SIAM J. Matrix Anal. Appl.
Vol. 18, No. 1, pp. 104–118, January 1997

STABILITY ISSUES IN THE FACTORIZATION OF STRUCTURED MATRICES*

MICHAEL STEWART† AND PAUL VAN DOOREN‡

Abstract. This paper provides an error analysis of the generalized Schur algorithm of Kailath and Chun [*SIAM J. Matrix Anal. Appl.*, 15 (1994), pp. 114–128]—a class of algorithms which can be used to factorize Toeplitz-like matrices, including block-Toeplitz matrices, and matrices of the form T^TT, where T is Toeplitz. The conclusion drawn is that if this algorithm is implemented with hyperbolic transformations in the factored form which is well known to provide numerical stability in the context of Cholesky downdating, then the generalized Schur algorithm will be stable. If a more direct implementation of the hyperbolic transformations is used, then it will be unstable. In this respect, the algorithm is analogous to Cholesky downdating; the details of implementation of the hyperbolic transformations are essential for stability. An example which illustrates this instability is given. This result is in contrast to the ordinary Schur algorithm for which an analysis by Bojanczyk, Brent, De Hoog, and Sweet [*SIAM J. Matrix Anal. Appl.*, 16 (1995), pp. 40–57] shows that the stability of the algorithm is not dependent on the implementation of the hyperbolic transformations.

Key words. Schur algorithm, structured matrices, Toeplitz matrices, stability

AMS subject classifications. 65F05, 65F30, 15A06, 15A23

PII. S089547989528692X

1. Introduction.

The Schur algorithm is a popular and fast method for the Cholesky factorization of a square, positive-definite Toeplitz matrix T. It performs reliably, and in [3] it was shown to be stable in the sense that if the algorithm runs to completion and $\hat{C}$ is the computed Cholesky factor, $\|T - \hat{C}^T\hat{C}\|$ is guaranteed to be small. This paper will perform a similar stability analysis which applies to several special cases of the generalized Schur algorithm [10]. In its full generality, the generalized Schur algorithm can be adapted to the factorization of a wide variety of structured matrices. The analysis given here is primarily of interest for block-Toeplitz and Toeplitz-block matrices, as well as for matrices of the form T^TT, where T is rectangular and Toeplitz. The key notion behind the general algorithm is the concept of displacement rank [10].

One of the most significant examples is the Cholesky factorization of T^TT. This factor is also the factor R in the QR factorization of the rectangular Toeplitz matrix T, and the obvious application of this fact to the solution of Toeplitz least squares problems is explored in [1]. However, the analysis given there assumes the use of the algorithm presented in [2] rather than the generalized Schur algorithm. The basic idea is to obtain R without bothering about finding Q, thus avoiding any problems associated with the loss of orthogonality which are common to all fast Toeplitz QR algorithms. The method of seminormal equations, possibly with iterative refinement, can then be used to find the least squares solution. The resulting equations are $R^TRx = T^Tb$ and a weak stability result can be given concerning their solution provided that for the computed R, $\|R^TR - T^TT\|$ is kept small. As will be shown

* Received by the editors May 30, 1995; accepted for publication (in revised form) by N. J. Higham January 9, 1996. This work was supported by ARPA grant 60NANB2D1272 and NSF grant CCR-9209349.

http://www.siam.org/journals/simax/18-1/28692.html

† Coordinated Science Laboratory, University of Illinois, Urbana, IL 61801 (stewart@monk.csl.uiuc.edu).

‡ Department of Mathematical Engineering, Université Catholique de Louvain, Louvain-la-Neuve, Belgium (vandooren@anma.ucl.ac.be).

here, this can be done using the generalized Schur algorithm as well as the algorithm in [2]. Consequently, the observations concerning Toeplitz least squares problems in [1] can be adapted to an alternate approach—that of finding R from the generalized Schur algorithm and then solving the seminormal equations to obtain the solution.

A comparison of the analysis in this paper can also be made with the analysis of the Schur algorithm given in [3]. Despite some major similarities, the conclusions of this paper will be, in one very important respect, quite different: the implementation details of the Schur algorithm are less critical to stability than those of the generalized Schur algorithm. Both can be implemented using hyperbolic transformations of the form

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

with $|\rho| < 1$ and where the transformations are applied with straightforward matrix multiplication; however there are reasons to be concerned about the stability of this approach. In many algorithms, such as Cholesky downdating [4], this is not a good idea. A downdating algorithm based on such transformations will not be stable unless the transformations are implemented in factored form,

$$H = \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{1-\rho^2}} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & \rho \\ 0 & 1 \end{bmatrix}.$$

It is worth noting that the manner in which the elements of these factors are computed is also important for stability. Details can be found in [4] and a correct implementation can be found in section 3. Such a seemingly minor difference in implementation makes the difference between stability and instability for Cholesky downdating, but the Schur algorithm is more robust—it is stable using either approach. Unfortunately, the generalized Schur algorithm will be shown to be analogous to the downdating problem in this regard; the factored transformations are essential for stability. A proof of the stability of the factored hyperbolic approach will be given in section 4 while an example which illustrates the instability of the other approach will be given in section 5. The stability proof in [1] also requires use of the factored hyperbolic transformations. This is not surprising since the presentation in [2] recasts the problem in the form of a downdating problem.

Finally, in section 5, some comments will be made about the nature of the instability. The propagation of errors is essentially stable regardless of which implementation is used; the difference is in the size of the local errors. From the fact that the error propagation is stable, it is possible to show that there will be many instances in which the instability does not completely destroy the accuracy of the computed Cholesky factors. This is true even for quite ill-conditioned problems. Numerical examples will be given to support this claim.

