Fortran 77 Subroutines for Computing the Eigenvalues of Hamiltonian Matrices II

PETER BENNER
Technische Universität Chemnitz
and
DANIEL KRESSNER
Technische Universität Berlin

This article describes Fortran 77 subroutines for computing eigenvalues and invariant subspaces of Hamiltonian and skew-Hamiltonian matrices. The implemented algorithms are based on orthogonal symplectic decompositions, implying numerical backward stability as well as symmetry preservation for the computed eigenvalues. These algorithms are supplemented with balancing and block algorithms, which can lead to considerable accuracy and performance improvements. As a by-product, an efficient implementation for computing symplectic QR decompositions is provided. We demonstrate the usefulness of the subroutines for several, practically relevant examples.

Categories and Subject Descriptors: D.3.2 [**Programming Languages**]: Language Classifications—*Fortran 77*; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems—*Computations on matrices*; G.1.3 [**Numerical Analysis**]: Numerical Linear Algebra—*Eigenvalues and eigenvectors (direct and iterative methods)*; G.4 [**Mathematics of Computing**]: Mathematical Software—*Algorithm design and analysis; Certification and testing; Documentation; Efficiency; Reliability and robustness*

General Terms: Algorithms, Documentation, Performance

Additional Key Words and Phrases: Algebraic Riccati equation, eigenvalues, Hamiltonian matrix, invariant subspaces, skew-Hamiltonian matrix, symplectic QR decomposition

1. INTRODUCTION

This article describes Fortran 77 subroutines for computing eigenvalues and invariant subspaces of a *Hamiltonian matrix*

$$H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}, \quad G = G^T, \quad Q = Q^T \tag{1}$$

This work was supported by the DFG Research Center "Mathematics for key technologies" (FZT 86) in Berlin.
Authors' addresses: P. Benner, Fakultät für Mathematik, Technische Universität Chemnitz, 09107 Chemnitz, FRG; email: benner@mathematik.tu-chemnitz.de. D. Kressner, Institut für Mathematik, MA4-5, Technische Universität Berlin, 10623 Berlin, FRG; email: kressner@math.tu-berlin.de.

where $A, G, Q \in \mathbb{R}^{n\times n}$. Our subroutines are based on a backward stable, structure-exploiting method developed by Benner, Mehrmann and Xu [1998; 1997]. We have also included recently developed balancing and block algorithms [Benner 2000; Kressner 2003a], which can improve the accuracy and efficiency of this method. The Hamiltonian eigenvalue problem has a number of applications in systems and control theory, see e.g. [Benner et al. 2000; Benner et al. 2003].

Additionally provided are subroutines for computing eigenvalues and invariant subspaces of a *skew-Hamiltonian matrix*

$$W = \begin{bmatrix} A & G \\ Q & A^T \end{bmatrix}, \quad G = -G^T, \quad Q = -Q^T, \tag{2}$$

where again $A, G, Q \in \mathbb{R}^{n\times n}$. They can be used to address complex Hamiltonian eigenvalue problems, see [Benner et al. 1999] and Section 2.4, and might be as well of independent interest, e.g., for solving certain matrix Riccati equations [Stefanovski and Trenčevski 1998] and quadratic eigenvalue problems [Mehrmann and Watkins 2000; Tisseur and Meerbergen 2001].

A Hamiltonian matrix H is equivalently defined by the property $HJ = (HJ)^T$, where

$$J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}. \tag{3}$$

Likewise, a matrix W is skew-Hamiltonian if and only if $WJ = -(WJ)^T$. These matrix structures induce particular spectral properties for H and W. Notably, the eigenvalues of H are symmetric with respect to the imaginary axis, and the eigenvalues of W have even algebraic and geometric multiplicities.

In principle, the eigenvalues of H and W could be obtained by any general-purpose method for computing eigenvalues of general matrices, e.g., the QR algorithm [Golub and Van Loan 1996, Sect. 7.5]. As such a method, however, does not exploit the structures of H and W it cannot be expected to preserve the spectral properties induced by the structure in finite precision arithmetic.

(Skew-)Hamiltonian structures are preserved if symplectic similarity transformations are used. A matrix $S \in \mathbb{R}^{2n\times 2n}$ is called *symplectic* if $S^T JS = J$. In the interest of numerical stability the employed similarity transformations should be orthogonal as well. An algorithm solely based on orthogonal *and* symplectic similarity transformations is *strongly backward stable* [Bunch 1987], i.e., the computed eigenvalues are the exact eigenvalues of a slightly perturbed (skew-)Hamiltonian matrix. Such an algorithm is known for computing the eigenvalues of a skew-Hamiltonian matrix but so far no completely satisfactory algorithm has been found for addressing the Hamiltonian eigenvalue problem. A detailed exposition of (skew-)Hamiltonian eigenvalue problems can be found in the survey [Benner et al. 2004].

This paper should be understood as a sequel to [Benner et al. 2000], where the implementation of an implicit version of the square-reduced method [Van Loan 1984] has been described. This method (implicitly) squares a Hamiltonian matrix and applies a strongly backward stable algorithm to the resulting skew-Hamiltonian matrix. The main disadvantage of the square-reduced method is the squaring part, leading to numerical instabilities which particularly affect eigenvalues of small magnitude. Moreover, computing invariant subspaces via this method is a rather subtle issue [Xu and Lu 1995; Hwang et al. 2003], which is not addressed in [Benner et al. 2000]. The method, on which our implementation is based, uses a similar idea but completely avoids the squaring part leading to numerically backward stably computed eigenvalues. Also, the computed eigenvalues are symmetric with respect to the imaginary axis. It should be said, however, that the computation of invariant subspaces may still suffer from numerical instabilities. Note that the balancing and block algorithms implemented in this paper can be used to improve the performance of the routines implemented in [Benner et al. 2000] as well.

This paper is organized as follows. In the next section we review basic algorithms for solving Hamiltonian and skew-Hamiltonian eigenvalue problems. Several important algorithmic details, such as balancing techniques, block algorithms and variants of the periodic QR algorithm, are summarized in Section 3. The major issues concerning the implementation of the described algorithms are detailed in Section 4. Finally, Section 5 contains some numerical examples illustrating the accuracy of our implementation compared to others.

2. ALGORITHMS

The algorithms implemented here are based on transformations involving orthogonal symplectic matrices. It is easy to see that any matrix belonging to this matrix group can be partitioned as

$$U = \begin{bmatrix} U_1 & U_2 \\ -U_2 & U_1 \end{bmatrix}, \quad U_1, U_2 \in \mathbb{R}^{n \times n}. \tag{4}$$

Two types of elementary orthogonal matrices have this form. These are $2n \times 2n$ Givens rotation matrices of the type

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

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

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

where v is a vector of length n with its first $j-1$ elements equal to zero.

A simple combination of these transformations, see Algorithm 1, can be used to map an arbitrary vector $x \in \mathbb{R}^{2n}$ into the linear space

$$\mathcal{E}_j = \operatorname{span}\{e_1, \dots, e_j, e_{n+1}, \dots, e_{n+j-1}\},$$

where e_i is the ith unit vector of length $2n$. Note that the elements $1, \dots, j-1$ and $n+1, \dots, n+j-1$ of x remain unaffected in Algorithm 1.

Orthogonal symplectic matrices of the form

$$E_j(x) \equiv E_j(v, w, \beta, \gamma, \theta) := (H_j \oplus H_j)(v, \beta) \cdot G_j(\theta) \cdot (H_j \oplus H_j)(w, \gamma), \tag{5}$$

as computed by this algorithm, will be called *elementary*. Let $F = \left[\begin{smallmatrix} 0 & I_n \\ I_n & 0 \end{smallmatrix}\right]$, then we obtain the following variant of elementary orthogonal symplectic matrices:

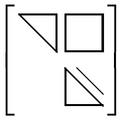
$$[F \cdot E_j(Fx) \cdot F]^T x \in \operatorname{span}\{e_1, \dots, e_{j-1}, e_{n+1}, \dots, e_{n+j}\}.$$

For the sake of brevity we set $E_{n+j}(x) := F \cdot E_j(Fx) \cdot F$, whenever $1 \le j \le n$.

2.1 Eigenvalues of Hamiltonian matrices

The structure-exploiting method for computing eigenvalues of a Hamiltonian matrix H proposed in [Benner et al. 1998] is based on the following idea. First, orthogonal symplectic matrices U and V are computed to reduce H to a so called *symplectic URV form*:

$$U^T H V = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix} =$$



, (6)

i.e., the matrix $R_{21} \in \mathbb{R}^{n \times n}$ is zero, $R_{11} \in \mathbb{R}^{n \times n}$ is upper triangular and $R_{22} \in \mathbb{R}^{n \times n}$ is lower Hessenberg. A simple calculation reveals

$$U^T H^2 U = \begin{bmatrix} -R_{11}R_{22}^T & R_{11}R_{12}^T - R_{12}R_{11}^T \\ 0 & -R_{22}R_{11}^T \end{bmatrix},$$

showing that the eigenvalues of H are the square roots of the eigenvalues of the upper Hessenberg matrix $-R_{11}R_{22}^T$. In a second step, the periodic QR algorithm [Bojanczyk et al. 1992; Hench and Laub 1994; Van Loan 1975] is applied to compute the eigenvalues of this matrix product in a numerically backward stable manner.

