A unified superfast divide-and-conquer algorithm for structured matrices over abstract fields

Victor Y. Pan*
Mathematics and Computer Science Department
Lehman College, CUNY, Bronx, NY 10468
internet: vpan@alpha.lehman.cuny.edu

June 9, 1999

Abstract

We propose a superfast divide-and-conquer algorithm that uses $2n-2$ random parameters, $O(n)$ memory space and $O((n\log^2 n)\log\log n)$ operations in a fixed arbitrary field in order to compute the rank and a basis for the null space of a structured $n \times n$ matrix X given with its short generator, as well as to solve a linear system $X\mathbf{y} = \mathbf{b}$ or to determine its inconsistency. If rank $X = n$, the algorithm also computes det X and a short generator of X^{-1}. The cost bounds cover correctness verification for the output but not the cost of the generation of random parameters. The algorithm gives a unified treatment of various classes of structured matrices including ones of Toeplitz, Hankel, Vandermonde and Cauchy types.

Key words: structured matrices, structured singular linear systems of equations, displacement rank, superfast divide-and-conquer algorithms, null space of a matrix, finite fields.

1991 Math. Subject Classification: 47B35, 68Q40, 65F05, 15A23, 15A33, 68Q25.

1 Introduction

1.1 Computations with structured matrices

The study of structured matrices is a classical subject with long history (cf. [C841], [Ho70], [KVM78], [KS95], [KS99]). Such matrices are omnipresent in signal and image processing, sciences and engineering. Some of the best known classical structured matrices are displayed in Table 1.

*Supported by NSF Grant CCR 9732206 and PSC-CUNY Award 669363

Table 1

Toeplitz matrices: $T = (t_{i-j})_{i,j=0}^{n-1}$	Hankel matrices: $\overline{H} = (\overline{h}_{i+j})_{i,j=0}^{n-1}$
Vandermonde and transposed Vandermonde matrices: $V = (t_i^j)_{i,j=0}^{n-1}$ and $V^T = (t_j^i)_{i,j=0}^{n-1}$	Cauchy matrices: $C = (\frac{1}{s_i - t_j})_{i,j=0}^{n-1}$

The most celebrated are *Toeplitz* and *Hankel* matrices, associated with operators of displacement (shift). *Vandermonde* and *Cauchy* matrices are associated with scaling operators (Vandermonde matrices – also with displacement operators) and are very popular and well recognized too, in particular for their application to celestial mechanics and algebraic decoding and their correlation to polynomial and rational interpolation and multipoint evaluation (see [PLST93], [PZHY97], [PACLS98], [OP98], [KS99], [OS99], [OP99] and bibliography therein).

There are various important classes of structured matrices, such as Bezoutians, Loewner, Pick, Sylvester and subresultant matrices, closely related to the above four classes. They are also associated with linear operators of displacement and/or scaling represented by a pair of matrices K and L and satisfy the Sylvester equation,

$$XK + LX = GH^T, \tag{1.1}$$

and/or its variations such as the Stein type equation,

$$X + LXM = GH^T, \tag{1.2}$$

where GH^T is a matrix of smaller rank (say, G, $H \in \mathbf{F}^{n \times l}$, assuming computations in a fixed field $\mathbf{F}$, whereas K, L, M, $X \in \mathbf{F}^{n \times n}$, for $l << n$). The matrices X satisfying (1.1), (1.2) can be expressed as bilinear or trilinear combinations of some basic structured matrices (such as diagonal, circulant and triangular Toeplitz matrices), each completely defined by K, L, and a single column of matrices G or H. The matrix pairs G, H are the memory efficient (short) generators for X. They represent $n \times n$ structured matrices X with $2ln$ entries of G and H and enable fast and/or superfast computations. (Hereafter, we will refer to arithmetic operations as *ops*, and we call computations with $n \times n$ matrices *fast* when they use $O(n^2)$ ops and *superfast* when they use $O(n)$ ops, up to polylogarithmic factors, versus order of n^3 for the same computations, like solving a linear system of n equations, but with general input matrices.) Depending on the choice of the scaling and/or displacement matrices K and L, the associated structured matrices may inherit the structure of Toeplitz, Hankel, Vandermonde or Cauchy matrices. For the sake of the unification of our study, we will call them *Toeplitz-like*, *Hankel-like*, *Vandermonde-like* and *Cauchy-like matrices* (cf. [BP94]), even though the nomenclature of "Vandermonde-like" appears in the literature also with distinct meaning. The first two classes have been studied most intensively [KKM79], [M80],

