SIMAX, VOL. 19, NO. 1, PP. 107–139, JAN. 1998

A FAST STABLE SOLVER FOR NONSYMMETRIC TOEPLITZ AND QUASI-TOEPLITZ SYSTEMS OF LINEAR EQUATIONS

S. CHANDRASEKARAN* AND ALI H. SAYED†

Abstract. We derive a stable and fast solver for nonsymmetric linear systems of equations with shift structured coefficient matrices (e.g., Toeplitz, quasi-Toeplitz, product of two Toeplitz matrices). The algorithm is based on a modified fast QR factorization of the coefficient matrix and relies on a stabilized version of the generalized Schur algorithm for matrices with displacement structure. All computations can be done in $O(n^2)$ operations, where n is the matrix dimension, and the algorithm is backward stable.

Key words. Displacement structure, generalized Schur algorithm, QR factorization, hyperbolic rotations, generator matrices, Schur complements, error analysis.

AMS subject classifications. 65F05, 65G05, 65F30, 15A23.

1. Introduction.

Linear systems of equations can be solved by resorting to the LDU factorization (Gaussian elimination) of the coefficient matrix. But for indefinite or nonsymmetric matrices, the LDU factorization is numerically unstable if done without pivoting. Moreover, since pivoting can destroy the structure of a matrix, it is not always possible to incorporate it into a fast algorithm for structured matrices without potential loss of computational efficiency.

Sometimes though, one can transform a given structured matrix to another structured form so that the new structure is insensitive to *partial* pivoting operations [9, 12]. While this technique can be satisfactory for certain situations, it may still pose numerical problems because partial pivoting by itself is not sufficient to guarantee numerical stability even for slow algorithms. It also seems difficult to implement *complete* pivoting in a fast algorithm without accruing a considerable loss of efficiency. Recently, Gu [11] has proposed a fast algorithm that incorporates an *approximate* complete pivoting strategy.

Another way to solve a structured linear system of equations is to compute the QR factorization of the coefficient matrix rapidly. Several fast methods have been proposed earlier in the literature [1, 6, 7, 8, 19] but none of them are numerically stable.

In this paper we resolve this open issue and derive an algorithm that is provably both fast *and* backward stable for solving linear systems of equations involving nonsymmetric structured coefficient matrices (e.g., Toeplitz, quasi-Toeplitz, Toeplitz-like). The algorithm is based on a modified fast QR factorization of the coefficient matrix T in $Tx = b$. It computes a factorization for T of the form

$$T = \Delta(\Delta^{-1}Q)R,$$

where Δ is lower-triangular, $(\Delta^{-1}Q)$ is orthogonal, and R is upper triangular. The factorization is then used to solve for x efficiently by using

$$x = R^{-1}(Q^T\Delta^{-T})\Delta^{-1}b. \tag{1.1}$$

* S. Chandrasekaran is with Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106. Fax (805) 893–3262. E-mail: shiv@ece.ucsb.edu

†A. H. Sayed is with Department of Electrical Engineering, University of California, Los Angeles, CA 90095. Fax (310) 206–8495. E-mail: sayed@ee.ucla.edu. The work of Dr. Sayed was supported in part by the National Science Foundation under Award No. MIP-9796147.

All computations can be done in $O(n^2)$ operations, where n is the matrix dimension, and the algorithm is backward stable in the sense that the computed solution $\hat{x}$ is shown to satisfy an equation of the form

$$(T + H)\hat{x} = b\,,$$

where the norm of the error matrix satisfies

$$\|H\| \;\leq\; c_1\epsilon\,\|T\| \;+\; O(\epsilon^2),$$

where ϵ denotes machine precision and c_1 is a low order polynomial in n.

The fast and stable algorithm to be derived in this paper is based on ideas of displacement structure theory [15]. The concept of displacement structure was introduced by Kailath et al. almost two decades ago [14] and has since proven to be a useful tool in matrix analysis. Its strength lies in the fact that it allows, in a systematic way, to describe and exploit varied forms of matrix structure. In this framework, matrix structures are described in terms of *displacement equations* and triangular factorizations are efficiently carried out by a *generalized Schur algorithm* [15].

However, the numerical behaviour of the generalized Schur algorithm has been an issue of concern until very recently, which is mainly due to the fact that the algorithm relies heavily on hyperbolic transformations. In recent work, Bojanczyk et al. [2] have shown that for a subclass of positive-definite shift structured matrices (known as quasi-Toeplitz), the Cholesky factorization provided by the generalized Schur algorithm is asymptotically stable despite the hyperbolic rotations.

The class of quasi-Toeplitz matrices refers to a special kind of structured matrices whose displacement rank (to be defined later) is equal to 2. Stewart and van Dooren [18] further considered the case of positive-definite shift structured matrices with displacement ranks larger than 2. They argued that the generalized Schur algorithm will still provide a stable Cholesky factorization provided the required rotations are now implemented in a special way (a combination of unitary rotations followed by a single hyperbolic rotation in mixed form).

Motivated by the work of Bojanczyk et al. [2], we have also pursued in [4] a detailed analysis of the numerical stability of the generalized Schur algorithm for a general class of positive-definite structured matrices. In particular, we have shown that along with proper implementations of the hyperbolic transformations, if further modifications are introduced while computing intermediate quantities, the algorithm will guarantee a Cholesky factorization that is provably backward stable. We further employed a perturbation analysis to indicate the best accuracy that can be expected from any finite precision algorithm (slow or fast), and then showed that the modified Schur algorithm of [4] essentially achieves this bound. For all practical purposes, the major conclusion of the analysis in [4] was that the modified Schur algorithm is backward stable for a large class of structured matrices.

The above results have further motivated us to tackle the standing issue of deriving an algorithm that is both *fast and stable* for the solution of *nonsymmetric* structured linear systems of equations $Tx = b$, where T is shift structured (to be defined later). The stability analyses of the generalized Schur algorithm that we referred to above do not apply in this case since the structured matrix T is not positive-definite (it is not even required to be symmetric). The only restriction on T is invertibility.

The way we approach the problem is motivated by embedding ideas pursued in [5, 13]. We first embed the given $n \times n$ matrix T into a larger $2n \times 2n$ matrix M that

is defined by

$$M = \begin{bmatrix} T^T T & T^T \\ T & \mathbf{0} \end{bmatrix}. \tag{1.2}$$

The matrix M is symmetric but still indefinite; while its leading $n \times n$ submatrix is positive-definite (equal to $T^T T$), its Schur complement with respect to the $(1,1)$ block is negative-definite (and equal to $-I$). [The product $T^T T$ is not formed explicitly, as explained later.]

We then apply $2n$ steps of the generalized Schur algorithm to M and obtain its *computed* triangular factorization, which is of the form

$$\begin{bmatrix} \hat{R}^T & \mathbf{0} \\ \hat{Q} & \Delta \end{bmatrix} \begin{bmatrix} \hat{R} & \hat{Q}^T \\ \mathbf{0} & -\Delta^T \end{bmatrix},$$

where $\hat{R}^T$ and Δ are $n \times n$ lower triangular matrices. The matrices $\{\hat{R}, \hat{Q}, \Delta\}$ are the quantities used in (1.1) to determine the computed solution $\hat{x}$ in a backward stable manner.

From a numerical point of view, the above steps differ in crucial ways from the embeddings suggested in [5, 13], and which turn out to mark the difference between a numerically stable and a numerically unstable implementation.

The discussion in [5] (pages 37,50,52) and [13] is mainly concerned with fast procedures for the QR factorization of Toeplitz-block and block-Toeplitz matrices. It employs an embedding of the form

$$M = \begin{bmatrix} T^T T & T^T \\ T & I \end{bmatrix}, \tag{1.3}$$

where the identity matrix I in (1.3) replaces the zero matrix in our embedding (1.2). The derivation in [5, 13] suggests applying n (rather than $2n$) steps of the generalized Schur algorithm to (1.3) and then uses the resulting $\hat{R}$ and $\hat{Q}$ as the QR factors of T. This procedure however *does not* guarantee a numerically orthogonal matrix $\hat{Q}$ and can not therefore be used to implement a stable solver for a linear system of equations $Tx = b$.

For this reason, we instead propose in this paper to proceed with the earlier embedding (1.2) since it seems difficult to obtain a stable algorithm that is solely based on the alternative embedding (1.3). We also apply $2n$ steps (rather than just n steps) of the generalized Schur algorithm to (1.2). This allows us to incorporate a correction procedure into the algorithm that is shown to ensure backward stability, when coupled with other modifications that are needed, especially while applying the hyperbolic rotations.

1.1. Notation. In the discussion that follows we use $\|\cdot\|$ to denote the $2-$norm of its argument. Also, the $\hat{}$ notation denotes computed quantities, and we use ϵ to denote the machine precision and n the matrix size. We also use subscripted δ's to denote quantities bounded by machine precision in magnitude, and subscripted c's to denote low order polynomials in n.

We assume that in our floating point model, additions, subtractions, multiplications, divisions, and square-roots are done to high relative accuracy, i.e.,

$$fl(x \circ y) = (x \circ y)(1 + \delta),$$

where $\circ$ denotes $+, -, \times, \div$ and $|\delta| \leq \epsilon$. Likewise for the square-root operation. This is true for floating point processors that adhere to the IEEE standards.

2. Displacement Structure. Consider an $n \times n$ symmetric matrix M and an $n \times n$ lower-triangular real-valued matrix F. The displacement of M with respect to F is denoted by ∇_F and defined as

$$\nabla_F = M - FMF^T. \tag{2.1}$$

The matrix M is said to have low displacement rank with respect to F if the rank of ∇_F is considerably lower than n. In this case, M is said to have displacement structure with respect to F [15].

Let $r \ll n$ denote the rank of ∇_F. It follows that we can factor ∇_F as

$$\nabla_F = GJG^T , \tag{2.2}$$

where G is an $n \times r$ matrix and J is a signature matrix of the form

$$J = \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix} , \quad p+q = r . \tag{2.3}$$

The integer p denotes the number of positive eigenvalues of ∇_F, while the integer q denotes the number of its negative eigenvalues. The factorization (2.2) is highly nonunique. If G satisfies (2.2) then $G\Theta$ also satisfies (2.2) for any $J-$unitary matrix Θ, i.e., for any Θ such that $\Theta J\Theta^T = J$. This follows from the trivial identity

$$(G\Theta)J(G\Theta)^T = G(\Theta J\Theta^T)G^T = GJG^T.$$

Combining (2.1) and (2.2), a matrix M is said to be structured with respect to the displacement operation defined by (2.1) if it satisfies a displacement equation of the form

$$M - FMF^T = GJG^T , \tag{2.4}$$

with a "low" rank matrix G. Equation (2.4) uniquely defines M (i.e., it has a unique solution M) if, and only if, the diagonal entries of the lower-triangular matrix F satisfy the condition

$$1 - f_i f_j \neq 0 \ \text{ for all } \ i, j.$$

This uniqueness condition will hold for the cases studied in this paper. (It can be relaxed in some instances [15]).

The pair (G, J) is said to be a generator pair for M since, along with F, it completely identifies M. Note however that, while M has n^2 entries, the matrix G has nr entries and r is usually much smaller than n. Therefore, algorithms that operate on the entries of G, with the purpose of obtaining a triangular factorization for M, will generally be an order of magnitude faster than algorithms that operate on the entries of M itself. The generalized Schur algorithm is one such fast $O(rn^2)$ procedure, which receives as input data the matrices (F, G, J) and provides as output data the triangular factorization of M. A recent survey on various other forms of displacement structure and on the associated forms of Schur algorithms can be found in [15].

The notion of structured matrices can also be extended to nonsymmetric matrices M. In this case, the displacement of M is generally defined with respect to two lower triangular matrices F and A (which can be the same, i.e., $F = A$ – see (2.10)),

$$\nabla_{F,A} = M - FMA^T, \tag{2.5}$$

and the low-rank difference matrix $\nabla_{F,A}$ is (nonuniquely) factored as

$$\nabla_{F,A} = GB^T \ , \tag{2.6}$$

where G and B are $n \times r$ generator matrices, i.e.,

$$M - FMA^T = GB^T. \tag{2.7}$$

Again, this displacement equation uniquely defines M iff the diagonal entries of F and A satisfy $1 - f_i a_j \neq 0$ for all i, j, a condition that will be met in this paper.

2.1. Toeplitz, Quasi-Toeplitz, and Shift Structured Matrices. The concept of displacement structure is perhaps best introduced by considering the much-studied special case of a symmetric Toeplitz matrix, $T = \left[t_{|i-j|}\right]_{i,j=1}^{n}$, $t_0 = 1$.

