Bunch-Kaufman Pivoting for Partially Reconstructible Cauchy-like Matrices, with Applications to Toeplitz-like Linear Equations and to Boundary Rational Matrix Interpolation Problems [*]

Thomas Kailath and Vadim Olshevsky [†]

Abstract

In an earlier paper [GKO95] we exploited the *displacement structure* of *Cauchy-like matrices* to derive for them a fast $O(n^2)$ implementation of Gaussian elimination with *partial pivoting*. One application is to the rapid and *numerically accurate* solution of linear systems with *Toeplitz-like* coefficient matrices, based on the fact that the latter can be transformed into Cauchy-like matrices by using the Fast Fourier, Sine or Cosine Transforms. However symmetry is lost in the process, and the algorithm of [GKO95] is not optimal for *Hermitian* coefficient matrices.

In this paper we present a new fast $O(n^2)$ implementation of *symmetric Gaussian elimination with partial diagonal pivoting* for *Hermitian Cauchy-like matrices*, and show how to transform Hermitian Toeplitz-like matrices to Hermitian Cauchy-like matrices, obtaining algorithms that are now twice as fast as those in [GKO95].

Numerical experiments indicate that in order to obtain not only fast but also *numerically accurate* methods, it is advantageous to explore the important case in which the corresponding *displacement operators* have nontrivial kernels; this situation gives rise to what we call *partially reconstructible matrices*, which are introduced and studied in the present paper. We extend the transformation technique and the generalized Schur algorithms (i.e., fast displacement-based implementations of Gaussian elimination) to partially reconstructible matrices. We show by a variety of computed examples that the incorporation of *diagonal pivoting* methods leads to high accuracy.

We focused in this paper on the design of new numerically reliable algorithms for Hermitian Toeplitz-like matrices. However, the proposed algorithms have other important applications; in particular, we briefly describe how they recursively solve a boundary interpolation problem for J-unitary rational matrix functions.

1 Introduction

Toeplitz linear systems. Linear systems of equations with Hermitian Toeplitz coefficient matrices of the form $T = \left[\begin{array}{c} t_{i-j} \end{array} \right]_{1 \le i,j \le n}$ appear in a variety of applications in the sciences and engineering; a partial list can be found in [B85]. They can be rapidly solved in only $O(n^2)$ operations by the well-known Levinson and Schur algorithms, which however often produce very inaccurate results for indefinite matrices. The reason is the recursive nature of the above algorithms, which sequentially process all leading submatrices, breaking down if some of them are singular. If one

[*]This work was supported in part by ARO contract DAAH04-96-1-0176 and also by DARPA contract F49620-95-1-0525. The views and conclusions contained in these documents are those of the author(s) and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Advanced Research Projects Agency, the Army Research office, or the U.S. Government.

[†]Information System Laboratory, Stanford University, Stanford CA 94305 – 4055, U.S.A.; E-mail : tk@rascals.stanford.edu olshevsk@rascals.stanford.edu

encounters nonsingular, but ill-conditioned submatrices, then *near-break-downs* occur, resulting in the loss of accuracy.

Recently *look-ahead approaches* have been suggested to overcome the above difficulty by exploring (look-ahead) steps in which one jumps from one well-conditioned submatrix to another; several versions of look-ahead Levinson and Schur algorithms have been designed, see, e.g. [CH92], [FZ93], [GH94], [BH94] and references therein. However the arithmetic complexity of these algorithms depends upon the number of ill-conditioned leading submatrices, and in the worst case one has to perform $O(n^3)$ arithmetic operations, thus losing superiority in speed over the standard algorithms.

An alternative fast and stable algorithm was recently developed in [GKO95], based on the transformation of a Toeplitz-like matrix to a Cauchy-like matrix (cf. [P90], [GO94b], [H95a]), followed by a fast implementation of Gaussian elimination with partial pivoting (fast GEPP) for the resulting Cauchy-like matrix. A large set of numerical experiments in [GKO95] showed that the GKO algorithm has reliable numerical behavior, and can be recommended for solving large indefinite Toeplitz-like systems.

Because of its importance in the sequel, we shall briefly review the GKO algorithm here; however before doing so we introduce the necessary notations, see, e.g., the recent review [KS95].

Displacement structure. A matrix R is said to have a displacement structure if it satisfies a so-called *displacement equation* of the form

$$\Omega \cdot R \cdot \Delta^* - F \cdot R \cdot A^* = G \cdot B^*, \tag{1.1}$$

where Ω, Δ, F and A are some lower triangular matrices, and the so-called *generator* matrices $\{G, B\}$ have a small number, say α, of columns. If the matrices Ω, Δ, A, F are simple in form, say they are shift or diagonal matrices, and the number α is smaller than n, then the structure of R is conveniently captured by the $2\alpha n$ entries of $\{G, B\}$. Particular choices for the Ω, Δ, A, F lead to the definitions of the basic structured classes of Toeplitz-like, Hankel-like, Vandermonde-like, Cauchy-like matrices and to several other classes of structured matrices. In particular *Toeplitz-like matrices* can be defined by using a displacement equation of the form,

$$Z_1 \cdot R - R \cdot Z_{-1} = G \cdot B^* \tag{1.2}$$

where Z_ϕ is a ϕ-circulant lower shift matrix, having ones on the first subdiagonal, ϕ in the $(1, n)$ position, and zeros elsewhere. Analogously *Cauchy-like matrices* will be defined here via the displacement equation

$$D_1 \cdot R - R \cdot D_2 = G \cdot B^*, \tag{1.3}$$

with diagonal D_1, D_2. The reason for the above designations is the easily verified fact that the corresponding displacement ranks of an arbitrary Toeplitz matrix $T = \left[\begin{array}{c} t_{i-j} \end{array} \right]_{1 \le i,j \le n}$ and a Cauchy matrix $C = \left[\begin{array}{c} \frac{1}{x_i - y_j} \end{array} \right]_{1 \le i,j \le n}$ are $\alpha \le 2$ and $\alpha = 1$, respectively.

The GKO algorithm. The GKO algorithm, mentioned above, consists of the following two steps :

- Transformation of a Toeplitz-like matrix to a Cauchy-like matrix.
- Running a fast GEPP algorithm for this Cauchy-like matrix.

The development of the GKO algorithm is based on the well-known fact that Z_ϕ is diagonalized by the (scaled) Discrete Fourier transform matrix $\mathcal{F}$, i.e.,

$$Z_\phi = D_\phi^{-1} \cdot \mathcal{F}^* \cdot \Lambda_\phi \cdot \mathcal{F} \cdot D_\phi, \tag{1.4}$$

where Λ_ϕ and D_ϕ are certain diagonal matrices. Therefore the transformation (cf. [P90], [GO94b], [H95a], [GKO95]) of a Toeplitz-like matrix R in (1.2) into a Cauchy-like matrix $\mathcal{F}RD^{-1}\mathcal{F}^*$ is reduced to just performing 2α FFT's on the columns of $\{G, B\}$:

$$\Lambda_1 \cdot (\mathcal{F}RD_{-1}^{-1}\mathcal{F}^*) - (\mathcal{F}RD_{-1}^{-1}\mathcal{F}^*) \cdot \Lambda_{-1} = (\mathcal{F}G) \cdot (\mathcal{F}D_{-1}B)^*. \tag{1.5}$$

This observation allowed efficient implementation of GEPP for the obtained Cauchy-like matrix.

A problem : loss of symmetry. The weakness of the definition of Toeplitz-like matrices via equation (1.2) is that it ignores the symmetry of R, in the sense that one still has two (different) generator matrices $\{G, B\}$; and as a result the Cauchy-like matrix $\mathcal{F}RD_{-1}^{-1}\mathcal{F}^*$ in (1.5) is no longer Hermitian. At first glance, the design of a symmetry-preserving transformation seems to cause no additional difficulties, because there are many equivalent forms of displacement equations for Toeplitz-like matrices. For example, for a Hermitian R one can simply return to the *ϕ-cyclic* displacement equation

$$\nabla_{Z_\phi}(R) = R - Z_\phi \cdot R \cdot Z_\phi^* = G \cdot J \cdot G^*, \tag{1.6}$$

where J is a $\alpha \times \alpha$ signature matrix, so the structure of R is now captured by the αn entries of only one generator matrix G. [1] For any $|\phi| \neq 1$ the identity (1.4) can be used to transform a *Hermitian* Toeplitz-like matrix R into a *Hermitian* Cauchy-like matrix $\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*$, satisfying

$$\nabla_{\Lambda_\phi}(\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*) = (\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*) - \Lambda_\phi \cdot (\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*) \cdot \Lambda_\phi^* = (\mathcal{F}D_\phi G) \cdot J \cdot (\mathcal{F}D_\phi G)^*. \tag{1.7}$$

Then instead of fast GEPP (exploited in the GKO algorithm), which would destroy the Hermitian character of $\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^*$, we could now implement a (symmetric) Gaussian elimination algorithm with Bunch-Kaufman pivoting [BK77]. The Bunch-Kaufman scheme is a *symmetry-preserving* partial pivoting technique for general Hermitian matrices, and it is the algorithm of choice in LAPACK, which is renowned for the numerical reliability of the algorithms recommended. Using a fast implementation of the Bunch-Kaufman factorization algorithm for Hermitian Cauchy-like matrices one can rapidly compute the factorization $\mathcal{F}D_\phi RD_\phi^*\mathcal{F}^* = PLDL^*P^T$ with unit lower triangular L, and block-diagonal D (with each diagonal block having size 1 or 2, see section 5 for the details); this factorization allows one to solve the associated linear system in $O(n^2)$ operations. For *any* choice of ϕ in (1.6) the resulting algorithm is about twice as fast as the original GKO algorithm.

The problem is, that while the different choices of ϕ are all theoretically equivalent, the corresponding computational schemes have different numerical properties. Our experiments indicate that choices with $|\phi| \neq 1$ are impractical, often leading to overflows, and losing accuracy for reasonably well-conditioned Toeplitz systems. By extensive testing we found that algorithms corresponding to the choice $\phi = 1$ allow good numerical properties.

Partially reconstructible matrices. However the choice $\phi = 1$ in (1.6) complicates the analysis, because both the displacement operators, $\nabla_{Z_\phi}(\cdot) : \mathbf{C}^{n\times n} \to \mathbf{C}^{n\times n}$ in (1.6) and $\nabla_{\Lambda_\phi}(\cdot) : \mathbf{C}^{n\times n} \to \mathbf{C}^{n\times n}$ in (1.7), acting in the linear space $\mathbf{C}^{n\times n}$ now have nontrivial kernels (i.e., non-empty nullspaces). Following a suggestion of Georg Heinig, we refer to such R as *partially reconstructible* matrices, because now only part of the information on R is contained in its generator $\{G, J\}$ on the right-hand side of (1.6). In order to develop accurate algorithms we extended several results of displacement structure theory to partially reconstructible matrices. We showed how partially reconstructible Toeplitz-like matrices can be efficiently transformed to partially reconstructible

[1] Such a displacement equation was mentioned in the concluding remarks of the first "displacement" papers [KKM79a],[KKM79b] and it was recently closely studied in [AG90] and in [GO92]. In [GO94b] we used it to transform Cauchy-like matrices and Vandermonde-like matrices to Toeplitz-like matrices to speed-up matrix-vector multiplication. See also [BP94], p. 199, 200 for the presentation of these results of [GO94b].

Cauchy-like matrices, while preserving a Hermitian structure. We also showed that the displacement structure of partially reconstructible matrices is inherited by their Schur complements, and extended to partially reconstructible matrices the *generalized Schur algorithms* (corresponding to *discrete-time* and *continuous-time* Lyapunov displacement operators, defined in Sec. 2 below), for the usual theory see the review [KS95]. We showed that for partially reconstructible Cauchy-like matrices symmetric and Bunch-Kaufman pivoting can be adopted by these fast triangularization algorithms, thus leading to fast $O(n^2)$ implementation of symmetric Gaussian elimination with Bunch-Kaufman pivoting (fast SGEBKP) for these matrices. Finally, we verified by a large number of tests that the resulting algorithms provide high relative accuracy.

Although we focused in this paper on the use of the fast SGEBKP algorithms to design new accurate Hermitian Toeplitz-like solvers, these algorithms have other important applications in system theory. For example, partially reconstructible Cauchy-like matrices appear in the context of the boundary homogeneous interpolation problem for J-unitary rational matrix functions, which in turn parameterizes all solutions for boundary non-homogeneous interpolation problems with norm constraints, including the classical boundary Nevanlinna-Pick matricial interpolation problem, see [BGR91]. The fast SGEBKP algorithms for Cauchy-like matrices of this paper suggest a recursive and computationally efficient solution to the above interpolation problems, over the right-half plane and over the unit disk.

Contents. The paper is structured as follows. In the next section we study a discrete-time Lyapunov displacement operator with a nontrivial kernel, and introduce *partially reconstructible Toeplitz-like and Cauchy-like* matrices. We further suggest an efficient method to transform partially reconstructible Toeplitz-like matrices into partially reconstructible Cauchy-like matrices, both defined via discrete-time Lyapunov displacement operators. Then in section 3 we show how a discrete-time Lyapunov displacement equation can be transformed into a continuous-time one, and vice versa, without changing the solution R. We show that in many cases (e.g., for Cauchy-like matrices) such transformation can be performed with linear complexity, thus allowing one to exploit any form (i.e., discrete-time and continuous-time) of displacement equation as convenient. In section 4 we extend to partially reconstructible matrices the generalized Schur algorithms for continuous-time Lyapunov and discrete-time Lyapunov displacement operators with nontrivial kernels. We further simplify these algorithms for partially reconstructible Cauchy-like matrices. In section 5 we incorporate symmetric and Bunch-Kaufman pivoting techniques into these algorithms. The results of our numerical experiments are described in Section 6. In Section 7 we briefly outline how to use the fast SGEBKP algorithms for partially reconstructible Cauchy-like matrices to obtain a recursive solution for the boundary homogeneous interpolation problem for J-unitary rational matrix functions. Some concluding remarks are presented in the last section.

2 Discrete-time Lyapunov displacement operators

2.1. Displacement structure. Let F be a given $n \times n$ matrix. Following [CKLA87], introduce in the linear space $\mathbf{C}^{n\times n}$ of all $n \times n$ complex matrices the *displacement operator* $\nabla_F(\cdot) : \mathbf{C}^{n\times n} \to \mathbf{C}^{n\times n}$ by

$$\nabla_F(R) = R - F \cdot R \cdot F^*, \tag{2.1}$$

where the superscript * stands for conjugate transpose. Often $\nabla_F(\cdot)$ is called a *discrete-time Lyapunov* displacement operator (the names *Toeplitz-like*, or *symmetric Stein* displacement operator are also in use), to distinguish it from the so-called *continuous-time Lyapunov* displacement operator, which will be considered in the next section.