[BA80], [CKL-A87]. Some earlier treatment of the structures of Vandermonde and Cauchy types can be found in [GKK86] and [GKKL87]. Then a unified algorithmic treatment exploiting the correlation and transformations among various classes of structured matrices (by using, in particular, *Vandermonde multipliers)* to improve the known algorithms was proposed in [P90], and applications of this approach to fast Toeplitz-like computations were shown in [GKO95] and to polynomial and rational interpolation and multipoint evaluation in [PLST93], [PZHY97] and [OP99].

1.2 Our results

Our main goal in this paper is to present a unified, working in any field, superfast divide-and-conquer algorithm for all the cited classes of matrices (both singular and nonsingular), which allows its immediate extension to various other classes of structured matrices X satisfying the Sylvester equation (1.1).

The algorithm computes the rank, a basis for the null space of an $n \times n$ structured matrix X, and a solution to a linear system $X\mathbf{y} = \mathbf{b}$ (or reports its inconsistency). If X is nonsigular, the algorithm also outputs det X and a short generator for X^{-1}. To counter possible singularities, the algorithm uses $2n-2$ random parameters, versus order of $n \log n$ in the superfast algorithm of [K95] (devised specially for the Toeplitz-like case), which also allows failure with double probability versus ours. Our algorithm requires linear space of $O(n)$ memory words. Being superfast, the algorithm runs in time $O((n \log^2 n) \log\log n)$ over any field and in time $O(n \log^2 n)$ if the field supports FFT. The bounds cover the cost of the computation of the output and testing its correctness but not the cost of the generation of random parameters. Unless we deal with nonsingular matrices, standard minor increase of the complexity estimates is required (see Remarks 7.1 and 7.2 in section 7) to bound from above by a fixed $\delta > 0$ the probability of failure (this requires the time increase by factor $O(\log(1/\delta)))$ and to handle the computations in the field of small cardinality c (this requires to increase by factor $O(\log(n^2/c))$ the memory space bound and by factor $O(\log(n^2/c) \log\log(n^2/c))$ the time bound).

1.3 Technical novelty and comparison with the known results

The main novelties of our work is the detailed and *unified treatment* of structured matrices of various classes and the elaboration of the computations with such matrices in finite fields and in the singular case. So far the latter cases (of finite fields and singular input) have been treated only for the Toeplitz-like matrices, in [K95], but this does not cover various important applications in particular to signal processing. For instance, [OS99] showed that the computations of a vector from the null space of a matrix having structure of Vandermonde type is a bottleneck of various algebraic and algebraic-geometric decoding computations and yielded acceleration by presenting a fast algorithm. Superfast solution of the same problem is immediately achieved as a special case of our algorithm.

Furthermore, our unified presentation should clarify the advantages of the divide-and-conquer approach to computations with structured matrices. We point out various directions for improvement, which can be further elaborated to give upper hand to the divide-and-conquer approach in its practical competition with other algorithims.

It is possible to view our work as an extension of the divide-and-conquer algorithms known for Toeplitz-like matrices [M80], [BA80], [K95]. As a goal, such an extension, assuming computations with positive definite input matrices in the complex field and with no singularities involved was stated in [OP98], where it was proposed to pursue the goal based on similar computations for rational interpolation. Although this approach has lead in [OP98] to highly important interpolation applications, in our present work we prefer more direct matrix approach, which enables simultaneously acceleration by factor $\log n$ and simplification and clarification versus [OP98] as well as achieving generality and universality of the resulting algorithm. In particular, the construction of the superfast structured matrix algorithms has become much clearer when we unified them as the variations of the divide-and-conquer algorithm of [S69], [AHU74], which reduced various computations with general matrices to matrix multiplications, additions and subtractions.