Let Z denote the $n \times n$ lower triangular shift matrix with ones on the first subdiagonal and zeros elsewhere (i.e., a lower triangular Jordan block with eigenvalue 0):

$$Z = \begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix} . \tag{2.8}$$

It can be easily checked that the difference $T - ZTZ^T$ has displacement rank 2 (except when all $t_i, i \neq 0$, are zero), and a generator for T is $\{G, (1 \oplus -1)\}$ where

$$T - ZTZ^T = \begin{bmatrix} 1 & 0 \\ t_1 & t_1 \\ \vdots & \vdots \\ t_{n-1} & t_{n-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_1 & t_1 \\ \vdots & \vdots \\ t_{n-1} & t_{n-1} \end{bmatrix}^T = GJG^T \ . \tag{2.9}$$

Similarly, for a nonsymmetric Toeplitz matrix, $T = [t_{i-j}]_{i,j=1}^{n}$, we can easily verify that the difference $T - ZTZ^T$ has displacement rank 2 and that a generator (G, B) for T is

$$T - ZTZ^T = \begin{bmatrix} t_0 & 1 \\ t_1 & 0 \\ \vdots & \vdots \\ t_{n-1} & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & t_{-1} \\ \vdots & \vdots \\ 0 & t_{-n+1} \end{bmatrix}^T = GB^T \ . \tag{2.10}$$

This is a special case of (2.7) with $F = A = Z$. In particular, any matrix T for which $(T - ZTZ^T)$ has rank 2 is called *quasi-Toeplitz*, i.e.,

$$T - ZTZ^T = GB^T \text{ has rank 2.} \tag{2.11}$$

For example, the inverse of a Toeplitz matrix is quasi-Toeplitz [15].

Later in the paper we shall focus on the class of *shift structured* matrices (cf. (4.1)), which includes Toeplitz and quasi-Toeplitz matrices as special cases. These are all matrices that are structured with respect to $F = A = Z$. For ease of reference, we define the terminology below.

DEFINITION 2.1.

1. *Any matrix that is structured with respect to the shift operators $F = Z$ and $A = Z$ will be said to be shift structured. That is, for shift structured matrices the rank of $\nabla_{Z,Z}$ (or displacement rank) is low compared to n.*
2. *A quasi-Toeplitz matrix is a shift structured matrix with displacement rank 2.*

For example, the product of two Toeplitz matrices is shift structured with displacement rank 4 [15].

3. The Generalized Schur Algorithm. An efficient algorithm for the triangular factorization of symmetric or nonsymmetric structured matrices (of either forms (2.4) or (2.7)) is the generalized Schur algorithm [15]. For our purposes, it is sufficient to describe the algorithm here for *symmetric* structured matrices M of the form (2.4), with a *strictly* lower-triangular matrix F. This includes, for example, the following special choices for F: $F = Z$, $F = Z^2$, $F = (Z \oplus Z)$, etc. The matrix M is further assumed to be strongly regular (i.e., all its leading submatrices are nonsingular).

A generator matrix G is said to be in *proper* form if its first nonzero row has a single nonzero entry, say in the first column

$$G = \begin{bmatrix} x & 0 & 0 & 0 & 0 \\ x & x & x & x & x \\ x & x & x & x & x \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x & x & x & x & x \end{bmatrix}, \tag{3.1}$$

or in the last column

$$G = \begin{bmatrix} 0 & 0 & 0 & 0 & x \\ x & x & x & x & x \\ x & x & x & x & x \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x & x & x & x & x \end{bmatrix}. \tag{3.2}$$

The generalized Schur algorithm operates on the entries of (F, G, J), which describe the displacement structure of M in (2.4) (assumed strongly regular), and provides the triangular factorization of M [15].

ALGORITHM 3.1 (The Generalized Schur Algorithm).

- *Input data: An $n \times n$ strictly lower-triangular matrix F, an $n \times r$ generator $G_1 = G$, and $J = (I_p \oplus -I_q)$.*
- *Output data: A lower-triangular factor L and a signature matrix D such that $M = LDL^T$, where M is the solution of (2.4) (assumed $n \times n$).*

The algorithm operates as follows: start with $G_1 = G$, $F_1 = F$, and repeat for $i = 1, 2, \ldots, n$:

1. *Let g_i denote the top row of G_i.*
2. *If $g_i J g_i^T > 0$ (we refer to this case as a positive step):*

- *Choose a $J-$unitary rotation Θ_i that converts g_i to proper form with respect to the first column, i.e.,*

$$g_i\Theta_i = \begin{bmatrix} x & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.3}$$

Let $\bar{G}_i = G_i\Theta_i$ (i.e., apply Θ_i to G_i).

- *The nonzero part of the $i-th$ column of L, denoted by $\bar{l}_i$, is the first column of $\bar{G}_i$,*

$$\bar{l}_i = \bar{G}_i \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}. \tag{3.4}$$

The $i-th$ column of L, denoted by l_i, is obtained by appending $(i-1)$ zero entries to $\bar{l}_i$,

$$l_i = \begin{bmatrix} \mathbf{0} \\ \bar{l}_i \end{bmatrix}. \tag{3.5}$$

The $i-th$ signature is $d_i = 1$.

- *Keep the last columns of $\bar{G}_i$ unchanged and multiply the first column by F_i, where F_i denotes the submatrix obtained by deleting the first $(i-1)$ rows and columns of F. This provides a new matrix whose first row is zero (since F_i is strictly lower triangular) and whose last rows are the rows of the next generator matrix G_{i+1}, i.e.,*

$$\begin{bmatrix} \mathbf{0} \\ G_{i+1} \end{bmatrix} = \begin{bmatrix} F_i\bar{l}_i & \bar{G}_i \begin{bmatrix} 0 & \\ & I \end{bmatrix} \end{bmatrix}. \tag{3.6}$$

3. *If $g_iJg_i^T < 0$ (we refer to this case as a negative step):*

- *Choose a $J-$unitary rotation Θ_i that converts g_i to proper form with respect to the last column, i.e.,*

$$g_i\Theta_i = \begin{bmatrix} 0 & 0 & 0 & 0 & x \end{bmatrix}. \tag{3.7}$$

Let $\bar{G}_i = G_i\Theta_i$ (i.e., apply Θ_i to G_i).

- *The nonzero part of the $i-th$ column of L, denoted by $\bar{l}_i$, is the last column of $\bar{G}_i$,*

$$\bar{l}_i = \bar{G}_i \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix}. \tag{3.8}$$

The $i-th$ column of L, denoted by l_i, is obtained by appending $(i-1)$ zero entries to $\bar{l}_i$,

$$l_i = \begin{bmatrix} \mathbf{0} \\ \bar{l}_i \end{bmatrix}. \tag{3.9}$$

The $i-th$ signature is $d_i = -1$.

- *Keep the first columns of $\bar{G}_i$ unchanged and multiply the last column by F_i. This provides a new matrix whose first row is zero (since F_i is strictly lower triangular) and whose last rows are the rows of the next generator matrix G_{i+1}, i.e.,*

$$\begin{bmatrix} \mathbf{0} \\ G_{i+1} \end{bmatrix} = \begin{bmatrix} \bar{G}_i \begin{bmatrix} I & \\ & 0 \end{bmatrix} & F_i\bar{l}_i \end{bmatrix}. \tag{3.10}$$

4. The case $g_i J g_i^T = 0$ is ruled out by the strong regularity of M.

Schematically, for the special case $r = 2$, we have the following simple array picture for a positive-step case (a similar picture holds for a negative-step case):

$$G_i = \begin{bmatrix} x & x \\ x & x \\ x & x \\ \vdots & \vdots \end{bmatrix} \xrightarrow{\Theta_i} \underbrace{\begin{bmatrix} x' & 0 \\ x' & x' \\ x' & x' \\ \vdots & \vdots \end{bmatrix}}_{\bar{G}_i} \xrightarrow{\text{apply } F_i} \begin{bmatrix} 0 & 0 \\ x'' & x' \\ x'' & x' \\ \vdots & \vdots \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ G_{i+1} \end{bmatrix} . \tag{3.11}$$

In words:

- Use the top row of G_i to define a $J-$unitary matrix Θ_i that transforms this row to the form $\begin{bmatrix} x' & 0 \end{bmatrix}$;
- Multiply G_i by Θ_i and keep the last columns unchanged;
- Apply F_i to the first column of $\bar{G}_i = G_i \Theta_i$;
- These two operations result in G_{i+1}.

The rotations Θ_i are always guaranteed to exist and they can be constructed in different ways (see, e.g., Lemma 4.3 and Sec. 4.4.1 in [15]).

After n steps, the algorithm provides the triangular decomposition [15]

$$M = \sum_{i=1}^{n} d_i l_i l_i^T , \tag{3.12}$$

at $O(rn^2)$ computational cost.

Moreover, the successive matrices G_i that are obtained via the algorithm have an interesting interpretation. Let M_i denote the Schur complement of M with respect to its leading $(i-1) \times (i-1)$ submatrix. That is, $M_1 = M$, M_2 is the Schur complement with respect to the $(1,1)$ top left entry of M, M_3 is the Schur complement with respect to the 2×2 top left submatrix of M, and so on. The matrices M_i are therefore $(n-i+1) \times (n-i+1)$. Recall also that F_i denotes the submatrix obtained by deleting the first $(i-1)$ rows and columns of F. Hence, M_i and F_i have the same dimensions.

While the M_i are never computed explicitly, it can be shown that (M_i, F_i, G_i) satisfy the displacement equation [15]

$$M_i - F_i M_i F_i^T = G_i J G_i^T . \tag{3.13}$$

Hence, G_i constitutes a generator matrix for the $i-th$ Schur complement M_i, which is therefore structured. Note further that $\bar{G}_i$ is also a generator matrix for the same Schur complement M_i since, due to the $J-$unitarity of Θ_i, we have $\bar{G}_i J \bar{G}_i^T = G_i \Theta_i J \Theta_i^T G_i^T = G_i J G_i^T$.

We summarize the above discussion in the following statement, deliberately stated in loose terms.

LEMMA 3.2. *The successive Schur complements of a structured matrix are also structured and the generalized Schur algorithm is a recursive procedure that provides generator matrices for the successive Schur complements. It also provides the triangular factors of the original matrix.*

We also indicate here, for later reference, that two successive Schur complements M_i and M_{i+1} are related via the Schur complementation step:

$$M_i = d_i \bar{l}_i \bar{l}_i^T + \begin{bmatrix} 0 & 0 \\ 0 & M_{i+1} \end{bmatrix}. \tag{3.14}$$

We now address the main issues of this paper.

4. Fast QR Factorization of Shift Structured Matrices. Let T be an $n \times n$ shift structured matrix (possibly nonsymmetric) with displacement rank r,

$$T - ZTZ^T = GB^T. \tag{4.1}$$

Special cases include the Toeplitz matrix of equation (2.10) and quasi-Toeplitz matrices of equation (2.11), whose displacement ranks are equal to 2 ($r = 2$).

Consider the $3n \times 3n$ augmented matrix

$$M = \begin{bmatrix} -I & T & \mathbf{0} \\ T^T & \mathbf{0} & T^T \\ \mathbf{0} & T & \mathbf{0} \end{bmatrix}. \tag{4.2}$$

The matrix M is also structured (as shown below) with respect to $Z_n \oplus Z_n \oplus Z_n$, where Z_n denotes the $n \times n$ lower shift triangular matrix (denoted earlier by Z – here we include the subscript n in order to explicitly indicate the size of Z).

It can be easily verified that $M - (Z_n \oplus Z_n \oplus Z_n)M(Z_n \oplus Z_n \oplus Z_n)^T$ is low rank since

$$M - (Z_n \oplus Z_n \oplus Z_n)M(Z_n \oplus Z_n \oplus Z_n)^T = \begin{bmatrix} -e_1 e_1^T & GB^T & \mathbf{0} \\ BG^T & \mathbf{0} & BG^T \\ \mathbf{0} & GB^T & \mathbf{0} \end{bmatrix}, \tag{4.3}$$

where $e_1 = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}^T$ is a basis vector of appropriate dimension. A generator matrix for M, with $3n$ rows and $(2r+1)$ columns, can be seen to be

$$\mathcal{G} = \frac{1}{\sqrt{2}} \begin{bmatrix} G & -G & e_1 \\ B & B & \mathbf{0} \\ G & -G & \mathbf{0} \end{bmatrix}, \quad \mathcal{J} = \begin{bmatrix} I_r & \\ & -I_{r+1} \end{bmatrix}. \tag{4.4}$$

That is,

$$M - \mathcal{F}M\mathcal{F}^T = \mathcal{G}\mathcal{J}\mathcal{G}^T,$$

where $\mathcal{F} = (Z_n \oplus Z_n \oplus Z_n)$ and $(\mathcal{G}, \mathcal{J})$ are as above.