2. Displacement rank and stable computation of the initial generators.

Given an $n \times n$ symmetric, positive-definite matrix A, which is not necessarily Toeplitz, we define the displacement of A to be

$$D_A = A - Z_A A Z_A^T, \tag{1}$$

where Z_A is strictly lower triangular (a matrix with zeros on the diagonal). The restriction of symmetry on A can easily be relaxed to allow A to be Hermitian, but for simplicity we deal only with the real symmetric case here.

In [10], the only additional restriction on Z is that it be a fast, $O(n)$ rather than $O(n^2)$, process to multiply a vector by Z. Otherwise, the factorization algorithm will have a complexity of $O(n^3)$ instead of $O(n^2)$. In addition to this, we will also make two assumptions which will guarantee the stability of the algorithm and simplify the analysis. For stability it will be necessary to assume that $\|Z\|_2 \leq 1$. Otherwise, repeated multiplication by Z will magnify errors. Further, to simplify the analysis, it will be assumed that a vector can be multiplied by Z without incurring any error. These two assumptions essentially limit Z to be a shift or a block shift. Since these are the forms that Z takes for Toeplitz least squares problems and block-Toeplitz Cholesky factorization, respectively, these assumptions are reasonable, even if they do remove a good deal of generality. The second assumption is not essential if the multiplication is done in a stable manner, but there don't seem to be any common examples to which the analysis could be made to apply by dropping it.

To be more specific about typical examples of Z, if A is square and Toeplitz or has the form T^TT for some rectangular Toeplitz matrix T, we choose $(Z_T)_{ij} = 1$ if $i = j + 1$ and $(Z_T)_{ij} = 0$ otherwise. A block-Toeplitz matrix B would require Z_B to be a block shift, with the ones on the subdiagonal replaced with identity matrices. The significance of the analysis here will be for algorithms based on displacements involving Z_T and Z_B.

It is well known that if T is a symmetric Toeplitz matrix, normalized to have ones on the diagonal, $(T)_{ij} = t_{i-j}$ with $t_i = t_{-i}$ and $t_0 = 1$, then D_T will be an indefinite rank 2 matrix and

$$D_T = T - Z_T T Z_T^T = G_T^T \Sigma_T G_T,$$

where

$$G_T = \begin{bmatrix} 1 & t_1 & \cdots & t_{n-1} \\ 0 & t_1 & \cdots & t_{n-1} \end{bmatrix} = \begin{bmatrix} u^T \\ v^T \end{bmatrix}$$

and $\Sigma_T = 1 \oplus -1$.

The vectors u and v are referred to as *generators* of T. Clearly, the matrix T is uniquely determined by these generators. Performing operations on the generators rather than on T directly is what allows efficient $O(n^2)$ algorithms for Toeplitz systems.

Assume that for some Z_A satisfying the previously specified restrictions A belongs to a class of positive-definite matrices for which D_A as defined by equation (1) has rank α. Clearly D_A will have a decomposition of the form

$$D_A = G^T \Sigma_A G,$$

where

$$G = \begin{bmatrix} u_{11} & u_{12}^T \\ u_{21} & U_{22} \\ v_{11} & v_{12}^T \\ v_{21} & V_{22} \end{bmatrix}$$

and $\Sigma_A = I_p \oplus I_q$. Here we take u_{11} and v_{11} to be scalars and u_{21} and v_{21} to be column vectors of size $p - 1$ and $q - 1$, respectively, with $p + q = \alpha$.

In section 3, we will give a summary of the general algorithm. More details and many special cases can be found in [10]. But first, since the numerical stability of the overall factorization will depend on the stability of the process of finding the generators for A, we will briefly discuss how this can be done in a stable manner.

As already seen, the generators of a Toeplitz matrix can be found trivially with no error. For a symmetric block-Toeplitz matrix, the process is only slightly more complicated and the errors in the initial generators will not cause a problem with stability. The generators for a Toeplitz least squares problem can also be found in a reliable manner with a minimal amount of computation, and again the process is stable.

In general, the numbers p and q correspond, respectively, to the number of positive and negative eigenvalues of D. When this decomposition cannot be obtained trivially, as in the Toeplitz or block-Toeplitz case, it can be obtained through an eigenvalue decomposition or through Gaussian elimination with a symmetric pivoting strategy to obtain an LDL^T factorization of the displacement [7].

Of course, computing a full eigenvalue decomposition will slow down the overall algorithm. In the absence of any specific knowledge of the form which the generators will take, and in the case in which D can be computed with $O(n^2)$ complexity, computing α steps of the LDL^T decomposition will give a set of generators without destroying the $O(n^2)$ complexity of the algorithm. However, it is important to note that when truncating such a decomposition, the pivoting scheme which is chosen can be more critical to stability than when the decomposition is carried out completely. The fact that the Bunch–Parlett strategy is a backward stable method for computing LDL^T is well known [7]. The stability of the Bunch–Kaufman procedure has been established more recently in [9]. However, it is simple to verify numerically that the rank 2 matrix

$$\begin{bmatrix} \epsilon & \sqrt{\epsilon} & \epsilon \\ \sqrt{\epsilon} & \frac{\beta^2}{4} & \frac{\beta}{2} \\ \epsilon & \frac{\beta}{2} & \frac{\beta^2\epsilon - 4\beta\sqrt{\epsilon} + \beta^2}{\beta^2 - 4} \end{bmatrix},$$