Let $R \in \mathbf{C}^{n\times n}$ be a Hermitian matrix, and p and q be the number of strictly positive and strictly negative eigenvalues of $\nabla_F(R)$, respectively. Then one can factor (nonuniquely)

$$\nabla_F(R) = R - F \cdot R \cdot F^* = G \cdot J \cdot G^*, \qquad \text{where} \qquad J = \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix} \in \mathbf{C}^{\alpha\times\alpha}, \tag{2.2}$$

with some rectangular $G \in \mathbf{C}^{n\times\alpha}$, where $\alpha = p + q$. The matrices $\{G, J\}$ are called the ∇_F-*generator* of R, and $\alpha = \text{rank}\nabla_F(R)$ of the ∇_F-generator is called the ∇_F-*displacement rank* of R. A matrix R is said to have a ∇_F-*displacement structure* if its ∇_F-displacement rank α is small compared to the size. As is now well-known, Toeplitz, Hankel, Toeplitz-plus-Hankel, Vandermonde, Chebyshev-Vandermonde, Cauchy, Pick, Loewner matrices, Bezoutians and many other important matrices, encountered in applications, have a displacement structure. Below we restrict ourself to Toeplitz-like and Cauchy-like matrices, the formal definitions of which will be given below.

2.2. Partially reconstructible matrices. In the literature on displacement structure, the operator $\nabla_F(\cdot)$ in (2.1) is generally assumed to be invertible, so that the ∇_F-generator $\{G, J\}$ contains all the information on R. However, displacement equations with a kernel (nullspace) of low dimension appear in certain problems, see, e.g., [CK91], [GO94a]. The task addressed in the present paper, i.e., a fast and *accurate* implementation of symmetric Gaussian elimination with diagonal pivoting for Cauchy-like matrices, which can be then utilized for Toeplitz-like matrices, exhibits another case where one gains by considering a displacement equation with a nontrivial kernel. Therefore $\mathcal{K} = \text{Ker}\nabla_F(\cdot)$ is assumed below to be nontrivial. In the latter situation a matrix R with displacement structure (2.2) will be called a *partially reconstructible* matrix, because only part of the information on R is contained in its ∇_F-generator $\{G, J\}$. To extend the definition of a ∇_F-generator to partially reconstructible matrices, let us represent $R \in \mathbf{C}^{n\times n}$ as

$$R = R_{\mathcal{K}} + R_{\mathcal{K}^\perp} \tag{2.3}$$

with respect to the orthogonal decomposition

$$\mathbf{C}^{n\times n} = \mathcal{K} \oplus \mathcal{K}^\perp. \tag{2.4}$$

A partially reconstructible matrix R is uniquely determined by the three matrices $\{G, J, R_{\mathcal{K}}\}$ in (2.2), (2.3), so we shall call them a ∇_F-*generator* of R. To complete the above definition, one should specify for $\mathbf{C}^{n\times n}$ the inner product in which the decomposition (2.4) is orthogonal. Here we choose

$$< A, B >= \text{tr}(B^* \cdot A), \qquad A, B \in \mathbf{C}^{n\times n}, \tag{2.5}$$

where $\text{tr}(A)$ denotes the sum of all diagonal entries of A. Note that the latter inner product induces the Frobenius norm in $\mathbf{C}^{n\times n}$:

$$< A, A >= \text{tr}(A^* \cdot A) = \sum_{i=1}^{n}\sum_{j=1}^{n} |a_{ij}|^2 = \|A\|_F^2, \qquad (A = \begin{bmatrix} a_{ij} \end{bmatrix}_{1\le i,j\le n}).$$

When $\mathcal{K} = \text{Ker}\nabla_F = \{0\}$, all the information about the n^2 entries of R is efficiently stored in only $\alpha n + 2$ entries of $\{G, J\}$. However if $\mathcal{K} = \text{Ker}\nabla_F$ is nontrivial, then $R_{\mathcal{K}}$ is a full $n \times n$ matrix, and at first glance the representation of a partially reconstructible matrix R by its generator $\{G, J, R_{\mathcal{K}}\}$ is no longer efficient. As one can realize, however, for the main structured classes, $\mathcal{K} = \text{Ker}\nabla_F$ has low dimension, and moreover, the matrix $R_{\mathcal{K}}$ has a simple form (say, circulant, or diagonal matrix), which in turn can be described by a smaller number of parameters. In the next two subsections we shall specify these definitions for Toeplitz-like and for Cauchy-like partially reconstructible matrices.

2.3. Partially reconstructible Toeplitz-like matrices. Here we shall use

$$\nabla_{Z_1}(R) = R - Z_1 \cdot R \cdot Z_1^*, \qquad \text{where} \qquad Z_1 = \begin{bmatrix} 0 & \cdots & \cdots & 0 & 1 \\ 1 & 0 & & & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \tag{2.6}$$

is the lower circulant shift matrix. It is easy to check that the ∇_{Z_1}-displacement rank of an arbitrary Hermitian Toeplitz matrix does not exceed 2 :

$$\nabla_{Z_1}(T) = T - Z_1 \cdot T \cdot Z_1^* = \begin{bmatrix} 0 & t_1^* - t_{n-1} & \cdots & t_{n-1}^* - t_1 \\ t_1 - t_{n-1}^* & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ t_{n-1} - t_1^* & 0 & \cdots & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ t_1 - t_{n-1}^* & t_1 - t_{n-1}^* \\ \vdots & \vdots \\ t_{n-1} - t_1^* & t_{n-1} - t_1^* \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{2} & t_1^* - t_{n-1} & \cdots & t_{n-1}^* - t_1 \\ -\frac{1}{2} & t_1^* - t_{n-1} & \cdots & t_{n-1}^* - t_1 \end{bmatrix}. \tag{2.7}$$

By analogy with (2.7), any matrix with a low (not just ≤ 2) ∇_{Z_1}-displacement rank will be called a *Toeplitz-like* matrix. As is well known (see, e.g., [GO92]), the kernel of $\nabla_{Z_1}(\cdot)$ in (2.6) is the subspace of all circulants, i.e.,

$$\mathcal{C} = \{\text{Circ}(c) : c \in \mathbf{C}^n\} \subset \mathbf{C}^{n\times n} \qquad \text{where} \qquad \text{Circ}(c) = \begin{bmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ c_{n-2} & & c_1 & c_0 & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix}. \tag{2.8}$$

Given arbitrary $R \in \mathbf{C}^{n\times n}$, the next lemma and corollary show how to compute in $O(n^2)$ arithmetic operations the decomposition of R with respect to $\mathbf{C}^{n\times n} = \mathcal{C} \oplus \mathcal{C}^\perp$.

Lemma 2.1 *Let $\mathcal{C} \subset \mathbf{C}^{n\times n}$ denote the subspace of all circulant matrices, use an inner product in $\mathbf{C}^{n\times n}$ defined by (2.5), and let $R = \begin{bmatrix} r_{ij} \end{bmatrix} \in \mathbf{C}^{n\times n}$ be arbitrary. Then $R \perp \mathcal{C}$ if and only if for $k = 1, 2, ..., n$ we have $\sum_{i=1}^n r_{i+k-1,i} = 0$ (Here the indices are integers modulo n, i.e., if $k > n$, then $k := k - n$).*

Briefly, a matrix R is orthogonal to all circulants if and only if for each of its circulant diagonals, the sum of the entries is zero.

Proof. It is easy to check that $< Z_1^k, R >= \sum_{i=1}^n r_{i+k-1,i}$. Since $\{I, Z_1, Z_1^2, ..., Z_1^{n-1}\}$ form a basis in $\mathcal{C} \subset \mathbf{C}^{n\times n}$, the assertions follow.

□

Corollary 2.2 *With $\mathcal{C} \subset \mathbf{C}^{n\times n}$ as above, a matrix $R = \begin{bmatrix} r_{ij} \end{bmatrix}_{1\leq i,j\leq n} \in \mathbf{C}^{n\times n}$ is decomposed with respect to $\mathbf{C}^{n\times n} = \mathcal{C} \oplus \mathcal{C}^\perp$ as*

$$R = R_{\mathcal{C}} + R_{\mathcal{C}^\perp}, \tag{2.9}$$

with

$$R_{\mathcal{C}} = \text{Circ}(c_0, c_1, ..., c_{n-1}), \qquad \text{where} \qquad c_{i-1} = \frac{1}{n}\sum_{k=1}^n r_{i+k-1,k}, \tag{2.10}$$

and the indices are again integers modulo n.

Here we may remark that $R_{\mathcal{C}}$ in (2.9) is the optimal Frobenius-norm circulant approximant of R. Such approximants are often used as preconditioners in iterative methods. For example, when $R = \left[\begin{array}{c} t_{i-j} \end{array} \right]$ is a Toeplitz matrix, then (2.10) reduces to the well-known example [C88] :

$$c_i = \frac{(n-i) \cdot t_i + i \cdot t_{i-n}}{n} \qquad (i = 0, 1, ..., n-1). \tag{2.11}$$

2.4. Partially reconstructible Cauchy-like matrices. Here we choose the displacement operator

$$\nabla_{D_f}(R) = R - D_f \cdot R \cdot D_f^*, \qquad \text{where} \qquad D_f = \operatorname{diag}(f_1, f_2, ..., f_n). \tag{2.12}$$

A matrix with low ∇_{D_f}-displacement rank is called a *Cauchy-like* matrix, when R is Cauchy, the displacement rank is unity. For our purposes in the present paper the numbers f_i are assumed to be on the unit circle (the case of arbitrary f_i is studied, for example, in [HR84], [GO94a], [GKO95]).

It is a trivial exercise to check that the kernel of $\nabla_{D_f}(\cdot)$ in (2.12)) is the subspace

$$\mathcal{D} = \{\operatorname{diag}(f) : f \in \mathbf{C}^n\} \subset \mathbf{C}^{n \times n}$$

of all diagonal matrices[2]. Here are the counterparts of Lemma 2.1 and Corollary 2.2 for Cauchy-like matrices.

Lemma 2.3 *Let $\mathcal{D} \subset \mathbf{C}^{n\times n}$ be as above, and the inner product in $\mathbf{C}^{n\times n}$ is defined by* (2.5), *and let $R = \left[\begin{array}{c} r_{ij} \end{array} \right] \in \mathbf{C}^{n\times n}$ be arbitrary. Then $\quad R \perp \mathcal{D} \quad$ if and only if its main diagonal is zero.*

Corollary 2.4 *Let $\mathcal{D} \subset \mathbf{C}^{n\times n}$ stands for the subspace of all diagonal matrices, than a matrix $R = \left[\begin{array}{c} r_{ij} \end{array} \right] \in \mathbf{C}^{n\times n}$ is decomposed with respect to $\mathbf{C}^{n\times n} = \mathcal{D} \oplus \mathcal{D}^{\perp}$ as*

$$R = R_{\mathcal{D}} + R_{\mathcal{D}^\perp} \qquad \text{where} \qquad R_{\mathcal{D}} = \operatorname{diag}(r_{11}, r_{22}, ..., r_{nn}). \tag{2.13}$$

Clearly, $R_{\mathcal{D}}$ in (2.13) is the optimal Frobenius-norm diagonal approximant of R.

2.5. Transformation of partially reconstructible Toeplitz-like matrices into partially reconstructible Cauchy-like matrices. As mentioned in the introduction, Cauchy-like matrices allow the incorporation of pivoting into fast algorithms, so they are attractive from the numerical point of view. It turns out that Toeplitz-like, Hankel-like, Toeplitz-plus-Hankel-like, Vandermonde-like and Chebyshev-Vandermonde-like matrices can all be transformed into Cauchy-like matrices by applying the Fast Fourier, Cosine or Sine transforms to the columns of their generators, see, e.g., [GKO95], [KO95] for the details. In particular, the transformation to a Cauchy-like matrix allowed us to design in [GKO95] a new fast and accurate Toeplitz solver. However the symmetry of the matrix was lost in the process. In order to design (in Sections 4, 5) an *accurate* algorithm that preserves, and also exploits, the symmetry of the matrix, we describe below the transformation of a partially reconstructible Toeplitz-like matrix into a partially reconstructible Cauchy-like matrix.

To this end recall that the lower circulant shift matrix Z_1 is diagonalized by the (normalized) Discrete Fourier Transform matrix $\mathcal{F} = \frac{1}{\sqrt{n}} \left[\begin{array}{c} \omega^{(i-1)(j-1)} \end{array} \right]_{1 \le i,j, \le n}$, which is a unitary matrix. Moreover,

$$Z_1 = \mathcal{F}^* \cdot D_1 \cdot \mathcal{F}, \qquad \text{with} \qquad D_1 = \operatorname{diag}(1, w, w^2, ..., w^{n-1}), \tag{2.14}$$

where $\omega = e^{\frac{2\pi i}{n}}$ is the n-th primitive root of unity. With these notations the following statement holds.

[2]Partially reconstructible Cauchy-like matrices appear in certain tangential rational matrix interpolation problems, where the presence of nontrivial kernel of $\nabla_F(\cdot)$ means that the corresponding $\alpha \times \alpha$ rational matrix function shares poles and zeros at the same points; see, e.g., [GO94a].

Lemma 2.5 *Let* $Z_1, D_1 \in \mathbf{C}^{n\times n}$ *be as in* (2.14), *and let* $R = \left[\begin{array}{c} r_{ij} \end{array}\right] \in \mathbf{C}^{n\times n}$ *be a partially reconstructible Toeplitz-like matrix, given by a* ∇_{Z_1}*-generator* $\{G_{\nabla_{Z_1}}, J, \mathrm{Circ}(c)\}$, *i.e.,*

$$\nabla_{Z_1}(R) = R - Z_1 \cdot R \cdot Z_1^* = G_{\nabla_{Z_1}} \cdot J \cdot G^*_{\nabla_{Z_1}},$$

and $c_i = \frac{1}{n}\sum_{k=1}^{n} r_{i+k-1}$. *Then* $\mathcal{F} \cdot R \cdot \mathcal{F}^*$ *is a partially reconstructible Cauchy-like matrix, satisfying*

$$\nabla_{D_1}(\mathcal{F} \cdot R \cdot \mathcal{F}^*) = (\mathcal{F} \cdot R \cdot \mathcal{F}^*) - D_1 \cdot (\mathcal{F} \cdot R \cdot \mathcal{F}^*) \cdot D_1^* = G_{\nabla_{D_1}} \cdot J \cdot G^*_{\nabla_{D_1}},$$

with a ∇_{D_1}*-generator* $\{G_{\nabla_{D_1}}, J, \mathrm{diag}(d)\}$, *where*

$$G_{\nabla_{D_1}} = \mathcal{F} \cdot G_{\nabla_{Z_1}}, \qquad \text{and} \qquad d = \sqrt{n} \cdot \mathcal{F} \cdot c. \tag{2.15}$$

Proof. The first equality in (2.15) follows immediately by comparing

$$R - Z_1 \cdot R \cdot Z_1^* = G_{\nabla_{Z_1}} \cdot J \cdot G^*_{\nabla_{Z_1}}, \qquad\qquad R - D_1 \cdot R \cdot D_1^* = G_{\nabla_{D_1}} \cdot J \cdot G^*_{\nabla_{D_1}},$$

and (2.14). To prove the second equality in (2.15), let

$$R = \mathrm{Circ}(c) + R_{\mathcal{C}^\perp}$$

be the decomposition (2.9) of R. As is well known, any circulant matrix is diagonalized by the Discrete Fourier matrix, and moreover, $\mathrm{diag}(d) = \mathcal{F} \cdot \mathrm{Circ}(c) \cdot \mathcal{F}^*$, see, e.g. , [D79]. Thus

$$\mathcal{F} \cdot R \cdot \mathcal{F}^* = \mathrm{diag}(d) + \mathcal{F} \cdot R_{\mathcal{C}^\perp} \cdot \mathcal{F}^*. \tag{2.16}$$

Since any unitary similarity preserves the inner product in (2.5), the equality (2.16) can be recognized as the decomposition (2.13) of the Cauchy-like matrix $\mathcal{F} \cdot R \cdot \mathcal{F}^*$, and the assertions of the Lemma follow.