The $n \times n$ leading submatrix of M is negative definite (in fact, equal to $-I$). Therefore, the first n steps of the generalized Schur algorithm applied to $(\mathcal{F}, \mathcal{G}, \mathcal{J})$ will be negative steps (cf. step 3 of Algorithm 3.1). These first n steps lead to a generator matrix, denoted by $\mathcal{G}_{n+1}$ (with $2n$ rows), for the Schur complement of M with respect to its leading $n \times n$ leading submatrix, viz.,

$$M_{n+1} - (Z_n \oplus Z_n)M_{n+1}(Z_n \oplus Z_n)^T = \mathcal{G}_{n+1}\mathcal{J}\mathcal{G}_{n+1}^T, \tag{4.5}$$

where M_{n+1} is $2n \times 2n$ and equal to

$$M_{n+1} = \begin{bmatrix} T^T T & T^T \\ T & \mathbf{0} \end{bmatrix}. \tag{4.6}$$

Clearly, M and its Schur complement M_{n+1} are related via the Schur complement relation (cf. (3.14))

$$M = \begin{bmatrix} I \\ -T^T \\ \mathbf{0} \end{bmatrix} (-I) \begin{bmatrix} I & -T^T & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & T^T T & T^T \\ \mathbf{0} & T & \mathbf{0} \end{bmatrix} .$$

Therefore, $(\mathcal{G}_{n+1}, \mathcal{J})$ is a generator for M_{n+1} with respect to $(Z_n \oplus Z_n)$, as shown by (4.5).

The leading $n \times n$ submatrix of M_{n+1} is now positive-definite (equal to $T^T T$). Therefore, the next n steps of the generalized Schur algorithm applied to $(Z_n \oplus Z_n, \mathcal{G}_{n+1}, \mathcal{J})$ will be positive steps (cf. step 2 of Algorithm 3.1). These steps lead to a generator matrix, denoted by $\mathcal{G}_{2n+1}$ (with n rows), for the Schur complement of M with respect to its leading $2n \times 2n$ leading submatrix, viz.,

$$M_{2n+1} - Z_n M_{2n+1} Z_n^T = \mathcal{G}_{2n+1} \mathcal{J} \mathcal{G}_{2n+1}^T ,$$

where M_{2n+1} is now $n \times n$ and equal to $-I$.

Again, M_{n+1} and M_{2n+1} are related via a (block) Schur complementation step (cf. (3.14)), written as

$$\begin{bmatrix} T^T T & T^T \\ T & \mathbf{0} \end{bmatrix} = M_{n+1} = \begin{bmatrix} R^T \\ Q \end{bmatrix} (I) \begin{bmatrix} R & Q^T \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -I \end{bmatrix}, \tag{4.7}$$

where we have denoted the first n columns of the triangular factor of M_{n+1} by

$$\begin{bmatrix} R^T \\ Q \end{bmatrix}$$

with R an $n \times n$ upper triangular matrix and Q an $n \times n$ matrix. The R and Q matrices are thus obtained by splitting the first n columns of the triangular factor of M_{n+1} into a leading lower triangular block followed by a full matrix Q.

By equating terms on both sides of (4.7) we can explicitly identify R and Q as follows:

$$T^T T = R^T R \ , \ T = QR \ , \quad QQ^T - I = \mathbf{0}.$$

These relations show that Q and R define the QR factors of the matrix T.

In summary, the above discussion shows the following: given a shift structured matrix T as in (4.1), its QR factorization can be computed efficiently by applying $2n$ steps of the generalized Schur algorithm to the matrices $(\mathcal{F}, \mathcal{G}, \mathcal{J})$ defined in (4.4). The factors Q and R can be obtained from the triangular factors $\{l_i\}$ for $i = n + 1, n + 2, \ldots, 2n$.

Alternatively, if a generator matrix is directly available for M_{n+1} in (4.6) (see Sec. 4.1), then we need only apply n Schur steps to the generator matrix and read the factors Q and R from the resulting n columns of the triangular factor.

In the later sections of this paper we shall establish, for convenience of exposition, the numerical stability of a fast solver for $Tx = b$ that starts with a generator matrix for the embedding (4.6) rather than the embedding (4.2). It will become clear however that the same conclusions will hold if we instead start with a generator matrix for the embedding (4.2).

The augmentation (4.2) was used in [16, 17] and it is based on embedding ideas originally pursued in [5, 13] (see Sec.4.2).

4.1. The Toeplitz Case. In some cases it is possible to find an explicit generator matrix for M_{n+1}. This saves the first n steps of the generalized Schur algorithm.

For example, consider the case when T is a Toeplitz matrix (which is a special case of (4.1) whose first column is $[t_0, t_1, \ldots, t_{n-1}]^T$ and whose first row is $[t_0, t_{-1}, \ldots, t_{-n+1}]$. Define the vectors

$$\begin{bmatrix} c_0 \\ \vdots \\ c_{n-1} \end{bmatrix} = \frac{Te_1}{\|Te_1\|}, \qquad \begin{bmatrix} s_0 \\ \vdots \\ s_{n-1} \end{bmatrix} = T^T \begin{bmatrix} c_0 \\ \vdots \\ c_{n-1} \end{bmatrix}.$$

It can be verified that a generator matrix for M_{n+1} in (4.6) is the following [5]

$$M_{n+1} - (Z_n \oplus Z_n) M_{n+1} (Z_n \oplus Z_n)^T = \mathcal{G}_{n+1} \mathcal{J} \mathcal{G}_{n+1}^T ,$$

where $\mathcal{J}$ is 5×5,

$$\mathcal{J} = \mathrm{diag}[1, 1, -1, -1, -1],$$

and $\mathcal{G}_{n+1}$ is $2n \times 5$,

$$\mathcal{G}_{n+1} = \begin{bmatrix} s_0 & 0 & 0 & 0 & 0 \\ s_1 & t_{-1} & s_1 & t_{n-1} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{n-1} & t_{-n+1} & s_{n-1} & t_1 & 0 \\ c_0 & 1 & c_0 & 0 & 1 \\ c_1 & 0 & c_1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \\ c_{n-1} & 0 & c_{n-1} & 0 & 0 \end{bmatrix}.$$

4.2. Other Augmentations. It is also possible to compute the QR factors of a structured matrix T satisfying (4.1) by using other augmented matrices, other than (4.2). For example, consider the $3n \times 3n$ augmented matrix

$$M = \begin{bmatrix} -I & T & \mathbf{0} \\ T^T & \mathbf{0} & T^T \\ \mathbf{0} & T & I \end{bmatrix}, \tag{4.8}$$

where an identity matrix replaces the zero matrix in the $(3,3)$ block entry of the matrix in (4.2). A generator matrix for M, with $3n$ rows and $(2r+2)$ columns, is now

$$\mathcal{G} = \frac{1}{\sqrt{2}} \begin{bmatrix} G & \mathbf{0} & -G & e_1 \\ B & \mathbf{0} & B & \mathbf{0} \\ G & e_1 & -G & \mathbf{0} \end{bmatrix}, \quad \mathcal{J} = \begin{bmatrix} I_{r+1} & \\ & -I_{r+1} \end{bmatrix}.$$

If T is Toeplitz, as in Sec. 4.1, then the rank of $\mathcal{G}$ can be shown to reduce to $2r = 4$ [5] (this is in contrast to the displacement rank 5 that follows from the earlier embedding (4.2), as shown in Sec. 4.1).

After $2n$ steps of the generalized Schur algorithm applied to the above $(\mathcal{G}, \mathcal{J})$, we obtain the following factorization (since now $M_{2n+1} = \mathbf{0}$),

$$M = \begin{bmatrix} I & \mathbf{0} \\ -T^T & R^T \\ \mathbf{0} & Q \end{bmatrix} \begin{bmatrix} -I & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix} \begin{bmatrix} I & \mathbf{0} \\ -T^T & R^T \\ \mathbf{0} & Q \end{bmatrix}^T ,$$

from which we can again read the QR factors of T from the triangular factors $\{l_i\}$ for $i = n+1, \ldots, 2n+1$. This augmentation was suggested in [5, p. 37] and [13].

However, from a numerical point of view, computing the QR factors of a structured matrix T using the generalized Schur algorithm on the augmented matrices M in (4.2) or (4.8) is not stable. The problem is that the computed Q matrix is not necessarily orthogonal. This is also true for other procedures for fast QR factorization [1, 7, 8, 19].

In the next section we show how to overcome this difficulty and develop a fast and stable algorithm for solving linear systems of equations with shift structured coefficient matrices T. For this purpose, we proceed with the embedding suggested earlier in (4.2) since it seems difficult to obtain a stable algorithm that is based solely on the alternative embedding (4.8). The reason is that the embedding (4.2) allows us to incorporate a correction procedure into the algorithm in order to ensure stability.

We first derive a stable algorithm for a well-conditioned coefficient matrix, and then modify it for the case when the coefficient matrix is ill-conditioned. The interested reader may consult at this time the summary of the final algorithm that is provided in Sec. 10.

5. Well-Conditioned T. In this section we develop a stable algorithm for the case of well-conditioned matrices T. A definition of what we mean by a well-conditioned matrix is given further ahead (see (5.19)). Essentially this refers to matrices whose condition number is less than the reciprocal of the square-root of the machine precision. Modifications to handle the ill-conditioned case will be introduced later in the paper.

We start with an $n \times n$ (possibly nonsymmetric) shift structured matrix T with displacement rank r,

$$T - Z_n T Z_n^T = GB^T, \tag{5.1}$$

and assume we have available a generator matrix $\mathcal{G}$ for the $2n \times 2n$ augmented matrix

$$M = \begin{bmatrix} T^T T & T^T \\ T & \mathbf{0} \end{bmatrix}, \tag{5.2}$$

that is,

$$M - \mathcal{F} M \mathcal{F}^T = \mathcal{G} \mathcal{J} \mathcal{G}^T, \tag{5.3}$$

where $\mathcal{F} = (Z_n \oplus Z_n)$. Note that, for ease of exposition, we have modified our notation. While we have earlier denoted the above matrix M by M_{n+1}, its generator by $\mathcal{G}_{n+1}$, and have used $\mathcal{F}$ to denote $(Z_n \oplus Z_n \oplus Z_n)$, we are now dropping the subscript $n+1$ from $(M_{n+1}, \mathcal{G}_{n+1})$ and are using $\mathcal{F}$ to denote the $2n \times 2n$ matrix $(Z_n \oplus Z_n)$.

In Section 4.1 we have discussed an example where we have shown a particular generator matrix $\mathcal{G}$ for M when T is Toeplitz. [We repeat that the error analysis of later sections will still apply if we instead start with the $3n \times 3n$ embedding (4.2) and its generator matrix (4.4)].

We have indicated earlier (at the end of Sec. 4) that by applying n steps of the generalized Schur algorithm to the matrix M in (5.2) we can obtain the QR factorization of T from the resulting n columns of the triangular factors of M. But this procedure is not numerically stable since the resulting Q is not guaranteed to be unitary. To fix this problem, we propose some modifications. The most relevant modification we introduce now is to run the Schur algorithm for $2n$ steps on M rather

than just n steps. As suggested in the paper [4], we also need to be careful in the application of the hyperbolic rotations. In particular, we assume that the hyperbolic rotations are applied using one of the methods suggested in the paper [4] (mixed downdating, OD-method, or H-procedure – see Appendices A and B at the end of this paper).

The matrix T is only required to be invertible. In this case, the leading submatrix of M in (5.2) is positive-definite and therefore the first n steps of the generalized Schur algorithm will be positive steps. Hence, the hyperbolic rotations needed for the first n steps will perform transformations of the form (3.3), where generators are transformed into proper form with respect to their first column. Likewise, the Schur complement of M with respect to its leading submatrix $T^T T$ is equal to $-I$, which is negative-definite. This means that the last n steps of the generalized Schur algorithm will be negative steps. Hence, the hyperbolic rotations needed for the last n steps will perform transformations of the form (3.7), where generators are transformed into proper form with respect to their last column.

During a positive step (a similar discussion holds for a negative step), a generator matrix G_i will be reduced to proper form by implementing the hyperbolic transformation Θ_i as a sequence of orthogonal transformations followed by a 2×2 hyperbolic rotation (see also [18]). The 2×2 rotation is implemented along the lines of [4], e.g., via mixed-downdating [3], or the OD-method, or the H-procedure (see Appendices A and B for a description of the OD- and H-procedures [4]). Details are given below.

5.1. Implementation of the $\mathcal{J}$-Unitary Rotations Θ_i. When the generalized Schur algorithm is applied to $(\mathcal{G}, \mathcal{F})$ in (5.3), we proceed through a sequence of generator matrices $(\mathcal{G}, \mathcal{G}_2, \mathcal{G}_3, \ldots)$ of decreasing number of rows $(2n, 2n-1, 2n-2, \ldots)$. Let g_i denote the top row of the generator matrix $\mathcal{G}_i$ at step i. In a positive step, it needs to be reduced to the form (3.3) via an $(I_p \oplus -I_q)-$unitary rotation Θ_i. We propose to perform this transformation as follows:

1. Apply a *unitary* (orthogonal) rotation (e.g., Householder) to the first p columns of $\mathcal{G}_i$ so as to reduce the top row of these p columns into proper form,

$$g_i = \begin{bmatrix} x & x & x & x & x & x \end{bmatrix} \overset{unitary\ \Theta_{i,1}}{\longrightarrow} \begin{bmatrix} x & 0 & 0 & x & x & x \end{bmatrix} = g_{i,1},$$

with a nonzero entry in the first column. Let

$$\mathcal{G}_{i,1} = \mathcal{G}_i \begin{bmatrix} \Theta_{i,1} & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix} \tag{5.4}$$

denote the modified generator matrix. Its last q columns coincide with those of $\mathcal{G}_i$.

