Numerical Methods and Software for General and Structured Eigenvalue Problems

vorgelegt von

Dipl.-Math. Daniel Kreßner

von der Fakultät II - Mathematik und Naturwissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
- Dr. rer. nat. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Günter M. Ziegler
Berichter:	Prof. Dr. Volker Mehrmann
Berichter:	Prof. Dr. Ralph Byers
Berichter:	Prof. Dr. Bo Kågström

Tag der wissenschaftlichen Aussprache: 14.05.2004

Berlin 2004
D 83

Zusammenfassung

Die Dissertationsschrift beschäftigt sich mit numerischen Verfahren zur Lösung allgemeiner und strukturierter Eigenwertprobleme. Weiterhin werden Software-Implementierungen der behandelten numerischen Verfahren vorgestellt. Die Arbeit enthält zahlreiche Beiträge zu verschiedenen Aspekten des QR-Algorithmus, des QZ-Algorithmus, des periodischen QR-Algorithmus, verschiedener strukturerhaltender Algorithmen für (schief-)Hamiltonische Eigenwertprobleme und Arnoldi-ähnlichen Verfahren.

Abstract

In this thesis, we have investigated numerical methods for the solution of general and structured eigenvalue problems. Moreover, we have presented software implementing these methods. Contributions have been made to various aspects of the QR algorithm, the QZ algorithm, the periodic QR algorithm, structure-preserving methods for (skew-)Hamiltonian matrices, and the Krylov-Schur algorithm.

Contents

Preface

Man schreibt nicht, weil man etwas zu sagen hat, sondern weil man Lust hat, etwas zu sagen.
—Emile M. Cioran

This text is my PhD thesis. It is all about computing eigenvalues and invariant subspaces of 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 [129] 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 [111, 237].

Chapter 1 of this thesis is concerned with the QR algorithm, which was introduced by Francis in 1961–1962 based on earlier work by Rutishauser [198]. 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 [253] for a complete overview of the practical QR algorithm as it is currently implemented in LAPACK [7, 13]. An award-winning improvement was made in 2002 when Braman, Byers and Mathias [51] presented their aggressive early deflation strategy. The combination of this deflation strategy with a tiny-bulge multishift QR algorithm [50, 155] 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 LAPACK implementation. One of the contributions of Chapter 1 is the proposal of some simple modifications, which can lead to even further improvements. Another contribution consists of a new block algorithm for the post-processing step that is necessary to compute invariant subspaces from the output of the QR algorithm. It achieves high performance by applying techniques borrowed from the tiny-bulge multishift QR algorithm to the eigenvalue reordering algorithm by Bai and Demmel [14]. Numerical experiments suggest that the new algorithm requires substantially less computing time than the corresponding LAPACK implementation. Furthermore, Chapter 1 summarizes known and also some new 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 numerically backward stable 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 [179] in 1973. Its probably most notable modification has been the high-performance pipelined QZ algorithm developed by Dackland and Kågström [74]. The same authors developed an efficient method for reducing a matrix pair to Hessenberg-triangular form, for which some improvements are described. One topic of Chapter 2 is the use of Householder matrices in the QZ algorithm. In particular, it is shown that using so called opposite Householder matrices does not sacrifice its backward stability. A tiny-bulge multishift QZ algorithm with aggressive early deflation in the spirit of [50, 155] was already sketched in [3]. We provide several implementation details and preliminary numerical experiments showing that this algorithm substantially outperforms existing implementations of the QZ algorithm.

In Chapter 3, the periodic eigenvalue problem, i.e., the computation of invariant subspace and eigenvalues of a matrix product $A^{(p)}A^{(p-1)}\cdots A^{(1)}$ is discussed. Well-known results from linear algebra tell that this problem can be related to a standard eigenvalue problem involving a block cyclic matrix. Applying the perfect shuffle permutation to this block cyclic matrix turns it into a cyclic block matrix. The main theme of Chapter 3 is the exploitation of these connections to address some problems associated with periodic eigenvalue problems in a considerably simple and elegant manner. In particular, we develop a simple approach to the perturbation analysis of periodic eigenvalue problems [39, 166]. More importantly, it is shown that the QR algorithm automatically preserves cyclic block matrices if the number of shifts is wisely chosen. This establishes numerical equivalence between the QR algorithm and the previously known periodic QR algorithm [46, 120, 241], a backward stable algorithm for solving periodic eigenvalue problems. The techniques developed in Chapter 3 may lead to a better understanding not only of the periodic QR algorithm but also of other algorithms addressing similar problems.

Two other classes of structured matrices are the subject of Chapter 4: skew-Hamiltonian and Hamiltonian matrices. Developing structure-preserving and numerically backward stable algorithms for computing eigenvalues and invariant subspaces of such matrices has a long tradition in numerical linear algebra, see, e.g., [25]. The main purpose of Chapter 4 is to provide an overview of theoretical results and numerical methods in this area. Particular emphasis is put on the structured perturbation analysis, where several new results are presented. Often, structure-preserving algorithms have a lower computational complexity than general-purpose methods. Turning this potential advantage into an actual reduction of computing time can be difficult due to the availability of very efficient implementations for these general-purpose methods. By developing block algorithms for orthogonal symplectic decompositions, we derive similarly efficient implementations for certain structure-preserving algorithms.

In Chapter 5, we focus on a descendant of the Arnoldi method, the recently introduced Krylov-Schur algorithm by Stewart [225]. This algorithm belongs to the class of Krylov subspace methods and is suitable for computing selected eigenvalues and invariant subspaces of large and sparse matrices. We explain how this algorithm can be adapted to periodic and (skew-)Hamiltonian eigenvalue problems giving rise to new structure-preserving Arnoldi-like algorithms. Another subject of Chapter 5 is the balancing of sparse matrices for eigenvalue computations [72]. It is shown how existing methods can be modified for the symplectic balancing of sparse Hamiltonian matrices.

Most of the numerical methods described in this thesis have been implemented, see Appendix B. In particular, we mention HAPACK, a comprehensive Fortran 77 library aimed at computing eigenvalues and invariant subspaces of skew-Hamiltonian and Hamiltonian matrices. This software library comes with Matlab interfaces and can be downloaded from `http://www.math.tu-berlin.de/~kressner/hapack/`.

How to read this text

Readers of this text need to be familiar with the basic concepts of numerical analysis and linear algebra. Those are covered by any of the text books [80, 112, 222, 223, 258]. 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 [114, 191, 205, 239, 269] are particularly suited for this purpose with respect to content and style of presentation.

Acknowledgments

Slightly more than two years have passed since I have started the research that led to this thesis. It has been an enjoyable time, possibly with the exception of the being stuck or writing it down parts. Many people have contributed to my work and I assent to a statement by Paul Smit [208]:

Some people caused delay in the evolution of this thesis.
For this, I thank them.
Some people caused progress in the evolution of this thesis.
For this, I thank them.

It is not necessary to provide details about people from the first category. Nevertheless, I would like to express my gratitude to my parents who have done a great job in being who they are and supporting my education. Moreover, I mention one person for posing problems which offer more subtlety and fascination than an eigenvalue problem could ever offer. For this, I thank her.

The first and foremost person from the second category is my supervisor Volker Mehrmann. It was him who hooked me on Hamiltonian eigenvalue problems when I was still an undergraduate student. His always enlightening advice and generous support have accompanied me all along the way from early attempts of implementing a Pascal program of the QR algorithm to the submission of this thesis. In

many ways, he has been a wonderful supervisor. It was also him who constantly introduced me to other researchers in numerical linear algebra and initiated joint work.

In this respect, I am grateful to Bo Kågström for several invitations to the Department of Computing Science at Umeå University, Sweden, and for his great interest in my work. I always enjoyed a very hospitable atmosphere in Umeå and lively discussions with members of the local numerical linear algebra group. I am indebted to Paul Van Dooren for inviting me to stay for five months at CESAME (Université catholique de Louvain, Belgium) in the first half of 2003. The experience gained through discussions with him, CESAME's PhD students and visitors has had a strong, positive influence on my research. I thank Andras Varga for several discussions on periodic eigenvalue problems and an invitation to DLR, Oberpfaffenhofen.

My appetite for traveling should not led to false conclusions about the "Modellierung, Numerik, Differentialgleichungen" group at TU Berlin. I have greatly enjoyed the very likeable working atmosphere of this group, well-attended seminars, illuminating discussions with visitors and much more. Among the researchers who have directly influenced my work are Ralph Byers and Michael Karow. I am particularly indebted to Peter Benner, who regrettably moved to Chemnitz, for showing constant interest in my work and maintaining fruitful collaboration.

Throughout the research on my thesis, I have received financial support from the DFG Research Center "Mathematics for key technologies" (FZT 86) in Berlin, the Sfb 557 "Beeinflussung komplexer turbulenter Scherströmungen" and a Marie Curie fellowship in the frame of the Control Training Site (MCFI-2001-00403).

Chapter 1

The QR Algorithm

Z'n eigenwaarde? Heeft ie een minderwaardigheitscomplex? —Paul Smit [208]

```
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 first chapter of this thesis is about the QR algorithm, a numerically backward stable method for computing eigenvalues and invariant subspaces of a real or complex matrix.

The organization of this chapter is as follows. Section 1 is used to introduce the standard eigenvalue problem and the associated notions of invariant subspaces and (real) Schur decompositions. In Section 2, we summarize and (slightly) extent existing perturbation results for eigenvalues and invariant subspaces. The very basic, explicit shifted QR iteration is introduced in the beginning of Section 3. In the following subsection, Section 3.1, known results on the convergence of the QR iteration are recalled and illustrated. The other subsections are concerned with important implementation details such as implicit shifting and deflation, which eventually leads to the implicit shifted QR algorithm as it is in use nowadays, see Algorithm 1.27. In Section 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 4, we recall balancing and its merits on subsequent eigenvalue computations. Block algorithms are the subject of Section 5. First, in Sections 5.1 and 5.2, the standard block algorithm for reducing a general matrix to Hessenberg form, (implicitly) based on compact WY representations, is summarized. Deriving a block QR algorithm is a more subtle issue. In Sections 5.3 and 5.4, we illustrate 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 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 6. Finally, Section 7 is concerned with the computation of selected invariant subspaces from a real Schur decomposition.

Contributions in this chapter

Most of the material presented in this chapter is of preparatory value for subsequent chapters. As such, it does not contain major new contributions to the understanding or improvement of the QR algorithm, despite the number of pages it occupies. Nevertheless, the author contributes some, to the best of our knowledge, novel pieces to this ample subject:

- a proof that the condition number for a complex eigenvalue of a real matrix with respect to real perturbations is at most a factor of $1/\sqrt{2}$ smaller than the corresponding condition number with respect to complex perturbations, see Theorem 1.6;
- a connection between pole assignment and the computation of the first column of the shift polynomial in the QR algorithm, yielding a partial explanation for the slow convergence of the large-bulge multishift QR algorithm, see Section 5.4;
- an extension of the tiny-bulge multishift QR algorithm to larger bulges, see Section 5.5;
- a preliminary investigation on the use of aggressive early deflation for exploiting linear convergence phenomena in the QR algorithm, see Section 6.2;
- an efficient block algorithm for reordering Schur decompositions, see Section 7.3.

1 The Standard Eigenvalue Problem

The *eigenvalues* of a real matrix $A \in \mathbb{R}^{n\times n}$ are the roots of its characteristic polynomial $\det(A - \lambda I)$. The set of all eigenvalues is 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 $\lambda = \lambda_1 + \imath\lambda_2$ with $\lambda_1 \in \mathbb{R}, \lambda_2 \in \mathbb{R}\backslash\{0\}$ be an eigenvalue of $A \in \mathbb{R}^{n\times n}$. If a corresponding eigenvector has the form $x = x_1 + \imath x_2$ with $x_1, x_2 \in \mathbb{R}^n$, then we find that*

$$\begin{aligned} 2Ax_1 &= A(x+\bar{x}) = \lambda x + \bar{\lambda}\bar{x} = 2(\lambda_1 x_1 - \lambda_2 x_2), \\ 2Ax_2 &= \imath A(x-\bar{x}) = \imath\lambda x - \imath\bar{\lambda}\bar{x} = 2(\lambda_2 x_1 + \lambda_1 x_2). \end{aligned}$$

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

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

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

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

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 can be proven by successively combining the block decomposition (1.1) with Example 1.1.

Theorem 1.2 (Real Schur decomposition [112]). *Let $A \in \mathbb{R}^{n\times n}$, then there exists an orthogonal matrix Q so that $AQ = QT$ 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 if 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 7.

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 $\|E\|_2 \le \varepsilon\|A\|_2$ and ε is not much larger than the *unit roundoff* $\mathbf{u}$. Such an algorithm is called *backward stable* and the matrix E is called the *backward error*. Fortunately, almost all algorithms discussed in this thesis 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 [226] and a recent report by Sun [231].

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

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

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

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

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 ([226, Thm. V.1.3]). *The Sylvester equation (1.4) has a unique solution 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

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

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 that $R \in \text{kernel}(\mathbf{T})\backslash\{0\}$ has the singular value decomposition $R = U \left[\begin{smallmatrix} \Sigma & 0 \\ 0 & 0 \end{smallmatrix}\right] V^H$, where $\Sigma \in \mathbb{C}^{l \times l}$ is nonsingular. If we partition

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

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 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 $\mathbb{R}^{k \times (n-k)} \to \mathbb{R}^{k \times (n-k)}$ that is induced by the Frobenius norm. 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 $\mathrm{vec}(Y) \in \mathbb{C}^{m\cdot n}$ in their natural order. The Kronecker product and the vec operator have many useful properties, see [126, 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 $k(n-k) \times k(n-k)$ 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

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

see also [226].

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 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 [22, 231].

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 $\hat{x} = f_x(E)$, which satisfies $x^H(\hat{x} - x) = 0$. Moreover, 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 $\mathrm{span}\{x\}^\perp$.

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 and then $\hat{\lambda}$ is an eigenvalue of $A+E$ with 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

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

Hence, the implicit function theorem (see e.g. [145]) 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:

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

where $P = (xy^H)/(y^Hx)$ is the spectral projector belonging to λ, see (1.3). Note that the utilized upper bound $|y^HEx| \le \|E\|_2$ is attained for any $E = \varepsilon yx^H$. This shows that the *absolute condition number for a simple eigenvalue* λ can be written as

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

Note that the employed perturbation $E = \varepsilon yx^H$ cannot be chosen to be real unless the eigenvalue λ itself is real. Hence, $c(\lambda)$ might not be the appropriate condition number if $\lambda \in \mathbb{C}$ and the perturbations can be restricted to the set of real matrices. This fact has found surprisingly little attention in the available literature. Often, this difficulty is not even mentioned in standard text books on numerical linear algebra. Fortunately, 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:

$$c^{\mathbb{R}}(\lambda) := \lim_{\varepsilon\to 0}\sup\left\{\frac{|\hat{\lambda}-\lambda|}{\varepsilon} : E\in\mathbb{R}^{n\times n}, \|E\|_F\le\varepsilon\right\}. \tag{1.15}$$

For real λ, we can choose a real rank-one perturbation that attains the supremum in (1.15) so that $c^{\mathbb{R}}(\lambda) = c(\lambda) = \|P\|_2$ holds. For complex λ, we clearly have $c^{\mathbb{R}}(\lambda) \le c(\lambda)$ 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. *Let $\lambda \in \mathbb{C}$ be a simple eigenvalue of $A \in \mathbb{R}^{n\times n}$ with normalized right and left eigenvectors $x = x_R + \imath x_I$ and $y = y_R + \imath y_I$, 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

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

Proof. The perturbation expansion (1.12) readily implies

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

This is a standard linear least-squares problem with the following solution [44]. Its maximum value is given by the largest 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 attaining the supremum in (1.16) is given by a left singular vector belonging to this singular value. The square of the largest singular value of X is given by the larger root $\theta_\star$ of the polynomial

$$\theta^2 - (b^T b + c^T c)\theta + (b^T b)(c^T c) - (b^T c)^2.$$

By direct calculation, it can be shown that $b^T b + c^T c = 1$ and $1/4 - (b^T b)(c^T c) = 1/4 \cdot (b^T b - c^T c)^2$, implying

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

which concludes the proof. $\square$

For the matrix $A = \left[\begin{smallmatrix} 0 & 1 \\ -1 & 0 \end{smallmatrix}\right]$, we have $c^{\mathbb{R}}(\imath) = c^{\mathbb{R}}(-\imath) = 1/\sqrt{2}$ and $c(\imath) = c(-\imath) = 1$, revealing that the bound $c^{\mathbb{R}}(\lambda) \ge c(\lambda)/\sqrt{2}$ can actually be attained. It is the use of the Frobenius norm in the definition (1.15) of $c^{\mathbb{R}}(\lambda)$ that leads to the esthetically unpleasing effect that this condition number can become less than the norm of A. A general framework allowing the use of a broad class of norms has been developed by Karow [138] based on the theory of spectral value sets and real μ-functions. However, it is not known how a simple bound of the form $c^{\mathbb{R}}(\lambda) \geq \alpha c(\lambda)$ can be obtained from this theory.

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 $\|\hat{x} - x\|$ 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.

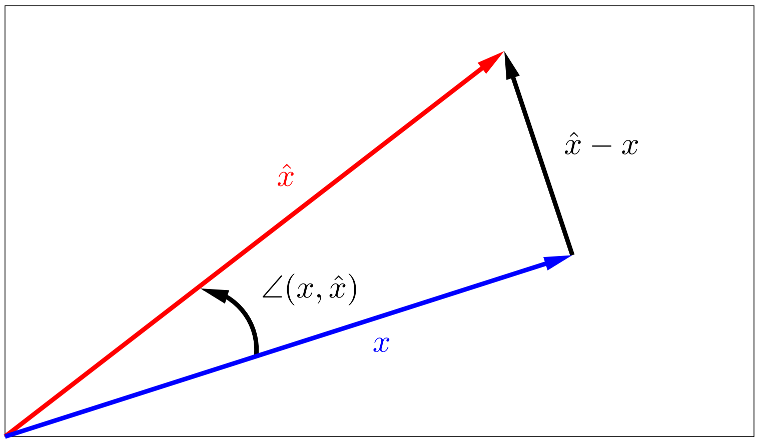


Figure 1.1. *Angle between two vectors.*

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

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

Proof. Using the fact that x is orthogonal to $(\hat{x} - x)$ we have

$$\cos \angle(x, \hat{x}) = 1/\|\hat{x}\|_2 = (1 + \|\hat{x} - x\|_2^2)^{-1/2}.$$

Expanding arccos 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. $\square$

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

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

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

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

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

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 [16]). *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. For example, if $\eta = 10^{-10}$ then the absolute value of these eigenvalues is $\eta^{1/10} = 0.1$ 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 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 helpful to consider clusters instead of individual eigenvalues. For this purpose, we assume that A has a block Schur decomposition of the form

$$AU = U \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \tag{1.19}$$

and that the eigenvalues of A_{11} form the considered cluster. If $\lambda(A_{11})$ only consists of values 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.

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 $A_{11} \in \mathbb{C}^{k\times k}$. Let the columns of Y form an orthonormal basis for the corresponding left invariant subspace. Assume that $\mathcal{X}$ is 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 $A + E$ with 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. 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 $\operatorname{range}(\hat{X})$ is an invariant subspace belonging to the eigenvalues of $\hat{A}_{11}$. 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

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

Again, the implicit function theorem guarantees the existence of functions $f_{A_{11}}$ and f_X on a sufficiently small, open neighborhood around 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 construct Newton and Newton-like methods for computing eigenvectors or invariant subspaces, see e.g. [79, 190]. Such methods are, however, not treated in this thesis although they are implicitly present in the QR algorithm [223, p. 418].

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

$$\left|\overline{\lambda(\hat{A}_{11})} - \overline{\lambda(A_{11})}\right| \le \frac{1}{k}\|P\|_2 \cdot \|E\|_{(1)} + \mathcal{O}(\|E\|^2) \le \|P\|_2 \cdot \|E\|_2 + \mathcal{O}(\|E\|^2), \tag{1.23}$$

where P is the projector belonging to $\lambda(A_{11})$ and $\|\cdot\|_{(1)}$ denotes the Schatten 1-norm [126].

Proof. The expansion (1.20) gives

$$\|\hat{A}_{11} - A_{11}\|_{(1)} \le \|(Y^H X)^{-1}\|_2 \cdot \|E\|_{(1)} + \mathcal{O}(\|E\|^2) = \|P\|_2 \cdot \|E\|_{(1)} + \mathcal{O}(\|E\|^2),$$

where we used (1.5). Combining this inequality with

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

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

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

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 $\sigma_1 \le \sigma_2 \le \cdots \le \sigma_k$ denote the singular values of $X^H Y$. Then the* canonical angles *between $\mathcal{X}$ and $\mathcal{Y}$ are defined by*

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

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

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

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

see also Figure 1.2.

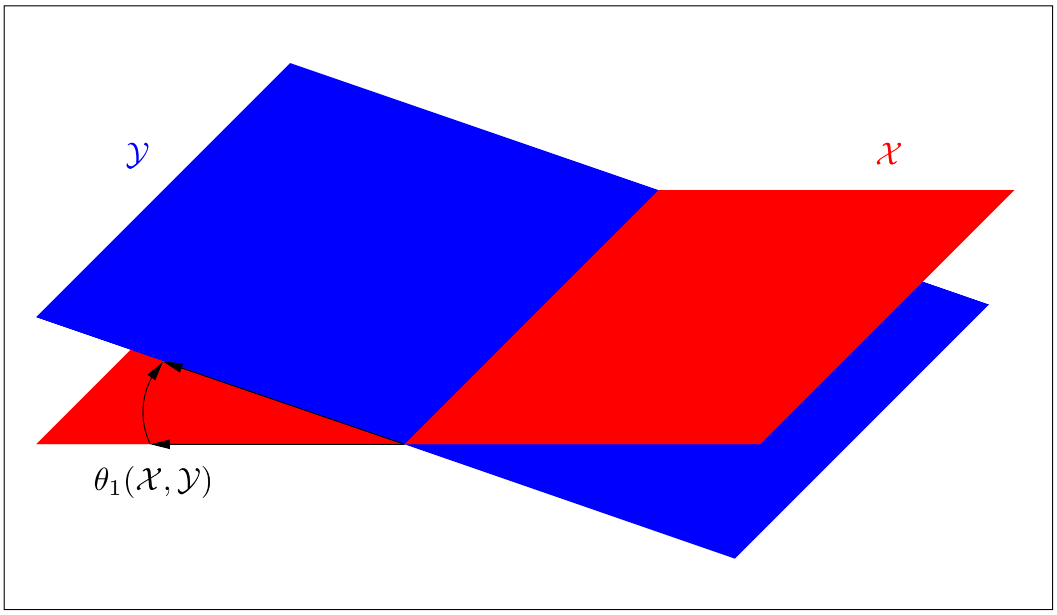


Figure 1.2. *Largest canonical angle between two subspaces.*

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 $d_\gamma(\mathcal{X}, \mathcal{Y}) =$

$\|\sin[\Theta(\mathcal{X},\mathcal{Y})]\|_\gamma$, see [226, 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 ([226]). *Let $\mathcal{X}$ be spanned by the columns of $[I,0]^H$, and $\mathcal{Y}$ by the columns of $[I,Q^H]^H$. If $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_k$ denote the singular values of Q then*

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

Proof. The columns of $[I,Q^H]^H(I+Q^HQ)^{-1/2}$ form an orthonormal basis for $\mathcal{Y}$. Let $Q = U\Sigma V^H$ with $\Sigma = \operatorname{diag}(\sigma_1,\dots,\sigma_k)$ be a singular value decomposition. Then

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

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. □

Now we are prepared to generalize Corollary 1.7 to invariant subspaces.

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

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

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

$$\|Q\|_F = \|\hat{X}-X\|_F \le \|E\|_F/\operatorname{sep}(A_{11},A_{22}).$$

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 $\|\mathbf{T}^{-1}V\|_F = 1/\operatorname{sep}(A_{11},A_{22})$. The existence of such a matrix is guaranteed by the well-known Weierstrass theorem. 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* $\mathcal{X}$:

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

see also [217, 219, 226].

On the computation of sep

The separation of two matrices $A_{11} \in \mathbb{C}^{k\times k}$ and $A_{22} \in \mathbb{C}^{(n-k)\times(n-k)}$ is 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 cheaper estimate of sep can be obtained by applying a norm estimator [122, 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 particular chosen right hand sides c and d or, equivalently, the solution of a few Sylvester equations $A_{11}X - XA_{22} = C$ and $A_{11}^T X - XA_{22}^T = D$. This approach becomes particularly attractive if A_{11} and A_{22} are already in (real) Schur form, see [18, 60, 130, 133].

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.

Using an approach taken by Stewart [219] and extended by Demmel [78], see also [70], we now derive exact perturbation bounds which are valid if (1.27) holds. 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 $W = \left[\begin{smallmatrix} I & 0 \\ -Q & I \end{smallmatrix}\right]$ so 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 implies that Q is a solution of the quadratic matrix equation

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

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

Lemma 1.14. *If $\|A_{12}\|_F \cdot \|A_{21}\|_F < \operatorname{sep}^2(A_{11}, A_{22})/4$ 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 [217, 219] or [226, 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 2.3, Chapter 4, to solve nonlinear matrix equations of a more complicated nature. □

An orthonormal basis for an invariant subspace $\hat{\mathcal{X}}$ of A is now given by

$$\hat{X} = \left[\begin{array}{c} I \\ -Q \end{array} \right] (I + Q^H Q)^{-1/2}, \tag{1.30}$$

see [78], 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*

$$AU = U \left[\begin{array}{cc} A_{11} & A_{12} \\ 0 & A_{22} \end{array} \right], \quad A_{11} \in \mathbb{R}^{k \times k}, A_{22} \in \mathbb{R}^{(n-k)\times(n-k)},$$

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

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

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 [78, Lemma 7.8]. It differs in the respect that Demmel provides an upper bound for $\| \tan[\Theta(\mathcal{X}, \hat{\mathcal{X}})] \|_2$ instead of $\| \tan[\Theta(\mathcal{X}, \hat{\mathcal{X}})] \|_F$. The following proof, however, is almost identical to the proof in [78].

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 $A + E$:

$$W_R^{-1}(A + E)W_R = \left[\begin{array}{cc} A_{11} & 0 \\ 0 & A_{22} \end{array} \right] + \left[\begin{array}{cc} F_{11} & F_{12} \\ F_{21} & F_{22} \end{array} \right].$$

By direct computation, $\|F_{11}\|_F$, $\|F_{21}\|_F$ and $\|F_{22}\|_F$ are bounded from above by $\|P\|_2\|E\|_F$. Furthermore,

$$\|F_{12}\|_F = \left\| \begin{bmatrix} I & R \end{bmatrix} E \begin{bmatrix} -R \\ I \end{bmatrix} \right\|_F / \|P\|_2 \leq \|P\|_2 \cdot \|E\|_F.$$

From the definition of sep it follows that

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

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

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

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

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

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

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

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 [112] (SVD) with $\Sigma = \operatorname{diag}(\sigma_1, \dots, \sigma_k)$ and $\sigma_1 \geq \cdots \geq \sigma_k$. Using (1.31), 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

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

and, since $\sigma_1 \leq \eta$, this concludes the proof. □

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

3 The Basic QR Algorithm

The QR iteration, introduced by Francis [100], generates a sequence of orthogonally similar matrices $A_0 \leftarrow A, A_1, A_2, \dots$ which, under suitable conditions, converges to a nontrivial block Schur form of A. Its name stems from the QR decompositions that are used in the course of the iteration. The second ingredient of the QR iteration is a 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:

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

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

3.1 Local Convergence

In the following we summarize the convergence analysis by Watkins and Elsner [261] of the sequence 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 seen that the matrix A_i has block Schur form

$$\left[\begin{array}{cc} A_{11}^{(i)} & A_{12}^{(i)} \\ 0 & A_{22}^{(i)} \end{array} \right], \quad A_{11}^{(i)} \in \mathbb{R}^{k \times k}, \tag{1.35}$$

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 small, 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

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

If we define $\mathcal{S}_0 := \operatorname{span}\{e_1, e_2, \dots, e_k\}$ 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 a more compact form as $\mathcal{S}_i = \hat{p}_i(A)\mathcal{S}_0$, 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*

$$AU = U \left[\begin{array}{cc} A_{11} & A_{12} \\ 0 & A_{22} \end{array} \right],$$

where $A_{11} \in \mathbb{R}^{k \times k}$ and $A_{22} \in \mathbb{R}^{(n-k)\times(n-k)}$. 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 $p_1, p_2, \dots,$ assume that $\hat{p}_i(A_{11})$ is *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 $\mathcal{S}_i = \hat{p}_i(A)\mathcal{S}_0$, $i = 1, 2, \dots$, and the invariant subspace $\mathcal{X}_1$ can be bounded by*

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

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 [261], 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 $W_R = \left[\begin{smallmatrix} I & -R/\|P\|_2 \\ 0 & I/\|P\|_2 \end{smallmatrix}\right]$, 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 [261]), it follows that

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

By direct computation,

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

which concludes the proof. □

We 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. In this case, the subspace condition is satisfied for $\mathcal{S}_0 = \text{span}\{e_1, e_2, \dots, e_k\}$ and 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 say something about the choice of shift polynomials. 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*:

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

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

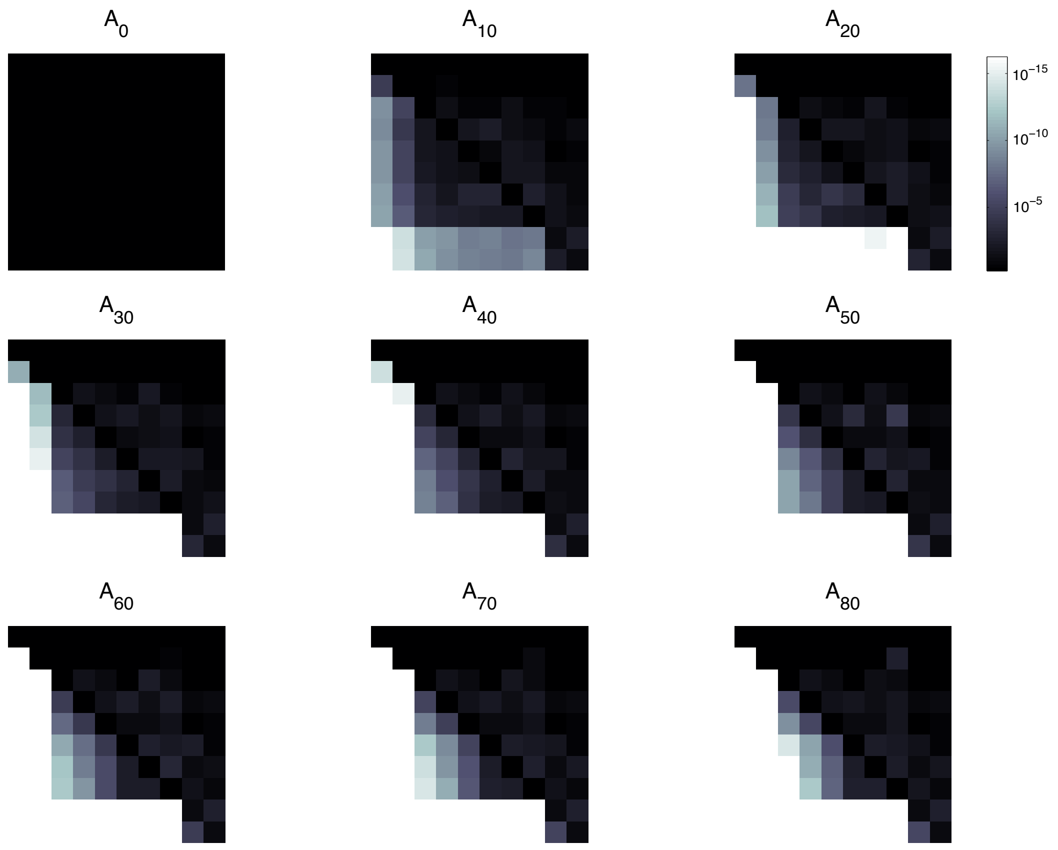


Figure 1.3. *Convergence pattern of the unshifted QR iteration.*

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

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

and X *is a random matrix with condition number* 10^3. *We applied* 80 *unshifted QR iterations to* $A_0 = A$; *the absolute values of the elements in* A_i *for* $i = 0, 10, \ldots, 80$ *are displayed in Figure 1.4. 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* $\mathbf{u} \cdot \|A\|_2$. *The diagonal elements of* A_i *approximate the diagonal elements of* Λ *in the same, 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 ([261]). *Let* $A \in \mathbb{R}^{n \times n}$ *and let* p *be a polynomial. Assume that there exists a partitioning* $\Lambda(A) = \Lambda_1 \cup \Lambda_2$ *so 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 $\mathcal{S}_i = p(A)^i \mathcal{S}_0$, $i = 1, 2, \dots$, and the invariant subspace $\mathcal{X}_1$ can be bounded by

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

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

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

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

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, 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 and $p_1(A_{22}^{(1)}) = 0$. 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:

$$\begin{aligned} p_i(\lambda) &\leftarrow \det(\lambda I_m - A_{22}^{(i-1)}), && (1.39a) \\ p_i(A_{i-1}) &= Q_i R_i, \quad \text{(QR decomposition)} && (1.39b) \\ A_i &\leftarrow Q_i^T A_{i-1} Q_i. && (1.39c) \end{aligned}$$

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 elements in A_i for $i = 0, 1, \dots, 8$ are displayed in Figure 1.4. What can be observed is that the* 2×2 *bottom right block, which contains approximations to the*

[1]It is not known to us who coined the term "Francis shifts". Uses of this term can be found in [80, 223]. Some authors prefer the terms "Wilkinson shifts" or "generalized Rayleigh quotient shifts".

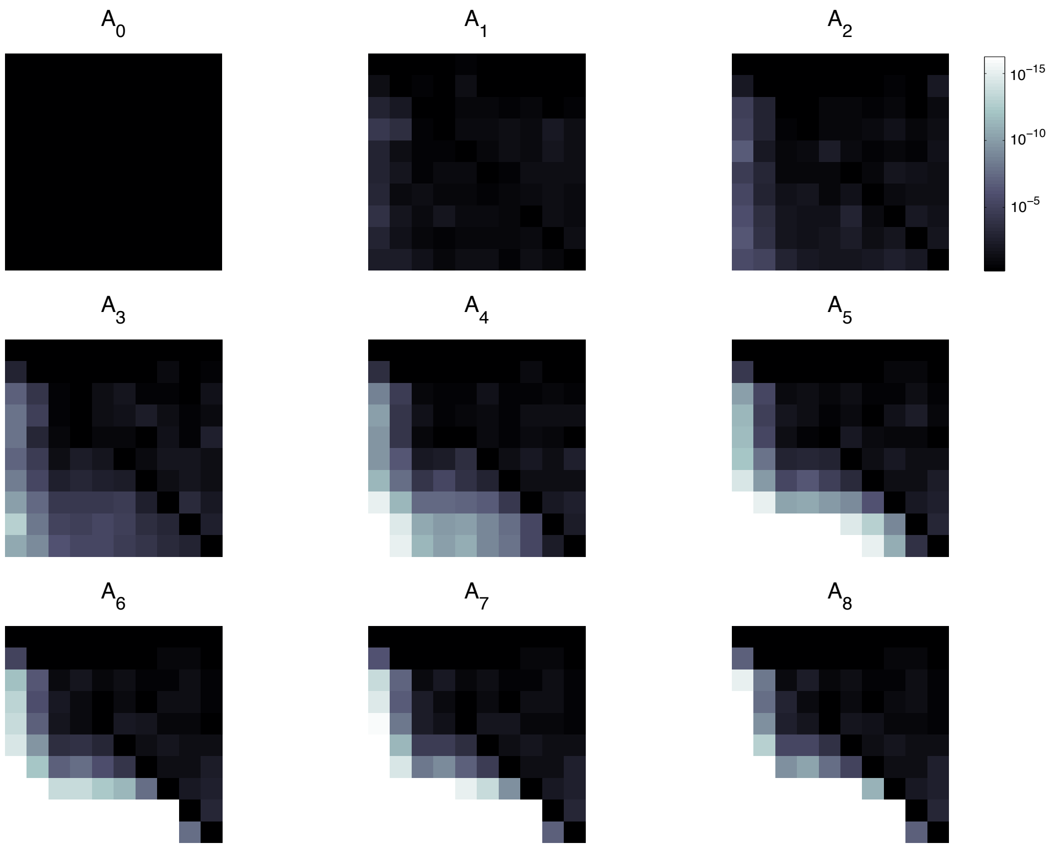


Figure 1.4. *Convergence pattern of the shifted QR iteration with two Francis shifts.*

eigenvalue cluster $\{0.59, 0.6\}$*, converges rather quickly. Already after six iterations all elements to the left of this block have magnitude less than* $\mathbf{u} \cdot \|A\|_2$*. Also, the rest of the matrix has made a lot of progress towards convergence. Most notably the eighth diagonal element 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 ([261]). *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 sequence of subspaces* $\mathcal{S}_i = \hat{p}_i(A)\mathcal{S}_0$ *with* $\mathcal{S}_0 = \operatorname{span}\{e_1, \dots, e_{n-m}\}$ *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 be the eigenvalues belonging to $\mathcal{X}_1$, and let $\mathcal{X}_2$ be the invariant subspace belonging to the eigenvalues $\Lambda_2 = \lambda(A) \backslash \Lambda_1$. For sufficiently small ε, we may assume that $\mathcal{S}_{i-1} \cap \mathcal{X}_2 = \{0\}$ as $\mathcal{X}_1$ and $\mathcal{X}_2$ are distinct subspaces. The $(i-1)$th iterate of (1.39) takes the form

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

We have $\|A_{21}^{(i-1)}\|_2 \leq \sqrt{2}\|A\|_2 \cdot \varepsilon$ [261, 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 $M = 2c_2(2\|A\|_2 + 1)^{m-1}\|A\|_2$. Since

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

we may choose a sufficiently small ε so that all roots of p_i have a distance of at least $\delta/2$ to the eigenvalues in Λ_1. 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 $\varepsilon \leq 1/\sqrt{2}$. □

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 show how we can 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 matrix A in upper Hessenberg form is called* unreduced[2] *if all its subdiagonal elements are nonzero.*

If one of the subdiagonal elements 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 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.*

[2]Some authors use the term *proper* instead of unreduced. Note that the occasionally used term *irreducible* is, strictly speaking, not correct as a matrix in unreduced Hessenberg form may be reducible.

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

$$K_n(A, e_1) = \left[\begin{array}{cccc} e_1 & Ae_1 & \cdots & A^{n-1}e_1 \end{array} \right]$$

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 Hessenberg form then, subtracting possible deflations, the shifted QR iteration preserves 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 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

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

and

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

Under this choice of v and β, we identify $H_j(x) \equiv H(v, \beta)$. Note that $H_j(x)$ does not alter the first $j - 1$ elements when applied to a vector.

We illustrate how Householder matrices can be used for reducing a matrix $A \in \mathbb{R}^{5 \times 5}$ 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:

$$A \leftarrow H_2(Ae_1)^T A H_2(Ae_1) = \left[\begin{array}{ccccc} a & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \end{array} \right].$$

Similar diagrams have been used by Wilkinson in his book [264] 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 in the current transformation. Consequently, an entry denoted by $\hat{0}$ is being annihilated in the current transformation.

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

$$A \leftarrow H_3(Ae_2)^T A H_3(Ae_2) = \begin{bmatrix} a & a & \hat{a} & \hat{a} & \hat{a} \\ a & a & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{a} \end{bmatrix},$$

and, finally, the $(5, 3)$ element:

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

For general n, the corresponding algorithm reads as follows.

Algorithm 1.23 (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 $H = Q^T A Q$ is in upper Hessenberg form. The matrix A is overwritten by H.*

```
$Q \leftarrow I_n$
FOR $j \leftarrow 1, \dots, n-2$
    $Q \leftarrow Q \cdot H_{j+1}(Ae_j)$
    $A \leftarrow H_{j+1}(Ae_j) \cdot A \cdot H_{j+1}(Ae_j)$
END FOR
```

Several remarks regarding the actual implementation of this algorithm 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 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 by two calls to 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 1.23 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 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.

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., [112, 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 [100], avoids these difficulties by making use of the following "uniqueness property" of the Hessenberg reduction.

Theorem 1.24 (Implicit Q theorem [112, Thm. 7.4.2]). *Let $U = [u_1, \dots, u_n]$ and $V = [v_1, \dots, v_n]$ 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 $D = \operatorname{diag}(1, \pm 1, \dots, \pm 1)$ 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 1.24 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.

The following algorithm constructs such a matrix Q_i for the shifted QR iteration (1.39) employing Francis shifts. It uses 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 1.23 has the form $Q = 1 \oplus \tilde{Q}$. Here, '$\oplus$' denotes the direct sum (or block diagonal concatenation) of two matrices.

Algorithm 1.25 (Implicit shifted QR iteration).

Input:	*A matrix $A_{i-1} \in \mathbb{R}^{n \times n}$ 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 $\sigma_1, \dots, \sigma_m$ as eigenvalues of $A_{i-1}(n-m+1:n, n-m+1:n)$.

2. Set $x = (A_{i-1} - \sigma_1 I_n) \cdot (A_{i-1} - \sigma_2 I_n) \cdots (A_{i-1} - \sigma_m I_n) e_1$.
3. Update $A_{i-1} \leftarrow H_1(x) \cdot A_{i-1} \cdot H_1(x)$.
4. Apply Algorithm 1.23 to compute an orthogonal matrix Q so that A_{i-1} is reduced to Hessenberg form.
5. Set $Q_i = H_1(x) Q$.

The shifts $\sigma_1, \dots, \sigma_m$ can be computed by an auxiliary implementation of the QR algorithm which employs at most two 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 1.29 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 $(A_{i-1} - \sigma_j I_n) \cdot (A_{i-1} - \bar{\sigma}_j I_n)$ 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 [90] have developed an algorithm which computes a multiple of x without determining the shifts.

Step 4 of Algorithm 1.25 should be based on a special-purpose implementation of Algorithm 1.23, 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., [112, 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 into a given orthogonal matrix.[3]

Let us illustrate Algorithm 1.25 for $n = 6$ and $m = 2$. First, the upper Hessenberg matrix A_{i-1} is overwritten by $H_1(x) A_{i-1} H_1(x)$, 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:

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

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 [260]. In the first step of Algorithm 1.23, the nonzero elements introduced in the first column of A_{i-1} are annihilated by applying the Householder

[3]Note that for $m = 2$, these flop counts differ from the $10n^2$ given in [112, p. 358] and $6n^2$ in [223, p. 123].

matrix $H_2(A_{i-1}e_1)$:

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

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:

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

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

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

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

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

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

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

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

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 [224] 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 example of such a graded matrix is given below.

Example 1.26 ([147]). *Let*

$$A = \begin{bmatrix} 10^{0} & 10^{-3} & 0 & 0 \\ 10^{-3} & 10^{-7} & 10^{-10} & 0 \\ 0 & 10^{-10} & 10^{-14} & 10^{-17} \\ 0 & 0 & 10^{-17} & 10^{-21} \end{bmatrix}.$$

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	$1.00000099999910\underline{0}1$	$1.000000999999099\underline{9}$
$-.8999991111128208\times10^{-06}$	$-.8999991111128\underline{2}12\times10^{-06}$	$-.8999991111128\underline{2}13\times10^{-06}$
$.2111111558732113\times10^{-13}$	$.211111\underline{1}085047986\times10^{-13}$	$.21111115587321\underline{1}4\times10^{-13}$
$-.3736841266803067\times10^{-20}$	$0.0999999999999999\times10^{-20}$	$-.37368412668030\underline{6}8\times10^{-20}$

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 [255] that even tiny subdiagonal elements (as small as $10^{-70}\cdot\|A\|_2$) do not affect the convergence of the QR algorithm.

3.5 The Overall Algorithm

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

Algorithm 1.27 (Basic QR algorithm).

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

```
$Q \leftarrow I_n$
$i \leftarrow 1$,   $l \leftarrow n$
FOR $it \leftarrow 0, \dots, 30 \cdot n$
  % The active submatrix is $H(i : l, i : l)$. Search for deflations.
```

```
    $k \leftarrow i$
    WHILE $k < l$ AND $|h_{k+1,k}| > \mathbf{u} \cdot (|h_{k,k}| + |h_{k+1,k+1}|)$
      $k \leftarrow k + 1$
    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.
        $l \leftarrow i - 1$;    $i \leftarrow 1$
        IF $i + 1 \geq l$ THEN
          % QR algorithm has converged.
          Exit.
        END IF
      END IF
    ELSE
      Apply implicit shifted QR iteration, Algorithm 1.25, with $\min\{m, l-i+1\}$
      shifts to $H(i : l, i : l)$. Let $\tilde{Q}$ be 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 $30 \cdot n$ iterations.
  Exit and error return.
```

Properly implemented, this algorithm 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.

Example 1.28. *We applied Algorithm 1.27 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. Blue curves correspond to matrices generated by the* MATLAB *command* `hess(rand(n))` *and red curves to matrices generated by* `triu(rand(n),-1)`. *It can be seen that the QR algorithm requires significantly more flops for the second class of matrices. Moreover, the flops are underestimated by a factor of $1.2 \ldots 1.7$ for $m = 4$. The latter effect will be explained in more detail in Sections 5.3 and 5.4.*

Remark 1.29. *In order to obtain a real Schur form, the 2×2 diagonal blocks of the matrix T returned by Algorithm 1.27 must be post-processed. Blocks with real eigenvalues are transformed to upper triangular form. Blocks with pairs of complex*

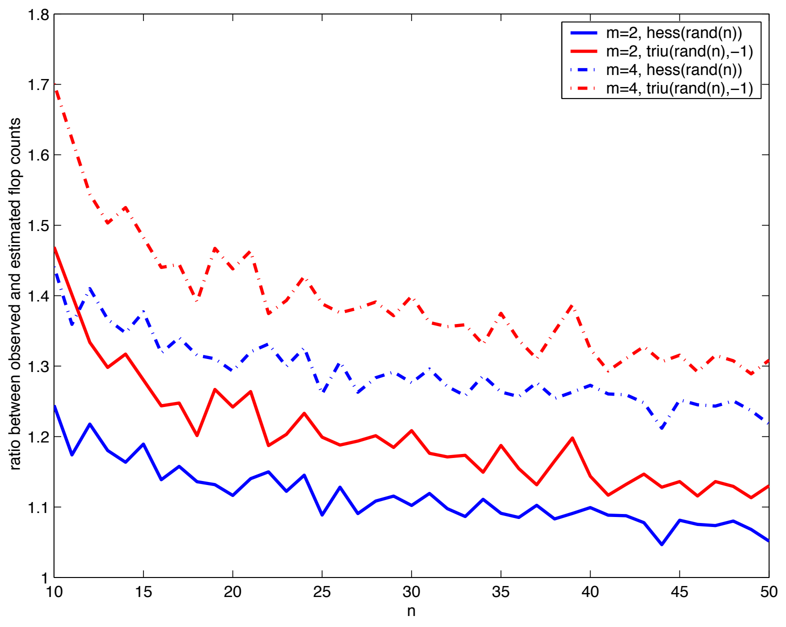


Figure 1.5. *Ratio between observed and estimated flops for reducing a Hessenberg matrix to a block upper triangular matrix using the QR algorithm (Algorithm 1.27).*

conjugate eigenvalues are 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 given by $a \pm \imath\sqrt{bc}$. *Both transformations can be achieved by Givens rotations using only basic arithmetic operations. The actual implementation, however, requires some care in order to guarantee the backward stability of the QR algorithm. More details can be found in recent work by Sima [206]. See also the LAPACK subroutine* DLANV2.

Algorithm 1.27 is implemented in LAPACK as subroutine DHSEQR, which uses only a small number of Francis shifts, typically $m = 6$. The auxiliary subroutine DLAHQR uses two shifts; it is used 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 [264]. Similar remarks apply to Algorithm 1.23, 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.

3.6 Failure of Global Converge

We have seen that the convergence of the QR algorithm is 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 [20, 187]. This statement, however, does not hold for nonsymmetric matrices [21]. Nevertheless, there are very few examples of practical relevance known, where the QR algorithm does not converge within $30 \cdot n$ iterations, even in finite precision arithmetic.

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

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

This matrix stays invariant under shifted QR iterations that use $m < n$ Francis shifts [170, 264]. This situation can be resolved by replacing after 10 and 20 shifted QR iterations without deflation the Francis shift polynomial by a so called *exceptional shift polynomial.* Martin, Peters and Wilkinson [170] proposed to use 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 1.27. Note that the EISPACK [209] implementation of the QR algorithm, subroutine HQR, uses $p_e(x - h_{ll})$ instead of $p_e(x)$.

Both exceptional shift strategies can be defeated [19, 77]. One such example is the matrix

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

whose discovery is attributed to Demmel [77, 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 [77] 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 [264, pg. 512].

Day's strategy is implemented in the LAPACK subroutine DLAHQR. Still, it can be defeated as shown by the following example, which is quoted in the beginning of this chapter.

Example 1.30. *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* $30 \cdot 4 = 120$ *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 a viable option if an application 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 Section 6.3, Chapter 5. Shift blurring is, especially for large numbers of shifts, a significant source of slow convergence in finite precision arithmetic, see Section 5.3 below.

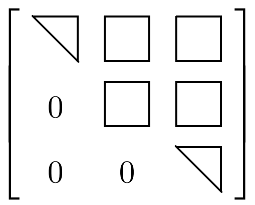
4 Balancing

Balancing is a preprocessing step, which often has positive effects on the performance and accuracy of numerical methods for computing eigenvalues and therefore has obtained early attention in the development of numerical linear algebra [184, 189]. 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.

4.1 Isolating Eigenvalues

In the first stage of the balancing algorithm by Parlett and Reinsch [189], a permutation matrix P is constructed so that P^TAP takes the form

$$P^TAP = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ 0 & A_{22} & A_{23} \\ 0 & 0 & A_{33} \end{bmatrix} = \qquad (1.47)$$



where $A_{11} \in \mathbb{R}^{(i_l-1)\times(i_l-1)}$ 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.

The matrix P can be composed of elementary permutation matrices

$$P_{ij} = I - e_ie_i^T - e_je_j^T + e_ie_j^T + e_je_i^T. \qquad (1.48)$$

If a matrix A is post-multiplied by P_{ij} then columns i and j of A are swapped. Correspondingly, a pre-multiplication by P_{ij} swaps rows i and j. It can be easily seen that the following algorithm produces a decomposition of the form (1.47).

Algorithm 1.31 ([189]).

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

Output: *A permutation matrix P so that P^TAP is in block triangular form (1.47) for some integers i_l, i_h. The matrix A is overwritten by P^TAP.*

```
$P \leftarrow I_n$
$i_l \leftarrow 1, \quad i_h \leftarrow n$
swapped $\leftarrow 1$
WHILE (swapped $= 1$)
  swapped $\leftarrow 0$
  % Search for row having only zero off-diagonal elements in active
  % submatrix $A(i_l : i_h, i_l : i_h)$ and swap with $(i_h)$th row.
  $i \leftarrow i_l$
  WHILE $(i \leq i_h)$ AND (swapped $= 0$)
    IF $\sum_{\substack{j=i_l \\ j \neq i}}^{i_h} |a_{ij}| = 0$ THEN
      swapped $\leftarrow 1$
      $A \leftarrow P_{i,i_h}^T \cdot A \cdot P_{i,i_h}$
      $P \leftarrow P \cdot P_{i,i_h}$
      $i_h \leftarrow i_h - 1$
    END IF
    $i \leftarrow i + 1$
  END WHILE
  % Search for column having only zero off-diagonal elements in active
  % submatrix $A(i_l : i_h, i_l : i_h)$ and swap with $(i_l)$th column.
  $j \leftarrow i_l$
  WHILE $(j \leq i_h)$ AND (swapped $= 0$)
    IF $\sum_{\substack{i=i_l \\ i \neq j}}^{i_h} |a_{ij}| = 0$ THEN
      swapped $\leftarrow 1$
      $A \leftarrow P_{i_l,j}^T \cdot A \cdot P_{i_l,j}$
      $P \leftarrow P \cdot P_{i_l,j}$
      $i_l \leftarrow i_l + 1$
    END IF
    $j \leftarrow j + 1$
  END WHILE
END WHILE
```

The computational work of this algorithm is somewhat difficult to measure as it can be implemented without any floating point operations. It requires $\mathcal{O}((i_l + i_h - n)n)$ comparisons. Experience gained by numerical experiments tells that performance benefits gained from the order reduction of the eigenvalue problem greatly outweigh the computational time needed by Algorithm 1.31.

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 (1.47). A 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 $j = 1, \ldots, n$. Under the assumption that A_{22} is irreducible, Osborne [184] 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 1.31 does not necessarily produce an irreducible matrix A_{22}, an algorithm that guarantees this property will be described in Section 5.1, Chapter 5.

The iterative procedure proposed by Osborne for computing the scaling matrix D is used in the balancing algorithm by Parlett and Reinsch [189]. Although this algorithm is able to balance a matrix in any vector norm, the LAPACK implementation `DGEBAL` uses the 1-norm because of efficiency considerations. Grad [113] 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 [71, 72, 93, 117, 137, 207, 228].

The following algorithm is basically LAPACK's implementation of the Parlett-Reinsch algorithm.

Algorithm 1.32.

Input: *A matrix $A \in \mathbb{R}^{n\times n}$ having the block triangular form (1.47) 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 $= 0$)
  converged $\leftarrow 1$
  FOR $j \leftarrow i_l, \dots, i_h$
    $c \leftarrow \sum_{i=i_l, i\neq j}^{i_h} |a_{ij}|, \quad r \leftarrow \sum_{k=i_l, k\neq j}^{i_h} |a_{jk}|, \quad s \leftarrow c + r, \quad \text{scal} \leftarrow 1$
    WHILE $c < r/\beta$
      $c \leftarrow c \cdot \beta, \quad r \leftarrow r/\beta, \quad \text{scal} \leftarrow \text{scal} \cdot \beta$
    END WHILE
    WHILE $c \geq r \cdot \beta$
      $c \leftarrow c/\beta, \quad r \leftarrow r \cdot \beta, \quad \text{scal} \leftarrow \text{scal}/\beta$
    END WHILE
    % Scaling is only applied if it results in a considerable reduction of
    % the column and row norms.
    IF $(c + r) < 0.95 \cdot s$ THEN
      converged $\leftarrow 0, \quad \tilde{d}_{jj} \leftarrow \text{scal} \cdot \tilde{d}_{jj}$
      $A(:, j) \leftarrow \text{scal} \cdot A(:, j), \quad A(j, :) \leftarrow 1/\text{scal} \cdot A(j, :)$
    END IF
  END FOR
END WHILE
```

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 Algorithm 1.32 to the matrix P^TAP in block triangular form (1.47), then, from the discussion above, we may conclude that the algorithm returns a nearly balanced matrix $D^{-1}A_{22}D$. Algorithm 1.32 requires at most $4kn^2 + \mathcal{O}(n)$ flops, where $k = \mathcal{O}(1)$ denotes the number of iterations until convergence.

Algorithm 1.32 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.

Example 1.33 ([72]). *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 is balanced by the diagonal matrix $D = \mathrm{diag}(4, 2, 1, 1/2)$. *Algorithm 1.32, however, returns the unbalanced matrix*

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

The two ingredients of balancing, Algorithms 1.31 and 1.32, 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 1.31. Otherwise, $\mathrm{scal}(j)$ contains $\tilde{d}_{jj}$, the jth diagonal entry of the diagonal matrix $\tilde{D}$ returned by Algorithm 1.32.

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

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 the subsequent orthogonal transformations. Occasionally, the norm of the matrix returned by Algorithm 1.32 is increased. This increase, however, 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 1.31 are read off without any roundoff errors. Also, eigenvalues of the unreduced, balanced block $D^{-1}A_{22}D$ are often more accurately computed, mainly due to the decrease of norm. Some numerical experiments illustrating this effect for matrices from the test matrix collection [12] can be found in [71, 72].

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 [12], see Section 5.5. An extreme example is given in Example 1.30 above, where the QR algorithm converges for the balanced matrix but does not converge for the unbalanced matrix within $30 \cdot n$ iterations.

5 Block Algorithms

The bulge of the computational work of Algorithm 1.23, reduction 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 [80, 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.

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 1.34 (Compact WY representation [202]). *Let $k \le n$ 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 (see equation (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

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

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

$$\tilde{T} = \begin{bmatrix} T & -\beta TW^T v \\ 0 & -\beta \end{bmatrix}, \quad \tilde{W} = \begin{bmatrix} W & v \end{bmatrix}, \tag{1.49}$$

concluding the proof. □

It is often the case that $j_i = i$. Then a proper implementation of the construction given in the proof of Theorem 1.34 requires $k^2 n - \frac{1}{3}k^3 + \mathcal{O}(k^2)$ flops, see also the LAPACK routine `DLARFT`. Furthermore, an inspection of (1.49) reveals that the upper $k \times k$ block of W is a unit lower triangular matrix. In this case, 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 1.35. *The representation $I_n + WTW^T$ is called compact WY representation because it has a "non-compact" predecessor: $I_n + WY$, where $Y = WT^T$ [42].*

The obvious disadvantage of using $I_n + WY$ is the extra workspace necessary for storing the $n \times k$ matrix Y instead of the $k \times k$ matrix T.

As an immediate application of compact WY representations we obtain a blocked algorithm for the computation of the orthogonal factor Q from a reduction to Hessenberg form, see Algorithm 1.23. The matrix Q is a 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.

Algorithm 1.36.
Input: *Householder matrices $H_2(x_1), H_3(x_2), \dots, H_{n-1}(x_{n-2})$. Integers N and n_b so that $n - 2 = N \cdot n_b$.*
Output: *An orthogonal matrix $Q = H_2(x_1) \cdots H_{n-1}(x_{n-2})$.*

```
FOR $p = N, N-1, \dots, 1$
  Set $s = (p-1)n_b + 1$.
  Apply the construction given in the proof of Theorem 1.34 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 $Q(s:n,:) \leftarrow Q(s:n,:)(I_n + WTW^T)$.
END FOR
```

In this algorithm, implemented as LAPACK routine `DORGHR`,

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

flops are required to generate the compact WY representations, while

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

flops are necessary to apply them to the matrix Q. On the other hand, the unblocked algorithm, based on calls to `DLARF`, requires $\frac{4}{3}n^3$ 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.

5.2 Block Hessenberg Reduction

Developing a block algorithm for the reduction to Hessenberg form, Algorithm 1.23, is complicated by dependencies between the individual Householder matrices. To resolve these dependencies, Dongarra, Sorensen and Hammarling [87] proposed to maintain a relationship of the form

$$A^{(k)} = \tilde{A}^{(k)} + WX^T + YW^T, \tag{1.50}$$

where $A^{(k)}$ is the partly reduced matrix obtained after k loops of Algorithm 1.23 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 1.23

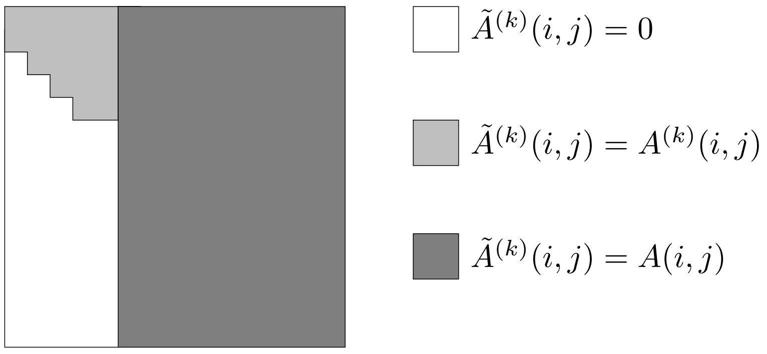

$\tilde{A}^{(k)}(i,j) = 0$
$\tilde{A}^{(k)}(i,j) = A^{(k)}(i,j)$
$\tilde{A}^{(k)}(i,j) = A(i,j)$

Figure 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.*

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 (1.50), a relationship of the form

$$A^{(k)} = (I + WTW^T)^T(\tilde{A}^{(k)} + Y\tilde{W}^T) \tag{1.51}$$

is maintained, where $I + WTW^T$ is the compact WY representation of the first k Householder matrices employed in Algorithm 1.23. The matrix $\tilde{W}$ equals W with the first k rows set to zero. The following algorithm produces (1.51) and is implemented in the LAPACK routine DLAHRD.

Algorithm 1.37 (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 (1.51). The matrix A is overwritten by $\tilde{A}^{(k)}$.*

```
$T \leftarrow [],\quad W \leftarrow [],\quad Y \leftarrow []$
FOR $j \leftarrow 1,\dots,k$
    IF $j > 1$ THEN
        % Update $j$th column of $A$.
        $A(:,j) \leftarrow A(:,j) + YV(j,:)^T$
        $A(:,j) \leftarrow A(:,j) + WT^TW^TA(:,j)$
    END IF
    Compute $j$th 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
```

After Algorithm 1.51 has been applied, the matrix $A^{(k)}$ in (1.51) is computed using level 3 BLAS operations. An analogous procedure is applied to columns $k+1,\dots,2k$ of $A^{(k)}$, which yields the matrix $A^{(2k)}$ having the first $2k$ columns

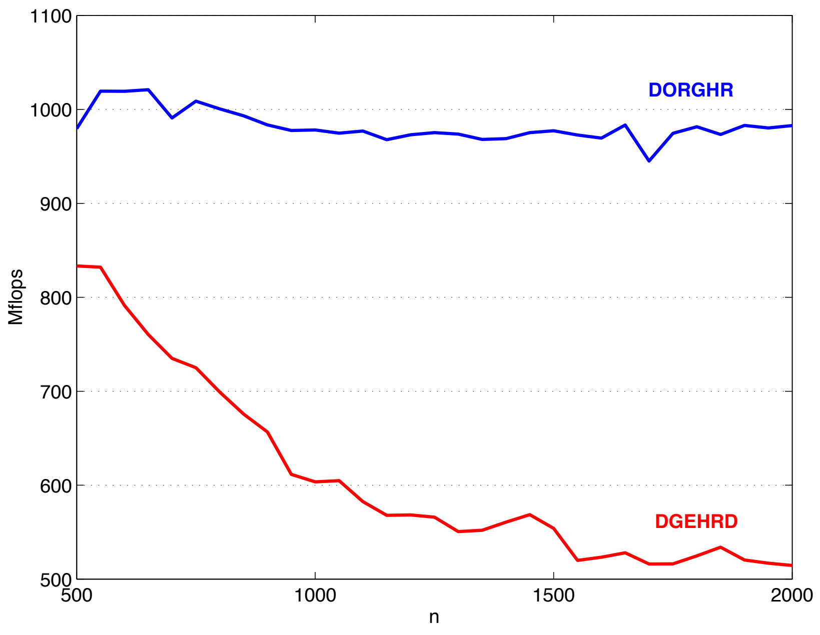


Figure 1.7. *Measured megaflops per second for* `DGEHRD` *and* `DORGHR` *applied to random matrices of order* $500, 550, \dots, 2000$.

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.

A two-stage approach, quite similar to the block algorithms for Hessenberg-triangular reduction described in Section 5.1, Chapter 2, has already been suggested by Bischof and Van Loan [42]. 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 [155, Sec. 2.4.3].

5.3 Multishifts and Bulge Pairs

Early attempts to improve the performance of the QR algorithm focused on using shift polynomials of high degree [13], 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 [89, 255, 256]. To explain this phenomenom, the notion of bulge pairs, introduced by Watkins [255, 256], is now briefly described.

Assume that the implicit shifted QR iteration with m shifts, Algorithm 1.25, 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.[4]

Theorem 1.38 ([256]). *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 26. Let $H^{(j)}$ denote the updated matrix H after the bulge has been chased $(j-1)$ steps. The bulge resides in the submatrix

$$B_j = H^{(j)}(j+1 : j+m+1, j : j+m+1), \tag{1.52}$$

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

Theorem 1.39 ([256]). *The shifts $\sigma_1, \ldots, \sigma_m$ are the finite eigenvalues of the jth* 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 (1.52) refers to entries outside of $H^{(j)}$. Such a situation is shown in (1.44). This issue can be resolved; by adding virtual rows and columns to the matrix $H^{(j)}$, see [256], Theorem 1.39 can be extended to $j > n - m - 1$.

Theorem 1.39 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. Several numerical experiments conducted in [256] 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 by 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 [256], it is intuitively clear that this sensitivity may affect the performance severely. Note that this does not 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. Figure 1.8

[4]For the definition of eigenvalues of matrix pair, the reader is referred to Section 1 in the next chapter.

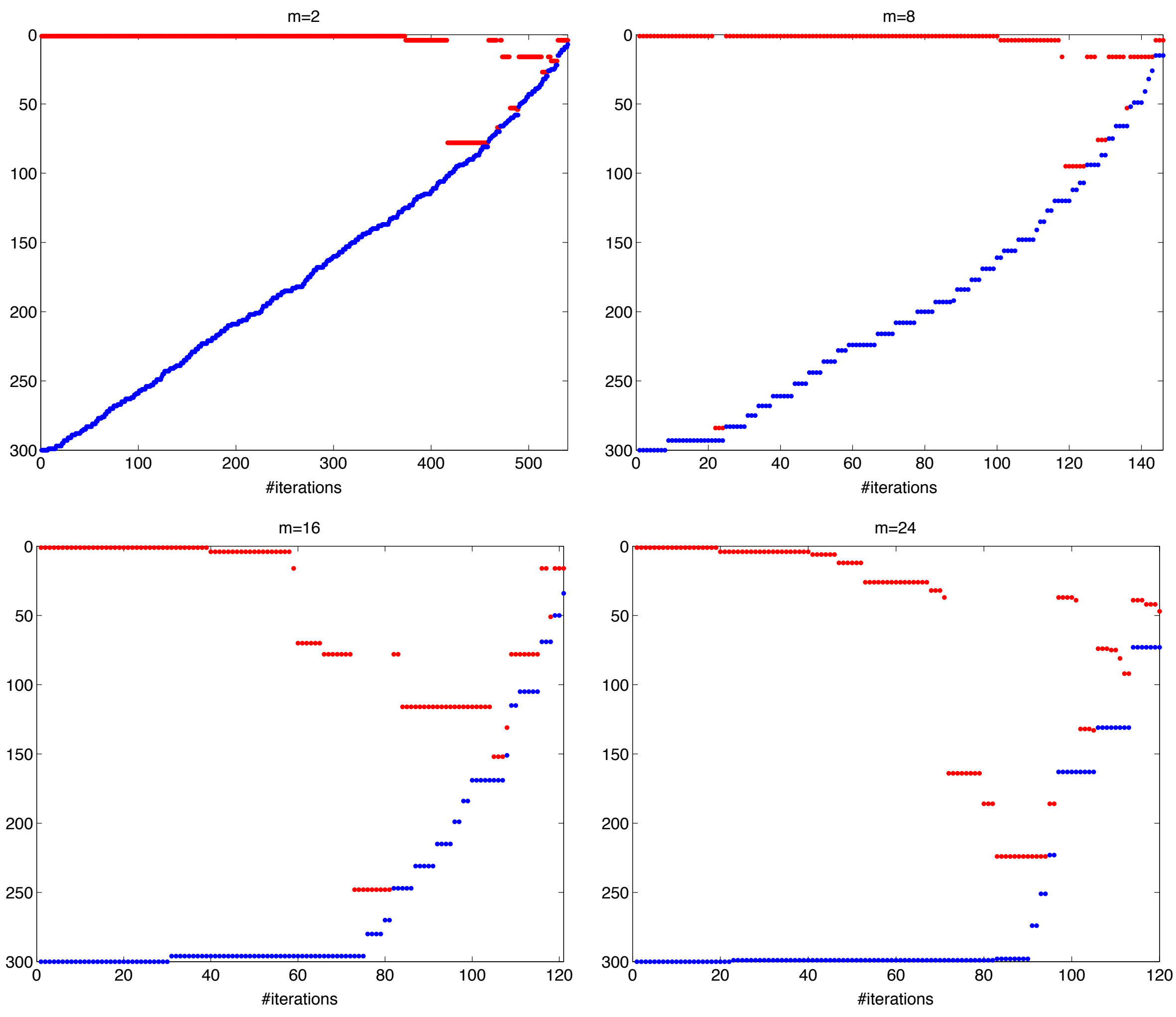


Figure 1.8. *Lower (red dots) and higher (blue dots) indices of the active submatrices that have order larger than m during the implicit shifted QR iteration with m Francis shifts.*

illustrates this phenomenom for matrices generated by 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). Note that the QR algorithm requires about $7.4 \cdot 10^8$ flops for $m = 24$, while it requires only 2.9×10^8 flops for $m = 2$.