where $\beta = (1+\sqrt{17})/8$ is a constant chosen to minimize element growth for symmetric pivoting and $\epsilon > 0$ is very small, leads to a large error if two steps of the Bunch–Kaufman approach are used to obtain a rank 2 LDL^T factorization. The Bunch–Parlett procedure will not have this problem. The reason for the difference can be found by looking at the sensitivity of the scalar Schur complement which is truncated by these two algorithms. An analysis of pivoted low rank Cholesky factorization which highlights the relevance of Schur complement sensitivity in the case of semidefinite matrices was given in [8]; the issues are the same for indefinite matrices, and the extension to the more complicated pivoting strategies is direct. The result can be summarized as follows: backward stability guarantees that if the computed Schur complement after two stages of the factorization process is small, then no large errors will be incurred in dropping it and terminating the factorization. Rank deficiency ensures that the exact value of the Schur complement will be zero, but if it is highly sensitive to perturbations in the original matrix, then the backward error can cause the computed value to become large. It is shown in [8] that the sensitivity of the Schur complement will depend on the size of the elements in L. A careful scrutiny of the two pivoting strategies shows that, although both are backward stable and give bounds on the norms of the Schur complements, only Bunch–Parlett gives bounds on the elements of L. However, this digression is included only for completeness to show that the process of obtaining the generators can always be done in a stable fashion. In practice, there is generally a simpler means of obtaining the generators—

for Toeplitz matrices, they can be obtained with no computation at all. A general factorization method would only be needed in the rare case in which D does not have a zero structure which makes the choice of generators obvious, or at least easy to compute. An example is the case of general non-Toeplitz matrices for which α equals two. The class of matrices for which $\alpha = 2$ is broader than the class of Toeplitz matrices, and D_A will not always have a zero structure which makes the process of finding the generators trivial.

3. The generalized Schur algorithm. Assuming that we have managed to find the generators for a structured matrix A, we can apply the generalized Schur algorithm to find a factorization $A = C^T C$. Much of the power and generality of this approach is due to the fact that it is a straightforward and intuitively clear generalization of the Schur algorithm; there is little essential difference between the cases $\alpha = 2$ and $\alpha > 2$. Since the former case is well known, the presentation here of the latter case can be kept brief. A more leisurely description can be found in [10].

Let $Z = Z_A$ and $Z_j = Z_A^j$. Since Z is strictly lower triangular, $Z_n = 0$, and from this it follows that

$$A = \sum_{j=0}^{n-1} Z_j (A - ZAZ^T) Z_j^T$$

or

$$A = \sum_{j=0}^{n-1} Z_j G^T \Sigma G Z_j^T. \tag{2}$$

For any transformation J such that $J^T \Sigma J = \Sigma$, we find that JG is also a set of generators for A. This follows immediately, since $G^T J^T \Sigma J G = G^T \Sigma G = D$.

From equation (2) and the positivity of A, we see that

$$0 < (A)_{1,1} = \left\| \begin{bmatrix} u_{11} \\ u_{12} \end{bmatrix} \right\|^2 - \left\| \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} \right\|^2 .$$

This means that if a transformation J_1 of the form

$$J_1 = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix},$$

where Q_1 and Q_2 are orthogonal, is applied to G to give

$$J_1 G = \begin{bmatrix} \hat{u}_{11} & \hat{u}_{12}^T \\ 0 & \hat{U}_{22} \\ \hat{v}_{11} & v_{12}^T \\ 0 & \hat{V}_{22} \end{bmatrix}$$

then $|\hat{u}_{11}| > |\hat{v}_{11}|$. Therefore it is possible to take $\rho = -\hat{v}_{11}/\hat{u}_{11}$ so that

$$J_2 J_1 G = \frac{1}{\sqrt{1-\rho^2}} \begin{bmatrix} 1 & 0 & \rho & 0 \\ 0 & I & 0 & 0 \\ \rho & 0 & 1 & 0 \\ 0 & 0 & 0 & I \end{bmatrix} J_1 G = \begin{bmatrix} \tilde{u}_{11} & \tilde{u}_{12}^T \\ 0 & \hat{U}_{22} \\ 0 & \tilde{v}_{12}^T \\ 0 & \hat{V}_{22} \end{bmatrix}.$$

Note that $J_1^T \Sigma J_1 = \Sigma$ and $J_2^T \Sigma J_2 = \Sigma$ so that $J_2 J_1 G$ is also a set of generators for A. We have proven that we can always find a set of generators for which the first column has only one nonzero element. Such generators are said to be proper.

This form of the generators is very useful. Equation (2) implies that the first row of A is not different from the first row of $G^T \Sigma G$. When the generators are in proper form, this means that

$$A(1,:) = \tilde{u}_{11} \begin{bmatrix} \tilde{u}_{11} & \tilde{u}_{12}^T \end{bmatrix}.$$

Clearly, the first row of the Cholesky factor of A will be

$$C(1,:) = \begin{bmatrix} \tilde{u}_{11} & \tilde{u}_{12}^T \end{bmatrix}.$$

If we let

$$A_S = A - \begin{bmatrix} \tilde{u}_{11} \\ \tilde{u}_{12} \end{bmatrix} \begin{bmatrix} \tilde{u}_{11} & \tilde{u}_{12}^T \end{bmatrix}$$

be the Schur complement of A, we see that

$$A_S - Z A_S Z^T = G^T \Sigma G - \begin{bmatrix} \tilde{u}_{11} \\ \tilde{u}_{12} \end{bmatrix} \begin{bmatrix} \tilde{u}_{11} & \tilde{u}_{12}^T \end{bmatrix} + Z \begin{bmatrix} \tilde{u}_{11} \\ \tilde{u}_{12} \end{bmatrix} \begin{bmatrix} \tilde{u}_{11} & \tilde{u}_{12}^T \end{bmatrix} Z^T.$$

This means that merely postmultiplying the first row of the generators

$$u^T = \begin{bmatrix} \tilde{u}_{11} & \tilde{u}_{12}^T \end{bmatrix}$$

by Z^T will give the generators of the Schur complement.

