LBL-37878
UC-405
Preprint

Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

Physics Division

Mathematics Department

To be submitted for publication

New Fast Algorithms for Structured Linear Least Squares Problems

M. Gu

December 1995

Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

LBL-37878

New Fast Algorithms for Structured Linear Least Squares Problems[1]

Ming Gu
Department of Mathematics and Lawrence Berkeley National Laboratory
University of California
Berkeley, CA 94720

December, 1995

[1] Supported by the Applied Mathematical Sciences Subprogram of the Office of Energy Research, U.S. Department of Energy under Contract DE-AC03-76SF00098.

New Fast Algorithms for Structured Linear Least Squares Problems

Ming Gu*

December 17, 1995

Abstract

We present new fast algorithms for solving the Toeplitz and the Toeplitz-plus-Hankel least squares problems. These algorithms are based on a new fast algorithm for solving the Cauchy-like least squares problem. We perform an error analysis and provide conditions under which these algorithms are numerically stable. We also develop implementation techniques that significantly reduce the execution time. Our numerical results indicate that these algorithms are highly efficient and numerically stable for problems ranging from well-conditioned to ill-conditioned to numerically singular.

Keywords: Linear least squares problem, Cauchy-like matrix, Hankel matrix, Toeplitz matrix, structured matrix, fast algorithm, displacement rank, generator, pivoting, iterative refinement, error analysis.

*Department of Mathematics, Lawrence Berkeley National Laboratory and University of California, Berkeley, CA 94720. The author was supported in part by the Applied Mathematical Sciences Subprogram of the Office of Energy Research, U.S. Department of Energy under Contract DE-AC03-76SF00098.

Contents

1 Introduction

1.1 Displacement Equations

The *Sylvester type displacement equation* for a matrix $M \in \mathbf{C}^{m\times n}$ is

$$\Omega \cdot M - M \cdot \Lambda = \Delta \tag{1.1}$$

where $\Omega \in \mathbf{C}^{m\times m}$ and $\Lambda \in \mathbf{C}^{n\times n}$; $\Delta \in \mathbf{C}^{m\times n}$ is called the *generator* of M with respect to Ω and Λ; and $r = \text{rank}(\Delta) \leq \min(m,n)$ is called the *displacement rank* of M with respect to Ω and Λ. M is considered to possess a *displacement structure* with respect to Ω and Λ if $r \ll \min(m,n)$.

In general, there is no simple relationship between r and $\text{rank}(M)$. For $r \ll \min(m,n)$, we usually factorize Δ as $\Delta = A\cdot B$ for matrices $A \in \mathbf{C}^{m\times r}$ and $B \in \mathbf{C}^{r\times n}$. This decomposition is not unique. For numerical stability reasons, we often choose A to be well-conditioned.

The displacement equation (1.1) does not in general reflect the potential symmetry structure in M. The *symmetric Stein type displacement equation* for a Hermitian matrix M ($m = n$ and $M^* = M$) is

$$M - \Omega^* \cdot M \cdot \Omega = \Theta \ , \tag{1.2}$$

where $\Omega \in \mathbf{C}^{m\times m}$; $\Theta \in \mathbf{C}^{m\times m}$ is Hermitian and is called the *generator* of M with respect to Ω; and $r = \text{rank}(\Theta) \leq m$ is called the *displacement rank* of M with respect to Ω. M is considered to possess a *displacement structure* with respect to Ω if $r \ll m$.

For $r \ll m$, we usually factorize Θ as $\Theta = A\cdot J\cdot A^*$ for matrices $A \in \mathbf{C}^{m\times r}$ and $J \in \mathbf{C}^{r\times r}$ with J being Hermitian. Similar to above, this decomposition is not unique, and we often choose A to be well-conditioned.

The concept of displacement structure was first introduced in Kailath, Kung, and Morf [31]; the symmetric variant of which, the displacement equation of the form (1.2), first appeared in Chun, Kailath, and Lev-Ari [12]. Displacement equations of the form (1.1) first appeared in Heinig and Rost [28] for the special case $m = n$. The most general form of displacement structure for square matrices, which includes equation (1.1) for $m = n$ and equation (1.2) as special cases, was introduced in Kailath and Sayed [33]. For a comprehensive discussion on the displacement structure theory and applications, see Kailath and Sayed [34].

1.2 Fast Algorithms for Structured Matrices

A special case in (1.1) is when both Ω and Λ are diagonal. Let $C \in \mathbf{C}^{m\times n}$ be a matrix satisfying

$$\Omega \cdot C - C \cdot \Lambda = A \cdot B \tag{1.3}$$

with

$$\Omega = \text{diag}(\omega_1, \cdots, \omega_m) \ , \ \Lambda = \text{diag}(\lambda_1, \cdots, \lambda_n) \ \text{ and } \ A = \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \ , \ B = (b_1, \cdots, b_n) \ ,$$

where ω_k, λ_j are scalars; and a_k and b_j are r-dimensional row and column vectors. C is called a *Cauchy-like matrix.* Usually we assume that $\omega_k \neq \lambda_j$ for all $1 \leq k \leq m$ and $1 \leq j \leq n$. In this case the (k, j) entry of C is $\dfrac{a_k \cdot b_j}{\omega_k - \lambda_j}$. In particular, C is a Cauchy matrix if $m = n$, $r = 1$, and $a_k = b_j = 1$ for all k and j (see Heinig [25]). In the case where $\omega_k = \lambda_j$ for some pairs of (k, j), equation (1.3) requires that $a_k \cdot b_j = 0$ and allows the corresponding entries in C to be arbitrary. A Cauchy-like matrix C has the interesting property that any submatrix of C is again a Cauchy-like matrix. In addition, if C is a nonsingular square Cauchy-like matrix, then C^{-1} is a Cauchy-like matrix as well. For square Cauchy-like matrices with $\omega_k \neq \lambda_j$ for all k and j, Heinig [25] developed a fast algorithm for computing an implicit LU factorization of C with partial pivoting in $O(n^2)$ floating point operations; and Gohberg, Kailath, and Olshevsky [18] developed an explicit fast Gaussian Elimination with Partial Pivoting (GEPP) procedure for factorizing C based on this work. Sweet and Brent [44] showed that the generator of C could suffer large internal element growth in the procedure of [18]; and Gu [23] presented a modified procedure that avoids such internal element growth and that can perform a fast variation of Gaussian Elimination with Complete Pivoting (GECP).

Some symmetric/Hermitian Cauchy-like matrices satisfy displacement equation (1.2). Let $\Omega \in \mathbf{C}^{m\times m}$ be a diagonal matrix and let $M = M^* \in \mathbf{C}^{m\times m}$ be a Hermitian matrix satisfying the displacement equation

$$M - \Omega^* \cdot M \cdot \Omega = A \cdot J \cdot A^{(*)} \ , \tag{1.4}$$

where $\Omega = \mathrm{diag}(\omega_1, \cdots, \omega_m)$; $A = \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix}$; and $J \in \mathbf{C}^{r\times r}$ is Hermitian. For $\omega_k^* \cdot \omega_j \neq 1$, the (k, j) entry of H is $\dfrac{a_k \cdot J \cdot a_j}{1 - \omega_k^* \cdot \omega_j}$. For $\omega_k^* \cdot \omega_j = 1$, Equation (1.4) requires that $a_k \cdot J \cdot a_j^* = 0$, and allows the corresponding entries in H to be arbitrary. Assume that $\omega_k \neq 0$ for all k, equation (1.4) can be rewritten in the form of (1.3) as

$$\Omega^{-*} \cdot M - M \cdot \Omega = (\Omega^{-*} \cdot A) \cdot (J \cdot A^{(*)}) \ .$$

Hence M is a Hermitian Cauchy-like matrix. Kailath and Olshevsky [32] have developed a fast Bunch-Kaufman Pivoting procedure for factorizing symmetric/Hermitian Cauchy-like matrices.

Other classes of structured matrices include the Toeplitz matrices and the Hankel matrices. A Toeplitz matrix T is a matrix whose entries are constant along every diagonal ($T = (t_{k-j})_{1\leq k\leq m, 1\leq j\leq n}$); and a Hankel matrix H is a matrix whose entries are constant along every anti-diagonal ($H = (h_{k+j-2})_{1\leq k\leq m, 1\leq j\leq n}$). Let $S_m \in \mathbf{R}^{m\times m}$ be the matrix that is 1 on the main anti-diagonal and 0 everywhere else (for example $S_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$). It is well-known that for every Hankel matrix $H \in \mathbf{C}^{m\times n}$, $S_m \cdot H$ is a Toeplitz matrix. Toeplitz and Hankel matrices are included in the larger class of Toeplitz-plus-Hankel matrices, which are sums of Toeplitz and Hankel matrices. These matrices often arise from signal processing and control theory applications (see, for example, Bunch [7] and Nagy [38]). We will discuss the displacement equations the Toeplitz matrix and the Toeplitz-plus-Hankel matrix satisfy in §3. Our main goal in this paper is to present new fast algorithms for solving the linear least squares problem

$$\min_{x} \|M \cdot x - h\|_2 , \tag{1.5}$$

where $M \in \mathbf{R}^{m \times n}$ is the Toeplitz or the Toeplitz-plus-Hankel matrix; and $h \in \mathbf{R}^m$ is a vector. We will also consider the case where M is a real or complex Cauchy-like matrix. Throughout this paper, we assume that M is non-singular and that $m \geq n + r$, where r is the displacement rank of M. The problem (1.5) has a unique solution

$$x_M = (M^* \cdot M)^{-1} \cdot M^* \cdot h . \tag{1.6}$$

Fast algorithms for solving the least squares problem (1.5) when M is a Toeplitz matrix have been developed by Bojanczyk, Brent, and de Hoog [4], Chun, Kailath, and Lev-Ari [12], Cybenko [14, 15], Nagy [37], Park and Eldén [39], Qiao [40], and Sweet [42] that require $O(mn)$ floating point operations, as opposed to $O(mn^2)$ floating point operations normally required for general dense linear least squares problems. Fast parallel algorithms have also been developed by Bojanczyk and Brent [3, 6]. However, some of these algorithms have unknown stability properties (see Brent [6]) and others are known to be unstable (see Luk and Qiao [36]); most of these methods suffer from loss of accuracy for very ill-conditioned problems.

A special case of the Toeplitz least squares problem is the Toeplitz linear system of equations, where one solves for x in $M \cdot x = h$ with M being a square Toeplitz matrix ($m = n$). For discussions on some of the earlier fast and superfast methods (performing $O(n^2)$ and $O(nlog_2^2 n)$ floating point operations, respectively) for solving such equations, see Bojanczyk, Brent and Sweet [5], Bunch [7], Cybenko [13], Sweet [43] and the references therein.

Recently, Heinig [25] showed that the square Toeplitz matrix can be transformed into a Cauchy-like matrix via the fast Fourier transforms; and Gohberg, Kailath, and Olshevsky [18] and Heinig [26] further showed that the Toeplitz-plus-Hankel matrix can be transformed into a Cauchy-like matrix via the fast trigonometric transforms. Hence fast algorithms for solving Cauchy-like linear systems of equations can be used to solve Toeplitz and Toeplitz-plus-Hankel linear systems of equations. Independently, Chandrasekaran and Sayed [9] presented a modified version of the QR type algorithm of [12] and showed it to be fast and backward stable for solving non-singular Toeplitz linear systems of equations.

1.3 Main Results

We present a new fast algorithm for solving the least squares problem (1.5) when M is a Cauchy-like matrix. This algorithm reduces the least squares problem into two Cauchy-like systems of linear equations and solves these systems by generalizing triangular factorization techniques developed in [18, 23, 25, 32]. Our error analysis shows that this algorithm is backward stable if the L matrix in the LU factorization of M with fast partial/complete pivoting is well-conditioned.

We also present new fast algorithms for solving the least squares problem (1.5) when M is a Toeplitz or Toeplitz-plus-Hankel matrix. These algorithms transform M into a Cauchy-like matrix via the fast Fourier or trigonometric transforms and solves the resulting Cauchy-like least squares problem using the fast algorithm above. Since the choices of transformations are not unique, we compare different choices in terms of efficiency and numerical accuracy in solving the Toeplitz and the Toeplitz-plus-Hankel least squares problems. Our error analysis shows that these algorithms are as backward stable as the algorithm for solving the resulting Cauchy-like least squares problem. Since the Hankel and Toeplitz matrices are simply related (see §1.2), the new fast algorithms for solving the Toeplitz and Toeplitz-plus-Hankel least squares problems are also new fast algorithms for solving the Hankel least squares problem; and the numerical stability results are the same.

We develop implementation techniques that significantly reduce the execution time. We also perform a large number of numerical experiments on the new fast Toeplitz and Toeplitz-plus-Hankel least squares problem solvers and compare them with the straightforward QR type least squares problem solver that ignores the Toeplitz and Toeplitz-plus-Hankel structures. Our numerical results indicate that these fast algorithms are significantly faster than the straightforward solver and yet are essentially as accurate on problems ranging from well-conditioned to ill-conditioned to numerically singular.

1.4 Overview

In §2 we present the new fast algorithm for solving the Cauchy-like least squares problem. In §3 we show how to transform Toeplitz and Toeplitz-plus-Hankel matrices into Cauchy-like matrices and compare different choices of transformations in terms of efficiency and numerical accuracy. In §4 we perform an error analysis for these algorithms. In §5 we develop implementation techniques that reduce the execution time and present numerical results. And in §6 we discuss some extensions, draw conclusions, and ask some open problems.

1.5 Notation and Conventions

i is the unit imaginary number ($i^2 = -1$). For a matrix M, M^* denotes its complex conjugate; in case M is real, both M^* and M^T denote its transpose. $\sigma_{\max}(M)$ and $\sigma_{\min}(M)$ denote the largest and the smallest singular values of M, respectively; and $\kappa(M) = \dfrac{\sigma_{\max}(M)}{\sigma_{\min}(M)}$ is the 2-norm condition number of M.

$M_{p:q,s:k}$ is a submatrix of M that selects rows p to q of columns s to k; $M_{:,s:k}$ and $M_{s:k,:}$ select s^{th} through k^{th} rows and columns, respectively; and when $s = k$, we replace $s : k$ by s. However, for matrices A and B with or without superscripts, we further set

$$A_{s:k} = A_{s:k,:} \quad \text{and} \quad B_{s:k} = B_{:,s:k}\ ;$$

for matrices C and L with or without superscripts, we set

$$C_1 = C_{1:n,1:n}\ ,\ C_2 = C_{n+1:m,1:n} \quad \text{and} \quad L_1 = L_{1:n,1:n}\ ,\ L_2 = L_{n+1:m,1:n}\ ;$$

and for matrices Ω and Λ, $\Omega_{s:k}$ and $\Lambda_{s:k}$ select both rows and columns s to k of Ω and Λ, respectively.

$|M|$ is the matrix of moduli of the $\{M_{k,j}\}$. We use the *max norm*, the ∞-*norm*, the *2-norm*, and the *Frobenius norm*:

$$\|M\|_{\max} = \max_{k,j} |M_{k,j}|\ ,\ \|M\|_{\infty} = \max_{k} \sum_{j} |M_{k,j}|\ ,\ \|M\|_2 = \max_{\|u\|_2=1} \|M \cdot u\|_2\ ,$$

and $\|M\|_F = \sqrt{\sum_{k,j} |M_{k,j}|^2}$. For a matrix $M \in \mathbf{C}^{m\times n}$, the following inequalities hold:

$$\frac{\|M\|_F}{\sqrt{m\,n}} \le \|M\|_{\max} \le \|M\|_2 \quad \text{and} \quad \frac{\|M\|_2}{\sqrt{m}} \le \|M\|_{\infty} \le \sqrt{n} \cdot \|M\|_2\ . \tag{1.7}$$

When the norm in $\|\cdot\|$ is not specified, it is one of the 1, 2, and ∞ norms.

I_k is the k-by-k identity matrix, $e_k \in \mathbf{R}^k$ is the vector of all 1's. $P_m \in \mathbf{R}^{m\times m}$ and $Q_n \in \mathbf{R}^{n\times n}$ are permutation matrices; and $P_m(j,k)$ and $Q_n(j,k)$ denote the permutations that interchange the j^{th} and k^{th} rows of matrices with m and n rows, respectively.

A *flop* is a real floating-point operation $\alpha \circ \beta$, where α and β are real floating-point numbers and $\circ$ is one of $+$, $-$, $\times$, and $\div$. Taking the absolute value or comparing two floating-point numbers is also counted as a flop. We count a complex addition or subtraction as 2 flops; a complex multiplication 6 flops; and a complex division 11 flops.

ϵ is the machine precision. In our error analysis, we take the usual model of arithmetic:[1]

$$\mathrm{fl}(\alpha \circ \beta) = (\alpha \circ \beta)(1+\eta)\ ,$$

where $\mathrm{fl}(\alpha \circ \beta)$ is the floating point result of the operation $\circ$; and $|\eta| \le \epsilon$. For simplicity, we ignore the possibility of overflow and underflow. It is well-known that for any $A \in \mathbf{C}^{m\times r}$ and $B \in \mathbf{C}^{r\times n}$, when the matrix-matrix product $A \cdot B$ is computed in the straightforward way, we have

$$|\mathrm{fl}\,(A \cdot B) - A \cdot B| \le \eta \cdot r \cdot |A| \cdot |B|\ , \tag{1.8}$$

where η is a small multiple of ϵ.

2 The Cauchy-like Least Squares Problem

2.1 Reducing One Cauchy-like Least Squares Problem into Two Cauchy-like Linear Systems

Let $C \in \mathbf{C}^{m\times n}$ be a Cauchy-like matrix satisfying (1.3) and define

$$\psi_{\max} = \max_{1\le k\le m, 1\le j\le n} |\omega_k| + |\lambda_j|\ ,\ \psi_{\min} = \min_{1\le k\le m, 1\le j\le n} |\omega_k - \lambda_j|\ ,\ \psi = \frac{\psi_{\max}}{\psi_{\min}} \tag{2.9}$$

$$\phi_{\max} = 2 \cdot \max_{1\le k\le m} |\omega_k|\ ,\ \phi_{\min} = \min_{1\le k\ne j\le m} |\omega_k - \omega_j|\ ,\ \phi = \frac{\phi_{\max}}{\phi_{\min}}\ . \tag{2.10}$$

[1] This model excludes some CRAY machines that do not have a guard digit. Our error analysis still holds for such machines with a few easy modifications.

We assume that C is a non-singular matrix and $\psi > 0$ and $\phi > 0$. Hence the diagonal entries of Ω are distinct and C is the unique solution to (1.3). In §2.1 we consider the least squares problem (1.5) for $M = C$. The notation established in §1.5 will be heavily used in this section.

The expression in the solution (1.6) is not suitable for direct numerical computation for ill-conditioned C. The standard approach is to use the QR factorization of C. Let

$$C = Q \cdot R\,,$$

where $Q \in \mathbf{C}^{m\times n}$ is column unitary and $R \in \mathbf{C}^{n\times n}$ is upper-triangular. The least squares solution is

$$x_C = (C^* \cdot C)^{-1} \cdot (C^* \cdot h) = (R^* \cdot Q^* \cdot Q \cdot R)^{-1} \cdot (R^* \cdot Q^* \cdot h) = R^{-1} \cdot (Q^* \cdot h)\,. \qquad (2.11)$$

Although this scheme is backward stable, it is slow for large m and n. The QR factorization requires $O(mn^2)$ flops to compute in general; and no known algorithm can stably compute a QR factorization of a Cauchy-like matrix in $O(mn)$ flops.

Fortunately, This scheme is not the only way to compute x_C. Partition (see §1.5 for notation)

$$C = \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \quad \text{and} \quad h = \begin{pmatrix} h_{1:n} \\ h_{n+1:m} \end{pmatrix}.$$

We assume that C_1 is non-singular and define

$$Z = C_2 \cdot C_1^{-1} \in \mathbf{C}^{(m-n)\times n} \quad \text{and} \quad K = I_n + Z^* \cdot Z \in \mathbf{C}^{n\times n}\,. \qquad (2.12)$$

The least squares solution x_C can now be rewritten as

$$\begin{aligned} x_C &= (C_1^* \cdot (I_n + Z^* \cdot Z) \cdot C_1)^{-1} \cdot C_1^* \cdot \left(\begin{pmatrix} I_n & Z^* \end{pmatrix} \cdot \begin{pmatrix} h_{1:n} \\ h_{n+1:m} \end{pmatrix} \right) \\ &= C_1^{-1} \cdot K^{-1} \cdot (h_{1:n} + Z^* \cdot h_{n+1:m})\,. \end{aligned}$$

In this formula, C_1 is a square Cauchy-like matrix (see §1.2), and hence x_C can be computed as the solution to the Cauchy-like linear system

$$C_1 \cdot x_C = g\,, \qquad (2.13)$$

where

$$g = (I_n + Z^* \cdot Z)^{-1} \cdot (h_{1:n} + Z^* \cdot h_{n+1:m})\,.$$

In Theorems 2.1 through 2.3 that follow, we show that for the types of Cauchy-like matrices C that are transformed from the Toeplitz or Toeplitz-plus-Hankel matrices, both Z and K are Cauchy-like matrices as well, with K being symmetric/Hermitian positive definite. Hence g can be computed as the solution to the symmetric/Hermitian positive definite Cauchy-like linear system

$$K \cdot g = h_{1:n} + Z^* \cdot h_{n+1:m}\,. \qquad (2.14)$$

Theorem 2.1 *Let C be a Cauchy-like matrix satisfying the displacement equation (1.3). Then the matrix Z in (2.12) is a Cauchy-like matrix satisfying the displacement equation*

$$\Omega_{n+1:m} \cdot Z - Z \cdot \Omega_{1:n} = \left(A_{n+1:m} - C_2 \cdot C_1^{-1} \cdot A_{1:n}\right) \cdot (B \cdot C_1^{-1}) \, . \tag{2.15}$$

Proof. Equation (1.3) can be rewritten as

$$\Omega_{1:n} \cdot C_1 - C_1 \cdot \Lambda = A_{1:n} \cdot B \quad \text{and} \quad \Omega_{n+1:m} \cdot C_2 - C_2 \cdot \Lambda = A_{n+1:m} \cdot B \, ,$$

which can be further rewritten as

$$C_1^{-1} \cdot \Omega_{1:n} = \Lambda \cdot C_1^{-1} + (C_1^{-1} \cdot A_{1:n}) \cdot (B \cdot C_1^{-1}) \, ,$$

and

$$\Omega_{n+1:m} \cdot C_2 = C_2 \cdot \Lambda + A_{n+1:m} \cdot B \, .$$

On the other hand,

$$\Omega_{n+1:m} \cdot Z - Z \cdot \Omega_{1:n} = (\Omega_{n+1:m} \cdot C_2) \cdot C_1^{-1} - C_2 \cdot (C_1^{-1} \cdot \Omega_{1:n}) \, .$$

Plugging in the above two relations and simplifying, we obtain (2.15). ∎

Since both factors in the generator for Z in (2.15) can be ill-conditioned, we rewrite the generator of Z as

$$Q \cdot W = \left(A_{n+1:m} - C_2 \cdot C_1^{-1} \cdot A_{1:n}\right) \cdot (B \cdot C_1^{-1}) \, , \tag{2.16}$$

where $Q \in \mathbf{C}^{(m-n)\times r}$ and $W \in \mathbf{C}^{r\times n}$ with Q being well-conditioned. In §2.2 we will discuss how to compute Q and W without computing $A_{n+1:m} - C_2 \cdot C_1^{-1} \cdot A_{1:n}$ and $B \cdot C_1^{-1}$.

Theorem 2.2 below shows that K is a Cauchy-like matrix if all the matrices involved in (1.3) are real.

Theorem 2.2 *Let C be a Cauchy-like matrix satisfying the displacement equation (1.3). Assume that all the matrices Ω, Λ, A, and B are real. Then the matrix K in (2.12) is a symmetric Cauchy-like matrix satisfying the displacement equation*

$$\Omega_{1:n} \cdot K - K \cdot \Omega_{1:n} = \mathcal{A} \cdot \mathcal{J} \cdot \mathcal{A}^T \, , \tag{2.17}$$

where

$$\mathcal{A} = \begin{pmatrix} Z^T \cdot Q & W^T \end{pmatrix} \quad \text{and} \quad \mathcal{J} = \begin{pmatrix} 0 & -I_r \\ I_r & 0 \end{pmatrix} ,$$

with Q and W being defined in (2.16).

Proof.

$$\begin{aligned} \Omega_{1:n} \cdot K - K \cdot \Omega_{1:n} &= \Omega_{1:n} \cdot (I_n + Z^T \cdot Z) - (I_n + Z^T \cdot Z) \cdot \Omega_{1:n} \\ &= (Z \cdot \Omega_{1:n})^T \cdot Z - Z^T \cdot (Z \cdot \Omega_{1:n}) \, . \end{aligned}$$

On the other hand, Combining (2.15) and (2.16) gives

$$Z \cdot \Omega_{1:n} = \Omega_{n+1:m} \cdot Z - Q \cdot W \, .$$

Theorem 2.2 follows by plugging this relation into above and simplifying. ▮

Remark 1: To compute $\mathcal{A}$ for a given generator $Q \cdot W$ of Z, we need to form the matrix Z and then perform the matrix-matrix product $Z^T \cdot Q$. The total cost is about $(4r+1)(m-n)n$ flops.

