URV decomposition based structured methods for palindromic and even eigenvalue problems

Christian Schröder*

March 14, 2007

Abstract

In this work numerical methods for the solution of two classes of structured generalized eigenvalue problems, $Ax = \lambda Bx$, are developed. Those classes are the palindromic ($B = A^T$) and the even ($A = A^T$, $B = -B^T$) eigenvalue problems. The spectrum of these problems is not arbitrary, rather do eigenvalues occur in pairs. We will construct methods for palindromic and even eigenvalue problems that are of cubic complexity and that are guaranteed to produce eigenvalues that are paired to working precision.

At the heart of both methods is a new URV-type matrix decomposition, that simultaneously transforms three matrices to skew triangular form, i.e., to a form that is triangular with respect to the Northeast-Southwest diagonal. The algorithm to compute this URV decomposition uses several other methods to reduce a single square matrix to skew triangular form: the skew QR factorization and the skew QRQ^T decomposition. Moreover, a method to compute the singular value decomposition of a complex, skew symmetric matrix is presented and used.

MSC2000 classification: 15A18, 15A22, 15A23

Keywords: URV decomposition, palindromic/even eigenvalue problem, structure preserving method, skew QR factorization, skew QRQ^T factorization, skew Takagi factorization

1 Introduction

A generalized eigenvalue problem of the form

$$Ax = \lambda A^T x \tag{1}$$

is called a *palindromic eigenvalue problem*. Here, $A \in \mathbb{C}^{n,n}$ is a given square complex matrix, A^T its transpose, and $x \in \mathbb{C}^n \setminus \{0\}$ and $\lambda \in \mathbb{C}$ denote the wanted eigenvector and eigenvalue, respectively. For the largest part of the paper we will assume that A is indeed complex and we will use complex transformations. However, many applications yield real palindromic problems which are discussed in a separate section. There is a third variant, the $*$-palindromic problem $Ax = \lambda A^* x$ (where A^* is the conjugate transpose of A), but this type is not considered here (see section 3 for details).

Palindromic eigenvalue problems of the form (1) are (up to a sign) the linear case of polynomial palindromic eigenvalue problems

$$P(\lambda)x = \left(\sum_{i=0}^{k} A_i \lambda^i\right) x = 0, \text{ where } A_{k-i}^T = A_i \in \mathbb{C}^{n,n}, \ i = 0, \ldots, k. \tag{2}$$

The underlying matrix polynomial $P(\lambda)$ is invariant under reversing the order of the coefficients (and transposing). This explains the term 'palindromic', as palindromes are words or sentences that are invariant under reversing the order of the letters, like 'mom', 'dad',

*`schroed@math.tu-berlin.de`, Institut für Mathematik, MA 4-5, Technische Universität Berlin, Germany. Supported by Deutsche Forschungsgemeinschaft through MATHEON, the DFG Research Center *Mathematics for key technologies* in Berlin.

'rotor' or 'A man, a plan, a canal, Panama'. Polynomial palindromic eigenvalue problems were introduced and analyzed in [18] and arise for example in the vibration analysis of rail tracks [13].

Here, we only consider linear palindromic problems of the form (1). This is not a severe restriction as polynomial problems of the form (2) can (under mild assumptions) be transformed into linear palindromic problems, see [18].

Another class of structured eigenvalue problems are symmetric/skew symmetric problems of the form

$$Mx = \lambda Nx, \text{ with } M = M^T, \ N = -N^T. \tag{3}$$

Also these problems can be generalized to polynomial problems,

$$P(\lambda)x = \left(\sum_{i=0}^{k} A_i \lambda^i\right) x = 0, \text{ where } A_i^T = (-1)^i A_i \in \mathbb{C}^{n,n}, \ i = 0, \ldots, k. \tag{4}$$

These problems are also called *even*, because $P(-\lambda) = P(\lambda)^T$. A polynomial even eigenvalue problem can again (under mild assumptions) be transformed into a linear even eigenvalue problem (3), see [18].

Even and palindromic eigenvalue problems are closely related by the generalized Cayley transformation [18]. It rewrites the generalized eigenvalue problem $Ax = \lambda Bx$ as

$$(A + B)x = \frac{\lambda + 1}{\lambda - 1}(A - B)x.$$

The generalized Cayley transform of a palindromic problem is even and vice versa.

The structure in the coefficient matrices of (1) and (3) results in a symmetry in the spectrum. Indeed, transposing the palindromic problem (1) we have $x^T A = \frac{1}{\lambda} x^T A^T$. So, if λ is an eigenvalue (and x an associated eigenvector), then so is $\frac{1}{\lambda}$ (with x^T as left eigenvector). This pairing also holds for a zero eigenvalue - its counterpart is an infinite eigenvalue (from here on, we will use the convention $\frac{1}{0} = \infty$ in order to unify the treatment of finite and infinite eigenvalues). Also the number and sizes of Jordan blocks corresponding to the eigenvalues λ and $\frac{1}{\lambda}$ coincide. This follows from structured Kronecker canonical forms for palindromic eigenvalue problems as presented in [21] or (as canonical forms under congruence) in [15, 16, 20].

Analogously, transposing (3) yields $x^T M = -\lambda x^T N$. Hence, the eigenvalues of an even problem come in pairs $\pm\lambda$. Canonical forms of a symmetric/ skew symmetric pencil under congruence (see [25] and the references therein) show that also the number and sizes of Jordan blocks corresponding to λ and $-\lambda$ coincide.

The spectral symmetries of palindromic and even eigenvalue problems call for efficient algorithms whose results comply with these symmetries. But, due to rounding errors, the spectrum computed by standard methods for the generalized eigenvalue problem (like the QZ algorithm) is in general not structured. Existing methods for the palindromic eigenvalue problem include a QR-like algorithm [22], a Jacobi-style method [13], and the Laub trick, a postprocessing step of the generalized Schur form [19]. However, the latter method works well, only if there are no eigenvalues near ± 1. The Jacobi method typically needs a multitude of computing time compared to the qz algorithm. It (probably) is of complexity $\mathcal{O}(n^3 \log(n))$, and the QR-like algorithm is of complexity $\mathcal{O}(n^4)$ in general, reducing to $\mathcal{O}(n^3)$ only if the matrix A in (1) is given in so-called skew Hessenberg form (defined below). A combination of these three methods, however, seems to be quite powerful [19]. In this paper we will present algorithms for both, the palindromic and the even problem that produce eigenvalues that fulfill the spectral symmetry also in finite precision arithmetic. They are of complexity $\mathcal{O}(n^3)$ and need less work (measured in flops) than the QZ algorithm. Both algorithms are based on a URV-type matrix decomposition.

A URV decomposition of a matrix A is a factorization of the form $A = URV^*$ where U and V are unitary and R is triangular. Such a factorization is far from being unique. Thus, it is not surprising that there are several URV decompositions for different applications, each posing special additional restrictions on U, V, and/or R.

The best known variant is probably the *rank revealing URV decomposition* of Stewart [23], [10, sec.12.5.5], where U, V are requested to be efficiently computable and R is of the form $\left[\begin{smallmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{smallmatrix}\right]$ with $R_{11} \in \mathbb{C}^{r,r}$ and $\sigma_{\min}(R_{11}) \gg \sigma_{\max}(R_{22})$. (Here, $\sigma_{\min}(A), \sigma_{\max}(A)$

denote the smallest and largest singular value of a matrix A.) This decomposition can be used to decide on the (numerical) rank of a matrix. In this it is an alternative to the singular value decomposition, but other than the SVD it is efficiently computable by a finite algorithm and it can be easily updated after rank-1 modifications.

Sometimes, this factorization is called *the* URV decomposition. However, we want to stress that in the context of this paper it is just one among many and that other URV decompositions may not reveal the (numerical) rank of a matrix and need not be computable by a finite algorithm.

A predecessor of the rank revealing URV decomposition is the *QR factorization with column pivoting*[6, 10], a URV decomposition where V is a permutation matrix and the elements of R are required to decrease in magnitude along the diagonal.

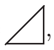
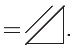
Another variant of a URV decomposition is used in the context of Hamiltonian and symplectic matrices. Here, we define $H \in \mathbb{C}^{2n,2n}$ to be Hamiltonian if $(JH)^T = JH$ where $J = \left[\begin{smallmatrix} 0 & I_n \\ -I_n & 0 \end{smallmatrix}\right]$. We call $S \in \mathbb{C}^{2n,2n}$ symplectic, if $S^T JS = J$. (Again, we could also, but do not, consider the $*$-case, $(JH)^* = JH$, $S^*JS = J$.) The *symplectic URV decomposition* [1, 2, 3] restricts U and V to be unitary and symplectic and R must be of the form

$$R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22}^T \end{bmatrix} \text{ with } R_{11}, R_{22} \in \mathbb{C}^{n,n} \text{ upper triangular.} \tag{5}$$

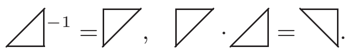
This URV decomposition can be used to solve the Hamiltonian eigenvalue problem. Our new URV decomposition is a generalization of the symplectic URV decomposition (see Section 7 for details). Recently, a postprocessing step for the symplectic URV decomposition was presented that yields a matrix R that is Hamiltonian itself, [8].

This paper is structured as follows. The use of a URV decomposition for the solution of structured eigenvalue problems is motivated in Section 2. This leads to the introduction of a new URV decomposition. Section 3 contains an algorithm to compute this factorization in even dimensional case. In Section 4 the method is generalized to arbitrary dimension. Sections 5 and 6 show how this URV decomposition can be used to solve the even and palindromic eigenvalue problems, respectively. Section 7.1 restricts the scope to real matrices.

Notation A square matrix A is called skew triangular, if $a_{ij} = 0$ whenever $i + j \leq n$. Similarly, a matrix B is called skew Hessenberg, if $b_{ij} = 0$ for $i + j < n$. Such matrices are depicted by

$A =$ , $B =$ .

The inverse of a skew triangular matrix (if it exists) is upper skew triangular. The product of an upper skew triangular matrix and a skew triangular matrix is upper triangular. Such relations are depicted by

 (6)

The direct sum of matrices is defined by $A \oplus B := \begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix}$.Throughout the paper, F denotes the flip matrix,

$$F = \begin{bmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{bmatrix}.$$

If a matrix A is premultiplied by F this causes A to be flipped upside down. Analogously, postmultiplication by F affects a flip leftside right.

We use MATLAB notation to address submatrices, so, $A(i : j, k : l)$ denotes the matrix that consists of the rows i to j of the columns k to l of the matrix A. $A(i : j, :)$ consists of the rows i to j and $A(:, k : l)$ consists of the columns k to l.

Rounding a real scalar α towards the nearest integer in positive or negative direction is denoted by $\lceil \alpha \rceil$ and $\lfloor \alpha \rfloor$, respectively.

In matrix diagrams we denote by ...

x: a potentially non-zero element of a matrix,

0: a zero element of a matrix,

space: same as 0,

y: an element that was affected by the last transformation,

$+$: fill-in, an element that was introduced by the last transformation, but has been zero before,

0: an matrix element that has been zeroed out in the last transformation (this was the reason to do that transform),

gray: rows/columns that have been affected by the last transformation.

2 Motivation

In order to motivate the use of a URV decomposition to solve a structured eigenvalue problem we consider the even eigenvalue problem $Mx = \lambda Nx$, where $M \in \mathbb{C}^{n,n}$ is symmetric and $N \in \mathbb{C}^{n,n}$ is skew symmetric. Further assume that N is non-singular (this implies that n is even).

Premultiplication by N^{-1} and squaring yields

$$Bx := N^{-1}MN^{-1}Mx = \lambda^2 x. \tag{7}$$

Since the eigenvalues of B are the squares of the eigenvalues of (M, N) and since the eigenvalues of (M, N) appear in pairs, we have

$$\Lambda(M,N) = \{\pm\sqrt{\mu}\,|\mu \in \Lambda(B)\}. \tag{8}$$

We are now solving the eigenvalue problem for B without explicitly forming the product or the inverses therein. To this aim assume we could determine unitary matrices $U, V \in \mathbb{C}^{n,n}$ such that

$$U^T MV =: R = \text{◢}, \quad U^T NU =: T = \text{◢}, \quad V^T NV =: P = \text{◢}. \tag{9}$$

Carrying out a similarity transformation with such a V on B results in

$$\begin{aligned}\tilde{B} = V^{-1}BV &= V^{-1}N^{-1}V^{-T}V^T MUU^{-1}N^{-1}U^{-T}U^T MV \\ &= (V^T NV)^{-1}(U^T MV)^T(U^T NU)^{-1}(U^T MV) \\ &= P^{-1}R^T T^{-1}R \\ &= \text{◢}^{-1}\cdot\text{◢}^{T}\cdot\text{◢}^{-1}\cdot\text{◢} \\ &= \text{◤}\cdot\text{◢}\cdot\text{◤}\cdot\text{◢} \\ &= \text{◥}.\end{aligned}$$

This means that V transforms B to Schur form and thus reveals the spectrum of B. The eigenvalues of B are the diagonal entries of $\tilde{B}$, which only depend on the skew diagonal entries of R, T, and P. Note, that B only has double eigenvalues, because T and P are skew symmetric. Indeed, (with $j := n + 1 - i$)

$$\tilde{b}_{ii} = \frac{r_{ij}r_{ji}}{p_{ji}t_{ji}} = \frac{r_{ji}r_{ij}}{(-p_{ij})(-t_{ij})} = \tilde{b}_{jj}.$$

This means that the eigenvalues of (M, N) can be read off the decomposition (9). Using (8) we have

$$\Lambda(M,N) = \left\{\pm\sqrt{\frac{r_{ij}r_{ji}}{p_{ji}t_{ji}}}\,\middle|\, i = 1,\ldots,\frac{n}{2}, j = n+1-i\right\}. \tag{10}$$

Note, that the eigenvalues are paired as desired. Section 5 shows how to handle the case when N is singular and how to extract eigenvectors form the transformation matrices U and V.