5.4 Connection to Pole Assignment

Although Watkins [256] provides convincing numerical experiments for the shift blurring effects described in the previous section, there has been no explanation why the bulge pairs are getting so sensitive as m increases. In this section, we connect the computation of x, a nonzero multiple of the first column of the shift polynomial, to pole assignment, see Section 1.3 in Appendix A, resulting in a semi-heuristic explanation for the sensitivity of the initial bulge pair.

First, let us assume that x is normalized so that $x_{m+1} = 1$. This assumption is justified, since $x_{m+1} = \alpha \prod_{i=1}^{m+1} h_{i+1,i} \neq 0$ is implied by the fact that H is in unreduced Hessenberg form. By applying a simple equivalence transformations to the initial bulge pair (B_0, N), Theorem 1.38 shows that the shifts $\sigma_1, \dots, \sigma_m$ are the eigenvalues of the matrix

$$\begin{aligned} C &= H(1:m,1:m) - h_{m+1,m}x(1:m)e_m^T \qquad (1.53) \\ &= \begin{bmatrix} h_{11} & h_{12} & \dots & h_{1,m-1} & h_{1m} - h_{m+1,m}x_1 \\ h_{21} & h_{22} & \dots & h_{2,m-1} & h_{2m} - h_{m+1,m}x_2 \\ 0 & h_{32} & \dots & h_{3,m-1} & h_{3m} - h_{m+1,m}x_3 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & h_{m,m-1} & h_{mm} - h_{m+1,m}x_m \end{bmatrix}. \end{aligned}$$

Next, consider the single-input control system

$$\dot{z}(t) = H(1:m,1:m)^T z(t) - (h_{m+1,m}e_m)u(t) \qquad (1.54)$$

with state vector $z(\cdot)$ and input $u(\cdot)$. The linear state feedback $u(t) = x(1:m)z(t)$ yields the closed-loop matrix C^T, where C is defined as in (1.53). Hence, the feedback vector $x(1:m)$ places the poles of the open loop system (1.54) to $\sigma_1, \dots, \sigma_m$. Since $x(1:m)$ is uniquely defined by this property, we obtain the following connection:

Any pole assignment 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 assignment algorithm [244]. A not-so-serious application is the expression of the QL iteration [91], a permuted version of the QR iteration, in three lines of MATLAB code, using functions of the MATLAB Control Toolbox [171]:

```
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 plenty of poles in a single-input problem is often very ill-conditioned [118, 123] 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 [176, 177]. Assume that the input data of the pole assignment problem – $A = H(1:m,1:m)^T, b = h_{m+1,m}e_m$ and $\sigma_1, \dots, \sigma_m$ – is perturbed by sufficiently small perturbations $\triangle A$, $\triangle b$ and $\triangle\sigma_1, \dots, \triangle\sigma_m$. Set

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

and let $\hat{f}$ be the feedback vector defined by the perturbed problem. Then, it was shown in [177, Thm. 1.1] that the eigenvalues $\hat{\sigma}_1, \dots, \hat{\sigma}_m$ of $A - b\hat{f}^T$ satisfy

$$|\hat{\sigma}_i - \sigma_i| \le \left(1 + \|G\|_2\|G^{-1}\|_2\sqrt{1 + \|\hat{f}\|_2}\right)\varepsilon + \mathcal{O}(\varepsilon^2), \qquad (1.55)$$

where G is the eigenvector matrix of the closed loop matrix $C^T = A + bf^T$, normalized so that all columns of G have unit norm. Although (1.55) is only an upper bound, numerical experiments in [176, 177] suggest that (1.55) catches the qualitative behavior of the maximal eigenvalue error rather well.

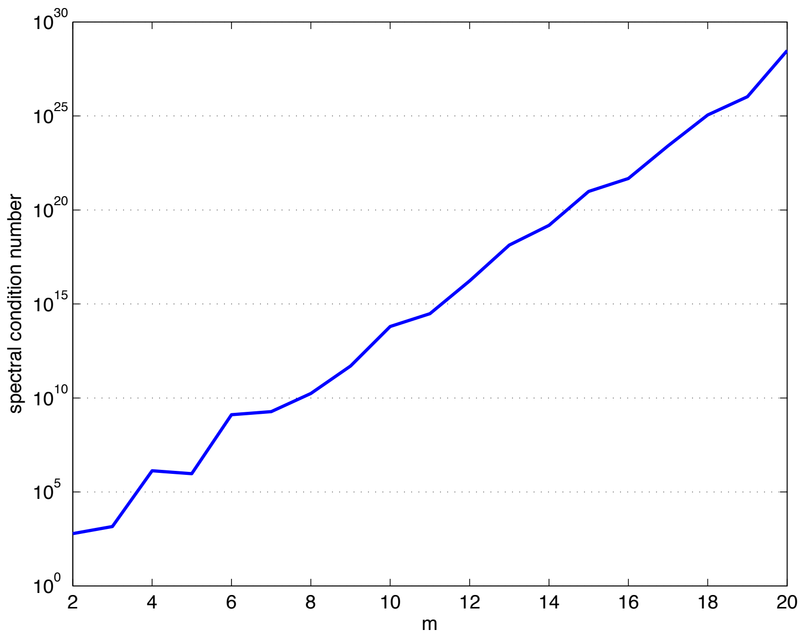


Figure 1.9. *Spectral condition numbers of the closed loop matrix C^T as defined in (1.53).*

It is especially the presence of the spectral condition number $\|G\|_2\|G^{-1}\|_2$ in (1.55) that is worrisome. Often, this term grows rapidly with m, even exponential growth can be proven for some cases [176, 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 (1.53), for a matrix H generated by the MATLAB command `hess(rand(250))` Figure 1.9 displays the spectral condition number of C^T for $m = 2, \dots, 15$, clearly exhibiting exponential growth.

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 [50, 89, 121, 141, 155, 156, 254]. In this section, we describe a slight extension of a variant which was developed independently by Braman, Byers and Mathias [50] as well as by Lang in his thesis [155].

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 H. Second, the whole chain at once 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.

Note that the only aspect in which our algorithm differs from the algorithms

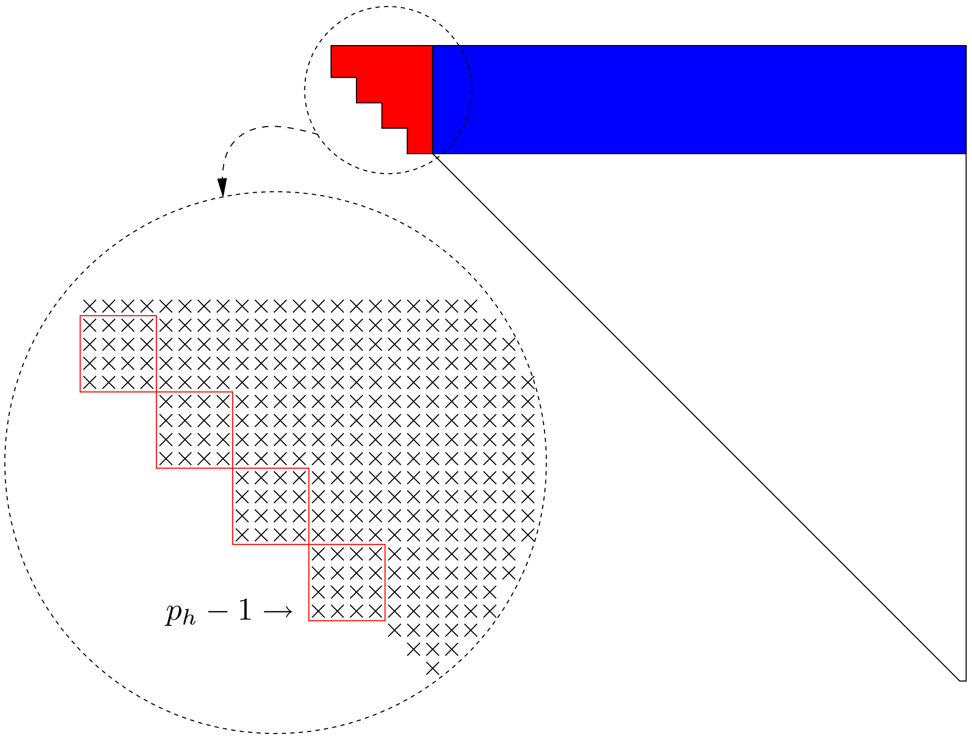


Figure 1.10. *Introducing a chain of $m/n_s = 4$ tightly coupled bulges, each of which contains $n_s = 3$ shifts.*

described in [50, 155] is that the latter are restricted to $n_s = 2$.

Introducing a chain of bulges

Given a set of m Francis shifts, we partition this set into subsets $\Sigma_1, \Sigma_2, \dots, \Sigma_{m/n_s}$. Each Σ_j contains n_s shifts and is closed under complex conjugation. We apply the implicit 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 that only this submatrix (painted red in Figure 1.10) must be updated during the bulge chasing process. To update the remaining part of H (painted blue), all employed orthogonal transformations are accumulated into a $p_h \times p_h$ matrix U. This enables us to use matrix-matrix multiplications:

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

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

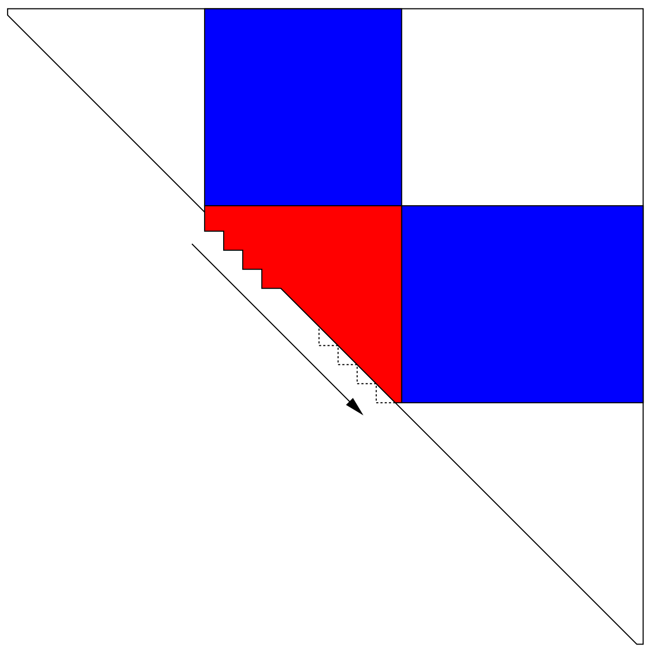


Figure 1.11. *Chasing a chain of $m/n_s = 4$ tightly coupled bulges.*

Figure 1.11. This is done in bottom-to-top order so that 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:

$$\begin{aligned} H(p_l : p_h + k, (p_h + 1) : n) &\leftarrow U^T \cdot H(p_l : p_h + k, (p_h + 1) : n), \\ H(1 : p_l - 1, p_l : p_h + k) &\leftarrow H(1 : p_l - 1, p_l : p_h + k) \cdot U. \end{aligned}$$

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

$$U = \begin{bmatrix} 1 & 0 & 0 \\ 0 & U_{11} & U_{12} \\ 0 & U_{21} & U_{22} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \square & \text{◺} \\ 0 & \text{◹} & \square \end{bmatrix}, \tag{1.56}$$

i.e., the matrices $U_{12} \in \mathbb{R}^{l_1 \times l_2}$ and $U_{21} \in \mathbb{R}^{l_2 \times l_1}$, 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.

The rare event of a zero subdiagonal entry between two consecutive bulges during the bulge chasing process is called a "vigilant" deflation [254]. 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 [50]. Note that it is not necessary to take care of vigilant deflations caused by small non-zero subdiagonal entries [255].

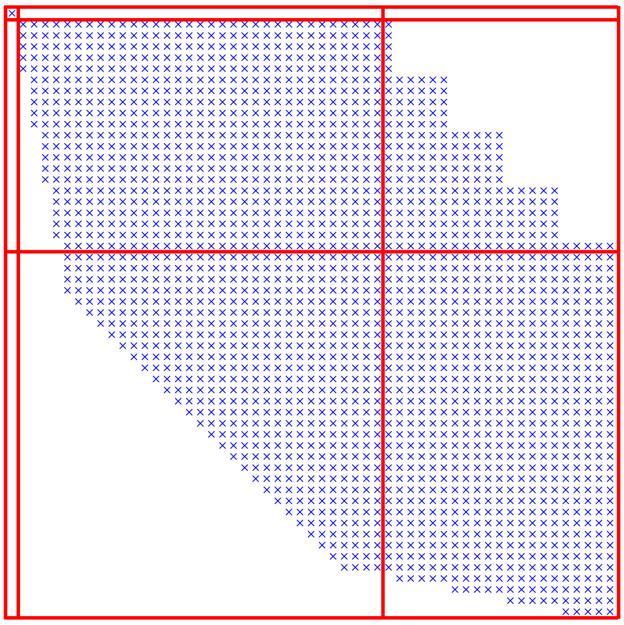


Figure 1.12. *Structure of the transformation matrix U for chasing a chain of $m/n_s = 5$ bulges, each of which contains $n_s = 4$ shifts, $k = 30$ steps.*

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

The described multishift QR algorithm with tightly coupled tiny bulges has been implemented in a Fortran 77 routine called `MTTQR`. Although this routine generally requires more flops than a doubleshift QR algorithm, see [50], it can be expected 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 [12], 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 in each iteration) and k (number of steps a chain of bulges is chased before off-diagonal parts are updated),

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 [12].*

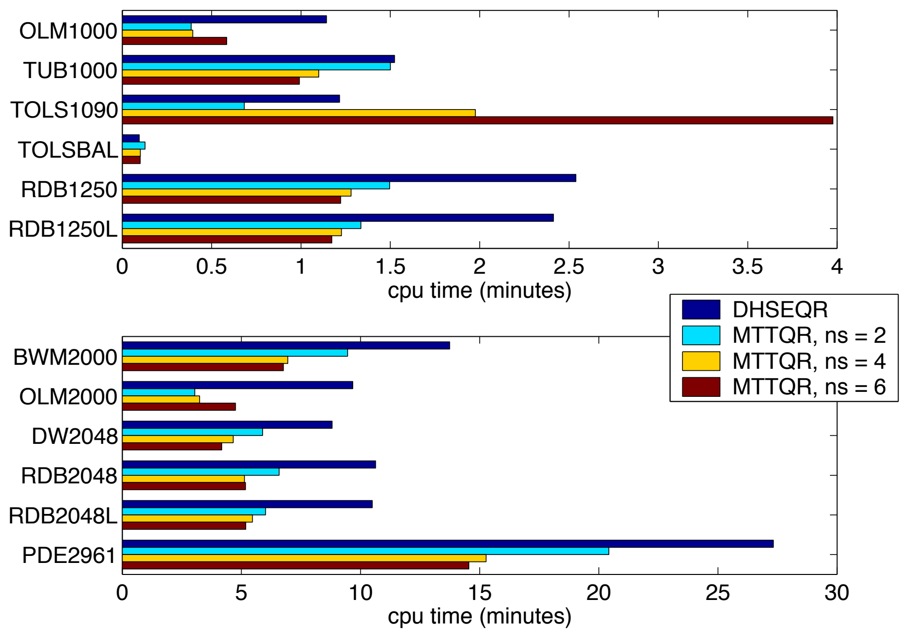


Figure 1.13. *Execution times for* `DHSEQR` *and* `MTTQR`*, the tiny-bulge multi-shift QR algorithm with $n_s \in \{2, 4, 6\}$ shifts per bulge, applied to matrices from the test matrix collection [12].*

we followed the recommendations given in [50]:

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

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

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. 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.

6 Advanced Deflation Techniques

There have been few attempts to modify the deflation criteria described in Section 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 [51]. In this section, we briefly describe their approach and show how it can be adapted to exploit linear convergence phenomena taking place at the top left part of a Hessenberg matrix.

6.1 Aggressive Early Deflation

To describe the 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 $\mathbf{u} \cdot \|H\|_F$. In the classical deflation strategies described in Section 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 the restriction from E gives the potential for more deflations. The price we have to pay is the extra effort for reducing $H + E$. In order to avoid that these additional expense dominates the computational time required by 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

$$H = \begin{array}{c} \\ {\scriptstyle n-w-1} \\ {\scriptstyle 1} \\ {\scriptstyle w} \end{array} \begin{array}{c} \begin{array}{ccc} {\scriptstyle n-w-1} & {\scriptstyle 1} & {\scriptstyle w} \end{array} \\ \left[\begin{array}{ccc} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ 0 & H_{32} & H_{33} \end{array} \right] \end{array}, \tag{1.57}$$

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

$$E = \begin{array}{c} \\ {\scriptstyle n-w-1} \\ {\scriptstyle 1} \\ {\scriptstyle w} \end{array} \begin{array}{c} \begin{array}{ccc} {\scriptstyle n-w-1} & {\scriptstyle 1} & {\scriptstyle w} \end{array} \\ \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & E_{32} & E_{33} \end{array} \right] \end{array}, \tag{1.58}$$

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 1.23 to $Q_1 \cdot (H_{33} + E_{33}) \cdot Q_1$ yields an orthogonal matrix Q_2 so that with $Q = Q_1 Q_2$ the matrix

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

is in Hessenberg form. We call E a *reducing perturbation* if some subdiagonal entry of $\tilde{H}$ is zero, i.e., one or more eigenvalues of $\tilde{H}$ are deflated. The following well-known lemma relates reducing perturbations to the concept of controllability, see Section 1.2 in Appendix A.

Lemma 1.40. *Let the matrices H and E have the form displayed in (1.57) and (1.58), 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 $Q^T(H_{33} + E_{33})Q$ to be an unreduced Hessenberg matrix and $Q^T(H_{32} + E_{32})$ to be a nonzero vector. □

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 3.2 in Chapter A. Braman, Byers and Mathias propose an efficient algorithm, also described in that section, in the context of the QR 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 $T_{33} = \left[\begin{smallmatrix} \tilde{T}_{11} & \tilde{T}_{12} \\ 0 & \tilde{T}_{22} \end{smallmatrix}\right]$ so that $\tilde{T}_{22}$ is either a real eigenvalue or a 2×2 block containing a pair of complex conjugate eigenvalues. The vector $U^T H_{32} = \left[\begin{smallmatrix} \tilde{s}_1 \\ \tilde{s}_2 \end{smallmatrix}\right]$ is correspondingly partitioned. Then the matrices $E_{32} = -U \left[\begin{smallmatrix} 0 \\ \tilde{s}_2 \end{smallmatrix}\right]$ and $E_{33} = 0$ correspond to a reducing perturbation E of the form (1.58). 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 $\mathbf{u} \cdot \|H\|_F$, otherwise the backward stability of the QR algorithm is lost. A more restrictive condition, in the spirit of the neighborwise deflation criterion described in Section 3.4, is given by

$$\|\tilde{s}_2\|_\infty \leq \sqrt{|\det(\tilde{T}_{22})|} \tag{1.59}$$

A variety of other criteria can be found in [51].

If the current ordering of the matrix T_{33} in real Schur form fails to fulfill the criterion (1.59), 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 7 below. If a reducing perturbation satisfying (1.59) 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

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

where T_{11}, T_{22} are in real Schur form and the entries of the vector s_2 satisfy criteria of the form (1.59). 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 [50].

Numerical results

The advantage of aggressive early deflation is best illustrated by numerical experiments. Some theoretical justification for the observed effect that this type of deflation is so much more effective than conventional deflation strategies can be found in [51]. We repeated the numerical experiments from Section 5.5 by combining aggressive early deflation with the tiny-bulge multishift QR algorithm described therein. This algorithm has been implemented in a Fortran 77 routine called `ATTQR`. Our choice of parameters for `ATTQR` is based on recommendations given in [51]. The number of shifts in each QR iteration was set to

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

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

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 `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%.

6.2 Aggressive Early Deflation at Top Left Corner

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 during a number of QR iterations, then the convergence theory described in Section 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.

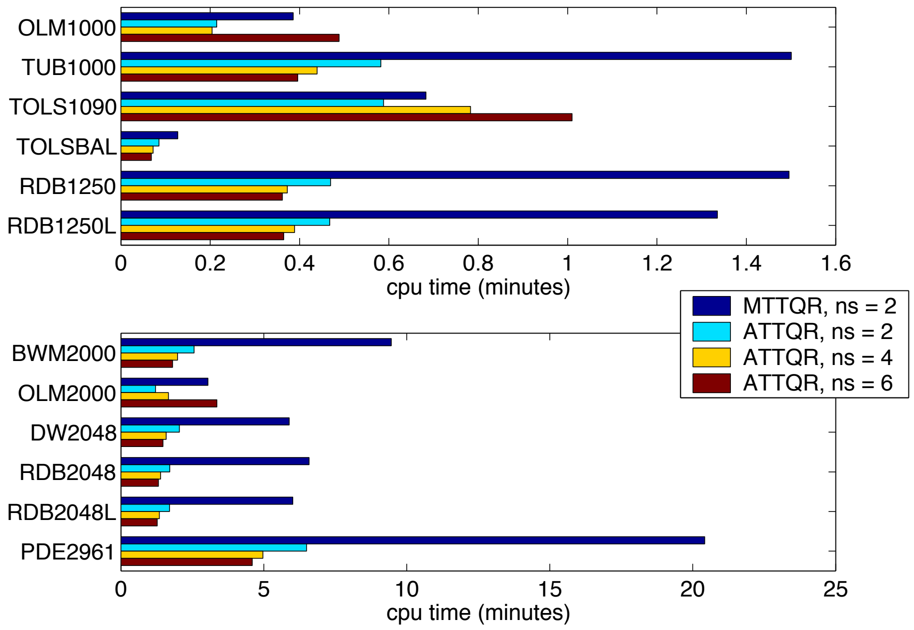


Figure 1.14. *Execution times for* ATTQR, *the tiny-bulge multishift QR algorithm with aggressive early deflation and $n_s \in \{2,4,6\}$ shifts per bulge, applied to matrices from the test matrix collection [12].*

Applying aggressive deflation to this part is simple. If F denotes the flip matrix, then the procedure described in Section 6.1 applied to FH^TF yields an orthogonal matrix Q so that

$$(Q \oplus I_{n-w})^T H (Q \oplus I_{n-w}) = \begin{array}{c} \\ d \\ w-d \\ 1 \\ n-w-1 \end{array} \begin{array}{c} \begin{array}{cccc} d & w-d & 1 & n-w-1 \end{array} \\ \left[\begin{array}{cccc} T_{11} & T_{12} & \tilde{H}_{13} & \tilde{H}_{13} \\ 0 & T_{22} & \tilde{H}_{23} & \tilde{H}_{24} \\ s_1 & s_2 & H_{33} & H_{34} \\ 0 & 0 & H_{43} & H_{44} \end{array}\right] \end{array},$$

where the matrices T_{11}, T_{22} are in real Schur form and s_1 is a row vector whose elements satisfy inequalities of the form (1.59) with respect to diagonal blocks of T_{11}. Setting s_1 to zero lets the d eigenvalues contained in T_{11} deflate.

Of course, as deflations at the top left corner depend on linear convergence phenomena, they cannot be expected to appear as frequently as deflations at the bottom right corner. Usually, a large number of QR iterations are necessary before the shifts start settling down and allow this kind of convergence to happen. To measure possible gains of performance, we repeated the numerical experiments from Section 6.1 with aggressive early deflation at the bottom right and top left corners applied after each tiny-bulge multishift QR iteration. The cpu times displayed by the cyan, yellow and red bars in Figure 1.15 do not include superfluous computing time spent for tests on early deflations at the top left corner that did not result in deflated eigenvalues. The transparent bars in the background include this extra computing time. Only for the matrix BWM2000 a substantial improvement (47%), gained through aggressive early deflation at the top left corner, can be observed. Merely minor improvements or even deterioration of performance can be observed for all other matrices. This shows that aggressive early deflation at the top left corner has to applied with care. Extra research is necessary to find cheap and reliable

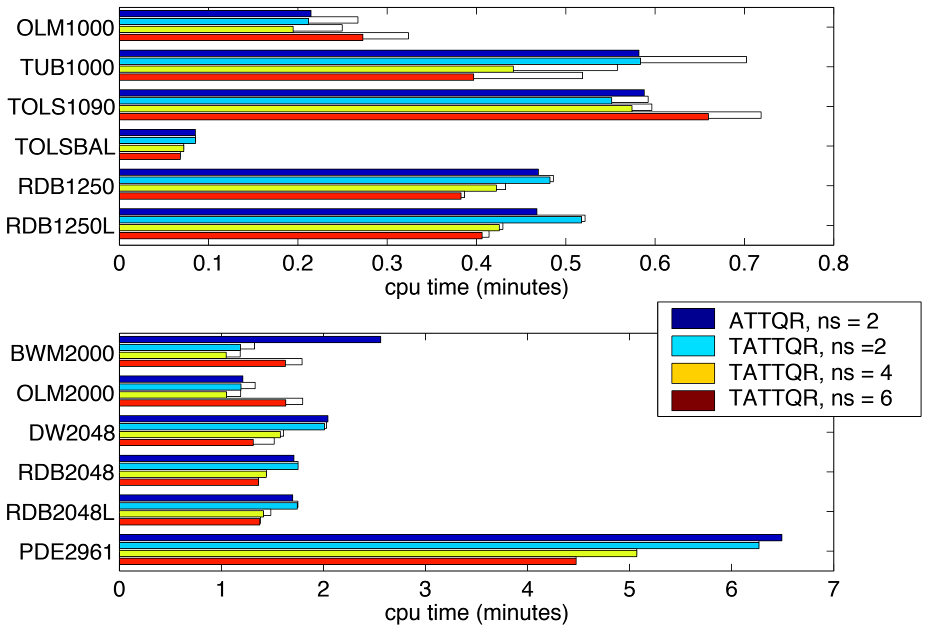


Figure 1.15. *Execution times for* `TATTQR`*, the tiny-bulge multishift QR algorithm with aggressive early deflation at bottom right and top left corners,* $n_s \in \{2,4,6\}$ *shifts per bulge, applied to matrices from the test matrix collection [12].*

estimates for the number of eigenvalues expected to get deflated by aggressive early deflation.

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:

$$Q^TAQ = T = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1m} \\ 0 & T_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_{m-1,m} \\ 0 & \cdots & 0 & T_{mm} \end{bmatrix}, \tag{1.60}$$

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 $j \le m$, 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 $\Lambda_s \subset \lambda(A)$, closed under complex conjugation. This section is about computing from a given Schur decomposition (1.60) a reordered Schur decomposition of the form

$$(Q\tilde{Q})^TA(Q\tilde{Q}) = \tilde{T} = \begin{bmatrix} \tilde{T}_{11} & \tilde{T}_{12} & \cdots & \tilde{T}_{1m} \\ 0 & \tilde{T}_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \tilde{T}_{m-1,m} \\ 0 & \cdots & 0 & \tilde{T}_{mm} \end{bmatrix}, \tag{1.61}$$

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

In the following two subsections we describe the reordering method by Bai and Demmel [14], which is implemented in LAPACK. The third subsection describes a new, more efficient method for eigenvalue reordering, inspired by the tiny-bulge multishift QR algorithm discussed in Section 5.5. Numerical results show that this new method can achieve remarkable performance improvements in comparison with the LAPACK implementation, largely independent of the distribution of selected eigenvalues over the block diagonal of T.

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

$$Q^T \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} Q = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ 0 & \tilde{A}_{22} \end{bmatrix}, \quad \begin{array}{l} \lambda(A_{11}) = \lambda(\tilde{A}_{22}), \\ \lambda(A_{22}) = \lambda(\tilde{A}_{11}), \end{array} \tag{1.62}$$

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} although this is, strictly speaking, a misnomer, since 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 [221] 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, the QR algorithm with the eigenvalues of A_{11} as shifts is applied. Numerical examples reported in [14, 86] demonstrate that this algorithm occasionally fails to converge or produces a block triangular matrix that does not correspond to the intended swapping.

Based on earlier work by other authors [67, 86, 182, 197], Bai and Demmel [14] developed a direct swapping algorithm. Assuming that $\lambda(A_{11}) \cap \lambda(A_{22}) = \emptyset$, it first computes the solution of the Sylvester equation

$$A_{11}X - XA_{22} = \gamma A_{12}, \tag{1.63}$$

where $\gamma \le 1$ is a scaling factor to prevent from possible overflow in X. This yields a 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 $Q = \left[\begin{smallmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{smallmatrix}\right]$ 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 (1.63) be obtained by applying Gaussian elimination with complete pivoting to the Kronecker product formulation of (1.63). 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

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

and ρ is a small constant of order $\mathcal{O}(1)$ [14]. 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 small $\mathrm{sep}(A_{11}, A_{22})$ only very rarely imply 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 $Q^T \left[\begin{smallmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{smallmatrix}\right] Q$ exceeds all elements of $20\mathbf{u} \cdot A_{ij}$ in magnitude, then the swapping is rejected. It was argued in [14] that such a rejection may only happen if the eigenvalues of A_{11} and A_{22} are so close that they are numerically indistinguishable.

7.2 Reordering

Having the direct swapping algorithm on hand, we can use bubble sort to reorder a selected set of eigenvalues to the top left corner of a given real Schur form.

Algorithm 1.41 (Reordering real Schur decomposition).

Input: *A matrix $T \in \mathbb{R}^{n\times n}$ in real Schur form (1.60), 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 $Q \cdot \tilde{Q}$, respectively.*

```
$j \leftarrow 0$
FOR $i \leftarrow 1, \dots, m$
  IF $\lambda(T_{ii}) \subset \Lambda_s$ THEN $j \leftarrow j+1$, $\mathrm{select}(j) \leftarrow i$ END IF
END FOR
$\mathrm{top} \leftarrow 0$
FOR $l \leftarrow 1, \dots, j$
  FOR $i \leftarrow \mathrm{select}(l), \mathrm{select}(l)-1, \dots, \mathrm{top}+1$
    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
  $\mathrm{top} \leftarrow \mathrm{top}+1$
END FOR
```

This algorithm is implemented in the LAPACK routine `DTRSEN`, which also provides (estimates of) condition numbers for the eigenvalue cluster Λ_s and the

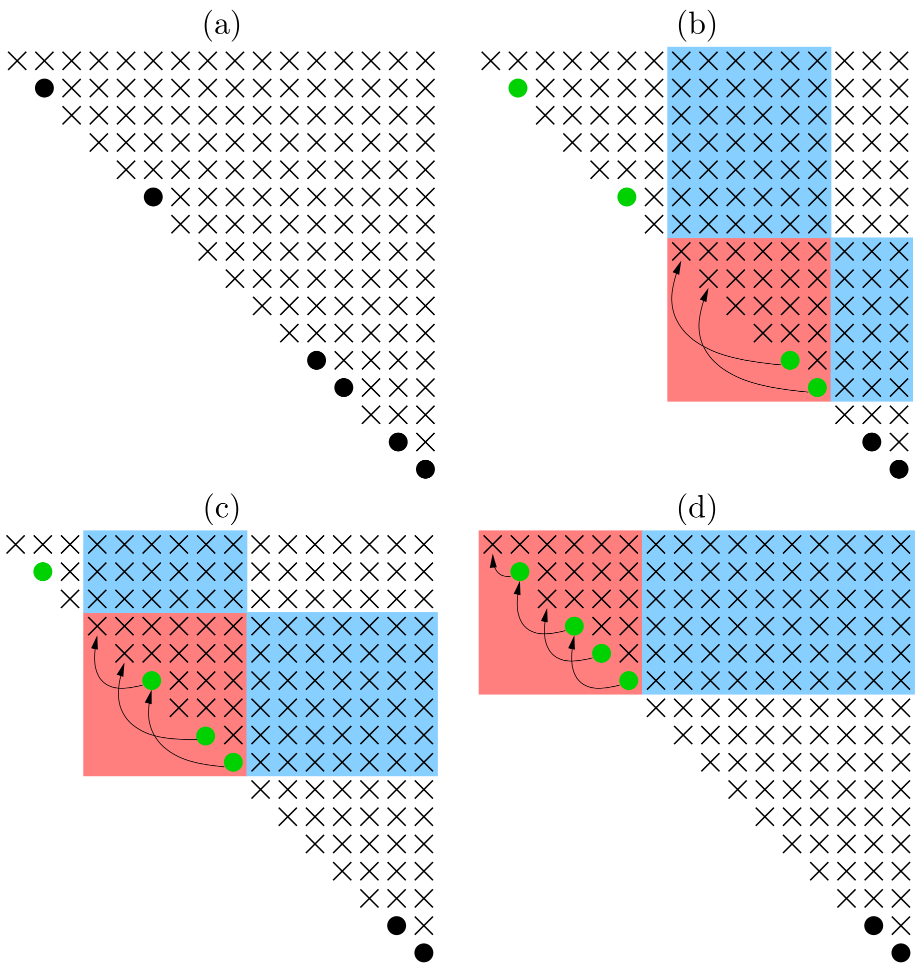


Figure 1.16.

corresponding invariant subspace. If Λ_s contains k eigenvalues, then Algorithm 1.41 requires $\mathcal{O}(kn^2)$ flops. The exact computational cost depends on the distribution of selected eigenvalues over the block diagonal of T.

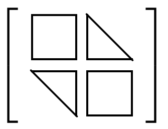
7.3 Block Algorithm

Each outer loop of Algorithm 1.41 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 1.41 can be improved using a block algorithm similar to the tiny-bulge multishift QR algorithm, see Section 5.5. The basic idea of this block algorithm is best explained by an example.

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 1.16 (a). We activate the eigenvalues in the $ev = 4$ upper-left-most positions $(2, 6, 12, 13)$, those are the green disks in Figure 1.16 (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 pink area in Figure 1.16 (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 1.16 (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; we feel that it will provide no further insight. Note, however, that the determination of the correct window positions in the presence of 2×2 diagonal blocks is a rather complicated process. For all technical details, the reader is referred to the Fortran 77 implementation `BLKORD` of the delineated block reordering algorithm, see Appendix B. `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 blue in Figure 1.16. In order to maintain locality of the memory reference pattern, all rows are updated in stripes of n_b columns. It there are sufficiently 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:

$$U = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} = \qquad (1.64)$$



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 Results

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` so that all eigenvalues have moderate condition numbers. The portion of selected eigenvalues was $d \in \{5\%, 25\%, 50\%\}$. To demonstrate that the performance improvement gained from using `BLKORD` is 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 $(d \cdot n) \times (d \cdot n)$ 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 $5/2 \cdot ev$ 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. Since all eigenvalues of T are real, it is possible to increase the efficiency by using the block triangular structure (1.64) of U. Timings for which this structure has been exploited are listed in columns with heading 'triu' in Table 1.2, while columns with heading 'full' correspond to timings for which the structure of U has not been exploited.

			Update T			Update T and Q		
n	sel.	distr.	`DTRSEN`	`BLKORD`		`DTRSEN`	`BLKORD`	
				full	triu		full	triu
500	5%	random	0.19	0.05	0.05	0.25	0.08	0.09
500	5%	bottom	0.24	0.07	0.07	0.34	0.13	0.12
500	25%	random	0.55	0.16	0.15	0.75	0.26	0.25
500	25%	bottom	0.86	0.29	0.28	1.24	0.50	0.45
500	50%	random	0.51	0.20	0.20	0.76	0.34	0.33
500	50%	bottom	1.06	0.38	0.37	1.54	0.63	0.58
1000	5%	random	2.23	0.34	0.34	2.87	0.62	0.60
1000	5%	bottom	2.98	0.50	0.47	4.03	0.88	0.78
1000	25%	random	6.51	0.95	0.91	8.40	1.71	1.57
1000	25%	bottom	11.65	1.94	1.80	15.83	3.39	3.04
1000	50%	random	7.60	1.31	1.21	10.08	2.34	2.10
1000	50%	bottom	15.56	2.53	2.34	21.23	4.49	4.03
1500	5%	random	7.92	1.00	0.96	9.46	1.77	1.69
1500	5%	bottom	11.36	1.61	1.51	14.21	2.91	2.64
1500	25%	random	25.02	3.01	2.75	30.53	5.42	4.88
1500	25%	bottom	45.12	6.09	5.58	56.64	11.04	10.20
1500	50%	random	28.11	4.02	3.64	35.93	7.38	6.55
1500	50%	bottom	60.47	7.98	7.23	75.79	14.64	12.93

Table 1.2. *Performance results in seconds for unblocked (*`DTRSEN`*) and blocked (*`BLKORD`*) reordering of an n-by-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`. In some cases, e.g. the reordering of 375 randomly selected eigenvalues in a 1500×1500 matrix, this ratio is as low as 11%. Further numerical experiments revealed that qualitatively similar results are obtained if some or all eigenvalues of T are complex. To test the numerical reliability of `BLKORD`, we measured the orthogonality of Q, $\|Q^T Q - I\|_F$, as well as the residual $\|Q^T T Q - \tilde{T}\|_F / \|T\|_F$ and found all values satisfactorily close to $\mathbf{u}$.

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 [157], see also [106, 158]. In this type of control problems, the dynamical system consists of a state equation given by a second-order differential equation

$$M\ddot{z}(t) + L\dot{z}(t) + Kz(t) = Su(t), \quad z(0) = z_0,\ \dot{z}(0) = z_1, \tag{1.65}$$

and an associated output equation

$$y(t) = Nz(t) + P\dot{z}(t), \tag{1.66}$$

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.

A first-order realization of this problem may 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

$$\dot{x}(t) = \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}L \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ M^{-1}S \end{bmatrix} u(t), \quad x(0) = x_0, \tag{1.67a}$$

$$y(t) = \begin{bmatrix} N & P \end{bmatrix} x(t), \tag{1.67b}$$

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:

$$\textit{Minimize} \quad J(u(\cdot)) = \frac{1}{2}\int_0^\infty \big(y(t)^T y(t) + u(t)^T u(t)\big)\ \mathrm{d}t \quad \textit{subject to (1.67).}$$

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 2 in Appendix A. 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:

$$H = \left[\begin{array}{cc|cc} 0 & I & 0 & 0 \\ -M^{-1}K & -M^{-1}L & 0 & -(M^{-1}S)(M^{-1}S)^T \\ \hline -N^TN & -N^TP & 0 & (M^{-1}K)^T \\ -P^TN & P^TP & -I & (M^{-1}L)^T \end{array}\right]. \tag{1.68}$$

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

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

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

$$T = (Q_1Q_2)^T H (Q_1Q_2)$$

has real Schur form.

[5]For more on Hamiltonian matrices, the reader is referred to Chapter 4.

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 1.42 ([119, 1, 152]). *The example is a model of a string consisting of coupled springs, dashpots, and masses as shown in Figure 1.17. The inputs are two forces, one acting on the left end of the string, the other one on the right end. For*

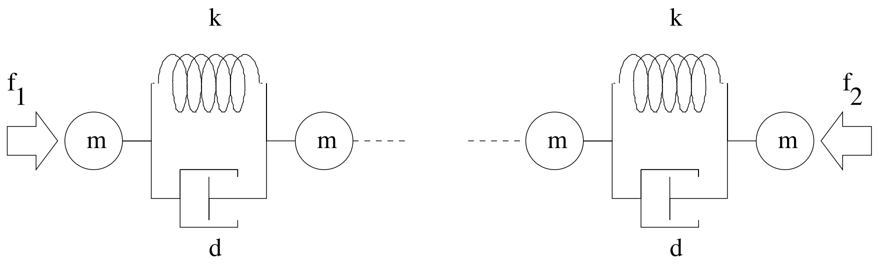


Figure 1.17. *Coupled springs experiment ($k \sim \kappa$, $m \sim \mu$, $d \sim \delta$).*

this problem, the matrices in (1.67) are

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

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

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

We used three different combinations of algorithms 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 state of the art variant of the QR algorithm, which is the tiny-bulge multishift QR algorithm (*`ATTQR`*) with $n_s = 2$ shifts per bulge and aggressive early deflation, see Sections 5.5 and 6.1. We are not aware of any existing implementations for Step 1 and 3 that are more efficient than LAPACK. The third combination uses the extension of* `ATTQR` *to $n_s = 6$, described in Section 5.5, and the newly developed block reordering routine* `BLKORD`*, see Section 7.3. All routine parameters were those used in the numerical experiments of Sections 6.1 and 7.3.*

The obtained performance results are displayed in Figure 1.18. 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

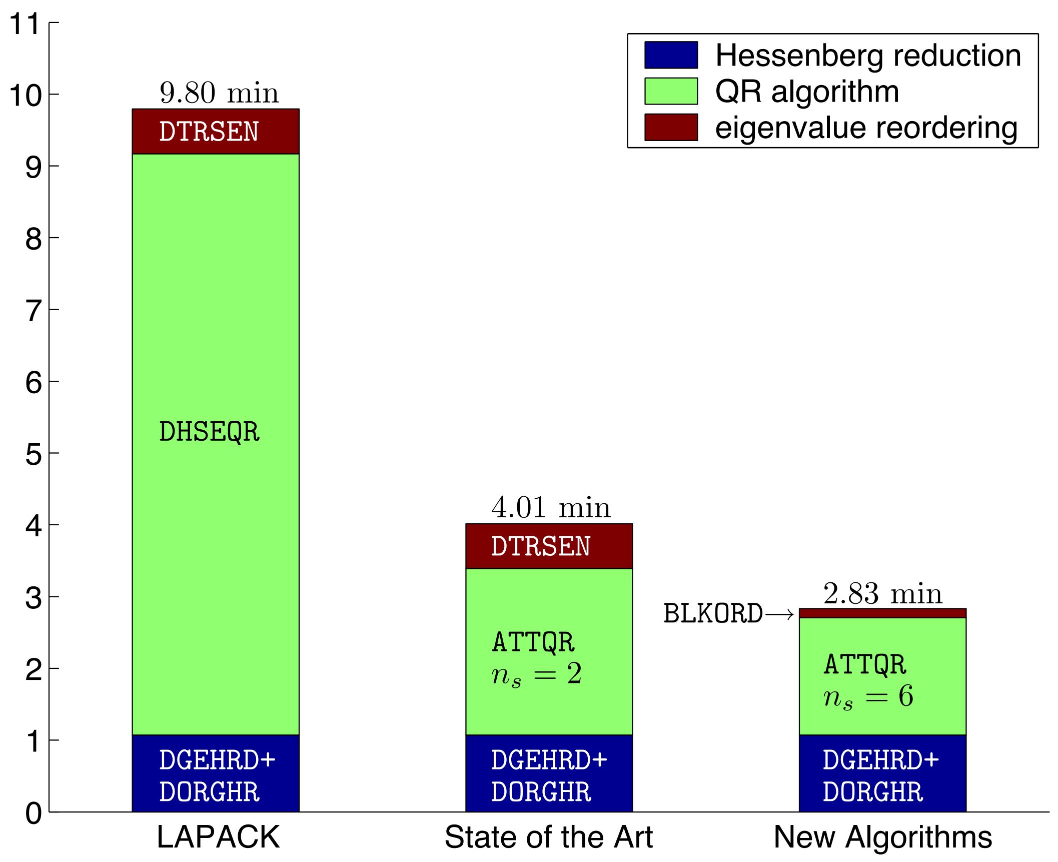


Figure 1.18. *Performance results (in minutes) for the three combinations of algorithms described in Example 1.42.*

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%.

Chapter 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 [179])*

This chapter is about the QZ algorithm, a numerically backward stable method for computing eigenvalues and deflating subspaces of matrix pairs.

In Section 1, we briefly recall basic definitions such as generalized eigenvalue, deflating subspace and generalized Schur decomposition. A more detailed treatment of this subject can be found in, e.g., [226]. Existing perturbation results for generalized eigenvalues and deflating subspaces are summarized in Section 2. The explicit shifted QZ iteration, the basis of the QZ algorithm, is introduced in the beginning Section 3. The remainder of that section is concerned with the other three ingredients of the implicit shifted QZ algorithm, reduction to Hessenberg-triangular form, bulge chasing and deflation. Particular emphasis is placed on the use of Householder matrices in the bulge chasing process, see Section 3.3. Section 4 is concerned with balancing matrix pairs. We describe two known approaches and demonstrate that both are more fragile than the Parlett-Reinsch algorithm for balancing matrices. In Section 5.1, block algorithms for reducing a matrix pair to Hessenberg-triangular form are investigated. In particular, we discuss an algorithm by Dackland and Kågström [74] and suggest certain improvements to this algorithm. Sections 5.2 and 5.3 deal with the transmission of shifts during an implicit shifted QZ iteration and the behavior of infinite eigenvalues. Finally, in Sections 5.5 and 6, we adapt techniques described in the previous chapter to obtain a tiny-bulge multishift QZ algorithm with aggressive early deflation. Some preliminary numerical experiments demonstrate the superior performance of this algorithm.

Contributions in this chapter

The author's contributions to the QZ algorithm are as follows:

- a brief error analysis explaining why opposite Householder matrices do not sacrifice the backward stability of the QZ algorithm, see Section 3.3;

- an algorithm for reducing a matrix pair to block Hessenberg-triangular form being more efficient than a similar, previously suggested algorithm [74], see Section 5.1;
- an investigation of the behavior of infinite eigenvalues under Householder-based implicit QZ iterations, see Section 5.3;
- the development of a tiny-bulge multishift QZ algorithm with aggressive early deflation based on earlier work by Adlerborn, Dackland and Kågström [3], see Sections 5.5 and 6.

The work on the tiny-bulge multishift QZ algorithm with aggressive early deflation is joint work with Bo Kågström and Björn Adlerborn.

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 this 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}. 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 tempts us 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 $(A, B) \cdot X \equiv (AX, BX)$ and $Y \cdot (A_{11}, B_{11}) \equiv (YA_{11}, YB_{11})$. The following example shows how deflating subspaces belonging to complex conjugate pairs of eigenvalues can be constructed.

Example 2.1. *Let $(\alpha, \beta) = (\alpha_1 + \imath\alpha_2, \beta)$ 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 $x = x_1 + \imath x_2$ is a corresponding eigenvector with $x_1, x_2 \in \mathbb{R}^n$, then $\beta Ax = \alpha Bx$ implies*

$$\begin{aligned} 2\beta Ax_1 &= \beta A(x + \bar{x}) = \alpha Bx + \bar{\alpha}B\bar{x} = 2(\alpha_1 Bx_1 - \alpha_2 Bx_2), \\ 2\beta Ax_2 &= \imath\beta A(x - \bar{x}) = \imath\alpha Bx - \imath\bar{\alpha}B\bar{x} = 2(\alpha_2 Bx_1 + \alpha_1 Bx_2), \end{aligned}$$

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

$$y_1 = \frac{1}{\beta}Bx_1, \quad y_2 = \frac{1}{\beta}Bx_2$$

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

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

Not surprisingly, deflating subspaces allow to deflate a generalized eigenproblem into two smaller subproblems, just as invariant subspaces can be used to deflate standard eigenproblems. 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 of $\mathcal{X}, \mathcal{X}^\perp, \mathcal{Y}, \mathcal{Y}^\perp$, respectively. Then $U = [Y, Y_\perp]$ and $V = [X, X_\perp]$ are unitary matrices with

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

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 as displayed on the right hand side of (2.1) is in *generalized block Schur form.* This block triangular matrix pair can subsequently be reduced to a triangular matrix pair, giving rise to a 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 [218]). *Let $(A, B) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$, then there exist orthogonal matrices Q and Z so that $Q^T \cdot (A, B) \cdot Z = (S, T)$, where S is in real Schur form (see Theorem 1.2) and T is upper triangular.*

The purpose of the QZ algorithm is to compute this decomposition. It 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 [132, 134, 238].

2 Perturbation Analysis

In this section we will investigate the influence of a perturbation (E, F) on the generalized eigenvalues and deflating subspaces of (A, B). The exposition will be briefer than for the standard eigenvalue problem; known results will not be proven as the proof techniques are quite similar and not needed later on.

2.1 Spectral Projectors and Dif

Spectral projectors and separation 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 we assume a generalized block Schur decomposition of the form

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

We partition the unitary matrices as $U = [Y, Y_\perp]$ and $V = [X, X_\perp]$ so that $(\mathcal{X}, \mathcal{Y}) = (\operatorname{span}(X), \operatorname{span}(Y))$ is a pair of deflating subspaces having 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 (R_r, R_l) satisfies the following system of matrix equations

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

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 ([218]). *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 $\mathbf{T}_u$:

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

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

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

The associated generalized Sylvester operator is given by

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

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

Example 2.5. *Let* $A_{11} = \left[\begin{smallmatrix} 10^5 & 0 \\ 0 & 10^{-5} \end{smallmatrix}\right], A_{22} = 1, B_{11} = \left[\begin{smallmatrix} 10^5 & 10^5 \\ 0 & 10^{-5} \end{smallmatrix}\right]$ *and* $B_{22} = 0$. *Then*

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

while

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

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

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

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

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

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 [226], largely based on works by Stewart [218, 220]. Further developments in this area can be found in a recent summary by Sun [231] and the references therein.

We start with a result on the perturbation expansion of a generalized eigenvalue.

Theorem 2.6 ([231, Thm. 4.1.1]). *Let the regular matrix pair (A, B) have a simple eigenvalue (α, β) with right and left eigenvectors x and z, respectively, normalized so that $z^HAx = \alpha$ and $z^HBx = \beta$. 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,*

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

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 12 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 well-known distance measure.

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

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

The only perturbation expansion for pairs of deflating subspaces we are aware of is a result by Sun [230, Theorem 3.1.1], who considers one-parameter families of perturbations. The following theorem is a variation of this result.

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*

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

with $(Q_r, Q_l) = \mathbf{T}_l^{-1}(Y_\perp^H E X, Y_\perp^H F X)$ *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 $(\operatorname{span}(\hat{X}),\operatorname{span}(\hat{Y}))$ 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

$$J = \frac{\partial f}{\partial(\hat{X},\hat{Y},\hat{A}_{11},\hat{B}_{11})}\Big|_{(0,0,X,Y,A_{11},B_{11})} = \begin{bmatrix} \multicolumn{2}{c}{\multirow{2}{*}{$\tilde{\mathbf{T}}$}} & -Y & 0 \\ & & 0 & -Y \\ X^H & 0 & 0 & 0 \\ 0 & Y^H & 0 & 0 \end{bmatrix}$$

with the linear matrix operator $\tilde{\mathbf{T}} : (Z_r, Z_l) :\rightarrow (AZ_r - Z_lA_{11}, BZ_r - Z_lB_{11})$. 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

$$J^{-1} = \begin{bmatrix} \multicolumn{2}{c}{\multirow{2}{*}{$\mathbf{S}$}} & X & 0 \\ & & 0 & Y \\ C_{11} & C_{12} & A_{11} & -A_{11} \\ C_{21} & C_{22} & B_{11} & -B_{11} \end{bmatrix},$$

where

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

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

$$\mathbf{T}_l^\star : (Q_1,Q_2) \rightarrow (Q_1A_{22}+Q_2B_{22}, -A_{11}Q_1 - B_{11}Q_2),$$

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

We have abstained from providing perturbation expansions for the representation of $(A+E,B+F)$ with respect to $(\hat{X},\hat{Y})$ as we found these expansions rather difficult to interpret. For a discussion on the appropriate condition number for a cluster of generalized eigenvalues, the reader is referred to [134].

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

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

Then

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

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

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 may severely overestimate 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

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

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.

Example 2.10 ([231, Ex. 4.2.10]). *Let the regular matrix pair* $(A,B) \in \mathbb{R}^{4\times 4} \times \mathbb{R}^{4\times 4}$ *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 $(\mathcal{X},\mathcal{Y}) = (\operatorname{span}([I,0]^T), \operatorname{span}([I,0]^T))$ *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 $c(\mathcal{X},\mathcal{Y})$ *significantly 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 $\|E\|$ is proportional to $\|A\|$ while $\|F\|$ is proportional to $\|B\|$. To balance the effects of perturbations, Sun [231] 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 [134, 135] 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 [73, 131, 136] for more details on the efficient solution of such generalized Sylvester equations.

We are not aware of any estimator 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.

2.3 Global Perturbation Bounds

Similar to Section 2.4 in Chapter 1, 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 [81] showed that this coalescence does not take place if

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

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 ([81, 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*

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

3 The Basic QZ Algorithm

The QZ algorithm is a numerically backward stable method for computing a (real) generalized Schur decomposition of a matrix pair (A, B). It goes back to Moler and Stewart in 1973 [179] and has undergone only a few modifications during the next 25 years, notably through works by Ward [251], Kaufman [140], Dackland and Kågström [74]. Non-orthogonal variants of the QZ algorithm include the LZ algorithm by Kaufman [139] and the AB algorithm for pencils by Kublanovskaya [153].

For the purpose of describing the QZ algorithm, let us temporarily assume that B is invertible. We will later on, in Sections 3.4 and 5.3, discuss the handling of a singular matrix B.

The fundamental ingredient is a fairly straight generalization of the QR iteration, the so called QZ iteration [179]. It generates a sequence of orthogonally equivalent matrix pairs $(A_0, B_0) \leftarrow (A, B)$, (A_1, B_1), (A_2, B_2), $\ldots$, 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:

$$p_i(A_{i-1}B_{i-1}^{-1}) = Q_i R_i, \quad \text{(QR decomposition)} \tag{2.13a}$$

$$\tilde{B}_i \leftarrow Q_i^T B_{i-1}, \tag{2.13b}$$

$$\tilde{B}_i = B_i Z_i, \quad \text{(RQ decomposition)} \tag{2.13c}$$

$$A_i \leftarrow Q_i^T A_{i-1} Z_i. \tag{2.13d}$$

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 $B_{i-1}^{-1}A_{i-1}$

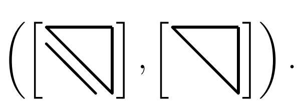
(yielding the orthogonal matrix Z_i). The advantage of the QZ iteration over QR iterations is that the explicit inversion of the possibly ill-conditioned matrix B is avoided in the formation of Q and Z. If the QZ iteration converges, it produces the block Schur form of a slightly perturbed matrix pair $(A+E, B+F)$, where $\|E\|_F$ and $\|F\|_F$ are of order $\mathbf{u} \cdot \|A\|_F$ and $\mathbf{u} \cdot \|B\|_F$, respectively, which implies numerical backward stability [179].

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 $A_{i-1}B_{i-1}^{-1}$. It should be mentioned, however, that a direct proof of convergence of the QZ iteration has its own advantages and leads to a better quantitative description of the convergence behavior, see [262].

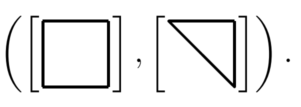
3.1 Hessenberg-Triangular Form

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 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 the computation of orthogonal matrices Q and Z so that

$$Q^T \cdot (A, B) \cdot Z =$$


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

$$(A, B) \leftarrow Q^T \cdot (A, B) =$$


The 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

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

for 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 $G_{ij}(\theta) \equiv G_{ji}(x)$ to identify the Givens rotation which maps the ith component of $G_{ij}(\theta)^T \cdot x$ 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.

We illustrate how these transformations 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. For general n, we obtain the following algorithm.

Algorithm 2.12 (Reduction to Hessenberg-triangular form [179]).

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.*

```
$Q \leftarrow I_n$,   $Z \leftarrow I_n$
FOR $j = 1, \dots, n-2$
  FOR $i = n-1, n-2, \dots, j+1$
    $G \leftarrow G_{i,i+1}(Ae_j)$
    $A \leftarrow GA$,   $B \leftarrow GB$,   $Q \leftarrow QG^T$
    $G \leftarrow G_{i+1,i}(e_{i+1}^T B)$
```

```
        $A \leftarrow AG, \quad B \leftarrow BG, \quad Z \leftarrow ZG^T$
      END FOR
    END FOR
```

This 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 $3n^3 + \mathcal{O}(n^2)$ 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 orthogonal factor.

3.2 Implicit Shifted QZ Iteration

By the implicit Q theorem, see Theorem 1.24, 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 $x = p_i(A_{i-1}B_{i-1}^{-1}) \cdot e_1$ denotes the first column of the selected shift polynomial. 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

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

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:

$$\left(\begin{bmatrix} \hat{a} & \hat{a} & \hat{a} & a & a & a \\ \hat{b}_a & \hat{b}_a & \hat{b}_a & a & a & a \\ \hat{b}_a & \hat{b}_a & \hat{b}_a & a & a & a \\ \hat{b}_a & \hat{b}_a & \hat{b}_a & a & a & a \\ 0 & 0 & 0 & a & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} \hat{b} & \hat{b} & \hat{b} & b & b & b \\ \hat{0} & \hat{b}_b & \hat{b}_b & b & b & b \\ 0 & \hat{0} & \hat{b}_b & b & b & b \\ 0 & 0 & 0 & b & b & b \\ 0 & 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right). \tag{2.14}$$

We have used the symbols b_a and b_b to designate elements of the so called bulge pair, see Section 5.2 below. Finally, the matrix pair (A_{i-1}, B_{i-1}) is returned to Hessenberg-triangular form using Algorithm 2.12. Overall, we obtain the following algorithm for performing a QZ iteration.

Algorithm 2.13 (Implicit shifted QZ iteration).

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

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

   $$\tilde{Q} = G_{12}(\theta_1) \cdots G_{m-1,m}(\theta_{m-1}) \cdot G_{m,m+1}(\theta_m)$$

   so that $\tilde{Q}x$ is a scalar multiple of e_1.

4. Update $(A_{i-1}, B_{i-1}) \leftarrow \tilde{Q} \cdot (A_{i-1}, B_{i-1})$.

5. Construct a sequence of Givens rotations

   $$\tilde{Z} = G_{m+1,m}(\theta_{m+m}) \cdot G_{m,m-1}(\theta_{m+m-1}) \cdots G_{21}(\theta_{m+1})$$

   so that $B_{i-1}\tilde{Z}$ is upper triangular.

6. Update $(A_{i-1}, B_{i-1}) \leftarrow (A_{i-1}, B_{i-1}) \cdot \tilde{Z}$.

7. Apply Algorithm 2.12 to compute orthogonal matrices Q and Z so that (A_{i-1}, B_{i-1}) 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 1.25, apply likewise to Algorithm 2.13. In particular, Step 7 should be based on a special-purpose implementation of Algorithm 2.12, 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 2.13 are negligible if we assume $m \ll n$.

Let us illustrate Algorithm 2.13 for $n = 6$ and $m = 2$. After Step 6 the matrix pair (A_{i-1}, B_{i-1}) takes the form displayed in (2.14). The bulge pair resides in rows $2, \dots, 4$ and columns $1, \dots, 3$. The subsequent reduction 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 [262]. Applying the first outer loop of Algorithm 2.12 to the matrix pair (A_{i-1}, B_{i-1}) amounts to moving the bulge pair one step downwards:

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

Each further execution of an outer loop of Algorithm 2.12 pushes the bulge pair further downwards until it vanishes at the bottom right corner:

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

$$(A_{i-1}, B_{i-1}) \leftarrow \left(\begin{bmatrix} a & a & a & \hat{a} & \hat{a} & \hat{a} \\ a & a & a & \hat{a} & \hat{a} & \hat{a} \\ 0 & a & a & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & 0 & \hat{0} & \hat{b}_a & \hat{b}_a & \hat{b}_a \\ 0 & 0 & \hat{0} & \hat{b}_a & \hat{b}_a & \hat{b}_a \end{bmatrix}, \begin{bmatrix} b & b & b & \hat{b} & \hat{b} & \hat{b} \\ 0 & b & b & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & b & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & 0 & \hat{0} & \hat{b}_b & \hat{b}_b \\ 0 & 0 & 0 & 0 & \hat{0} & \hat{b}_b \end{bmatrix} \right), \quad (2.15)$$

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

3.3 On the Use of Householder Matrices

Early attempts

Algorithm 2.13 differs in one aspect significantly from the originally proposed implicit QZ iteration, described for the case $m = 2$ in [179]. Moler and Stewart suggest to use transformations based on Householder matrices instead of Givens rotations to move a bulge pair a step downwards. Their idea is well explained by a 5×5 example:

$$(A,B) = \left(\begin{bmatrix} a & a & a & a & a \\ b_a & b_a & b_a & a & a \\ b_a & b_a & b_a & a & a \\ b_a & b_a & b_a & a & a \\ 0 & 0 & 0 & a & a \end{bmatrix}, \begin{bmatrix} b & b & b & b & b \\ 0 & b_b & b_b & b & b \\ 0 & 0 & b_b & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & b \end{bmatrix} \right). \tag{2.16}$$

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

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

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:

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

It seems baffling why such a combination of orthogonal transformations has been favored. Using this type of transformation, Algorithm 2.13 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. As a matter of fact, this combination is used in the EISPACK implementation `QZIT` of the QZ algorithm. Later on, Ward [251] recognized this increase of flops and proposed to use a single shift iteration if both shifts are real. The resulting *combination shift QZ algorithm* partly avoids the extra costs mentioned above.

Opposite Householder matrices

A cheaper alternative has been proposed by Watkins and Elsner [262]. 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 2.14. *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 $B \cdot H_1(B^{-1}e_1)e_1 = 1/\gamma \cdot e_1$. $\square$

Normally, a Householder matrix that is used for annihilating several entries in one column is applied from the left, which tempts us to call $H_1(B^{-1}e_1)$ informally 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., [74, 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 2.15. *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 $(I - \hat{\beta}\hat{v}\hat{v}^T)\hat{x} = -\sqrt{1+1/100}\cdot e_1$. The $(2,1)$ entry of $B(I - \hat{\beta}\hat{v}\hat{v}^T)$ is approximately given by 10^{-16}.

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

$$(A+E)\hat{x} = e_1, \quad \|E\|_2 \le c_A\|A\|_2. \tag{2.19}$$

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 $H_1(\hat{x}) \equiv I - \tilde{\beta}\tilde{v}\tilde{v}^T$, where $\tilde{\beta} \in \mathbb{R}, \tilde{v} \in \mathbb{R}^n$, 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 $\tilde{\beta}, \tilde{v}$ defining $H_1(\hat{x})$ is subject to roundoff errors. Using a standard algorithm, 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 [122, p. 365]. It follows that

$$\begin{aligned}\|A\cdot(I-\hat\beta\hat v\hat v^T)e_1 - 1/\hat\gamma\cdot e_1\|_2 &\le \|A\cdot(I-\tilde\beta\tilde v\tilde v^T)e_1 - 1/\hat\gamma\cdot e_1\|_2\\ &\quad + (c_\beta+2c_v)\|A\|_2 + \mathcal{O}(\mathbf{u}^2)\\ &\le (c_A+c_\beta+2c_v)\|A\|_2+\mathcal{O}(\mathbf{u}^2).\end{aligned} \tag{2.20}$$

This shows that if $\hat x$ is computed by a backward stable method, then the last $n-1$ elements in the first column $A(I-\hat\beta\hat v\hat v^T)$ 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 (2.16). 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 (2.17). The submatrix $B_{22} = B(2:4,2:4)$, since B itself is assumed to be invertible. Next, the opposite Householder matrix $H_1(B_{22}^{-1}e_1)$ is applied to columns $2,\dots,4$ of A and B, which yields

$$(A,B)\leftarrow\left(\begin{bmatrix} a & \hat a & \hat a & \hat a & a\\ a & \hat a & \hat a & \hat a & a\\ 0 & \hat b_a & \hat b_a & \hat b_a & a\\ 0 & \hat b_a & \hat b_a & \hat b_a & a\\ 0 & \hat b_a & \hat b_a & \hat b_a & a\end{bmatrix},\begin{bmatrix} b & \hat b & \hat b & \hat b & b\\ 0 & \hat b & \hat b & \hat b & b\\ 0 & \hat 0 & \hat b_b & \hat b_b & b\\ 0 & \hat 0 & \hat b_b & \hat b_b & b\\ 0 & 0 & 0 & 0 & b\end{bmatrix}\right). \tag{2.21}$$

Note that – in contrast to (2.18) – 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 reads as follows.

Algorithm 2.16 (Algorithm 2.13 based on Householder matrices).
Input and Output: *See Algorithm 2.13.*