$\square$

2.6. Implementation details. Let R be a Toeplitz-like matrix given by its ∇_{Z_1}-generator $\{G_{\nabla_{Z_1}}, J, \mathrm{Circ}(c)\}$. Lemma 2.5 allows us to efficiently compute the ∇_{D_1}-generator $\{G_{\nabla_{D_1}}, J, \mathrm{diag}(d)\}$ of the Cauchy-like matrix $\mathcal{F} \cdot R \cdot \mathcal{F}^*$, i.e., one avoids the operations on the entries of matrices, and manipulates only their generators:

- The computation of $G_{\nabla_{D_1}}$ requires applying α FFT's of order n to the columns of the generator $G_{\nabla_{Z_1}}$ of R, see e.g., the first equality in (2.15).
- The computation of d in (2.15) in fact requires one FFT of order only $\frac{n}{2}$, thanks to the property stated next.

Corollary 2.6 *Let* $R \in \mathbf{C}^{n\times n}$ *be any Hermitian matrix, and the entries of* $c = \left[\begin{array}{cccc} c_0 & c_1 & \cdots & c_{n-1} \end{array}\right]^T$ *be given by* (2.10). *Then* c *is a conjugate-even vector, i.e.,*

$$c = Q \cdot \bar{c}, \qquad \text{where} \qquad Q = \left[\begin{array}{c|cccc} 1 & 0 & \cdots & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 & 1 \\ \vdots & \vdots & \iddots & \iddots & 0 \\ \vdots & 0 & 1 & \iddots & \vdots \\ 0 & 1 & 0 & \cdots & 0 \end{array}\right], \tag{2.17}$$

and the bar denotes the operation of componentwise conjugation.

It is well-known that the FFT of a conjugate-even vector of the order n can be performed by the application of the FFT of order $\frac{n}{2}$. Moreover, being the Discrete Fourier transform of a conjugate-even vector, d in (2.15) is a real vector, which is not surprising given the fact that the components of d are the diagonal entries of the Hermitian matrix $\mathcal{F} \cdot R \cdot \mathcal{F}^*$.

Therefore, the transformation of a partially reconstructible Toeplitz-like matrix into a partially reconstructible Cauchy-like matrix can be computed in $(\alpha + \frac{1}{2})$ FFT's, where α is the corresponding displacement rank.

2.7. Further simplifications for Toeplitz matrices. We conclude this section with several remarks showing that when R is Toeplitz, its transformation into a Cauchy-like matrix can be achieved more cheaply than above. Indeed, from (2.7) and (2.11) it follows that for a Hermitian Toeplitz matrix T its ∇_{Z_1}-generator is given by $\{G, J, T_\mathcal{C}\}$, where

$$\{G = \begin{bmatrix} r + \frac{1}{2}e_0 & r - \frac{1}{2}e_0 \end{bmatrix}, \qquad J = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad T_\mathcal{C} = \mathrm{Circ}(c)\}, \tag{2.18}$$

where $r = \begin{bmatrix} 0, & t_1 - t_{n-1}^*, & t_2 - r_{n-2}^*, & \cdots, & t_{n-2} - t_2^*, & t_{n-1} - t_1^* \end{bmatrix}^T$, e_0 is the first unit vector, and the entries of the vector $c \in \mathbf{C}^n$ are given by (2.11).

The Hermitian Toeplitz structure implies the following property, useful for efficient transformation of T into a Cauchy-like matrix.

Corollary 2.7 *Let $T \in \mathbf{C}^{n\times n}$ be a Hermitian Toeplitz matrix, and a triple of matrices in* (2.18) *be its generator. Then r will be a conjugate-odd vector, i.e.,*

$$r = -Q \cdot \bar{r}, \tag{2.19}$$

with Q defined in (2.17).

Thus the first generator matrix $G_{\nabla_{D_1}}$ of the Cauchy-like matrix $\mathcal{F}T\mathcal{F}^*$ can be computed by applying only one FFT to the vector r in (2.18). In accordance with Corollary 2.7, r is a conjugate-odd vector, and hence the computation of $\mathcal{F}r$ can be done via only one FFT of order $\frac{n}{2}$. Also the conjugate-oddness of r implies that the entries of $is := \mathcal{F}r$ are purely imaginary numbers. Hence the first equality in (2.15) reduces to

$$G_{\nabla_{D_1}} = \begin{bmatrix} t + is & -t + is \end{bmatrix}, \tag{2.20}$$

where $t = \frac{1}{2\sqrt{n}} \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$ is a real vector. Thus one sees that the entries of the second column of $G_{\nabla_{D_1}}$ are obtained from the entries of its first column by conjugating and changing the sign. As we shall see in section 3, this property of $G_{\nabla_{D_1}}$ is inherited under Schur complementation, thus allowing a more efficient implementation of the corresponding generalized Schur algorithm.

Finally, the conclusion is that the transformation of a Hermitian Toeplitz matrix into a Hermitian Cauchy-like matrix is reduced to performing of two FFT's of order $\frac{n}{2}$.

3 Continuous-time Lyapunov displacement operator.

It has been just shown how partially reconstructible matrices can be defined via the discrete-time Lyapunov displacement operators of the form (2.1), and moreover, how partially reconstructible Toeplitz-like matrices can be transformed into partially reconstructible Cauchy-like matrices, while preserving Hermitian structure. Below we equivalently define the same classes of partially reconstructible matrices using an alternative so-called *continuous-time Lyapunov displacement operators.*

The goal of this section is to state a counterpart of Lemma 2.5, namely to provide a recipe for transforming a partially reconstructible Toeplitz-like matrix, defined via discrete-time Lyapunov displacement operator, into a partially reconstructible Cauchy-like matrix, defined via continuous-time Lyapunov displacement operator.

Both the transformations in Lemma 2.5 and those proposed here will be used below to devise a variety of different algorithms for solving linear systems with Hermitian partially reconstructible Toeplitz-like matrices. Moreover to devise such algorithms, it will be more convenient to exploit an alternative (to (2.1)) form of the displacement operator; this form is considered below.

3.1. Continuous-time Lyapunov displacement operator. In this subsection we consider an alternative continuous-time Lyapunov displacement operator of the form

$$\triangle_A(R) = A \cdot R + R \cdot A^*, \tag{3.1}$$

(the names *Hankel-like*, or *symmetric Sylvester* displacement operator are also in use). Although $\triangle_A(\cdot)$ differs from $\nabla_F(\cdot)$ in (2.1), both displacement operators can be used for introducing *the same* types (i.e., Toeplitz-like, Cauchy-like, etc.) of displacement structure. This is seen, for example, from the following two simple lemmas, which allow us to change the form of the displacement equation from the discrete-time form,

$$\nabla_F(R) = R - F \cdot R \cdot F^* = G_{\nabla_F} \cdot J \cdot G^*_{\nabla_F} \tag{3.2}$$

to the continuous-time form,

$$\triangle_A(R) = A \cdot R + R \cdot A^* = G_{\triangle_A} \cdot J \cdot G^*_{\triangle_A}, \tag{3.3}$$

and vice versa, while keeping the same solution R.

Lemma 3.1 *Let $F \in \mathbf{C}^{n \times n}$ be arbitrary, and set*

$$A = (\tau I + F) \cdot (\tau I - F)^{-1}, \tag{3.4}$$

where τ is any number such that $|\tau| = 1$ is chosen to guarantee the invertibility of the second factor in (3.4). *Let the displacement operators ∇_F and $\triangle_A$ be defined by* (3.2) *and* (3.3), *respectively. Then*

(i) $\mathrm{Ker}\nabla_F = \mathrm{Ker}\triangle_A$.

(ii) *For any $R \in \mathbf{C}^{n\times n}$, specified by the ∇_F-generator $\{G_{\nabla_F}, J, R_{\mathcal{K}}\}$, its $\triangle_A$-generator is $\{G_{\triangle_A}, J, R_{\mathcal{K}}\}$, where*

$$G_{\triangle_A} = \sqrt{2} \cdot (\tau I - F)^{-1} G_{\nabla_F}. \tag{3.5}$$

In particular, the ∇_F-displacement rank and the $\triangle_A$-displacement rank of R are the same.

Proof. Both assertions follow from the identity

$$2 \cdot (\tau I - F)^{-1} \cdot (R - F \cdot R \cdot F^*) \cdot (\tau^* I - F^*)^{-1} =$$

$$= (\tau I - F)^{-1} \cdot (\tau I + F) \cdot R + R \cdot (\tau^* I + F^*) \cdot (\tau^* I - F^*)^{-1}.$$

□

This Lemma originates in the Möbius transformations between the unit disk and the right half plane, see, e.g., [K80], p.180. The above Lemma shows that the classes of partially reconstructible

Toeplitz-like and Cauchy-like matrices can be equivalently introduced via the continuous-time Lyapunov displacement operators (3.1), via a circulant matrix,

$$A = \frac{2\tau}{\tau^n - 1} \cdot \mathrm{Circ}(\frac{\tau^n + 1}{2\tau}, \tau^{n-2}, \tau^{n-3}, ..., \tau, 1),$$

and a diagonal matrix

$$A = \mathrm{diag}(\frac{\tau + f_1}{\tau - f_1}, ..., \frac{\tau + f_n}{\tau - f_n}), \tag{3.6}$$

respectively.

Recall that the subspace $\mathcal{D} \subset \mathbf{C}^{n\times n}$ of all diagonal matrices is the kernel of the discrete-time Lyapunov displacement operator ∇_F in (2.12). By Lemma 3.1,the same $\mathcal{D}$ is the kernel of the continuous-time Lyapunov displacement operator $\triangle_A$ in (3.1) with A as in (3.6). Note that this can easily be seen directly from the fact that $\frac{\tau+f_i}{\tau-f_i}$ in (3.6) are purely imaginary numbers.

The next lemma is the converse to Lemma 3.1, and it can be deduced by exactly the same arguments.

Lemma 3.2 *Let $A \in \mathbf{C}^{n\times n}$ be arbitrary, and set*

$$F = (tI + A)^{-1} \cdot (tI - A), \tag{3.7}$$

where $t > 0$ is any number that will guarantee the invertibility of the second factor in (3.7)*. Let the displacement operators ∇_F and $\triangle_A$ be defined by* (2.1) *and* (3.1)*, resp. Then*

(i) $\mathrm{Ker}\nabla_F = \mathrm{Ker}\triangle_A$.

(ii) *Any $R \in \mathbf{C}^{n\times n}$ given by the $\triangle_A$-generator $\{G_{\triangle_A}, J, R_{\mathcal{K}}\}$ has a ∇_F-generator $\{G_{\nabla_F}, J, R_{\mathcal{K}}\}$, where*

$$G_{\nabla_F} = \sqrt{2t} \cdot (tI + A)^{-1} \cdot G_{\triangle_A} \tag{3.8}$$

In particular, the ∇_F-displacement rank and the $\triangle_A$-displacement rank of R are the same.

Proof. Both assertions follow from the easily checked identity

$$2 \cdot t \cdot (tI + A)^{-1} \cdot (A \cdot R + R \cdot A^*) \cdot (tI + A^*)^{-1} =$$

$$= R - (tI + A)^{-1} \cdot (tI - A) \cdot R \cdot (tI - A^*) \cdot (tI + A^*)^{-1}.$$

□

We conclude this section with a recipe for transforming a partially reconstructible Toeplitz-like matrix, defined via a discrete-time Lyapunov displacement operator, into a partially reconstructable Cauchy-like matrix defined via a continuous-time Lyapunov displacement operator.

Corollary 3.3 *Let $Z_1, D_1 \in \mathbf{C}^{n\times n}$ be defined as in* (2.14)*, and let $R \in \mathbf{C}^{n\times n}$ be a Toeplitz-like matrix, satisfying*

$$\nabla_{Z_1}(R) = R - Z_1 \cdot R \cdot Z_1^* = G_{\nabla_{Z_1}} \cdot J \cdot G^*_{\nabla_{Z_1}}, \tag{3.9}$$

so that it is given by a ∇_{Z_1}-generator $\{G_{\nabla_{Z_1}}, J, \mathrm{Circ}(c)\}$, and

$$A_1 = \mathrm{diag}(\frac{\tau+1}{\tau-1}, \frac{\tau+w}{\tau-w}, \frac{\tau+w^2}{\tau-w^2}, ..., \frac{\tau+w^{n-1}}{\tau-w^{n-1}}), \qquad \text{where} \qquad w = e^{\frac{2\pi i}{n}},$$

where τ is any number with $|\tau| = 1$ so that $\tau^n \neq 1$. Then $\mathcal{F} \cdot R \cdot \mathcal{F}^*$ is a Cauchy-like matrix, satisfying, , satisfying

$$\triangle_{A_1}(\mathcal{F} \cdot R \cdot \mathcal{F}^*) = A_1 \cdot (\mathcal{F} \cdot R \cdot \mathcal{F}^*) + (\mathcal{F} \cdot R \cdot \mathcal{F}^*) \cdot A_1^* = G_{\nabla_{A_1}} \cdot J \cdot G_{\nabla_{A_1}}^*, \tag{3.10}$$

with a $\triangle_{A_1}$-generator $\{G_{\triangle_{A_1}}, J, \mathrm{diag}(d)\}$, where

$$G_{\triangle_{A_1}} = \sqrt{2} \cdot (\tau I - D_1) \cdot \mathcal{F} \cdot G_{\nabla_{Z_1}}, \qquad \text{and} \qquad d = \sqrt{n} \cdot \mathcal{F} \cdot c. \tag{3.11}$$

Thus, the transformation of a partially reconstructable Toeplitz-like matrix, defined via discrete-time Lyapunov displacement equation (3.9) into a partially reconstructable Cauchy-like matrix, defined via continuous-time Lyapunov displacement equation (3.10), requires performing two FFT's of order $\frac{n}{2}$ (cf. subsection 2.7).