The following theorem shows that H is a Cauchy-like matrix if all diagonal entries in Ω are on the unit circle in the complex plane.

Theorem 2.3 *Let C be a Cauchy-like matrix satisfying the displacement equation (1.3). Assume that $|\omega_j| = 1$ for $1 \le j \le m$. Then the matrix K in (2.12) is a Cauchy-like matrix satisfying the displacement equation*

$$K - \Omega^*_{1:n} \cdot K \cdot \Omega_{1:n} = \mathcal{A} \cdot \mathcal{J} \cdot \mathcal{A}^* \tag{2.18}$$

where

$$\mathcal{A} = \begin{pmatrix} W^* & Z^* \cdot \Omega^*_{n+1:m} \cdot Q \end{pmatrix} \quad \textit{and} \quad \mathcal{J} = \begin{pmatrix} -Q^* \cdot Q & I_r \\ I_r & 0 \end{pmatrix},$$

with Q and W being defined in (2.16).

Proof. By assumption, we have

$$\Omega^*_{1:n} \cdot \Omega_{1:n} = I_n \quad \text{and} \quad \Omega^*_{n+1:m} \cdot \Omega_{n+1:m} = I_{m-n} \, .$$

Hence

$$\begin{aligned} K - \Omega^*_{1:n} \cdot K \cdot \Omega_{1:n} &= (I_n + Z^* \cdot Z) - \Omega^*_{1:n} \cdot (I_n + Z^* \cdot Z) \cdot \Omega_{1:n} \\ &= Z^* \cdot Z - (Z \cdot \Omega_{1:n})^* \cdot (Z \cdot \Omega_{1:n}) \, . \end{aligned}$$

On the other hand, Combining (2.15) and (2.16) gives

$$Z \cdot \Omega_{1:n} = \Omega_{n+1:m} \cdot Z - Q \cdot W \, .$$

Theorem 2.3 follows by plugging this relation into above and simplifying. ▮

Remark 2: To compute $\mathcal{A}$ for a given generator $Q \cdot W$ of Z, we need to form the matrix Z and then perform the matrix-matrix product $Z^* \cdot (\Omega^*_{n+1:m} \cdot Q)$. The total cost is about $(16r+11)(m-n)n$ flops.

In §2.2 through§ 2.4 that follow, we will solve linear systems (2.13) and (2.14) by developing fast algorithms for computing the generator of Z and factorizing C_1 and K.

2.2 Gaussian Elimination for Cauchy-like Matrices

Let C be a Cauchy-like matrix satisfying (1.3) and let the LU-factorization of C be

$$C = L \cdot U \,, \tag{2.19}$$

where $L \in \mathbf{C}^{m\times n}$ is unit lower triangular and $U \in \mathbf{C}^{n\times n}$ is upper triangular. This factorization, when exists, can be computed via Gaussian Elimination (GE). Given (2.19), the factorization for C_1 is simply $L_1 \cdot U$ (see §1.5 for notation).

The first step of Gaussian elimination is to zero-out the first column of C below the diagonal entry:

$$C = \begin{pmatrix} \gamma & u \\ v & C_{2:m,2:n} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ l & I_{m-1} \end{pmatrix} \cdot \begin{pmatrix} \gamma & u \\ 0 & C^{(2)} \end{pmatrix} \,, \tag{2.20}$$

where $U_{1,1} = \gamma$ is the pivot; $L_{2:m,1} \equiv l = v/\gamma$; $U_{1,2:n} = u$; and $C^{(2)} = C_{2:m,2:n} - l \cdot u$ is the Schur complement of γ. Theorem 2.4 below asserts that $C^{(2)}$ is also a Cauchy-like matrix. Variations and generalizations of this theorem have appeared in [18, 19, 20, 25, 33]. The algorithms of Gohberg, Kailath, and Olshevsky [18] are based on it.

Theorem 2.4 *Let matrix C in (2.20) satisfy the displacement equation (1.3) with $\omega_k \neq \lambda_j$ for all $1 \leq k \leq m$ and $1 \leq j \leq n$. Assume that $\gamma \neq 0$. Then $C^{(2)}$ satisfies the displacement equation*

$$\Omega_{2:m} \cdot C^{(2)} - C^{(2)} \cdot \Lambda_{2:n} = A^{(2)} \cdot B^{(2)} \tag{2.21}$$

with $A^{(2)} = A_{2:m} - l \cdot A_{1,:} \in \mathbf{C}^{(m-1)\times r}$ and $B^{(2)} = B_{2:n} - B_{:,1} \cdot u/\gamma \in \mathbf{C}^{r\times(n-1)}$.

Hence the first step of Gaussian elimination on the matrix C involves computing γ, v, and u of C from equation (1.3); computing the vector l; and computing the matrices $A^{(2)}$ and $B^{(2)}$ in equation (2.21). Gaussian elimination then proceeds by recursively applying this step to $C^{(2)}$. If all the pivots are non-zero, then at the end of this procedure, C is factored into (2.19).

While the generator of Z can be computed directly from formula (2.15) using the factorization (2.19), numerical accuracy could be compromised for ill-conditioned C_1. In the following we present an alternative procedure that computes the generator for Z during the course of Gaussian elimination on C. To this end, we define $C^{(1)} = C$, $L^{(0)} = I_m$, and

$$L^{(k)} = \begin{pmatrix} L_{1:k,1:k} & 0 \\ L_{k+1:m,1:k} & I_{m-k} \end{pmatrix} \in \mathbf{C}^{m\times m} \tag{2.22}$$

and

$$C^{(k+1)} = C_{k+1:m,k+1:n} - C_{k+1:m,1:k} \cdot C_{1:k,1:k}^{-1} \cdot C_{1:k,k+1:n} \,,$$

for $k = 1, 2, \cdots, n-1$. $C^{(k+1)}$ is the Schur complement of $C_{1:k,1:k}$.

Lemma 2.1 *Let the Cauchy-like matrix C in (1.3) have LU factorization (2.19). Then for $2 \le k \le n$, the Schur complements $\{C^{(k)}\}_{k=1}^{n}$ satisfy the displacement equation*

$$\Omega_{k:m} \cdot C^{(k)} - C^{(k)} \cdot \Lambda_{k:n} = A^{(k)} \cdot B^{(k)} , \tag{2.23}$$

where $\{A^{(k)}\}$ and $\{B^{(k)}\}$ satisfy the recursion: $A^{(1)} = A$, $B^{(1)} = B$ and

$$A^{(k)} = A^{(k-1)}_{2:m-k} - L_{k:m,k-1} \cdot A^{(k-1)}_{1,:} \ , \ B^{(k)} = B^{(k-1)}_{2:n-k} - B^{(k-1)}_{:,1} \cdot (U_{k-1,k:n}/U_{k-1,k-1}) \ , \tag{2.24}$$

as well as the identities

$$\begin{aligned} A^{(k)} &= A_{k:m} - C_{k:m,1:k-1} \cdot C^{-1}_{1:k-1,1:k-1} \cdot A_{1:k-1} && (2.25)\\ B^{(k)} &= B_{k:n} - B_{1:k-1} \cdot C^{-1}_{1:k-1,1:k-1} \cdot C_{1:k-1,k:n} \ . && (2.26) \end{aligned}$$

Proof. It is well-known (see, for example, Wilkinson [48]) that after the first $(k-1)^{st}$ steps, Gaussian elimination performs the elimination procedure on $C^{(k)}$. Partition

$$C^{(k)} = \begin{pmatrix} \gamma^{(k)} & u^{(k)} \\ v^{(k)} & C^{(k)}_{2:m-k+1,2:n-k+1} \end{pmatrix} .$$

Then $U_{k,k} = \gamma^{(k)}$, $U_{k,k+1:n} = u^{(k)}$, $L_{k+1:m,k} \equiv l^{(k)} = v^{(k)}/\gamma^{(k)}$, and

$$C^{(k+1)} = C^{(k)}_{2:m-k+1,2:n-k+1} - L_{k+1:m,k} \cdot u^{(k)} \ .$$

Equations (2.23) and (2.24) follow immediately by applying Theorem 2.4 to every $C^{(k)}$.

To show (2.25), we note that

$$L^{(k)} = L^{(k-1)} \cdot \begin{pmatrix} I_{k-1} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & l^{(k)} & I_{m-k} \end{pmatrix} , \ (L^{(k)})^{-1} = \begin{pmatrix} I_{k-1} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -l^{(k)} & I_{m-k} \end{pmatrix} \cdot (L^{(k-1)})^{-1} \ .$$

Now a straightforward induction using this relation and (2.24) shows that $A^{(k)}$ is the last $m-k+1$ rows of $(L^{(k)})^{-1} \cdot A$ for all k. On the other hand

$$(L^{(k)})^{-1} = \begin{pmatrix} L^{-1}_{1:k,1:k} & 0 \\ -L_{k+1:m,1:k} \cdot L^{-1}_{1:k,1:k} & I_{m-k} \end{pmatrix} ,$$

where

$$\begin{aligned} L_{k+1:m,1:k} \cdot L^{-1}_{1:k,1:k} &= (L_{k+1:m,1:k} \cdot U_{1:k,1:k}) \cdot (L_{1:k,1:k} \cdot U_{1:k,1:k})^{-1} \\ &= C_{k+1:m,1:k} \cdot C^{-1}_{1:k,1:k} \ . \end{aligned}$$

Hence

$$(L^{(k)})^{-1} \cdot A = \begin{pmatrix} L^{-1}_{1:k,1:k} A_{1:k} \\ A_{k+1:m} - C_{k+1:m,1:k} \cdot C^{-1}_{1:k,1:k} \cdot A_{1:k} \end{pmatrix} ,$$

from which equation (2.25) follows. Similar arguments show that equation (2.26) holds. ∎

To compute the generator of Z, we define

$$Z^{(k)} = C_{k+1:m,1:k} \cdot C^{-1}_{1:k,1:k} \quad \text{for} \quad k = 1, 2, \cdots, n \; . \tag{2.27}$$

In particular, for $k = n$ we have $Z^{(n)} = Z$ (see (2.12)). Theorem 2.1 implies that every $Z^{(k)}$ is a Cauchy-like matrix of displacement rank at most r. In the following we introduce a simple recursion for computing $\{Z^{(k)}\}$ via $\{A^{(k)}\}$ and $\{B^{(k)}\}$.

First, by applying Theorem 2.1 to $Z^{(1)}$ we have

$$\Omega_{2:m,2:m} \cdot Z^{(1)} - Z^{(1)} \cdot \omega_1 = A^{(2)} \cdot Y^{(1)} \; ,$$

where $Y^{(1)} = B_{:,1}/\gamma$.

To compute the generator for $Z^{(k)}$, we set

$$y^{(k)} = B^{(k)}_{:,1}/U_{k,k} \quad , \quad z^{(k)} = \left(A^{(k)}_{1,:} \cdot Y^{(k-1)} \right) \cdot (\omega_k I_k - \Lambda_{1:k-1,1:k-1})^{-1}$$

and

$$Y^{(k)} = \left(Y^{(k-1)} - y^{(k)} \cdot z^{(k)} \quad y^{(k)} \right) \in \mathbf{C}^{r \times k} \; . \tag{2.28}$$

Theorem 2.5 *for $k = 2, \cdots, n$,*

$$\Omega_{k+1:m} \cdot Z^{(k)} - Z^{(k)} \cdot \Omega_{1:k} = A^{(k+1)} \cdot Y^{(k)} \; .$$

Proof. Combining Theorem 2.1 and equation (2.25), we see that the generator of $Z^{(k)}$ is simply $A^{(k+1)} \cdot B_{:,1:k} \cdot C^{-1}_{1:k,1:k}$ for all k. Hence in the following we will show inductively that

$$Y^{(k-1)} = B_{:,1:k-1} \cdot C^{-1}_{1:k-1,1:k-1} \; . \tag{2.29}$$

Equation (2.29) is obviously true for $k = 2$ by construction. Assuming that it holds for some $k \geq 2$, we want to show it to hold for $k + 1$. To this end, partition

$$C_{1:k,1:k} = \begin{pmatrix} C_{1:k-1,1:k-1} & C_{1:k-1,k} \\ C_{k,1:k-1} & C_{k,k} \end{pmatrix} \; .$$

It is well-known that $U_{k,k}$ is the Schur complement of $C_{1:k-1,1:k-1}$,

$$U_{k,k} = C_{k,k} - C_{k,1:k-1} \cdot C^{-1}_{1:k-1,1:k-1} \cdot C_{1:k-1,k} \; .$$

Hence

$$C^{-1}_{1:k,1:k} = \begin{pmatrix} C^{-1}_{1:k-1,1:k-1} + f \cdot g/U_{k,k} & -f/U_{k,k} \\ -g/U_{k,k} & 1/U_{k,k} \end{pmatrix} \; ,$$

where $f = C^{-1}_{1:k-1,1:k-1} \cdot C_{1:k-1,1:k}$ and $g = C_{k,1:k-1} \cdot C^{-1}_{1:k-1,1:k-1}$. Consequently

$$\begin{aligned} B_{1:k} \cdot C^{-1}_{1:k,1:k} &= (B_{1:k-1} \quad B_{:,k}) \cdot \begin{pmatrix} C^{-1}_{1:k-1,1:k-1} + f \cdot g/U_{k,k} & -f/U_{k,k} \\ -g/U_{k,k} & 1/U_{k,k} \end{pmatrix} \\ &= \left(B_{1:k-1} \cdot C^{-1}_{1:k-1,1:k-1} - \hat{f} \cdot g/U_{k,k} \quad \hat{f}/U_{k,k} \right) \; , \end{aligned} \tag{2.30}$$

where

$$\hat{f} = B_{:,k} - B_{1:k-1} \cdot f = B_{:,k} - B_{1:k-1} \cdot C^{-1}_{1:k-1,1:k-1} \cdot C_{1:k-1,1:k} \ ,$$

which, according to (2.26), is simply $B^{(k)}_{:,1}$.

On the other hand, we observe that g is the first row of $Z^{(k-1)}$, which satisfies the displacement equation

$$\Omega_{k:m} \cdot Z^{(k-1)} - Z^{(k-1)} \cdot \Omega_{1:k-1} = A^{(k)} \cdot Y^{(k-1)} \ .$$

Hence

$$g = \Big((A^{(k)}_{1,:} \cdot Y^{(k)}\Big) \cdot (\omega_k I_{k-1} - \Omega_{1:k-1})^{-1} \ .$$

By setting $\hat{f}/U_{k,k} = y^{(k)}$ and $g = z^{(k)}$ in (2.30) and comparing the resulting equation with (2.28) and (2.29), we have $Y^{(k)} = B_{1:k} \cdot C^{-1}_{1:k,1:k}$. ∎

2.3 Pivoting and Generator Re-decomposition

The factorization (2.19) does not always exit. One celebrated property of the Cauchy-like matrix C is that performing partial and complete pivoting on C does not change the Cauchy-like structure (see Heinig [25]). To perform pivoting on C, one finds a large magnitude entry $(k_{\max}, j_{\max})$ in C, permutes it to the $(1,1)$ entry to get $\hat{C} \equiv P_m(1, k_{\max}) \cdot C \cdot Q_n(1, j_{\max})$, and then applies the elimination step to $\hat{C}$. Let C be a Cauchy-like matrix satisfying equation (1.3). Then for every $1 \leq k \leq m$ and $1 \leq j \leq n$,

$$\begin{aligned} & (P_m(1,k) \cdot \Omega \cdot P_m(1,k)) \cdot \hat{C} - \hat{C} \cdot (Q_n(1,j) \cdot \Lambda \cdot Q_n(1,j)) \\ = \ & (P_m(1,k) \cdot A) \cdot (B \cdot Q_n(1,j)) \ . \end{aligned}$$

It follows that $\hat{C}$ is still a Cauchy-like matrix.

To perform partial pivoting, one chooses $j_{max} = 1$; finds the largest magnitude entry $(k_{\max}, 1)$ in the first column of C; and permutes the $k^{th}_{\max}$ and the first rows of C. As in straightforward GEPP for dense matrices, performing partial pivoting ensures that an LU factorization can always be computed. There is, however, a potential problem of *element growth.* Let

$$g_{PP} \equiv \max_{1 \leq k \leq n} \|C^{(k)}\|_{\max} / \|C\|_{\max}$$

be the *element growth factor.* It is well-known that $g_{PP} \leq 2^{n-1}$ for GEPP, and although very rare, this bound is attainable for certain dense matrices (see Golub and van Loan [21, pages 115-116]). It is not clear whether this bound is attainable for Cauchy-like matrices with low displacement rank. When large element growth does occur, the computed LU factorizations can have a large backward error.

One way to reduce this element growth is to perform complete pivoting, whereby one chooses the largest magnitude entry in the entire matrix C and permutes it to the $(1,1)$ entry. This is an overall $O(mn^2)$ procedure. To reduce the cost, we adopt the fast variation of complete pivoting proposed in Gu [23] to find an entry that is sufficiently large in magnitude. Since this method involves re-decomposing the generators of both $C^{(k)}$ and $Z^{(k-1)}$, in the following we show how it works on $C^{(k)}$ instead of on C.

In equation (2.23), we QR factorize $A^{(k)}$ to get $A^{(k)} = \mathcal{A} \cdot R$, where $\mathcal{A}$ is column unitary, and R is upper-triangular. We then compute $\mathcal{B} = R \cdot B^{(k)}$ and $\mathcal{Y} = R \cdot Y^{(k-1)}$. It follows that

$$A^{(k)} \cdot B^{(k)} = \mathcal{A} \cdot \mathcal{B} \quad \text{and} \quad A^{(k)} \cdot Y^{(k-1)} = \mathcal{A} \cdot \mathcal{Y} .$$

In other words, we have re-decomposed the generators of $C^{(k)}$ and $Z^{(k)}$ as $\mathcal{A} \cdot \mathcal{B}$ and $\mathcal{A} \cdot \mathcal{Y}$, respectively.

Since $\mathcal{A}$ is column unitary, the j^{th} columns of $A^{(k)} \cdot B^{(k)}$ and $\mathcal{B}$ have the same 2-norm, for all j. Hence we choose $j_{\max}$ by looking for the largest 2-norm column of $\mathcal{B}$. We then choose the $(k_{\max}, j_{\max})$ entry to be the largest magnitude entry in the $j_{\max}^{th}$ column of $C^{(k)}$. The following lemma relates $|C^{(k)}_{k_{\max},j_{\max}}|$ to the max and Frobenius norms of $C^{(k)}$; it is a straightforward generalization of Lemma 2.1 in [23].

Lemma 2.2 *Let C be a Cauchy-like matrix satisfying equation (1.3). Then*

$$\|C^{(k)}\|_{\max} \leq \sqrt{m} \cdot \psi \cdot |C^{(k)}_{k_{\max},j_{\max}}| \quad \textit{and} \quad \|C^{(k)}\|_F \leq \sqrt{m\,n} \cdot \psi \cdot |C^{(k)}_{k_{\max},j_{\max}}| ,$$

where ψ is defined in (2.9).

In addition to the potential element growth in the LU factorization, Sweet and Brent [44] show that the matrices $A^{(k)}$ and $B^{(k)}$, if updated as in equation (2.24), could also grow so that

$$\left\| |A^{(k)}| \cdot |B^{(k)}| \right\|_2 \gg \left\| A^{(k)} \cdot B^{(k)} \right\|_2$$

for some k. And if this happens, the backward error in the LU factorization could be large. The same arguments show that if

$$\left\| |A^{(k)}| \cdot |Y^{(k-1)}| \right\|_2 \gg \left\| A^{(k)} \cdot Y^{(k-1)} \right\|_2 ,$$

then the backward error in the computed Z matrix could be large. However, such element growth goes away when re-decomposition is performed; in fact it will not occur as long as $A^{(k)}$ is well-conditioned (see [23] and §4).

Algorithm 1 below computes an LU-factorization of the form

$$C = P_m \cdot L \cdot U \cdot (Q_n)^T , \tag{2.31}$$

where $L \in \mathbf{C}^{m\times n}$ is unit lower triangular and $U \in \mathbf{C}^{n\times n}$ is upper triangular; and $P_m \in \mathbf{R}^{m\times m}$ and $Q_n \in \mathbf{R}^{n\times n}$ are permutation matrices. Let

$$\tilde{C} = (P_m)^T \cdot C \cdot Q_n .$$

Algorithm 1 also computes a generator of the matrix $Z = \tilde{C}_2 \cdot \tilde{C}_1^{-1}$. It performs column pivoting at every ζ steps, where ζ is a user supplied positive integer. In particular, Algorithm 1 performs column pivoting at every step if $\zeta = 1$. In practice we set $\zeta \gg r$. Algorithm 1 assumes that the matrix A is initially column unitary.

Algorithm 1 LU factorization of C with pivoting.

```
$L := 0 \in \mathbf{C}^{m \times n}$; $U := 0 \in \mathbf{C}^{n \times n}$; $P_m := I_m$; $Q_n := I_n$;
for $k := 1$ to $n$ do
    if $(\mathrm{mod}(k, \zeta) = 0)$ then
        $A_{k:m} := A_{k:m} \cdot R$ (QR factorizes $A_{k:m}$);
        $B_{k:n} := R \cdot B_{k:n}$; $Y_{:,1:k-1} := R \cdot Y_{:,1:k-1}$;
        $j_{\max} := \mathrm{argmax}_{k \le j \le n} \|B_{:,j}\|_2$;
        if $j_{\max} > k$ then
            $Q_n := Q_n(k, j_{\max}) \cdot Q_n$; $\Lambda := Q_n(k, j_{\max}) \cdot \Lambda \cdot Q_n(k, j_{\max})$;
            $B := B \cdot Q_n(k, j_{\max})$; $U := U \cdot Q_n(k, j_{\max})$;
        endif
    endif
    $L_{k:m,k} := (\Omega_{k:m} - \lambda_k I_{m-k+1})^{-1} \cdot A_{k:m,:} \cdot B_{:,k}$;
    $k_{\max} := \mathrm{argmax}_{k \le j \le m} |L_{j,k}|$;
    if $k_{\max} > k$ then
        $P_m := P_m \cdot P_m(k, k_{\max})$; $\Omega := P_m(k, k_{\max}) \cdot \Omega \cdot P_m(k, k_{\max})$;
        $A := P_m(k, k_{\max}) \cdot A$; $L := P_m(k, k_{\max}) \cdot L$;
    endif
    $U_{k,k} := L_{k,k}$; $U_{k,k+1:n} := A_{k,:} \cdot B_{k+1:n} \cdot (\omega_k I_{n-k} - \Lambda_{k+1:n})^{-1}$;
    $L_{k,k} := 1$; $L_{k+1:m,k} := L_{k+1:m,k} / U_{k,k}$;
    $A_{k+1:m} := A_{k+1:m} - L_{k+1:m,k} \cdot A_{k,:}$; $B_{k+1:n} := B_{k+1:n} - B_{:,k} \cdot U_{k,k+1:n} / U_{k,k}$;
    $z := A_{k,:} \cdot Y_{:,1:k-1} \cdot (\omega_k I_{k-1} - \Lambda_{1:k-1})^{-1}$; $y := B_{:,k} / U_{k,k}$;
    $Y_{:,1:k} := (Y_{:,1:k-1} - y \cdot z \quad y)$;
endfor
```

Remark 3: If the input data A, B, Ω and Λ are real, Algorithm 1 costs about $(4r + 4)mn + (2r - 1)n^2$ flops, plus an extra cost of about $(4m - n)nr^2/\zeta$ flops for redecomposing the generators; there is also potentially about $n^2(\zeta + 1)/(2\zeta)$ swaps of memory locations. For a matrix transformed into a Cauchy-like matrix from a Toeplitz-plus-Hankel matrix (see §1 and §3), the displacement rank r is at most 4. If we choose $\zeta \gg 4$, then the cost of Algorithm 1 is about $(20m + 7n)n$ flops.

Remark 4: If the input data are complex, Algorithm 1 costs about $(16r + 22)mn + 8rn^2$ flops, plus the cost for comparisons and the cost of about $4(4m - n)nr^2/\zeta$ flops for redecomposing the generators; there is also potentially about $n^2(\zeta + 1)/\zeta$ swaps of memory locations. For a matrix transformed into a Cauchy-like matrix from a Toeplitz matrix (see §3), the displacement rank r is at most 2. If we choose $\zeta \gg 2$, and choose pivots that have the largest sum of real and imaginary parts in absolute value, the cost for comparisons is about $4mn - 2n^2$ flops, and the cost of Algorithm 1 is about $2(29m + 7n)n$ flops.

Remark 5: If $r \gg \max(\zeta, 1)$, then the cost of redecomposing the generators dominates the computation of Algorithm 1. In this case we use QR updating techniques similar to those of Daniel, Gragg, Kaufman and Stewart [16] (see also Golub and van Loan [21, §12]) to perform the QR factorization at every step, bringing the total redecomposition cost down to $O(mnr)$.

To get an upper bound on the element growth factor for Algorithm 1, let

$$\mathcal{W}(k) = \left(k \prod_{s=2}^{k} s^{1/(s-1)}\right)^{1/2} = O\left(k^{\frac{1}{2}+\frac{1}{4}\ln k}\right).$$

This is Wilkinson's upper bound on the growth factor for GECP on a $k \times k$ general dense matrix. Although $\mathcal{W}(k)$ is not a polynomial in k, it does not grow very fast either [47]. The following theorem trivially generalizes a similar result in [23].