Summarizing, we use the URV–like decomposition (9) as structured generalized *periodic Schur form* [4, 12, 17, 26] for the related problem (7).

This motivates the following definition. Let $A, N, S \in \mathbb{C}^{n,n}$ be three given square matrices with N, S skew symmetric. Unitary matrices $U, V \in \mathbb{C}^{n,n}$ are said to define a *skew URV decomposition* of (A, N, S) if

$$U^T AV = \triangle, \quad U^T NU = \triangle, \quad V^T SV = \triangle. \tag{11}$$

Equations (11) can be interpreted as a URV decomposition of A (as $A = \bar{U}RV^*$ with skew triangular R) under the restrictions that $U^T NU$ and $V^T NV$ are skew triangular.

Here the matrix A plays the role of the symmetric matrix M in the motivating example (9), but A is not assumed to be symmetric, and even if it is, symmetry will be lost for $U^T AV$.

3 A special URV decomposition – the complex, even dimensional case

In the following we present a method to compute a skew URV decomposition (11) for given A, N, S. In this section, we restrict ourselves to even dimensional matrices in order to simplify the presentation. Later, the method is modified to arbitrary n reusing much of the material in this section.

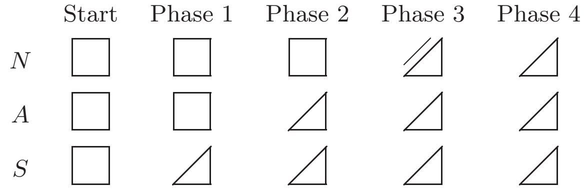
Our algorithm for the computation of decomposition (11) for even n can be divided into four phases. First, S is transformed to skew triangular form. To this aim, let V_1 be a unitary matrix such that $S_1 = V_1^T SV_1$ is skew triangular. Such an V_1 is generated during the skew QRQ^T decomposition which is described in Section 3.1. This V_1 has to be applied to A, too, yielding $A_1 = AV_1$.

Then, A_1 is skew triangularized while keeping S_1 unchanged. This can be accomplished by a skew QR decomposition, discussed in Section 3.2. The result is a unitary matrix U_2 such that $A_2 = U_2^T A_1$ is skew triangular. Subsequently, U_2 has to be applied to N giving $N_2 = U_2^T NU_2$.

In the third phase, N_2 is transformed to skew Hessenberg form without destroying the skew triangular form of A_2 and S_1, i.e., two unitary matrices U_3, V_3 have to be determined such that $N_3 = U_3^T N_2 U_3$ is in skew Hessenberg form, while $A_3 = U_3^T A_2 V_3$ and $S_3 = V_3^T S_1 V_3$ are skew triangular. Section 3.3 presents a process resulting in such U_3, V_3.

Finally, in the fourth phase all, N_3, A_3, and S_3, are transformed to skew triangular form. In Section 3.4 we describe a method to compute unitary matrices U_4, V_4 such that $N_4 := U_4^T N_3 U_4$, $A_4 := U_4^T A_3 V_4$, and $S_4 := V_4^T S_3 V_4$ are skew triangular. This is the only phase where iterative methods have to be used.

Choosing $U = U_2U_3U_4$ and $V = V_1V_3V_4$ we obtain (11), as requested. Summarizing, the forms of N, A, S after accomplishing the various phases are depicted in the following table.



Note, that Phases 1 and 3 make heavy use of the fact that skew symmetric matrices have zeros on their diagonal. Because of that, there is no analogous algorithm for the case that N and S are skew Hermitian matrices, which may have non-zero (purely imaginary) entries on their diagonal.

On the other hand, if A, N and S are real, an almost identical algorithm can be derived that stays in real arithmetic. Section 7.1 discusses the necessary changes.

3.1 Phase 1: Skew QRQ^T factorization

In this section, we describe a method to compute a decomposition of a skew symmetric matrix S of the form

$$S = QRQ^T,$$

where Q is unitary, and R is skew triangular.

This can be achieved by a series of Householder transformations. The process is demonstrated for an 8-by-8 matrix.

$$S = \left[\begin{array}{cccc|cccc} 0 & x & x & x & x & x & x & x \\ x & 0 & x & x & x & x & x & x \\ x & x & 0 & x & x & x & x & x \\ x & x & x & 0 & x & x & x & x \\ \hline x & x & x & x & 0 & x & x & x \\ x & x & x & x & x & 0 & x & x \\ x & x & x & x & x & x & 0 & x \\ x & x & x & x & x & x & x & 0 \end{array}\right]$$

Let $\tilde{H}_1$ be a Householder reflector, such that $\tilde{H}_1 S(2:8,1) = \alpha_1 e_7$. Set $H_1 := 1 \oplus \tilde{H}_1$. Then

$$S_1 := H_1 S H_1^T = \left[\begin{array}{cccc|cccc} 0 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & y \\ \mathbf{0} & 0 & y & y & y & y & y & y \\ \mathbf{0} & y & 0 & y & y & y & y & y \\ \mathbf{0} & y & y & 0 & y & y & y & y \\ \hline \mathbf{0} & y & y & y & 0 & y & y & y \\ \mathbf{0} & y & y & y & y & 0 & y & y \\ \mathbf{0} & y & y & y & y & y & 0 & y \\ y & y & y & y & y & y & y & 0 \end{array}\right].$$

The remainder of the process consists of the recursive application of the scheme to the submatrix $S_1(2:7,2:7)$: so let $\tilde{H}_2$ be a Householder matrix that reflects $S_1(3:7,2)$ to a multiple of e_5 and define $H_2 := I_2 \oplus \tilde{H}_2 \oplus 1$. Then

$$S_2 := H_2 H_1 S H_1^T H_2^T = \left[\begin{array}{cccc|cccc} & & & & & & & x \\ & 0 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & y & x \\ & \mathbf{0} & 0 & y & y & y & y & y \\ & \mathbf{0} & y & 0 & y & y & y & y \\ \hline & \mathbf{0} & y & y & 0 & y & y & y \\ & \mathbf{0} & y & y & y & 0 & y & y \\ & y & y & y & y & y & 0 & y \\ x & x & y & y & y & y & y & 0 \end{array}\right].$$

Finally, defining a Householder reflector, such that $\tilde{H}_3 S_1(4:6,3) = \alpha_3 e_3$ and $H_3 := I_3 \oplus \tilde{H}_3 \oplus I_2$ gives

$$R := H_3 H_2 H_1 S H_1^T H_2^T H_3^T = \left[\begin{array}{cccc|cccc} & & & & & & & x \\ & & & & & & x & x \\ & & 0 & \mathbf{0} & \mathbf{0} & y & x & x \\ & & \mathbf{0} & 0 & y & y & y & y \\ \hline & & \mathbf{0} & y & 0 & y & y & y \\ & & y & y & y & 0 & y & y \\ & x & x & y & y & y & 0 & x \\ x & x & x & y & y & y & x & 0 \end{array}\right].$$

Defining $Q := H_1^* H_2^* H_3^*$, we have $S = QRQ^T$ with skew triangular R, as desired.

A pseudocode for this algorithm is provided in Appendix A.1 (code lines 1–7). The algorithm can be implemented working only on the upper triangular part of S. Then it costs $\frac{5}{6}n^3 + \mathcal{O}(n^2)$ flops to reduce S to skew triangular form. Applying the unitary factor Q to an $n \times k$ matrix costs $n^2 k + \mathcal{O}(nk)$ further operations. Forming Q itself takes $\frac{2}{3}n^3 + \mathcal{O}(n^2)$ flops.

As the method consists only of a sequence of Householder updates of the matrix S standard analysis techniques [10] can be used to show that the computed skew triangular factor is the exact factor of a nearby skew symmetric matrix, i.e., there is a unitary matrix $\tilde{Q}$ such that $\|S - \tilde{Q}R\tilde{Q}^T\|_2 \approx \varepsilon \|S\|_2$ and $\|Q^*Q - I\|_2 \approx \varepsilon$. So the algorithm is strongly backward stable.

3.2 Phase 2: Skew QR factorization

In this subsection we dicuss the factorization of a matrix A of the form

$$A = QR$$

where Q is unitary, and R is skew triangular.

Being a variant of the standard QR factorization, the skew QR factorization can be implemented as a series of $n-1$ Householder reflections in an obvious way: the first reflection, H_1, zeros out all but the last entries in the first column of A yielding A_1; the second reflection, H_2, zeros out all but the last two entries in the second column of A_1 yielding A_2; ...the ith reflection, H_i, zeros out all but the last i entries in the ith column of A_{i-1} yielding A_i, for $i = 1, \ldots, n-1$. Then, with $Q = H_1^* \cdots H_{n-1}^*$ and $R = A_{n-1}$, $A = QR$ is a skew QR factorization.

Also the stability and cost aspects of this method equal those of the standard QR factorization: it is backward stable, and it requires $\frac{4}{3}n^3$ flops to manipulate A, and further $\frac{4}{3}n^3$ flops to generate Q.

Another way to compute a skew QR factorization is to use a QL factorization: if $AF = QL$, where L is lower triangular, then $A = QR$ where $R = LF$ is skew triangular. Also a standard QR factorization can be used if a QL factorization is not at hand (like in MATLAB). If $FA = QR$ (with R upper triangular), then $A = (FQF)(FR)$ is a skew QR factorization.

3.3 Phase 3: URV-Hessenberg reduction

In this subsection we will describe how to transform a skew symmetric matrix N to skew Hessenberg form, while preserving the skew triangular forms of A and $S = -S^T$ by transformations of the form $A \mapsto U^T A V$, $N \mapsto U^T N U$, $S \mapsto V^T S V$. The process is illustrated for an 8-by-8 example. After phase 2 the matrices N, A, S are of the form

$$
\begin{array}{ccc}
N & A & S \\
\left[\begin{array}{cccc|cccc}
0 & x & x & x & x & x & x & x \\
x & 0 & x & x & x & x & x & x \\
x & x & 0 & x & x & x & x & x \\
x & x & x & 0 & x & x & x & x \\
\hline
x & x & x & x & 0 & x & x & x \\
x & x & x & x & x & 0 & x & x \\
x & x & x & x & x & x & 0 & x \\
x & x & x & x & x & x & x & 0
\end{array}\right]
& \triangle &
\left[\begin{array}{cccc|cccc}
 & & & & & & & x \\
 & & & & & & x & x \\
 & & & & & x & x & x \\
 & & & & x & x & x & x \\
\hline
 & & & x & 0 & x & x & x \\
 & & x & x & x & 0 & x & x \\
 & x & x & x & x & x & 0 & x \\
x & x & x & x & x & x & x & 0
\end{array}\right]
\end{array}
$$

In the following, U_{ij} is a rotation in the i,j-plane acting on N as a congruence and on A from the left, while V_{ij} denotes a rotation in the i,j-plane acting on S as a congruence and on A from the right.

We begin by eliminating the (2,1) and (1,2) elements of N by a rotation U_{23} in the (2,3) plane. This rotation, when applied to A from the left, will generate fill-in at position (2,6). This fill-in can be annihilated by a rotation V_{67} applied from the right. This restoring rotation V_{67} has to be applied to S as a congruence affecting fill-in at positions (2,6) and (6,2).

$$
\begin{array}{cccc}
N & & A & S \\
\downarrow U_{23} & \searrow & & \\
\left[\begin{array}{cccc|cccc}
0 & \mathbf{0} & y & x & x & x & x & x \\
\mathbf{0} & 0 & y & y & y & y & y & y \\
y & y & 0 & y & y & y & y & y \\
x & y & y & 0 & x & x & x & x \\
\hline
x & y & y & x & 0 & x & x & x \\
x & y & y & x & x & 0 & x & x \\
x & y & y & x & x & x & 0 & x \\
x & y & y & x & x & x & x & 0
\end{array}\right]
& & \triangle \xrightarrow{V_{67}} &
\left[\begin{array}{cccc|cccc}
 & & & & & & & x \\
 & & & & & + & y & x \\
 & & & & & y & y & x \\
 & & & & x & y & y & x \\
\hline
 & & & x & 0 & y & y & x \\
 & + & y & y & y & 0 & y & y \\
 & y & y & y & y & y & 0 & y \\
x & x & x & x & x & y & y & 0
\end{array}\right]
\end{array}
$$

In order to annihilate these new elements a congruence rotation V_{23} is applied to S. This, in turn, will introduce a non-zero element in A at position (6,2). Now, a further rotation U_{67} can be used to zero out this element again. In general, if a rotation in the $(i, i+1)$-plane, applied from either side, destroys the skew triangular structure of A, a second rotation in the $(n-i, n-i+1)$-plane applied from the other side can be used to restore the skew triangular form.

