SIAM J. MATRIX ANAL. APPL.
Vol. 17, No. 4, pp. 950–983, October 1996

014

STABILIZING THE GENERALIZED SCHUR ALGORITHM*

S. CHANDRASEKARAN† AND ALI H. SAYED†

Abstract. This paper provides a detailed analysis that shows how to stabilize the *generalized* Schur algorithm, which is a fast procedure for the Cholesky factorization of positive-definite structured matrices R that satisfy displacement equations of the form $R - FRF^T = GJG^T$, where J is a 2×2 signature matrix, F is a stable lower-triangular matrix, and G is a generator matrix. In particular, two new schemes for carrying out the required hyperbolic rotations are introduced and special care is taken to ensure that the entries of a Blaschke matrix are computed to high relative accuracy. Also, a condition on the smallest eigenvalue of the matrix, along with several computational enhancements, is introduced in order to avoid possible breakdowns of the algorithm by assuring the positive-definiteness of the successive Schur complements. We use a perturbation analysis to indicate the best accuracy that can be expected from *any* finite-precision algorithm that uses the generator matrix as the input data. We then show that the modified Schur algorithm proposed in this work essentially achieves this bound when coupled with a scheme to control the generator growth. The analysis further clarifies when pivoting strategies may be helpful and includes illustrative numerical examples. For all practical purposes, the major conclusion of the analysis is that the modified Schur algorithm is backward stable for a large class of structured matrices.

Key words. displacement structure, generalized Schur algorithm, Cholesky factorization, hyperbolic rotations, generator matrices, pivoting, Schur functions, error analysis

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

1. Introduction.

We show how to stabilize the generalized Schur algorithm and give a finite-precision error analysis to support our conclusions. The notion of structured matrices, along with the algorithm itself, is reviewed in the next two sections. Here we proceed with a general overview of earlier relevant work in the literature.

One of the most frequent structures, at least in signal processing applications, is the Toeplitz structure, with constant entries along the diagonals of the matrix. A classical algorithm for the Cholesky factorization of the *inverses* of such matrices is the so-called Levinson–Durbin algorithm [14, 8], an error analysis of which was provided by Cybenko [7]. He showed that, in the case of positive reflection coefficients, the residual error produced by the Levinson–Durbin procedure is comparable to the error produced by the Cholesky factorization [8, p. 191].

A related analysis was carried out by Sweet [22] for the Bareiss algorithm [2], which is also closely related to an algorithm of Schur [20, 12]. These are fast procedures for the Cholesky factorization of the Toeplitz matrix itself rather than its inverse. Sweet concluded that the Bareiss algorithm is asymptotically stable.

In recent work, Bojanczyk et al. [3] further extended and strengthened the conclusions of Sweet [22] by employing elementary downdating techniques [1, 4, 5] that are also characteristic of array formulations of the Schur algorithm [13, 17]. They considered the larger class of quasi-Toeplitz matrices [13], which includes the Toeplitz matrix as a special case, and provided an error analysis that establishes that the Schur algorithm for this class of matrices is asymptotically stable.

The interesting formulation of Bojanczyk et al. [3] motivated us to take a closer look at the numerical stability of a generalized Schur algorithm [13, 15, 18] that applies

* Received by the editors June 9, 1995; accepted for publication (in revised form) by N. J. Higham November 28, 1995. This work was supported in part by National Science Foundation award MIP-9409319.

† The Center for Control Engineering and Computation, Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106 (shiv@ece.ucsb.edu, sayed@ece.ucsb.edu).

to a wider class of positive-definite structured matrices R that satisfy displacement equations of the form $R - FRF^T = GJG^T$, where J is a signature matrix, F is a stable lower-triangular matrix, and G is a generator matrix. This class is briefly introduced in the next section, where the lower-triangular matrix F is shown to be pivotal in characterizing the structure of the matrix. For example, in the Toeplitz or quasi-Toeplitz case, the matrix F is equal to the shift matrix Z (i.e., a Jordan block with zero eigenvalue and ones on the first subdiagonal). Multiplying a column vector u by Z simply corresponds to shifting down the entries of u by one position. In general, however, the matrix F can be any lower-triangular matrix (for example, diagonal, bidiagonal, strictly lower triangular, etc.). This creates several complications that we address closely in order to guarantee a reliable algorithm.

For this purpose, we propose several modifications to the generalized Schur algorithm (Matlab codes for the new modified algorithm are provided at the end of this paper). In particular, two new schemes for carrying out the required hyperbolic rotations are introduced and special care is taken to ensure that the entries of the Blaschke matrix are computed to high relative accuracy. Also, a condition on the smallest eigenvalue of the matrix, along with several computational enhancements, is introduced in order to avoid possible breakdowns of the algorithm by assuring the positive-definiteness of the successive Schur complements.

We further use a perturbation analysis to indicate the best accuracy that can be expected from *any* finite-precision algorithm (slow or fast) that uses the generator matrix as the input data. We then show that the modified Schur algorithm proposed in this work essentially achieves this bound when coupled with a scheme to control the generator growth.

Another interesting idea that was recently suggested by Heinig [10] is the introduction of pivoting into algorithms for structured matrices when F is diagonal. In this paper, we have tried to clarify when pivoting may be helpful for positive-definite matrices. Numerical examples are included to support our observations. In particular, we emphasize that, in the diagonal F case, pivoting becomes necessary only when $\|F\|$ is very close to one. Furthermore, we note the following.

- If F is positive (or negative), a good strategy is shown to be the reordering of the entries of F in increasing order of magnitude.
- If F has both positive and negative entries, then numerical examples indicate that pivoting may not help in controlling the growth of the generators.

In our opinion, for positive-definite structured matrices, with diagonal or strictly lower-triangular F, the stabilization of the generalized Schur algorithm is critically dependent on the following:

- proper implementations of the hyperbolic rotations,
- proper evaluation of the Blaschke matrix–vector product,
- enforcement of positive-definiteness to avoid early breakdowns,
- control of the generator growth.

1.1. Notation. In the discussion that follows we use $\|\cdot\|$ to denote the 2-norm of its argument. We further assume, without loss of generality, that F is represented exactly in the computer. Also, the $\hat{\ }$ notation denotes computed quantities, while the $\bar{\ }$ notation denotes intermediate exact quantities. We further let ϵ 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, multiplica-

tions, 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 $|\delta| \leq \epsilon$. The same holds 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 positive-definite matrix R and an $n \times n$ lower-triangular real-valued matrix F. The displacement of R with respect to F is denoted by ∇_F and defined as

$$\nabla_F = R - FRF^T. \tag{1}$$

The matrix R is said to have low displacement rank with respect to F if the rank of ∇_F is considerably lower than n. In this case, R is said to have displacement structure with respect to F [13].

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

$$\nabla_F = GJG^T, \tag{2}$$

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

$$J = \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix}, \quad p + q = r. \tag{3}$$

The integer p denotes the number of positive eigenvalues of ∇_F, while the integer q denotes the number of its negative eigenvalues. Factorization (2) is highly nonunique. If G satisfies (2) then $G\Theta$ also satisfies (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 (1) and (2), a matrix R is said to be structured with respect to the displacement operation defined by (1) if it satisfies a displacement equation of the form

$$R - FRF^T = GJG^T, \tag{4}$$

with a "low" rank matrix G. Equation (4) uniquely defines R (i.e., it has a unique solution R) 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 be assumed throughout the paper, although it can be relaxed in some instances [13].

The pair (G, J) is said to be a generator pair for R since, along with F, it completely identifies R. Note, however, that while R 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 R, will generally be an order of magnitude faster than algorithms that operate on the entries of R 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 Cholesky factor of R. A recent survey on various other forms of displacement structure and on the associated forms of Schur algorithms is [13].

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

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):

$$Z = \begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix}. \tag{5}$$

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

$$T - ZTZ^T = \begin{bmatrix} 1 & 0 \\ t_1 & t_1 \\ \vdots & \vdots \\ t_{n-1} & t_{n-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_1 & t_1 \\ \vdots & \vdots \\ t_{n-1} & t_{n-1} \end{bmatrix}^T = GJG^T. \tag{6}$$

Another example is the so-called Pick matrix, which arises in the study of interpolation problems in the unit disc [19]:

$$R = \left[\frac{1 - \beta_i \beta_j}{1 - f_i f_j}\right]_{i,j=1}^{n},$$

where the β_i are real scalars and the f_i are distinct real points in the interval $(-1, 1)$. Let F denote the diagonal matrix $F = \operatorname{diag}[f_1, f_2, \ldots, f_n]$; then it can be verified that the above Pick matrix R has displacement rank 2 with respect to F since

$$R - FRF^T = \begin{bmatrix} 1 & \beta_1 \\ 1 & \beta_2 \\ \vdots & \vdots \\ 1 & \beta_n \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & \beta_1 \\ 1 & \beta_2 \\ \vdots & \vdots \\ 1 & \beta_n \end{bmatrix}^T. \tag{7}$$

More generally, one can allow for complex-valued quantities and define the Pick matrix [19] as

$$R = \left[\frac{x_i x_j^H - y_i y_j^H}{1 - f_i f_j^H}\right]_{i,j=1}^{n},$$

where ^{H} denotes Hermitian conjugation (complex conjugation for scalars), x_i and y_i are $1 \times p$ and $1 \times q$ row vectors, and f_i are complex points inside the open unit disc ($|f_i| < 1$). For the same diagonal matrix $F = \operatorname{diag}[f_1, f_2, \ldots, f_n]$, the above Pick matrix has displacement rank $r = (p + q)$, since

$$R - FRF^H = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix} \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix}^H. \tag{8}$$

Without loss of generality, the analysis provided in this paper focuses on real-valued data, i.e., on displacement equations of form (4) and on the important special case of matrices with displacement rank 2 (i.e., G has two columns and $J = (1 \oplus -1)$). The results can be extended to *higher displacement ranks* and to the complex case.

The displacement structure implied by (4) applies to symmetric matrices R. The case of nonsymmetric matrices will be pursued elsewhere since it also includes important matrices as special cases such as the Vandermonde matrix

$$V = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \dots & \alpha_1^n \\ 1 & \alpha_2 & \alpha_2^2 & \dots & \alpha_2^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha_n & \alpha_n^2 & \dots & \alpha_n^n \end{bmatrix}.$$

It is immediate to verify that the matrix V has displacement rank 1 since

$$V - FVZ^T = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}, \tag{9}$$

where F is now the diagonal matrix

$$F = \text{diag}\,[\alpha_1, \dots, \alpha_n].$$

3. The generalized Schur algorithm. The discussion in what follows focuses on symmetric positive-definite matrices R with displacement rank 2 with respect to a lower-triangular matrix F, viz., matrices R that satisfy displacement equations of the form

$$R - FRF^T = \begin{bmatrix} u_1 & v_1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} u_1 & v_1 \end{bmatrix}^T, \tag{10}$$

where u_1 and v_1 denote the $n \times 1$ column vectors of G. The diagonal entries of F are further assumed to be strictly inside the open unit disc ($|f_i| < 1$). In this case, the matrix F is said to be stable. This condition is clearly satisfied for the Toeplitz case (5), where $f_i = 0$, and for the Pick matrix (7). In applications, the following forms are the most frequent occurrences for F: $F = Z$, $F =$ a diagonal matrix with distinct entries, $F =$ a Jordan block,

$$F = \begin{bmatrix} f_1 & & & \\ 1 & f_1 & & \\ & \ddots & \ddots & \\ & & 1 & f_1 \end{bmatrix},$$

F in bidiagonal form,

$$F = \begin{bmatrix} f_1 & & & \\ 1 & f_2 & & \\ & \ddots & \ddots & \\ & & 1 & f_n \end{bmatrix},$$

or F strictly lower triangular such as Z, Z^2, $(Z \oplus Z)$, etc.