Theorem 2.6 *Let C be a Cauchy-like matrix satisfying (1.3); and let equation (2.31) be the LU factorization generated by Algorithm 1 in exact arithmetic for $\zeta = 1$. Then the element growth factor $g_{CP} \equiv \max_{1 \le k \le n} \|C^{(k)}\|_{\max}/\|C\|_{\max}$ satisfies*

$$g_{CP} \le \sqrt{n} \cdot \psi^{2+\sum_{k=1}^{n-1} 1/k} \cdot \mathcal{W}(n).$$

2.4 Factorizing A Positive Definite Cauchy-like Matrix

In this subsection, we compute the Cholesky factorization of a symmetric/Hermitian positive definite Cauchy-like matrix K of the form (2.12):

$$K = \mathcal{L} \cdot \mathcal{D} \cdot \mathcal{L}^*, \tag{2.32}$$

where $\mathcal{L} \in \mathbf{C}^{n\times n}$ is unit lower triangular and $\mathcal{D} \in \mathbf{R}^{n\times n}$ is positive diagonal. We will assume that K satisfies either equation (2.17) or equation (2.18). In both cases, the displacement equation specifies all the off-diagonal entries of K; the diagonal entries of K have to be provided separately.

The first step of Cholesky factorization is to zero-out the first column and row of K below and above the diagonal entry:

$$K = \begin{pmatrix} \gamma & v^* \\ v & K_{2:n,2:n} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ l & I_{n-1} \end{pmatrix} \cdot \begin{pmatrix} \gamma & 0 \\ 0 & K^{(2)} \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ l & I_{n-1} \end{pmatrix}^*, \tag{2.33}$$

where $\mathcal{D}_{1,1} = \gamma$; $\mathcal{L}_{2:m,1} \equiv l = v/\gamma$; and $K^{(2)} = K_{2:m,2:n} - v \cdot v^*/\gamma$ is the Schur complement of γ.

2.4.1 Real Cholesky Factorization

First assume that K satisfies equation (2.17). Then K is a real symmetric positive definite matrix. Similar to (2.20), $K^{(2)}$ is a Cauchy-like matrix. Theorem 2.7 below is a direct generalization of Theorem 2.4. More discussions on factorizing symmetric Cauchy-like matrices can be found in Kailath and Olshevsky [32].

Theorem 2.7 *Let matrix K satisfy the displacement equation (2.17). Then $K^{(2)}$ satisfies the displacement equation*

$$\Omega_{2:n} \cdot K^{(2)} - K^{(2)} \cdot \Omega_{2:n} = \mathcal{A}^{(2)} \cdot \mathcal{J} \cdot \left(\mathcal{A}^{(2)}\right)^T$$

with $\mathcal{A}^{(2)} = \mathcal{A}_{2:n} - l \cdot \mathcal{A}_{1,:}$. The diagonal entries of $K^{(2)}$ satisfy

$$K^{(2)}_{k-1,k-1} = K_{k,k} - v_k^2/\gamma\,, \quad \text{for} \quad k = 2, \cdots, n\,.$$

Hence the first step of Cholesky factorization on the matrix K involves computing v of K from equation (2.17); computing the vector l; computing the matrix $\mathcal{A}^{(2)}$; and computing the diagonal entries of $K^{(2)}$. Cholesky factorization then proceeds by recursively applying this step to $K^{(2)}$. Since K is symmetric positive definite, all the diagonal entries are positive and hence this procedure never breaks. At the end of this procedure, K is factored into (2.32) with $\mathcal{L} \in \mathbf{R}^{n\times n}$.

It is well-known that Cholesky factorization on a general dense symmetric positive definite matrix is always backward stable, and there is no element growth in $\mathcal{D}$. However, the above Cholesky factorization procedure may not necessarily be backward stable for K. This is because, as in Algorithm 1 without generator re-decomposition, there might be potential element growth in the generators. To avoid this problem, we perform diagonal pivoting on K and re-decompose the generators as well. Since K is symmetric positive definite, γ is the largest magnitude entry in the entire matrix K after diagonal pivoting. Algorithm 2 below computes a Cholesky factorization with diagonal pivoting:

$$K = Q_n \cdot \mathcal{L} \cdot \mathcal{D} \cdot \mathcal{L}^T \cdot (Q_n)^T\,,$$

where $\mathcal{L} \in \mathbf{R}^{n\times n}$ is unit lower triangular and $\mathcal{D} \in \mathbf{R}^{n\times n}$ is positive diagonal. Algorithm 2 performs generator redecomposition every ζ steps; and it assumes that on input, the diagonal entries of $\mathcal{D}$ are those of K and that $\mathcal{A}$ is well-conditioned.

Algorithm 2 Real Cholesky Factorization

```
  $\mathcal{L} := 0 \in \mathbf{R}^{n\times n}$; $Q_n := I_n$;
  for $k := 1$ to $n$ do
      if $(\mathrm{mod}(k,\zeta) = 0)$ then
          $\mathcal{A}_{k:n} := \mathcal{A}_{k:n} \cdot R$ (QR factorizes $\mathcal{A}_{k:n}$); $\mathcal{J} := R \cdot \mathcal{J} \cdot R^T$;
      endif
      $k_{\max} := \mathrm{argmax}_{k\le j\le n}\, \mathcal{D}_{j,j}$;
      if $k_{\max} > k$ then
          $Q_n := Q_n \cdot Q_n(k, k_{\max})$; $\Omega_{1:n} := Q_n(k, k_{\max}) \cdot \Omega_{1:n} \cdot Q_n(k, k_{\max})$;
          $\mathcal{D} := Q_n(k, k_{\max}) \cdot \mathcal{D} \cdot Q_n(k, k_{\max})$;
          $\mathcal{A} := Q_n(k, k_{\max}) \cdot \mathcal{A}$; $\mathcal{L}_{:,1:k} := Q_n(k, k_{\max}) \cdot \mathcal{L}_{:,1:k}$;
      endif
```

```
        $\mathcal{L}_{k+1:n,k} := (\Omega_{k+1:n} - \omega_k I_{n-k})^{-1} \cdot \mathcal{A}_{k+1:n} \cdot \mathcal{J} \cdot \mathcal{A}^T_{k,:}/\mathcal{D}_{k,k}$;
        $\mathcal{A}_{k+1:n} := \mathcal{A}_{k+1:n} - \mathcal{L}_{k+1:n,k} \cdot \mathcal{A}_{k,:}$;
        for $j := k+1$ to $n$ do
            $\mathcal{D}_{j,j} := \mathcal{D}_{j,j} - \mathcal{L}^2_{j,k} \cdot \mathcal{D}_{k,k}$.
        endfor
    endfor
```

Remark 6: Algorithm 2 costs about $(4r+2.5)n^2$ flops, where $2r$ is the displacement rank of K (see equation (2.17)), plus an extra cost of about $2nr^2/\zeta$ flops for redecomposing the generators; there is also potentially about $n^2/2$ swaps of memory locations. For a matrix transformed into a Cauchy-like matrix from a Toeplitz-plus-Hankel matrix (see §1 and §3), the displacement rank of K is $2r \leq 8$. In this case, Algorithm 2 costs about $18.5n^2$ flops for $\zeta \gg 4$.

2.4.2 Complex Cholesky Factorization

Now assume that K satisfies equation (2.18). Then K is a complex Hermitian positive definite matrix. Again $K^{(2)}$ in (2.33) is a Cauchy-like matrix. Theorem 2.8 below is a slight modification of Lemma 3.2 in [32].

Theorem 2.8 *Let matrix K satisfy the displacement equation (2.18). Then $K^{(2)}$ satisfies the displacement equation*

$$K^{(2)} - \Omega^*_{2:n} \cdot K^{(2)} \cdot \Omega_{2:n} = \mathcal{A}^{(2)} \cdot \mathcal{J} \left(\mathcal{A}^{(2)}\right)^*$$

with

$$\mathcal{A}^{(2)} = \mathcal{A}_{2:n} - (l - t/2) \cdot \mathcal{A}_{1,:} \quad \textit{where} \quad t = \mathcal{A}_{2:n} \cdot \mathcal{J} \cdot \mathcal{A}^*_{1,:}/\gamma \ .$$

The diagonal entries of $K^{(2)}$ satisfy

$$K^{(2)}_{k-1,k-1} = K_{k,k} - |v_k|^2/\gamma \ , \quad \textit{for} \quad k = 2, \cdots, n \ .$$

Similar to Algorithm 2, Algorithm 3 belows computes a Cholesky factorization with diagonal pivoting:

$$H = Q_n \cdot \mathcal{L} \cdot \mathcal{D} \cdot \mathcal{L}^* \cdot (Q_n)^T \ , \tag{2.34}$$

where $\mathcal{L} \in \mathbf{C}^{n\times n}$ is unit lower triangular and $\mathcal{D} \in \mathbf{R}^{n\times n}$ is positive diagonal. Algorithm 3 performs generator redecomposition every ζ steps; and it assumes that on input, the diagonal entries of $\mathcal{D}$ are those of K and that $\mathcal{A}$ is well-conditioned.

Algorithm 3 Complex Cholesky Factorization

```
    $\mathcal{L} := 0 \in \mathbf{R}^{n\times n}$; $Q_n := I_n$;
    for $k := 1$ to $n$ do
        if $(\mathrm{mod}(k, \zeta) = 0)$ then
        $\mathcal{A}_{k:n} := \mathcal{A}_{k:n} \cdot R$ (QR factorizes $\mathcal{A}_{k:n}$); $\mathcal{J} := R \cdot \mathcal{J} \cdot R^*$;
    endif
    $k_{\max} := \mathrm{argmax}_{k \le j \le n} \mathcal{D}_{j,j}$;
    if $k_{\max} > k$ then
        $Q_n := Q_n \cdot Q_n(k, k_{\max})$; $\Omega_{1:n} := Q_n(k, k_{\max}) \cdot \Omega_{1:n} \cdot Q_n(k, k_{\max})$;
        $\mathcal{D} := Q_n(k, k_{\max}) \cdot \mathcal{D} \cdot Q_n(k, k_{\max})$;
        $\mathcal{A} := Q_n(k, k_{\max}) \cdot \mathcal{A}$; $\mathcal{L}_{:,1:k} := Q_n(k, k_{\max}) \cdot \mathcal{L}_{:,1:k}$;
    endif
    $t := \mathcal{A}_{k+1:n} \cdot \mathcal{J} \cdot \mathcal{A}^*_{k,:} / \mathcal{D}_{k,k}$; $\mathcal{L}_{k+1:n,k} := \left(I_{n-k} - \omega_k \cdot \bar{\Omega}_{k+1:n}\right)^{-1} \cdot t$;
    $\mathcal{A}_{k+1:n} := \mathcal{A}_{k+1:n} - (\mathcal{L}_{k+1:n,k} - t/2) \cdot \mathcal{A}_{k,:}$;
    for $j := k + 1$ to $n$ do
        $\mathcal{D}_{j,j} := \mathcal{D}_{j,j} - |\mathcal{L}_{j,k}|^2 \cdot \mathcal{D}_{k,k}$.
    endfor
endfor
```

Remark 7: Algorithm 3 costs about $(16r + 14)n^2$ flops, where $2r$ is the displacement rank of K (see equation (2.18)), plus an extra cost of about $8nr^2/\zeta$ flops for redecomposing the generators; there is also potentially about n^2 swaps of memory locations. For a matrix transformed into a Cauchy-like matrix from a Toeplitz matrix (see §3), the displacement rank of K is $2r \le 4$. In this case, Algorithm 1 costs about $46n^2$ flops for for $\zeta \gg 2$.

3 Toeplitz and Toeplitz-plus-Hankel Least Squares Problems

3.1 Toeplitz Least Squares Problems

For any integer $k > 0$, define

$$T_k^{(\delta)} = \begin{pmatrix} 0 & 0 & \cdots & 0 & \delta \\ 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix} \in \mathbf{R}^{k \times k} .$$

Let $\Omega = T_m^{(1)}$ and $\Lambda = T_n^{(\delta)}$. It is easy to show that every m-by-n Toeplitz matrix satisfies the displacement equation (1.1) with Δ having non-zero entries only in its first row and last column, and hence the displacement rank is $\mathrm{rank}(\Delta) \le 2$. To ensure that equation (1.1) always has a unique solution, we choose $\delta > 1$ in Λ. The following result is a generalization of that in Heinig [25].

Lemma 3.1 *Let $M \in \mathbf{C}^{m \times n}$ be a matrix satisfying the displacement equation*

$$T_m^{(1)} \cdot M - M \cdot T_n^{(\delta)} = A \cdot B \, , \tag{3.35}$$

where $A \in \mathbf{C}^{m\times r}$ and $B \in \mathbf{C}^{r\times n}$. Then $C \equiv \mathcal{F}_m \cdot M \cdot \mathcal{G}^{-1} \cdot \mathcal{F}_n^*$ is a Cauchy-like matrix:

$$\Omega \cdot C - C \cdot \Lambda = (\mathcal{F}_m \cdot A) \cdot \left(B \cdot \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^*\right) , \tag{3.36}$$

where

$$\mathcal{F}_m = \sqrt{\frac{1}{m}} \cdot \left(e^{\frac{2\pi i}{m}(k-1)(j-1)}\right)_{1\leq k,j\leq m} \quad \textit{and} \quad \mathcal{F}_n = \sqrt{\frac{1}{n}} \cdot \left(e^{\frac{2\pi i}{n}(k-1)(j-1)}\right)_{1\leq k,j\leq n}$$

are the normalized Inverse Discrete Fourier Transform matrices;

$$\mathcal{G} = \mathrm{diag}(1, \tilde{\delta}, \cdots, \tilde{\delta}^{n-1}) \quad \textit{where} \quad \tilde{\delta} = \delta^{\frac{1}{n}} ;$$

and

$$\Omega = \mathrm{diag}\left(1, e^{\frac{2\pi i}{m}}, \cdots, e^{\frac{2\pi i}{m}(m-1)}\right) \quad \textit{and} \quad \Lambda = \tilde{\delta} \cdot \mathrm{diag}\left(1, e^{\frac{2\pi i}{n}}, \cdots, e^{\frac{2\pi i}{n}(n-1)}\right) .$$

The diagonals of Ω and Λ satisfy

$$\psi_{\max} \leq \tilde{\delta} + 1 \; , \; \psi \leq \frac{\tilde{\delta}+1}{\tilde{\delta}-1} \; , \; \phi_{\max} \leq 2 \;\; \textit{and} \;\; \phi \leq \left(\sin\frac{\pi}{m}\right)^{-1} ,$$

where $\psi_{\max}$, ψ, and $\phi_{\max}$, ϕ are defined in (2.9) and (2.10).

Proof. It is well-known that

$$T_m^{(1)} = (\mathcal{F}_m)^* \cdot \Omega \cdot \mathcal{F}_m .$$

On the other hand, $T_n^{(\delta)} = \tilde{\delta} \cdot \mathcal{G}^{-1} \cdot T_n^{(1)} \cdot \mathcal{G}$ and hence

$$T_n^{(\delta)} = \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^* \cdot \Lambda \cdot \mathcal{F}_n \cdot \mathcal{G} .$$

Equation (3.36) follows by plugging these relations into (3.36) and simplifying.

The relations for $\psi_{\max}$, ψ, and $\phi_{\max}$, ϕ follow from direct verification. ▮

A matrix M is called *Toeplitz-like* if it satisfies the displacement equation (3.35) with $r \ll n$ (cf. [18]). To solve the least squares problem (1.5) when M is a Toeplitz-like matrix, we transform M into a Cauchy-like matrix using Lemma 3.1. Setting

$$M = (\mathcal{F}_m)^* \cdot C \cdot \mathcal{F}^{(n)} \cdot \mathcal{G}$$

in (1.6) and simplifying,

$$\begin{aligned} x_M &= \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^* \cdot (C^* \cdot C)^{-1} \cdot C^* \cdot (\mathcal{F}_m \cdot K) \\ &= \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^* \cdot x_C , \end{aligned} \tag{3.37}$$

where x_C is the solution to the Cauchy-like least squares problem

$$\min_x \|C \cdot x - (\mathcal{F}_m \cdot h)\|_2 .$$

The idea of transforming a Toeplitz linear system of equations into a Cauchy-like system of linear equations was first proposed by Heinig [25].

We summarize the above into Algorithm 4 that follows, assuming that M satisfies equation (3.35) with A column orthogonal.

Algorithm 4 Solving the Toeplitz-like Least Squares Problem

1. Choose δ; compute Ω and Λ in Lemma 3.1;
2. compute $h := \mathcal{F}_m \cdot h$; $A := \mathcal{F}_m \cdot A$; and $B := B \cdot \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^*$;
3. choose ζ_1; compute the LU factorization for C_1 in (2.13) and the generator for Z in (2.12) via Algorithm 1;
4. choose ζ_3; compute the initial generator for K in (2.12) and compute its Cholesky factorization via Algorithm 3;
5. compute the right hand side in (2.14) and compute x_C by solving the linear systems (2.13) and (2.14) via forward and backward substitution;
6. compute $x_M := \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^* \cdot x_C$.

Remark 8: The total cost of Steps 1, 2, and 6 is $O(m \log_2 m)$ flops via the forward and backward FFTs; the cost of Step 3 is about $(16r + 22)mn + 8rn^2$ flops, plus the cost for comparisons and the cost of about $4(4m - n)nr^2/\zeta_1$ flops for redecomposing the generators (see Remark 4); the cost of Step 4 is about $(16r + 11)mn + 3n^2$ flops, plus an extra cost of about $8nr^2/\zeta_3$ flops for redecomposing the generators (see Remarks 2 and 7); Since the matrix Z is computed in Step 4 (see Remark 2), the cost for Step 5 is about $8(m + n)n$ flops. Hence the total cost of Algorithm 4 is about $(32r + 41)mn + (8r + 11)n^2$ flops, plus the cost for comparisons and generator redecompositions. In particular, let M be a Toeplitz matrix, then $r \leq 2$. We choose pivots in Algorithm 1 to have the largest sum of real and imaginary parts in absolute value so that the cost for comparisons is about $4mn - 2n^2$ flops (see Remark 4), and we choose $\zeta_1 \gg 2$ and $\zeta_3 \gg 2$. Thus the total cost of Algorithm 4 is about $(109m + 25n)n$ flops.

3.2 Toeplitz-plus-Hankel Least Squares Problems

For any integer $k > 0$, define

$$\mathcal{T}_k^{(\delta)} = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 1 & \cdots & \vdots \\ 0 & 1 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & 0 & 1 & \delta \end{pmatrix} \in \mathbf{R}^{k \times k} .$$

Since $\mathcal{T}_k^{(\delta)}$ is symmetric, it has an eigendecomposition of the form

$$\mathcal{T}_k^{(\delta)} = \mathcal{Q}_k^{(\delta)} \cdot \mathcal{D}_k^{(\delta)} \cdot \left(\mathcal{Q}_k^{(\delta)}\right)^T ,$$

where $\mathcal{Q}_k^{(\delta)}$ is orthogonal and $\mathcal{D}_k^{(\delta)}$ is diagonal. $\mathcal{Q}_k^{(\delta)}$ is a fast cosine transformation matrix for $\delta = \pm 1$:

$$\begin{aligned} \mathcal{Q}_k^{(1)} &= \sqrt{\frac{2}{k}} \cdot \left(q_j \cos \frac{(2p-1)(j-1)\pi}{2n} \right)_{1 \le p,j \le k} , \\ \mathcal{Q}_k^{(-1)} &= \sqrt{\frac{2}{k}} \cdot \left(\cos \frac{(2p-1)(2j-1)\pi}{4k} \right)_{1 \le p,j \le k} , \end{aligned}$$

where $q_1 = \frac{1}{2}$ and $q_j = 1$ for $2 \le j \le k$; and the corresponding diagonal matrices are

$$\begin{aligned} \mathcal{D}_k^{(1)} &= 2 \cdot \mathrm{diag}\left(1, \cos \frac{\pi}{k}, \cdots, \cos \frac{(k-1)\pi}{k}\right) , \\ \mathcal{D}_k^{(-1)} &= 2 \cdot \mathrm{diag}\left(\cos \frac{\pi}{2k}, \cos \frac{3\pi}{2k}, \cdots, \cos \frac{(2k-1)\pi}{2k}\right) . \end{aligned}$$

Let $\Omega = \mathcal{T}_m^{(\delta_1)}$ and $\Lambda = \mathcal{T}_n^{(\delta_2)}$. It is easy to verify that every Toeplitz-plus-Hankel matrix satisfies the displacement equation (1.1) with Δ having non-zero entries only in its first and last rows and columns, thus the displacement rank of a Toeplitz-plus-Hankel matrix is $\mathrm{rank}(\Delta) \le 4$ (cf. [18, 27]). In particular, This result is true for every Toeplitz or Hankel matrix. Let $\gcd(m,n)$ be the *greatest common divider* of m and n.

Lemma 3.2 *Let $M \in \mathbf{C}^{m \times n}$ is a matrix satisfying the displacement equation*

$$\mathcal{T}_m^{(\delta_1)} \cdot M - M \cdot \mathcal{T}_n^{(\delta_2)} = A \cdot B , \tag{3.38}$$

where $A \in \mathbf{C}^{m \times r}$ and $B \in \mathbf{C}^{r \times n}$. Then $C \equiv \left(\mathcal{Q}_m^{(\delta_1)}\right)^T \cdot M \cdot \mathcal{Q}_n^{(\delta_2)}$ is a Cauchy-like matrix satisfying:

$$\Omega \cdot C - C \cdot \Lambda = \left(\left(\mathcal{Q}_m^{(\delta_1)}\right)^T \cdot A \right) \cdot \left(B \cdot \mathcal{Q}_n^{(\delta_2)} \right) ,$$

where $\Omega = \mathcal{D}_m^{(\delta_1)}$ and $\Lambda = \mathcal{D}_n^{(\delta_2)}$. In particular, we choose $\delta_2 = -\delta_1$ where

$$\delta_1 = \begin{cases} 1 & \text{if } \dfrac{m}{\gcd(m,n)} \text{ is odd;} \\ -1 & \text{otherwise.} \end{cases}$$

Then the diagonals of Ω and Λ satisfy

$$\psi_{\max} \le 4 \ , \ \psi \le \left(\sin \frac{\gcd(m,n) \cdot \pi}{4m \cdot n} \right)^{-2} , \ \phi_{\max} \le 4 \ \text{and} \ \phi \le \left(\sin \frac{\pi}{2m} \right)^{-2} ,$$

where $\psi_{\max}$, ψ, and $\phi_{\max}$, ϕ are defined in (2.9) and (2.10).

Proof. The displacement equation can be proved using arguments similar to those in the proof of Lemma 3.1.

As to the relations for $\psi_{\max}$, ψ, and $\phi_{\max}$, ϕ, it is obvious that $\psi_{\max} \le 4$ and $\phi_{\max} \le 4$. Let $m_1 = m/\gcd(m,n)$ and $n_1 = n/\gcd(m,n)$. Then both m_1 and n_1 are integers and $\gcd(m_1, n_1) = 1$.

First assume that m_1 is odd. Then $\delta_1 = 1$ and $\delta_2 = -1$. It follows that

$$\begin{aligned}
|\omega_k - \lambda_j| &= 2 \cdot \left|\cos\frac{(k-1)\pi}{m} - \cos\frac{(2j-1)\pi}{2n}\right| \\
&= 4\left|\sin\left(\frac{2\left((k-1)\cdot n_1 - j\cdot m_1\right) + m_1}{4m_1 \cdot n_1 \cdot \gcd(m,n)}\cdot\pi\right)\right| \\
&\quad \cdot\left|\sin\left(\frac{2\left((k-1)\cdot n_1 + j\cdot m_1\right) - m_1}{4m_1 \cdot n_1 \cdot \gcd(m,n)}\cdot\pi\right)\right| \\
&\geq 4\left|\sin\left(\frac{\pi}{4m_1\cdot n_1\cdot\gcd(m,n)}\right)\cdot\sin\left(\frac{\pi}{4m_1\cdot n_1\cdot\gcd(m,n)}\right)\right| \\
&= 4\sin^2\frac{\gcd(m,n)\cdot\pi}{4m\cdot n},
\end{aligned}$$

where we have used the fact that both

$$2\left((k-1)\cdot n_1 - j\cdot m_1\right) + m_1 \quad\text{and}\quad 2\left((k-1)\cdot n_1 + j\cdot m_1\right) - m_1$$

are odd integers and hence are never integer multiples of $4m_1 \cdot n_1 \cdot \gcd(m,n)$, the denominator.

On the other hand, if m_1 is even, then it follows that n_1 is odd, and that $\delta_1 = -1$ and $\delta_2 = 1$. Similar arguments show that $|\omega_k - \lambda_j| \geq 4\sin^2\frac{\gcd(m,n)\cdot\pi}{4m\cdot n}$.

In both cases, the bound on ϕ follows from direct verification. ∎

A matrix M is called *Toeplitz-plus-Hankell-like* if it satisfies the displacement equation (3.38) with $r \ll n$ (cf. [18]). In the least squares problem (1.5), if the matrix M is real Toeplitz-plus-Hankel-like, we transform it into a real Cauchy-like matrix C using Lemma 3.1:

$$M = \mathcal{Q}_m^{(\delta_1)} \cdot C \cdot \left(\mathcal{Q}_n^{(\delta_2)}\right)^T .$$

Plugging this relation into (1.6) and simplifying,

$$x_M = \mathcal{Q}_n^{(\delta_2)} \cdot x_C ,$$

where

$$x_C = \left(C^T \cdot C\right)^{-1} \cdot C^T \cdot \left(\left(\mathcal{Q}_m^{(\delta_1)}\right)^T \cdot h\right) .$$

Similar to Algorithm 4, Algorithm 5 that follows solves the least squares problem with M satisfying (3.38). We assume that both M and h are real.