```
Apply Steps 1 and 2 of Algorithm 2.13.
$(A_{i-1},B_{i-1}) \leftarrow H_1(x)\cdot(A_{i-1},B_{i-1})$
$Q\leftarrow H_1(x)$,    $Z\leftarrow H_1(B_{i-1}^{-1}e_1)$
$(A_{i-1},B_{i-1}) \leftarrow (A_{i-1},B_{i-1})\cdot Z$
FOR $j\leftarrow 1,\dots,n-1$
   $\tilde Q \leftarrow H_{j+1}(A_{i-1}e_j)$
   $(A_{i-1},B_{i-1}) \leftarrow \tilde Q\cdot(A_{i-1},B_{i-1})$
   $Q\leftarrow Q\tilde Q$
   $\tilde Z \leftarrow H_{j+1}(B_{i-1}^{-1}e_{j+1})$
   $(A_{i-1},B_{i-1}) \leftarrow (A_{i-1},B_{i-1})\cdot\tilde Z$
   $Z\leftarrow Z\tilde Z$
END FOR
```

If $m\ll n$, a proper implementation of this algorithm 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 approximately 1/11-th more flops than one double shift QZ 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 is Gaussian elimination with partial pivoting. Note, however, that the constant c_A, which bounds the backward error in (2.19), can be proportional to 2^n if this method is used [122]. The famous Wilkinson matrix [264, p. 212] is such an "admittedly highly artificial" example. Examples of practical relevance have been discovered by Wright [267] and Foster [99].

Example 2.17 ([267]). *The following $2k \times 2k$ matrix arises after multiple shooting has been applied to a certain two-point boundary value problem:*

$$B = \begin{bmatrix} I_2 & & & & I_2 \\ -e^{Mh} & I_2 & & & \\ & -e^{Mh} & I_2 & & \\ & & \ddots & \ddots & \\ & & & -e^{Mh} & I_2 \end{bmatrix} \tag{2.22}$$

with $h = 0.3$ and

$$M = \begin{bmatrix} -1/6 & 1 \\ 1 & -1/6 \end{bmatrix}.$$

We solved the linear system $Bx = e_1$ for several values of k using the Matlab *operator \, which employs Gaussian elimination with partial pivoting. The following table displays the measured relative residual of the computed solution and the norm of the vector $f \in \mathbb{R}^{2k-1}$ containing the subdiagonal elements in the first column of $BH_1(\hat{x})e_1$.*

k	25	50	75	100
$\Vert B\hat{x} - e_1\Vert_2/\Vert B\Vert_2$	9.9×10^{-15}	4.7×10^{-12}	3.3×10^{-09}	7.9×10^{-07}
$\Vert f\Vert_2/\Vert B\Vert_2$	6.6×10^{-15}	3.1×10^{-12}	2.2×10^{-09}	5.3×10^{-07}

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 $H(Q^T e_1)$ is an opposite Householder matrix. Even if B is singular, it can be shown that the first column of $B \cdot H(Q^T e_1)$ is mapped to a multiple of e_1:

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

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

The RQ decomposition enjoys a favorable backward error analysis, the constant c_A in (2.19) can be bounded by roughly $n^2\mathbf{u}$, see, e.g., [122, 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.

Algorithm 2.18 (Update of 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

   $$Q_1 = G_{21}(\theta_1) \cdot G_{32}(\theta_2) \cdots G_{n,n-1}(\theta_{n-1})$$

   so that $Q_1 x = \gamma e_1$ for some $\gamma \in \mathbb{R}$.

2. Update $R \leftarrow RQ_1$ and $Q \leftarrow Q_1 Q$. *% 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

   $$Q_2 = G_{n-1,n}(\theta_{n-1}) \cdots G_{23}(\theta_2) \cdot G_{12}(\theta_1)$$

   so that RQ_2 is upper triangular.

5. Update $R \leftarrow RQ_2$ and $Q \leftarrow Q_2 Q$.

This algorithm requires $\mathcal{O}(n^2)$ flops instead of $\mathcal{O}(n^3)$ flops that are needed for computing an RQ decomposition from scratch. This decrease of flops may not be relevant in the context of Algorithm 2.16. The number of shifts per implicit QZ iteration must be kept tiny in order to avoid shift blurring phenomena, see Section 5.3 in Chapter 1. This implies that the cost for determining the opposite Householder matrices are expected to be negligible anyway. Nevertheless, Algorithm 2.18 has useful applications in other parts of the QZ algorithm, see Sections 5.1 and 6.

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

$$|a_{k+1,k}| \le \mathbf{u} \cdot \|A\|_F. \qquad (2.24)$$

All known public implementations of the QZ algorithm employ this deflation criterion.[6] A variation of Example 1.26 can be used to show that the neighbor-wise criterion

$$|a_{k+1,k}| \leq \mathbf{u} \cdot (|a_{k,k}| + |a_{k+1,k+1}|) \tag{2.25}$$

may produce more accurately computed generalized eigenvalues in the presence of graded matrix pairs, see also [147]. We therefore propose 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, e.g., in [112, Sec. 7.7.5]. 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:

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

[6] A possible exception is the single-shift complex QZ algorithm implemented in MATLAB before MATLAB's `eig` function was based on LAPACK, see [178].

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

Finally, a Givens rotation acting on rows 1 and 2 can be used to deflate the infinite eigenvalue at top:

$$(A,B) \leftarrow \left(\left[\begin{array}{c|cccc} \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ \hline \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & a & a & a & a \\ 0 & 0 & a & a & a \\ 0 & 0 & 0 & a & a \end{array} \right], \left[\begin{array}{c|cccc} 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hline 0 & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & 0 & b & b & b \\ 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & b \end{array} \right] \right).$$

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 [112, Sec. 7.7.5].

Later on, in Section 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 tiny value for b_{kk} can have a negative effect on the convergence of the QZ iteration [140, 257]. 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

$$|b_{kk}| \leq \mathbf{u} \cdot \|B\|_F,$$

as implemented in the LAPACK routine DHGEQZ, or the possibly more reliable neighbor-wise criterion

$$|b_{kk}| \leq \mathbf{u} \cdot (|b_{k-1,k}| + |b_{k,k+1}|).$$

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 [105], is larger than one [179]. Much more reliable decisions on the nature of infinite eigenvalues can be met using algorithms that reveal Kronecker structures, such as GUPTRI [82, 83]. In some cases, infinite eigenvalues can be cheaply and reliably deflated by exploiting the structure of A and B [10]. Stykel [229] describes such a case in which (A,B) arise from a semi-discretized Stokes equation.

3.5 The Overall Algorithm

Glueing implicit QZ iterations and deflation together yields the QZ algorithm for Hessenberg-triangular matrix pairs.

Algorithm 2.19 (Basic QZ algorithm).

Input: *A matrix pair $(A,B) \in \mathbb{R}^{n \times n} \times \mathbb{R}^{n \times n}$ in Hessenberg-triangular form, an integer $m \in [2, n]$.*

Output: *Orthogonal matrices $Q, Z \in \mathbb{R}^{n \times n}$ so that the matrix $(S,T) = Q^T \cdot (A,B) \cdot Z$ 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 shifts are used.*

```
$Q \leftarrow I_n, \quad Z \leftarrow I_n$
$i \leftarrow 1, \quad l \leftarrow n$
FOR $it \leftarrow 0, \dots, 30 \cdot n$
  % The active matrix pair is $(A(i:l, i:l), B(i:l, i:l))$.
  % Search for deflations in $B$.
  $k \leftarrow i$
  $\mathrm{tol} \leftarrow \mathbf{u} \cdot |b_{k,k+1}|$
  WHILE $k \leq l$ AND $|b_{kk}| > \mathrm{tol}$
    IF $k < l$ THEN
      $\mathrm{tol} \leftarrow \mathbf{u} \cdot (|b_{k,k+1}| + |b_{k-1,k}|)$
    ELSE
      $\mathrm{tol} \leftarrow \mathbf{u} \cdot |b_{k-1,k}|$
    END IF
    $k \leftarrow k + 1$
  END WHILE
  IF $k < l$ THEN
    % Deflation at position $(k, k)$ found.
    $b_{kk} \leftarrow 0$
    IF $k < (l + i)/2$ THEN
      Deflate infinite eigenvalue at top using procedure described in
      Section 3.4.
    ELSE
      Deflate infinite eigenvalue at bottom using procedure described in
      Section 3.4.
    END IF
  END IF
  % Search for deflations in $A$.
  $k \leftarrow i$
  WHILE $k < l$ AND $|a_{k+1,k}| > \mathbf{u} \cdot (|a_{k,k}| + |a_{k+1,k+1}|)$
    $k \leftarrow k + 1$
  END WHILE
  IF $k < l$ THEN
    % Deflation at position $(k + 1, k)$ found.
    $a_{k+1,k} \leftarrow 0$
    $i \leftarrow k + 1$
    IF $i + 1 \geq l$ THEN
      % Blocks of size at most two have converged.
      $l \leftarrow i - 1; \quad i \leftarrow 1$
      IF $i + 1 \geq l$ THEN
        % QZ algorithm has converged.
        Exit.
      END IF
    END IF
  ELSE
    Apply implicit shifted QZ iteration, Algorithm 2.16, 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.
    Update $A(i:l, l+1:n) \leftarrow \tilde{Q}^T A(i:l, l+1:n)$.
    Update $A(1:i-1, i:l) \leftarrow A(1:i-1, i:l)\tilde{Z}$.
    Update $B(i:l, l+1:n)) \leftarrow \tilde{Q}^T B(i:l, l+1:n)$.
```

```
        Update $B(1:i-1,i:l) \leftarrow B(1:i-1,i:l)\tilde{Z}$.
        Update $Q(1:n,i:l) \leftarrow Q(1:n,i:l)\tilde{Q}$,   $Z(1:n,i:l) \leftarrow Z(1:n,i:l)\tilde{Z}$.
    END IF
END FOR
% The QZ algorithm did not converge within $30 \cdot n$ iterations.
Exit and error return.
```

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 2.19 needs about twice the number of flops required by the QR algorithm applied to an $n \times n$ matrix, see Page 29.

Remark 2.20. *Similar to the QR algorithm, see Remark 1.29, post-processing must be applied in order to guarantee that Algorithm 2.19 returns a real generalized Schur form. Based on work by Van Loan [240, 241, 179], the LAPACK routine* `DLAGV2` *transforms the 2×2 diagonal blocks of the matrix pair (S, T) returned by Algorithm 1.27 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

$$\left(\begin{bmatrix} s_{ii} & s_{i,i+1} \\ 0 & s_{i+1,i+1} \end{bmatrix}, \begin{bmatrix} t_{ii} & 0 \\ 0 & t_{i+1,i+1} \end{bmatrix} \right), \quad t_{ii} \geq t_{i+1,i+1} \geq 0,,$$

in the case of complex eigenvalues.

Algorithm 2.19 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 2.19 inherits the (rare) possibility of global convergence failures from the QR algorithm. Exceptional shift strategies similar to those described in Section 3.6, Chapter 1, can be developed for the QZ algorithm and are partly incorporated in `DHGEQZ`.

4 Balancing

Balancing a matrix pair is a preprocessing step which can have positive effects on the performance and accuracy of subsequent methods for computing generalized eigenvalues and deflating subspaces [163, 252]. As for general matrices, see Section 4 in Chapter 1, 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 similarity transformation (scaling), which aims at reducing the sensitivity of the generalized eigenvalues.

4.1 Isolating Eigenvalues

In the first stage of the balancing algorithm by Ward [252], 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 $(A_{11}, B_{11}) \in \mathbb{R}^{(i_l-1)\times(i_l-1)} \times \mathbb{R}^{(i_l-1)\times(i_l-1)}$ and $(A_{33}, B_{33}) \in \mathbb{R}^{(n-i_h)\times(n-i_h)} \times \mathbb{R}^{(n-i_h)\times(n-i_h)}$ are upper triangular matrix pairs. The *isolated generalized eigenvalues* contained in these 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 1.31.

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 [252] 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 [252] for more details. This balancing strategy together with the permutation algorithm outlined above is implemented in the LAPACK routine `DGGBAL`.

Ward's algorithm must be applied with some care. In fact, it is simple to construct examples where a balancing strategy based on (2.27) severely deteriorates the eigenvalue sensitivities.

Example 2.21. *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 [163] recently proposed a balancing strategy for matrix pairs which is closer in spirit to the Parlett-Reinsch algorithm for balancing matrices, see Algorithm 1.32. It produces diagonal matrices D_R and D_C so that every row of $D_R[A, B]$ and every column of $\left[\begin{smallmatrix} A \\ B \end{smallmatrix}\right] D_C$ 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 $D_R(A + \imath B)D_C$ 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 [181, 196] and the references therein.

The algorithm proposed in [163] 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 $\left[\begin{smallmatrix} A \\ B \end{smallmatrix}\right] D_C$ 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 [163], 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.

Example 2.22. *Consider the matrices*

$$A = 10^{-6} \times \begin{bmatrix} 1 & 2 & 0 \\ 2 & 2 & 2 \\ 0 & 1 & 1 \end{bmatrix}, \quad D = \operatorname{diag}(2^{-17}, 1, 1).$$

The eigenvalues of A are $\{-10^{-6}, 10^{-6}, 4 \times 10^{-6}\}$. If the algorithm by Lemonnier and Van Dooren [163] 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.

5 Block Algorithms

5.1 Reduction to Hessenberg-Triangular Form

The reduction to Hessenberg-triangular form as implemented in Algorithm 2.12 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 2.12 performs very poorly for larger matrix pencils. Dackland and Kågström [74] 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 2.12. 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, see [43, 42, 155] and the references therein.

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} with block size nb. The implicit assumption that n is an integer multiple nb is for notational convenience only. Let us illustrate the block partitioning for $n = 6 \cdot nb$:

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} & A_{26} \\ A_{31} & A_{32} & A_{33} & A_{34} & A_{35} & A_{36} \\ A_{41} & A_{42} & A_{43} & A_{44} & A_{45} & A_{46} \\ A_{51} & A_{52} & A_{53} & A_{54} & A_{55} & A_{56} \\ A_{61} & A_{62} & A_{63} & A_{64} & A_{65} & A_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} & B_{13} & B_{14} & B_{15} & B_{16} \\ 0 & B_{22} & B_{23} & B_{24} & B_{25} & B_{26} \\ 0 & 0 & B_{33} & B_{34} & B_{35} & B_{36} \\ 0 & 0 & 0 & B_{44} & B_{45} & B_{46} \\ 0 & 0 & 0 & 0 & B_{55} & B_{56} \\ 0 & 0 & 0 & 0 & 0 & B_{66} \end{bmatrix} \right).$$

As B is upper triangular each of its diagonal blocks B_{ii} is of course also upper triangular. Our goal is to reduce the pencil (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 each subdiagonal block $A_{i+1,i}$ and each diagonal block B_{ii} has upper triangular form.

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 its last $p-1$ blocks. For example if $p = 3$, this amounts to computing a QR factorization

$$\begin{bmatrix} A_{41} \\ A_{51} \\ A_{61} \end{bmatrix} = GR = (I + VTV^T)\hat{A}_{41},$$

where $I + VTV^T$ with $T \in \mathbb{R}^{nb \times nb}, V \in \mathbb{R}^{p \cdot nb \times nb}$ is the compact WY representation of the orthogonal factor G. Applying G^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 factorization

$$\begin{bmatrix} B_{44} & B_{45} & B_{46} \\ B_{54} & B_{55} & B_{56} \\ B_{64} & B_{65} & B_{66} \end{bmatrix} = R\tilde{G}, \tag{2.28}$$

where R is upper triangular and $\tilde{G}$ is orthogonal. The obvious disadvantage is that there is no compact WY representation with a thin V factor for $\tilde{G}$. Consequently, the application of $\tilde{G}$ to the last three columns of A and B becomes rather expensive, particularly for larger p. One can avoid this extra expense using a technique similar to opposite Householder matrices. Let

$$\tilde{G}^T \begin{bmatrix} I_{nb} \\ 0 \\ 0 \end{bmatrix} = \check{G} \begin{bmatrix} \tilde{G}_1 \\ 0 \\ 0 \end{bmatrix}$$

be a QR factorization 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 $\check{T} \in \mathbb{R}^{nb \times nb}, \check{V} \in \mathbb{R}^{p \cdot nb \times nb}$. Applying $\check{G}$ 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 newly created zero entries in B are of order $\mathbf{u} \cdot \|R\|_F$. These facts can be proven along the line of argument used in Section 3.2. The blocks A_{31} and A_{41} are similarly annihilated:

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ \hat{A}_{21} & \hat{A}_{22} & \hat{A}_{23} & \hat{A}_{24} & \hat{A}_{25} & \hat{A}_{26} \\ \hat{0} & \hat{A}_{32} & \hat{A}_{33} & \hat{A}_{34} & \hat{A}_{35} & \hat{A}_{36} \\ \hat{0} & \hat{A}_{42} & \hat{A}_{43} & \hat{A}_{44} & \hat{A}_{45} & \hat{A}_{46} \\ 0 & A_{52} & A_{53} & A_{54} & A_{55} & A_{56} \\ 0 & A_{62} & A_{63} & A_{64} & A_{65} & A_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & B_{12} & B_{13} & B_{14} & B_{15} & B_{16} \\ 0 & \hat{B}_{22} & \hat{B}_{23} & \hat{B}_{24} & \hat{B}_{25} & \hat{B}_{26} \\ 0 & \hat{B}_{32} & \hat{B}_{33} & \hat{B}_{34} & \hat{B}_{35} & \hat{B}_{36} \\ 0 & \hat{B}_{42} & \hat{B}_{43} & \hat{B}_{44} & \hat{B}_{45} & \hat{B}_{46} \\ 0 & 0 & 0 & 0 & B_{55} & B_{56} \\ 0 & 0 & 0 & 0 & B_{65} & B_{66} \end{bmatrix} \right),$$

$$\left(\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} & A_{16} \\ A_{21} & \hat{A}_{22} & \hat{A}_{23} & \hat{A}_{24} & A_{25} & A_{26} \\ 0 & \hat{A}_{32} & \hat{A}_{33} & \hat{A}_{34} & A_{35} & A_{36} \\ 0 & \hat{A}_{42} & \hat{A}_{43} & \hat{A}_{44} & A_{45} & A_{46} \\ 0 & \hat{A}_{52} & \hat{A}_{53} & \hat{A}_{54} & A_{55} & A_{56} \\ 0 & \hat{A}_{62} & \hat{A}_{63} & \hat{A}_{64} & A_{65} & A_{66} \end{bmatrix}, \begin{bmatrix} B_{11} & \hat{B}_{12} & \hat{B}_{13} & \hat{B}_{14} & B_{15} & B_{16} \\ 0 & \hat{B}_{22} & \hat{B}_{23} & \hat{B}_{24} & B_{25} & B_{26} \\ 0 & \hat{0} & \hat{B}_{33} & \hat{B}_{34} & B_{35} & B_{36} \\ 0 & \hat{0} & \hat{B}_{43} & \hat{B}_{44} & B_{45} & B_{46} \\ 0 & 0 & 0 & 0 & B_{55} & B_{56} \\ 0 & 0 & 0 & 0 & B_{65} & B_{66} \end{bmatrix} \right).$$

The reduction of the second block column proceeds in the same fashion. First, only the blocks A_{62} and B_{65} are annihilated and second, the blocks A_{42}, A_{52} and B_{43}, B_{53}. The algorithm for general n and p is given below.

Algorithm 2.23 (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 nb 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.*

```
$Q \leftarrow I_n, \quad Z \leftarrow I_n$
$l \leftarrow n - nb - [(n - 2\cdot nb - 1) \bmod ((p-1)\cdot nb)]$
FOR $j = 1, nb+1, 2\cdot nb + 1 \ldots, n - 3\cdot nb + 1$
  FOR $i = l, l - (p-1)\cdot nb, l - 2(p-1)\cdot nb, \ldots, j + nb$
    $m \leftarrow \min\{n - i + 1, p\cdot nb\}$
    Compute a QR factorization $A_{i:i+m-1,j:j+k-1} = GR$.
    $A_{i:i+m-1,:} \leftarrow G^T A_{i:i+m-1,:}$
    $B_{i:i+m-1,:} \leftarrow G^T B_{i:i+m-1,:}$
    $Q_{:,i:i+m-1} \leftarrow Q_{:,i:i+m-1} G$
    Compute an RQ factorization $B_{i:i+m-1,i:i+m-1} = R\tilde{G}$.
    Compute a QR factorization $\tilde{G}^T_{1:nb,:} = \check{G}R$.
    $A_{:,i:i+m-1} \leftarrow A_{:,i:i+m-1}\check{G}$
    $B_{:,i:i+m-1} \leftarrow B_{:,i:i+m-1}\check{G}$
    $Z_{:,i:i+m-1} \leftarrow Z_{:,i:i+m-1}\check{G}$
  END FOR
  $l \leftarrow l + k$
  IF $nb + l > n$ THEN $l \leftarrow l - (p-1)\cdot nb$ END IF
END FOR
```

Table 2.1 contains timings that have been obtained from a Fortran implementation of Algorithm 2.23 applied to random matrix pencils. 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 2.23, with greater benefit for moderate block sizes nb. This is particularly useful considering the fact that the computational expenses for the second stage may grow with nb.

Algorithm 2.23 is almost identical with an algorithm proposed by Dackland and Kågström [74, Sec. 2.3]. The only difference is that the latter algorithm uses an RQ factorization of the bottom $(p-1)\cdot nb \times p\cdot nb$ matrix to annihilate the nonzero block entries that have been introduced by a transformation from the left

Alg. 2.23		w/o compt. of Q and Z				with compt. of Q and Z			
n	nb	$p=2$	$p=4$	$p=6$	$p=8$	$p=2$	$p=4$	$p=6$	$p=8$
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 2.23.*

in a $p \cdot nb \times p \cdot nb$ diagonal block of B:

$$\left[\begin{array}{c|ccc} B_{ii} & B_{i,i+1} & \dots & B_{i,i+p-1} \\ \hline B_{i+1,i} & B_{i+1,i+1} & \dots & B_{i+1,i+p-1} \\ \vdots & \vdots & & \vdots \\ B_{i+p-1,i} & B_{i+p-1,i+1} & \dots & B_{i+p-1,i+p-1} \end{array}\right] = \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & R_{22} \end{array}\right] Q,$$

where $R_{22} \in \mathbb{R}^{(p-1)\cdot nb \times (p-1)\cdot nb}$ is upper triangular matrix.

A comparison between both algorithms with optimal choices for $p \in [2, 10]$ (p_{DK} refers to the Dackland/Kågström algorithm and $p_{\mathrm{Alg.\ 2.23}}$ to Algorithm 2.23) is presented in Table 2.2. Columns 4 and 7 contain the ratios between the runtimes of both algorithms without and with accumulation of orthogonal transformations, respectively. The optimal choice for the Dackland/Kågström algorithm is always $p_{\mathrm{DK}} \leq 3$. We observe that the performance of this algorithm is almost always worse than the performance of Algorithm 2.23 for the chosen matrix and block sizes. It requires up to 1.88 times more runtime.

In some cases, the structure of a matrix pair allows a cheaper reduction to block Hessenberg-triangular form.

Example 2.24. *Consider the following matrix pair of block Hankel matrices*

$$(A, B) = \left(\left[\begin{array}{cccc} H_1 & H_2 & \cdots & H_N \\ H_2 & H_3 & \cdots & H_{N+1} \\ \vdots & \vdots & \ddots & \vdots \\ H_N & H_{N+1} & \cdots & H_{2N-1} \end{array}\right], \left[\begin{array}{cccc} H_0 & H_1 & \cdots & H_{N-1} \\ H_1 & H_2 & \cdots & H_N \\ \vdots & \vdots & \ddots & \vdots \\ H_{N-1} & H_N & \cdots & H_{2N-2} \end{array}\right] \right),$$

		w/o compt. of Q and Z			with compt. of Q and Z		
n	nb	p_{DK}	$p_{\text{Alg. 2.23}}$	$\frac{t_{\text{DK}}}{t_{\text{Alg. 2.23}}}$	p_{DK}	$p_{\text{Alg. 2.23}}$	$\frac{t_{\text{DK}}}{t_{\text{Alg. 2.23}}}$
500	8	3	7	1.51	3	9	1.60
500	16	2	5	1.31	3	7	1.41
500	32	2	5	1.13	2	5	1.29
500	64	2	3	0.99	2	3	1.11
1000	8	3	9	1.74	3	9	1.81
1000	16	3	5	1.52	3	6	1.62
1000	32	2	5	1.30	3	7	1.40
1000	64	2	5	1.15	2	5	1.30
1000	128	2	3	1.05	2	5	1.17
2000	8	3	9	1.82	3	9	1.88
2000	16	3	9	1.62	3	10	1.72
2000	32	2	7	1.59	3	7	1.71
2000	64	2	5	1.41	3	7	1.64
2000	128	2	5	1.30	2	5	1.42

Table 2.2. *Performance comparison between an algorithm by Dackland and Kågström and Algorithm 2.23 for the reduction of a matrix pair to block Hessenberg form.*

where $H_0, H_1, \dots, H_{2N-1} \in \mathbb{R}^{n_b \times n_b}$. Such matrix pairs play a role in algorithms for reconstructing polygonal shapes from moments, see [109]. Also, $B^{-1}A$ is the companion matrix of a matrix polynomial, revealing a close relation to this area [108].

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.

Stage 2

In Stage 2, the unwanted $nb-1$ subdiagonals of A are annihilated while the triangular structure of B is preserved. The basic algorithm that applies here is a variant of the QZ iteration, Algorithm 2.19, with the major difference that bulges are chased along the nbth instead of the first subdiagonal. Let us illustrate the first few steps for $nb = 3$ 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 $2, \dots, 4$ 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 $2, \dots, 4$:

$$\left(\left[\begin{array}{c|ccc|cccc} a & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ a & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ \hline 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ 0 & \hat{a} & \hat{a} & \hat{a} & a & a & a & a \\ \hline 0 & 0 & 0 & 0 & a & a & a & a \end{array} \right], \begin{bmatrix} b & \hat{b} & \hat{b} & \hat{b} & b & b & b & b \\ 0 & \hat{b} & \hat{b} & \hat{b} & b & b & b & b \\ 0 & \hat{0} & \hat{b} & \hat{b} & b & b & b & b \\ 0 & \hat{0} & \hat{b} & \hat{b} & b & b & b & b \\ 0 & 0 & 0 & 0 & b & b & b & b \\ 0 & 0 & 0 & 0 & 0 & b & b & b \\ 0 & 0 & 0 & 0 & 0 & 0 & b & b \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \right).$$

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 a Householder matrix applied to columns $2, \dots, 4$ in order to annihilate some of the created 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 can be repeated to annihilate the unwanted subdiagonal entries in columns $2, \dots, 7$ of A.

Dackland and Kågström [74] proposed a similar algorithm based on Givens rotations instead of Householder matrices. High efficiency is attained by delaying the update of parts which are far off from the diagonal. Extensive numerical experiments in [74] show that this algorithm combined with Stage 1 outperforms the LAPACK routine DGGHRD by a factor 2–3. It can be expected that the use of Householder matrices instead of Givens rotations may lead to some further speedup. Surprisingly, our preliminary numerical experiments revealed only minor improvements, if any. Some further investigation of this phenomenon is necessary.

5.2 Multishifts and Bulge Pairs

A 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 [262]. Analogous to the QR iteration, see Section 5.3 in Chapter 1, Watkins [257] has shown that the indented shifts are the finite eigenvalues of these bulge pairs.

Suppose that the implicit shifted QZ iteration with m shifts, Algorithm 2.16, 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 B is an upper triangular matrix. We do not assume that B is nonsingular, only parts used for the shift computation ($T(n-m+1:n, n-m+1:n)$) and parts involved in the introduction of the bulge pair ($T(1:m, 1:m)$) are required to be nonsingular. Let x be a multiple of the first column of the shift polynomial

$$p(HT^{-1}) = (HT^{-1} - \sigma_1 I)\cdots(HT^{-1} - \sigma_m I).$$

For the proper definition of x, it is only necessary that the mth leading principal submatrix of T is nonsingular [262].

The *initial bulge pair* is the matrix pair $\big(B_0^{(H)}, B_0^{(T)}\big)$, 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 2.25 ([257]). *If the mth leading principal submatrix of T is nonsingular, then the shifts $\sigma_1,\ldots,\sigma_m$ are the finite eigenvalues of the initial bulge pair $\big(B_0^{(H)}, B_0^{(T)}\big)$.*

Along the lines of Section 5.4 in Chapter 1, the computation of x can be related to the pole assignment problem for linear descriptor systems [75].

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 $\big(H^{(j)}, T^{(j)}\big)$ denote the updated matrix pair (H,T) obtained after the bulge pair has been chased $(j-1)$ steps. Then, the *jth bulge pair* $\big(B_j^{(H)}, B_j^{(T)}\big)$

is given by

$$B_j^{(H)} = H^{(j)}(j+1:j+m+1, j:j+m+1), \tag{2.29a}$$
$$B_j^{(T)} = T^{(j)}(j+1:j+m+1, j:j+m+1), \tag{2.29b}$$

which corresponds to the submatrices designated by the entries $\hat{b}_a$ and $\hat{b}_b$ in (2.21).

Theorem 2.26 ([257]). *If the mth leading principal submatrix of T is nonsingular, then the shifts $\sigma_1, \dots, \sigma_m$ are the finite eigenvalues of the jth bulge pair $(B_j^{(H)}, B_j^{(T)})$.*

5.3 Deflation of Infinite Eigenvalues Revisited

It should be emphasized that Theorem 2.26 only requires the assumptions that the intended shifts are finite and that the mth leading principal submatrix of T is nonsingular. Zero diagonal entries below this submatrix 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 [250, 251] for $m \le 2$, and by Watkins for general m [257]. The answer is that the zero diagonal entry moves m positions upwards along the diagonal. However, it is assumed in [257] that a QZ iteration based on Givens rotations, i.e., Algorithm 2.13, is used. In the following, we show that the same answer holds for a QZ iteration based on (opposite) Householder matrices, i.e., Algorithm 2.16.

For this purpose, 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:

$$H^{(j)}(j+1:j+m+1, j:j+m+1) = \begin{array}{c c} & \begin{array}{ccc} 1 & m & 1 \end{array} \\ \begin{array}{c} 1 \\ m-1 \\ 1 \end{array} & \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \end{array},$$

$$T^{(j)}(j+1:j+m+1, j:j+m+1) = \begin{array}{c c} & \begin{array}{ccc} 1 & m & 1 \end{array} \\ \begin{array}{c} 1 \\ m-1 \\ 1 \end{array} & \begin{bmatrix} 0 & B_{12} & B_{13} \\ 0 & B_{22} & B_{23} \\ 0 & 0 & 0 \end{bmatrix} \end{array}.$$

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 [257]. The matrix $\left[\begin{smallmatrix} B_{12} \\ B_{22} \end{smallmatrix}\right]$ must be nonsingular, otherwise the jth bulge pair contains less than m finite eigenvalues, which contradicts Theorem 2.26.

If the bulge pair is chased downwards, then first a Householder matrix $I - \beta v v^T$ is constructed so 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 $\left[\begin{smallmatrix} \tilde{B}_{22} \\ \tilde{B}_{32} \end{smallmatrix}\right]$ must be invertible. Otherwise, there exists a vector $\left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right] \neq 0$ 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 so 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 [44]. 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.

5.4 Tightly Coupled Tiny Bulge Pairs

Analogous to the tiny-bulge multishift QR algorithm, see Section 5.5 in Chapter 1, a tiny-bulge multishift QZ algorithm can be developed. The idea of such an algorithm was already mentioned in [3] but not explained in detail. For the purpose of describing some of the details, 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. As for the QR algorithm, tiny values for n_s, say $n_s \in [2, 6]$, must be used in order to avoid shift blurring phenomena.

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). Every bulge pair contains a set of n_s shifts. It is assumed 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

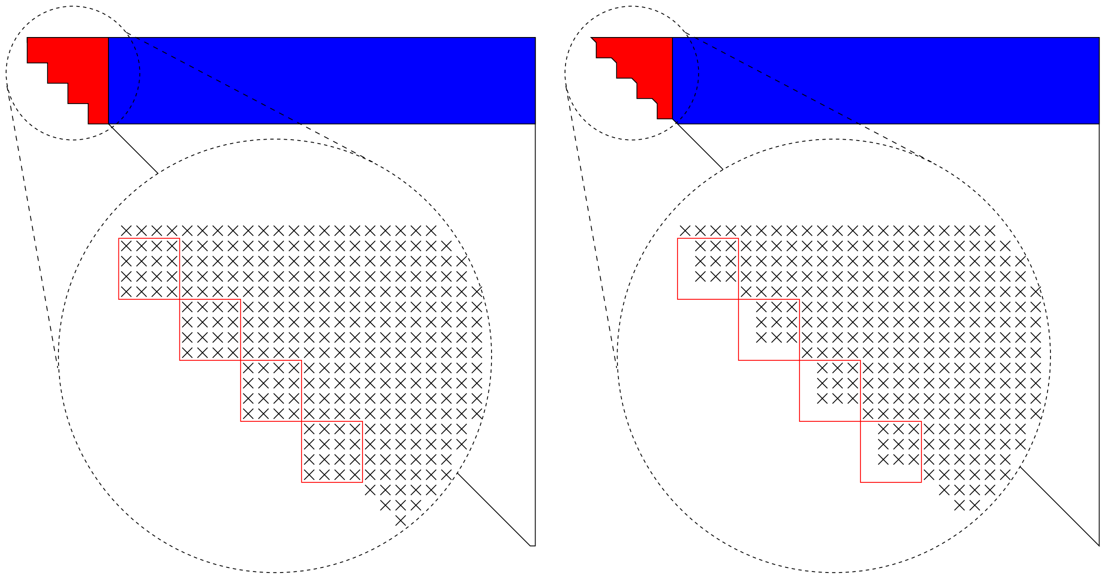


Figure 2.1. *Introducing a chain of $m/n_s = 4$ tightly coupled bulge pairs, each of which contains $n_s = 3$ shifts.*

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 the submatrices painted red in Figure 2.1 must be updated during the bulge chasing process. To update the remaining parts (painted blue), 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:

$$\begin{aligned} H(1:p_h,(p_h+1):n) &\leftarrow U^T \cdot H(1:p_h,(p_h+1):n), \\ T(1:p_h,(p_h+1):n) &\leftarrow U^T \cdot T(1:p_h,(p_h+1):n). \end{aligned}$$

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 5.5. One such step is illustrated in Figure 2.2. Again, only the principal submatrices painted red 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, matrix-matrix multiplications can be used to update the rest of the matrix pair (painted blue in Figure 2.2):

$$\begin{aligned} H(p_l:p_h+k,(p_h+1):n) &\leftarrow U^T \cdot H(p_l:p_h+k,(p_h+1):n), \\ T(p_l:p_h+k,(p_h+1):n) &\leftarrow U^T \cdot T(p_l:p_h+k,(p_h+1):n), \\ H(1:p_l-1,p_l:p_h+k) &\leftarrow H(1:p_l-1,p_l:p_h+k) \cdot V. \\ T(1:p_l-1,p_l:p_h+k) &\leftarrow T(1:p_l-1,p_l:p_h+k) \cdot V. \end{aligned}$$

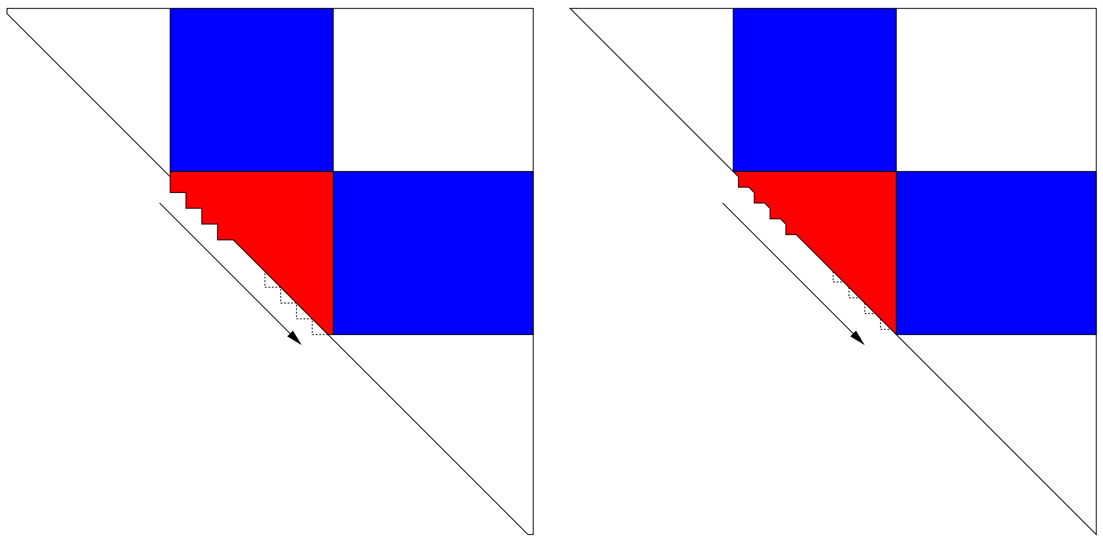


Figure 2.2. *Chasing a chain of $m/n_s = 4$ tightly coupled bulge pairs.*

Note that both matrices, U and V, have the following block structure:

$$\begin{array}{cccc} & 1 & l_2 & l_1 \\ 1 & 1 & 0 & 0 \\ l_1 & 0 & \square & \text{◺} \\ l_2 & 0 & \text{◹} & \square \end{array},$$

where $l_1 = (m/n_s)(n_s + 1) - n_s$ and $l_2 = k + n_s$. Exploiting this structure may 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 5.3.

Getting rid off a chain of bulge pairs

Once the bottom bulge pair of the chain has reached the bottom right corner of the matrix pair, the whole chain is bulge-by-bulge chased off this corner, similarly to the introduction of bulge pairs.

Numerical results

The described tiny-bulge multishift QZ algorithm has been implemented in a Fortran 77 routine called `MTTQZ` and some preliminary numerical experiments have been performed. 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

$$m = \begin{cases} 48, & \text{if } n < 1000, \\ 60, & \text{if } 1000 \leq n < 2000, \end{cases}$$

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 [74]. 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

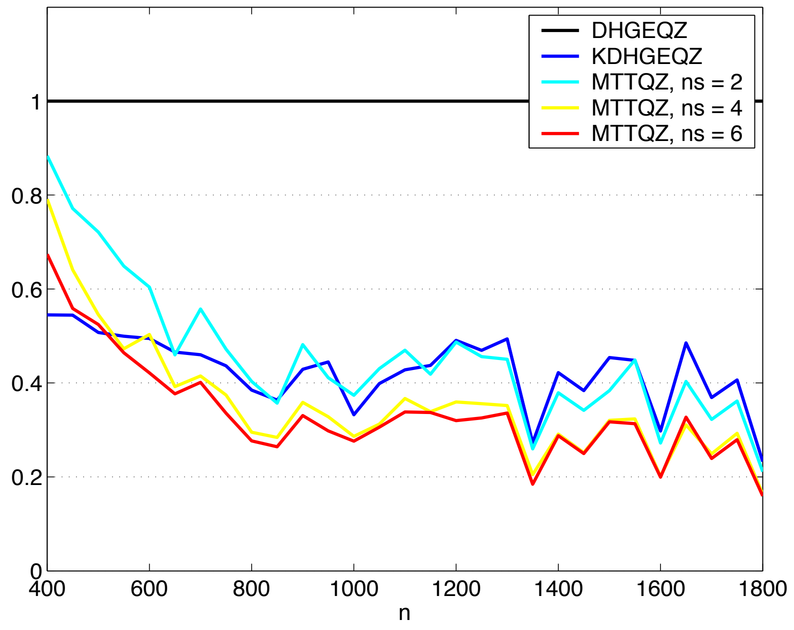


Figure 2.3. *Performance of the tiny-bulge multishift QZ algorithm (*MTTQZ*) relative to the LAPACK routine* DHGEQZ.

80% less cpu time than DHGEQZ for sufficiently large matrix pairs. The improvement upon KDHGEQZ is not very significant. In fact, KDHGEQZ outperforms MTTQZ for matrix pairs of order smaller than 500.

6 Aggressive Early Deflation

Aggressive early deflation, see Section 6.1 in Chapter 1, can be adapted to the QZ algorithm in the following way [3]. Consider a matrix pair (A, B) in Hessenberg-triangular form, partitioned as follows:

$$A = \begin{array}{c} \\ n-w-1 \\ 1 \\ w \end{array} \begin{array}{c} \begin{array}{ccc} n-w-1 & 1 & w \end{array} \\ \left[\begin{array}{ccc} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ 0 & A_{32} & A_{33} \end{array}\right] \end{array}, \quad B = \begin{array}{c} \\ n-w-1 \\ 1 \\ w \end{array} \begin{array}{c} \begin{array}{ccc} n-w-1 & 1 & w \end{array} \\ \left[\begin{array}{ccc} B_{11} & B_{12} & B_{13} \\ 0 & B_{22} & B_{23} \\ 0 & 0 & B_{33} \end{array}\right] \end{array}.$$

Our goal is to construct a correspondingly partitioned perturbation of the form

$$E = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & E_{32} & 0 \end{bmatrix} \tag{2.30}$$

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 $U^T A_{32} = \left[\begin{smallmatrix} \tilde{s}_1 \\ \tilde{s}_2 \end{smallmatrix}\right]$ is correspondingly partitioned. Then the matrix $E_{32} = -U \left[\begin{smallmatrix} 0 \\ \tilde{s}_2 \end{smallmatrix}\right]$ corresponds to a reducing perturbation E of the form (2.30). The Frobenius norm of E is given by $\|\tilde{s}_2\|_2$. We make use of this perturbation if $\tilde{s}_2$ satisfies

$$\|\tilde{s}_2\|_\infty \leq \sqrt{|\det(\tilde{S}_{22})|}. \tag{2.31}$$

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 [51].

If the current ordering of the matrix pair (S_{33}, T_{33}) fails to fulfill the criterion (2.31), we may test other possible choices for $(\tilde{S}_{22}, \tilde{T}_{22})$ by reordering the generalized Schur form, see [132, 134, 238]. If it happens, however, that a reducing perturbation satisfying (2.31) 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 (1.59). 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 2.18 (update of RQ decomposition). The matrix S_{11} is correspondingly updated. Finally, applying Algorithm 2.12, reduction to Hessenberg-triangular form, to the updated matrix pair (S_{11}, T_{11}) returns the large matrix pair (A, B) to Hessenberg-triangular form.

Aggressive early deflation is performed after each multishift QZ iteration. As in the case of QR iterations, it must be combined with the conventional deflation strategy described in Section 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`. In order to present some preliminary numerical results, we have applied `ATTQZ` to the same test matrix pairs that have been used for the numerical results presented in Section 5.4. Also, the employed routine parameters were identical to those used in Section 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 [74] 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 chose an "optimal" window size from the set $\{10, 20, 30, 40, 50\}$ by running five numerical experiments for each matrix pair.

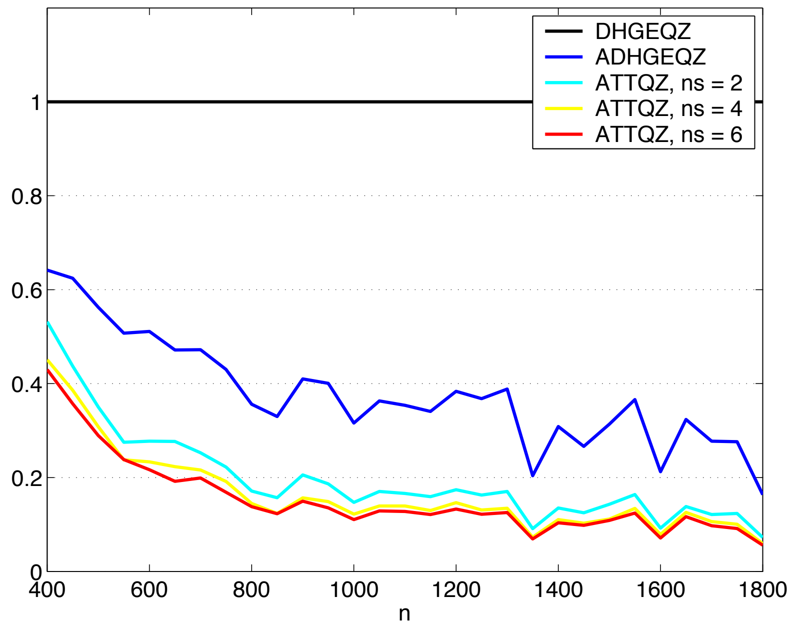


Figure 2.4. *Performance of the tiny-bulge multishift QZ algorithm with aggressive early deflation (*`ATTQZ`*) relative to the LAPACK routine* `DHGEQZ`*.*

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. Further numerical experiments involving more realistic test matrix pairs are necessary in order to confirm these findings. Assembling a collection of such matrix pairs from a variety of application areas is work under progress.

Chapter 3

The Periodic QR Algorithm

In this chapter, we consider the periodic eigenvalue problem, which consists of computing eigenvalues and invariant subspaces of a matrix product

$$\Pi_{\mathcal{A}} = A^{(p)} A^{(p-1)} \cdots A^{(1)}, \tag{3.1}$$

where $A^{(1)}, \dots, A^{(p)} \in \mathbb{R}^{n \times n}$. In principal, one could apply the QR algorithm to the explicitly formed matrix $\Pi_{\mathcal{A}}$. This would yield the exact eigenvalues of a perturbed matrix $\Pi_{\mathcal{A}} + \triangle\Pi_{\mathcal{A}}$, where the norm of the backward error, $\|\triangle\Pi_{\mathcal{A}}\|$, is of order unit roundoff times $\|A^{(p)}\| \cdot \|A^{(p-1)}\| \cdots \|A^{(1)}\|$. Unfortunately, this generally does not imply small backward errors in the factors of $\Pi_{\mathcal{A}}$ [122, Sect. 3.5]. Hence, the eigenvalues obtained with such an approach might be much less accurate than the data deserve.

The *periodic QR algorithm* [46, 120, 241] is a special-purpose method designed for eigenvalue problems of the form (3.1). It achieves a small factor-wise backward error, i.e., the computed eigenvalues are the exact eigenvalues of

$$(A^{(p)} + E^{(p)})(A^{(p-1)} + E^{(p-1)}) \cdots (A^{(1)} + E^{(1)}). \tag{3.2}$$

where each $\|E^{(l)}\|$ is of order unit roundoff times $\|A^{(l)}\|$. The periodic QR algorithm earned its name for being a useful tool in the analysis and design of linear discrete-time periodic systems, see [247] and the references therein. Other applications include queueing network models [48, 227], multiple shooting methods [11, 168], and Hamiltonian eigenvalue problems [37, 38], see also Section 3.6 in Chapter 4.

Contributions in this Chapter

It is a basic linear algebra result that the periodic eigenvalue problem (3.1) can be related to a structured eigenvalue problem involving a $pn \times pn$ block cyclic matrix $\mathcal{A}$. By applying a certain permutation to $\mathcal{A}$, we obtain a cyclic block matrix $\tilde{\mathcal{A}}$. The main contribution in this chapter is to show that if these relations are fully exploited then some problems associated with periodic eigenvalue problems can be addressed in a relatively simple and elegant manner. In particular, it is shown that:

- some known perturbation results for the periodic eigenvalue problem (3.1) with respect to factor-wise perturbations of the form (3.2), see [39, 166], can be obtained as corollaries from perturbation results for the standard eigenvalue problem, see Section 2;

- the periodic QR algorithm is numerically equivalent to the QR algorithm applied to $\tilde{\mathcal{A}}$, see Section 3.

There is also a brief discussion on reordering eigenvalues as well as possible extensions of the obtained results to more general periodic eigenvalue problems. The work on the numerical equivalence between the periodic QR algorithm and the QR algorithm has been accepted for publication, see [149].

1 The Periodic Eigenvalue Problem

The following decomposition plays the same eminent role for the periodic eigenvalue problem (3.1) that the Schur decomposition plays for the standard eigenvalue problem or the generalized Schur decomposition for the generalized eigenvalue problem.

Theorem 3.1 (Periodic Schur decomposition [46, 120]). *Let $A^{(1)},\dots,A^{(p)} \in \mathbb{R}^{n\times n}$, then there exist orthogonal matrices $Q^{(1)},\dots,Q^{(p)} \in \mathbb{R}^{n\times n}$ so that*

$$\begin{array}{rcl} T^{(p)} & = & Q^{(1)T}A^{(p)}Q^{(p)}, \\ T^{(p-1)} & = & Q^{(p)T}A^{(p-1)}Q^{(p-1)}, \\ & \vdots & \\ T^{(1)} & = & Q^{(2)T}A^{(1)}Q^{(1)}, \end{array} \tag{3.3}$$

where $T^{(p)}$ has real Schur form and $T^{(1)},\dots,T^{(p-1)}$ are upper triangular matrices.

The periodic Schur decomposition (3.3) can be written in the more compact form

$$T^{(l)} = Q^{(l+1)T}A^{(l)}Q^{(l)}, \quad l = 1,\dots,p,$$

if we identify $Q^{(p+1)}$ with $Q^{(1)}$. More generally spoken, we will make use of the following convention:

Throughout the entire chapter 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 (3.3), i.e., there are unitary matrices $Q^{(1)},\dots,Q^{(p)}$ so that the matrices

$$T^{(l)} = Q^{(l+1)H}A^{(l)}Q^{(l)}, \quad l = 1,\dots,p, \tag{3.4}$$

are upper triangular. This implies a Schur decomposition for $\Pi_{\mathcal{A}}$:

$$Q^{(1)H}\Pi_{\mathcal{A}}Q^{(1)} = T^{(p)}T^{(p-l)}\cdots T^{(1)} = \left[\ \diagdown\!\!\mid\ \right]. \tag{3.5}$$

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 $t_{ii}^{(p)}\cdot t_{ii}^{(p-1)}\cdots t_{ii}^{(1)}$, $i = 1,\dots,n$. By a suitable reordering of the periodic Schur decomposition, see [120] and Section 4, we can let the eigenvalues of Π_A appear in any desirable order on the diagonals of $T^{(l)}$.

The Schur decomposition (3.5) 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

$$\Pi_{\mathcal{A}}^{(l)} = A^{(p+l-1)}A^{(p+l-2)}\cdots A^{(l)}, \quad l = 1,\dots,p, \tag{3.6}$$

then the first k columns of $Q^{(l)}$ form 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

$$\mathcal{A} = \begin{bmatrix} 0 & & & A^{(p)} \\ A^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & A^{(p-1)} & 0 \end{bmatrix}. \tag{3.7}$$

To see this, let us partition

$$Q^{(l)} = \begin{array}{c} \begin{array}{cc} k & n-k \end{array} \\ \begin{bmatrix} X^{(l)} & X_{\perp}^{(l)} \end{bmatrix} \end{array}, \quad T^{(l)} = \begin{array}{c|cc} & k & n-k \\ \hline k & A_{11}^{(l)} & A_{12}^{(l)} \\ n-k & 0 & A_{22}^{(l)} \end{array}.$$

By setting

$$X = X^{(1)} \oplus X^{(2)} \oplus \cdots \oplus X^{(p)}, \quad X_{\perp} = X_{\perp}^{(1)} \oplus X_{\perp}^{(2)} \oplus \cdots \oplus X_{\perp}^{(p)}, \tag{3.8}$$

and

$$\mathcal{A}_{ij} = \begin{bmatrix} 0 & & & A_{ij}^{(p)} \\ A_{ij}^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & A_{ij}^{(p-1)} & 0 \end{bmatrix}, \tag{3.9}$$

we obtain a block Schur decomposition for $\mathcal{A}$:

$$\mathcal{A}[X, X_{\perp}] = [X, X_{\perp}] \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ 0 & \mathcal{A}_{22} \end{bmatrix}. \tag{3.10}$$

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 $\mathcal{A}$ has a block cyclic representation but in the context of periodic eigenvalue problems it is sufficient to consider this type of subspace.

Definition 3.2 ([167]). *Let $\mathcal{X}$ be a $(p \cdot k)$-dimensional (left) invariant subspace of a block cyclic matrix $\mathcal{A} \in \mathbb{R}^{pn \times pn}$. If there exist matrices $X^{(1)}, X^{(2)}, \dots, X^{(p)} \in \mathbb{R}^{n \times k}$ so that*

$$\mathcal{X} = \operatorname{span}(X^{(1)} \oplus X^{(2)} \oplus \cdots \oplus X^{(p)}),$$

then $\mathcal{X}$ is called a (right) periodic invariant subspace *of $\mathcal{A}$.*

By direct computation, it can be seen that every periodic invariant subspace has 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 3.3. *Let λ be an eigenvalue of the matrix product $\Pi_{\mathcal{A}}$ having the form (3.1). Then $\lambda^{1/p}, \omega\lambda^{1/p}, \dots, \omega^{p-1}\lambda^{1/p}$, where ω is the pth primitive root of unity, are eigenvalues of the block cyclic matrix $\mathcal{A}$ having the form (3.7).*

Proof. By the (complex) periodic Schur decomposition and the construction given above, there exists an invariant subspace of $\mathcal{A}$ corresponding to a representation of the form (3.10) with

$$\mathcal{A}_{11} = \begin{bmatrix} 0 & & & t_{11}^{(p)} \\ t_{11}^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & t_{11}^{(p-1)} & 0 \end{bmatrix}, \quad \lambda = t_{11}^{(p)} \cdot t_{11}^{(p-1)} \cdots t_{11}^{(1)}.$$

The result follows by observing $\mathcal{A}_{11}^p = \lambda I_p$. $\square$

Bhatia [41, Sec. VIII.5] calls a p-tuple of the form $\{\alpha, \omega\alpha, \dots, \omega^{k-1}\alpha\}$ a *p-Carrollian* tuple in honor of the writer and mathematician Lewis Carroll. As any periodic invariant subspace corresponds to a block cyclic representation it does also belong to eigenvalues that form a set of p-Carrollian tuples.

2 Perturbation Theory

Benner, Mehrmann and Xu [39] as well as Lin and Sun [166] independently developed perturbation theories for more general variants of the periodic eigenvalue problem. In this section, we show how the perturbation analysis for (3.1) can be derived via a different approach; by treating the periodic 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 2 in Chapter 1, we start with a block Schur decomposition

$$\mathcal{A}[X, X_\perp] = [X, X_\perp] \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ 0 & \mathcal{A}_{22} \end{bmatrix}. \tag{3.11}$$

However, we assume some extra structure: X and $X_\perp$ have block diagonal form (3.8); $\mathcal{A}_{11} \in \mathbb{C}^{pk \times pk}$, $\mathcal{A}_{12} \in \mathbb{C}^{pk \times p(n-k)}$ and $\mathcal{A}_{22} \in \mathbb{C}^{p(n-k) \times p(n-k)}$ are block cyclic matrices of the form (3.9).

The Sylvester operator associated with (3.11) has a number of useful properties that are summarized in the following lemma.

Lemma 3.4. *Let $\mathbf{T}_\mathcal{A} : \mathbb{C}^{pk \times p(n-k)} \to \mathbb{C}^{pk \times p(n-k)}$ be the Sylvester operator defined by $\mathbf{T}_\mathcal{A} : R \mapsto \mathcal{A}_{11}R - R\mathcal{A}_{22}$, where $\mathcal{A}_{11}$ and $\mathcal{A}_{22}$ are block cyclic matrices of the form (3.9). Then the following statements hold:*

1. *Let $\mathcal{Z}^{(j)}$, $j = 0, \dots, p-1$, be the matrix subspaces*

$$\mathcal{Z}^{(j)} := \Big\{ \mathcal{C}_k^j \cdot \Big(Z^{(1)} \oplus Z^{(2)} \oplus \cdots \oplus Z^{(p)} \Big) \mid Z^{(l)} \in \mathbb{C}^{pk \times p(n-k)} \Big\}, \tag{3.12}$$

where

$$\mathcal{C}_k = \begin{bmatrix} 0 & & & I_k \\ I_k & \ddots & & \\ & \ddots & \ddots & \\ & & I_k & 0 \end{bmatrix},$$

then

$$\mathbf{T}_{\mathcal{A}}\mathcal{Z}^{(0)} \subseteq \mathcal{Z}^{(1)}, \ \ldots, \ \mathbf{T}_{\mathcal{A}}\mathcal{Z}^{(p-2)} \subseteq \mathcal{Z}^{(p-1)}, \ \mathbf{T}_{\mathcal{A}}\mathcal{Z}^{(p-1)} \subseteq \mathcal{Z}^{(0)}. \tag{3.13}$$

2. $\mathbf{T}_{\mathcal{A}}$ *is invertible if and only if* $\lambda(\mathcal{A}_{11}) \cap \lambda(\mathcal{A}_{22}) = \emptyset$, *which is equivalent to the condition* $\lambda\Big(A_{11}^{(p)}A_{11}^{(p-1)}\cdots A_{11}^{(1)}\Big) \cap \lambda\Big(A_{22}^{(p)}A_{22}^{(p-1)}\cdots A_{22}^{(1)}\Big) = \emptyset$.
3. *If* $\mathbf{T}_{\mathcal{A}}$ *is invertible, then*

$$\mathcal{Z}^{(0)} = \mathbf{T}_{\mathcal{A}}^{-1}\mathcal{Z}^{(1)}, \ \ldots, \ \mathcal{Z}^{(p-2)} = \mathbf{T}_{\mathcal{A}}^{-1}\mathcal{Z}^{(p-1)}, \ \mathcal{Z}^{(p-1)} = \mathbf{T}_{\mathcal{A}}^{-1}\mathcal{Z}^{(0)}.$$

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

$$\begin{aligned} \mathcal{A}_{11}R &= \Big(A_{11}^{(p)} \oplus A_{11}^{(1)} \oplus \cdots A_{11}^{(p-1)}\Big) \cdot \mathcal{C}_k^{j+1} \cdot R_1 \\ &\in \Big(A_{11}^{(p)} \oplus A_{11}^{(1)} \oplus \cdots A_{11}^{(p-1)}\Big) \cdot \mathcal{Z}^{(j+1)} \subseteq \mathcal{Z}^{(j+1)} \end{aligned}$$

and

$$\begin{aligned} R\mathcal{A}_{22} &= R_2 \cdot \mathcal{C}_{n-k}^{j+1} \cdot \Big(A_{22}^{(1)} \oplus A_{22}^{(2)} \oplus \cdots A_{22}^{(p)}\Big) \\ &\in \mathcal{Z}^{(j+1)} \cdot \Big(A_{22}^{(1)} \oplus A_{22}^{(2)} \oplus \cdots A_{22}^{(p)}\Big) \subseteq \mathcal{Z}^{(j+1)}, \end{aligned}$$

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 $\mathbf{T}_{\mathcal{A}}\mathcal{Z}^{(j)} \subseteq \mathcal{Z}^{(j+1)}$.

2. This statement follows from Lemma 1.3 and

$$\lambda \in \lambda(\mathcal{A}_{ii}) \quad \Leftrightarrow \quad \lambda^p \in \lambda\Big(A_{ii}^{(p)}A_{ii}^{(p-1)}\cdots A_{ii}^{(1)}\Big), \quad i \in \{1,2\}.$$

3. The fact that $\{\mathcal{Z}^{(0)}, \mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(p-1)}$ is 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 $\mathbf{T}_{\mathcal{A}}$ that the set inclusions (3.13) become

$$\mathbf{T}_{\mathcal{A}}\mathcal{Z}^{(0)} = \mathcal{Z}^{(1)}, \ \ldots, \ \mathbf{T}_{\mathcal{A}}\mathcal{Z}^{(p-2)} = \mathcal{Z}^{(p-1)}, \ \mathbf{T}_{\mathcal{A}}\mathcal{Z}^{(p-1)} = \mathcal{Z}^{(0)},$$

which concludes the proof.

□

If $\mathbf{T}_{\mathcal{A}}$ is invertible, then the left invariant subspace belonging to the eigenvalues of $\mathcal{A}_{11}$ is given by

$$\mathcal{Y} = \operatorname{span}\left([X, X_\perp] \cdot [I, \mathbf{T}_{\mathcal{A}}^{-1}\mathcal{A}_{12}]^H\right).$$

The third statement of Lemma 3.4 shows that $\mathbf{T}_{\mathcal{A}}^{-1}\mathcal{A}_{12} \in \mathcal{Z}^{(0)}$, i.e., this matrix is block diagonal. Hence, $\mathcal{Y}$ is a left *periodic* invariant subspace.

We are now prepared to formulate the main result of this section, a structured expansion theorem for invariant subspaces of block cyclic matrices.

Theorem 3.5. *Let the block cyclic matrix $\mathcal{A} \in \mathbb{C}^{pn\times pn}$ have a block Schur decomposition of the form (3.11) so that $\mathcal{X} = \operatorname{span} X$ is a periodic invariant subspace belonging to the eigenvalues of $\mathcal{A}_{11} \in \mathbb{C}^{pk\times pk}$. Let the columns of $Y = Y^{(1)} \oplus \cdots \oplus Y^{(p)}$ form an orthonormal basis for the corresponding left periodic invariant subspace. Assume that $\lambda(\mathcal{A}_{11})$ is simple and let $\mathcal{E} \in \mathcal{B}(0)$ be a perturbation having the same block cyclic structure as $\mathcal{A}$, where $\mathcal{B}(0) \subset \mathbb{C}^{pn\times pn}$ is a sufficiently small open neighborhood of the origin. Then there exist analytic functions $f_{\mathcal{A}_{11}} : \mathcal{B}(0) \to \mathbb{C}^{pk\times pk}$ and $f_X : \mathcal{B}(0) \to \mathbb{C}^{pn\times pk}$ so that $\mathcal{A}_{11} = f_{\mathcal{A}_{11}}(0)$, $X = f_X(0)$, and the columns of $\hat{X} = f_X(\mathcal{E})$ span a periodic invariant subspace of $\mathcal{A}+\mathcal{E}$ having the block cyclic representation $\hat{\mathcal{A}}_{11} = f_{\mathcal{A}_{11}}(\mathcal{E})$. Moreover $X^H(\hat{X}-X) = 0$, and we have the expansions*

$$\hat{A}_{11} = A_{11} + (Y^H X)^{-1} Y^H \mathcal{E} X + O(\|\mathcal{E}\|^2), \tag{3.14}$$

$$\hat{X} = X - X_\perp \mathbf{T}_{\mathcal{A}}^{-1} X_\perp^H \mathcal{E} X + O(\|\mathcal{E}\|^2), \tag{3.15}$$

with the Sylvester operator $\mathbf{T}_{\mathcal{A}} : Q \mapsto \mathcal{A}_{22} Q - Q \mathcal{A}_{11}$.

Proof. Almost all statements of this theorem follow directly from Theorem 1.9. The only parts to be proven are that $\hat{\mathcal{X}}$ is periodic and that $\hat{\mathcal{A}}_{11}$ is block cyclic.

By applying a Newton-like iteration to the nonlinear equations

$$\begin{aligned} (\mathcal{A}+\mathcal{E})Z - Z\mathcal{B} &= 0, \\ X^H(Z - X) &= 0, \end{aligned}$$

we construct a sequence

$$\begin{bmatrix} Z_0 \\ \mathcal{B}_0 \end{bmatrix} \leftarrow \begin{bmatrix} X \\ \mathcal{A}_{11} \end{bmatrix}, \quad \begin{bmatrix} Z_{i+1} \\ \mathcal{B}_{i+1} \end{bmatrix} \leftarrow \begin{bmatrix} Z_i - X_\perp \mathbf{T}_{\mathcal{A}}^{-1} X_\perp^H \mathcal{R}_i - X(X^H Z_i - I) \\ \mathcal{B}_i + (Y^H X)^{-1} Y^H \mathcal{R}_i \end{bmatrix},$$

where $\mathcal{R}_i = (\mathcal{A}+\mathcal{E})Z_i - Z_i\mathcal{B}$. If $\mathcal{E}$ is sufficiently small then Z_i and R_i converge (linearly) to $\hat{X}$ and $\hat{\mathcal{A}}_{11}$, respectively, see [183]. Note that the third statement of Lemma 3.4 implies that all matrices Z_i stay block diagonal, i.e., there exist matrices $Z_i^{(l)} \in \mathbb{C}^{n\times k}$ so that $Z_i = Z_i^{(1)} \oplus \cdots \oplus Z_i^{(p)}$. Thus, $\hat{X}$ has the form $\hat{X} = \hat{X}^{(1)} \oplus \cdots \oplus \hat{X}^{(p)}$ for some matrices $\hat{X}^{(l)} \in \mathbb{C}^{n\times k}$ implying that $\hat{\mathcal{X}}$ is a periodic invariant subspace of $\mathcal{A}+\mathcal{E}$. Moreover, all iterates $\mathcal{B}_i$ are block cyclic matrices, which shows that the representation $\hat{\mathcal{A}}_{11}$ is also a block cyclic matrix. $\square$

2.1 Eigenvalues

In this section, we apply Theorem 3.5 to derive eigenvalue condition numbers for the periodic eigenvalue problem.

We have seen that to any eigenvalue λ of the periodic eigenvalue problem (3.1) there exists a p-dimensional periodic invariant subspace $\mathcal{X}$ of the corresponding block cyclic matrix $\mathcal{A}$ belonging to the eigenvalues $\lambda^{1/k}, \omega\lambda^{1/k}, \dots, \omega^{k-1}\lambda^{1/k}$. Now, we can choose vectors $x^{(1)}, \dots, x^{(p)} \in \mathbb{C}^n$ 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 $y^{(1)}, \dots, y^{(p)} \in \mathbb{C}^n$, $\|y^{(1)}\|_2 = \cdots = \|y^{(p)}\|_2 = 1$, so that $\mathcal{Y} = \operatorname{span}(Y)$ 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 $\alpha^{(1)}, \dots, \alpha^{(p)}$. 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 (3.14) in Theorem 1.9 proves the existence of a cyclic matrix $\hat{\mathcal{A}}_{11}$ which satisfies

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

where the eigenvalues of $\hat{\mathcal{A}}_{11}$ are eigenvalues of $\mathcal{A} + \mathcal{E}$. This expression can be rewritten in terms of the entries $\hat{\alpha}^{(l)}$ of $\hat{\mathcal{A}}_{11}$:

$$\hat{\alpha}^{(l)} = \alpha^{(l)} + \frac{1}{y^{(l+1)H} x^{(l)}} y^{(l+1)H} E^{(l)} x^{(l)} + \mathcal{O}(\|\mathcal{E}\|^2), \quad l = 1, \dots, p.$$

The condition number for each individual $\alpha^{(l)}$ is thus given by $c(\alpha^{(l)}) = 1/|y^{(l+1)H} x^{(l)}|$. 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

$$\triangle\alpha = \|(Y^H X)^{-1} Y^H \mathcal{E} X\|_F + \mathcal{O}(\|\mathcal{E}\|^2) \leq \|(Y^H X)^{-1}\|_2 \cdot \|\mathcal{E}\|_F + \mathcal{O}(\|\mathcal{E}\|^2).$$

For the block cyclic perturbations $\mathcal{E} = \varepsilon Y \mathcal{C}_1 X^H$ (the matrix $\mathcal{C}_1$ is defined in Lemma 3.4), this inequality is attained in first order. Hence, the structured condition number for $\triangle\alpha$ satisfies

$$c(\triangle\alpha) := \lim_{\varepsilon \to 0} \sup_{\substack{\|\mathcal{E}\|_F \leq \varepsilon \\ \mathcal{E} \text{ is block cyclic}}} \frac{\triangle\alpha}{\varepsilon} = \|(Y^H X)^{-1}\|_2 = \max_{l \in [1,p]} c(\alpha^{(l)}).$$

Note that $c(\triangle\alpha)$ is equal to the condition number for the eigenvalue mean of $\mathcal{A}_{11}$, see Section 2.3 in Chapter 1.

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

$$\begin{aligned}
|\hat{\lambda} - \lambda| &= \Big|\prod_{l=1}^{p}[(\hat{\alpha}^{(l)} - \alpha^{(l)}) + \alpha^{(l)}] - \prod_{l=1}^{p}\alpha^{(l)}\Big| \\
&= \Big|\sum_{l=1}^{p}\Big[(\hat{\alpha}^{(l)} - \alpha^{(l)})\prod_{k\neq l}\alpha^{(k)}\Big]\Big| + \mathcal{O}(\|\mathcal{E}\|^2) \\
&\leq \sum_{l=1}^{p}\Big[|\hat{\alpha}^{(l)} - \alpha^{(l)}| \cdot \prod_{k\neq l}|\alpha^{(k)}|\Big] + \mathcal{O}(\|\mathcal{E}\|^2) \\
&\leq \sum_{l=1}^{p}\Big[c(\alpha^{(l)}) \cdot \|E^{(l)}\|_2 \cdot \prod_{k\neq l}|\alpha^{(k)}|\Big] + \mathcal{O}(\|\mathcal{E}\|^2).
\end{aligned}$$

2.2 Invariant Subspaces

Under the assumptions of Theorem 3.5, the perturbation expansion (3.15) implies

$$\|\hat{X} - X\|_F = \|X_\perp \mathbf{T}_{\mathcal{A}}^{-1} X_\perp^H \mathcal{E} X\|_F + O(\|\mathcal{E}\|^2). \tag{3.16}$$

Let us consider the matrix subspaces $\mathcal{Z}^{(0)}, \dots, \mathcal{Z}^{(p-1)}$ defined as in (3.12) but with the roles of k and $n-k$ interchanged. By setting

$$c_{\mathcal{A}}(\mathcal{X}) = \sup_{R\in\mathcal{Z}^{(1)}\backslash\{0\}} \frac{\|\mathbf{T}_{\mathcal{A}}^{-1}(R)\|_F}{\|R\|_F}$$

and observing $X_\perp^H \mathcal{E} X \in \mathcal{Z}^{(1)}$, it can be seen that (3.16) yields the perturbation bound

$$\|\hat{X} - X\|_F \leq c_{\mathcal{A}}(\mathcal{X}) \cdot \|\mathcal{E}\|_F + O(\|\mathcal{E}\|^2).$$

This inequality can be approximately attained by any block cyclic matrix $\mathcal{E} = \varepsilon X_\perp R_0 X^H$, where $R_0 \in \mathcal{Z}^{(1)}$ satisfies $\|R_0\|_F = 1$ and $c_{\mathcal{A}}(\mathcal{X}) = \|\mathbf{T}_{\mathcal{A}}^{-1}(R_0)\|_F$. The existence of such a matrix R_0 is guaranteed by the Weierstrass theorem. Using the same arguments as in Section 2.3, Chapter 1, $c_{\mathcal{A}}(\mathcal{X})$ can be seen as the *structured condition number for a periodic invariant subspace* $\mathcal{X}$:

$$\lim_{\varepsilon\to 0} \sup_{\substack{\|\mathcal{E}\|_F\leq\varepsilon \\ \mathcal{E}\text{ is block cyclic}}} \frac{\|\Theta(\mathcal{X}, \hat{\mathcal{X}})\|_F}{\varepsilon} = c_{\mathcal{A}}(\mathcal{X}).$$

In contrast, the unstructured condition number for $\mathcal{X}$ is given by

$$c(\mathcal{X}) = \sup_{R\neq 0} \frac{\|\mathbf{T}_{\mathcal{A}}^{-1}(R)\|_F}{\|R\|_F} = \max_{j\in[0,p-1]} \left\{ \sup_{R\in\mathcal{Z}^{(j)}\backslash\{0\}} \frac{\|\mathbf{T}_{\mathcal{A}}^{-1}(R)\|_F}{\|R\|_F} \right\}.$$

Note that $c_{\mathcal{A}}(\mathcal{X})$ and $c(\mathcal{X})$ may differ significantly as shown by the following example.

Example 3.6. *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

$$c_{\mathcal{A}}(\mathcal{X}) = \left\| \begin{bmatrix} C & -I_2 \\ 0 & D \end{bmatrix}^{-1} \right\|_2 = \sqrt{2} \times 10^5,$$

while the unstructured condition number is much higher,

$$c(\mathcal{X}) = \max \left\{ c_{\mathcal{A}}, \left\| \begin{bmatrix} D & -I_2 \\ 0 & C \end{bmatrix}^{-1} \right\|_2 \right\} = 10^{10}.$$

The quantity $c_{\mathcal{A}}(\mathcal{X})$ measures the overall conditioning of $\mathcal{X}$, which can be written as the direct sum of the subspaces $\mathcal{X}^{(1)} = \operatorname{span} X^{(1)}, \ \ldots, \ \mathcal{X}^{(p)} = \operatorname{span} X^{(p)}$. 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 condition number is given by

$$c_{\mathcal{A}}^{(1)}(\mathcal{X}) = \sup_{R \in \mathcal{Z}^{(1)} \setminus \{0\}} \frac{\|E_1^T \cdot \mathbf{T}_{\mathcal{A}}^{-1}(R)\|_F}{\|R\|_F},$$

where E_1 contains the first $(n-k)$ columns of $I_{p(n-k)}$, see also [166]. In a similar fashion, one can derive condition numbers for the individual subspaces $\mathcal{X}^{(2)}, \ldots, \mathcal{X}^{(p)}$.

On the computation of $c_{\mathcal{A}}(\mathcal{X})$

Computing the structured condition number $c_{\mathcal{A}}(\mathcal{X})$ is equivalent to computing the norm of the inverse of the Sylvester operator $\mathbf{T}_{\mathcal{A}} : \mathcal{Z}^{(0)} \to \mathcal{Z}^{(1)}$ with $\mathbf{T}_{\mathcal{A}} : Q \mapsto \mathcal{A}_{22} Q - Q \mathcal{A}_{11}$. If we let

$$\begin{aligned} Q &= (Q^{(1)} \oplus \cdots \oplus Q^{(p)}), \\ \mathbf{T}_{\mathcal{A}}(Q) &= \mathcal{C}_{n-k}(R^{(1)} \oplus \cdots \oplus R^{(p)}), \end{aligned}$$

then

$$\begin{array}{rcl} A_{22}^{(p)} Q^{(p)} - Q^{(1)} A_{11}^{(p)} &=& R^{(p)}, \\ A_{22}^{(1)} Q^{(1)} - Q^{(2)} A_{11}^{(1)} &=& R^{(1)}, \\ &\vdots& \\ A_{22}^{(p-1)} Q^{(p-1)} - Q^{(p)} A_{11}^{(p-1)} &=& R^{(p-1)}. \end{array} \tag{3.17}$$

The system of matrix equations (3.17) is called a *periodic Sylvester equation.* It is equivalent to the linear system of equations

$$K_{\mathbf{T}_{\mathcal{A}}} \begin{bmatrix} \operatorname{vec}(Q^{(1)}) \\ \vdots \\ \operatorname{vec}(Q^{(p-1)}) \\ \operatorname{vec}(Q^{(p)}) \end{bmatrix} = \begin{bmatrix} \operatorname{vec}(R^{(1)}) \\ \vdots \\ \operatorname{vec}(R^{(p-1)}) \\ \operatorname{vec}(R^{(p)}) \end{bmatrix}, \tag{3.18}$$

where $K_{\mathbf{T}_{\mathcal{A}}}$ can be written as the sum of a block diagonal and a block cyclic matrix having the form

$$K_{\mathbf{T}_{\mathcal{A}}} = \begin{bmatrix} I_k \otimes A_{22}^{(1)} & -A_{11}^{(1)} \otimes I_{n-k} & & \\ & \ddots & \ddots & \\ & & I_k \otimes A_{22}^{(p-1)} & -A_{11}^{(p-1)} \otimes I_{n-k} \\ -A_{11}^{(p)} \otimes I_{n-k} & & & I_k \otimes A_{22}^{(p)} \end{bmatrix}.$$

Thus $\|K_{\mathbf{T}_\mathcal{A}}^{-1}\|_2 = c_\mathcal{A}(\mathcal{X})$, which can be estimated by applying a norm estimator [122, Ch. 14] to $K_{\mathbf{T}_\mathcal{A}}^{-1}$. This amounts to the solution of a few linear equations $K_{\mathbf{T}_\mathcal{A}} q_1 = r_1$ and $K_{\mathbf{T}_\mathcal{A}}^T q_2 = r_2$ 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 (3.17) and periodic matrix equations of similar type can be found in [66, 215, 245].

3 Derivation of the Periodic QR Algorithm

The periodic QR algorithm was developed independently by Bojanczyk/Golub/Van Dooren [46] and Hench/Laub [120] in the beginning of the nineties. Already in 1975, Van Loan [241] published an algorithm for the case $p = 2$ containing the main ideas behind the periodic QR algorithm. Van Loan's paper is based on his PhD thesis [240], 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 was always introduced as an independent algorithm. In this section, we show that the standard QR algorithm applied to a shuffled version of the block cyclic matrix $\mathcal{A}$ is numerically equivalent to the periodic QR algorithm applied to the matrix product $\Pi_\mathcal{A}$.

3.1 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, e.g., in the construction of algorithms for block Toeplitz matrices [104]. The block structure of our interest is a block cyclic matrix of the form

$$\mathcal{A} = \begin{bmatrix} 0 & & & A^{(p)} \\ A^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & A^{(p-1)} & 0 \end{bmatrix}. \tag{3.19}$$

The perfect shuffle permutation can be obtained as follows. Let

$$z = [\, z^{(1)}, z^{(2)}, \dots, z^{(p)} \,],$$

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

$$\tilde{z} = [\, z_1^{(1)}, z_1^{(2)}, \dots, z_1^{(p)}, z_2^{(1)}, \dots, z_2^{(p)}, \dots, z_n^{(1)}, \dots, z_n^{(p)} \,].$$

There exists a unique permutation matrix $P \in \mathbb{R}^{pn \times pn}$ such that $\tilde{z} = zP$. Applying this permutation to the block cyclic matrix $\mathcal{A}$ turns it into an $n \times n$ block matrix with cyclic blocks:

$$\tilde{\mathcal{A}} := P^T \mathcal{A} P = \begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & & \vdots \\ A_{n1} & \cdots & A_{nn} \end{bmatrix}, \; A_{ij} := \begin{bmatrix} 0 & & & a_{ij}^{(p)} \\ a_{ij}^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & a_{ij}^{(p-1)} & 0 \end{bmatrix}. \quad (3.20)$$

Any matrix of the form (3.20) will be called *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 $p \times p$ diagonal matrices as entries. We refer to any matrix of the latter form as *diagonal block matrix.* The following straightforward lemma shows a useful relation between cyclic and diagonal block matrices.

Lemma 3.7. *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

Now, the periodic Schur decomposition can be interpreted as a block Schur decomposition for cyclic block matrices, see also Figure 3.1.

Corollary 3.8. *Let $\tilde{\mathcal{A}} \in \mathbb{R}^{pn \times pn}$ be a cyclic block matrix of the form (3.20). 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 3.1 to the $n \times n$ coefficient matrices $A^{(1)}, \dots, A^{(p)}$ associated with $\tilde{\mathcal{A}}$ and setting

$$\tilde{Q} = P^T (Q^{(1)} \oplus Q^{(2)} \oplus \cdots \oplus Q^{(p)}) P,$$

where P is the perfect shuffle matrix. $\square$

3.2 Reduction to 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 1.23 is the basic algorithm for such a reduction based on Householder matrices. Let us recall that a Householder matrix that maps the last $n-j$ elements of a vector $x \in \mathbb{R}^n$ to zero is given by

$$H_j(x) = I - \beta \cdot v_j(x) v_j(x)^T,$$

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 1.23 preserves cyclic block structures.

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

$$\left[\begin{array}{ccc|ccc|ccc|ccc} 0 & 0 & c_{11} & 0 & 0 & c_{12} & 0 & 0 & c_{13} & 0 & 0 & c_{14} \\ a_{11} & 0 & 0 & a_{12} & 0 & 0 & a_{13} & 0 & 0 & a_{14} & 0 & 0 \\ 0 & b_{11} & 0 & 0 & b_{12} & 0 & 0 & b_{13} & 0 & 0 & b_{14} & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & c_{22} & 0 & 0 & c_{23} & 0 & 0 & c_{24} \\ 0 & 0 & 0 & a_{22} & 0 & 0 & a_{23} & 0 & 0 & a_{24} & 0 & 0 \\ 0 & 0 & 0 & 0 & b_{22} & 0 & 0 & b_{23} & 0 & 0 & b_{24} & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & c_{32} & 0 & 0 & c_{33} & 0 & 0 & c_{34} \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{33} & 0 & 0 & a_{34} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{33} & 0 & 0 & b_{34} & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c_{44} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{44} & 0 \end{array}\right]$$

Figure 3.1. *Relation between periodic Schur form and block Schur form of a cyclic block matrix.*

Theorem 3.9. *If Algorithm 1.23 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 1.23 the matrix $\tilde{\mathcal{A}}$ has been overwritten by a cyclic block matrix. Then,

$$\tilde{\mathcal{A}} e_j = y \otimes e_{l'}, \quad y = \left[\, a_{1k}^{(l)}, a_{2k}^{(l)}, \dots, a_{nk}^{(l)} \, \right]^T, \tag{3.21}$$

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 3.7 shows that the j-th loop of Algorithm 1.23 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. $\square$

Hence, Algorithm 1.23 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 $A^{(1)}, \dots, A^{(p)}$. In the following, we will derive such a reformulation. First note that the proof of Theorem 3.9 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

$$\begin{cases} l < p: & A^{(l+1)} \leftarrow A^{(l+1)} \cdot H_k(A^{(l)} e_k), \quad A^{(l)} \leftarrow H_k(A^{(l)} e_k)^T \cdot A^{(l)}, \\ l = p: & A^{(1)} \leftarrow A^{(1)} \cdot H_{k+1}(A^{(p)} e_k), \quad A^{(p)} \leftarrow H_{k+1}(A^{(p)} e_k)^T \cdot A^{(p)}, \end{cases}$$

where the quantities k and l are defined as in (3.21). Furthermore, if we set

$$\tilde{\mathcal{Q}} = P^T \mathrm{diag}(Q^{(1)}, Q^{(2)}, \dots, Q^{(p)}) P,$$

then $\tilde{Q} \leftarrow \tilde{Q}H_{j+1}(\tilde{A}e_j)$ 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 1.23 in the following way.

Algorithm 3.10.

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 $k \leftarrow 1, \ldots, n-1$
  FOR $l \leftarrow 1, \ldots, p-1$
    $Q^{(l+1)} \leftarrow Q^{(l+1)} \cdot H_k(A^{(l)}e_k)$
    $A^{(l+1)} \leftarrow A^{(l+1)} \cdot H_k(A^{(l)}e_k)$
    $A^{(l)} \leftarrow H_k(A^{(l)}e_k)A^{(l)}$
  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

$$\left[\begin{array}{ccc|ccc|ccc|ccc}
0 & 0 & c_{11} & 0 & 0 & c_{12} & 0 & 0 & c_{13} & 0 & 0 & c_{14} \\
a_{11} & 0 & 0 & a_{12} & 0 & 0 & a_{13} & 0 & 0 & a_{14} & 0 & 0 \\
0 & b_{11} & 0 & 0 & b_{12} & 0 & 0 & b_{13} & 0 & 0 & b_{14} & 0 \\ \hline
0 & 0 & c_{21} & 0 & 0 & c_{22} & 0 & 0 & c_{23} & 0 & 0 & c_{24} \\
0 & 0 & 0 & a_{22} & 0 & 0 & a_{23} & 0 & 0 & a_{24} & 0 & 0 \\
0 & 0 & 0 & 0 & b_{22} & 0 & 0 & b_{23} & 0 & 0 & b_{24} & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & c_{32} & 0 & 0 & c_{33} & 0 & 0 & c_{34} \\
0 & 0 & 0 & 0 & 0 & 0 & a_{33} & 0 & 0 & a_{34} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{33} & 0 & 0 & b_{34} & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c_{43} & 0 & 0 & c_{44} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{44} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_{44} & 0
\end{array}\right]$$

Figure 3.2. *Relation between periodic Hessenberg form and Hessenberg form of a cyclic block matrix.*

Note that this Algorithm corresponds to the reduction to periodic Hessenberg form described in [46, Pg. 5–7]. It should be emphasized that Algorithm 3.10 performs exactly the same operations as Algorithm 1.23 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 3.11. *If $p = 2$ and $A^{(1)} = A^{(2)T}$ then Algorithm 1.23 reduces the symmetric matrix $\tilde{\mathcal{A}}$ to tridiagonal form. On the other hand, Algorithm 3.10 returns $Q^{(2)T}A^{(1)}Q^{(1)}$ in bidiagonal form. Hence, as a special case we obtain that bidiagonal reduction (see, e.g, [223, 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 [185].*

3.3 QR Iteration

A QR iteration applied to a general Hessenberg matrix A as described in Algorithm 1.25 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 1.23. The following theorem shows that QR iterations preserve cyclic block structures if m is wisely chosen.

Theorem 3.12. *If Algorithm 1.25 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 3.9 concludes the proof. □

The subdiagonal of $\tilde{\mathcal{A}}$ consists of the diagonals of $A^{(1)}, \dots, A^{(p-1)}$ and the subdiagonal of $A^{(p)}$. Hence, the Hessenberg matrix $\tilde{\mathcal{A}}$ 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 3.12 gives a way to rewrite Algorithm 1.25 in terms of operations on the factors of $\tilde{\mathcal{A}}$.

Algorithm 3.13 (Periodic QR iteration).

Input: *Nonsingular upper triangular matrices $A^{(1)}, \dots, A^{(p-1)} \in \mathbb{R}^{n\times n}$ 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 $Q^{(1)}, \dots, Q^{(p)}$ so that $Q^{(l+1)T} A^{(l)} Q^{(l)}$, $l = 1, \dots, p$, 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 $\{\gamma_1, \dots, \gamma_t\}$ 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 3.10 to compute orthogonal matrices $Q^{(1)}, \dots, Q^{(p)}$ so that $A^{(1)} \leftarrow Q^{(2)T} A^{(1)} Q^{(1)}, \dots, A^{(p-1)} \leftarrow Q^{(p)T} A^{(p-1)} Q^{(p-1)}$ are upper triangular and $A^{(p)} \leftarrow Q^{(1)T} A^{(p)} Q^{(p)}$ is in Hessenberg form.
5. Update $Q^{(1)} \leftarrow H_1(x) \cdot Q^{(1)}$.

This algorithm is a 'Householder version' of the periodic QR iteration with t shifts [46, pg. 11–12].

Example 3.14. *This is a continuation of Example 3.11. If $A^{(1)}$ and $A^{(2)} = A^{(1)T}$ satisfy the assumptions of Algorithm 3.13 then $A^{(1)}$ is a bidiagonal matrix with nonzero diagonal and supdiagonal elements. Algorithm 1.25 applied to the tridiagonal matrix $\tilde{\mathcal{A}}$ performs an implicit shifted symmetric QR iteration, while Algorithm 3.13 performs a bidiagonal QR iteration, see, e.g., [223, Alg. 3.4]. This shows that both QR iterations are numerically equivalent.*

3.4 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 if it is small compared to the neighboring diagonal elements, see Section 3.4. This is however not a very sensible choice for matrices with zero diagonal 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

$$|a^{(p)}_{j+1,j}| \le \mathbf{u}(|a^{(p)}_{j,j}| + |a^{(p)}_{j+1,j+1}|), \tag{3.22}$$

$$|a^{(l)}_{j,j}| \le \mathbf{u}(|a^{(l)}_{j-1,j}| + |a^{(l)}_{j,j+1}|), \quad l = 1, \dots, p-1, \tag{3.23}$$

where an entry on the right hand side of (3.23) is replaced by zero if the index is out of range. Note that inequality (3.23) may only be satisfied if the 2-norm condition number of $A^{(l)}$ is not less than $1/(2\mathbf{u})$.

Situation (3.22) 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 (3.23) yields a deflation into two smaller eigenproblems which do not carry the structure of $\tilde{\mathcal{A}}$. For illustration, consider the case $p = 3$, $n = 4$ and $a_{22}^{(1)} = 0$:

$$\left[\begin{array}{ccc|c||cc|ccc|ccc}
0 & 0 & a_{11}^{(3)} & 0 & 0 & a_{12}^{(3)} & 0 & 0 & a_{13}^{(3)} & 0 & 0 & a_{14}^{(3)} \\
a_{11}^{(1)} & 0 & 0 & a_{12}^{(1)} & 0 & 0 & a_{13}^{(1)} & 0 & 0 & a_{14}^{(1)} & 0 & 0 \\
0 & a_{11}^{(2)} & 0 & 0 & a_{12}^{(2)} & 0 & 0 & a_{13}^{(2)} & 0 & 0 & a_{14}^{(2)} & 0 \\ \hline
0 & 0 & a_{21}^{(3)} & 0 & 0 & a_{22}^{(3)} & 0 & 0 & a_{23}^{(3)} & 0 & 0 & a_{24}^{(3)} \\ \hline\hline
0 & 0 & 0 & 0 & 0 & 0 & a_{23}^{(1)} & 0 & 0 & a_{24}^{(1)} & 0 & 0 \\
0 & 0 & 0 & 0 & a_{22}^{(2)} & 0 & 0 & a_{23}^{(2)} & 0 & 0 & a_{24}^{(2)} & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & a_{32}^{(3)} & 0 & 0 & a_{33}^{(3)} & 0 & 0 & a_{34}^{(3)} \\
0 & 0 & 0 & 0 & 0 & 0 & a_{33}^{(1)} & 0 & 0 & a_{34}^{(1)} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{33}^{(2)} & 0 & 0 & a_{34}^{(2)} & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{43}^{(3)} & 0 & 0 & a_{44}^{(3)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{44}^{(1)} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{44}^{(2)} & 0
\end{array}\right]$$

Fortunately, there is an easy way to force deflations at $a_{21}^{(3)}$ and $a_{32}^{(3)}$ so that afterwards the deflation stemming from $a_{22}^{(1)}$ 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 $a_{21}^{(3)}$ 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 shifts preserves the structure of $\tilde{\mathcal{A}}$ and gives the zero $a_{32}^{(3)}$ element. Using the results of Section 3.3, it can be shown that this procedure is numerically equivalent to the deflation strategy presented in [46, pg. 7–9]. A notable difference is that the deflation criteria suggested in [46] are based on the norms of the factors $A^{(l)}$ instead of the magnitudes of nearby entries.

3.5 Summary

We have shown that reduction to Hessenberg form as well as QR iterations preserve cyclic block structures. If the factors $A^{(1)}, \dots, A^{(p-1)}$ 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. We hope that this connection may lead to a better understanding not only of the periodic QR algorithm but also of other algorithms used for analyzing and designing periodic systems [247].

4 Computation of Invariant Subspaces

Let us assume that the periodic QR algorithm has computed a (real) periodic Schur decomposition

$$T^{(l)} = Q^{(l+1)T} A^{(l)} Q^{(l)}, \quad l = 1, \dots, p,$$

where $T^{(p)}$ has real Schur form and $T^{(1)}, \dots, T^{(p-1)}$ 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 similar bubble-sort-like fashion as for a standard Schur decomposition described in Section 7, Chapter 1, it is sufficient to develop a swapping algorithm for the periodic Schur decomposition. That is, the computation of orthogonal matrices $Q^{(1)}, \dots, Q^{(p)}$ so that

$$Q^{(l+1)T} \left[\begin{array}{cc} A_{11}^{(l)} & A_{12}^{(l)} \\ 0 & A_{22}^{(l)} \end{array} \right] Q^{(l)} = \left[\begin{array}{cc} B_{11}^{(l)} & B_{12}^{(l)} \\ 0 & B_{22}^{(l)} \end{array} \right], \quad l = 1, \dots, p, \tag{3.24}$$

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

$$\begin{array}{rcl} \lambda(A_{11}^{(p)} A_{11}^{(p-1)} \cdots A_{11}^{(1)}) & = & \lambda(B_{22}^{(p)} B_{22}^{(p-1)} \cdots B_{22}^{(1)}), \\ \lambda(A_{22}^{(p)} A_{22}^{(p-1)} \cdots A_{22}^{(1)}) & = & \lambda(B_{11}^{(p)} B_{11}^{(p-1)} \cdots B_{11}^{(1)}). \end{array}$$

Not surprisingly, swapping blocks in a periodic Schur decomposition can be related to swapping blocks in a standard Schur decomposition. If we partition

$$Q^{(l)} = \begin{array}{c c} & \begin{array}{cc} n_2 & n_1 \end{array} \\ \begin{array}{c} n_1 \\ n_2 \end{array} & \left[\begin{array}{cc} Q_{11}^{(l)} & Q_{12}^{(l)} \\ Q_{21}^{(l)} & Q_{22}^{(l)} \end{array} \right] \end{array}$$

and set

$$\mathcal{Q} = \left[\begin{array}{cc} Q_{11}^{(1)} \oplus \cdots \oplus Q_{11}^{(p)} & Q_{12}^{(1)} \oplus \cdots \oplus Q_{12}^{(p)} \\ Q_{21}^{(1)} \oplus \cdots \oplus Q_{21}^{(p)} & Q_{22}^{(1)} \oplus \cdots \oplus Q_{22}^{(p)} \end{array} \right], \tag{3.25}$$

then (3.24) is equivalent to

$$\mathcal{Q}^T \left[\begin{array}{cc} \mathcal{A}_{11} & \mathcal{A}_{12} \\ 0 & \mathcal{A}_{22} \end{array} \right] \mathcal{Q} = \left[\begin{array}{cc} \mathcal{B}_{11} & \mathcal{B}_{12} \\ 0 & \mathcal{B}_{22} \end{array} \right], \tag{3.26}$$

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 (3.9).

In principal, the orthogonal factor $\mathcal{Q}$ in (3.26) can be constructed as described in Section 7.1, Chapter 1. Assuming that $\lambda(\mathcal{A}_{11}) \cap \lambda(\mathcal{A}_{22}) = \emptyset$, the solution of the Sylvester equation

$$\mathcal{A}_{11}\mathcal{X} - \mathcal{X}\mathcal{A}_{22} = \gamma \mathcal{A}_{12} \tag{3.27}$$

for some scaling factor $\gamma \leq 1$ is computed. Now, $\mathcal{Q}$ can be determined from a QR decomposition

$$\mathcal{Q}^T \left[\begin{array}{c} -\mathcal{X} \\ \gamma I_{pn_2} \end{array} \right] = \left[\begin{array}{c} \mathcal{R} \\ 0 \end{array} \right].$$

Since Lemma 3.4 implies that $\mathcal{X}$ is a block diagonal matrix, the factor $\mathcal{Q}$ can always be chosen so that it has the form (3.25).

The finite precision properties of this algorithm are as follows. Let $\hat{Q}^{(1)}, \dots, \hat{Q}^{(p)}$ denote the computed orthogonal factors, then the exact swapping relation (3.24) is perturbed to

$$\hat{Q}^{(l+1)T} \left[\begin{array}{cc} A_{11}^{(l)} & A_{12}^{(l)} \\ 0 & A_{22}^{(l)} \end{array} \right] \hat{Q}^{(l)} = \left[\begin{array}{cc} \hat{B}_{11}^{(l)} & \hat{B}_{12}^{(l)} \\ \hat{B}_{21}^{(l)} & \hat{B}_{22}^{(l)} \end{array} \right], \quad l = 1, \dots, p.$$

Let us assume that the Sylvester equation (3.27) is solved by applying Gaussian elimination with complete pivoting to its Kronecker product formulation, see also (3.18). Then from the discussion in Section 7.1, Chapter 1, it can be expected that the subdiagonal blocks $\hat{B}_{21}^{(l)}$ almost always satisfy

$$\|\hat{B}_{21}^{(l)}\|_F \leq c^{(l)}\mathbf{u}(\|\mathcal{A}_{11}\|_F + \|\mathcal{A}_{12}\|_F + \|\mathcal{A}_{22}\|_F) \tag{3.28}$$

for some small 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 (3.28) 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 developed by Kågström and Poromaa [132, 134].

5 Further Notes and References

The periodic 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 [46, 120]. If $\mathcal{B}$ is invertible, this corresponds to the computation of eigenvalues and invariant subspaces of the general matrix product

$$A^{(p)}[B^{(p)}]^{-1}A^{(p-1)}[B^{(p-1)}]^{-1}\cdots A^{(1)}[B^{(1)}]^{-1}.$$

Perturbation analyses for this type of generalized periodic eigenvalue problems can be found in [39, 166]. The approach developed in Section 2 can be generalized to this situation without any difficulties using perturbation results for the (standard) generalized eigenvalue problem, see Section 2 in Chapter 2. If P denotes the perfect shuffle matrix, then the reduction to Hessenberg-triangular form as described in Algorithm 2.12 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, see also [46, 120, 241]. Along the lines of the proof of Theorem 3.12, it can be shown that the QZ iteration based on Householder matrices, see Algorithm 2.16, preserves the structure of $(\tilde{\mathcal{A}}, \tilde{\mathcal{B}})$. This leads to the periodic QZ iteration described in [46, 120].

Even more general products of the form

$$[A^{(p)}]^{s^{(p)}}[A^{(p-1)}]^{s^{(p-1)}}\cdots[A^{(1)}]^{s^{(1)}}, \quad s^{(1)},\dots,s^{(p)} \in \{1,-1\}.$$

can be found in [28, 39, 110]. Such products can be related to block cyclic/diagonal matrix pairs by introducing identities at appropriate places [39].

Another generalization of the periodic eigenvalue problem is to allow varying dimensions, i.e., $A^{(l)} \in \mathbb{R}^{n^{(l)}\times n^{(l+1)}}$ for some integers $n^{(1)},\dots,n^{(p)}$. Varga [246] has developed a finite algorithm reducing this problem to an equivalent periodic eigenvalue problem involving p square factors of order $n = \min\{n^{(1)},\dots,n^{(p)}\}$.

The classification of (generalized) periodic eigenvalue problems with respect to similarity/equivalence transformations is well-understood in the field of representation theory, see e.g. [88, 180]. Very readable introductions to this area can be found in papers by Sergeichuk [203, 204]. A GUPTRI-like algorithm for (generalized) periodic eigenvalue problems is described in [204].

There exist several implementations of the periodic QR/QZ algorithms for computing eigenvalues of real periodic eigenvalue problems [36, 146, 147, 168]. An implementation of the periodic QZ algorithm for complex periodic eigenvalue problems is described in Section 4, Appendix B. It also includes a routine (`ZPTORD`) for computing periodic invariant subspaces based on an alternative algorithm proposed by Bojanczyk and Van Dooren [45]. For some preliminary work on balancing periodic eigenvalue problems, see [146].

Chapter 4

QR-Based Algorithms for Skew-Hamiltonian and Hamiltonian Matrices

This chapter is devoted to two classes of structured matrices, skew-Hamiltonian and Hamiltonian matrices. A *skew-Hamiltonian matrix* has the form

$$W = \begin{bmatrix} A & G \\ Q & A^T \end{bmatrix}, \quad G = -G^T,\ Q = -Q^T, \tag{4.1}$$

while a *Hamiltonian matrix* reads as

$$H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}, \quad G = G^T,\ Q = Q^T, \tag{4.2}$$

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 [62, 49, 107], see also Section 3.1 in Appendix A;
- linear quadratic optimal control and the solution of continuous-time algebraic Riccati equations [25, 173, 205];
- the solution of anti-symmetric Riccati equations [216];
- H_∞ control [29];
- passivity preserving model reduction [8, 214];
- quadratic eigenvalue problems [175, 236];
- computation of pseudospectra [58] and the distance to uncontrollability [115, 57], see also Section 3.2 in Appendix A.

We have already seen in Chapter 3 that an important question to be discussed when dealing with structured eigenvalue problems is what kind of advantages can principally be expected from exploiting structures. With respect to accuracy of computed eigenvalues and invariant subspaces this question leads to the notion of structured condition numbers and their relationship to unstructured ones. It is interesting to note that the two matrix structures under consideration differ significantly in this aspect. While it is absolutely necessary to use a structure-preserving algorithm for computing invariant subspaces of skew-Hamiltonian matrices, the merits of structure preservation for Hamiltonian matrices are of a more subtle nature and not always relevant in applications. If one is interested in efficiency then there is not so much that can be expected. Both matrix classes depend on $2n^2 + \mathcal{O}(n)$ parameters compared to $4n^2$ parameters of a general $2n \times 2n$ matrix. Hence, a structure-preserving algorithm can be expected to be at best a decent factor faster than a general-purpose method; for the matrix classes considered here, this factor is usually in the range of 2–3, see [27, 33, 38] and Section 6.

Another important question is whether it is actually possible to design an algorithm capable of achieving the possible advantages mentioned above. An ideal method tailored to the matrix structure would

- be strongly backward stable in the sense of Bunch described in [52], i.e., the computed solution is the exact solution corresponding to a nearby matrix with the same structure;
- be reliable, i.e., capable to solve (almost) all eigenvalue problems in the considered matrix class; and
- require $\mathcal{O}(n^3)$ floating point operations (flops), preferably less than a competitive general-purpose method.

We have seen that such a method exists for block cyclic matrices and there is also an ideal method for skew-Hamiltonian matrices by Van Loan [243]. However, it has been a long-standing open problem to develop an ideal method for the Hamiltonian eigenvalue problem. So far there is no method known that meets all three requirements satisfactorily.

The main purpose of this chapter is to survey theory and algorithms for small to medium-sized (skew-)Hamiltonian eigenvalue problems. With respect to algorithms, the account will necessarily be rather incomplete, simply because of the vast number of algorithms that have been developed. Instead, our focus will be on methods that are based on orthogonal transformations.

The structure of this chapter is as follows. After having introduced some notation and preliminary material in the first section, we devote the second section to the skew-Hamiltonian eigenvalue problem. We review structured Hessenberg-like, Schur-like and block diagonal decompositions. This is followed by a discussion on structured condition numbers for eigenvalues and invariant subspaces. The section is concluded by a description of the ideal method for skew-Hamiltonian matrices that was mentioned above. Section 3 contains similar results for the Hamiltonian eigenvalue problem, with a more extensive treatment of structure-preserving algorithms. In particular, we present an explicit version of the Hamiltonian QR algorithm, describe the method given in [38] via an embedding in skew-Hamiltonian matrices, and give an example of an iterative refinement algorithm. The last three sections include some computational aspects such as balancing and block algorithms, as well as numerical results.

This chapter is accompanied by HAPACK, a Fortran 77/MATLAB software library for solving skew-Hamiltonian and Hamiltonian eigenvalue problems, see [33, 151] and Section 5 in Appendix B.

Contributions in this chapter

The author contributes the following novel pieces to the (skew-) Hamiltonian puzzle:

- perturbation bounds and structured condition numbers for eigenvalues and invariant subspaces of skew-Hamiltonian matrices, see Sections 2.2 and 2.3;
- block algorithms for orthogonal symplectic decompositions, see Section 5.

Smaller bits and pieces are:

- a discussion on structured condition numbers for eigenvalues and invariant subspaces of Hamiltonian matrices and their relationship to unstructured ones, see Sections 3.2 and 3.3;
- an alternative algorithm for reordering Hamiltonian Schur decompositions based on symplectic QR decompositions, see Section 3.5;
- the derivation of the symplectic URV decomposition from the PVL decomposition, see Section 3.6;
- a simple description of the permutation stage in symplectic balancing algorithms, see Section 4.

Most of the presented material has already been published or submitted for publication, see [33, 34, 148, 150].

1 Preliminaries

An ubiquitous matrix in this chapter is the skew-symmetric matrix

$$J_{2n} = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}, \tag{4.3}$$

where I denotes the $n \times n$ identity matrix. In the following we will drop the subscript $2n$ whenever the dimension of the corresponding matrix 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 $S^TJS = SJS^T = J$ is called *symplectic*, and since

$$(S^{-1}HS)J = S^{-1}HJS^{-T} = S^{-1}J^TH^TS^{-T} = [(S^{-1}HS)J]^T,$$

we see that symplectic equivalence transformations preserve Hamiltonian structures. There are cases, however, where both H and $S^{-1}HS$ are Hamiltonian but S is not a symplectic matrix [101]. In a similar fashion the same can be shown for skew-Hamiltonian matrices.

1.1 Elementary Orthogonal Symplectic Matrices

From a numerical point of view it is desirable that a symplectic matrix $U \in \mathbb{R}^{2n\times 2n}$ that is 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

$$G_{j,n+j}(\theta) = \begin{bmatrix} I_{j-1} & & & & \\ & \cos\theta & & \sin\theta & \\ & & I_{n-1} & & \\ & -\sin\theta & & \cos\theta & \\ & & & & I_{n-j} \end{bmatrix}, \quad 1 \le j \le n,$$

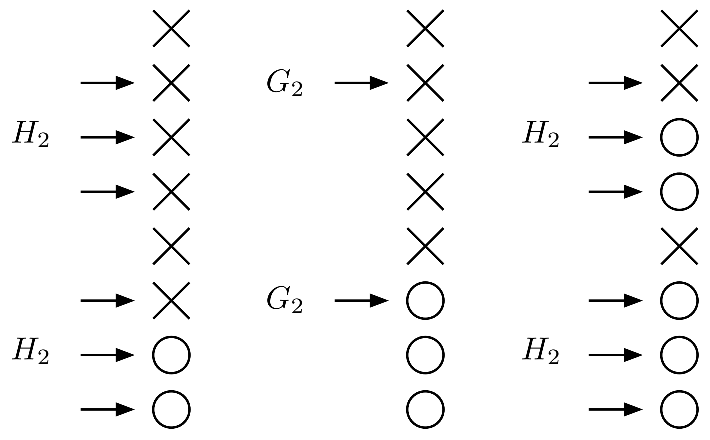


Figure 4.1. *The three steps of Algorithm 4.1 for $n = 4$ and $j = 2$.*

for some angle $\theta \in [-\pi/2, \pi/2)$ and the direct sum of two identical $n \times n$ Householder matrices

$$(H_j \oplus H_j)(v, \beta) = \begin{bmatrix} I_n - \beta vv^T & \\ & I_n - \beta vv^T \end{bmatrix},$$

where v is a vector of length n with its first $j-1$ elements equal to zero and β a scalar satisfying $\beta(\beta v^T v - 2) = 0$. Here, '$\oplus$' denotes the direct sum of matrices.

A simple combination of these transformations can be used to map an arbitrary vector $x \in \mathbb{R}^{2n}$ into the linear space

$$\mathcal{E}_j = \text{span}\{e_1, \dots, e_j, e_{n+1}, \dots, e_{n+j-1}\},$$

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 the following algorithm, where it should be noted that elements $1, \dots, j-1$ and $n+1, \dots, n+j-1$ of the vector x remain unaffected.

Algorithm 4.1.
Input: *A vector $x \in \mathbb{R}^{2n}$ and an index $j \leq n$.*
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.

The three steps of this algorithm are illustrated in Figure 4.1. Orthogonal symplectic matrices of the form

$$E_j(x) \equiv E_j(v, w, \beta, \gamma, \theta) := (H_j \oplus H_j)(v, \beta) \cdot G_{j,n+j}(\theta) \cdot (H_j \oplus H_j)(w, \gamma), \quad (4.4)$$

as computed by Algorithm 4.1 and with $1 \leq j \leq n$, will be called *elementary.*

Remark 4.2. *Let* $F = \left[\begin{smallmatrix} 0 & I_n \\ I_n & 0 \end{smallmatrix}\right]$, *then*

$$[F \cdot E_j(Fx) \cdot F]^T x \in \operatorname{span}\{e_1, \dots, e_{j-1}, e_{n+1}, \dots, e_{n+j}\}.$$

For the sake of brevity we set $E_{n+j}(x) := F \cdot E_j(Fx) \cdot F$, *whenever* $1 \leq j \leq n$.

The following lemma shows that every orthogonal symplectic matrix can be factorized into elementary matrices.

Lemma 4.3 ([263, 53]). *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.*

Proof. The first part is a consequence of the two relations $U^T J U = J$ and $U^T U = I$ which imply $JUJ^T = U^{-T} = U$. We prove the second part by induction over n. The elementary matrix $E_1(Ue_1) \equiv E_j(v, w, \beta, \gamma, \theta)$ maps the first column of U to αe_1 with $\alpha = \pm \|Ue_1\|_2 = \pm 1$. By a suitable choice of w and γ in the last step of Algorithm 4.1 we may assume w.l.o.g. that $\alpha = 1$. Thus,

$$U = E_j(Ue_1)\tilde{U}, \quad \tilde{U} = \left[\begin{array}{cc|cc} 1 & U_{12} & 0 & U_{14} \\ 0 & U_{22} & 0 & U_{24} \\ \hline 0 & -U_{14} & 1 & U_{12} \\ 0 & -U_{24} & 0 & U_{22} \end{array}\right],$$

where we have applied the first part of this lemma to the orthogonal symplectic matrix $\tilde{U}$. The orthogonality of $\tilde{U}$ implies that its first row has unit norm and thus $U_{12} = U_{14} = 0$. The proof of this lemma is completed by applying the induction hypothesis to the $2(n-1) \times 2(n-1)$ orthogonal symplectic matrix $\left[\begin{smallmatrix} U_{22} & U_{24} \\ -U_{24} & U_{22} \end{smallmatrix}\right]$. □

1.2 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 4.4 ([53]). *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*

$$R = \begin{bmatrix} R_{11} \\ R_{21} \end{bmatrix}, \quad R_{11} = \begin{bmatrix} \diagdown\!\!| \\ 0 \end{bmatrix}, \quad R_{21} = \begin{bmatrix} {}^{0}\!\ddots\!{}_{0}\!\diagdown\!\!| \\ 0 \end{bmatrix}, \tag{4.5}$$

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 (4.5) is called a *symplectic QR decomposition*, it is useful for computing and refining invariant subspaces of Hamiltonian matrices, see Sections 3.5 and 3.8 below. Other applications include the symplectic integration of Hamiltonian systems [162, 211]. The following algorithm proves the first part of Lemma 4.4 by construction.

Algorithm 4.5 (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 (4.5).*