Also, since a generator matrix G is highly nonunique, it can always be chosen to be of the form

$$G = \begin{bmatrix} x & 0 \\ x & x \\ x & x \\ \vdots & \vdots \\ x & x \end{bmatrix}. \tag{11}$$

That is, the top entry of v_1, v_{11}, can always be chosen to be zero. Indeed, assume that a generator G for R is found that does not satisfy this requirement, say

$$G = \begin{bmatrix} u_{11} & v_{11} \\ x & x \\ \vdots & \vdots \\ x & x \end{bmatrix}.$$

It then follows from (10) that the $(1,1)$ entry of R, which is positive, is given by

$$R_{11} = \frac{|u_{11}|^2 - |v_{11}|^2}{1 - |f_1|^2} > 0.$$

Consequently, $|u_{11}| > |v_{11}|$ and a hyperbolic rotation Θ can always be found in order to reduce the row $\begin{bmatrix} u_{11} & v_{11} \end{bmatrix}$ to the form $\begin{bmatrix} \sqrt{|u_{11}|^2 - |v_{11}|^2} & 0 \end{bmatrix}$. The matrix $G\Theta$ can then be used instead of G as a generator for R.

A generator matrix of form (11) is said to be in proper form. Note that in the Toeplitz case (6), the generator G is already in proper form.

The following algorithm is known as the generalized Schur algorithm: it operates on the entries of (F, G, J) and provides the Cholesky factor of R. (We remark that the algorithm can be extended to more general scenarios, e.g., an unstable F, nonsymmetric matrices R, etc.—see [13, 18, 15].)

ALGORITHM 3.1 (the generalized Schur algorithm).

- *Input data: A stable lower-triangular matrix F, a generator $G_1 = G$ in proper form, with columns denoted by u_1 and v_1, and $J = (1 \oplus -1)$.*
- *Output data: The lower-triangular Cholesky factor L of the unique matrix R that satisfies* (10), *$R = LL^T$.*

The algorithm operates as follows: start with (u_1, v_1) and repeat for $i = 1, 2, \ldots, n$:

1. *Compute the $n \times n$ matrix $\Phi_i = (F - f_i I)(I - f_i F)^{-1}$. Note that the (i,i) diagonal entry of Φ_i is zero.*
2. *Form the prearray of numbers $\begin{bmatrix} \Phi_i u_i & v_i \end{bmatrix}$. At step i, the top i entries of $\Phi_i u_i$ and v_i will be zero.*
3. *Apply a hyperbolic rotation Θ_i in order to annihilate the $(i+1)$ entry of v_i. Denote the resulting column vectors by (u_{i+1}, v_{i+1}):*

$$\begin{bmatrix} u_{i+1} & v_{i+1} \end{bmatrix} = \begin{bmatrix} \Phi_i u_i & v_i \end{bmatrix} \Theta_i. \tag{12}$$

The matrix $G_{i+1} = \begin{bmatrix} u_{i+1} & v_{i+1} \end{bmatrix}$ *will also be in proper form, with the top* i *rows equal to zero:*

$$G_{i+1} = \begin{bmatrix} 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ x & 0 \\ \vdots & \vdots \\ x & x \end{bmatrix}.$$

4. *The* i*th column of the Cholesky factor* L *is given by*

$$l_i = \sqrt{1 - |f_i|^2}(I - f_i F)^{-1} u_i. \tag{13}$$

The top $(i-1)$ *entries of* l_i *are zero.*

After n steps, the algorithm provides the Cholesky decomposition

$$R = \sum_{i=1}^{n} l_i l_i^T, \tag{14}$$

as shown below. Moreover, the successive matrices G_i that are obtained via the recursion have an interesting interpretation. Let R_i denote the Schur complement of R with respect to its leading $(i-1) \times (i-1)$ submatrix. That is, $R_1 = R$, R_2 is the Schur complement with respect to the $(1,1)$ top left entry of R, R_3 is the Schur complement with respect to the 2×2 top left submatrix of R, and so on. The matrix R_i is therefore $(n-i+1) \times (n-i+1)$. Define the $n \times n$ embedding

$$\tilde{R}_i = \begin{bmatrix} 0 & 0 \\ 0 & R_i \end{bmatrix}.$$

Then it can be shown that [13]

$$\tilde{R}_i - F\tilde{R}_i F^T = G_i J G_i^T. \tag{15}$$

In other words, G_i is a generator matrix for the ith Schur complement, which is also structured.

THEOREM 3.2. *The generalized Schur algorithm provides the Cholesky decomposition of* R*, viz.,*

$$R = \sum_{i=1}^{n} l_i l_i^T. \tag{16}$$

Proof. It follows from relation (12) that

$$u_{i+1}u_{i+1}^T - v_{i+1}v_{i+1}^T = \Phi_i u_i u_i^T \Phi_i^T - v_i v_i^T, \tag{17}$$

where, in view of (13),

$$u_i = \frac{1}{\sqrt{1 - |f_i|^2}}(I - f_i F) l_i, \quad \Phi_i u_i = \frac{1}{\sqrt{1 - |f_i|^2}}(F - f_i I) l_i.$$

Summing (17) over i, up to $n-1$, we obtain

$$\sum_{i=1}^{n-1} u_{i+1}u_{i+1}^T - \sum_{i=1}^{n-1} \Phi_i u_i u_i^T \Phi_i^T = -v_1 v_1^T \text{ since } v_n = 0,$$

which is equivalent to

$$\sum_{i=1}^{n} u_i u_i^T - \sum_{i=1}^{n} \Phi_i u_i u_i^T \Phi_i^T = u_1 u_1^T - v_1 v_1^T \text{ since } \Phi_n u_n = 0.$$

Using the above expressions for u_i and $\Phi_i u_i$ in terms of l_i we obtain

$$\sum_{i=1}^{n} \left[\frac{(I - f_i F) l_i l_i^T (I - f_i F)^T}{1 - |f_i|^2} - \frac{(F - f_i I) l_i l_i^T (F - f_i I)^T}{1 - |f_i|^2} \right] = u_1 u_1^T - v_1 v_1^T.$$

Expanding and simplifying the ith term of the sum on the left-hand side we get

$$\sum_{i=1}^{n} \frac{(1 - |f_i|^2) l_i l_i^T - (1 - |f_i|^2) F l_i l_i^T F^T}{1 - |f_i|^2} = u_1 u_1^T - v_1 v_1^T.$$

Therefore,

$$\sum_{i=1}^{n} l_i l_i^T - F \left(\sum_{i=1}^{n} l_i l_i^T \right) F^T = u_1 u_1^T - v_1 v_1^T = GJG^T.$$

This shows that $\sum_{i=1}^{n} l_i l_i^T$ satisfies the displacement equation (10). Hence, by uniqueness,

$$R = \sum_{i=1}^{n} l_i l_i^T. \qquad \square$$

4. Limits to numerical accuracy. Given a symmetric positive-definite matrix R (not necessarily structured), if its Cholesky factor is evaluated by any standard backward stable method that operates on the entries of R, e.g., Gaussian elimination [8, Chap. 4], the corresponding error bound is given by

$$\|R - \hat{L}\hat{L}^T\| \leq c_1 \epsilon \|R\|,$$

where ϵ is the machine precision and c_1 is a low-order polynomial in n, the matrix size.

A fundamental question that needs to be answered then is the following: Given (F, G, J), but not R, how accurately can we expect to be able to compute the Cholesky factorization of R *irrespective* of the algorithm used (*slow or fast*)?

To address this issue we note that just representing (F, G) in finite precision already induces round-off errors. This fact in turn imposes limits on how accurate an algorithm that employs (F, G) can be. We demonstrate this point by the following example.

Let F be a stable diagonal matrix with distinct entries $\{f_i\}$ and assume f_1 is the largest in magnitude. Let the entries of the column vectors u_1 and v_1 be given by

$$u_{i1} = \left(\frac{1}{2}\right)^{i-1}, \quad v_{i1} = \gamma f_i u_{i1}, \quad i \geq 1,$$

where γ is chosen such that $0 < \gamma < 1$.

The unique matrix R that solves (10) for the given (F, u_1, v_1) is symmetric positive definite. This can be verified by invoking Theorem A.1 and by noting that

$$v_{i1} = u_{i1}\, s(f_i),$$

where $s(z)$ is the Schur function (i.e., analytic and strictly bounded by one in $|z| < 1$)

$$s(z) = \gamma z.$$

For this example, we have

$$\frac{|u_{11}|^2}{\|u_1\|^2} \geq \frac{1}{1/(1-\frac{1}{4})} = \frac{3}{4}. \tag{18}$$

Now define the perturbed vectors $\hat{u}_1$ and $\hat{v}_1$ with

$$\hat{u}_{11} = u_{11}(1+\delta), \quad \hat{u}_{i1} = u_{i1}, \quad i \geq 2, \quad \hat{v}_1 = v_1.$$

That is, we make only a relative perturbation in the first entry of u_1 and keep all other entries of u_1 and v_1 unchanged. Here, δ is a small number (for example, for round-off errors, $|\delta|$ is smaller than machine precision).

Let $\hat{R}$ be the unique solution of the displacement equation with the perturbed generator matrix,

$$\hat{R} - F\hat{R}F^T = \hat{G}J\hat{G}^T, \quad \hat{G} = \begin{bmatrix} \hat{u}_1 & \hat{v}_1 \end{bmatrix},$$

and introduce the error matrix $E = R - \hat{R}$. Then E is the unique solution of

$$E - FEF^T = GJG^T - \hat{G}J\hat{G}^T = u_1u_1^T - \hat{u}_1\hat{u}_1^T,$$

from which we find

$$|E_{11}| = \left| \frac{u_{11}^2(2\delta + \delta^2)}{1 - f_1^2} \right|.$$

Therefore,

$$|E_{11}| = \frac{u_{11}^2}{\|u_1\|^2} \frac{(2|\delta| + \delta^2)}{1 - f_1^2} \|u_1\|^2. \tag{19}$$

But since F is diagonal and $|f_1| < 1$ is its largest entry, we have

$$(1 - f_1^2)^{-1} = \|(I - F \otimes F)^{-1}\|, \tag{20}$$

where $\otimes$ denotes the Kronecker product of two matrices.

Using (18), we conclude that

$$|E_{11}| \geq \frac{3}{4} 2|\delta|\, \|(I - F \otimes F)^{-1}\|\, \|u_1\|^2,$$

from which we get a *lower bound* on the norm of the error matrix

$$\|E\| \geq |E_{11}| \geq \frac{3}{2}|\delta|\, \|(I - F \otimes F)^{-1}\|\, \|u_1\|^2.$$

We now show that by suitably choosing γ, the norm of R can be much smaller than the above bound. Indeed,

$$\begin{aligned}\|R\| &\le n \max_i R_{ii} \\ &= n \max_i \frac{u_{ii}^2 - v_{ii}^2}{1 - f_i^2} \\ &= n \max_i \left[\frac{1 - \gamma^2 f_i^2}{1 - f_i^2} \, u_{ii}^2 \right],\end{aligned}$$

from which we see that as $\gamma \to 1$, the norm of R can be bounded by $n\|u_1\|^2$. Therefore, in this example, $\|R\|$ can be made much smaller than $\|(I - F \otimes F)^{-1}\| \, \|u_1\|^2$ by choosing f_1 and γ close to one.

