Computation of Stable Invariant Subspaces of Hamiltonian Matrices

R. V. Patel  
Concordia University - Montreal

Z. Lin  
Concordia University - Montreal

Pradeep Misra  
Wright State University - Main Campus, pradeep.misra@wright.edu

Follow this and additional works at: https://corescholar.libraries.wright.edu/ee

Part of the Electrical and Computer Engineering Commons

Repository Citation
https://corescholar.libraries.wright.edu/ee/1

This Article is brought to you for free and open access by the Electrical Engineering at CORE Scholar. It has been accepted for inclusion in Electrical Engineering Faculty Publications by an authorized administrator of CORE Scholar. For more information, please contact corescholar@www.libraries.wright.edu, library-corescholar@wright.edu.
COMPUTATION OF STABLE INVARIANT SUBSPACES OF HAMILTONIAN MATRICES*

R. V. PATEL†, Z. LIN‡, AND P. MISRA†

Abstract. This paper addresses some numerical issues that arise in computing a basis for the stable invariant subspace of a Hamiltonian matrix. Such a basis is required in solving the algebraic Riccati equation using the well-known method due to Laub. Two algorithms based on certain properties of Hamiltonian matrices are proposed as viable alternatives to the conventional approach.

Key words. Hamiltonian matrices, eigenvalues, invariant subspaces, algebraic Riccati equation

AMS subject classifications. 65F15, 65H10, 93D15

1. Introduction. A matrix $Z \in \mathbb{R}^{2n \times 2n}$ is called a Hamiltonian matrix if

$$JZJT = -ZT,$$

where

$$J = \begin{bmatrix} O_n & I_n \\ -I_n & O_n \end{bmatrix}.$$  

In (1.2), $I_n$ denotes the $n \times n$ identity matrix and $O_n$ denotes the $n \times n$ null matrix. In this paper we consider some numerical issues related to the problem of finding a basis for the invariant subspace of a Hamiltonian matrix corresponding to a certain subset of its eigenvalues. Such a problem arises in one of the more commonly used methods [1] for solving the well-known continuous-time algebraic Riccati equation (ARE):

$$\text{CR}(X) \equiv ATX + XA - XGX + F = O,$$

where $A \in \mathbb{R}^{n \times n}$; $G = BB^T \in \mathbb{R}^{n \times n}$ and $F = CTC \in \mathbb{R}^{n \times n}$ are positive semidefinite matrices. If $(A, B)$ is a stabilizable pair and $(A, C)$ is a detectable pair [2], then (1.3) has a unique positive semidefinite solution $X = X^T \in \mathbb{R}^{n \times n}$, which is a stabilizing solution, i.e., $A - GX$ has all its eigenvalues in the open left half-plane. Numerical solutions of equations of the form (1.3) are required in several control system applications, e.g., the linear quadratic optimal control problem, Kalman filtering, and $H^\infty$ robust control. It is not surprising, therefore, that many methods have been proposed for solving the ARE, e.g., see [3], [4] for an extensive bibliography. In this paper, we

---

*Received by the editors September 18, 1989; accepted for publication (in revised form) May 7, 1992. This work was supported by the Natural Sciences and Engineering Research Council of Canada grant OGP0001345 and by Wright Research and Development Center, Wright Patterson Air Force Base grant F33615-88-C-3605.

†Department of Electrical and Computer Engineering, Concordia University, Montréal, Québec, Canada H3G 1M8 (rajni@davinci.concordia.ca). Current address for Z. Lin, International Submarine Engineering Ltd., 1734 Broadway Street, Port Coquitlam, British Columbia, Canada V3C 2M8 (ise@cs.sfu.ca).

‡Department of Electrical Engineering, Wright State University, Dayton, Ohio 45435 (pmisra@valhalla.cs.wright.edu).
shall concentrate on one of the more reliable and elegant methods for solving (1.3). The Hamiltonian matrix corresponding to (1.3) is given by

\[
Z = \begin{bmatrix} A & G \\ F & -AT \end{bmatrix}.
\]

The method proposed by Laub [1] is based on the following results concerning the matrix Z:

(a) If \( \lambda \in \mathbb{C} \) is an eigenvalue of Z, then so is \(-\lambda\).

(b) A symmetric matrix X is the desired stabilizing solution of (1.3) if and only if \( X = -U_21 U_{11}^{-1} \), where the columns of \( [U_{11}^T \quad U_{21}^T]^T \) span the \( n \)-dimensional invariant subspace of Z corresponding to its stable eigenvalues.

The algorithm proposed by Laub for computing X involves the following steps:

**Algorithm I**

**Step 1.** Reduce Z to a real Schur form (RSF), \( \hat{R} \in \mathcal{R}^{2n \times 2n} \). Accumulate the orthogonal transformations in a matrix \( \hat{U} \in \mathcal{R}^{2n \times 2n} \), i.e.,

\[
\hat{R} = \hat{U}^T Z \hat{U}.
\]

*Comment:* This step can be performed by first reducing Z to upper Hessenberg form and then using the QR Algorithm [5]. The reduction to upper Hessenberg form requires approximately \( \frac{5}{3} (2n)^3 \) flops (floating point operations) and the reduction of the resulting upper Hessenberg matrix to an RSF requires approximately \( 4\kappa (2n)^3 \) flops, where \( \kappa \) represents the average number of QR steps required per eigenvalue (\( \approx 1.5 \)).

**Step 2.** Rearrange the eigenvalues of \( \hat{R} \) so that the \( n \) stable eigenvalues are in the top left corner.

*Comment:* This can be achieved by means of orthogonal transformations on \( \hat{R} \) using the subroutines EXCHNG and HQR3 [6] (also note the corrections in [7]) and requires more than \( 4\kappa (2n)^3 \) flops. Let this eigenvalue rearrangement operation be represented by

\[
\hat{R} = \hat{U}^T \hat{R} \hat{U},
\]

where the orthogonal transformations resulting from using EXCHNG and HQR3 are accumulated in \( \hat{U} \), and the submatrix \( \hat{R}_{11} \in \mathcal{R}^{n \times n} \) is in RSF with all its eigenvalues in the open left half-plane.

**Step 3.** Set \( U = \hat{U} \hat{U} \) and let

\[
U = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix}, \quad \text{where } U_{11}, U_{21} \in \mathcal{R}^{n \times n}.
\]

**Step 4.** Solve \( X U_{11} = -U_{21} \) for X.

*Comment:* This step can be performed using the appropriate subroutines from LINPACK and requires approximately \( \frac{3}{4} n^3 \) flops.