It remained to elaborate these operations in terms of the associated short generators. Such a task entailed many technicalities, which included the computation of the generators of the matrix products, inverses, block submatrices, and Schur complements (in terms of their generators), as well as tracing the matrix structure back and forth between the input matrix and all the matrices of smaller sizes involved (to solve the latter task, we examined the divide-and-conquer algorithm twice – as the *descending and lifting recursive processes*) and the techniques enabling the computations in the presence of singularities (over abstract fields). We solved the cited main task and its various subtasks in a unified way and in more details and greater generality than in the known literature.

Among the techniques that we developed and used, we would like to single out our transformations of various classes of structured matrices into each other and our novel randomization algorithms, which not only counter the problems of singularities for all listed classes of structured matrices over abstract fields but also, as a by-product, enabled simple additional acceleration by logarithmic factor for the Cauchy-like and Vandermonde-like computations over the fields containing the n-th roots of 1, for n being a power of 2. Note that at this stage we propose new *Cauchy-like multipliers* (as well as new f-circulant multipliers) as preconditioners, in addition to the known triangular Toeplitz multipliers.

We hope that some of the techniques developed are of independent interest for the study of structured matrices.

1.4 Organization of our paper

We organize our paper as follows. In the next section, we describe the known divide-and-conquer algorithm for general matrices, with its extension to the singular case. In section 3 we define four basic classes of structured matrices as well as various associated linear operators of scaling and displacement and memory efficient generators for the matrices. In section 4 we prove some basic results that enable manipulation with the generators instead of matrices. In section 5 we specify the divide-and-conquer algorithm for structured matrices and estimate its memory space and running time. In section 6 we accelerate the computations in the Vandermonde-like and Cauchy-like cases by factor $\log n$. In sections 7 and 8 we use randomization to treat singularities in the case of general and structured matrices, respectively. In section 9 we cover in some detail the stage of the computation of Schur's complements.

2 A divide-and-conquer algorithm for general matrices

Hereafter, $\mathbf{F}$ denotes a fixed field.

Definition 2.1 *We write $I = I_m \in \mathbf{F}^{m\times m}$ for the $m \times m$ identity matrix, 0 for a null matrix of appropriate size, W^T and W^H for the transpose and the Hermitian transpose of a matrix or a vector W, respectively.*

Definition 2.2 *$W^{(k)} \in \mathbf{F}^{k\times k}$ is the $k \times k$* leading principal *(northwestern)* submatrix *of an $m \times n$ matrix W, $k = 1, \ldots, \min(m, n)$. A matrix W of rank ρ has* generic rank profile *if its $k \times k$ leading principal submatrices $W^{(k)}$ are nonsingular for $k = 1, \ldots, \rho$. A matrix is* strongly nonsingular *if it is both nonsingular and has generic rank profile.*

We will first describe a divide-and-conquer algorithm for the factorization of a general $n \times n$ strongly nonsingular matrix,

$$X = \begin{pmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{pmatrix}, \tag{2.1}$$

where $X_{11} = X^{(k)}$ is a $k \times k$ matrix. The $(n-k) \times (n-k)$ matrix,

$$S = X_{22} - X_{21}X_{11}^{-1}X_{12}, \tag{2.2}$$

is called the *Schur complement of* X_{11} *in* X and is denoted by $S(X_{11}, X)$. We have

$$X = \begin{pmatrix} I & 0 \\ X_{21}X_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} X_{11} & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & X_{11}^{-1}X_{12} \\ 0 & I \end{pmatrix}, \tag{2.3}$$

$$X^{-1} = \begin{pmatrix} I & -X_{11}^{-1}X_{12} \\ 0 & I \end{pmatrix} \begin{pmatrix} X_{11}^{-1} & 0 \\ 0 & S^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -X_{21}X_{11}^{-1} & I \end{pmatrix}. \tag{2.4}$$