In summary, we have shown that, at the same time,

- $\|R - \hat{R}\|$ can be larger than $|\delta| \, \|(I - F \otimes F)^{-1}\| \, \|u_1\|^2$ and
- $\|R\|$ can be much smaller than $\|(I - F \otimes F)^{-1}\| \, \|u_1\|^2$.

Hence, in general, we cannot expect the error norm, $\|R - \hat{L}\hat{L}^T\|$, for any algorithm (slow or fast) that uses (F, G, J) as input data (but not R), to be as small as $c_1|\delta|\|R\|$ for some constant c_1.

Therefore, we conclude that *irrespective* of the algorithm we use (*slow or fast*), if the input data is (F, G, J), for a general lower-triangular F, we cannot expect a better bound than

$$\|R - \hat{L}\hat{L}^T\| \le c_2 \epsilon \, \|(I - F \otimes F)^{-1}\| \, \|u_1\|^2. \tag{21}$$

5. Hyperbolic rotation. Each step (12) of the generalized Schur algorithm requires the application of a hyperbolic rotation Θ_i. The purpose of the rotation is to rotate the $(i+1)$th row of the prearray $\begin{bmatrix} \Phi_i u_i & v_i \end{bmatrix}$ to proper form (recall that the top i rows of $\begin{bmatrix} \Phi_i u_i & v_i \end{bmatrix}$ are zero by construction). If we denote the top nonzero row of the prearray by

$$\begin{bmatrix} (\Phi_i u_i)_{i+1} & (v_i)_{i+1} \end{bmatrix} = \begin{bmatrix} \alpha_i & \beta_i \end{bmatrix},$$

then the expression for a hyperbolic rotation that transforms it to the form

$$\begin{bmatrix} \sqrt{|\alpha_i|^2 - |\beta_i|^2} & 0 \end{bmatrix}$$

is given by

$$\Theta_i = \frac{1}{\sqrt{1 - \rho_i^2}} \begin{bmatrix} 1 & -\rho_i \\ -\rho_i & 1 \end{bmatrix}, \quad \text{where } \rho_i = \frac{\beta_i}{\alpha_i}. \tag{22}$$

The positive-definiteness of R guarantees $|\rho_i| < 1$.

For notational convenience, we rewrite equation (12) in the compact form

$$G_{i+1} = \bar{G}_{i+1} \Theta_i, \tag{23}$$

where we have denoted the prearray $\begin{bmatrix} \Phi_i u_i & v_i \end{bmatrix}$ by $\bar{G}_{i+1}$. Note that both G_{i+1} and $\bar{G}_{i+1}$ can be regarded as generator matrices for the $(i+1)$th Schur complement.

Expression (23) shows that in infinite precision, the generator matrices G_{i+1} and $\bar{G}_{i+1}$ must satisfy the fundamental requirement

$$G_{i+1} J G_{i+1}^T = \bar{G}_{i+1} J \bar{G}_{i+1}^T. \tag{24}$$

Obviously, this condition cannot be guaranteed in finite precision. But it turns out that with an appropriate implementation of transformation (23), equality (24) can be guaranteed to within a "small" error. (The need to enforce the condition in finite precision was first observed for the $F = Z$ case by Bojanczyk et al. [3].) To see how, we consider the case when $\bar{G}_{i+1}$ is available exactly in the following subsections.

5.1. Direct implementation. A naive implementation of the hyperbolic transformation (23) can lead to large errors. Indeed, in finite precision, if we apply Θ_i directly to $\bar{G}_{i+1}$ we obtain a computed matrix $\hat{G}_{i+1}$ such that

$$\hat{G}_{i+1} = \bar{G}_{i+1}\Theta_i \; + \; E_{i+1},$$

where the norm of the error matrix E_{i+1} satisfies [8, p. 66]

$$\|E_{i+1}\| \; \leq \; c_3\epsilon \, \|\bar{G}_{i+1}\| \, \|\Theta_i\|.$$

The constant c_3 is a low-order polynomial in the size of the matrices and ϵ is the machine precision. Consequently,

$$\hat{G}_{i+1}J\hat{G}_{i+1}^T = \bar{G}_{i+1}J\bar{G}_{i+1}^T + E_{i+1}JE_{i+1}^T + \bar{G}_{i+1}\Theta_i E_{i+1}^T + E_{i+1}\Theta_i\bar{G}_{i+1}^T,$$

which shows that

$$\|\hat{G}_{i+1}J\hat{G}_{i+1}^T - \bar{G}_{i+1}J\bar{G}_{i+1}^T\| \leq c_4\epsilon \, \|\bar{G}_{i+1}\|^2 \, \|\Theta_i\|^2. \tag{25}$$

But since $\|\Theta_i\|$ can be large, the computed quantities are not guaranteed to satisfy relation (24) to sufficient accuracy.[1] This possibly explains the disrepute to which fast algorithms have fallen.

5.2. Mixed downdating. One possible way to ameliorate the above problem is to employ the mixed-downdating procedure as suggested by Bojanczyk et al. [3, 4]. This scheme guarantees that

$$\|\hat{G}_{i+1}J\hat{G}_{i+1}^T - \bar{G}_{i+1}J\bar{G}_{i+1}^T\| \leq c_5\epsilon \, \left(\|\bar{G}_{i+1}\|^2 + \|\hat{G}_{i+1}\|^2\right).$$

This bound is sufficient, when combined with other modifications suggested in §§6 and 8.4, to make the algorithm numerically reliable (§7).

5.3. A new method: The OD procedure. An alternate scheme is now proposed which is based on using the SVD of the hyperbolic rotation Θ_i (it is a modification of a scheme in [6]). Its good numerical properties come from the fact that the hyperbolic rotation is applied as a sequence of orthogonal and diagonal matrices, which we shall refer to as the OD (orthogonal–diagonal) procedure. Its other advantage is that it is a general technique that can be applied in other situations, such as the Schur algorithms for nonsymmetric matrices. It can be implemented with the same operation count as the mixed-downdating algorithm of [3].

[1] Interestingly, though, Bojanczyk et al. [3] showed that for the special case $F = Z$ and displacement rank $r = 2$, the direct implementation of the hyperbolic rotation still leads to an asymptotically backward stable algorithm. This conclusion, however, does not hold for higher displacement ranks. Stewart and Van Dooren [21] showed that for $F = Z$ and $r > 2$, the direct implementation of the hyperbolic rotation can be unstable.

It is straightforward to verify that any hyperbolic rotation of form (22) admits the following eigen(svd-)decomposition:

$$\Theta_i = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} \sqrt{\frac{\alpha_i+\beta_i}{\alpha_i-\beta_i}} & 0 \\ 0 & \sqrt{\frac{\alpha_i-\beta_i}{\alpha_i+\beta_i}} \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \frac{1}{\sqrt{2}} = Q_i D_i Q_i^T, \tag{26}$$

where the matrix

$$Q_i = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$$

is orthogonal ($Q_i Q_i^T = I$).

If the eigendecomposition $Q_i D_i Q_i^T$ is now applied to the prearray $\bar{G}_{i+1}$ in (23), then it can be shown that the computed generator matrix $\hat{G}_{i+1}$ satisfies (see Appendix C)

$$(\hat{G}_{i+1} + E_{2,i+1}) = (\bar{G}_{i+1} + E_{1,i+1})\Theta_i, \tag{27}$$

with

$$\|E_{1,i+1}\| \le c_6 \epsilon \, \|\bar{G}_{i+1}\|, \quad \|E_{2,i+1}\| \le c_7 \epsilon \, \|\hat{G}_{i+1}\|.$$

It further follows from (27) that $\hat{G}_{i+1}$ satisfies

$$(\hat{G}_{i+1} + E_{2,i+1}) J (\hat{G}_{i+1} + E_{2,i+1})^T = (\bar{G}_{i+1} + E_{1,i+1}) J (\bar{G}_{i+1} + E_{1,i+1})^T, \tag{28}$$

which shows that

$$\|\hat{G}_{i+1} J \hat{G}_{i+1}^T - \bar{G}_{i+1} J \bar{G}_{i+1}^T\| \le c_8 \epsilon \left(\|\bar{G}_{i+1}\|^2 + \|\hat{G}_{i+1}\|^2 \right). \tag{29}$$

ALGORITHM 5.1 (the OD procedure). *Given a hyperbolic rotation Θ with reflection coefficient $\rho = \beta/\alpha$, $|\rho| < 1$, and a prearray row vector $\begin{bmatrix} x & y \end{bmatrix}$, the postarray row vector $\begin{bmatrix} x_1 & y_1 \end{bmatrix}$ can be computed as follows:*

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

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

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

The algorithm guarantees (cf. (27)–(29)) the following error bounds:

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

with

$$\left\| \begin{bmatrix} e_1 & e_2 \end{bmatrix} \right\| \le c_9 \epsilon \left\| \begin{bmatrix} \hat{x}_1 & \hat{y}_1 \end{bmatrix} \right\|, \quad \left\| \begin{bmatrix} e_3 & e_4 \end{bmatrix} \right\| \le c_{10} \epsilon \left\| \begin{bmatrix} x & y \end{bmatrix} \right\|. \tag{30}$$

5.4. Another new method: 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,

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

The advantage of the method to be described in this section is that the computed quantities $\hat{x}_1$ and $\hat{y}_1$ satisfy the equation

$$\begin{bmatrix} \hat{x}_1 + e_1' & \hat{y}_1 + e_2' \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \Theta, \tag{31}$$

where the error terms satisfy

$$|e_1'| \le c_{11}\epsilon|\hat{x}_1|, \quad |e_2'| \le c_{12}\epsilon(|\hat{x}_1| + |\hat{y}_1|). \tag{32}$$

Compare this with (27), where the prearray is also perturbed. Moreover, we shall show in §10.2 that by a slight modification we can further enforce that $|\hat{x}_1| > |\hat{y}_1|$, which is needed to prevent breakdown in the algorithm. (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 $|x| > |y|$.)

The expression for x_1 can be written in the form

$$x_1 = \frac{|\alpha| x}{\sqrt{(\alpha-\beta)(\alpha+\beta)}} \left[1 - \frac{\beta}{\alpha}\frac{y}{x}\right].$$

The term $\xi = 1 - \frac{\beta}{\alpha}\frac{y}{x}$ can be evaluated to high relative accuracy as follows:

If $\frac{\beta}{\alpha}\frac{y}{x} < 1/2$
  then $\xi = 1 - \frac{\beta}{\alpha}\frac{y}{x}$
else
  $d_1 = \frac{|\alpha|-|\beta|}{|\alpha|}$, $d_2 = \frac{|x|-|y|}{|x|}$
  $\xi = d_1 + d_2 - d_1 d_2$

The argument employed in §6.1 establishes that

$$\hat{\xi} = \xi(1 + 33\delta_1),$$

where δ_1 denotes a quantity that is smaller than the machine precision in magnitude. Therefore, x_1 can be computed to high relative accuracy from the expression

$$x_1 = \frac{|\alpha| x \xi}{\sqrt{(\alpha-\beta)(\alpha+\beta)}};$$

i.e.,

$$\hat{x}_1 = x_1(1 + c_{13}\delta_2)$$

for some constant c_{13}.

To compute y_1 we use the expression

$$y_1 = x_1 - \sqrt{\frac{\alpha+\beta}{\alpha-\beta}}\,(x-y).$$

Then the computed y_1 satisfies

$$\hat{y}_1 = \left(x_1(1+c_{13}\delta_2) - \sqrt{\frac{\alpha+\beta}{\alpha-\beta}}\,(x-y)(1+c_{14}\delta_3) \right)(1+\delta_4),$$

from which we get

$$\hat{y}_1 = y_1 + x_1 c_{15}\delta_5 + \hat{y}_1 c_{16}\delta_6.$$

Therefore,

$$|\hat{y}_1 - y_1| \le c_{17}\epsilon\left[|\hat{x}_1| + |\hat{y}_1|\right].$$

In summary, the H procedure is the following.