4 Symmetric block Gaussian elimination for matrices with displacement structure

4.1. A general scheme for generalized Schur algorithms for partially reconstructable matrices. Symmetric block Gaussian elimination is based on recursively apply the well-known Schur complementation formula

$$R_1 = \begin{bmatrix} R_{11} & R_{21}^* \\ R_{21} & R_{22} \end{bmatrix} = \begin{bmatrix} I & 0 \\ R_{21} \cdot R_{11}^{-1} & I \end{bmatrix} \cdot \begin{bmatrix} R_{11} & 0 \\ 0 & R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^* \end{bmatrix} \cdot \begin{bmatrix} I & R_{11}^{-1} \cdot R_{21}^* \\ 0 & I \end{bmatrix}. \tag{4.1}$$

As was mentioned in the introduction, in the symmetric case, numerical stability cannot be achieved using only scalar elimination steps, and sometimes one has to perform the step (4.1) with an 2×2 block R_{11}. The choice of the size m of the pivot R_{11} depends upon the particular pivoting strategy; this issue will be addressed in Section 4 below.

When the kernel of a displacement operator is trivial, then it is well-known that the displacement structure of a matrix is inherited by its Schur complements. This fact allows one to avoid directly computing the $(n-m)^2$ entries of the Schur complement $R_2 = R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^*$ in (4.1), and to compute instead only the $O(\alpha(n-m))$ entries of its generator. Algorithms of this type have been called generalized Schur algorithms, because the classical Schur algorithm [S17] belongs to the class (see [KS95]). In this section we allow the displacement equation to have a nontrivial kernel, and show that also in this case Schur complements inherit a displacement structure. We further extend the generalized Schur algorithms to partially reconstructable matrices defined via discrete-time (section 4.2), and continuous-time (section 4.1) Lyapunov displacement operators. Before doing so, let us give a general scheme for devising such algorithms.

A general scheme for generalized Schur algorithms for partially reconstructable matrices, defined via either discrete-time or continuous-time Lyapunov displacement operators

Input : A generator $\{G_1, J, R_{\mathcal{K}_1}\}$ of $R_1 \in \mathbf{C}^{n \times n}$ in (4.1), where $G_1 \in \mathbf{C}^{n \times \alpha}$.

Output : The block LDL^* decomposition of R_1.

Steps : **1.** If $n = 0$ then stop.

2. Chose size m for the pivot $R_{11} \in \mathbf{C}^{m \times m}$.

3. Recover from $\{G_1, J, R_{\mathcal{K}_1}\}$ the entries of the first m columns $\begin{bmatrix} R_{11} \\ R_{21} \end{bmatrix}$ of R_1.

4. Write down the first $m \times m$ block diagonal entry R_{11} of the matrix D and the first m columns $\left[\begin{array}{c} I \\ R_{21} \cdot R_{11}^{-1} \end{array} \right]$ of the matrix L in the LDL^* decomposition of R_1.

5. Compute a generator $\{G_2, J, R_{\mathcal{K}_2}\}$ of the Schur complement R_2 :

5.1. Compute $G_2 \in \mathbf{C}^{(n-m)\times\alpha}$.

5.2. Compute $R_{\mathcal{K}_2} \in \mathbf{C}^{(n-m)\times(n-m)}$.

6. Set $n := n-m$, and repeat the steps 1 - 6 for the Schur complement R_2, given by its generator $\{G_2, J, R_{\mathcal{K}_2}\}$.

In order to implement the above general scheme for a particular displacement structure, one has to specify the computations in the steps 2, 3, 5.1 and 5.2, which depend upon the form of displacement equation. In the next two subsections these steps are specified for continuous-time Lyapunov and discrete-time Lyapunov displacement operators.

4.2. Generalized Schur algorithms for partially reconstructable matrices, defined via continuous-time Lyapunov displacement equation. The following lemma from [GO94a], [GKO95] is the basis for generalized Schur algorithms corresponding to partially reconstructable matrices, defined via continuous-time Lyapunov displacement operators.

Lemma 4.1 *Let $A \in \mathbf{C}^{n\times n}$ be a block lower triangular matrix, $R_1 \in \mathbf{C}^{n\times n}$ be a Hermitian matrix, satisfying a continuous-time Lyapunov displacement equation of the form*

$$\triangle_A(R) = A \cdot R_1 + R_1 \cdot A^* = G_1 \cdot J \cdot G_1^*, \tag{4.2}$$

with some $G_1 \in \mathbf{C}^{n\times\alpha}$, and some Hermitian matrix $J \in \mathbf{C}^{\alpha\times\alpha}$. Let the matrices in (4.2) *be partitioned as*

$$A = \left[\begin{array}{cc} A_{11} & 0 \\ A_{21} & A_{22} \end{array} \right], \qquad R_1 = \left[\begin{array}{cc} R_{11} & R_{21}^* \\ R_{21} & R_{22} \end{array} \right], \qquad G_1 = \left[\begin{array}{c} G_{11} \\ G_{21} \end{array} \right].$$

If the upper left block $R_{11} \in \mathbf{C}^{m\times m}$ of R_1 is invertible, then the Schur complement $R_2 = R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^* \in \mathbf{C}^{(n-m)\times(n-m)}$ satisfies

$$A_{22} \cdot R_2 + R_2 \cdot A_{22} = G_2 \cdot J \cdot G_2^*, \tag{4.3}$$

with

$$G_2 = G_{21} - R_{21} \cdot R_{11}^{-1} \cdot G_{11}. \tag{4.4}$$

Proof. Substituting (4.1) into (4.2), and then multiplying by $\left[\begin{array}{cc} I & 0 \\ -R_{21} \cdot R_{11}^{-1} & I \end{array} \right]$ from the left, and by $\left[\begin{array}{cc} I & -R_{11}^{-1} \cdot R_{21}^* \\ 0 & I \end{array} \right]$ from the right, we obtain

$$\left[\begin{array}{cc} A_{11} & 0 \\ ? & A_{22} \end{array} \right] \cdot \left[\begin{array}{cc} R_{11} & 0 \\ 0 & R_2 \end{array} \right] + \left[\begin{array}{cc} R_{11} & 0 \\ 0 & R_2 \end{array} \right] \cdot \left[\begin{array}{cc} A_{11}^* & ? \\ 0 & A_{22}^* \end{array} \right] =$$

$$\left[\begin{array}{c} G_{11} \\ G_{21} - R_{21} \cdot R_{11}^{-1} \end{array} \right] \cdot J \cdot \left[\begin{array}{c} G_{11} \\ G_{21} - R_{21} \cdot R_{11}^{-1} \end{array} \right]^*,$$

where ? denotes an irrelevant entry. Formula (4.4) then follows by equating the lower right block entries.

$\square$

4.3. Generalized Schur algorithm for partially reconstructable Cauchy-like matrices, defined via continuous-time Lyapunov displacement equation. As was mentioned above, in order to devise a generalized Schur algorithm for a particular displacement structure, one has to specify the steps 2, 3, 5.1 and 5.2 in the general scheme of section 4.1. Here we do so for the continuous-time Lyapunov displacement equation with diagonal matrix A, namely

$$\triangle_{\mathcal{D}_a}(R_1) = D_a \cdot R_1 + R_1 \cdot D_a^* = G_1 \cdot J \cdot G_1^* \qquad \text{with} \qquad D_a = \operatorname{diag}(a_1, a_2, ..., a_n), \tag{4.5}$$

where for our purposes in the present paper we restrict ourself to purely imaginary a_i.

It is easy to verify to check that $\operatorname{Ker}\triangle_{D_a}$ is the subspace $\mathcal{D} \in \mathbf{C}^{n\times n}$ of all diagonal matrices. Furthermore, by Corollary 2.4, matrix $R_{\mathcal{D}}$ in (2.13) is simply the diagonal part of R_1, which we shall denote by $R_{\mathcal{D}} = \operatorname{diag}(d_1^{(1)}, d_2^{(1)}, ..., d_n^{(1)})$.

<u>Continuous-Cauchy algorithm</u>

Let $R_1 \in \mathbf{C}^{n\times n}$ be a Cauchy-like matrix given by its $\triangle_{D_a}$-generator $\{G_1, J, \operatorname{diag}(d_1)\}$. Then the steps 2, 3, 5.1 and 5.2 above can be implemented as follows.

2. The size m will be chosen to be 1 or 2, to enhance the stability of the algorithm, we postpone the discussion till section 4.

3. From (4.5) it is easily seen that the nondiagonal entries of the first m columns of $R_1 = \left[\begin{array}{c} r_{ij} \end{array}\right]$ are given by the formula

$$r_{ij} = \frac{g_i \cdot J \cdot g_i^*}{a_i + a_j^*}, \tag{4.6}$$

where g_i is the i-th row of G_1. The diagonal entries of R_1 are simply stored in $\operatorname{diag}(d_1)$.

5.1. The matrix G_2 is computed by the formula (4.4).

5.2. The diagonal part $R_{\mathcal{K}_2} = \operatorname{diag}(d_2^{(2)}, d_3^{(2)}, ..., d_n^{(2)})$ of the Schur complement $R_2 = R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^*$ is obtained by computing <u>only</u> the diagonal entries in $\operatorname{diag}(d_1) - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^*$, namely :

- If $m = 1$ then $d_k^{(2)} = d_k^{(1)} - r_{k1}\frac{1}{r_{11}}r_{k1}^*$.
- If $m = 2$ then $d_k^{(2)} = d_k^{(1)} - \left[\begin{array}{cc} r_{k1} & r_{k2} \end{array}\right] \cdot \left[\begin{array}{cc} r_{11} & r_{21}^* \\ r_{21} & r_{22} \end{array}\right]^{-1} \cdot \left[\begin{array}{c} r_{k1}^* \\ r_{k2}^* \end{array}\right]$.

□

The complexity of computing the factorization $PR_1P^T = LDL^*$ is $4\alpha n^2$ arithmetic operations, or $4\alpha n$ parallel operations with n processors.

If $R = \mathcal{F} \cdot T \cdot \mathcal{F}^*$ is a partially reconstructable Cauchy-like matrix, transformed from a Toeplitz matrix, then the complexity of its triangular factorization by the continuous-Cauchy algorithm is $12n^2$ operations.

4.4. Generalized Schur algorithms for partially reconstructable matrices defined via a discrete-time Lyapunov displacement equation. Here we use the results of the previous subsection to derive an algorithm for the discrete-time Lyapunov displacement operator. Here is the counterpart of Lemma 4.1.

Lemma 4.2 *Let $F \in \mathbf{C}^{n\times n}$ be a block lower triangular matrix, $R_1 \in \mathbf{C}^{n\times n}$ be a Hermitian matrix, satisfying a discrete-time Lyapunov displacement equation of the form*

$$\nabla_F(R) = R_1 - F \cdot R_1 \cdot F^* = G_1 \cdot J \cdot G_1^*, \tag{4.7}$$

for some $G_1 \in \mathbf{C}^{n\times\alpha}$, and some Hermitian matrix $J \in \mathbf{C}^{\alpha\times\alpha}$. Let the matrices in (4.7) *be partitioned as*

$$F = \left[\begin{array}{cc} F_{11} & 0 \\ F_{21} & F_{22} \end{array}\right], \qquad R_1 = \left[\begin{array}{cc} R_{11} & R_{21}^* \\ R_{21} & R_{22} \end{array}\right], \qquad G_1 = \left[\begin{array}{c} G_{11} \\ G_{21} \end{array}\right].$$

If the upper left block $R_{11} \in \mathbf{C}^{m\times m}$ of R_1 is invertible, then the Schur complement $R_2 = R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^* \in \mathbf{C}^{(n-m)\times(n-m)}$ satisfies

$$R_2 - F_{22} \cdot R_2 \cdot F_{22}^* = G_2 \cdot J \cdot G_2^*, \tag{4.8}$$

with

$$G_2 = G_{21} - ((\tau I - F_{22}) \cdot R_{21} \cdot R_{11}^{-1} - F_{21}) \cdot (\tau I - F_{11})^{-1} \cdot G_{11}, \tag{4.9}$$

where τ is any number on the unit circle, which is not an eigenvalue of F_1.

Proof. A relatively quick method to prove (4.9) is to use the connection between discrete-time and continuous-time Lyapunov displacement operators, described by Lemma 3.1. More precisely, let us convert (4.7) into (4.2), then perform one step of the generator recursion (4.4), and finally convert (4.3) back to (4.8). To this end let us rewrite (4.7) as

$$\nabla_{F_1}(R_1) = R_1 - F_1 \cdot R_1 \cdot F_1^* = G_{\nabla_{F_1}} \cdot J \cdot G_{\nabla_{F_1}}^*, \tag{4.10}$$

and set $A_1 = (\tau I + F_1) \cdot (\tau I - F_1)^{-1}$, where τ is some number on the unit circle, to guarantee the invertibility of the second factor in the latter expression. Clearly, A_1 is a lower triangular matrix :

$$A_1 = \begin{bmatrix} A_{11} & 0 \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} (\tau I + F_{11}) \cdot (\tau I - F_{11})^{-1} & 0 \\ ? & (\tau I + F_{22}) \cdot (\tau I - F_{22})^{-1} \end{bmatrix}, \tag{4.11}$$

where by ? we denote an irrelevant entry. Then by Lemma 3.1

$$\triangle_{A_1}(R_1) = A_1 \cdot R_1 + R_1 \cdot A_1^* = G_{\triangle_{A_1}} \cdot J \cdot G_{\triangle_{A_1}}^*, \tag{4.12}$$

where the matrices $G_{\nabla_{F_1}}$ in (4.10) and $G_{\triangle_{A_1}}$ in (4.12) are related by

$$G_{\nabla_{F_1}} = \frac{1}{\sqrt{2}} \cdot (\tau I - F_1) \cdot G_{\triangle_{A_1}},$$

or in the partitioned form

$$\begin{bmatrix} G_{\nabla_{F_1,11}} \\ G_{\nabla_{F_1,21}} \end{bmatrix} = \frac{1}{\sqrt{2}} \cdot \begin{bmatrix} (\tau I - F_{11}) \cdot G_{\triangle_{A_1,11}} \\ (\tau I - F_{22}) \cdot G_{\triangle_{A_1,21}} - F_{21} \cdot G_{\triangle_{A_1,11}} \end{bmatrix}. \tag{4.13}$$

Then by Lemma 4.1 the Schur complement R_2 satisfies

$$\triangle_{A_{22}}(R_2) = A_{22} \cdot R_2 + R_2 \cdot A_{22}^* = G_{\triangle_{A_2}} \cdot J \cdot G_{\triangle_{A_2}}^*,$$

with

$$G_{\triangle_{A_2}} = G_{\triangle_{A_1,21}} - R_{21} \cdot R_{11}^{-1} \cdot G_{\triangle_{A_1,11}}. \tag{4.14}$$

Comparing the form of A_{22} in (4.11) with (3.4), one sees that

$$\nabla_{F_{22}}(R_2) = R_2 - F_{22} \cdot R_2 \cdot F_{22}^* = G_{\nabla_{F_2}} \cdot J \cdot G_{\nabla_{F_2}}^*,$$

with

$$G_{\nabla_{F_2}} = \frac{1}{\sqrt{2}} \cdot (\tau I - F_{22}) \cdot G_{\triangle_{A_2}}. \tag{4.15}$$

Hence multiplying (4.14) by $\frac{1}{\sqrt{2}} \cdot (\tau I - F_{22})$ from the left, and using (4.13), one obtains (4.9). The Lemma is now proved.