Algorithm 2 can be used to compute a symplectic URV decomposition (6). This algorithm is implemented in the subroutine `DGESUV`, which requires $\frac{80}{3}n^3 + \mathcal{O}(n^2)$ floating point operations (flops) to reduce H. The subroutine `DOSGSU` can be used to generate the orthogonal symplectic factors U and V, which requires an additional amount of $\frac{16}{3}n^3 + \mathcal{O}(n^2)$ flops for each factor. Note that Algorithm 2 does not require H to be a Hamiltonian matrix, but even if H is Hamiltonian, this structure will be destroyed making it necessary to provide all elements of H explicitely in a $2n \times 2n$ array.

Algorithm 1

Input: A vector $x \in \mathbb{R}^{2n}$ and an index $j \le n$.
Output: Vectors $v, w \in \mathbb{R}^n$ and $\beta, \gamma, \theta \in \mathbb{R}$ so that

$$[(H_j \oplus H_j)(v,\beta) \cdot G_j(\theta) \cdot (H_j \oplus H_j)(w,\gamma)]^T x \in \mathcal{E}_j.$$

---

(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, see [Golub and Van Loan 1996, p.209].

(2) Determine $\theta \in [-\pi/2, \pi/2)$ such that the $(n+j)$th element of $x \leftarrow G_j(\theta)x$ is zero, see [Golub and Van Loan 1996, p.215].

(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.

Algorithm 2 Symplectic URV decomposition

Input: A matrix $H \in \mathbb{R}^{2n\times 2n}$.
Output: Orthogonal symplectic matrices $U, V \in \mathbb{R}^{2n\times 2n}$; H is overwritten with $U^T HV$ having the form (6).

$U \leftarrow I_{2n},\ V \leftarrow I_{2n}$.
for $j \leftarrow 1, 2, \dots, n$ **do**
  Set $x \leftarrow He_j$.
  Apply Algorithm 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 1 to compute $E_{j+1}(y)$.
    Update $H \leftarrow HE_{n+j+1}(y),\ V \leftarrow VE_{n+j+1}(y)$.
  end if
end for

In the following, we give a brief description of the periodic QR algorithm for the product of a Hessenberg matrix A and an upper triangular matrix B. This algorithm is an iterative procedure aiming at computing orthogonal matrices Q and Z so that $S = Q^T AZ$ is in real Schur form, see [Golub and Van Loan 1996], while $T = Z^T BQ$ remains upper triangular. The eigenvalues of AB equal those of ST and can be easily computed from the diagonal blocks of S and T. Algorithm 3 performs one iteration of the periodic QR algorithm. It can be considered as an implicit application of the standard QR iteration to AB without actually forming this product.

After a few iterations of Algorithm 3 have been applied, one or more subdiagonal elements of A can be expected to become small. We decide on the negligibility of a subdiagonal element using the deflation criteria of the LAPACK [Anderson et al. 1999] subroutines `DHSEQR` and `DLAHQR`. Basically, a subdiagonal element $a_{k+1,k}$ is set to zero if it satisfies

$$|a_{k+1,k}| \le \epsilon \cdot (|a_{kk}| + |a_{k+1,k+1}|), \tag{7}$$

where ϵ denotes the machine precision. After such a deflation has been found, we can partition

$$AB = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix}$$

and apply Algorithm 3 to the matrix products $A_{11}B_{11}$ and $A_{22}B_{22}$ separately. This procedure is repeated until the matrix A is reduced to block diagonal form with one-by-one and two-by-two blocks on the diagonal. Two-by-two blocks corresponding to real eigenvalues of AB are further reduced by applying a single shift version of Algorithm 3.

If one of the diagonal elements of B happens to be small then a “zero chasing” algorithm described in [Bojanczyk et al. 1992] can be used to deflate one zero eigenvalue and two smaller eigenvalue problems. We based the decision whether a

Algorithm 3 Periodic QR iteration

Input: A Hessenberg matrix $A \in \mathbb{R}^{n\times n}$; an upper triangular matrix $B \in \mathbb{R}^{n\times n}$; $n > 2$.

Output: Orthogonal matrices $Q, Z \in \mathbb{R}^{n\times n}$; A and B are overwritten with the Hessenberg matrix $Q^T AZ$ and the upper triangular matrix $Z^T BQ$, respectively. This algorithm applies one iteration of the periodic QR algorithm to AB.

```
Q ← I_n, Z ← I_n.
Compute shifts σ_1 and σ_2 as the eigenvalues of the 2 × 2 bottom right submatrix of AB.
Set x ← (AB − σ_1 I_n)(AB − σ_2 I_n)e_1.
Update A ← H_1(x)A, B ← BH_1(x), Q ← QH_1(x).
Set y ← Be_1.
Update A ← AH_1(y), B ← H_1(y)B, Z ← ZH_1(y).
for j ← 2, 3, ..., n − 1 do
  Set x ← Ae_{j−1}.
  Update A ← H_j(x)A, B ← BH_j(x), Q ← QH_j(x).
  Set y ← Be_j.
  Update A ← AH_j(y), B ← H_j(y)B, Z ← ZH_j(y).
end for
```

diagonal element b_{kk} is negligible on a criterion similar to (7):

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

Algorithm 3 together with the described deflation strategies constitute the periodic QR algorithm, which is implemented in the subroutine `DLAPQR`. Assuming that it takes an average of about two periodic QR iterations to deflate an eigenvalue, this subroutine requires $22n^3 + \mathcal{O}(n^2)$ flops for computing the Schur form of AB. If only the eigenvalues are requested some computational work can be saved by not updating converged parts of the matrices A and B leading to $\frac{44}{3}n^3 + \mathcal{O}(n^2)$ flops. Another $11n^3 + \mathcal{O}(n^2)$ flops are needed to compute each of the orthogonal factors Q and Z.

The subroutine `DHAESU` is a combination of the subroutines `DGESUV` and `DLAPQR`; it computes a symplectic URV decomposition (6) – with R_{22}^T in real Schur form – as well as the eigenvalues of a Hamiltonian matrix H.

2.2 Stable invariant subspaces of Hamiltonian matrices

Note that the approach presented in the previous section only provides the eigenvalues of a Hamiltonian matrix. Invariant subspaces can be obtained by employing the following relationship between the eigenvalues and invariant subspaces of a matrix and an appropriate extension.

THEOREM 2.1 [BENNER ET AL. 1997]. *Let $A \in \mathbb{R}^{n\times n}$ and define $B = \begin{bmatrix} 0 & A \\ A & 0 \end{bmatrix}$. Then $\lambda(B) = \lambda(A) \cup (-\lambda(A))$, where $\lambda(\cdot)$ denotes the set of all eigenvalues of a matrix. Further, let $\lambda(A) \cap \imath\mathbb{R} = \emptyset$. If the columns of the matrix $[Q_1^T, Q_2^T]^T$ span a B-invariant subspace associated with eigenvalues in the open right half plane, then* *the columns of* $Q_1 - Q_2$ *span an* A*-invariant subspace belonging to eigenvalues in the open left half plane.*

An orthogonal basis for the subspace spanned by the columns of $U_1 - U_2$ can be obtained, e.g., from a rank-revealing QR decomposition [Golub and Van Loan 1996] of $U_1 - U_2$. For general matrices it is of course not advisable to use the above result in order to compute invariant subspaces of the matrix A as it would unnecessarily double the dimension of the problem. But if A is a Hamiltonian matrix then the results from the previous section can be used to compute invariant subspaces of the extended matrix B which in turn yield invariant subspaces of A.

To see this, let $H \in \mathbb{R}^{2n\times 2n}$ be Hamiltonian with $\lambda(H) \cap \imath\mathbb{R} = \emptyset$. Then we apply Algorithms 2 and 3 to H. From this we obtain orthogonal symplectic matrices $U = \begin{bmatrix} U_{11} & U_{12} \\ -U_{12} & U_{22} \end{bmatrix}$ and $V = \begin{bmatrix} V_{11} & V_{12} \\ -V_{12} & V_{22} \end{bmatrix}$ such that

$$R := U^T H V = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}$$

where R_{11} is upper triangular and R_{22}^T is quasi upper triangular.

Then

$$B := \begin{bmatrix} U^T & 0 \\ 0 & V^T \end{bmatrix} \begin{bmatrix} 0 & H \\ H & 0 \end{bmatrix} \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} = \begin{bmatrix} 0 & R \\ (JRJ)^T & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & R_{11} & R_{12} \\ 0 & 0 & 0 & R_{22} \\ -R_{22}^T & R_{12}^T & 0 & 0 \\ 0 & -R_{11}^T & 0 & 0 \end{bmatrix}.$$

Swapping the middle block rows/columns of B corresponds to $P^T BP$, where P is the appropriate permutation matrix, and transforms B to block upper triangular form.