Applying U_{67} to N does not generate fill-in in N.

$$
\begin{array}{ccc}
N & A & S \\
\hline
 & \swarrow & V_{23}\downarrow \\
\left[\begin{array}{cccc|cccc}
0 & 0 & x & x & x & y & y & x \\
0 & 0 & x & x & x & y & y & x \\
x & x & 0 & x & x & y & y & x \\
x & x & x & 0 & x & y & y & x \\
\hline
x & x & x & x & 0 & y & y & x \\
y & y & y & y & y & 0 & y & y \\
y & y & y & y & y & y & 0 & y \\
x & x & x & x & x & y & y & 0
\end{array}\right]
& \xleftarrow{U_{67}} \quad \triangle &
\left[\begin{array}{cccc|cccc}
 & & & & & & & x \\
 & & & & & \mathbf{0} & y & y \\
 & & & & & y & y & y \\
 & & & & x & x & x & x \\
\hline
 & & & x & 0 & x & x & x \\
 & \mathbf{0} & y & x & x & 0 & x & x \\
 & y & y & x & x & x & 0 & x \\
x & y & y & x & x & x & x & 0
\end{array}\right]
\end{array}
$$

This process is repeated to zero out elements (3,1) and (1,3) (in general: elements $3, \ldots, \frac{n}{2} - 1$ in the first row/column) of N:

$$
\begin{array}{ccc}
N & A & S \\
\hline
\downarrow U_{34} & \searrow & \\
\left[\begin{array}{cccc|cccc}
 & & \mathbf{0} & y & x & x & x & x \\
 & 0 & y & y & x & x & x & x \\
\mathbf{0} & y & 0 & y & y & y & y & y \\
y & y & y & 0 & y & y & y & y \\
\hline
x & x & y & y & 0 & x & x & x \\
x & x & y & y & x & 0 & x & x \\
x & x & y & y & x & x & 0 & x \\
x & x & y & y & x & x & x & 0
\end{array}\right]
& \triangle \quad \xrightarrow{V_{56}} &
\left[\begin{array}{cccc|cccc}
 & & & & & & & x \\
 & & & & & & x & x \\
 & & & & + & y & x & x \\
 & & & & y & y & x & x \\
\hline
 & & + & y & 0 & y & y & y \\
 & & y & y & y & 0 & y & y \\
 & x & x & x & y & y & 0 & x \\
x & x & x & x & y & y & x & 0
\end{array}\right] \\
 & \swarrow & V_{34}\downarrow \\
\left[\begin{array}{cccc|cccc}
 & & & x & y & y & x & x \\
 & 0 & x & x & y & y & x & x \\
 & x & 0 & x & y & y & x & x \\
x & x & x & 0 & y & y & x & x \\
\hline
y & y & y & y & 0 & y & y & y \\
y & y & y & y & y & 0 & y & y \\
x & x & x & x & y & y & 0 & x \\
x & x & x & x & y & y & x & 0
\end{array}\right]
& \xleftarrow{U_{56}} \quad \triangle &
\left[\begin{array}{cccc|cccc}
 & & & & & & & x \\
 & & & & & & x & x \\
 & & & & \mathbf{0} & y & y & y \\
 & & & & y & y & y & y \\
\hline
 & & \mathbf{0} & y & 0 & x & x & x \\
 & & y & y & x & 0 & x & x \\
 & x & y & y & x & x & 0 & x \\
x & x & y & y & x & x & x & 0
\end{array}\right]
\end{array}
$$

Next, we eleminate the elements $(4, 1)$ and $(1, 4)$ of N by a rotation U_{45}. Restoring the skew triangular shape of A results in a rotation V_{45} that does not generate fill-in in S, because the diagonal entries of skew symmetric matrices are necessarily zero.

$$
\begin{array}{ccc}
N & A & S \\
\hline
\downarrow U_{45} & \searrow & \\
\left[\begin{array}{cccc|cccc}
 & & & \mathbf{0} & x & x & x & x \\
 & 0 & x & y & y & x & x & x \\
 & x & 0 & y & y & x & x & x \\
\mathbf{0} & y & y & 0 & y & y & y & y \\
\hline
y & y & y & y & 0 & y & y & y \\
x & x & x & y & y & 0 & x & x \\
x & x & x & y & y & x & 0 & x \\
x & x & x & y & y & x & x & 0
\end{array}\right]
& \triangle \quad \xrightarrow{V_{45}} &
\left[\begin{array}{cccc|cccc}
 & & & & & & & x \\
 & & & & & & x & x \\
 & & & & & x & x & x \\
 & & & & y & y & y & y \\
\hline
 & & & y & 0 & y & y & y \\
 & & x & y & y & 0 & x & x \\
 & x & x & y & y & x & 0 & x \\
x & x & x & y & y & x & x & 0
\end{array}\right]
\end{array}
$$

Now, the second half of the first row/column of N can be reduced leaving just the last two elements non-zero. This is accomplished in an analogous manner as for the first half.

$$N \qquad\qquad A \qquad\qquad S$$

$$\downarrow U_{56} \quad \searrow$$

$$\left[\begin{array}{cccc|cccc} & & & & \mathbf{0} & y & x & x \\ & 0 & x & x & y & y & x & x \\ & x & 0 & x & y & y & x & x \\ & x & x & 0 & y & y & x & x \\ \hline \mathbf{0} & y & y & y & 0 & y & y & y \\ y & y & y & y & y & 0 & y & y \\ x & x & x & x & y & y & 0 & x \\ x & x & x & x & y & y & x & 0 \end{array}\right] \qquad \triangle \quad \xrightarrow{V_{34}} \quad \left[\begin{array}{cccc|cccc} & & & & & & & x \\ & & & & & & x & x \\ & & & & + & y & y & y \\ & & & & y & y & y & y \\ \hline & & + & y & 0 & x & x & x \\ & & y & y & x & 0 & x & x \\ & x & y & y & x & x & 0 & x \\ x & x & y & y & x & x & x & 0 \end{array}\right]$$

$$\swarrow \quad V_{56} \downarrow$$

$$\left[\begin{array}{cccc|cccc} & & & & & x & x & x \\ & 0 & y & y & x & x & x & x \\ & y & 0 & y & y & y & y & y \\ & y & y & 0 & y & y & y & y \\ \hline & x & y & y & 0 & x & x & x \\ x & x & y & y & x & 0 & x & x \\ x & x & y & y & x & x & 0 & x \\ x & x & y & y & x & x & x & 0 \end{array}\right] \quad \xleftarrow{U_{34}} \quad \triangle \qquad \left[\begin{array}{cccc|cccc} & & & & & & & x \\ & & & & & & x & x \\ & & & & \mathbf{0} & y & x & x \\ & & & & y & y & x & x \\ \hline & & \mathbf{0} & y & 0 & y & y & y \\ & & y & y & y & 0 & y & y \\ & x & x & x & y & y & 0 & x \\ x & x & x & x & y & y & x & 0 \end{array}\right]$$

$$\downarrow U_{67} \quad \searrow$$

$$\left[\begin{array}{cccc|cccc} & & & & & \mathbf{0} & y & x \\ & 0 & x & x & x & y & y & x \\ & x & 0 & x & x & y & y & x \\ & x & x & 0 & x & y & y & x \\ \hline & x & x & x & 0 & y & y & x \\ \mathbf{0} & y & y & y & y & 0 & y & y \\ y & y & y & y & y & y & 0 & y \\ x & x & x & x & x & y & y & 0 \end{array}\right] \qquad \triangle \quad \xrightarrow{V_{23}} \quad \left[\begin{array}{cccc|cccc} & & & & & & & x \\ & & & & & + & y & y \\ & & & & & y & y & y \\ & & & & x & x & x & x \\ \hline & & & x & 0 & x & x & x \\ & + & y & x & x & 0 & x & x \\ & y & y & x & x & x & 0 & x \\ x & y & y & x & x & x & x & 0 \end{array}\right]$$

$$\swarrow \quad V_{67} \downarrow$$

$$\left[\begin{array}{cccc|cccc} & & & & & & x & x \\ & 0 & y & y & y & y & y & y \\ & y & 0 & y & y & y & y & y \\ & y & y & 0 & x & x & x & x \\ \hline & y & y & x & 0 & x & x & x \\ & y & y & x & x & 0 & x & x \\ x & y & y & x & x & x & 0 & x \\ x & y & y & x & x & x & x & 0 \end{array}\right] \quad \xleftarrow{U_{23}} \quad \triangle \qquad \left[\begin{array}{cccc|cccc} & & & & & & & x \\ & & & & & \mathbf{0} & y & x \\ & & & & & y & y & x \\ & & & & x & y & y & x \\ \hline & & & x & 0 & y & y & x \\ & \mathbf{0} & y & y & y & 0 & y & y \\ & y & y & y & y & y & 0 & y \\ x & x & x & x & x & y & y & 0 \end{array}\right]$$

At this point, the first row/column of N is in skew Hessenberg form. Note, that the first and last elements of the first row/column of N were not touched during the process so far. Thus, applying this procedure recursively to the (2:n-1,2:n-1) submatrices preserves the just generated zeros. This recursive application yields U, V such that N is in skew Hessenberg form, while A and S are skew triangular, as required.

$$N \qquad A \qquad S$$

$$\downarrow \tilde{U} \qquad \downarrow \qquad \downarrow \tilde{V}$$

$$\left[\begin{array}{cccc|cccc} & & & & & & x & x \\ & & & & & y & x & x \\ & & & & y & y & y & y \\ & & & 0 & y & y & y & y \\ \hline & & y & y & 0 & y & y & y \\ & y & y & y & y & 0 & y & y \\ x & x & y & y & y & y & 0 & x \\ x & x & y & y & y & y & x & 0 \end{array}\right] \qquad \triangle \qquad \left[\begin{array}{cccc|cccc} & & & & & & & x \\ & & & & & & x & x \\ & & & & & y & y & y \\ & & & & y & y & y & y \\ \hline & & & y & 0 & y & y & y \\ & & y & y & y & 0 & y & y \\ & x & y & y & y & y & 0 & x \\ x & x & y & y & y & y & x & 0 \end{array}\right]$$

A pseudocode for general even n can be found in Appendix A.4.

3.4 Phase 4: URV-triangularization

Finally, we will describe how N can be skew triangularized while keeping A and S skew triangular. Note, that, because N is skew symmetric, (N, A, S) can be partitioned as

$$N = \begin{bmatrix} 0 & -(FH)^T \\ FH & N_{22} \end{bmatrix}, A = \begin{bmatrix} 0 & (FR_1)^T \\ FR_3 & A_{22} \end{bmatrix}, S = \begin{bmatrix} 0 & -(FR_2)^T \\ FR_2 & S_{22} \end{bmatrix}, \tag{12}$$

where $H, R_1, R_2, R_3 \in \mathbb{C}^{\frac{n}{2}, \frac{n}{2}}$ and H is upper Hessenberg and R_1, R_2, R_3 are upper triangular. Let Q_1, Q_2 and Z_1, Z_2 be unitary matrices that transform H, R_1, R_2, R_3 to generalized periodic Schur form, i.e.,

$$\begin{aligned} Q_1^* H Z_2 &= T_4 = \diagdown\!\!\rceil, \\ Q_2^* R_1 Z_2 &= T_1 = \diagdown\!\!\rceil, \\ Q_2^* R_2 Z_1 &= T_2 = \diagdown\!\!\rceil, \\ Q_1^* R_3 Z_1 &= T_3 = \diagdown\!\!\rceil. \end{aligned} \tag{13}$$

These can be computed, e.g., by the periodic QZ algorithm [4, 12, 17, 26] applied to the matrix product $HR_1^{-1}R_2R_3^{-1}$. Note, that R_1 and R_3 do not have to be non-singular for the method to work. Note further, that the first step of the periodic QZ algorithm, the reduction to Hessenberg-triangular form is not necessary, because H, R_1, R_2, R_3 are already in this form.

Setting $U = Z_2 \oplus F\bar{Q}_1F$ and $V = Z_1 \oplus F\bar{Q}_2F$, we have

$$\begin{aligned} U^T N U &= \begin{bmatrix} 0 & -Z_2^T (FH)^T F\bar{Q}_1F \\ FQ_1^*FFHZ_2 & \tilde{N}_{22} \end{bmatrix} = \begin{bmatrix} 0 & -(FT_4)^T \\ FT_4 & \tilde{N}_{22} \end{bmatrix} = \begin{bmatrix} 0 & \diagup\!\!\rfloor \\ \diagup\!\!\rfloor & \square \end{bmatrix}, \\ U^T A V &= \begin{bmatrix} 0 & Z_2^T (FR_1)^T F\bar{Q}_2F \\ FQ_1^*FFR_3Z_1 & \tilde{A}_{22} \end{bmatrix} = \begin{bmatrix} 0 & (FT_1)^T \\ FT_3 & \tilde{A}_{22} \end{bmatrix} = \begin{bmatrix} 0 & \diagup\!\!\rfloor \\ \diagup\!\!\rfloor & \square \end{bmatrix}, \\ V^T S V &= \begin{bmatrix} 0 & -Z_1^T (FR_2)^T F\bar{Q}_2F \\ FQ_2^*FFR_2Z_1 & \tilde{S}_{22} \end{bmatrix} = \begin{bmatrix} 0 & -(FT_2)^T \\ FT_2 & \tilde{S}_{22} \end{bmatrix} = \begin{bmatrix} 0 & \diagup\!\!\rfloor \\ \diagup\!\!\rfloor & \square \end{bmatrix}. \end{aligned}$$

Here, $\tilde{N}_{22} = FQ_1^*FN_{22}F\bar{Q}_1F$, $\tilde{A}_{22} = FQ_1^*FA_{22}F\bar{Q}_2F$, and $\tilde{S}_{22} = FQ_2^*FS_{22}F\bar{Q}_2F$.

At this point, phase 4 and with it the URV-decomposition (11) is completed.

4 The general complex case

The process presented in the last section does not work if the matrices are of odd dimension. In this section we will generalize that process to cover problems of any dimension. This generalization consists mainly of a modification of phase one and the introduction of an additional fifth phase.

During the first phase S is now reduced to the form

$$S_1 := V_1^T S V_1 = \begin{array}{c c} & \begin{array}{ccc} m & r & r \end{array} \\ \begin{array}{c} m \\ r \\ r \end{array} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \diagup\!\!\rfloor \\ 0 & \diagup\!\!\rfloor & \square \end{bmatrix} \end{array}, \tag{14}$$

where r, m are such that $n = 2r + m$. Further define $A_1 := AV_1$, as before.

There are several methods to achieve this reduction. If n is even, the skew QRQ^T factorization yields form (14) with $m = 0$. If n is odd, the reduced skew QRQ^T factorization (described in Section 4.1) yields form (14) with $m = 1$. If S is highly rank defective, the skew Takagi factorization can be used. This method is described in Section 4.2 and results in form (14) with $2r = \operatorname{rank}(S)$.