□

Formula (4.9) is a generalization of the result in [KS95b], where it appeared for the case of invertible displacement operators $\nabla_F(\cdot)$.

4.5. Generalized Schur algorithm for partially reconstructable Cauchy-like matrices, defined via a discrete-time Lyapunov displacement equation. Here are the specializations of the steps 2, 3, 5.1 and 5.2 of the general scheme in section 4.1 for the case of a discrete-time Lyapunov displacement equation with diagonal matrix F, i.e.,

$$\nabla_{\mathcal{D}_f}(R_1) = R_1 - D_f \cdot R_1 \cdot D_f^* = G_1 \cdot J \cdot G_1^* \qquad \text{with} \qquad D_f = \text{diag}(f_1, f_2, ..., f_n), \tag{4.16}$$

where for our purposes in the present paper we restrict ourself to unit magnitude f_i.

<u>Discrete-Cauchy algorithm</u>

Let $R_1 \in \mathbf{C}^{n\times n}$ be a Cauchy-like matrix given by its ∇_{D_f}-generator $\{G_1, J, \text{diag}(d_1)\}$. Then the steps 2, 3, 5.1 and 5.2 of the generalized Schur algorithm can be implemented as follows.

2. The size m will be chosen to be 1 or 2, to enhance the stability of the algorithm, we postpone the discussion till section 5.

3. From (4.16) it is easily seen that the nondiagonal entries of the first m columns of $R_1 = \left[\begin{array}{c} r_{ij} \end{array}\right]$ are given by the formula

$$r_{ij} = \frac{g_i \cdot J \cdot g_j^*}{1 - f_i \cdot f_j^*}, \tag{4.17}$$

where g_i is the i-th row of G_1. The diagonal entries of R_1 are simply stored in $\text{diag}(d_1)$.

5.1. The matrix G_2 is to be computed by formula (4.9), which is specified here to

$$G_2 = G_{21} - \text{diag}(\tau - f_{m+1}, ..., \tau - f_n) \cdot R_{21} \cdot R_{11}^{-1} \cdot \text{diag}(\frac{1}{\tau - f_1}, ..., \frac{1}{\tau - f_m}) \cdot G_{11}, \tag{4.18}$$

5.2. The diagonal part $R_{\mathcal{K}_2} = \text{diag}(d_2)$ of the Schur complement $R_2 = R_{22} - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^*$ is obtained by computing <u>only</u> the diagonal entries in $\text{diag}(d_1) - R_{21} \cdot R_{11}^{-1} \cdot R_{21}^*$, namely :

- If $m = 1$ then $d_k^{(2)} = d_k^{(1)} - r_{k1}\frac{1}{r_{11}}r_{k1}^*$.
- If $m = 2$ then $d_k^{(2)} = d_k^{(1)} - \left[\begin{array}{cc} r_{k1} & r_{k2} \end{array}\right] \cdot \left[\begin{array}{cc} r_{11} & r_{21}^* \\ r_{21} & r_{22} \end{array}\right]^{-1} \cdot \left[\begin{array}{c} r_{k1}^* \\ r_{k2}^* \end{array}\right]$.

□

The complexity of the above algorithm is $6\alpha n^2$ operations, or $6\alpha n$ parallel operations with n processors.

4.6. Further simplifications for Toeplitz matrices. Let $R = \mathcal{F} \cdot T \cdot \mathcal{F}^*$ be a partially reconstructable Cauchy-like matrix, transformed from a Toeplitz matrix. Then by (2.20) its $\nabla_{\mathcal{D}_f}$-generator $\{G_1, J, \text{diag}(d_1)\}$ has a nice pattern :

$$G_1 = \left[\begin{array}{cc} g & -\bar{g} \end{array}\right], \qquad g \in \mathbf{C}^n.$$

A straightforward computation by (4.18) shows that since $|\tau| = |f_1| = ... = |f_n| = 1$, this pattern is inherited by the generator G_2 of the Schur complement. This means that it is sufficient to apply (4.18) only to compute the first column of G_2, its second column is then obtained for free. Therefore if the discrete-Cauchy algorithm is used as part of a Hermitian Toeplitz solver, the complexity of $12n^2$ operations for triangular factorization of $R = \mathcal{F} \cdot T \cdot \mathcal{F}^*$ is exactly the same as if the continuous-Cauchy algorithm would be employed.

5 Symmetric and Bunch-Kaufman pivoting for Cauchy-like matrices

5.1. Diagonal pivoting with partially reconstructable Cauchy-like matrices. A single step (4.1) of standard symmetric Gaussian elimination is valid if the upper left block R_{11} is invertible. Now, to enhance the accuracy of computations, it is recommended to precede (4.1) with symmetric row and column permutations (diagonal pivoting)

$$R_1 \leftarrow P \cdot R_1 \cdot P^*. \tag{5.1}$$

Assume now that R_1 is a partially reconstructable Cauchy-like matrix, given by its generator $\{G_1, J, \mathrm{diag}(d_1)\}$. Here we do not specify a particular form (i.e., discrete-time, or continuous-time) of displacement operator. Since both the continuous-Cauchy and discrete-Cauchy algorithms only work on the entries of the generators, to incorporate diagonal pivoting we have to express (5.1) in terms of the appropriate manipulations on the generator $\{G_1, J, \mathrm{diag}(d_1)\}$ of R_1. Multiplying the corresponding displacement equation by a permutation P, one immediately sees that (5.1) is equivalent to the following updating

$$F \leftarrow P \cdot F \quad (\text{or} \quad A \leftarrow P \cdot A), \qquad G_1 \leftarrow P \cdot G_1, \qquad d_1 \leftarrow P \cdot d_1. \tag{5.2}$$

If the permutation P is known, then (5.2) can be immediately incorporated into both continuous-Cauchy and discrete-Cauchy algorithms. However, in the known pivoting schemes one chooses the permutation P by analyzing the entries of R_1, which are not available in these fast algorithms. Therefore in order to incorporate a particular pivoting scheme, one has to recover by (4.6) or (4.17) the necessary entries of R_1 from its generator. The key question here is how many entries of R_1 have to be examined to decide on the permutation. For example, in the Bunch-Parlett scheme [BP71] one has to scan all $n(n+1)/2$ entries of the $n \times n$ symmetric matrix, so the overall complexity of the algorithm would jump to $O(n^3)$ operations. Below we describe two schemes that require scanning only $O(n)$ entries of the matrix at each step, which will not slow down fast $O(n^2)$ algorithms[3].

5.2. Symmetric pivoting. Symmetric pivoting employs only 1×1 elimination steps, and it is based on the observation that the diagonal of $P_1 R_1 P_1^T$ is a reordering of the diagonal of R_1. This technique moves the maximal magnitude entry on the main diagonal to the (1,1) position. Since the main diagonal of a partially reconstructable Cauchy-like matrix is a part of its generator, therefore the incorporation of symmetric pivoting into the continuous-Cauchy and discrete-Cauchy algorithms does not require any additional computations.

Symmetric pivoting technique is useful with positive definite matrices, for which it is equivalent to *complete pivoting* [GVL89], but clearly this method fails with indefinite matrices in the case of a zero main diagonal. Moreover, even if the main diagonal will never become zero during elimination, symmetric pivoting cannot ensure stability, so the continuous-Cauchy and discrete-Cauchy algorithms with symmetric pivoting will likely break down numerically for indefinite matrices.

5.3. Bunch-Kaufman pivoting. This method is the one of choice in LAPACK [LAPACK]. At each step the Bunch-Kaufman algorithm scans only two columns of the matrix $R_1 = \left[\begin{array}{c} r_{ij} \end{array} \right]_{1 \le i,j \le n}$ to determine the size m (i.e., 1 or 2) of R_{11} and the permutation matrix P_1 in (4.1)). To understand

[3] In [GO94a] an interpretation of *partial* and *diagonal* pivoting techniques is given in terms of a reordering of the interpolation points (i.e., the poles and zeros) for a rational $\alpha \times \alpha$ matrix function corresponding to the partially reconstructable Cauchy-like matrix, cf., e.g., the footnote in section 2.2.

the Bunch-Kaufman technique it helps to consider the matrix $R = \begin{bmatrix} r_{11} & \cdots & \lambda^* & \cdots & \cdots & \cdots \\ \vdots & & \vdots & & & \\ \lambda & \cdots & r_{tt} & \cdots & \sigma^* & \cdots \\ \vdots & & \vdots & & & \\ \vdots & & \sigma & & & \\ \vdots & & \vdots & & & \end{bmatrix}$,

and to note that the pivot R_{11} is either r_{11} or r_{tt} or $\begin{bmatrix} r_{11} & \lambda^* \\ \lambda & r_{tt} \end{bmatrix}$.

<u>The Bunch-Kaufman algorithm</u>

```
$\alpha = (1+\sqrt{17})/8$
$\lambda = |r_{t,1}| = \max\{|r_{2,1}|, ..., |r_{n,1}|\}$
if $\lambda \neq 0$
    if $|r_{1,1} \geq \alpha\lambda$
            $m = 1;\ P = I$
    else
            $\sigma = |r_{p,t}| = \max\{|r_{1,t}|, ..., |r_{t-1,t}|, |r_{t+1,t}|, ..., |r_{n,t}|\}$
            if $\sigma|r_{1,1}| \geq \alpha\lambda^2$
                    $s = 1;\ P = I$
            else if $|r_{t,t}| \geq \alpha\sigma$
                    $m = 1$ and choose $P$ so $(P \cdot R_1 \cdot P^*)_{1,1} = r_{t,t}$
            else
                    $m = 2$ and choose $P$ so $(P \cdot R_1 \cdot P^*)_{2,1} = r_{t,p}$
            end
    end
end
```

The value of the constant $\alpha = (1+\sqrt{17})/8$ is determined to bound the element growth [BK77], [B71]; this method was has been recently proven to be backward stable in [Hig95].

Since the Bunch-Kaufman algorithm requires knowledge of the entries of only one more column of R_1 (the first column should be computed in any case), this pivoting technique can be incorporated into both the continuous-Cauchy and discrete-Cauchy algorithms at the additional cost of $2\alpha n^2$ operations.

6 Numerical results

We applied the algorithms developed in the present paper to solving Hermitian Toeplitz systems of equations, and performed a large number of numerical experiments to compare them to other available algorithms. In the description of the results we shall use the following abbreviations :

Table 3. Algorithms.

GEPP	$O(n^3)$	Gaussian elimination with partial pivoting. The driver routine **sgesv** of LAPACK.
SGEBKP	$O(n^3)$	Symmetric Gaussian elimination with Bunch-Kauffman pivoting. The driver routine **sspsv** of LAPACK.
Levinson	$O(n^2)$	The classical Levinson algorithm.
Schur	$O(n^2)$	The classical Schur algorithm for positive definite Toeplitz matrices, or its immediate modification to compute the LDL^* factorization for indefinite Toeplitz matrices.
GKO	$O(n^2)$	The algorithm of [GKO95]; transformation to a Cauchy-like matrix, and then running a fast GEPP.

Table 3. Continuation.

disc-C	$O(n^2)$	Transformation to a Cauchy-like matrix, on the basis of Lemma 2.5 (for the implementation details see sections 2.6, 2.7), and then running the discrete-Cauchy algorithm of subsection 4.5 (see the implementation details in section 4.6).
cont-C	$O(n^2)$	Transformation to a Cauchy-like matrix, on the basis of Corollary 3.3 (for the implementation details cf. sections 2.6, 2.7), and then running the continuous-Cauchy algorithm of subsection 4.3.

For the latter two algorithms in Table 3 we used several pivoting techniques, the abbreviations for which appear in Table 4.

Table 4. Pivoting techniques.

WP	Without pivoting.
SP	Symmetric pivoting, see, e.g., section 5.2.
BKP	Bunch-Kaufman pivoting, see, e.g., section 5.3.
SBKP	A combination of SP and BKP, i.e., one first maximizes the (1,1) entry by applying the symmetric pivoting step, and then performs the BKP step.

Besides the standard routines from LAPACK for GEPP and SGEBKP, all the algorithms were implemented in single precision using the C language on a DEC 5000/133 RISC workstation (unit roundoff in single precision $u \approx 1.19 \cdot 10^{-7}$), and for the FFT we used FORTRAN single precision routines from FFTPACK. We considered the solution x of $Tx = b$ computed in double precision by GEPP as being exact, and for the solutions $\hat{x}$, computed by each algorithm, we evaluated the forward, residual and backward errors by

$$e = \frac{\|x - \hat{x}\|_2}{\|x\|_2}, \qquad r = \frac{\|T\hat{x} - b\|_2}{\|b\|_2}, \qquad b = \frac{\|T\hat{x} - b\|_2}{\|b\|_2 + \|T\|_2\|\hat{x}\|_2}.$$

For the disc-C and cont-C algorithms we also evaluated the componentwise matrix residual error

$$mr = \max_{ij} |PLDL^*P^T - \mathcal{F}T\mathcal{F}^*|$$

to check the accuracy of triangular factorization. We now present the results of several numerical experiments.

6.1 Positive definite Toeplitz matrices.

Example 1. The prolate Toeplitz matrix. The prolate matrix is defined by

$$T_n = \left[\; t_{i-j} \; \right]_{1 \le n} \qquad \text{where} \qquad t_k = \begin{cases} 2\omega & \text{if} \quad k = 0, \\ \frac{sin(2\pi\omega k)}{\pi k} & \text{otherwise,} \end{cases} \qquad (0 \le \omega \le \frac{1}{2}).$$

Background on the prolate matrix can be found in [V93]; we mention here only that it possesses remarkable spectral and conditioning properties. For small ω, its eigenvalues are clustered around 0 and 1, which makes it extremely ill-conditioned. In fact $k_2(T_n) \approx \frac{1}{p(n,\omega)} e^{\gamma n}$, for some γ and $p(n, \omega)$, where $k_2(T) = \|T\| \cdot \|T^{-1}\|$ is the spectral condition number.

In this example we solved $Tx = \left[\begin{array}{ccccc} 1 & -1 & 1 & -1 & \cdots \end{array} \right]^T$ for $\omega = 1/4$. Table 5 shows that in this case the condition number (computed in double precision) of the prolate matrix is indeed very large, making it a suitable example for testing different Toeplitz solvers.