Now let $W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix}$ be orthogonal such that

$$W^T \begin{bmatrix} 0 & R_{11} \\ -R_{22}^T & 0 \end{bmatrix} W = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} =: T \tag{8}$$

is quasi upper triangular with all eigenvalues of $T_{11} \in \mathbb{R}^{n\times n}$ and $-T_{22} \in \mathbb{R}^{n\times n}$ located in the open right half plane. Note that this is possible as the eigenvalues of $\begin{bmatrix} 0 & R_{11} \\ -R_{22}^T & 0 \end{bmatrix}$ are exactly those of H, and $\lambda(H) \cap \imath\mathbb{R} = \emptyset$. Hence,

$$\tilde{B} := \begin{bmatrix} W^T & 0 \\ 0 & W^T \end{bmatrix} P^T BP \begin{bmatrix} W & 0 \\ 0 & W \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & C_{11} & C_{12} \\ 0 & T_{22} & C_{12}^T & C_{22} \\ 0 & 0 & -T_{11}^T & 0 \\ 0 & 0 & -T_{12}^T & -T_{22}^T \end{bmatrix}. \tag{9}$$

This implies that the first n columns of $\begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} P \begin{bmatrix} W & 0 \\ 0 & W \end{bmatrix}$ span an invariant subspace of $\begin{bmatrix} 0 & H \\ H & 0 \end{bmatrix}$ belonging to the eigenvalues of T_{11}. Thus, as a corollary of Theorem 2.1 we obtain that the columns of

$$Q = \begin{bmatrix} U_{11}W_{11} - V_{11}W_{21} \\ -U_{12}W_{11} + V_{12}W_{21} \end{bmatrix} \tag{10}$$

span the invariant subspace of H associated with all stable eigenvalues, i.e., the n eigenvalues in the open left half plane. Computing the matrix W in (8) can be implemented efficiently using the underlying structure; for details see [Benner et al. 1997]. The number of flops needed by the overall algorithm is approximately 60% of the number of flops the standard QR algorithm would require to compute the invariant subspace under consideration [Benner et al. 1997]. The subroutine `DHASUB` is based on the described algorithm and computes invariant subspaces of H from the output of the subroutine `DHAESU`. It should be emphasized that this algorithm might encounter numerical difficulties if H has eigenvalues close to the imaginary axis, i.e., it is *not* guaranteed to be *backward stable* for such cases.

The described procedure admits several extensions, also implemented in `DHASUB`. First, assume that not the invariant subspace belonging to all stable eigenvalues but the invariant subspace belonging to $k < n$ selected stable eigenvalues is to be computed. This can be achieved by reordering the corresponding eigenvalues in the quasi-upper triangular matrix product $R_{11}R_{22}^T$ to the top left corner, see [Kressner 2004], and restricting all computations to the first k columns of the matrix Q in (10). By setting $k = 1$ eigenvectors for specified real eigenvalues can be computed. Complex eigenvectors can not be computed directly, but with $k = 2$ and choosing a pair of conjugate complex eigenvalues, eigenvectors can be obtained from the 2-dimensional basis of the provided invariant subspace. Second, invariant subspaces of H belonging to eigenvalues in the open right half plane can be computed by a variant of Theorem 2.1 saying that the columns of $Q_1 + Q_2$ span such an invariant subspace. Finally, if the Hamiltonian matrix H has eigenvalues on or close to the imaginary axis then the numerical rank of the matrix Q in (10) is likely to be less than n. In this case, it is preferable to reorder the eigenvalues of the $4n \times 4n$ matrix $\tilde{B}$ in (9) such that all eigenvalues in the upper $2n \times 2n$ block are in the open right half plane. This can be achieved, e.g., by symplectic reordering algorithms, see [Byers 1983; Benner et al. 2004]; the subroutine `DHAORD` is an implementation of the reordering algorithm described in [Benner et al. 2004]. With these algorithms it is possible to determine an orthogonal symplectic matrix Z such that

$$Z^T \tilde{B} Z = \begin{bmatrix} T_{11} & \tilde{T}_{12} & C_{11} & \tilde{C}_{12} \\ 0 & \tilde{T}_{22} & \tilde{C}_{12}^T & \tilde{C}_{22} \\ 0 & 0 & -T_{11}^T & 0 \\ 0 & 0 & -\tilde{T}_{12}^T & -\tilde{T}_{22}^T \end{bmatrix},$$

where $\tilde{B}$ is the matrix defined in (9) and all the eigenvalues of $\tilde{T}_{22}$ lie in the open right half plane. Now define

$$\tilde{Q} := \begin{bmatrix} \tilde{Q}_{11} & \tilde{Q}_{12} \\ \tilde{Q}_{21} & \tilde{Q}_{22} \end{bmatrix} := \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} P \begin{bmatrix} W & 0 \\ 0 & W \end{bmatrix} Z, \tag{11}$$

then by construction the columns of the matrix $[\tilde{Q}_{11}^T, \tilde{Q}_{21}^T]^T$ span the invariant subspace for $\left[\begin{smallmatrix} 0 & H \\ H & 0 \end{smallmatrix}\right]$ associated with all eigenvalues in the open right half plane. Again by an application of Theorem 2.1, we obtain the invariant subspace of H associated with all stable eigenvalues using a rank-revealing QR decomposition of $\tilde{Q}_{11} - \tilde{Q}_{21}$.

2.3 Enforcing isotropy via the symplectic QR decomposition

Any invariant subspace $\mathcal{X} \subseteq \mathbb{R}^{2n}$ belonging to a set of stable eigenvalues of a Hamiltonian matrix is isotropic, i.e., $\mathcal{X}$ is perpendicular to $J\mathcal{X}$. This implies that any matrix X with $\mathcal{X} = \text{span}(X)$ satisfies $X^T J X = 0$. Enforcing this property in finite-precision arithmetic is sometimes important in applications. For example, it is the isotropy of the invariant subspace belonging to all stable eigenvalues of a Hamiltonian matrix which implies that the stabilizing solution of the corresponding algebraic Riccati equation is symmetric [Lancaster and Rodman 1995].

Unfortunately, the methods described in Section 2.2 are not capable to guarantee the isotropy of a computed invariant subspace. Nevertheless, a basis for a nearby isotropic subspace can be constructed via the symplectic QR decomposition, which is defined as follows. Let $X \in \mathbb{R}^{2n\times k}$ with $n \geq k$, then there exists an orthogonal symplectic matrix Q so that $X = QR$ and

$$R = \begin{bmatrix} R_{11} \\ R_{21} \end{bmatrix}, \quad R_{11} = \begin{bmatrix} \diagdown \\ 0 \end{bmatrix}, \quad R_{21} = \begin{bmatrix} {}^{0}\!\diagdown_{0} \\ 0 \end{bmatrix}, \tag{12}$$

i.e., 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 [Bunse-Gerstner 1986]. Then the first k columns of Q span an isotropic subspace that will be close to $\mathcal{X}$ under the assumption that $\mathcal{X}$ admits a nearby isotropic subspace, see e.g. [Benner et al. 2004].

Algorithm 4, which is implemented in the subroutines `DGESQR` and `DOSGSQ`, provides a straightforward way to compute a symplectic QR decomposition (12). It requires $8(k^2 n - k^3/3) + \mathcal{O}(k^2)$ flops for computing the reduced matrix R, and additionally $16kn^2 - 16k^2 n + \frac{16}{3}k^3 + \mathcal{O}(k^2)$ flops for accumulating the orthogonal symplectic factor Q in reversed order.

Algorithm 4 Symplectic QR decomposition

Input: A general matrix $X \in \mathbb{R}^{2n\times k}$ with $n \geq k$.
Output: An orthogonal symplectic matrix $Q \in \mathbb{R}^{2n\times 2n}$; X is overwritten with $R = Q^T X$ having the form (12).

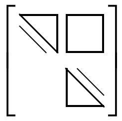
$Q \leftarrow I_{2n}$.
for $j \leftarrow 1, \ldots, k$ **do**
  Set $x \leftarrow X e_j$.
  Apply Algorithm 1 to compute $E_j(x)$.
  Update $X \leftarrow E_j(x)^T X$, $Q \leftarrow Q E_j(x)$.
end for

Note that the symplectic QR decomposition has a number of other applications, such as the symplectic integration of Hamiltonian systems [Leimkuhler and Van Vleck 1997].

2.4 Eigenvalues of skew-Hamiltonian matrices

Computing the eigenvalues of a skew-Hamiltonian matrix W as proposed by Van Loan [1984] is considerably simple. First, an orthogonal symplectic matrix U is computed to reduce W to a so called *Paige/Van Loan (PVL) form*:

$$U^T W U = \begin{bmatrix} R_{11} & R_{12} \\ R_{12} & R_{11}^T \end{bmatrix} = \tag{13}$$



i.e., the matrix $R_{21} \in \mathbb{R}^{n \times n}$ is zero and $R_{11} \in \mathbb{R}^{n \times n}$ is upper Hessenberg. Second, the standard QR algorithm is applied to compute the eigenvalues of R_{11}, which are the eigenvalues of W with halved multiplicities.

Algorithm 5 PVL decomposition

Input:	A skew-Hamiltonian matrix $W \in \mathbb{R}^{2n \times 2n}$.
Output:	An orthogonal symplectic matrix $U \in \mathbb{R}^{2n \times 2n}$; W is overwritten with $U^T W U$ having the form (13).

$U \leftarrow I_{2n}$.
for $j \leftarrow 1, 2, \ldots, n-1$ **do**
  Set $x \leftarrow W e_j$.
  Apply Algorithm 1 to compute $E_{j+1}(x)$.
  Update $W \leftarrow E_{j+1}(x)^T W E_{j+1}(x)$, $U \leftarrow U E_{j+1}(x)$.
end for

Algorithm 5 computes a PVL decomposition of the form (13). The subroutine `DSHPVL` is based on this algorithm and requires $\frac{40}{3}n^3 + \mathcal{O}(n^2)$ flops for reducing W. The generation of the orthogonal symplectic factor U is implemented in the subroutine `DOSGPV`, which requires an additional amount of $\frac{16}{3}n^3 + \mathcal{O}(n^2)$ flops.