Factorization (2.3) represents the block Gauss-Jordan elimination applied to the 2×2 block matrix X of (2.1). Since the matrix X_{11} is strongly nonsingular, the Schur complement matrix S can be obtained in $n - k$ steps of Gaussian elimination. By expanding (2.3), we obtain

$$X^{-1} = \begin{pmatrix} X_{11}^{-1} + X_{11}^{-1}X_{12}S^{-1}X_{21}X_{11}^{-1} & -X_{11}^{-1}X_{12}S^{-1} \\ -S^{-1}X_{21}X_{11}^{-1} & S^{-1} \end{pmatrix}. \tag{2.5}$$

Proposition 2.1 *If (2.3) holds, then* $\det X = (\det X_{11}) \det S$.

Proposition 2.2 *[*GL96*]. If X is strongly nonsingular, so are X_{11} and S.*

Proposition 2.3 *(cf. (2.5)). Let X and X_{11} be nonsingular matrices of (2.1) and let $S = S(X_{11}, X)$. Then S^{-1} forms the respective southeastern block of X^{-1}.*

Proposition 2.4 *[*BP94*], [*GL96*]. Suppose that the Schur complements $S = S(X^{(k)}, X)$ and $S_1 = S(S^{(h)}, S)$ are defined, that is, the matrices $X^{(k)}$ and $S^{(h)}$ are nonsingular. Then $S_1 = S(X^{(k+h)}, X)$.*

Due to Proposition 2.2, we may extend factorization (2.3) from X to X_{11} and S and then recursively continue such a *descending process* until we arrive at 1×1 matrices (compare [S69], [AHU74]). In actual computation, we apply *lifting process* that begins with the inversion of the 1×1 matrix $X^{(1)}$. Then we compute and invert its 1×1 Schur complement S_1 in the 2×2 matrix $X^{(2)}$ (this defines the factorization of $X^{(2)}$ and of its inverse), compute the inverse of $X^{(2)}$ and its 2×2 Schur complement S_2 in the 4×4 matrix $X^{(4)}$, and so on. In other words, we recursively proceed *bottom up*, that is, we invert 1×1 matrices and, other than that, use only matrix multiplications and subtractions to compute all matrices specified in the recursive descending process, until we finally arrive at X^{-1}. The algorithm emulates Gaussian elimination steps except that it combines their scalar multiplications and subtractions into the similar operations with matrix blocks. The entire computation will be called the *CRF* (or *complete recursive factorization*) of X. In the *balanced* CRFs, X_{11} of (2.1) is a $\lfloor \frac{n}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor$ submatrix of X, and similar balancing is maintained in all subsequent recursive steps. The balanced CRF has depth at most $d = \lceil \log_2 n \rceil$.

Let us summarize our description in the form of a recursive algorithm.

Algorithm 2.1. *Recursive triangular factorization and inversion of a strongly nonsingular matrix.*

Input: a field $\mathbf{F}$ and a strongly nonsingular $n \times n$ matrix X of (2.1), $X \in \mathbf{F}^{n \times n}$.

Output: the balanced CRF of X, including the matrix X^{-1}.

Computations:

1. Apply Algorithm 2.1 to the matrix X_{11} (replacing X as its input) in order to compute the balanced CRF of X_{11} (including X_{11}^{-1}). Compute X_{11}^{-1} via a single division if X_{11} is a 1×1 matrix or, otherwise, based on the extension of (2.4) to the factorization of X_{11}^{-1}.

2. Compute the Schur complement $S = X_{22} - X_{21} X_{11}^{-1} X_{12}$.

3. Apply Algorithm 2.1 to the matrix S (replacing X as its input) to compute the balanced CRF of S (including S^{-1}).

4. Compute X^{-1} from (2.4).

As a by-product, Algorithm 2.1 may immediately compute the vector $\mathbf{y} = X^{-1}\mathbf{b}$ of the solution to a linear system $X\mathbf{y} = \mathbf{b}$ for a given vector $\mathbf{b}$. If we also seek $\det X$, then it suffices to add the request for computing $\det X_{11}$, $\det S$, and $\det X$ (see Proposition 2.1) at stages 1, 3, and 4, respectively.