Algorithm 5 Solving the Toeplitz-plus-Hankel-like Least Squares Problem

1. Choose δ_1 and δ_2 and compute Ω and Λ in Lemma 3.2;
2. compute $h := \left(\mathcal{Q}_m^{(\delta_1)}\right)^T \cdot h$; $A := \left(\mathcal{Q}_m^{(\delta_1)}\right)^T \cdot A$; and $B := B \cdot \mathcal{Q}_n^{(\delta_2)}$;
3. choose ζ_1; compute the LU factorization for C_1 in (2.13) and the generator for Z in (2.12) via Algorithm 1;
4. choose ζ_2; compute the initial generator for K in (2.12) and compute its Cholesky factorization via Algorithm 2;

5. compute the right hand side in (2.14) and compute x_C by solving the linear systems (2.13) and (2.14) via forward and backward substitution;

6. compute $x_M := \mathcal{Q}_n^{(\delta_2)} \cdot x_C$.

Remark 9: The total cost of Steps 1, 2, and 6 is $O(m \log_2 m)$ flops via fast cosine transforms; the cost of Step 3 is about $(4r + 4)mn + (2r - 1)n^2$ flops, plus the cost of about $(4m - n)nr^2/\zeta_1$ flops for redecomposing the generators (see Remark 3); the cost of Step 4 is about $(4r + 1)mn + 1.5n^2$ flops, plus an extra cost of about $2nr^2/\zeta_2$ flops for redecomposing the generators (see Remarks 1 and 6); Since the matrix Z is computed in Step 4 (see Remark 1), the cost for Step 5 is about $2(m + n)n$ flops. Hence the total cost of Algorithm 5 is about $(8r + 7)mn + (2r + 2.5)n^2$ flops, plus the cost for generator redecompositions. In particular, let M be a Toeplitz-plus-Hankel matrix, then $r \leq 4$. We choose $\zeta_1 \gg 4$ and $\zeta_2 \gg 4$. Thus the total cost of Algorithm 5 is about $(39m + 10.5n)n$ flops.

3.3 Efficiency and Accuracy Considerations

Every Toeplitz matrix is a Toeplitz-plus-Hankel matrix, hence Algorithm 5 is an algorithm for solving Toeplitz least squares problem as well. For a real Toeplitz least squares problem, Algorithm 4 performs 2.7 times as many flops as Algorithm 5 (see Remarks 8 and 9).

However, Algorithm 4 could be more accurate than Algorithm 5. The upper bound on g_{CP} in Theorem 2.6 and the error analysis of §4 suggest that the smaller ψ and ϕ are, the smaller the potential element growth and backward error. In our numerical experiments, we took $\delta = n$ in Algorithm 4. Hence $\psi = O(n)$ and $\phi = O(m)$ for Algorithm 4, whereas $\psi = O(m^2n^2/\gcd^2(m,n))$ and $\phi = O(m^2)$ for Algorithm 5. Thus the upper bound on g_{CP} in Theorem 2.6 is smaller for Algorithm 4. We will talk more about this trade-off between efficiency and stability in §4 and §5.

4 Error Analysis

In this section, we do a backward error analysis for Algorithms 1 through 5. In §4.1 we analyze Algorithm 1; in §4.2 we analyze Algorithms 2 and 3; and in §4.3 we analyze Algorithms 4 and 5.

4.1 Backward Errors in LU factorization

We assume that $\zeta = 1$ in Algorithm 1, so that column pivoting is done at every step; we also assume without loss of generality that no column or row swaps actually occur during the execution of Algorithm 1. At the end of §4 we discuss the case $\zeta > 1$.

Define

$$\hat{L} \cdot \hat{U} = C + H , \tag{4.39}$$

where $C \in \mathbf{C}^{m\times n}$ is the Cauchy-like matrix to be factored; $\hat{L}\cdot\hat{U}$ is the computed LU factorization; and $H \in \mathbf{C}^{m\times n}$ is the backward error matrix. Since the part of Algorithm 1 that computes the LU factorization is exactly Algorithm 2 of [23] on a rectangular matrix, we can obtain an ∞-norm upper bound on H by applying the error analysis of [23] to Algorithm 1 with a few simple changes on the matrix dimensions and related constants (see Theorem 4.1).

Let $\hat{Z}^{(k)} \in \mathbf{C}^{(m-k)\times k}$ be the computed $Z^{(k)}$ matrix and let $\hat{Z} = \hat{Z}^{(n)}$. To analyze the accuracy in $\hat{Z}^{(k)}$, we define $X^{(k)} \in \mathbf{C}^{m\times m}$ to be a lower triangular matrix with unit diagonal entries such that

1. $X^{(k)}_{k+1:m,1:k} = -Z^{(k)}$; and $X^{(k)}_{j,1:j-1} = -Z^{(j-1)}_{1,1:j-1}$ for $2 \le j \le k-1$.
2. All other entries of $X^{(k)}$ are zero.

Lemma 4.1 *Let $L^{(k)}$ be as defined in (2.22). Then $L^{(k)}\cdot X^{(k)} = I_m$.*

Proof. By definition, we can partition $X^{(k)}$ as

$$X^{(k)} = \begin{pmatrix} X^{(k)}_{1:k,1:k} & 0 \\ -Z^{(k)} & I_{n-k} \end{pmatrix} .$$

Since

$$(L^{(k)})^{-1} = \begin{pmatrix} L^{-1}_{1:k,1:k} & 0 \\ -L_{k+1:m,1:k}\cdot L^{-1}_{1:k,1:k} & I_{m-k} \end{pmatrix} = \begin{pmatrix} L^{-1}_{1:k,1:k} & 0 \\ -Z^{(k)} & I_{m-k} \end{pmatrix} ,$$

to show that $X^{(k)} = (L^{(k)})^{-1}$, it is sufficient to show that the first $k-1$ rows of $X^{(k)}$ and $(L^{(k)})^{-1}$ are the same. Since $(L^{(k)})^{-1}$ is unit lower triangular, for any $1 \le j \le k-1$, its j^{th} row is

$$\left(-L_{j,1:j-1}\cdot L^{-1}_{1:j-1,1:j-1}\ ,\quad 1\ ,\quad 0\right) ,$$

which is exactly the j^{th} row of $X^{(k)}$ by construction. ▮

Let $\hat{L}^{(k)}$ and $\hat{X}^{(k)}$ be the finite-precision counter-parts of $L^{(k)}$ and $X^{(k)}$, respectively. In §4.1.3, we inductively establish an ∞-norm upper bound on the matrix $E^{(k)}$ in the equation

$$\hat{L}^{(k)}\cdot\hat{X}^{(k)} = I_m + E^{(k)} ; \tag{4.40}$$

we also establish an ∞-norm upper bound on the matrix F in the equation

$$\hat{Z}\cdot\hat{L}_1\cdot\hat{U} = C_2 + F . \tag{4.41}$$

Relations (4.39) and (4.41) provide the backward error matrix produced by Algorithm 1:

$$\begin{pmatrix} I_n \\ \hat{Z} \end{pmatrix}\cdot\left(\hat{L}_1\cdot\hat{U}\right) = C + \begin{pmatrix} H_{1:n,1:n} \\ F \end{pmatrix} . \tag{4.42}$$

4.1.1 Preliminary Results

Assume that after the first $k-1$ steps of Gaussian elimination in finite arithmetic, we arrive at the the Cauchy-like matrix

$$\hat{C}^{(k)} = \begin{pmatrix} \gamma^{(k)} & u^{(k)} \\ v^{(k)} & \hat{C}^{(k)}_{2:m-k+1,2:n-k+1} \end{pmatrix}$$

that satisfies the displacement equation

$$\Omega_{k:m} \cdot \hat{C}^{(k)} - \hat{C}^{(k)} \cdot \Lambda_{k:n} = \hat{A}^{(k)} \cdot \hat{B}^{(k)} \,, \tag{4.43}$$

with $\hat{A}^{(k)}$ numerically column unitary. Let $\hat{Z}^{(k-1)}$ be the finite precision counter-part of $Z^{(k-1)}$ and partition it as

$$\hat{Z}^{(k-1)} = \begin{pmatrix} z^{(k-1)} \\ \hat{Z}^{(k-1)}_{2:m-k+1,1:k-1} \end{pmatrix} .$$

$\hat{Z}^{(k-1)}$ is a Cauchy-like matrix satisfying the displacement equation

$$\Omega_{k:m} \cdot \hat{Z}^{(k-1)} - \hat{Z}^{(k-1)} \cdot \Omega_{1:k-1} = \hat{A}^{(k)} \cdot \hat{Y}^{(k-1)} \,. \tag{4.44}$$

The k^{th} step of Gaussian elimination is to eliminate the first column of $\hat{C}^{(k)}$ below the diagonal:

$$\hat{C}^{(k)} = \begin{pmatrix} 1 & 0 \\ l^{(k)} & I_{m-k} \end{pmatrix} \cdot \begin{pmatrix} \gamma^{(k)} & u^{(k)} \\ 0 & C^{(k+1)} \end{pmatrix} ,$$

where

$$l^{(k)} = v^{(k)}/\gamma^{(k)} \quad \text{and} \quad C^{(k+1)} = \hat{C}^{(k)}_{2:m-k+1,2:n-k+1} - l^{(k)} \cdot u^{(k)} \,.$$

$C^{(k+1)}$ is a Cauchy-like matrix satisfying the displacement equation

$$\Omega_{k+1:m} \cdot C^{(k+1)} - C^{(k+1)} \cdot \Lambda_{k+1:n} = A^{(k+1)} \cdot B^{(k+1)} \,,$$

where $A^{(k+1)} = \hat{A}^{(k)}_{2:m-k+1} - l^{(k)} \cdot \hat{a}^{(k)}$ and $B^{(k+1)} = \hat{B}_{2:n-k+1} - \hat{b}^{(k)} \cdot u^{(k)}/\gamma^{(k)}$, with $\hat{a}^{(k)}$ and $\hat{b}^{(k)}$ being the first row of $\hat{A}^{(k)}$ and first column of $\hat{B}^{(k)}$, respectively. It follows from (4.43) that

$$\gamma^{(k)} = \frac{\hat{a}^{(k)} \cdot \hat{b}^{(k)}}{\omega_k - \lambda_k} \quad \text{and} \quad l^{(k)} = \frac{\left(\Omega_{k+1:m} - \lambda_k I_{m-k}\right)^{-1} \cdot \hat{A}^{(k)}_{2:m-k+1} \cdot \hat{b}^{(k)}}{\gamma^{(k)}} \,. \tag{4.45}$$

Similarly from (4.44) we have

$$z^{(k-1)} = \left(\hat{a}^{(k)} \cdot \hat{Y}^{(k-1)}\right) \cdot \left(\omega_k I_{k-1} - \Omega_{1:k-1,1:k-1}\right)^{-1} \,. \tag{4.46}$$

The matrix $Z^{(k)}$ is a Cauchy-like matrix satisfying the displacement equation

$$\Omega_{k+1:m} \cdot \hat{Z}^{(k)} - \hat{Z}^{(k)} \cdot \Omega_{1:k} = A^{(k+1)} \cdot Y^{(k)} \,, \tag{4.47}$$

where

$$Y^{(k)} = \left(\hat{Y}^{(k-1)} - y^{(k)} \cdot z^{(k-1)} \,, \quad y^{(k)}\right)$$

with $y^{(k)} = \hat{b}^{(k)}/\gamma^{(k)}$.

Lemma 4.2 *In above notation, we have*

$$Z^{(k)} = \left(Z^{(k)}_{1:m-k,1:k-1}\,,\quad Z^{(k)}_{1:m-k,k}\right) = \left(\hat{Z}^{(k-1)}_{2:m-k+1,1:k-1} - l^{(k)} \cdot z^{(k-1)}\,,\quad l^{(k)}\right)\,.$$

Proof. From equation (4.45) we have

$$\hat{a}^{(k)} \cdot y^{(k)} = \omega_k - \lambda_k \quad \text{and} \quad \hat{A}^{(k)}_{2:m-k+1} \cdot y^{(k)} = (\Omega_{k+1:m} - \lambda_k I_{m-k}) \cdot l^{(k)}\,.$$

Hence

$$\begin{aligned} A^{(k+1)} \cdot y^{(k)} &= A^{(k)}_{2:m-k+1} \cdot y^{(k)} - l^{(k)} \cdot \hat{a}^{(k)} \cdot y^{(k)} \\ &= (\Omega_{k+1:m} - \lambda_k I_{m-k}) \cdot l^{(k)} - (\omega_k - \lambda_k) \cdot l^{(k)} \\ &= (\Omega_{k+1:m} - \omega_k I_{m-k}) \cdot l^{(k)}\,. \end{aligned}$$

It follows from equation (4.47) that

$$Z^{(k)}_{1:m-k,k} = (\Omega_{k+1:m} - \omega_k I_{m-k})^{-1} \cdot A^{(k+1)} \cdot y^{(k)} = l^{(k)}\,.$$

On the other hand, from equation (4.46) we have

$$\begin{aligned} & A^{(k+1)} \cdot \left(\hat{Y}^{(k-1)} - y^{(k)} \cdot z^{(k)}\right) \\ = \;& \hat{A}^{(k)}_{2:m-k+1} \cdot \hat{Y}^{(k-1)} - l^{(k)} \cdot \hat{a}^{(k)} \cdot \hat{Y}^{(k-1)} - A^{(k+1)} \cdot y^{(k)} \cdot z^{(k)} \\ = \;& \hat{A}^{(k)}_{2:m-k+1} \cdot \hat{Y}^{(k-1)} - l^{(k)} \cdot z^{(k)} \cdot (\omega_k I_{k-1} - \Omega_{1:k-1}) \\ & - (\Omega_{k+1:m} - \omega_k I_{m-k}) \cdot l^{(k)} \cdot z^{(k)} \\ = \;& \hat{A}^{(k)}_{2:m-k+1} \cdot \hat{Y}^{(k-1)} + l^{(k)} \cdot z^{(k)} \cdot \Omega_{1:k-1} - \Omega_{k+1:m} \cdot l^{(k)} \cdot z^{(k)}\,. \end{aligned}$$

Combining this with equation (4.47) gives

$$\Omega_{k+1:m} \cdot \left(Z^{(k)}_{1:m-k,1:k-1} + l^{(k)} \cdot z^{(k)}\right) - \left(Z^{(k)}_{1:m-k,1:k-1} + l^{(k)} \cdot z^{(k)}\right) \cdot \Omega_{1:k-1} = \hat{A}^{(k)}_{2:m-k+1} \cdot \hat{Y}^{(k-1)}\,.$$

Comparing this relation with equation (4.44) yields Lemma 4.2. ∎

Let the finite precision counterparts of $\gamma^{(k)}$, $v^{(k)}$, $z^{(k)}$, and $y^{(k)}$ be $\hat{\gamma}_k$, $\hat{v}^{(k)}$, $\hat{z}^{(k)}$, and $\hat{y}^{(k)}$. We assume that $l^{(k)}$ is computed as $\hat{l}^{(k)} = \mathrm{fl}(\hat{v}^{(k)}/\hat{\gamma}_k)$.

Let $G^{(k)} = A^{(k+1)} \cdot Y^{(k)}$ and $\tilde{G}^{(k)} = \tilde{A}^{(k+1)} \cdot \tilde{Y}^{(k)}$ with

$$\tilde{A}^{(k+1)} = \hat{A}^{(k)}_{k+1:m} - \hat{l}^{(k)} \cdot \hat{a}^{(k)}_k \quad \text{and} \quad \tilde{Y}^{(k)} = \left(\hat{Y}^{(k-1)} - \hat{y}^{(k)} \cdot \hat{z}^{(k)}\,,\quad \hat{y}^{(k)}\right)\,.$$

It follows from Algorithm 1 that the generator of $\hat{Z}^{(k)}$ is $\hat{G}^{(k)} \doteq \hat{A}^{(k+1)} \cdot \hat{Y}^{(k)}$, where $\hat{A}^{(k+1)}$ is the computed Q factor in the QR factorization of $\mathrm{fl}(\tilde{A}^{(k+1)})$; and $\hat{Y}^{(k)}$ is the numerical product of the R factor with $\mathrm{fl}(\tilde{Y}^{(k)})$.

Define

$$\mu = \max_{1 \le k \le n} \|\hat{C}^{(k)}\|_\infty \quad \text{and} \quad \nu = \max_{1 \le k \le n} \|\hat{Z}^{(k)}\|_\infty\,.$$

Using arguments similar to those in the proof of Lemma 4.1, it can be shown that

$$\mu \le (\nu + 1) \cdot \|C\|_\infty + O(\epsilon) \quad \text{and} \quad \|\hat{Z}\|_\infty \le \|\hat{L}^{-1}\|_\infty \le 1 + \nu \le n \cdot \|\hat{L}^{-1}\|_\infty\,.$$

In particular, ν is large only if $\hat{L}$ is ill-conditioned.

The round-off errors of the QR factorization in redecomposing the generators and the LU factorization (4.39) for the case $m = n$ are analyzed in [23]. Different error analysis has also been done by Sweet and Brent [44] for the algorithm of [18] that LU factorizes Cauchy-like matrices, and by Chandrasekaran and Sayed [10] for a generalized Schur algorithm that LU factorizes more general structured matrices.

Theorem 4.1 below is a direct generalization of related results in § 4.2 of [23]. Throughout §4 we will use the same η in many error bounds; and every η is a small multiples of ϵ. We will also assume that both $\psi_{\max}$ in (2.9) and $\phi_{\max}$ in (2.10) are constants at most 2, as in the case where the Cauchy-like matrix is transformed from a Toeplitz or Toeplitz-plus-Hankel matrix.

Theorem 4.1 *The round-off errors of the QR factorization in generator redecomposition satisfy*

$$\|\hat{G}^{(k)} - \tilde{G}^{(k)}\|_\infty \leq r^3 \cdot m^{\frac{3}{2}} \cdot \phi_{\max} \cdot \nu \cdot \eta + O(\epsilon^2) \ ,$$

and the backward error H in the LU factorization of C in (4.39) satisfies

$$\|H\|_\infty \leq r^3 \cdot m^{\frac{3}{2}} \cdot n \cdot \psi \cdot \left(\|\hat{U}\|_\infty + \mu\right) \cdot \eta + O(\epsilon^2) \ .$$

As remarked in [23], both $O(\epsilon^2)$ terms in Theorem 4.1 can be avoided by more detailed error analysis. This is also true for all the $O(\epsilon^2)$ terms that appear in §4.

4.1.2 Error Propagation at the k^{th} Step of Elimination

Our objective in §4.1.2 is to derive an upper bound on $\|\hat{Z}^{(k)} - Z^{(k)}\|_\infty$, from which an upper bound on $\|E^{(k)}\|_\infty$ of (4.40) will be derived in §4.1.3.

Lemma 4.3 *In the notation of §4.1.1,*

$$\begin{aligned}
\frac{|\hat{\gamma}^{(k)} - \gamma^{(k)}|}{|\gamma^{(k)}|} &\leq \rho + O(\epsilon^2) \ , \ |\hat{l}^{(k)} - l^{(k)}| \leq \rho \cdot e_{m-k} + O(\epsilon^2) && (4.48) \\
|\hat{y}^{(k)} - y^{(k)}| &\leq \rho \cdot |y^{(k)}| + O(\epsilon^2) \ , \ \|y^{(k)}\|_{\max} \leq \sqrt{m} \cdot \psi_{\max} + O(\epsilon) && (4.49) \\
\|\hat{z}^{(k)} - z^{(k)}\|_\infty &\leq \eta \cdot r \cdot \phi \cdot \sqrt{m \cdot n} \cdot \nu + O(\epsilon^2) \ , && (4.50)
\end{aligned}$$

where $\rho = \eta \cdot r \cdot \sqrt{m} \cdot \psi$.

Proof. The fact that Algorithm 1 performs row pivoting gives

$$\begin{aligned}
|\gamma^{(k)}| &= \|(\Omega_{k:m} - \lambda_k I_{m-k+1})^{-1} \cdot \hat{A}^{(k)} \cdot \hat{b}^{(k)}\|_{\max} + O(\epsilon) \\
&\geq \frac{1}{\sqrt{m}} \cdot \|(\Omega_{k:m} - \lambda_k I_{m-k+1})^{-1} \cdot \hat{A}^{(k)} \cdot \hat{b}^{(k)}\|_2 + O(\epsilon) \\
&\geq \frac{\|\hat{b}^{(k)}\|_2}{\sqrt{m} \cdot \psi_{\max}} + O(\epsilon) \geq \frac{|\hat{a}^{(k)}| \cdot |\hat{b}^{(k)}|}{\sqrt{m} \cdot \psi_{\max}} + O(\epsilon) \ ,
\end{aligned}$$

where we have used the fact that $\hat{A}^{(k)}$ is numerically column unitary.

On the other hand, it follows from (1.8) and (4.43) that

$$|\hat{\gamma}^{(k)} - \gamma^{(k)}| \leq \eta \cdot r \cdot \frac{|\hat{a}^{(k)}| \cdot |\hat{b}^{(k)}|}{|\omega_k - \lambda_k|} .$$

Combining these two relations,

$$|\hat{\gamma}^{(k)} - \gamma^{(k)}| \leq \eta \cdot r \cdot \frac{\sqrt{m} \cdot \psi_{\max} \cdot |\gamma^{(k)}|}{|\omega_k - \lambda_k|} + O(\epsilon) \leq \eta \cdot r \cdot \sqrt{m} \cdot \psi \cdot |\gamma^{(k)}| + O(\epsilon) ,$$

which implies the first relation in (4.48).

To derive an upper bound on $|\hat{l}^{(k)} - l^{(k)}|$, we first note from (1.8) and (4.45) that

$$\begin{aligned} |\hat{v}^{(k)} - v^{(k)}| &\leq r\eta |\Omega_{k+1:m} - \lambda_k I_{m-k}|^{-1} \cdot |A^{(k)}_{2:m-k+1}| \cdot |b^{(k)}| \\ &\leq \frac{r\eta \cdot \|b^{(k)}\|_2}{\psi_{\min}} \cdot e_{m-k} + O(\epsilon) \leq r\sqrt{m} \cdot \eta \cdot |\gamma^{(k)}| \cdot \psi \cdot e_{m-k} + O(\epsilon) . \end{aligned}$$

Since Algorithm 1 performs row pivoting, we have $|\gamma^{(k)}| \geq \|v^{(k)}\|_{\max} + O(\epsilon)$ and

$$|\hat{l}^{(k)} - l^{(k)}| \leq \left| \hat{l}^{(k)} - \frac{\hat{v}^{(k)}}{\hat{\gamma}^{(k)}} \right| + \frac{|\hat{v}^{(k)} - v^{(k)}|}{|\hat{\gamma}^{(k)}|} + \frac{|v^{(k)}|}{|\hat{\gamma}^{(k)}|} \cdot \frac{\left|\hat{\gamma}^{(k)} - \gamma^{(k)}\right|}{|\gamma^{(k)}|} . \tag{4.51}$$

By our model of computation (see §1.5), we have

$$\left| \hat{l}^{(k)} - \frac{\hat{v}^{(k)}}{\hat{\gamma}^{(k)}} \right| \leq \eta \cdot |\hat{l}^{(k)}| \leq \eta \cdot e_{m-k} + O(\epsilon) .$$

Plugging this relation and the upper bounds on $|\hat{v}^{(k)} - v^{(k)}|$ and $\left|\hat{\gamma}^{(k)} - \gamma^{(k)}\right|$ into (4.51) and simplifying, we obtain the second relation in (4.48).

The bound on $|\hat{y}^{(k)} - y^{(k)}|$ and $\|y^{(k)}\|_{\max}$ follow from similar arguments. As to $\|\hat{z}^{(k)} - z^{(k)}\|_\infty$, we note from (1.8) and (4.46) that

$$\|\hat{z}^{(k)} - z^{(k)}\|_\infty \leq \eta \cdot r \cdot \frac{\left\| |\hat{a}^{(k)}| \cdot |Y^{(k-1)}| \right\|_\infty}{\phi_{\min}} + O(\epsilon^2) . \tag{4.52}$$

On the other hand,

$$\begin{aligned} \left\| |\hat{a}^{(k)}| \cdot |Y^{(k-1)}| \right\|_\infty &\leq \sqrt{n} \cdot \left\| |\hat{a}^{(k)}| \cdot |Y^{(k-1)}| \right\|_F \leq \sqrt{n} \cdot \left\| Y^{(k-1)} \right\|_F + O(\epsilon) \\ &= \sqrt{n} \cdot \left\| \hat{A}^{(k)} \cdot Y^{(k-1)} \right\|_F + O(\epsilon) \leq \phi_{\max} \cdot \sqrt{n} \cdot \left\| \hat{Z}^{k-1} \right\|_F + O(\epsilon) \\ &\leq \phi_{\max} \cdot \sqrt{m \cdot n} \cdot \left\| \hat{Z}^{k-1} \right\|_\infty + O(\epsilon) \leq \phi_{\max} \cdot \sqrt{m \cdot n} \cdot \nu + O(\epsilon) . \end{aligned}$$

Plugging this relation into (4.52) we obtain (4.50). ▌

Lemma 4.4 below gives an upper bound on $\|\tilde{G}^{(k+1)} - G^{(k+1)}\|_\infty$.