The above approach (also called the Schur vector approach) can be regarded as a generalization, and a numerically much more robust implementation, of the eigenvector approach of MacFarlane [8]. In the rest of this paper, we concentrate on Steps 1

\[1\] By a real Schur form, we mean an upper quasi-triangular real matrix with a scalar along the diagonal for each real eigenvalue and a \( 2 \times 2 \) block for each pair of complex-conjugate eigenvalues.
and 2, and more specifically on the problem of numerically computing the columns of \([U_{11}^T \quad U_{21}^T]^T\) that span the \(n\)-dimensional invariant subspace of \(Z\) corresponding to its stable eigenvalues. Our intention is to suggest some ways of improving the efficiency and accuracy of the computations that are required in these steps.

2. Computing the eigenvalues of a Hamiltonian matrix. The method proposed by Laub uses an algorithm for reduction of a general matrix to upper Hessenberg form and the QR Algorithm to find the eigenvalues of Hessenberg matrices. Therefore, it does not take into account the special structure of \(Z\), so that the transformations employed in this algorithm destroy the Hamiltonian structure of \(Z\). On the other hand, if similarity transformations on \(Z\) are carried out using symplectic matrices, then its Hamiltonian structure will be preserved. In other words, the matrix \(Z_0 = V^{-1}ZV\) will be a Hamiltonian matrix if \(V\) is a symplectic matrix. A matrix \(V \in \mathcal{R}^{2n \times 2n}\) is symplectic if \(V^TJ = J\), where \(J\) is defined by (1.2). From the point of view of numerical reduction of \(Z\) to a condensed form, such as a block upper triangular form, it is desirable to perform the required transformations using orthogonal symplectic matrices. The following result shows the existence of one such condensed form.

**Theorem 2.1.** If \(Z\) has no eigenvalues on the imaginary axis, then there exists an orthogonal symplectic matrix

\[
V = \begin{bmatrix}
V_{11} & V_{21}
- V_{21} & V_{11}
\end{bmatrix}
\]

with \(V_{11}, V_{12} \in \mathcal{R}^{n \times n}\), such that

\[
VTZV = \begin{bmatrix}
R & \hat{G}
O_n & - R^T
\end{bmatrix} \equiv \hat{Z},
\]

where \(\hat{G} = \hat{G}^T \in \mathcal{R}^{n \times n}\), and \(R \in \mathcal{R}^{n \times n}\) is in RSF with eigenvalues in the open left half-plane.

**Proof.** See [9].

There are two types of orthogonal symplectic matrices that are particularly useful in performing reductions on Hamiltonian matrices. The first type consists of Householder symplectic matrices defined by

\[
P(k, u) = \begin{bmatrix}
\hat{P} & O_n \\
O_n & \hat{P}
\end{bmatrix}, \quad \hat{P} \in \mathcal{R}^{n \times n},
\]

where

\[
\hat{P} = I_n - \frac{2uu^T}{u^Tu}
\]

and

\[
u^T = [0, \ldots, 0, u_k, \ldots, u_n] \neq 0^T.
\]

The second type consists of Givens symplectic matrices defined by

\[
J(k, c, s) = \begin{bmatrix}
C & S \\
-S & C
\end{bmatrix}, \quad C, S \in \mathcal{R}^{n \times n},
\]
where

\[
C = \text{diag}\{1, \ldots, 1, c, 1, \ldots, 1\},
\]

\[
S = \text{diag}\{0, \ldots, 0, s, 0, \ldots, 0\},
\]

and \(c^2 + s^2 = 1\). Algorithms are given in [12] for computing \(P(k, u)\) and \(J(k, c, s)\) to zero specific entries in a vector.

Theorem 2.1 and its proof in [9] show that it is possible to reduce \(Z\) using the structure-preserving orthogonal symplectic transformations to the block upper triangular form (2.2), but no algorithm for doing so is provided. In fact, so far success in developing an efficient QR-type algorithm for this reduction has been reported only for a special case [10], [17], namely, when \(\text{rank}(G) = 1\) or \(\text{rank}(F) = 1\). In this case, Byers has provided an extension of the implicitly shifted QR algorithm that uses orthogonal symplectic transformations. However, because the method is only applicable for a special case, we shall not consider it further in this paper. It suffices to mention that in the algorithm proposed by Byers, a reordering of the eigenvalues is required to ensure that \(R\) is a stable matrix. This reordering is performed using EXCHNG and HQR3 to bring an unstable eigenvalue of \(R\) to its \((n, n)\)th position followed by a Givens symplectic transformation, \(J(n, c, s)\), to interchange the \((n, n)\)th entry of \(R\) with that of \(-R^T\). Also, it is worth mentioning that reduction of \(Z\) to a block triangular condensed form has been achieved in the general case using nonorthogonal symplectic similarity transformations [11]. However, the use of such transformations may cause numerical instability.

An elegant method that uses orthogonal symplectic matrices to “approximate” the eigenvalues of a Hamiltonian matrix has been proposed by Van Loan [12]. The algorithm given in [12] computes the eigenvalues of \(Z^2\), i.e., of

\[
(2.7) \quad M = \begin{bmatrix}
M_{11} & M_{12} \\
M_{21} & M_{22}
\end{bmatrix} = \begin{bmatrix}
A & G \\
F & -A^T
\end{bmatrix}^2,
\]

where

\[
(2.8a) \quad M_{11} = A^2 + GF = M_{22}^T,
\]

\[
(2.8b) \quad M_{12} = AG - GAT = -M_{12}^T,
\]

and

\[
(2.8c) \quad M_{21} = FA - A^TF = -M_{21}^T.
\]

Note that \(M_{12}\) and \(M_{21}\) are skew-symmetric matrices. Also, it can be easily shown that the structure of \(M\) is preserved under symplectic similarity transformations [12]. Furthermore, if \(Z\) has eigenvalues \(\{\lambda_1, -\lambda_1, \ldots, \lambda_n, -\lambda_n\}\), then the eigenvalues of \(M\) are \(\{\lambda_1^2, \lambda_1^2, \ldots, \lambda_n^2, \lambda_n^2\}\). The eigenvalues of \(Z\) can, therefore, be easily obtained from those of \(M\). However, it should be noted that the algorithm in [12] does not give us an RSF of \(Z\), nor does it enable us to compute the stable invariant subspace of \(Z\). The eigenvalues of \(M\) can be computed using the following steps:
Algorithm II

Step 1. Form \( M = \begin{bmatrix} A & G \\ F & -AT \end{bmatrix} \)^2

Step 2. Compute an orthogonal symplectic matrix \( Q \) such that
\[
Q^T MQ = \begin{bmatrix} H & N \\ O_n & -HT \end{bmatrix}
\]
where \( H \) is in upper Hessenberg form (see Algorithm SR in [12]).