ALGORITHM 5.2 (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
$x_1 \leftarrow \frac{|\alpha| x \xi}{\sqrt{(\alpha-\beta)(\alpha+\beta)}}$
$y_1 \leftarrow x_1 - \sqrt{\frac{\alpha+\beta}{\alpha-\beta}}\,(x-y).$

This algorithm guarantees (31) and (32). We remark that 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. (31)), whereas the OD method is only stable (cf. (27)).

From now on we shall denote by $\hat{u}_{i+1}$ and $\hat{v}_{i+1}$ the computed generator columns at step i; i.e.,

$$\hat{G}_{i+1} = \begin{bmatrix} \hat{u}_{i+1} & \hat{v}_{i+1} \end{bmatrix},$$

starting with

$$\bar{G}_i = \begin{bmatrix} \Phi_i \hat{u}_i & \hat{v}_i \end{bmatrix}.$$

6. Blaschke matrix. Each step of the algorithm also requires multiplying the Blaschke matrix Φ_i by u_i. (Note that the top i rows of $\Phi_i u_i$ are zero and, hence, can be ignored in the computation.) In this section, we consider the following two cases.

- F is stable and diagonal, in which case Φ_i itself is diagonal and given by

$$(\Phi_i)_{jj} = \frac{f_j - f_i}{1 - f_i f_j}.$$

- F is strictly lower triangular; e.g., $F = Z$, $F = (Z \oplus Z)$, or other more involved choices. In these situations, the matrix Φ_i is equal to F since the f_i are all zero,

$$\Phi_i = F.$$

6.1. The case of diagonal F. The goal of this section is to show how to compute $\Phi_i \hat{u}_i$ to high componentwise relative accuracy (i.e., high relative accuracy for each component of the computed vector). Here, $\hat{u}_i$ denotes the computed value of u_i.

The numerator of $(\Phi_i)_{jj}$ can be computed to high relative accuracy as

$$fl(f_j - f_i) = (f_j - f_i)(1 + \delta_1).$$

Computing the denominator $x_{ij} = (1 - f_i f_j)$ to high relative accuracy is a bit trickier, as the following example shows.

Let $f_1 = f_2 = 0.998842$. Then in 6-digit arithmetic $1 - f_1 f_2 \approx 2.31500 \times 10^{-3}$, whereas the actual answer is $2.31465903600 \times 10^{-3}$. Therefore, the relative error is approximately 1.5×10^{-4}. Using the scheme given below, we find $1 - f_1 f_2 \approx 2.31466 \times 10^{-3}$. The relative error is now approximately 4.2×10^{-7}.

The scheme we use to compute x_{ij} is as follows:

If $f_i f_j < 1/2$
  then $x_{ij} = 1 - f_i f_j$
else
  $d_j = 1 - |f_j|$, $d_i = 1 - |f_i|$
  $x_{ij} = d_i + d_j - d_i d_j$

We now show that this scheme ensures that x_{ij} is computed to high relative accuracy. Indeed, when $f_i f_j < 1/2$, we have $|1 - f_i f_j| > 1/2$. Moreover,

$$\begin{aligned} \hat{x}_{ij} &= (1 - f_i f_j (1 + \delta_2))(1 + \delta_3) \\ &= x_{ij} \left(1 - \frac{f_i f_j}{x_{ij}} \delta_2\right)(1 + \delta_3). \end{aligned}$$

Since $|f_i f_j / x_{ij}| < 1$, we have $\hat{x}_{ij} = x_{ij}(1 + 3\delta_4)$. On the other hand, for $f_i f_j \geq 1/2$, we note that

$$\hat{d}_j = d_j(1 + \delta_5), \quad \hat{d}_i = d_i(1 + \delta_6),$$

and

$$\hat{x}_{ij} = \left[(\hat{d}_i + \hat{d}_j)(1 + \delta_7) - \hat{d}_i \hat{d}_j (1 + \delta_8)\right](1 + \delta_9),$$

which, when simplified, gives

$$\hat{x}_{ij} = x_{ij} \left(1 + 11 \frac{(d_i + d_j + d_i d_j)}{x_{ij}} \delta_{10}\right).$$

We shall now show that $(d_i + d_j + d_i d_j)/x_{ij} < 3$. Indeed, first note that d_i and d_j are positive numbers smaller than $1/2$ since $|f_i| > 1/(2|f_j|) > 1/2$. It then follows from

$$d_i < \frac{1}{2} < \frac{1}{2} + \frac{d_i}{2d_j}$$

that $d_i + d_j > 2d_i d_j$. Therefore,

$$\frac{d_i + d_j + d_i d_j}{d_i + d_j - d_i d_j} < \frac{\frac{3}{2}(d_i + d_j)}{\frac{1}{2}(d_i + d_j)} = 3,$$

which shows that

$$\hat{x}_{ij} = x_{ij}\,(1 + 33\delta_{11})\,.$$

In summary, we have shown how to compute Φ_i to componentwise accuracy. Therefore, since Φ_i is diagonal, $\Phi_i \hat{u}_i$ can be computed to componentwise high relative accuracy. More specifically,

$$fl\,(\Phi_i \hat{u}_i)_j = (\Phi_i \hat{u}_i)_j (1 + 72\delta_{12}). \tag{33}$$

We should remark that if the denominator entries $(1 - f_i f_j)$ were instead computed directly, the error in computing $(\Phi_i \hat{u}_i)_j$ would also depend on the norm of $(I - f_i F)^{-1}$, which can be large. For this reason, we have introduced the above computational scheme for evaluating $(1 - f_i f_j)$. This scheme, however, is not totally successful when F is a general triangular matrix (for example, when F is bidiagonal). A way around this difficulty will be addressed elsewhere. But for a strictly lower-triangular F, the situation is far simpler, as shown in the next subsection.

But for now, let us consider how to compute the l_i's. Define

$$\bar{l}_i = \sqrt{1 - f_i^2}\;(I - f_i F)^{-1} \hat{u}_i. \tag{34}$$

We use the expression

$$(\bar{l}_i)_j = \frac{\sqrt{(1 - f_i)(1 + f_i)}}{1 - f_i f_j} (\hat{u}_i)_j$$

to compute $\bar{l}_i$ with the technique explained above for the denominator $(1 - f_i f_j)$. Then we can show that

$$(\hat{l}_i)_j = (\bar{l}_i)_j (1 + c_{18}\delta_{13}). \tag{35}$$

6.2. The case of strictly lower-triangular F. For a strictly lower-triangular F, we use the standard matrix–vector multiplication. In this case, the computed quantities satisfy the relation [8, p. 66]

$$\|fl(F\hat{u}_i) - F\hat{u}_i\| \leq c_{19}\epsilon \|F\|\,\|\hat{u}_i\|.$$

Also, since $f_i = 0$,

$$\bar{l}_i = \hat{u}_i = \hat{l}_i. \tag{36}$$

6.3. Enforcing positive-definiteness. Condition (44) on the matrix R in §7.1 guarantees the positive-definiteness of the successive Schur complements and, therefore, that $|\Phi_i \hat{u}_i|_{i+1} > |\hat{v}_i|_{i+1}$. This assures that the reflection coefficients will be smaller than one in magnitude, a condition that we now enforce in finite precision as follows:

If $|fl(\Phi_i \hat{u}_i)|_{i+1} < |\hat{v}_{i+1,i}|$ then
$fl(\Phi_i \hat{u}_i)_{i+1} \leftarrow |\hat{v}_{i+1,i}|(1+3\epsilon)\text{sign}(fl(\Phi_i \hat{u}_i)_{i+1})$.

This enhancement, along with condition (44), will be shown in §7.1 to guarantee that the algorithm will complete without any breakdowns.

7. Error analysis of the algorithm. The argument in this section is motivated by the analysis in Bojanczyk et al. [3].

Note that for diagonal and for strictly lower-triangular F we can write

$$\|fl(\Phi_i \hat{u}_i) - \Phi_i \hat{u}_i\| \leq c_{20}\epsilon \|\Phi_i\| \|\hat{u}_i\|.$$

Therefore, from the error analysis (27) of the hyperbolic rotation we obtain

$$(\hat{G}_{i+1} + E_{2,i+1}) = (\begin{bmatrix} \Phi_i \hat{u}_i & \hat{v}_i \end{bmatrix} + E_{3,i+1})\Theta_i, \tag{37}$$

where

$$\|E_{3,i+1}\| \leq c_{21}\epsilon(\|\Phi_i\|\|\hat{u}_i\| + \|\hat{v}_i\|).$$

It then follows that

$$\hat{u}_{i+1}\hat{u}_{i+1}^T - \hat{v}_{i+1}\hat{v}_{i+1}^T = \Phi_i \hat{u}_i \hat{u}_i^T \Phi_i^T - \hat{v}_i \hat{v}_i^T - M_{i+1}, \tag{38}$$

where

$$\|M_{i+1}\| \leq c_{22}\epsilon(\|\hat{u}_{i+1}\|^2 + \|\Phi_i\|^2\|\hat{u}_i\|^2 + \|\hat{v}_{i+1}\|^2 + \|\hat{v}_i\|^2). \tag{39}$$

Since

$$\bar{l}_i = \sqrt{1-f_i^2}\,(I - f_i F)^{-1}\hat{u}_i,$$

the following two equations hold:

$$\hat{u}_i = \frac{1}{\sqrt{1-f_i^2}}(I - f_i F)\bar{l}_i, \quad \Phi_i \hat{u}_i = \frac{1}{\sqrt{1-f_i^2}}(F - f_i I)\bar{l}_i.$$

Hence, following the proof of Theorem 3.2, we can establish that

$$\sum_{i=1}^{n} \bar{l}_i \bar{l}_i^T - F\left(\sum_{i=1}^{n} \bar{l}_i \bar{l}_i^T\right) F^T = \hat{u}_1 \hat{u}_1^T - \hat{v}_1 \hat{v}_1^T - \sum_{i=1}^{n} M_i. \tag{40}$$

Define

$$\bar{R} = \sum_{i=1}^{n} \bar{l}_i \bar{l}_i^T$$

and

$$\bar{E} = R - \bar{R}.$$

Then note that $\bar{E}$ satisfies

$$\bar{E} - F\bar{E}F^T = \sum_{i=1}^{n} M_i \tag{41}$$

since, we assume, $\hat{u}_1 = u_1$ and $\hat{v}_1 = v_1$.

Now if

$$E = R - \sum_{i=1}^{n} \hat{l}_i \hat{l}_i^T = (R - \bar{R}) + \left(\bar{R} - \sum_{i=1}^{n} \hat{l}_i \hat{l}_i^T \right),$$

then

$$\|E\| \le \|\bar{E}\| + \left\| \sum_{i=1}^{n} \bar{l}_i \bar{l}_i^T - \sum_{i=1}^{n} \hat{l}_i \hat{l}_i^T \right\|.$$

Using (35) and (36) we can establish that, for diagonal or strictly lower-triangular F,

$$\left\| \sum_{i=1}^{n} \bar{l}_i \bar{l}_i^T - \sum_{i=1}^{n} \hat{l}_i \hat{l}_i^T \right\| \le c_{23}\epsilon \|\bar{R}\|.$$

Therefore,

$$\begin{aligned} \|E\| &\le c_{24}\epsilon \|\bar{R}\| + \|(I - F \otimes F)^{-1}\| \sum_{i=1}^{n} \|M_i\| \\ &\le c_{25}\epsilon \left[\|\bar{R}\| + \|(I - F \otimes F)^{-1}\| \sum_{i=1}^{n} \left\{ (1 + \|\Phi_i\|^2) \, \|\hat{u}_i\|^2 + \|\hat{v}_i\|^2 \right\} \right]. \end{aligned} \tag{42}$$

7.1. Avoiding breakdown. The above error analysis assumes that the algorithm does not break down. That is, at every iteration the reflection coefficients of the hyperbolic rotations are assumed to be strictly less than one in magnitude. In this section we show that this can be guaranteed by imposing condition (44) on R *and* by employing the enhancement suggested in §6.3.

