–316, 1993.
263. B. N. Parlett and C. Reinsch. Balancing a matrix for calculation of eigenvalues and eigenvectors. *Numer. Math.*, 13:293–304, 1969. Also in [365, pp.315–326].
264. G. Peters and J. H. Wilkinson. Inverse iteration, ill-conditioned equations and Newton's method. *SIAM Rev.*, 21(3):339–360, 1979.
265. P. H. Petkov, N. D. Christov, and M. M. Konstantinov. *Computational Methods for Linear Control Systems*. Prentice-Hall, Hertfordshire, UK, 1991.
266. G. Quintana-Ortí and R. van de Geijn. Improving the performance of reduction to Hessenberg form. FLAME working note #14, Department of Computer Sciences, The University of Texas at Austin, 2004.
267. P. J. Rabier and W. C. Rheinboldt. *Nonholonomic motion of rigid mechanical systems from a DAE viewpoint*. SIAM, Philadelphia, PA, 2000.

268. A. C. Raines and D. S. Watkins. A class of Hamiltonian–symplectic methods for solving the algebraic Riccati equation. *Linear Algebra Appl.*, 205/206:1045–1060, 1994.
269. A. C. M. Ran and L. Rodman. Stability of invariant maximal semidefinite subspaces. I. *Linear Algebra Appl.*, 62:51–86, 1984.
270. A. C. M. Ran and L. Rodman. Stability of invariant Lagrangian subspaces. I. In *Topics in operator theory*, volume 32 of *Oper. Theory Adv. Appl.*, pages 181–218. Birkhäuser, Basel, 1988.
271. A. C. M. Ran and L. Rodman. Stability of invariant Lagrangian subspaces. II. In *The Gohberg anniversary collection, Vol. I (Calgary, AB, 1988)*, volume 40 of *Oper. Theory Adv. Appl.*, pages 391–425. Birkhäuser, Basel, 1989.
272. R. M. Reid. Some eigenvalue properties of persymmetric matrices. *SIAM Rev.*, 39(2):313–316, 1997.
273. N. H. Rhee and V. Hari. On the cubic convergence of the Paardekooper method. *BIT*, 35(1):116–132, 1995.
274. U. G. Rothblum and H. Schneider. Scalings of matrices which have prespecified row sums and column sums via optimization. *Linear Algebra Appl.*, 114/115:737–764, 1989.
275. A. Ruhe. An algorithm for numerical determination of the structure of a general matrix. *BIT*, 10:196–216, 1969.
276. S. M. Rump. Structured perturbations. I. Normwise distances. *SIAM J. Matrix Anal. Appl.*, 25(1):1–30, 2003.
277. S. M. Rump. Eigenvalues, pseudosprectrum and structured perturbations, 2005. Submitted.
278. H. Rutishauser. Solution of eigenvalue problems with the LR transformation. *Nat. Bur. Stand. App. Math. Ser.*, 49:47–81, 1958.
279. Y. Saad. On the rates of convergence of the Lanczos and the block Lanczos methods. *SIAM Journal on Numerical Analysis*, 17:687–706, 1980.
280. Y. Saad. Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems. *Math. Comp.*, 42(166):567–588, 1984.
281. Y. Saad. *Numerical Methods for Large Eigenvalue Problems: Theory and Algorithms*. John Wiley, New York, 1992.
282. R. Schreiber and C. F. Van Loan. A storage-efficient WY representation for products of Householder transformations. *SIAM J. Sci. Statist. Comput.*, 10(1):53–57, 1989.
283. V. V. Sergeichuk. Canonical matrices for linear matrix problems. *Linear Algebra Appl.*, 317(1-3):53–102, 2000.
284. V. V. Sergeichuk. Computation of canonical matrices for chains and cycles of linear mappings. *Linear Algebra Appl.*, 376:235–263, 2004.
285. V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics*. Marcel Dekker, Inc., New York, NY, 1996.
286. V. Sima. Accurate computation of eigenvalues and real Schur form of 2x2 real matrices. In *Proceedings of the Second NICONET Workshop on "Numerical Control Software: SLICOT, a Useful Tool in Industry", December 3, 1999, INRIA Rocquencourt, France*, pages 81–86, 1999.
287. V. Simoncini. Variable accuracy of matrix-vector products in projection methods for eigencomputation, 2005. To appear in SIAM J. Numer. Anal.