```
$Q = I_{2m}$.
FOR $j = 1, \dots, n$
  Set $x = Ae_j$.
  Apply Algorithm 4.1 to compute $E_j(x)$.
  Update $A \leftarrow E_j(x)^T A$, $Q \leftarrow QE_j(x)$.
END FOR
```

This algorithm, implemented in the HAPACK 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 4.5 are as follows. Similarly as for the standard QR decomposition [112] 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 to 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 $\hat{Q} = \left[\begin{smallmatrix} \hat{Q}_1 & \hat{Q}_2 \\ -\hat{Q}_2 & \hat{Q}_1 \end{smallmatrix}\right]$. This implies, together with the following lemma, that $\hat{Q}$ is close to an orthogonal symplectic matrix.

Lemma 4.6. *Let $\hat{Q} = \left[\begin{smallmatrix} Q_1 & Q_2 \\ -Q_2 & Q_1 \end{smallmatrix}\right]$ 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*

$$\frac{\|\hat{Q}^T\hat{Q} - I\|_2}{\|\hat{Q}\|_2 + 1} \leq \|\hat{Q} - Q\|_2 \leq \|\hat{Q}^T\hat{Q} - I\|_2. \tag{4.6}$$

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 [38] or Section 3.6, 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 $R_{11} = U_1 D V_1^T$ 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 (4.6) is a well-known result, see e.g. [122, p.389]. □

2 The Skew-Hamiltonian Eigenvalue Problem

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

$$S^{-1}WS = \begin{bmatrix} W_{11} & 0 \\ 0 & W_{11}^T \end{bmatrix}. \tag{4.7}$$

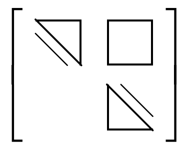
This decomposition – among others – will be described in the following section.

2.1 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 4.7 (PVL decomposition [243]). *Let $W \in \mathbb{R}^{2n\times 2n}$ be skew-Hamiltonian. Then there exists an orthogonal symplectic matrix U so that U^TWU has* Paige/Van Loan (PVL) *form, i.e.,*

$$U^TWU = \begin{bmatrix} W_{11} & W_{12} \\ 0 & W_{11}^T \end{bmatrix} =$$



$, \qquad (4.8)$

where $W_{11} \in \mathbb{R}^{n\times n}$ is an upper Hessenberg matrix.

The PVL decomposition (4.8) is a consequence of Algorithm 4.8 below. Let us illustrate its idea for $n = 4$. First, $E_2(We_1)$ is used to annihilate entries $3, \dots, 8$ 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:

$$W \leftarrow E_3(We_2)^T W E_3(We_2) = \left[\begin{array}{cccc|cccc} a & a & \hat{a} & \hat{a} & 0 & g & \hat{g} & \hat{g} \\ a & a & \hat{a} & \hat{a} & g & 0 & \hat{g} & \hat{g} \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & 0 & \hat{g} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & 0 \\ \hline 0 & 0 & 0 & 0 & a & a & 0 & 0 \\ 0 & 0 & \hat{0} & \hat{0} & a & a & \hat{a} & \hat{0} \\ 0 & \hat{0} & 0 & \hat{q} & \hat{a} & \hat{a} & \hat{a} & \hat{a} \\ 0 & \hat{0} & \hat{q} & 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \end{array}\right],$$

$$W \leftarrow E_3(We_2)^T W E_3(We_2) = \left[\begin{array}{cccc|cccc} a & a & a & \hat{a} & 0 & g & g & \hat{g} \\ a & a & a & \hat{a} & g & 0 & g & \hat{g} \\ 0 & a & a & \hat{a} & g & g & 0 & \hat{g} \\ 0 & 0 & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & 0 \\ \hline 0 & 0 & 0 & 0 & a & a & 0 & 0 \\ 0 & 0 & 0 & 0 & a & a & a & \hat{0} \\ 0 & 0 & 0 & \hat{0} & a & a & a & \hat{a} \\ 0 & 0 & \hat{0} & 0 & \hat{a} & \hat{a} & \hat{a} & \hat{a} \end{array}\right].$$

Algorithm 4.8.
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 $U^T WU$ having PVL form (4.8).*

