An Efficient Algorithm for the Discrete-Time Algebraic Riccati Equation

L.-Z. Lu, W.-W. Lin, and C. E. M. Pearce

Abstract—In this paper the authors develop a new algorithm to solve the standard discrete-time algebraic Riccati equation by using a skew-Hamiltonian transformation and the square-root method. The algorithm is structure-preserving and efficient because the Hamiltonian structure is fully exploited and only orthogonal transformations are used. The efficiency and stability of the algorithm are analyzed. Numerical examples are included.

Index Terms—Algebraic Riccati equation, Hamiltonian structure, square-root method, structure-preserving algorithms.

I. INTRODUCTION

Algebraic Riccati equations arise in a number of scientific and engineering application areas, such as optimal control, filtering theory, and signal processing. Therefore, their numerical solution is a problem of practical importance and has been a topic of considerable interest.

In this paper we develop an efficient algorithm for solving the standard discrete-time algebraic Riccati equation

\[ A^T X A - X - A^T X B (R + B^T X B)^{-1} B^T X A + K = 0 \]

(1)

where \( A, K \in \mathbb{R}^{n \times n} \), \( B \in \mathbb{R}^{n \times m} \), and \( R = \mathbb{R}^{m \times m} \). The matrices \( R \) and \( K \) are symmetric with \( R \) positive definite and \( K \) nonnegative definite. The coefficient matrices in (1) can be assembled to give a \( 2n \times 2n \) symplectic pencil

\[ N - \lambda L = \begin{bmatrix} A & 0 \\ K & I \end{bmatrix} - \lambda \begin{bmatrix} I & -G \\ 0 & A^T \end{bmatrix} \]

(2)

Manuscript received April 29, 1998. Recommended by Associate Editor, C.Kenney.
L.-Z. Lu is with the Department of Mathematics, Xiamen University, Xiamen, China.
W.-W. Lin is with the Department of Applied Mathematics, Tsinghua University, Taiwan, China.
C. E. M. Pearce is with the Department of Applied Mathematics, University of Adelaide, Adelaide, Australia.

II. BASIS OF THE METHOD

For completeness we first recall some essential definitions.

Definition 2.1:

1) A \( 2n \times 2n \) real matrix \( H \) is called Hamiltonian if \( JH \) is symmetric, where \( J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix} \) with \( I_n \), the \( n \times n \) identity matrix. A \( 2n \times 2n \) real matrix \( \tilde{N} \) is called skew-Hamiltonian if \( JN \) is skew-symmetric.

2) A \( 2n \times 2n \) regular pencil \( E - \lambda F \) is called symplectic if \( EF^T = FJF^T \). A \( 2n \times 2n \) matrix \( S \) is called symplectic if \( JS^T = J \).

Since

\[ L J L^T = N J N^T = \begin{bmatrix} 0 & A \\ -A^T & 0 \end{bmatrix} \]

(3)

it follows that the pencil \( N - \lambda L \) given in (2) is symplectic.

Definition 2.2: Suppose \( \bar{H} \) is a nonsingular matrix. A matrix \( Y \) is called the positive square root of \( H \) if \( Y^2 = H \) and all eigenvalues of \( Y \) have positive real parts.

The following well-known result (see [6]) bears on this definition.
Theorem 2.1: Let $H$ be a nonsingular matrix with no negative real eigenvalues. Then $H$ has a unique positive square root [denoted by $\sqrt{H}$].

Combining Theorem 2.1 with [12, Ths. 2.5 and 2.6] gives immediately the following result.

Theorem 2.2: Let the symplectic pencil $N - \lambda L$ defined by (2) have no eigenvalue $-1$. Then

$$H = (N + L)^{-1} (N - L)$$

(4)

is Hamiltonian. Furthermore, the columns of $H - \sqrt{H^2}$ span the invariant subspace of $N - \lambda L$ corresponding to the eigenvalues inside the unit circle.