In phase two, A_1 is transformed to skew triangular form. This can be done by the skew QR factorization as in the even case. This results in $A_2 := U_2^T A_1 = \triangle$ and $N_2 := U_2^T N U_2$.

Next, N_2 and A_2 are partitioned according to S_1 as follows:

$$N_2 = \begin{array}{c} \\ r \\ r \\ m \end{array} \!\!\begin{array}{c} \begin{array}{ccc} r & r & m \end{array} \\ \begin{bmatrix} N_{11} & -N_{21}^T & -N_{31}^T \\ N_{21} & N_{22} & -N_{32}^T \\ N_{31} & N_{32} & N_{33} \end{bmatrix} \end{array}, \quad A_2 = \begin{array}{c} \\ r \\ r \\ m \end{array} \!\!\begin{array}{c} \begin{array}{ccc} m & r & r \end{array} \\ \begin{bmatrix} 0 & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \end{array}, \quad S_1 = \begin{array}{c} \\ m \\ r \\ r \end{array} \!\!\begin{array}{c} \begin{array}{ccc} m & r & r \end{array} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & S_{23} \\ 0 & -S_{23}^T & S_{33} \end{bmatrix} \end{array}.$$

Now, the usual phases three and four (described in the Sections 3.3 and 3.4) can be applied to the even dimensional triple

$$\left(\begin{bmatrix} N_{11} & -N_{21}^T \\ N_{21} & N_{22} \end{bmatrix}, \begin{bmatrix} 0 & A_{13} \\ A_{22} & A_{23} \end{bmatrix}, \begin{bmatrix} 0 & S_{23} \\ -S_{23}^T & S_{33} \end{bmatrix} \right).$$

This yields unitary matrices $\tilde{U}_3, \tilde{U}_4, \tilde{V}_3, \tilde{V}_4$. Setting $U_i := \tilde{U}_i \oplus I_m$, $V_i := I_m \oplus \tilde{V}_i$, $i = 3, 4$, we have

$$\begin{array}{rcl} (N_4, A_4, S_4) & := & (U_4^T U_3^T N_2 U_3 U_4,\ U_4^T U_3^T A_2 V_3 V_4,\ V_4^T V_3^T S_1 V_3 V_4) \\ & = & \left(\begin{array}{c} \\ r \\ r \\ m \end{array} \!\!\begin{array}{c} \begin{array}{ccc} r & r & m \end{array} \\ \begin{bmatrix} 0 & \triangle & \square \\ \triangle & \square & \square \\ \square & \square & \square \end{bmatrix} \end{array}, \begin{array}{c} \\ r \\ r \\ m \end{array} \!\!\begin{array}{c} \begin{array}{ccc} m & r & r \end{array} \\ \begin{bmatrix} 0 & 0 & \triangle \\ 0 & \triangle & \square \\ \triangle & \square & \square \end{bmatrix} \end{array}, \begin{array}{c} \\ m \\ r \\ r \end{array} \!\!\begin{array}{c} \begin{array}{ccc} m & r & r \end{array} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \triangle \\ 0 & \triangle & \square \end{bmatrix} \end{array} \right) \end{array} \quad (15)$$

It remains to transform N_4 and S_4 to skew triangular form while preserving the form of A_4. More precisely, unitary matrices U_5, V_5 have to be determined such that

$$\begin{array}{rcl} (N_5, A_5, S_5) & := & (U_5^T N_4 U_5,\ U_5^T A_4 V_5,\ V_5^T S_4 V_5) \\ & = & \left(\begin{array}{c} \\ r \\ m \\ r \end{array} \!\!\begin{array}{c} \begin{array}{ccc} r & m & r \end{array} \\ \begin{bmatrix} 0 & 0 & \triangle \\ 0 & \triangle & \square \\ \triangle & \square & \square \end{bmatrix} \end{array}, \begin{array}{c} \\ r \\ m \\ r \end{array} \!\!\begin{array}{c} \begin{array}{ccc} r & m & r \end{array} \\ \begin{bmatrix} 0 & 0 & \triangle \\ 0 & \triangle & \square \\ \triangle & \square & \square \end{bmatrix} \end{array}, \begin{array}{c} \\ r \\ m \\ r \end{array} \!\!\begin{array}{c} \begin{array}{ccc} r & m & r \end{array} \\ \begin{bmatrix} 0 & 0 & \triangle \\ 0 & 0 & \square \\ \triangle & \square & \square \end{bmatrix} \end{array} \right) \end{array} \quad (16)$$

Note, that the dimensions of the blocks have changed. The procedure is described in Section 4.3 below.

At this point, the URV decomposition is completed.

4.1 Phase 1a: Reduced skew QRQ^T factorization

We will discuss a method to decompose a skew symmetric matrix $S = -S^T \in \mathbb{C}^{n,n}$ of odd dimension n into

$$S = QRQ^T,$$

where Q is unitary and R is of the form (14) with $m = 1$.

The method builds upon the QRQ^T factorization, described in Section 3.1. Indeed, the method can be extended as depicted in the following for an example of size 7. Application of the process in Section 3.1 brings S to skew triangular form.

$$S = \left[\begin{array}{cccc|ccc} 0 & x & x & x & x & x & x \\ x & 0 & x & x & x & x & x \\ x & x & 0 & x & x & x & x \\ x & x & x & 0 & x & x & x \\ \hline x & x & x & x & 0 & x & x \\ x & x & x & x & x & 0 & x \\ x & x & x & x & x & x & 0 \end{array}\right], S_1 = \tilde{Q}S\tilde{Q}^T = \left[\begin{array}{cccc|ccc} & & & & & & x \\ & & & & & x & x \\ & & & & x & x & x \\ & & & 0 & x & x & x \\ \hline & & x & x & 0 & x & x \\ & x & x & x & x & 0 & x \\ x & x & x & x & x & x & 0 \end{array}\right]$$

Now, we will eleminate the skew diagonal elements, starting in the center going outwards. A two sided rotation in the (3,4) plane can be used to eleminate the elements (3,5) and (5,3):

$$S_2 = \left[\begin{array}{cccc|ccc} & & & & & & x \\ & & & & & x & x \\ & & & & \mathbf{0} & y & y \\ & & & & y & y & y \\ \hline & & \mathbf{0} & y & 0 & x & x \\ & x & y & y & x & 0 & x \\ x & x & y & y & x & x & 0 \end{array}\right].$$

Analogously, rotations in the $(i, i+1)$ plane can be used to eleminate the elements $(i, n+1-i)$ and $(n+1-i, i)$, for $i = \frac{n-1}{2} - 1, \frac{n-1}{2} - 2, \ldots, 1$.

$$S_3 = \left[\begin{array}{cccc|ccc} & & & & & & x \\ & & & & & \mathbf{0} & y \\ & & & & & y & y \\ & & & & x & x & x \\ \hline & & & x & 0 & x & x \\ & \mathbf{0} & y & x & x & 0 & x \\ x & y & y & x & x & x & 0 \end{array}\right], R = \left[\begin{array}{cccc|ccc} & & & & & & \mathbf{0} \\ & & & & & & y \\ & & & & & x & x \\ & & & & x & x & x \\ \hline & & & x & 0 & x & x \\ & & x & x & x & 0 & x \\ \mathbf{0} & y & x & x & x & x & 0 \end{array}\right]$$

At this point S is of the form (14) with $m = 1, r = \frac{n-1}{2}$.

A pseudocode for this algorithm is provided in Appendix A.1. The complexity is of the same order as that of the unreduced QRQ^T factorization, as the added reduction is an $\mathcal{O}(n^2)$ process.

4.2 Phase 1b: Skew Takagi factorization

In this section we show how to transform a skew symmetric matrix $S = -S^T \in \mathbb{C}^{n,n}$ to the form

$$QSQ^T = \begin{array}{c} \\ r \\ \\ r \end{array} \begin{array}{c} \begin{array}{ccc} r & \quad & r \end{array} \\ \left[\begin{array}{ccc} & & D \\ & 0 & \\ -D^T & & \end{array}\right] \end{array}, \quad \text{with } D = \diagup \text{ skew diagonal, real, positive.} \tag{17}$$

This factorization may be thought of as a structured version of the singular value decomposition, as $S = U\Sigma V^*$ with $U = Q^*, \Sigma = DF \oplus 0_{n-2r} \oplus FD, V = Q^T F(I_{n-r} \oplus -I_r)$. We call it *skew Takagi factorization.*

The name is inspired by the Takagi factorization [24, according to [5]], a decomposition of a complex symmetric matrix $M = M^T \in \mathbb{C}^{n,n}$ into $M = U\Sigma U^T$, where U is unitary and Σ is real diagonal. This factorization can be seen as a symmetric variant of the singular value decomposition. An algorithm to compute the Takagi factorization was described by Bunse-Gerstner and Gragg, [5].

The process for the skew symmetric case is demonstrated for a 7-by-7 example.

$$S = \begin{bmatrix} 0 & x & x & x & x & x & x \\ x & 0 & x & x & x & x & x \\ x & x & 0 & x & x & x & x \\ x & x & x & 0 & x & x & x \\ x & x & x & x & 0 & x & x \\ x & x & x & x & x & 0 & x \\ x & x & x & x & x & x & 0 \end{bmatrix},$$

We begin by transforming S to skew bidiagonal form. First, the first row and column are reduced to the last entry. Then the last row/column are reduced to the first two entries.

$$
\begin{bmatrix}
 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & y \\
\mathbf{0} & 0 & y & y & y & y & y \\
\mathbf{0} & y & 0 & y & y & y & y \\
\mathbf{0} & y & y & 0 & y & y & y \\
\mathbf{0} & y & y & y & 0 & y & y \\
\mathbf{0} & y & y & y & y & 0 & y \\
y & y & y & y & y & y & 0
\end{bmatrix},
\begin{bmatrix}
 & & & & & & x \\
 & 0 & y & y & y & y & y \\
 & y & 0 & y & y & y & \mathbf{0} \\
 & y & y & 0 & y & y & \mathbf{0} \\
 & y & y & y & 0 & y & \mathbf{0} \\
 & y & y & y & y & 0 & \mathbf{0} \\
x & y & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} &
\end{bmatrix},
$$

Then, this scheme is recursively applied to the $S(2:n-1, 2:n-1)$ submatrix,

$$
\begin{bmatrix}
 & & & & & & x \\
 & & \mathbf{0} & \mathbf{0} & \mathbf{0} & y & x \\
 & \mathbf{0} & 0 & y & y & y & \\
 & \mathbf{0} & y & 0 & y & y & \\
 & \mathbf{0} & y & y & 0 & y & \\
 & y & y & y & y & 0 & \\
x & x & & & & &
\end{bmatrix},
\begin{bmatrix}
 & & & & & & x \\
 & & & & & x & x \\
 & & 0 & y & y & y & \\
 & & y & 0 & y & \mathbf{0} & \\
 & & y & y & 0 & \mathbf{0} & \\
 & x & y & \mathbf{0} & \mathbf{0} & & \\
x & x & & & & &
\end{bmatrix},
$$

yielding S in the form

$$
S_1 = \left[\begin{array}{cccc|ccc}
 & & & & & & x \\
 & & & & & x & x \\
 & & & \mathbf{0} & y & x & \\
 & & \mathbf{0} & 0 & y & & \\
\hline
 & & y & y & 0 & & \\
 & x & x & & & & \\
x & x & & & & &
\end{array}\right].
$$

At this point, the matrix decouples. It can be partitioned as

$$
S_1 = \begin{array}{c} \\ \lceil \frac{n}{2} \rceil \\ \lfloor \frac{n}{2} \rfloor \end{array}
\begin{array}{c}
\begin{array}{cc} \lceil \frac{n}{2} \rceil & \lfloor \frac{n}{2} \rfloor \end{array} \\
\begin{bmatrix} 0 & -(FB)^T \\ FB & 0 \end{bmatrix}
\end{array}
$$

where B is upper bidiagonal. Let $B = U \left[\begin{smallmatrix} \Sigma & 0 \\ 0 & 0 \end{smallmatrix}\right] V^*$ with $\Sigma \in \mathbb{R}^{r,r}$ be the singular value decomposition of B. Then with $Q = V^T \oplus FU^*F$ we have

$$
QS_1Q^T = \begin{array}{c} \\ \lceil \frac{n}{2} \rceil \\ \lfloor \frac{n}{2} \rfloor \end{array}
\begin{array}{c}
\begin{array}{cc} \lceil \frac{n}{2} \rceil & \lfloor \frac{n}{2} \rfloor \end{array} \\
\begin{bmatrix} 0 & -V^T(FB)^T F\bar{U}F \\ FU^*FFBV & 0 \end{bmatrix}
\end{array}
=
\begin{array}{c} \\ r \\ \\ \\ r \end{array}
\begin{array}{c}
\begin{array}{cccc} r & & & r \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & -(F\Sigma)^T \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
F\Sigma & 0 & 0 & 0
\end{bmatrix}
\end{array}.
$$

This matrix is in form (17).

A pseudocode for this algorithm can be found in Appendix A.2. Reducing S to bidiagonal form costs $\frac{4}{3}n^3$ flops. The following SVD computation is neglectable, if an $\mathcal{O}(n^2)$ algorithm (like MRRR, see [27] and the references therein) is used. Generating Q costs another $2n^3$ flops.

Note, that the form (17) is easily permuted to the form (14).

4.3 Phase 5

In this section we discuss how matrices N, A, S of the form (15) are transformed to the form (16).