It is well known [S69], [AHU74], [BP94], p.99, that the complexity of these computations (in terms of the number of ops involved) satisfies

$$RF(n) = O(\sum_{i=1}^{h} 2^i M(n/2^i), \tag{2.6}$$

where w.l.o.g. we assume $n = 2^h$ for an integer h, and $M(n)$ is the complexity of $n \times n$ matrix multiplication. Theoretically, for general matrices, $M(n) = O(n^{2.376})$, though practical algorithms support the bounds $M(n) = O(n^{\beta})$, for $2.8 < \rho \leq 3$ [GL96], [K,a]. For structured matrices, we have $M(n) = O(n \log^c n)$, $c < 3$, and we deduce from (2.6) that

$$RF(n) = O(M(n)) \ \text{ if } \ M(n) \geq n^{1+d}, \ d > 0, \tag{2.7}$$

$$RF(n) = O(M(n) \log n) \ \text{ if } \ M(n) = O(n \log^c n) \tag{2.8}$$

for a constant c.

Now, let X have generic rank profile. In this case we apply **generalized Algorithm 2.1,** which includes a counter for the number of the inversions of 1×1 matrices involved (that is, for the number of divisions). If division by 0 occurs, the computations stop, and at this point, we have ρ =rank X in the counter and a CRF of $X^{(\rho)}$ available, which immediately gives us $(X^{(\rho)})^{-1}$.

Let us also show a simple extension (see e.g. [BP94], p. 110). Recall (2.1) with $X_{11} = X^{(\rho)}$ and write

$$F = \begin{pmatrix} I_\rho & -X_{11}^{-1} X_{12} \\ 0 & I_{n-\rho} \end{pmatrix}, \quad N = F \begin{pmatrix} 0 \\ I_{n-\rho} \end{pmatrix}. \tag{2.9}$$

Then

$$XF = \begin{pmatrix} X_{11} & 0 \\ X_{21} & 0 \end{pmatrix}, \tag{2.10}$$

the columns of N (which are the last $n-\rho$ columns of the matrix F) form a basis for the null space of X, and the substitution $\mathbf{y} = F\mathbf{z}$ reduces the solution of a linear system $X\mathbf{y} = \mathbf{b}$ (or the determination of its inconsistency) to the case of the system $(XF)\mathbf{z} = \mathbf{b}$, for which the problem is simple because we have (2.10) and already know X_{11}^{-1}.

Definition 2.3 *A* full output set *of generalized Algorithm 2.1 consists of the set of the matrices of a CRF of a largest nonsingular submatrix of X, complemented by the rank ρ of X, a basis for the null space of X, a solution $\mathbf{y}$ to the linear system $X\mathbf{y} = \mathbf{b}$ for a given vector $\mathbf{b}$ (or the determination of its inconsistency), and if $\rho = n$, then also by X^{-1} and det X. Without the CRF (but including a largest nonsingular submatrix), this is a partial output set.*

The next extension of (2.7), (2.8) to the computation of a full output set is immediately verified:

Theorem 2.1 *For an $n \times n$ matrix X having generic rank profile, generalized Algorithm 2.1 computes its full output set at the cost bounded according to (2.7), (2.8).*

One may extend the bound (2.7) of Theorem 2.1 to an arbitrary matrix X by using pivoting [IMH82], which only adds $O(n^2)$ comparison and some permutations of matrix rows and columns, so that the overall cost bound is still dominated by $O(M(n))$ under (2.7).

For a nonsingular real or complex matrix X, one may apply symmetrization instead of pivoting in order to compute (within the cost bounds (2.7), (2.8)) the balanced CRFs of the strongly nonsingular matrices $X^H X$ and/or XX^H (at the price of possible squaring the condition number of X), and then obtain X^{-1} as $(X^H X)^{-1} X^H$ or as $X^H (XX^H)^{-1}$ and $(\det X)^2 = \det(X^H X) = \det(XX^H)$. Symmetrization has no effect over the fields of positive characteristic as well as in the case of a singular input matrix X.