Step 3. Compute the eigenvalues of \( H \) \((\mu_i, i = 1, \ldots, n)\) using the QR Algorithm.

Step 4. Compute \( \lambda_i = \sqrt[4]{\mu_i}, i = 1, \ldots, n \). Set \( \lambda_{n+i} = -\lambda_i, i = 1, \ldots, n \).

The symplectic orthogonal matrix \( Q \) in Step 2 is a product of symplectic Householder and Givens transformation matrices and is structure-preserving. The complete algorithm for the reduction is given in [12] with implementation details and numerical properties. It suffices to mention here that the algorithm requires approximately \( 53n^3/3 \) flops which is about 25–30% of the computation required by the QR Algorithm applied to \( Z \). Furthermore, if floating point arithmetic with base \( b \) and precision \( t \) is used, then it can be shown that the computed eigenvalues of \( Z \) obtained using Algorithm II are the exact eigenvalues of a matrix \( Z + E_s \), where \( E_s \in \mathbb{R}^{2n \times 2n} \) satisfies
\[
\| E_s \|_2 \approx \sqrt{b^{-t}} \| Z \|_2.
\]

As a comparison, if the eigenvalues of \( Z \) are computed using the QR Algorithm (as is done in [1]), then the computed eigenvalues are the exact eigenvalues of a matrix \( Z + E_Q \), where \( E_Q \in \mathbb{R}^{2n \times 2n} \) satisfies
\[
\| E_Q \|_2 \approx b^{-t} \| Z \|_2.
\]
This implies that the error in computing the eigenvalues of \( Z \) using Algorithm II may be up to \( \sqrt{b} \) times as large as that using the QR Algorithm. Also, in general, Algorithm II gives better accuracy for eigenvalues with larger magnitudes than those with smaller magnitudes.

The fact that Algorithm II computes the eigenvalues of \( Z \) less accurately than the QR Algorithm is not a matter of concern in our application. It is sufficient at this point to know approximately the set of \( n \) stable or unstable eigenvalues of \( Z \). More accurate values of these will be obtained and at the same time reordered to get a stabilizing solution of the ARE.

3. Condensed forms for \( Z \) with specified eigenvalue ordering. We now consider the problem of reducing \( Z \) to condensed forms in which the eigenvalues of \( Z \) are separated into sets of stable and unstable eigenvalues, i.e., \( Z \) is reduced to a block triangular form
\[
Z = \begin{bmatrix} Z_{11} & Z_{12} \\ O_n & Z_{22} \end{bmatrix},
\]
where \( Z_{11} \) and \( Z_{22} \in \mathbb{R}^{n \times n} \) and have only stable and unstable eigenvalues, respectively. In this section, we show how two such condensed forms can be computed. Our approach uses a modification of the QR Algorithm. The algorithms described in this section can be regarded as alternatives to using the EXCHNG and HQR3 subroutines [6]. In this context, it is worth mentioning that since no interchanging of eigenvalues is done in our approach, we avoid the numerical difficulties that may be encountered in attempting to exchange nearly equal eigenvalues using EXCHNG and HQR3 [13].
We start by showing how a specified eigenvalue of $Z$ can be made to appear in the bottom right (Algorithm QR-down) or top left (Algorithm QR-up) position. For the sake of brevity, we shall present only Algorithm QR-down in detail. Algorithm QR-up can be stated in an analogous manner. We assume that $Z$ is in unreduced upper Hessenberg form and the eigenvalue to be positioned is given.

**Algorithm QR-down** ($Z, n, \lambda, D$)

Step 1. If the eigenvalue to be shifted ($\lambda$) is complex, go to Step 4.

Step 2. Form a real shift corresponding to $\lambda$: $Z := Z - \lambda I_{2n}$

Step 3. For $k = 1, 2, \ldots, 2n - 1$, determine a Householder matrix $\hat{D}_k \in \mathbb{R}^{2 \times 2}$ such that

\[
\hat{D}_k \begin{bmatrix} z_{kk} \\ z_{k+1,k} \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}
\]

$Z := \hat{D}_k Z D_k^T$, where $D_k = \text{diag}\{I_{k-1}, \hat{D}_k, I_{2n-k-1}\}$

$D := D_k D$

end

$Z := Z + \lambda I_{2n}$

exit

Step 4. Form an implicit double shift corresponding to the complex-conjugate pair $(\lambda, \lambda^*)$:

$p_1 := z_{11}^2 - z_{11}(\lambda + \lambda^*) + \lambda \lambda^* + z_{12} z_{21}$

$q_1 := z_{21}[z_{11} + z_{22} - (\lambda + \lambda^*)]$  

$r_1 := z_{21} z_{32}$

Determine a Householder matrix $\hat{D}_0 \in \mathbb{R}^{3 \times 3}$ such that

\[
\hat{D}_0 \begin{bmatrix} p_1 \\ q_1 \\ r_1 \end{bmatrix} = \begin{bmatrix} \times \\ 0 \\ 0 \end{bmatrix}
\]

$Z := \hat{D}_0 Z D_0^T$, where $D_0 = \text{diag}\{\hat{D}_0, I_{2n-3}\}$

$D := D_0 D$

Step 5. For $k = 1, 2, \ldots, 2n - 3$, determine a Householder matrix $\hat{D}_k \in \mathbb{R}^{3 \times 3}$ such that

\[
\hat{D}_k \begin{bmatrix} z_{k+1,k} \\ z_{k+2,k} \\ z_{k+3,k} \end{bmatrix} = \begin{bmatrix} \times \\ 0 \\ 0 \end{bmatrix}
\]

$Z := \hat{D}_k Z D_k^T$, where $D_k = \text{diag}\{I_k, \hat{D}_k, I_{2n-k-3}\}$

$D := D_k D$

end

Determine a Householder matrix $\hat{D}_{2n-2} \in \mathbb{R}^{2 \times 2}$ such that

\[
\hat{D}_{2n-2} \begin{bmatrix} z_{2n-1,2n-2} \\ z_{2n,2n-2} \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}
\]

$Z := \hat{D}_{2n-2} Z D_{2n-2}^T$, where $D_{2n-2} = \text{diag}\{I_{2n-2}, \hat{D}_{2n-2}\}$

$D := D_{2n-2} D$

exit

**Remark 3.1.** Algorithm QR-up ($Z, n, \lambda, U$) can be defined in a similar manner to the way in which QR-down was defined. In this case, the eigenvalue(s) corresponding to the shift are made to appear in the top left corner of $Z$. 