Lemma 4.4 *In the notation of §4.1.1,*

$$\|\tilde{G}^{(k+1)} - G^{(k+1)}\|_\infty \leq \eta \cdot r^2 \cdot m \cdot \sqrt{n} \cdot (\psi_{\max} + \phi_{\max}) \cdot (\psi + \phi) \cdot (\nu + 1) + O(\epsilon^2) .$$

Proof. First of all, we note that

$$\begin{aligned}\tilde{G}^{(k+1)} - G^{(k+1)} &= \tilde{A}^{(k+1)} \cdot \tilde{Y}^{(k)} - A^{(k+1)} \cdot Y^{(k)} \\ &= \left(\tilde{A}^{(k+1)} - A^{(k+1)}\right) \cdot \tilde{Y}^{(k)} + A^{(k+1)} \cdot \left(\tilde{Y}^{(k)} - Y^{(k)}\right) . \qquad (4.53)\end{aligned}$$

Now we derive ∞-norm upper bounds on the two terms in the right hand sum. Since

$$\tilde{A}^{(k+1)} - A^{(k+1)} = \left(l^{(k)} - \hat{l}^{(k)}\right) \cdot a^{(k)} ,$$

it follows from Lemma 4.3 and its proof that

$$\| \left(\tilde{A}^{(k+1)} - A^{(k+1)}\right) \cdot \tilde{Y}^{(k)}\|_\infty \le \rho \cdot \phi_{\max} \cdot \sqrt{m \cdot n} \cdot \nu + O(\epsilon^2) .$$

On the other hand,

$$\begin{aligned}\left\|\tilde{Y}^{(k)} - Y^{(k)}\right\|_\infty &= \left\|\left(-\hat{y}^{(k)} \cdot \hat{z}^{(k)} + y^{(k)} \cdot z^{(k)} , \quad \hat{y}^{(k)} - y^{(k)}\right)\right\|_\infty \\ &= \left\|\left(-\left(\hat{y}^{(k)} - y^{(k)}\right) \cdot \hat{z}^{(k)} - y^{(k)} \cdot \left(\hat{z}^{(k)} - z^{(k)}\right) , \quad \hat{y}^{(k)} - y^{(k)}\right)\right\|_\infty \\ &\le \left\|\left(\hat{y}^{(k)} - y^{(k)}\right) \cdot \hat{z}^{(k)}\right\|_\infty + \left\|y^{(k)} \cdot \left(\hat{z}^{(k)} - z^{(k)}\right)\right\|_\infty \\ &\quad + \left\|\hat{y}^{(k)} - y^{(k)}\right\|_\infty .\end{aligned}$$

Applying Lemma 4.3 and simplifying,

$$\left\|\tilde{Y}^{(k)} - Y^{(k)}\right\|_\infty \le \eta \cdot r \cdot m \cdot \psi_{\max} \cdot (\psi + \sqrt{n} \cdot \phi) \cdot (\nu + 1) + O(\epsilon^2) .$$

Taking norms on both sides of (4.53) and plugging in these relations, we have

$$\begin{aligned}\left\|\tilde{G}^{(k+1)} - G^{(k+1)}\right\|_\infty &\le \left\|\left(\tilde{A}^{(k+1)} - A^{(k+1)}\right) \cdot \tilde{Y}^{(k)}\right\|_\infty + \left\|A^{(k+1)} \cdot \left(\tilde{Y}^{(k)} - Y^{(k)}\right)\right\|_\infty \\ &\le \rho \cdot \phi_{\max} \cdot \sqrt{m \cdot n} \cdot \nu + r \cdot \left\|\tilde{Y}^{(k)} - Y^{(k)}\right\|_\infty + O(\epsilon^2) \\ &\le \rho \cdot \phi_{\max} \cdot \sqrt{m \cdot n} \cdot \nu \\ &\quad + \eta \cdot r^2 \cdot m \cdot \psi_{\max} \cdot (\psi + \sqrt{n} \cdot \phi) \cdot (\nu + 1) + O(\epsilon^2) .\end{aligned}$$

Lemma 4.4 follows immediately by simplifying the last expression. ▌

4.1.3 Backward Errors in $\hat{Z}$

Our objective in §4.1.3 is to derive an ∞-norm backward error bound on $E^{(k)}$.

Theorem 4.2 *Let $E^{(k)}$ be as defined in (4.40). Then for all $1 \le k \le n$,*

$$\left\|E^{(k)}\right\|_\infty \le \eta \cdot r^3 \cdot m \cdot n \cdot \phi \cdot (\sqrt{n} \cdot (\psi + \phi) + \sqrt{m}) \cdot (\nu + 1) + O(\epsilon^2) .$$

Proof. By definition, $E^{(k)}$ is a lower triangular matrix and $E^{(k)}_{1:m,k+1:n} = 0$. In addition, $\hat{L}^{(k)}$ can be constructed from $\hat{L}^{(k-1)}$ by replacing the zero vector $\hat{L}^{(k-1)}_{k+1:m,k}$ by $\hat{l}^{(k)}$; and $\hat{X}^{(k)}$ can be constructed from $\hat{X}^{(k-1)}$ by replacing the matrix

$$\hat{X}^{(k-1)}_{k+1:m,1:k} = \left(-\hat{Z}^{(k-1)}_{2:m-k+1,1:k-1} , 0\right)$$

by $-\hat{Z}^{(k)}$. It follows that $E^{(k)}_{1:k,1:k} = E^{(k-1)}_{1:k,1:k}$. Consequently,

$$\begin{aligned}
& E^{(k)}_{k+1:m,1:k} = \hat{L}^{(k)}_{k+1:m,1:k} \cdot \hat{X}^{(k)}_{1:k,1:k} - \hat{Z}^{(k)} \\
= & \left(\hat{L}^{(k)}_{k+1:m,1:k-1} \cdot \hat{X}^{(k)}_{1:k-1,1:k-1} + \hat{L}^{(k)}_{k+1:m,k} \cdot \hat{X}^{(k)}_{k,1:k-1} - \hat{Z}^{(k)}_{1:m-k,1:k-1}\, , \quad \hat{L}^{(k)}_{k+1:m,k} - \hat{Z}^{(k)}_{1:m-k,k}\right) \\
= & \left(\hat{Z}^{(k-1)}_{2:m-k+1,1:k-1} + E^{(k-1)}_{k+1:m,1:k-1} - \hat{l}^{(k)} \cdot \hat{z}^{(k-1)} - \hat{Z}^{(k)}_{1:m-k,1:k-1}\, , \quad \hat{l}^{(k)} - \hat{Z}^{(k)}_{1:m-k,k}\right)\, , \qquad (4.54)
\end{aligned}$$

where we have used the fact that

$$\hat{L}^{(k-1)}_{k+1:m,k} = \hat{l}^{(k)} \quad , \quad \hat{X}^{(k)}_{k,1:k-1} = -\hat{z}^{(k-1)}$$

and that

$$\hat{L}^{(k)}_{k+1:m,1:k-1} \cdot \hat{X}^{(k)}_{1:k-1,1:k-1} = \hat{Z}^{(k-1)}_{2:m-k+1,1:k-1} + E^{(k-1)}_{k+1:m,1:k-1} \hat{Z}^{(k)}\, .$$

Applying Lemma 4.2 to (4.54), we have

$$\begin{aligned}
E^{(k)}_{k+1:m,1:k} \quad = & \quad \left(E^{(k-1)}_{k+1:m,1:k-1}\, , \quad 0\right) + \left(Z^{(k)} - \hat{Z}^{(k)}\right) \\
& + \left(-\hat{l}^{(k)} \cdot \hat{z}^{(k-1)} + l^{(k)} \cdot z^{(k-1)}\, , \quad \hat{l}^{(k)} - l^{(k)}\right)\, .
\end{aligned}$$

Taking the ∞-norm on both sides, and recalling the fact that entries of $E^{(k)}$ and $E^{(k-1)}$ are identical elsewhere, we get

$$\begin{aligned}
\left\|E^{(k)}\right\|_\infty \quad \leq & \quad \left\|E^{(k-1)}\right\|_\infty + \left\|Z^{(k)} - \hat{Z}^{(k)}\right\|_\infty \\
& + \left\|\left(\hat{l}^{(k)} \cdot \hat{z}^{(k-1)} - l^{(k)} \cdot z^{(k-1)}\, , \quad \hat{l}^{(k)} - l^{(k)}\right)\right\|_\infty\, . \qquad (4.55)
\end{aligned}$$

In this relation,

$$\begin{aligned}
\left\|Z^{(k)} - \hat{Z}^{(k)}\right\|_\infty \quad \leq & \quad \frac{\left\|G^{(k)} - \hat{G}^{(k)}\right\|_\infty}{\phi_{\min}} \\
\leq & \quad \frac{\left\|G^{(k)} - \tilde{G}^{(k)}\right\|_\infty}{\phi_{\min}} + \frac{\left\|\tilde{G}^{(k)} - \hat{G}^{(k)}\right\|_\infty}{\phi_{\min}}\, .
\end{aligned}$$

Combining the results in Theorem 4.1 and Lemma 4.4 and simplifying, we have

$$\left\|Z^{(k)} - \hat{Z}^{(k)}\right\|_\infty \leq \eta \cdot r^3 \cdot m \cdot \phi \cdot \left(\sqrt{n} \cdot (\psi + \phi) + \sqrt{m}\right) \cdot (\nu + 1) + O(\epsilon^2)\, .$$

On the other hand,

$$\begin{aligned}
& \left\|\left(-\hat{l}^{(k)} \cdot \hat{z}^{(k)} + l^{(k)} \cdot z^{(k)}\, , \quad \hat{l}^{(k)} - l^{(k)}\right)\right\|_\infty \\
= & \left\|\left(-\left(\hat{l}^{(k)} - l^{(k)}\right) \cdot \hat{z}^{(k)} - l^{(k)} \cdot \left(\hat{z}^{(k)} - z^{(k)}\right)\, , \quad \hat{l}^{(k)} - l^{(k)}\right)\right\|_\infty \\
\leq & \left\|\left(\hat{l}^{(k)} - l^{(k)}\right) \cdot \hat{z}^{(k)}\right\|_\infty + \left\|l^{(k)} \cdot \left(\hat{z}^{(k)} - z^{(k)}\right)\right\|_\infty \\
& + \left\|\hat{l}^{(k)} - l^{(k)}\right\|_\infty\, .
\end{aligned}$$

Applying Lemma (4.3) to above and simplifying,

$$\left\|\left(-\hat{l}^{(k)} \cdot \hat{z}^{(k)} + l^{(k)} \cdot z^{(k)}\, , \quad \hat{l}^{(k)} - l^{(k)}\right)\right\|_\infty \leq \eta \cdot r \cdot \sqrt{m} \cdot \left(\psi + \sqrt{n} \cdot \phi\right) \cdot (\nu + 1) + O(\epsilon^2)\, .$$

Plugging these relations into (4.55) and simplifying,

$$\left\|E^{(k)}\right\|_\infty \le \left\|E^{(k-1)}\right\|_\infty + \eta \cdot r^3 \cdot m \cdot \phi \cdot (\sqrt{n} \cdot (\psi + \phi) + \sqrt{m}) \cdot (\nu + 1) + O(\epsilon^2) .$$

Theorem 4.2 is proved by solving this recursion with $E^{(0)} = 0$. ▮

Now we analyze the backward error in $\hat{Z}$. In equation (4.40), we set $k = n$ to get

$$\hat{L}_1 \cdot \hat{X}^{(n)}_{1:n,1:n} = I_n + E^{(n)}_{1:n,1:n} \quad \text{and} \quad \hat{L}_2 \cdot \hat{X}^{(n)}_{1:n,1:n} - \hat{Z} = E^{(n)}_{n+1:m,1:n} .$$

It follows that

$$\begin{aligned} \hat{Z} \cdot \hat{L}_1 - \hat{L}_2 &= \hat{L}_2 \cdot \hat{X}^{(n)}_{1:n,1:n} \cdot \hat{L}_1 - \hat{L}_2 - E^{(n)}_{n+1:m,1:n} \cdot \hat{L}_1 \\ &= \hat{L}_2 \cdot \hat{L}_1^{-1} \cdot \left(\hat{L}_1 \cdot X^{(n)}_{1:n,1:n}\right) \cdot \hat{L}_1 - \hat{L}_2 - E^{(n)}_{n+1:m,1:n} \cdot \hat{L}_1 \\ &= \hat{L}_2 \cdot \hat{L}_1^{-1} \cdot \left(I_n + E^{(n)}_{1:n,1:n}\right) \cdot \hat{L}_1 - \hat{L}_2 - E^{(n)}_{n+1:m,1:n} \cdot \hat{L}_1 \\ &= \hat{L}_2 \cdot \hat{L}_1^{-1} \cdot E^{(n)}_{1:n,1:n} \cdot \hat{L}_1 - E^{(n)}_{n+1:m,1:n} \cdot \hat{L}_1 . \end{aligned}$$

On the other hand,

$$\begin{aligned} \hat{L}_2^{(n)} \cdot \hat{L}_1^{-1} &= \hat{L}_2 \cdot \hat{X}^{(n)}_{1:n,1:n} \cdot \left(\hat{L}_1 \cdot \hat{X}^{(n)}_{1:n,1:n}\right)^{-1} \\ &= \left(\hat{Z} + E^{(n)}_{n+1:m,1:n}\right) \cdot \left(I_n + E^{(n)}_{1:n,1:n}\right)^{-1} = \hat{Z} + O(\epsilon) . \end{aligned}$$

Hence

$$\hat{Z} \cdot \hat{L}_1 - \hat{L}_2 = \hat{Z} \cdot E^{(n)}_{1:n,1:n} \cdot \hat{L}_1 - E^{(n)}_{n+1:m,1:n} \cdot \hat{L}_1 + O(\epsilon^2) . \tag{4.56}$$

Theorem 4.3 *Let F be defined in (4.41). Then*

$$\|F\|_\infty \le \eta \cdot r^3 \cdot m \cdot n \cdot \phi \cdot (\sqrt{n} \cdot (\psi + \phi) + \sqrt{m}) \cdot (\nu + 1) \cdot (\|\hat{Z}\|_\infty + 1) \cdot \|C\|_\infty + O(\epsilon^2) .$$

Proof. From equation (4.39) we have $C_2 = \hat{L}_2 \cdot \hat{U} - H_{n+1:m,1:n}$. Hence

$$\begin{aligned} F &= \hat{Z} \cdot \hat{L}_1 \cdot \hat{U} - C_2 \\ &= \hat{Z} \cdot \hat{L}_1 \cdot \hat{U} - \hat{L}_2 \cdot \hat{U} + H_{n+1:m,1:n} \\ &= \left(\hat{Z} \cdot \hat{L}_1 - \hat{L}_2\right) \cdot \hat{U} + H_{n+1:m,1:n} \\ &= \hat{Z} \cdot E^{(n)}_{1:n,1:n} \cdot \hat{L}_1 \cdot \hat{U} - E^{(n)}_{n+1:m,1:n} \cdot \hat{L}_1 \cdot \hat{U} + H_{n+1:m,1:n} + O(\epsilon^2) , \end{aligned}$$

where we have used relation (4.56). Taking the ∞-norm,

$$\begin{aligned} \|F\|_\infty &\le \|\hat{Z}\|_\infty \cdot \|E^{(n)}\|_\infty \cdot \|\hat{L}_1 \cdot \hat{U}\|_\infty + \|E^{(n)}\|_\infty \cdot \|\hat{L}_1 \cdot \hat{U}\|_\infty + \|H\|_\infty + O(\epsilon^2) \\ &\le \left(\|\hat{Z}\|_\infty + 1\right) \cdot \|E^{(n)}\|_\infty \cdot \|C\|_\infty + \|H\|_\infty + O(\epsilon^2) . \end{aligned}$$

Theorem 4.3 is proved by plugging in related upper bounds in Theorems 4.1 and 4.2, and observing that

$$\max\{\mu, \|U\|_\infty\} \le (1 + \nu) \cdot \|C\|_\infty + O(\epsilon) .$$

▮

4.2 Backward Errors in Cholesky Factorization

We assume without loss of generality that no column or row swaps actually occur during the execution of Algorithms 2 and 3. Define

$$\hat{\mathcal{L}} \cdot \hat{\mathcal{D}} \cdot \hat{\mathcal{L}}^* = K + \mathcal{H} , \tag{4.57}$$

where K is the symmetric/Hermitian positive definite Cauchy-like matrix to be factored; $\hat{\mathcal{L}} \cdot \hat{\mathcal{D}} \cdot \hat{\mathcal{L}}^*$ is the computed Cholesky factorization with $\hat{\mathcal{L}}$ being unit lower triangular; and $\mathcal{H}$ is the backward error matrix. For Algorithm 2, K, $\mathcal{H}$, $\hat{\mathcal{L}} \in \mathbf{R}^{n \times n}$, and for Algorithm 3, K, $\mathcal{H}$, $\hat{\mathcal{L}} \in \mathbf{C}^{n \times n}$. Our objective in § 4.2 is to derive an upper bound on $\|\mathcal{H}\|_2$.

4.2.1 Preliminary Results

Assume that after the first $k-1$ steps of Cholesky Factorization in finite arithmetic, we arrive at the symmetric/Hermitian positive definite Cauchy-like matrix

$$\hat{K}^{(k)} = \begin{pmatrix} \gamma^{(k)} & (v^{(k)})^* \\ v^{(k)} & \hat{K}^{(k)}_{2:m-k+1,2:m-k+1} \end{pmatrix} . \tag{4.58}$$

For Algorithm 2, $\hat{K}^{(k)}$ is real and satisfies the displacement equation

$$\Omega_{k:m} \cdot \hat{K}^{(k)} - \hat{K}^{(k)} \cdot \Omega_{k:m} = \hat{\mathcal{A}}^{(k)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\hat{\mathcal{A}}^{(k)}\right)^T ; \tag{4.59}$$

and for Algorithm 3, $\hat{K}^{(k)}$ is complex and satisfies the displacement equation

$$\hat{K}^{(k)} - \Omega^*_{k:m} \cdot \hat{K}^{(k)} \cdot \Omega_{k:m} = \hat{\mathcal{A}}^{(k)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\hat{\mathcal{A}}^{(k)}\right)^* \tag{4.60}$$

with diagonal entries of Ω being on the unit disk in the complex plane. In both cases, the diagonal entries of $\hat{K}^{(k)}$ are positive and are provided separately from the displacement equation.

The k^{th} step of Cholesky factorization eliminates the first column and row of $\hat{K}^{(k)}$ below and above the diagonal:

$$\hat{K}^{(k)} = \begin{pmatrix} 1 & 0 \\ l^{(k)} & I_{n-k} \end{pmatrix} \cdot \begin{pmatrix} \gamma^{(k)} & 0 \\ 0 & K^{(k+1)} \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ l^{(k)} & I_{n-k} \end{pmatrix}^* ,$$

where

$$l^{(k)} = v^{(k)}/\gamma^{(k)} \quad \text{and} \quad K^{(k+1)} = \hat{K}^{(k)}_{2:m-k+1,2:n-k+1} - v^{(k)} \cdot (v^{(k)})^*/\gamma^{(k)} .$$

Let $\hat{a}^{(k)}$ be the first row of $\hat{\mathcal{A}}^{(k)}$. For Algorithm 2, $K^{(k+1)}$ is a symmetric positive definite Cauchy-like matrix satisfying the displacement equation

$$\Omega_{k+1:m} \cdot K^{(k+1)} - K^{(k+1)} \cdot \Omega_{k+1:m} = \mathcal{A}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\mathcal{A}^{(k+1)}\right)^T ,$$

where $\mathcal{A}^{(k+1)} = \hat{\mathcal{A}}^{(k)}_{2:m-k+1} - l^{(k)} \cdot \hat{a}^{(k)}$. It follows from (4.43) that

$$l^{(k)} = \frac{\left(\Omega_{k+1:m} - \omega_k I_{m-k}\right)^{-1} \cdot \hat{\mathcal{A}}^{(k)}_{2:m-k+1} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\hat{a}^{(k)}\right)^T}{\gamma^{(k)}} . \tag{4.61}$$

For Algorithm 3, $K^{(k+1)}$ is a Hermitian positive definite Cauchy-like matrix satisfying the displacement equation

$$K^{(k+1)} - \Omega^*_{k+1:m} \cdot K^{(k+1)} \cdot \Omega_{k+1:m} = \mathcal{A}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\mathcal{A}^{(k+1)}\right)^* ,$$

where $\mathcal{A}^{(k+1)} = \hat{\mathcal{A}}^{(k)}_{2:m-k+1} - \left(l^{(k)} - t^{(k)}/2\right) \cdot \hat{a}^{(k)}$, with $t^{(k)} = \hat{\mathcal{A}}^{(k)}_{2:m-k+1} \cdot \hat{\mathcal{J}} \cdot \left(\hat{a}^{(k)}\right)^* / \gamma^{(k)}$ and $l^{(k)} = \left(I_{n-k} - \omega_k \cdot \bar{\Omega}_{k+1:n}\right)^{-1} \cdot t^{(k)}$. For both Algorithms 2 and 3, the diagonal entries of $K^{(k+1)}$ satisfy

$$K^{(k+1)}_{j,j} = K^{(k)}_{j+1,j+1} - |v^{(k)}_j|^2/\gamma^{(k)} , \quad \text{for} \quad j = 1, \cdots, n-k .$$

Let the finite precision counterparts of $v^{(k)}$, $t^{(k)}$, and $l^{(k)}$ be $\hat{v}^{(k)}$, $\hat{t}^{(k)}$, and $\hat{l}^{(k)}$; and let the finite precision counterpart of $K^{(k+1)}_{j,j}$ be $\hat{K}^{(k+1)}_{j,j}$ for $j = 1, \cdots, m-k$. We also define

$$\Delta^{(k+1)} = \mathcal{A}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\mathcal{A}^{(k+1)}\right)^* \quad \text{and} \quad \tilde{\Delta}^{(k+1)} = \tilde{\mathcal{A}}^{(k+1)} \cdot \mathcal{J}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)}\right)^*$$

with

$$\tilde{\mathcal{A}}^{(k+1)} = \hat{\mathcal{A}}^{(k)}_{2:m-k+1} - \left(\hat{l}^{(k)} - \hat{t}^{(k)}/2\right) \cdot \hat{a}^{(k)} .$$

It follows from Algorithms 2 and 3 that the generator of $\hat{K}^{(k+1)}$ is

$$\hat{\Delta}^{(k+1)} = \hat{\mathcal{A}}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k+1)} \cdot \left(\hat{\mathcal{A}}^{(k+1)}\right)^* ,$$

where $\hat{\mathcal{A}}^{(k+1)}$ is the computed Q factor in the QR factorization of $\text{fl}(\tilde{\mathcal{A}}^{(k+1)})$; and

$$\hat{\mathcal{J}}^{(k+1)} = \text{fl}\left(\text{fl}\left(\mathcal{R} \cdot \hat{\mathcal{J}}^{(k)}\right) \cdot \mathcal{R}^*\right)$$

with $\mathcal{R}$ being the computed R factor in the QR factorization.

4.2.2 Error Propagation at the k^{th} Step of Cholesky Factorization

Our objective in §4.2.2 is to derive an upper bound on $\|K^{(k+1)} - \hat{K}^{(k+1)}\|_2$.