3 Structured matrices and their generators (definitions)

Definition 3.1 *$T = (t_{i,j})_{i,j=0}^{n-1} \in \mathbf{F}^{n\times n}$ is a* Toeplitz matrix *if $t_{i+1,j+1} = t_{i,j}, i, j = 0, \dots, n-2$. $\overline{H} = (\overline{h}_{i,j})_{i,j=0}^{n-1} \in \mathbf{F}^{n\times n}$ is a* Hankel matrix *if $\overline{h}_{i+1,j-1} = \overline{h}_{i,j}$, $i = 0, \dots, n-2$; $j = 1, \dots, n-1$. $V(\mathbf{t}) = (t_i^j)_{i,j=0}^{n-1} \in \mathbf{F}^{n\times n}$ is a* Vandermonde matrix , *and $C(\mathbf{s}, \mathbf{t}) = (\frac{1}{s_i - t_j})_{i,j=0}^{n-1} \in \mathbf{F}^{n\times n}$ is a* Cauchy matrix, *for any pair of vectors $\mathbf{s} = (s_i)_{i=0}^{n-1}$, $\mathbf{t} = (t_j)_{j=0}^{n-1}$ where $s_i \neq t_j$ for every pair (i, j). (Many authors use the name "Vandermonde matrix" for $V^T(\mathbf{t})$.)*

Definition 3.2 *$\mathbf{e}_i \in \mathbf{F}^{n\times 1}$ is the i-th* coordinate vector *having 1 as its i-th coordinate and 0 as all its other coordinates. $\mathbf{0}$ is the* null vector *of appropriate dimension. $J = (\mathbf{e}_i)_{i=n-1}^{0}$ is the* reflection matrix, *$J\mathbf{v} = (v_i)_{i=n-1}^{0}$ for any vector $\mathbf{v} = (v_i)_{i=0}^{n-1}$. $D(\mathbf{v}) = \mathrm{diag}(v_0, \dots, v_{n-1}) = (v_i \mathbf{e}_i)_{i=0}^{n-1} \in \mathbf{F}^{n\times n}$ is the* diagonal matrix *with diagonal entries $v_0, \dots, v_{n-1}$. $\mathbf{v}^k = (v_i^k)_{i=0}^{n-1}$, for $\mathbf{v} = (v_i)_{i=0}^{n-1}$ and an integer k.*

Definition 3.3 $C_f = (\mathbf{e}_1, \ldots, \mathbf{e}_{n-1}, f\mathbf{e}_0) \in \mathbf{F}^{n\times n}$ *(for a scalar f) is the* unit f-circulant matrix, *called the* unit circulant *if $f = 1$. ($C_f\mathbf{v} = (fv_{n-1}, v_0, \ldots, v_{n-2})$ for $\mathbf{v} = (v_i)_{i=0}^{n-1}$.) $C_f(\mathbf{v}) = \sum_{i=0}^{n-1} v_i C_f^i$ is an f-*circulant matrix. *$C_1(\mathbf{v})$ is a* circulant matrix, *and $C_0(\mathbf{v})$ is a lower triangular Toeplitz matrix.*

Remark 3.1 *To show more options in choosing the operators associated with structured matrices, we allow any f when we define the matrices C_f and $C_f(\mathbf{v})$. In our presentation in this paper, however, we could have always stayed with the triangular Toeplitz case $f = 0$ or with $f = 1$ (the circulant case) and later on (in (3.2), Remarks 3.3 and 6.1, and Proposition 6.3) $e = -1$ for the matrices $C_e(\mathbf{v})$ (the skew circulant case).*

Fact 3.1 $J^2 = I,\ C_0^n = 0,\ C_f^{-1} = C_{1/f}^T$ *for* $f \neq 0,\ JC_fJ = C_f^T$ *for any* f. $JD(\mathbf{v})J = D(J\mathbf{v})$ *for any* $\mathbf{v}$.