Table 5. Auxiliary information. Prolate matrix with $\omega = 1/4$;
RHS : $b = \begin{bmatrix} 1 & -1 & 1 & -1 & \cdots \end{bmatrix}^T$.

n	10	40	70	110	120	130	140	150
$k_2(T)$	1.8e+06	4.6e+16	5.9e+16	1.1e+17	1.8e+18	2.7e+17	6.6e+17	9.9e+17
$\|T\|$	1.0e+00	1.0e+00	1.0e+00	1.0e+00	1.0e+00	1.0e+00	1.0e+00	1.0e+00
$\|x\|$	4.6e+06	1.3e+17	2.6e+17	9.3e+17	2.2e+17	1.2e+18	5.2e+17	3.2e+17
$\|b\|$	0.0e+00	5.5e+00	7.7e+00	1.0e+01	1.0e+01	1.1e+01	1.1e+01	1.2e+01

The data in Table 5 show that the norm of the solution (computed in double precision) is extremely large, reflecting the extreme ill-conditioning of the coefficient matrix. Correspondingly the forward and residual errors turned out to be drastically large for all compared algorithms. Table 6 displays the backward error, which is smaller and allows the better comparison of the algorithms.

Table 6. Backward error. Prolate matrix with $a = 1/4$; RHS : $b = \begin{bmatrix} 1 & -1 & 1 & -1 & \cdots \end{bmatrix}^T$.

n	10	40	70	110	120	130	140	150
GEPP	7.2e-08	6.5e-08	1.3e-07	1.2e-07	1.1e-07	1.4e-07	1.1e-07	1.4e-07
SGEBKP	5.0e-08	5.2e-08	5.6e-08	5.6e-08	7.6e-08	7.0e-08	6.5e-08	7.5e-08
Schur	4.8e-08	3.8e-08	3.7e-08	3.7e-08	3.5e-08	3.6e-08	3.7e-08	3.6e-08
Levinson	3.8e-08	6.6e-08	1.0e-07	1.4e-07	4.4e-07	6.2e-06	4.8e-06	5.1e-06
GKO	7.0e-08	1.3e-07	8.3e-08	2.0e-07	1.4e-07	1.1e-07	9.4e-08	1.2e-07
cont-C+WP	9.1e-06	1.4e-03	3.8e-02	1.3e-01	3.7e-01	6.0e-01	1.8e-01	NaN
disc-C+WP	1.4e-06	2.3e-04	6.1e-02	2.7e-03	9.4e-01	6.4e-01	1.5e-01	NaN
cont-C+BKP	7.2e-06	3.7e-04	3.1e-04	1.6e-04	1.7e-04	3.6e-04	1.4e-04	6.0e-04
disc-C+BKP	5.8e-08	9.1e-08	8.1e-08	7.1e-08	7.6e-08	1.1e-07	7.3e-08	8.8e-08
cont-C+SP	4.2e-08	4.2e-07	3.4e-07	4.7e-07	1.7e-07	1.3e-06	1.4e-07	2.4e-06
disc-C+SP	2.7e-08	5.7e-08	8.4e-08	8.6e-08	7.0e-08	1.1e-07	8.6e-08	9.9e-08

We shall analyze the data of Table 6 below. However before doing so, let us describe the result of a test with another positive definite Toeplitz matrix.

Example 2. Gaussian Toeplitz matrix. This matrix has the form $T_n = \begin{bmatrix} t_{i-j} \end{bmatrix}_{1 \le n}$ where $t_k = a^{k^2}$ $0 < a < 1$, with a close to 1. Background on such matrices can be found in [PD92]. We mention here only that this positive definite matrix can arise as a discretization of Gaussian convolution and that

$$k_2(T_n) \ge \frac{1+a}{(1-a^2)(1-a^4)\cdots(1-a^{2(n-1)})}.$$

In order to avoid the situation of the previous example, where the ill-conditioning of the coefficient matrix is reflected in the growth of the solution norm (see, e.g., Table 5), here we chose a right hand side b by accumulation, $b = T \cdot \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$. Table 7 displays some useful information for such Gaussian Toeplitz systems.

Table 7. Auxiliary information. The Gaussian Toeplitz matrix with $a = 0.9$;
RHS : $b = T \cdot \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$.

n	10	30	50	70	90	110	130
$k_2(T)$	1.4e+06	1.2e+09	3.9e+09	5.4e+09	6.2e+09	6.6e+09	6.8e+09
$\|T\|$	4.7e+00	5.3e+00	5.4e+00	5.4e+00	5.4e+00	5.5e+00	5.5e+00
$\|x\|$	3.2e+00	1.1e+01	2.5e+01	2.9e+01	3.0e+01	3.1e+01	3.1e+01
$\|b\|$	0.0e+00	2.3e+01	3.3e+01	4.1e+01	4.8e+01	5.4e+01	5.9e+01

In Tables 8, 9 we list the the measurements of the corresponding backward and the residual errors.

Table 8. Backward error. The Gaussian Toeplitz matrix with $a = 0.9$;
RHS : $b = T \cdot \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$.

n	10	30	50	70	90	110	130
GEPP	2.1e-08	3.1e-08	2.1e-08	1.9e-08	1.9e-08	1.8e-08	2.0e-08
SGEBKP	4.6e-08	1.5e-08	2.3e-08	1.8e-08	1.7e-08	1.7e-08	1.6e-08
Schur	2.5e-08	1.4e-08	1.0e-08	1.1e-08	1.3e-08	1.2e-08	1.2e-08
Levinson	1.2e-06	9.0e-07	3.0e-06	3.7e-06	3.9e-06	4.2e-06	4.0e-06
GKO	1.1e-07	1.2e-07	3.1e-07	2.7e-07	1.8e-07	3.1e-07	4.0e-07
cont-C+WP	3.6e-08	1.1e-05	1.2e-05	6.8e-05	1.9e-05	1.1e-04	5.4e-05
disc-C+WP	3.8e-08	8.8e-06	1.7e-05	9.6e-05	2.7e-05	5.4e-05	3.2e-05
cont-C+BKP	3.1e-08	3.3e-06	5.7e-06	6.0e-05	8.7e-08	1.7e-06	1.1e-06
disc-C+BKP	6.1e-08	4.0e-08	6.4e-08	7.3e-08	3.3e-07	7.8e-08	6.5e-08
cont-C+SP	4.3e-08	2.5e-08	7.2e-08	6.5e-08	1.5e-07	8.4e-08	1.9e-07
disc-C+SP	7.5e-08	2.5e-08	7.5e-08	4.8e-08	4.3e-08	6.5e-08	8.7e-08

Table 9. Residual error. The Gaussian Toeplitz matrix with $a = 0.9$;
RHS : $b = T \cdot \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T$.

n	10	30	50	70	90	110	130
GEPP	4.3e-08	1.1e-07	4.4e-07	2.0e-07	2.9e-06	1.7e-06	2.6e-06
SGEBKP	9.4e-08	3.0e-07	1.6e-07	1.4e-06	3.7e-07	3.7e-07	3.3e-07
Schur	5.0e-08	7.5e-08	1.2e-07	1.2e-07	1.4e-07	1.1e-07	1.1e-07
Levinson	2.5e-06	6.0e-05	2.8e-04	2.9e-04	2.9e-04	3.0e-04	2.9e-04
GKO	2.2e-07	3.9e-07	7.9e-07	7.3e-07	7.6e-07	1.3e-06	1.5e-06
cont-C+WP	7.3e-08	2.3e-05	2.5e-05	1.9e-04	3.9e-05	4.9e-04	1.3e-04
disc-C+WP	7.7e-08	1.8e-05	3.5e-05	1.5e-03	5.7e-05	1.2e-04	1.1e-04
cont-C+BKP	6.4e-08	6.7e-06	1.2e-05	1.3e-04	1.8e-07	3.3e-06	2.1e-06
disc-C+BKP	1.2e-07	8.5e-08	1.6e-07	2.7e-07	7.2e-07	3.1e-07	9.3e-07
cont-C+SP	8.8e-08	5.2e-08	1.8e-07	1.6e-07	3.0e-07	3.3e-07	4.3e-07
disc-C+SP	1.5e-07	5.4e-08	2.0e-07	2.8e-07	1.1e-07	6.1e-07	7.1e-07

The analysis of the data in Tables 6-9 suggests the following conclusions for positive definite Toeplitz matrices.

- Tables 6, 8 confirm the analytical results of [BBHS95] on the backward stability of the classical Schur algorithm[4]. Moreover, one sees that with positive definite Toeplitz matrices the classical Schur algorithm produces the smallest backward error among all the compared algorithms.

- Recall that the positive definiteness of a Toeplitz matrix is equivalent to the fact that the corresponding reflection coefficients belong to the interval (-1,1). Cybenko proved in [C80] that if the reflection coefficients belong to the smaller interval (0,1) then the Levinson algorithm is guaranteed to produce a residual comparable to the one of stable Cholesky factorization. Varah observed in [V93] that the Prolate matrix is an example where the reflection coefficients are of both signs (so the Cybenko results are not applicable), and where the Levinson algorithm produces a residual about 10^3 times larger than with the standard numerically stable algorithms. In [GKO95] we presented two more such examples with positive definite Toeplitz matrices with sign-alternating reflection coefficients, for which the Levinson algorithm produced large residuals. The coefficient matrix in one of these examples was a Gaussian Toeplitz matrix, and the data in Table 9 on the residual error of the Levinson algorithm reinforce the conclusions of [V93] (see also [GKO95]). Moreover the data on the Levinson algorithm in Table 6 for $n > 130$ and in Table 8 suggest that if the reflection coefficients are of both signs,

[4]See also [GKO95] for a numerical example showing some limitations of the results of [BBHS95], which in fact are only valid for not extremely ill-conditioned matrices.

then not only the residual, but also the corresponding backward error, for this algorithm is larger than the one of the standard GEPP and SGEBKP. The numerical comparison of the classical Levinson and Schur algorithms indicates that the wide spread use of the better known Levinson algorithm in engineering and scientific applications may be not appropriate from the numerical point of view; the Schur algorithm can be much better.

- Pivoting (even for positive definite matrices) plays a crucial role in achieving high accuracy with the cont-C and disc-C algorithms, without pivoting they are not reliable, will give less correct digits, and often encounter overflows, denoted by NaN in Tables 6, 8.

- Tables 6 and 8 show that the cont-C and disc-C algorithms have different numerical behavior for positive definite matrices. Moreover, the disc-C algorithm demonstrated the best numerical performance among all the compared transformation-and-pivoting algorithms. The algorithm disc-C produces approximately the same accuracy when combined with the BK pivoting or with SP pivoting.

- The accuracy of the cont-C algorithm is more significantly dependent upon the choice of pivoting strategy. The standard BK pivoting technique is not the optimal choice, and the corresponding backward error is not as favorable as the one for the disc-C algorithm. However the cont-C algorithm with SP pivoting does quite satisfactory results.

6.2 Indefinite Hermitian Toeplitz matrices

Example 3. Random symmetric Toeplitz matrix. Here we generated a Toeplitz matrix with random entries in the interval (-1,1), and solved a linear system $T \cdot x = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}$. For $10 \leq n \leq 150$ the condition number of such T usually varies from 10 to 10^2. Here are the measurements of the forward error.

Table 10. Forward error. Matrix - Random; RHS - Ones.

n	10	30	50	70	90	110	130
GEPP	2.5e-07	5.2e-07	1.9e-06	1.3e-06	1.3e-05	1.1e-05	5.9e-06
SGEBKP	4.4e-07	4.8e-07	2.1e-06	1.6e-06	5.1e-05	2.4e-05	1.3e-05
Schur	1.4e-03	1.4e-05	1.8e-05	5.7e-05	5.4e-03	4.9e-02	3.6e-01
Levinson	8.0e-05	1.1e-05	1.6e-05	1.2e-04	1.3e-04	1.1e-04	1.8e-04
GKO	4.1e-07	1.0e-06	2.6e-06	7.7e-07	4.1e-05	3.9e-05	5.9e-06
cont-C+WP	6.1e-06	4.4e-05	1.1e+01	5.3e+00	1.3e+01	NaN	NaN
disc-C+WP	2.4e-06	8.9e-05	1.0e+01	NaN	2.6e+01	NaN	NaN
cont-C+BKP	4.1e-06	5.5e-06	9.3e-05	3.3e-06	1.2e-01	6.4e-05	1.4e-02
disc-C+BKP	2.0e-06	2.2e-06	6.1e-06	1.0e-05	6.1e-04	3.5e-05	1.2e-03
cont-C+SBKP	9.3e-07	2.3e-06	6.1e-05	4.6e-06	5.6e-04	5.7e-05	1.5e-04
disc-C+SBKP	5.2e-07	1.7e-06	1.1e-05	1.7e-06	4.1e-05	2.3e-04	2.5e-05

The analysis of the data in Table 10 suggest the following conclusions

- Although the classical Levinson and Schur algorithms are much faster than GEPP and SGEBKP, they cannot provide the same accuracy for indefinite matrices.

- The data confirm the observation of [GKO95] that the accuracy shown in practice by the GKO algorithm is about the same as the one of standard numerically stable methods.

- Pivoting plays a profound role in achieving high accuracy with the disc-C and cont-C algorithms for indefinite Toeplitz matrices. In fact without pivoting these algorithms fail to compute even one correct digit for $n \geq 50$ and encounter overflows denoted in the Table by

NaN. BK pivoting almost always gives satisfactory results; however the case $n = 90$ shows that occasionally the forward error for the cont-C and disc-C algorithms can jump.

- In practice the use of the SBKP pivoting allows to avoid such jumps, and both disc-C and cont-C show almost the same high accuracy as the GKO algorithm. However the latter is the most reliable among the compared fast algorithms for indefinite Toeplitz matrices.
- The GEPP algorithm is slightly more accurate than the SGEBKP algorithm. Therefore it is not surprising that the same fact holds valid for their fast implementations, and that the GKO algorithm is a little bit more accurate than the disc-C+SBKP and cont-C+SBKP algorithms.

Table 11. Componentwise matrix residual error in $|PLDL^*P^* - \mathcal{F}T\mathcal{F}^*|$.
Matrix - Random.

n	10	30	50	70	90	110	130
disc-C+WP	8.8e-06	1.5e-03	7.1e+02	6.5e+14	1.4e+05	NaN	NaN
disc-C+WP	5.3e-06	3.0e-03	4.0e+02	0.0e+00	1.5e+04	NaN	NaN
cont-C+BKP	5.8e-06	7.0e-05	1.1e-03	1.1e-03	3.2e-01	8.4e-05	2.3e+00
disc-C+BKP	8.2e-07	1.2e-05	4.9e-05	2.4e-03	2.2e-03	1.7e-04	1.2e-02
cont-C+SBKP	3.8e-07	8.9e-06	3.1e-04	1.8e-05	2.4e-03	1.5e-04	1.5e-03
disc-C+SBKP	3.0e-07	3.3e-06	2.6e-05	1.1e-05	2.2e-04	1.8e-05	1.8e-04

- Table 11 shows that the algorithm disc-C provides a more precise triangular factorization for Cauchy-like matrix $\mathcal{F}T\mathcal{F}^*$ then the algorithm cont-C, and that the SBKP is the best pivoting strategy for this case.