From Theorem 2.2 the key point for obtaining the unique positive semidefinite solution is the computation of $\sqrt{H^2}$. This is quite difficult from a numerical point of view, due to the presence of $(N + L)^{-1}$. Theorem 2.3 below provides the foundation for the efficient and stable computation of $\sqrt{H^2}$ in our algorithm. For convenience we put $\Gamma = N + L$

$$M = \Gamma^{-T} (N - L)^T$$

and

$$E = \Gamma J \Gamma^T, \quad F = (N - L) J^T (N - L)^T.$$  

(6)

Theorem 2.3: Let $H$ be defined by (4). Then

$$H^2 = \Gamma^{-1} (M^2)^T \Gamma \quad \text{and} \quad M^2 = E^{-1} F.$$  

(7)

Proof: By (3) we have $(N - L) J^T \Gamma^T = \Gamma J (N - L)^T$, hence

$$E^{-1} F = \Gamma^{-T} J^{-1} \Gamma^{-1} (N - L) J^T (N - L)^T = \Gamma^{-T} J^{-1} \Gamma^{-1} \Gamma J (N - L)^T \Gamma^{-T} (N - L)^T = M^2.$$  

Also

$$\Gamma^{-1} (M^2)^T \Gamma \Gamma^{-1} (N - L) \Gamma^{-1} (N - L) \Gamma^{-1} \Gamma = \Gamma^{-1} (N - L) \Gamma^{-1} (N - L) \Gamma^{-1} \Gamma = H^2.$$  

In the next section we use this theorem and the special forms of $E$, $F$ to develop an efficient algorithm to compute $H - \sqrt{H^2}$.

### III. The Basic Algorithm

Since $\sqrt{S^{-1} A S} = S^{-1} \sqrt{A} S$, we have from Theorem 2.3 that

$$H - \sqrt{H^2} = \Gamma^{-1} (N - L - \sqrt{M^2})^T \Gamma \quad \text{and} \quad M^2 = E^{-1} F.$$  

(8)

First we direct our attention to the computation of $\sqrt{E^{-1} F}$. From (6), $J E$ and $J F$ are both skew-Hamiltonian. We can apply Patel’s algorithm [15] to reduce them simultaneously to block-triangular forms. That is, we can find orthogonal matrices $Q$, $Z$ such that

$$Q J E Z = \begin{bmatrix} \hat{E}_{11} & \hat{E}_{12} \\ 0 & \hat{E}_{11}^T \end{bmatrix}$$  

and

$$Q J F Z = \begin{bmatrix} \hat{F}_{11} & \hat{F}_{12} \\ 0 & \hat{F}_{11}^T \end{bmatrix}.$$  

(9)

where $\hat{E}_{11}$ is $n \times n$ upper Hessenberg, $\hat{F}_{11} n \times n$ upper triangular, and $\hat{E}_{12}, \hat{F}_{12} n \times n$ skew-symmetric.

Next we use the $QZ$ algorithm to produce $n \times n$ orthogonal matrices $Q_1$, $Z_1$, a quasi upper-triangular matrix $E_{11}$ (where there may be $2 \times 2$ blocks on the main diagonal) and an upper-triangular matrix $F_{11}$ such that

$$Q_1 \hat{E}_{11} Z_1 = E_{11} \quad \text{and} \quad Q_1 \hat{F}_{11} Z_1 = F_{11}.$$  

(10)

Put $\hat{Q} = \text{diag}(Q_1, Z_1^T)$, $\hat{Z} = \text{diag}(Z_1, Q_1^T)$, $E_{12} = Q_1 \hat{E}_{12} Q_1^T$, and $F_{12} = Q_1 \hat{F}_{12} Q_1^T$. It is easy to verify from (9) and (10) that

$$\hat{Q} Q J E Z \hat{Z} = \begin{bmatrix} E_{11} & E_{12} \\ 0 & E_{11}^T \end{bmatrix}$$  

and

$$\hat{Q} Q J F Z \hat{Z} = \begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{11}^T \end{bmatrix}.$$  

(11)

Accordingly, we have

$$\hat{Z}^T Z^T (E^{-1} F) Z \hat{Z} = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \equiv T$$  

(12)

where

$$T_{11} = E_{11}^T F_{11} + T_{22} = E_{11}^T T_{11} E_{11}^T + T_{22}$$  

(13)

and

$$T_{12} = E_{11}^T (F_{12} - E_{12} E_{11}^T F_{11}^T) = E_{11}^T (F_{12} - E_{12} T_{22}).$$  

(14)