2. Apply another *unitary* (orthogonal) rotation (e.g., Householder) to the last q columns of $\mathcal{G}_{i,1}$ so as to reduce the top row of these last q columns into proper form with respect to their last column,

$$g_{i,1} = \begin{bmatrix} x & 0 & 0 & x & x & x \end{bmatrix} \overset{unitary\ \Theta_{i,2}}{\longrightarrow} \begin{bmatrix} x & 0 & 0 & 0 & 0 & x \end{bmatrix} = g_{i,2},$$

with a nonzero entry in the last column. Let

$$\mathcal{G}_{i,2} = \mathcal{G}_{i,1} \begin{bmatrix} I & \mathbf{0} \\ \mathbf{0} & \Theta_{i,2} \end{bmatrix} \tag{5.5}$$

denote the modified generator matrix. Its first p columns coincide with those of $\mathcal{G}_{i,1}$.

3. Employ an elementary hyperbolic rotation $\Theta_{i,3}$ acting on the first and last columns (in mixed-downdating [3] form, or according to the OD or the H methods of [4] – see also Appendices A and B) in order to annihilate the nonzero entry in the last column,

$$g_{i,2} = \begin{bmatrix} x & 0 & 0 & 0 & x \end{bmatrix} \overset{hyperbolic\ \Theta_{i,3}}{\longrightarrow} \begin{bmatrix} x & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

4. The combined effect of the above steps is to reduce g_i to the proper form (3.3) and, hence,

$$\bar{\mathcal{G}}_i = \mathcal{G}_i \begin{bmatrix} \Theta_{i,1} & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix} \begin{bmatrix} I & \mathbf{0} \\ \mathbf{0} & \Theta_{i,2} \end{bmatrix} \Theta_{i,3}. \tag{5.6}$$

Expression (5.6) shows that, in infinite precision, the generator matrices $\mathcal{G}_i$ and $\bar{\mathcal{G}}_i$ must satisfy the fundamental requirement

$$\mathcal{G}_i \mathcal{J} \mathcal{G}_i^T = \bar{\mathcal{G}}_i \mathcal{J} \bar{\mathcal{G}}_i^T. \tag{5.7}$$

Obviously, this condition cannot be guaranteed in finite precision. But with the above implementation of the transformation (5.6) (as a sequence of two orthogonal transformations and a hyperbolic rotation in mixed, OD, or H-forms), equality (5.7) can be guaranteed to within a "small" error (see (5.8)). Indeed, it follows from (5.4) and (5.5), and from the orthogonality of $\Theta_{i,1}$ and $\Theta_{i,2}$, that

$$\|\hat{\mathcal{G}}_{i,2} \mathcal{J} \hat{\mathcal{G}}_{i,2}^T - \mathcal{G}_i \mathcal{J} \mathcal{G}_i^T\| \leq c_2 \epsilon \|\mathcal{G}_i\|^2,$$

and

$$\left| \|\hat{\mathcal{G}}_{i,2}\|^2 - \|\mathcal{G}_i\|^2 \right| \leq c_3 \epsilon \|\mathcal{G}_i\|^2.$$

It further follows from the error bound (A.3) (in the appendix) that

$$\|\hat{\bar{\mathcal{G}}}_i \mathcal{J} \hat{\bar{\mathcal{G}}}_i^T - \hat{\mathcal{G}}_{i,2} \mathcal{J} \hat{\mathcal{G}}_{i,2}^T\| \leq c_4 \epsilon \left(\|\hat{\bar{\mathcal{G}}}_i\|^2 + \|\hat{\mathcal{G}}_{i,2}\|^2 \right).$$

Combining the above error bounds we conclude that the following holds:

$$\|\hat{\bar{\mathcal{G}}}_i \mathcal{J} \hat{\bar{\mathcal{G}}}_i^T - \mathcal{G}_i \mathcal{J} \mathcal{G}_i^T\| \leq c_5 \epsilon \left(\|\hat{\bar{\mathcal{G}}}_i\|^2 + \|\mathcal{G}_i\|^2 \right). \tag{5.8}$$

A similar analysis holds for a negative step, where the hyperbolic rotation $\Theta_{i,3}$ is again implemented as a sequence of two unitary rotations and one elementary hyperbolic rotation in order to guarantee the transformation (3.7). We forgo the details here.

We finally remark that in the algorithm, the incoming generator matrix $\mathcal{G}_i$ will in fact be the computed version, which we denote by $\hat{\mathcal{G}}_i$. This explains why in the error analysis of the next section (see (5.11) and (5.13)) we replace $\mathcal{G}_i$ by $\hat{\mathcal{G}}_i$ in the error bound (5.8).

Note that we are implicitly assuming that the required hyperbolic rotation $\Theta_{i,3}$ exists. While that can be guaranteed in infinite precision, it is possible that in finite precision we can experience break downs. This matter is handled in section 5.3.

5.2. Error Analysis of the First n Steps. After the first n steps of the generalized Schur algorithm applied to $(\mathcal{F}, \mathcal{G})$ in (5.3), we let

$$\begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix}$$

denote the computed factors that correspond to expression (4.7). We further define the matrix S_{n+1} that solves the displacement equation

$$S_{n+1} - Z_n S_{n+1} Z_n^T = \hat{\mathcal{G}}_{n+1} \mathcal{J} \hat{\mathcal{G}}_{n+1}^T. \tag{5.9}$$

Note that S_{n+1} is an $n \times n$ matrix, which in infinite precision would have been equal to the Schur complement $-I$ (cf. (4.7)). We can now define

$$\hat{M} = \begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix} \begin{bmatrix} \hat{R} & \hat{Q}^T \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & S_{n+1} \end{bmatrix}. \tag{5.10}$$

We also define the difference

$$N_i = \hat{\bar{\mathcal{G}}}_i \mathcal{J} \hat{\bar{\mathcal{G}}}_i^T - \hat{\mathcal{G}}_i \mathcal{J} \hat{\mathcal{G}}_i^T, \tag{5.11}$$

and introduce the error matrix $E = M - \hat{M}$. Using (5.8), the error analysis in [4] (Sec. 7, eq. (41)) can be extended to show that the $2n \times 2n$ error matrix satisfies the equation

$$E - \mathcal{F} E \mathcal{F}^T = \sum_{i=1}^{n} N_i.$$

Consequently, since $\mathcal{F} = (Z_n \oplus Z_n)$ is nilpotent,

$$E = \sum_{k=0}^{n-1} \mathcal{F}^k \left(\sum_{i=1}^{n} N_i \right) (\mathcal{F}^k)^T .$$

If we further invoke the fact that $\mathcal{F}$ is contractive we conclude that

$$\|E\| \leq \sum_{k=0}^{n-1} \left\| \sum_{i=1}^{n} N_i \right\| \leq \sum_{k=0}^{n-1} \sum_{i=1}^{n} \|N_i\| = n \sum_{i=1}^{n} \|N_i\| , \tag{5.12}$$

where, according to (5.8),

$$\|N_i\| \leq c_5 \epsilon \left(\|\hat{\bar{\mathcal{G}}}_i\|^2 + \|\hat{\mathcal{G}}_i\|^2 \right) . \tag{5.13}$$

But since all columns of $\hat{\mathcal{G}}_{i+1}$ and $\hat{\bar{\mathcal{G}}}_i$ coincide, except for one column in $\hat{\bar{\mathcal{G}}}_i$ that is shifted down (mulitplied by $\mathcal{F}_i$) to produce the corresponding column in $\hat{\mathcal{G}}_{i+1}$, then we clearly have

$$\|\hat{\mathcal{G}}_{i+1}\|^2 \leq \|\hat{\bar{\mathcal{G}}}_i\|^2.$$

We can therefore rewrite (5.13) as

$$\|N_i\| \leq c_6 \epsilon \left(\|\hat{\mathcal{G}}_{i+1}\|^2 + \|\hat{\mathcal{G}}_i\|^2 \right) . \tag{5.14}$$

Substituting into (5.12) we obtain the following error bound

$$\|E\| \leq c_7 \epsilon \sum_{i=1}^{n} \left(\|\hat{\mathcal{G}}_{i+1}\|^2 + \|\hat{\mathcal{G}}_i\|^2 \right) \leq c_8 \epsilon \sum_{i=1}^{n} \|\hat{\mathcal{G}}_i\|^2 . \tag{5.15}$$

5.3. Avoiding Breakdown. The above error analysis assumes that the first n steps of the generalized Schur algorithm applied to $(\mathcal{G}, \mathcal{F})$ in (5.3) do not breakdown. That is, during the first n steps, the $\mathcal{J}$-unitary rotations Θ_i are well defined. This further requires that the leading submatrices of the first n successive Schur complements remain positive-definite. We now show that this can be guaranteed by imposing a lower bound on the minimum singular value of the matrix T (see (5.19) – this corresponds to requiring a well-conditioned T, an assumption that will be dropped in Sec. 7 when the algorithm is extended for ill-conditioned T).

The argument is inductive. We assume that the algorithm has successfully completed the first $(i-1)$ steps and define the matrix S_i that solves the displacement equation

$$S_i - \mathcal{F}_i S_i \mathcal{F}_i^T = \hat{\mathcal{G}}_i \mathcal{J} \hat{\mathcal{G}}_i^T, \quad 1 \leq i \leq (n+1) \tag{5.16}$$

where $\mathcal{F}_i$ is the submatrix obtained from $\mathcal{F}$ in (5.3) by deleting its first $(i-1)$ rows and columns. In particular, $\mathcal{F}_1 = \mathcal{F}$ and $\mathcal{F}_n = Z_n$. Note that S_i is an $(2n-i+1) \times (2n-i+1)$ matrix, which in infinite precision would have been equal to the Schur complement of M with respect to its leading $(i-1) \times (i-1)$ submatrix.

We further define, for $1 \leq i \leq n+1$, the matrices $\hat{M}_i$,

$$\hat{M}_i = \sum_{j=1}^{i-1} \hat{l}_i \hat{l}_i^T \; + \; S_i, \tag{5.17}$$

where the $\hat{l}_i$ are the computed triangular factors, given by (cf. (3.4) and (3.5)). We can again establish, by following the arguments of Sec. 7.1 in [4], that the error matrices $E_i = M - \hat{M}_i$ satisfy

$$E_i - \mathcal{F}_i E_i \mathcal{F}_i^T = \sum_{j=1}^{i-1} N_j.$$

This relation again establishes, along the lines of (5.15), that

$$\|M - \hat{M}_i\| \; \leq \; c_9 \epsilon \sum_{j=1}^{i-1} \|\hat{\mathcal{G}}_j\|^2.$$

Therefore, if the minimum eigenvalue of the leading $n \times n$ submatrix of M (which is equal to $T^T T$) meets the lower bound

$$\lambda_{\min}(T^T T) \; > \; c_9 \epsilon \sum_{j=1}^{i-1} \|\hat{\mathcal{G}}_j\|^2 \, , \tag{5.18}$$

then the leading $n \times n$ submatrix of $\hat{M}_i$ will be guaranteed to be positive-definite and the algorithm can continue to the next iteration.

This analysis suggests the following lower bound on the minimum singular value of T in order to avoid breakdown in the first n steps of the algorithm:

$$\sigma_{\min}^2(T) \; > \; 2 \, c_9 \epsilon \sum_{j=1}^{n} \|\hat{\mathcal{G}}_j\|^2 \, . \tag{5.19}$$

We refer to a matrix T that satisfies the above requirement as being well-conditioned [the scalar multiple 2 is made explicit for convenience in later discussion – see (5.29)].

THEOREM 5.1 (Error Bound). *The first n steps of the generalized Schur algorithm applied to $(\mathcal{F}, \mathcal{G})$ in (5.3), for a matrix T satisfying (5.19), and with the rotations Θ_i implemented as discussed in Sec. 5.1, guarantees the following error bound on the matrix $(M - \hat{M})$ (with $\hat{M}$ defined in (5.10)):*

$$\|M - \hat{M}\| \leq c_9 \epsilon \sum_{j=1}^{n} \|\hat{\mathcal{G}}_j\|^2. \tag{5.20}$$

5.4. Growth of Generators. The natural question then is how big can the norm of the generator matrices be? The analysis that follows is motivated by an observation in [18] that for matrices of the form $T^T T$, with T Toeplitz, there is no appreciable generator growth.