```
$U \leftarrow I_{2n}$
FOR $j = 1, \dots, n-1$
   Set $x = We_j$.
   Apply Algorithm 4.1 to compute $E_{j+1}(x)$.
   Update $W \leftarrow E_{j+1}(x)^T W E_{j+1}(x)$, $U \leftarrow UE_{j+1}(x)$.
END FOR
```

This algorithm 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.

An immediate consequence of the PVL decomposition (4.8) 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 4.9. *The Sylvester equation*

$$W_{11}P - PW_{11}^T = -W_{12} \tag{4.9}$$

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 (4.9) is real and symmetric.*

Proof. The first part of this result can be found in [105, 97]. The second part is not explicitly stated there. 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 (4.9) into

$$\begin{bmatrix} J_1 & 0 \\ 0 & J_2 \end{bmatrix}\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} - \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}\begin{bmatrix} J_1^T & 0 \\ 0 & J_2^T \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} \\ -B_{12}^T & B_{22} \end{bmatrix}, \quad (4.10)$$

where $J_2 \in \mathbb{R}^{m\times m}$ is a single real Jordan block. The Sylvester equation

$$J_1 P_{12} - P_{12} J_2^T = B_{12}$$

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

$$J_2 P_{22} - P_{22} J_2^T = B_{22} \quad (4.11)$$

If J_2 is a scalar then every $P_{22} \in \mathbb{R}$ is a solution of (4.11). 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 (4.11) 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 (4.11) can be obtained by combining the above results with backward substitution. It is left to prove that any solution P_{22} of (4.11) 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., [125]) 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 Hankel matrix that is skew-symmetric 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 (4.10) 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}. $\square$

We now use this proposition to block-diagonalize a skew-Hamiltonian matrix in PVL form (4.8) assuming that W_{11} is nonderogatory. For this purpose let R be a solution of (4.9), then the symmetry of R implies that $\left[\begin{smallmatrix} I & R \\ 0 & I \end{smallmatrix}\right]$ is symplectic. Applying the corresponding symplectic equivalence transformation yields the transformed matrix

$$\begin{bmatrix} I & R \\ 0 & I \end{bmatrix}^{-1} \begin{bmatrix} W_{11} & W_{12} \\ 0 & W_{11}^T \end{bmatrix} \begin{bmatrix} I & R \\ 0 & I \end{bmatrix} = \begin{bmatrix} W_{11} & 0 \\ 0 & W_{11}^T \end{bmatrix}. \tag{4.12}$$

Note that there is a lot of freedom in the choice of R as equation (4.9) 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 2.3.

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 [97] have shown that any skew-Hamiltonian matrix can be reduced to block diagonal form (4.12) using symplectic equivalence transformations. The proof, however, is much more involved than the simple derivation given above.

Another way to go from a skew-Hamiltonian matrix W in PVL form (4.8) 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:

$$\tilde{U}^T W \tilde{U} = \begin{bmatrix} T & \tilde{G} \\ 0 & T^T \end{bmatrix}, \tag{4.13}$$

where $\tilde{G} = Q_1^T W_{12} Q_1$ is skew-symmetric.

2.2 Structured Condition Numbers for Eigenvalues

In this and the next 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 4.10. *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.

We now turn to the perturbation theory for an eigenvalue λ of a matrix W under a perturbation E, where both W and E are skew-Hamiltonian matrices. As

λ is necessarily a multiple eigenvalue we cannot apply the eigenvalue perturbation expansion given in Theorem 1.5 to λ alone. 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 corresponding 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 $A_{11} = \operatorname{diag}(\lambda, \lambda)$. 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

$$\begin{aligned} |\hat{\lambda} - \lambda| &= \|(X^TJX)^{-1}X^TJEX\|_2 + \mathcal{O}(\|E\|^2) \\ &\leq \|(X^TJX)^{-1}\|_2 + \mathcal{O}(\|E\|^2), \end{aligned} \tag{4.14}$$

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 4.11. *Let $W, E \in \mathbb{R}^{2n\times 2n}$ be skew-Hamiltonian matrices, where $\|E\|_2 > 0$ is sufficiently small. Assume that λ is an eigenvalue of W with multiplicity two. Then there exists an eigenvalue $\hat{\lambda}$ of $W + E$ so that*

$$|\hat{\lambda} - \lambda| \leq \|P\|_2 \cdot \|E\|_2 + O(\|E\|^2), \tag{4.15}$$

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 (4.14) 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 eigenvalue condition number for an eigenvalue $\lambda \in \mathbb{R}$ of a skew-Hamiltonian matrix* satisfies

$$c_W(\lambda) := \lim_{\varepsilon\to 0} \sup_{\substack{\|E\|_2\leq\varepsilon \\ E \text{ skew-Hamiltonian}}} \frac{|\hat{\lambda} - \lambda|}{\varepsilon} = \|P\|_2.$$

Likewise for complex λ, we can use perturbations of the form $E = \varepsilon J_{2n}^T \bar{X} J_2 X^H$. Note that E satisfies $(EJ_{2n})^T = -(EJ_{2n})$, i.e., E may be regarded as a *complex* skew-Hamiltonian matrix. It is an open problem whether one can construct a real skew-Hamiltonian perturbation to show $c_W(\lambda) = \|P\|_2$ for complex eigenvalues.

There is a simple expression for computing $\|P\|_2$ from the eigenvectors belonging to λ.

Lemma 4.12. *Under the assumptions of Corollary 4.11,*

$$\|P\|_2 = \frac{1}{|x_1^T J x_2|}\sqrt{\|x_1\|_2^2 \cdot \|x_2\|_2^2 - |x_1^H x_2|^2},$$

where $x_1, x_2 \in \mathbb{C}^{2n}$ are two linearly independent eigenvectors belonging to λ.

Proof. The spectral projector can be expressed as

$$P = \frac{1}{x_1^T J x_2} \left(x_1 (J\bar{x}_2)^H - x_2 (J\bar{x}_1)^H \right).$$

Now, let $x_2' = x_2 - (x_1^H x_2)/(x_1^H x_1) \cdot x_1$, then $x_1^H x_2' = 0$ and we get

$$\begin{aligned} |x_1^T J x_2| \cdot \|P\|_2 &= \left\| x_1 (J\bar{x}_2)^H - v_2 (J\bar{x}_1)^H \right\|_2 \\ &= \left\| v_1 (J\bar{x}_2')^H - x_2' (J\bar{x}_1)^H \right\|_2 \\ &= \|x_1\|_2 \cdot \|x_2'\|_2 = \|x_1\|_2 \sqrt{\|x_2\|^2 - |x_1^H x_2|^2 / \|x_1\|_2^2}, \end{aligned}$$

which concludes the proof. □

Structured backward errors and condition numbers for eigenvalues of skew-Hamiltonian matrices with additional structures can be found in [235].

2.3 Structured Condition Numbers for Invariant Subspaces

The invariant subspaces of a skew-Hamiltonian matrix W that are usually of interest in applications are those which are isotropic.

Definition 4.13. *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.

Obviously, any eigenvector of W spans an isotropic invariant subspace but also the first $k \leq n$ columns of the matrix $\tilde{U}$ in a skew-Hamiltonian Schur decomposition (4.13) 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. Necessarily, $\mathcal{X}$ is not simple, which makes the application of perturbation expansions, such as the one given in Theorem 1.9, impossible. Instead, we develop a global, structured perturbation analysis for isotropic invariant subspaces, much in the spirit of the approach by Stewart [217, 219] described in Section 2.4, Chapter 1.

This will be done as follows. First, a connection between finding a nearby isotropic subspace and solving a quadratic matrix equation will be explained. The solution of this equation is complicated by an artificial singularity, its lengthy derivation compromises the bulge of the work. Based on this solution we present the central global perturbation result, Theorem 4.24, which gives an upper bound for the sensitivity of isotropic invariant subspaces. This will lead us to define a corresponding condition number, and we will briefly touch the question how this quantity may be computed.

Isotropic invariant subspaces are related to some kind of block skew-Hamiltonian Schur form, just as invariant subspaces of general matrices are related to block Schur forms.

Lemma 4.14. *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*

$$U^T W U = \begin{array}{c|cccc} & k & n-k & k & n-k \\ k & A_{11} & A_{12} & G_{11} & G_{12} \\ n-k & 0 & A_{22} & -G_{12}^T & G_{22} \\ k & 0 & 0 & A_{11}^T & 0 \\ n-k & 0 & H_{22} & A_{12}^T & A_{22}^T \end{array}. \tag{4.16}$$

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 (4.16). The other direction is straightforward. $\square$

As the eigenvalue properties of $A_{11} = X^T W X$ and

$$\begin{bmatrix} A_{22} & G_{22} \\ H_{22} & A_{22}^T \end{bmatrix} = [Z, J^T Z]^T W [Z, J^T Z]$$

do not depend on the choice of basis, the following definition can be used to adapt the notion of simple invariant subspaces to skew-Hamiltonian matrices.

Definition 4.15. *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 (4.16). Then* $\mathcal{X}$ *is called* semi-simple *if* $\lambda(A_{11}) \cap \lambda\left(\left[\begin{smallmatrix} A_{22} & G_{22} \\ H_{22} & A_{22}^T \end{smallmatrix}\right]\right) = \emptyset$ *and* A_{11} *is nonderogatory, i.e., each eigenvalue of* A_{11} *has geometric multiplicity one.*

Isotropic invariant subspaces and a quadratic matrix equation

In the following we assume that $\mathcal{X}$ is a semi-simple isotropic invariant subspace of a skew-Hamiltonian matrix W. Given a skew-Hamiltonian perturbation E of small norm we now relate the question whether $W+E$ has an isotropic invariant subspace $\hat{\mathcal{X}}$ close to $\mathcal{X}$ to the solution of a quadratic matrix equation. For this purpose, let the columns of X form an orthonormal basis for $\mathcal{X}$. Apply Lemma 4.14 to construct a matrix $Y = [Z, J^T Z, J^T X]$ so that $\tilde{U} = [X, Y]$ is an orthogonal matrix. Note that $\tilde{U}^T (W + E)\tilde{U}$ is a permuted skew-Hamiltonian matrix and can be partitioned as

$$\tilde{U}^T (W + E)\tilde{U} = \begin{array}{c|ccc} & k & 2(n-k) & k \\ k & W_{11} & W_{23}^T J_{n-k}^T & W_{13} \\ 2(n-k) & W_{21} & W_{22} & W_{23} \\ k & W_{31} & W_{21}^T J_{n-k} & W_{11}^T \end{array}, \tag{4.17}$$

where W_{13} and W_{31} are skew-symmetric matrices, and W_{22} is skew-Hamiltonian. For $E = 0$, the matrices W_{21} and W_{31} are zero and the other blocks in (4.17) correspond to the block representation (4.16) as follows:

$$W_{11} = A_{11},\ W_{13} = G_{11},\ W_{22} = \begin{bmatrix} A_{22} & G_{22} \\ H_{22} & A_{22}^T \end{bmatrix},\ W_{23} = \begin{bmatrix} -G_{12}^T \\ A_{12}^T \end{bmatrix}.$$

Now, let

$$\hat{X} = \left(X + Y \begin{bmatrix} P \\ Q \end{bmatrix} \right) (I + P^T P + Q^T Q)^{-1/2}, \tag{4.18}$$

$$\hat{Y} = (Y - X \begin{bmatrix} P^T & Q^T \end{bmatrix}) \left(I + \begin{bmatrix} P \\ Q \end{bmatrix} \begin{bmatrix} P^T & Q^T \end{bmatrix} \right)^{-1/2}, \tag{4.19}$$

where $P \in \mathbb{R}^{2(n-k)\times k}$ and $Q \in \mathbb{R}^{k\times k}$ are matrices to be determined so that $\hat{\mathcal{X}} = \operatorname{span}(\hat{X})$ is an isotropic invariant subspace of $W + E$. This is equivalent to the conditions $Q^T - Q = P^T J P$ and $\hat{Y}^T (W + E)\hat{X} = 0$. In terms of (4.17), the latter can be written as

$$\begin{bmatrix} P \\ Q \end{bmatrix} W_{11} - \begin{bmatrix} W_{22} & W_{23} \\ W_{21}^T J & W_{11}^T \end{bmatrix} \begin{bmatrix} P \\ Q \end{bmatrix} + \begin{bmatrix} P \\ Q \end{bmatrix} \begin{bmatrix} JW_{23} \\ W_{13}^T \end{bmatrix}^T \begin{bmatrix} P \\ Q \end{bmatrix} = \begin{bmatrix} W_{21} \\ W_{31} \end{bmatrix}. \tag{4.20}$$

Once we have solved (4.20) the sines of the canonical angles between $\mathcal{X}$ and $\hat{\mathcal{X}}$ are the singular values of

$$Y^T \hat{X} = \begin{bmatrix} P \\ Q \end{bmatrix} (I + P^T P + Q^T Q)^{-1/2},$$

see Section 2.3 in Chapter 1. We will see that (4.20) may admit infinitely many solutions satisfying $Q^T - Q = P^T J P$. In the interest of a small distance between $\mathcal{X}$ and $\hat{\mathcal{X}}$ a solution of small norm should be preferred.

Solving (4.20) is complicated by two facts. First, we have to guarantee that the solution satisfies $Q^T - Q = P^T J P$ and second, the linear part of (4.20) is close to a singular linear matrix equation if $W_{21} \approx 0$. Unfortunately, it is not easy to see from the present formulation of (4.20) that this singularity is, due to the special structure of the nonlinearities and the right hand side, artificial. Both issues can be more easily addressed after a reformulation of (4.20).

Skew-symmetrizing the bottom part

Let

$$R = Q + P^T \tilde{J} P, \quad \tilde{J} = \begin{bmatrix} 0 & I_{n-k} \\ 0 & 0 \end{bmatrix}, \tag{4.21}$$

then R is symmetric if and only if $Q^T - Q = P^T J P$. The following lemma reveals a particular useful nonlinear matrix equation satisfied by (P, R).

Lemma 4.16. *Let $R = Q + P^T \tilde{J} P$ be symmetric. Then the matrix pair (P, Q) is a solution of (4.20) if and only if (P, R) is a solution of*

$$\begin{bmatrix} P \\ R \end{bmatrix} W_{11} - \begin{bmatrix} W_{22} & W_{23} \\ W_{21}^T J & W_{11}^T \end{bmatrix} \begin{bmatrix} P \\ R \end{bmatrix} + \begin{bmatrix} \Phi_1(P,R) \\ \Phi_2(P,R) - P^T J W_{21} \end{bmatrix} = \begin{bmatrix} W_{21} \\ W_{31} \end{bmatrix}, \tag{4.22}$$

where

$$\begin{aligned} \Phi_1(P,R) &= W_{23}(P^T \tilde{J} P) + P(JW_{23})^T P + PW_{13}(R - P^T \tilde{J} P), \\ \Phi_2(P,R) &= (R - P^T \tilde{J}^T P) W_{23}^T J^T P - P^T J W_{23} (R - P^T \tilde{J}^T P)^T \\ &\quad + (R - P^T \tilde{J} P)^T W_{13} (R - P^T \tilde{J} P) - P^T J W_{22} P. \end{aligned}$$

Proof. Adding the top part of (4.20) premultiplied by $P^T J$,

$$P^T JW_{21} = P^T JPW_{11} - P^T JW_{22}P - P^T JW_{23}Q + P^T JP(JW_{23})^T P + P^T JPW_{13}Q,$$

to the bottom part of (4.20) yields the transformed equation (4.22) after some basic algebraic manipulations. □

The reformulated equation (4.22) has the advantage that the nonlinearity $\Phi_2(P, R)$, the right hand side term W_{31}, as well as the coupling term $-W_{21}^T JP - P^T JW_{21}$ are skew-symmetric and thus in the range of the operator $R \mapsto RW_{11} - W_{11}^T R$ provided that W_{11} is nonderogatory. This indicates that the singularity caused by this operator is indeed artificial.

Solving the decoupled linearized equation

Linearizing (4.22) around $(P, R) = (0, 0)$ yields

$$\tilde{\mathbf{T}}(P, R) = \begin{bmatrix} W_{21} \\ W_{31} \end{bmatrix}, \tag{4.23}$$

where the operator $\tilde{\mathbf{T}} : \mathbb{R}^{2(n-k)\times k} \times \mathbb{R}^{k\times k} \to \mathbb{R}^{2(n-k)\times k} \times \mathbb{R}^{k\times k}$ is given by

$$\tilde{\mathbf{T}} : (P, R) \mapsto \begin{bmatrix} P \\ R \end{bmatrix} W_{11} - \begin{bmatrix} W_{22} & W_{23} \\ W_{21}^T J & W_{11}^T \end{bmatrix} \begin{bmatrix} P \\ R \end{bmatrix} - \begin{bmatrix} 0 \\ P^T JW_{21} \end{bmatrix}. \tag{4.24}$$

Note that we sometimes identify $(X, Y) \sim \begin{bmatrix} X \\ Y \end{bmatrix}$ for notational convenience. It is assumed that the perturbation E is sufficiently small, implying that W_{21} is small. Hence, $W_{21}^T JP$ and $P^T JW_{21}$ can be regarded as weak coupling terms. Let us neglect these terms and consider the operator

$$\mathbf{T} : (P, R) \mapsto \begin{bmatrix} P \\ R \end{bmatrix} W_{11} - \begin{bmatrix} W_{22} & W_{23} \\ 0 & W_{11}^T \end{bmatrix} \begin{bmatrix} P \\ R \end{bmatrix}, \tag{4.25}$$

which allows an easy characterization. In the following lemma, $\mathrm{Sym}(k)$ denotes the set of all symmetric $k \times k$ matrices, and $\mathrm{Skew}(k)$ the set of all skew-symmetric $k \times k$ matrices.

Lemma 4.17. *Consider the operator* $\mathbf{T}$ *defined by (4.25) with domain and codomain restricted to* $\mathrm{dom}\,\mathbf{T} = \mathbb{R}^{2(n-k)\times k} \times \mathrm{Sym}(k)$ *and* $\mathrm{codom}\,\mathbf{T} = \mathbb{R}^{2(n-k)\times k} \times \mathrm{Skew}(k)$, *respectively. Then* $\mathbf{T}$ *is onto if and only if* W_{11} *is nonderogatory and* $\lambda(W_{11}) \cap \lambda(W_{22}) = \emptyset$.

Proof. If W_{11} is nonderogatory and $\lambda(W_{11}) \cap \lambda(W_{22}) = \emptyset$, then we can apply Proposition 4.9 and Lemma 1.3 combined with backward substitution to show that $\mathbf{T}$ is onto. For the other direction, assume that $\mathbf{T}$ is onto. The nonderogatority of W_{11} is a consequence of Proposition 4.9; it remains to show that $\lambda(W_{11}) \cap \lambda(W_{22}) = \emptyset$. By continuity, we may assume w.l.o.g. that there exists a nonsingular matrix X so that $\Lambda = X^{-1}W_{11}X$ is diagonal with diagonal elements $\lambda_1, \ldots, \lambda_k \in \mathbb{C}$. Then there exists a matrix $\tilde{R}_0 \in \mathbb{C}^{k\times k}$ so that every solution of the transformed equation $\tilde{R}\Lambda - \bar{\Lambda}\tilde{R} = X^{-1}W_{31}X$ has the form

$$\tilde{R} = \tilde{R}_0 + \sum_{i=1}^{k} \alpha_i e_i e_i^T, \quad \alpha_1, \ldots, \alpha_k \in \mathbb{C}.$$

Inserting this representation into the equation $\tilde{P}\Lambda - W_{22}\tilde{P} - W_{23}X^{-T}\tilde{R} = W_{13}X$ leads to the k separate equations

$$\begin{bmatrix} \lambda_i I - W_{22} & b_i \end{bmatrix} \begin{bmatrix} \tilde{p}_i \\ \alpha_i \end{bmatrix} = (W_{13}X + W_{23}\tilde{R}_0)e_i, \tag{4.26}$$

where $\tilde{p}_i$ and b_i denote the ith columns of $\tilde{P}$ and $W_{23}X^{-T}$, respectively. Equation (4.26) has a solution for any $W_{13} \in \mathbb{R}^{2(n-k)\times k}$ if and only if $[\lambda_i I - W_{22}, b_i]$ has full rank $2(n-k)$. This implies $\lambda_i \notin \lambda(W_{22})$, since otherwise

$$\operatorname{rank}\left(\begin{bmatrix} \lambda_i I - W_{22} & b_i \end{bmatrix}\right) \leq \operatorname{rank}(\lambda_i I - W_{22}) + 1 \leq 2(n-k) - 1,$$

where we used the fact that the geometric multiplicity of each eigenvalue of the skew-Hamiltonian matrix W_{22} is at least two. Thus $\lambda(W_{11}) \cap \lambda(W_{22}) = \emptyset$, which concludes the proof. □

For the remainder of this section only the restricted operator $\mathbf{T}$ will be considered and it will be assumed that this operator is onto. Note that for $E = 0$ the latter is equivalent to the assumption that $\mathcal{X}$ is semi-simple, see Definition 4.15. The dimensions of the matrix spaces Skew(k) and Sym(k) differ by k. More precisely, it can be shown that the set of solutions corresponding to a particular right hand side in the codomain of $\mathbf{T}$ form an affine subspace of dimension k [97]. In view of an earlier remark one should pick a solution that has minimal norm. The following lemma shows that this solution is uniquely determined if the Frobenius norm is used.

Lemma 4.18. *Let* $\mathbf{T}$ *be defined as in (4.25) and let* $(W_{21}, W_{31}) \in \operatorname{codom} \mathbf{T}$. *Then there exists one and only one matrix pair* $(P_\star, R_\star) \in \operatorname{dom} \mathbf{T}$ *satisfying*

$$\|(P_\star, R_\star)\|_F = \min_{(P,R)\in \operatorname{dom} \mathbf{T}} \left\{ \|(P,R)\|_F \mid \mathbf{T}(P,R) = \begin{bmatrix} W_{21} \\ W_{31} \end{bmatrix} \right\}. \tag{4.27}$$

Proof. Using the second part of Proposition 4.9 the constraint $(P,R) \in \operatorname{dom} \mathbf{T}$ in (4.27) can be dropped. Let us define

$$K_{\mathbf{T}} := W_{11}^T \otimes I - I \otimes \begin{bmatrix} W_{22} & W_{23} \\ 0 & W_{11}^T \end{bmatrix}.$$

Using the vec operator, the minimization problem (4.27) can be written in the form

$$\min_{x \in \mathbb{R}^{(2n-k)\times k}} \{ \|x\|_2 \ : \ K_{\mathbf{T}} \cdot x = w \}, \tag{4.28}$$

where $w = \operatorname{vec}(\left[\begin{smallmatrix} W_{21} \\ W_{31} \end{smallmatrix}\right])$. Well-known results about linear least-squares problems show that (4.28) has a unique minimum given by $K_{\mathbf{T}}^\dagger \cdot w$, where $K_{\mathbf{T}}^\dagger$ denotes the pseudo-inverse of $K_{\mathbf{T}}$ [112, Sec. 5.5.4]. □

This lemma allows us to define an operator

$$\mathbf{T}^\dagger : \operatorname{codom} \mathbf{T} \to \operatorname{dom} \mathbf{T}$$

which maps a matrix pair (W_{21}, W_{31}) to the solution of (4.27). A sensible choice of norm for $\mathbf{T}^\dagger$ is the one induced by the Frobenius norm,

$$\|\mathbf{T}^\dagger\| := \sup_{\substack{\|(W_{21},W_{31})\|_F=1 \\ (W_{21},W_{31})\in \operatorname{codom} \mathbf{T}}} \|\mathbf{T}^\dagger(W_{21}, W_{31})\|_F. \tag{4.29}$$

Solving the coupled linearized equation

The key to solving the coupled equation (4.23) is to note that $\tilde{\mathbf{T}}$ can be decomposed into $\mathbf{T} - \triangle\mathbf{T}$, where $\triangle\mathbf{T} : \operatorname{dom} \mathbf{T} \to \operatorname{codom} \mathbf{T}$ is defined by

$$\triangle\mathbf{T} : (P, R) \mapsto \begin{bmatrix} 0 \\ P^T J W_{21} + W_{21}^T J P \end{bmatrix}. \tag{4.30}$$

This implies that the composed operator $\mathbf{T}^\dagger \circ \triangle\mathbf{T} : \operatorname{dom} \mathbf{T} \to \operatorname{dom} \mathbf{T}$ is well-defined, its norm is again the one induced by the Frobenius norm.

Lemma 4.19. *If* $\mathbf{T}$ *is onto and* $\delta := \|\mathbf{T}^\dagger \circ \triangle\mathbf{T}\| < 1$ *then*

$$\mathbf{X}(W_{21}, W_{31}) := \sum_{i=0}^{\infty} (\mathbf{T}^\dagger \circ \triangle\mathbf{T})^i \circ \mathbf{T}^\dagger(W_{21}, W_{31}) \tag{4.31}$$

is a solution of (4.23).

Proof. If $\delta < 1$ then

$$\Big\| \sum_{i=0}^{\infty} (\mathbf{T}^\dagger \circ \triangle\mathbf{T})^i \circ \mathbf{T}^\dagger \Big\| \le \sum_{i=0}^{\infty} \delta^i \|\mathbf{T}^\dagger\| = \frac{\|\mathbf{T}^\dagger\|}{1-\delta}, \tag{4.32}$$

implying that the infinite sum in (4.31) converges absolutely. Moreover, pre-multiplying (4.31) with $\mathbf{T} - \triangle\mathbf{T}$ shows that $\mathbf{X}(W_{21}, W_{31})$ solves (4.23). □

Inequality (4.32) yields the bound

$$\|\mathbf{X}(W_{21}, W_{31})\|_F \le \frac{\|\mathbf{T}^\dagger\|}{1-\delta} \cdot \|(W_{21}, W_{31})\|_F. \tag{4.33}$$

An upper bound for the quantity δ is clearly given by $2\|\mathbf{T}^\dagger\| \cdot \|W_{21}\|_F$. It should be stressed that $\mathbf{X} : \operatorname{codom} \mathbf{T} \to \operatorname{dom} \mathbf{T}$ may not give the solution of smallest norm. However, if $\|\triangle\mathbf{T}\|$ is sufficiently small it can be expected to be rather close to it.

Lemma 4.20. *Under the assumptions of Lemma 4.19 let* $\tilde{\mathbf{T}}^\dagger : \operatorname{codom} \mathbf{T} \to \operatorname{dom} \mathbf{T}$ *denote the operator that maps a pair* (W_{21}, W_{31}) *to the minimal norm solution of the coupled equation (4.23). Then*

$$\lim_{\triangle\mathbf{T}\to 0} \mathbf{X} = \lim_{\triangle\mathbf{T}\to 0} \tilde{\mathbf{T}}^\dagger = \mathbf{T}^+. \tag{4.34}$$

Proof. Lemma 4.19 shows that the coupled equation (4.23) has, for a given right hand side in codom $\mathbf{T}$, a nonempty set of solutions. This set is, according to Proposition 4.9, a subset of dom $\mathbf{T}$. The solution of minimal norm is uniquely defined, for reasons similar to those that have been used in the proof of Lemma 4.18. Hence, the operator $\tilde{\mathbf{T}}^\dagger$ is well-defined. By checking the four Penrose conditions it can be shown that $\mathbf{X} = (\mathbf{T} - \triangle\mathbf{T} \circ (\mathbf{T}^\dagger \circ \mathbf{T}))^\dagger$. Equalities (4.34) follow from the fact that the ranges of $(\mathbf{T} - \triangle\mathbf{T} \circ (\mathbf{T}^\dagger \circ \mathbf{T}))$, $\tilde{\mathbf{T}}$ and $\mathbf{T}$ have equal dimensions [226, Sec. III.3]. □

We remark that Lemma 4.19 and Lemma 4.20 are not restricted to perturbations of the form (4.30). In fact, they hold for any $\triangle\mathbf{T} : \operatorname{dom}\mathbf{T} \to \operatorname{codom}\mathbf{T}$ satisfying $\|\mathbf{T}^\dagger \circ \triangle\mathbf{T}\| < 1$.

Solving the quadratic matrix equation

Using the terminology developed above, we can rewrite the nonlinear equation (4.22) in the more convenient form

$$\tilde{\mathbf{T}}(P,R) + \Phi(P,R) = \begin{bmatrix} W_{21} \\ W_{31} \end{bmatrix}, \tag{4.35}$$

where $\Phi(P,R) = [\Phi_1(P,R)^T, \Phi_2(P,R)^T]^T$.

Theorem 4.21. *Let the matrices W_{ij} be defined as in (4.17) and assume that the operator $\mathbf{T}$ defined by (4.25) is onto in the sense of Lemma 4.17. Assume that $\delta = 2\|\mathbf{T}^\dagger\| \cdot \|W_{21}\|_F < 1$, where $\|\mathbf{T}^\dagger\|$ is defined by (4.29). Set*

$$\gamma = \|(W_{21}, W_{31})\|_F, \quad \eta = \left\| \begin{bmatrix} W_{23}^T J^T & W_{13} \\ W_{22} & W_{23} \end{bmatrix} \right\|_F, \quad \kappa = \frac{\|\mathbf{T}^\dagger\|}{1-\delta}.$$

Then if

$$8\gamma\kappa < 1, \qquad 20\gamma\eta\kappa^2 < 1,$$

there exists a solution (P,R) of (4.35) satisfying

$$\|(P,R)\|_F \le 2\gamma\kappa. \tag{4.36}$$

Proof. We adapt the technique used by Stewart [219, Sec. 3] and solve (4.35) by constructing an iteration. First, some facts about the nonlinearities are required:

$$\begin{aligned} &\|\Phi_1(P,R)\|_F \le \|W_{13}\|_F(\|P\|_F\|R\|_F + \|P\|_F^3) + 2\|W_{23}\|_F\|P\|_F^2, \\ &\|\Phi_2(P,R)\|_F \le \eta\|(P,R)\|_F^2 + \|W_{13}\|_F(2\|P\|_F^2\|R\|_F + \|P\|_F^4) + 2\|W_{23}\|_F\|P\|_F^3, \\ \Rightarrow\ &\|\Phi(P,R)\|_F \le (1+\sqrt{3})\eta\|(P,R)\|_F^2 + (\sqrt{2}+\sqrt{3})\eta\|(P,R)\|_F^3 + \eta\|(P,R)\|_F^4. \end{aligned}$$

Using a rough estimate, we have $\|\Phi(P,R)\|_F \le 4\eta\|(P,R)\|_F^2$ for $\|(P,R)\|_F \le 1/4$. Similarly, it can be shown that

$$\begin{aligned} \|\Phi(\hat{P},\hat{R}) - \Phi(P,R)\|_F \le\ & [2(1+\sqrt{3})\eta \max\{\|(\hat{P},\hat{R})\|_F, \|(P,R)\|_F\} \\ & +4(\sqrt{2}+\sqrt{3})\eta \max\{\|(\hat{P},\hat{R})\|_F, \|(P,R)\|_F\}^2 \\ & +8\eta \max\{\|(\hat{P},\hat{R})\|_F, \|(P,R)\|_F\}^3] \cdot \|(\hat{P}-P, \hat{R}-R)\|_F \\ \le\ & 10\eta \max\{\|(\hat{P},\hat{R})\|_F, \|(P,R)\|_F\} \cdot \|(\hat{P}-P, \hat{R}-R)\|_F, \end{aligned}$$

where the latter inequality holds for $\max\{\|(P,R)\|_F, \|(\hat{P},\hat{R})\|_F\} \le 1/4$. Next, we define a sequence by $(P_0, R_0) = (0,0)$ and

$$(P_{k+1}, R_{k+1}) = \mathbf{X}(W_{21}, W_{31}) + \mathbf{X} \circ \Phi(P_k, R_k).$$

Note that this iteration is well-defined as $\Phi : \operatorname{dom} \mathbf{T} \to \operatorname{codom} \mathbf{T}$. We show by induction that the iterates stay bounded. Under the assumption $\|(P_k, R_k)\| < 2\gamma\kappa \leq 1/4$ it follows that

$$\|(P_{k+1}, R_{k+1})\|_F \leq \kappa(\gamma + 4\eta\|(P_k, R_k)\|_F^2) < 2\gamma\kappa.$$

The operator $\mathbf{X} \circ \Phi$ is a contraction on $\mathcal{D} = \{(P, R) \; : \; \|(P, R)\|_F < 2\gamma\kappa\}$ since

$$\|\mathbf{X} \circ \Phi(\hat{P}, \hat{R}) - \mathbf{X} \circ \Phi(P, R)\|_F \leq 20\gamma\eta\kappa^2\|(\hat{P} - P, \hat{R} - R)\|_F < \|(\hat{P} - P, \hat{R} - R)\|_F$$

for all $(P, R) \in \mathcal{D}$ and $(\hat{P}, \hat{R}) \in \mathcal{D}$. Thus, the contraction mapping theorem [183] shows that the sequence (P_k, R_k) converges to a fixed point, which solves (4.35). □

Corollary 4.22. *Under the assumptions of Theorem 4.21 there exists a solution (P, Q) of the quadratic matrix equation (4.20) satisfying $Q^T - Q = P^T JP$ and*

$$\|(P, Q)\|_F \leq 2\gamma\kappa + 4\gamma^2\kappa^2 < 2.5\gamma\kappa.$$

Proof. The result is a direct consequence of the relationship $Q = R - P^T\tilde{J}P$. □

Perturbation bounds and a condition number

From the discussion in the beginning of this section it follows that Corollary 4.22 yields the existence of an isotropic invariant subspace $\hat{\mathcal{X}}$ of $W + E$ close to $\mathcal{X}$, which is an isotropic invariant subspace of the unperturbed matrix W.

Corollary 4.23. *Under the assumptions of Theorem 4.21 there is an isotropic invariant subspace $\hat{\mathcal{X}}$ of the skew-Hamiltonian matrix $W + E$ so that*

$$\|\tan\Theta(\mathcal{X}, \hat{\mathcal{X}})\|_F \leq 2\gamma\kappa + 4\gamma^2\kappa^2 < 2.5\gamma\kappa. \tag{4.37}$$

Proof. Inequality (4.37) follows from Corollary 4.22 using the fact that $\tan\theta_i(\mathcal{X}, \hat{\mathcal{X}})$, $i = 1, \ldots, k$, are the singular values of the matrix $[P^T, Q^T]^T$. □

The catch of this corollary is that it works with quantities that are usually not known. For example, the operator $\mathbf{T}$, used to define κ, explicitly depends on the matrix $W + E$. However, often not the perturbation E itself but only an upper bound on its norm is given. For this reason, given a partitioning (4.16) let us use the unperturbed data to define an operator $\mathbf{T}_W : \operatorname{dom} \mathbf{T} \to \operatorname{codom} \mathbf{T}$ as follows:

$$\mathbf{T}_W : (P, Q) \mapsto \begin{bmatrix} P \\ Q \end{bmatrix} A_{11} - \begin{bmatrix} A_{22} & G_{22} & -G_{12}^T \\ H_{22} & A_{22}^T & A_{12}^T \\ 0 & 0 & A_{11}^T \end{bmatrix} \begin{bmatrix} P \\ Q \end{bmatrix} \tag{4.38}$$

The operator $\mathbf{T}_W^\dagger$ and its norm are defined in the same way as $\mathbf{T}^\dagger$ and $\|\mathbf{T}^\dagger\|$.

Theorem 4.24. *Let $U = [X, Z, J^T X, J^T Z]$ be orthogonal symplectic and suppose that $\mathcal{X} = \operatorname{span} X$ is a semi-simple isotropic invariant subspace of the skew-Hamiltonian matrix W so that*

$$U^T W U = \begin{bmatrix} A_{11} & A_{12} & G_{11} & G_{12} \\ 0 & A_{22} & -G_{12}^T & G_{22} \\ 0 & 0 & A_{11}^T & 0 \\ 0 & H_{22} & A_{12}^T & A_{22}^T \end{bmatrix}. \tag{4.39}$$

Given a skew-Hamiltonian perturbation E, let

$$U^T E U = \begin{bmatrix} E_{11} & E_{12} & E_{13} & E_{14} \\ E_{21} & E_{22} & -E_{14}^T & E_{24} \\ E_{31} & E_{32} & E_{11}^T & E_{21}^T \\ -E_{32}^T & E_{42} & E_{12}^T & E_{22}^T \end{bmatrix}.$$

Assume that $\hat{\delta} = \sqrt{3}\|\mathbf{T}_W^\dagger\| \cdot \|E\|_F < 1$, where $\mathbf{T}_W^\dagger$ is defined by (4.38). Set

$$\hat{\gamma} = \left\| \begin{bmatrix} E_{21} \\ E_{31} \\ E_{32}^T \end{bmatrix} \right\|_F, \quad \hat{\eta} = \left\| \begin{bmatrix} A_{12} & G_{11} & G_{12} \\ A_{22} & -G_{12}^T & G_{22} \\ H_{22} & A_{12}^T & A_{22}^T \end{bmatrix} \right\|_F + \left\| \begin{bmatrix} E_{12} & E_{13} & E_{14} \\ E_{22} & -E_{14}^T & E_{24} \\ E_{42} & E_{12}^T & E_{22}^T \end{bmatrix} \right\|_F$$

and $\hat{\kappa} = \|\mathbf{T}_W^\dagger\|/(1-\hat{\delta})$. Then if

$$8\hat{\gamma}\hat{\kappa} < 1, \qquad 20\hat{\gamma}\hat{\eta}\hat{\kappa}^2 < 1,$$

there are matrices P and Q satisfying

$$\|(P, Q)\|_F \leq 2\hat{\gamma}\hat{\kappa} + 4\hat{\gamma}^2\hat{\kappa}^2 < 2.5\hat{\gamma}\hat{\kappa}$$

so that the columns of

$$\hat{X} = \left(X + [Z, J^T Z, J^T X] \begin{bmatrix} P \\ Q \end{bmatrix} \right) (I + P^T P + Q^T Q)^{-1/2}$$

form an orthonormal basis for an isotropic invariant subspace of $\hat{W} = W + E$.

Proof. First, note that the semi-simplicity of $\mathcal{X}$ implies that $\mathbf{T}_W$ is onto. The operator $\tilde{\mathbf{T}}$, defined in (4.24), is decomposed into $\mathbf{T}_W - \triangle\mathbf{T}_W$, where $\triangle\mathbf{T}_W$: dom $\mathbf{T} \to$ codom $\mathbf{T}$ is given by

$$\triangle\mathbf{T}_W : \begin{bmatrix} P \\ R \end{bmatrix} \mapsto \begin{bmatrix} P \\ R \end{bmatrix} E_{11} - \begin{bmatrix} E_{22} & E_{24} & -E_{14}^T \\ E_{42} & E_{22}^T & E_{12}^T \\ -E_{32} & -E_{21}^T & E_{11}^T \end{bmatrix} \begin{bmatrix} P \\ R \end{bmatrix} - \begin{bmatrix} 0 \\ F \end{bmatrix}$$

with $F = P^T \begin{bmatrix} E_{32}^T \\ E_{21} \end{bmatrix}$. Hence, $\|\triangle\mathbf{T}_W\| \leq \sqrt{3}\|E\|_F$ and Lemma 4.19 implies that

$$\mathbf{X} = \sum_{i=0}^{\infty} (\mathbf{T}_W^\dagger \circ \triangle\mathbf{T}_W)^i \circ \mathbf{T}_W^\dagger$$

converges absolutely and satisfies $\|\mathbf{X}\| \leq \hat{\kappa}$. The remainder of the proof is analogous to the proof of Theorem 4.21. □

The bound (4.37) on the canonical angles between $\mathcal{X}$ and $\hat{\mathcal{X}}$ holds with the quantities γ and κ replaced by $\hat{\gamma}$ and $\hat{\kappa}$:

$$\|\tan\Theta(\mathcal{X},\hat{\mathcal{X}})\|_F \leq 2\hat{\gamma}\hat{\kappa} + 4\hat{\gamma}^2\hat{\kappa}^2 < 2.5\hat{\gamma}\hat{\kappa}. \tag{4.40}$$

Hence, the *structured condition number $c_W(\mathcal{X})$ for a semi-simple isotropic subspace of a skew-Hamiltonian matrix* satisfies

$$c_W(\mathcal{X}) := \lim_{\varepsilon\to 0} \sup_{\substack{\|E\|_F\leq\varepsilon \\ E \text{ skew-Hamiltonian}}} \frac{\|\Theta(\mathcal{X},\hat{\mathcal{X}})\|_F}{\varepsilon} \leq \alpha\|\mathbf{T}_W^\dagger\|$$

for some $\alpha \leq 2$. The presence of the factor α in this bound is artificial; a slight modification of the proof of Theorem 4.21 shows that α can be made arbitrarily close to one under the assumption that the perturbation E is sufficiently small. Thus, $\|\mathbf{T}_W^\dagger\|$ is an upper bound on the structured condition number of $\mathcal{X}$.

To show that $c_W(\mathcal{X})$ and $\|\mathbf{T}_W^\dagger\|$ actually coincide we construct perturbations E so that

$$\lim_{\|E\|_F\to 0} \|\Theta(\mathcal{X},\hat{\mathcal{X}})\|_F/\|E\|_F \geq \|\mathbf{T}_W^\dagger\|.$$

holds. By the well-known Weierstrass theorem we can choose E_{21} and E_{31} so that $\|(E_{21},E_{31})\|_F = 1$ and $\|\mathbf{T}_W^\dagger(E_{21},E_{31})\|_F = \|\mathbf{T}_W^\dagger\|$ hold. Given this and a block Schur decomposition of the form (4.39), we consider the perturbation

$$E = \varepsilon\cdot[Z, J^TX, J^TZ]\begin{bmatrix} E_{21} \\ E_{31}\end{bmatrix} X^T.$$

By choosing a sufficiently small ε we may assume w.l.o.g. that there exists an invariant subspace $\hat{\mathcal{X}}$ of $W+E$ satisfying $\|\Theta(\mathcal{X},\hat{\mathcal{X}})\|_2 < \frac{\pi}{2}$. This implies the existence of matrices P and Q so that the columns of

$$\hat{X} = \left(X + [Z, J^TZ, J^TX]\begin{bmatrix} P \\ Q\end{bmatrix}\right)(I + P^TP + Q^TQ)^{-1/2}$$

form an orthonormal basis of $\hat{\mathcal{X}}$. We have seen that any such matrix pair (P,Q) must satisfy the nonlinear matrix equation

$$\mathbf{T}_W(P,R) - \triangle\mathbf{T}_W(P,R) + \Phi(P,R) = \varepsilon\begin{bmatrix} E_{21} \\ E_{31}\end{bmatrix}, \tag{4.41}$$

where R, $\triangle\mathbf{T}_W$ and Φ are defined as in (4.21), (4.30) and (4.35), respectively. If we decompose

$$(P,R) = (P_1+P_2, R_1+R_2), \quad (P_1,R_1)\in\text{kernel}(\mathbf{T}_W),\ (P_2,R_2)\in\text{kernel}(\mathbf{T}_W)^\perp,$$

then

$$(P_2,R_2) = \varepsilon\cdot\mathbf{T}_W^\dagger(E_{21},E_{31}) + \mathbf{T}_W^\dagger\circ[\triangle\mathbf{T}_W(P,R) - \Phi(P,R)].$$

Since $\|(P,R)\| = \mathcal{O}(\varepsilon)$, it follows that $\|\triangle\mathbf{T}_W(P,R) - \Phi(P,R)\|_F = \mathcal{O}(\varepsilon^2)$ and thus

$$\lim_{\varepsilon\to 0}\|(P_2,R_2)\|_F/\varepsilon = \|\mathbf{T}_W^\dagger(E_{21},E_{31})\|_F = \|\mathbf{T}_W^\dagger\|.$$

Combining this equality with $\|(P,R)\|_F \geq \|(P_2,R_2)\|_F$ and $\|\Theta(\mathcal{X},\hat{\mathcal{X}})\|_F = \|(P,R)\|_F + \mathcal{O}(\varepsilon^2)$ yields the desired result:

$$\lim_{\varepsilon\to 0}\|\Theta(\mathcal{X},\hat{\mathcal{X}})\|_F/\varepsilon \geq \|\mathbf{T}_W^\dagger\|.$$

On the computation of $\|\mathbf{T}_W^\dagger\|$

We have shown that $\|\mathbf{T}_W^\dagger\|$ is the appropriate structured condition number for a semi-simple isotropic invariant subspace $\mathcal{X}$ of a skew-Hamiltonian matrix W. It remains to compute this quantity. It turns out that $\|\mathbf{T}_W^\dagger\|$ is easy to compute if $k = 1$ (real eigenvectors).

Lemma 4.25. *Let $\lambda \in \mathbb{R}$ be an eigenvalue of a skew-Hamiltonian matrix W having algebraic multiplicity two, and let x be an associated eigenvector satisfying $\|x\|_2 = 1$. Given a partitioning of the form (4.39) with respect to x it follows that*

$$\|\mathbf{T}_W^\dagger\| = \sigma_{\min}(W_\lambda)^{-1},$$

where $\sigma_{\min}(W_\lambda)$ denotes the minimum singular value of the matrix

$$W_\lambda = \begin{bmatrix} A_{22} - \lambda I & G_{22} & -G_{12}^T \\ H_{22} & A_{22}^T - \lambda I & A_{12}^T \end{bmatrix}.$$

Proof. The operator $\mathbf{T}_W$ can be identified with $\left[\begin{smallmatrix} W_\lambda \\ 0 \end{smallmatrix}\right]$. Hence,

$$\|\mathbf{T}_W^\dagger\| = \sup_{\|x\|_2=1} \|\mathbf{T}_W^\dagger(x,0)\|_2 = \sup_{\|x\|_2=1} \|W_\lambda^\dagger x\|_2 = \sigma_{\min}(W_\lambda)^{-1},$$

using the fact that the space of 1×1 skew-symmetric matrices is $\{0\}$. $\square$

If U^TWU is in skew-Hamiltonian Schur form (4.13) then $H_{22} = 0$ and A_{22} is in real Schur form. Then the computation of $\|\mathbf{T}_W^\dagger\|$ becomes particularly cheap. Construct an orthogonal matrix Q so that

$$W_\lambda Q = \begin{bmatrix} T_{11} & T_{12} & 0 \\ 0 & T_{22}^T & 0 \end{bmatrix}$$

with upper triangular matrices T_{11} and T_{22}. Since Q can be represented as a product of $\mathcal{O}(n)$ Givens rotations, see [112, Sec. 12.5], the computation of T_{11}, T_{12} and T_{22} requires $\mathcal{O}(n^2)$ flops. In this case, one of the condition number estimators for triangular matrices [122, Ch. 14] can be used to estimate

$$\left\| \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22}^T \end{bmatrix}^{-1} \right\|_2 = \sigma_{\min}(W_\lambda U)^{-1} = \sigma_{\min}(W_\lambda)^{-1}$$

within $\mathcal{O}(n^2)$ flops.

The case $k > 1$ is more complicated. A possible but quite expensive option is provided by the Kronecker product approach that was already used in the proof of Lemma 4.18. Let

$$K_{\mathbf{T}_W} := A_{11}^T \otimes I - I \otimes \begin{bmatrix} A_{22} & G_{22} & -G_{12}^T \\ H_{22} & A_{22}^T & A_{12}^T \\ 0 & 0 & A_{11}^T \end{bmatrix}$$

and let the columns of K_{Skew} form an orthonormal basis for all vectors in vec(codom $\mathbf{T}$). Then $\|\mathbf{T}_W^\dagger\|$ is given by the minimum singular value of the matrix $K_{\text{Skew}}^T K_{\mathbf{T}_W}$. Note that this is an $(2nk - k(3k+1)/2) \times (2nk - k^2)$ matrix and thus a direct method for computing its minimum singular value requires $O(k^3 n^3)$ flops.

Another approach would consist of adapting a condition estimator for Sylvester equations [60, 136], see also Section 2.3 in Chapter 1, to estimate $\|\mathbf{T}_W^\dagger\|$. This would require the application of $\mathbf{T}_W^\dagger$ (and its dual) to particular elements of codom $\mathbf{T}$ (and dom $\mathbf{T}$). The efficient and reliable computation of these "matrix-vector products" is a delicate task, see e.g. [124], and beyond the scope of this treatment.

Numerical Example

An algorithm for computing the derived condition number for a real eigenvector of a skew-Hamiltonian matrix has been implemented in the HAPACK routine `DSHSNA`. Let us illustrate its use with the following $2n \times 2n$ skew-Hamiltonian matrix:

$$W_n = \left[\begin{array}{cccc|cccc} 0 & -1 & \cdots & -1 & 0 & 1 & \cdots & 1 \\ 0 & -1 & \cdots & -1 & -1 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & 1 \\ 0 & \cdots & 0 & -1 & -1 & \cdots & -1 & 0 \\ \hline 0 & \cdots & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & & & \vdots & -1 & -1 & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & -1 & -1 & \cdots & -1 \end{array} \right].$$

We computed exact values of $\|\mathbf{T}_W^\dagger\|$ for the eigenvector e_1 of W_n, $n = 2, \dots, 30$. Furthermore, we applied the algorithm proposed in the previous section to produce estimates of $\|\mathbf{T}_W^\dagger\|$. These theoretical results were compared with practical observations in the following way. A skew-Hamiltonian matrix E with random entries chosen from a standard normal distribution with zero mean, scaled so that $\|E\|_F = 10^{-10}$. Using the HAPACK routine `DSHEVC`, we computed eigenvectors v and w corresponding to two identical eigenvalues of $W_n + E$ that have smallest absolute value. Let the columns of U form an orthonormal basis for $\operatorname{span}\{v, w\}^\perp$. Then the sine of the angle between $\operatorname{span}\{e_1\}$ and $\operatorname{span}\{v, w\}$ is given by $\|U^H e_1\|_2$. The observed value of $\|\mathbf{T}_W^\dagger\|$ was taken as the maximum over all quantities $10^{10} \cdot \|U^H e_1\|_2$ for 500 different samples of E. The results of the computations are displayed in Figure 4.2. It turns out that the exact value of $\|\mathbf{T}_W^\dagger\|$ is underestimated for $n = 2$ by a factor of 0.88 and overestimated for all other values of n by a factor of at most 2.2. Furthermore, the exact value is consistently larger than the observed value, by a factor of at most 20.

2.4 An Algorithm for Computing the Skew-Hamiltonian Schur Decomposition

In Section 2.1 we used a constructive approach to prove the skew-Hamiltonian Schur decomposition

$$U^T W U = \left[\begin{array}{cc} T & \tilde{G} \\ 0 & T^T \end{array} \right], \tag{4.42}$$

where U is orthogonal symplectic and T has real Schur form. The following algorithm summarizes this construction.

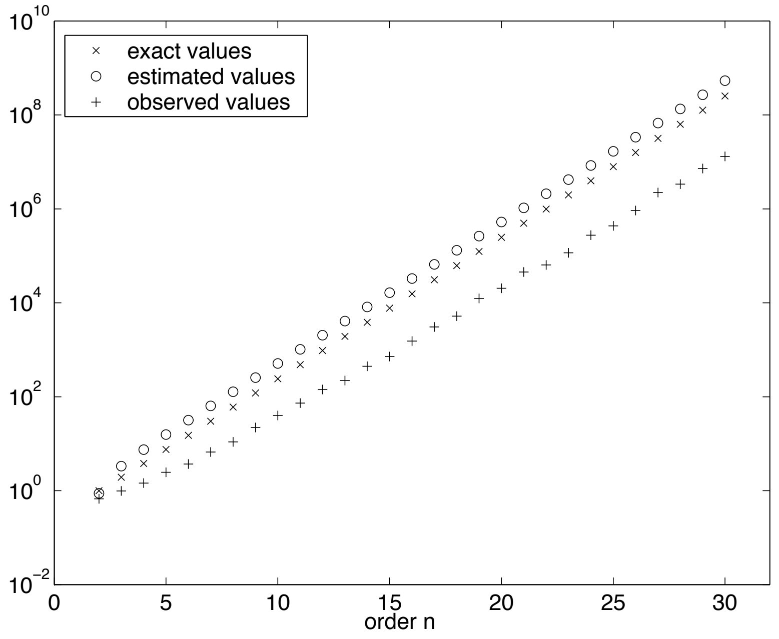


Figure 4.2. *Exact, estimated and observed values of* $\|\mathbf{T}_W^\dagger\|$ *for the eigenvector* e_1 *of* W_n.

Algorithm 4.26 (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^TWU *having skew-Hamiltonian Schur form (4.42).*

```
Apply Algorithm 4.8 to compute an orthogonal symplectic matrix $U$ so
that $W \leftarrow U^TWU$ has PVL form.

Apply the QR algorithm to the $(1,1)$ block $W_{11}$ of W to compute an
orthogonal matrix $Q$ so that $Q^TW_{11}Q$ has real Schur form.

Update $W \leftarrow (Q \oplus Q)^T W (Q \oplus Q)$, $U \leftarrow U(Q \oplus Q)$.
```

This algorithm is implemented in the HAPACK routine `DSHES`; it requires around $\frac{62}{3}n^3$ 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 double-shift QR algorithm listed on page 29. 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 4.26 are as follows. Similarly as for the QR algorithm [264] one can show that there exists an orthogonal symplectic matrix V which transforms the computed skew-Hamiltonian Schur form $\hat{W} = \left[\begin{smallmatrix} \hat{T} & \hat{G} \\ 0 & \hat{T}^T \end{smallmatrix}\right]$ to a skew-Hamiltonian matrix near to W, i.e., $V\hat{W}V^T = W + E$, 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 $\hat{U} = \left[\begin{smallmatrix} \hat{U}_1 & \hat{U}_2 \\ -\hat{U}_2 & \hat{U}_1 \end{smallmatrix}\right]$. This implies that $\hat{U}$ is close to an orthogonal symplectic matrix, see Lemma 4.6.

2.5 Computation of Invariant Subspaces

Once a skew-Hamiltonian Schur decomposition (4.42) 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 7, Chapter 1.

2.6 Other Algorithms

Similarly as the Hessenberg form of a general matrix can be computed by Gauss transformations [112, Sec. 7.4.7] it has been shown by Stefanovski and Trenčevski [216] how non-orthogonal symplectic transformations can be used to compute the PVL form of a skew-Hamiltonian matrix. A modification of the Arnoldi method, suitable for computing eigenvalues and isotropic invariant subspaces of large and sparse skew-Hamiltonian matrices, has been proposed by Mehrmann and Watkins [175], see also Section 4 in Chapter 5.

3 The Hamiltonian Eigenvalue Problem

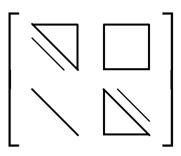
One of the most remarkable properties of a Hamiltonian matrix $H = \left[\begin{smallmatrix} A & G \\ Q & -A^T \end{smallmatrix}\right] \in \mathbb{R}^{2n\times 2n}$ is that its eigenvalues always occur in pairs $\{\lambda, -\lambda\}$, if $\lambda \in \mathbb{R} \cup \imath\mathbb{R}$, or in quadruples $\{\lambda, -\lambda, \bar{\lambda}, -\bar{\lambda}\}$, 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 of using a structure-preserving algorithm for computing the eigenvalues of H.

Generally, 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 [193, 194, 195, 164, 165, 101] 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.

3.1 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 4.8 can be used to construct an orthogonal symplectic matrix U so that

$$U^T H U = \left[\begin{array}{cc} \tilde{A} & \tilde{G} \\ \tilde{Q} & -\tilde{A}^T \end{array}\right] =$$



,

i.e., $\tilde{A}$ has upper Hessenberg form and $\tilde{Q}$ is a diagonal matrix, this Hamiltonian PVL decomposition [186] is of limited use. The Hamiltonian QR algorithm, see Section 3.4 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 [59] 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 [6], 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

$$x^T J H^{2i-1} x = 0, \quad i = 1, \dots, n.$$

This result can even be extended to non-orthogonal symplectic transformations [192].

A Schur-like form for Hamiltonian matrices is given by the following theorem [186, 165].

Theorem 4.27. *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.,*

$$U^T H U = \begin{bmatrix} T & \tilde{G} \\ 0 & -T^T \end{bmatrix}, \tag{4.43}$$

where $T \in \mathbb{R}^{n \times n}$ has real Schur form.

3.2 Structured Condition Numbers for Eigenvalues

An extensive perturbation analysis of (block) Hamiltonian Schur forms for the case that H has no purely imaginary eigenvalues has been presented in [144]. The analysis used therein is based on the technique of splitting operators and Lyapunov majorants. The approach used in this and the next section is somewhat simpler; it is based on the perturbation expansions given in Theorems 1.5 and 1.9.

Let λ be a simple eigenvalue of a Hamiltonian matrix H with right and left eigenvectors x and y, respectively. The perturbation expansion (1.20) implies that for a sufficiently small perturbation E, there exists an eigenvalue $\hat{\lambda}$ of $W + E$ so that

$$|\hat{\lambda} - \lambda| = \frac{|y^H E x|}{|y^H x|} + \mathcal{O}(\|E\|^2) \leq \frac{\|x\|_2 \cdot \|y\|_2}{|y^H x|} \|E\|_2 + \mathcal{O}(\|E\|^2). \tag{4.44}$$

If λ is real then we may assume that x and y are real and normalized, i.e., $\|x\|_2 = \|y\|_2 = 1$. For the Hamiltonian perturbation $E = \varepsilon[y, Jx] \cdot [x, J^T y]^H$ we have $|y^H E x| = \varepsilon(1 + |y^H J x|^2)$ and

$$\|E\|_2 = \varepsilon \|[x, Jy]\|_2^2 = \varepsilon(1 + |y^H J x|).$$

The minimum of $(1 + |y^H J x|^2)/(1 + |y^H J x|)$ is $\beta = 2\sqrt{2} - 2$. This implies that for $\varepsilon \to 0$ both sides in (4.44) differ at most by a factor $1/\beta$. Hence, the *structured eigenvalue condition number for a simple eigenvalue of a Hamiltonian matrix*,

$$c_H(\lambda) := \lim_{\varepsilon \to 0} \sup_{\substack{\|E\|_2 \leq \varepsilon \\ E \text{ is Hamiltonian}}} \frac{|\hat{\lambda} - \lambda|}{\varepsilon},$$

satisfies the inequalities

$$(2\sqrt{2} - 2) c(\lambda) \leq c_H(\lambda) \leq c(\lambda),$$

if $\lambda \in \mathbb{R}$, where $c(\lambda)$ stands for the standard condition number of λ, see Section 2.2 in Chapter 1. This inequality still holds for complex λ if one allows *complex* Hamiltonian perturbations E, i.e., $(EJ)^H = EJ$. A tight lower bound for the structured condition number of a complex eigenvalue under real perturbations is an open problem.

Again, structured backward errors and condition numbers for eigenvalues of Hamiltonian matrices with additional structures can be found in [234, 235].

3.3 Structured Condition Numbers for 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 always 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

$$U^T H U = \begin{array}{c} \\ k \\ n-k \\ k \\ n-k \end{array} \begin{array}{c} \begin{array}{cccc} k & n-k & k & n-k \end{array} \\ \left[\begin{array}{cccc} A_{11} & A_{12} & G_{11} & G_{12} \\ 0 & A_{22} & G_{12}^T & G_{22} \\ 0 & 0 & -A_{11}^T & 0 \\ 0 & Q_{22} & -A_{12}^T & -A_{22}^T \end{array}\right] \end{array}.$$

Assuming that the perturbation E is sufficiently small, the perturbation expansion (1.21) implies that there exists a matrix $\hat{X}$ so that $\hat{\mathcal{X}} = \operatorname{range}(\hat{X})$ is an invariant subspace of $H + E$ satisfying

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

and $\hat{X}^T(\hat{X} - X) = 0$. The involved Sylvester operator $\mathbf{T}_H$ is given by

$$\mathbf{T}_H : (R_1, R_2, R_3) \mapsto \begin{bmatrix} A_{22} & G_{12}^T & G_{22} \\ 0 & -A_{11}^T & 0 \\ Q_{22} & -A_{12}^T & -A_{22}^T \end{bmatrix} \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} - \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} A_{11}.$$

If the perturbation E is Hamiltonian, then $X_\perp^T E X$ takes the form $S = [A_{21}^T, Q_{11}^T, Q_{21}^T]^T$, where $Q_{11} \in \operatorname{Sym}(k)$ and A_{21}, Q_{21} are general $(n-k)\times k$ matrices. Hence, if we let

$$\|\mathbf{T}_H^{-1}\| := \sup_{S\neq 0} \left\{ \frac{\|\mathbf{T}_H^{-1}(S)\|_F}{\|S\|_F} \mid S \in \mathbb{R}^{(n-k)\times k} \times \operatorname{Sym}(k) \times \mathbb{R}^{(n-k)\times k} \right\},$$

then the *structured condition number for an isotropic invariant subspace of a Hamiltonian matrix* satisfies

$$c_H(\mathcal{X}) = \lim_{\varepsilon\to 0} \sup_{\substack{\|E\|_F \leq \varepsilon \\ E \text{ Hamiltonian}}} \frac{\|\Theta(\mathcal{X}, \hat{\mathcal{X}})\|_F}{\varepsilon} = \|\mathbf{T}_H^{-1}\|.$$

Obviously, this quantity coincides with the unstructured condition number 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 left half plane. To show this, first note that in this case

$$\|\mathbf{T}_H^{-1}\| = \sup_{\substack{S\neq 0 \\ S\in\operatorname{Sym}(n)}} \frac{\|\mathbf{T}_H^{-1}(S)\|_F}{\|S\|_F} = \inf_{\substack{S\neq 0 \\ S\in\operatorname{Sym}(n)}} \frac{\|A_{11}S + SA_{11}^T\|_F}{\|S\|_F},$$

where $\lambda(A_{11}) \subset \mathbb{C}^-$. Using a result by Byers and Nash [65], we have

$$\inf_{\substack{S \neq 0 \\ S \in \mathrm{Sym}(n)}} \frac{\|A_{11}S + SA_{11}^T\|_F}{\|S\|_F} = \inf_{S \neq 0} \frac{\|A_{11}S + SA_{11}^T\|_F}{\|S\|_F},$$

which indeed shows that the structured and unstructured condition numbers for the maximal stable invariant subspace coincide.

However, there is a severe loss if we do not require E to be Hamiltonian; the subspace $\hat{\mathcal{X}}$ may not be isotropic. To obtain a nearby isotropic subspace one can apply the symplectic QR decomposition to an orthonormal basis $\hat{X}$ of $\hat{\mathcal{X}}$. This yields the orthonormal basis Z of an isotropic subspace $\mathcal{Z} = \operatorname{span} Z$ so that

$$\|Z - X\|_F \leq 2\|\hat{X} - X\|_F \leq 2c_H(\mathcal{X})\|E\|_F + \mathcal{O}(\|E\|_F^2).$$

Note that for the original subspace $\hat{\mathcal{X}}$ we have the desirable property $\|\hat{X}_\perp^T H \hat{X}\|_F = \|E\|_F$, where the columns of $\hat{X}_\perp$ form an orthonormal basis for $\hat{\mathcal{X}}^\perp$. For the isotropic subspace $\mathcal{Z}$, however, we can only guarantee

$$\|(JZ)^T HZ\|_F \leq 4c_H(\mathcal{X}) \cdot \|H\|_F \cdot \|E\|_F + \mathcal{O}(\|E\|_F^2),$$

which signals a severe loss of backward stability. The following numerical example demonstrates the undesirable appearance of the factor $c_H(\mathcal{X})$ in $\|(JZ)^T HZ\|_F$.

Example 4.28. *Let*

$$H = \begin{bmatrix} -10^{-5} & -1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 10^{-5} & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

and consider the stable invariant subspace spanned by the columns of $X = [I_2, 0]^T$, which has condition number 10^5. If we add a random (non-Hamiltonian) perturbation E with $\|E\|_F = 10^{-10}$ to H, and compute (using MATLAB*) an orthonormal basis $\hat{X}$ for the invariant subspace $\hat{\mathcal{X}}$ of $H + E$ belonging to the eigenvalues in the open left half plane, we observe that*

$$\|\hat{X}_\perp^T H \hat{X}\|_F \approx 4.0 \times 10^{-11}.$$

By computing a symplectic QR decomposition of $\hat{X}$ we constructed an orthonormal basis Z satisfying $Z^T(JZ) = 0$ and observed

$$\|(J\tilde{Z})^T H \tilde{Z}\|_F \approx 4.7 \times 10^{-6}.$$

3.4 An Explicit Hamiltonian QR Algorithm

Byers' Hamiltonian QR algorithm [59] 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 [59, 61].

One iteration of the Hamiltonian QR algorithm computes the symplectic QR decomposition of the first n columns of the symplectic matrix

$$M = [(H - \sigma_1 I)(H - \sigma_2 I)][(H + \sigma_1 I)(H + \sigma_2 I)]^{-1}, \tag{4.45}$$

where $\{\sigma_1, \sigma_2\}$ is a pair of real or complex conjugate shifts. This yields an orthogonal symplectic matrix U so that

$$U^T M = \left[\begin{array}{cc} ◹ & \square \\ & ◺ \end{array}\right]. \tag{4.46}$$

The next iterate is obtained by updating $H \leftarrow U^T H U$. Let us partition H as follows:

$$H = \begin{array}{c} \\ 2 \\ n-2 \\ 2 \\ n-2 \end{array} \begin{array}{c} \begin{array}{cccc} 2 & n-2 & 2 & n-2 \end{array} \\ \left[\begin{array}{cccc} A_{11} & A_{12} & G_{11} & G_{12} \\ A_{21} & A_{22} & G_{12}^T & G_{22} \\ Q_{11} & Q_{12} & -A_{11}^T & -A_{21}^T \\ Q_{12}^T & Q_{22} & -A_{12}^T & -A_{22}^T \end{array}\right] \end{array}. \tag{4.47}$$

In Section 3.1, Chapter 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 s_1, s_2 as those 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 has two imaginary eigenvalues, then we suggest to choose the one eigenvalue with positive real part twice, and if there are four purely imaginary eigenvalues, then our suggestion is to choose ad hoc shifts.

If the norms of the blocks A_{21}, Q_{11} and Q_{12} become less than $\mathbf{u} \cdot \|H\|_F$, then we may safely regard them as zero and apply the iteration to the submatrix $\left[\begin{smallmatrix} A_{22} & G_{22} \\ Q_{22} & -A_{22}^T \end{smallmatrix}\right]$. This will finally yield 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. In our numerical experiments, however, we often observed 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 4.29. One can avoid the explicit computation of the potentially ill-conditioned matrix M in (4.45) 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 (4.46). This can be achieved by a minor modification of the standard RQ decomposition [38]. Secondly, the orthogonal symplectic matrix U is computed from the symplectic QR decomposition of the first n columns of $(H - \sigma_1 I)(H - \sigma_2 I)Q_r^T$.

3.5 Reordering a Hamiltonian Schur Decomposition

If the Hamiltonian QR algorithm has successfully computed a Hamiltonian Schur decomposition,

$$U^T H U = \begin{bmatrix} T & \tilde{G} \\ 0 & -T^T \end{bmatrix} \tag{4.48}$$

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 (4.48) 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 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 7.2, Chapter 1. Now, let G_{mm} denote the corresponding block in $\tilde{G}$; it remains to find an orthogonal symplectic matrix U_{mm} so that

$$U_{mm}^T \begin{bmatrix} T_{mm} & G_{mm} \\ 0 & -T_{mm}^T \end{bmatrix} U_{mm} = \begin{bmatrix} \tilde{T}_{mm} & \tilde{G}_{mm} \\ 0 & -\tilde{T}_{mm}^T \end{bmatrix} \tag{4.49}$$

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 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 shown that U_{mm} is an orthogonal symplectic matrix which produces a reordering of the form (4.49). As for the swapping algorithm described in Section 7.1, Chapter 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 [59].

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.

3.6 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 [243]. 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 4.26 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, has been implemented in [27]. 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 [243] shows that for an eigenvalue λ of H the computed $\hat{\lambda}$ satisfies

$$|\hat{\lambda} - \lambda| \lessapprox c(\lambda) \cdot \min\{\mathbf{u}\|H\|_2^2/|\lambda|, \sqrt{\mathbf{u}}\|H\|_2\}.$$

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^TA, see e.g. [223, 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 [38]. In the following, we show how this algorithm can be directly derived from Algorithm 4.26, quite similar to the derivation of the periodic QR algorithm in Section 3, Chapter 3.

In lieu of H^2 we make use of the skew-Hamiltonian matrix

$$W = \begin{bmatrix} 0 & A & 0 & G \\ -A & 0 & -G & 0 \\ 0 & Q & 0 & -A^T \\ -Q & 0 & A^T & 0 \end{bmatrix} \in \mathbb{R}^{4n\times 4n}, \tag{4.50}$$

for given $H = \left[\begin{smallmatrix} A & G \\ Q & -A^T \end{smallmatrix}\right]$. As W is permutationally similar to $\left[\begin{smallmatrix} 0 & H \\ -H & 0 \end{smallmatrix}\right]$, we see that $\pm\lambda$ is an eigenvalue of H if and only if $\pm\sqrt{-\lambda^2}$ is an eigenvalue of W. Note that the matrix W has a lot of extra structure besides being skew-Hamiltonian, which is not exploited if we apply Algorithm 4.26 directly to W.

Instead, we consider the shuffled matrix $\tilde{W} = (P \oplus P)^T W (P \oplus P)$, where

$$P = \begin{bmatrix} e_1 & e_3 & \cdots & e_{2n-1} & e_2 & e_4 & \cdots & e_{2n} \end{bmatrix}.$$

This matrix has the form

$$\tilde{W} = \begin{bmatrix} \tilde{W}_A & \tilde{W}_G \\ \tilde{W}_Q & -\tilde{W}_A^T \end{bmatrix}, \tag{4.51}$$

where each of the matrices $\tilde{W}_A, \tilde{W}_G$ and $\tilde{W}_Q$ is a block matrix composed of two-by-two blocks having the form

$$\tilde{W}_X = \left(\begin{bmatrix} 0 & x_{ij} \\ -x_{ij} & 0 \end{bmatrix} \right)_{i,j=1}^{n}.$$

If an orthogonal symplectic matrix $\tilde{Q}$ has the form

$$\tilde{Q} = (P \oplus P)^T \begin{bmatrix} U_1 & 0 & U_2 & 0 \\ 0 & V_1 & 0 & V_2 \\ -U_2 & 0 & U_1 & 0 \\ 0 & -V_2 & 0 & V_1 \end{bmatrix} (P \oplus P), \tag{4.52}$$

then $\tilde{Q}^T \tilde{W} \tilde{Q}$ is skew-Hamiltonian and has the same zero pattern as $\tilde{W}$.

Lemma 4.30. *If Algorithm 4.8 is applied to the skew-Hamiltonian matrix $\tilde{W}$ defined in 4.51 then the orthogonal symplectic factor of the computed PVL decomposition has the form (4.52).*

Proof. Assume that after $(j-1)$ loops of Algorithm 4.8 the matrix $\tilde{W}$ has been overwritten by a matrix with the same zero pattern as $\tilde{W}$. Let $\tilde{x}$ denote the jth column of $\tilde{W}$. If j is odd then $\tilde{x}$ can be written as $\tilde{x} = x \otimes e_2$, where x is a vector of length $2n$ and $e_2 = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]$. This implies that the elementary orthogonal symplectic matrix

$$E_{j+1}(\tilde{x}) = E(x \otimes e_2)(H_{j+1} \oplus H_{j+1})(\tilde{v}, \beta) \cdot G_{j+1,n+j+1}(\theta) \cdot (H_{j+1} \oplus H_{j+1})(\tilde{w}, \gamma)$$

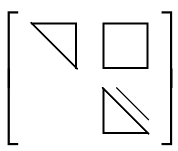
computed by Algorithm 4.1 satisfies $\tilde{v} = v \otimes e_2$ and $\tilde{w} = w \otimes e_2$ for some $v, w \in \mathbb{R}^{2n}$. Thus, $E_{j+1}(\tilde{x})$ has the same zero pattern as the matrix $\tilde{Q}$ in (4.52). By similar arguments the same holds for even j. This shows that the jth loop of Algorithm 4.8 preserves the zero pattern of $\tilde{W}$. The proof is concluded by using the fact that the set of matrices having the form (4.52) is closed under multiplication. $\square$

Note that this lemma also shows that the PVL decomposition returned by Algorithm 4.8 applied to $\tilde{W}$ must take the form

$$\tilde{Q}^T \tilde{W} \tilde{Q} = (P \oplus P)^T \begin{bmatrix} 0 & R_{11} & 0 & R_{12} \\ -R_{22} & 0 & -R_{12}^T & 0 \\ 0 & 0 & 0 & -R_{22}^T \\ 0 & 0 & R_{11}^T & 0 \end{bmatrix} (P \oplus P), \tag{4.53}$$

where R_{11} is an upper triangular matrix and R_{22} is an upper Hessenberg matrix. Rewriting (4.53) in terms of the block entries of $\tilde{W}$ and $\tilde{Q}$ yields

$$U^T H V = \begin{bmatrix} R_{11} & R_{12} \\ 0 & -R_{21}^T \end{bmatrix} = \tag{4.54}$$



with the orthogonal symplectic matrices $U = \left[\begin{smallmatrix} U_1 & U_2 \\ -U_2 & U_1 \end{smallmatrix}\right]$ and $V = \left[\begin{smallmatrix} V_1 & V_2 \\ -V_2 & V_1 \end{smallmatrix}\right]$. This is a so called *symplectic URV decomposition* [38].

As Algorithm 4.8 exclusively operates on the nonzero entries of $\tilde{W}$ it is possible to reformulate it purely in terms of these entries. It can be shown that this amounts to the following algorithm [38, Alg. 4.4].

Algorithm 4.31 (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 $U^T H V$ having the form (4.54).*

```
$U \leftarrow I_{2n}$, $V \leftarrow I_{2n}$.
FOR $j \leftarrow 1, 2, \dots, n$
  Set $x \leftarrow He_j$.
  Apply Algorithm 4.1 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 4.1 to compute $E_{n+j+1}(y)$.
    Update $H \leftarrow HE_{n+j+1}(y)$, $V \leftarrow VE_{n+j+1}(y)$.
  END IF
END FOR
```

Algorithm 4.31 is implemented in the HAPACK routines `DGESUV`; 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.

Let us illustrate the first two loops of Algorithm 4.31 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 entries $(2:8,1)$ of H:

$$H \leftarrow E_1(He_1)^T H = \left[\begin{array}{cccc|cccc} \hat{a} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hat{0} & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hline \hat{0} & \hat{q} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hat{0} & \hat{q} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hat{0} & \hat{q} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ \hat{0} & \hat{q} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \end{array}\right].$$

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:

$$H \leftarrow E_2(He_2)^T H = \left[\begin{array}{cccc|cccc} a & a & a & a & g & g & g & g \\ 0 & \hat{a} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ 0 & \hat{0} & \hat{a} & \hat{a} & \hat{g} & \hat{g} & \hat{g} & \hat{g} \\ \hline 0 & 0 & 0 & 0 & b & b & 0 & 0 \\ 0 & \hat{0} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{0} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \\ 0 & \hat{0} & \hat{q} & \hat{q} & \hat{b} & \hat{b} & \hat{b} & \hat{b} \end{array}\right],$$

$$H \leftarrow HE_7(H^T e_6) = \left[\begin{array}{cccc|cccc} a & a & \hat{a} & \hat{a} & g & g & \hat{g} & \hat{g} \\ 0 & a & \hat{a} & \hat{a} & g & g & \hat{g} & \hat{g} \\ 0 & 0 & \hat{a} & \hat{a} & g & g & \hat{g} & \hat{g} \\ 0 & 0 & \hat{a} & \hat{a} & g & g & \hat{g} & \hat{g} \\ \hline 0 & 0 & 0 & 0 & b & b & 0 & 0 \\ 0 & 0 & \hat{0} & \hat{0} & b & b & \hat{b} & \hat{0} \\ 0 & 0 & \hat{q} & \hat{q} & b & b & \hat{b} & \hat{b} \\ 0 & 0 & \hat{q} & \hat{q} & b & b & \hat{b} & \hat{b} \end{array}\right].$$

Now, the second step of Algorithm 4.26 applied to the $4n \times 4n$ skew-Hamiltonian matrix $\tilde{W}$ as defined in (4.51) consists of applying the QR algorithm to the upper left $2n \times 2n$ block of the PVL form (4.53). We have seen in Section 3, Chapter 3, that this is equivalent to applying the periodic QR algorithm to the matrix product $-R_{22} \cdot R_{11}$, which constructs orthogonal matrices Q_1 and Q_2 so that $Q_1^T R_{22} Q_2$ is reduced to real Schur form while $Q_2^T R_{11} Q_1$ stays upper triangular. The periodic QR algorithm is a backward stable method for computing the eigenvalues of $R_{22} \cdot R_{11}$. The positive and negative square roots of these eigenvalues are the eigenvalues of H.

The procedure, as described above, 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 its complexity is $\mathcal{O}(n^3)$. 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 clear whether the method is strongly backward stable or not.

3.7 Computation of Invariant Subspaces Based on H^2

Having computed an invariant subspace for the skew-Hamiltonian matrix H^2 it is possible to extract invariant subspaces for H from it [128, 268]. However, we have already observed that the explicit computation of H^2 can lead to numerical instabilities and should be avoided. The above idea of embedding H in a skew-Hamiltonian matrix W of double dimension can be extended for computing invariant subspaces, see [37]. However, it should be noted that this approach might encounter numerical difficulties if H has eigenvalues on or close to the imaginary axis.

3.8 Refinement of Stable Invariant Subspaces

With all the difficulties in deriving a strongly backward stable method it might be preferable to use some kind of iterative refinement algorithm to improve the quantities computed by a less stable method. This idea is used, for example, in the multishift algorithm [5] and hybrid methods for solving algebraic Riccati equations [31].

In the following we describe a method for improving an isotropic subspace $\hat{\mathcal{X}}$ that approximates the stable invariant subspace $\mathcal{X}$ of a Hamiltonian matrix H. Let the columns of $\hat{X}$ form an orthonormal basis for $\hat{\mathcal{X}}$ and consider

$$\begin{bmatrix} \hat{X} & J\hat{X} \end{bmatrix}^T H \begin{bmatrix} \hat{X} & J\hat{X} \end{bmatrix} = \begin{bmatrix} \tilde{A} & \tilde{G} \\ \tilde{Q} & -\tilde{A}^T \end{bmatrix}.$$

If $\hat{X}$ has been computed by a strongly backward stable method then $\|\tilde{Q}\|_F$ is of order $\mathbf{u} \cdot \|H\|_F$ and it is not possible to refine $\hat{X}$ much further. However, as we have seen in Example 4.28, if a less stable method has been used then $\|\tilde{Q}\|_F$ might be much larger. In this case we can apply the following algorithm to improve the accuracy of $\hat{X}$.

Algorithm 4.32.

Input: *A Hamiltonian matrix* $H \in \mathbb{R}^{2n \times 2n}$*, a matrix* $\hat{X} \in \mathbb{R}^{2n \times n}$ *so that* $[\hat{X}, J\hat{X}]$ *is orthogonal, and a tolerance* $\mathrm{tol} > 0$*.*

Output: *The matrix* $\hat{X}$ *is updated until* $\|(J\hat{X})^T H \hat{X}\|_F \leq \mathrm{tol} \cdot \|H\|_F$*.*

```
WHILE $\|(J\hat{X})^T H\hat{X}\|_F > \mathrm{tol} \cdot \|H\|_F$ DO
```

```
        Set $\tilde{A} \leftarrow \hat{X}^T H \hat{X}$ and $\tilde{Q} \leftarrow (J\hat{X})^T H \hat{X}$.
        Solve the Lyapunov equation $R\tilde{A} + \tilde{A}^T R = -\tilde{Q}$.
        Compute $Y \in \mathbb{R}^{2n \times n}$ so that $[Y, JY]$ is orthogonal and

                 $\mathrm{range}(Y) = \mathrm{range}\left(\left[\begin{array}{c} I \\ -R \end{array}\right]\right)$,

        using a symplectic QR decomposition.
        Update $[\hat{X}, J\hat{X}] \leftarrow [\hat{X}, J\hat{X}] \cdot [Y, JY]$.
    END WHILE
```

As this algorithm is a special instance of a Newton method for refining invariant subspaces [69, 79, 219] or a block Jacobi-like algorithm [127] it converges locally quadratic. On the other hand, Algorithm 4.32 can be seen as a particular implementation of a Newton method for solving algebraic Riccati equation [142, 154, 173, 174]. By a more general result in [116], this implies under some mild conditions global convergence if H has no eigenvalues on the imaginary axis and if the iteration is initialized with a matrix $\hat{X}$ so that all eigenvalues of $\tilde{A} = \hat{X}^T H \hat{X}$ are in the open left half plane $\mathbb{C}^-$.

In finite precision arithmetic, the minimal attainable tolerance is tol $\approx n^2 \cdot \mathbf{u}$ under the assumption that a forward stable method such as the Bartels-Stewart method [18] is used to solve the Lyapunov equations $R\tilde{A} + \tilde{A}^T R = -\tilde{Q}$ [122, 233].

3.9 Other Algorithms

As mentioned in the introduction there is a vast number of algorithms for the Hamiltonian eigenvalue problem available. Other algorithms based on orthogonal transformations are the Hamiltonian Jacobi algorithm [63, 54], its variants for Hamiltonian matrices that have additional structure [96] and the multishift algorithm [5]. Algorithms based on symplectic but non-orthogonal transformations include the SR algorithm [55, 53, 173] and related methods [56, 192]. A completely different class of algorithms is based on the matrix sign function, see, e.g., [25, 173, 205] and the references therein. Other Newton-like methods directed towards the computation of invariant subspaces for Hamiltonian matrices can be found in [2, 116].

A structure-preserving Arnoldi method based on the H^2 approach was developed in [175], see also Section 4 in Chapter 5. There are a number of symplectic Lanczos methods available, see [30, 98, 259].

4 Symplectic Balancing

We have seen in Section 4.3, Chapter 1, that balancing is 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 [26] has developed a special-purpose balancing algorithm that is based on symplectic equivalence transformations and thus preserves the structure of H. As general balancing, it consists of two stages, which are described in the following two subsections.

4.1 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 [26, Eq. (3.10)], indicating that the group of symplectic permutation matrices is too restrictive to obtain useful classifications for $P^T HP$. Instead, we propose to broaden the range of similarity transformations to $\tilde{P}^T H\tilde{P}$, where $\tilde{P} = DP$ is symplectic, $D = \operatorname{diag}\{\pm 1, \dots, \pm 1\}$ 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:

$$P_{ij}^{(d)} = P_{ij} \oplus P_{i,j}, \tag{4.55}$$

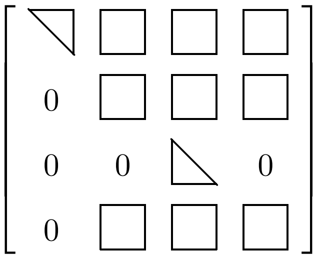
where $1 \le i < j \le n$, $P_{kl} \in \mathbb{R}^{n\times n}$ defined as in (1.48), and

$$P_i^{(s)} = I_{2n} - \begin{bmatrix} e_i & e_{i+n} \end{bmatrix} \begin{bmatrix} e_i^T \\ e_{i+n}^T \end{bmatrix} + \begin{bmatrix} e_i & -e_{i+n} \end{bmatrix} \begin{bmatrix} e_{i+n}^T \\ e_i^T \end{bmatrix}, \tag{4.56}$$

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 $P_i^{(s)}$.

Algorithm 1.31, which permutes a general matrix, uses permutations that swap the jth column with the i_lth column, which is the first column of the active submatrix. The same effect can be achieved by using $P_{i_l,j}^{(d)}$ (if $j \le n$) or $P_{i_l,(j-n)}^{(d)} \cdot P_{i_l}^{(s)}$ (if $j > n$). Note that there is no need for searching a row having zero off-diagonal elements in the active Hamiltonian submatrix as this is only the case if there is a column sharing this property. With this modifications, Algorithm 1.31 produces a symplectic generalized permutation matrix $\tilde{P}$ so that

$$\tilde{P}^T H \tilde{P} = \begin{bmatrix} A_{11} & A_{21} & G_{11} & G_{12} \\ 0 & A_{22} & G_{12}^T & G_{22} \\ 0 & 0 & -A_{11}^T & 0 \\ 0 & Q_{22} & -A_{21}^T & -A_{22}^T \end{bmatrix} = \tag{4.57}$$



where $A_{11} \in \mathbb{R}^{i_l \times i_l}$ is an upper triangular matrix. The unreduced Hamiltonian submatrix $\left[\begin{smallmatrix} A_{22} & G_{22} \\ Q_{22} & A_{22}^T \end{smallmatrix}\right]$ is characterized by the property that all columns have at least one nonzero off-diagonal element. The modified algorithm reads as follows.

Algorithm 4.33.

Input: *A Hamiltonian matrix* $H = \left[\begin{smallmatrix} A & G \\ Q & -A^T \end{smallmatrix}\right] \in \mathbb{R}^{2n\times 2n}$.

Output: *A symplectic generalized permutation matrix* $\tilde{P} \in \mathbb{R}^{2n\times 2n}$ *so that* $\tilde{P}^T H\tilde{P}$ *is in block triangular form (4.57) for some integer* i_l. *The matrix* H *is overwritten by* $\tilde{P}^T H\tilde{P}$.

```
$\tilde{P} \leftarrow I_{2n}$
$i_l \leftarrow 1$
```

```
swapped $\leftarrow 1$
WHILE (swapped $= 1$)
  swapped $\leftarrow 0$
  % Search for column having only zero off-diagonal elements in active
  % submatrix $\begin{bmatrix} A(i_l{:}n,i_l{:}n) & G(i_l{:}n,i_l{:}n) \\ Q(i_l{:}n,i_l{:}n) & -A(i_l{:}n,i_l{:}n)^T \end{bmatrix}$ and swap with $i_l$th column.
  $j \leftarrow i_l$
  WHILE $(j \leq n)$ AND (swapped $= 0$)
    IF $\sum_{\substack{i=i_l \\ i\neq j}}^{n} |a_{ij}| + \sum_{i=i_l}^{n} |q_{ij}| = 0$ THEN
      swapped $\leftarrow 1$
      $H \leftarrow \big(P^{(d)}_{i_l,j}\big)^T H P^{(d)}_{i_l,j}$
      $\tilde{P} \leftarrow \tilde{P} \cdot P^{(d)}_{i_l,j}$
      $i_l \leftarrow i_l + 1$
    END IF
    $j \leftarrow j + 1$
  END WHILE
  $j \leftarrow i_l$
  WHILE $(j \leq n)$ AND (swapped $= 0$)
    IF $\sum_{i=i_l}^{n} |g_{ij}| + \sum_{\substack{i=i_l \\ i\neq j}}^{n} |a_{ji}| = 0$ THEN
      swapped $\leftarrow 1$
      $H \leftarrow \big(P^{(d)}_{i_l,j} P^{(s)}_{i_l}\big)^T H \big(P^{(d)}_{i_l,j} P^{(s)}_{i_l}\big)$
      $\tilde{P} \leftarrow \tilde{P} \cdot \big(P^{(d)}_{i_l,j} P^{(s)}_{i_l}\big)$
      $i_l \leftarrow i_l + 1$
    END IF
    $j \leftarrow j + 1$
  END WHILE
END WHILE
```

The difference between this algorithm and a similar algorithm proposed by Benner [26, Alg. 3.4] is that the latter algorithm does not make use of the generalized permutation matrices $P^{(s)}_{i_l}$, which results in a more complicated reduced form than (4.57). However, it should be emphasized that Benner also describes a simple procedure to obtain (4.57) from this more complicated form. Counting the number of required comparisons, it is argued in [26] that Algorithm 4.33 is about half as expensive as the general-purpose Algorithm 1.31 applied to H.

4.2 Scaling

Benner [26] proposed an algorithm for computing a diagonal matrix D so that the matrix

$$(D \oplus D^{-1})^{-1} \begin{bmatrix} A_{22} & G_{22} \\ Q_{22} & -A_{22}^T \end{bmatrix} (D \oplus D^{-1}) = \begin{bmatrix} D^{-1}A_{22}D & D^{-1}G_{22}D^{-1} \\ DQ_{22}D & -(D^{-1}A_{22}D)^T \end{bmatrix}$$

is nearly balanced in 1-norm. The algorithm is in the spirit of the Parlett-Reinsch algorithm for equilibrating the row and column norms of a general matrix, Algorithm 1.32. 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. As in the general case, this assumption is not satisfied by the output of Algorithm 4.33. A structure-preserving algorithm that yields irreducible Hamiltonian submatrices is presented in Section 6.1, Chapter 5.

Note that $\tilde{D} = I_{i_l-1} \oplus D \oplus I_{i_l-1} \oplus D^{-1}$ is symplectic. Thus, a similarity transformation involving $\tilde{D}$ preserves Hamiltonian structures. The following algorithm is basically HAPACK's implementation of Benner's algorithm.

Algorithm 4.34.

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 (4.57) 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 $= 0$
  converged $\leftarrow 1$
  FOR $j \leftarrow i_l, \dots, n$
    $c \leftarrow \sum_{i=i_l, i\neq j}^{n} (|a_{ij}| + |q_{ij}|)$,   $r \leftarrow \sum_{k=i_l, k\neq j}^{n} (|a_{jk}| + |g_{jk}|)$,   $\delta_q = |q_{jj}|$,   $\delta_g = |g_{jj}|$
    $s \leftarrow c + r$,   scal $\leftarrow 1$
    WHILE $((r + \delta_g/\beta)/\beta) \geq ((c + \delta_q \cdot \beta)\cdot \beta)$
      $c \leftarrow c\cdot\beta$,   $r \leftarrow r/\beta$,   $\delta_q \leftarrow \delta_q\cdot\beta^2$,   $\delta_g \leftarrow \delta_g/\beta^2$
      scal $\leftarrow$ scal $\cdot\, \beta$
    END WHILE
    WHILE $((r + \delta_g\cdot\beta)\cdot\beta) \leq ((c + \delta_q/\beta)/\beta)$
      $c \leftarrow c/\beta$,   $r \leftarrow r\cdot\beta$,   $\delta_q \leftarrow \delta_q/\beta^2$,   $\delta_g \leftarrow \delta_g\cdot\beta^2$
      scal $\leftarrow$ scal$/\beta$
    END WHILE
    % Balance if necessary.
    IF scal $\neq 1$ THEN
      converged $\leftarrow 0$,   $\tilde{d}_{jj} \leftarrow$ scal $\cdot\, \tilde{d}_{jj}$,   $\tilde{d}_{n+j,n+j} \leftarrow 1/$scal $\cdot\, \tilde{d}_{n+j,n+j}$
      $A(:,j) \leftarrow$ scal $\cdot\, A(:,j)$,       $A(j,:) \leftarrow 1/$scal $\cdot\, A(j,:)$
      $G(:,j) \leftarrow 1/$scal $\cdot\, G(:,j)$,   $G(j,:) \leftarrow 1/$scal $\cdot\, G(j,:)$
      $Q(:,j) \leftarrow$ scal $\cdot\, Q(:,j)$,       $Q(j,:) \leftarrow$ scal $\cdot\, Q(j,:)$
    END IF
  END FOR
END WHILE
```

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 $1 \leq i \leq n$, Algorithm 4.34 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 flops required by the general-purpose Algorithm 1.32 applied to H.

Both ingredients of symplectic balancing, Algorithms 4.33 and 4.34, 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 $\mathrm{scal}(j) \leq n$) 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 Algorithm 4.33. Otherwise, $\mathrm{scal}(j)$ contains $\tilde{d}_{jj}$, the jth diagonal entry of the diagonal matrix $\tilde{D}$ returned by Algorithm 4.34.