Finally, from (8) and (12), we obtain that

$$H - \sqrt{H^2} = \Gamma^{-1} (N - L - [Z \hat{Z} \text{sqrt}(\hat{Z}^T Z^T E^{-1} F Z \hat{Z}) \hat{Z}^T Z^T]^T \Gamma)$$  

(15)

We may now address the computation of the positive square root of (15). From (13), we have

$$\text{sqrt}(T_{22}) = E_{11}^T \text{sqrt}(T_{11})^T E_{11}^T.$$  

(16)

Therefore, if

$$U \equiv \text{sqrt}(T) = \begin{bmatrix} \text{sqrt}(T_{11}) & W \\ 0 & \text{sqrt}(T_{22}) \end{bmatrix}$$  

(17)

to find $U$ we need only:

1) compute $\text{sqrt}(T_{11})$ (which can be done by an algorithm such as in [9] or [1]);
2) solve the equation

$$\text{sqrt}(T_{11}) W + W \text{sqrt}(T_{22}) = T_{12}$$  

(18)
(which can be done by an algorithm such as in [8, pp. 242–244]).

By the above discussion, a basic algorithm can be set up as follows.

Algorithm 3.1: This algorithm computes the first $n$ columns of $H = \text{sqrt}(B^2)$, using $H^2$ as given by (6) and (7).

Step 1: Compute $J E$ and $J F$ from (6).

Step 2: Reduce $J E$, $J F$ to block upper-triangular form (9) using the algorithm proposed by Patel [15].

Step 3: Use the $Q Z$ algorithm [8, pp. 256–262] to produce a quasi-upper-triangular $E_{11}$ and an upper-triangular $F_{11}$ as in (10).

Step 4: Compute $T_{11}$, $T_{22}$, and $T_{12}$ from (13) and (14), respectively.

Step 5: Compute $\text{sqrt}(T_{11})$ and solve (18) for $W$ and compute $U$.

Step 6: Compute the $2n \times n$ matrix

$$
\begin{bmatrix}
W_1 \\
W_2
\end{bmatrix} = \Gamma^{-1} \begin{bmatrix} N - L - Z \bar{Z} U^T \bar{Z}^T \Gamma \\ I_n \end{bmatrix}.
$$

Step 7: Solve $X W_1 = W_2$ for $X$.

IV. AN ANALYSIS OF THE EFFICIENCY AND STABILITY OF THE ALGORITHM

We now consider the roundoff errors of Algorithm 3.1. Let $\bar{E}^{-1}$ be the inverse of $E$ as computed from (6) by a digital computer with a machine precision of $\varepsilon \approx 2^{-p}$. The computation is assumed to be stable (see Byers [4]), that is, $\bar{E}^{-1}$ satisfies

$$
\bar{E}^{-1} = \left( E + \delta E^{(1)} \right)^{-1} + \delta E^{(2)} \equiv \bar{E}^{-1} + \delta E^{(2)}
$$

(19)

where $||\delta E^{(1)}|| \leq \delta ||E||$ and $||\delta E^{(2)}|| \leq \epsilon ||\bar{E}^{-1}||$. The constant $\Gamma$ depends on the details of the arithmetic and the inversion algorithm and on the order of the matrix $E$. Assume that $||E^{-1}|| ||\delta E^{(1)}|| < \frac{1}{2}$.

Under this mild assumption

$$
||\delta E^{(2)}|| \leq \frac{2 \epsilon \Gamma}{1 - \delta \epsilon} ||\bar{E}^{-1}|| \leq 4 \epsilon ||\bar{E}^{-1}||
$$

where $||\bar{E}^{-1}|| \leq 2 ||E^{-1}||$. For the computation of $M^2 = E^{-1} F$ in (7) we get

$$
\bar{E}^{-1} F = \left[ \bar{E}^{-1} + \delta E^{(2)} \right] F \equiv \bar{E}^{-1} F + \delta E^{(2)} F
$$

(20)

with $||\delta E^{(2)} F|| \leq 4 \epsilon ||\bar{E}^{-1}|| ||F||$.