Subsequent to Algorithm 5, the standard QR algorithm is applied to the matrix R_{11}, producing an orthogonal matrix Q so that

$$\tilde{U}^T W \tilde{U} = \begin{bmatrix} \tilde{R}_{11} & \tilde{R}_{12} \\ 0 & \tilde{R}_{11}^T \end{bmatrix} \tag{14}$$

where $\tilde{U} = U \left[\begin{smallmatrix} Q & 0 \\ 0 & Q \end{smallmatrix}\right]$ and $\tilde{R}_{11}$ has real Schur form. A decomposition of the form (14) is called a *skew-Hamiltonian Schur decomposition* of W. It is produced by the subroutine `DSHES`, which requires only approximately 21% the number of flops the standard QR algorithm would require to compute the unstructured, real Schur decomposition of W, see e.g. [Kressner 2004].

2.5 Invariant subspaces of skew-Hamiltonian matrices

Having computed a real Schur decomposition of the form (14), the first $k \leq n$ columns of $\tilde{U}$ span an isotropic invariant subspace of the skew-Hamiltonian matrix W if the $(k+1, k)$ entry of $\tilde{R}_{11}$ is zero. Other isotropic invariant subspaces can be obtained by swapping the diagonal blocks of $\tilde{R}_{11}$ as described, e.g., in [Bai and Demmel 1993]. This is implemented in the subroutine `DHAORD`, which can also be used to swap diagonal blocks between $\tilde{R}_{11}$ and $\tilde{R}_{11}^T$.

2.6 Eigenvalues of complex Hamiltonian matrices

A *complex Hamiltonian matrix* $H \in \mathbb{C}^{2n \times 2n}$ can be defined by the property $(JH)^\star = JH$, with the matrix J as in (3). As in the real case, the eigenvalues of H come in pairs $\{\lambda, -\lambda\}$. Algorithm 5 can be used to compute these eigenvalues in a structure-preserving manner.

To see this, let us decompose $H = H_R + \imath H_I$ such that $H_R, H_I \in \mathbb{R}^{2n \times 2n}$. Then the relationship $(JH)^\star = JH$ implies that the two matrices H_R and H_I have the following structure:

$$H_R = \begin{bmatrix} A_R & G_R \\ Q_R & -A_R^T \end{bmatrix}, \quad H_I = \begin{bmatrix} A_I & G_I \\ Q_I & A_I^T \end{bmatrix},$$

where $G_R = G_R^T, Q_R = Q_R^T, G_I = -G_I^T, Q_I = -Q_I^T$, i.e., the matrix H_R is Hamiltonian and the matrix H_I is skew-Hamiltonian. Note that the $4n \times 4n$ skew-Hamiltonian matrix

$$W = \left[\begin{array}{cc|cc} A_I & A_R & G_I & G_R \\ -A_R & A_I & -G_R & G_I \\ \hline Q_I & Q_R & A_I^T & -A_R^T \\ -Q_R & Q_I & A_R^T & A_I^T \end{array}\right] \tag{15}$$

is permutationally similar to the matrix $\left[\begin{smallmatrix} H_I & H_R \\ -H_R & H_I \end{smallmatrix}\right]$. This implies that λ is an eigenvalue of H if and only if $\{-\imath\lambda, \imath\bar{\lambda}\}$ is an eigenvalue pair of W.

Now, if we compute a PVL decomposition (13) of W, then it is sufficient to consider the $2n$ eigenvalues of W belonging to the real matrix R_{11}. These eigenvalues come in pairs $\{\mu, \bar{\mu}\}$, each of which corresponds to an eigenvalue pair $\{\imath\mu, -\imath\bar{\mu}\}$ of H. For a non-real eigenvalue μ of W, the pair $\{\mu, \bar{\mu}\}$ can be easily detected as a two-by-two block in the real Schur form of R_{11}. A real eigenvalue μ, which corresponds to a purely imaginary eigenvalue of H, has multiplicity two. However, this property will be lost in finite-precision arithmetic. Thus, we are faced with the decision which of the computed real eigenvalues of R_{11} are actually duplicates and can be neglected. In lack of more sophisticated decision criteria, we propose to sort the real eigenvalues of W in descending order and neglect every other real eigenvalue.

Note that an embedding of the form (15) can also be used to obtain invariant subspaces of complex Hamiltonian matrices, see [Benner et al. 1999].

3. ALGORITHMIC DETAILS

This section summarizes various algorithmic details such as balancing and block algorithms. These techniques aim at improving the algorithms described in Section 2, making them competitive to the general-purpose eigenvalue solvers implemented in LAPACK.

3.1 Symplectic Balancing

Balancing is a beneficial pre-processing step for computing eigenvalues of matrices and has been studied, e.g., in [Osborne 1960; Parlett and Reinsch 1969]. A special-purpose balancing algorithm that is based on symplectic equivalence transformations and consequently preserves Hamiltonian structures has been proposed in [Benner 2000]. It consists of two stages, which are described in the following two subsections.

3.1.1 *Isolating Eigenvalues.* The first stage consists of permuting a Hamiltonian matrix H in order to isolate as many of its eigenvalues as possible. Although it is tempting to require the facilitated permutations to be symplectic, it has turned out that such a requirement leads to rather complicated reduced forms [Benner 2000]. Instead, it was proposed in [Benner 2000; Kressner 2004] to broaden the range of similarity transformations to $\tilde{P}^T H \tilde{P}$, where $\tilde{P} = DP$ is symplectic, $D = \mathrm{diag}\{\pm 1, \ldots, \pm 1\}$ and P is a permutation matrix. These *symplectic generalized permutation matrices* clearly form a group, which can be generated by the following two classes of elementary matrices:

$$P_{ij}^{(d)} = P_{ij} \oplus P_{ij}, \tag{16}$$

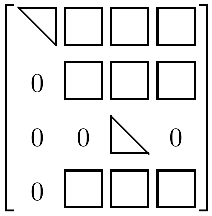
where $1 \le i < j \le n$, $P_{ij} = I - e_i e_i^T - e_j e_j^T + e_i e_j^T + e_j e_i^T$, and

$$P_i^{(s)} = I_{2n} - \begin{bmatrix} e_i & e_{i+n} \end{bmatrix} \begin{bmatrix} e_i^T \\ e_{i+n}^T \end{bmatrix} + \begin{bmatrix} e_i & -e_{i+n} \end{bmatrix} \begin{bmatrix} e_{i+n}^T \\ e_i^T \end{bmatrix}, \tag{17}$$

where $1 \le i < n$. If H is post-multiplied by $P_{ij}^{(d)}$ then columns $i \leftrightarrow j$ and columns $(n+i) \leftrightarrow (n+j)$ of H are swapped. A post-multiplication by $P_i^{(s)}$ swaps columns $i \leftrightarrow (n+i)$ and scales the ith column by -1. Analogous statements hold for the rows of H if this matrix is pre-multiplied by $P_{ij}^{(d)}$ or $P_i^{(s)}$.

Combinations of these matrices can be used to compute a symplectic generalized permutation matrix $\tilde{P}$ so that

$$\tilde{P}^T H \tilde{P} = \begin{bmatrix} A_{11} & A_{21} & G_{11} & G_{12} \\ 0 & A_{22} & G_{12}^T & G_{22} \\ 0 & 0 & -A_{11}^T & 0 \\ 0 & Q_{22} & -A_{21}^T & -A_{22}^T \end{bmatrix} =$$



, (18)

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. An algorithm that produces the block triangular form (18) can be developed along the lines of algorithms for isolating eigenvalues of general matrices, see [Kressner 2004] for more details.

3.1.2 *Scaling.* The second stage of symplectic balancing consists of finding a diagonal matrix D so that

$$(D \oplus D^{-1})^{-1} \begin{bmatrix} A_{22} & G_{22} \\ Q_{22} & -A_{22}^T \end{bmatrix} (D \oplus D^{-1}) = \begin{bmatrix} D^{-1} A_{22} D & D^{-1} G_{22} D^{-1} \\ D Q_{22} D & -(D^{-1} A_{22} D)^T \end{bmatrix}$$

is nearly balanced in 1-norm, i.e., the rows and columns of this matrix are nearly equal in 1-norm.

An iterative procedure achieving this aim has been developed in [Benner 2000], in the spirit of the Parlett-Reinsch algorithm for equilibrating the row and column norms of a general matrix [Parlett and Reinsch 1969]. It converges if there is no restriction on the diagonal entries of D and under the assumption that $\begin{bmatrix} A_{22} & G_{22} \\ Q_{22} & A_{22}^T \end{bmatrix}$ is irreducible. Note that, strictly speaking, this assumption is not satisfied by submatrices of the block triangular form (18). A structure-preserving block triangular form yielding irreducible Hamiltonian submatrices has been presented in [Benner and Kressner 2003]. The construction of this form, however, requires graph-theoretic tools that are more suitable for large and sparse matrices.

Algorithm 6 is basically our implementation of the symplectic scaling procedure described in [Benner 2000]. To avoid any roundoff errors in this algorithm, the scaling factor β should be a power of the machine base (usually 2). By exploiting

Algorithm 6 Symplectic Scaling

Input: A Hamiltonian matrix $H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix} \in \mathbb{R}^{2n\times 2n}$ having the block triangular form (18) for an integer i_l. A scaling factor $\beta \in \mathbb{R}$.