Additional unitary and hyperbolic transformations can now be used to introduce zeros into the second elements of these vectors to obtain the first row of the Cholesky factor of the Schur complement—the second row of the Cholesky factor of A. The process can be continued recursively to obtain the complete Cholesky factor of A. At each stage, the positivity of the Schur complements guarantees that an appropriate ρ can be found.

The resulting algorithm, with the hyperbolic transformations implemented in the stable form in MATLAB code, is as follows:

```
function [C]=gschur(G,n,p,q)
C=zeros(n,n); C(1,1:n)=G(1,1:n);
for i=1:n,
  for j=2:p
    Q=givens(G(1,i),G(j,i));
    G(1:j-1:j,i:n)=Q*G(1:j-1:j,i:n);
  end
  for j=2:q
    Q=givens(G(p+1,i),G(p+j,i));
    G(p+1:j-1:p+j,i:n)=Q*G(p+1:j-1:p+j,i:n);
  end
  s=G(p+1,i)/G(1,i);
  c=sqrt(G(1,i)^2-G(p+1,i)^2)/G(1,i);
  G(1,i+1:n)=([1 -s]*G(1:p:p+1,i+1:n)/c;
  G(p+1,i+1:n)=[-s c]*G(1:p:p+1,i+1:n);
  G(1,i)=sqrt(G(1,i)^2-G(p+1,i)^2);
  C(i,i:n)=G(1,i:n);
  G(1,i+1:n)=G(1,i:n-1);
end;
```

It was mentioned in section 1 that the computation of the elements in the factors is significant for stability. It is also important that the leading element of the first generator be computed separately, as shown here. The analysis in [4] does not apply to other implementations.

Although the algorithm given above will suffice to find the factors of a block-Toeplitz matrix, it is significantly different from some of the approaches given in the literature. The numerical properties are not different, but a more block-oriented perspective on block-Toeplitz factorization can be found in [6].

In the two-generator case, the matrix J_2 which zeros the first elements of all the generators except the first is almost uniquely defined by the constraints

$$H\begin{bmatrix} u_{11} \\ v_{11} \end{bmatrix} = \begin{bmatrix} x \\ 0 \end{bmatrix}$$

and $H^T\Sigma H = \Sigma$. The only possible variation is that each row of H can be multiplied by -1. This is not the case when there are more generators. The obvious example is the possibility of applying an arbitrary hyperbolic transformation acting on the last $\alpha - 1$ generators after the appropriate zeros have been introduced into their leading elements. This paper will show that when J is formed from a block diagonal orthogonal matrix and an appropriately implemented hyperbolic transformation, the algorithm will be stable.

4. Stability analysis.

4. Stability analysis. A stability analysis of the Schur algorithm for Toeplitz factorization is presented in [3]. The analysis can be broken into two parts: one which shows how local errors propagate through the algorithm and one which bounds the local errors. The propagation of errors is essentially the same for the generalized Schur algorithm, but the problem of bounding local errors is slightly more difficult. We will first modify results from [3] to apply to the general algorithm and then later develop new inequalities which will complete the analysis.

At the end of the kth stage of the algorithm, let the generators stored in the computer be

$$\tilde{G}_k = \begin{bmatrix} 0_{k-1}^T & \tilde{u}_{11,k} & \tilde{u}_{12,k}^T \\ 0_{p-1,k-1} & 0_{p-1} & \tilde{U}_{22,k} \\ 0_{k-1}^T & 0 & \tilde{v}_{12,k}^T \\ 0_{q-1,k-1} & 0_{q-1} & \tilde{V}_{22,k} \end{bmatrix} = \begin{bmatrix} \tilde{u}_{1,k}^T \\ \tilde{U}_{2,k} \\ \tilde{v}_{1,k}^T \\ \tilde{V}_{2,k} \end{bmatrix}.$$

Using MATLAB notation, let $\tilde{G}_{k,Z}$ be defined by

$$\tilde{G}_{k,Z}(1,:) = \tilde{G}_k(1,:)Z^T$$

and

$$\tilde{G}_{k,Z}(2:\alpha,:) = \tilde{G}_k(2:\alpha,:).$$

Also, let G_k and $G_{k,Z}$ be the generators which would result from exact computations.

The computed generators will satisfy

$$\tilde{G}_{k+1}^T \Sigma \tilde{G}_{k+1} = \tilde{G}_{k,Z}^T \Sigma \tilde{G}_{k,Z} + \epsilon F_k + O(\epsilon^2),$$

where ϵ is the machine precision and ϵF_k are errors incurred locally in computing the new form of the generators at step k through the use of orthogonal and hyperbolic transformations.

Since it is possible that G_1 will already be in proper form, we will assume that it is and treat any errors which appear in it separately from the other errors. Summing this equation from $k = 1$ to $k = n-1$ and grouping the terms related to the top rows of $\tilde{G}_k$ and $\tilde{G}_{k,Z}$ in the left-hand side gives

$$\sum_{k=1}^{n-1} \left(\tilde{u}_{1,k+1}\tilde{u}_{1,k+1}^T - Z\tilde{u}_{1,k}\tilde{u}_{1,k}^T Z^T \right) = \sum_{k=1}^{n-1} \Big((\tilde{G}_{k,Z}(2:\alpha,:))^T \Sigma (\tilde{G}_{k,Z}(2:\alpha,:)) \\ - (\tilde{G}_{k+1}(2:\alpha,:))^T \Sigma (\tilde{G}_{k+1}(2:\alpha,:)) + \epsilon F_k \Big) + O(\epsilon^2).$$