Downloaded 09/25/12 to 130.108.121.217. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php
Remark 3.2. Algorithms QR-down and QR-up perform one step (single for a real eigenvalue and double for a complex-conjugate pair of eigenvalues) of the QR Algorithm [5]. Explicit single shifts are used for real eigenvalues and implicit double shifts for complex-conjugate pairs of eigenvalues. The implicit shifts introduce nonzero terms in certain locations below the subdiagonal of the upper Hessenberg matrix. Algorithm QR-down uses row operations to “chase” these nonzero terms to the bottom right corner, whereas Algorithm QR-up uses column operations to “chase” the terms to the top left.

Remark 3.3. If a shift is an accurately computed eigenvalue of $Z$, then Algorithm QR-down will transform the matrix $Z$ to the form

$$
\begin{bmatrix}
\hat{Z}_{11} & \hat{Z}_{12} \\
0 & \lambda
\end{bmatrix}
$$

for a real single shift $\lambda$, or to the form

$$
\begin{bmatrix}
\hat{Z}_{11} & \hat{Z}_{12} \\
O & \Phi
\end{bmatrix},
$$

where $\Phi \in \mathbb{R}^{2 \times 2}$ for a double shift. Similarly, Algorithm QR-up will position $\lambda$ and $\Phi$ in the top left corner of a quasi-triangular matrix. Now, if a shift is not equal to an accurately computed eigenvalue of $Z$, then the subdiagonal element(s) next to $\lambda$ (or below $\lambda$ (or $\Phi$) in the case of QR-up) may not become zero after one iteration, in which case two or more iterations of the algorithm may be required. In this case the shifts for the second and subsequent iterations would correspond to the scalar (for a real eigenvalue) or the $2 \times 2$ matrix (for a complex-conjugate pair of eigenvalues) in the bottom right corner (Algorithm QR-down) or top left corner (Algorithm QR-up). The effect of performing these additional iterations would be to reduce the appropriate subdiagonal term(s) to zero and yield more accurate value(s) for the eigenvalue(s).

Remark 3.4. The orthogonal similarity transformations on $Z$ are accumulated in $D \in \mathbb{R}^{2n \times 2n}$ for Algorithm QR-down and in $U \in \mathbb{R}^{2n \times 2n}$ for Algorithm QR-up.

Remark 3.5. It has been assumed in algorithms QR-down and QR-up that the upper Hessenberg matrix $Z$ is unreduced. If this is not the case, then the sequence of transformations $D_k$ (in QR-down) and $U_k$ (in QR-up) cannot be completed. However, this is not a limitation because, if $Z$ is not unreduced, then it can be made unreduced by applying an arbitrary QR shift to introduce coupling between the corresponding blocks [6].

3.1. A real Schur form with eigenvalue rearrangement. In this section we show how the algorithms QR-down and QR-up can be used to obtain the condensed form (3.1), in which $Z_{11} \in \mathbb{R}^{n \times n}$ and $Z_{22} \in \mathbb{R}^{n \times n}$ are in RSF and have only stable and unstable eigenvalues, respectively.

Algorithm III (RSF)

**Input:** A Hamiltonian matrix $Z \in \mathbb{R}^{2n \times 2n}$

**Output:** An orthogonal matrix $\hat{U} \in \mathbb{R}^{2n \times 2n}$ such that

$$
\hat{U}^T Z \hat{U} = 
\begin{bmatrix}
R_1 & \hat{Z}_{12} \\
O & R_2
\end{bmatrix}
$$

where $R_1 \in \mathbb{R}^{n \times n}$ and $R_2 \in \mathbb{R}^{n \times n}$ are in RSF with stable and unstable eigenvalues, respectively.

Step 1. Reduce $Z$ to an upper Hessenberg form $Z_1$: 

STABLE INVARIANT SUBSPACES

\(Z_1 := \hat{U}^T Z \hat{U}_1\)
\(n_1 := n\)
\(\rho := 0\)

Step 2. For \(k = 1, 2, \ldots, n\), compute an eigenvalue \(\lambda_k\) of \(Z_1\) (using one or more iterations of Algorithm QR-down); accumulate the transformations in \(\hat{U}_2\).

If \(\text{Re}(\lambda_k) > 0\),
Call QR-up \((Z_k, n_k, -\lambda_k, U_k)\)
else
Call QR-down \((Z_k, n_k, \lambda_k, U_k^T)\)
Call QR-up \((Z_k, n_k, -\lambda_k, U_k)\)
end
\(\hat{U}_2 := \text{diag}\{I_p, U_k, I_\rho\}\)
If \(\lambda_k\) is real,
\(Z_{k+1} := Z_k(2 : 2n_k - 1, 2 : 2n_k - 1)\)
\(n_{k+1} := n_k - 1\)
\(\rho := \rho + 1\)
else
\(Z_{k+1} := Z_k(3 : 2n_k - 2, 3 : 2n_k - 2)\)
\(n_{k+1} := n_k - 2\)
\(\rho := \rho + 2\)
end
end

Step 3. \(\hat{U} := \hat{U}_1 \hat{U}_2\)

Remark 3.6. Algorithm III computes only \(n\) eigenvalues of \(Z\). After an eigenvalue \(\lambda_k\) has been determined using Algorithm QR-down, if \(\lambda_k\) is an unstable eigenvalue, then a shift \(-\lambda_k\) is applied using Algorithm QR-up to position the stable eigenvalue \(-\lambda_k\) in the top left corner. On the other hand, if \(\lambda_k\) is a stable eigenvalue, then a shift \(-\lambda_k\) using Algorithm QR-down and another shift \(\lambda_k\) using Algorithm QR-up are applied to position the unstable eigenvalue \((-\lambda_k)\) in the bottom right corner and the stable eigenvalue \((\lambda_k)\) in the top left corner, respectively. As an illustration, let us consider the case \(k = 2, n = 4\), with computed real eigenvalues \(\lambda_1 > 0\) and \(\lambda_2 > 0\). Then the structure of the resulting matrix would be

\[
\begin{bmatrix}
-\lambda_1 & x & x & x & x & x & x \\
n_2 & -\lambda_2 & x & x & x & x & x \\
n_0 & 0 & x & x & x & x & x \\
n_0 & 0 & 0 & x & x & x & x \\
n_0 & 0 & 0 & 0 & x & x & x \\
n_0 & 0 & 0 & 0 & 0 & 0 & 0 \\
n_0 & 0 & 0 & 0 & 0 & 0 & \lambda_1
\end{bmatrix}
\]