The backward transformation, i.e., multiplication with $(P\tilde{D})^{-1}$, is implemented in the HAPACK routine DHABAK. Slight modifications of Algorithms 4.33 and 4.34 can be used for balancing skew-Hamiltonian matrices, see [26]. These modified algorithms are implemented in the HAPACK subroutine DSHBAL.

Symplectic balancing a (skew-)Hamiltonian matrix has essentially the same positive effects that are described in Section, Chapter 1, for balancing a general matrix. Several numerical experiments confirming this statement can be found in [26] and Section 6, as well as Section 6.3 in Chapter 5.

5 Block Algorithms

We have seen in Section 5.2, Chapter 1, that the LAPACK subroutines for computing Hessenberg decompositions attain high efficiency by (implicitly) employing compact WY representations of the involved orthogonal transformations. A variant of this representation can be used to derive efficient block algorithms for computing orthogonal symplectic decompositions, such as the symplectic QR and URV decompositions.

5.1 A WY-like representation for products of elementary orthogonal symplectic matrices

We recall the reader that a $2n \times 2n$ elementary (orthogonal symplectic) matrix $E_j(x)$ takes the form

$$E_j(x) \equiv E_j(v, w, \beta, \gamma, \theta) := (H_j \oplus H_j)(v, \beta) \cdot G_{j,n+j}(\theta) \cdot (H_j \oplus H_j)(w, \gamma), \quad (4.58)$$

where $(H_j \oplus H_j)(v, \beta)$, $(H_j \oplus H_j)(v, \gamma)$ are direct sums of two identical $n \times n$ Householder matrices and $G_{j,n+j}(\theta)$ is a Givens rotation matrix. In order to develop block algorithms for orthogonal symplectic decompositions we have to derive a modified WY representation theorem for products of such elementary matrices. Of course, since $G_{j,n+j}$ and $(H_j \oplus H_j)$ can be written as the products of Householder matrices, we could use a standard WY representation to obtain such a representation. However, such an approach would ignore the structures in $G_{j,n+j}$, $(H_j \oplus H_j)$ and would consequently lead to considerably higher memory and run-time requirements. The following theorem presents a modified representation where these structures are exploited.

Theorem 4.35. *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 matrices $E_{j_i}(x_i)$ are defined as in (4.58) 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*

$$Q = \begin{bmatrix} I_n + WTW^T & WRSW^T \\ -WRSW^T & I_n + WTW^T \end{bmatrix}. \quad (4.59)$$

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.

Proof. We prove the representation (4.59) by induction over k. The case $k = 0$ is clear. Let Q be represented as in (4.59). Consider for $j := j_{k+1}$ the product

$$\tilde{Q} := Q \cdot E_j(v, w, \beta, \gamma, \theta) = Q \cdot (H_j \oplus H_j)(v, \beta) \cdot G_{j,n+j}(\theta) \cdot (H_j \oplus H_j)(w, \gamma).$$

We now show how to find a representation for $\tilde{Q}$. Similar to the construction of the compact WY representation, see Theorem 1.34 or [202], the first Householder matrix $H_j(v, \beta)$ is incorporated by updating

$$R_1 \leftarrow \begin{bmatrix} R_1 \\ 0 \end{bmatrix}, \; S_1 \leftarrow \begin{bmatrix} S_1 & -\beta S W^T v \end{bmatrix}, \tag{4.60}$$

$$T_{11} \leftarrow \begin{bmatrix} T_{11} & -\beta T_{1,:} W^T v \\ 0 & -\beta \end{bmatrix}, \; T_{i1} \leftarrow \begin{bmatrix} T_{i1} & -\beta T_{i,:} W^T v \end{bmatrix}, \tag{4.61}$$

$$T_{1l} \leftarrow \begin{bmatrix} T_{1l} \\ 0 \end{bmatrix}, \; W_1 \leftarrow \begin{bmatrix} W_1 & v \end{bmatrix}, \tag{4.62}$$

where $i, l \in \{2, 3\}$ and $T_{i,:}$ denotes the ith block row of T. By straightforward computation, it can be shown that the following update yields a representation for $Q \cdot (H_j \oplus H_j)(v, \beta) \cdot G_{j,n+j}(\theta)$,

$$R_2 \leftarrow \begin{bmatrix} R_2 & T_{2,:} W^T e_j \\ 0 & 1 \end{bmatrix}, \; R_i \leftarrow \begin{bmatrix} R_i & T_{i,:} W^T e_j \end{bmatrix}, \; S_2 \leftarrow \begin{bmatrix} S_2 & \bar{c} S_2 W^T \\ 0 & -\bar{s} \end{bmatrix},$$

$$S_i \leftarrow \begin{bmatrix} S_i \\ 0 \end{bmatrix}, \; T_{22} \leftarrow \begin{bmatrix} T_{22} & (\bar{s} R_2 S + \bar{c} T_{2,:}) W^T e_j \\ 0 & \bar{c} \end{bmatrix},$$

$$T_{i2} \leftarrow \begin{bmatrix} T_{i2} & (\bar{s} R_i S + \bar{c} T_{i,:}) W^T e_j \end{bmatrix}, \; T_{2i} \leftarrow \begin{bmatrix} T_{2i} \\ 0 \end{bmatrix}, \; W_2 \leftarrow \begin{bmatrix} W_2 & e_j \end{bmatrix},$$

where $\bar{c} = 1 - \cos\theta$, $\bar{s} = \sin\theta$ and $i, l \in \{1, 3\}$. The second Householder matrix $H_j(w, \gamma)$ is treated similar to (4.60)–(4.62). □

An inspection of the preceding proof reveals that the matrices R_3, S_1, T_{21}, T_{31} and T_{32} are actually strictly upper triangular and the matrix R_2 is unit upper triangular. If $j_i = i$ then the upper $k \times k$ blocks of W_1, W_3 consist of unit lower triangular matrices and W_2 contains the first k columns of the identity matrix. In this case a thorough implementation of the construction given in the proof of Theorem 4.35 requires $(4k - 2)kn + \frac{19}{3}k^3 + \frac{1}{2}k^2 + \mathcal{O}(k)$ flops.

The application of the WY-like representation (4.59) to a $2n \times q$ matrix requires $(16k(n-k) + 38k - 2)q$ flops using an implementation of the following algorithm which takes care of all the generic structures present in R, S, T and W.

Algorithm 4.36.

Input: *Matrices $A_1, A_2 \in \mathbb{R}^{n \times q}$; matrices $R \in \mathbb{R}^{3k \times k}$, $S \in \mathbb{R}^{k \times 3k}$, $T \in \mathbb{R}^{3k \times 3k}$, $W \in \mathbb{R}^{n \times 3k}$ representing the orthogonal symplectic matrix Q as described in Theorem 4.35 for $j_i = i$.*

Output: *The matrix $\begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$ is overwritten with $Q^T \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$.*

```
$V_1 = A_1^T W,\ V_2 = A_2^T W$
$Y_1 = WT^T - WS^T R^T$
$Y_2 = WT^T + WS^T R^T$
$A_1 \leftarrow A_1 + Y_1 V_1^T,\ A_2 \leftarrow A_2 + Y_2^T V_2^T$
```

This algorithm is implemented in the HAPACK routine `DLAESB`, using calls to the Level 3 BLAS `DGEMM` and `DTRMM`.

5.2 Block Symplectic QR Decomposition

Using the results of the previous section we can now easily derive a block oriented version of Algorithm 4.5 for computing the symplectic QR decomposition of a general $2m \times n$ matrix A.

Let us partition A into block columns

$$A = \begin{bmatrix} A_1 & A_2 & \dots & A_N \end{bmatrix},$$

For convenience only, we will assume that each A_i has n_b columns so that $n = N \cdot n_b$. Our block algorithm for the symplectic QR decomposition goes hand in hand with block algorithms for the standard QR decomposition [42]. The idea is as follows. At the beginning of step p $(1 \le p \le N)$ the matrix A has been overwritten with

$$Q_{j-1} \cdots Q_1 A = \begin{array}{c} \\ {\scriptstyle (p-1)n_b} \\ {\scriptstyle r} \\ {\scriptstyle (p-1)n_b} \\ {\scriptstyle r} \end{array} \begin{array}{c} \begin{array}{ccc} {\scriptstyle (p-1)n_b} & {\scriptstyle n_b} & {\scriptstyle q} \end{array} \\ \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \\ 0 & R_{42} & R_{43} \end{bmatrix} \end{array},$$

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 $\begin{bmatrix} R_{23} \\ R_{43} \end{bmatrix}$. The following algorithm is a formal description of this procedure.

Algorithm 4.37.

Input: *A matrix $A \in \mathbb{R}^{2m \times n}$ with $m \ge n$ 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 (4.5). In contrast to Algorithm 4.5 a block oriented method is used.*

```
$Q \leftarrow I_{2m}$
FOR $p = 1, \dots, N$
    Set $s = (p-1)n_b + 1$.
```

```
        Apply Algorithm 4.5 and the construction given in the proof of
        Theorem 4.35 to compute the WY-like representation (4.59) of an
        orthogonal symplectic matrix $Q_p$ so that
            $Q_p^T \left[ \begin{array}{c} A(s:m, s:s+n_b-1) \\ A(m+s:2m, s:s+n_b-1) \end{array} \right]$
        has the form (4.5).
        Update $\left[ \begin{smallmatrix} A(s:m,s+n_b:n) \\ A(m+s:2m,s+n_b:n) \end{smallmatrix} \right] \leftarrow Q_p^T \left[ \begin{smallmatrix} A(s:m,s+n_b:n) \\ A(m+s:2m,s+n_b:n) \end{smallmatrix} \right]$ using Alg. 4.36.
        Update $\left[ \begin{smallmatrix} Q(:,s:m) & Q(:,m+s:2m) \end{smallmatrix} \right] \leftarrow \left[ \begin{smallmatrix} Q(:,s:m) & Q(:,m+s:2m) \end{smallmatrix} \right] Q_p$ using Alg. 4.36.
    END FOR
```

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 4.5 requires $8(mn^2 - n^3/3) + \mathcal{O}(n^2)$ flops to $Q^T A$. Hence, Algorithm 4.37 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 DORGSB and DORGSQ for more details.

5.3 Block Symplectic URV Decomposition

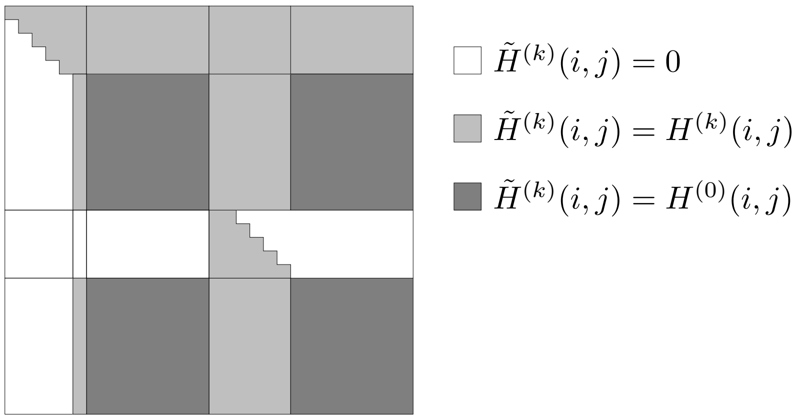
In the same manner WY-like representations allow us to develop block algorithms for virtually any kind of one-sided orthogonal symplectic decomposition. Some new difficulties arise when we consider two-sided decompositions. In this case and in contrast to the symplectic QR decomposition it is often impossible to reduce a subset of columns without touching other parts of the matrix. Hence, more effort is necessary to resolve the dependencies between the individual elementary transformations used to construct a two-sided decomposition. Let us illuminate this point with the symplectic URV decomposition as computed by Algorithm 4.31.

Assume that Algorithm 4.31 is stopped after $k < n$ loops. Denote the so far updated matrix by $H^{(k)}$ and partition

$$H^{(k)} = \left[\begin{array}{cc} A^{(k)} & G^{(k)} \\ Q^{(k)} & B^{(k)} \end{array} \right], \tag{4.63}$$

where each block is $n \times n$. According to the usual terminology for block algorithms we say that $H^{(k)}$ is *k-panel reduced*. The matrix $H^{(k)}$ emerged from $H^{(0)} = H$ after k elementary matrices have been applied to both sides of $H^{(0)}$. Applying Theorem 4.59 to these transformations and multiplying $H^{(0)}$ from both sides by the corresponding WY-like representations show that there exist $n \times 3k$ matrices $\tilde{U}, \tilde{V}, \tilde{X}_{\{A,B,G,Q\}}, \tilde{Y}_{\{A,B,G,Q\}}$ so that

$$H^{(k)} = \left[\begin{array}{cc} A^{(0)} + \tilde{U}\tilde{X}_A^T + \tilde{Y}_A\tilde{V}^T & G^{(0)} + \tilde{U}\tilde{X}_G^T + \tilde{Y}_G\tilde{V}^T \\ Q^{(0)} + \tilde{U}\tilde{X}_Q^T + \tilde{Y}_Q\tilde{V}^T & B^{(0)} + \tilde{U}\tilde{X}_B^T + \tilde{Y}_B\tilde{V}^T \end{array} \right]. \tag{4.64}$$



□ $\tilde{H}^{(k)}(i,j) = 0$
▨ $\tilde{H}^{(k)}(i,j) = H^{(k)}(i,j)$
■ $\tilde{H}^{(k)}(i,j) = H^{(0)}(i,j)$

Figure 4.3. *Structure of $\tilde{H}^{(k)}$ for $k = 5, n = 15$. White and pale-gray parts contain the reduced k-panel, these are elements of the matrix $H^{(k)}$. Dark-gray parts contain elements of the original matrix $H^{(0)}$.*

Clearly, the above representation of $H^{(k)}$ would directly lead to a block version of Algorithm 4.31. Unfortunately, things are not that simple because the definition of the elementary transformations used in Algorithm 4.31 requires that columns $1 : k$ and rows $n+1 : n+k$ are updated. Also, the first instruction in loop $k+1$ would require that the $(k+1)$th column of $H^{(k)}$ is known at this time. We therefore remove the parts from $\tilde{U}, \tilde{V}, \tilde{X}_{\{A,B,G,Q\}}$ and $\tilde{Y}_{\{A,B,G,Q\}}$ that correspond to these portions of the matrix $H^{(k)}$. In turn, the matrices $A^{(0)}, B^{(0)}, G^{(0)}, Q^{(0)}$ in (4.64) must be altered to compensate these removals. Let $\tilde{H}^{(k)}$ be equal to $H^{(0)}$ with columns $1 : k+1$, $n+1 : n+k+1$ and rows $1 : k$, $n+1 : n+k$ superseded by the corresponding entries of $H^{(k)}$ as illustrated in Figure 4.3. Furthermore, $\tilde{H}^{(k)}$ is partitioned into blocks $\tilde{A}^{(k)}, \tilde{B}^{(k)}, \tilde{G}^{(k)}$ and $\tilde{Q}^{(k)}$ similarly to (4.63).

Altogether, we consider the modified representation

$$H^{(k)} = \left[\begin{array}{cc} \tilde{A}^{(k)} + UX_A^T + Y_A V^T & \tilde{G}^{(k)} + UX_G^T + Y_G V^T \\ \tilde{Q}^{(k)} + UX_Q^T + Y_Q V^T & \tilde{B}^{(k)} + UX_B^T + Y_B V^T \end{array} \right], \tag{4.65}$$

where $U, V, X_{\{A,B,G,Q\}}, Y_{\{A,B,G,Q\}}$ have been reduced to $n \times 2k$ matrices.

We now show how to pass from (4.65) to an analogous representation for $H^{(k+1)}$. In the following algorithm the symbol '$\star$' denotes a placeholder which may take any value in the set $\{A, B, G, Q\}$.

Algorithm 4.38.

Input: *A k-panel reduced matrix $H^{(k)} \in \mathbb{R}^{2n \times 2n}$ represented as in (4.65).*

Output: *A representation of the form (4.65) for the $(k+1)$-panel reduced matrix $H^{(k+1)}$.*

```
% Incorporate transformations from the left.
Apply Algorithm 4.1 to compute $E_{k+1}(\tilde{H}^{(k)} e_{k+1}) = E_{k+1}(v, w, \beta, \gamma, \theta)$ and
update the $(k+1)$th column of $\tilde{H}^{(k)}$.
FOR EACH $\star \in \{A, B, G, Q\}$ DO
    $X_\star \leftarrow [X_\star, -\beta((\tilde{\star}^{(k)})^T v + X_\star U^T v + V Y_\star^T v)]$, $U \leftarrow [U, v]$
    $\tilde{\star}^{(k)}(k+1,:) \leftarrow \tilde{\star}^{(k)}(k+1,:) + X_\star(k+1,:) U^T + V(k+1,:) Y_\star^T$
    $X_\star(k+1,:) = 0$, $V(k+1,:) = 0$
END FOR
```

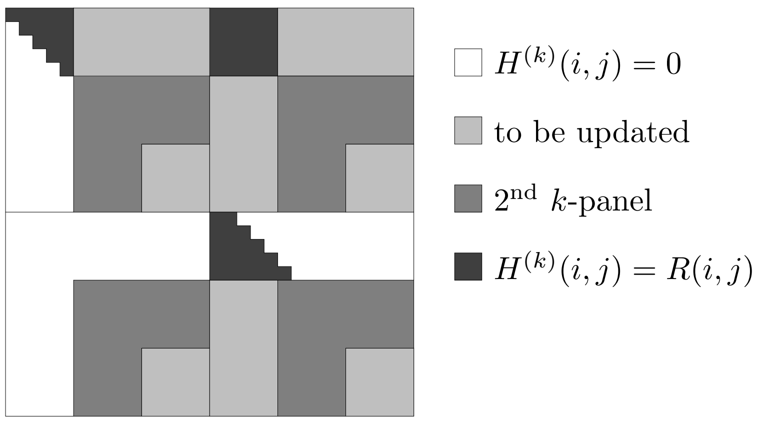


Figure 4.4. *Reduction of the second k-panel for $k = 5, n = 15$. White and black parts partly contain the first k-panel and are not affected by subsequent panel reductions. Dark-gray parts contain the second k-panel and pale-gray parts must be updated after the second panel has been reduced.*

```
$\tilde{H}^{(k)} \leftarrow G_{k+1}(\theta)\tilde{H}^{(k)}$
FOR EACH $\star \in \{A, B, G, Q\}$ DO
    $w(k+1) = 0$, $x_\star = -\gamma((\tilde{\star}^{(k)})^T w + X_\star U^T w + V Y_\star^T w)$
    $X_\star \leftarrow [X_\star, x_\star]$, $U \leftarrow [U, w]$
    $\tilde{\star}^{(k)}(k+1, :) \leftarrow \tilde{\star}^{(k)}(k+1, :) + x_\star^T$
END FOR
% Incorporate transformations from the right.
Apply Algorithm 4.1 to compute $E_{k+2}((\tilde{H}^{(k)})^T e_{n+k+1}) = E_{k+2}(v, w, \beta, \gamma, \theta)$
and update the $(n+k+1)$th row of $\tilde{H}^{(k)}$.
FOR EACH $\star \in \{A, B, G, Q\}$ DO
    $Y_\star \leftarrow [Y_\star, -\beta(\tilde{\star}^{(k)} v + U X_\star^T v + Y_\star V^T v)]$, $V \leftarrow [V, v]$
    $\tilde{\star}^{(k)}(:, k+2) \leftarrow \tilde{\star}^{(k)}(:, k+2) + X_\star U(k+2, :)^T + V Y_\star(k+2, :)^T$
    $U(k+2, :) = 0$, $Y_\star(k+2, :) = 0$
END FOR
$\tilde{H}^{(k)} \leftarrow \tilde{H}^{(k)} G_{k+2}(\theta)$
FOR EACH $\star \in \{A, B, G, Q\}$ DO
    $w(k+2) = 0$, $y_\star = -\gamma(\tilde{\star}^{(k)} w + U X_\star^T w + Y_\star V^T w)$
    $Y_\star \leftarrow [Y_\star, y_\star]$, $V \leftarrow [V, w]$
    $\tilde{\star}^{(k)}(:, k+2) \leftarrow \tilde{\star}^{(k)}(:, k+2) + y_\star$
END FOR
$\tilde{H}^{(k+1)} = \tilde{H}^{(k)}$
```

Subsequent application of Algorithm 4.38 yields representation (4.65) requiring

$$16 \cdot (2kn^2 + 7k^2 n - 13k^3/3) + 42kn + \mathcal{O}(k^2)$$

flops.

The rest of the story is easily told. Using (4.65) the matrix $H^{(k)}$ is computed via eight rank-$2k$ updates of order $n - k$. The next panel to be reduced resides in rows and columns $k+1 : 2k$, $n+k+1 : n+2k$ of $H^{(k)}$ as illustrated in Figure 4.4. Algorithm 4.38 is repeatedly applied to the matrix

$$\begin{bmatrix} A^{(k)}(k+1:n,k+1:n) & G^{(k)}(k+1:n,k+1:n) \\ Q^{(k)}(k+1:n,k+1:n) & B^{(k)}(k+1:n,k+1:n) \end{bmatrix}$$

to reduce its leading k-panel. Again, eight rank-$2k$ updates, now of order $n-2k$, yield $\star^{(2k)}(k+1:n,k+1:n)$ for $\star \in \{A,B,G,Q\}$. It remains to update rows $1:k$ and columns $n+1:n+k+1$ of $H^{(k)}$. This could be achieved by applying WY-like representations of the orthogonal symplectic transformations involved in the reduction of the second panel. In our implementation, however, we include these parts in Algorithm 4.38 so that the subsequent rank-$2k$ updates readily yield the matrix $H^{(2k)}$. For more details the reader is referred to the Fortran implementation of this algorithm, see Section 5 in Appendix B.

Assume that $n = N \cdot k$, then the procedure described above requires

$$80n^3/3 + 64n^3/N - 16n^3/N^2 + \mathcal{O}(n^2)$$

flops to compute the R-factor in the symplectic URV decomposition of a $2n \times 2n$ matrix. Since the unblocked version, Algorithm 4.31, requires $80n^3/3 + \mathcal{O}(n^2)$ flops for the same task, we see that blocking is more expensive by approximately a factor of $(1 + 2.4/N)$.

Remark 4.39. *Block algorithms for PVL decompositions of skew-Hamiltonian or Hamiltonian matrices can be derived in a similar way, see the HAPACK routines* `DHAPVB` *and* `DSHPVB`*. However, they are somewhat less relevant for this kind of decompositions. The only use of PVL decompositions of Hamiltonian matrices we are aware of is in the OSMARE algorithm [5]. However, with increasing matrix dimensions, OSMARE suffers from the poor convergence of the multishift QR algorithm explained in Section 5.4, Chapter 1. On the other hand, we have seen that the PVL decomposition of a skew-Hamiltonian matrix constitutes an important preprocessing step for eigenvalue computations. However, developing an efficient block algorithm for this reduction would require to have an efficient BLAS for skew-symmetric block updates of the form $C \leftarrow C + AB^T - BA^T$ handy. Unfortunately, such a subroutine is not yet defined in the BLAS standard.*

5.4 Numerical Stability

The derived block algorithms would be unedifying if they flawed the favorable error analysis of orthogonal decompositions. To show backward stability for the construction and the application of the WY-like representation in Section 5.1 we use techniques described in the book by Higham [122]. First, let us establish the following inequalities.

Lemma 4.40. *Let $\hat{R}, \hat{S}, \hat{T}$ and $\hat{W}$ be the computed factors of the block representation constructed as in the proof of Theorem 4.35 and set*

$$\hat{Q} = \begin{bmatrix} I + \hat{W}\hat{T}\hat{W}^T & \hat{W}\hat{R}\hat{S}\hat{W}^T \\ -\hat{W}\hat{R}\hat{S}\hat{W}^T & I + \hat{W}\hat{T}\hat{W}^T \end{bmatrix}.$$

Then

$$\|\hat{Q}^T\hat{Q} - I\|_2 \leq \mathbf{u} d_Q, \tag{4.66}$$

$$\|\hat{R}\|_2 \leq d_R, \quad \|\hat{S}\|_2 \leq d_S, \quad \|\hat{T}\|_2 \leq d_T, \quad \|\hat{W}\|_2 \leq d_W, \tag{4.67}$$

for modest constants d_Q, d_R, d_S, d_T and d_W.

Proof. Inequality (4.66) is shown by induction. For the evaluation of $\hat{Q}_1$ with $Q_1 := Q(H_j \oplus H_j)(v, \beta)$ we may assume that $|\hat{v} - v| \leq \gamma_{cn}|v|$, $\|v\|_2 = \sqrt{2}$, and $\beta = 1$, where the quantity γ_{cn} denotes $\mathbf{u}cn/(1-\mathbf{u}cn)$ with a small constant $c > 1$. Then, using formulas (4.60)–(4.61) and Lemma 18.3 of [122] on the backward error in products of Householder matrices, it follows that

$$\|\hat{Q}_1^T \hat{Q}_1 - I\|_2 \leq \mathbf{u}d_Q + \sqrt{n}\gamma_{cn} =: \mathbf{u}d_Q'. \tag{4.68}$$

Similarly, $|\cos\hat{\theta} - \cos\theta| + |\sin\hat{\theta} - \sin\theta| \leq \gamma_{c'}$ with a small constant $c' > 0$. The factored matrix $\hat{Q}_2$ with $Q_2 := Q_1 G_j(\theta)$ satisfies

$$\|\hat{Q}_2^T \hat{Q}_2 - I\|_2 \leq \mathbf{u}d_Q' + \sqrt{n}\gamma_c.$$

Repeated application of (4.68) to $Q_3 := Q_2 H_j(w, \gamma)$ proves (4.66). The inequalities in (4.67) readily follow from the construction of R, S, T and W. $\square$

Inequality (4.66) implies that the matrix $\hat{Q}$ is close to orthogonality. Together with the block structure of $\hat{Q}$ this implies that $\hat{Q}$ is actually close to an orthogonal symplectic matrix, see Lemma 4.6.

Lemma 4.40 also shows that under the usual assumptions on matrix multiplication, the forward errors of the computed matrices $\hat{B}_1$ and $\hat{B}_2$ in Algorithm 4.36 satisfy

$$\begin{aligned} \left\|\hat{B}_1 - B_1\right\|_2 &\leq \mathbf{u}\|A_1\|_2 + \mathbf{u}d_W^2[c_T(k,n)\|A_1\|_2 + c_R(k,n)\|A_2\|_2], \\ \left\|\hat{B}_2 - B_2\right\|_2 &\leq \mathbf{u}\|A_2\|_2 + \mathbf{u}d_W^2[c_T(k,n)\|A_2\|_2 + c_R(k,n)\|A_1\|_2], \end{aligned}$$

where

$$c_T(k,n) := d_T(18k^2 + n^2 + 2), \quad c_R(k,n) := d_R d_S(19k^2 + n^2 + 2).$$

Lemma 4.6 together with inequality (4.66) imply the existence of an orthogonal symplectic matrix U so that $\hat{Q} = U + \triangle U$ with $\|\triangle U\|_2 \leq \mathbf{u}d_Q$. This enables us to bound the backward error of Algorithm 4.36,

$$\begin{bmatrix} \hat{B}_1 \\ \hat{B}_2 \end{bmatrix} = U \begin{bmatrix} \hat{A}_1 + \triangle A_1 \\ \hat{A}_2 + \triangle A_2 \end{bmatrix},$$

where

$$\left\| \begin{bmatrix} \triangle A_1 \\ \triangle A_2 \end{bmatrix} \right\|_2 \leq \sqrt{2}\mathbf{u}\left[1 + d_Q + d_W^2(c_T(k,n) + c_R(k,n))\right] \left\| \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \right\|_2.$$

This immediately verifies numerical backward stability for symplectic QR decompositions constructed as described in Algorithm 4.37. The analysis of the block algorithm for symplectic URV decompositions presented in Section 5.3 is complicated by the fact that parts from the WY-like representations for the left and right elementary transformations are removed to keep the k-panel updated. However, it can be expected that these removals do not introduce numerical instabilities in the symplectic URV decomposition.

5.5 Numerical Results

The described block algorithms are implemented in HAPACK. To demonstrate the efficiency of these implementations we present numerical examples run on an Origin2000 computer equipped with 400MHz IP27 R12000 processors and sixteen gigabytes of memory. The implementations were compiled with version 7.30 of the MIPSpro Fortran 77 compiler with options `-64 TARG:platform=ip27 -Ofa st=ip27 -LNO`. The programs called optimized BLAS and LAPACK subroutines from the SGI/Cray Scientific Library version 1.2.0.0. Timings were carried out using matrices with pseudorandom entries uniformly distributed in the interval $[-1, 1]$. Unblocked code was used for all subproblems with column or row dimension smaller than 65 ($n_x = 64$). Each matrix was stored in an array with leading dimension slightly larger than the number of rows to avoid unnecessary cache conflicts.

`DGESQB`		$n = 128$		$n = 256$		$n = 512$		$n = 1024$	
m	n_b	R	Q	R	Q	R	Q	R	Q
128	1	0.10	0.10	0.26	0.10	0.75	0.10	2.71	0.10
128	8	*0.08*	*0.08*	*0.21*	*0.08*	*0.50*	*0.08*	1.54	*0.08*
128	16	0.09	*0.08*	0.22	*0.08*	0.51	*0.08*	1.52	*0.08*
128	24	0.10	0.09	0.23	0.09	0.52	0.09	*1.50*	0.09
128	32	0.10	0.10	0.25	0.10	0.56	0.10	1.65	0.10
256	1	0.24	0.24	0.88	0.88	3.32	0.88	9.71	0.90
256	8	*0.18*	*0.18*	*0.54*	0.54	1.46	0.54	3.87	0.54
256	16	*0.18*	*0.18*	*0.54*	*0.50*	*1.42*	*0.50*	*3.63*	*0.51*
256	24	0.20	0.19	0.57	0.53	1.46	0.53	3.72	0.53
256	32	0.20	0.20	0.61	0.56	1.52	0.56	3.93	0.56
512	1	0.59	0.60	3.25	3.27	13.10	13.16	35.16	13.16
512	16	0.40	*0.39*	*1.31*	1.28	4.33	4.20	11.34	4.15
512	24	*0.39*	*0.39*	*1.31*	*1.27*	4.26	4.11	11.04	4.07
512	32	0.42	0.41	*1.31*	*1.27*	*4.17*	*3.96*	*10.78*	*3.94*
512	48	0.46	0.45	1.40	1.35	4.26	4.06	11.05	4.04
1024	1	1.86	1.89	9.15	9.20	34.99	35.11	113.29	113.58
1024	16	0.87	0.88	2.94	2.92	10.43	10.21	33.85	33.17
1024	24	*0.85*	*0.84*	2.89	2.85	9.84	9.71	31.90	31.06
1024	32	0.89	0.89	*2.81*	*2.77*	*9.29*	*9.13*	29.68	28.65
1024	48	0.97	0.96	2.95	2.92	9.34	9.15	*29.17*	*27.93*

Table 4.1. *Performance results in seconds for the symplectic QR decomposition of an $m \times n$ matrix.*

Table 4.1 shows the result for `DGESQB`, an implementation of Algorithm 4.37. Rows with block size $n_b = 1$ correspond to the unblocked variant of this algorithm. Columns with heading R show timings when only the reduced matrix $R = Q^T A$ was computed. Additional times necessary to generate the orthogonal symplectic factor Q are displayed in columns with heading Q. The results show that the block algorithm outperforms the unblocked one for all chosen matrix dimensions under the assumption that a suitable value for the block size n_b has been used. The best improvement has been obtained for $m = 1024, n = 1024$ where the block algorithm saved 74.8% of the execution time for the computation of both factors.

DGESUB				
n	n_b	R	U	V
128	1	0.53	0.11	0.11
128	8	*0.48*	*0.08*	*0.09*
128	16	0.52	*0.08*	*0.09*
256	1	6.91	0.87	0.89
256	8	*4.74*	*0.50*	*0.52*
256	16	5.12	*0.50*	0.53
256	32	5.82	0.55	0.58
512	1	66.79	13.04	12.90
512	8	42.17	4.82	5.15
512	16	*42.05*	4.07	4.34
512	32	44.02	*3.88*	*4.11*
1024	1	563.16	113.40	114.02
1024	16	377.55	32.52	33.42
1024	32	*318.84*	*28.13*	*29.29*
1024	64	350.11	28.98	30.32

Table 4.2. *Performance results in seconds for the symplectic URV decomposition of an $2n \times 2n$ matrix H.*

The results for DGESUB, an implementation of the block algorithm described in Section 5.3, are displayed in Table 4.2. Again, the column with heading R refers to timings when only the reduced matrix $R = U^T HV$ was computed. The additional times for generating the orthogonal symplectic factors U and V are displayed in columns four and five, respectively. Albeit not so dramatic as for the symplectic QR decomposition the results show considerable improvements when the matrix order is sufficiently large. At its best, the block algorithm saved 47.6% of the execution time when the complete symplectic URV decomposition of a 2048×2048 ($n = 1024$) matrix was computed.

The accuracy of the block algorithms has been tested for various random matrices as well as Hamiltonian matrices obtained from the Riccati benchmark collections [1, 35]. We measured orthogonality of the factors Q, U, V and the relative residuals $\|QR - A\|_F / \|A\|_F$, $\|URV^T - H\|_F / \|H\|_F$. The results for the standard algorithms and the new block algorithms are qualitatively the same.

6 Numerical Experiments

To give a flavour of the numerical behaviour 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 [35] and contains several examples of continuous-time algebraic Riccati equations (CAREs) of the form

$$Q + A^T X + XA - XGX = 0. \tag{4.69}$$

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 2 in Appendix A and [25, 173, 205].

6.1 Accuracy of Eigenvalues

We compared the accuracy of the eigenvalues computed by the following implementations:

QR – the Matlab command `eig(H,'nobalance')`, which facilitates the LAPACK implementation of the QR algorithm;

SQRED – a straight Matlab implementation of the square reduced method proposed in [27];

UrvPsd – the Matlab command `haeig(H,'nobalance')`, which facilitates the HAPACK implementation `DHAESU` of the symplectic URV/periodic Schur decomposition, see also Section 5 in Appendix B;

HamQR – a straight Matlab implementation of the explicit Hamiltonian QR algorithm described in Section 3.4.

The maximal relative error in the computed eigenvalues $|\hat{\lambda} - \lambda|/|\lambda|$ was measured using the "exact" eigenvalue λ obtained from the QR algorithm in quadruple precision. Numerical results obtained without any prior balancing are listed in Table 4.3. Those, where prior (symplectic) balancing resulted in a considerable damping of the relative error, are listed in Table 4.4.

Example	QR	SqRed	UrvPsd	HamQR
1.1	2.4×10^{-08}	0	0	7.4×10^{-09}
1.2	1.8×10^{-14}	3.5×10^{-15}	3.9×10^{-15}	3.1×10^{-15}
1.3	6.7×10^{-15}	6.2×10^{-15}	5.8×10^{-15}	5.5×10^{-15}
1.4	4.1×10^{-14}	7.7×10^{-14}	4.7×10^{-14}	4.2×10^{-14}
1.5	4.5×10^{-14}	2.3×10^{-12}	8.0×10^{-15}	8.5×10^{-14}
1.6	6.6×10^{-11}	4.7×10^{-08}	7.9×10^{-11}	3.7×10^{-05}
2.1	1.1×10^{-15}	0	6.6×10^{-16}	2.2×10^{-16}
2.2	1.2×10^{-09}	1.0×10^{-08}	1.2×10^{-09}	1.2×10^{-09}
2.3	1.4×10^{-11}	1.1×10^{-16}	1.1×10^{-16}	3.5×10^{-14}
2.4	6.0×10^{-04}	1.0×10^{-02}	3.7×10^{-11}	2.4×10^{-04}
2.5	4.0×10^{-08}	5.6×10^{-16}	2.9×10^{-08}	1.5×10^{-08}
2.6	6.2×10^{-16}	7.8×10^{-16}	4.7×10^{-16}	1.5×10^{-15}
2.7	8.1×10^{-04}	7.8×10^{-10}	2.4×10^{-05}	7.0×10^{-05}
2.8	1.0×10^{-12}	1.0×10^{-12}	1.0×10^{-12}	1.0×10^{-12}

Table 4.3. *Relative errors in the computed eigenvalues of Hamiltonian matrices from [1]* without *prior balancing.*

One can observe some interesting anomalies in the presented figures:

Example	QR	SqRed	UrvPsd	HamQR
1.6	2.7×10^{-13}	2.0×10^{-09}	1.6×10^{-13}	2.6×10^{-12}
2.1	0	0	2.2×10^{-16}	0
2.3	9.2×10^{-16}	1.1×10^{-16}	1.1×10^{-16}	1.1×10^{-16}
2.7	5.5×10^{-11}	1.8×10^{-07}	3.1×10^{-10}	1.8×10^{-10}

Table 4.4. *Maximal relative errors in the computed eigenvalues of Hamiltonian matrices from [1]* with *prior (symplectic) balancing.*

- Although the explicit Hamiltonian QR algorithm enjoys strong backward stability, it sometimes computes eigenvalues with much poorer accuracy than QR or UrvPsd. This effect is particularly drastic in Example 2.4. We believe that this can be attributed to the fact that the explicit Hamiltonian QR algorithm employs a norm-wise deflation criterion while QR and UrvPsd both employ less generous neighbour-wise deflation criteria, see also the discussion in Section 3.4, Chapter 1.
- The square reduced method yields sometimes (Examples 2.5 and 2.7 in Table 4.3, Example 2.5 in Table 4.4) the highest accuracy among all algorithms. It is not clear whether this effect must be attributed to fortunate circumstances or is caused by a special property of the square reduced method.
- Example 1.5 in Table 4.3 as well as Examples 1.6 and 2.7 in Table 4.4 seem to be affected by the numerical instability of the square reduced method. This is confirmed by the following table, which lists the norm of the (balanced) Hamiltonian matrix, the affected eigenvalue λ, its absolute condition number $c(\lambda)$ and the relative error $|\hat{\lambda}-\lambda|/|\lambda|$ obtained with the square reduced method.

Example	$\|H\|_2$	λ	$c(\lambda)$	$\|\hat{\lambda}-\lambda\|/\|\lambda\|$
1.5	1.9×10^{2}	0.3366	1.2	1.1×10^{-12}
1.6	9.4×10^{2}	0.1824	1.3	2.0×10^{-09}
2.7	2.6	1.414×10^{-07}	3.5×10^{06}	1.8×10^{-07}

Several numerical examples comparing the described methods for computing the solution of CARE can be found in [37, 26].

6.2 Computing Times

To compare computing times we computed the eigenvalues and/or the stable invariant subspace of the Hamiltonian matrix (1.68), which corresponds to the solution of the optimal control problem described in Section 2, Chapter 1. The following Fortran 77 implementations have been used:

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;

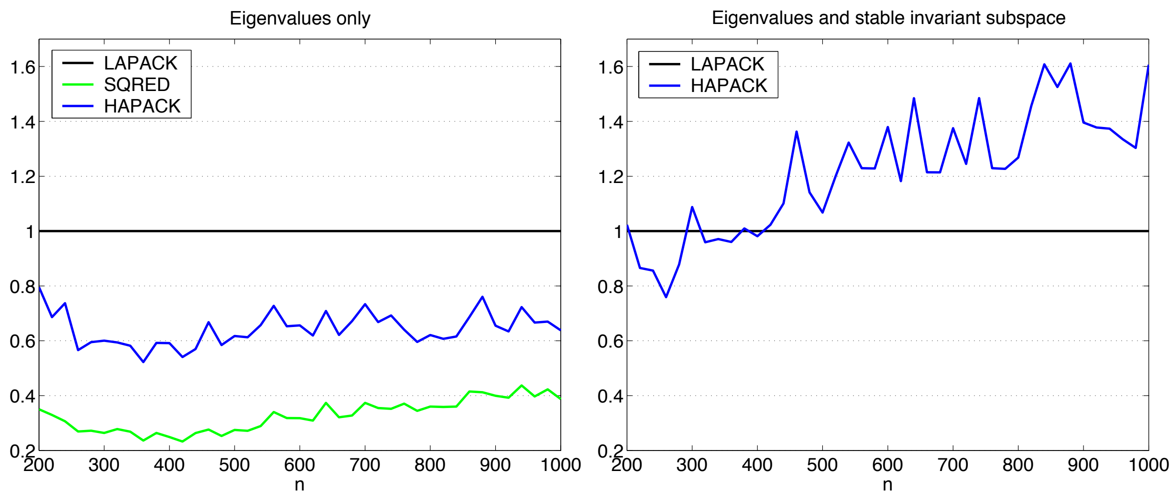


Figure 4.5. *Computational times of SQRED and HAPACK for computing eigenvalues and invariant subspaces of* $2n \times 2n$ *Hamiltonian matrices* ($n = 200, 220, \dots, 1000$)*, relative to the computational time of LAPACK.*

SQRED – routines `DHASRD` and `DHAEVS` from [27] 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 4.5. 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.

Chapter 5

Krylov-Schur Algorithms

So far, all discussed algorithms for computing eigenvalues and invariant subspaces are based on orthogonal transformations and are suitable for small to medium-sized, dense matrices. Nevertheless, these algorithms play an important role in methods designed for the eigenvalue computation of large and sparse matrices, such as Arnoldi, Lanczos or Jacobi-Davidson methods; see, e.g., the eigenvalue templates book [15] for a comprehensive overview.

In this chapter, we focus on a descendant of the Arnoldi method, the recently introduced Krylov-Schur algorithm by Stewart [225]. 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 one of 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 algorithms described in previous chapters. Furthermore, it can be easily adapted to periodic and (skew-)Hamiltonian eigenvalue problems as will be shown below.

This chapter is organized as follows. In the first section, we briefly review Krylov subspaces, their properties and the basic Arnoldi method. A more detailed account of this subject can be found, e.g., in the monographs [201, 223]. Section 2 describes the Krylov-Schur algorithm as introduced by Stewart. In Section 3, we show that replacing (reordered) Schur decompositions in this algorithm by periodic (reordered) Schur decompositions leads to a product Krylov-Schur algorithm suitable to address eigenvalue problems associated with products of large and sparse matrices or, equivalently, associated with large and sparse block cyclic matrices. Based on work by Mehrmann and Watkins [175], we develop structure-preserving variants of the Krylov-Schur algorithm for skew-Hamiltonian and Hamiltonian matrices in Section 4. Balancing sparse matrices for eigenvalue computations has been a subject of recent interest. In Section 5, we summarize the work of Chen and Demmel [72] in this area, which is in the subsequent section modified for the symplectic balancing of sparse Hamiltonian matrices.

Contributions in this chapter

This chapter contains the following new contributions to the world of Krylov subspace methods:

1. a special-purpose Krylov-Schur algorithm for products of matrices, which can produce much more accurate results than a general-purpose Krylov-Schur algorithm, see Section 3;

2. a two-sided Krylov-Schur algorithm for Hamiltonian matrices, which is numerically backward stable and preserves eigenvalue pairings, see Section 4;

3. a structure-preserving irreducible form for Hamiltonian matrices, which can be used for balancing but might as well be of independent interest, see Section 6.1;

4. a slight modification of a Krylov-based balancing algorithm by Chen and Demmel [72] yielding a symplectic variant for Hamiltonian matrices, see Section 6.2.

The work on balancing sparse Hamiltonian matrices is joint work with Peter Benner, TU Chemnitz, and has been accepted for publication [32].

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 [201].

1.1 Krylov Subspaces

First, let us define a Krylov subspace and recall some of its elementary properties.

Definition 5.1. *Let $A \in \mathbb{R}^{n\times n}$ and $u \in \mathbb{R}^n$ with $u \neq 0$, then*

$$K_k(A,u) = \left[\begin{array}{ccccc} u & Au & A^2u & \dots & A^{k-1}u \end{array}\right]$$

is called the kth Krylov matrix *associated with A and u. The corresponding subspace*

$$\mathcal{K}_k(A,u) = \mathrm{range}(K_k(A,u)) = \mathrm{span}\{u, Au, A^2u, \dots, A^{k-1}u\}$$

is called the kth Krylov subspace *associated with A and u.*

Lemma 5.2. *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. $\mathcal{K}_l(A,u) = \mathcal{K}_{l+1}(A,u) \quad \Rightarrow \quad \mathcal{K}_l(A,u) = \mathcal{K}_k(A,u), \quad \forall k \geq l.$

The last item in this lemma covers an exceptional situation. The relation $\mathcal{K}_{l+1}(A,u) = \mathcal{K}_l(A,u)$ 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 5.3 ([223, Ex. 4.3.1]). *Let* $A = \mathrm{diag}(1, 0.95, 0.95^2, \dots,)$ *and let* u *be a random vector of length* n. *The solid blue line in Figure 5.1 shows the tangent*

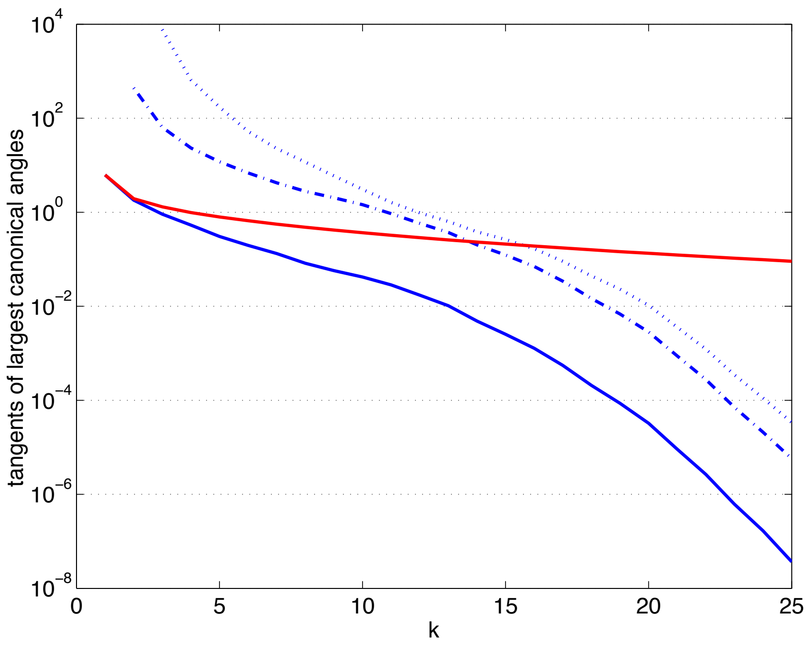


Figure 5.1. *Approximations of eigenvectors and invariant subspaces by Krylov subspaces.*

of the angle between e_1, *the eigenvector belonging to the dominant eigenvalue* 1, *and* $\mathcal{K}_k(A,u)$ *for* $k = 1,\dots,25$. *The dash-dotted line shows the tangent of the largest canonical angle between* $\mathrm{span}\{e_1, e_2\}$, *the invariant subspace belonging to* $\{1, 0.95\}$, *and* $\mathcal{K}_k(A,u)$, *while the dotted line shows the corresponding figures for* $\mathrm{span}\{e_1, e_2, e_3\}$. *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 red curve.*

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 [187] and the references therein.

Theorem 5.4 ([223, Thm. 4.3.8]). *Let* $A \in \mathbb{R}^{n \times n}$ *be* symmetric *and let*

$x_1, \dots, x_n$ 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

$$\tan \theta_1(x_1, \mathcal{K}_k(A,u)) \leq \frac{\tan \angle(x_1, u)}{(1 + 2\sqrt{\eta + \eta^2})^{k-1} + (1 + 2\sqrt{\eta + \eta^2})^{1-k}}. \tag{5.1}$$

For large k, the bound (5.1) 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

$$\tan \theta_1(x_2, \mathcal{K}_k(A,u)) \lesssim \frac{\tan \angle(x_2, u)}{(1 + 2\sqrt{\eta_2 + \eta_2^2})^{k-2} + (1 + 2\sqrt{\eta_2 + \eta_2^2})^{2-k}}.$$

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. [199]. For more recent work in this area, see [23, 24].

1.2 The Arnoldi Method

An explicit Krylov basis of the form

$$K_k(A,u) = \left[\begin{array}{cccccc} u & Au & A^2u & \dots & A^{k-1}u \end{array} \right]$$

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 method which implicitly constructs an orthonormal basis for $\mathcal{K}_k(A,u)$. The Arnoldi method is such a method 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 5.5. *Let the columns of*

$$U_{k+1} = \left[\begin{array}{cccc} u_1 & u_2 & \dots & u_{k+1} \end{array}\right] \in \mathbb{R}^{n\times(k+1)}$$

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

$$AU_k = U_{k+1}\hat{H}_k. \tag{5.2}$$

Conversely, a matrix U_{k+1} with orthonormal columns satisfies a relation of the form (5.2) 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. [223, Thm. 5.1.1]. Nevertheless, a proof is provided as it reveals some useful relationships between Krylov matrices, QR and Arnoldi decompositions.

Let us partition

$$\left[\begin{array}{cc} K_k(A,u_1) & A^k u_1 \end{array}\right] = \left[\begin{array}{cc} U_k & u_{k+1} \end{array}\right] \left[\begin{array}{cc} R_k & r_{k+1} \\ 0 & r_{k+1,k+1}, \end{array}\right],$$

then $U_k R_k$ is a QR decomposition of $K_k(A, u_1)$. Setting $S_k = R_k^{-1}$, we obtain

$$\begin{aligned} AU_k = AK_k(A,u_1)S_k &= K_{k+1}(A,u_1)\left[\begin{array}{c} 0 \\ S_k \end{array}\right] \\ &= U_{k+1}R_{k+1}\left[\begin{array}{c} 0 \\ S_k \end{array}\right] = U_{k+1}\hat{H}_k, \end{aligned}$$

where

$$\hat{H}_k = R_{k+1}\left[\begin{array}{c} 0 \\ S_k \end{array}\right].$$

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 (5.2) 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

$$\hat{H}_k = \left[\begin{array}{cc} \hat{H}_{k-1} & h_k \\ 0 & h_{k+1,k} \end{array}\right],$$

then (5.2) 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 U_{k+1} form an orthonormal basis for $K_{k+1}(A, u_1)$. □

This theorem justifies the following definition.

Definition 5.6. *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*

$$AU_k = U_{k+1}\hat{H}_k, \tag{5.3}$$

then (5.3) is called an (unreduced) Arnoldi decomposition of order k*.*

By a suitable partition of $\hat{H}_k$ we can rewrite (5.3) as follows:

$$AU_k = [\ U_k,\ u_{k+1}\ ] \begin{bmatrix} H_k \\ h_{k+1,k}e_k^T \end{bmatrix} = U_kH_k + h_{k+1,k}u_{k+1}e_k^T. \tag{5.4}$$

This shows that U_k satisfies the "invariant subspace relation" $AU_k = U_kH_k$ 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 5.7. *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* $H_k = U_k^T AU_k$ *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_kw *is called a* Ritz vector *belonging to* λ*.*

Given an Arnoldi decomposition of the form (5.4), 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 2 in Chapter 1, 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_kw belonging to a Ritz value λ with $\|w\|_2 = 1$ may correspond to a backward error much smaller than $|h_{k+1,k}|$:

$$[A - (h_{k+1,k}w_k)u_{k+1}(U_kw)^T] \cdot U_kw = \lambda \cdot U_kw,$$

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_kw so that the backward error $\|(h_{k+1,k}w_k)u_{k+1}(U_kw)^T\|_2 = |h_{k+1,k}w_k|$ converges to zero, see [223, Sec. 4.4.2].

The Basic Algorithm

The Arnoldi decomposition (5.2) almost immediately leads to the following algorithm for its computation.

Algorithm 5.8 (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* $k \le n$*.*

Output: *A matrix* $U_{k+1} = [u_1, \dots, u_{k+1}] \in \mathbb{R}^{n\times(k+1)}$ *with orthonormal columns and an upper Hessenberg matrix* $\hat{H}_k \in \mathbb{R}^{(k+1)\times k}$*, defining an Arnoldi decomposition (5.2) of* k*th order.*

```
$\hat{H}_0 \leftarrow []$
FOR $j \leftarrow 1, 2, \dots, k$
    $h_j \leftarrow U_j^T A u_j$
    $v \leftarrow A u_j - U_j h_j$
    $h_{j+1,j} \leftarrow \|v\|_2$
    $u_{j+1} \leftarrow v / h_{j+1,j}$
    $\hat{H}_j \leftarrow \begin{bmatrix} \hat{H}_{j-1} & h_j \\ 0 & h_{j+1,j} \end{bmatrix}$
END FOR
```

Several remarks concerning the implementation of this algorithm are necessary:

1. The algorithm can be implemented so that exactly one $n \times n$ matrix-vector multiplication is needed per step. The computational cost of Algorithm 5.8 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-by-$(k+1)$ matrix U_{k+1} makes Algorithm 5.8 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.

3. The algorithm can be seen as an implicit implementation of Gram-Schmidt orthonormalization. As such, it inherits the numerical instabilities of this orthonormalization method, i.e., the orthogonality of the columns of U_k can be severely affected in the presence of roundoff errors, see [44, 187]. 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 [76, 187]. Another stable alternative is based on Householder matrices [249].

4. If not the extremal eigenvalues but eigenvalues close to a given value σ are desired, then Algorithm 5.8 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 [17].

2 Restarting and the Krylov-Schur Algorithm

One of the drawbacks of Algorithm 5.8 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 Ritz values. Also, the cost for computing the Ritz values of the $k \times k$ matrix H_k grows cubically with k.

2.1 Restarting an Arnoldi Decomposition

A way out of this dilemma has been suggested by Saad [200] based on earlier work by Manteuffel [169] for the iterative solution of linear systems. Given an Arnoldi decomposition of order m,

$$AU_m = [\ U_m,\ u_{m+1}\ ] \begin{bmatrix} H_m \\ h_{m+1,m}e_m^T \end{bmatrix}, \tag{5.5}$$

Saad proposes to choose a so called *filter polynomial* ψ, based on information contained in $\lambda(H_m)$, and 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 [213] nicely 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 [212] 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 (5.5) of order m to an unreduced Arnoldi decomposition of order $k < m$,

$$A\tilde{U}_k = [\ \tilde{U}_k,\ \tilde{u}_{k+1}\ ] \begin{bmatrix} \tilde{H}_k \\ \tilde{h}_{k+1,k}e_k^T \end{bmatrix}, \tag{5.6}$$

so that

$$\operatorname{span}(\tilde{U}_k) = \mathcal{K}_k(A, \psi(A)u_1).$$

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 [212] and to the documentation of the software package ARPACK [161], which provides a Fortran 77 implementation of the implicitly restarted Arnoldi algorithm. Moreover, ARPACK facilitates similar techniques, developed by Lehoucq and Sorensen [160], 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.

2.2 The Krylov Decomposition

Stewart [225] 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 [188], is well demonstrated by the following example.

Example 5.9 ([188, Ex.2.1]). *Consider an Arnoldi decomposition*

$$A[e_1, \dots, e_6] = [e_1, \dots, e_6]H_6 + h_{7,6}e_7e_6^T$$

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 $\hat{H}_6 = fl(Q^T H_6 Q)$ 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 $[e_1, \dots, e_6]Q$. Theoretically, the Ritz vector x belonging to λ should have no components in $\operatorname{span}(\tilde{U}_5)$, *i.e.,*

$$\theta_1(\operatorname{span}(x), \operatorname{span}(\tilde{U}_5)^\perp) = 0$$

Again, this relationship is violated in finite precision arithmetic, the computed matrix $\hat{U}_5$ satisfies

$$\theta_1(\operatorname{span}(x), \operatorname{span}(\hat{U}_5)^\perp) \approx 0.0024.$$

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 [225]. It consists of relaxing the definition of an Arnoldi decomposition and using the eigenvalue reordering techniques described in Section 7, Chapter 1, for the restarting, locking and purging operations. The relaxed definition reads as follows.

Definition 5.10. *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*

$$AU_k = U_k B_k + u_{k+1} b_{k+1}^T, \tag{5.7}$$

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 5.11 ([225]). *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 $\hat{B}_k = \left[\begin{smallmatrix} B_k \\ b_{k+1}^T \end{smallmatrix}\right]$, 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 3.2 in Chapter 1). Use Algorithm 1.23 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

$$A\tilde{U}_k = \tilde{U}_{k+1}\left[\begin{array}{c} H_k \\ h_{k+1,k}e_k^T \end{array}\right],$$

which concludes the proof. $\square$

Particularly useful are Krylov decompositions having the factor B_k in real Schur form. Such a decomposition will be called *Krylov-Schur decomposition.*

2.3 Restarting a Krylov Decomposition

Given a Krylov decomposition of order m,

$$AU_m = U_{m+1}\left[\begin{array}{c} B_m \\ b_{m+1}^T \end{array}\right], \tag{5.8}$$

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 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:

$$A\tilde{U}_k = [\tilde{U}_k, \tilde{u}_{k+1}]\left[\begin{array}{c} T_w \\ b_w^T \end{array}\right]. \tag{5.9}$$

The described process is depicted in Figure 5.2. It can be shown that the transition from the extended Krylov decomposition (5.8) to the reduced Krylov

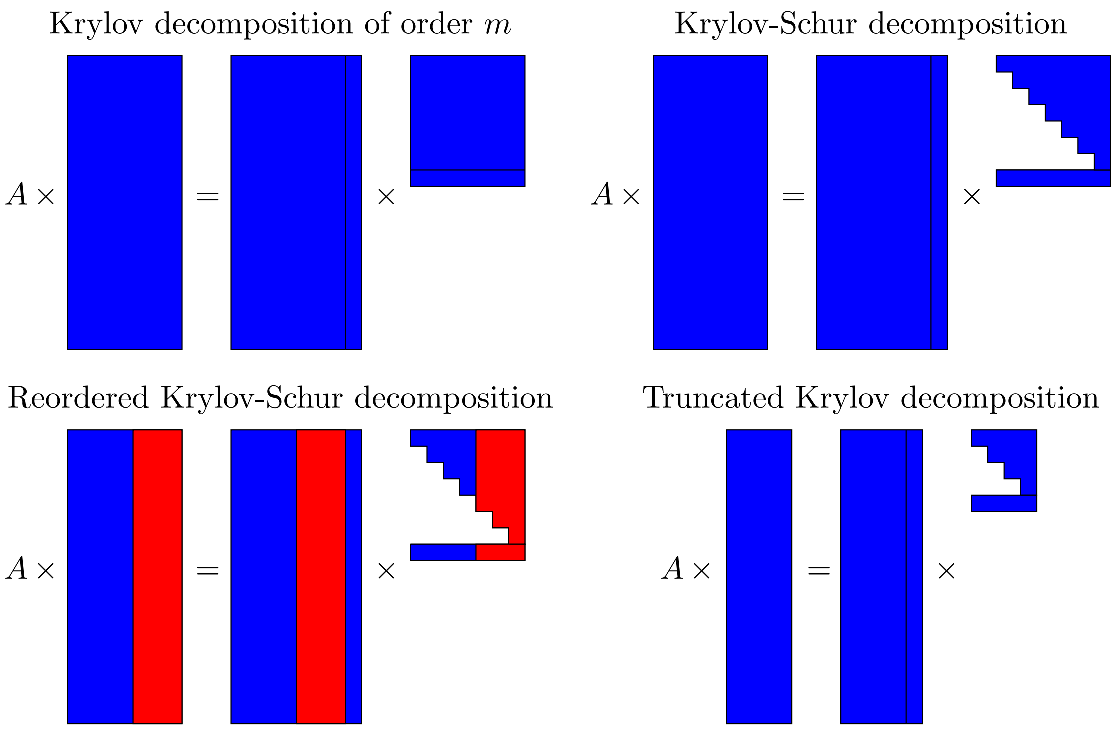

Krylov decomposition of order m
Krylov-Schur decomposition
Reordered Krylov-Schur decomposition
Truncated Krylov decomposition
$A \times$
$=$
$\times$

Figure 5.2. *Restarting a Krylov decomposition.*

decomposition (5.9) 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 [225].

After being truncated, the Krylov decomposition is again expanded to a decomposition of order m using the following algorithm.

Algorithm 5.12 (Expanding a Krylov decomposition).

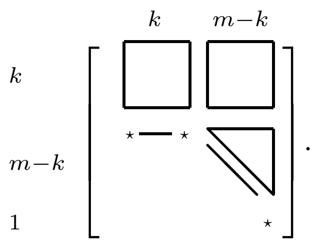
Input: *A Krylov decomposition of order k: $AU_k = U_{k+1}\hat{B}_k$. An integer $m > k$.*

Output: *A Krylov decomposition of order m: $AU_m = U_{m+1}\hat{B}_m$.*

```
FOR $j \leftarrow k+1, 2, \ldots, m$
  $h_j \leftarrow U_j^T A u_j$
  $v \leftarrow Au_j - U_j h_j$
  $h_{j+1,j} \leftarrow \|v\|_2$
  $u_{j+1} \leftarrow v / h_{j+1,j}$
  $\hat{B}_j \leftarrow \begin{bmatrix} \hat{B}_{j-1} & h_j \\ 0 & h_{j+1,j} \end{bmatrix}$
END FOR
```

The remarks concerning the proper implementation of the Arnoldi method, Algorithm 5.8, apply likewise to Algorithm 5.12. The returned factor $\hat{B}_m$ has the following structure:

$\hat{B}_m =$



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 1.23 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 5.11.

2.4 Deflating a Krylov Decomposition

The process of restarting and expanding Krylov decompositions is repeated until convergence occurs. In ARPACK [161, Sec. 4.6], a Ritz value λ of an order m Arnoldi decomposition (5.5) is regarded as converged if the associated Ritz vector $U_m w$ ($\|w\|_2 = 1$) satisfies

$$\|A(U_m w) - \lambda(U_m w)\|_2 = |h_{m+1,m} e_m^T w| \le \max\{\mathbf{u}\|H_m\|_F, \texttt{tol} \cdot |\lambda|\}, \tag{5.10}$$

where `tol` is a chosen user tolerance. A reasonable extension of this criterion to an order m Krylov decomposition (5.8) is given by

$$\|A(U_m w) - \lambda(U_m w)\|_2 = |b_{m+1}^T w| \le \max\{\mathbf{u}\|B_m\|_F, \texttt{tol} \cdot |\lambda|\}. \tag{5.11}$$

Note, that this implies a small backward error for λ, since λ is an eigenvalue of the slightly perturbed matrix $(A+E)$ with $E = -b_{m+1}^T w \cdot u_{m+1}(U_m w)^T$. Similarly, a matrix Q_d containing an orthonormal basis for the space spanned by d Ritz vectors $U_m w_1, \dots, U_m w_d$ is regarded as converged to a basis of an invariant subspace if it satisfies

$$\|AQ_d - Q_d(Q_d^T A Q_d)\|_F = \|b_{m+1}^T Q_d\|_2 \le \max\{\mathbf{u}\|H_m\|_F, \texttt{tol} \cdot \|Q_d^T A Q_d\|_F\|\}, \tag{5.12}$$

It should be noted that convergence of the individual Ritz vectors $U_m w_1, \dots, U_m w_d$ does not yield (nearby) convergence of their 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 [225]. Given a Krylov decomposition of order m, the factor B_m is reduced to a reordered Schur form $\tilde{B}_m = \begin{bmatrix} \lambda & \star \\ 0 & \tilde{B}_{m-1} \end{bmatrix}$. The Krylov decomposition is partitioned as

$$A[\tilde{u}_1, \tilde{U}_{m-1}] = [\tilde{u}_1, \tilde{U}_{m-1}] \left[\begin{array}{cc} \lambda & \star \\ 0 & \tilde{B}_{m-1} \\ \hline \tilde{b}_1 & \tilde{b}_{m-1}^T \end{array}\right],$$

where it is assumed that $|\tilde{b}_1| = |b_{m+1}^T w|$ satisfies inequality (5.11), i.e., the Ritz value λ is 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 $A\tilde{U}_{m-1} = \tilde{U}_{m-1}\tilde{B}_{m-1}$ for the remainder of the computation, which, if properly implemented, can lead to some performance improvements.

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:

$$A[Q_d, U_{m-d}] = [Q_d, U_{m-d+1}] \left[\begin{array}{cc} T_d & \star \\ 0 & B_{m-d} \\ \hline 0 & b_{m-d}^T \end{array}\right], \tag{5.13}$$

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:

$$A[Q_d, \tilde{u}_1, \tilde{U}_{m-d-1}] = [Q_d, \tilde{u}_1, \tilde{U}_{m-d}] \left[\begin{array}{ccc} T_d & \star & \star \\ 0 & \lambda & \star \\ 0 & 0 & \tilde{B}_{m-d-1} \\ \hline 0 & \tilde{b}_1 & \tilde{b}_{m-d-1}^T \end{array}\right]. \tag{5.14}$$

The Ritz value $\lambda \in \mathbb{R}$ is regarded as converged if the scalar $\tilde{b}_1$ satisfies:

$$|\tilde{b}_1| \le \max\{\mathbf{u}\|B_{m-d}\|_F, \mathtt{tol}\cdot\lambda\|\} \tag{5.15}$$

Let $\tilde{Q}$ be an orthogonal matrix so 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 $v = U_{m-d}\tilde{Q}e_1$ is a Ritz vector belonging to λ, which satisfies

$$\|Av - \lambda v\|_2 = |[0, \tilde{b}_1]\tilde{Q}e_1| \le |\tilde{b}_1|.$$

Thus, the Ritz value λ has also converged in the sense of (5.11). If the selected Ritz value λ does not satisfy (5.15), 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 (5.14). It has been pointed out by Byers [64] that this deflation strategy, originally suggested by Stewart [225], can be considered as a variant of aggressive early deflation described in Section 6.1, Chapter 1. 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 (5.15) instead of (5.11) has the advantage that the orthonormal basis spanned by the deflated Ritz vectors nearly satisfies the convergence criterion (5.12). To see this, assume that the d locked Ritz values in (5.13) have been deflated based upon a criterion of the form (5.15). Then there exists a vector $\tilde{b}_d \in \mathbb{R}^d$, where each component satisfies an inequality of type (5.15), so that $\|AQ_d - Q_dT_d\|_F = \|\tilde{b}_d\|_2$. Thus,

$$\|AQ_d - Q_dT_d\|_F \le \sqrt{d}\max\{\mathbf{u}, \mathtt{tol}\}\cdot\|A\|_F.$$

Both algorithms, the implicitly restarted Arnoldi algorithm and the described Krylov-Schur algorithm, have their advantages and disadvantages. 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 A Krylov-Schur Algorithm for Periodic Eigenvalue Problems

This section is concerned with the eigenvalue computation of a product of large and sparse matrices or, equivalently, with the structure-preserving eigenvalue computation of a large and sparse block cyclic matrix. Surprisingly little attention has been paid to this subject in the available literature. Although there exist a number of Krylov subspace methods for block cyclic matrices in the context of solving linear systems, see [47, 48, 95, 102], we are not aware of the use of such methods for eigenvalue computations. This might be attributed to the fact that the numerical instability of a general-purpose method applied to matrix products is not a concern if the desired accuracy of the method is lower than any error caused by working in finite precision arithmetic. However, the following simple example reveals that an Arnoldi or Krylov-Schur algorithm directly applied to a matrix product can lead to computed eigenvalues that have no accuracy at all.