After simplifying the terms that cancel in the right-hand side, we obtain

$$\tilde{R}^T\tilde{R} - \tilde{u}_{1,1}\tilde{u}_{1,1}^T - Z^T\tilde{R}^T\tilde{R}Z + Z\tilde{u}_{1,n}\tilde{u}_{1,n}^T Z^T = (\tilde{G}_1(2:\alpha,:))^T \Sigma (\tilde{G}_1(2:\alpha,:)) \\ - (\tilde{G}_n(2:\alpha,:))^T \Sigma (\tilde{G}_n(2:\alpha,:)) + \epsilon \sum_{k=1}^{n-1} F_k + O(\epsilon^2),$$

where

$$\tilde{R} = \begin{bmatrix} \tilde{u}_{1,1}^T \\ \tilde{u}_{1,2}^T \\ \vdots \\ \tilde{u}_{1,n}^T \end{bmatrix}.$$

Since $G_{n,Z} = 0$,

$$\tilde{R}^T\tilde{R} - Z^T R^T R Z = \tilde{u}_{1,1}\tilde{u}_{1,1}^T + (\tilde{G}_1(2:\alpha,:))^T \Sigma (\tilde{G}_1(2:\alpha,:)) + \epsilon \sum_{k=1}^{n-1} F_k + O(\epsilon^2).$$

Using this to find the displacement

$$(A - \tilde{R}^T\tilde{R}) - Z(A - \tilde{R}^T\tilde{R})Z^T = G_1^T \Sigma G_1 - \tilde{G}_1^T \Sigma \tilde{G}_1 + \epsilon \sum_{k=1}^{n-1} F_k + O(\epsilon^2)$$

and applying equation (2) gives

$$(3) \qquad A - \tilde{R}^T\tilde{R} = \sum_{j=0}^{n-1} Z_j [G_1^T \Sigma G_1 - \tilde{G}_1^T \Sigma \tilde{G}_1] Z_j^T - \epsilon \sum_{j=0}^{n-1} \sum_{k=1}^{n-j-1} Z_j F_k Z_j^T + O(\epsilon^2),$$

where the sum over k has been reduced by noting that $Z_j^T F_k Z_j^T = 0$ whenever $k > n - j - 1$. This shows that if we can bound errors in the computation of the initial generators and the local errors, then the algorithm will be stable. The errors in the initial generators are not a problem, since from D the generators can be obtained in a backward stable manner using Bunch–Parlett pivoting to compute LDL^T or, more typically, in a more direct fashion. Since the methods for obtaining the generators may vary, in the analysis and error bounds which follow we will ignore this source of error and only concern ourselves with the effects of local errors due to the unitary and hyperbolic transformations.

The local errors are given by the expression

$$\epsilon F_k = \tilde{G}_{k+1}^T \Sigma \tilde{G}_{k+1} - \tilde{G}_{k,Z}^T \Sigma \tilde{G}_{k,Z} + O(\epsilon^2).$$

Because any bounds on the errors produced by the transformations will depend on the norm of the generators, it is essential to bound the generators.

THEOREM 4.1. *When the generators are computed by applying a sequence of plane rotations and a hyperbolic transformation, they satisfy*

$$\|G_k\|_F^2 \leq 2\sqrt{k-1}\|A\|_F + \|G_1\|_F^2.$$

Proof. Let $\hat{u}_1$ and $\hat{v}_1$ be the two generators u_1 and v_1 after orthogonal transformations have been performed on the generators from step $k-1$. Then

$$\begin{aligned}
\begin{bmatrix} u_{1,k}^T \\ v_{1,k}^T \end{bmatrix} &= \frac{1}{\sqrt{1-\rho^2}} \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \begin{bmatrix} \hat{u}_1^T \\ \hat{v}_1^T \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{1-\rho^2}} & \frac{\rho}{\sqrt{1-\rho^2}} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{u}_1^T \\ \hat{v}_1^T \end{bmatrix} \\
&= \begin{bmatrix} u_{1,k}^T \\ \rho u_{1,k}^T \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} \hat{u}_1^T \\ \hat{v}_1^T \end{bmatrix}.
\end{aligned}$$

Taking Frobenius norms gives

$$\begin{aligned}
\left\| \begin{bmatrix} u_{1,k}^T \\ v_{1,k}^T \end{bmatrix} \right\|_F^2 &= (1+\rho^2)\|u_{1,k}\|^2 + (1-\rho^2)\|\hat{v}_1\|^2 + 2\rho\sqrt{1-\rho^2}u_{1,k}^T\hat{v}_1 \\
&\leq \|u_{1,k}\|^2 + (|\rho|\|u_{1,k}\| + \sqrt{1-\rho^2}\|\hat{v}_1\|)^2 \\
&\leq 2\|u_{1,k}\|^2 + \|v_1\|^2.
\end{aligned}$$

To see why the final inequality is true, let $x = |\rho|$ and look for the minimum value of

$$f(x) = \|u_{1,k}\| + (x\|u_{1,k}\|^2 + \sqrt{1-x^2}\|\hat{v}_1\|)^2$$

on $0 \leq x \leq 1$. If

$$0 = f'(x) = 2\left(x\|u_{1,k}\| + \sqrt{1-x^2}\|\hat{v}_1\|\right)\left(\|u_{1,k}\| - \frac{x}{\sqrt{1-x^2}}\|\hat{v}_1\|\right),$$

then

$$x = \frac{\|u_{1,k}\|}{\sqrt{\|u_{1,k}\|^2 + \|\hat{v}_1\|^2}}$$

and

$$f(x) = 2\|u_{1,k}\|^2 + \|\hat{v}_1\|^2.$$

It is easy to see that f assumes lower values at $x = 0$ and $x = 1$, so this point is the only possible maximum. Since the plane rotations do not affect the norm of the generators and since $\|Z\|_2 \le 1$,

$$\|G_k\|_F^2 \le 2\|u_{1,k}\|^2 + \|G_{k-1}\|_F^2.$$

This inequality can be expanded recursively to give

$$\|G_k\|_F^2 \le 2\sum_{j=2}^{n} \|u_{1,j}\|^2 + \|G_1\|_F^2 = 2\|C(2:k,:)\|_F^2 + \|G_1\|_F^2.$$