This process is illustrated for a 7×7 example with $r = 2$, $m = 3$.

$$
\begin{array}{ccc}
N & A & S \\
\hline
\left[\begin{array}{cc|cc|ccc}
 & & & x & x & x & x \\
 & & x & x & x & x & x \\
\hline
 & x & 0 & x & x & x & x \\
x & x & x & 0 & x & x & x \\
\hline
x & x & x & x & 0 & x & x \\
x & x & x & x & x & 0 & x \\
x & x & x & x & x & x & 0
\end{array}\right]
&
\triangle
&
\left[\begin{array}{ccc|cc|cc}
 & & & & & & 0 \\
 & & & & & & 0 \\
 & & & & & & 0 \\
\hline
 & & & & & & x \\
 & & & & & x & x \\
\hline
 & & & & x & 0 & x \\
0 & 0 & 0 & x & x & x & 0
\end{array}\right]
\end{array}
$$

We begin by eliminating elements $(4,1)$ and $(1,4)$ of N by a rotation in the $(4,5)$ plane. Pushing the transformation through A results in a rotation in the $(3,4)$ plane introducing fill in in S at the positions $(7,3)$ and $(3,7)$.

$$
\begin{array}{cccc}
N & A & & S \\
\hline
\downarrow U_{45} & \searrow & & \\
\left[\begin{array}{cc|ccccc}
 & & & \mathbf{0} & y & x & x \\
 & & x & y & y & x & x \\
\hline
 & x & 0 & y & y & x & x \\
\mathbf{0} & y & y & 0 & y & y & y \\
y & y & y & y & 0 & y & y \\
x & x & x & y & y & 0 & x \\
x & x & x & y & y & x & 0
\end{array}\right]
&
\triangle
&
\xrightarrow{V_{34}}
&
\left[\begin{array}{cccccc|c}
 & & & & & & 0 \\
 & & & & & & 0 \\
 & & & & & & + \\
 & & & & & & y \\
 & & & & & x & x \\
\hline
 & & & & x & 0 & x \\
0 & 0 & + & y & x & x & 0
\end{array}\right]
\end{array}
$$

Continuing in this manner, we reduce the first row/column of N to a multiple of e_n. During this process the last row/column of S becomes fully populated.

$$
\begin{array}{cccc}
N & A & & S \\
\hline
\downarrow U_{56} & \searrow & & \\
\left[\begin{array}{cc|ccccc}
 & & & & \mathbf{0} & y & x \\
 & & x & x & y & y & x \\
\hline
 & x & 0 & x & y & y & x \\
 & x & x & 0 & y & y & x \\
\mathbf{0} & y & y & y & 0 & y & y \\
y & y & y & y & y & 0 & y \\
x & x & x & x & y & y & 0
\end{array}\right]
&
\triangle
&
\xrightarrow{V_{23}}
&
\left[\begin{array}{cccccc|c}
 & & & & & & 0 \\
 & & & & & & + \\
 & & & & & & y \\
 & & & & & & x \\
 & & & & & x & x \\
\hline
 & & & & x & 0 & x \\
0 & + & y & x & x & x & 0
\end{array}\right]
\\
\downarrow U_{67} & \searrow & & \\
\left[\begin{array}{cc|ccccc}
 & & & & & \mathbf{0} & y \\
 & & x & x & x & y & y \\
\hline
 & x & 0 & x & x & y & y \\
 & x & x & 0 & x & y & y \\
 & x & x & x & 0 & y & y \\
\mathbf{0} & y & y & y & y & 0 & y \\
y & y & y & y & y & y & 0
\end{array}\right]
&
\triangle
&
\xrightarrow{V_{12}}
&
\left[\begin{array}{cccccc|c}
 & & & & & & + \\
 & & & & & & y \\
 & & & & & & x \\
 & & & & & & x \\
 & & & & & x & x \\
\hline
 & & & & x & 0 & x \\
+ & y & x & x & x & x & 0
\end{array}\right]
\end{array}
$$

Now apply the same procedure to the remainder of the first r rows/columns of N.

$$
\begin{array}{cccc}
N & A & & S \\
\hline
\downarrow U_{34}, U_{45}, U_{56} & \searrow & & \\
\left[\begin{array}{cc|ccc|cc}
 & & & & & & x \\
 & & \mathbf{0} & \mathbf{0} & \mathbf{0} & y & x \\
\hline
 & \mathbf{0} & 0 & y & y & y & y \\
 & \mathbf{0} & y & 0 & y & y & y \\
 & \mathbf{0} & y & y & 0 & y & y \\
\hline
 & y & y & y & y & 0 & y \\
x & x & y & y & y & y & 0
\end{array}\right]
&
\triangle
&
\xrightarrow{V_{56},\, V_{45},\, V_{34}}
&
\left[\begin{array}{cc|ccc|cc}
 & & & & & & x \\
\hline
 & & & & & y & y \\
 & & & & & y & y \\
 & & & & & y & y \\
 & & & & & y & y \\
\hline
 & y & y & y & y & 0 & x \\
x & y & y & y & y & x & 0
\end{array}\right]
\end{array}
$$

Finally, the middle block, $N(3:5,3:5)$ is skew triangularized by a skew QRQ^T factorization. Pulling this transformation through A does not change the structure of S.

$$
\begin{array}{ccc}
N & A & S \\
\left[\begin{array}{cc|ccc|cc}
 & & & & & & x \\
 & & & & & x & x \\
\hline
 & & & & y & y & y \\
 & & & 0 & y & y & y \\
 & & y & y & 0 & y & y \\
\hline
 & x & y & y & y & 0 & x \\
x & x & y & y & y & x & 0
\end{array}\right]
& &
\left[\begin{array}{cc|ccc|cc}
 & & & & & & x \\
 & & & & & x & x \\
\hline
 & & & & & y & y \\
 & & & & & y & y \\
 & & & & & y & y \\
\hline
 & x & y & y & y & 0 & x \\
x & x & y & y & y & x & 0
\end{array}\right]
\end{array}
$$

At this point N, A, S are of the form (16) and the URV decomposition is complete. A pseudocode for this algorithm is given in Appendix A.5.

Summarizing, for any square matrix A and any two skew symmetric matrices N, S of the same size we showed how to compute unitary matrices U, V such that all three, U^TAV, U^TNU, and V^TSV are skew triangular. In the next sections we demonstrate how eigenvalues and eigenvectors of even or palindromic problems can be extracted from these transformations.

5 Application to even eigenvalue problems

In this section we return to the even eigenvalue problem $Mx = \lambda Nx$ with $M = M^T \in \mathbb{C}^{n,n}$, $N = -N^T \in \mathbb{C}^{n,n}$.

In Section 2 we have shown, how the eigenvalues of (M, N) may be read off a skew URV decomposition of (M, N, N) provided that N is non-singular. Now, we want to generalize this idea to general pencils (M, N). We can assume w.l.o.g. that the pencil is regular, as the singular part can be deflated off as a preliminary step, see [7].

The role of the matrix $B = N^{-1}MN^{-1}M$ is taken by the $2n$–dimensional pencil

$$(\tilde{M}, \tilde{N}) = \left(\begin{bmatrix} 0 & M \\ M & 0 \end{bmatrix}, \begin{bmatrix} N & 0 \\ 0 & N \end{bmatrix} \right). \tag{18}$$

The following lemma relates the Weierstraß form [9] of $(\tilde{M}, \tilde{N})$ to that of (M, N).

Lemma 1 *Let $M - \lambda N \in \mathbb{C}^{n,n}$ be a regular even pencil. Define $(\tilde{M}, \tilde{N})$ as in (18). Then, if there exist m Jordan blocks of size k associated with the eigenvalue λ in the Weierstraß form of (M, N), then there are $2m$ Jordan blocks of size k for eigenvalue λ in the Weierstraß form of $(\tilde{M}, \tilde{N})$.*

Proof: Consider matrices $X, Y \in \mathbb{C}^{n,k}$ of full column rank k, and $J_1, J_2 \in \mathbb{C}^{k,k}$, one being a Jordan block and the other one the identity matrix, such that $MX = YJ_1$, $NX = YJ_2$. This means that span(X) is a deflating subspace of (M, N). Then, with $\tilde{X}_\pm = [X^T, \pm X^T]^T$ and $Y_\pm = [Y^T, \pm Y^T]^T$ we have $\tilde{M}\tilde{X}_\pm = \tilde{Y}_\pm(\pm J_1)$ and $\tilde{N}\tilde{X}_\pm = \tilde{Y}_\pm J_2$. So, both, $\tilde{X}_+$ and $\tilde{X}_-$ span deflating subspaces of $(\tilde{M}, \tilde{N})$. Since (M, N) was assumed to be regular, $(\tilde{M}, \tilde{N})$ has no further eigenvalues. The result follows as λ and $-\lambda$ agree in the number and sizes of associated Jordan blocks in the Weierstraß form of (M, N),[25]. ■

Summarizing, $(\tilde{M}, \tilde{N})$ has the same eigenvalues as (M, N), but of double multiplicity.

Next, we determine the spectrum of $(\tilde{M}, \tilde{N})$. To this end, let U, V define a skew URV decomposition of (M, N, N), i.e.,

$$U^TMV = R = , \quad U^TNU = T = , \quad V^TNV = P = .$$

Then, with $Q = U \oplus V$, it follows that

$$(\hat{M}, \hat{N}) = Q^T(\tilde{M}, \tilde{N})Q = \left(\begin{bmatrix} 0 & R \\ R^T & 0 \end{bmatrix}, \begin{bmatrix} T & 0 \\ 0 & P \end{bmatrix} \right).$$

Evaluating the determinant of $\hat{M} - \lambda\hat{N}$, the eigenvalues of $(\tilde{M}, \tilde{N})$ are given by

$$\Lambda(\tilde{M}, \tilde{N}) = \left\{ \pm\sqrt{\frac{r_{ij}r_{ji}}{p_{ji}t_{ji}}} \,\middle|\, i = 1, \ldots, n, j = n + 1 - i \right\}. \tag{19}$$

So, the eigenvalues of (M, N) are given by the same formula (19), but with i ranging from 1 to $\lceil \frac{n}{2} \rceil$. (Note that if n is odd, then for $i = \lceil \frac{n}{2} \rceil$ we have $i = j$ and $p_{ji} = t_{ji} = 0$. This corresponds to infinite eigenvalues.) This nicely coincides with formula (10) for the case that N is non-singular.

Also the eigenvectors of (M, N) can be derived using the skew URV decomposition. Evaluating the first columns of $MV = \bar{U}R$, $MU = \bar{V}R^T$, $NU = \bar{U}T$, and $NV = \bar{V}P$ we have

$$\begin{aligned} M[u_1, v_1] &= [\bar{u}_n, \bar{v}_n] \begin{bmatrix} 0 & r_{n1} \\ r_{1n} & 0 \end{bmatrix}, \\ N[u_1, v_1] &= [\bar{u}_n, \bar{v}_n] \begin{bmatrix} t_{n1} & 0 \\ 0 & p_{n1} \end{bmatrix}. \end{aligned}$$

So, $\operatorname{span}([u_1, v_1])$ is a deflating subspace of (M, N) corresponding to the eigenvalues $\lambda_{1;2} = \pm\sqrt{\frac{r_{1n} r_{n1}}{t_{n1} p_{n1}}}$. The eigenvectors are given by $x_{1;2} = \sqrt{r_{n1} p_{n1}}\, u_1 \pm \sqrt{r_{1n} t_{n1}}\, v_1$.

Note, that it is possible, that u_1 and v_1 are linearly dependent. In this case the algorithm, as it is now, breaks down. It is topic of future research to circumvent this failure.

Eigenvectors corresponding to other eigenvalue pairs can be obtained by reordering the generalized periodic Schur form [11].

6 Application to palindromic eigenvalue problems

Consider a palindromic eigenvalue problem $Ax = \lambda A^T x$. Of course, it could be addressed by treating its Cayley transform $(A + A^T, A - A^T)$. But cancellation errors could occur during these computations. In the following we will present a method, that still needs the matrix $A - A^T$, but circumvents the computation of $A + A^T$.

Again, we use a pencil whose spectrum is related to that of (A, A^T),

$$(\tilde{M}, \tilde{N}) = \left(\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}, \begin{bmatrix} A - A^T & 0 \\ 0 & A - A^T \end{bmatrix} \right). \tag{20}$$

The next lemma relates the Weierstraß form of $(\tilde{M}, \tilde{N})$ to that of (A, A^T).

Lemma 2 *Let $A - \lambda A^T \in \mathbb{C}^{n,n}$ be a regular palindromic pencil. Let $\mu(\lambda) := \frac{\sqrt{\lambda}}{\lambda - 1}$. Define $(\tilde{M}, \tilde{N})$ as in (20). Then, if there exist m Jordan blocks of size k for eigenvalue λ in the Weierstraß form of (A, A^T) there are*

(case 1: $\lambda \neq 0, \infty, \pm 1$) $2m$ Jordan blocks of size k for eigenvalue $\mu(\lambda)$,

(case 2: $\lambda = 0, \infty$) m Jordan blocks of size $2k$ for eigenvalue $0 = \mu(0) = \lim_{\lambda \to \infty} \mu(\lambda)$,

(case 3: $\lambda = 1$) $2m$ Jordan blocks of size k for eigenvalue $\infty = \lim_{\lambda \to 1} \mu(\lambda)$,

(case 4: $\lambda = -1$) m Jordan blocks of each size $\lceil \frac{k}{2} \rceil, \lfloor \frac{k}{2} \rfloor$ for both eigenvalues $\pm \frac{i}{2} = \pm\mu(-1)$ in the Weierstraß form of $(\tilde{M}, \tilde{N})$.