Example 4. The Chebyshev-Toeplitz matrix. The data in Table 10 show that the classical Levinson and Schur algorithms provide less accuracy with random indefinite Toeplitz matrices than do the other algorithms that were compared. However, note that the Levinson algorithm computes the triangular factorization for T^{-1}, while the classical Schur algorithm computes the triangular factorization for T itself. Both factorizations can be used for iterative refinement, which will not increase by much the $O(n^2)$ complexity of these algorithms, but will provide more accurate solutions. Another possibility of enhancing the computed solution is to explore a look-ahead approach, which is based on the observation that the reason for the lack of the accuracy is the recursive nature of the Levinson and Schur algorithms, which sequentially process all leading submatrices, and break down numerically if some of them are near-singular. In the look-ahead versions for these two algorithms, one jumps over such ill-conditioned submatrices, which often produces satisfactory results. However, both the iterative refinement and look-ahead enhancements have limitations, and here is an example for which neither of them will work.

Consider a symmetric $2n \times 2n$ Chebyshev-Toeplitz matrix T, whose first row is given by

$$\begin{bmatrix} T_0(a) & \cdots & T_{n-1}(a) & 0 & \cdots & 0 \end{bmatrix}.$$

The well-known recurrence relations for Chebyshev polynomials $\{T_k(x)\}$ imply that for $k = 3, 4, ..., n$ all $k \times k$ leading minors of T are zero. Thus the classical Levinson and Schur algorithms both break down, so there is nothing to refine. Furthermore, using any of the look-ahead algorithms, one has to jump over half of the matrix, thus performing $O(n^3)$ operations, thus failing to achieve any superiority in speed over the slow GEPP and SGEBKP algorithms. On the other hand Table 12 shows that even for Chebyshev-Toeplitz systems the fast $O(n^2)$ GKO, disc-C and cont-C algorithms provide very satisfactory results. Here the parameter defining the coefficient matrix is given by $a = 0.2$, and the right-hand side b was computed by accumulating $T \cdot x$ for x with components randomly distributed in (-1,1).

Table 12. Forward error. Chebyshev-Toeplitz matrix for $a = 0.2$;
RHS : $b = T \cdot x$, where x is random.

n	10	30	50	70	90	100
GEPP	1.1e-06	3.2e-07	4.6e-07	7.8e-06	1.7e-06	1.5e-06
SGEBKP	3.3e-07	5.2e-07	7.1e-07	1.6e-05	1.7e-06	2.1e-06
Schur	NaN	NaN	NaN	NaN	NaN	NaN
Levinson	1.0e+00	2.6e+07	1.5e+12	2.3e+12	1.5e+13	6.3e+15
GKO	3.4e-06	6.0e-07	9.3e-07	2.5e-05	2.3e-06	2.4e-06
cont-C+WP	2.7e-05	4.8e-07	2.3e-05	1.3e+00	2.1e-06	4.9e-03
disc-C+WP	4.6e-05	5.5e-07	2.0e-05	6.1e-01	1.9e-06	6.1e-03
cont-C+BKP	5.1e-06	4.9e-07	8.2e-07	3.9e-05	2.2e-06	1.1e-04
disc-C+BKP	6.3e-05	3.4e-07	1.0e-06	5.7e-04	1.3e-06	1.1e-05
cont-C+SBKP	7.5e-06	4.2e-07	1.6e-06	2.2e-06	1.4e-06	3.6e-06
disc-C+SBKP	1.4e-05	3.0e-07	1.0e-06	4.5e-06	1.4e-06	3.7e-06

- The classical Schur and Levinson algorithms indeed fail, whereas for other compared algorithms the data in Table 12 are very similar to those in Table 10, thus leading to the same conclusions.

7 Recursive solution for boundary homogeneous interpolation problem for J-unitary rational matrix functions

7.1. Boundary homogeneous interpolation problem for J-unitary rational matrix functions. Continuous time. In this paper we focused on the use of the discrete-Cauchy and continuous-Cauchy algorithms to design new fast and accurate Toeplitz-like solvers. However these algorithms also have other important applications in the framework of rational matrix interpolation theory. We shall pursue the full details elsewhere, but it seems to be important and instructive to reveal these connections and to briefly outline the main points here.

It is known (see [BGR91]) that many boundary rational matrix interpolation problems with norm constraints, including the celebrated boundary matrix Nevanlinna-Pick interpolation problem, can be reduced to the construction of a rational matrix function $\Theta(z)$ that is J-unitary on the imaginary axis, i.e., for those $z \in i\mathbf{R}$ that are not poles of $\Theta(z)$ we have

$$(\Theta(z))^* \cdot J \cdot \Theta(z) = J. \tag{7.1}$$

The problem is the following.

Boundary homogeneous interpolation problem for J-unitary rational matrix functions

- Given n points $a_1, ..., a_n$ on the imaginary axis $i\mathbf{R}$, n $1 \times \alpha$ row vectors $\varphi_1, ..., \varphi_n$, and n real numbers $\rho_1, ..., \rho_n$, construct a J-unitary rational $\alpha \times \alpha$ matrix function $\Theta(z)$ with value I at infinity such that
  - For $k = 1, 2, ..., n$, both $\Theta(z)$ and $\Theta(z)^{-1}$ have a simple pole at a_k, i.e., in the neighborhood of a_k we have
  $$\Theta(z) = R_{-1,k} \cdot \frac{1}{z - a_k} + [\text{analytic} \quad \text{at} \quad a_k],$$
  where
  $$\text{Res}_{z=a_k}\Theta(z) := R_{-1,k}$$
  is one-dimensional; and with a similar decomposition at a_k for $\Theta(z)^{-1}$.
  - The points $a_1, ..., a_n$ are the only poles of $\Theta(z)^{-1}$.
  - For $k = 1, 2, ..., n$,
  $$\text{Im}_l \text{Res}_{z=a_k}\Theta(z)^{-1} = \text{span}\{\varphi_k\}, \tag{7.2}$$
  i.e., $R_{-1,k} = \varphi_k \cdot \psi_k$ for some $\alpha \times 1$ column vector ψ_k.

– For $k = 1, 2, ..., n$,

$$\varphi_{k,1} \cdot J \cdot \varphi_k = -\rho_k, \tag{7.3}$$

where $\varphi_{k,1}$ is a row vector such that the row vector function

$$[\varphi_k + \varphi_{k,1}(z - a_k)] \cdot \Theta(z)$$

is analytic at a_k with value 0 at a_k. The numbers $\rho_1, ..., \rho_n$ are called *coupling numbers.*

The equality (7.1) is known to imply that the poles of $\Theta(z)$ are given by $-a_1^*, ..., -a_n^*$. If the zeros $\{a_k\}$ of $\Theta(z)$ (i.e., the poles of $\Theta(z)^{-1}$), and the poles $\{-a_k^*\}$ of $\Theta(z)$ would be two disjoint sets, then the analyticity of $\Theta(z)$ at $\{a_k\}$ would allow us to express (7.2) as the standard left tangential interpolation conditions

$$\varphi_i \cdot \Theta(a_i) = 0, \qquad (i = 1, 2, ..., n). \tag{7.4}$$

We however consider in this section the less studied case where the zeros lie on the imaginary axis, i.e., $a_k = -a_k^* \in i\mathbf{R}$, so that $\Theta(z)$ has zeros and poles at the same points. In this situation $\Theta(z)$ is no longer analytic at $\{a_k\}$, so the interpolation conditions have to be captured by the more delicate and more general formulation used in (7.2).

The solution of the above boundary homogeneous matrix interpolation problem (as given, e.g., in [BGR91]), can be formulated in our language as follows. Define

$$A_1 = \operatorname{diag}(a_1, ..., a_k), \qquad G_1 = \begin{bmatrix} \varphi_1 \\ \vdots \\ \varphi_n \end{bmatrix}, \qquad d_1 = \begin{bmatrix} -\rho_1 & -\rho_2 & \cdots & -\rho_n \end{bmatrix}. \tag{7.5}$$

If there exists a nonsingular partially reconstructable Cauchy-like matrix R_1, with a $\triangle_{A_1}$-generator $\{G_1, J, \operatorname{diag}(d_1)\}$, i.e., if

- R_1 satisfies
$$A_1 \cdot R_1 + R_1 \cdot A_1^* = G_1 \cdot J \cdot G_1^* \tag{7.6}$$
so that the rows φ_k in (7.5) are necessarily J-neutral vectors (i.e., $\varphi_k \cdot J \cdot \varphi_k^* = 0$);
- The diagonal part of R_1 is given by the coupling numbers, $\operatorname{diag}(R_1) = \operatorname{diag}(-\rho_1, ..., -\rho_n)$;

then the unique solution for the above stated interpolation problem is given by the (so-called global state-space) formula

$$\Theta(z) = I - J \cdot G_1^* \cdot (zI - A_1)^{-1} \cdot R_1^{-1} \cdot G_1. \tag{7.7}$$

Because this formula explicitly involves R_1^{-1}, where R_1 has to be found from (7.6), this solution, as it stands, cannot be regarded as a computational procedure for obtaining a solution $\Theta(z)$. For example, evaluating the value $\Theta(z)$ at a given point z requires $O(n^3)$ operations, if standard matrix inversions methods are employed. In the next subsection we shall clarify how the $O(\alpha n^2)$ continuous-Cauchy algorithm suggests a more convenient representation for $\Theta(z)$, allowing us to further evaluate $\Theta(z)$ in only $O(\alpha n)$ operations.

7.2. Cascade decomposition. The continuous-Cauchy algorithm of Section 4 conveniently represents the same $\Theta(z)$ as a cascade

$$\Theta(z) = \Theta_1(z) \cdot \Theta_2(z) \cdot ... \cdot \Theta_m(z) \tag{7.8}$$

where each J-unitary factor $\Theta_k(z)$ has the same form as in (as (7.7))

$$\Theta_i(z) = I - J \cdot \tilde{G}_i^* \cdot (zI - \tilde{A}_i)^{-1} \cdot \tilde{R}_i^{-1} \cdot \tilde{G}_i, \tag{7.9}$$

except that instead of the $n \times n$ matrices A_1 and R_1, we now have only 1×1 or 2×2 matrices $\tilde{A}_i$ and $\tilde{R}_i$. More precisely, given the data in (7.5), then the first recursive step of the continuous-Cauchy algorithm represents $\Theta(z)$ in (7.7) as the product of two (J-unitary on $i\mathbf{R}$) factors

$$\Theta(z) = \Theta_1(z) \cdot W_1(z). \tag{7.10}$$

The particular form of the factors in (7.10) depends upon the size m of the block elimination step in step **2** of the continuous-Cauchy algorithm (see Section 4). If $m = 1$, $\Theta(z)$ is a first-order section

$$\Theta_1(z) = I - J\varphi_1^* \cdot \frac{1}{z - a_1} \cdot \frac{1}{\rho_1} \cdot \varphi_1. \tag{7.11}$$

The quantities $\{a_1, \varphi_1, \rho_1\}$ appearing in (7.11) are simply borrowed from (7.5), so that $a_1 \in i\mathbf{R}$, and φ_1 is a J-neutral vector; such factors (7.11) are called *Brune sections* in the engineering literature, see, e.g., [DD84]. If $m = 2$, $\Theta(z)$ is a second-order section

$$\Theta_1(z) = I - J \begin{bmatrix} \varphi_1^* & \varphi_2^* \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{(z-a_1)} & 0 \\ 0 & \frac{1}{(z-a_2)} \end{bmatrix} \cdot \begin{bmatrix} \rho_1 & r_{21}^* \\ r_{21} & \rho_2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \varphi_1 \\ \varphi_2 \end{bmatrix}. \tag{7.12}$$

Again the quantities $\{a_1, a_2, \varphi_1, \varphi_2, \rho_1, \rho_2\}$ are borrowed from (7.5) and r_{21} is the entry of the first column of R_1 computed in step **3.** of the continuous-Cauchy algorithm. Summarizing, in both cases $m = 1$ and $m = 2$, the first cascade factor $\Theta_1(z)$ is constructed without computation from the input data of the algorithm.

We next clarify how to obtain the data describing the "quotient" $W_1(z)$ in (7.10). To this end recall that the continuous-Cauchy algorithm in both cases $m = 1$ and $m = 2$ computes the $\triangle_{A_2}$-generator $\{G_2, J, d_2\}$ of the Schur complement R_2 . As was shown in [GO94a], the J-unitary quotient $W_1(z)$ is given by

$$W_1(z) = I - J \cdot G_2^* \cdot (zI - A_2)^{-1} \cdot R_2^{-1} \cdot G_2, \tag{7.13}$$

Since $\{G_2, J, d_2\}$ is an input of the second recursive step of the continuous-Cauchy algorithm, the latter further computes the factorization $W_1(z) = \Theta_2(z) \cdot W_2(z)$, and thus after at most n steps we shall obtain in $O(\alpha n^2)$ operations the whole factorization (7.8).

7.3. Incorporation of pivoting. Observe that the matrix $\Theta(z)$ in (7.7) can be equivalently written as

$$\Theta(z) = I - J \cdot (PG)^* \cdot (zI - (PAP^T))^{-1} \cdot (PRP^T)^{-1} \cdot (PG),$$

where P is any permutation, i.e., one can reorder the interpolation data $\{a_k, \varphi_k, \rho_k\}$ in (7.5) at will. In particular this means that the cascade decomposition (7.8) for a given $\Theta(z)$ is highly nonunique, because at each recursive step of the continuous-Cauchy algorithm we have a variety of choices for the zero and pole a_k of the next cascade factor $\Theta_k(z)$. Though all theoretically equivalent, these decompositions have different numerical properties. Moreover, because the reordering of interpolation data $\{a_k, \varphi_k, \rho_k\}$ is equivalent to row and column permutation of a partially reconstructable Cauchy-like matrix $R_1 \leftarrow PR_1P^T$ (cf. with [GO94a]), the continuous-Cauchy algorithm combined with Bunch-Kaufman pivoting suggests not only a recursive, but also an accurate, solution for the above boundary homogeneous interpolation problem for J-unitary rational matrix function.