Lemma 4.5 *In the notation of §4.2,*

$$\begin{aligned}
\|\hat{\mathcal{J}}^{(k)}\|_F &\le \phi_{\max} \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon) , \ \|\hat{l}^{(k)} - l^{(k)}\|_2 \le \eta \cdot r^{1.5} \cdot n \cdot \phi + O(\epsilon^2) , \\
\|\hat{t}^{(k)} - t^{(k)}\|_2 &\le \eta \cdot r^{1.5} \cdot n \cdot \phi + O(\epsilon^2) , \ \|\hat{t}\|_2 \le 3 \cdot \sqrt{n} + O(\epsilon) , \\
|\hat{K}^{(k+1)}_{j,j} - K^{(k+1)}_{j,j}| &\le \eta \cdot \|K^{(k)}\|_{\max} + O(\epsilon) \quad \textit{for} \quad j = 1, 2, \cdots, n-k .
\end{aligned}$$

Proof. We start by proving the F-norm upper bounds on $\mathcal{J}^{(k)}$. Since $\hat{\mathcal{A}}^{(k)}$ is numerically column unitary, by relation (1.7) we have

$$\|\hat{\mathcal{J}}^{(k)}\|_F = \|\hat{\mathcal{A}}^{(k)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\hat{\mathcal{A}}^{(k)}\right)^* \|_F + O(\epsilon) \leq \phi_{\max} \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon) ,$$

where we have used the fact that the diagonal entries in the matrix $\hat{\mathcal{A}}^{(k)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\hat{\mathcal{A}}^{(k)}\right)^*$ are $O(\epsilon)$. Since both Algorithms 2 and 3 perform diagonal pivoting on a symmetric/Hermitian positive definite matrix, it follows that $\gamma^{(k)} = \|\hat{K}^{(k)}\|_{\max} + O(\epsilon)$ and that (see (1.7))

$$\|\hat{\mathcal{J}}^{(k)}\|_F \leq n \cdot \gamma^{(k)} + O(\epsilon) .$$

Let $y = \hat{\mathcal{A}}^{(k)}_{2:m-k+1} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\hat{a}^{(k)}\right)^*$ and $\hat{y}$ be its finite precision counterpart. Then we get from (1.8) that

$$\begin{aligned}
\|\hat{y} - y\|_2 &\leq r \cdot \eta \cdot \left\| |\hat{\mathcal{A}}^{(k)}_{2:m-k+1}| \cdot |\hat{\mathcal{J}}^{(k)}| \cdot | \left(a^{(k)}\right)^T | \right\|_2 \\
&\leq r \cdot \eta \cdot \left\| |\hat{\mathcal{A}}^{(k)}_{2:m-k+1}| \right\|_2 \cdot \left\| |\hat{\mathcal{J}}^{(k)}| \right\|_2 \cdot \left\| | \left(a^{(k)}\right)^T | \right\|_2 \\
&\leq r^{1.5} \cdot \eta \cdot \|\hat{\mathcal{J}}^{(k)}\|_F + O(\epsilon^2) \\
&\leq r^{1.5} \cdot \eta \cdot \phi_{\max} \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) ,
\end{aligned}$$

where we have used relation (1.7) and the fact that $\hat{\mathcal{A}}^{(k)}$ is numerically column unitary. Similarly,

$$\|\hat{y}\|_2 \leq \sqrt{r} \cdot \phi_{\max} \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon) .$$

We now derive 2-norm upper bound on $\hat{l}^{(k)} - l^{(k)}$ for Algorithm 2. From (1.8) we have

$$\|\hat{l}^{(k)} - l^{(k)}\|_2 \leq \left\| \hat{l}^{(k)} - \frac{\hat{v}^{(k)}}{\gamma^{(k)}} \right\|_2 + \frac{\left\| \hat{v}^{(k)} - v^{(k)} \right\|_2}{\gamma^{(k)}} \leq \eta \cdot \frac{\|\hat{v}^{(k)}\|_2}{\gamma^{(k)}} + \frac{\left\| \hat{v}^{(k)} - v^{(k)} \right\|_2}{\gamma^{(k)}} ,$$

where $\|\hat{v}^{(k)}\|_{\max} \leq \gamma^{(k)} + O(\epsilon)$; and

$$\begin{aligned}
\|\hat{v}^{(k)} - v^{(k)}\|_2 &\leq \left\| \hat{v}^{(k)} - \left(\Omega_{k+1:m} - \omega_k I_{m-k}\right)^{-1} \cdot \hat{y} \right\|_2 + \left\| \left(\Omega_{k+1:m} - \omega_k I_{m-k}\right)^{-1} \cdot (\hat{y} - y) \right\|_2 \\
&\leq \frac{\eta \cdot \|\hat{y}\|_2}{\phi_{\min}} + \frac{\|\hat{y} - y\|_2}{\phi_{\min}} .
\end{aligned}$$

Plugging the corresponding upper bounds on $\|\hat{y} - y\|_2$, $\|\hat{y}\|_2$, we obtain the desired 2-norm upper bound.

To derive 2-norm upper bound on $\hat{t}^{(k)} - t^{(k)}$ for Algorithm 3, we note that from (1.8)

$$\|\hat{t}^{(k)} - t^{(k)}\|_2 \leq \left\| \hat{t}^{(k)} - \frac{\hat{y}}{\gamma^{(k)}} \right\|_2 + \frac{\|\hat{y} - y\|_2}{\gamma^{(k)}} \leq \eta \cdot \frac{\|\hat{y}\|_2}{\gamma^{(k)}} + \frac{\|\hat{y} - y\|_2}{\gamma^{(k)}} .$$

Plugging the corresponding upper bounds on $\|\hat{y} - y\|_2$, $\|\hat{y}\|_2$, and $\|K^{(k)}\|_F$, we obtain the desired upper bound. Since $l^{(k)} = \left(I_{n-k} - \omega_k \cdot \bar{\Omega}_{k+1:n}\right)^{-1} \cdot t^{(k)}$, it follows that $t^{(k)} = \left(I_{n-k} - \omega_k \cdot \bar{\Omega}_{k+1:n}\right) \cdot l^{(k)}$ and hence

$$\|\hat{t}\|_2 \leq 3 \cdot \sqrt{n} .$$

As to the upper bound on $|\hat{K}^{(k+1)}_{j,j} - K^{(k+1)}_{j,j}|$, we write (see the model of arithmetic in §1.5)

$$\hat{K}^{(k+1)}_{j,j} = K^{(k)}_{j+1,j+1} \cdot (1+\eta_1) - |v^{(k)}_j|^2/\gamma^{(k)} \cdot (1+\eta_2)$$

for $|\eta_1| \leq \eta$ and $|\eta_2| \leq \eta$. It follows that

$$|\hat{K}^{(k+1)}_{j,j} - K^{(k+1)}_{j,j}| \leq \eta \cdot \left(K^{(k)}_{j+1,j+1} + |v^{(k)}_j|^2/\gamma^{(k)} \right) \leq 2 \cdot \eta \cdot \|K^{(k)}\|_{\max} + O(\epsilon) \, .$$

▮

Lemma 4.6 below gives an upper bound on $\|\tilde{\Delta}^{(k+1)} - \Delta^{(k+1)}\|_2$.

Lemma 4.6 *In the notation of §4.2,*

$$\|\tilde{\Delta}^{(k+1)} - \Delta^{(k+1)}\|_2 \leq \eta \cdot (r \cdot n)^{1.5} \cdot \phi \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) \, .$$

Proof. Similar to the proof of Lemma 4.4, we have

$$\begin{aligned} \tilde{\Delta}^{(k+1)} - \Delta^{(k+1)} &= \tilde{\mathcal{A}}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)}\right)^* - \mathcal{A}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\mathcal{A}^{(k+1)}\right)^* \\ &= \left(\tilde{\mathcal{A}}^{(k+1)} - \mathcal{A}^{(k+1)}\right) \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)}\right)^* \\ &\quad + \mathcal{A}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)} - \mathcal{A}^{(k+1)}\right)^* \, . \end{aligned} \tag{4.62}$$

Now we derive 2-norm upper bounds on the two terms in the right hand sum for Algorithm 3. These bounds for Algorithm 2 can be derived similarly.

Since

$$\tilde{\mathcal{A}}^{(k+1)} - \mathcal{A}^{(k+1)} = \left(l^{(k)} - \hat{l}^{(k)}\right) \cdot a^{(k)} - \left(t^{(k)} - \hat{t}^{(k)}\right) \cdot a^{(k)}/2 \, ,$$

we have

$$\begin{aligned} &\left\| \left(\tilde{\mathcal{A}}^{(k+1)} - \mathcal{A}^{(k+1)}\right) \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\mathcal{A}^{(k+1)}\right)^* \right\|_2 \\ \leq\ & \left(\|\hat{l}^{(k)} - l^{(k)}\|_2 \cdot \|a^{(k)}\|_2 + \|\hat{t}^{(k)} - t^{(k)}\|_2 \cdot \|a^{(k)}\|_2/2 \right) \cdot \left\|\mathcal{J}^{(k)}\right\|_2 \cdot \left\| \left(\mathcal{A}^{(k+1)}\right)^* \right\|_2 \\ \leq\ & 2\eta \cdot r^{1.5} \cdot n \cdot \phi \cdot \left\|\mathcal{A}^{(k+1)}\right\|_2 + O(\epsilon^2) \end{aligned}$$

Since $\hat{\mathcal{A}}^{(k)}$ is column unitary, it follows from Lemma 4.5 that

$$\begin{aligned} \left\|\mathcal{A}^{(k+1)}\right\|_2 &\leq \left\|\hat{\mathcal{A}}^{(k)}_{2:n-k+1}\right\|_2 + \|\hat{l}^{(k)}\|_2 \cdot \|a^{(k)}\|_2 + \|t^{(k)}\|_2 \cdot \|a^{(k)}\|_2/2 \\ &\leq 3 \cdot \sqrt{n} \, . \end{aligned}$$

Combining all these bounds, and apply Lemma 4.5 and simplifying, we obtain

$$\left\| \left(\tilde{\mathcal{A}}^{(k+1)}\right)^* - \mathcal{A}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\mathcal{A}^{(k+1)}\right)^* \right\|_2 \leq 3\eta \cdot (r \cdot n)^{1.5} \cdot \phi \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) \, .$$

Similar arguments give

$$\left\| \mathcal{A}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)} - \mathcal{A}^{(k+1)}\right)^* \right\|_2 \leq 3\eta \cdot (r \cdot n)^{1.5} \cdot \phi \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) \, .$$

Taking 2-norms on both sides of (4.62) and plugging in the above upper bounds, we arrive at the lemma. ▮

The round-off errors of the QR factorization in redecomposing the generators are analyzed in [23] and are generalized in Theorem 4.1. Lemma 4.7 performs a similar analysis.

Lemma 4.7 *The round-off errors of the QR factorization in redecomposing the generators satisfy*

$$\|\hat{\Delta}^{(k)} - \tilde{\Delta}^{(k)}\|_2 \leq \eta \cdot r \cdot n^2 \cdot \phi_{\max} \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) \, .$$

Proof. Define

$$\hat{\mathcal{A}}^{(k+1)} \cdot \mathcal{R} = \tilde{\mathcal{A}}^{(k+1)} + E_1 \quad \text{and} \quad \hat{\mathcal{J}}^{(k+1)} = \mathcal{R} \cdot \hat{\mathcal{J}}^{(k)} \cdot \mathcal{R}^* + E_2 \, ,$$

where E_1 and E_2 are round-off error matrices.

Since the numerical QR factorization is performed on $\mathrm{fl}(\tilde{\mathcal{A}}^{(k+1)})$, instead of $\tilde{\mathcal{A}}^{(k+1)}$ itself, we write

$$\|E_1\|_2 \leq \left\|\hat{\mathcal{A}}^{(k+1)} \cdot \mathcal{R} - \mathrm{fl}(\tilde{\mathcal{A}}^{(k+1)})\right\|_2 + \left\|\mathrm{fl}(\tilde{\mathcal{A}}^{(k+1)}) - \tilde{\mathcal{A}}^{(k+1)}\right\|_2 \, . \qquad (4.63)$$

In the following, we derive 2- norm upper bound on E_1. Applying the standard round-off error bounds for QR factorization (see [21]), we have

$$\begin{aligned}
\left\|\hat{\mathcal{A}}^{(k+1)} \cdot \mathcal{R} - \mathrm{fl}(\tilde{\mathcal{A}}^{(k+1)})\right\|_2 &\leq \eta \cdot r \cdot n \left\|\mathrm{fl}(\tilde{\mathcal{A}}^{(k+1)})\right\|_2 \\
&\leq \eta \cdot r \cdot n \left(\|\hat{\mathcal{A}}^{(k)}_{2:m-k+1}\|_2 + \left(\|\hat{l}^{(k)}\|_2 + \|\hat{t}^{(k)}\|_2/2\right) \cdot \|\hat{a}^{(k)}\|_2\right) \\
&\quad + O(\epsilon^2) \\
&\leq \eta \cdot r \cdot n \, (1 + (\sqrt{n} + 2 \cdot \sqrt{n}/2)) + O(\epsilon^2) \\
&\leq 3\eta \cdot r \cdot n^{1.5} + O(\epsilon^2) \, ,
\end{aligned}$$

where we have used Lemma 4.5 and the fact that $\hat{\mathcal{A}}^{(k)}$ is numerically column unitary.

On the other hand,

$$\begin{aligned}
\left\|\mathrm{fl}(\tilde{\mathcal{A}}^{(k+1)}) - \tilde{\mathcal{A}}^{(k+1)}\right\|_2 &\leq \eta \cdot \left(\|\hat{\mathcal{A}}^{(k)}_{2:m-k+1}\|_2 + \left(\|\hat{l}^{(k)}\|_2 + \|\hat{t}^{(k)}\|_2/2\right) \cdot \|\hat{a}^{(k)}\|_2\right) + O(\epsilon^2) \\
&\leq \eta \cdot (1 + (\sqrt{n} + 2 \cdot \sqrt{n}/2) \cdot 1) + O(\epsilon^2) \leq 3\eta \cdot \sqrt{n} + O(\epsilon^2) \, .
\end{aligned}$$

Similarly $\left\|\tilde{\mathcal{A}}^{(k+1)}\right\|_2 \leq 3\sqrt{n}$. Plugging these upper bounds in (4.63) and simplifying, we obtain

$$\|E_1\|_2 \leq \eta \cdot r \cdot n^{1.5} + O(\epsilon^2) \, .$$

Now we derive a 2-norm upper bound on E_2. It follows from (1.8) that

$$\|E_2\|_2 \leq \eta \cdot r \cdot |||\mathcal{R}|||_2 \cdot |||\hat{\mathcal{J}}^{(k)}|||_2 \cdot |||\mathcal{R}^*|||_2 \leq \eta \cdot r \cdot \|\mathcal{R}\|_F^2 \cdot \|\hat{\mathcal{J}}^{(k)}\|_F \, .$$

Since

$$\|\mathcal{R}\|_F = \left\|\mathrm{fl}(\tilde{\mathcal{A}}^{(k+1)})\right\|_F + O(\epsilon) \leq 3 \cdot \sqrt{n} \, ,$$

it follows that

$$\begin{aligned}
\|E_2\|_2 &\leq 9\eta \cdot r \cdot n \cdot \|\hat{\mathcal{J}}^{(k)}\|_F + O(\epsilon^2) \\
&\leq 9\eta \cdot r \cdot n \cdot \|\hat{\mathcal{A}}^{(k)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\hat{\mathcal{A}}^{(k)}\right)^* \|_F + O(\epsilon^2) \\
&\leq 9\eta \cdot \phi_{\max} \cdot r \cdot n \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) \, .
\end{aligned}$$

With all these relations, we now have

$$\begin{aligned}
\hat{\Delta}^{(k)} - \tilde{\Delta}^{(k)} &= \hat{\mathcal{A}}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k+1)} \cdot \left(\hat{\mathcal{A}}^{(k+1)}\right)^* - \tilde{\mathcal{A}}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)}\right)^* \\
&= \hat{\mathcal{A}}^{(k+1)} \cdot \left(\mathcal{R} \cdot \hat{\mathcal{J}}^{(k)} \cdot \mathcal{R}^* + E_2\right) \cdot \left(\hat{\mathcal{A}}^{(k+1)}\right)^* - \tilde{\mathcal{A}}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)}\right)^* \\
&= \left(\hat{\mathcal{A}}^{(k+1)} \cdot \mathcal{R}\right) \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\hat{\mathcal{A}}^{(k+1)} \cdot \mathcal{R}\right)^* + \hat{\mathcal{A}}^{(k+1)} \cdot E_2 \cdot \left(\hat{\mathcal{A}}^{(k+1)}\right)^* \\
&\quad -\tilde{\mathcal{A}}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)}\right)^* \\
&= \left(\tilde{\mathcal{A}}^{(k+1)} + E_1\right) \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)} + E_1\right)^* + \hat{\mathcal{A}}^{(k+1)} \cdot E_2 \cdot \left(\hat{\mathcal{A}}^{(k+1)}\right)^* \\
&\quad -\tilde{\mathcal{A}}^{(k+1)} \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)}\right)^* \\
&= E_1 \cdot \hat{\mathcal{J}}^{(k)} \cdot \left(\tilde{\mathcal{A}}^{(k+1)}\right)^* + \left(\tilde{\mathcal{A}}^{(k+1)} + E_1\right) \cdot \hat{\mathcal{J}}^{(k)} \cdot E_1^* + \hat{\mathcal{A}}^{(k+1)} \cdot E_2 \cdot \left(\hat{\mathcal{A}}^{(k+1)}\right)^* .
\end{aligned}$$

Taking 2-norm on both sides,

$$\begin{aligned}
\left\|\hat{\Delta}^{(k)} - \tilde{\Delta}^{(k)}\right\|_2 &\leq \|E_1\|_2 \cdot \left\|\hat{\mathcal{J}}^{(k)}\right\|_2 \cdot \left\|\left(\tilde{\mathcal{A}}^{(k+1)}\right)^*\right\|_2 + \left\|\left(\tilde{\mathcal{A}}^{(k+1)} + E_1\right)\right\|_2 \cdot \left\|\hat{\mathcal{J}}^{(k)}\right\|_2 \cdot \|E_1^*\|_2 \\
&\quad + \left\|\hat{\mathcal{A}}^{(k+1)}\right\|_2 \cdot \|E_2\|_2 \cdot \left\|\left(\hat{\mathcal{A}}^{(k+1)}\right)^*\right\|_2 .
\end{aligned}$$

Lemma 4.7 follows by plugging all the corresponding upper bounds into this relation and simplifying. ∎

4.2.3 An Upper Bound on the Backward Errors

Our objective in §4.2.3 is to derive a 2-norm backward error bound on $\mathcal{H}$ in (4.57). Let

$$\hat{\mathcal{L}}^{(k)} \cdot \hat{\mathcal{D}}^{(k)} \cdot \left(\hat{\mathcal{L}}^{(k)}\right)^* = \hat{K}^{(k)} + \mathcal{H}^{(k)} \tag{4.64}$$

be the computed Cholesky factorization of $\hat{K}^{(k)}$, where $\hat{K}^{(k)}$ is the symmetric/Hermitian positive definite Cauchy-like matrix in (4.58); and $\mathcal{H}^{(k)}$ is the backward error matrix. In the following we derive an upper bound on $\|\mathcal{H}^{(k)}\|_2$.

It is well-known that

$$\hat{\mathcal{D}}^{(k)}_{1,1} = \gamma^{(k)} \quad \text{and} \quad \hat{\mathcal{L}}^{(k)}_{2:n-k+1} = \hat{l}^{(k)} ,$$

and that the Cholesky factorization can be rewritten as

$$\hat{\mathcal{L}}^{(k)} \cdot \hat{\mathcal{D}}^{(k)} \cdot \left(\hat{\mathcal{L}}^{(k)}\right)^* = \begin{pmatrix} \gamma^{(k)} & \gamma^{(k)} \cdot \left(\hat{l}^{(k)}\right)^* \\ \gamma^{(k)} \cdot \hat{l}^{(k)} & \gamma^{(k)} \cdot \hat{l}^{(k)} \cdot \left(\hat{l}^{(k)}\right)^* \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & \hat{\mathcal{L}}^{(k+1)} \cdot \hat{\mathcal{D}}^{(k+1)} \cdot \left(\hat{\mathcal{L}}^{(k+1)}\right)^* \end{pmatrix} .$$

Replacing the two Cholesky factorizations by the corresponding right hand sides in (4.64), we get

$$\hat{K}^{(k)} + \mathcal{H}^{(k)} = \begin{pmatrix} \gamma^{(k)} & \gamma^{(k)} \cdot \left(\hat{l}^{(k)}\right)^* \\ \gamma^{(k)} \cdot \hat{l}^{(k)} & \gamma^{(k)} \cdot \hat{l}^{(k)} \cdot \left(\hat{l}^{(k)}\right)^* \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & \hat{K}^{(k+1)} + \mathcal{H}^{(k+1)} \end{pmatrix} .$$

On the other hand, §4.2.1 implies that

$$\hat{K}^{(k)} = \begin{pmatrix} \gamma^{(k)} & (v^{(k)})^* \\ v^{(k)} & v^{(k)} \cdot (v^{(k)})^*/\gamma^{(k)} \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & K^{(k+1)} \end{pmatrix} .$$

Plugging this into above and simplifying,

$$\begin{aligned} \mathcal{H}^{(k)} &= \begin{pmatrix} 0 & \left(v^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)}\right)^* \\ v^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)} & v^{(k)} \cdot (v^{(k)})^*/\gamma^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)} \cdot \left(\hat{l}^{(k)}\right)^* \end{pmatrix} \\ &\quad + \begin{pmatrix} 0 & 0 \\ 0 & \left(\hat{K}^{(k+1)} - K^{(k+1)}\right) + \mathcal{H}^{(k+1)} \end{pmatrix} . \end{aligned}$$

Taking 2-norm on both sides and simplifying,

$$\begin{aligned} \|\mathcal{H}^{(k)}\|_2 &\leq \|\mathcal{H}^{(k+1)}\|_2 + \left\|\hat{K}^{(k+1)} - K^{(k+1)}\right\|_2 + \left\|\left(v^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)}\right)^*\right\|_2 \\ &\quad + \left\|v^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)}\right\|_2 + \left\|v^{(k)} \cdot (v^{(k)})^*/\gamma^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)} \cdot \left(\hat{l}^{(k)}\right)^*\right\|_2 . \end{aligned} \tag{4.65}$$

With arguments similar to those used in the proof of Lemma 4.5, it is easy to show that

$$\left\|v^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)}\right\|_2 \leq r^{1.5} \cdot n \cdot \eta \cdot \phi \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) .$$

Similarly, since $v^{(k)} = \gamma^{(k)} \cdot l^{(k)}$, it follows that

$$\begin{aligned} & \left\|v^{(k)} \cdot (v^{(k)})^*/\gamma^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)} \cdot \left(\hat{l}^{(k)}\right)^*\right\|_2 \\ \leq & \left\|\left(l^{(k)} - \hat{l}^{(k)}\right) \cdot \left(v^{(k)}\right)^*\right\|_2 + \left\|\hat{l}^{(k)} \cdot \left(v^{(k)} - \gamma^{(k)} \cdot \hat{l}^{(k)}\right)^*\right\|_2 \\ \leq & r^{1.5} \cdot n \cdot \eta \cdot \phi \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) . \end{aligned}$$

On the other hand,

$$\begin{aligned} \left\|\hat{K}^{(k+1)} - K^{(k+1)}\right\|_2 &\leq \frac{\left\|\hat{\Delta}^{(k+1)} - \Delta^{(k+1)}\right\|_2}{\phi_{\min}} \\ &\leq \frac{\left\|\hat{\Delta}^{(k+1)} - \tilde{\Delta}^{(k+1)}\right\|_2}{\phi_{\min}} + \frac{\left\|\tilde{\Delta}^{(k+1)} - \Delta^{(k+1)}\right\|_2}{\phi_{\min}} \\ &\leq \eta \cdot r \cdot n^2 \cdot \phi \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) . \end{aligned}$$

Plugging all these bounds into (4.65) and simplifying,

$$\|\mathcal{H}^{(k)}\|_2 \leq \|\mathcal{H}^{(k+1)}\|_2 + \eta \cdot r \cdot n^2 \cdot \phi \cdot \|\hat{K}^{(k)}\|_F + O(\epsilon^2) .$$

Since $\hat{K}$ is Hermitian positive definite, it follows that $\|\hat{K}^{(k)}\|_F \leq \|\hat{K}\|_F$ for all k. Solve the above recursion we obtain

Theorem 4.4 *Let $\mathcal{H}$ be defined as in (4.57). Then*

$$\|\mathcal{H}\|_2 \leq \eta \cdot r \cdot n^3 \cdot \phi \cdot \|\hat{K}\|_F + O(\epsilon^2) .$$

4.3 The Toeplitz and Toeplitz-plus-Hankel least squares problems

We will only analyze the round-off errors in Algorithm 4; the analysis and results for Algorithm 5 are similar. We set $\delta = n$ and $\zeta_1 = \zeta_3 = 1$. Let $\hat{x}_M$ be the solution computed by Algorithm 4. We want to identify conditions under which $\hat{x}_M$ can be written as $\hat{x}_M = \tilde{x}_M + \delta\tilde{x}_M$, where $\delta\tilde{x}_M$ is a small perturbation to $\tilde{x}_M$, which in turn is the exact solution to the perturbed linear least squares problem

$$\min_x \| (M + \delta M) \cdot x - (h + \delta h) \|_2 , \tag{4.66}$$

with δM, δh being small perturbations to M and h, respectively. As in Stewart [41, pp. 75–76], we say that Algorithm 4 is stable if such small perturbations $\delta\tilde{x}_M$, δM, and δh exist.

Let M be a Toeplitz matrix satisfying equation (3.35) with A numerically column unitary. In finite arithmetic, Algorithm 4 performs the following computations.

1. Compute $\hat{\Omega} = \mathrm{fl}(\Omega)$ and $\hat{\Lambda} = \mathrm{fl}(\Lambda)$;
2. compute $\tilde{h} := \mathrm{fl}\left(\mathcal{F}_m \cdot h\right)$, $\tilde{A} := \mathrm{fl}\left(\mathcal{F}_m \cdot A\right)$, and $\tilde{B} := \mathrm{fl}\left(B \cdot \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^*\right)$;
3. compute the LU factorization for $\tilde{C}_1$ and the generator for $Z = \tilde{C}_2 \cdot \tilde{C}_1^{-1}$, where $\tilde{C} \in \mathbf{C}^{m \times n}$ is the Cauchy-like matrix that satisfies the displacement equation $\Omega \cdot \tilde{C} - \tilde{C} \cdot \Lambda = \tilde{A} \cdot \tilde{B}$. Let $\hat{Z}$ be the computed Z.
4. compute the initial generator for $\tilde{K} = I_n + \hat{Z}^* \cdot \hat{Z}$ and solve $\tilde{K} \cdot g = \tilde{h}_{1:n} + \hat{Z}^* \cdot \tilde{h}_{n+1:m}$ to get $\hat{g}$.
5. solve $\tilde{C}_1 \cdot x_C = \hat{g}$ to get $\hat{x}_C$.
6. compute $\hat{x}_M := \mathrm{fl}\left(\mathcal{G}^{-1} \cdot (\mathcal{F}_n)^* \cdot \hat{x}_C\right)$.