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

$\tilde{D} \leftarrow I_n$
converged $\leftarrow 0$
while converged $= 0$ **do**
  converged $\leftarrow 1$
  for $j \leftarrow i_l, \dots, n$ **do**
    $c \leftarrow \sum_{\substack{i=i_l \\ i\neq j}}^{n} (|a_{ij}| + |q_{ij}|), \quad r \leftarrow \sum_{\substack{k=i_l \\ k\neq j}}^{n} (|a_{jk}| + |g_{jk}|), \quad \delta_q \leftarrow |q_{jj}|, \quad \delta_g \leftarrow |g_{jj}|$
    $s \leftarrow c + r, \quad \mathrm{scal} \leftarrow 1$
    while $((r + \delta_g/\beta)/\beta) \geq ((c + \delta_q \cdot \beta) \cdot \beta)$ **do**
      $c \leftarrow c \cdot \beta, \quad r \leftarrow r/\beta, \quad \delta_q \leftarrow \delta_q \cdot \beta^2, \quad \delta_g \leftarrow \delta_g/\beta^2$
      scal $\leftarrow$ scal $\cdot \beta$
    end while
    while $((r + \delta_g \cdot \beta) \cdot \beta) \leq ((c + \delta_q/\beta)/\beta)$ **do**
      $c \leftarrow c/\beta, \quad r \leftarrow r \cdot \beta, \quad \delta_q \leftarrow \delta_q/\beta^2, \quad \delta_g \leftarrow \delta_g \cdot \beta^2$
      scal $\leftarrow$ scal$/\beta$
    end while
    % Balance if necessary.
    if scal $\neq 1$ **then**
      converged $\leftarrow 0$, $\quad \tilde{d}_{jj} \leftarrow \mathrm{scal} \cdot \tilde{d}_{jj}, \quad \tilde{d}_{n+j,n+j} \leftarrow 1/\mathrm{scal} \cdot \tilde{d}_{n+j,n+j}$
      $A(:,j) \leftarrow \mathrm{scal} \cdot A(:,j), \qquad A(j,:) \leftarrow 1/\mathrm{scal} \cdot A(j,:)$
      $G(:,j) \leftarrow 1/\mathrm{scal} \cdot G(:,j), \qquad G(j,:) \leftarrow 1/\mathrm{scal} \cdot G(j,:)$
      $Q(:,j) \leftarrow \mathrm{scal} \cdot Q(:,j), \qquad Q(j,:) \leftarrow \mathrm{scal} \cdot Q(j,:)$
    end if
  end for
end while

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 6 only needs to balance the first n rows and columns of H. It can thus be concluded that it requires about half the number of operations required by the Parlett-Reinsch algorithm applied to H.

Both ingredients of symplectic balancing, the construction of the block triangular form (18) and Algorithm 6, are implemented in the subroutine `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 $P^{(d)}_{\mathrm{scal}(j)-n,j} P^{(s)}_{\mathrm{scal}(j)}$ (if $\mathrm{scal}(j) > n$) has been applied in the course of constructing the block triangular form (18). Otherwise, $\mathrm{scal}(j)$ contains $\tilde{d}_{jj}$, the jth diagonal entry of the diagonal matrix $\tilde{D}$ returned by Algorithm 6.

The backward transformation, i.e., multiplication with $(P\tilde{D})^{-1}$, is implemented in the subroutine `DHABAK`. Slight modifications of the described algorithms can be used for balancing skew-Hamiltonian matrices, see [Benner 2000]. These modified algorithms are implemented in the subroutine `DSHBAL`.

Symplectic balancing a (skew-)Hamiltonian matrix has essentially the same positive effects that are attributed to balancing a general matrix. First, the norm of the matrix H is often decreased which equally decreases the norm of the backward error caused by the subsequent orthogonal transformations. Second, eigenvalues isolated by the block triangular form (18) are read off without any roundoff errors. Finally, balancing can have a positive impact on the computational time needed by subsequent methods for computing eigenvalues. Numerical experiments confirming these statements can be found in [Benner 2000; Kressner 2004] and Section 5.

3.2 Block algorithms

The LAPACK subroutines for computing QR, Hessenberg, bidiagonal and similar decompositions attain high efficiency by (implicitly) employing compact WY representations [Bischof and Van Loan 1987; Schreiber and Van Loan 1989] of the involved orthogonal transformations. The following theorem describes a variant of this representation, suitable for elementary orthogonal symplectic matrices as defined in (5).

THEOREM 3.1. *[Kressner 2003a] Let $k \leq n$ and $Q = E_{j_1}(x_1) \cdot E_{j_2}(x_2) \cdots E_{j_k}(x_k)$, where the elementary matrices $E_{j_i}(x_i)$ are defined as in (5) with $j_i \in [1, n]$ and $x_i \in \mathbb{R}^{2n}$. Then there exist matrices $R \in \mathbb{R}^{3k \times k}, S \in \mathbb{R}^{k \times 3k}, T \in \mathbb{R}^{3k \times 3k}$ and $W \in \mathbb{R}^{n \times 3k}$ so that*

$$Q = \begin{bmatrix} I_n + WTW^T & WRSW^T \\ -WRSW^T & I_n + WTW^T \end{bmatrix}. \tag{19}$$

Furthermore, these matrices can be partitioned as

$$R = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix}, \quad S = \begin{bmatrix} S_1 & S_2 & S_3 \end{bmatrix}, \quad T = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix},$$

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* i*th column* e_{j_i}*, the* j_i*th column of the* $n \times n$ *identity matrix.*

The subroutine DLAEST can be used to construct WY-like representations of the form (19) and requires $(4k-2)kn + \frac{19}{3}k^3 + \mathcal{O}(k^2)$ flops. Taking care of the generic structures present in R, S, T and W, the subroutine DLAESB applies the WY-like representation (19) to a $2n \times q$ matrix, which requires $16k(n-k)q + \mathcal{O}(kq)$ flops. This subroutine attains high efficiency by making exclusive use of calls to level 3 BLAS [Dongarra et al. 1990].

Similar to the development of block algorithms for orthogonal decompositions, see [Dongarra et al. 1989], the WY-like representation (19) can be used to derive block variants of Algorithms 2, 4 and 5 for computing symplectic URV, QR and PVL decompositions. These block algorithms are implemented in subroutines DGESQB, DGESUB, and DHAPVB, respectively. For further details, the reader is referred to [Kressner 2003a].

3.3 Multi-shift variants of the periodic QR algorithm

The LAPACK implementation of the QR algorithm is capable of employing an arbitrary number of shifts simultaneously in each QR iteration [Bai and Demmel 1989]. Larger shift numbers lead to larger Householder matrices during the QR iteration and enable the more effective use of level 2 BLAS, which in turn improves the performance of the QR algorithm. Note, however, that this approach must be carefully considered; it has been shown that the convergence of such a multi-shift QR algorithm is severely affected by roundoff-errors if the number of simultaneous shifts is too large [Dubrulle 1991; Watkins 1995]. The LAPACK subroutine DHSEQR therefore uses a limited number of shifts (by default six).

Similarly, multi-shift variants of the periodic QR algorithm can be developed by admitting more than two shifts in the definition of the vector x in Algorithm 3, see [Kressner 2003b] for more details. Such a variant has been implemented in the subroutine DHGPQR.

4. IMPLEMENTATION

All subroutines have been implemented in Fortran 77 in accordance with the SLICOT [Benner et al. 1999] implementation and documentation standards [1996]. This implies that the header of each subroutine contains an elaborate documentation describing inputs, outputs and functionality. In this section, our focus will therefore only be on the most important implementation issues.

A list of all driver, computational and auxiliary subroutines can be found in Appendix A.

4.1 Naming convention

Similar to LAPACK [Anderson et al. 1999] the name of each subroutine has the form **XYYZZZ**. Here, the letter **X** specifies the data type of the matrices to be processed. Since we only support double real precision data, this letter is D for all subroutines discussed in this paper. The letters **YY** indicate the type and structure of the involved matrices. As an extension of the LAPACK naming scheme and similar to [Benner et al. 2000], we use HA, OS, SH and SK to denote Hamiltonian, orthogonal symplectic, skew-Hamiltonian and skew-symmetric matrices, respectively. The last three letters **ZZZ** are used to designate the computation to be performed.

4.2 Storage layout

A $2n \times 2n$ Hamiltonian matrix $H = \left[\begin{smallmatrix} A & G \\ Q & -A^T \end{smallmatrix}\right]$ can be represented by $2n^2 + n$ parameters. To avoid redundancy in the data representation of H we use the packed storage layout proposed in [Benner et al. 2000]. While the submatrix A is stored in a conventional $n \times n$ array `A`, the symmetric submatrices G and Q are stored in an $n \times (n+1)$ array `QG` as illustrated in Figure 1. The skew-symmetric parts of a skew-Hamiltonian matrix are similarly stored, with the notable difference that the parts containing the diagonal and the first superdiagonal of the array `QG` are not referenced.

Hamiltonian	skew-Hamiltonian
$\mathtt{QG} = \begin{bmatrix} q_{11} & g_{11} & g_{12} & g_{13} & \cdots \\ q_{21} & q_{22} & g_{22} & g_{23} & \cdots \\ q_{31} & q_{32} & q_{33} & g_{33} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$	$\mathtt{QG} = \begin{bmatrix} \star & \star & g_{12} & g_{13} & \cdots \\ q_{21} & \star & \star & g_{23} & \cdots \\ q_{31} & q_{32} & \star & \star & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$

Fig. 1. Storage layout for the (skew-)symmetric submatrices G and Q of a (skew-)Hamiltonian matrix.