Example 5.13. *Let* $A^{(1)} = A^{(2)} = A^{(3)} = \mathrm{diag}(1, 0.1, 0.01, 0.001, \dots)$*, and let* u_1 *be a random starting vector. We applied the Krylov-Schur algorithm to the matrix product* $A^{(3)}A^{(2)}A^{(1)}$ *using the convergence criterion (5.11) with the parameter* `tol` *set to machine precision. The following table displays the number of correct significant decimal digits of the six largest eigenvalues computed by the Krylov-Schur algorithm as well as by the periodic Krylov-Schur algorithm, which will be described below.*

eigenvalue	*Krylov-Schur*	*periodic Krylov-Schur*
1	15	15
10^{-03}	14	14
10^{-06}	10	14
10^{-09}	8	14
10^{-12}	4	13
10^{-15}	1	12
10^{-18}	0	11

It can be seen that the accuracy of eigenvalues computed by Krylov-Schur drops rapidly with decreasing magnitude; the eigenvalue 10^{-18} *has no accuracy at all. In contrast, the periodic Krylov-Schur computes each displayed eigenvalue to at least* 11 *significant digits correctly.*

3.1 Periodic Arnoldi and Krylov Decompositions

To explain our new periodic variation of the Krylov-Schur algorithm let us recall some material from Chapter 3. The periodic eigenvalue problem consists of finding eigenvalues and invariant subspaces of the matrix product

$$\Pi_{\mathcal{A}} = A^{(p)}A^{(p-1)}\cdots A^{(1)},$$

where $A^{(1)}, \dots, A^{(p)} \in \mathbb{R}^{n\times n}$. As explained in detail in Section 1, Chapter 3, this is to some extent equivalent to computing eigenvalues and certain invariant subspaces of the block cyclic matrix

$$\mathcal{A} = \begin{bmatrix} 0 & & & A^{(p)} \\ A^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & A^{(p-1)} & 0 \end{bmatrix}. \tag{5.16}$$

Here, the invariant subspaces of interest are periodic, i.e., invariant subspaces having a basis of the form

$$X^{(1)} \oplus X^{(2)} \oplus \cdots \oplus X^{(p)} \tag{5.17}$$

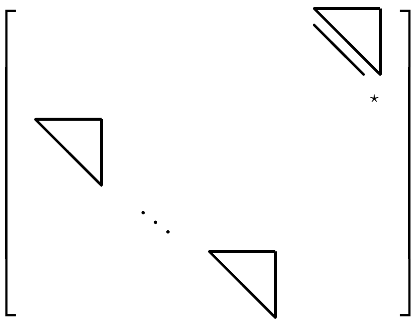
for some matrices $X^{(1)}, X^{(2)}, \dots, X^{(p)} \in \mathbb{R}^{n\times k}$.

From this point of view, it is desirable for a basis of the Krylov subspace $\mathcal{K}_k(\mathcal{A}, u)$ to have a similar structure as (5.17). Indeed, it will be shown that if a starting vector of the form $u_1 = [u_1^{(1)T}, 0, \dots, 0]^T$ is used, then one can construct, by a minor modification of the Arnoldi method, an Arnoldi decomposition of the following type:

$$\mathcal{A} \cdot (U_k^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) = (U_{k+1}^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) \cdot \hat{\mathcal{H}}_k, \tag{5.18}$$

where all matrices $U_k^{(l)} = [u_1^{(l)}, \dots, u_k^{(l)}]$, $l = 1, \dots, p-1$, and $U_{k+1}^{(p)} = [u_1^{(p)}, \dots, u_{k+1}^{(p)}]$ have orthonormal columns. Moreover, the factor $\hat{\mathcal{H}}_k$ in (5.18) takes the form

$$\hat{\mathcal{H}}_k = \begin{bmatrix} 0 & & & \hat{H}_k^{(p)} \\ H_k^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & H_k^{(p-1)} & 0 \end{bmatrix} = \tag{5.19}$$



i.e., $H_k^{(1)}, \dots, H_k^{(p-1)} \in \mathbb{R}^{k\times k}$ 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 upper Hessenberg form.

Any decomposition of the form (5.18)–(5.19) will be called *periodic Arnoldi decomposition* of order k.

Corollary 5.14. *Consider a kth order periodic Arnoldi decomposition of the form (5.18)–(5.19) and assume that the upper triangular matrices $H_k^{(1)}, \dots, H_k^{(p-1)}$ are invertible and that $\hat{H}_k^{(p)}$ has unreduced Hessenberg form. Then the columns of $(U_{k+1}^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)})$ form a basis for $\mathcal{K}_{pk+1}(\mathcal{A}, [u_1^{(1)T}, 0, \dots, 0]^T)$.*

Proof. This result follows from Theorem 5.5 after permuting the $(k+1)$th row of $\hat{\mathcal{H}}_k$ to the bottom row and applying a perfect shuffle permutation to the rest of the matrix . $\square$

An alternative characterization is suggested by the observation that the columns of $U_{k+1}^{(1)}, U_k^{(2)}, \dots, U_k^{(p)}$ form orthonormal bases for the Krylov subspaces

$$\begin{gathered}\mathcal{K}_{k+1}(A^{(p)}A^{(p-1)}\cdots A^{(2)}A^{(1)}, u_1^{(1)}),\\ \mathcal{K}_k(A^{(1)}A^{(p)}\cdots A^{(3)}A^{(2)}, u_1^{(2)}),\\ \vdots\\ \mathcal{K}_k(A^{(p-1)}A^{(p-2)}\cdots A^{(1)}A^{(p)}, u_1^{(p)}),\end{gathered}$$

respectively. Although this point of view is closer to the periodic eigenvalue problem, it hides, to some extent, the connection of the periodic Arnoldi algorithm to the standard Arnoldi algorithm and will henceforth be disregarded.

The following algorithm produces a periodic Arnoldi decomposition. It is formally and numerically equivalent to the Arnoldi method, Algorithm 5.8, applied to the block cyclic matrix $\mathcal{A}$ with starting vector $u_1 = [u_1^{(1)T}, 0, \dots, 0]^T$, with the only difference that the columns of the produced basis U_k are sorted in a particular order.

Algorithm 5.15 (Periodic Arnoldi method).

Input: *Matrices $A^{(1)}, \dots, A^{(p)} \in \mathbb{R}^{n\times n}$, a starting vector $u_1^{(1)} \in \mathbb{R}^n$ with $\|u_1^{(1)}\|_2 = 1$, and an integer $k \le n$.*

Output: *Matrices $U_{k+1}^{(1)} \in \mathbb{R}^{n\times(k+1)}$, $U_k^{(2)}, \dots, U_k^{(p)} \in \mathbb{R}^{n\times k}$ having orthonormal columns, upper triangular matrices $H_k^{(1)}, \dots, H_k^{(p-1)} \in \mathbb{R}^{k\times k}$ and an upper Hessenberg matrix $\hat{H}_k^{(p)} \in \mathbb{R}^{(k+1)\times k}$, defining a periodic Arnoldi decomposition (5.18)–(5.19) of kth order.*

```
$H_0^{(1)} \leftarrow []$, $\dots$, $H_0^{(p-1)} \leftarrow []$, $\hat{H}_0^{(p)} \leftarrow []$
FOR $j \leftarrow 1, 2, \dots, k$
  FOR $l \leftarrow 1, 2, \dots, p-1$
    $h_j^{(l)} \leftarrow U_{j-1}^{(l+1)T} A^{(l)} u_j^{(l)}$
    $v \leftarrow A^{(l)} u_j^{(l)} - U_{j-1}^{(l+1)} h_j^{(l)}$
    $h_{jj}^{(l)} \leftarrow \|v\|_2$
    $u_j^{(l+1)} \leftarrow v / h_{jj}^{(l)}$
    $H_j^{(l)} \leftarrow \begin{bmatrix} H_{j-1}^{(l)} & h_j^{(l)} \\ 0 & h_{jj}^{(l)} \end{bmatrix}$
  END FOR
  $h_j^{(p)} \leftarrow U_j^{(1)T} A^{(p)} u_j^{(p)}$
  $v \leftarrow A^{(p)} u_j^{(p)} - U_j^{(1)} h_j^{(p)}$
  $h_{j+1,j}^{(p)} \leftarrow \|v\|_2$
  $u_{j+1}^{(1)} \leftarrow v / h_{j+1,j}^{(p)}$
  $\hat{H}_j^{(p)} \leftarrow \begin{bmatrix} \hat{H}_{j-1}^{(p)} & h_j^{(p)} \\ 0 & h_{j+1,j}^{(p)} \end{bmatrix}$
END FOR
```

Again, some remarks are necessary:

1. Algorithm 5.15 can be implemented so that in each outer loop exactly p matrix-vector multiplications, involving each of the matrices $A^{(1)}, \dots, A^{(p)}$,

are needed. It can be expected that the computational cost of Algorithm 5.15 is dominated by these matrix-vector multiplications making it comparable to the cost of Algorithm 5.8 applied to the matrix product $A^{(p)}A^{(p-1)}\cdots A^{(1)}$. Note that this statement is only true under the assumption that the matrices $A^{(1)}, A^{(2)}, \dots, A^{(p)}$ are subsequently applied to a vector whenever Algorithm 5.8 requests a matrix-vector product. If the matrix product or parts of it can be condensed in a cheap manner, then the computational cost of Algorithm 5.15 can be higher than the cost of Algorithm 5.8.

2. A major drawback of using Algorithm 5.15 instead of Algorithm 5.8 is the increase of memory requirements by roughly a factor of p due to the need for storing each basis $U_{k+1}^{(1)}, U_k^{(2)}, \dots, U_k^{(p)}$ instead of only one $n \times (k+1)$ basis.

3. Algorithm 5.15 breaks down as soon as it encounters $h_{jj}^{(l)}$, $l = 1, \dots, p-1$, or $h_{j+1,j}^{(p)} = 0$. In both cases, a periodic invariant subspace has been found and the methods described in the next section can be used to deflate this subspace. Afterwards, the iteration is restarted with a random, unit vector orthogonal to the converged invariant subspace.

4. A computed vector $u_j^{(l)}$ may need to be reorthogonalized against previously computed vectors contained in $U_{j-1}^{(l)}$ due to numerical instabilities inherited from the Gram-Schmidt orthonormalization process, see also the remarks concerning the implementation of Algorithm 5.8.

5. It seems to be difficult to construct a periodic shift-and-invert Arnoldi method as $(\mathcal{A} - \sigma I)^{-1}$ does not preserve the block cyclic structure of $\mathcal{A}$ as long as $\sigma \neq 0$. Alternatively, one can use

$$(\mathcal{A} - \sigma^{(1)} I)^{-1} \cdot (\mathcal{A} - \sigma^{(2)} I)^{-1} \cdots (\mathcal{A} - \sigma^{(p)} I)^{-1} = \left(\mathcal{A}^p - \prod_{l=1}^{p} \sigma^{(l)} I \right)^{-1},$$

where $\{\sigma^{(1)}, \dots, \sigma^{(p)}$ is a p-Carrollian tuple. It can be expected, however, that such an approach suffers from numerical instabilities similar to those illustrated in Example 5.13.

For the purpose of restarting and deflation it will be helpful to relax the definition of a periodic Arnoldi decomposition and introduce the notion of a periodic Krylov decomposition.

Definition 5.16. *Let $\mathcal{A}$ be a block cyclic matrix of the form (5.16). The decomposition*

$$\mathcal{A} \cdot (U_k^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) = (U_{k+1}^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) \cdot \hat{\mathcal{B}}_k, \qquad (5.20)$$

is called a periodic Krylov decomposition *of order k if all matrices $U_k^{(l)} = [u_1^{(l)}, \dots, u_k^{(l)}]$, $l = 1, \dots, p-1$, and $U_{k+1}^{(p)} = [u_1^{(p)}, \dots, u_{k+1}^{(p)}]$ have orthonormal columns and if $\hat{\mathcal{B}}_k$ has the form*

$$\hat{\mathcal{B}}_k = \begin{bmatrix} 0 & & & \hat{B}_k^{(p)} \\ B_k^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & B_k^{(p-1)} & 0 \end{bmatrix} \qquad (5.21)$$

for some matrices $B_k^{(1)}, \dots, B_k^{(p-1)} \in \mathbb{R}^{k\times k}$ *and* $\hat{B}_k^{(p)} = \begin{bmatrix} B_k^{(p)} \\ b_k^{(p)T} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k}$.

The following lemma is the periodic equivalent to Lemma 5.11, it shows that any periodic Krylov decomposition can be reduced to a periodic Arnoldi decomposition.

Lemma 5.17. *Let*

$$\mathcal{A} \cdot (U_k^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) = (U_{k+1}^{(1)} \oplus U_k^{(2)} \cdots \oplus U_k^{(p)}) \cdot \hat{\mathcal{B}}_k$$

be a periodic Krylov decomposition. Then there exists a periodic Arnoldi decomposition

$$\mathcal{A} \cdot (\tilde{U}_k^{(1)} \oplus \tilde{U}_k^{(2)} \cdots \oplus \tilde{U}_k^{(p)}) = (\tilde{U}_{k+1}^{(1)} \oplus \tilde{U}_k^{(2)} \cdots \oplus \tilde{U}_k^{(p)}) \cdot \hat{\mathcal{H}}_k$$

so that $\operatorname{span}(U_{k+1}^{(1)}) = \operatorname{span}(\tilde{U}_{k+1}^{(1)})$ *and* $\operatorname{span}(U_k^{(l)}) = \operatorname{span}(\tilde{U}_k^{(l)})$ *for* $l = 1, \dots, p$.

Proof. The proof of this result proceeds in the same fashion as the proof of Lemma 5.11. First, let us partition $\hat{B}_{k+1}^{(p)} = \begin{bmatrix} B_k^{(p)} \\ b_{k+1}^{(p)T} \end{bmatrix}$. Setting $Z = F \cdot H_1(Fb_{k+1}^{(p)})$, where F denotes the flip matrix, yields $b_{k+1}^{(p)T} ZF = h_{k+1,k}^{(p)} e_k^T$. We can apply reduction to periodic Hessenberg form (Algorithm 3.10) to construct orthogonal matrices $Q^{(1)}, \dots, Q^{(p)}$ so that the formal matrix product

$$(FB_k^{(p)T} Z) \cdot (Z^T B_k^{(1)T} F) \cdot (FB_k^{(2)T} F) \cdots (FB_k^{(p-1)T} F)$$

is transformed to periodic Hessenberg form, i.e., $Q^{(p)T} \cdot (FB_k^{(p)T} Z) \cdot Q^{(1)}$ has upper Hessenberg form while the matrices

$$\begin{gathered} Q^{(1)T} \cdot (Z^T B_k^{(1)T} F) \cdot Q^{(2)}, \\ Q^{(2)T} \cdot (FB_k^{(2)T} F) \cdot Q^{(3)}, \\ \vdots \\ Q^{(p-1)T} \cdot (FB_k^{(p-1)T} F) \cdot Q^{(p)} \end{gathered}$$

are upper triangular. This implies that the formal matrix product

$$\begin{aligned} &\overbrace{[(ZQ^{(1)}F)^T \cdot B_k^{(p)} \cdot (FQ^{(p)}F)]}^{:=H_k^{(p)}} \cdot \overbrace{[(FQ^{(p)}F)^T \cdot B_k^{(p-1)} \cdot (FQ^{(p-1)}F)]}^{:=H_k^{(p-1)}} \cdots \\ &\underbrace{[(FQ^{(3)}F)^T \cdot B_k^{(2)} \cdot (FQ^{(2)}F)]}_{:=H_k^{(2)}} \cdot \underbrace{[(FQ^{(2)}F)^T \cdot B_k^{(1)} \cdot (ZQ^{(1)}F)]}_{:=H_k^{(1)}} \end{aligned}$$

is as well in periodic Hessenberg form. Since $Q^{(1)}$ takes the form $1 \oplus \tilde{Q}^{(1)}$, we still have $b_{k+1}^{(p)T} \cdot (ZQ^{(1)}F) = h_{k+1,k}^{(p)} e_k^T$. The proof is concluded by setting

$$\tilde{U}_k^{(1)} = U_k^{(1)} \cdot (ZQ^{(1)}F),\ \tilde{U}_k^{(2)} = U_k^{(2)} \cdot (FQ^{(2)}F),\ \dots,\ \tilde{U}_k^{(p)} = U_k^{(p)} \cdot (FQ^{(p)}F),$$

and $\tilde{U}_{k+1}^{(1)} = [\tilde{U}_k^{(1)}, u_{k+1}^{(1)}]$. □

In analogy to the standard case, (5.20)–(5.21) will be called *periodic Krylov-Schur decomposition* if the matrices $B_k^{(1)}, \dots, B_k^{(p-1)}$ are upper triangular and $B_k^{(p)}$ is in real Schur form.

3.2 Restarting a Periodic Krylov Decomposition

Given a periodic Krylov decomposition of order m,

$$\mathcal{A} \cdot (U_m^{(1)} \oplus U_m^{(2)} \oplus \cdots \oplus U_m^{(p)}) = (U_{m+1}^{(1)} \oplus U_m^{(2)} \oplus \cdots \oplus U_m^{(p)}) \cdot \hat{\mathcal{B}}_m, \tag{5.22}$$

where $\hat{\mathcal{B}}_m$ is a block cyclic matrix having the form (5.21), implicit restarting proceeds as follows. First, orthogonal matrices $Q_1^{(1)}, \dots, Q_1^{(p)} \in \mathbb{R}^{m\times m}$ are constructed so that $T_m^{(p)} = Q_1^{(1)T} H_m^{(p)} Q_1^{(p)}$ is in real Schur form while $T_m^{(1)} = Q_1^{(2)T} B_m^{(1)} Q_1^{(1)}$, $\dots$, $T_m^{(p-1)} = Q_1^{(p)T} B_m^{(p-1)} Q_1^{(p-1)}$ are upper triangular matrices. Next, $k < m$ wanted p-Carrollian tuples of Ritz values are reordered to the top of these factors, using the reordering methods described in Section 4, Chapter 3. This yields further orthogonal matrices $Q_2^{(1)}, \dots, Q_2^{(p)} \in \mathbb{R}^{m\times m}$ so that

$$\begin{aligned} &\mathcal{A} \cdot (U_m^{(1)} Q_1^{(1)} Q_2^{(1)} \oplus U_m^{(2)} Q_1^{(2)} Q_2^{(2)} \oplus \cdots \oplus U_m^{(p)} Q_1^{(p)} Q_2^{(p)}) \\ &= ([U_m^{(1)} Q_1^{(1)} Q_2^{(1)}, u_{m+1}^{(1)}] \oplus U_m^{(2)} Q_1^{(2)} Q_2^{(2)} \oplus \cdots \oplus U_m^{(p)} Q_1^{(p)} Q_2^{(p)}) \cdot \hat{\mathcal{T}}_m, \end{aligned}$$

where $\hat{\mathcal{T}}_m$ is a block cyclic matrix of the form (5.21) corresponding to triangular factors $T_m^{(l)} = \left[\begin{smallmatrix} T_w^{(l)} & \star \\ 0 & T_u^{(l)} \end{smallmatrix}\right]$, $l = 1, \dots, p-1$, and

$$\hat{T}_m^{(p)} = \left[\begin{array}{cc} T_w^{(p)} & \star \\ 0 & T_u^{(p)} \\ \hline b_w^{(p)T} & \star \end{array}\right],$$

with $T_w^{(l)} \in \mathbb{R}^{k\times k}$ for $l = 1, \dots, p$. Note that the block cyclic matrices

$$\mathcal{T}_w = \begin{bmatrix} 0 & & & T_w^{(p)} \\ T_w^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & T_w^{(p-1)} & 0 \end{bmatrix}, \quad \mathcal{T}_u = \begin{bmatrix} 0 & & & T_u^{(p)} \\ T_u^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & T_u^{(p-1)} & 0 \end{bmatrix}.$$

contain the wanted and the unwanted p-Carrollian tuples of Ritz values, respectively. Finally, this periodic Krylov-Schur decomposition is truncated. I.e., by letting $\tilde{U}_k^{(l)}$ contain the first k columns of $U_m^{(l)} Q_1^{(l)} Q_2^{(l)}$, $l = 1, \dots, p$, and setting $\tilde{u}_{k+1}^{(1)} = u_{m+1}^{(1)}$, we obtain the following Krylov-Schur decomposition of order k:

$$\mathcal{A} \cdot (\tilde{U}_k^{(1)} \oplus \tilde{U}_k^{(2)} \oplus \cdots \oplus \tilde{U}_k^{(p)}) = ([\tilde{U}_k^{(1)}, \tilde{u}_{k+1}^{(1)}] \oplus \tilde{U}_k^{(2)} \oplus \cdots \oplus \tilde{U}_k^{(p)}) \cdot \hat{\mathcal{T}}_w,$$

where $\hat{\mathcal{T}}_w$ is identical to $\mathcal{T}_w$, with the only difference that the $k \times k$ matrix $T_w^{(p)}$ is replaced by the $(k+1) \times k$ matrix $\hat{T}_w^{(p)} = \left[\begin{smallmatrix} T_w^{(p)} \\ b_w^{(p)T} \end{smallmatrix}\right]$.

The described restarting process is depicted in Figure 5.3. The obtained Krylov basis is the same basis that would have been obtained if implicit restarting with the filter polynomial $\psi(z) = \prod\limits_{\lambda\in\lambda(\mathcal{T}_u)} (z - \lambda)$ was applied to the (standard) Arnoldi decomposition corresponding to (5.22).

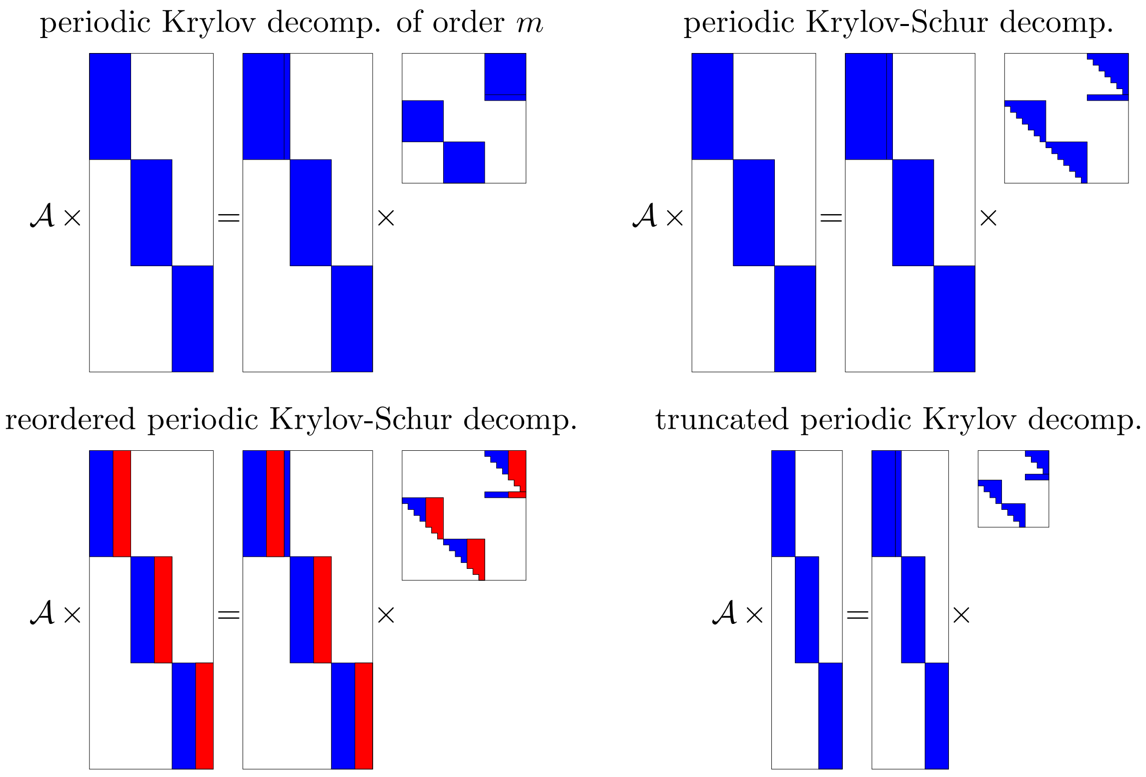

periodic Krylov decomp. of order m
periodic Krylov-Schur decomp.
$\mathcal{A}\times$
$=$
$\times$
reordered periodic Krylov-Schur decomp.
truncated periodic Krylov decomp.

Figure 5.3. *Restarting a periodic Krylov decomposition.*

3.3 Deflating a Periodic Krylov Decomposition

After a periodic Krylov decomposition has been truncated to order k it can be expanded to order m by a variant of the periodic Arnoldi method, Algorithm 5.15, see also Algorithm 5.12. This process of restarting is repeated until convergence occurs. We suggest to use a convergence criterion similar to (5.15) for deflating a set of p Ritz vectors belonging to a p-Carrollian tuple of Ritz values.

For the purpose of describing this deflation strategy in detail, assume that we have an mth order periodic Krylov decomposition of the following particular form:

$$\begin{aligned}&\mathcal{A}\cdot([Q_d^{(1)},U_{m-d}^{(1)}]\oplus[Q_d^{(2)},U_{m-d}^{(2)}]\cdots\oplus[Q_d^{(p)},U_{m-d}^{(p)}])\\&=([Q_d^{(1)},U_{m-d+1}^{(1)}]\oplus[Q_d^{(2)},U_{m-d}^{(2)}]\cdots\oplus[Q_d^{(p)},U_{m-d}^{(p)}])\cdot\hat{\mathcal{B}}_m,\end{aligned}\tag{5.23}$$

where $Q_d^{(1)},\dots,Q_d^{(p)}\in\mathbb{R}^{n\times d}$ and $\hat{\mathcal{B}}_m$ is a block cyclic matrix of the form (5.21) corresponding to factors

$$B_m^{(l)}=\left[\begin{array}{cc}T_d^{(l)} & \star\\ 0 & B_{m-d}^{(l)}\end{array}\right],\ l=1,\dots,p-1,\quad \hat{B}_m^{(p)}=\left[\begin{array}{cc}T_d^{(p)} & \star\\ 0 & \hat{B}_{m-d}^{(p)}\end{array}\right],\tag{5.24}$$

with $T_d^{(1)},\dots,T_d^{(p)}\in\mathbb{R}^{d\times d}$. This implies that the columns of $Q_d=Q_d^{(1)}\oplus\cdots\oplus Q_d^{(p)}$ span a periodic invariant subspace having the block cyclic representation $\mathcal{T}_d$ corresponding to the factors $T_d^{(1)},\dots,T_d^{(p)}$. Thus, the p-Carrollian tuples that are eigenvalues of $\mathcal{T}_d$ have been deflated. To deflate another p-Carrollian tuple of Ritz values, we assume, e.g., by using a periodic Schur decomposition, that the submatrices $B^{(1)}_{m-d}, \dots, B^{(p-1)}_{m-d}, \hat{B}^{(p)}_{m-d}$ in (5.24) take the form

$$B^{(l)}_m = \begin{bmatrix} \lambda^{(l)} & \star \\ 0 & \tilde{B}^{(l)}_{m-d-1} \end{bmatrix}, \; l = 1, \dots, p-1, \quad \hat{B}^{(p)}_m = \left[\begin{array}{cc} \lambda^{(p)} & \star \\ 0 & \tilde{B}^{(p)}_{m-d-1} \\ \hline \tilde{b}^{(p)}_1 & \tilde{b}^{(p)T}_{m-d-1} \end{array}\right].$$

The eigenvalues of the cyclic matrix

$$\Lambda = \begin{bmatrix} 0 & & & \lambda^{(p)} \\ \lambda^{(1)} & \ddots & & \\ & \ddots & \ddots & \\ & & \lambda^{(p-1)} & 0 \end{bmatrix}$$

are regarded as a converged p-Carrollian tuple of Ritz values if the scalar $\tilde{b}^{(p)}_1$ satisfies

$$|\tilde{b}^{(p)}_1| \le \max\{\mathbf{u}\|\hat{B}^{(p)}_m\|_F, \texttt{tol} \cdot |\lambda^{(p)}|\}. \tag{5.25}$$

This implies that the $n \times p$ matrix $V = U^{(1)}_{m-d} e_1 \oplus \cdots \oplus U^{(p)}_{m-d} e_1$ satisfies

$$\|(I - Q_d Q_d^T)\mathcal{A}(I - Q_d Q_d^T)V - V\Lambda\|_F \le \max\{\mathbf{u}\|\hat{B}^{(p)}_m\|_F, \texttt{tol} \cdot \lambda^{(p)}\|\},$$

Thus, the p-Carrollian tuple of Ritz values contained in Λ satisfies the convergence criterion (5.12). If the selected p-Carrollian tuple does not satisfy (5.25), we may test any other tuple of Ritz values by reordering the periodic Schur form of the formal matrix product

$$\begin{bmatrix} \lambda^{(p)} & \star \\ 0 & \tilde{B}^{(p)}_{m-d-1} \end{bmatrix} \begin{bmatrix} \lambda^{(p-1)} & \star \\ 0 & \tilde{B}^{(p-1)}_{m-d-1} \end{bmatrix} \cdots \begin{bmatrix} \lambda^{(1)} & \star \\ 0 & \tilde{B}^{(1)}_{m-d-1} \end{bmatrix}. \tag{5.26}$$

As already pointed out in Section 2.4 the use of a convergence criterion of the form (5.25), leads to a backward stable approximation of the orthonormal basis spanned by all deflated Ritz vectors. To see this for the periodic Krylov-Schur algorithm, let us assume that the d locked p-Carrollian tuples of Ritz values in (5.23) have been deflated based upon a criterion of the form (5.25). Then the matrix $Q_d = (Q^{(1)}_d \oplus \cdots \oplus Q^{(p)}_d)$ satisfies

$$\mathcal{A}Q_d - Q_d \mathcal{T}_d = R_d,$$

where $R_d \in \mathbb{R}^{np \times dp}$ is a block cyclic matrix with

$$\|R_d\|_F \le \sqrt{d} \max\{\mathbf{u}\|A^{(p)}\|_F, \texttt{tol} \cdot \|T^{(p)}_d\|_F\|\}.$$

Note that the subspace spanned by the columns of Q_d is the exact periodic invariant subspace of the slightly perturbed block cyclic matrix $\mathcal{A} - R_d Q_d^T$. Neglecting the effects of roundoff errors, this shows the strong backward stability of the periodic Krylov-Schur algorithm for computing periodic invariant subspaces of block cyclic matrices. Similarly as for the standard Krylov-Schur algorithm [225], roundoff errors do not introduce numerical instabilities, if Algorithm 5.15 is carefully implemented.

To preserve the realness of the Krylov decomposition, it may be necessary to deflate two p-Carrollian tuples of Ritz values, belonging to a pair of complex conjugate eigenvalues of the periodic eigenvalue problem, at once. This can be achieved in a similar way, by reordering the corresponding two-by-two blocks to the top left corners of the factors in the matrix product (5.26).

Deflation of singular factors

In the case that one of the triangular factors $H_m^{(1)}, \dots, H_m^{(p-1)}$ in an mth order periodic Arnoldi decomposition (5.18)–(5.19) happens to be (almost) singular, we can use a deflation procedure similar to the one described in Section 3.4, Chapter 3, see also [46]. Let us outline this deflation by assuming $p = 3$ and $m = 5$, where the third diagonal element of $H_5^{(2)}$ happens to be zero:

$$\hat H_5^{(3)} H_5^{(2)} H_5^{(1)} = \left[\begin{array}{ccccc} x&x&x&x&x\\ x&x&x&x&x\\ 0&x&x&x&x\\ 0&0&x&x&x\\ 0&0&0&x&x\\ \hline 0&0&0&0&x\end{array}\right] \left[\begin{array}{ccccc} x&x&x&x&x\\ 0&x&x&x&x\\ 0&0&0&x&x\\ 0&0&0&x&x\\ 0&0&0&0&x\end{array}\right] \left[\begin{array}{ccccc} x&x&x&x&x\\ 0&x&x&x&x\\ 0&0&x&x&x\\ 0&0&0&x&x\\ 0&0&0&0&x\end{array}\right].$$

Applying a sequence of two Givens rotations, the subdiagonal elements $(2,1)$ and $(3,2)$ of $H_5^{(3)}$ can be annihilated at the expense of introducing two nonzeros in the subdiagonal of $H_5^{(1)}$:

$$\left[\begin{array}{ccccc} \hat x&\hat x&\hat x&\hat x&\hat x\\ \hat 0&\hat x&\hat x&\hat x&\hat x\\ 0&\hat 0&\hat x&\hat x&\hat x\\ 0&0&x&x&x\\ 0&0&0&x&x\\ \hline 0&0&0&0&x\end{array}\right] \left[\begin{array}{ccccc} x&x&x&x&x\\ 0&x&x&x&x\\ 0&0&0&x&x\\ 0&0&0&x&x\\ 0&0&0&0&x\end{array}\right] \left[\begin{array}{ccccc} \hat x&\hat x&\hat x&x&x\\ \hat x&\hat x&\hat x&x&x\\ 0&\hat x&\hat x&x&x\\ 0&0&0&x&x\\ 0&0&0&0&x\end{array}\right].$$

These two newly created nonzeros can be annihilated in a similar fashion, but now – owning to the zero diagonal element – only one nonzero entry is introduced in $H_5^{(2)}$:

$$\left[\begin{array}{ccccc} x&x&x&x&x\\ 0&x&x&x&x\\ 0&0&x&x&x\\ 0&0&x&x&x\\ 0&0&0&x&x\\ \hline 0&0&0&0&x\end{array}\right] \left[\begin{array}{ccccc} \hat x&\hat x&\hat x&x&x\\ \hat x&\hat x&\hat x&x&x\\ 0&0&0&x&x\\ 0&0&0&x&x\\ 0&0&0&0&x\end{array}\right] \left[\begin{array}{ccccc} \hat x&\hat x&\hat x&\hat x&\hat x\\ \hat 0&\hat x&\hat x&\hat x&\hat x\\ 0&\hat 0&\hat x&\hat x&\hat x\\ 0&0&0&x&x\\ 0&0&0&0&x\end{array}\right].$$

Annihilating this entry returns the product to periodic Hessenberg form:

$$\left[\begin{array}{cc|ccc} \hat x&\hat x&x&x&x\\ \hat x&\hat x&x&x&x\\ \hline 0&0&x&x&x\\ 0&0&x&x&x\\ 0&0&0&x&x\\ \hline 0&0&0&0&x\end{array}\right] \left[\begin{array}{cc|ccc} \hat x&\hat x&\hat x&\hat x&\hat x\\ \hat 0&\hat x&\hat x&\hat x&\hat x\\ \hline 0&0&0&x&x\\ 0&0&0&x&x\\ 0&0&0&0&x\end{array}\right] \left[\begin{array}{cc|ccc} x&x&x&x&x\\ 0&x&x&x&x\\ \hline 0&0&x&x&x\\ 0&0&0&x&x\\ 0&0&0&0&x\end{array}\right]. \tag{5.27}$$

This yields a deflated periodic invariant subspace of dimension $2p$. Although another p-Carrollian tuple of eigenvalues is known to consist of zero eigenvalues there seems to be no straightforward way to deflate the corresponding periodic invariant subspace unless the $(4,3)$ subdiagonal entry of $\hat H_5^{(3)}$ is sufficiently small. A simple remedy is to ignore this additional information and perform an explicit restart with a random starting vector orthogonal to the deflated subspace.

A technically more involved remedy is to allow varying dimensions and relabel the periodic Krylov decomposition. Let us repartition (5.27) as follows:

$$\left[\begin{array}{cc|ccc} x & x & x & x & x \\ x & x & x & x & x \\ 0 & 0 & x & x & x \\ \hline 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{array}\right] \left[\begin{array}{ccc|cc} x & x & x & x & x \\ 0 & x & x & x & x \\ \hline 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & x & x \\ \hline 0 & 0 & 0 & 0 & x \end{array}\right] \left[\begin{array}{ccc|cc} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ \hline 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{array}\right].$$

By applying two further steps of the periodic Arnoldi method, Algorithm 5.15, one obtains the following periodic Hessenberg form:

$$\left[\begin{array}{cc|ccc} x & x & x & x & x \\ x & x & x & x & x \\ 0 & 0 & x & x & x \\ \hline 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{array}\right] \left[\begin{array}{ccc|ccc} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ \hline 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \\ \hline 0 & 0 & 0 & 0 & 0 & x \end{array}\right] \left[\begin{array}{ccc|ccc} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ \hline 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{array}\right].$$

This corresponds to a periodic Arnoldi decomposition of the form

$$\begin{aligned} A^{(3)} \cdot \left[\overset{2}{Q^{(3)}}, \overset{3}{U^{(3)}}\right] &= \left[\overset{3}{Q^{(1)}}, \overset{3}{U^{(1)}}\right] \cdot \begin{array}{c} \\ 3 \\ 3 \end{array} \begin{array}{c} \begin{array}{cc} 2 & 3 \end{array} \\ \left[\begin{array}{cc} T^{(3)} & \star \\ 0 & H^{(3)} \end{array}\right] \end{array}, \\ A^{(2)} \cdot \left[\overset{3}{Q^{(2)}}, \overset{3}{U^{(2)}}\right] &= \left[\overset{2}{Q^{(3)}}, \overset{3}{U^{(3)}}, \overset{1}{u^{(3)}}\right] \cdot \begin{array}{c} \\ 2 \\ 4 \end{array} \begin{array}{c} \begin{array}{cc} 3 & 3 \end{array} \\ \left[\begin{array}{cc} T^{(2)} & \star \\ 0 & \hat{H}^{(2)} \end{array}\right] \end{array}, \qquad (5.28) \\ A^{(1)} \cdot \left[\overset{3}{Q^{(1)}}, \overset{3}{U^{(1)}}\right] &= \left[\overset{3}{Q^{(2)}}, \overset{3}{U^{(2)}}\right] \cdot \begin{array}{c} \\ 3 \\ 3 \end{array} \begin{array}{c} \begin{array}{cc} 3 & 3 \end{array} \\ \left[\begin{array}{cc} T^{(1)} & \star \\ 0 & H^{(1)} \end{array}\right] \end{array}. \end{aligned}$$

Note that the deflated matrix product $T^{(3)}T^{(2)}T^{(1)}$ consists of rectangular factors. Slight variations of the algorithms described in Chapter 3 can be used to compute eigenvalues and invariant subspaces of such matrix products in a numerically backward stable manner, see [246] for more details. The undeflated part in (5.28) corresponds to the decomposition

$$A^{(3)}U^{(3)} = U^{(1)}H^{(3)}, \; A^{(2)}U^{(2)} = [U^{(3)}, u^{(3)}]\hat{H}^{(2)}, \; A^{(1)}U^{(1)} = U^{(2)}H^{(1)}$$

By the index transformation

$$(2) \to (3), \quad (3) \to (1), \quad (1) \to (2),$$

it can be seen that this decomposition is actually a periodic Arnoldi decomposition. Hence, the periodic Arnoldi method/Krylov-Schur algorithm can be continued from this decomposition.

4 Krylov-Schur Algorithms for Skew-Hamiltonian and Hamiltonian Eigenvalue Problems

4.1 SHIRA

Shira [175] is a structure-preserving variant of the implicitly restarted Arnoldi algorithm suitable for computing a few eigenvalues and eigenvectors of a large and sparse skew-Hamiltonian matrix

$$W = \begin{bmatrix} A & G \\ Q & A^T \end{bmatrix}, \quad G = -G^T, \; Q = -Q^T,$$

where A, G and Q are real $n \times n$ matrices. SHIRA is based on the following fact.

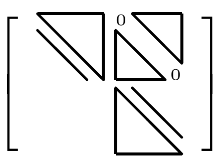
Lemma 5.18 ([175]). *Let $W \in \mathbb{R}^{2n\times 2n}$ be a skew-Hamiltonian matrix and let $u_1 \in \mathbb{R}^{2n}$. Then any Krylov subspace $\mathcal{K}_k(W, u_1)$ is isotropic.*

Let us now consider an unreduced Arnoldi decomposition of order $k < n$:

$$WU_k = [\, U_k, \; u_{k+1} \,] \begin{bmatrix} H_k \\ h_{k+1,k}e_k^T \end{bmatrix}. \tag{5.29}$$

Lemma 5.18 implies that the columns of $[U_{k+1}, JU_{k+1}]$, where $J = \left[\begin{smallmatrix} 0 & I \\ -I & 0 \end{smallmatrix}\right]$, form an orthonormal basis. Roundoff error due to finite precision arithmetic will cloud the situation, as usual. It is thus necessary to modify the Arnoldi method, Algorithm 5.8, 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 $x_1^T J x_2 = 0$, see Section 2 in Chapter 4. An isotropic Krylov subspace can only contain approximations to a one-dimensional invariant subspace belonging to an eigenvalue of algebraic multiplicity two. Consequently, the eigenvalues do not appear in duplicate in the Hessenberg factor H_k. Note that (5.29) can be considered as a truncated PVL decomposition (see Section 2.1 in Chapter 4):

$$[\, U_k, \; JU_k \,]^T W [\, U_k, \; JU_k \,] = \begin{bmatrix} H_k & U_k^T W J U_k \\ 0 & H_k^T \end{bmatrix} =$$

.

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 [9, 175] 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 of enforcing isotropic Krylov subspaces. Assume that the first d Ritz values of the mth order Krylov decomposition

$$W[Q_d, U_{m-d}] = [Q_d, U_{m-d+1}] \left[\begin{array}{cc} T_d & \star \\ 0 & B_{m-d} \\ \hline 0 & b_{m-d}^T \end{array}\right],$$

have been deflated based upon a criterion of the form (5.15). Then the Frobenius norm of the residual $R_d = WQ_d - Q_d T_d$ can be bounded by $\sqrt{d}\max\{\mathbf{u}, \texttt{tol}\|\} \cdot \|A\|_F$.

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

$$\hat{W} = W - R_d Q_d^T + J(R_d Q_d^T)^T J(I - Q_d Q_d^T),$$

see also [235].

4.2 A Two-Sided Krylov-Schur Algorithm for Hamiltonian Matrices

It has been demonstrated in [175] that a variety of other eigenvalue problems can be addressed by SHIRA. For example, consider the Hamiltonian matrix

$$H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}, \quad G = G^T, \; Q = Q^T,$$

where, once again, A, G and Q are real $n \times n$ matrices. In Section 3.6, Chapter 4, we have already seen that $W_0 = H^2$ is a skew-Hamiltonian matrix. Thus, we can apply SHIRA to W_0 and compute the positive and negative square roots of the returned eigenvalues to obtain eigenvalue pairs of H. Of course, since it is assumed that H is large and sparse, one would not square H explicitly but apply the matrix twice to a vector whensoever SHIRA requests the computation of a matrix-vector product. For finding eigenvalues closest to a quadruplet $(\lambda_0, \bar{\lambda}_0, -\lambda_0, -\bar{\lambda}_0)$, where $\lambda_0 \notin \lambda(H)$, Mehrmann and Watkins [175] suggest to use the rational transformation

$$W_1 = (H - \lambda_0 I)^{-1}(H + \lambda_0 I)^{-1}(H - \bar{\lambda}_0 I)^{-1}(H + \bar{\lambda}_0 I)^{-1}, \tag{5.30}$$

or, if $\lambda_0 \in \mathbb{R} \cup \imath\mathbb{R}$,

$$W_2 = (H - \lambda_0 I)^{-1}(H + \lambda_0 I)^{-1}. \tag{5.31}$$

Direct calculation reveals that both matrices, W_1 and W_2, are skew-Hamiltonian.

The basis for our new variant of the Krylov-Schur algorithm is the observation that the matrices W_0, W_1, W_2 can be written as a matrix product $A^{(1)}A^{(2)}$ for some simpler matrices $A^{(2)}, A^{(1)} \in \mathbb{R}^{2n \times 2n}$ so that the reversed matrix product $A^{(2)}A^{(1)}$ is also skew-Hamiltonian. Our choice is as follows:

$$\begin{array}{ll} W_0: & A^{(1)} = A^{(2)} = H, \\ W_1: & A^{(1)} = (H - \lambda_0 I)^{-1}(H - \bar{\lambda}_0 I)^{-1}, \; A^{(2)} = (H + \lambda_0 I)^{-1}(H + \bar{\lambda}_0 I)^{-1}, \\ W_2: & A^{(1)} = (H - \lambda_0 I)^{-1}, \; A^{(2)} = (H + \lambda_0 I)^{-1}, \; \lambda_0 \in \mathbb{R}. \end{array}$$

The periodic Krylov-Schur algorithm applied to the matrix product $A^{(2)}A^{(1)}$ with some starting vector $u_1^{(1)} \in \mathbb{R}^n$ produces an orthonormal basis $U_{k+1}^{(1)} \oplus U_k^{(2)}$ for the Krylov subspace

$$\mathcal{K}_{2k+1}\left(\begin{bmatrix} 0 & A^{(2)} \\ A^{(1)} & 0 \end{bmatrix}, \begin{bmatrix} u_1^{(1)} \\ 0 \end{bmatrix} \right).$$

The corresponding Krylov decomposition reads as follows:

$$\begin{bmatrix} 0 & A^{(2)} \\ A^{(1)} & 0 \end{bmatrix} \begin{bmatrix} U_k^{(1)} & 0 \\ 0 & U_k^{(2)} \end{bmatrix} = \begin{bmatrix} U_{k+1}^{(1)} & 0 \\ 0 & U_k^{(2)} \end{bmatrix} \begin{bmatrix} 0 & \hat{B}_k^{(2)} \\ B_k^{(1)} & 0 \end{bmatrix}, \tag{5.32}$$

where $\hat{B}_k^{(2)} = \begin{bmatrix} B_k^{(2)} \\ b_k^{(2)T} \end{bmatrix}$.

Equivalently, the columns of the matrices $U_{k+1}^{(1)}$ and $U_k^{(2)}$ form orthonormal bases for the Krylov subspaces $\mathcal{K}_{k+1}(A^{(2)}A^{(1)}, u_1^{(1)})$ and $\mathcal{K}_k(A^{(1)}A^{(2)}, A^{(1)}u_1^{(1)})$, respectively. Since the matrix products $A^{(2)}A^{(1)}$ and $A^{(1)}A^{(2)}$ are skew-Hamiltonian, it follows that both Krylov subspaces are isotropic which in turn implies that the columns of $[U_{k+1}^{(1)}, JU_{k+1}^{(1)}]$ as well as those of $[U_{k+1}^{(2)}, JU_{k+1}^{(2)}]$ form orthonormal bases. As for Shira, this property must be enforced in finite precision arithmetic by a slight modification of the periodic Arnoldi method, Algorithm 5.15. Newly produced vectors $u_j^{(2)}$ and $u_{j+1}^{(1)}$ must be orthogonalized against the columns of $JU_{j-1}^{(2)}$ and $JU_j^{(1)}$, respectively.

Restarting and deflation of (5.32) are carried out in precisely the same manner as for general periodic Krylov decomposition, see Sections 3.2 and 3.3.

For the rest of this section, we consider only the case $A^{(1)} = A^{(2)} = H$, for which (5.32) implies

$$\left[U_k^{(2)},\ JU_k^{(2)} \right]^T H \left[U_k^{(1)},\ JU_k^{(1)} \right] = \left[\begin{array}{cc} B_k^{(1)} & U_k^{(2)T} H J U_k^{(1)} \\ 0 & B_k^{(2)T} \end{array} \right]. \tag{5.33}$$

If the periodic Krylov decomposition (5.32) happens to be a periodic Arnoldi decomposition, then $B_k^{(1)}$ is an upper triangular matrix and $B_k^{(2)}$ is an upper Hessenberg matrix, i.e., the two-sided decomposition (5.33) can be considered as a truncated symplectic URV decomposition, see Section 3.6 in Chapter 4.

If a deflation criterion of the form (5.25) is used, then the periodic Krylov-Schur algorithm produces matrices $[Q, JQ], [Z, JZ] \in \mathbb{R}^{n\times 2d}$ having orthonormal columns such that

$$(H+E)[Q, JQ] = [Z, JZ] \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & R_{22}^T \end{array} \right], \quad R_{ij} \in \mathbb{R}^{d\times d},$$

holds for some backward error $E \in \mathbb{R}^{n\times n}$ with

$$\|E\|_F \leq \sqrt{d}\max\{\mathbf{u}\cdot \|H\|_F, \mathtt{tol}\cdot \|R_{22}\|_F\|\}.$$

The positive and negative square roots of $\lambda(-R_{11}R_{22})$, which can be computed by the periodic QR algorithm, are the eigenvalues of the matrix $H+E$. Thus, the described two-sided Krylov-Schur algorithm is a backward stable method for computing the eigenvalues of a Hamiltonian matrix. Moreover, it is structure-preserving in the sense that the eigenvalue symmetries are preserved. Approximations to invariant subspaces of H can be obtained from the matrices Q and Z using methods described in [37]. However, it should be noted that these methods are not backward stable. It is thus not clear how to guarantee that the computed invariant subspace satisfies a specified backward error bound. This subject requires further investigation.

5 Balancing Sparse General Matrices

In Section 4, Chapter 1, 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. Therefore, Chen and Demmel [72] developed a two-stage balancing algorithm, particularly suited for large and sparse matrices and with similar positive effects.

5.1 Irreducible forms

The first stage of the balancing algorithm proposed by Chen and Demmel 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

$$P^T A P = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \tag{5.34}$$

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:

$$P^T A P = \begin{bmatrix} A_1 & \star & \cdots & \star \\ 0 & A_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \star \\ 0 & \cdots & 0 & A_r \end{bmatrix}, \quad A_i \in \mathbb{R}^{n_i\times n_i}, \quad n_i \geq 1. \tag{5.35}$$

Constructing this final *irreducible form* (a pre-stage of the so called Frobenius normal form [40]) is equivalent to finding the strongly connected components of the incidence graph of A and numbering them in their topological order. To see this, we will briefly introduce the necessary graph theoretic tools. More details can be found, e.g., in [143, 248].

Definition 5.19. *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*

$$V = \{v_1, \dots, v_n\}, \quad E = \{(v_i, v_j) : \; a_{ij} \neq 0\},$$

respectively.

Definition 5.20. *A directed graph is called* strongly connected *if for any pair of vertices v and w there is a path from v to w and one from w to v. The* strongly connected components *of a directed graph are its maximal strongly connected subgraphs.*

It is well-known that the strongly connected components of the incidence graph of a matrix in irreducible form (5.35) are the subgraphs belonging to the vertex sets

$$V_i = \{v_{k_{i-1}+1}, \dots, v_{k_i}\}, \quad i = 1, \dots, r, \quad k_i = \sum_{j=1}^{i} n_j,$$

see e.g. [143]. 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 [232] 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

$$V_i = \{v_{l_{k_{i-1}+1}}, \dots, v_{l_{k_i}}\}, \quad i = 1, \dots, r, \quad k_i = \sum_{j=1}^{i} n_j,$$

for some integers $l_1, \dots, l_n \in [1, n]$ and integers $n_1, \dots, n_r$ satisfying $\sum n_i = n$. W.l.o.g., we may assume $V_i \preceq V_j$ for $i < j$. Next, let us construct 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 Algorithm 1.31, see [72]. 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 [72] that Algorithm 1.31 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 given.

5.2 Krylov-Based Balancing

Algorithm 1.32, 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 by Chen and Demmel [72], who called them *Krylov-based algorithms*, although it must be mentioned that none of the proposed algorithms exploits a complete Krylov subspace.

One such Krylov-based algorithm, the so called KrylovAtz, is based on the following fact.

Lemma 5.21. *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*

$$D = \operatorname{diag}(\sqrt{x_1/y_1}, \sqrt{x_2/y_2}, \dots, \sqrt{x_n/y_n}), \tag{5.36}$$

then $\|D^{-1}AD\|_2 = \rho(A)$.

Proof. See, e.g., [72]. □

This scaling achieves minimal 2-norm 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 5.21 to $|A| := [|a_{ij}|]_{i,j=1}^{n}$ to construct a (possibly suboptimal) diagonal scaling matrix D. It was observed in [72] 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 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 [72]. It makes use of the fact that multiplying A by z approximates one step of the power method applied to $|A|$ with starting vector $[1, 1, \ldots, 1]^T$.

Algorithm 5.22 (KrylovAtz [72])**.**
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 5.21.*

```
$D \leftarrow I_n$
FOR $k = 1, 2, \ldots$
  $z \leftarrow$ vector of length $n$ with random $\pm 1$ entries
  $p \leftarrow D^{-1}(A(Dz)), \quad r \leftarrow D(A^T(D^{-1}z))$
  FOR $i = 1, \ldots, n$
    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 5.23. *Based on the experimental results presented in [72] it was proposed to replace the conditions $p_i \neq 0$ $r_i \neq 0$ in the inner loop of Algorithm 5.22 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.*

Algorithm 5.22 can be applied to the diagonal blocks of an irreducible form (5.35). If this form is not available, Algorithm 5.22 can be applied to the complete matrix A with less theoretical justification but often to an equal norm-reducing effect. For numerical experiments, see [72] or Section 6.3.

Chen and Demmel [72] also proposed a one-sided variant of Algorithm 5.22, 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.

6 Balancing Sparse Hamiltonian Matrices

In this section, we describe modifications of the algorithms in the previous section in order to obtain symplectic balancing algorithms for large and sparse Hamiltonian matrices. For this purpose, a structure-preserving irreducible form is derived and it is shown that Krylov balancing can be easily modified to yield symplectic scaling matrices.

6.1 Irreducible forms

A structure-preserving irreducible form can be obtained by investigating the properties of the incidence graph of a Hamiltonian matrix. For notational convenience, the vertex labels of such an incidence graph differ slightly from the labels used in Definition 5.19.

Definition 5.24. *Let $H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix} \in \mathbb{R}^{2n\times 2n}$ be a Hamiltonian matrix, then the* incidence graph of H, *denoted by $\mathcal{G}_H(V,E)$, is a directed graph with vertex and edge sets*

$$\begin{aligned} V &= \{v_1,\dots,v_n,w_1,\dots,w_n\}, \\ E &= \{(v_i,v_j):\ a_{ij}\neq 0\}\cup\{(v_i,w_j):\ g_{ij}\neq 0\}\cup \\ &\quad \{(w_i,v_j):\ q_{ij}\neq 0\}\cup\{(w_i,w_j):\ -a_{ji}\neq 0\}. \end{aligned}$$

Lemma 5.25. *The incidence graph $\mathcal{G}_H(V,E)$ of a Hamiltonian matrix H satisfies the following:*

a) there exists a path from v_i to v_j if and only if there exists a path from w_j to w_i,

b) there exists a path from v_i to w_j if and only if there exists a path from v_j to w_i,

c) there exists a path from w_i to v_j if and only if there exists a path from w_j to v_i.

Proof. By induction on the path length k. For $k = 2$, the conclusions are direct consequences of the definition of the incidence graph and the symmetries of G and Q. Now, assume that a)–c) hold for any path of maximum length $\bar{k}$ and that there is a path of length $\bar{k}+1$ from v_i to w_j. We cut this path in two pieces by choosing a vertex $p = v_l$ or $p = w_l$ so that there are paths of length not greater than $\bar{k}$ from v_i to p and from p to w_j. In the case $p = v_l$, the induction hypothesis shows the existence of paths from w_l to w_i and from v_j to w_l. In the other case, we obtain paths from v_j to v_l and from v_l to w_i. In any case, there is a path from v_j to w_i. The converse as well as assertions a) and c) are analogously proved. □

Lemma 5.25 implies consequences for the structure of the strongly connected components of $\mathcal{G}_H(V,E)$.

Corollary 5.26. *Let $\mathcal{G}_H(V,E)$ be the incidence graph of a Hamiltonian matrix H.*

$$V_1 := \{v_{i_1},\dots,v_{i_k},w_{j_1},\dots,w_{j_l}\}$$

is the vertex set of a strongly connected component if and only if

$$\check{V}_1 := \{v_{j_1},\dots,v_{j_l},w_{i_1},\dots,w_{i_k}\}$$

is the vertex set of a strongly connected component.

If $V_1 \cap \check{V}_1 \neq \emptyset$ then the strong connectivity property and Corollary 5.26 enforce $V_1 = \check{V}_1$. The corresponding component will be called of *type (II)*. In the other case, $V_1 \cap \check{V}_1 = \emptyset$, we say that the corresponding components are of *type (I)*. Further information is available about edges between components.

Lemma 5.27. *Let $\mathcal{G}_H(V, E)$ be the incidence graph of a Hamiltonian matrix H and let V_1 and V_2 correspond to strongly connected components of type (I). Then, there exists an edge from a vertex in V_1 to a vertex in V_2 if and only if there exists an edge from a vertex in $\check{V}_2$ to a vertex in $\check{V}_1$, where $\check{V}_1$ and $\check{V}_2$ are defined as in Corollary 5.26. Moreover, there are no edges between components of type (II).*

Proof. Let $(v_i, v_j) \in E$ with $v_i \in V_1$ and $v_j \in V_2$. Then, by definition, $w_i \in \check{V}_1$, $w_j \in \check{V}_2$ and by Lemma 5.25a), $(w_j, w_i) \in E$. For the second part, let V_3 and V_4 correspond to two distinct strongly connected components of type (II) and assume that there are vertices $v_i \in V_3$, $v_j \in V_4$ with $(v_i, v_j) \in E$. This implies $w_i \in V_3$ and $w_j \in V_4$ because $V = \check{V}$ for type (II) components. Again, by Lemma 5.25a), $(w_j, w_i) \in E$, which means that $V_3 \cup V_4$ is the vertex set of a strongly connected component. This contradicts the assumption. The proof is analogous for edges of the form (v_i, w_j), (w_i, v_j) or (w_i, w_j). $\square$

Lemma 5.27 implies that there is always a numbering of the strongly connected components so that their topological order takes the form

$$V_1 \preceq \cdots \preceq V_r \preceq V_{r+1} \preceq \cdots \preceq V_{r+s} \preceq \check{V}_r \preceq \cdots \preceq \check{V}_1, \tag{5.37}$$

where $V_1, \dots, V_r$, $\check{V}_1, \dots, \check{V}_r$ correspond to type (I) components and $V_{r+1}, \dots, V_{r+s}$ correspond to type (II) components of $\mathcal{G}_H(V, E)$.

This shows, together with Corollary 5.26, the existence of a permutation matrix P so that

$$P^T H P = \begin{bmatrix} A_1 & \star & \cdots & \star & \cdots & \cdots & \star & \cdots & \star \\ & \ddots & \ddots & \vdots & & & \vdots & & \vdots \\ & & A_r & \star & \cdots & \cdots & \star & \cdots & \star \\ & & & H_{r+1} & 0 & 0 & \vdots & & \vdots \\ & & & & \ddots & 0 & \vdots & & \vdots \\ & & & & & H_{r+s} & \star & \cdots & \star \\ & & & & & & -A_r^T & \ddots & \vdots \\ & & & & & & & \ddots & \star \\ & & & & & & & & -A_1^T \end{bmatrix}, \tag{5.38}$$

is block upper triangular matrix, where all diagonal blocks are irreducible. Moreover, each vertex set belonging to a principal submatrix H_{r+i} satisfies $V_{r+i} = \check{V}_{r+i}$, implying that H_{r+i} can be chosen to be Hamiltonian: $H_{r+i} = \left[\begin{smallmatrix} A_{r+i} & G_{r+i} \\ Q_{r+i} & -A_{r+i}^T \end{smallmatrix}\right]$. Thus, the spectrum of H contains the eigenvalues of the unstructured matrices $A_i, -A_i^T$ and those of the Hamiltonian matrices H_{r+i}. Unfortunately, the irreducible form (5.38) does not respect the Hamiltonian structure of H. Therefore, it is now of interest to construct a permutation matrix $\tilde{P}$ which is structure-preserving and leads to a form $\tilde{P}^T H \tilde{P}$ where the essential information of (5.38) can be easily read off.

It was already noted in Section 4.1, Chapter 4, that the group of generalized symplectic permutation matrices can be used to obtain useful classifications for $\tilde{P}^T H \tilde{P}$.

Theorem 5.28. *For any Hamiltonian matrix H there exists a symplectic generalized permutation matrix $\tilde{P}$ so that*

$$\tilde{H} := \tilde{P}^T H \tilde{P} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{21} & \tilde{G}_{11} & \tilde{G}_{12} \\ 0 & \tilde{A}_{22} & \tilde{G}_{12}^T & \tilde{G}_{22} \\ 0 & 0 & -\tilde{A}_{11}^T & 0 \\ 0 & \tilde{Q}_{22} & -\tilde{A}_{21}^T & -\tilde{A}_{22}^T \end{bmatrix}, \tag{5.39}$$

where

$$\tilde{A}_{11} = \begin{bmatrix} A_1 & \star & \dots & \star \\ 0 & A_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \star \\ 0 & \dots & 0 & A_r \end{bmatrix}, \qquad \tilde{A}_{22} = \begin{bmatrix} A_{r+1} & 0 & \dots & 0 \\ 0 & A_{r+2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & A_{r+s} \end{bmatrix},$$

$$\tilde{G}_{22} = \begin{bmatrix} G_{r+1} & 0 & \dots & 0 \\ 0 & G_{r+2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & G_{r+s} \end{bmatrix}, \ \tilde{Q}_{22} = \begin{bmatrix} Q_{r+1} & 0 & \dots & 0 \\ 0 & Q_{r+2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & Q_{r+s} \end{bmatrix},$$

and all matrices A_i, $i = 1, \dots, r$, and $\left[\begin{smallmatrix} A_{r+i} & G_{r+i} \\ Q_{r+i} & -A_{r+i}^T \end{smallmatrix}\right]$, $i = 1, \dots, s$, are irreducible.

Proof. Using the elementary symplectic matrices $P_j^{(s)}$ defined in (4.56) we can construct a product of such matrices, $\tilde{P}_1 = P_{j_1}^{(s)} P_{j_2}^{(s)} \dots P_{j_k}^{(s)}$, so that in all type (I) components of $\mathcal{G}_{\tilde{P}_1^T H \tilde{P}_1}(V, E)$ the vertex sets V_i contain only v-vertices, $\check{V}_i$ only w-vertices and $V_i \succeq \check{V}_i$ for $i = 1, \dots, r$. By a simultaneous reordering of the v- and w-vertices, there exists a permutation matrix P_2 such that

$$\tilde{H} = \begin{bmatrix} P_2^T & 0 \\ 0 & P_2^T \end{bmatrix} (\tilde{P}_1^T H \tilde{P}_1) \begin{bmatrix} P_2 & 0 \\ 0 & P_2 \end{bmatrix}$$

has an incidence graph whose strongly connected components correspond to the vertex sets

$$V_1 = \{v_1, \dots, v_{k_1}\}, \ V_2 = \{v_{k_1+1}, \dots, v_{k_2}\}, \ \dots, \ V_r = \{v_{k_{r-1}+1}, \dots, v_{k_r}\},$$

$$V_{r+1} = \{v_{k_r+1}, \dots, v_{k_{r+1}}, w_1, \dots, w_{l_1}\}, \ \dots,$$

$$V_{r+s} = \{v_{k_{r+s-1}+1}, \dots, v_{k_{r+s}}, w_{l_{s-1}+1}, \dots, w_{l_s}\},$$

$$\check{V}_1 = \{w_1, \dots, w_{k_1}\}, \ \check{V}_2 = \{w_{k_1+1}, \dots, w_{k_2}\}, \ \dots, \ \check{V}_r = \{w_{k_{r-1}+1}, \dots, w_{k_r}\},$$

where the topological order is given by (5.37). Now, the structure of $\tilde{H}$ is a direct consequence of Lemma 5.27. □

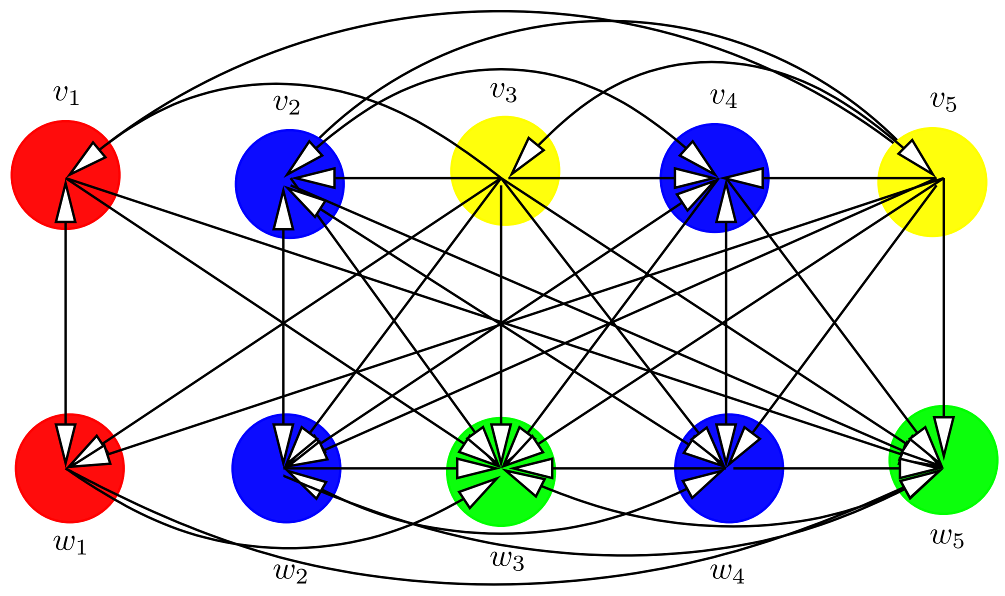


Figure 5.4. *Incidence graph of the Hamiltonian matrix H in Example 5.29.*

Example 5.29. *Consider a 10×10 Hamiltonian matrix H with the sparsity pattern*

$$H = \left[\begin{array}{ccccc|ccccc}
\mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} \\
\mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} \\
\mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} \\
\mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} \\
\mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} \\
\hline
\mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} \\
\mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} \\
\mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} \\
\mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} \\
\mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{X}
\end{array}\right].$$

Here, the symbol $\mathtt{0}$ *denotes a zero entry and* $\mathtt{X}$ *denotes an arbitrary nonzero entry. Its incidence graph $G_H(V, E)$ is shown in Figure 5.4. The vertex sets of the strongly connected components are*

$$V_1 = \{w_3, w_5\},\ \check{V}_1 = \{v_3, v_5\},\ V_2 = \{v_1, w_1\},\ V_3 = \{v_2, v_4, w_2, v_4\},$$

V_1 and $\check{V}_1$ belong to type (I) components, while V_2 and V_3 belong to type (II) components. A suitable topological order, as illustrated in Figure 5.5, is given by $V_1 \preceq V_2 \preceq V_3 \preceq \check{V}_1$. Using the construction given in the proof of Theorem 5.28, we obtain the following structure-preserving irreducible form:

$$\tilde{P}^T H \tilde{P} = \left[\begin{array}{ccccc|ccccc}
\mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} \\
\mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} \\
\mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{0} & \mathtt{0} \\
\mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{X} \\
\mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{X} \\
\hline
\mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{0} \\
\mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{0} \\
\mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{0} & \mathtt{0} \\
\mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{X} \\
\mathtt{0} & \mathtt{0} & \mathtt{0} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{X} & \mathtt{0} & \mathtt{X} & \mathtt{X}
\end{array}\right].$$

In short, given a Hamiltonian matrix H, it can be reduced to the irreducible form (5.39) by first computing a topological sort of its incidence graph, classifying

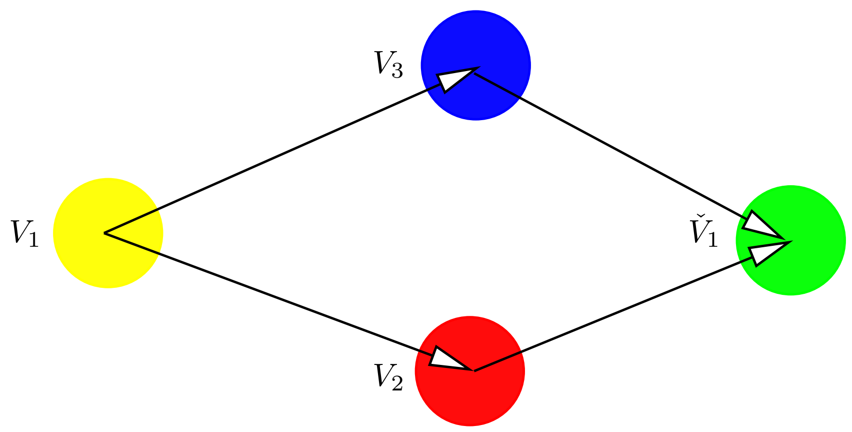


Figure 5.5. *Strongly connected components of the incidence graph of the Hamiltonian matrix H in Example 5.29.*

the type (I) and type (II) components, and permuting the columns and rows of H in the corresponding order. The following algorithm implements the described procedure. It calls a subroutine topsort, which is discussed below.

Algorithm 5.30.

Input: *A Hamiltonian matrix $H = \left[\begin{smallmatrix} A & G \\ Q & -A^T \end{smallmatrix}\right] \in \mathbb{R}^{2n \times 2n}$.*

Output: *A symplectic generalized permutation matrix $\tilde{P}$ so that $\tilde{P}^T H \tilde{P}$ has the form (5.39). The matrix H is overwritten by $\tilde{P}^T H \tilde{P}$.*