The argument is inductive. We know that $|\Phi_1 u_1|_2 > |v_1|_2$. Now assume that the algorithm has successfully gone through the first i steps and define the $\bar{l}_i$ as in (34). Also define the matrix S_i that solves the displacement equation

$$S_i - FS_iF^T = \hat{u}_i \hat{u}_i^T - \hat{v}_i \hat{v}_i^T, \tag{43}$$

as well as the matrix

$$\bar{R}_i = \sum_{j=1}^{i-1} \bar{l}_j \bar{l}_j^T + S_i.$$

Following the proof of Theorem 3.2, we can establish that

$$\bar{R}_i - F\bar{R}_iF^T = \hat{u}_1 \hat{u}_1^T - \hat{v}_1 \hat{v}_1^T - \sum_{j=1}^{i-1} M_j.$$

If we further introduce the error matrix

$$\bar{E}_i = R - \bar{R}_i,$$

we then note that it satisfies

$$\bar{E}_i - F\bar{E}_i F^T = \sum_{j=1}^{i-1} M_j$$

since, we assume, $\hat{u}_1 = u_1$ and $\hat{v}_1 = v_1$.

Then, as before, we can establish that

$$\|R - \bar{R}_i\| \leq c_{26}\epsilon\|\bar{R}_i\| + \|(I - F \otimes F)^{-1}\| \sum_{j=1}^{i-1} \|M_j\|$$

$$\leq c_{27}\epsilon \left[\|\bar{R}_i\| + \|(I - F \otimes F)^{-1}\| \sum_{j=1}^{i-1} \{(1 + \|\Phi_j\|^2)\, \|\hat{u}_j\|^2 \; + \; \|\bar{R}_i\|(1 + \|F\|^2)\} \right],$$

where in the second step we use Lemma B.1. Now note that for diagonal and stable F, $\|\Phi_i\| \leq 1$, and if F is strictly lower triangular then $\|\Phi_i\| \leq \|F\|$. Therefore, combining both cases, we get $1 + \|\Phi_i\|^2 \leq 2 + \|F\|^2$, which leads to

$$\|R - \bar{R}_i\| \leq c_{28}\epsilon\|(I - F \otimes F)^{-1}\|\, (2 + \|F\|^2) \left[\|\bar{R}_i\| + \sum_{j=1}^{i-1} \|\hat{u}_j\|^2 \right].$$

It now follows that if the minimum eigenvalue of R meets the lower bound

$$\lambda_{\min}(R) \; > \; c_{28}\epsilon\|(I - F \otimes F)^{-1}\|\, (2 + \|F\|^2) \left[\|\bar{R}_i\| + \sum_{j=1}^{i-1} \|\hat{u}_j\|^2 \right],$$

then $\bar{R}_i$ will be guaranteed to be positive definite and, consequently, S_i will be positive definite. Then, by positive-definiteness,

$$|\Phi_i \hat{u}_i|_{i+1} \; > \; |\hat{v}_i|_{i+1}.$$

But since we enforce $fl(|\Phi_i \hat{u}_i|_{i+1}) > |\hat{v}_i|_{i+1}$, the algorithm can continue to the next iteration.

This suggests the following lower bound on the smallest eigenvalue of R in order to avoid breakdown of the algorithm (i.e., in order to ensure the positive-definiteness of the computed Schur complements):

$$\lambda_{\min}(R) \; > \; c_{28}\epsilon\|(I - F \otimes F)^{-1}\|\, (2 + \|F\|^2) \left[\|\bar{R}\| + \sum_{j=1}^{n} \|\hat{u}_j\|^2 \right]. \tag{44}$$

7.2. Error bound. From the discussion in the previous section, we can conclude the following.

THEOREM 7.1 (error bound). *The generalized Schur algorithm for a structured positive-definite matrix R satisfying* (44), *with a stable diagonal or strictly lower-triangular F, implemented as detailed in this paper (see listing in Appendix* D*), guarantees the following error bound:*

$$\|E\| \leq c_{29}\epsilon\|(I - F \otimes F)^{-1}\| \, (2 + \|F\|^2) \left[\|\bar{R}\| + \sum_{j=1}^{n} \|\hat{u}_j\|^2 \right]. \tag{45}$$

The term $\|(I - F \otimes F)^{-1}\|$ in the error bound is expected from the perturbation analysis of §4. However, the presence of the norms of the successive generators makes the error bound larger than the bound suggested by the perturbation analysis, which depends only on the norm of the first generator matrix.

7.3. Growth of generators. The natural question then is as follows: How big can the norm of the generators be? An analysis based on the norm of the hyperbolic rotations used in the algorithm gives the following bound:

$$\|\hat{u}_i\| \leq c_{30}\|u_1\| \prod_{k=1}^{i} \sqrt{\frac{1 + |\rho_k|}{1 - |\rho_k|}}, \tag{46}$$

which is reminiscent of the error bounds of Cybenko for the Levinson algorithm [7]. A tighter bound can be obtained by using the fact that R is positive definite to get (recall (13))

$$\|u_i\|^2 \leq \|(I - F \otimes F)^{-1}\| \, (1 + \|F\|^2)^2 \, \|R\|. \tag{47}$$

This shows that the growth of the generators depends on $\|(I - F \otimes F)^{-1}\|$. But for a strictly lower-triangular F a better bound is $\|u_i\|^2 \leq \|R\|$. Therefore, for strictly lower-triangular F the error bound is as good as can be expected from the perturbation analysis of §4.

What does this say about the stability of the generalized Schur algorithm for a diagonal and stable F? Clearly, when the eigenvalues of F are sufficiently far from 1 the method has excellent numerical stability. The algorithm degrades as the eigenvalues of F get closer to 1. This is to be expected from the perturbation analysis (whether we use a slow or a fast algorithm). However, if the generators grow rapidly (i.e., as fast as (47)) then the algorithm degrades faster than the rate predicted by the perturbation analysis.

Is there anything further we can do to ameliorate this problem? One thing we have not considered yet is *pivoting*, which is possible only when F is diagonal. We discuss this in §8.

7.4. F strictly lower triangular. When F is strictly lower triangular the error bound can be written in the alternate form

$$\|E\| \leq c_{31}\epsilon(2 + \|F\|^2) \left(\sum_{i=1}^{n} \|F^i\|^2 \right) \left[\|\bar{R}\| + \left(\sum_{i=1}^{n} \|\hat{u}_i\|^2 \right) \right].$$

This shows that when F is contractive ($\|F\| \leq 1$), the error bound is as good as can be expected from the perturbation analysis; i.e.,

$$\|E\| \leq c_{32}\epsilon \left[\|\bar{R}\| + \left(\sum_{i=1}^{n} \|\hat{u}_i\|^2 \right) \right].$$

We observed in §7.3 that $\|u_i\|^2 \leq \|R\|$. Therefore, if F is strictly lower triangular and contractive, then the algorithm is backward stable.

This includes the important class of positive-definite quasi-Toeplitz matrices, which correspond to $F = Z$. In this case, we strengthen the result of Bojanczyk et al. [3], which states that for quasi-Toeplitz symmetric positive-definite matrices, the Schur algorithm is asymptotically backward stable. Our analysis shows that the modified algorithm proposed here is backward stable provided the smallest eigenvalue of the quasi-Toeplitz matrix satisfies

$$\lambda_{\min}(R) \; > \; c_{32}\epsilon \left[\|\bar{R}\| + \sum_{j=1}^{n} \|\hat{u}_j\|^2 \right].$$

If F is strictly lower triangular but noncontractive then the error norm can possibly depend on $\|(I - F \otimes F)^{-1}\|$.

7.5. F diagonal. For the special case of a stable diagonal F, the bound in (45) may suggest that the norm of the error can become very large when the magnitude of the diagonal entries of F become close to one. But this is not necessarily the case (see also the numerical example in the next section).

First note that the bound in (39) can be strengthened since the hyperbolic rotations are applied to each row independently. By assumption (44), the Schur complement generated by $(\hat{u}_i, \hat{v}_i)$ is positive definite. Hence, by Theorem A.1,

$$|\hat{u}_i| \; > \; |\Phi_i \hat{u}_i| \; > \; |\hat{v}_i|,$$

and we conclude that

$$|M_{i+1}| \; \leq c_{33}\epsilon(|\hat{u}_{i+1}||\hat{u}_{i+1}|^T + |\hat{u}_i||\hat{u}_i|^T).$$

Define

$$\rho_{j,i} = \frac{\hat{v}_{j,i}}{\hat{u}_{j,i}}.$$

It then follows from (43) that

$$(S_i)_{j,k} = \frac{\hat{u}_{j,i}\hat{u}_{k,i}}{1 - f_j f_k} \left(1 - \rho_{j,i}\rho_{k,i}\right).$$

Therefore,

$$\frac{|\hat{u}_{j,i}\hat{u}_{k,i}|}{1 - f_j f_k} \; \leq \; \frac{|(S_i)_{j,k}|}{1 - \rho_{j,i}\rho_{k,i}} \leq \frac{\|\bar{R}_i\|}{1 - \max_j \left(\rho_{j,i}^2\right)}.$$

Now using expression (41) we obtain

$$|\bar{E}|_{j,k} = \frac{|\sum_{i=1}^{n} M_i|_{j,k}}{1 - f_j f_k} \leq c_{34}\epsilon \frac{\sum_{i=1}^{n} \|\bar{R}_i\|}{1 - \max_{j,i} \left(\rho_{j,i}^2\right)}.$$

This establishes the following alternative bound for the error matrix E:

$$\|E\| \leq c_{35}\epsilon \, \frac{\sum_{i=1}^{n} \|\bar{R}_i\|}{1 - \max_{j,i} \left(\rho_{j,i}^2\right)},$$

which is independent of the $\{f_i\}$. In other words, if the coefficients $\rho_{j,i}$ are sufficiently smaller than one, then the algorithm will be backward stable irrespective of how close the $\{|f_i|\}$ are to one.

8. Pivoting with diagonal F. When F is diagonal it is possible to accommodate pivoting into the algorithm, as suggested by Heinig [10]; it corresponds to reordering the f_j's, $u_{j,i}$'s, and $v_{j,i}$'s identically at the ith iteration of the algorithm. This has the effect of computing the Cholesky factorization of PRP^T, where P is the product of all the permutations that were carried out during the algorithm.

In finite precision, pivoting strategies are employed in classical Cholesky factorization algorithms when the positive-definite matrix is numerically singular. In the context of the generalized Schur algorithm of this paper, the main motivation for pivoting should be to keep the norm of the generator matrices as small as possible! This is suggested by the expression for the error bound in (45), which depends on the norm of the generators. Note that this motivation has little to do with the size of the smallest eigenvalue of the matrix.

We would like to emphasize that pivoting is necessary only when the norm of F is very close to one since otherwise the generators do not grow appreciably (47).