Proof: Let $X, V \in \mathbb{C}^{n,k}$ be such that $AX = A^T XJ$ and $AVJ = A^T V$ where J is a Jordan block of size k for eigenvalue $\lambda \neq \pm 1$, i.e. $\operatorname{span}(X)$ is a deflating subspace for λ and $\operatorname{span}(V)$ is a deflating subspace for $\frac{1}{\lambda}$. Since $\lambda \neq \frac{1}{\lambda}$, $[X, V]$ is of rank $2k$. Then we have

$$\begin{aligned} \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} \pm X & V \\ -V & \pm X \end{bmatrix} &= \begin{bmatrix} \pm A^T X & -AV \\ AV & \pm A^T X \end{bmatrix} \begin{bmatrix} 0 & J \\ I & 0 \end{bmatrix}, \\ \begin{bmatrix} A - A^T & 0 \\ 0 & A - A^T \end{bmatrix} \begin{bmatrix} \pm X & V \\ -V & \pm X \end{bmatrix} &= \begin{bmatrix} \pm A^T X & -AV \\ AV & \pm A^T X \end{bmatrix} \begin{bmatrix} J - I & 0 \\ 0 & J - I \end{bmatrix}. \end{aligned}$$

The Weierstraß form of $\left(\left[\begin{smallmatrix} 0 & J \\ I & 0 \end{smallmatrix}\right], \left[\begin{smallmatrix} J-I & 0 \\ 0 & J-I \end{smallmatrix}\right] \right)$ consists of Jordan blocks of size k for the eigenvalues $\pm\frac{\sqrt{\lambda}}{\lambda - 1}$ (if $\lambda \neq 0, \pm 1$) or of one Jordan block of size $2k$ for eigenvalue 0 (if $\lambda = 0$). Using the fact that λ and $\frac{1}{\lambda}$ agree in the number and sizes of associated Jordan blocks in the Weierstraß form of (A, A^T), e.g.,[21], this proves the cases 1 and 2.

To treat the cases 3 and 4, let $X, Y \in \mathbb{C}^{n,k}$ be such that $AX = YJ$ and $A^T X = Y$, where $J \in \mathbb{C}^{k,k}$ is a Jordan block for eigenvalue ± 1. Set $C = J^{-\frac{1}{2}}$. (Here, we mean the square root of J^{-1} that can be written as polynomial in J^{-1}. For existence see [14].) So, $JC^2 = I$. Further, let $\Lambda = (J - I)C$. Note, that J, C, Λ are all upper triangular Toeplitz matrices and thus commute. (This all follows from the fact, that upper triangular Toeplitz matrices can be represented as polynomials in the nilpotent Jordan block.) The diagonal elements of Λ are given by $\frac{\lambda-1}{\sqrt{\lambda}}$. Then we have

$$\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} X \\ \pm XC \end{bmatrix} (\pm\Lambda) = \begin{bmatrix} \pm YJC \\ Y \end{bmatrix} (\pm\Lambda) = \begin{bmatrix} Y(J-I) \\ \pm Y(J-I)C \end{bmatrix} = \begin{bmatrix} A-A^T & 0 \\ 0 & A-A^T \end{bmatrix} \begin{bmatrix} X \\ \pm XC \end{bmatrix}.$$

So, $\mathrm{span}([X^T, (\pm XC)^T]^T)$ is an invariant subspace of $(\tilde{M}, \tilde{N})$ corresponding to the inverses of the eigenvalues of $\pm\Lambda$. It remains to determine the Jordan form of Λ.

In case 3, i.e., $\lambda = 1$, Λ is a strict upper triangular matrix. Since it is the product of the non-singular matrix C and the matrix $(J-I)$ of rank $k-1$ also Λ is of rank $k-1$. Thus Λ is similar to the nilpotent Jordan block, which proves case 3.

In case 4, i.e. $\lambda = -1$, C is of the form

$$C = \pm i \begin{bmatrix} 1 & \frac{1}{2} & \frac{3}{8} & * & \cdots & * \\ & 1 & \frac{1}{2} & \frac{3}{8} & \ddots & \vdots \\ & & 1 & \frac{1}{2} & \ddots & * \\ & & & 1 & \ddots & \frac{3}{8} \\ & & & & \ddots & \frac{1}{2} \\ & & & & & 1 \end{bmatrix}.$$

This follows from the Taylor series expansion of $f(x) = x^{-\frac{1}{2}}$ at $x = -1$ yielding $f(-1+x) = i(1 + \frac{1}{2}x + \frac{3}{8}x^2 + \mathcal{O}(x^3))$. Since Λ is the product of C with $(J-I)$ it is a upper triangular Toeplitz matrix with $\pm 2i$ on the diagonal. Further, Λ has a vanishing super diagonal and a non-vanishing second super diagonal. Considering powers of $(\Lambda \mp 2iI)$ it is clear that the Jordan form of Λ consists of a Jordan block of each size $\lceil \frac{k}{2} \rceil, \lfloor \frac{k}{2} \rfloor$ for both eigenvalues $\pm \frac{i}{2} = \pm\mu(-1)$. ■

In other words, when going from (A, A^T) to $(\tilde{M}, \tilde{N})$ the eigenvalues are transformed from λ to $\pm\mu(\lambda)$. Note, that $\mu(\frac{1}{\lambda}) = -\mu(\lambda)$. So, $(\tilde{M}, \tilde{N})$ has only double eigenvalues. Inverting the formula for $\mu(\lambda)$ it follows that if μ is eigenvalue of $(\tilde{M}, \tilde{N})$, then

$$\lambda_{1;2} = \frac{1 + 2\mu^2 \pm \sqrt{1+4\mu^2}}{2\mu^2} \tag{21}$$

are eigenvalues of (A, A^T). Eigenvalues computed in this way are paired as $\lambda_1\lambda_2 = 1$. In order to avoid cancellation one would use (21) to compute the value of λ that is larger in modulus (say, λ_1) and then set $\lambda_2 = \frac{1}{\lambda_1}$.

The spectrum of $(\tilde{M}, \tilde{N})$ can be computed analogously to the even case: let U, V define a skew URV decomposition of $(A,\ A-A^T,\ A-A^T)$, i.e.,

$$U^TAV = R =, \quad U^T(A-A^T)U = T =, \quad V^T(A-A^T)V = P =.$$

Then, with $Q = U \oplus V$ we have

$$(\hat{M}, \hat{N}) = Q^T(\tilde{M}, \tilde{N})Q = \left(\begin{bmatrix} 0 & R \\ R^T & 0 \end{bmatrix}, \begin{bmatrix} T & 0 \\ 0 & P \end{bmatrix} \right).$$

The eigenvalues of $(\tilde{M}, \tilde{N})$ are given by (19). So, the eigenvalues of (A, A^T) are given by

$$\Lambda(A, A^T) = \left\{ \frac{1 + 2\mu_i^2 \pm \sqrt{1+4\mu_i^2}}{2\mu_i^2} \,\Big|\, \mu_i^2 = \frac{r_{ij}r_{ji}}{p_{ji}t_{ji}}, i = 1, \ldots, \lceil \frac{n}{2} \rceil, j = n+1-i \right\}.$$

Again, also eigenvectors can be extracted from U, V. Evaluation of the first columns of $AV = \bar{U}R$, $A^TU = \bar{V}R^T$, $(A-A^T)U = \bar{U}T$, and $(A-A^T)V = \bar{V}P$ gives

$$\begin{aligned} A[u_1, v_1] &= [\bar{u}_n, \bar{v}_n] \begin{bmatrix} t_{n1} & r_{n1} \\ r_{1n} & 0 \end{bmatrix}, \\ A^T[u_1, v_1] &= [\bar{u}_n, \bar{v}_n] \begin{bmatrix} 0 & r_{n1} \\ r_{1n} & -p_{n1} \end{bmatrix}. \end{aligned}$$

phase \ target	A, N, S	U	V	all
1a, skew QRQ^T	$\frac{11}{6}n^3$	–	$\frac{2}{3}n^3$	
1b, skew Takagi	$\frac{13}{3}n^3$	–	n^3	
2, skew QR	$\frac{10}{3}n^3$	$\frac{4}{3}n^3$	–	
3, URV-Hessenberg	$30nr^2 - \frac{5}{6}r^3$	$12nr^2$	$12nr^2$	
4, periodic QZ	$8nr^2 + 75\frac{1}{3}r^3$	$4nr^2$	$4nr^2$	
5, triangularization	$\frac{19}{6}n^3 - n^2r$ $-10nr^2 - \frac{4}{3}r^3$	$n^3 + 2n^2r$ $-8nr^2$	$2n^3 - 2n^2r$ $-4nr^2$	
URV(1a)	$\approx 24n^3$	$\frac{16}{3}n^3$	$\frac{14}{3}n^3$	$\approx 34n^3$
URV(1b)	$\approx 11n^3 - n^2r$ $+28nr^2 + 73r^3$	$\frac{7}{3}n^3 + 2n^2r$ $+8nr^2$	$3n^3 - 2n^2r$ $+12nr^2$	$\approx \frac{49}{3}n^3 - n^2r$ $+48nr^2 + 73r^3$
URV(1b,$r = \frac{n}{3}$)	$\approx \frac{33}{2}n^3$	$\approx 4n^3$	$\frac{11}{3}n^3$	$\approx 24n^3$
QZ	$30n^3$	$16n^3$	$20n^3$	$66n^3$

Table 13: Flop counts of the URV algorithm. The numbers are in terms of the problem size n and the parameter r in the partition (14). It is assumed that only the upper or lower triangular part of N and S are stored/updated. Only dominant terms are shown, so every number should be understood as 'plus $\mathcal{O}(n^2 + nr + r^2)$'. The row labeled URV(1a) shows numbers for the whole algorithm using the (reduced) skew QRQ^T factorization in the first phase (so, $r = \lfloor \frac{n}{2} \rfloor$). If the skew Takagi factorization is used in the first phase (row 'URV(1b)'), then $r = \frac{\text{rank}(S)}{2}$. The next row lists the terms for the URV(1b) algorithm for the special case of $\text{rank}(S) = \frac{2}{3}n$. For comparison, the terms for the QZ algorithm are given, too. The four numbers are the flop counts to compute the generalized Schur form, the transformation matrices Q, Z, and the sum of all. Flop counts for classic algorithms (QR-factorization, QZ-algorithm, etc.) are taken from [10]. The iterative part of the perodic QZ algorithm is taken to consume 4 times as many flops as the iterative part of the standard QR algorithm.

So, $\text{span}([u_1, v_1])$ is a deflating subspace of (A, A^T) corresponding to the eigenvalues

$$\lambda_{1;2} = \frac{p_{n1}t_{n1} + 2r_{n1}r_{1n} \pm \sqrt{p_{n1}^2 t_{n1}^2 + 4p_{n1}t_{n1}r_{n1}r_{1n}}}{2r_{1n}r_{n1}}.$$

Eigenvectors of (A, A^T) are given by

$$x_{1;2} = (p_{n1}t_{n1} \pm \sqrt{p_{n1}t_{n1}(p_{n1}t_{n1} + 4r_{n1}r_{1n})})\, u_1 + 2r_{1n}t_{n1}\, v_1.$$

Again, eigenvectors corresponding to other eigenvalue pairs can be obtained by reordering the generalized periodic Schur form [11].

7 Various topics

In this section we discuss several topics.

We begin by discussing the performance of our algorithm. Table 13 shows flop counts for the method, separated by phases and whether only A, N, S or additionally U and/or V need to be formed. For comparison, we also list flop counts of the QZ algorithm applied to (A, A^T) or (M, N). Notably, the URV algorithm needs less flops then the QZ algorithm, if the entire factorization is computed even by a factor of almost two ($34n^3$ vs. $66n^3$). Note further that in the case of a highly rank deficient matrix S the higher cost of the skew Takagi factorization in the first phase pays off by lower cost for the third and fourth phases resulting in an overall lower amount for the whole algorithm ($24n^3$ vs. $34n^3$ for $\text{rank}(S) = \frac{2}{3}n$).

The URV algorithm as presented above has been implemented in Matlab. (In Phase 4 we used a preliminary version of the periodic QZ algorithm, that will be part of the Fortran77 periodic eigenvalue toolbox announced in [11].) For randomly generated matrices (Matlab:

`A=randn(n)+i*randn(n)`) of medium size ($n \approx 700$), our implementation is slower than Matlabs QZ algorithm by only a factor of ca. 2.5. Given that we use a research implementation using only BLAS1 operations while qz uses highly optimized LAPACK routines, this is a promising result.

Now, we turn to the subtle topic of error analysis. Let $\hat{U}, \hat{V}, \hat{R}, \hat{T}, \hat{P}$ be the computed decomposition (11). Note, that every step of the algorithm consists either of a Householder or Givens update or of setting a small element to zero. Thus, by standard techniques [10], the URV algorithm is backward stable for (A, N, S), i.e., $\hat{U}$ and $\hat{V}$ are unitary to machine precision and $\hat{R}, \hat{T}, \hat{P}$ are the exact triangular factors of a nearby problem. This in turn means that there exist $\tilde{A}, \tilde{N}, \tilde{S}$ with $\|A-\tilde{A}\|_2 = \mathcal{O}(\varepsilon)\|A\|_2$, $\|N-\tilde{N}\|_2 = \mathcal{O}(\varepsilon)\|N\|_2$, $\|S-\tilde{S}\|_2 = \mathcal{O}(\varepsilon)\|S\|_2$, $\tilde{N}^T = -\tilde{N}$, $\tilde{S}^T = -\tilde{S}$ and perfectly unitary matrices U, V such that $U^T\tilde{A}V = \hat{R}$, $U^T\tilde{N}U = \hat{T}$, $V^T\tilde{S}V = \hat{P}$.

Using these facts, it is clear that using (19) to compute eigenvalues of an even pencil $\left(\left[\begin{smallmatrix} & A \\ A^T & \end{smallmatrix}\right], \left[\begin{smallmatrix} N & \\ & S\end{smallmatrix}\right]\right)$ yields the exact eigenvalues of the nearby pencil $\left(\left[\begin{smallmatrix} & \tilde{A} \\ \tilde{A}^T & \end{smallmatrix}\right], \left[\begin{smallmatrix} \tilde{N} & \\ & \tilde{S}\end{smallmatrix}\right]\right)$. But it is not clear, if this implies backward stability for even or palindromic eigenvalue problems, because there is more structure in the corresponding double size pencils (18) ($N = S$, A symmetric) or (20) ($N = S = A - A^T$). In general, the perturbed pencil will not share these extra properties. More research is needed in this field. However, it is enough to achieve paired eigenvalues, which is the primary interest in this paper. (In the Hamiltonian case, in [3] the symplectic URV algorithm was shown to produce eigenvalues that are of the same quality as those computed by a backward stable method, i.e., the error in an eigenvalue λ is of the order of $\frac{\varepsilon}{y^T x}$, where x and y are right and left eigenvectors corresponding to λ.)