Let $\hat{U}$ be the computed square root of the estimate $\hat{T}$ of $T$. From (12) and (17) we have

$$
\bar{Z}^T \bar{Z}^T (\bar{E}^{-1} F) \bar{Z} \hat{T} = \hat{T} \quad \text{and} \quad \hat{U} = \text{sqrt}(\hat{T}) = U + \delta U.
$$

(21)

By Björck and Hammarling [1], the roundoff error of the computed square root of $U$ of $\hat{T}$ is given by

$$
||\delta U|| \leq \left( 2 + \epsilon \right) \alpha ||\hat{T}|| \leq 4 \epsilon \alpha ||E^{-1}|| ||F|| + O(\epsilon^2)
$$

(22)

where $\alpha = ||F||^2 / ||\bar{E}^{-1} F||$. Now from (15) we compute

$$
V \equiv \text{sqrt}(H^2) = \Gamma^{-1} \text{sqrt}(E^{-1} F)^T \Gamma.
$$

Let $\bar{E}^{-1}$ be the computed inverse of $\Gamma^{-1}$, and have the same error estimate as $E^{-1}$ in (19). Assume that $||\Gamma^{-1}|| ||\delta \Gamma^{(0)}|| < \frac{1}{2}$. Then

$$
||\delta \Gamma^{(2)}|| \leq 4 \epsilon \Gamma ||\Gamma^{-1}|| \quad \text{and} \quad ||\Gamma^{-1}|| \leq 2 ||\Gamma^{-1}||
$$

(23)

From (19) and (21), we have

$$
\hat{V} = \bar{E}^{-1} (\hat{U})^T \Gamma = (\Gamma^{-1} + \delta \Gamma^{(2)}) (U + \delta U)^T \Gamma = (I + \Gamma^{-1} \delta \Gamma^{(1)})^{-1} \Gamma^{-1} U^T \Gamma + \delta \Gamma^{(2)} U^T \Gamma + \Gamma^{-1} (\delta U)^T \Gamma + \delta \Gamma^{(2)} (\delta U)^T \Gamma.
$$

(24)

where

$$
\delta V = \left( -\Gamma^{-1} \delta \Gamma^{(1)} + \left( \Gamma^{-1} \delta \Gamma^{(1)} \right)^2 + \cdots \right) \Gamma^{-1} U^T \Gamma + \delta \Gamma^{(2)} U^T \Gamma + \Gamma^{-1} (\delta U)^T \Gamma + \delta \Gamma^{(2)} (\delta U)^T \Gamma.
$$

(25)

Because $\alpha = ||U||^2 / ||\bar{E}^{-1} F||$, we have

$$
||U|| = \alpha^{1/2} ||\bar{E}^{-1} F||^{1/2}
$$

From (20), we derive

$$
||U|| \leq \alpha^{1/2} \left( 2 + 4 \epsilon \Gamma \right)^{1/2} ||E^{-1}|| ||F||^{1/2} ||E^{-1} F||^{1/2}
$$

(26)

It follows from (6) that $||E^{-1}||^{1/2} \leq ||\Gamma^{-1}||$ and $||F||^{1/2} \leq ||N - L||$, and therefore we get from (22) and (26) that

$$
||U|| \leq \epsilon \alpha ||\Gamma^{-1}|| ||N - L||
$$

and

$$
||\delta U|| \leq 4 \alpha ||\Gamma^{-1}|| ||N - L||^2 + O(\epsilon^2).
$$

(27)

Thus the perturbation bound of the square root of $E^{-1} F$ computed by the Schur method [1] is $O(||\Gamma^{-1}||^2 \epsilon)$. From (19), (23), and (27), the upper bound of $\delta V$ in (25) becomes

$$
||\delta V|| \leq c_1 \epsilon ||\Gamma^{-1}||^2 ||\Gamma||^2 ||N - L||
$$

$$
+ c_2 \epsilon ||\Gamma^{-1}||^2 ||\Gamma|| ||N - L||^2 + O(\epsilon^2)
$$

(28)

where $c_1$ and $c_2$ are constants depending, respectively, on $\Gamma$ and $\alpha$.

Generally speaking, the perturbation bound of the square root of $H^2$ is roughly $O(||\Gamma^{-1}||^2 \epsilon)$.