Thus, as \(k\) increases by 1, the eigenvalue problem deflates by 2. Therefore, since we compute only \(n\) eigenvalues of \(Z\), the amount of computation required is approximately \(\frac{1}{2}(4\kappa)(2n)^3\) flops, where \(\kappa\) represents the average number of QR steps required per eigenvalue and is usually overestimated by a factor of 1.5. Also, once \(\lambda_k\) has been determined, a shift of \(-\lambda_k\) normally yields the eigenvalue \(-\lambda_k\) in the appropriate location in one sweep of the QR Algorithm (\(\kappa = 1\)). Therefore, the remaining \(n\) eigenvalues require approximately \(16n^3\) operations. The complete reduction of the upper Hessenberg matrix \(Z_1\) to an RSF can thus be achieved using approximately \(40n^3\)
flops. For comparison, we note that the reduction of a $2n \times 2n$ upper Hessenberg matrix to RSF using the QR Algorithm requires approximately $4\kappa(2n)^3$ operations. Using $\kappa = 1.5$, this amounts to approximately $48n^3$ operations. If the ordering of the eigenvalues in the RSF requires about 25% additional operations, then we have approximately $60n^3$ operations for computing all $2n$ eigenvalues of an upper Hessenberg matrix and rearranging them in groups of stable and unstable eigenvalues using EXCHNG and HQR3.

Remark 3.7. Further speedup in Algorithm III may be achieved by first using Algorithm II to compute the eigenvalues of $Z$. The approach would then be reduced to performing shifts corresponding to these eigenvalues to position them in the appropriate blocks on the diagonal. However, as we shall see in the next section, we need only apply shifts corresponding to the unstable eigenvalues, thereby further reducing the computational effort required.

3.2. A Hessenberg–Schur form. We now describe the reduction of $Z$ to a block upper triangular form (3.1) in which $Z_{11} \in \mathcal{R}^{n \times n}$ is in upper Hessenberg form and $Z_{22} \in \mathcal{R}^{n \times n}$ is in real Schur form. Furthermore, $Z_{11}$ will have all its eigenvalues in the open left half of the complex plane. The corresponding columns of the accumulated transformation matrix will then immediately give us a basis for the stable invariant subspace of $Z$. The stabilizing solution of the ARE can then be obtained as mentioned in §1.

Algorithm IV (Hessenberg–Schur Form)

Input: A Hamiltonian matrix $Z \in \mathcal{R}^{2n \times 2n}$

Output: An orthogonal matrix $U \in \mathcal{R}^{2n \times 2n}$ such that

$$U^T Z U = \begin{bmatrix} H & Z_{12} \\ O_n & R \end{bmatrix}$$

where $H \in \mathcal{R}^{n \times n}$ is an upper Hessenberg matrix with only stable eigenvalues and $R \in \mathcal{R}^{n \times n}$ is in an RSF with only unstable eigenvalues.

Step 1. Compute the "approximate" unstable eigenvalues, $\tilde{\lambda}_i$, $i = 1, \ldots, n$ of $Z$ using Algorithm II.

Step 2. Reduce $Z$ to upper Hessenberg form: $Z_1 := U_1^T Z U_1$, and

$n_1 := n$, $\rho := 0$, $U_2 := I_{2n}$.

Step 3. For $k = 1, 2, \ldots, n$,

Call QR-down $(Z_k, n_k, \tilde{\lambda}_k, \tilde{U}_k^T)$

$U_2 := U_2 \text{ diag}\{\tilde{U}_k, I_\rho\}$

If $\tilde{\lambda}_k$ is real

$Z_{k+1} := Z_k(1:2n_k - 1, 1:2n_k - 1)$

$n_{k+1} := n_k - 1$

$\rho := \rho + 1$

else

$Z_{k+1} := Z_k(1:2n_k - 2, 1:2n_k - 2)$

$n_{k+1} := n_k - 2$

$\rho := \rho + 2$

end

end

Step 4. $U := U_1 U_2$

Remark 3.8. By partitioning $U$ as

$U = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix}$,
where \( U_{11}, U_{21} \in \mathbb{R}^{n \times n} \), it is easy to see that \([U_{11}^T \quad U_{21}^T]^T\) is a basis for the invariant subspace corresponding to the stable eigenvalues of \( Z \). Therefore, a stabilizing solution of the ARE is \( X = -U_{21}U_{11}^{-1} \).

**Remark 3.9.** It should be noted that only \( n \) shifts are applied to \( Z_1 \) using the QR Algorithm and these shifts correspond to the unstable eigenvalues of \( Z \). If the transformations that are accumulated in \( U_1 \) and \( U_2 \) are performed on \( Z \) (via Algorithm QR-down), then the result will be a real Schur matrix \( R \) with the unstable eigenvalues of \( Z \) in the bottom right corner, leaving an upper Hessenberg matrix \( H \) with the stable eigenvalues of \( Z \) in the top left corner. However, if we are only interested in the solution \( X \) of the ARE, we do not need to store the matrix resulting from the operations on \( Z \), since \( X \) can be computed from the first \( n \) columns of \( U \).

**Remark 3.10.** If the eigenvalues \( \lambda_i, i = 1, \ldots, n \), computed using Algorithm II are accurate, then the process of reducing \( Z \) to the Hessenberg–Schur form becomes finite because the required shifts are known a priori. In this case Algorithm IV will be slightly faster than Algorithm III. On the other hand, if the eigenvalues \( \lambda_i, i = 1, \ldots, n \) are not computed accurately, then it may be necessary to perform two or more iterations of Algorithm QR-down with updated values of \( \lambda_i \) to provide more accurate shifts. This will result in finer tuning of the shifts yielding more accurate values for the unstable eigenvalues, which in turn will improve the accuracy of \( H \) and \( U \).