We now want to specify our statement in Section 1 that the skew URV decomposition generalizes the symplectic URV decomposition.

Lemma 3 *Let $A \in \mathbb{C}^{2n,2n}$. Let U, V, R, T, P define a skew URV decomposition of the triple $(J^T A, J, J)$, where $J = \left[\begin{smallmatrix} 0 & I_n \\ -I_n & 0\end{smallmatrix}\right]$, i.e., all $R = U^T(J^T A)V$, $\quad T = U^T JU$, and $P = V^T JV$ are skew triangular.*

Then there are unitary diagonal matrices $D, D_2 \in \mathbb{C}^{n,n}$, such that $\tilde{U} := U(I \oplus D^* F)$, $\tilde{V} := V(I \oplus D_2^* F)$ and $\tilde{R} := J(I \oplus D^* F)^T R(I \oplus D_2^* F)$ make up a symplectic URV decomposition, $A = \tilde{U}\tilde{R}\tilde{V}^*$ of A.

Proof: With U and J also T is unitary. Since T is also skew symmetric and skew triangular, it must be of the form

$$T = \begin{bmatrix} 0 & FD \\ -D^T F & 0 \end{bmatrix},$$

where D is unitary and diagonal. $\tilde{U}$ is symplectic as $\tilde{U}^T J\tilde{U} = (I \oplus D^* F)^T U^T JU(I \oplus D^* F) = (I \oplus D^* F)^T T(I \oplus D^* F) = J$. By an analogous argument, there is a diagonal, unitary matrix D_2, such that $\tilde{V}$ is symplectic. The matrix $\tilde{R}$ is of the right shape (5), as

$$\tilde{R} = J \begin{bmatrix} I & \\ & FD^* \end{bmatrix}^T \begin{bmatrix} 0 & \diagup\!\!| \\ |\!\!\diagup & \square \end{bmatrix} \begin{bmatrix} I & \\ & FD_2^* \end{bmatrix} = \begin{bmatrix} \diagdown & \square \\ 0 & \diagdown \end{bmatrix}.$$

Finally,

$$\tilde{R} = J(I \oplus D^* F)^T R(I \oplus D_2^* F) = J(I \oplus D^* F)^T U^T J^T AV(I \oplus D_2^* F) = J\tilde{U}^T J^T A\tilde{V} = \tilde{U}^* A\tilde{V},$$

thus $A = \tilde{U}\tilde{R}\tilde{V}^*$. (Here, we used $J\tilde{U}J^T = U^{-1}$, a property of symplectic matrices.) ■

In other words, a skew URV decomposition of $(J^T A, J, J)$ defines a symplectic URV decomposition of A.

7.1 The real case

Until now, complex matrices A, N, S were modified by complex transformation matrices U, V. However, most physical problems result in real matrices. Of course, we could just treat the real problem as a complex one, but there are good reasons to stay in real arithmetic, e.g., the execution time of complex floating point operations is three to four times that of real operations. Moreover, real eigenvalue problems have more structure: the eigenvalues appear in complex conjugate pairs.

Luckily, almost the whole URV algorithm works in real arithmetic just as well as with complex numbers. Householder reflections, Givens rotations, skew QR-, QRQ^T- and Takagi factorizations all yield real results for real problems. Only one aspect changes: the real periodic QZ algorithm in phase 4 returns a real periodic generalized Schur form, i.e., the matrix T_4 in (13) is not upper triangular, but quasi upper triangular with 1×1 and 2×2 blocks on the diagonal, the 2×2 blocks corresponding to a pair of conjugate eigenvalues. Thus, at the end of phase 4, N, A, S will have the form

$$N_4 = \begin{array}{cc} & \begin{matrix} r & r & m \end{matrix} \\ \begin{matrix} r \\ r \\ m \end{matrix} & \begin{bmatrix} 0 & -N_{21}^T & -N_{31}^T \\ N_{21} & N_{22} & -N_{32}^T \\ N_{31} & N_{32} & N_{33} \end{bmatrix} \end{array}, \quad A_4 = \begin{array}{cc} & \begin{matrix} m & r & r \end{matrix} \\ \begin{matrix} r \\ r \\ m \end{matrix} & \begin{bmatrix} 0 & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \end{array}, \quad S_4 = \begin{array}{cc} & \begin{matrix} m & r & r \end{matrix} \\ \begin{matrix} m \\ r \\ r \end{matrix} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & S_{23} \\ 0 & -S_{23}^T & S_{33} \end{bmatrix} \end{array}$$

with $A_{13}, A_{22}, A_{31}, S_{23}$ skew triangular and N_{21} quasi skew triangular.

At the end of phase 5, N, A, S will have the form (the symbols N_{ij}, A_{ij}, S_{ij} are reused for different matrices)

$$N_5 = \begin{array}{cc} & \begin{matrix} r & m & r \end{matrix} \\ \begin{matrix} r \\ m \\ r \end{matrix} & \begin{bmatrix} 0 & 0 & -N_{31}^T \\ 0 & N_{22} & -N_{32}^T \\ N_{31} & N_{32} & N_{33} \end{bmatrix} \end{array}, \quad A_5 = \begin{array}{cc} & \begin{matrix} r & m & r \end{matrix} \\ \begin{matrix} r \\ m \\ r \end{matrix} & \begin{bmatrix} 0 & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \end{array}, \quad S_5 = \begin{array}{cc} & \begin{matrix} r & m & r \end{matrix} \\ \begin{matrix} r \\ m \\ r \end{matrix} & \begin{bmatrix} 0 & 0 & S_{13} \\ 0 & 0 & S_{23} \\ -S_{13}^T & -S_{23}^T & S_{33} \end{bmatrix} \end{array}$$

with $A_{13}, A_{22}, A_{31}, N_{22}, N_{31}$ skew triangular and S_{13} quasi skew triangular.

8 Conclusion and acknowledgments

In this paper we have introduced a new URV decomposition affecting a matrix triple. It was shown how this URV decomposition can be used to solve palindromic or even eigenvalue problems. The eigenvalues computed in this way are paired in compliance with the spectral symmetry that palindromic and even eigenproblems show.

An algorithm for the computation of the URV decomposition has been developed. It consists of several phases, some of them could be of interest in their own right: reduction of a square matrix to skew triangular form by one sided transformations, reduction of a skew symmetric matrix to (reduced) skew triangular form by congruence, or the structure preserving singular value decomposition of a skew symmetric matrix.

Thanks go to Volker Mehrmann, Christian Mehl, and Hongguo Xu for discussions on the topic and giving comments on draft versions of this work. Special thanks go to Daniel Kressner and Bo Kågström for their hospitality during a research stay in Umeå in fall 2006.

A Algorithms

In this section we list pseudocodes of the discussed algorithms.

In the following `house(x,i)` returns a Householder reflector H, such that $Hx = \alpha e_i$, with $\alpha \in \mathbb{C}$. The variant `house(x,end)` is short for `house(x,n)`, where $x \in \mathbb{C}^n$. Further, `givens(`α, β`,i)` returns a Givens rotation G such that $G \left[\begin{smallmatrix} \alpha \\ \beta \end{smallmatrix} \right] = \gamma e_i$, with $\gamma \in \mathbb{C}$. The functions `qr(A)` and `rq(A)` perform the standard QR- and RQ factorizations. `fliplr(A)` and `flipud(A)` flip the argument leftside-right, or upside-down, respectively.

A.1 (Reduced) Skew QRQ^T factorization

function $[Q, R]$=`skewqrqt(`S`,want_reduced)`

Input: skew symmetric matrix $S \in \mathbb{C}^{n,n}$; flag `want_reduced`, indicating whether a reduced factorization is wanted

Output: unitary Q, skew triangular, skew symmetric R such that $S = QRQ^T$, S is overwritten by R

$Q \leftarrow I_n$
for $i = 1 : \lceil \frac{n}{2} \rceil - 1$ **do**
  $H \leftarrow$ `house`$(S(i+1 : n+1-i, i), end)$

$S \leftarrow HS(i+1:n+1-i,:)$
$S \leftarrow S(:,i+1:n+1-i)H^T$
$Q \leftarrow HQ(i+1:n+1-i,:)$
end for
if n is odd and `want_reduced` **then**
for $i = \frac{n-1}{2} : -1 : 1$ **do**
$G \leftarrow$ `givens`$(S(i,n+1-i), S(i+1,n+1-i), 2)$
$S \leftarrow GS(i:i+1,:)$
$S \leftarrow S(:,i:i+1)G^T$
$Q \leftarrow GQ(i:i+1,:)$
end for
end if

A.2 Skew Takagi factorization

function $[Q,R]$=`skewtakagi(`S`)`
Input: skew symmetric matrix $S \in \mathbb{C}^{n,n}$
Output: unitary Q, skew symmetric, skew diagonal, real R such that $S = QRQ^T$, S is overwritten by R
$Q \leftarrow I_n$
for $i = 1 : \lfloor \frac{n}{2} \rfloor - 1$ **do**
$H \leftarrow$ `house`$(S(i+1:n+1-i,i), end)$
$S \leftarrow HS(i+1:n+1-i,:)$
$S \leftarrow S(:,i+1:n+1-i)H^T$
$Q \leftarrow HQ(i+1:n+1-i,:)$
$H \leftarrow$ `house`$(S(i+1:n-i,n+1-i),1)$
$S \leftarrow HS(i+1:n-i,:)$
$S \leftarrow S(:,i+1:n-i)H^T$
$Q \leftarrow HQ(i+1:n-i,:)$
end for
if n is odd **then**
$i \leftarrow \frac{n+1}{2}$
$G \leftarrow$ `givens`$(S(i,i-1), S(i+1,i-1), 2)$
$S \leftarrow GS(i:i+1,:)$
$S \leftarrow S(:,i:i+1)G^T$
$Q \leftarrow GQ(i:i+1,:)$
end if
$B \leftarrow$ `flipud`$(S(\lceil \frac{n}{2} \rceil : n, 1 : \lfloor \frac{n}{2} \rfloor))$
$[U,\Sigma,V] \leftarrow$ `svd`(B)
$S(\lceil \frac{n}{2} \rceil + 1 : n, 1 : \lfloor \frac{n}{2} \rfloor) \leftarrow$ `flipud`(Σ)
$S(1 : \lfloor \frac{n}{2} \rfloor, \lceil \frac{n}{2} \rceil + 1 : n) \leftarrow -$`fliplr`$(\Sigma)$
$Q(1 : \lceil \frac{n}{2} \rceil, :) \leftarrow V^T Q(1 : \lceil \frac{n}{2} \rceil, :)$
$Q(\lfloor \frac{n}{2} \rfloor + 1 : n, :) \leftarrow FU^* FQ(\lfloor \frac{n}{2} \rfloor + 1 : n, :)$

A.3 Skew QR factorization

function $[Q,R] =$ `skewqr`(A)
Input: matrix $A \in \mathbb{C}^{n,n}$
Output: unitary Q, skew triangular R such that $A = QR$
$A \leftarrow$ `fliplr`(A)
$[Q,R] \leftarrow$ `ql`(A)
$R \leftarrow$ `fliplr`(R)

function $[R,Q] =$ `skewrq`(A)
Input: matrix $A \in \mathbb{C}^{n,n}$
Output: unitary Q, skew triangular R such that $A = RQ$
$A \leftarrow$ `flipud`(A)
$[R,Q] \leftarrow$ `rq`(A)

$R \leftarrow \texttt{flipud}(R)$

A.4 URV-Hessenberg reduction

function $[U, V, \tilde{A}, \tilde{N}, \tilde{S}]$=`urvhess`$(A, N, S)$

Input: matrices $A, N = -N^T, S = -S^T \in \mathbb{C}^{n,n}$, n even, A, S skew triangular

Output: unitary U, V such that $\tilde{N} = U^T N U$ is skew Hessenberg, while $\tilde{A} = U^T A V$ and $\tilde{S} = V^T S V$ are skew triangular, A, N, S are overwritten by $\tilde{A}, \tilde{N}, \tilde{S}$