In the following we assume that $\hat{\Omega} = \Omega$ and $\hat{\Lambda} = \Lambda$ so that we can apply the results in §4 directly. In actual computations, there are round-off errors in $\hat{\Omega}$ and $\hat{\Lambda}$ with $|\hat{\Omega} - \Omega| = O(\epsilon)$ and $|\hat{\Lambda} - \Lambda| = O(\epsilon)$, and hence Ω and Λ must be replaced by $\hat{\Omega}$ and $\hat{\Lambda}$, respectively. However the corresponding error analysis is almost identical to that in §4, with small increases in the value of η. Without loss of generality, we assume that no permutation actually takes place during the factorization of $\tilde{C}$ and $\tilde{K}$.

By relation (1.8), we can bound the round-off errors in $\tilde{h}$, $\tilde{A}$, and $\tilde{B}$ as

$$\begin{aligned}
\left\|\tilde{h} - \mathcal{F}_m \cdot h\right\|_2 &\leq \eta \cdot m \cdot \| |\mathcal{F}_m| \|_2 \cdot \| |h| \|_2 \leq \eta \cdot m^{1.5} \cdot \|h\|_2 \\
\left\|\tilde{A} - \mathcal{F}_m \cdot A\right\|_2 &\leq \eta \cdot r \cdot m \cdot \| |\mathcal{F}_m| \|_2 \cdot \| |A| \|_2 \leq \eta \cdot (r \cdot m)^{1.5} \\
\left\|\tilde{B} - B \cdot \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^*\right\|_2 &\leq \eta \cdot r \cdot n \cdot \| |B| \|_2 \cdot \left\|\mathcal{G}^{-1}\right\|_2 \cdot \| |\mathcal{F}_n|^* \|_2 \leq \eta \cdot (r \cdot n)^{1.5} \cdot \|B\|_F \ .
\end{aligned}$$

It now follows that

$$\left\|C - \tilde{C}\right\|_2 \leq \eta \cdot (r \cdot m)^{1.5} \cdot \psi \cdot \|C\|_F \ .$$

The same error bound holds (with different η) if M is given as a Toeplitz matrix and both A and B are computed from M for $\mathcal{G}$ chosen in Lemma 3.1.

We also have

$$\begin{pmatrix} I_n \\ \hat{Z} \end{pmatrix} \cdot \left(\hat{L}_1 \cdot \hat{U}\right) = \tilde{C} + \begin{pmatrix} H_{1:n,1:n} \\ F \end{pmatrix} = C - (C - \tilde{C}) + \begin{pmatrix} H_{1:n,1:n} \\ F \end{pmatrix} ,$$

where $\hat{L}_1 \cdot \hat{U}$ is the numerical LU factorization of $\tilde{C}_1$.

The round-off errors in solving $\tilde{K} \cdot g = \tilde{h}_{1:n} + \hat{Z}^* \cdot \tilde{h}_{n+1:m}$ come from four parts: computing the right hand side, computing the initial generator for $\tilde{K}$; factorizing $\tilde{K}$; and backward and forward substitutions. We will put all these round-off errors into $\tilde{h}_{1:n}$. First we set

$$\hat{h}_{1:n} + \hat{Z}^* \cdot \tilde{h}_{n+1:m} = \mathrm{fl}\left(\tilde{h}_{1:n} + \hat{Z}^* \cdot \tilde{h}_{n+1:m}\right) .$$

Applying (1.8) we have

$$\|\hat{h}_{1:n} - \tilde{h}_{1:n}\|_2 \leq \eta \cdot m \cdot \left(1 + \||\hat{Z}|\|_2\right) \cdot \||\tilde{h}|\|_2 \leq \eta \cdot m \cdot \left(1 + \|\hat{Z}\|_F\right) \cdot \|\tilde{h}\|_2 .$$

Let the numerical Cholesky factorization of $\tilde{K}$ satisfy

$$\hat{\mathcal{L}} \cdot \hat{\mathcal{G}} \cdot \hat{\mathcal{L}}^* = \tilde{K} + \mathcal{H} ,$$

where $\mathcal{H}$ is the round-off error matrix satisfying Theorem 4.4. We also put the round-off errors in computing the initial generator into $\mathcal{H}$; this only increases the bound on $\mathcal{H}$ by a constant factor. Hence $\hat{g}$ satisfies

$$\left(\tilde{K} + \mathcal{H} + \mathcal{H}_1\right) \cdot \hat{g} = \hat{h}_{1:n} + \hat{Z}^* \cdot \tilde{h}_{n+1:m} ,$$

where $\mathcal{H}_1$ is the error matrix due to substitutions and hence satisfies $\|\mathcal{H}_1\|_2 \leq \eta \cdot n^3 \cdot \|\tilde{K}\|_2$. The above equation can be re-written as

$$\tilde{K} \cdot \hat{g} = \hat{h}_{1:n} + \hat{Z}^* \cdot \tilde{h}_{n+1:m} - (\mathcal{H} + \mathcal{H}_1) \cdot \hat{g} .$$

In the following, we consider the round-off errors in $\hat{x}_C$. Since we only need to perform substitutions, it follows that

$$\left(\hat{L}_1 \cdot \hat{U} + \mathcal{H}_2\right) \cdot \hat{x}_C = \hat{g} ,$$

where $\|\mathcal{H}_2\|_\infty \leq \eta \cdot n^3 \cdot \|\tilde{C}_1\|_\infty$.

Set $\tilde{x}_M = \mathcal{G}^{-1} \cdot (\mathcal{F}_n)^* \cdot \hat{x}_C$ and let $\hat{x}_M = \tilde{x}_M + \delta\tilde{x}_M$. Then it is easy to show that $\|\delta\tilde{x}_M\|_2 \leq \eta \cdot n^2 \|\tilde{x}_M\|_2$. In the following, we identify δM and δh in (4.66). Set

$$\begin{aligned} M + \delta M &= \mathcal{F}_m^* \cdot \begin{pmatrix} I_n \\ \hat{Z} \end{pmatrix} \cdot \left(\hat{L}_1 \cdot \hat{U} + \mathcal{H}_2\right) \cdot \mathcal{G} \cdot \mathcal{F}_n \\ &= M + \mathcal{F}_m^* \cdot \left(\begin{pmatrix} I_n \\ \hat{Z} \end{pmatrix} \cdot \mathcal{H}_2 - (C - \tilde{C}) + \begin{pmatrix} H_{1:n,1:n} \\ F \end{pmatrix}\right) \cdot \mathcal{G} \cdot \mathcal{F}_n , \end{aligned}$$

and hence

$$\delta M = \mathcal{F}_m^* \cdot \left(\begin{pmatrix} I_n \\ \hat{Z} \end{pmatrix} \cdot \mathcal{H}_2 - (C - \tilde{C}) + \begin{pmatrix} H_{1:n,1:n} \\ F \end{pmatrix} \right) \cdot \mathcal{G} \cdot \mathcal{F}_n \, .$$

Similarly,

$$\begin{aligned} h + \delta h &= (\mathcal{F}_M)^* \cdot \begin{pmatrix} \hat{h}_{1:n} - (\mathcal{H} + \mathcal{H}_1) \cdot \hat{g} \\ \tilde{h}_{n+1:m} \end{pmatrix} \\ &= (\mathcal{F}_M)^* \cdot \left(\mathcal{F}_M \cdot h + \left(\tilde{h} - \mathcal{F}_M \cdot h \right) + \begin{pmatrix} \left(\hat{h}_{1:n} - \tilde{h}_{1:n} \right) - (\mathcal{H} + \mathcal{H}_1) \cdot \hat{g} \\ 0 \end{pmatrix} \right) , \end{aligned}$$

and hence

$$\delta h = (\mathcal{F}_M)^* \cdot \left(\left(\tilde{h} - \mathcal{F}_M \cdot h \right) + \begin{pmatrix} \left(\hat{h}_{1:n} - \tilde{h}_{1:n} \right) - (\mathcal{H} + \mathcal{H}_1) \cdot \hat{g} \\ 0 \end{pmatrix} \right) .$$

Taking 2-norm on δM we have

$$\begin{aligned} \|\delta M\|_2 &\leq \left(\left\| \begin{pmatrix} I_n \\ \hat{Z} \end{pmatrix} \right\|_2 \cdot \|\mathcal{H}_2\| + \|C - \tilde{C}\|_2 + \|H_{1:n,1:n}\|_2 + \|F\|_2 \right) \cdot \|\mathcal{G}\|_2 \\ &\leq \left((1 + \|\hat{Z}\|_2) \cdot \|\mathcal{H}_2\| + \|C - \tilde{C}\|_2 + \|H_{1:n,1:n}\|_2 + \|F\|_2 \right) \cdot n \, , \end{aligned}$$

where we have used the fact that we have chosen $\delta = n$ in Algorithm 4 and hence $\|\mathcal{G}\|_2 \leq n$. It now follows from Theorems 4.1 and 4.3, as well as (1.7) that

$$\|\delta M\|_2 \leq \eta \cdot r^3 \cdot m^{1.5} \cdot n^2 \cdot \phi \cdot (\sqrt{n} \cdot (\psi + \phi) + \sqrt{m}) \cdot (\nu + 1) \cdot (\|\hat{Z}\|_\infty + 1) \cdot \|C\|_\infty + O(\epsilon^2) \, .$$

To derive an upper bound on δh, we observe that

$$\|\tilde{K}^{-1}\|_2 \leq 1 \quad \text{and} \quad \|\tilde{K}^{-1} \cdot \hat{Z}\|_2 \leq 1 \, ,$$

and hence $\|\hat{g}\|_2 \leq 2 \cdot \|h\|_2 + O(\epsilon)$. Taking 2-norms on δh we have

$$\|\delta h\|_2 \leq \left\| \tilde{h} - \mathcal{F}_M \cdot h \right\|_2 + \left\| \hat{h}_{1:n} - \tilde{h}_{1:n} \right\|_2 + \|\mathcal{H} + \mathcal{H}_1\|_2 \cdot \|\hat{g}\|_2 \, .$$

Applying theorem 4.4 and the analysis in § 4.3 and simplifying,

$$\begin{aligned} \|\delta h\|_2 &\leq \eta \cdot r \cdot (m \cdot n)^{1.5} \cdot \phi \cdot \|\tilde{K}\|_F \cdot \|h\|_2 + O(\epsilon^2) \\ &\leq \eta \cdot r \cdot (m \cdot n)^{1.5} \cdot \phi \cdot (\sqrt{n} + \|\hat{Z}\|_F^2) \cdot \|h\|_2 + O(\epsilon^2) \, . \end{aligned}$$

Similar results hold for Algorithm 5. Theorem 4.5 below summarizes these results.

Theorem 4.5 *The computed solutions of Algorithms 4 and 5 to the least squares problem (1.5) can be written as $\hat{x}_M = \tilde{x}_M + \delta\tilde{x}_M$, where $\tilde{x}_M$ is the exact solution to (4.66). $\delta\tilde{x}_M$, δM and δh satisfy*

$$\begin{aligned} \|\delta\tilde{x}_M\|_2 &\leq \eta \cdot n^2 \|\tilde{x}_M\|_2 \\ \|\delta M\|_2 &\leq \eta \cdot r^3 \cdot m^{1.5} \cdot n^2 \cdot \phi \cdot (\sqrt{n} \cdot (\psi + \phi) + \sqrt{m}) \cdot (\nu + 1) \cdot (\|\hat{Z}\|_\infty + 1) \cdot \|C\|_\infty \\ &\quad + O(\epsilon^2) \\ \|\delta h\|_2 &\leq \eta \cdot r \cdot (m \cdot n)^{1.5} \cdot \phi \cdot (\sqrt{n} + \|\hat{Z}\|_F^2) \cdot \|h\|_2 + O(\epsilon^2) \, . \end{aligned}$$

Remark 10: It can be shown that similar norm upper bounds hold for the case $\zeta_1 > 1$, $\zeta_2 > 1$, and $\zeta_3 > 1$ as well, provided that there is little element growth within the generators for Algorithms 1 through 3 (see §2.3).

Remark 11: Since $\|\hat{Z}\|_\infty \leq 1 + \nu$, Algorithms 4 and 5 are stable if ν is not large, and unstable otherwise. It is known that in the LU factorization (4.39), $\kappa(\hat{L})$ could be as large as $O(2^n)$ in the worst case if C were a general dense matrix; consequently both ν and $\|\hat{Z}\|_\infty$ could be as large as $O(2^n)$. Hence the bounds on $\|\delta M\|_2$ and $\|\delta h\|_2$ could be as large as $O(4^n)$ in the worst case. It is not clear whether a sharper bound on ν exists for Cauchy-like matrices with low displacement rank.

Remark 12: It follows from Lemmas 3.1 and 3.2 that the parameters ψ and ϕ for Algorithm 4 are much smaller than for Algorithm 5. Although the above norm upper bounds on δM and δh are by no means tight, they do suggest that Algorithm 4 could be more accurate than Algorithm 5 if ν and $\|\hat{Z}\|_\infty$ are about the same for both algorithms. On the other hand, for both Algorithms 4 and 5, the norm upper bounds on δM and δh are much larger than those for the standard QR method, which are $\eta \cdot m \cdot n \|M\|_2$ and 0, respectively (see [21]). Hence Algorithms 4 and 5 appear to be less accurate than QR. Our numerical experiments indicate that Algorithm 5 is in general less accurate than Algorithm 4, which in turn is in general less accurate than QR; but the lost accuracy can be recovered by one step of iterative refinement. See §5 for more details.

5 Numerical Experiments

We have implemented Algorithms 1 through 5 in Fortran and have performed a large number of numerical experiments with them to investigate their behavior in finite arithmetic and to compare Algorithms 4 and 5 with other available algorithms. In this section we discuss some issues related to measuring backward errors and implementation, and report some of the numerical results. We chose $\zeta_1 = 10, \zeta_2 = \zeta_3 = 0$, and $\delta = n$ in Algorithms 4 and 5.

5.1 Backward Perturbation Bounds

In order to conveniently measure the round-off errors in Theorem 4.5, we use the following theorem of Gu [22] to push all of them into perturbations in M.

Theorem 5.1 *Assume that $\delta\tilde{x}_M$, δM, and δh in (4.66) are small perturbations to $\tilde{x}_M$, M, and h, respectively. Then there exists a $\widehat{\delta M}$ with*

$$\frac{\|\widehat{\delta M}\|_2}{\|M\|_2} \leq \frac{\|\delta M\|_2}{\|M\|_2} + 2\left(1 + \frac{\|\delta M\|_2}{\|M\|_2}\right) \cdot \left(\frac{\|\delta\tilde{x}_M\|_2}{\|\hat{x}_M\|_2} + \frac{\|\delta h\|_2}{\|h\|_2}\right) \Big/ \left(1 - 2 \cdot \frac{\|\delta h\|_2}{\|h\|_2}\right) ,$$

such that $\hat{x}_M = \tilde{x}_M + \delta\tilde{x}_M$ is the exact solution to the least squares problem

$$\min_x \|(M + \widehat{\delta M}) \cdot x - h\|_2 . \tag{5.67}$$

Let $\hat{x}_M$ be the numerical solution to the least squares problem (1.5) computed by Algorithm 4, Algorithm 5, or QR, we look for the smallest possible $\|\widehat{\delta M}\|_2$ in (5.67). If $\|\widehat{\delta M}\|_2$ is tiny compared to $\|M\|_2$, then $\hat{x}_M$ is a good approximation, otherwise it is not.

To compute such a $\|\widehat{\delta M}\|_2$, we use the following theorem of Gu [22], which is a modification of a result of Waldén, Karlson, and Sun [46] and Higham [29, Chapter 19].. Let $M = Q \cdot \begin{pmatrix} D \\ 0 \end{pmatrix} \cdot W^T$ be the singular value decomposition of M, where $Q \in \mathbf{R}^{m\times m}$ and $W \in \mathbf{R}^{n\times n}$ are orthogonal; and $D \in \mathbf{R}^{n\times n}$ is non-negative diagonal. Let

$$r = h - M \cdot \hat{x}_M = Q \cdot \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} \quad \text{and} \quad \eta = \frac{\|r\|_2}{\|\hat{x}_M\|_2}.$$

r is the *residual.* $r_1 = 0$ if $\hat{x}_M$ is the exact solution to (1.5). We assume that $\hat{x}_M \neq 0$.

Theorem 5.2 *Define*

$$\begin{aligned} \mathcal{E}(\hat{x}_M) &= \min\{\|\widehat{\delta M}\|_F, \ \textit{where } \widehat{\delta M} \textit{ satisfies (5.67).}\} \\ \tilde{\mathcal{E}}(\hat{x}_M) &= \min(\eta, \tilde{\sigma}), \quad \textit{where } \ \tilde{\sigma} = \sqrt{\frac{r_1^T \cdot D^2 \cdot (D^2 + \eta^2 I)^{-1} \cdot r_1}{\gamma^2/\eta^2 + \eta^2 \cdot r_1^T \cdot (D^2 + \eta^2 I_n)^{-2} \cdot r_1}}, \end{aligned}$$

where $\gamma = \|r_2\|_2$. *Then*

$$1 \leq \frac{\tilde{\mathcal{E}}(\hat{x}_M)}{\mathcal{E}(\hat{x}_M)} \leq \frac{\sqrt{5}+1}{2}.$$

We measure $\|\widehat{\delta M}\|_2$ by computing $\tau = \dfrac{\tilde{\mathcal{E}}(\hat{x}_M)}{\sqrt{m} \cdot \|M\|_2 \cdot \epsilon}$. Hence $\hat{x}_M$ is a good approximate solution if and only if $\tau \leq O(1)$.

5.2 Implementation Issues

A natural way to implement Algorithm 1 is to keep P_m and Q_n in vectors and keep both L and U in a single matrix W by storing L in the strict lower triangular part of W and U upper triangular part of W. However, arrays are stored column-wise in Fortran. Since U is generated and stored row by row in Algorithm 1, columns of W have to be moved into and brought out of fast memory for most steps of elimination for large n. This causes a significant amount of memory traffic between slow and fast memory levels in the memory hierarchy. For more detailed discussions on memory traffic, see for example [17, §2.6]. As in Gu [23], we reduce this memory traffic by storing rows of U column-wise in Algorithm 1. Let $S_n \in \mathbf{R}^{n\times n}$ be the matrix defined in §1.2. It follows that $\tilde{U} \equiv S_n \cdot U^T \cdot S_n$ is an upper triangular matrix, whose k^{th} column is the $(n-k+1)^{st}$ row of U in the reverse order. The backward substitution procedure for computing $U^{-1} \cdot y$ in Algorithm 1 can be rewritten as a forward substitution as $S_n \cdot \left(\left(\tilde{U}^T\right)^{-1} \cdot (S_n \cdot y)\right)$. Gu [23] shows that this technique significantly reduces memory traffic.

We further reduce the memory traffic by delaying permuting rows of L in Algorithms 1 and 2 until the factorization is completed. To be more specific, assume that at k^{th} step of

Matrix	Order		$\frac{1}{\kappa(T)}$	Execution Time (seconds)				
Type	m	n		NEW-I	NEW-II	NEW-III	NEW-IV	QR
1	320	300	2.38×10^{-3}	2.30×10^{-1}	2.60×10^{-1}	3.60×10^{-1}	4.30×10^{0}	9.40×10^{-1}
	640	600	1.28×10^{-3}	8.02×10^{-1}	8.84×10^{-1}	1.60×10^{0}	1.82×10^{0}	6.09×10^{0}
	1280	1200	1.08×10^{-3}	3.28×10^{0}	3.37×10^{0}	6.62×10^{0}	7.05×10^{0}	4.85×10^{1}
	2560	2400	7.95×10^{-4}	1.26×10^{1}	1.35×10^{1}	2.55×10^{1}	2.76×10^{1}	3.81×10^{2}
2	320	300	5.78×10^{-17}	2.02×10^{-1}	2.14×10^{-1}	3.80×10^{-1}	4.08×10^{-1}	8.04×10^{-1}
	640	600	4.07×10^{-17}	8.22×10^{-1}	8.66×10^{-1}	1.58×10^{0}	1.70×10^{0}	6.08×10^{0}
	1280	1200	2.69×10^{-17}	3.17×10^{0}	3.37×10^{0}	6.32×10^{0}	6.92×10^{0}	4.78×10^{1}
	2560	2400	1.02×10^{-17}	1.29×10^{1}	1.33×10^{1}	2.56×10^{1}	2.79×10^{1}	3.81×10^{2}
3	320	300	3.81×10^{-9}	2.16×10^{-1}	2.16×10^{-1}	3.92×10^{-1}	4.24×10^{-1}	7.92×10^{-1}
	640	600	1.93×10^{-13}	8.66×10^{-1}	8.24×10^{-1}	1.55×10^{0}	1.71×10^{0}	5.93×10^{0}
	1280	1200	2.01×10^{-15}	3.16×10^{0}	3.38×10^{0}	6.29×10^{0}	6.92×10^{0}	4.72×10^{1}
	2560	2400	5.19×10^{-15}	1.28×10^{1}	1.35×10^{1}	2.55×10^{1}	2.76×10^{1}	3.82×10^{2}

Matrix	Order		$\frac{1}{\kappa(T)}$	Backward Error (τ)				
Type	m	n		NEW-I	NEW-II	NEW-III	NEW-IV	QR
1	320	300	2.38×10^{-3}	2.07×10^{1}	2.18×10^{-1}	1.34×10^{0}	1.19×10^{-1}	3.00×10^{-2}
	640	600	1.28×10^{-3}	8.13×10^{1}	2.45×10^{-1}	1.41×10^{0}	2.48×10^{-1}	1.79×10^{-2}
	1280	1200	1.08×10^{-3}	1.33×10^{3}	3.11×10^{-1}	2.14×10^{0}	3.10×10^{-1}	1.72×10^{-2}
	2560	2400	7.95×10^{-4}	2.89×10^{2}	9.67×10^{-2}	7.65×10^{0}	9.69×10^{-2}	9.63×10^{-3}
2	320	300	5.78×10^{-17}	1.06×10^{0}	8.85×10^{-1}	5.97×10^{-1}	5.35×10^{-1}	2.57×10^{-1}
	640	600	4.07×10^{-17}	1.48×10^{0}	9.92×10^{-1}	5.18×10^{-1}	5.22×10^{-1}	2.63×10^{-1}
	1280	1200	2.69×10^{-17}	7.31×10^{-1}	5.42×10^{-1}	2.01×10^{0}	2.32×10^{0}	2.19×10^{-1}
	2560	2400	1.02×10^{-17}	2.23×10^{0}	2.34×10^{0}	2.87×10^{0}	2.48×10^{0}	2.20×10^{-1}
3	320	300	3.81×10^{-9}	6.61×10^{1}	4.48×10^{-1}	3.50×10^{0}	4.57×10^{-1}	1.24×10^{-1}
	640	600	1.93×10^{-13}	9.18×10^{1}	1.55×10^{2}	4.25×10^{0}	5.04×10^{-1}	1.33×10^{-1}
	1280	1200	2.01×10^{-15}	1.67×10^{1}	2.49×10^{1}	8.66×10^{0}	1.13×10^{1}	1.45×10^{-1}
	2560	2400	5.19×10^{-15}	2.79×10^{1}	4.26×10^{1}	1.76×10^{1}	3.35×10^{1}	1.35×10^{-1}

Table 1: Fortran Blas, Large Residuals

elimination, we need to swap two rows of L in order to bring the pivot to the (k,k) position. We achieve this by only swapping the corresponding rows in the generator matrix. After the factorization is completed, we restore rows of L to their proper positions. This technique is also used to swap columns of U in Algorithm 1. Our numerical experiments indicate that Algorithms 4 and 5 are less accurate than straightforward QR algorithm in many cases. To enhance accuracy we perform one step of Björck iterative refinement [2].

Algorithms 4 and 5 are unstable if the parameter ν in Theorem 4.5 is exceptionally large. We monitor the size of ν by computing $\|\hat{Y}^{(k)}\|_{\max}$ in Algorithm 1. Throughout our experiments, $\|\hat{Y}^{(k)}\|_{\max}$ never exceeded 200.

5.3 Numerical Results

The computations were done on an IBM RS6000 workstation in double precision, where the machine precision is $\epsilon \approx 1.1 \times 10^{-16}$.