An orthogonal symplectic matrix $U = \left[\begin{smallmatrix} U_1 & U_2 \\ -U_2 & U_1 \end{smallmatrix}\right]$ is stored in two $n \times n$ arrays `U1` and `U2` containing the submatrices U_1 and U_2, respectively.

4.3 Examples and testing

Following the style of the SLICOT library, each of the driver and main computational subroutines is accompanied by an example program with a set of corresponding input and output data. The purpose of these example programs is to give users a demonstration of the straightforward application of the subroutine to solve a simple problem. If the name of the subroutine is **XYYZZZ** then the example program can be found in the file `T`**XYYZZZ**`.f` while the input and output data can be found in **XYYZZZ**`.dat` and **XYYZZZ**`.res`, respectively. In order to validate the functionality of a compiled subroutine, users can compare the delivered output with the output contained in the corresponding `.res` file.

4.4 Matlab interfaces

To enhance user-friendliness, MATLAB [The MathWorks, Inc. 2002] mex interfaces provide the main functionality of the discussed subroutines in a convenient way to MATLAB users. These interfaces, described in more detail in [Kressner 2004], are available at `http://www.tu-chemnitz.de/mathematik/hapack/matlab/`.

5. NUMERICAL EXAMPLES

In this section, we provide some numerical examples to illustrate the accuracy of the presented subroutines in comparison with the standard QR algorithm implemented in LAPACK [Anderson et al. 1999] and the square-reduced method implemented in [Benner et al. 2000]. For experiments comparing the execution times, we refer to [Benner et al. 2003; Kressner 2003a; 2004]. Summarizing these experiments, it can be said that the observed execution times essentially confirm the expectations raised by the flop counts. Furthermore, the block algorithms described in Section 3.2 make the performance of our subroutines competitive with the corresponding LAPACK subroutines. For example, when the eigenvalues of a Hamiltonian matrix are to be computed, the subroutine `DHAESU` requires approximately 60% of the time needed by LAPACK while being only slightly slower than the square-reduced method.

5.1 Accuracy of eigenvalues

The algorithms implemented in this paper preserve the eigenvalue symmetries of a Hamiltonian matrix. This symmetry is of particular importance if small perturbations, caused by roundoff errors, could move eigenvalues across the imaginary axis. Applications like designing (sub-)optimal H_∞ controllers [Benner et al. 2004; Zhou et al. 1996], computing H_∞ norms, stability radii and pseudospectra [Byers 1988; Boyd et al. 1989; Burke et al. 2003] require a safe decision whether an eigenvalue is on or close to the imaginary axis. Moreover, a false decision can make it impossible to find the stable invariant subspace.

Let us consider the following Hamiltonian matrix, see [Arnold, III and Laub 1984, Example 2] and [Abels and Benner 1999, Example 2.8]:

$$H = \begin{bmatrix} A & ee^T \\ ee^T & -A^T \end{bmatrix}, \quad A = \begin{bmatrix} -10^{-6} & 1 & 0 & 0 \\ -1 & -10^{-6} & 0 & 0 \\ 0 & 0 & 10^{-6} & 1 \\ 0 & 0 & -1 & 10^{-6} \end{bmatrix},$$

where e is the vector of all ones. The Hamiltonian matrix H has a quadruple of eigenvalues very close to the imaginary axis:

$$\lambda = \pm 0.500000000000375 \cdot 10^{-12} \pm 0.999999999999500\imath.$$

When applying the QR algorithm to this matrix then the real parts of these eigenvalues are perturbed by a relative error of up to 5.77×10^{-4}. This compares to 7.81×10^{-6} for the subroutine `DHAESU` and 1.36×10^{-4} for the square-reduced method. Hence, for this example, our routines gain two digits of accuracy in the imaginary parts of the critical eigenvalues.

Table I gives an account on the eigenvalue accuracy of the QR algorithm with balancing (`QR`), the square-reduced method with balancing as proposed in [Benner et al. 2000] (`SQRED`) and with symplectic balancing (`SQBAL`), as well as the subroutine `DHAESU` for all Hamiltonian matrices from the benchmark collection [Abels and Benner 1999]. The following quantities are displayed:

$$\text{fwd} = \max_i \frac{|\hat{\lambda}_i - \lambda_i|}{\|H\|_2}, \quad \text{bwd} = \max_i \frac{\sigma_{\min}(H - \hat{\lambda}_i I_{2n})}{\|H\|_2},$$

Ex	QR	SQRED	SQBAL	DHAESU
1.1	$2.8{\cdot}10^{-16}(1.0{\cdot}10^{-08})$	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
1.2	$7.4{\cdot}10^{-17}(6.9{\cdot}10^{-18})$	$7.4{\cdot}10^{-17}(6.9{\cdot}10^{-18})$	$7.4{\cdot}10^{-17}(6.9{\cdot}10^{-18})$	$7.1{\cdot}10^{-17}(1.0{\cdot}10^{-16})$
1.3	$6.6{\cdot}10^{-16}(1.3{\cdot}10^{-15})$	$6.3{\cdot}10^{-17}(1.8{\cdot}10^{-16})$	$8.8{\cdot}10^{-17}(1.9{\cdot}10^{-16})$	$2.5{\cdot}10^{-16}(4.2{\cdot}10^{-16})$
1.4	$1.1{\cdot}10^{-15}(1.1{\cdot}10^{-15})$	$3.3{\cdot}10^{-16}(2.3{\cdot}10^{-16})$	$3.3{\cdot}10^{-16}(2.3{\cdot}10^{-16})$	$1.5{\cdot}10^{-15}(1.4{\cdot}10^{-15})$
1.5	$2.1{\cdot}10^{-16}(2.5{\cdot}10^{-15})$	$1.6{\cdot}10^{-15}(4.5{\cdot}10^{-15})$	$6.3{\cdot}10^{-16}(1.8{\cdot}10^{-15})$	$6.7{\cdot}10^{-17}(8.0{\cdot}10^{-16})$
1.6	$4.1{\cdot}10^{-20}(9.6{\cdot}10^{-20})$	$1.0{\cdot}10^{-17}(2.9{\cdot}10^{-16})$	$2.1{\cdot}10^{-20}(2.9{\cdot}10^{-19})$	$3.3{\cdot}10^{-20}(6.8{\cdot}10^{-21})$
2.1	$1.0{\cdot}10^{-16}(1.5{\cdot}10^{-17})$	$1.0{\cdot}10^{-16}(1.5{\cdot}10^{-17})$	$1.0{\cdot}10^{-16}(1.5{\cdot}10^{-17})$	$1.1{\cdot}10^{-16}(6.0{\cdot}10^{-17})$
2.2	$2.4{\cdot}10^{-18}(1.3{\cdot}10^{-17})$	$9.5{\cdot}10^{-16}(9.6{\cdot}10^{-14})$	$9.5{\cdot}10^{-16}(9.6{\cdot}10^{-14})$	$2.4{\cdot}10^{-18}(5.9{\cdot}10^{-18})$
2.3	$4.5{\cdot}10^{-19}(9.8{\cdot}10^{-19})$	$1.6{\cdot}10^{-19}(8.4{\cdot}10^{-20})$	$1.6{\cdot}10^{-19}(6.5{\cdot}10^{-20})$	$2.3{\cdot}10^{-19}(8.0{\cdot}10^{-20})$
2.4	$7.2{\cdot}10^{-16}(3.3{\cdot}10^{-11})$	$1.9{\cdot}10^{-16}(6.4{\cdot}10^{-10})$	$1.9{\cdot}10^{-16}(6.4{\cdot}10^{-10})$	$2.1{\cdot}10^{-16}(2.0{\cdot}10^{-16})$
2.5	$7.8{\cdot}10^{-17}(2.4{\cdot}10^{-09})$	$2.5{\cdot}10^{-17}(1.6{\cdot}10^{-09})$	$2.5{\cdot}10^{-17}(1.6{\cdot}10^{-09})$	$7.8{\cdot}10^{-17}(1.9{\cdot}10^{-09})$
2.6	$7.2{\cdot}10^{-16}(8.7{\cdot}10^{-16})$	$1.3{\cdot}10^{-16}(5.4{\cdot}10^{-17})$	$1.3{\cdot}10^{-16}(5.4{\cdot}10^{-17})$	$3.2{\cdot}10^{-16}(2.2{\cdot}10^{-16})$
2.7	$1.4{\cdot}10^{-22}(7.0{\cdot}10^{-22})$	$1.0{\cdot}10^{-22}(3.9{\cdot}10^{-22})$	$3.7{\cdot}10^{-21}(2.7{\cdot}10^{-20})$	$1.4{\cdot}10^{-22}(9.4{\cdot}10^{-22})$
2.8	$3.4{\cdot}10^{-16}(5.2{\cdot}10^{-16})$	$3.4{\cdot}10^{-16}(3.3{\cdot}10^{-16})$	$3.4{\cdot}10^{-16}(3.3{\cdot}10^{-16})$	$9.2{\cdot}10^{-17}(6.3{\cdot}10^{-17})$
2.9	$9.1{\cdot}10^{-23}(1.8{\cdot}10^{-23})$	$6.0{\cdot}10^{-20}(4.4{\cdot}10^{-14})$	$5.6{\cdot}10^{-23}(1.3{\cdot}10^{-20})$	$2.8{\cdot}10^{-23}(5.4{\cdot}10^{-23})$
3.1	$3.0{\cdot}10^{-16}(7.4{\cdot}10^{-16})$	$1.3{\cdot}10^{-16}(8.4{\cdot}10^{-16})$	$1.3{\cdot}10^{-16}(8.4{\cdot}10^{-16})$	$1.5{\cdot}10^{-16}(6.1{\cdot}10^{-16})$
3.2	$2.4{\cdot}10^{-15}(2.4{\cdot}10^{-15})$	$2.7{\cdot}10^{-15}(2.7{\cdot}10^{-15})$	$2.7{\cdot}10^{-15}(2.7{\cdot}10^{-15})$	$3.4{\cdot}10^{-15}(3.4{\cdot}10^{-15})$
4.1	$2.3{\cdot}10^{-15}(2.4{\cdot}10^{-15})$	$8.6{\cdot}10^{-16}(9.2{\cdot}10^{-16})$	$8.6{\cdot}10^{-16}(9.2{\cdot}10^{-16})$	$1.2{\cdot}10^{-15}(1.3{\cdot}10^{-15})$
4.2	$9.5{\cdot}10^{-15}$	$3.1{\cdot}10^{-15}$	$3.1{\cdot}10^{-15}$	$4.9{\cdot}10^{-15}$
4.3	$1.2{\cdot}10^{-15}(8.4{\cdot}10^{-15})$	$9.3{\cdot}10^{-16}(7.6{\cdot}10^{-15})$	$9.3{\cdot}10^{-16}(7.6{\cdot}10^{-15})$	$9.1{\cdot}10^{-16}(1.7{\cdot}10^{-15})$
4.4	$4.5{\cdot}10^{-20}$	$2.9{\cdot}10^{-13}$	$3.4{\cdot}10^{-16}$	$6.2{\cdot}10^{-20}$
(20)	$5.9{\cdot}10^{-16}(1.3{\cdot}10^{-16})$	$1.1{\cdot}10^{-09}(1.1{\cdot}10^{-09})$	$1.1{\cdot}10^{-09}(1.1{\cdot}10^{-09})$	$1.6{\cdot}10^{-16}(1.3{\cdot}10^{-16})$