V. COMPUTATION COSTS AND NUMERICAL EXPERIMENTS

We now have the $Q Z$ algorithm (see [14] and [16]), the iterative algorithm (see [12]), and our new Algorithm 3.1 for solving the algebraic Riccati equation (1). The cost of the new algorithm compares
with those of the QZ algorithm and the iterative algorithm as follows:

<table>
<thead>
<tr>
<th>Method</th>
<th>Total amount of work</th>
</tr>
</thead>
<tbody>
<tr>
<td>QZ Algorithm</td>
<td>202n³ flops</td>
</tr>
<tr>
<td>Iterative Algorithm</td>
<td>72n³ flops</td>
</tr>
<tr>
<td>Algorithm 3.1</td>
<td>54n³ flops</td>
</tr>
</tbody>
</table>

The flop count, or number of floating-point additions and multiplications, for the QZ algorithm is derived from [14] or [16], for the iterative algorithm from [12] and for Algorithm 3.1 by summing the counts in the seven steps: 2) from [15]; 3) from the QZ algorithm [8, pp. 256–262]; 5) from [9] or [1] for sqrt(T_i) (i = 1, 2) and from [8, pp. 242–244] for W_i; and 1), 4), 6), and 7) by direct computation.

If the condition number of Γ is large, then from (27), the sensitivity of Γ−1 is roughly proportional to ||Γ−1||³, provided α is not far from unity. Thus if ||Γ−1|| ≈ O(ε−1/3), then Step 4 of Algorithm 3.1 produces a larger perturbation bound and provides a less accurate solution than does the QZ algorithm. On the other hand, if the condition number of N − L is O(1), we can remedy this feature of Algorithm 3.1 by replacing H by H−1. The perturbation bound of the square root of H−2 satisfies

\[
\left| \delta V^{-(1)} \right| \leq \hat{c}_1 \epsilon \left( \|N - L\| \right)^3 \left( \|L\| \right)^2 \|\Gamma\|
+ \hat{c}_2 \epsilon \left( \|N - L\| \right)^3 \left( \|L\| \right)^2 \|\Gamma\| \| \epsilon \right|^2 + O(\epsilon^2)
\]

where \( \hat{c}_1 \) and \( \hat{c}_2 \) are constants corresponding to \( \alpha \) and \( \Gamma \). Thus we can compute a positive semidefinite solution for the discrete algebraic Riccati equation as well. We exhibit this in Example 5.3. If both \( N - L \) and \( \Gamma \) are ill-conditioned we can switch from our algorithm to the QZ algorithm. Some examples are given below illustrating the application of Algorithm 3.1. The first two are taken from [14] with \( \Gamma \) well-conditioned and the third is constructed with \( \Gamma \) ill-conditioned.

Example 5.1: Let

\[
A = \begin{bmatrix} 0, 5, e_1, e_2, \cdots, e_{n-1} \end{bmatrix}
\]

\[
K_1 = e_1
\]

\[
K = K_1, K_1^T = e_1 e_1^T
\]

\[
B = e_4
\]

\[
R = 0.25
\]

\[
G = B R^{-1} B^T = 4 e_4 e_4^T
\]

where \( e_i \) denotes column \( i \) of the \( n \times n \) identity. The four eigenvectors corresponding to the stable eigenvalues of (2) (those inside the unit circle) are given exactly by

\[
\begin{bmatrix}
X_1 \\
X_2
\end{bmatrix} = \begin{bmatrix}
2 & -2 & 1 \\
-1 & -1 & 1 \\
0 & 1 & 1
\end{bmatrix} \begin{bmatrix}
19/4 \\
103/2 \\
536/2
\end{bmatrix}
\]

Let \( X^* \) be the exact positive semidefinite solution \( X_2 X_1^{-1} \). A procedure DARE(F, B, K, H) was written in MATLAB based on the QZ + EXCHQZ and ORDER algorithms [16] to compute the positive semidefinite solution \( X^{QZ+} \). Let \( X^1 \) be the positive semidefinite solution computed by Algorithm 3.1 and \( X^B \) that of the iterative [14, Algorithm 4.1b]. Let \( \text{Err}(X) \) denote the residual matrix when substituting the approximate solution \( X \) into the Riccati equation (1). The numerical results are shown in Table I.

Example 5.2: Let

\[
A = [0, \delta e_1, e_1, e_2, \cdots, e_{n-1}]
\]

\[
K = I_n
\]

\[
B = e_n^T
\]

\[
R = 1
\]

\[
G := e_n e_n^T
\]

The exact solution of (1) is \( X^* = \text{diag}(1, 2, \cdots, n) \). The numerical results obtained for \( n = 15 \) are shown in Table II.