**Remark 3.11.** As mentioned in §2, computation of the eigenvalues of \( Z \) using Van Loan’s method [12] requires approximately \( 53n^3/3 \) operations. The ordering of \( n \) unstable eigenvalues to get the Hessenberg–Schur structure involves approximately \( 4\kappa(2n)^3/2 \) operations. We can use \( \kappa = 1 \), since the shifts employed are the eigenvalues computed using Van Loan’s method and, in general, one iteration per eigenvalue should suffice to “fine tune” the eigenvalues to the accuracy of the QR Algorithm. This results in an operations count of approximately \( 33.7n^3 \) flops for the reduction of the upper Hessenberg matrix \( Z_1 \) to the Hessenberg–Schur form (3.5).

**Remark 3.12.** It is worth mentioning here that in a recent report [14], an algorithm has been proposed that uses the “approximate” eigenvalues computed using Van Loan’s method to implement block shifts (of \( n \) eigenvalues at a time) on a condensed form of \( Z \) obtained by Paige and Van Loan [9] using orthogonal symplectic transformations. The algorithm in [14], therefore, uses only orthogonal symplectic transformations to obtain the RSF in (2.2); but no analysis or numerical experiments are given to show the efficiency or accuracy of the algorithm. As mentioned in [9], performing orthogonal symplectic updates of the condensed form destroys its nice zero structure. Consequently, carrying out implicit double shifts in this case becomes computationally expensive, so that no advantage is gained even though only orthogonal symplectic transformations are used. The use of implicit \( n \) simultaneous (block) shifts overcomes this problem, especially if the shifts are the “approximate” eigenvalues of \( Z \). However, the buildup of rounding errors in implementing a large number of shifts simultaneously can cause difficulties with convergence of this symplectic block QR-type algorithm, particularly if one or more of the eigenvalues are badly conditioned.

### 4. Numerical examples

In this section, we illustrate the numerical performance and properties of the algorithms proposed in the preceding sections. All computations were performed on a SUN 4/370 computer using the f77 compiler.

**Example 1.** In this example, we use the model for the high-speed vehicle control problem described in [1]. State matrices of orders 9, 19, 29, 39, and 49 were considered and, with appropriate choice of matrices \( G \) and \( F \), Hamiltonian matrices of orders 18, 38, 58, 78, and 98, respectively, were obtained. The eigenvalue problems for these
matrices are quite well conditioned. For each Hamiltonian matrix, the comparisons were performed for the following computations:

(a) Computing the eigenvalues and separating them into groups of stable and unstable eigenvalues.

(b) Computing the positive semidefinite solution of the corresponding ARE from the stable invariant subspace.

The above computations were done for three cases:

1. The QR Algorithm and the EXCHNG subroutine [6] were used to compute the eigenvalues and order them into groups of stable and unstable eigenvalues. The computations were performed using both single and double precision arithmetic.

2. Algorithm III was used in single precision to obtain the RSF in (3.4).

3. Algorithm IV was used in single precision to reduce the Hamiltonian matrix to the Hessenberg–Schur form in (3.5).

Note that case 3 requires the use of Van Loan’s method [12] (Algorithm II) to “approximate” the unstable eigenvalues of the Hamiltonian matrix. In fact, since the Hamiltonian matrices in this example are well conditioned, the eigenvalues computed using Van Loan’s method are almost as accurate as those obtained using the QR Algorithm (case 1). The accuracies of the unstable eigenvalues obtained using single precision arithmetic (as compared with the double precision results obtained in case 1) are shown in Table 4.1 for the three cases and the five Hamiltonian matrices.

<table>
<thead>
<tr>
<th>Size of Z</th>
<th>HQR3-EXCHNG</th>
<th>Algorithm III</th>
<th>Algorithm IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>18 × 18</td>
<td>9.68 × 10⁻⁷</td>
<td>4.37 × 10⁻⁷</td>
<td>2.76 × 10⁻⁷</td>
</tr>
<tr>
<td>38 × 38</td>
<td>1.68 × 10⁻⁶</td>
<td>6.97 × 10⁻⁷</td>
<td>5.96 × 10⁻⁷</td>
</tr>
<tr>
<td>58 × 58</td>
<td>2.29 × 10⁻⁶</td>
<td>8.29 × 10⁻⁷</td>
<td>1.43 × 10⁻⁶</td>
</tr>
<tr>
<td>78 × 78</td>
<td>2.48 × 10⁻⁶</td>
<td>4.53 × 10⁻⁶</td>
<td>3.59 × 10⁻⁶</td>
</tr>
<tr>
<td>98 × 98</td>
<td>3.14 × 10⁻⁶</td>
<td>6.70 × 10⁻⁶</td>
<td>2.35 × 10⁻⁶</td>
</tr>
</tbody>
</table>

In this table and elsewhere, the “relative error” in the eigenvalues is the value |
\lambda_i^s - \lambda_i^d|/|\lambda_i^d|, where \lambda_i^s and \lambda_i^d denote the ith eigenvalue computed using single and double precision arithmetic, respectively. The maximum is taken over all the eigenvalues of Z. It is clear from Table 4.1 that the accuracies obtained using Algorithms III and IV is comparable to or better than that obtained using the conventional approach (with the HQR3 and EXCHNG subroutines).

For the three cases and the five Hamiltonian matrices, we obtained bases for the stable invariant subspaces from the accumulated transformations and computed the positive semidefinite solutions of the corresponding AREs. All the computations were performed in single precision. For comparison, we also obtained the solution of each ARE in double precision for case 1. We used this to compute the relative errors in the single precision solutions of the AREs in the three cases, i.e., ||X_d - X_s||_2/||X_d||_2, where X_d is the double precision solution of the ARE using Algorithm I with EXCHNG and HQR3 (case 1) and X_s is the single precision solution of the ARE computed using Algorithm I (case 1), Algorithm III (case 2), and Algorithm IV (case 3). The relative errors in the solutions are shown in Table 4.2 for the three cases and the five Hamiltonian matrices.