288. V. Simoncini and D. B. Szyld. Theory of inexact Krylov subspace methods and applications to scientific computing. *SIAM J. Sci. Comput.*, 25(2):454–477, 2003.

289. R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Statist.*, 35:876–879, 1964.
290. P. Smit. *Numerical Analysis of Eigenvalue Algorithms Based on Subspace Iterations*. PhD thesis, Catholic University Brabant, The Netherlands, 1997.
291. B. T. Smith, J. M. Boyle, J. J. Dongarra, B. S. Garbow, Y. Ikebe, V. C. Klema, and C. B. Moler. *Matrix Eigensystem Routines–EISPACK Guide. Lecture Notes in Computer Science*. Springer-Verlag, New York, second edition, 1976.
292. R. L. Smith. Algorithm 116: Complex division. *Comm. ACM*, 5(8):435, 1962.
293. M. Sofroniou and G. Spaletta. Symplectic methods for separable Hamiltonian systems. In P.M.A. Sloot et al., editor, *ICCS 2002*, LNCS 2329, pages 506–515. Springer-Verlag, 2002.
294. D. C. Sorensen. Implicit application of polynomial filters in a k-step Arnoldi method. *SIAM J. Matrix Anal. Appl.*, 13:357–385, 1992.
295. D. C. Sorensen. Implicitly restarted Arnoldi/Lanczos methods for large scale eigenvalue calculations. In *Parallel numerical algorithms (Hampton, VA, 1994)*, volume 4 of *ICASE/LaRC Interdiscip. Ser. Sci. Eng.*, pages 119–165. Kluwer Acad. Publ., Dordrecht, 1997.
296. D. C. Sorensen. Passivity preserving model reduction via interpolation of spectral zeros. Technical report TR02-15, ECE-CAAM Depts, Rice University, 2002.
297. J. Sreedhar and P. Van Dooren. A Schur approach for solving some periodic matrix equations. In U. Helmke, R. Mennicken, and J. Saurer, editors, *Systems and Networks : Mathematical Theory and Applications*, volume 77, pages 339–362. Akademie Verlag, Berlin, 1994.
298. J. Stefanovski and K. Trenčevski. Antisymmetric Riccati matrix equation. In *1st Congress of the Mathematicians and Computer Scientists of Macedonia (Ohrid, 1996)*, pages 83–92. Sojuz. Mat. Inform. Maked., Skopje, 1998.
299. G. W. Stewart. Error bounds for approximate invariant subspaces of closed linear operators. *SIAM J. Numer. Anal.*, 8:796–808, 1971.
300. G. W. Stewart. On the sensitivity of the eigenvalue problem $Ax = \lambda Bx$. *SIAM J. Numer. Anal.*, 9:669–686, 1972.
301. G. W. Stewart. Error and perturbation bounds for subspaces associated with certain eigenvalue problems. *SIAM Rev.*, 15:727–764, 1973.
302. G. W. Stewart. Gerschgorin theory for the generalized eigenvalue problem $Ax = \lambda Bx$. *Mathematics of Computation*, 29:600–606, 1975.
303. G. W. Stewart. Algorithm 407: HQR3 and EXCHNG: FORTRAN programs for calculating the eigenvalues of a real upper Hessenberg matrix in a prescribed order. *ACM Trans. Math. Software*, 2:275–280, 1976.
304. G. W. Stewart. *Matrix Algorithms. Vol. I.* SIAM, Philadelphia, PA, 1998. Basic decompositions.
305. G. W. Stewart. *Matrix Algorithms. Vol. II.* SIAM, Philadelphia, PA, 2001. Eigensystems.
306. G. W. Stewart. On the eigensystems of graded matrices. *Numer. Math.*, 90(2):349–370, 2001.
307. G. W. Stewart. A Krylov-Schur algorithm for large eigenproblems. *SIAM J. Matrix Anal. Appl.*, 23(3):601–614, 2001/02.
308. G. W. Stewart and J.-G. Sun. *Matrix Perturbation Theory*. Academic Press, New York, 1990.

309. W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, Princeton, NJ, 1994.
310. T. Ström. Minimization of norms and logarithmic norms by diagonal similarities. *Computing (Arch. Elektron. Rechnen)*, 10:1–7, 1972.
311. T. Stykel. Balanced truncation model reduction for semidiscretized Stokes equation. Technical report 04-2003, Institut für Mathematik, TU Berlin, 2003.