```
$\tilde{P} \leftarrow I_{2n}$
$(V_1, \dots, V_k) \leftarrow \text{topsort}\left( \left[\begin{smallmatrix} A & G \\ Q & -A^T \end{smallmatrix}\right] \right)$
FOR $i = 1, \dots, k$
  IF $\exists \{v_j, w_j\} \in V_i$ THEN type$(i) \leftarrow 2$ ELSE type$(i) \leftarrow 1$ END IF
END FOR
$l \leftarrow 0$
FOR $i = 1, \dots, k$
  IF type$(i) = 1$ AND $l < \#(\text{type}(i) = 1)/2$ THEN
    $l \leftarrow l + 1$
    FOR EACH $w_j \in V_i$
      $H \leftarrow (P_j^{(s)})^T H P_j^{(s)}$     % see (4.56) for the definition of $P_j^{(s)}$
      $\tilde{P} \leftarrow \tilde{P} P_j^{(s)}$
      delete $w_j$ from $V_i$,   insert $v_j$ in $V_i$
    END FOR
  END IF
END FOR
$p \leftarrow 0$
FOR $i = 1, \dots, k$
  IF $p < n$ THEN
    FOR EACH $v_j \in V_i$
      $p \leftarrow p + 1$
      $P_1 \leftarrow I_n + e_j e_p^T + e_p e_j^T - e_j e_j^T - e_p e_p^T$
      $H \leftarrow (P_{pj}^{(d)})^T H P_{pj}^{(d)}$     % see (4.55) for the definition of $P_{pj}^{(d)}$
```

```
          $\tilde{P} \leftarrow \tilde{P} P_{pj}^{(d)}$
        END FOR
      END IF
    END FOR
```

The subroutine topsort, called in line 2 of Algorithm 5.30, accepts an arbitrary square matrix as input and returns the vertex sets of the strongly connected components of its incidence graph in their topological order. This can be achieved by Tarjan's algorithm [232], which uses two depth first searches and has found an excellent implementation in Fortran 77 [92]. In an object-oriented programming environment, it is preferable to use an implementation which is able to handle arbitrarily defined graphs. In this case, no information about the incidence graph has to be stored. Such subroutines are for example provided by the C++ library LEDA [172]. Note that it is an open problem to modify Tarjan's algorithm so that it requires less computational time by taking advantage of the Hamiltonian structure. The complete Algorithm 5.30 runs in $\mathcal{O}(n + nz)$ time, where nz is the number of nonzeros in H.

Algorithm 4.33, the structure-preserving algorithm for isolating eigenvalues of a Hamiltonian, is in some sense a suboptimal variant of Algorithm 5.30. In graph-theoretic terms, it works on sink and source nodes of $\mathcal{G}_H(V,E)$ or subgraphs of $\mathcal{G}_H(V,E)$. A sink (source) node s corresponds to a strongly connected component of $\mathcal{G}_H(V,E)$ that satisfies $s \preceq V_j$ ($s \succeq V_j$) for any other strongly connected component V_j. In the first phase, the algorithm seeks a sink node s of $\mathcal{G}_H(V,E)$ and permutes the corresponding diagonal element to the (n,n) position of H by means of a symplectic generalized permutation matrix $\tilde{P}$. For the remainder, only the matrix $\tilde{P}^T H \tilde{P}$ with the first and $(n+1)$-th rows and columns expunged is considered. The algorithm iterates until no sink node can be found. In the second phase, the procedure is repeated for source nodes, which are permuted to the $(1,1)$ position. The worst case complexity of this algorithm is $O(n \cdot nz)$ which compares unfavorably with the complexity of Algorithm 5.30. However, it was experimentally observed that Algorithm 5.30 required more time than Algorithm 4.33 for matrices with density nz/n^2 greater than a certain ratio γ. As already mentioned at the end of Section 5.1, the exact value of γ depends on the sparsity pattern and the particular implementation but a typical observation was $\gamma \approx 1/10$.

6.2 Krylov-Based Balancing

Suppose that we have transformed the Hamiltonian matrix to the form (5.39). The Hamiltonian eigenvalue problem now decomposes into eigenvalue problems for matrices of the form

$$H_I = \begin{bmatrix} A_I & G_I \\ 0 & -A_I^T \end{bmatrix}, \quad H_{II} = \begin{bmatrix} A_{II} & G_{II} \\ Q_{II} & -A_{II}^T \end{bmatrix},$$

where A_I and H_{II} are irreducible matrices. In this section we describe a Krylov-based balancing algorithm that aims to reduce the norms of these matrices while preserving their Hamiltonian structure.

For H_I, we can simply apply Algorithm 5.22 to A_I to construct a diagonal matrix D so that $D^{-1}AD$ is nearly balanced. If the matrix H_I is transformed to

$$\begin{bmatrix} D^{-1} & 0 \\ 0 & D \end{bmatrix} \begin{bmatrix} A_I & G_I \\ 0 & -A_I^T \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & D^{-1} \end{bmatrix} = \begin{bmatrix} D^{-1}A_I D & D^{-1}G_I D^{-1} \\ 0 & -(D^{-1}A_I D)^T \end{bmatrix},$$

then also the matrix $-(D^{-1}A_I D)^T$ is balanced in this sense.

For H_{II}, we propose a simple modification of Algorithm 5.22 so that it returns a symplectic scaling matrix. This modification is based on the following fact.

Lemma 5.31. *Let $H \in \mathbb{R}^{2n\times 2n}$ be an irreducible Hamiltonian matrix and $S = |H|$. If x and y are the normalized right and left Perron vectors of S, then the diagonal matrix D defined by (5.36) in Lemma 5.21 is symplectic.*

Proof. Let $P = \left[\begin{smallmatrix} 0 & I_n \\ I_n & 0 \end{smallmatrix}\right]$, then $PS = S^T P$ and thus $x = Py$ which implies that D is symplectic. □

The Perron vectors x and y can be approximated by matrix-vector products using the statistically motivated procedure in Algorithm 5.22. For Hamiltonian matrices, the following lemma shows how to guarantee that these approximations yield symplectic scaling matrices.

Lemma 5.32. *Let $H = \left[\begin{smallmatrix} A & G \\ Q & -A^T \end{smallmatrix}\right] \in \mathbb{R}^{2n\times 2n}$ be a Hamiltonian matrix and let $z \in \mathbb{R}^{2n}$. If*

$$\bar{H} = \begin{bmatrix} A & G \\ Q & A^T \end{bmatrix}, \quad P = \begin{bmatrix} 0 & I_n \\ I_n & 0 \end{bmatrix}, \tag{5.40}$$

then $p = \bar{H}z$ and $r = \bar{H}^T P z$ satisfy $r = Pp$, implying that

$$\begin{aligned} D &= \mathrm{diag}(\sqrt{r_1/p_1}, \dots, \sqrt{r_{2n}/p_{2n}}) \\ &= \mathrm{diag}(\sqrt{p_{n+1}/p_1}, \dots, \sqrt{p_{2n}/p_n}, \sqrt{p_1/p_{n+1}}, \dots, \sqrt{p_n/p_{2n}}) \end{aligned}$$

is symplectic.

Proof. The statement is a direct consequence of the fact that $P\bar{H}$ is symmetric. □

This leads us to the following adaption of Algorithm 5.22 to Hamiltonian matrices.

Algorithm 5.33 (KrylovAtz for Hamiltonian matrices).

Input: *An irreducible Hamiltonian matrix $H \in \mathbb{R}^{2n\times 2n}$.*

Output: *A symplectic diagonal matrix D so that $D^{-1}HD$ is approximately balanced in the sense of Lemma 5.21.*

```
$D \leftarrow I_{2n}$
FOR $k = 1, 2, \dots$
  $z \leftarrow$ vector of length $2n$ with random $\pm 1$ entries
  $z \leftarrow Dz$,   $p \leftarrow \bar{H}z$,   $p \leftarrow D^{-1}p$
  FOR $i = 1, \dots, n$
    IF $p_i \neq 0$ AND $p_{n+i} \neq 0$ THEN
      $d_{ii} \leftarrow d_{ii} \cdot \sqrt{|p_i|/|p_{n+i}|}$
      $d_{n+i,n+i} \leftarrow d_{n+i,n+i} \cdot \sqrt{|p_{n+i}|/|p_i|}$
    END IF
  END FOR
END FOR
```

Example	n	Algorithm 4.33	Algorithm 5.30
Ex. 1.6 [1]	30	$8 \times (1 \times 1)$	$8 \times (1 \times 1)$
			$2 \times (2 \times 2)$
		$1 \times (52 \times 52)$	$1 \times (48 \times 48)$
Ex. 2.4 [1] ($\varepsilon = 0$)	4	$1 \times (4 \times 4)$	$2 \times (2 \times 2)$
Ex. 2.9 [1]	55	$4 \times (1 \times 1)$	$10 \times (1 \times 1)$
			$2 \times (2 \times 2)$
		$1 \times (106 \times 106)$	$1 \times (96 \times 96)$
Ex. 4.3 [1]	60	$56 \times (1 \times 1)$	$56 \times (1 \times 1)$
($\mu = 4, \delta = 0, \kappa = 0$)		$1 \times (8 \times 8)$	$2 \times (4 \times 4)$

Table 5.1. *Sizes of the decoupled eigenvalue problems after application of Algorithms 4.33 and 5.30.*

The remarks following Algorithm 5.22 apply also to Algorithm 5.33.

Remark 5.34. *Algorithm 5.33 works with $\bar{H}$, see (5.40), instead of H implying that the action of the matrices A, G and Q on a vector must be known. Alternatively, one could make use of the relation*

$$\bar{H} \begin{bmatrix} z_u \\ z_l \end{bmatrix} = H \begin{bmatrix} z_u \\ 0 \end{bmatrix} + \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} H \begin{bmatrix} 0 \\ z_l \end{bmatrix}.$$

6.3 Merits of Balancing

Most applications of Hamiltonian eigenvalue problems have their background in control theory. Therefore, we used two benchmark collections from this area for our numerical experiments. The first collection [1] was already used in Section 6, Chapter 1, and contains examples for continuous-time algebraic Riccati equations (CAREs) corresponding to Hamiltonian matrices. The second collection by Chahlaoui and Van Dooren [68] aims at model reduction problems for linear time-invariant systems. Each example provides matrices $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, and $C \in \mathbb{R}^{p \times n}$. Here, the corresponding Hamiltonian matrix $H = \left[\begin{smallmatrix} A & BB^T \\ C^T C & -A^T \end{smallmatrix}\right]$ can be used to determine the closed-loop poles of the system which in turn can help to evaluate the quality of the closed-loop performance of the reduced-order model.

Permutation algorithms

We compared Algorithm 4.33, which isolates eigenvalues of a Hamiltonian matrix, with Algorithm 5.30, which computes a structure-preserving irreducible form of a Hamiltonian matrix. Both algorithms attempt to decouple the Hamiltonian eigenvalue problem into smaller-sized problems. Algorithm 5.30 is potentially more successful, as explained at the end of Section 6.1. Indeed, we observed this phenomena in the four examples that are listed in Table 5.1. For instance, Algorithm 4.33 applied to Example 2.9 [1] isolates four eigenvalues, which means that the other eigenvalues can be computed from a 106×106 Hamiltonian matrix. Algorithm 5.30 isolates ten eigenvalues. The other eigenvalues are contained in two 2×2 matrices and a 96×96 Hamiltonian matrix. As costs for computing eigenvalues crucially depend on the size of the largest decoupled eigenvalue problem, we may conclude that it will be beneficial to use Algorithm 5.30 as a cheap preliminary reduction step. However, it should be noted that all examples of [68] correspond to irreducible Hamiltonian matrices, showing the limitation of such an approach.

Matrix norm reduction

We have examined the capability of Algorithm 5.33, in the following denoted by HTZ, to reduce the norms of Hamiltonian matrices. If a cutoff value δ, see Remark 5.23, was used, then Algorithm 5.33 is denoted by CUT. We let the number of iterations in HTZ and CUT vary from 1 to 10, the cutoff δ from 0.1 to 10^{-10} by powers of 10, and measured the minimal Frobenius norm of the scaled matrices. These norms were compared with the Frobenius norms of the scaled matrices returned by Algorithm 4.34, the symplectic scaling algorithm for Hamiltonian matrices, or for short BAL. All tests were done in MATLAB. Table 5.2 summarizes the results we obtained with the two benchmark collections. For the examples not listed either scaling strategy makes no or little difference to the matrix norm. In most cases,

Example	n	$\|H\|_F$	BAL	HTZ	CUT
Ex. 1.6 [1]	30	1.4×10^{08}	1.2×10^{03}	1.3×10^{03}	1.3×10^{03}
Ex. 2.2 [1]	2	1.0×10^{06}	2.9×10^{05}	5.9×10^{05}	2.7×10^{05}
Ex. 2.3 [1]	2	1.4×10^{06}	2.0×10^{04}	1.4×10^{06}	1.8×10^{05}
Ex. 2.7 [1]	4	1.0×10^{12}	2.1×10^{06}	1.9×10^{06}	1.9×10^{06}
Ex. 2.9 [1]	55	4.4×10^{10}	4.0×10^{03}	4.1×10^{03}	2.7×10^{03}
Ex. 4.4 [1]	421	8.6×10^{11}	2.5×10^{06}	7.2×10^{09}	3.5×10^{06}
Beam [68]	348	1.0×10^{05}	5.0×10^{03}	6.3×10^{03}	5.7×10^{03}
Build [68]	48	2.2×10^{04}	8.0×10^{02}	5.4×10^{03}	2.9×10^{03}
CDPlayer [68]	120	1.5×10^{06}	3.3×10^{05}	3.6×10^{05}	3.4×10^{05}
ISS [68]	270	2.9×10^{04}	8.8×10^{02}	3.4×10^{04}	3.4×10^{04}

Table 5.2. *Norms of Hamiltonian matrices with and without scaling.*

BAL, HTZ and CUT give very similar results. A notable exception is Example 4.4, where HTZ reduces the norm of H only by two orders of magnitude while BAL and CUT reduce it by more than five orders of magnitude. Furthermore, only BAL is capable to reduce the norm of the ISS example from [68].

It was proposed in [72] to use 5 as the default number of iterations and $\delta = 10^{-8}$ as the default cutoff value. Using these values instead of optimal values, the norms of the scaled matrices returned by CUT are usually no more than a factor of ten larger. The only exception in the benchmark collections is Example 1.6, where the norm of the scaled matrix, using CUT with default values, is 1.7×10^5.

Eigenvalue computation

Balancing may have a strong positive impact on the accuracy of eigenvalue computations. The first point we want to illuminate is the merits of decoupling. Let us consider Example 2.9 [1]. We applied a MATLAB implementation of the square-reduced method [27] (see also Section 3.6 in Chapter 4), SQRED, to the corresponding 110×110 Hamiltonian matrix. The relative errors of seven selected eigenvalues are displayed in the second column of Table 5.3. The 'exact' eigenvalues used to obtain these errors were computed with the QR algorithm in quadruple precision. Next, we used Algorithm 4.33 as a preliminary reduction step, which isolates $\pm\lambda_1, \pm\lambda_2$. Consequently, these eigenvalues are computed without any round-off error. All the other eigenvalues were computed using SQRED applied to the remaining 106×106 block. The third column of Table 5.3 contains the resulting relative errors. With Algorithm 5.30, ten eigenvalues, $\pm\lambda_1, \pm\lambda_2, \dots, \pm\lambda_5$, are isolated and four eigenvalues $\pm\lambda_6, \pm\lambda_7$ are contained in two 2×2 blocks. The latter eigenvalues were computed applying the QR algorithm to the 2×2 blocks which yields, as can be seen in the last column of Table 5.3, relatively small errors. In fact, they are almost 10 orders more accurate than the eigenvalues obtained by SQRED with and without Algorithm 4.33.

Eigenvalue	SQRED	Alg. 4.33+SQRED	Algorithm 5.30
$\lambda_1 = -20$	1.7×10^{-05}	0	0
$\lambda_2 = -20$	1.7×10^{-05}	0	0
$\lambda_3 \approx -5.30$	1.2×10^{-10}	4.5×10^{-11}	0
$\lambda_4 \approx -33.3$	1.2×10^{-12}	7.7×10^{-11}	0
$\lambda_5 \approx -221$	3.8×10^{-13}	4.2×10^{-12}	0
$\lambda_6 \approx -5.16 + 5.26\imath$	1.9×10^{-06}	2.6×10^{-05}	5.5×10^{-15}
$\lambda_7 \approx -5.16 - 5.26\imath$	1.9×10^{-06}	2.6×10^{-05}	5.5×10^{-15}

Table 5.3. *Relative errors of eigenvalues computed by the square-reduced method with and without permuting.*

We have also investigated the influence of scaling on the accuracy of sparse eigensolvers. For this purpose, we applied the Fortran implementation of ARPACK [161] to the 96×96 irreducible Hamiltonian matrix $\tilde{H}$ obtained after Algorithm 5.30 had been applied to Example 2.9 in [1]. The parameter `tol` in the stopping criterion, see [161, Sec. 2.3.5] or Section 2.4, was set to machine precision and the dimension of the Arnoldi basis was limited to 40. ARPACK computed the 20 eigenvalues of largest magnitude, the relative errors of those eigenvalues which have negative real part are displayed in the left graph of Figure 5.6. Also displayed are the relative errors when ARPACK is applied to the operators $D_{\text{BAL}}^{-1}\tilde{H}D_{\text{BAL}}$ and $D_{\text{CUT}}^{-1}\tilde{H}D_{\text{CUT}}$, where D_{BAL} and D_{CUT} are the symplectic scaling matrices computed by BAL and CUT, respectively. The graph on the right shows the same quantities, but computed using a Fortran implementation of SHIRA, see [175] and Section 4.1, instead of ARPACK. Figure 5.6 shows that for both, ARPACK and SHIRA, either scaling strategy yields considerable improvements with respect to eigenvalue accuracies. It should be noted, though, that such drastic improvements cannot always be expected. In the case that a matrix is well-balanced and no eigenvalues (or blocks) can be isolated, there is often no considerable effect of any balancing strategy. On the other hand, is is quite common for real-world applications to be badly scaled or to lead to a natural decoupling of eigenvalues so that improvements can often be observed.

Example 4.4 from [1] demonstrates that balancing is a must in some applications. The QR algorithm applied to the corresponding Hamiltonian matrix without balancing does *not* converge. ARPACK encounters a similar error, the QR algorithm fails to compute the eigenvalues of some Ritz block during the Arnoldi iteration. Scaling resolves these problems. Both, BAL+ARPACK and CUT+ARPACK compute

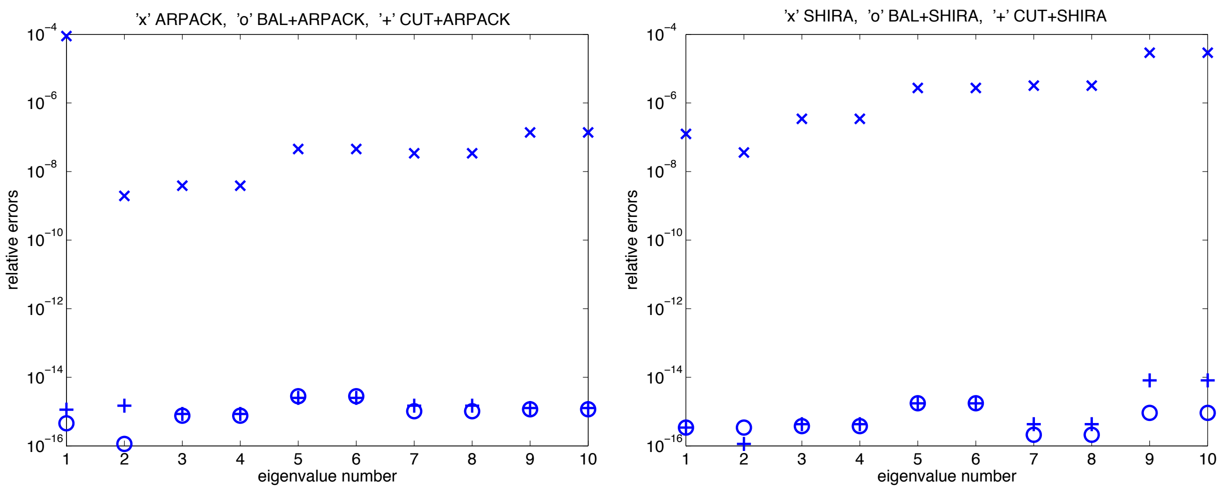


Figure 5.6. *Relative errors of eigenvalues computed by* Arpack *and* Shira *with and without scaling for Example 2.9.*

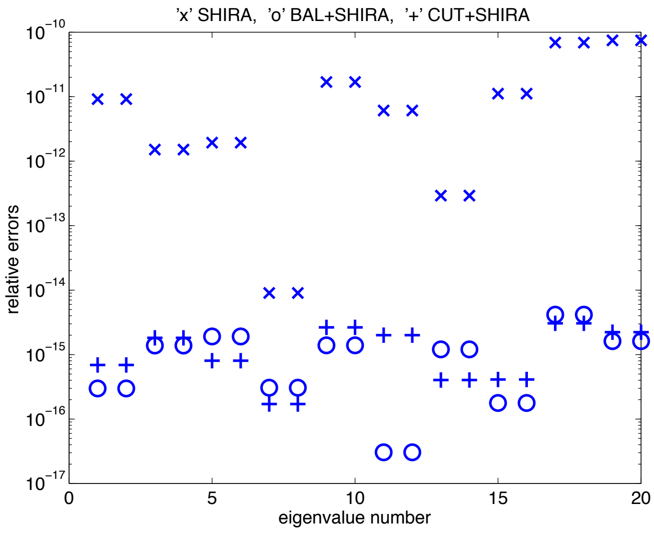


Figure 5.7. *Relative errors of eigenvalues computed by* Shira *with and without scaling for Example 4.4.*

eigenvalues with a relative error close to machine precision. On the other hand, Shira runs to completion, even for the unscaled matrix. The relative errors of the 20 largest eigenvalues with negative real part computed with Shira, Bal+Shira and Cut+Shira are displayed in Figure 5.7. Again, Figure 5.7 shows that scaling leads to considerably more accurate eigenvalues.

Chapter 6

Conclusions and Future Research

In this thesis, we have investigated numerical methods for the solution of general and structured eigenvalue problems. Moreover, we have presented software implementing these methods. Contributions have been made to various aspects of the QR algorithm, the QZ algorithm, the periodic QR algorithm, structure-preserving methods for (skew-)Hamiltonian matrices, and the Krylov-Schur algorithm. Details about the exact nature of these contributions can be found in the beginning of each chapter. Here follows a list of subjects that might be of most interest to the reader:

- In Section 2.2, Chapter 1, we show that the condition number for a complex eigenvalue of a real matrix with respect to real perturbations is at most a factor of $1/\sqrt{2}$ smaller than the corresponding condition number with respect to complex perturbations.
- In Section 5.4, Chapter 1, it is shown that the failure of the large-bulge multishift QR algorithm can be related to the intrinsic ill-conditioning of the pole assignment problem.
- In Section 7.3, Chapter 1, we describe an efficient block algorithm for reordering Schur decompositions.
- In Sections 5.5 and 6, Chapter 2, a tiny-bulge multishift QZ algorithm with aggressive early deflation is developed.
- In Section 3, Chapter 3, it is shown that the periodic QR algorithm is numerically equivalent to the QR algorithm applied to a cyclic block matrix.
- In Sections 2.2 and 2.3, Chapter 4, we derive perturbation bounds and structured condition numbers for eigenvalues and invariant subspaces of skew-Hamiltonian matrices.
- In Section 5, Chapter 4, we develop block algorithms for orthogonal symplectic decompositions.
- In Section 3, Chapter 5, a special-purpose Krylov-Schur algorithm for products of matrices is developed.
- In Section 6.1, Chapter 5, we prove a structure-preserving irreducible form for Hamiltonian matrices.

- In Section 5, Appendix B, a comprehensive Fortran 77/MATLAB software library aimed at computing eigenvalues and invariant subspaces of skew-Hamiltonian and Hamiltonian matrices is presented.

Of course, several problems related to the subjects studied in this thesis remain open. This is only a partial list:

- Although we have seen that the exploitation of linear convergence phenomena in the QR algorithm by aggressive early deflation has the potential to yield some performance improvements, there is still a need for a cheap and reliable method for detecting early deflations in order to take full advantage of this potential.

- Presently, the implementation of the tiny-bulge multishift QZ algorithm with aggressive early deflation is in an experimental stage. The development of a LAPACK-like implementation is work under progress.

- So far, there is no numerical method for Hamiltonian eigenvalue problems known that meets all three requirements of an ideal method (see Page 122) satisfactorily.

- The application of the two-sided Hamiltonian Krylov-Schur algorithm to large algebraic Riccati equations and passivity preserving model reduction is subject to future research.

Appendix A

Background in Control Theory

In this chapter, we give a brief introduction to some concepts of systems and control theory. The presentation is restricted to subjects related to this thesis; 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 [114, 191, 205, 239, 269], which should be consulted for proofs and further details.

1 Basic Concepts

A *linear continuous-time system* with constant coefficients can be described by a set of matrix differential and algebraic equations

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad x(0) = x_0, \tag{A.1a}$$
$$y(t) = Cx(t), \tag{A.1b}$$

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.

The popularity of such systems is mainly due to the fact that many computational tasks related to (A.1) are mathematically well understood and can be solved with reliable numerical algorithms. In practice, however, the dynamics of a process is described by physical laws that rarely lead to linear ordinary differential equations with constant coefficients. Often, discretization and linearization techniques are used to construct linear systems that describe such processes, albeit only in an approximate and local sense.

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 shall be 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,

$$x(t) = \begin{bmatrix} a(t,h) & a(t,2h) & \cdots & a(t,(N-1)h) \end{bmatrix}^T.$$

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

$$\dot{x}(t) = Ax(t) + B \begin{bmatrix} a_l(t) & a_r(t) \end{bmatrix}^T, \tag{A.5}$$

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 $C = e^T_{(N-1)/2}$ under the assumption that N is odd.

The tractability of linear 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* $u(\cdot) \to y(\cdot)$.

1.1 Stability

For a moment, let us only 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.

(i) *The system (A.10) is called* asymptotically stable *if for each x_0 the solution $x(t)$ satisfies $\lim_{t\to\infty} x(t) = 0$.*

(ii) *If for each x_0 there exists a constant $C > 0$ so that $\|x(t)\| \leq C$ for all $t > 0$ then the system is called* stable*.*

(iii) *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 the norm of a matrix.

Lemma A.4. *Let*

$$\rho(A) := \max\{|\lambda| : \lambda \textit{ is an eigenvalue of } A\}$$

denote the spectral radius *of A.*

(i) *The spectral radius is dominated by any induced matrix norm $\|\cdot\|$, i.e., $\rho(A) \leq \|A\|$.*

(ii) *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 $\|A\| \leq \rho(A) + \epsilon$.*

(iii) *Moreover, there is an induced 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.

(i) *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.*

(ii) *It 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, A restricted to the subspace $\mathcal{U}$ 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 motivates us to call $\mathcal{U}^-$ the *asymptotically stable*, $\mathcal{U}^- \oplus \mathcal{U}^0$ the *stable* and $\mathcal{U}^+$ the *unstable subspace* of system (A.1).

1.2 Controllability and Observability

Let us focus on the state equation

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0. \tag{A.11}$$

A control 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 $x(t_f) = x_f$. Note that for a linear control system (A.11) it can be assumed w.l.o.g. that $x_f = 0$. Controllability can be completely characterized by algebraic criteria.

Theorem A.6. *The following statements are equivalent:*

(i) *the linear system (A.11) is controllable;*
(ii) $\operatorname{rank}([B, AB, \dots, A^{n-1}B]) = n$*;*
(iii) $\operatorname{rank}([A - \lambda I, B]) = n, \ \forall \lambda \in \lambda(A)$*.*

If the system (A.11) is controllable, the *matrix pair* (A, B) is also 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 5.8, with starting vector $u_1 = B$ does not break down.*

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. Once again for $m = 1$, the relation $\mathcal{K}_n(A, B) = \mathcal{C}$ reveals the close relationship between Krylov subspaces and controllability.

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.

1.3 Pole Assignment

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 becomes a subtle issue, see [239, Sec. 4.1] or the discussion in Section 5.4, Chapter 1.

2 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)

$$C^T Q C + XA + A^T X - XGX = 0, \tag{A.14}$$

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

$$u_\star(t) = -R^{-1}B^T X_\star,$$

where $X_\star$ is the unique solution of CARE (A.14) 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.14) can be obtained from certain invariant subspaces of the Hamiltonian matrix

$$H = \left[\begin{array}{cc} A & G \\ -C^T Q C & -A^T \end{array}\right].$$

To see this, suppose X is a symmetric solution of (A.14). Then

$$H\left[\begin{array}{cc} I_n & 0 \\ -X & I_n \end{array}\right] = \left[\begin{array}{cc} I_n & 0 \\ -X & I_n \end{array}\right]\left[\begin{array}{cc} A-GX & G \\ 0 & -(A-GX)^T \end{array}\right].$$

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 $X_\star = -X_2X_1^{-1}$ is the stabilizing solution of (A.14). 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 [29, 173].

3 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.

3.1 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 [62]: 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 $\|A + A^T\|_F/2$ [242]). 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 [62] 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 $\mathcal{O}(\mathbf{u}) \cdot \|A\|_2$-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 [112] to this matrix.

It is not known whether the use of the symplectic URV/periodic Schur algorithm [38], see also Section 3.6 in Chapter 4, also results in a computed $\gamma(A)$ which is within an $\mathcal{O}(\mathbf{u}) \cdot \|A\|_2$-distance of the exact stability radius.

3.2 Distance to Uncontrollability

The distance of an controllable matrix pair (A, B) to an uncontrollable one is defined as

$$\rho(A,B) = \min\{\|[E,F]\|_2 \,:\, (A+E, B+F) \text{ is uncontrollable}\}. \qquad \text{(A.15)}$$

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.15) can actually be attained.

Eising [94] has shown how the multi-parameter optimization problem (A.15) 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.15).*

A number of algorithms for minimizing $\sigma_{\min}([A-\mu I, B])$ have been developed, see [115, 57] for some recent developments in this area, a more comprehensive list of references can be found in [51]. As far as we know, none of these algorithms is capable to produce a reliable estimate of (A.15) 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 6.1 in Chapter 1. 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 [51], 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 $[E, F] = -\|\tilde{b}\|_2 \cdot [0, y\tilde{b}^H]$. 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 $\rho(A, B) \leq \|C\|_2$. 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 7, Chapter 1, are used, then the described algorithm requires $\mathcal{O}(n^3)$ flops.

Appendix B
Software

1 Computational Environment

If not otherwise stated, we have used an IBM Power3 based SMP system for performing the numerical experiments described in this thesis. 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 [84, 85, 159] and standard LAPACK [7] 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.

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 [210].

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	$[\, x,\ y \,] \leftarrow [\, x,\ y \,] \left[\begin{smallmatrix} c & -s \\ s & c \end{smallmatrix}\right]$	$x, y \in \mathbb{R}^n, c, s \in \mathbb{R}$	$6n$
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	$[\, x,\ y \,] \leftarrow [\, x,\ y \,] \left[\begin{smallmatrix} c & -\bar{s} \\ s & c \end{smallmatrix}\right]$	$x, y \in \mathbb{R}^n, c \in \mathbb{R}, s \in \mathbb{C}$	$18n$
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).*

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, $A \in \mathbb{F}^{m \times n}$, $H \in \mathbb{C}^{n \times n}$ Hermitian, $S \in \mathbb{R}^{n \times n}$ symmetric, $T \in \mathbb{F}^{n \times n}$ upper triangular, $U \in \mathbb{F}^{n \times n}$ unit upper triangular and $\alpha, \beta \in \mathbb{F}$.

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, $C \in \mathbb{F}^{m \times n}$, $H \in \mathbb{C}^{n \times n}$ Hermitian, $S \in \mathbb{R}^{n \times n}$ symmetric, $T \in \mathbb{F}^{m \times m}$ upper triangular, $U \in \mathbb{F}^{m \times m}$ unit upper triangular and $\alpha, \beta \in \mathbb{F}$.

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 selected LAPACK subroutines.*

3 QR and QZ Algorithms

The Fortran 77 implementations of the block QR and QZ algorithms with aggressive deflation described in Chapters 1 and 2 are still in an experimental stage and available on request from the author.

4 Periodic Eigenvalue Problems

Fortran 77 routines for computing eigenvalues and invariant subspaces of real periodic eigenvalue problems have already been presented in the author's diploma thesis [147]. The following routines cover complex periodic eigenvalue problems associated with general matrix products of the form

$$[A^{(p)}]^{s^{(p)}}[A^{(p-1)}]^{s^{(p-1)}} \cdots [A^{(1)}]^{s^{(1)}},$$

where $A^{(1)}, \dots, A^{(p)} \in \mathbb{C}^{n\times n}$ and $s^{(1)}, \dots, s^{(p)} \in \{1, -1\}$. All routines satisfy the SLICOT [36] implementation and documentation standards [266]

4.1 Fortran Routines

`ZPGHRD` Reduces a general matrix product to periodic Hessenberg form.

`ZPHEQZ` Computes the periodic Schur form of a general matrix product in periodic Hessenberg form.

`ZPTORD` Reorders the periodic Schur form of a general matrix product so that a selected cluster of eigenvalues appears in the top left corner of the matrix product.

4.2 Matlab functions

The following MATLAB functions use the MEX gateway routine `percomplex.f` to access the Fortran routines listed above for computing eigenvalues and invariant subspaces of periodic eigenvalue problems.

`pqzhess.m` Reduces a general matrix product to periodic Hessenberg form.

`pqzreorder.m` Reorders the periodic Schur form of a general matrix product.

`pqzschur.m` Computes the periodic Schur form of a general matrix product.

5 HAPACK

HAPACK 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 [36] implementation and documentation standards [266] 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.math.tu-berlin.de/~kressner/hapack/` for further information.

5.1 Fortran Routines

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.

Auxiliary Routines

`DCROOT` Complex square root in real arithmetic.

`DHAEX2` Swaps adjacent diagonal blocks in a (skew-)Hamiltonian Schur decomposition.

`DLABMX` Auxiliary routine for `DHASUB`.

`DLABTR` Solution of a certain triangular block system.

`DLAESB` Applies the WY representation for a product of elementary orthogonal symplectic transformation.

`DLAEST` Constructs the WY representation for a product of elementary orthogonal symplectic transformation.

`DLANHA` Norm of a (skew-)Hamiltonian matrix.

`DLAPQR` Periodic Schur decomposition of a product of two small matrices.

`DLAPV2` Periodic Schur decomposition of a product of two 2-by-2 matrices.

`DLAPVB` Panel reduction for PVL decomposition.

`DLASUB` Panel reduction for symplectic URV decomposition.

`DSKMV` Skew-symmetric matrix-vector product.

`DSKR2` Skew-symmetric rank-2 update.

`DSKR2K` Skew-symmetric rank-$2k$ update.

`DSKRKB` Computes $\alpha C + \beta ABA^T$ for skew-symmetric matrices B and C.

`DSKUPD` Computes ZAZ^T for a skew-symmetric matrix A.

`DTGPX2` Swaps adjacent diagonal blocks in a periodic Schur decomposition.

`DTGPY2` Solution of a small periodic Sylvester equation.

`DTRQML` Computes matrix-matrix products involving a quasi-triangular matrix.

`ILAHAP` Problem-dependent parameters for the local environment.

5.2 Matlab functions

Functions related to Hamiltonian eigenvalue problems

The following MATLAB functions use the MEX gateway routine `hapack_haeig.f` to access HAPACK routines for computing eigenvalues and invariant subspaces of Hamiltonian matrices, see also Figure B.1.

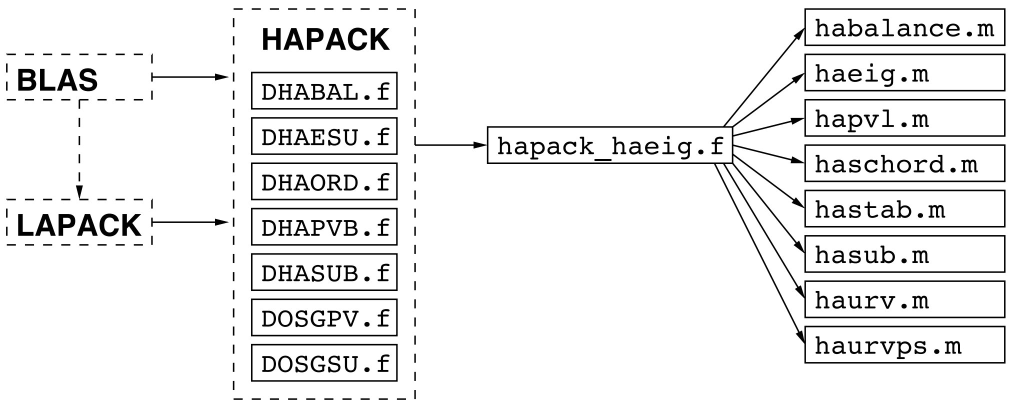


Figure B.1. *Dependencies between HAPACK routines,* MATLAB *functions and the MEX gateway routine* `hapack_haeig.f`.

`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.

Functions related to skew-Hamiltonian eigenvalue problems

The following MATLAB functions use the MEX gateway routine `hapack_sheig.f` to access HAPACK routines for computing eigenvalues and invariant subspaces of skew-Hamiltonian matrices, see also Figure B.2.

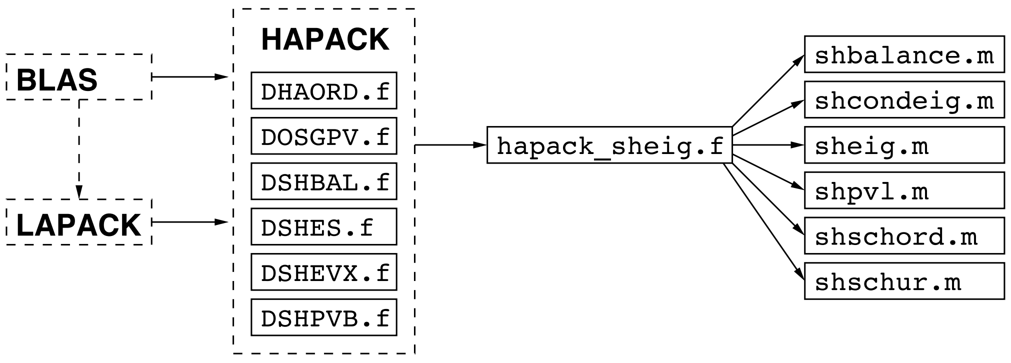


Figure B.2. *Dependencies between HAPACK routines,* MATLAB *functions and the MEX gateway routine* `hapack_sheig.f`.

`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.

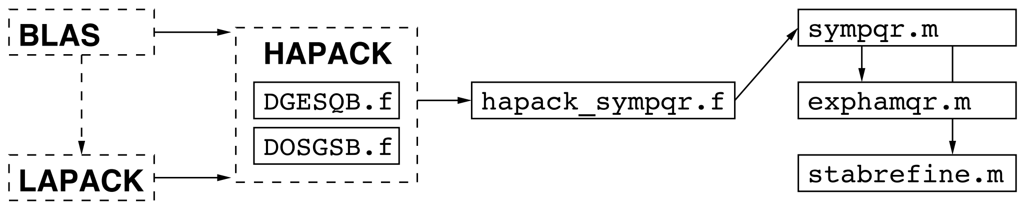


Figure B.3. *Dependencies between HAPACK routines,* MATLAB *functions and the MEX gateway routine* `hapack_sympqr.f`.

Functions related to the symplectic QR decomposition

The following MATLAB functions are (indirectly) based on the MEX gateway routine `hapack_sympqr.f` to access HAPACK routines for computing a symplectic QR decomposition, see also Figure B.3.

`sympqr.m` Symplectic QR decomposition.

`exphamqr.m` MATLAB implementation of the explicit Hamiltonian QR algorithm as described in Section 3.4, Chapter 4.

`stabrefine.m` MATLAB implementation of Algorithm 4.32 for refining an approximate stable invariant subspace of a Hamiltonian matrix.

Bibliography

[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.

[2] P.-A. Absil and P. Van Dooren. Two-sided Grassmann Rayleigh quotient iteration, 2002.

[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 and V. Mehrmann. On Hamiltonian and symplectic Hessenberg forms. *Linear Algebra Appl.*, 149:55–72, 1991.

[7] E. Anderson, Z. Bai, C. 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.

[8] 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.

[9] 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.

[10] P. Arbenz and Z. Drmač. On positive semidefinite matrices with known null space. *SIAM J. Matrix Anal. Appl.*, 24(1):132–149, 2002.

[11] U. M. Ascher, R. M. M. Mattheij, and R. D. Russell. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, volume 13 of *Classics in Applied Mathematics*. SIAM, Philadelphia, PA, 1995.

[12] 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`.

[13] 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.

[14] Z. Bai and J. W. Demmel. On swapping diagonal blocks in real Schur form. *Linear Algebra Appl.*, 186:73–95, 1993.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] S. Batterson. Convergence of the shifted QR algorithm on 3×3 normal matrices. *Numer. Math.*, 58(4):341–352, 1990.

[20] S. Batterson and J. Smillie. The dynamics of Rayleigh quotient iteration. *SIAM J. Numer. Anal.*, 26(3):624–636, 1989.

[21] S. Batterson and J. Smillie. Rayleigh quotient iteration for nonsymmetric matrices. *Math. Comp.*, 55:169–178, 1990.

[22] H. Baumgärtel. *Endlichdimensionale Analytische Störungstheorie.* Akademie-Verlag, Berlin, 1972.

[23] 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.

[24] 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.

[25] P. Benner. Computational methods for linear-quadratic optimization. *Supplemento ai Rendiconti del Circolo Matematico di Palermo, Serie II*, No. 58:21–56, 1999.

[26] P. Benner. Symplectic balancing of Hamiltonian matrices. *SIAM J. Sci. Comput.*, 22(5):1885–1904, 2000.

[27] 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.

[28] 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.

[29] P. Benner, R. Byers, V. Mehrmann, and H. Xu. Robust numerical methods for robust control. In preparation, 2003.

[30] P. Benner and H. Faßbender. An implicitly restarted symplectic Lanczos method for the Hamiltonian eigenvalue problem. *Linear Algebra Appl.*, 263:75–111, 1997.

[31] P. Benner and H. Faßbender. A hybrid method for the numerical solution of discrete-time algebraic Riccati equations. *Contemporary Mathematics*, 280:255–269, 2001.

[32] P. Benner and D. Kressner. Balancing sparse Hamiltonian eigenproblems, 2003. To appear in *Linear Algebra Appl.*

[33] P. Benner and D. Kressner. Fortran 77 subroutines for computing the eigenvalues of Hamiltonian matrices II. In preparation. See also `http://www.math.tu-berlin.de/~kressner/hapack/`, 2004.

[34] P. Benner, D. Kressner, and V. Mehrmann. Skew-Hamiltonian and Hamiltonian eigenvalue problems: Theory, algorithms and applications, 2003. Submitted. Online available from `http://www.math.tu-berlin.de/~kressner/`.

[35] 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.

[36] 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.

[37] 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.

[38] P. Benner, V. Mehrmann, and H. Xu. A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils. *Numerische Mathematik*, 78(3):329–358, 1998.

[39] 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.

[40] 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.

[41] R. Bhatia. *Matrix Analysis*. Springer-Verlag, New York, 1997.

[42] C. Bischof and C. F. Van Loan. The *WY* representation for products of Householder matrices. *SIAM J. Sci. Statist. Comput.*, 8(1):S2–S13, 1987. Parallel processing for scientific computing (Norfolk, Va., 1985).

[43] C. H. Bischof, B. Lang, and X. Sun. A framework for symmetric band reduction. *ACM Trans. Math. Software*, 26(4):581–601, 2000.

[44] Å. Björck. *Numerical Methods for Least Squares Problems.* SIAM, Philadelphia, PA, 1996.

[45] A. Bojanczyk and P. Van Dooren. *On propagating orthogonal transformations in a product of 2x2 triangular matrices*, pages 1–9. de Gruyter, 1993.

[46] 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.

[47] 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 *Lecture Notes in Computational Science and Engineering*, pages 293–300, Berlin, Heidelberg, London, 2001. Springer Verlag.

[48] 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.

[49] 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.

[50] 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.

[51] 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.

[52] J. R. Bunch. The weak and strong stability of algorithms in numerical linear algebra. *Linear Algebra Appl.*, 88/89:49–66, 1987.

[53] A. Bunse-Gerstner. Matrix factorizations for symplectic *QR*-like methods. *Linear Algebra Appl.*, 83:49–77, 1986.

[54] 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.

[55] 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.

[56] 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.

[57] J. V. Burke, A. S. Lewis, and M. L. Overton. Pseudospectral components and the distance to uncontrollability, 2003. Submitted to SIAM J. Matrix Anal. Appl.

[58] 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.

[59] R. Byers. *Hamiltonian and Symplectic Algorithms for the Algebraic Riccati Equation.* PhD thesis, Cornell University, Dept. Comp. Sci., Ithaca, NY, 1983.

[60] R. Byers. A LINPACK-style condition estimator for the equation $AX - XB^T = C$. *IEEE Trans. Automat. Control*, 29(10):926–928, 1984.

[61] R. Byers. A Hamiltonian QR algorithm. *SIAM J. Sci. Statist. Comput.*, 7(1):212–229, 1986.

[62] R. Byers. A bisection method for measuring the distance of a stable to unstable matrices. *SIAM J. Sci. Statist. Comput.*, 9:875–881, 1988.

[63] R. Byers. A Hamiltonian-Jacobi algorithm. *IEEE Trans. Automat. Control*, 35:566–570, 1990.

[64] R. Byers, 2004. Personal communication.

[65] R. Byers and S. Nash. On the singular "vectors" of the Lyapunov operator. *SIAM J. Algebraic Discrete Methods*, 8(1):59–66, 1987.

[66] 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.

[67] 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 [14].

[68] 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.

[69] F. Chatelin. Simultaneous Newton's iteration for the eigenproblem. In *Defect correction methods (Oberwolfach, 1983)*, volume 5 of *Comput. Suppl.*, pages 67–74. Springer, Vienna, 1984.

[70] F. Chatelin. *Eigenvalues of matrices.* John Wiley & Sons Ltd., Chichester, 1993.

[71] T.-Y. Chen. Balancing sparse matrices for computing eigenvalues. Master's thesis, University of California at Berkeley, 1998.

[72] T.-Y. Chen and J. W. Demmel. Balancing sparse matrices for computing eigenvalues. *Linear Algebra Appl.*, 309:261–287, 2000.

[73] 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.

[74] 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.

[75] L. Dai. *Singular control systems*, volume 118 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin, 1989.

[76] 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.

[77] 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.

[78] J. W. Demmel. Computing stable eigendecompositions of matrices. *Linear Algebra Appl.*, 79:163–193, 1986.

[79] J. W. Demmel. Three methods for refining estimates of invariant subspaces. *Computing*, 38:43–57, 1987.

[80] J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997.

[81] J. W. Demmel and B. Kågström. Computing stable eigendecompositions of matrix pencils. *Linear Algebra Appl.*, 88/89:139–186, 1987.

[82] 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.

[83] 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.

[84] 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.

[85] 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.

[86] 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.

[87] 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.

[88] 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.

[89] A. A. Dubrulle. The multishift QR algorithm–is it worth the trouble? Tr 6320-3558, IBM Scientific Center, Palo Alto, CA, 1991.

[90] A. A. Dubrulle and G. H. Golub. A multishift QR iteration without computation of the shifts. *Numer. Algorithms*, 7(2-4):173–181, 1994.

[91] A. A. Dubrulle, R. S. Martin, and J. H. Wilkinson. The implicit QL algorithm. *Numerische Mathematik*, 12:377–383, 1968. Also in [265, pp.241–248].

[92] I. S. Duff and J. K. Reid. An implementation of Tarjan's algorithm for the block triangularization of a matrix. *ACM Trans. Math. Software*, 4:137–147, 1978.

[93] 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.

[94] R. Eising. Between controllable and uncontrollable. *Systems Control Lett.*, 4(5):263–264, 1984.

[95] O. G. Ernst. Equivalent iterative methods for p-cyclic matrices. *Numer. Algorithms*, 25(1-4):161–180, 2000.

[96] 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.

[97] 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.

[98] 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.

[99] L. V. Foster. Gaussian elimination with partial pivoting can fail in practice. *SIAM J. Matrix Anal. Appl.*, 15(4):1354–1362, 1994.

[100] J. G. F. Francis. The QR transformation, parts I and II. *Computer Journal*, 4:265–271, 332–345, 1961, 1962.

[101] 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.

[102] 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 [103].

[103] 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.

[104] 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.

[105] F.R. Gantmacher. *The Theory of Matrices.* Chelsea, New York, 1960.

[106] J. D. Gardiner. Stabilizing control for second-order models and positive real systems. *AIAA J. Guidance, Dynamics and Control*, 15(1):280–282, 1992.

[107] 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.

[108] I. Gohberg, P. Lancaster, and L. Rodman. *Matrix polynomials.* Academic Press Inc. [Harcourt Brace Jovanovich Publishers], New York, 1982. Computer Science and Applied Mathematics.

[109] 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, 1999/00.

[110] 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.

[111] G. H. Golub and H. van der Vorst. Eigenvalue computation in the 20th century. *J. Comput. Appl. Math.*, 123(1-2):35–65, 2000.

[112] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[113] J. Grad. Matrix balancing. *Comput. J.*, 14:280–284, 1971.

[114] M. Green and D. J. N. Limebeer. *Linear Robust Control.* Prentice-Hall, Englewood Cliffs, NJ, 1995.

[115] M. Gu. New methods for estimating the distance to uncontrollability. *SIAM J. Matrix Anal. Appl.*, 21(3):989–1003, 2000.

[116] C.-H. Guo and P. Lancaster. Analysis and modification of Newton's method for algebraic Riccati equations. *Math. Comp.*, 67:1089–1105, 1998.

[117] D. J. Hartfiel. Concerning diagonal similarity of irreducible matrices. *Proc. Amer. Math. Soc.*, 30:419–425, 1971.

[118] 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.

[119] 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.

[120] 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.

[121] 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.

[122] N. J. Higham. *Accuracy and Stability of Numerical Algorithms.* SIAM, Philadelphia, PA, 1996.

[123] N. J. Higham, M. Konstantinov, V. Mehrmann, and P. Petkov. Sensitivity of computational control problems. Numerical Analysis Report No. 424, Manchester Centre for Computational Mathematics, Manchester, England, February 2003. To appear in IEEE Control Systems Magazine.

[124] A. S. Hodel and P. Misra. Least-squares approximate solution of overdetermined Sylvester equations. *SIAM J. Matrix Anal. Appl.*, 18(2):279–290, 1997.

[125] R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, Cambridge, 1985.

[126] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis.* Cambridge University Press, Cambridge, 1991.

[127] K. Hüper and P. Van Dooren. New algorithms for the iterative refinement of estimates of invariant subspaces. *Journal Future Generation Computer Systems*, 19:1231–1242, 2003.

[128] T.-M. Hwang, W.-W. Lin, and V. Mehrmann. Numerical solution of quadratic eigenvalue problems with structure-preserving methods. *SIAM J. Sci. Comput.*, 24(4):1283–1302, 2003.

[129] 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 [237].

[130] 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.

[131] 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.

[132] B. Kågström. A direct method for reordering eigenvalues in the generalized real Schur form of a regular matrix pair (A, B). In *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.

[133] 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.

[134] 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.

[135] 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.

[136] 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.

[137] B. Kalantari, L. Khachiyan, and A. Shokoufandeh. On the complexity of matrix balancing. *SIAM J. Matrix Anal. Appl.*, 18(2):450–463, 1997.

[138] M. Karow. *Geometry of Spectral Value Sets.* PhD thesis, Universität Bremen, Fachbereich 3 (Mathematik & Informatik), Bremen, Germany, 2003.

[139] L. Kaufman. The LZ-algorithm to solve the generalized eigenvalue problem. *SIAM J. Numer. Anal.*, 11:997–1024, 1974.

[140] L. Kaufman. Some thoughts on the QZ algorithm for solving the generalized eigenvalue problem. *ACM Trans. Math. Software*, 3(1):65–75, 1977.

[141] L. Kaufman. A parallel QR algorithm for the symmetric tridiagonal eigenvalue problem. *Journal of Parallel and Distributed Computing*, Vol 3:429–434, 1994.

[142] D. L. Kleinman. On an iterative technique for Riccati equation computations. *IEEE Trans. Automat. Control*, AC-13:114–115, 1968.

[143] 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.

[144] 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.

[145] S. G. Krantz. *Function Theory of Several Complex Variables.* John Wiley & Sons Inc., New York, 1982.

[146] D. Kressner. An efficient and reliable implementation of the periodic QZ algorithm. In *IFAC Workshop on Periodic Control Systems*, 2001.

[147] D. Kressner. Numerical methods for structured matrix factorizations, 2001. Diploma thesis, Fakultät für Mathematik, TU Chemnitz.

[148] D. Kressner. Block algorithms for orthogonal symplectic factorizations. *BIT*, 43(4):775–790, 2003.

[149] D. Kressner. The periodic QR algorithm is a disguised QR algorithm, 2003. To appear in *Linear Algebra Appl.*

[150] D. Kressner. Perturbation bounds for isotropic invariant subspaces of skew-Hamiltonian matrices, 2003. To appear in *SIAM J. Matrix Anal. Appl.*. Online available from `http:/www.math.tu-berlin.de/~kressner/`.

[151] D. Kressner. A Matlab toolbox for solving skew-Hamiltonian and Hamiltonian eigenvalue problems, 2004. Online available from `http:/www.math.tu-berlin.de/~kressner/hapack/matlab/`.

[152] 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.

[153] V. N. Kublanovskaya. *AB*-algorithm and its modifications for the spectral problems of linear pencils of matrices. *Numer. Math.*, 43(3):329–342, 1984.

[154] P. Lancaster and L. Rodman. *The Algebraic Riccati Equation*. Oxford University Press, Oxford, 1995.

[155] B. Lang. Effiziente Orthogonaltransformationen bei der Eigen- und Singulärwertzerlegung. Habilitationsschrift, 1997.

[156] B. Lang. Using level 3 BLAS in rotation-based algorithms. *SIAM J. Sci. Comput.*, 19(2):626–634, 1998.

[157] A. J. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Trans. Automat. Control*, AC-24:913–921, 1979.

[158] 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.

[159] 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.

[160] 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.

[161] 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.

[162] B. J. Leimkuhler and E. S. Van Vleck. Orthosymplectic integration of linear Hamiltonian systems. *Numer. Math.*, 77(2):269–282, 1997.

[163] D. Lemonnier and P. Van Dooren. Balancing regular matrix pencils, 2004. Submitted to *SIAM J. Matrix Anal. Appl.*

[164] 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.

[165] 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.

[166] W.-W. Lin and J.-G. Sun. Perturbation analysis for the eigenproblem of periodic matrix pairs. *Linear Algebra Appl.*, 337:157–187, 2001.

[167] 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.

[168] 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.

[169] T. A. Manteuffel. Adaptive procedure for estimating parameters for the non-symmetric Tchebychev iteration. *Numer. Math.*, 31(2):183–208, 1978.

[170] R. S. Martin, G. Peters, and J. H. Wilkinson. The QR algorithm for real Hessenberg matrices. *Numerische Mathematik*, 14:219–231, 1970. Also in [265, pp.359–371].

[171] The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, Mass, 01760. *The* MATLAB *Control Toolbox, Version 5*, 2000.

[172] K. Mehlhorn and S. Näher. *LEDA*. Cambridge University Press, Cambridge, 1999. A platform for combinatorial and geometric computing.

[173] 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.

[174] V. Mehrmann and E. Tan. Defect correction methods for the solution of algebraic Riccati equations. *IEEE Trans. Automat. Control*, 33(7):695–698, 1988.

[175] 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.

[176] 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 (electronic), 1996.

[177] 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.

[178] C. B. Moler. Cleve's corner: MATLAB incorporates LAPACK, 2000. See `http://www.mathworks.com/company/newsletter/win00/index.shtml`.

[179] C. B. Moler and G. W. Stewart. An algorithm for generalized matrix eigenvalue problems. *SIAM J. Numer. Anal.*, 10:241–256, 1973.

[180] L. A. Nazarova. Representations of quivers of infinite type. *Izv. Akad. Nauk SSSR Ser. Mat.*, 37:752–791, 1973.

[181] 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).

[182] 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 [14].

[183] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, New York, 1970.

[184] E. E. Osborne. On preconditioning of matrices. *Journal of the ACM*, 7:338–345, 1960.

[185] C. C. Paige. Bidiagonalization of matrices and solutions of the linear equations. *SIAM J. Numer. Anal.*, 11:197–209, 1974.

[186] C. C. Paige and C. F. Van Loan. A Schur decomposition for Hamiltonian matrices. *Linear Algebra Appl.*, 41:11–32, 1981.

[187] B. N. Parlett. *The Symmetric Eigenvalue Problem*, volume 20 of *Classics in Applied Mathematics.* SIAM, Philadelphia, PA, 1998. Corrected reprint of the 1980 original.

[188] B. N. Parlett and J. Le. Forward instability of tridiagonal QR. *SIAM J. Matrix Anal. Appl.*, 14(1):279–316, 1993.

[189] B. N. Parlett and C. Reinsch. Balancing a matrix for calculation of eigenvalues and eigenvectors. *Numerische Mathematik*, 13:293–304, 1969. Also in [265, pp.315–326].

[190] G. Peters and J. H. Wilkinson. Inverse iteration, ill-conditioned equations and Newton's method. *SIAM Rev.*, 21(3):339–360, 1979.

[191] P. H. Petkov, N. D. Christov, and M. M. Konstantinov. *Computational Methods for Linear Control Systems.* Prentice-Hall, Hertfordshire, UK, 1991.

[192] 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.

[193] A. C. M. Ran and L. Rodman. Stability of invariant maximal semidefinite subspaces. I. *Linear Algebra Appl.*, 62:51–86, 1984.

[194] 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.

[195] 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.

[196] 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.

[197] A. Ruhe. An algorithm for numerical determination of the structure of a general matrix. *BIT*, 10:196–216, 1969.

[198] H. Rutishauser. Solution of eigenvalue problems with the LR transformation. *Nat. Bur. Stand. App. Math. Ser.*, 49:47–81, 1958.

[199] Y. Saad. On the rates of convergence of the Lanczos and the block Lanczos methods. *SIAM Journal on Numerical Analysis*, 17:687–706, 1980.

[200] Y. Saad. Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems. *Math. Comp.*, 42(166):567–588, 1984.

[201] Y. Saad. *Numerical Methods for Large Eigenvalue Problems: Theory and Algorithms.* John Wiley, New York, 1992.

[202] 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.

[203] V. V. Sergeichuk. Canonical matrices for linear matrix problems. *Linear Algebra Appl.*, 317(1-3):53–102, 2000.

[204] V. V. Sergeichuk. Computation of canonical matrices for chains and cycles of linear mappings. *Linear Algebra Appl.*, 376:235–263, 2004.

[205] V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics.* Marcel Dekker, Inc., New York, NY, 1996.

[206] 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.

[207] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Statist.*, 35:876–879, 1964.

[208] P. Smit. *Numerical Analysis of Eigenvalue Algorithms Based on Subspace Iterations.* PhD thesis, Catholic University Brabant, The Netherlands, 1997.

[209] 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.

[210] R. L. Smith. Algorithm 116: Complex division. *Comm. ACM*, 5(8):435, 1962.

[211] 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.

[212] D. C. Sorensen. Implicit application of polynomial filters in a k-step Arnoldi method. *SIAM J. Matrix Anal. Appl.*, 13:357–385, 1992.

[213] 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.

[214] D. C. Sorensen. Passivity preserving model reduction via interpolation of spectral zeros. Technical report TR02-15, ECE-CAAM Depts, Rice University, 2002.

[215] 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.

[216] 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.

[217] G. W. Stewart. Error bounds for approximate invariant subspaces of closed linear operators. *SIAM J. Numer. Anal.*, 8:796–808, 1971.

[218] G. W. Stewart. On the sensitivity of the eigenvalue problem $Ax = \lambda Bx$. *SIAM J. Numer. Anal.*, 9:669–686, 1972.

[219] G. W. Stewart. Error and perturbation bounds for subspaces associated with certain eigenvalue problems. *SIAM Rev.*, 15:727–764, 1973.

[220] G. W. Stewart. Gerschgorin theory for the generalized eigenvalue problem $Ax = \lambda Bx$. *Mathematics of Computation*, 29:600–606, 1975.

[221] 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.

[222] G. W. Stewart. *Matrix Algorithms. Vol. I.* SIAM, Philadelphia, PA, 1998. Basic decompositions.

[223] G. W. Stewart. *Matrix Algorithms. Vol. II.* SIAM, Philadelphia, PA, 2001. Eigensystems.

[224] G. W. Stewart. On the eigensystems of graded matrices. *Numer. Math.*, 90(2):349–370, 2001.

[225] G. W. Stewart. A Krylov-Schur algorithm for large eigenproblems. *SIAM J. Matrix Anal. Appl.*, 23(3):601–614, 2001/02.

[226] G. W. Stewart and J.-G. Sun. *Matrix Perturbation Theory.* Academic Press, New York, 1990.

[227] W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains.* Princeton University Press, Princeton, NJ, 1994.

[228] T. Ström. Minimization of norms and logarithmic norms by diagonal similarities. *Computing (Arch. Elektron. Rechnen)*, 10:1–7, 1972.

[229] T. Stykel. Balanced truncation model reduction for semidiscretized Stokes equation. Technical report 04-2003, Institut für Mathematik, TU Berlin, 2003.

[230] J.-G. Sun. Perturbation analysis of singular subspaces and deflating subspaces. *Numer. Math.*, 73(2):235–263, 1996.

[231] 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.

[232] R. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.

[233] F. Tisseur. Newton's method in floating point arithmetic and iterative refinement of generalized eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 22(4):1038–1057, 2001.

[234] F. Tisseur. Stability of structured Hamiltonian eigensolvers. *SIAM J. Matrix Anal. Appl.*, 23(1):103–125, 2001.

[235] F. Tisseur. A chart of backward errors for singly and doubly structured eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 24(3):877–897, 2003.

[236] F. Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Rev.*, 43(2):235–286, 2001.

[237] 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.

[238] 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.

[239] P. Van Dooren. *Numerical Linear Algebra for Signal Systems and Control.* Draft notes prepared for the Graduate School in Systems and Control, 2003.

[240] C. F. Van Loan. *Generalized Singular Values with Algorithms and Applications.* PhD thesis, The University of Michigan, 1973.

[241] C. F. Van Loan. A general matrix eigenvalue algorithm. *SIAM J. Numer. Anal.*, 12(6):819–834, 1975.

[242] 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.

[243] C. F. Van Loan. A symplectic method for approximating all the eigenvalues of a Hamiltonian matrix. *Linear Algebra Appl.*, 61:233–251, 1984.

[244] A. Varga. A multishift Hessenberg method for pole assignment of single-input systems. *IEEE Trans. Automat. Control*, 41(12):1795–1799, 1996.

[245] A. Varga. Periodic Lyapunov equations: some applications and new algorithms. *Internat. J. Control*, 67(1):69–87, 1997.

[246] A. Varga. Balancing related methods for minimal realization of periodic systems. *Systems Control Lett.*, 36(5):339–349, 1999.

[247] 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.

[248] R. S. Varga. *Matrix Iterative Analysis.* Prentice–Hall, Englewood Cliffs, NJ, 1962.

[249] H. F. Walker. Implementation of the GMRES method using Householder transformations. *SIAM J. Sci. Stat. Comp.*, 9:152–163, 1988.

[250] R. C. Ward. *A numerical solution to the generalized eigenvalue problem.* PhD thesis, University of Virginia, Charlottesville, Va., 1974.

[251] R. C. Ward. The combination shift QZ algorithm. *SIAM J. Numer. Anal.*, 12(6):835–853, 1975.

[252] R. C. Ward. Balancing the generalized eigenvalue problem. *SIAM J. Sci. Statist. Comput.*, 2(2):141–152, 1981.

[253] D. S. Watkins. Some perspectives on the eigenvalue problem. *SIAM Review*, 35:430–471, 1993.

[254] D. S. Watkins. Shifting strategies for the parallel QR algorithm. *SIAM J. Sci. Comput.*, 15(4):953–958, 1994.

[255] D. S. Watkins. Forward stability and transmission of shifts in the QR algorithm. *SIAM J. Matrix Anal. Appl.*, 16(2):469–487, 1995.

[256] D. S. Watkins. The transmission of shifts and shift blurring in the QR algorithm. *Linear Algebra Appl.*, 241/243:877–896, 1996.

[257] D. S. Watkins. Performance of the QZ algorithm in the presence of infinite eigenvalues. *SIAM J. Matrix Anal. Appl.*, 22(2):364–375, 2000.

[258] D. S. Watkins. *Fundamentals of Matrix Computations.* Pure and Applied Mathematics. Wiley-Interscience [John Wiley & Sons], New York, 2002. Second editon.

[259] D. S. Watkins. On Hamiltonian and symplectic Lanczos processes, 2002. To appear in Linear Algebra Appl.

[260] D. S. Watkins and L. Elsner. Chasing algorithms for the eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 12(2):374–384, 1991.

[261] D. S. Watkins and L. Elsner. Convergence of algorithms of decomposition type for the eigenvalue problem. *Linear Algebra Appl.*, 143:19–47, 1991.

[262] 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.

[263] H. Weyl. *The Classical Groups.* Princeton University Press, Princeton, NJ, 1973. 8th Printing.

[264] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Clarendon Press, Oxford, 1965.

[265] J. H. Wilkinson and C. Reinsch. *Handbook for Automatic Computation. Vol. II Linear Algebra.* Springer-Verlag, New York, 1971.

[266] 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.

[267] 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.

[268] H. Xu and L. Z. Lu. Properties of a quadratic matrix equation and the solution of the continuous-time algebraic Riccati equation. *Linear Algebra Appl.*, 222:127–145, 1995.

[269] K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control.* Prentice-Hall, Upper Saddle River, NJ, 1996.

Index