Table I. Backward (and forward) errors of the eigenvalues computed by the QR algorithm, the square-reduced method, and the subroutine DHAESU.

where $\hat{\lambda}_i$ denotes the computed approximation to the exact eigenvalue λ_i of H and $\sigma_{\min}$ is the smallest singular value of a matrix. The quantity fwd can be considered as the maximal forward error while bwd represents the maximal backward error of the computed eigenvalues. All computations have been performed in Matlab 6.1 using the mex interfaces mentioned in Section 4.4. The "exact" eigenvalues λ_i have been computed in variable precision arithmetic (64 decimal digits) as provided by the Symbolic Math Toolbox in Matlab. Note, however, that this toolbox failed to deliver "exact" eigenvalues for Examples 4.2 and 4.4, where it returned with an error message.

In general, the perturbation analysis predicts that structure preservation will not lead to a higher accuracy in the real and imaginary parts together, see [Benner et al. 2004]. Though not explained by this analysis, Example 1.1 and Example 2.4 of Table I show that the structure-preserving algorithms may return significantly more accurate eigenvalues. The known possible loss of accuracy of the square-reduced method can be observed in Examples 1.6, 2.2, 2.9, and 4.4. In Examples 1.6 and 2.9, preliminary symplectic balancing cures this problem. It is remarkable that the square-reduced method displays its numerical backward instability only for Example 4.4, where the measured backward error $2.9 \cdot 10^{-13}$ is significantly larger than the machine precision. A simple matrix leading to an even larger backward error can be constructed by setting

$$H = U^T \begin{bmatrix} A & 0 \\ 0 & -A^T \end{bmatrix} U, \quad A = \mathrm{diag}(1, 10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}), \tag{20}$$

Ex 1.1	Ex 1.2	Ex 1.3	Ex 1.4	Ex 1.5	Ex 1.6	Ex 2.1
$1.8{\cdot}10^{-16}$	$9.3{\cdot}10^{-17}$	$3.8{\cdot}10^{-15}$	$1.7{\cdot}10^{-15}$	$2.8{\cdot}10^{-16}$	$2.5{\cdot}10^{-16}$	$1.4{\cdot}10^{-16}$
$3.2{\cdot}10^{-16}$	$3.7{\cdot}10^{-16}$	$3.0{\cdot}10^{-16}$	$4.4{\cdot}10^{-16}$	$4.7{\cdot}10^{-16}$	$5.8{\cdot}10^{-16}$	$2.9{\cdot}10^{-16}$
Ex 2.2	**Ex 2.3**	**Ex 2.4**	**Ex 2.5**	**Ex 2.6**	**Ex 2.7**	**Ex 2.8**
$1.1{\cdot}10^{-16}$	$6.1{\cdot}10^{-17}$	$4.5{\cdot}10^{-02}$	$6.7{\cdot}10^{-17}$	$1.6{\cdot}10^{-04}$	$1.8{\cdot}10^{-17}$	$5.1{\cdot}10^{-16}$
$6.6{\cdot}10^{-16}$	$2.4{\cdot}10^{-16}$	$6.5{\cdot}10^{-16}$	$2.5{\cdot}10^{-16}$	$5.4{\cdot}10^{-16}$	$7.3{\cdot}10^{-16}$	$2.9{\cdot}10^{-16}$
Ex 2.9	**Ex 3.1**	**Ex 3.2**	**Ex 4.1**	**Ex 4.2**	**Ex 4.3**	**Ex 4.4**
$1.1{\cdot}10^{-10}$	$5.0{\cdot}10^{-16}$	$4.2{\cdot}10^{-15}$	$1.5{\cdot}10^{-15}$	$7.8{\cdot}10^{-16}$	$4.8{\cdot}10^{-15}$	$8.9{\cdot}10^{-14}$
$1.0{\cdot}10^{-15}$	$7.2{\cdot}10^{-16}$	$1.1{\cdot}10^{-15}$	$6.1{\cdot}10^{-16}$	$9.8{\cdot}10^{-16}$	$9.1{\cdot}10^{-16}$	–

Table II. Relative residuals of invariant subspaces computed by `DHASUB` with `METH = 'S'` (upper row) and `METH = 'L'` (lower row).

where U is a random orthogonal symplectic matrix obtained from the symplectic QR decomposition of a random matrix, see also [Benner et al. 2000]. The errors obtained for this matrix are displayed in the last row of Table I.

5.2 Accuracy of invariant subspaces

To test the accuracy of the stable invariant subspaces computed by `DHASUB`, we repeated the experiment from the previous section with this subroutine and measured the relative residual

$$\text{res} = \|H\hat{X} - \hat{X}(\hat{X}^T H\hat{X})\|_F / \|H\|_F,$$

where the columns of $\hat{X}$ form the computed orthonormal basis for the stable invariant subspace. The parameter `METH` in `DHASUB` can be used to control the choice of method for computing the stable invariant subspace. If `METH = 'S'`, an orthonormal basis for this subspace is obtained by applying a QR decomposition to the matrix Q in (10). If `METH = 'L'`, this basis is obtained by applying a rank-revealing QR decomposition to the matrix $\tilde{Q}_{11} - \tilde{Q}_{21}$ from (10). The results in Table II suggest that `METH = 'S'` often behaves like a numerically backward stable method, except for Examples 2.4, 2.6, and 2.9, while the computationally more expensive choice `METH = 'L'` seems to be numerically backward stable in general. Note that Example 4.4 represents a highly unbalanced Hamiltonian matrix, for which the QR algorithm fails to converge [Benner and Kressner 2003]. Also, `DHASUB` encounters convergence problems, which could only be avoided when symplectic balancing (`BALANC = 'B'`) was combined with `METH = 'S'`. For all other examples, symplectic balancing had no significant positive effect on the numerical behavior of `DHASUB`.

To demonstrate the accuracy of the subroutines `DSHES` and `DHAORD` for computing isotropic invariant subspaces of skew-Hamiltonian matrices, let us consider the following simple matrix:

$$W = U^T \begin{bmatrix} A & 0 \\ 0 & A^T \end{bmatrix} U, \quad A = \operatorname{diag}\left(1, \frac{1}{2^5}, \frac{1}{3^5}, \ldots, \frac{1}{100^5}\right),$$

where, as above, U is obtained from the symplectic QR decomposition of a random 200×200 matrix. Let $\hat{X} = [\hat{x}_1, \ldots, \hat{x}_{100}]$, where $\hat{x}_k$ is a normalized eigenvector belonging to the eigenvalue $1/k^5$, as computed by the QR algorithm. As W is a symmetric, skew-Hamiltonian matrix, the columns of $\hat{X}$ theoretically form an orthonormal basis spanning an isotropic invariant subspace. While the orthonormality is well preserved in finite-precision arithmetic, the isotropy is severely violated:

$$\|\hat{X}^T X - I\|_F = 2.3 \cdot 10^{-14}, \quad \|\hat{X}^T J X\|_F = 8.1 \cdot 10^{-6}.$$

On the other hand, if the subroutines DSHES and DHAORD are used to compute $\hat{X}$, both properties are well preserved in finite-precision arithmetic:

$$\|\hat{X}^T X - I\|_F = 4.4 \cdot 10^{-14}, \quad \|\hat{X}^T J X\|_F = 8.9 \cdot 10^{-15}.$$

6. CONCLUSIONS

We have presented a comprehensive library for solving Hamiltonian and skew-Hamiltonian eigenvalue problems as well as for computing several related orthogonal symplectic decompositions. The described subroutines form the basis of the HAPACK project, which can be found under

`http://www.tu-chemnitz.de/mathematik/hapack/`.