312. T. Stykel. Gramian-based model reduction for descriptor systems. *Math. Control Signals Systems*, 16:297–319, 2004.
313. J.-G. Sun. Perturbation expansions for invariant subspaces. *Linear Algebra Appl.*, 153:85–97, 1991.
314. J.-G. Sun. A note on backward perturbations for the Hermitian eigenvalue problem. *BIT*, 35(3):385–393, 1995.
315. J.-G. Sun. Perturbation analysis of singular subspaces and deflating subspaces. *Numer. Math.*, 73(2):235–263, 1996.
316. J.-G. Sun. Backward errors for the unitary eigenproblem. Report UMINF-97.25, Department of Computing Science, Umeå University, Umeå, Sweden, 1997.
317. J.-G. Sun. Stability and accuracy: Perturbation analysis of algebraic eigenproblems. Technical report UMINF 98-07, Department of Computing Science, University of Umeå, Umeå, Sweden, 1998. Revised 2002.
318. J.-G. Sun. Perturbation analysis of algebraic Riccati equations. Technical report UMINF 02-03, Department of Computing Science, University of Umeå, Umeå, Sweden, 2002.
319. R. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.
320. R. C. Thompson. Pencils of complex and real symmetric and skew matrices. *Linear Algebra Appl.*, 147:323–371, 1991.
321. C. Tischendorf. Topological index calculation of differential-algebraic equations in circuit simulation. *Surveys Math. Indust.*, 8(3-4):187–199, 1999.
322. F. Tisseur. Backward stability of the QR algorithm. TR 239, UMR 5585 Lyon Saint-Etienne, 1996.
323. F. Tisseur. A chart of backward errors for singly and doubly structured eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 24(3):877–897, 2003.
324. F. Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Rev.*, 43(2):235–286, 2001.
325. M.S. Tombs and I. Postlethwaite. Truncated balanced realization of a stable non-minimal state-space system. *Internat. J. Control*, 46(4):1319–1330, 1987.
326. L. N. Trefethen and M. Embree. *Spectra and Pseudospectra: The Behavior of Non-Normal Matrices and Operators*. Princeton University Press, 2005. To appear.
327. H. van der Vorst and G. H. Golub. 150 years old and still alive: eigenproblems. In *The state of the art in numerical analysis (York, 1996)*, volume 63 of *Inst. Math. Appl. Conf. Ser. New Ser.*, pages 93–119. Oxford Univ. Press, New York, 1997.
328. P. Van Dooren. Algorithm 590: DSUBSP and EXCHQZ: Fortran subroutines for computing deflating subspaces with specified spectrum. *ACM Trans. Math. Softw.*, 8:376–382, 1982.
329. P. Van Dooren. *Numerical Linear Algebra for Signal, Systems and Control*. Draft notes prepared for the Graduate School in Systems and Control, 2003.

330. C. F. Van Loan. *Generalized Singular Values with Algorithms and Applications*. PhD thesis, The University of Michigan, 1973.
331. C. F. Van Loan. A general matrix eigenvalue algorithm. *SIAM J. Numer. Anal.*, 12(6):819–834, 1975.
332. C. F. Van Loan. How near is a matrix to an unstable matrix? *Lin. Alg. and its Role in Systems Theory*, 47:465–479, 1984.
333. C. F. Van Loan. A symplectic method for approximating all the eigenvalues of a Hamiltonian matrix. *Linear Algebra Appl.*, 61:233–251, 1984.
334. A. Varga. A multishift Hessenberg method for pole assignment of single-input systems. *IEEE Trans. Automat. Control*, 41(12):1795–1799, 1996.
335. A. Varga. Periodic Lyapunov equations: some applications and new algorithms. *Internat. J. Control*, 67(1):69–87, 1997.
336. A. Varga. Balancing related methods for minimal realization of periodic systems. *Systems Control Lett.*, 36(5):339–349, 1999.
337. A. Varga. Computation of Kronecker-like forms of periodic matrix pairs. In *Proc. of Sixteenth International Symposium on Mathematical Theory of Networks and Systems (MTNS 2004), Leuven, Belgium*, 2004.
338. A. Varga and P. Van Dooren. Computational methods for periodic systems - an overview. In *Proc. of IFAC Workshop on Periodic Control Systems, Como, Italy*, pages 171–176, 2001.
339. R. S. Varga. *Matrix Iterative Analysis*. Prentice–Hall, Englewood Cliffs, NJ, 1962.
340. H. Voss. A symmetry exploiting Lanczos method for symmetric Toeplitz matrices. *Numer. Algorithms*, 25(1-4):377–385, 2000.