8.1. A numerical example. The first question that arises then is whether there exists a pivoting strategy that guarantees a small growth in the norm of the generators. Unfortunately, we have numerical examples that show that irrespective of what pivoting strategy is employed, the norms of the generators may not exhibit significant reduction.

Consider the matrix R the satisfies the displacement equation $R - FRF^T = GJG^T$ with

$$G = \begin{bmatrix} 0.26782811166721 & 0.26782805810159 \\ 0.65586390188981 & -0.65586311485320 \\ 0.65268528182561 & 0.65268365011256 \\ 0.26853783287812 & -0.26853149538590 \end{bmatrix}, \quad J = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

and

$$F = \text{diagonal}\{0.9999999, -0.9999989, 0.9999976, -0.9999765\}.$$

The matrix R is positive definite since the entries of the column vector v_1 were generated from the relation $v_{i,1} = u_{i,1}s(f_i)$, where $s(z)$ is the Schur function $s(z) = 0.9999999z$. Table 8.1 lists the values of

$$\sum_{i=1}^{n} \|\hat{u}_i\|^2$$

for all 24 possible pivoting options of the rows of the generator matrix G. The results indicate that none of the pivoting options significantly reduces the size of the generators. Indeed, note that the norm of u_1 is approximately one, while the best growth rate we achieve with pivoting is approximately 10^4. This best case is achieved when the diagonal entries of F are approximately in increasing order of magnitude.

8.2. The case of positive F. This raises the next question: Is pivoting useful at all? It *is* useful when the the entries of the F matrix are strictly positive (or negative). In this case, we permute the entries of F (and, correspondingly, the entries of u_1 and v_1) such that the diagonal of F is in increasing order of magnitude. Then it is shown below that

$$\|\hat{u}_i\| \leq \|\bar{R}\|^{1/2},$$

TABLE 8.1

$10^{-6}\sum_{i=1}^{n}\|\hat{u}_i\|^2$	
5.30	0.41
5.30	0.41
5.21	0.40
5.21	0.40
5.03	0.40
5.03	0.40
0.83	0.04
0.83	0.04
0.83	0.04
0.83	0.04
0.83	0.04
0.83	0.04

which makes the first-order term of the upper bound on E depend only on the first power of $\|(I - F \otimes F)^{-1}\|$. Indeed, we know that

$$\hat{u}_i = \frac{1}{\sqrt{1-|f_i|^2}}\,(I - f_i F)\bar{l}_i,$$

where the top $(i-1)$ entries of u_i and l_i are zero. For $j \geq i$,

$$\hat{u}_{j,i} = \frac{1 - f_i f_j}{\sqrt{1-|f_i|^2}}\,\bar{l}_{j,i},$$

and due to the ordering of the entries of F, and since $|f_i| < 1$, we have

$$1 - f_i f_j \;\leq\; 1 - f_i^2 \;\leq\; \sqrt{1-|f_i|^2}.$$

Therefore,

$$\frac{1 - f_i f_j}{\sqrt{1-|f_i|^2}} \leq 1,$$

and we conclude that

$$\|\hat{u}_i\| \leq \|\bar{l}_i\| \;\leq\; \|\bar{R}\|^{1/2},$$

as desired.

8.3. The nonpositive case. When F is not positive, the example in §8.1 suggests that pivoting may not help in general. However, it may still be beneficial to try a heuristic pivoting strategy to control the growth of the generators. Ideally, at the ith iteration we should pick the row of the prearray $\bar{G}_{i+1}$ which would lead to the smallest (in norm) postarray G_{i+1}. Since there seems to be no efficient way to do this we suggest picking the row that leads to the smallest reflection coefficient (in magnitude) for the hyperbolic rotation Θ_i. As suggested by the example of §8.1, an alternate strategy would be to order the f_i's in increasing order of magnitude.

We stress that pivoting is relevant *only* when the norm of F is very close to one, as indicated by the error bound (45) and (47).

8.4. Controlling the generator growth. We have shown in §§7.4 and 8.2 that the generators do not grow (i) if F is strictly lower triangular and contractive or (ii) if F is a positive diagonal matrix with increasing diagonal entries. We now show how to control the generator growth in general using an idea suggested by Gu [9].

It follows from (43) that

$$\|\hat{G}_i J \hat{G}_i^T\| = \|S_i - F S_i F^T\|.$$

Let $W_i \Lambda_i W_i^T$ denote the eigendecomposition of $\hat{G}_i J \hat{G}_i^T$, where Λ_i is a 2×2 real diagonal matrix with $(\Lambda_i)_{11} > 0$ and $(\Lambda_i)_{22} < 0$. Then $W_i \sqrt{|\Lambda_i|}$ can be taken as a generator for S_i with the desirable property that

$$\left\| W_i \sqrt{|\Lambda_i|} \right\|^2 = \|\Lambda_i\| = \|S_i - F S_i F^T\| \le \|S_i\|(1 + \|F\|^2) \le \|\bar{R}_i\|(1 + \|F\|^2),$$

where $\|\bar{R}_i\| \approx \|R\|$ to first order in ϵ.

Therefore, whenever the generator grows, i.e., $\|\hat{G}_i\|^2$ becomes larger than a given threshold (say, $2\|R\|(1 + \|F\|^2)$), we can replace it by $W_i \sqrt{|\Lambda_i|}$. This computation can be done in $O((n-i)r^2 + r^3)$ flops ($r = 2$ in the case under consideration) by first computing the QR factorization of $\hat{G}_i$, say

$$\hat{G}_i = Q_i P_i, \quad Q_i Q_i^T = I,$$

and then computing the eigendecomposition of the 2×2 matrix $P_i J P_i^T$. We can then get W_i by multiplying Q_i by the orthogonal eigenvector matrix of $P_i J P_i^T$.

9. Solution of linear systems of equations. The analysis in the earlier sections suggests that for $\|F\|$ sufficiently close to one, the error norm can become large. However, if our original motivation is the solution of the linear system of equations

$$Rx = b,$$

then the error can be improved by resorting to iterative refinement if either the matrix R is given or if it can be computed accurately from (F, G). In what follows we show that for a diagonal F, the matrix R can be evaluated to high relative accuracy if u_1 and v_1 are exact.

9.1. Computing the matrix R. Given a positive-definite structured matrix R that satisfies

$$R - FRF^T = u_1 u_1^T - v_1 v_1^T,$$

with F diagonal and stable, its entries can be computed to high relative accuracy, as we explain below.

It follows from the displacement equation that

$$r_{ij} = \frac{u_{i,1}u_{j,1} - v_{i,1}v_{j,1}}{1 - f_i f_j} = \frac{u_{i,1}u_{j,1}\left(1 - \frac{v_{i,1}}{u_{i,1}}\frac{v_{j,1}}{u_{j,1}}\right)}{1 - f_i f_j},$$

where, by positive-definiteness, the ratios

$$\frac{v_{i,1}}{u_{i,1}} \quad \text{and} \quad \frac{v_{j,1}}{u_{j,1}}$$

are strictly less than one.

The term $\xi = 1 - \frac{v_{i,1}}{u_{i,1}}\frac{v_{j,1}}{u_{j,1}}$ can be evaluated to high relative accuracy, as explained earlier in the paper in §5.4, viz.,

If $\frac{v_{i,1}}{u_{i,1}}\frac{v_{j,1}}{u_{j,1}} < 1/2$
  then $\xi = 1 - \frac{v_{i,1}}{u_{i,1}}\frac{v_{j,1}}{u_{j,1}}$
else
  $d_1 = \frac{|u_{i,1}|-|v_{i,1}|}{|u_{i,1}|}$, $d_2 = \frac{|u_{j,1}|-|v_{j,1}|}{|u_{j,1}|}$
  $\xi = d_1 + d_2 - d_1 d_2$

Likewise, we evaluate $\mu = (1 - f_i f_j)$ and then

$$r_{ij} = \frac{u_{i,1}u_{j,1}\xi}{\mu}.$$

This guarantees that

$$\hat{r}_{i,j} = r_{i,j}(1 + c_{36}\delta_7).$$

9.2. Iterative refinement. If the factorization $\hat{L}\hat{L}^T$ is not too inaccurate and if R is not too ill conditioned, then it follows from the analysis in [11] that the solution $\hat{x}$ of $Rx = b$ can be made backward stable by iterative refinement.

ALGORITHM 9.1 (iterative refinement).

Set $\hat{x}_0 = \hat{x}$, $r = b - R\hat{x}_0$
repeat until $\|r\| \leq c_{37}\epsilon\|R\|\,\|\hat{x}\|$
  solve $\hat{L}\hat{L}^T\delta x = r$
  set $\hat{x}_i = \hat{x}_{i-1} + \delta x$
  $r = b - R\hat{x}_i$
endrepeat

10. Enhancing the robustness of the algorithm. We now suggest enhancements to further improve the robustness of the algorithm.

To begin with, carrying out the hyperbolic rotation as in (26) enforces the relation (28),

$$\hat{u}_{i+1}\hat{u}_{i+1}^T - \hat{v}_{i+1}\hat{v}_{i+1}^T = \Phi_i\hat{u}_i\hat{u}_i^T\Phi_i^T - \hat{v}_i\hat{v}_i^T - N_{i+1}, \tag{48}$$

where

$$\|N_{i+1}\| \leq c_{38}\epsilon(\|\hat{u}_{i+1}\|^2 + \|\Phi_i\|^2\|\hat{u}_i\|^2 + \|\hat{v}_{i+1}\|^2 + \|\hat{v}_i\|^2). \tag{49}$$

But the positive-definiteness of R further imposes conditions on the columns of the generator matrix. Indeed,

- for a diagonal and stable F, by Theorem A.1, a necessary condition for the positive-definiteness of the matrix is that we must have

  $$|\hat{u}_{i+1}| > |\hat{v}_{i+1}|, \tag{50}$$

  where the inequality holds componentwise;
- for a lower-triangular contractive F, Lemma B.2 shows that a necessary condition for positive-definiteness is

  $$\|u_i\| \geq \|v_i\|;$$

- in all cases, the condition $|\Phi_i u_i|_{i+1} > |v_i|_{i+1}$ is required to ensure that the reflection coefficient of the hyperbolic rotation Θ_i is less than 1.

We have found that if all these necessary conditions are enforced explicitly the algorithm is more reliable numerically. An example of this can be found in §10.3.

We now show how the OD and H methods can be modified to preserve the sign of the J-norm of each row of the prearray.

10.1. Enhancing the OD method. The OD method can be enhanced to preserve the sign of the J-norm of the row it is being applied to. For this purpose, assume that

$$|\Phi_i \hat{u}_i|_j > |\hat{v}_i|_j.$$

Then from (27) we see that if the jth row of the perturbed prearray has a positive J-norm then by adding a small perturbation to the jth row of the computed postarray we can guarantee a positive J-norm. If the jth row of the perturbed prearray does not have a positive J-norm, then in general there does not exist a small perturbation for the jth row of the postarray that will guarantee a positive J-norm. For such a row, the prearray must be perturbed to make its J-norm sufficiently positive and then the hyperbolic rotation must be reapplied by the OD method to that row. The new jth row of the postarray can now be made to have a positive J-norm by a small perturbation. The details are given in the algorithm below. For the case of a diagonal and stable F, all the rows of the prearray should have a positive J-norm. The algorithm should enforce this property.