Example 2. In this example, we generated random matrices of orders 9, 19, 29, 39, and 49 for A. Also, the matrices G and F were constructed as G = BB^T and
TABLE 4.2  
**Relative error in X.**  
<table>
<thead>
<tr>
<th>Size of Z</th>
<th>HQR3-EXCHNG</th>
<th>Algorithm III</th>
<th>Algorithm IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>18x18</td>
<td>7.46x10^{-7}</td>
<td>7.29x10^{-7}</td>
<td>6.73x10^{-7}</td>
</tr>
<tr>
<td>38x38</td>
<td>1.61x10^{-6}</td>
<td>1.39x10^{-6}</td>
<td>1.37x10^{-6}</td>
</tr>
<tr>
<td>58x58</td>
<td>2.22x10^{-6}</td>
<td>2.25x10^{-6}</td>
<td>1.63x10^{-6}</td>
</tr>
<tr>
<td>78x78</td>
<td>2.86x10^{-6}</td>
<td>2.31x10^{-6}</td>
<td>2.29x10^{-6}</td>
</tr>
<tr>
<td>98x98</td>
<td>3.72x10^{-6}</td>
<td>2.71x10^{-6}</td>
<td>2.89x10^{-6}</td>
</tr>
</tbody>
</table>

$F = C^TC$, where $B$ and $C$ were random matrices. Therefore, once again we have Hamiltonian matrices of orders 18, 38, 58, 78, and 98, respectively. The computations (a) and (b) for the three cases (1–3) mentioned in Example 1 were performed on these matrices. The corresponding results are shown in Tables 4.3 and 4.4.

TABLE 4.3  
**Maximum relative error in $\lambda(Z)$.**  
<table>
<thead>
<tr>
<th>Size of Z</th>
<th>HQR3-EXCHNG</th>
<th>Algorithm III</th>
<th>Algorithm IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>18x18</td>
<td>2.97x10^{-6}</td>
<td>1.05x10^{-6}</td>
<td>6.78x10^{-7}</td>
</tr>
<tr>
<td>38x38</td>
<td>3.49x10^{-6}</td>
<td>2.08x10^{-6}</td>
<td>1.74x10^{-6}</td>
</tr>
<tr>
<td>58x58</td>
<td>1.31x10^{-5}</td>
<td>1.11x10^{-5}</td>
<td>1.26x10^{-5}</td>
</tr>
<tr>
<td>78x78</td>
<td>4.44x10^{-6}</td>
<td>3.43x10^{-6}</td>
<td>2.05x10^{-6}</td>
</tr>
<tr>
<td>98x98</td>
<td>1.89x10^{-5}</td>
<td>1.90x10^{-5}</td>
<td>7.75x10^{-6}</td>
</tr>
</tbody>
</table>

In this example, we note again that the accuracy obtained using Algorithms III and IV is comparable to or better than that obtained using the conventional approach (with the HQR3 and EXCHNG subroutines).

TABLE 4.4  
**Relative error in X.**  
<table>
<thead>
<tr>
<th>Size of Z</th>
<th>HQR3-EXCHNG</th>
<th>Algorithm III</th>
<th>Algorithm IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>18x18</td>
<td>4.82x10^{-6}</td>
<td>5.54x10^{-6}</td>
<td>3.26x10^{-6}</td>
</tr>
<tr>
<td>38x38</td>
<td>3.23x10^{-6}</td>
<td>3.15x10^{-6}</td>
<td>2.47x10^{-6}</td>
</tr>
<tr>
<td>58x58</td>
<td>7.91x10^{-6}</td>
<td>6.84x10^{-6}</td>
<td>7.87x10^{-6}</td>
</tr>
<tr>
<td>78x78</td>
<td>5.99x10^{-6}</td>
<td>5.55x10^{-6}</td>
<td>5.24x10^{-6}</td>
</tr>
<tr>
<td>98x98</td>
<td>6.79x10^{-6}</td>
<td>6.20x10^{-6}</td>
<td>5.03x10^{-6}</td>
</tr>
</tbody>
</table>

**Example 3.** This is an example of a Hamiltonian matrix of order 24 with some very ill-conditioned eigenvalues. The example is the same as that used by Van Loan [12], and the matrix $A$ and its eigenvalues are given in [15]. The latter were used in obtaining the relative errors in the computed eigenvalues. The computations in Example 1 (a) were carried out for cases 1–3 in double precision. The results for the four most ill-conditioned eigenvalues are shown in Table 4.5, where the quantity $s(A_i)$ is the cosine of the angle between the left and right eigenvectors associated with the eigenvalue $\lambda_i$. The reciprocal of $s(A_i)$ denotes the conditioning of $\lambda_i$ [15].

In this example, because of the ill conditioning, only double precision arithmetic was used. For the ill-conditioned eigenvalues, Van Loan’s method (Algorithm II) was significantly less accurate than the QR Algorithm (case 1). Therefore, two or more QR steps (QR-down) are needed in Algorithm IV to improve the accuracy and to
isolate the unstable eigenvalues in the matrix $R$ in (3.5). Table 4.6 shows the relative errors in the ill-conditioned eigenvalues computed using Van Loan's method and those obtained using Algorithm IV.

### Table 4.5

<table>
<thead>
<tr>
<th>$\lambda_i \approx$</th>
<th>$s(\lambda_i)$</th>
<th>HQR3-EXCHNG</th>
<th>Algorithm III</th>
<th>Algorithm IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1436</td>
<td>$10^{-7}$</td>
<td>5.54 x $10^{-8}$</td>
<td>4.61 x $10^{-8}$</td>
<td>6.44 x $10^{-8}$</td>
</tr>
<tr>
<td>0.0812</td>
<td>$10^{-8}$</td>
<td>4.73 x $10^{-7}$</td>
<td>4.13 x $10^{-7}$</td>
<td>5.08 x $10^{-7}$</td>
</tr>
<tr>
<td>0.0495</td>
<td>$10^{-8}$</td>
<td>1.21 x $10^{-6}$</td>
<td>1.08 x $10^{-6}$</td>
<td>1.26 x $10^{-6}$</td>
</tr>
<tr>
<td>0.0310</td>
<td>$10^{-8}$</td>
<td>9.49 x $10^{-7}$</td>
<td>8.49 x $10^{-7}$</td>
<td>9.72 x $10^{-7}$</td>
</tr>
</tbody>
</table>

### Table 4.6

<table>
<thead>
<tr>
<th>$\lambda_i \approx$</th>
<th>Algorithm II</th>
<th>Algorithm IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1436</td>
<td>4.74 x $10^{-6}$</td>
<td>6.44 x $10^{-8}$</td>
</tr>
<tr>
<td>0.0812</td>
<td>5.66 x $10^{-5}$</td>
<td>5.08 x $10^{-7}$</td>
</tr>
<tr>
<td>0.0495</td>
<td>2.16 x $10^{-4}$</td>
<td>1.26 x $10^{-6}$</td>
</tr>
<tr>
<td>0.0310</td>
<td>2.55 x $10^{-4}$</td>
<td>9.72 x $10^{-7}$</td>
</tr>
</tbody>
</table>