Future work will be directed towards generalized eigenvalue problems involving skew-Hamiltonian/Hamiltonian matrix pencils.

REFERENCES

Abels, J. and Benner, P. 1999. CAREX - a collection of benchmark examples for continuous-time algebraic Riccati equations (version 2.0). SLICOT working note 1999-14, http://www.win.tue.nl/niconet.

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J. W., Dongarra, J. J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., and Sorensen, D. C. 1999. *LAPACK Users' Guide*, Third ed. SIAM, Philadelphia, PA.

Arnold, III, W. and Laub, A. 1984. Generalized eigenproblem algorithms and software for algebraic Riccati equations. *Proc. IEEE 72*, 1746–1754.

Bai, Z. and Demmel, J. W. 1989. On a block implementation of the Hessenberg multishift QR iterations. *Internat. J. High Speed Comput. 1*, 97–112.

Bai, Z. and Demmel, J. W. 1993. On swapping diagonal blocks in real Schur form. *Linear Algebra Appl. 186*, 73–95.

Benner, P. 2000. Symplectic balancing of Hamiltonian matrices. *SIAM J. Sci. Comput. 22,* 5, 1885–1904.

Benner, P., Byers, R., and Barth, E. 2000. Algorithm 800: Fortran 77 subroutines for computing the eigenvalues of Hamiltonian matrices I: The square-reduced method. *ACM Trans. Math. Software 26*, 49–77.

Benner, P., Byers, R., Mehrmann, V., and Xu, H. 2004. Robust numerical methods for robust control. Preprint 2004-06, Institut für Mathematik, TU Berlin, D-10623 Berlin, FRG, http://www.math.tu-berlin.de/preprints.

Benner, P. and Kressner, D. 2003. Balancing sparse Hamiltonian eigenproblems. To appear in *Linear Algebra Appl.*

Benner, P., Kressner, D., and Mehrmann, V. 2003. Structure preservation: A challenge in computational control. *Future Generation Computer Systems 19,* 7, 1243–1252.

Benner, P., Kressner, D., and Mehrmann, V. 2004. Skew-Hamiltonian and Hamiltonian eigenvalue problems: Theory, algorithms and applications. To appear in Proceedings of ApplMath03, Brijuni (Croatia), June 23-27, 2003, Kluwer.

Benner, P., Mehrmann, V., Sima, V., Van Huffel, S., and Varga, A. 1999. SLICOT—a subroutine library in systems and control theory. In *Applied and computational control, signals, and circuits, Vol. 1.* Birkhäuser Boston, Boston, MA, 499–539.

Benner, P., Mehrmann, V., and Xu, H. 1997. A new method for computing the stable invariant subspace of a real Hamiltonian matrix. *J. Comput. Appl. Math. 86*, 17–43.

Benner, P., Mehrmann, V., and Xu, H. 1998. A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils. *Numerische Mathematik 78,* 3, 329–358.

Benner, P., Mehrmann, V., and Xu, H. 1999. A note on the numerical solution of complex Hamiltonian and skew-Hamiltonian eigenvalue problems. *Electron. Trans. Numer. Anal. 8*, 115–126.

Bischof, C. and Van Loan, C. F. 1987. The WY representation for products of Householder matrices. *SIAM J. Sci. Statist. Comput. 8,* 1, 2–13.

Bojanczyk, A., Golub, G. H., and Van Dooren, P. 1992. The periodic Schur decomposition; algorithm and applications. In *Proc. SPIE Conference*. Vol. 1770. 31–42.

Boyd, S., Balakrishnan, V., and Kabamba, P. 1989. A bisection method for computing the $\mathcal{H}_\infty$ norm of a transfer matrix and related problems. *Math. Control, Signals, Sys. 2*, 207–219.

Bunch, J. R. 1987. The weak and strong stability of algorithms in numerical linear algebra. *Linear Algebra Appl. 88/89*, 49–66.

Bunse-Gerstner, A. 1986. Matrix factorizations for symplectic QR-like methods. *Linear Algebra Appl. 83*, 49–77.

Burke, J. V., Lewis, A. S., and Overton, M. L. 2003. Robust stability and a criss-cross algorithm for pseudospectra. *IMA J. Numer. Anal. 23,* 3, 359–375.

Byers, R. 1983. Hamiltonian and symplectic algorithms for the algebraic Riccati equation. Ph.D. thesis, Cornell University, Dept. Comp. Sci., Ithaca, NY.

Byers, R. 1988. A bisection method for measuring the distance of a stable to unstable matrices. *SIAM J. Sci. Statist. Comput. 9*, 875–881.

Dongarra, J. J., Du Croz, J., Duff, I. S., and Hammarling, S. 1990. A set of level 3 basic linear algebra subprograms. *ACM Trans. Math. Software 16*, 1–17.

Dongarra, J. J., Sorensen, D. C., and Hammarling, S. J. 1989. Block reduction of matrices to condensed forms for eigenvalue computations. *J. Comput. Appl. Math. 27,* 1-2, 215–227. Reprinted in *Parallel algorithms for numerical linear algebra*, 215–227, North-Holland, Amsterdam, 1990.

Dubrulle, A. A. 1991. The multishift QR algorithm–is it worth the trouble? TR 6320-3558, IBM Scientific Center, Palo Alto, CA.

Golub, G. H. and Van Loan, C. F. 1996. *Matrix Computations*, third ed. Johns Hopkins University Press, Baltimore, MD.

Hench, J. J. and Laub, A. J. 1994. Numerical solution of the discrete-time periodic Riccati equation. *IEEE Trans. Automat. Control 39,* 6, 1197–1210.

Hwang, T.-M., Lin, W.-W., and Mehrmann, V. 2003. Numerical solution of quadratic eigenvalue problems with structure-preserving methods. *SIAM J. Sci. Comput. 24,* 4, 1283–1302.

Kressner, D. 2003a. Block algorithms for orthogonal symplectic factorizations. *BIT 43,* 4, 775–790.

Kressner, D. 2003b. The periodic QR algorithm is a disguised QR algorithm. To appear in *Linear Algebra Appl.*

Kressner, D. 2004. Numerical methods and software for general and structured eigenvalue problems. Ph.D. thesis, TU Berlin, Institut für Mathematik, Berlin, Germany.

Lancaster, P. and Rodman, L. 1995. *The Algebraic Riccati Equation*. Oxford University Press, Oxford.

Leimkuhler, B. J. and Van Vleck, E. S. 1997. Orthosymplectic integration of linear Hamiltonian systems. *Numer. Math. 77,* 2, 269–282.

Mehrmann, V. and Watkins, D. S. 2000. Structure-preserving methods for computing eigenpairs of large sparse skew-Hamiltonian/Hamiltonian pencils. *SIAM J. Sci. Comput. 22,* 6, 1905–1925.

Osborne, E. E. 1960. On preconditioning of matrices. *Journal of the ACM 7*, 338–345.

Parlett, B. N. and Reinsch, C. 1969. Balancing a matrix for calculation of eigenvalues and eigenvectors. *Numerische Mathematik 13*, 293–304. Also in [Wilkinson and Reinsch 1971, pp.315–326].

Schreiber, R. and Van Loan, C. F. 1989. A storage-efficient WY representation for products of Householder transformations. *SIAM J. Sci. Statist. Comput. 10,* 1, 53–57.

Stefanovski, J. and Trenčevski, K. 1998. Antisymmetric Riccati matrix equation. In *1st Congress of the Mathematicians and Computer Scientists of Macedonia (Ohrid, 1996)*. Sojuz. Mat. Inform. Maked., Skopje, 83–92.

The MathWorks, Inc. 2002. Matlab *Version 6.5.* The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, Mass, 01760.

The Working Group on Software (WGS) 1996. *SLICOT Implementation and Documentation Standards 2.1.* The Working Group on Software (WGS), Available from `http://www.win.tue.nl/wgs/reports.html`. WGS-report 96-1.

Tisseur, F. and Meerbergen, K. 2001. The quadratic eigenvalue problem. *SIAM Rev. 43,* 2, 235–286.

Van Loan, C. F. 1975. A general matrix eigenvalue algorithm. *SIAM J. Numer. Anal. 12,* 6, 819–834.

Van Loan, C. F. 1984. A symplectic method for approximating all the eigenvalues of a Hamiltonian matrix. *Linear Algebra Appl. 61*, 233–251.

Watkins, D. S. 1995. Forward stability and transmission of shifts in the QR algorithm. *SIAM J. Matrix Anal. Appl. 16,* 2, 469–487.

Wilkinson, J. H. and Reinsch, C. 1971. *Handbook for Automatic Computation. Vol. II Linear Algebra*. Springer-Verlag, New York.

Xu, H. and Lu, L. Z. 1995. Properties of a quadratic matrix equation and the solution of the continuous-time algebraic Riccati equation. *Linear Algebra Appl. 222*, 127–145.

Zhou, K., Doyle, J. C., and Glover, K. 1996. *Robust and Optimal Control.* Prentice-Hall, Upper Saddle River, NJ.

A. LIST OF AVAILABLE SUBROUTINES

A.1 Driver subroutines

`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.

A.2 Computational subroutines

`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.

A.3 Auxiliary subroutines

DCROOT	Computes the square root of a complex number in real arithmetic.
DHAEX2	Swaps adjacent diagonal blocks in a (skew-)Hamiltonian Schur decomposition.
DLABMX	Auxiliary subroutine for DHASUB.
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.