To establish an upper bound on the generator norm, we consider the generator matrix $\hat{\mathcal{G}}_i$ (at the $i-th$ step) and recall from the discussion that led to (5.6) that, in a positive-step, $\hat{\mathcal{G}}_i$ is transformed via three rotation steps: a unitary rotation $\Theta_{i,1}$ that reduces the first p columns of $\hat{\mathcal{G}}_i$ into proper form, a second unitary rotation $\Theta_{i,2}$ that reduces the last q columns of $\hat{\mathcal{G}}_i$ into proper form, and a last elementary hyperbolic rotation $\Theta_{i,3}$ that reduces the overall generator matrix $\hat{\mathcal{G}}_i$ into proper form.

We denote the first and last columns of $\hat{\mathcal{G}}_i$ by $\hat{u}_i$ and $\hat{v}_i$, respectively, and denote the remaining columns by the block matrices $\hat{U}_i$ and $\hat{V}_i$, i.e., we write

$$\hat{\mathcal{G}}_i = \begin{bmatrix} \hat{u}_i & \hat{U}_i & \hat{V}_i & \hat{v}_i \end{bmatrix}.$$

After the above sequence of three rotations we obtain a new generator matrix $\hat{\tilde{\mathcal{G}}}_i$ that we partition accordingly,

$$\hat{\tilde{\mathcal{G}}}_i = \begin{bmatrix} \hat{\tilde{u}}_i & \hat{\tilde{U}}_i & \hat{\tilde{V}}_i & \hat{\tilde{v}}_i \end{bmatrix}.$$

The last $(r-1)$ columns of $\hat{\tilde{\mathcal{G}}}_i$ remain unchanged and provide the columns of the next generator matrix $\hat{\mathcal{G}}_{i+1}$, while the first column $\hat{\tilde{u}}_i$ is multiplied by $\mathcal{F}_i$ (which essentially corresponds to a simple shifting operation). Hence, we have

$$\hat{\mathcal{G}}_{i+1} = \begin{bmatrix} \hat{u}_{i+1} & \hat{U}_{i+1} & \hat{V}_{i+1} & \hat{v}_{i+1} \end{bmatrix} = \begin{bmatrix} \mathcal{F}_i \hat{\tilde{u}}_i & \hat{\tilde{U}}_i & \hat{\tilde{V}}_i & \hat{\tilde{v}}_i \end{bmatrix}.$$

The first unitary rotation $\Theta_{i,1}$ operates on $\{\hat{u}_i, \hat{U}_i\}$ and provides $\{\tilde{u}_i, \hat{\tilde{U}}_i\}$. This step guarantees the following norm relation,

$$\|\hat{\tilde{U}}_i\| \ \leq \ (1 + c_{10}\epsilon) \left(\|\hat{U}_i\| \ + \ \|\hat{u}_i\| \right).$$

But since $\hat{\tilde{U}}_i = \hat{U}_{i+1}$, we also have

$$\|\hat{U}_{i+1}\| \ \leq \ (1 + c_{10}\epsilon) \left(\|\hat{U}_i\| \ + \ \|\hat{u}_i\| \right).$$

By repeatedly applying the above inequality we obtain

$$\|\hat{U}_{i+1}\| \ \leq \ (1 + c_{11}\epsilon)^i \sum_{j=1}^{i} \|\hat{u}_j\|.$$

Consequently,

$$\| \begin{bmatrix} \hat{u}_{i+1} & \hat{U}_{i+1} \end{bmatrix} \| \leq (1 + c_{12}\epsilon)^i \sum_{j=1}^{i+1} \|\hat{u}_j\|. \tag{5.21}$$

But the $\hat{u}_i$, for $i = 2, \ldots, n+1$, are shifted versions of the (nonzero parts of the) columns of the block matrix

$$\begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix}.$$

Therefore,

$$\sum_{j=1}^{i+1} \|\hat{u}_j\| \leq n \left\| \begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix} \right\| + \|\hat{u}_1\|.$$

Now further recall that

$$S_{i+1} - \mathcal{F}_{i+1} S_{i+1} \mathcal{F}_{i+1}^T = \hat{\mathcal{G}}_{i+1} \mathcal{J} \hat{\mathcal{G}}_{i+1}^T,$$

where $\mathcal{F}_{i+1}$ is nilpotent (in fact, composed of shift matrices). It thus follows that

$$\| \begin{bmatrix} \hat{V}_{i+1} & \hat{v}_{i+1} \end{bmatrix} \|^2 \leq \| \begin{bmatrix} \hat{u}_{i+1} & \hat{U}_{i+1} \end{bmatrix} \|^2 + 2\|S_{i+1}\|. \tag{5.22}$$

Combining (5.21) and (5.22) we conclude that

$$\|\hat{\mathcal{G}}_{i+1}\|^2 \leq 8n^2(1 + c_{12}\epsilon)^{2i} \left\| \begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix} \right\|^2 + 8\|\hat{u}_1\|^2 + 4\|S_{i+1}\|. \tag{5.23}$$

We will now show that $\|S_{i+1}\|$ is bounded (at least in infinite precision).

For this purpose, we partition T into $T = \begin{bmatrix} T_1 & T_2 \end{bmatrix}$ where T_1 has i columns and T_2 has $(n-i)$ columns. Commensurately partition M as follows:

$$M = \begin{bmatrix} T_1^T T_1 & T_1^T T_2 & T_1^T \\ T_2^T T_1 & T_2^T T_2 & T_2^T \\ T_1 & T_2 & 0 \end{bmatrix}.$$

Therefore, the Schur complement S_{i+1} in infinite precision is given by

$$S_{i+1} = \begin{bmatrix} T_2^T T_2 - T_2^T T_1 (T_1^T T_1)^{-1} T_1^T T_2 & T_2^T - T_2^T T_1 (T_1^T T_1)^{-1} T_1^T \\ T_2 - T_1 (T_1^T T_1)^{-1} T_1^T T_2 & -T_1 (T_1^T T_1)^{-1} T_1^T \end{bmatrix}.$$

Let the partitioned QR factorization of T in infinite precision be

$$T = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}.$$

Then

$$T_1 (T_1^T T_1)^{-1} T_1^T = Q_1 Q_1^T,$$

which is an orthogonal projector with 2−norm equal to one. It then follows that $\|S_{i+1}\|$ is bounded as follows

$$\|S_{i+1}\| \ \leq \ 1 + 2\|T\|^2 + 2\|T\|. \tag{5.24}$$

The derivation of the above bound can be extended to finite precision by following the technique used in the next section for $\|S_{n+1}\|$. We omit the details here.

Therefore, a first-order bound for the sum of the norms of the generators in (5.20) is given by

$$\begin{aligned}\sum_{i=1}^{n} \|\mathcal{G}_i\|^2 &\leq \sum_{i=1}^{n} \left[8n^2 \left\| \begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix} \right\|^2 \ + \ 8\|\hat{u}_1\|^2 \ + \ 4\|S_{i+1}\| \right] \ + \ O(\epsilon^2) \\ &\leq 8n^3\|M\| + 8n\|M\| \ + \ 4n(1 + 2\|T\|^2 + 2\|T\|) \ + \ O(\epsilon^2) \\ &\leq 16n(1+n^2)(1 + \|T\| + \|T^2\|) \ + \ O(\epsilon^2). \end{aligned} \tag{5.25}$$

5.5. Error Analysis of the Last n Steps. It follows from (5.10), and from the definition of $E = M - \hat{M}$, that

$$M - E = \begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix} \begin{bmatrix} \hat{R} & \hat{Q}^T \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & S_{n+1} \end{bmatrix}. \tag{5.26}$$

If we partition the error matrix $-E$ into sub-blocks, say

$$-E = \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix}, \ \ E_{21} = E_{12}^T \, ,$$

and use the definition of M in (5.2), we obtain from (5.26) that

$$S_{n+1} = E_{22} - (T + E_{21})(T^T T + E_{11})^{-1}(T^T + E_{12}).$$

Therefore,

$$\begin{aligned} S_{n+1} &= E_{22} - T(T^T T + E_{11})^{-1}T^T - T(T^T T + E_{11})^{-1}E_{12} - \\ &\quad E_{21}(T^T T + E_{11})^{-1}T^T - E_{21}(T^T T + E_{11})^{-1}E_{12} \\ &= -(I + T^{-T}E_{11}T^{-1})^{-1} + \bar{E}, \end{aligned} \tag{5.27}$$

where several terms have been collected into the matrix $\bar{E}$,

$$\bar{E} = E_{22} - T(T^T T + E_{11})^{-1}E_{12} - E_{21}(T^T T + E_{11})^{-1}T^T - E_{21}(T^T T + E_{11})^{-1}E_{12}.$$

Its norm satisfies the bound

$$\|\bar{E}\| \ \leq \ \|E\| + \ \frac{2\,\|T\|\,\|E\|}{\lambda_{\min}(T^T T) - \|E\|} \ + \frac{\|E\|^2}{\lambda_{\min}(T^T T) - \|E\|} \, ,$$

and the denominator is positive in view of (5.19) and (5.20). At this stage we make the following normalization assumption:

$$\|T\| \leq \frac{1}{5} \, , \tag{5.28}$$

which can always be guaranteed by proper scaling (as explained in the statement of the algorithm in Sec. 10).

We also recall that the well-conditioned assumption (5.19), along with (5.25) and the error bound (5.20), guarantees the following condition:

$$\lambda_{\min}^{-1}(T^T T)\|E\| \leq \frac{1}{2}. \tag{5.29}$$

Remark. This essentially means that the condition number of T should be smaller than $1/\sqrt{\epsilon}$. We will relax this condition in Sec. 7.

¿From assumptions (5.28) and (5.29) we obtain $\|E\|^2 \leq \|T\| \, \|E\|$, since

$$\|E\| \leq \frac{\sigma_{min}^2(T)}{2} \leq \frac{\|T\|^2}{2} \leq \frac{1}{5}\frac{\|T\|}{2} \leq \|T\|.$$

Therefore,

$$\|\bar{E}\| \leq \|E\| + \frac{3\,\|T\|\,\|E\|}{\lambda_{\min}(T^T T) - \|E\|}.$$

Applying Corollary 8.3.2 in [10][p. 428] to expression (5.27), we get

$$\sigma_{\min}(S_{n+1}) \geq \frac{1}{1 + \lambda_{\min}^{-1}(T^T T)\|E\|} - \|\bar{E}\| \tag{5.30}$$

Using (5.28) and (5.29) we get

$$\sigma_{\min}(S_{n+1}) \geq \frac{2}{3} - \|\bar{E}\|, \tag{5.31}$$

and

$$\|\bar{E}\| \leq \frac{11}{5}\|E\| \leq \frac{11}{25}. \tag{5.32}$$

It then follows from (5.31) that

$$\sigma_{\min}(S_{n+1}) \geq \frac{17}{75} \geq \frac{1}{5}. \tag{5.33}$$

We now derive an upper bound for $\|S_{n+1}\|$. Applying Corollary 8.3.2 in [10][p. 428] to expression (5.27), and using (5.29) and (5.32), we get

$$\sigma_{\max}(S_{n+1}) \leq \frac{1}{1 - \lambda_{\min}^{-1}(T^T T)\|E\|} + \|\bar{E}\| \leq 2 + \frac{11}{25} < 3. \tag{5.34}$$

Therefore, the condition number of S_{n+1} satisfies

$$\kappa(S_{n+1}) \leq 15. \tag{5.35}$$

This establishes that S_{n+1} is a well-conditioned matrix.

By Corollary 8.3.2 in [10][p. 428], the matrix $(I + T^{-T}E_{11}T^{-1})^{-1}$ in (5.27) is positive definite since by (5.29) $1 - \lambda_{min}^{-1}(T^T T)\|E\| \geq 1/2 > 0$. Furthermore,

$$\|\bar{E}\| \leq \frac{11}{25} < \frac{1}{2} < \frac{1}{1 - \lambda_{\min}^{-1}(T^T T)\|E\|} \leq 2.$$

Therefore, applying Corollary 8.3.2 in [10][p. 428] again to expression (5.27) we conclude that S_{n+1} is negative-definite.