Finally, for an arbitrary positive semidefinite rank $k-1$ matrix with a factorization $A = C^T C$,

$$\|C\|_F^2 \le \sqrt{k}\|A\|_F.$$

This follows from the fact that the Frobenius norm squared equals the sum of squares of the singular values and from a standard inequality relating the vector 2-norm and the vector 1-norm. This completes the proof of the theorem.

To complete a stability analysis all that is needed is to show that the orthogonal and hyperbolic transformations produce a local error, ϵF_k, which is proportional to the norm of the generators. Note that this does not necessarily refer to the norm of the generators prior to the transformation. In fact, in the case of the hyperbolic transformation, it is necessary to look at the norm of one of the generators which is produced by the hyperbolic transformation.

An error analysis of hyperbolic transformations is given in [4]. The result is that if the transformations are applied in factored form

$$(4)\qquad \frac{1}{\sqrt{1-\rho^2}}\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix}\begin{bmatrix} \frac{1}{\sqrt{1-\rho^2}} & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & \rho \\ 0 & 1 \end{bmatrix}$$

then

$$(5)\qquad \begin{bmatrix} \tilde{u}_{1,k+1}^T + \widehat{\Delta u}^T \\ \tilde{v}_{1,k+1}^T \end{bmatrix} = H \begin{bmatrix} \hat{u}_{1,k}^T \\ \hat{v}_{1,k}^T + \widehat{\Delta v}^T \end{bmatrix}.$$

The mixed error vectors $\widehat{\Delta u}$ and $\widehat{\Delta v}$ satisfy

$$(6)\qquad \left\| \begin{bmatrix} \widehat{\Delta u}^T \\ \widehat{\Delta v}^T \end{bmatrix} \right\|_F \le 6.25\epsilon \left\| \begin{bmatrix} \tilde{u}_{1,k+1}^T \\ \hat{v}_{1,k}^T \end{bmatrix} \right\|_F,$$

where ϵ is the unit roundoff.

In addition to this, there is a result in [11] concerning the application of plane rotations. In particular, there will exist orthogonal $\hat{Q}_1$ and $\hat{Q}_2$ such that

$$(7)\qquad \left[\begin{array}{c|c} \hat{Q}_1 & 0 \\ \hline 0 & \hat{Q}_2 \end{array}\right] \left[\begin{array}{c} \tilde{u}_{1,k}^T Z^T + \Delta u_1 \\ \hat{U}_{2,k} + \Delta U_2 \\ \hline \tilde{v}_{1,k}^T + \Delta v_1 \\ \hat{V}_{2,k} + \Delta V_2 \end{array}\right] = \begin{bmatrix} \hat{u}_{1,k}^T \\ \hat{U}_{2,k} \\ \hat{v}_{1,k}^T \\ \hat{V}_{2,k} \end{bmatrix} = \hat{G}_k,$$

where for $m \doteq \max\{p-1, q-1\}$

$$\left\| \begin{bmatrix} \Delta u_1^T \\ \Delta U_2 \end{bmatrix} \right\|_F \le 6m\epsilon \left\| \begin{bmatrix} \tilde{u}_{1,k}^T Z^T \\ \tilde{U}_{2,k} \end{bmatrix} \right\|_F \tag{8}$$

and

$$\left\| \begin{bmatrix} \Delta v_1^T \\ \Delta V_2 \end{bmatrix} \right\|_F \le 6m\epsilon \left\| \begin{bmatrix} \tilde{v}_{1,k}^T Z^T \\ \tilde{V}_{2,k} \end{bmatrix} \right\|_F . \tag{9}$$

If we let

$$\Delta G_k = \begin{bmatrix} \Delta u_1 & \Delta U_2^T & \Delta v_1 & \Delta V_2^T \end{bmatrix}^T$$

then clearly

$$\|\Delta G_k\|_F \le 6m\epsilon \|G_{k,Z}\|_F \le 6m\epsilon \|G_k\|_F .$$

Also, if we let

$$\widehat{\Delta G_k} = \begin{bmatrix} \widehat{\Delta u} & \widehat{\Delta v} \end{bmatrix}^T$$

then the error bounds, (5) and (7), can be used to show that

$$\hat{G}_k^T \Sigma \hat{G}_k = (\tilde{G}_{k,Z} + \Delta G_k)^T \Sigma (\tilde{G}_{k,Z} + \Delta G_k)$$

and

$$(\tilde{G}_{k+1} + e_1 \widehat{\Delta u}^T)^T \Sigma (\tilde{G}_{k+1} + e_1 \widehat{\Delta u}^T) = (\hat{G}_k + e_{p+1} \widehat{\Delta v}^T)^T \Sigma (\hat{G}_k + e_{p+1} \widehat{\Delta v}^T),$$

where e_1 and e_{p+1} are standard basis vectors. This gives

$$\epsilon F_k = \tilde{G}_{k,Z}^T \Sigma \Delta G_k + \Delta G_k^T \Sigma \tilde{G}_{k,Z} - \begin{bmatrix} \tilde{u}_{1,k+1} & \hat{v}_{1,k} \end{bmatrix} \widehat{\Delta G_k} - \widehat{\Delta G_k}^T \begin{bmatrix} \tilde{u}_{1,k+1}^T \\ \hat{v}_{1,k}^T \end{bmatrix},$$