7.4. Discrete-time case. Analogously the discrete-Cauchy algorithm with Bunch-Kaufman pivoting suggests an accurate recursive solution for the analogous boundary homogeneous interpolation problem of Section 7.1, with interpolation points $f_1, ..., f_k$ on the unit circle. The solution is now reduced to finding a rational matrix function $\Psi(z)$ that is now J-unitary on the unit circle, i.e.,

$$(\Psi(z))^* \cdot J \cdot \Psi(z) = J,$$

for those $|z| = 1$ that are not poles of $\Theta(z)$. The solution for such problem is given by the global state-space formula [BGR91]

$$\Psi(z) = [I - JG_1^*(zI - F_1)^{-1}R_1^{-1}F_1^{-*}G_1]D_\alpha, \tag{7.14}$$

where

$$D_\alpha = I - JG_1^*R_1^{-1}(I - \alpha F_1^*)^{-1}G_1,$$

α is any number on the unit circle such that $(I - \alpha F_1^*)$ is nonsingular, and R_1 is a nonsingular partially reconstructable Cauchy-like matrix, satisfying a discrete-time Lyapunov displacement equation

$$R_1 - F_1R_1F_1^* = G_1 \cdot J \cdot G_1^*,$$

where

$$F_1 = \text{diag}(f_1, ..., f_n), \qquad G_1 = \begin{bmatrix} \varphi_1 \\ \vdots \\ \varphi_n \end{bmatrix}.$$

Again, the formula (7.14) explicitly involves R_1^{-1} and it does not provide yet an efficient computational procedure. In fact such a procedure is suggested by the discrete-Cauchy algorithm for the LDL^* factorization of R_1, which in fact computes a convenient decomposition of $\Psi(z)$ into the cascade

$$\Psi(z) = \Psi_1(z) \cdot \Psi_2(z) \cdot ... \cdot \Psi_m(z) \tag{7.15}$$

of the first-order and second-order J-unitary on the circle sections. The formulas for the factors have the same form as in (7.14), and for brevity are omitted here.

However it is instructive to get more insight into the relation between the discrete-Cauchy and continuous-Cauchy algorithms. To this end recall that since $\Psi(z)$ in (7.14) is J-unitary on the circle,

$$\Theta(z) = \Psi(\tau\frac{z-1}{z+1}) \tag{7.16}$$

is a J-unitary on $i\mathbf{R}$ rational matrix function, having a state-space realization

$$\Theta(z) = I + J\sqrt{2}G^*(\tau^*I - F^*)^{-1}(zI - A)^{-1}R^{-1}\sqrt{2}(\tau I - F)G, \tag{7.17}$$

where $A = (\tau I + F)(\tau I - F)^{-1}$. By inspecting the formula (7.17) and Lemma 3.1, one sees that the latter Lemma converts the J-unitary-on-the-circle rational matrix function $\Psi(z)$ in (7.14) into the J-unitary-on-$i\mathbf{R}$ rational matrix function $\Theta(z)$ in (7.17), which satisfies, in fact, the same interpolation conditions, cf. (7.16). Moreover the two cascades (7.15) and (7.8) for $\Theta(z)$ in (7.17), computed by the discrete-Cauchy and continuous-Cauchy algorithms, respectively, are essentially the same, with the sections $\Theta_k(z) = \Psi_k(\tau\frac{z-1}{z+1})$ related as in (7.17) and (7.14).

This observation gives a factorization interpretation for the derivation in section 4 of the discrete-Cauchy algorithm from the continuous-Cauchy algorithm (recall that the proof of Lemma 4.2 was based on Lemma 3.1, transforming a discrete-time Lyapunov displacement equation into a continuous-time one). Indeed, in the language of this section, our aim in Lemma 4.2 was to obtain the formulas for factorization of $\Psi(z)$ in (7.14) into the product

$$\Psi(z) = \Psi_1(z) \cdot U_1(z) \tag{7.18}$$

of J-unitary on the circle factors. We used Lemma 3.1 to convert this desired factorization problem (7.18) into

$$\Psi(\tau\frac{z-1}{z+1}) = \Psi_1(\tau\frac{z-1}{z+1}) \cdot U_1(\tau\frac{z-1}{z+1}), \tag{7.19}$$

where all factors are now J-unitary on $i\mathbf{R}$, so the formulas for the factors in (7.19) are described by Lemma 4.1. Using these formulas and Lemma 3.1, we again obtain the decomposition in (7.18). Summarizing, the continuous-Cauchy and discrete-Cauchy algorithms compute essentially the same cascade decompositions, which are related to each other via an appropriate change of variable (i.e., via the Möbius transformation of the interior of the unit disk onto the right half plane, see Sec. 3).

8 Concluding remarks

In an earlier paper [GKO95] we designed a fast $O(n^2)$ Gaussian elimination with partial pivoting algorithm for Cauchy-like matrices, and as an application suggested a new fast Toeplitz-like solver GKO. A backward error analysis for the GKO algorithm appeared in [SB95], where the bound for the backward error involved (along with the usual *element growth* term) a so-called *generator growth* term. Since the corresponding bound for the standard $O(n^3)$ GEPP algorithm involves only the element growth term, this analysis indicated that the GKO algorithm may be less accurate if during elimination the entries of the generators grow faster than do the matrix entries. However current experiments do not reveal actual examples of Toeplitz matrices for which such generator growth does indeed occur; finding such an example would indicate that the bound of [SB95] is tight. Moreover in [G95], Ming Gu showed how potential generator growth can be suppressed by incorporation into the GKO algorithm of steps consisting of QR factorization of the generator, at the expense of $O(n^2)$ additional operations.

In this paper we focused on *Hermitian* matrices, and implemented fast $O(n^2)$ symmetric Gaussian elimination with diagonal pivoting for partially reconstructable Cauchy-like matrices. Although there are many theoretically equivalent schemes of that kind, not all of them have good numerical properties. By extensive testing we found several numerically accurate algorithms that are about twice as fast as the original GKO algorithm (which does not preserve symmetry). The high accuracy of the new algorithms is achieved by allowing the corresponding displacement equations to have nontrivial kernels (i.e., nontrivial nullspaces). This situation gives rise to the class of *partially reconstructable matrices*, introduced and studied in this paper. We extended to partially reconstructable matrices the transformation technique, and the generalized Schur algorithms, corresponding to the *continuous-time* and *discrete-time* Lyapunov displacement operators. An application of these algorithms combined with Bunch-Kaufman pivoting to accurate recursive solution of certain boundary homogeneous interpolation problem was described in Section 7.

Finally we note that it is likely that the results of [SB95] (which contains a general methodology for the error analysis of the GKO-like algorithms) and of [G95] can be extended to partially reconstructable matrices and to the new Toeplitz solvers proposed in the present paper, a useful topic for further investigation.

Acknowledgment. After the first 5 sections of this paper were finished, and we had performed numerical tests, the authors received from Thomas Huckle a preprint [Hu95], which also presents a symmetry-preserving variant of the GKO algorithm. The scheme of [Hu95] transforms a symmetric Toeplitz (but not any Toeplitz-like) matrix to a Hermitian Cauchy-like matrix, for which it implements scalar elimination steps only. For this reason, Huckle's algorithm is limited to positive definite Toeplitz matrices. In the present paper we show how to transform *any* Hermitian *Toeplitz-like* matrix to a Hermitian Cauchy-like matrix, for which we implement not only scalar, but also block, elimination steps. This allows us to incorporate the Bunch-Kaufman pivoting procedure, and to propose a group of algorithms applicable to *indefinite* Toeplitz-*like* matrices. Recently Georg Heinig informed us that Adam Bojanczyk and he are also pursuing a symmetry-exploiting transformation-and-pivoting algorithm [H95b].

References

[AG90] G.Ammar and P.Gader, *New decompositions of the inverse of a Toeplitz matrix*, Signal processing, Scattering and Operator Theory, and Numerical Methods, Proc. Int. Symp. MTNS-89, vol. III, 421-428, Birkhauser, Boston, 1990.

[B71] J.R.Bunch, *Analysis of the diagonal pivoting method*, SIAM J. Num. Anal., **8** (1971), 656 -680.

[B85] J.Bunch, *Stability of methods for solving Toeplitz systems of equations*, SIAM J. of Matrix Analysis, **6** (1985), 349-364.

[BBHS95] A.W.Bojanczyk, R.P.Brent, F.R. de Hoog and D.R.Sweet, *On the stability of the Bareiss and related Toeplitz factorization algorithms*, SIAM J. on Matrix Analysis Appl., **16**No.1 (1995).

[BGR90] J. Ball, I.Gohberg and L.Rodman, *Interpolation of rational matrix functions*, OT45, Birkhäuser Verlag, Basel, 1990.

[BGR91] J. Ball, I.Gohberg and L.Rodman, *Boundary Nevanlinna-Pick Interpolation for Rational Matrix Functions*, J. of Mathematical Systems, Estimation and Control, **1** No 2. (1991), 131-164.

[BH94] A.Bojanczyk and G.Heinig, *A multi-step algorithm for Hankel matrices*, J. of Complexity, 1994.

[BK77] J.R.Bunch and L.Kaufman, *Some stable methods for calculating inertia and solving symmetric linear systems*, Math. Comp., **31**, 162 - 179.

[BP71] J.R.Bunch and B.Parlett, *Direct methods for solving symmetric indefinite systems of linear equations*, SIAM J. Num. Anal., **8**(1971), 639 - 655.

[BP94] D.Bini and V.Pan, *Polynomial and Matrix Computations*, Volume 1, Birkhauser, Boston, 1994.

[C80] G.Cybenko, *The numerical stability of Levinson–Durbin algorithm for Toeplitz systems of equations*, SIAM J. Sci. Statist. Comput., **1** (1980), 303 – 319.

[C88] T.Chan, *An optimal circulant preconditioner for Toeplitz systems*, SIAM J. Sci. Stat. Comput. **9 (4)** (1988), 166 - 771.

[CH92] T.Chan and P.Hansen, *A look-ahead Levinson algorithm for indefinite l Toeplitz systems*, SIAM J. on Matrix Anal. and Appl., 13(1992), 1079-1090.

[CK91] J.Chun and T.Kailath, *Displacement structure for Hankel, Vandermonde and related (derived) matrices*, Linear Algebra Appl., **151**(1991), 199-227.

[CKLA87] J.Chun, T.Kailath and H.Lev-Ari, *Fast parallel algorithms for QR and triangular factorization*, SIAM J. Sci.Stat. Comput., **8** (1987), 899 – 913.

[D79] P. Davis, *Circulant Matrices*, John Wiley, New York, 1979.

[DD84] P.Dewilde and H.Dym, *Lossless inverse scattering, digital filters, and estimation theory*, IEEE Transactions on Information Theory, **IT-30**, No. 4 (1984), 644 - 661.

[FZ93] R.Freund and H.Zha, *A look-ahead strategy for the solution of general Hankel systems*, Numerische Mathematik, **64**, (1993), 295-322.

[G95] Ming Gu, *Stable and efficient algorithms for structured systems of linear equations*, preprint, 1995.

[GH94] M.Gutknecht and M.Hochbruck, *Look-ahead Levinson and Schur recurrences in the Pad'e table*, Electronic Transactions on Numerical Analysis, **2** (1994), 104-129.

[GO92] I.Gohberg and V.Olshevsky, *Circulants, displacements and decompositions of matrices*, Integral Equations and Operator Theory, **15**, No. 5 (1992), 730 -743.

[GO94a] I.Gohberg and V.Olshevsky, *Fast state space algorithms for matrix Nehari and Nehari-Takagi interpolation problems*, Integral Equations and Operator Theory, **20**, No. 1 (1994), 44 – 83.

[GO94b] I.Gohberg and V.Olshevsky, *Complexity of multiplication with vectors for structured matrices*, Linear Algebra Appl., **202** (1994), 163 – 192.

[GKO95] I.Gohberg, T.Kailath and V.Olshevsky, *Fast Gaussian elimination with partial pivoting for matrices with displacement structure*, Math. of Computation, **64** (1995), 1557-1576.

[GVL89] G. Golub and C, Van Loan, *Matrix Computations*, second edition, John Hopkins U. P., Baltimore, 1989.

[H95a] Heinig G. *Inversion of generalized Cauchy matrices and other classes of structured matrices*, in : Linear Algebra in Signal Processing (Proc. of IMA-92 Workshop on linear algebra in signal processing), IMA volumes in Mathematics and its Applications, vol. **69** (1995), 95 – 114.

[H95b] G.Heinig, *Private communication*, 1995.

[HR84] Heinig,G., Rost, K. (1984), *Algebraic methods for Toeplitz-like matrices and operators*, Operator Theory, vol. 13, Birkauser Verlag, Basel.

[Hig95] N.Higham, *Stability of the diagonal pivoting method with partial pivoting*, Preprint, 1995.

[Hu95] T.Huckle, *Symmetric Gaussian elimination for Cauchy-like matrices with application to positive definite Toeplitz matrices*, preprint, 1995.

[K80] T.Kailath, *Linear systems*, Prentice Hall, Englewood Cliffs, 1980.

[KKM79a] T.Kailath, S.Kung and M.Morf, *Displacement ranks of matrices and linear equations*, J. Math. Anal. and Appl., **68** (1979), 395-407.

[KKM79b] T.Kailath, S.Kung and M.Morf, *Displacement ranks of matrices and linear equations*, Bull. Amer. Math. Society **1** (1979), 769-773.

[KO95] T.Kailath and V.Olshevsky, *Displacement structure approach to Chebyshev–Vandermonde and related matrices*, Integral Equations and Operator Theory,**22** (1995), 65-92..

[KS92] T. Kailath and A.H.Sayed, *Fast algorithms for generalized displacement structures*, in Recent advances in mathematical theory of systems, control, networks and signal processing II, Proc. of the MTNS-91 (H.Kimura, S.Kodama, Eds.), Mita Press, Japan, 1992, 27 – 32.

[KS95b] T.Kailath and A.H.Sayed, *A look-ahead block Schur algorithm for Toeplitz-like matrices*, SIAM J. of Matrix Analysis Appl., **16** (1995), 388-414.

[KS95] T.Kailath and A.H.Sayed, *Displacement structure : Theory and Applications*, SIAM Review, **37** No.3 (1995), 297-386.

[LAPACK] E.Anderson, Z.Bai, C.Bishof, J.Demmel, J.Dongarra, J. Du Croz, A.Greenbaum, S.Hammarling, A.McKenney, S.Ostrouchov and D.Sorensen, *LAPACK User's Guide, Release 2.0*, SIAM, Philadelphia, PA, USA, second edition, 1995.

[P90] V.Pan, *On computations with dense structured matrices*, Math. of Computation, **55**, No. 191 (1990), 179 – 190.

[PD92] J. Pasupathy and R.A. Damodar, *The Gaussian Toeplitz matrix*, Linear Algebra and Appl., **171** (1992),133 – 147.

[SKLAC94] A.H.Sayed, T.Kailath, H. Lev-Ari and T.Constantinescu, *Recursive Solutions of Rational Interpolation Problems via Fast Matrix Factorization*, Integral Equations and Operator Theory, **20** (1994), 84-118.

[S17] I.Schur, *Über potenzreihen die im Inneren des Einheitskreises beschränkt sind*, Journal für die Reine und Angewandte Mathematik, **147** (1917), 205 – 232. English translation in *Operator Theory : Advances and Applications* (I.Gohberg. ed.), vol. **18**, 31 – 88, Birkhäuser, Boston, 1986.

[SB95] D.Sweet and R.Brent, *Error analysis of a partial pivoting method for structured matrices*, Advanced Signal processing algorithms, Proc of SPIE-1995, vol. **2563**, 266-280.

[V93] J.M. Varah, *The Prolate Matrix*, Linear Algebra and Appl., **187** (1993), 269 – 278.