MATRIX	ORDER		$\frac{1}{\kappa(T)}$	EXECUTION TIME (SECONDS)				
TYPE	m	n		NEW-I	NEW-II	NEW-III	NEW-IV	QR
1	320	300	2.38×10^{-3}	2.40×10^{-1}	2.50×10^{-1}	4.30×10^{-1}	4.50×10^{-1}	8.30×10^{-1}
	640	600	1.28×10^{-3}	8.30×10^{-1}	8.80×10^{-1}	1.62×10^{0}	1.77×10^{0}	6.26×10^{0}
	1280	1200	1.08×10^{-3}	3.21×10^{0}	3.41×10^{0}	6.38×10^{0}	7.06×10^{0}	4.80×10^{1}
	2560	2400	7.95×10^{-4}	1.27×10^{1}	1.32×10^{1}	2.54×10^{1}	2.75×10^{1}	3.81×10^{2}
2	320	300	5.78×10^{-17}	2.02×10^{-1}	2.18×10^{-1}	4.24×10^{-1}	4.46×10^{-1}	7.84×10^{-1}
	640	600	4.07×10^{-17}	8.40×10^{-1}	8.68×10^{-1}	1.57×10^{0}	1.77×10^{0}	5.96×10^{0}
	1280	1200	2.69×10^{-17}	3.24×10^{0}	3.41×10^{0}	6.45×10^{0}	7.10×10^{0}	4.75×10^{1}
	2560	2400	1.02×10^{-17}	1.28×10^{1}	1.36×10^{1}	2.55×10^{1}	2.73×10^{1}	3.80×10^{2}
3	320	300	3.81×10^{-9}	2.10×10^{-1}	2.18×10^{-1}	4.22×10^{-1}	4.44×10^{-1}	7.60×10^{-1}
	640	600	1.93×10^{-13}	7.96×10^{-1}	9.08×10^{-1}	1.59×10^{0}	1.74×10^{0}	6.01×10^{0}
	1280	1200	2.01×10^{-15}	3.14×10^{0}	3.38×10^{0}	6.30×10^{0}	6.90×10^{0}	4.73×10^{1}
	2560	2400	5.19×10^{-15}	1.29×10^{1}	1.37×10^{1}	2.54×10^{1}	2.75×10^{1}	3.80×10^{2}

MATRIX	ORDER		$\frac{1}{\kappa(T)}$	NORMALIZED RESIDUAL (τ)				
TYPE	m	n		NEW-I	NEW-II	NEW-III	NEW-IV	QR
1	320	300	2.38×10^{-3}	6.94×10^{2}	2.65×10^{0}	7.15×10^{0}	2.63×10^{0}	3.41×10^{-1}
	640	600	1.28×10^{-3}	3.91×10^{3}	8.40×10^{0}	1.04×10^{0}	8.42×10^{0}	4.45×10^{-1}
	1280	1200	1.08×10^{-3}	8.86×10^{4}	1.40×10^{1}	1.38×10^{1}	1.40×10^{1}	2.57×10^{-1}
	2560	2400	7.95×10^{-4}	3.89×10^{4}	4.66×10^{0}	4.54×10^{1}	4.65×10^{0}	3.25×10^{-1}
2	320	300	5.78×10^{-17}	2.01×10^{1}	9.34×10^{-1}	1.11×10^{0}	4.07×10^{-1}	2.62×10^{-1}
	640	600	4.07×10^{-17}	2.10×10^{2}	7.54×10^{-1}	2.80×10^{0}	4.13×10^{-1}	2.42×10^{-1}
	1280	1200	2.69×10^{-17}	4.53×10^{1}	5.13×10^{-1}	2.51×10^{0}	2.52×10^{0}	2.29×10^{-1}
	2560	2400	1.02×10^{-17}	1.69×10^{1}	2.34×10^{0}	3.75×10^{0}	3.34×10^{0}	2.12×10^{-1}
3	320	300	3.81×10^{-9}	3.88×10^{1}	1.22×10^{0}	1.46×10^{1}	1.21×10^{0}	3.33×10^{-1}
	640	600	1.93×10^{-13}	2.15×10^{2}	4.43×10^{2}	1.34×10^{1}	4.62×10^{0}	2.73×10^{-1}
	1280	1200	2.01×10^{-15}	5.07×10^{1}	2.60×10^{1}	1.27×10^{1}	1.97×10^{1}	3.19×10^{-1}
	2560	2400	5.19×10^{-15}	9.21×10^{1}	7.27×10^{1}	2.49×10^{1}	3.31×10^{1}	3.09×10^{-1}

Table 2: FORTRAN BLAS, SMALL RESIDUALS

We compared the following algorithms:

- **NEW-I:** Implementation of Algorithm 5 *without* iterative refinement.
- **NEW-II:** Implementation of Algorithm 5 *with* iterative refinement.
- **NEW-III:** Implementation of Algorithm 4 *without* iterative refinement.
- **NEW-IV:** Implementation of Algorithm 4 *with* iterative refinement.
- **QR:** LAPACK [1] subroutine DGELS for solving a general dense linear least squares problem using the QR method (see §2.1).

The cost for the first four algorithms is $O(mn)$ flops, and the cost for QR is $O(mn^2)$ flops.

We solved the Toeplitz linear least squares problem

$$\min_x \|T \cdot x - h\|_2$$

Matrix Type	Order		$\frac{1}{\kappa(T)}$	Execution Time (seconds)				
	m	n		NEW-I	NEW-II	NEW-III	NEW-IV	QR
1	320	300	2.38×10^{-3}	2.10×10^{-1}	2.10×10^{-1}	4.30×10^{-1}	4.60×10^{-1}	2.90×10^{-1}
	640	600	1.28×10^{-3}	7.56×10^{-1}	7.70×10^{-1}	1.46×10^{0}	1.53×10^{0}	2.15×10^{0}
	1280	1200	1.08×10^{-3}	2.89×10^{0}	2.96×10^{0}	5.71×10^{0}	6.05×10^{0}	1.53×10^{1}
	2560	2400	7.95×10^{-4}	1.14×10^{1}	1.20×10^{1}	2.31×10^{1}	2.43×10^{1}	1.15×10^{2}
2	320	300	4.02×10^{-17}	2.00×10^{-1}	2.24×10^{-1}	3.62×10^{-1}	4.10×10^{-1}	3.00×10^{-1}
	640	600	3.62×10^{-17}	7.24×10^{-1}	7.88×10^{-1}	1.43×10^{0}	1.49×10^{0}	2.19×10^{0}
	1280	1200	2.40×10^{-17}	2.86×10^{0}	2.98×10^{0}	5.72×10^{0}	6.07×10^{0}	1.53×10^{1}
	2560	2400	7.15×10^{-18}	1.13×10^{1}	1.18×10^{1}	2.32×10^{1}	2.41×10^{1}	1.16×10^{2}
3	320	300	3.81×10^{-9}	2.06×10^{-1}	1.98×10^{-1}	3.80×10^{-1}	3.72×10^{-1}	3.26×10^{-1}
	640	600	1.93×10^{-13}	7.72×10^{-1}	7.90×10^{-1}	1.43×10^{0}	1.53×10^{0}	2.17×10^{0}
	1280	1200	2.02×10^{-15}	2.83×10^{0}	2.98×10^{0}	5.83×10^{0}	5.93×10^{0}	1.52×10^{1}
	2560	2400	5.25×10^{-15}	1.16×10^{1}	1.19×10^{1}	2.30×10^{1}	2.44×10^{1}	1.16×10^{2}

Matrix Type	Order		$\frac{1}{\kappa(T)}$	Backward Error (τ)				
	m	n		NEW-I	NEW-II	NEW-III	NEW-IV	QR
1	320	300	2.38×10^{-3}	2.40×10^{-1}	2.50×10^{-1}	4.30×10^{-1}	4.50×10^{-1}	8.30×10^{-1}
	640	600	1.28×10^{-3}	8.65×10^{1}	2.45×10^{-1}	1.42×10^{0}	2.48×10^{-1}	2.36×10^{-2}
	1280	1200	1.08×10^{-3}	1.01×10^{3}	3.10×10^{-1}	2.15×10^{0}	3.11×10^{-1}	1.69×10^{-2}
	2560	2400	7.95×10^{-4}	3.51×10^{2}	9.67×10^{-2}	7.70×10^{0}	9.68×10^{-2}	1.06×10^{-2}
2	320	300	4.02×10^{-17}	1.02×10^{0}	6.55×10^{-1}	7.52×10^{-1}	7.06×10^{-1}	2.38×10^{-1}
	640	600	3.62×10^{-17}	1.52×10^{0}	1.30×10^{0}	5.56×10^{-1}	7.14×10^{-1}	2.66×10^{-1}
	1280	1200	2.40×10^{-17}	9.02×10^{-1}	6.48×10^{-1}	1.92×10^{0}	1.90×10^{0}	2.09×10^{-1}
	2560	2400	7.15×10^{-18}	2.07×10^{0}	2.37×10^{0}	2.82×10^{0}	2.76×10^{0}	2.30×10^{-1}
3	320	300	3.81×10^{-9}	6.07×10^{1}	4.55×10^{-1}	3.78×10^{0}	4.55×10^{-1}	1.36×10^{-1}
	640	600	1.93×10^{-13}	2.11×10^{2}	8.39×10^{2}	4.41×10^{0}	5.09×10^{-1}	1.32×10^{-1}
	1280	1200	2.02×10^{-15}	2.58×10^{1}	4.97×10^{1}	1.12×10^{1}	2.02×10^{1}	1.34×10^{-1}
	2560	2400	5.25×10^{-15}	2.22×10^{1}	3.76×10^{1}	1.75×10^{1}	3.20×10^{1}	1.38×10^{-1}

Table 3: Optimized Blas, Large Residuals

for the following types of Toeplitz matrices $T = (t_{k-j})_{1\le k\le m, 1\le j\le n}$:

- **Type 1:** $\{t_k\}$ randomly generated from uniform distribution on $(0,1)$. A Type 1 matrix is usually well-conditioned.

- **Type 2:** $t_0 = 2\omega$ and $t_k = \frac{\sin(2\pi\omega k)}{\pi k}$ for $k \neq 0$, where $\omega = 0.25$. A Type 2 matrix is also called the Prolate matrix in Gohberg, Kailath, and Olshevsky [18] and Varah [45]; T is very ill-conditioned.

- **Type 3:** A set of data was generated as

$$x_k = e^{-\alpha\cdot k}\cdot \sum_{j=1}^{\lfloor m/3\rfloor} \cdot \frac{j}{\lfloor m/3\rfloor + 1}\cos\left(\frac{j\cdot k\cdot\pi}{\lfloor m/3\rfloor+1}\right) + \beta\cdot r_k\,,$$

where $\alpha = \pi/n$; r_k is taken from a normal distribution, $r_k \in N(0,1)$; and $\beta = 10^{-7}$, 10^{-11}, 10^{-15}, and 10^{-18}. The matrix $T \in \mathbf{R}^{m\times n}$ was constructed from $t_{k-s} =$

x_{k-s+n}. This is a modification of Test V in Park and Eldén [39], which, in turn, is a modification of an example from Kolbel and Schafer [35]. T becomes more and more ill-conditioned as β becomes smaller and smaller.

We chose two types of h vectors:

- components of h are randomly generated from uniform distribution on $(0, 1)$. The Toeplitz linear least squares problem has a *large* residual.
- components of x_M are randomly generated from uniform distribution on $(0, 1)$ and $h = T \cdot x_M$. The problem has a *small* residual.

Numerical results are summarized in Tables 1 through 4. These results confirm that both Algorithms 4 and 5 are capable of solving problems ranging from well-conditioned to ill-conditioned to numerically singular, both for large residuals and small residuals; and they are significantly faster than QR.

Algorithm 5 is the fastest algorithm, whereas QR is the slowest. For $m = 2560$ and $n = 2400$, Algorithm 5 is up to 30 times faster than QR in Fortran BLAS and up to 10 times faster in optimized BLAS; and Algorithm 4 is up to 15 times faster than QR in Fortran BLAS and up to 5 times faster in optimized BLAS. However, if these algorithms are implemented in systolic arrays, one might expect the speedups of Algorithms 4 and 5 to be more like the Fortran BLAS speedups.

Algorithm 5 is the least accurate algorithm, whereas QR is the most accurate. For $m = 2560$ and $n = 2400$, the backward error in Algorithm 5 is upto 10^5 times larger than that in QR. With iterative refinement, the backward error in Algorithm 5 is at most 200 times as large. While Algorithm 4 costs at least twice as much as Algorithm 5, it is significantly more accurate. With or without iterative refinement, the backward error in Algorithm 4 is at most 200 times larger than that in QR.

6 Conclusions and Extensions

We have presented fast algorithms for solving the Toeplitz and Toeplitz-plus-Hankel linear least squares problems and shown them to be numerically stable under certain conditions. We have discussed implementation techniques that further improve their efficiency. Numerical experiments indicate that they are both numerically stable and efficient in practice.

The algorithms presented in this paper can be modified to solve *Mosaic Toeplitz* or *Block Toeplitz* linear least squares problems (cf. [11, 18]).

As Theorem 4.5 indicates, these new algorithms could be numerically unstable if the parameter ν is large. The best available upper bound on ν is $O(2^n)$, which has never been seen in practice. It is not clear whether sharper bound on ν exists for Cauchy-like matrices with low displacement rank.

One way to reduce the upper bound on ν is to perform a *rank-revealing LU* (RRLU) factorization on C instead of using GEPP/GECP (see, for example, Chan [8], Gu and Eisen-

MATRIX	ORDER		$\frac{1}{\kappa(T)}$	EXECUTION TIME (SECONDS)				
TYPE	m	n		NEW-I	NEW-II	NEW-III	NEW-IV	QR
1	320	300	2.38×10^{-3}	2.20×10^{-1}	2.50×10^{-1}	4.00×10^{-1}	4.90×10^{-1}	2.50×10^{-1}
	640	600	1.28×10^{-3}	7.62×10^{-1}	7.74×10^{-1}	1.43×10^{0}	1.54×10^{0}	2.15×10^{0}
	1280	1200	1.08×10^{-3}	2.88×10^{0}	3.03×10^{0}	5.73×10^{0}	6.04×10^{0}	1.52×10^{1}
	2560	2400	7.95×10^{-4}	1.13×10^{1}	1.18×10^{1}	2.32×10^{1}	2.39×10^{1}	1.16×10^{2}
2	320	300	4.02×10^{-17}	2.00×10^{-1}	2.12×10^{-1}	3.56×10^{-1}	3.74×10^{-1}	2.78×10^{-1}
	640	600	3.62×10^{-17}	7.32×10^{-1}	7.56×10^{-1}	1.45×10^{0}	1.55×10^{0}	2.15×10^{0}
	1280	1200	2.40×10^{-17}	2.85×10^{0}	2.94×10^{0}	5.77×10^{0}	6.02×10^{0}	1.52×10^{1}
	2560	2400	7.15×10^{-18}	1.12×10^{1}	1.19×10^{1}	2.31×10^{1}	2.40×10^{1}	1.17×10^{2}
3	320	300	3.81×10^{-9}	2.06×10^{-1}	2.10×10^{-1}	3.50×10^{-1}	4.16×10^{-1}	3.02×10^{-1}
	640	600	1.93×10^{-13}	7.78×10^{-1}	7.82×10^{-1}	1.40×10^{0}	1.46×10^{0}	2.12×10^{0}
	1280	1200	2.02×10^{-15}	2.83×10^{0}	3.05×10^{0}	5.69×10^{0}	6.08×10^{0}	1.55×10^{1}
	2560	2400	5.25×10^{-15}	1.17×10^{1}	1.20×10^{1}	2.32×10^{1}	2.44×10^{1}	1.17×10^{2}

MATRIX	ORDER		$\frac{1}{\kappa(T)}$	BACKWARD ERROR (τ)				
TYPE	m	n		NEW-I	NEW-II	NEW-III	NEW-IV	QR
1	320	300	2.38×10^{-3}	2.40×10^{-1}	2.50×10^{-1}	4.30×10^{-1}	4.50×10^{-1}	8.30×10^{-1}
	640	600	1.28×10^{-3}	4.15×10^{3}	8.43×10^{0}	1.01×10^{0}	8.45×10^{0}	4.28×10^{-1}
	1280	1200	1.08×10^{-3}	6.73×10^{4}	1.40×10^{1}	1.18×10^{1}	1.40×10^{1}	5.07×10^{-1}
	2560	2400	7.95×10^{-4}	4.60×10^{4}	4.65×10^{0}	4.69×10^{1}	4.65×10^{0}	4.29×10^{-1}
2	320	300	4.02×10^{-17}	1.47×10^{1}	9.79×10^{-1}	8.19×10^{-1}	3.65×10^{-1}	2.63×10^{-1}
	640	600	3.62×10^{-17}	2.45×10^{2}	7.37×10^{-1}	6.97×10^{-1}	6.85×10^{-1}	2.51×10^{-1}
	1280	1200	2.40×10^{-17}	6.51×10^{1}	6.58×10^{-1}	2.07×10^{0}	1.16×10^{0}	2.23×10^{-1}
	2560	2400	7.15×10^{-18}	4.00×10^{1}	2.30×10^{0}	4.14×10^{0}	3.47×10^{0}	2.24×10^{-1}
3	320	300	3.81×10^{-9}	4.21×10^{1}	1.21×10^{0}	1.88×10^{1}	1.21×10^{0}	3.73×10^{-1}
	640	600	1.93×10^{-13}	2.33×10^{2}	8.25×10^{2}	1.08×10^{1}	4.60×10^{0}	3.98×10^{-1}
	1280	1200	2.02×10^{-15}	2.74×10^{1}	4.90×10^{1}	1.67×10^{1}	2.15×10^{1}	3.97×10^{-1}
	2560	2400	5.25×10^{-15}	6.44×10^{1}	7.86×10^{1}	2.40×10^{1}	3.22×10^{1}	3.90×10^{-1}

Table 4: OPTIMIZED BLAS, SMALL RESIDUALS

stat [24] and Hwang, Lin, and Yang [30]). It is interesting to see if fast RRLU factorization algorithms can be developed to guarantee that ν is *always* modest.

Acknowledgements. The author is grateful to Prof. S. Chandrasekaran for helpful discussions, and Dr. V. Olshevsky for preprints of his papers.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide.* SIAM, Philadelphia, PA, second edition, 1994.

[2] A. Björck. Iterative refinement of linear least squares solutions I. *BIT*, 7:257–278, 1967.

[3] A. Bojanczyk and R. P. Brent. Parallel solution of certain Toeplitz least-squares problems. *Lin. Alg. Appl.*, 77:43–60, 1986.

[4] A. W. Bojanczyk, R. P. Brent, and F. de Hoog. QR factorization of Toeplitz matrices. *Numer. Math.*, 49:81–94, 1986.

[5] A. W. Bojanczyk, R. P. Brent, and D. R. Sweet. On the stability of the Bareiss and related Toeplitz factorization algorithms. *SIAM J. Matrix Anal. Appl.*, 16:40–57, 1995.

[6] R. P. Brent. Parallel algorithms for Toeplitz systems. In G. H. Golub and P. van Dooren, editors, *Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms*. Springer, 1990.

[7] J. R. Bunch. Stability of methods for solving Toeplitz systems of equations. *SIAM J. Sci. Stat. Comp.*, 6:349–364, 1985.

[8] T. F. Chan. On the existence and computation of LU-factorizations with small pivots. *Math. Comp.*, 42:535–547, 1984.

[9] S. Chandrasekaran and A. H. Sayed. A fast and stable solver for nonsymmetric structured systems. Preprint, submitted for publication, 1995.

[10] S. Chandrasekaran and A. H. Sayed. Stabilizing the fast generalized Schur algorithm. Preprint, submitted for publication, 1995.

[11] J. Chun and T. Kailath. Generalized displacement structure for block-Toeplitz, Toeplitz-block, and Toeplitz-derived matrices. *SIAM J. Matrix Anal. Appl.*, 15:114–128, 1994.

[12] J. Chun, T. Kailath, and H. Lev-Ari. Fast parallel algorithms for QR and triangular factorization. *SIAM J. Sci. Stat. Comput.*, 8:899–913, 1987.

[13] G. Cybenko. The numerical stability of Levinson-Durbin algorithm for Toeplitz systems of equations. *SIAM J. Sci. Stat. Comput.*, 1:303–319, 1980.

[14] G. Cybenko. A general orthorgonalization technique with applications to time series analysis and signal processing. *Math. Comp.*, 40:323–336, 1983.

[15] G. Cybenko. Fast Toeplitz orthorgonalization using inner products. *SIAM J. Sci. Stat. Comput.*, 8:734–740, 1987.

[16] J. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization. *Math. Comp.*, 30:772–795, 1976.

[17] J. Demmel. Berkeley Lecture Notes in Numerical Linear Algebra. Mathematics Department, University of California, 1993.

[18] I. Gohberg, T. Kailath, and V. Olshevsky. Fast Gaussian elimination with partial pivoting for matrices with displacement structure. *Math. Comp.*, 1995. To appear.

[19] I. Gohberg and V. Olshevsky. Fast algorithm for matrix Nehari problem. In U. Helmke, R. Mennicken, and J. Sauers, editors, *Systems and networks: mathematical theory and applications, proceedings of the international symposium MTNS-93*, volume 2, pages 687–690, 1994.

[20] I. Gohberg and V. Olshevsky. Fast state space algorithms for matrix Nehari and Nehari-Takagi interpolation problems. *Integral Equations and Operator Theory*, 20:44–83, 1994.

[21] G. Golub and C. van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, MD, 2nd edition, 1989.

[22] M. Gu. Backward perturbation bounds for linear least squares problems. LBL Report LBL-37976, Lawrence Berkeley National Laboratory, November 1995.

[23] M. Gu. Stable and efficient algorithms for structured systems of linear equations. LBL Report LBL-37690, Lawrence Berkeley National Laboratory, September 1995.

[24] M. Gu and S. C. Eisenstat. An efficient algorithm for computing a rank-revealing QR factorization. Research Report YALEU/DCS/RR-967, Department of Computer Science, Yale University, June 1993. To appear in SIAM J. Num. Anal.

[25] G. Heinig. Inversion of generalized Cauchy matrices and other classes of structured matrices. In *Linear algebra in signal processing, IMA volumes in mathematics and its applications*, volume 69, pages 95 – 114, 1994.

[26] G Heinig. Transformation techniques for structured matrices, 1995. Seminar talk, Computer Science Division, University of California.

[27] G. Heinig, P. Jankowski, and K. Rost. Fast inversion of Toeplitz-plus-Hankel matrices. *Numer. Math*, 52:665–682, 1988.

[28] G. Heinig and K. Rost. Algebraic methods for Toeplitz-like matrices and operators. *Operator Theory*, 13:109–127, 1984.

[29] N. J. Higham. *Accuracy and Stability of Numerical Algorithms.* SIAM, 1996. To appear.

[30] T.-M. Hwang, W.-W. Lin, and E. K. Yang. Rank revealing LU factorizations. *Lin. Alg. Applics.*, 175:115–141, 1992.

[31] T. Kailath, S. Kung, and M. Morf. Displacement ranks of matrices and linear equations. *J. Math. Anal. and Appl.*, 68:395–407, 1979.

[32] T. Kailath and V. Olshevsky. Symmetric and Bunch-Kaufman pivoting for Cauchy-like matrices with applications to Toepliz-like matrices. Unpublished manuscript, 1995.

[33] T. Kailath and A. H. Sayed. Fast algorithms for generalized displacement structures. In H. Kimura and S. Kodama, editors, *Recent advances in mathematical theory of systems, control, networks and signal processing*, volume II, pages 27–32. Mita press, Japan, 1992.

[34] T. Kailath and A. H. Sayed. Displacement structure: Theory and applications. *SIAM Review*, 1995. To appear.

[35] W. Kolbel and H. Schafer. Improvement and automation of the LPSVD algorithm by continuous regularization of the singular values. *J. Magnetic Resonance*, 100:598–603, 1992.

[36] F. T. Luk and S. Qiao. A fast but unstable orthogonal triangularization technique for Toeplitz matrices. *Lin. Alg. Appl.*, 88/89:495–506, 1987.

[37] J. G. Nagy. Fast inverse QR factorization for Toeplitz matrices. *SIAM J. Sci. Stat. Comput.*, 14:1174–1183, 1993.

[38] J. G. Nagy. Applications of Toeplitz systems. SIAM News, October 1995.

[39] H. Park and L. Eldén. Stability analysis and fast algorithms for triangularization of Toeplitz matrices. Technical Report Lith-Mat-R-95-16, Department of Mathematics, Linköping University, May 1995.

[40] S. Qiao. Hybrid algorithm for fast Toeplitz orthogonalization. *Numer. Math.*, 53:351–366, 1988.

[41] G. W. Stewart. *Introduction to Matrix Computations.* Academic Press, New York, 1973.

[42] D. R. Sweet. Fast Toeplitz orthogonalization. *Numer. Math.*, 43:1–21, 1984.

[43] D. R. Sweet. The use of pivoting to improve the numerical performance of Toeplitz matrix algorithms. *SIAM J. Matrix Anal. Appl.*, 14:468–493, 1993.

[44] D. R. Sweet and R. P Brent. Error analysis of a fast partial pivoting method for structured matrices. In T. Luk, editor, *Advanced Signal Processing algorithms, Proc. of SPIE*, volume 2363, pages 266–280, 1995.

[45] J. M. Varah. The Prolate matrix. *Lin. Alg. Appl.*, 187:269–278, 1993.

[46] B. Waldén, R. Karlson, and J.-G. Sun. Optimal backward perturbation bounds for the linear least square problem. *Numer. Lin. Alg. Appl.*, 2:271–286, 1995.

[47] J. H. Wilkinson. Error analysis of direct methods of matrix inversion. *J. ACM*, 10:281–330, 1961.

[48] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Clarendon Press, Oxford, 1965.

LAWRENCE BERKELEY NATIONAL LABORATORY
UNIVERSITY OF CALIFORNIA
TECHNICAL & ELECTRONIC INFORMATION DEPARTMENT
BERKELEY, CALIFORNIA 94720