$U \leftarrow I_n$
$V \leftarrow I_n$
for $j = 1 : \frac{n}{2} - 1$ **do**
  for $i = j + 1 : \frac{n}{2} - 1$ **do**
    {annihilate N(i,j) by Givens rotation in (i,i+1) plane}
    $G \leftarrow \texttt{givens}(N(i, j), N(i + 1, j), 2)$
    {apply rotation to N as congruence}
    $N(i : i + 1, :) \leftarrow GN(i : i + 1, :)$
    $N(:, i : i + 1) \leftarrow N(:, i : i + 1)G^T$
    {apply rotation to U from the right}
    $U(:, i : i + 1) \leftarrow U(:, i : i + 1)G^T$
    {apply rotation to A from the left generating fill-in at (i,n-i)}
    $A(i : i + 1, :) \leftarrow GA(i : i + 1, :)$
    {annihilate A(i,n-i) by Givens rotation in (n-i,n-i+1) plane}
    $G \leftarrow \texttt{givens}(A(i, n - i), A(i, n - i + 1), 2)$
    {apply rotation to A from the right}
    $A(:, n - i : n - i + 1) \leftarrow A(:, n - i : n - i + 1)G^T$
    {apply rotation to V from the right}
    $V(:, n - i : n - i + 1) \leftarrow V(:, n - i : n - i + 1)G^T$
    {apply rotation to S as congruence generating fill-in at (i,n-i)}
    $S(:, n - i : n - i + 1) \leftarrow S(:, n - i : n - i + 1)G^T$
    $S(n - i : n - i + 1, :) \leftarrow GS(n - i : n - i + 1, :)$
    {annihilate S(i,n-i) by Givens rotation in (i,i+1) plane}
    $G \leftarrow \texttt{givens}(S(i, n - i), S(i + 1, n - i), 2)$
    {apply rotation to S as congruence}
    $S(i : i + 1, :) \leftarrow GS(i : i + 1, :)$
    {apply rotation to V from the right}
    $V(:, i : i + 1) \leftarrow V(:, i : i + 1)G^T$
    {apply rotation to A from the right generating fill-in at (n-i,i)}
    $A(:, i : i + 1) \leftarrow A(:, i : i + 1)G^T$
    {annihilate A(n-i,i) by Givens rotation in (n-i,n-i+1) plane}
    $G \leftarrow \texttt{givens}(A(n - i, i), A(n - i + 1, i), 2)$
    {apply rotation to A from the left}
    $A(n - i : n - i + 1, :) \leftarrow GA(n - i : n - i + 1, :)$
    {apply rotation to U from the right}
    $U(:, n - i : n - i + 1) \leftarrow U(:, n - i : n - i + 1)G^T$
    {apply rotation to N as congruence, no fill-in}
    $N(n - i : n - i + 1, :) \leftarrow GN(n - i : n - i + 1, :)$
    $N(:, n - i : n - i + 1) \leftarrow N(:, n - i : n - i + 1)G^T$
  end for
  $i \leftarrow \frac{n}{2}$
  {annihilate N(i,j) by Givens rotation in (i,i+1) plane}
  $G \leftarrow \texttt{givens}(N(i, j), N(i + 1, j), 2)$
  {apply rotation to N as congruence}
  $N(i : i + 1, :) \leftarrow GN(i : i + 1, :)$
  $N(:, i : i + 1) \leftarrow N(:, i : i + 1)G^T$
  {apply rotation to U from the right}
  $U(:, i : i + 1) \leftarrow U(:, i : i + 1)G^T$
  {apply rotation to A from the left generating fill-in at (i,n-i)}
  $A(i : i + 1, :) \leftarrow GA(i : i + 1, :)$

{annihilate A(i,n-i) by Givens rotation in (n-i,n-i+1) plane}
$G \leftarrow$ `givens`$(A(i,n-i), A(i,n-i+1), 2)$
{apply rotation to A from the right}
$A(:,n-i:n-i+1) \leftarrow A(:,n-i:n-i+1)G^T$
{apply rotation to V from the right}
$V(:,n-i:n-i+1) \leftarrow V(:,n-i:n-i+1)G^T$
{apply rotation to S as congruence, no fill-in}
$S(n-i:n-i+1,:) \leftarrow GS(n-i:n-i+1,:)$
$S(:,n-i:n-i+1) \leftarrow S(:,n-i:n-i+1)G^T$
for $i = m+1 : n-j-1$ **do**
  {annihilate N(i,j) by Givens rotation in (i,i+1) plane}
  $G \leftarrow$ `givens`$(N(i,j), N(i+1,j), 2)$
  {apply rotation to N as congruence}
  $N(i:i+1,:) \leftarrow GN(i:i+1,:)$
  $N(:,i:i+1) \leftarrow N(:,i:i+1)G^T$
  {apply rotation to U from the right}
  $U(:,i:i+1) \leftarrow U(:,i:i+1)G^T$
  {apply rotation to A from the left generating fill-in at (i,n-i)}
  $A(i:i+1,:) \leftarrow GA(i:i+1,:)$
  {annihilate A(i,n-i) by Givens rotation in (n-i,n-i+1) plane}
  $G \leftarrow$ `givens`$(A(i,n-i), A(i,n-i+1), 2)$
  {apply rotation to A from the right}
  $A(:,n-i:n-i+1) \leftarrow A(:,n-i:n-i+1)G^T$
  {apply rotation to V from the right}
  $V(:,n-i:n-i+1) \leftarrow V(:,n-i:n-i+1)G^T$
  {apply rotation to S as congruence generating fill-in at (n-i,i)}
  $S(n-i:n-i+1,:) \leftarrow GS(n-i:n-i+1,:)$
  $S(:,n-i:n-i+1) \leftarrow S(:,n-i:n-i+1)G^T$
  {annihilate S(n-i,i) by Givens rotation in (i,i+1) plane}
  $G \leftarrow$ `givens`$(S(n-i,i), S(n-i,i+1), 2)$
  {apply rotation to S as congruence}
  $S(:,i:i+1) \leftarrow S(:,i:i+1)G^T$
  $S(i:i+1,:) \leftarrow GS(i:i+1,:)$
  {apply rotation to V from the right}
  $V(:,i:i+1) \leftarrow V(:,i:i+1)G^T$
  {apply rotation to A from the right generating fill-in at (n-i,i)}
  $A(:,i:i+1) \leftarrow A(:,i:i+1)G^T$
  {annihilate A(n-i,i) by Givens rotation in (n-i,n-i+1) plane}
  $G \leftarrow$ `givens`$(A(n-i,i), A(n-i+1,i), 2)$
  {apply rotation to A from the left}
  $A(n-i:n-i+1,:) \leftarrow GA(n-i:n-i+1,:)$
  {apply rotation to U from the right}
  $U(:,n-i:n-i+1) \leftarrow U(:,n-i:n-i+1)G^T$
  {apply rotation to N as congruence, no fill-in}
  $N(n-i:n-i+1,:) \leftarrow GN(n-i:n-i+1,:)$
  $N(:,n-i:n-i+1) \leftarrow N(:,n-i:n-i+1)G^T$
end for
end for

A.5 Phase 5

function $[U, V, \tilde{A}, \tilde{N}, \tilde{S}]$=`phase5`$(A, N, S, r)$
Input: matrices $A, N = -N^T, S = -S^T \in \mathbb{C}^{n,n}$ of the form (15)
Output: unitary U, V such that $\tilde{N} = U^T N U$, $\tilde{A} = U^T A V$, $\tilde{S} = V^T S V$ are of the form (16), A, N, S are overwritten by $\tilde{A}, \tilde{N}, \tilde{S}$
for $j = 1 : r$ **do**
  for $i = 2r+1-j : n-j$ **do**
    {annihilate N(i,j) by Givens rotation in (i,i+1) plane}

$G \leftarrow$ `givens`$(N(i,j), N(i+1,j), 2)$
{apply rotation to N as congruence}
$N(i:i+1,:) \leftarrow GN(i:i+1,:)$
$N(:,i:i+1) \leftarrow N(:,i:i+1)G^T$
{apply rotation to U from the right}
$U(:,i:i+1) \leftarrow U(:,i:i+1)G^T$
{apply rotation to A from the left generating fill-in at (i,n-i)}
$A(i:i+1,:) \leftarrow GA(i:i+1,:)$
{annihilate A(i,n-i) by Givens rotation in (n-i,n-i+1) plane}
$G \leftarrow$ `givens`$(A(i,n-i), A(i,n-i+1), 2)$
{apply rotation to A from the right}
$A(:,n-i:n-i+1) \leftarrow A(:,n-i:n-i+1)G^T$
{apply rotation to V from the right}
$V(:,n-i:n-i+1) \leftarrow V(:,n-i:n-i+1)G^T$
{apply rotation to S as congruence}
$S(n-i:n-i+1,:) \leftarrow GS(n-i:n-i+1,:)$
$S(:,n-i:n-i+1) \leftarrow S(:,n-i:n-i+1)G^T$
end for
end for
{triangularize middle block in N}
$[Q, N(r+1:n-r, r+1:n-r)] \leftarrow$ `skewqrqt(`$N(r+1:n-r, r+1:n-r)$`,false)`
{apply transformation to rest of N as congruence}
$N(r+1:n-r, n-r+1:n) \leftarrow Q^* N(r+1:n-r, n-r+1:n)$
$N(n-r+1:n, r+1:n-r) \leftarrow N(n-r+1:n, r+1:n-r)\bar{Q}$
{apply transformation to U from the right}
$U(:,r+1:n-r) \leftarrow U(:,r+1:n-r)\bar{Q}$
{apply transformation to A from the left destroying triangular form}
$A(r+1:n-r,:) \leftarrow Q^* A(r+1:n-r,:)$
{retriangularize middle block of A(i,n-i) by skew RQ factorization}
$[A(r+1:n-r, r+1:n-r), Q] \leftarrow$ `skewrq`$(A(r+1:n-r, r+1:n-r))$
{apply transformation to rest of A from the right}
$A(n-r+1:n, r+1:n-r) \leftarrow A(n-r+1:n, r+1:n-r)Q^*$
{apply transformation to V from the right}
$V(:,r+1:n-r) \leftarrow V(:,r+1:n-r)Q^*$
{apply transformation to S as congruence}
$S(r+1:n-r,:) \leftarrow \bar{Q}S(r+1:n-r,:)$
$S(:,r+1:n-r) \leftarrow S(:,r+1:n-r)Q^*$

References

[1] Peter Benner, Ralph Byers, Volker Mehrmann, and Hongguo Xu. Numerical computation of deflating subspaces of skew-Hamiltonian/Hamiltonian pencils. *SIAM J. Matrix Anal. Appl.*, 24(1):165–190 (electronic), 2002.

[2] Peter Benner, Volker Mehrmann, and Hongguo Xu. A new method for computing the stable invariant subspace of a real Hamiltonian matrix. *J. Comput. Appl. Math.*, 86(1):17–43, 1997.

[3] Peter Benner, Volker Mehrmann, and Hongguo Xu. A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils. *Numer. Math.*, 78(3):329–358, 1998.

[4] Adam Bojanczyk, Gene H. Golub, and Paul Van Dooren. The periodic schur decomposition; algorithms and applications. In *Proc. SPIE conference*, volume 1770, pages 31–42, 1992.

[5] Angelika Bunse-Gerstner and William B. Gragg. Singular value decompositions of complex symmetric matrices. *J. Comput. Appl. Math.*, 21(1):41–54, 1988.

[6] Peter Businger and Gene H. Golub. Handbook series linear algebra. Linear least squares solutions by Householder transformations. *Numer. Math.*, 7:269–276, 1965.

[7] Ralph Byers, Volker Mehrmann, and Hongguo Xu. A structured staircase algorithm for skew-symmetric/symmetric pencils. *ETNA*, 26:1–33, 2007.

[8] Delin Chu, Xinmin Liu, and Volker Mehrmann. A numerical method for computing the Hamiltonian Schur form. *Numer. Math.*, 105(3):375–412, 2007.

[9] Felix R. Gantmacher. *The theory of matrices. Vols. 1, 2.* Translated by K. A. Hirsch. Chelsea Publishing Co., New York, 1959.

[10] Gene H. Golub and Charles F. Van Loan. *Matrix computations.* Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[11] Robert Granat, Bo Kågström, and Daniel Kressner. Computing periodic deflating subspaces associated with a specified set of eigenvalues. *BIT Numerical Mathematics*, 43(1):001–018, 2003.

[12] John J. Hench and Alan J. Laub. Numerical solution of the discrete-time periodic Riccati equation. *IEEE Trans. Automat. Control*, 39(6):1197–1210, 1994.

[13] Andreas Hilliges, Christian Mehl, and Volker Mehrmann. On the solution of palindromic eigenvalue problems. In *Proceedings of the 4th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS)*. Jyväskylä, Finland, 2004. CD-ROM.

[14] Roger A. Horn and Charles R. Johnson. *Topics in matrix analysis.* Cambridge University Press, Cambridge, 1991.

[15] Roger A. Horn and Vladimir V. Sergeichuk. Canonical forms for complex matrix congruence and *congruence. *Linear Algebra and its Appl.*, 2006. in press.

[16] Roger A. Horn and Vladimir V. Sergeichuk. A regularization algorithm for matrices of bilinear and sesquilinear forms. *Linear Algebra Appl.*, 412(2-3):380–395, 2006.

[17] Daniel Kressner. An effcient and reliable implementation of the periodic QZ algorithm. In *IFAC Workshop on Periodic Control Systems.*

[18] D. Steven Mackey, Niloufer Mackey, Christian Mehl, and Volker Mehrmann. Structured polynomial eigenvalue problems: Good vibrations from good linearizations. *SIAM Journal on Matrix Analysis and Applications*, 28(4):1029–1051, 2006.

[19] D. Steven Mackey, Niloufer Mackey, Christian Mehl, and Volker Mehrmann. Numerical methods for palindromic eigenvalue problems, 2007. in preparation.

[20] Leiba Rodman. Bounded and stably bounded palindromic difference equations of first order. *ELA*, 15:22–49, January 2006.

[21] Christian Schröder. A canonical form for palindromic pencils and palindromic factorizations. Preprint 316, TU Berlin, Matheon, Germany, 2006.

[22] Christian Schröder. A QR-like algorithm for the palindromic eigenvalue problem. Preprint, TU Berlin, Matheon, Germany, in preparation.

[23] G. W. Pete Stewart. Updating a rank-revealing ULV decomposition. *SIAM J. Matrix Anal. Appl.*, 14(2):494–499, 1993.

[24] T. Takagi. On an algebraic problem related to an analytic Theorem of Carathédory and Fejér and on an allied theorem of Landau. *Japan J. Math.*, 1:82–93, 1924.

[25] Robert C. Thompson. Pencils of complex and real symmetric and skew matrices. *Linear Algebra Appl.*, 147:323–371, 1991.

[26] Andreas Varga and Paul Van Dooren. Computational methods for periodic systems — an overview. In *Proc. of IFAC Workshop on Periodic Control Systems, Como, Italy*, pages 171–176, 2001.

[27] Paul R. Willems, Bruno Lang, and Christof Vömel. Computing the bidiagonal SVD using multiple relatively robust representations. preprint BUW-SC 2005/5, Bergische Universität Wuppertal, 2005.