corresponding to a bound

$$\begin{aligned} \|\epsilon F_k\|_F &\le 2\|\tilde{G}_{k,Z}\|_F \|\Delta G_k\|_F + 2(\|\hat{G}_k\|_F + \|\tilde{G}_{k+1}\|_F)\|\widehat{\Delta G_k}\|_F \\ &\le 12m\epsilon \|\tilde{G}_{k,Z}\|_F^2 + 12.5\epsilon(\|\hat{G}_k\|_F + \|\tilde{G}_{k+1}\|_F)^2 \\ &\le 12m\epsilon \|G_k\|_F^2 + 12.5\epsilon\left(\|G_k\|_F + \|G_{k+1}\|_F\right)^2 + O(\epsilon^2). \end{aligned}$$

Since Theorem 4.1 shows that

$$\|G_k\|_F^2, \|G_{k+1}\|_F^2 \le 2\sqrt{k}\|A\|_F + \|G_1\|_F^2,$$

we get a bound on the local error of

$$\|\epsilon F_k\|_F \le (50 + 12m)\epsilon(2\sqrt{k}\|A\|_F + \|G_1\|_F^2).$$

From (3), we see that

$$\|A - R^T R\|_F \le (25 + 6m)(n-1)n\epsilon \left(2\sqrt{n}\|A\|_F + \|G_1\|_F^2\right). \tag{10}$$

5. An unstable implementation. In the last section the stability of the generalized Schur algorithm was proven for the case in which the hyperbolic transformations are applied in factored form. It has already been noted that the implementation is not unique. There are many transformations which introduce the needed zeros in the generators, and they can be applied in multiple ways: the obvious example is the application of the hyperbolic transformation in factored form, (4), versus the direct multiplication approach.

It turns out that the factored form of the hyperbolic transformation is crucial to the stability of the algorithm. To see this, take $\alpha = 4$ and the generators

$$G_0 = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} - \frac{1}{2} & \frac{1}{\sqrt{2}} - \frac{3}{2} & 1 \\ 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} + \frac{1}{2} & \frac{1}{\sqrt{2}} + \frac{3}{2} \\ 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & 0 \\ 0 & 0 & 1-\eta & 1+2\sqrt{\eta} \end{bmatrix}. \tag{11}$$

For small η these generators correspond to an ill-conditioned A for which $\alpha = 4$. If the Schur algorithm with the hyperbolic transformations applied in factored form is used, then a small error $A - R^T R$ will be achieved, independent of the size of η. However, if the hyperbolic transformations are multiplied directly, then the backward error will depend to a significant extent on η.

To see why, note that this example gives, after two steps of the generalized Schur algorithm, the following generators:

$$G_Z = \begin{bmatrix} 1 & 1 \\ 0 & 2 \\ 1-\eta & 1+2\sqrt{\eta} \\ 0 & 1 \end{bmatrix}. \tag{12}$$

In fact, the example was obtained by performing a reversal of the algorithm on equation (12). The construction of the matrix in equation (11) was not really necessary since the essential mechanism of instability is represented in equation (12); this construction merely removes any objection as to whether the generators (12), corresponding to a displacement rank 2 matrix, could actually occur in the factorization of a displacement rank 4 matrix. If not, then one might argue that the instability shown when applying the Schur algorithm to equation (12) is not really relevant to the stability of the factorization of displacement rank 4 matrices. Since the matrix in equation (11) corresponds to four linearly independent generators which reduce to equation (11) in two steps of the Schur procedure, this objection can be dismissed.

Table 1 shows the dependence of the errors on η. It was necessary to go to very ill-conditioned matrices to demonstrate a significant loss of accuracy. The increase of the backward error by a factor 10^5 is more than seems reasonable in a backward stable algorithm, yet the increase in error is quite modest for an unstable algorithm in which the condition number is increased by a factor of 10^{10}. It turns out that this is primarily a result of the fact that instability is due to large local errors rather than unstable propagation of errors.

In contrast, over this very wide range of condition numbers, the error in the Cholesky factors computed by the stable form of the algorithm are all around 10^{-15}. This is consistent with the conclusion of the last section that the algorithm is stable.

It is interesting to note that the algorithm for $\alpha = 2$ is stable even when the hyperbolic transformations are applied directly. This is proven in [3]. The reason is

Table 1
Errors for different values of η.

η	$K(A)$	$\|A - CC^T\|$
10^{-3}	9.6×10^4	2.7×10^{-15}
10^{-8}	1×10^{10}	1.3×10^{-12}
10^{-13}	1×10^{15}	7×10^{-10}

that whenever a hyperbolic transformation of large norm which might magnify errors is performed, it can be proven that the norm of the generators drops drastically. Since the error bounds [4] for the unstabilized form of the hyperbolic transformation are proportional to the norm of the new generators multiplied by the norm of the transformation, the large norm of the hyperbolic transformation is canceled by a proportional decrease in the norm of the generators.

The reason for the decrease in norm of the generators is that whenever ρ is very close to negative one, the action of a hyperbolic transformation is close to just taking the difference between the two generators and scaling the result by $1/\sqrt{1-\rho^2}$. For the case $\alpha = 2$, whenever the leading nonzero elements are $O(\eta)$ apart, leading to a ρ very close to one, it can be proven that all the other components of the generators are within $O(\eta)$ of each other. As can be seen in the example here, this is clearly not the case for more than two generators: the leading elements from which the hyperbolic transformation is computed are within $O(\eta)$, but the other components of the two generators are only within $O(\sqrt{\eta})$. Their difference isn't small enough to completely cancel out the large norm of the hyperbolic transformation.