LEMMA 5.2. *The matrix S_{n+1} defined in (5.9) is negative-definite and well-conditioned. In particular, its condition number is at most* 15 *(cf. (5.35)).*

We can now proceed with the last n steps of the generalized Schur algorithm applied to $\hat{\mathcal{G}}_{n+1}$, since $\hat{\mathcal{G}}_{n+1}$ is a generator matrix for S_{n+1}:

$$S_{n+1} - Z_n S_{n+1} Z_n^T = \hat{\mathcal{G}}_{n+1} \mathcal{J} \hat{\mathcal{G}}_{n+1}^T.$$

All steps will now be negative steps. Hence, the discussion of Sec. 5.1 applies. The only difference will be that we make the generator proper with respect to its last column. In other words, the third step of the algorithm in Sec. 5.1 should be modified as follows:

$$g_{i,2} = \begin{bmatrix} x & 0 & 0 & 0 & x \end{bmatrix} \overset{hyperbolic\ \Theta_{i,3}}{\longrightarrow} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & x \end{bmatrix}. \tag{5.36}$$

Let $-\Delta\Delta^T$ be the computed triangular factorization of S_{n+1}. A similar error analysis to that of Sec. 5.2 (or the results of [4]) can be used to show that

$$\|S_{n+1} - (-\Delta\Delta^T)\| \leq c_{13}\epsilon \sum_{i=n+1}^{2n} \|\hat{\mathcal{G}}_i\|^2. \tag{5.37}$$

The norm of the generators $\{\hat{\mathcal{G}}_i\}$ appearing in the above error expression can be shown to be bounded as follows. Similar to equation (5.21) we have

$$\left\| \begin{bmatrix} \hat{V}_{i+1} & \hat{v}_{i+1} \end{bmatrix} \right\| \leq (1 + c_{14}\epsilon)^{i-n} \sum_{j=n+1}^{i} \|\hat{v}_j\|. \tag{5.38}$$

Moreover, the $\hat{v}_i$, for $i = n+2, \ldots, 2n$, are shifted versions of the (nonzero parts of the) columns of Δ. Hence,

$$\sum_{j=n+1}^{i} \|\hat{v}_j\| \leq n\|\Delta\| + \|\hat{v}_{n+1}\|.$$

By using the fact that Z_n is lower triangular and contractive and that S_{n+1} is negative-definite, Lemma B.2 in [4] can be extended to show that

$$\left\| \begin{bmatrix} \hat{u}_{i+1} & \hat{U}_{i+1} \end{bmatrix} \right\| \leq \left\| \begin{bmatrix} \hat{V}_{i+1} & \hat{v}_{i+1} \end{bmatrix} \right\|.$$

Therefore,

$$\begin{aligned} \|\hat{\mathcal{G}}_i\| &\leq \left\| \begin{bmatrix} \hat{u}_{i+1} & \hat{U}_{i+1} \end{bmatrix} \right\| + \left\| \begin{bmatrix} \hat{V}_{i+1} & \hat{v}_{i+1} \end{bmatrix} \right\| \\ &\leq 2 \left\| \begin{bmatrix} \hat{V}_{i+1} & \hat{v}_{i+1} \end{bmatrix} \right\| \\ &\leq 2(1 + c_{14}\epsilon)^{i-n} [n\|\Delta\| + \|\hat{v}_{n+1}\|] , \end{aligned} \tag{5.39}$$

where in infinite precision,

$$\|\Delta\|^2 = \|S_{n+1}\| \leq 3,$$

from relation (5.34). Similarly, the bound for $\hat{v}_{n+1}$ follows from (5.23) and (5.24).

Summary. We have shown so far that if we apply $2n$ steps of the generalized Schur algorithm to the matrices $(\mathcal{F}, \mathcal{G})$ in (5.3), with proper implementation of the $\mathcal{J}$-unitary rotations (as explained in Sec. 5.1), then the error in the computed factorization of M is bounded as follows:

$$\left\| M - \begin{bmatrix} \hat{R}^T & 0 \\ \hat{Q} & \Delta \end{bmatrix} \begin{bmatrix} \hat{R} & \hat{Q}^T \\ 0 & -\Delta^T \end{bmatrix} \right\| \leq c_{15}\epsilon \sum_{i=1}^{2n} \|\hat{\mathcal{G}}_i\|^2. \tag{5.40}$$

We have also established (at least in infinite precision) that the norm of the generators is bounded. Therefore, the computed factorization is (at least asymptotically) backward stable with respect to M.

6. Solving Linear Systems. We now return to the problem of solving the linear system of equations $Tx = b$, where T is a well-conditioned nonsymmetric shift structured matrix (e.g., Toeplitz, quasi-Toeplitz, product of two Toeplitz matrices).

Note from the bound (5.40) that

$$\|\hat{Q}\hat{Q}^T - \Delta\Delta^T\| \leq c_{15}\epsilon \sum_{i=1}^{2n} \|\hat{\mathcal{G}}_i\|^2.$$

Therefore,

$$\|(\Delta^{-1}\hat{Q})(\Delta^{-1}\hat{Q})^T - I\| \leq c_{15}\epsilon \|\Delta^{-1}\|^2 \sum_{i=1}^{2n} \|\hat{\mathcal{G}}_i\|^2.$$

It follows from (5.33) and (5.35) that

$$\sigma_{min}(\Delta\Delta^T) \geq \sigma_{min}(S_{n+1}) - c_{15}\epsilon \sum_{i=n+1}^{2n} \|\hat{\mathcal{G}}_i\|^2 \geq \frac{1}{5} - c_{15}\epsilon \sum_{i=n+1}^{2n} \|\hat{\mathcal{G}}_i\|^2 \approx \frac{1}{5}.$$

Therefore, $\|\Delta^{-1}\|^2$ is bounded by $1/5$ (approximately), from which we can conclude that $\Delta^{-1}\hat{Q}$ is numerically orthogonal.

Furthermore from (5.40) we also have

$$\|T - \hat{Q}\hat{R}\| \leq c_{15}\epsilon \sum_{i=1}^{2n} \|\hat{\mathcal{G}}_i\|^2.$$

This shows that we can compute x by solving the nearby linear system

$$\Delta\Delta^{-1}\hat{Q}\hat{R}x = b\ ,$$

in $O(n^2)$ flops by exploiting the fact that $\Delta^{-1}\hat{Q}$ is numerically orthogonal and Δ is triangular as follows:

$$\hat{x} \leftarrow \hat{R}^{-1}(\hat{Q}^T\Delta^{-T})\Delta^{-1}b\ . \tag{6.1}$$

The fact that this scheme for computing x is backward stable will be established in Section 8 (see remark after expression (8.2)).

7. Ill-conditioned T. We now consider modifications to the algorithm when the inequality (5.29) is not satisfied by T. This essentially means that the condition number of T is larger than the square root of the reciprocal of the machine precision. We will refer to such matrices T as being ill-conditioned.

There are several potential numerical problems now, all of which have to be eliminated. First, the $(1,1)$-block of M can fail to factorize as it is not sufficiently positive-definite. Second, even if the first n steps of the Schur algorithm are completed successfully, the Schur complement S_{n+1} of the $(2,2)$-block may no longer be negative-definite making the algorithm unstable. Third, the matrix Δ may no longer be well-conditioned, in which case it is not clear how one can solve the linear system $Tx = b$ in a stable manner. We now show how these problems can be resolved.

To resolve the first two problems we add small multiples of the identity matrix to the $(1,1)$ and $(2,2)$ blocks of M, separately:

$$M = \begin{bmatrix} T^T T + \alpha I & T \\ T^T & -\beta I \end{bmatrix} , \tag{7.1}$$

where α and β are positive numbers that will be specified later.[1] This leads to an increase in the displacement rank of M. For Toeplitz matrices the rank increases only by one and the new generators are given as follows:

$$M - (Z_n \oplus Z_n) M (Z_n \oplus Z_n)^T = \mathcal{G} \mathcal{J} \mathcal{G}^T , \tag{7.2}$$

where $\mathcal{J}$ is 6×6,

$$\mathcal{J} = \mathrm{diag}[1, 1, 1, -1, -1, -1], \tag{7.3}$$

and $\mathcal{G}$ is $2n \times 6$,

$$\mathcal{G} = \begin{bmatrix} \sqrt{\alpha} & s_0 & 0 & 0 & 0 & 0 \\ 0 & s_1 & t_{-1} & s_1 & t_{n-1} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & s_{n-1} & t_{-n+1} & s_{n-1} & t_1 & 0 \\ 0 & c_0 & 1 & c_0 & 0 & \sqrt{1+\beta} \\ 0 & c_1 & 0 & c_1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ 0 & c_{n-1} & 0 & c_{n-1} & 0 & 0 \end{bmatrix} . \tag{7.4}$$

Had we instead started with the embedding (4.2) for more general shift structured matrices, we would then modify the generators as explained later in the remark in Sec. 9.

Assume α is chosen such that

$$\alpha \ \geq \ c_{16} \epsilon \sum_{j=1}^{n} \|\hat{\mathcal{G}}_j\|^2 , \tag{7.5}$$

then since

$$\lambda_{\min}(T^T T + \alpha I) \ > \ c_{16} \epsilon \sum_{j=1}^{n} \|\hat{\mathcal{G}}_j\|^2 , \tag{7.6}$$

[1] We continue to use M for the new matrix in (7.1) for convenience of notation.

it follows from the analysis in Sec. 5.3 that the first n steps of the generalized Schur algorithm applied to $\mathcal{G}$ in (7.4) will complete successfully. As in (5.10), define the matrix

$$\hat{M} = \begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix} \begin{bmatrix} \hat{R} & \hat{Q}^T \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & S_{n+1} \end{bmatrix}, \tag{7.7}$$

where S_{n+1} is the solution of

$$S_{n+1} - Z_n S_{n+1} Z_n^T = \hat{\mathcal{G}}_{n+1} \mathcal{J} \hat{\mathcal{G}}_{n+1}.$$

(Recall that $\hat{\mathcal{G}}_{n+1}$ has now six columns and $\mathcal{J}$ is 6×6). Then following the analysis of the first n steps of Sec. 5.2 we obtain (cf. (5.20))

$$\|E\| = \left\| M - \hat{M} \right\| \leq c_{19} \epsilon \sum_{j=1}^{n} \|\hat{\mathcal{G}}_j\|^2 ,$$

where, as shown earlier in (5.23),

$$\|\hat{\mathcal{G}}_{i+1}\|^2 \leq 8n^2(1 + c_{16}\epsilon)^{2i} \left\| \begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix} \right\|^2 + 8\|\hat{u}_1\|^2 + 4\|S_{i+1}\|^2 . \tag{7.8}$$

The proof that S_{i+1} is bounded is similar to the proof that S_{n+1} is bounded, which we now give. First we assume that β satisfies the following bound,

$$\beta \geq \frac{1 + c_{16}\epsilon}{1 - c_{16}\epsilon}(\|E\| + 4) . \tag{7.9}$$

Recall that S_{n+1} satisfies the relation

$$M - E = \begin{bmatrix} \hat{R}^T \\ \hat{Q} \end{bmatrix} \begin{bmatrix} \hat{R} & \hat{Q}^T \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & S_{n+1} \end{bmatrix}. \tag{7.10}$$

If we partition the error matrix $-E$ into sub-blocks, say

$$-E = \begin{bmatrix} E_{11} & E_{12}^T \\ E_{12} & E_{22} \end{bmatrix},$$

and use the definition of M in (7.1), we obtain from (7.10) that

$$S_{n+1} = -\beta I + E_{22} - (T + E_{12})(T^T T + \alpha I + E_{11})^{-1}(T^T + E_{12}^T). \tag{7.11}$$

Since α and β satisfy (7.5) and (7.9), we have that

$$\alpha \geq \|E\| \geq \|E_{11}\| , \quad \beta \geq \frac{1 + c_{16}\epsilon}{1 - c_{16}\epsilon}(\|E\| + 4) \geq \|E\| \geq \|E_{22}\|.$$

Therefore, $(\alpha I + E_{11})$ is positive-definite and $(-\beta I + E_{22})$ is negative-definite. This shows, in view of (7.11), that S_{n+1} is negative-definite. We now proceed to bound the smallest and the largest eigenvalues of S_{n+1}.

Using (7.11) we write

$$S_{n+1} = -\beta I + E_{22} - (I + E_{12}T^{-1})(I + \alpha T^{-T}T^{-1} + T^{-T}E_{11}T^{-1})^{-1}(I + T^{-T}E_{12}^T) ,$$

and note that

$$\|(I + \alpha T^{-T}T^{-1} + T^{-T}E_{11}T^{-1})^{-1}\| = \left\| \left(I + \alpha T^{-T} \left[I + \frac{E_{11}}{\alpha} \right] T^{-1} \right)^{-1} \right\| \leq 1 ,$$

since $\|E_{11}\|/\alpha < 1$.

We now make the assumption

$$\|T^{-1}\| \, \|E\| \leq 1 , \tag{7.12}$$

which is considerably weaker than the assumption (5.29) used in the well-conditioned case. Assumption (7.12) essentially means that the condition number of T should be less than the reciprocal of the machine precision.

It then follows that

$$\|S_{n+1}\| \leq \beta + \|E\| + 4 .$$

Since technically, $\|E\|$ depends upon $\|S_{n+1}\|$, we have only shown that $\|S_{n+1}\|$ is bounded to first order in ϵ. With more effort, this restriction can be removed.