Tables I and II show that Algorithm 3.1 loses about three orders of magnitude compared with the QZ algorithm on well-conditioned matrices. The reason is that because \( \|\Gamma^{-1}\| \approx 10 \) for both examples, the perturbation bound of the square root of \( H^T \) as in (28) is \( O(\|\Gamma^{-1}\|^{3/2} \epsilon) \approx O(10^{-13}) \).
Example 5.3: Let
\[
\bar{A} = \begin{bmatrix}
0.4 & 0 & 0 & 0 \\
1 & 0.6 & 0 & 0 \\
0 & 1 & 0.8 & 0 \\
0 & 0 & 0 & -0.999982
\end{bmatrix}
\]
\[
\bar{K} = \begin{bmatrix}
2 & -1 & 0 & 0 \\
-1 & 2 & -1 & 0 \\
0 & -1 & 2 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\]
\[
\bar{B} = I_t
\]
\[
U = \begin{bmatrix}
1 & -1 & -1 & -1 \\
0 & 1 & -1 & -1 \\
0 & 0 & 1 & -1 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]
\[
U^{-1} = \begin{bmatrix}
1 & 1 & 2 & 4 \\
0 & 1 & 1 & 2 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]
We define
\[
A := U \bar{A} U^{-1} = \begin{bmatrix}
-0.6 & -2.2 & -3.6 & -5.400018 \\
1 & 0.6 & 0.8 & 3.399982 \\
0 & 1 & 1.8 & 3.799982 \\
0 & 0 & 0 & -0.999982
\end{bmatrix}
\]
\[
K := U^{-T} \bar{K} U^{-1} = \begin{bmatrix}
2 & 1 & 3 & 6 \\
1 & 2 & 2 & 5 \\
3 & 2 & 6 & 11 \\
6 & 5 & 11 & 22
\end{bmatrix}
= K_1 K_1^T \text{ (full-rank factorization)}
\]
\[
B := U \bar{B}
\]
\[
G := B R^{-1} B^T = \begin{bmatrix}
4 & 1 & 0 & -1 \\
1 & 3 & 0 & -1 \\
0 & 0 & 2 & -1 \\
-1 & -1 & -1 & 1
\end{bmatrix}
\]

The pairs \((A, B)\) and \((K_1^T, A)\) can be verified to be, respectively, stabilizable and detectable. Since the condition number of \(\Gamma\) is large \((\approx 4.01 \times 10^{11})\), the example is constructed so that the inverse of \(\Gamma\) is near \(1/\sqrt{\gamma}\). We know that the \(QZ\) algorithm and the improved iterative method \([14, \text{Algorithm } 4.1b]\) are applicable. But Algorithm 3.1 fails for this case; the error of the square root thus obtained is too large. If we replace \(H\) by \(H^{-1}\) in Algorithm 3.1 (to give Algorithm 3.2, say), then we can obtain a more accurate positive semidefinite solution to (1). Let \(X^2\) be the positive semidefinite solution computed by Algorithm 3.2. We show the numerical results in Table III.

Many random examples have been tested. The computed solutions with our algorithm have at least 12 digits accuracy.

VI. Conclusions

We have proposed a new algorithm to solve the discrete-time algebraic Riccati equation. The method appears attractive and efficient. Since \(JF\) and \(JF\) in (6) are skew-Hamiltonian, we use Paté’s structure-preserving algorithm to reduce them to block upper-triangular form. Thus we save on arithmetic operations for computing \(E^{-1} F\) and the square root of \(H^2\). The cost of computations is about 25% that of the \(QZ\) algorithm and is less than that of the iterative algorithm and may accordingly be termed “efficient.” Algorithm 3.1 is numerically unstable with the calculation of the inverse of an ill-conditioned matrix \(\Gamma\). If the condition number of \(N - L\) is \(O(1)\), then we replace \(H\) by \(H^{-1}\) in Algorithm 3.1 to compute the positive semidefinite solution for (1). If both \(\Gamma\) and \(N - L\) are ill-conditioned, it is appropriate to switch from our algorithm to the \(QZ\) algorithm.

References