In the statement of the algorithm, $\begin{bmatrix} x & y \end{bmatrix}$ stands for a particular row of the prearray $\bar{G}_{i+1}$, $\begin{bmatrix} \hat{x}_1 & \hat{y}_1 \end{bmatrix}$ stands for the corresponding row of the postarray $\hat{G}_{i+1}$, and Θ stands for the hyperbolic rotation. Here we are explicitly assuming that $|x| > |y|$, which is automatically the case when F is diagonal and stable. Otherwise, since $\begin{bmatrix} y & x \end{bmatrix}\Theta = \begin{bmatrix} y_1 & x_1 \end{bmatrix}$, the technique must be used with the elements of the input row interchanged.

ALGORITHM 10.1 (enhanced OD method).

Assumption: $|x| > |y|$.
if $|\hat{x}_1| < |\hat{y}_1|$
  $\gamma_1 \leftarrow c_7\epsilon(|\hat{x}_1| + |\hat{y}_1|)\text{sign}(\hat{x}_1)$
  $\gamma_2 \leftarrow c_7\epsilon(|\hat{x}_1| + |\hat{y}_1|)\text{sign}(\hat{y}_1)$
  if $|\hat{x}_1 + \gamma_1| > |\hat{y}_1 - \gamma_2|$ then
    $\hat{x}_1 \leftarrow \hat{x}_1 + \gamma_1$
    $\hat{y}_1 \leftarrow \hat{y}_1 - \gamma_2$
  else
    $\eta_1 \leftarrow c_6\epsilon(|x| + |y|)\text{sign}(x)$
    $\eta_2 \leftarrow c_6\epsilon(|x| + |y|)\text{sign}(y)$
    $\begin{bmatrix} \hat{x}_1 & \hat{y}_1 \end{bmatrix} \leftarrow \begin{bmatrix} x + \eta_1 & y - \eta_2 \end{bmatrix}\Theta$ (via the OD method)
    if $|\hat{x}_1| > |\hat{y}_1|$ then $\hat{x}_1 \leftarrow \hat{x}_1$ and $\hat{y}_1 \leftarrow \hat{y}_1$
    else
      $\gamma_1 \leftarrow c_7\epsilon(|\hat{x}_1| + |\hat{y}_1|)\text{sign}(\hat{x}_1)$
      $\gamma_2 \leftarrow c_7\epsilon(|\hat{x}_1| + |\hat{y}_1|)\text{sign}(\hat{y}_1)$
      $\hat{x}_1 \leftarrow \hat{x}_1 + \gamma_1$
      $\hat{y}_1 \leftarrow \hat{y}_1 - \gamma_2$
    endif

endif
endif

The computed columns $\hat{u}_{i+1}$ and $\hat{v}_{i+1}$ continue to satisfy a relation of form (27).

10.2. Enhancing the H procedure. Here again, $\begin{bmatrix} x & y \end{bmatrix}$ stands for a particular row of the prearray $\bar{G}_{i+1}$ and $\begin{bmatrix} \hat{x}_1 & \hat{y}_1 \end{bmatrix}$ stands for the corresponding row of the postarray. We shall again assume that $|x| > |y|$. If that is not the case then the procedure must be applied to $\begin{bmatrix} y & x \end{bmatrix}$, since $\begin{bmatrix} y & x \end{bmatrix} \Theta = \begin{bmatrix} y_1 & x_1 \end{bmatrix}$.

It follows from $|x| > |y|$ and relation (31) that

$$|\hat{x}_1 + e_1|^2 - |\hat{y}_1 + e_2|^2 > 0$$

for the H procedure. Therefore, by adding small numbers to $\hat{x}_1$ and $\hat{y}_1$ we can guarantee $|\hat{x}_1| > |\hat{y}_1|$.

ALGORITHM 10.2 (enhanced H method).

Assumption: $|x| > |y|$.
Apply the hyperbolic rotation Θ to $\begin{bmatrix} x & y \end{bmatrix}$ using the H procedure.
If $|\hat{x}_1| < |\hat{y}_1|$ then
  $\hat{y}_1 \leftarrow |\hat{x}_1|(1 - 3\epsilon)\text{sign}(\hat{y}_1)$

10.3. A numerical example. The following example exhibits a positive-definite matrix R for which a direct implementation of the Schur algorithm, without the enhancements and modifications proposed herein, breaks down. On the other hand, the modified Schur algorithm enforces positive-definiteness and avoids breakdown, as the example shows. The data is given in Appendix E.

A straightforward implementation of the generalized Schur algorithm (i.e., with a naive implementation of the hyperbolic rotation and the Blaschke matrix–vector multiply) breaks down at the 8th step and declares the matrix indefinite.

On the other hand, our implementation, using the enhanced H procedure (§10.2) and the enhanced Blaschke matrix–vector multiply (§6.3), successfully completes the matrix factorization and yields a relative error

$$\frac{\|R - \hat{L}\hat{L}^T\|}{\epsilon(1 - \|F\|^2)^{-2}\|R\|} \approx 0.15.$$

Furthermore, the relative backward error $\|R - \hat{L}\hat{L}^T\|/\|R\|$ is approximately 10^{-11} (using a machine precision of approximately 10^{-16}).

11. Summary of results. The general conclusion is the following.

The modified Schur algorithm is backward stable for a large class of structured matrices. Generally, it is as stable as can be expected from the analysis in §4.

More specifically, we have the following.

- If F is strictly lower triangular and contractive (e.g., $F = Z$), then the modified algorithm is backward stable with no generator growth.
- If F is stable, diagonal, and positive, then by reordering the entries of F in increasing order, there will be no generator growth and the algorithm will be as stable as can be expected from §4. In particular, it will be backward stable if $\|F\|$ is not too close to one (e.g., $\|F\|^2 \leq 1 - \frac{1}{n^2}$).

- In all other cases, we can use the technique outlined in §8.4 to control the generator growth and make the algorithm as stable as can be expected from §4. In particular, it is backward stable if $\|F\|$ is not too close to one (e.g., $\|F\|^2 \leq 1 - \frac{1}{n^2}$).
- If R is given or can be computed accurately (e.g., when F is diagonal), iterative refinement can be used to make the algorithm backward stable for the solution of linear equations.

As far as pivoting is concerned, in the diagonal F case, we emphasize that it is necessary only when $\|F\|$ is very close to one.

- If F is positive (or negative), a good strategy is to reorder the entries of F in increasing order of magnitude.
- If F has both positive and negative entries, then our numerical example of §8.1 indicates that pivoting may not help control the growth of the generators.

In our opinion, for positive-definite structured matrices, with diagonal or strictly lower-triangular F, the stabilization of the generalized Schur algorithm is critically dependent on the following:

- proper implementations of the hyperbolic rotations (using the OD or H procedures),
- proper evaluation of the Blaschke matrix–vector product,
- enforcement of positive-definiteness to avoid early breakdowns,
- control of the generator growth.

12. Concluding remarks. The analysis and results of this paper can be extended to positive-definite structured matrices with displacement rank larger than 2, as well as to other forms of displacement structure, say the Hankel-like case

$$FR + RF^T = GJG^T.$$

While the current analysis can also be extended to the bidiagonal F case, the error bound will further depend on the norm of the Blaschke matrices, which need not be smaller than one. Further improvements seem possible and will be discussed elsewhere.

We are currently pursuing the extension of our results to general nonsymmetric structured matrices. In these cases, the hyperbolic rotations are replaced by coupled rotations [16] and the OD and H procedures can be generalized to implement these rotations accurately.

Moreover, the results of this work suggest improvements to certain fast algorithms in adaptive filtering and state–space estimation in view of the connections of these algorithms to the Schur algorithm [15]. This is a subject of ongoing investigation.

The H procedure can also be extended to other elementary transformations like Gauss transforms and Givens rotations. The implications of this fact will be addressed elsewhere.

Appendix A. Schur functions. A function $s(z)$ that is analytic and strictly bounded by one (in magnitude) in the closed unit disc ($|z| \leq 1$) will be referred to as a Schur function. Such functions arise naturally in the study of symmetric positive-definite matrices R that satisfy (10) with a stable diagonal matrix F with distinct entries. Indeed, let $\{u_{i1}\}$ and $\{v_{i1}\}$ denote the entries of the generator column vectors

u_1 and v_1:

$$u_1 = \begin{bmatrix} u_{11} \\ u_{21} \\ \vdots \\ u_{n1} \end{bmatrix}, \quad v_1 = \begin{bmatrix} v_{11} \\ v_{21} \\ \vdots \\ v_{n1} \end{bmatrix}.$$

The following theorem guarantees the existence of a Schur function $s(z)$ that maps the u_{i1} to the v_{i1} [19, Thm. 2.1], [15, 17].

THEOREM A.1. *The matrix R that solves* (10) *with a stable diagonal matrix F with distinct entries is positive definite if and only if there exists a Schur function $s(z)$ such that*

$$v_{i1} = u_{i1} s(f_i). \tag{A.51}$$

It follows from (A.51) that $|v_{i1}| < |u_{i1}|$ and, consequently, that $\|v_1\| < \|u_1\|$.

Appendix B. Useful lemmas. The following two results are used in the body of the paper.

LEMMA B.1. *If $R - FRF^T = uu^T - vv^T$ then*

$$\|v\|^2 \leq \|u\|^2 + \|R\| \, (1 + \|F\|^2).$$

The following is an extension of a result in [3].

LEMMA B.2. *If $R - FRF^T = uu^T - vv^T$, R is a positive-definite matrix, and F is a contractive matrix, then*

$$\|v\|^2 \leq \|u\|^2.$$

Proof. Taking the trace of both sides of the displacement equation we get

$$\operatorname{tr}(R) - \operatorname{tr}(FRF^T) = \|u\|^2 - \|v\|^2.$$

Introduce the SVD of F: $F = U\Sigma V^T$ with $\Sigma_{ii} \leq 1$. Then $\operatorname{tr}(FRF^T) = \operatorname{tr}(\Sigma V^T R V \Sigma) \leq \operatorname{tr}(V^T R V) = \operatorname{tr}(R)$, from which the result follows. □

Appendix C. Error analysis of the OD method. In this section we analyze the application of a hyperbolic rotation Θ using its eigendecomposition $\Theta = QDQ^T$, where

$$Q = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix},$$

and

$$D = \begin{bmatrix} \sqrt{\frac{\alpha+\beta}{\alpha-\beta}} & 0 \\ 0 & \sqrt{\frac{\alpha-\beta}{\alpha+\beta}} \end{bmatrix}$$

for given number α and β such that $|\alpha| > |\beta|$.

Let $B = \Theta A$. We shall now show that

$$(\hat{B} + E_2) = \Theta(A + E_1),$$

where $\|E_2\| \leq c_{39}\epsilon\|\hat{B}\|$ and $\|E_1\| \leq c_{40}\epsilon\|A\|$.

First note that using the above expressions for D and Q, their entries can be computed to high componentwise relative accuracy (assuming that the square roots can be computed to high relative accuracy).

Next observe that, for any vector x,

$$fl((\hat{D}x)_j) = (Dx)_j(1 + c_{41}\delta_8).$$

Therefore, $\|\hat{D}x - Dx\| \leq c_{42}\epsilon\|Dx\|$. Also, note that

$$fl(\hat{Q}_{ij}x_j) = Q_{ij}x_j(1 + c_{43}\delta_9).$$

Hence, we can show that

$$fl(\hat{Q}x) = Q(x + e) = Qx + Qe = Qx + e_1,$$

where $\|e_1\| = \|e\| \leq c_{44}\epsilon\|x\| = c_{45}\epsilon\|fl(Qx)\|$.