Before proceeding, we mention that the error in factorizing S_{n+1} into $-\Delta\Delta^T$ by the generalized Schur algorithm can be written in the form

$$\|S_{n+1} - (-\Delta\Delta^T)\| \leq c_{17}\epsilon\|S_{n+1}\| ,$$

where c_{17} can be obtained by extending the analysis of Sec. 5.2.

As mentioned earlier (cf. (5.18)), S_{n+1} can be factorized by the Schur algorithm if its minimum eigenvalue satisfies

$$|\lambda_{min}(S_{n+1})| \geq c_{17}\epsilon\|S_{n+1}\| .$$

But since $|\lambda_{min}(S_{n+1})| \geq \beta - \|E_{22}\|$, the above condition can be guaranteed by choosing

$$\begin{aligned} \beta &\geq c_{17}\epsilon\|S_{n+1}\| + \|E_{22}\| \\ &\geq c_{17}\epsilon(\beta + \|E\| + 4) + \|E\| \\ &\geq \frac{1}{1 - c_{17}\epsilon} [c_{17}\epsilon(\|E\| + 4) + \|E\|] \\ &\geq \frac{1 + c_{17}\epsilon}{1 - c_{17}\epsilon}(\|E\| + 4) , \end{aligned}$$

which is assumption (7.9) on β (with $c_{17} = c_{16}$).

Therefore, the last n steps of the generalized Schur algorithm can be completed to give the following error bound in the factorization of M in (7.1):

$$\left\| M - \begin{bmatrix} \hat{R}^T & \mathbf{0} \\ \hat{Q} & \Delta \end{bmatrix} \begin{bmatrix} \hat{R} & \hat{Q}^T \\ \mathbf{0} & -\Delta^T \end{bmatrix} \right\| \leq \alpha + \beta + c_{18}\epsilon \sum_{i=1}^{2n} \|\hat{\mathcal{G}}_i\|^2 , \tag{7.13}$$

where the norm of the generators is again bounded by arguments similar to those in Sec. 5.4. In other words, we have a backward stable factorization of M.

Since Δ is no longer provably well-conditioned, we can not argue that $\Delta^{-1}\hat{Q}$ is numerically orthogonal. For this reason, we now discuss how to solve the linear system of equations $Tx = b$ in the ill-conditioned case.

8. Solving the Linear System. Note that if x solves $Tx = b$ then it also satisfies

$$\begin{bmatrix} T^T T & T^T \\ T & \mathbf{0} \end{bmatrix} \begin{bmatrix} x \\ -b \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ b \end{bmatrix} .$$

Using the above backward stable factorization (7.13) we can solve the above linear system of equations to get

$$\left(\begin{bmatrix} T^T T & T^T \\ T & \mathbf{0} \end{bmatrix} + H \right) \begin{bmatrix} \hat{y} \\ \hat{z} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ b \end{bmatrix} , \tag{8.1}$$

where the error matrix H satisfies

$$\|H\| \leq \alpha + \beta + c_{18}\epsilon \sum_{i=1}^{2n} \|\hat{\mathcal{G}}_i\|^2 + c_{19}\epsilon \left\| \begin{bmatrix} \hat{R}^T & \mathbf{0} \\ \hat{Q} & \Delta \end{bmatrix} \right\|^2 .$$

Note that $\hat{y}$ is computed by the expression

$$R^{-1}\hat{Q}^T \Delta^{-T} \Delta^{-1} b , \tag{8.2}$$

which is identical to the earlier formula (6.1) we obtained by assuming $\Delta^{-1}\hat{Q}$ is numerically orthogonal! Therefore, the subsequent error analysis holds equally well for the well-conditioned case.

Moreover, it follows from (8.1) that

$$(T + H_{21})\hat{y} + H_{22}\hat{z} = b.$$

Therefore, we can write this as

$$\left(T + H_{21} + \frac{H_{22}\hat{z}\hat{y}^T}{\hat{y}^T\hat{y}} \right) \hat{y} = b , \tag{8.3}$$

where

$$\left\| H_{21} + \frac{H_{22}\hat{z}\hat{y}^T}{\hat{y}^T\hat{y}} \right\| \leq \|H_{21}\| + \|H_{22}\| \frac{\|\hat{z}\|}{\|\hat{y}\|}. \tag{8.4}$$

If we assume

$$\|T\| \leq 1 \tag{8.5}$$

(which is implied by (5.28)), then in infinite precision

$$\frac{\|\hat{z}\|}{\|\hat{y}\|} = \frac{\|b\|}{\|x\|} = \frac{\|b\|}{\|T^{-1}b\|} \leq \frac{\|b\|}{\|b\|} \|T\| \leq 1.$$

Under the assumptions in Theorem C.1, which are of a similar nature to assumptions we have already made, we can show that $\|\hat{z}\|/\|\hat{y}\|$ is also bounded in finite precision. Therefore, our algorithm is backward stable for solving shift structured linear systems.

Theorem C.1 imposes a bound on $\kappa(M)$, the condition number of M. We now verify that $\kappa(M)$ is of the same order as $\kappa(T)$. First note that

$$\|M\| \leq 2\|T\| + \|T\|^2 \leq 3\|T\| ,$$

since $\|T\| \leq 1$. Moreover,

$$M^{-1} = \begin{bmatrix} \mathbf{0} & T^{-1} \\ T^{-T} & -I \end{bmatrix},$$

from which we conclude that

$$\|M^{-1}\| \ \leq \ 1 + 2\|T^{-1}\|.$$

Hence,

$$\kappa(M) \ \leq \ (1 + 2\|T^{-1}\|)(3\|T\|) \ \leq \ 9\kappa(T).$$

Therefore, the restriction on $\kappa(M)$ can be considered a restriction on $\kappa(T)$, which will be similar to our earlier assumption (7.12).

For convenience we now give a simple first-order bound for the backward error in (8.3). Indeed,

$$\begin{aligned}
\left\| H_{21} + \frac{H_{22}\hat{z}\hat{y}^T}{\hat{y}^T\hat{y}} \right\| &\leq \ \|H_{21}\| \ + \ \|H_{22}\| \ + \ O(\epsilon^2) \\
&\leq 2\|H\| \ + \ O(\epsilon^2) \\
&\leq 2\left[\alpha + \beta + c_{20}\epsilon \sum_{i=1}^{2n} \|\hat{\mathcal{G}}_i\|^2 \ + \ c_{21}\epsilon \left\| \begin{bmatrix} \hat{R}^T & \mathbf{0} \\ \hat{Q} & \Delta \end{bmatrix} \right\|^2 \right] \ + \ O(\epsilon^2) \\
&\leq 2(\alpha + \beta) + \\
&\quad c_{22}\epsilon \left[\|M\| \ + \ \sum_{i=1}^{2n} \left(8n^2\|M\| + 4(1 + 2\|T\|^2 + 2\|T\|)\right) \right] \ + \ O(\epsilon^2) \\
&\leq 2(\alpha + \beta) + c_{23}\epsilon \left[\|M\| \ + \ 4n(1 + 2\|T\|^2 + 2\|T\|) \right] \ + \ O(\epsilon^2) \\
&\leq 2(\alpha + \beta) + c_{24}\epsilon[\|M\| + 1] \ + \ O(\epsilon^2) \\
&\leq 2(\alpha + \beta) + c_{25}\epsilon[\|T\| + 1] \ + \ O(\epsilon^2). \qquad (8.6)
\end{aligned}$$

Note that $\|T\|$ should be approximately one for the algorithm to be backward stable. This can be satisfied by appropriately normalizing $\|T\|$.

8.1. Conditions on the Coefficient Matrix. For ease of reference, we list here the conditions imposed on the coefficient matrix T in order to guarantee a fast backward stable solver of $Tx = b$:

1. $\|T\|$ is suitably normalized to guarantee $\|T\| \approx 1$ (cf. (5.28) and (8.5)).
2. $\|T^{-1}\|$ satisfies (7.12), which essentially means that the condition number of T should be less than the reciprocal of the machine precision.

9. A Remark. Had we instead started with the embedding (4.2), we first perform n steps of the generalized Schur algorithm to get a generator matrix $\hat{\mathcal{G}}_{n+1}$ for the computed version of the $2n \times 2n$ embedding (4.6). We then add two columns to $\hat{\mathcal{G}}_{n+1}$ as follows:

$$\begin{bmatrix} \sqrt{\alpha} & & 0 \\ 0 & & 0 \\ 0 & & \sqrt{\beta} \\ \vdots & \hat{\mathcal{G}}_{n+1} & \vdots \\ 0 & & 0 \\ 0 & & 0 \end{bmatrix},$$

where the entry $\sqrt{\beta}$ occurs in the $(n+1)-th$ row of the last column. The new first column has a positive signature and the new last column has a negative signature.

10. Pseudo-Code of the Algorithm for Toeplitz Systems. For convenience we summarize the algorithm here for the case of nonsymmetric Toeplitz systems. We hasten to add though that the algorithm also applies to more general shift structured matrices T (such as quasi-Toeplitz or with higher displacement ranks, as demonstrated by the analysis in the earlier sections). The only difference will be in the initial generator matrix $\mathcal{G}$ and signature matrix $\mathcal{J}$ for M in (7.1) and (7.2). The algorithm will also be essentially the same, apart from an additional n Schur steps, if we instead employ the embedding (4.2).

Input: A nonsymmetric $n \times n$ Toeplitz matrix T and an $n-$dimensional column vector b. The entries of the first column of T are denoted by $[t_0, t_1, \ldots, t_{n-1}]^T$, while the entries of the first row of T are denoted by $[t_0, t_{-1}, \ldots, t_{-n+1}]$.

Output: A backward stable solution of $Tx = b$.

Algorithm:

- Normalize T and b. Since the Frobenius norm of $\|T\|$ is less than

$$\gamma \;=\; \sqrt{n \sum_{i=-n+1}^{n-1} t_i^2}\;,$$

  we can normalize T by setting t_i to be $t_i/(5\gamma)$ for all i. Similarly, divide the entries of b by 5γ. In the sequel, T and b will refer to these normalized quantities.

- Define the vectors

$$\begin{bmatrix} c_0 \\ \vdots \\ c_{n-1} \end{bmatrix} = \frac{Te_1}{\|Te_1\|}\;, \qquad \begin{bmatrix} s_0 \\ \vdots \\ s_{n-1} \end{bmatrix} = T^T \begin{bmatrix} c_0 \\ \vdots \\ c_{n-1} \end{bmatrix}.$$

- Construct the 6×6 signature matrix

$$\mathcal{J} = \mathrm{diag}[1, 1, 1, -1, -1, -1],$$

  and the $2n \times 6$ generator matrix $\mathcal{G}$,

$$\mathcal{G} = \begin{bmatrix} \sqrt{\alpha} & s_0 & 0 & 0 & 0 & 0 \\ 0 & s_1 & t_{-1} & s_1 & t_{n-1} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & s_{n-1} & t_{-n+1} & s_{n-1} & t_1 & 0 \\ 0 & c_0 & 1 & c_0 & 0 & \sqrt{1+\beta} \\ 0 & c_1 & 0 & c_1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ 0 & c_{n-1} & 0 & c_{n-1} & 0 & 0 \end{bmatrix},$$

  where the small positive numbers α and β are chosen as follows (by experimental tuning):

$$\alpha = n^{1/2}\epsilon\,\|\mathcal{G}\|^2 \quad , \quad \beta = 4(2n)^{1/4}\epsilon.$$

(If T is well-conditioned then we set $\beta = 0 = \alpha$, delete the first columns of $\mathcal{G}$ and $\mathcal{J}$, which then become $2n \times 5$ and 5×5, respectively).

- Apply n steps of the generalized Schur algorithm starting with $\mathcal{G}_1 = \mathcal{G}$ and $\mathcal{F} = (Z_n \oplus Z_n)$, and ending with $\mathcal{G}_{n+1}$ and $\mathcal{F} = Z_n$. These are positive steps according to the description of Algorithm 3.1 (step 2), where the successive generators are reduced to proper form relative to their first column. Note that this must be performed with care for numerical stability as explained in Sec. 5.1.
- Apply n more steps of the generalized Schur algorithm starting with $\mathcal{G}_{n+1}$. These are negative steps according to the description of Algorithm 3.1 (step 3), where the successive generators are reduced to proper form relative to their last column. This has also to be performed with care as explained prior to equation (5.36).
- Each of the above $2n$ steps provides a column of the triangular factorization of the matrix M in (7.1), as described in Algorithm 3.1 (steps 2 and 3). The triangular factor of M is then partitioned to yield the matrices $\{\hat{R}, \hat{Q}, \Delta\}$,

$$\begin{bmatrix} \hat{R}^T & \mathbf{0} \\ \hat{Q} & \Delta \end{bmatrix},$$

  where $\hat{R}$ is upper triangular and Δ is lower triangular.
- The solution $\hat{x}$ is obtained by evaluating the quantity

$$R^{-1}\hat{Q}^T\Delta^{-T}\Delta^{-1}b\,,$$

  via a sequence of back-substitutions and matrix-vector multiplications. The computed solution is backward stable. It satisfies

$$(T + H)\hat{x} = b\,,$$

  where the norm of the error matrix is bounded by

$$(10.1)\quad \|H\| \;\leq\; 2(\alpha+\beta) \;+ c_{26}\epsilon[1 + \|T\|] + O(\epsilon^2) \;\leq c_{27}\epsilon\,\|T\| + O(\epsilon^2)\,.$$

10.1. Operation Count. The major computational cost is due to the application of the successive steps of the generalized Schur algorithm. The overhead operations that are required for the normalization of T, and for the determination of the generator matrix $\mathcal{G}$, amount at most to $O(n \log n)$ flops. Table 10.1 shows the number of flops needed at each step of the algorithm [i denotes the iteration number and it runs from $i = 2n$ down to $i = 1$]. The operation count given below assumes that, for each iteration, two Householder transformations are used to implement the reduction to proper form of Sec. 5.1, combined with an elementary hyperbolic rotation in OD form.