341. H. F. Walker. Implementation of the GMRES method using Householder transformations. *SIAM J. Sci. Stat. Comp.*, 9:152–163, 1988.
342. T.-L. Wang and W. B. Gragg. Convergence of the shifted QR algorithm for unitary Hessenberg matrices. *Math. Comp.*, 71(240):1473–1496, 2002.
343. T.-L. Wang and W. B. Gragg. Convergence of the unitary QR algorithm with a unimodular Wilkinson shift. *Math. Comp.*, 72(241):375–385, 2003.
344. R. C. Ward. *A numerical solution to the generalized eigenvalue problem*. PhD thesis, University of Virginia, Charlottesville, Va., 1974.
345. R. C. Ward. The combination shift QZ algorithm. *SIAM J. Numer. Anal.*, 12(6):835–853, 1975.
346. R. C. Ward. Balancing the generalized eigenvalue problem. *SIAM J. Sci. Statist. Comput.*, 2(2):141–152, 1981.
347. R. C. Ward and L. J. Gray. Eigensystem computation for skew-symmetric matrices and a class of symmetric matrices. *ACM Trans. Math. Software*, 4(3):278–285, 1978.
348. D. S. Watkins. Some perspectives on the eigenvalue problem. *SIAM Review*, 35:430–471, 1993.
349. D. S. Watkins. Shifting strategies for the parallel QR algorithm. *SIAM J. Sci. Comput.*, 15(4):953–958, 1994.
350. D. S. Watkins. Forward stability and transmission of shifts in the QR algorithm. *SIAM J. Matrix Anal. Appl.*, 16(2):469–487, 1995.
351. D. S. Watkins. The transmission of shifts and shift blurring in the QR algorithm. *Linear Algebra Appl.*, 241/243:877–896, 1996.
352. D. S. Watkins. Bulge exchanges in algorithms of QR type. *SIAM J. Matrix Anal. Appl.*, 19(4):1074–1096, 1998.

353. D. S. Watkins. Performance of the QZ algorithm in the presence of infinite eigenvalues. *SIAM J. Matrix Anal. Appl.*, 22(2):364–375, 2000.
354. D. S. Watkins. *Fundamentals of Matrix Computations*. Pure and Applied Mathematics. Wiley-Interscience [John Wiley & Sons], New York, 2002. Second editon.
355. D. S. Watkins. On Hamiltonian and symplectic Lanczos processes. *Linear Algebra Appl.*, 385:23–45, 2004.
356. D. S. Watkins. A case where balancing is harmful, 2005. Submitted.
357. D. S. Watkins. Product eigenvalue problems. *SIAM Rev.*, 47:3–40, 2005.
358. D. S. Watkins and L. Elsner. Chasing algorithms for the eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 12(2):374–384, 1991.
359. D. S. Watkins and L. Elsner. Convergence of algorithms of decomposition type for the eigenvalue problem. *Linear Algebra Appl.*, 143:19–47, 1991.
360. D. S. Watkins and L. Elsner. Theory of decomposition and bulge-chasing algorithms for the generalized eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 15:943–967, 1994.
361. J. R. Weaver. Centrosymmetric (cross-symmetric) matrices, their basic properties, eigenvalues, and eigenvectors. *Amer. Math. Monthly*, 92(10):711–717, 1985.
362. H. Weyl. *The Classical Groups*. Princeton University Press, Princeton, NJ, 1973. 8th Printing.
363. N. J. Wildberger. Diagonalization in compact Lie algebras and a new proof of a theorem of Kostant. *Proc. Amer. Math. Soc.*, 119(2):649–655, 1993.
364. J. H. Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, 1965.
365. J. H. Wilkinson and C. Reinsch. *Handbook for Automatic Computation. Vol. II Linear Algebra*. Springer-Verlag, New York, 1971.
366. The Working Group on Software: WGS, Available from `http://www.win.tue.nl/wgs/reports.html`. *SLICOT Implementation and Documentation Standards 2.1*, 1996. WGS-report 96-1.
367. S. J. Wright. A collection of problems for which Gaussian elimination with partial pivoting is unstable. *SIAM J. Sci. Comput.*, 14(1):231–238, 1993.
368. K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control*. Prentice-Hall, Upper Saddle River, NJ, 1996.

Index