In Table 1, the backward error seems to display a proportionality to the square root of the condition number. This is a property which is suggested by Theorem 4.1 and the fact that the overall backward error is just a sum of local errors. These imply that neither the generators nor the effect of previous errors will be unduly magnified by the hyperbolic transformations. This makes it possible to concentrate our attention on just one stage of the algorithm. We do not expect the results in the overall factorization to be significantly worse than the worst possible loss of accuracy in a single stage.

Note that we have already proven that when applying a hyperbolic transformation the result is always a set of generators which satisfy the bound

$$\|G_k\|_F^2 \leq 2\sqrt{k-1}\|A\|_F + \|G_1\|_F^2$$

or, just looking at the two generators on which the hyperbolic transformation has acted,

$$\left\| \begin{bmatrix} u_{1,k}^T \\ v_{1,k}^T \end{bmatrix} \right\|_F^2 \leq 2\sqrt{k-1}\|A\|_F + \|G_1\|_F^2.$$

As a direct result of the analysis in [4], the local error introduced by the hyperbolic transformation which produces $u_{1,k}$ and $v_{1,k}$ will be proportional to the norm of the generators and the norm of the transformation, which can be bounded as follows:

$$\epsilon\|H_k\| \left\| \begin{bmatrix} u_{1,k}^T \\ v_{1,k}^T \end{bmatrix} \right\| \leq \frac{2\epsilon}{\sqrt{1-\rho^2}} \left\| \begin{bmatrix} u_{1,k}^T \\ v_{1,k}^T \end{bmatrix} \right\| \leq \frac{2\epsilon}{\sqrt{1-\rho^2}}(2\sqrt{k-1}\|A\|_F + \|G_1\|_F^2).$$

Although there does not appear to be a general theory relating the condition number of an arbitrary low α-rank matrix to the values ρ_k, it is proven in [5] that for a positive-definite Toeplitz matrix

$$\prod_{k=1}^{n} \frac{1+\rho_k}{1-\rho_k^2} \leq \|T^{-1}\|_1 \leq \prod_{k=1}^{n-1} \frac{(1+|\rho_k|)^2}{1-\rho_k^2}.$$

If similar inequalities hold for matrices of displacement rank α, then this is enough to suggest that the errors will tend to be proportional to the square root of the condition number.

We can get a somewhat more conclusive result. If we assume that the factorization goes to completion, we can write of the leading elements of $\hat{u}_{11,k-1}$ and $\hat{v}_{11,k-1}$ from which ρ is computed,

$$\hat{v}_{11,k-1} = \hat{u}_{11,k-1}(1-\epsilon_1)$$

and $1 \geq \epsilon_1 \geq \epsilon$. We have assumed without loss of generality that $\hat{v}_{11,k-1} > 0$ and $\hat{u}_{11,k-1} > 0$. If this is not the case, then either generator can be multiplied by -1 without causing any problems.

This means that

$$\frac{1}{1-\rho^2} = \frac{\hat{u}_{11,k-1}^2}{\hat{u}_{11,k-1}^2 - \hat{v}_{11,k-1}^2} = \frac{1}{2\epsilon_1 - \epsilon_1^2} < \frac{1}{\epsilon_1} \leq \frac{1}{\epsilon}.$$

So the error incurred during the hyperbolic transformation will be at worst proportional to

$$2\sqrt{\epsilon}(2\sqrt{k-1}\|A\|_F + \|G_1\|_F^2).$$

This explains the results in Table 1: because of the stability of the error propagation in the algorithm, as well as the bound on the generators, the errors will be at worst a modest multiple of $\sqrt{\epsilon}$. Although the backward errors are dependent on the reflection coefficients, there is a limit to this dependence.

6. Summary. This paper has proven the stability of a method for Cholesky factorization of low α-rank matrices. The stability result is dependent on the particular implementation. Since the result depends on having a bound on the norm of the generators, it is important to choose transformations which permit such a bound. Working with transformations which can be factored as a product of a sequence of plane rotations and one hyperbolic transformation gives such a bound. In the case in which the algorithm is implemented with unfactored hyperbolic transformations, this bound on the generators is not sufficient for stability; however, bounds were given which support the assertion that the stability of the error propagation keeps the instability from being as bad as might otherwise be expected.

Although not discussed here, the approach in this paper can be used in a manner similar to that of [1] to provide a fast, weakly stable solution of Toeplitz least squares problems. The conclusions would be essentially the same as those in [1].

REFERENCES

[1] A. W. Bojanczyk, R. P. Brent, and F. R. De Hoog, *A weakly stable algorithm for general Toeplitz systems*, Numer. Algorithms, to appear.

[2] A. W. Bojanczyk, R. P. Brent, and F. R. De Hoog, *QR factorization of Toeplitz matrices*, Numer. Math., 49 (1986), pp. 81–94.

[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. 210–220.

[5] 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.

[6] K. A. Gallivan, S. Thirumalai, P. Van Dooren, and V. Vermaut, *High performance algorithms for Toeplitz and block Toeplitz matrices*, Linear Algebra Appl., 241–243 (1996), pp. 343–388.

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

[8] N. J. Higham, *Analysis of the Cholesky decomposition of a semi-definite matrix*, in Reliable Numerical Computation, M. G. Cox and S. J. Hammarling, eds., Oxford University Press, London, 1990, pp. 161–185.

[9] N. J. Higham, *Stability of the diagonal pivoting method with partial pivoting*, Numerical Analysis Report No. 265, Manchester Centre for Computational Mathematics, Manchester, England, July 1995.

[10] 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.

[11] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, London, 1965.