Table 10.2 indicates the specific costs for different classes of structured matrices.

11. Conclusions. We performed extensive experiments to verify the theoretical bounds for both well-conditioned and ill-conditioned Toeplitz matrices. The error was always better than the bounds predicted by the theory. Interested readers can get Matlab codes of the algorithm by contacting the authors.

The results of this work can be extended to Toeplitz least-squares problems, which will be addressed in a companion paper. Furthermore, there are also useful applications of these ideas in filtering theory, which will be reported elsewhere.

TABLE 10.1
Complexity analysis of the algorithm.

During each iteration of the algorithm	Count in flops
Compute two Householder transformations	$3r$
Apply the Householder transformations	$4 \cdot i \cdot r$
Compute the hyperbolic transformation	7
Apply the hyperbolic transformation using OD	$6 \cdot i$
Shift columns	i
Total for $i = 2n$ down to 1	$(14 + 8r)n^2 + 10nr + 21n$
Cost of 3 back-substitution steps	$3n^2$
Cost of matrix-vector multiplication	$2n^2$
Startup costs	$n(24 \log n + r + 52)$
Total cost of the algorithm	$(19 + 8r)n^2 + n(24 \log n + 11r + 73)$

TABLE 10.2
Computational cost for some structured matrices.

Matrix type	Cost
Well-conditioned Toeplitz matrix	$59n^2 + n(24 \log n + 128)$
Ill-conditioned Toeplitz matrix	$67n^2 + n(24 \log n + 139)$

APPENDICES

Appendix A. The OD Procedure. Let $\rho = \beta/\alpha$ be the reflection coefficient of a hyperbolic rotation Θ,

$$\Theta = \frac{1}{\sqrt{1-\rho^2}} \begin{bmatrix} 1 & -\rho \\ -\rho & 1 \end{bmatrix},$$

with $|\rho| < 1$. Let $\begin{bmatrix} x_1 & y_1 \end{bmatrix}$ and $\begin{bmatrix} x & y \end{bmatrix}$ be the postarray and prearray rows, respectively,

$$\begin{bmatrix} x_1 & y_1 \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \Theta.$$

The advantage of the OD-method is that the computed quantities $\hat{x}_1$ and $\hat{y}_1$ satisfy the equation

$$\begin{bmatrix} \hat{x}_1 + e_1 & \hat{y}_1 + e_2 \end{bmatrix} = \begin{bmatrix} x + e_3 & y + e_4 \end{bmatrix} \Theta, \tag{A.1}$$

with

$$\left\| \begin{bmatrix} e_1 & e_2 \end{bmatrix} \right\| \leq c_{28}\epsilon \left\| \begin{bmatrix} \hat{x}_1 & \hat{y}_1 \end{bmatrix} \right\|, \quad \left\| \begin{bmatrix} e_3 & e_4 \end{bmatrix} \right\| \leq c_{29}\epsilon \left\| \begin{bmatrix} x & y \end{bmatrix} \right\|, \tag{A.2}$$

and, consequently,

$$|(\hat{x}_1^2 - \hat{y}_1^2) - (x_1^2 - y_1^2)| \leq c_{30}\epsilon(\hat{x}_1^2 + \hat{y}_1^2 + x^2 + y^2). \tag{A.3}$$

ALGORITHM A.1 (The OD Procedure). *Consider a hyperbolic rotation Θ with reflection coefficient $\rho = \beta/\alpha$, $|\rho| < 1$. Given a row vector $\begin{bmatrix} x & y \end{bmatrix}$ as a prearray, the transformed (postarray) row vector $\begin{bmatrix} x_1 & y_1 \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \Theta$ is computed as follows:*

$$\begin{bmatrix} x' & y' \end{bmatrix} \leftarrow \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} x'' & y'' \end{bmatrix} \leftarrow \begin{bmatrix} x' & y' \end{bmatrix} \begin{bmatrix} \frac{1}{2}\sqrt{\frac{\alpha+\beta}{\alpha-\beta}} & 0 \\ 0 & \frac{1}{2}\sqrt{\frac{\alpha-\beta}{\alpha+\beta}} \end{bmatrix}$$

$$\begin{bmatrix} x_1 & y_1 \end{bmatrix} \leftarrow \begin{bmatrix} x'' & y'' \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

Appendix B. The H Procedure. Let $\rho = \beta/\alpha$ be the reflection coefficient of a hyperbolic rotation Θ,

$$\Theta = \frac{1}{\sqrt{1-\rho^2}} \begin{bmatrix} 1 & -\rho \\ -\rho & 1 \end{bmatrix},$$

with $|\rho| < 1$. Let $\begin{bmatrix} x_1 & y_1 \end{bmatrix}$ and $\begin{bmatrix} x & y \end{bmatrix}$ be the postarray and prearray rows, respectively,

$$\begin{bmatrix} x_1 & y_1 \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \Theta, \quad \text{with} \quad |x| > |y|.$$

The advantage of the H-method is that the computed quantities $\hat{x}_1$ and $\hat{y}_1$ satisfy the equation

$$\begin{bmatrix} \hat{x}_1 + e_1' & \hat{y}_1 + e_2' \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \Theta, \tag{B.1}$$

where the error terms satisfy

$$|e_1'| \le c_{31}\epsilon|\hat{x}_1|, \quad |e_2'| \le c_{32}\epsilon(|\hat{x}_1| + |\hat{y}_1|). \tag{B.2}$$

If $|x| < |y|$, then it can be seen that $\begin{bmatrix} y & x \end{bmatrix} \Theta = \begin{bmatrix} y_1 & x_1 \end{bmatrix}$. Therefore, without loss of generality, we shall only consider the case $|x| > |y|$.

ALGORITHM B.1 (The H Procedure). *Given a hyperbolic rotation Θ with reflection coefficient $\rho = \beta/\alpha$, $|\rho| < 1$, and a prearray $\begin{bmatrix} x & y \end{bmatrix}$ with $|x| > |y|$, the postarray $\begin{bmatrix} x_1 & y_1 \end{bmatrix}$ can be computed as follows:*

If $\frac{\beta}{\alpha}\frac{y}{x} < 1/2$
  then $\xi \leftarrow 1 - \frac{\beta}{\alpha}\frac{y}{x}$
else
  $d_1 \leftarrow \frac{|\alpha|-|\beta|}{|\alpha|}$, $d_2 \leftarrow \frac{|x|-|y|}{|x|}$
  $\xi \leftarrow d_1 + d_2 - d_1 d_2$
endif
$x_1 \leftarrow \frac{|\alpha| x \xi}{\sqrt{(\alpha-\beta)(\alpha+\beta)}}$
$y_1 \leftarrow x_1 - \sqrt{\frac{\alpha+\beta}{\alpha-\beta}}\,(x-y)$.

The H procedure requires $5n$ to $7n$ multiplications and $3n$ to $5n$ additions. It is therefore costlier than the OD procedure, which requires $2n$ multiplications and $4n$ additions. But the H procedure is forward stable (cf. (B.1)) whereas the OD method is only stable (cf. (A.1)).

Appendix C. Miscellaneous Error Bounds. The following is an extension of Lemma 2.7.1 and Theorem 2.7.2 of [10, p. 82].

THEOREM C.1. *Suppose*

$$M \begin{bmatrix} y \\ z \end{bmatrix} = b$$

where M is an $n \times n$ matrix, b is an $n-$dimensional vector, and $\|z\| \leq \|y\|$. Let

$$(M + H) \begin{bmatrix} \hat{y} \\ \hat{z} \end{bmatrix} = b \, ,$$

where H is an $n \times n$ matrix such that $\|H\| \leq c_{33}\epsilon\|M\|$. If $c_{33}\epsilon\kappa(M) = r < \frac{1}{5}$, where $\kappa(M) = \|M\| \, \|M^{-1}\|$, then

$$\frac{\|\hat{z}\|}{\|\hat{y}\|} \leq \frac{1+3r}{1-5r} \, .$$

Proof. From Theorem 2.7.2 in [10] it follows that

$$\|y\| - \frac{2r}{1-r}[\|y\| + \|z\|] \leq \|\hat{y}\| \leq \|y\| + \frac{2r}{1-r}[\|y\| + \|z\|] \, .$$

By interchanging y and z we can obtain a similar inequality for $\hat{z}$. Then

$$\begin{aligned} \frac{\|\hat{z}\|}{\|\hat{y}\|} &\leq \frac{\|z\| + \frac{2r}{1-r}[\|y\| + \|z\|]}{\|y\| - \frac{2r}{1-r}[\|y\| + \|z\|]} \\ &\leq \frac{1+3r}{1-5r} \, , \end{aligned}$$

since $\|z\| \leq \|y\|$.
□

REFERENCES

[1] A.W.BOJANCZYK, R.P.BRENT, AND F. DE HOOG, *QR factorization of Toeplitz matrices*, Numerische Mathematik, 49 (1986), pp. 81–94.

[2] A. W. BOJANCZYK, R. P. BRENT, F. R. DE HOOG, AND D. R. SWEET, *On the stability of the Bareiss and related Toeplitz factorization algorithms*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 40–57.

[3] A. W. BOJANCZYK, R. P. BRENT, P. VAN DOOREN, AND F. R. DE HOOG, *A note on downdating the Cholesky factorization*, SIAM J. Sci. Stat. Comput., 8 (1987), pp. 210–221.

[4] S. CHANDRASEKARAN AND A. H. SAYED, *Stabilizing the generalized Schur algorithm*, SIAM J. Matrix Analysis and Applications, 17 (1996), pp. 950–983.

[5] J. CHUN, *Fast Array Algorithms for Structured Matrices*, PhD thesis, Stanford University, Stanford, CA, 1989.

[6] J. CHUN, T. KAILATH, AND H. LEV-ARI, *Fast parallel algorithms for QR and triangular factorization*, SIAM J. Sci. Stat. Comput., 8 (1987), pp. 899–913.

[7] G. CYBENKO, *A general orthogonalization technique with applications to time series analysis and signal processing*, Math. Comp., 40 (1983), pp. 323–336.

[8] ———, *Fast Toeplitz orthogonalization using inner products*, SIAM J. Scientific Comp., (1987), pp. 734–740.

[9] I. GOHBERG, T. KAILATH, AND V. OLSHEVSKY, *Fast Gaussian elimination with partial pivoting for matrices with displacement structure.* submitted for publication.

[10] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, second ed., 1989.

[11] M. GU, *Stable and efficient algorithms for structured systems of linear equations*, Technical Report LBL-37690, Lawrence Berkeley Laboratory, University of California, Berkeley, (1995).

[12] G. HEINIG, *Inversion of generalized Cauchy matrices and other classes of structured matrices*, 69 (1995), pp. 63–81. Linear Algebra for Signal Processing, eds. A. Bojanczyk and G. Cybenko.

[13] T. KAILATH AND J. CHUN, *Generalized displacement structure for block-Toeplitz, Toeplitz-block, and Toeplitz-derived matrices*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 114–128.

[14] T. KAILATH, S. Y. KUNG, AND M. MORF, *Displacement ranks of a matrix*, Bulletin of the American Mathematical Society, 1 (1979), pp. 769–773.

[15] T. KAILATH AND A. H. SAYED, *Displacement structure: Theory and applications*, SIAM Review, 37 (1995), pp. 297–386.

[16] A. H. SAYED, *Displacement Structure in Signal Processing and Mathematics*, PhD thesis, Stanford University, Stanford, CA, August 1992.

[17] A. H. SAYED AND T. KAILATH, *A look-ahead block Schur algorithm for Toeplitz-like matrices*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 388–413.

[18] M. STEWART AND P. VAN DOOREN, *Stability issues in the factorization of structured matrices*, SIAM J. Matrix Analysis and Appl., (1996). To appear.

[19] D. R. SWEET, *Fast Toeplitz orthogonalization*, Numerische Mathematik, 43 (1984), pp. 1–21.