Now, let $y = QDQ^Tx$. Then in finite precision we have

$$\begin{aligned} fl(\hat{Q}^Tx) &= Q^T(x + e_3), \\ fl(\hat{D}Q^T(x + e_3)) &= DQ^T(x + e_3) + e_4 \end{aligned}$$

and

$$\begin{aligned} fl(\hat{Q}(DQ^T(x + e_3) + e_4)) &= Q(DQ^T(x + e_3) + e_4) + e_5 \\ &= QDQ^T(x + e_1) - e_6 = \hat{y}. \end{aligned}$$

Thus,

$$\hat{y} + e_6 = QDQ^T(x + e_3) = \Theta(x + e_3),$$

where, by the bounds above,

$$\|e_3\| \leq c_{46}\epsilon\|x\|, \quad \|e_6\| \leq c_{47}\epsilon\|\hat{y}\|.$$

Appendix D. Matlab programs. We include here a Matlab listing of the stabilized Schur algorithm suggested in this paper. The program assumes that the input matrix is positive definite and tries to enforce it. The algorithm listed here can be easily modified to test if a structured matrix is positive definite.

D.1. The H procedure. Input data: The ratio *beta/alpha* represents the reflection coefficient, which is smaller than one in magnitude. Also, y/x is assumed smaller than one in magnitude.

Output data: The entries $\begin{bmatrix} x_1 & y_1 \end{bmatrix}$ that result by applying a hyperbolic rotation to $\begin{bmatrix} x & y \end{bmatrix}$ with $|x_1| > |y_1|$.

function $\begin{bmatrix} x1, & y1 \end{bmatrix} = h\_procedure(x, y, beta, alpha)$

```
c = (beta * y)/(alpha * x);
if c < 0.5
  xi = 1 - c;
else
```

```
    d1 = (abs(alpha) − abs(beta))/abs(alpha);
    d2 = (abs(x) − abs(y))/abs(x);
    xi = d1 + d2 − d1 * d2;
end
x1 = (abs(alpha) * x * xi)/sqrt((alpha − beta) * (alpha + beta));
y1 = x1 − sqrt((alpha + beta)/(alpha − beta)) * (x − y);
if abs(x1) < abs(y1)
  y1 = abs(x1) * (1 − 3 * eps) * sign(y1)
end
```

D.2. The Blaschke matrix–vector product. We now list the program that computes $\Phi_i u_i$ for both a diagonal F and a strictly lower-triangular F.

Input data: An $n \times n$ stable and diagonal matrix F, a vector u, a vector v (such that $|v| < |u|$), and an index i $(1 \leq i \leq n)$.

Output data: The matrix–vector product $z = \Phi_i u$, where $\Phi_i = (I - f_i F)^{-1}(F - f_i I)$, and the vector $ub = (I - f_i F)^{-1} u$.

function $[\ z,\ \ ub\ ] = blaschke\_1(F, u, v, i, n)$

```
ub = u;
z = u;
for j = i : n
    if F(i,i) * F(j,j) < 0.5
      xi = 1/(1 − F(j,j) * F(i,i));
    else
        d1 = 1 − abs(F(i,i));
        d2 = 1 − abs(F(j,j));
        xi = 1/(d1 + d2 − d1 * d2);
    end
    ub(j) = xi * z(j);
    z(j) = (F(j,j) − F(i,i)) * ub(j);
    if abs(z(j)) < abs(v(j))
      z(j) = abs(v(j)) * (1 + 3 * eps) * sign(z(j));
    end
end
```

For a strictly lower-triangular F we use the following.

Input data: An $n \times n$ strictly lower-triangular matrix F, a vector u, a vector v (such that $|v| < |u|$), and an index i $(1 \leq i \leq n)$.

Output data: The matrix–vector product $z = Fu$ and $ub = u$.

function $[\ z,\ \ ub\ ] = blaschke\_2(F, u, v, i, n)$

```
ub = u;
z = F * u;
z(i) = 0;
```

```
if abs(z(i + 1)) < abs(v(i + 1))
  z(i + 1) = abs(v(i + 1)) * (1 + 3 * eps) * sign(z(i + 1));
end
```

D.3. The stable Schur algorithm. We now list two versions of the stable modified Schur algorithm—one for diagonal stable F and the other for strictly lower-triangular F.

Input data: An $n \times n$ diagonal and stable matrix F, a generator $G = \begin{bmatrix} u & v \end{bmatrix}$ in proper form (i.e., $v_1 = 0$) with column vectors u, v.

Output data: A lower-triangular Cholesky factor L such that $\|R - LL^T\|$ satisfies (45).

function $L = stable\_schur\_1(u, v, F)$

```
n = size(F, 1);
for i = 1 : n − 1
    [ u,   ub ] = blaschke_1(F, u, v, i, n);
    L(:, i) = sqrt((1 − F(i, i)) * (1 + F(i, i))) * ub;
    a = v(i + 1);
    b = u(i + 1);
    for j = i + 1 : n
        [ u(j),   v(j) ] = h_procedure(u(j), v(j), a, b);
    end
    v(i + 1) = 0;
end
L(n, n) = (1/sqrt((1 − F(n, n)) * (1 + F(n, n)))) * u(n);
L(1 : n − 1, n) = zeros(n − 1, 1);
```

Input data: An $n \times n$ strictly lower-triangular matrix F, a generator $G = \begin{bmatrix} u & v \end{bmatrix}$ in proper form (i.e., $v_1 = 0$) with column vectors u, v.

Output data: A lower-triangular Cholesky factor L such that $\|R - LL^T\|$ satisfies (45).

function $L = stable\_schur\_2(u, v, F)$

```
n = size(F, 1);
for i = 1 : n − 1
    [ u,   ub ] = blaschke_2(F, u, v, i, n);
    L(:, i) = sqrt((1 − F(i, i)) * (1 + F(i, i))) * ub;
    a = v(i + 1);
    b = u(i + 1);
    for j = i + 1 : n
        if abs(u(j)) > abs(v(j))
            [ u(j),   v(j) ] = h_procedure(u(j), v(j), a, b);
        else
            [ temp_v,   temp_u ] = h_procedure(v(j), u(j), a, b);
            v(j) = temp_v; u(j) = temp_u;
```

```
        endif
    end
    v(i+1) = 0;
end
L(n,n) = (1/sqrt((1 - F(n,n)) * (1 + F(n,n)))) * u(n);
L(1 : n-1, n) = zeros(n-1, 1);
```

Appendix E. Example of breakdown.

$$G = \begin{bmatrix} 0.29256168393970 & 0 \\ 0.28263551029525 & -0.10728616660709 \\ 0.09633626413940 & 0.01541380240248 \\ 0.06797943459994 & -0.02572176567354 \\ 0.55275012712414 & 0.22069874528633 \\ 0.42631253478657 & 0.06821000412583 \\ 0.50468895704517 & 0.20125628531328 \\ 0.23936358366577 & -0.09527653751206 \\ 0.14608901804405 & 0.02337424345679 \end{bmatrix}, \quad F = \operatorname{diag} \begin{bmatrix} 0.40000000000000 \\ 0.97781078411630 \\ -0.00000000433051 \\ 0.97646762001746 \\ -0.99577002371173 \\ 0.00000001005313 \\ -0.99285659894698 \\ 0.99789820799463 \\ -0.00000001100000 \end{bmatrix}.$$

The matrix R that solves $R - FRF^T = GJG^T$ is positive definite since the entries of v were computed from

$$v_i = u_i s(f_i),$$

where $s(z)$ is the Schur function

$$s(z) = 0.4\,\frac{0.4 - z}{1 - 0.4z}.$$

Acknowledgments. The authors would like to thank Dr. Ming Gu for useful discussions and the anonymous referees and the Editor for their helpful comments.

REFERENCES

[1] S. T. Alexander, C.-T. Pan, and R. J. Plemmons, *Analysis of a recursive least-squares hyperbolic rotation algorithm for signal processing*, Linear Algebra Appl., 98 (1988), pp. 3–40.

[2] E. H. Bareiss, *Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices*, Numer. Math., 13 (1969), pp. 404–424.

[3] 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.

[4] 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.

[5] A. W. Bojanczyk and A. O. Steinhardt, *Matrix downdating techniques for signal processing*, in Proc. SPIE Conference on Advanced Algorithms and Architectures for Signal Processing, San Diego, CA, Society of Photo-optical Instrumentation Engineers, 1988, pp. 68–75.

[6] S. Chandrasekaran and M. Gu, *Notes on the PSVD and QSVD*, in preparation.

[7] G. Cybenko, *The numerical stability of the Levinson–Durbin algorithm for Toeplitz systems of equations*, SIAM J. Sci. Stat. Comput., 1 (1980), pp. 303–319.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed., The Johns Hopkins University Press, Baltimore, MD, 1989.

[9] M. Gu, *Stable and Efficient Algorithms for Structured Systems of Linear Equations*, Tech. report LBL-37690, Lawrence Berkeley Laboratory, University of California, Berkeley, CA, 1995.

[10] G. Heinig, *Inversion of generalized Cauchy matrices and other classes of structured matrices*, in Linear Algebra for Signal Processing, Vol. 69, A. Bojanczyk and G. Cybenko, eds., Springer-Verlag, New York, Berlin, 1995, pp. 63–81.

[11] M. JANKOWSKI AND M. WOZNIAKOWSKI, *Iterative refinement implies numerical stability*, BIT, 17 (1977), pp. 303–311.

[12] T. KAILATH, *A theorem of I. Schur and its impact on modern signal processing*, in Operator Theory: Advances and Applications, Vol. 18, I. Gohberg, ed., Birkhäuser, Boston, 1986, pp. 9–30.

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

[14] N. LEVINSON, *The Wiener r.m.s. (root-mean-square) error criterion in filter design and prediction*, J. Math. Phys., 25 (1947), pp. 261–278.

[15] A. H. SAYED, *Displacement Structure in Signal Processing and Mathematics*, Ph.D. thesis, Stanford University, Stanford, CA, 1992.

[16] ———, *Time-variant structured matrices: An application to instrumental variable methods*, in Proc. SPIE Conference on Advanced Signal Processing: Algorithms, Architectures, and Implementations, San Diego, CA, 1994, Society of Photo-optical Instrumentation Engineers, pp. 516–527.

[17] A. H. SAYED, T. CONSTANTINESCU, AND T. KAILATH, *Square-Root Algorithms for Structured Matrices, Interpolation, and Completion Problems*, IMA Volumes in Mathematics and Its Applications, Vol. 69, Springer-Verlag, 1995, pp. 153–184. Linear Algebra for Signal Processing, A. Bojanczyk and G. Cybenko, eds.

[18] A. H. SAYED AND T. KAILATH, *Fast algorithms for generalized displacement structures and lossless systems*, Linear Algebra Appl., 219 (1995), pp. 49–78.

[19] A. H. SAYED, T. KAILATH, H. LEV-ARI, AND T. CONSTANTINESCU, *Recursive solutions of rational interpolation problems via fast matrix factorization*, Integral Equations Operator Theory, 20 (1994), pp. 84–118.

[20] I. SCHUR, *Über potenzreihen die im Inneren des Einheitskreises beschränkt sind*, Journal für die Reine und Angewandte Mathematik, 147 (1917), pp. 205–232. (English translation in *Operator Theory: Advances and Applications*, Vol. 18, I. Gohberg, ed., Birkhäuser, Boston, 1986, pp. 31–88.)

[21] M. STEWART AND P. VAN DOOREN, *Stability Issues in the Factorization of Structured Matrices*, SIAM J. Matrix Anal. Appl., to appear.

[22] D. R. SWEET, *Numerical Methods for Toeplitz Matrices*, Ph.D. thesis, University of Adelaide, Adelaide, Australia, 1982.