**Example 4.** Here we tested the algorithms for two scenarios: when some of the eigenvalues of the Hamiltonian matrices as well as the corresponding AREs are poorly conditioned, and when there are multiple eigenvalues. The computations (a) and (b) for the three cases (1–3) mentioned in Example 1 were performed for five Hamiltonian matrices. The relative errors in the eigenvalues and in the solutions of the AREs are shown in Tables 4.7 and 4.8, respectively. The three poorly conditioned examples correspond to Hamiltonian matrices, which were generated using the Frank matrix, the data for the boiler model [16], and an example given by Byers [17]. It should be noted that the values of $\min\{s(\lambda_i)\}$ in Tables 4.7 and 4.8 are for the closed-loop eigenvalues, i.e., for the eigenvalues of the Hamiltonian matrices with nonzero $F$ and $G$ matrices, whereas in Table 4.5, the values of $s(\lambda_i)$ are for the open-loop case, i.e., for the Frank matrix. The term $\kappa_R$ in Table 4.8 denotes the "condition number" of the ARE [17] corresponding to a given Hamiltonian matrix, and is given by

$$\kappa_R = \frac{(1 + \|X\|)^2 \|Z\|}{\|X\|SER[(A-GX), -(A-GX)^T]}$$

where $SER[N,-N^T] = \min\{\|PN + NTP\| \mid \|P\| = 1\}$. A large value of $\kappa_R$ implies an ill-conditioned ARE.

From Tables 4.7 and 4.8, we note that the eigenvalues and the AREs corresponding to the Hamiltonian matrices with multiple eigenvalues are well conditioned. As expected, for these two cases the three algorithms give very good and comparable accuracy. In the first case, the Hamiltonian matrix has eigenvalues at $\pm 20$, $\pm 30$, and $\pm 50$ with multiplicity 3; in the second case, the eigenvalues are at $\pm 4$ with multiplicity 6, and at $\pm 10$, $\pm 20$, and $\pm 30$ each with multiplicity 2.

The three other Hamiltonian matrices considered in Tables 4.7 and 4.8 are relatively poorly conditioned. The boiler problem has the worst conditioning of the three cases, both with respect to the eigenvalues of the Hamiltonian matrix as well as the
TABLE 4.7
Maximum relative error in $\lambda(Z)$.

<table>
<thead>
<tr>
<th>Examples</th>
<th>$\min{s(\lambda_i)} \approx HQR3$-EXCHNG</th>
<th>Algorithm III</th>
<th>Algorithm IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boiler problem (18 x 18)</td>
<td>$10^{-8}$</td>
<td>$1.80 \times 10^{-3}$</td>
<td>$1.04 \times 10^{-3}$</td>
</tr>
<tr>
<td>Frank matrix (24 x 24)</td>
<td>$10^{-4}$</td>
<td>$8.62 \times 10^{-4}$</td>
<td>$7.05 \times 10^{-4}$</td>
</tr>
<tr>
<td>Byers’s example (10 x 10)</td>
<td>$10^{-3}$</td>
<td>$1.64 \times 10^{-5}$</td>
<td>$1.52 \times 10^{-5}$</td>
</tr>
<tr>
<td>Multiple case 1 (24 x 24)</td>
<td>$10^{-1}$</td>
<td>$1.24 \times 10^{-6}$</td>
<td>$9.53 \times 10^{-7}$</td>
</tr>
<tr>
<td>Multiple case 2 (24 x 24)</td>
<td>$10^{-1}$</td>
<td>$1.20 \times 10^{-6}$</td>
<td>$1.04 \times 10^{-6}$</td>
</tr>
</tbody>
</table>

TABLE 4.8
Relative error in $X$.

<table>
<thead>
<tr>
<th>Examples</th>
<th>$\kappa_R \approx HQR3$-EXCHNG</th>
<th>Algorithm III</th>
<th>Algorithm IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boiler problem (18 x 18)</td>
<td>$10^{15}$</td>
<td>$1.20 \times 10^{-3}$</td>
<td>$1.20 \times 10^{-3}$</td>
</tr>
<tr>
<td>Frank matrix (24 x 24)</td>
<td>$10^{7}$</td>
<td>$1.10 \times 10^{-3}$</td>
<td>$9.71 \times 10^{-4}$</td>
</tr>
<tr>
<td>Byers’s example (10 x 10)</td>
<td>$10^{9}$</td>
<td>$4.01 \times 10^{-6}$</td>
<td>$4.14 \times 10^{-6}$</td>
</tr>
<tr>
<td>Multiple case 1 (24 x 24)</td>
<td>$20$</td>
<td>$4.99 \times 10^{-8}$</td>
<td>$3.99 \times 10^{-8}$</td>
</tr>
<tr>
<td>Multiple case 2 (24 x 24)</td>
<td>$40$</td>
<td>$5.10 \times 10^{-8}$</td>
<td>$4.46 \times 10^{-8}$</td>
</tr>
</tbody>
</table>

corresponding ARE. This results in slightly lower accuracy for the computed closed-loop eigenvalues and the solution of the ARE than in the other two cases, for which the measures of conditioning have similar orders of magnitude.

From Tables 4.7 and 4.8, we note that both Algorithms III and IV perform as well as or slightly better than the conventional approach (HQR3-EXCHNG) in all cases except one. The exception corresponds to the results obtained using Algorithm IV for the boiler model, which is the most ill conditioned of the examples considered in Tables 4.7 and 4.8. The slightly lower accuracy in this case can be explained as follows: For a Hamiltonian matrix with some ill-conditioned eigenvalues, there is a significant loss of accuracy in computing the corresponding eigenvalues of the square of the Hamiltonian matrix using Van Loan’s method [12]. Consequently, the shifts used in Algorithm IV for very ill-conditioned eigenvalues will have poor accuracy.

5. Concluding remarks. In this paper, some numerical issues in computing a basis for the stable invariant subspace of a Hamiltonian matrix have been discussed. In particular, two alternatives to the use of the EXCHNG subroutine for reordering eigenvalues of a Hamiltonian matrix have been proposed. These were derived using certain properties of Hamiltonian matrices and were shown to require significantly less computation than the conventional approach (using the HQR3 and EXCHNG subroutines). Numerical experiments that have been carried out suggest that the proposed algorithms give accuracy that is often comparable to or better than that obtained using the conventional approach.

Acknowledgments. The authors wish to thank the referees and the review editor for their helpful comments and suggestions.

REFERENCES