Fact 3.2 *TJ and JT are Hankel matrices if T is a Toeplitz matrix, and vice versa, $\overline{H}J$ and $J\overline{H}$ are Toeplitz matrices if $\overline{H}$ is a Hankel matrix.*

Next, we will show some generalizations of the classes of the structured matrices of Definition 2.1 in the form of (scaled or shifted) bilinear or trilinear combinations X of the structured matrices of Definition 3.1 and diagonal matrices. Equivalently, such matrices are the solutions to Sylvester's basic matrix equation (1.1), which we now display again, for convenience:

$$XK + LX = GH^T. \tag{3.1}$$

Here $K, L \in \mathbf{F}^{n\times n}$ and $G, H \in \mathbf{F}^{n\times \ell}$ are fixed matrices, and ℓ is small relative to n. We will write $T(G,H), V(\mathbf{t},G,H)$ (with some subscripts) and $C(\mathbf{s},\mathbf{t},G,H)$ for matrices whose structure generalizes one of $T, V(\mathbf{t})$ and $C(\mathbf{s},\mathbf{t})$, respectively.

Definition 3.4 *Given two scalars e and f, $ef \neq 1$, two vectors $\mathbf{s}, \mathbf{t} \in \mathbf{F}^{n\times 1}$, with $s_i \neq t_j$ for all pairs (i,j), and a pair of $n \times \ell$ matrices $G = (\mathbf{g}_i)_{i=1}^{\ell}$, $H = (\mathbf{h}_i)_{i=1}^{\ell} \in \mathbf{F}^{n\times\ell}$, we write*

$$T_f = T_f(G,H) = (fC_f(T_f\mathbf{e}_0) + \frac{1}{1-ef}\sum_{k=1}^{\ell} C_f(\mathbf{g}_k)C_e^T(\mathbf{h}_k))C_{1/f}^T, \ \text{for } f \neq 0, \tag{3.2}$$

$$T_0 = T_0(G,H) = C_0(T_0\mathbf{e}_0) + \sum_{k=1}^{\ell} C_0(\mathbf{g}_k)C_0^T(C_0\mathbf{h}_k), \tag{3.3}$$

$$V_f(\mathbf{t},G,H) = D((\mathbf{t} - f\mathbf{t}^{n+1})^{-1})\sum_{k=1}^{\ell} D(\mathbf{g}_k)V(\mathbf{t})C_f^T(\mathbf{h}_k),\ ft_i^{n+1} \neq t_i,\ i = 0, \ldots, n-1, \tag{3.4}$$

$$C(\mathbf{s},\mathbf{t},G,H) = \sum_{k=1}^{\ell} D(\mathbf{g}_k)C(\mathbf{s},\mathbf{t})D(\mathbf{h}_k). \tag{3.5}$$

*The matrix pair (G,H) is called a (K,L)-*generator *(or just* a generator) of length ℓ *for a matrix X where $K = -L = C_f$ for $X = T_f$ and for any scalar f; $K = -C_f^T$, $L = D^{-1}(\mathbf{t})$ for $X = V_f(\mathbf{t},G,H)$, and $K = -D(\mathbf{t}), L = D(\mathbf{s})$ for $X = C(\mathbf{s},\mathbf{t},G,H)$. For a fixed (K,L), the minimum ℓ in all (K,L)-generators of X is called the (K,L)-*rank *or a* generator rank *of X and is denoted by $r_{K,L}(X)$. The operator $X \to XK + LX$ is called a* basic operator *for X, and the pair (K,L) is called a* basic matrix pair *for X.*

The latter definition is motivated by the following results, defining matrix structure in terms of the associated linear operators of scaling and displacement (shift) (cf. Remark 3.5 at the end of this section).

Theorem 3.1 *A matrix X satisfies Sylvester's matrix equation (3.1) if (G,H) is a (K,L)-generator of X for the triples X,K,L defined above, that is, if*

a) $T_f C_f - C_f T_f = GH^T$ *under (3.2), (3.3),*
b) $(JT_f)C_f - C_f^T(JT_f) = (JG)H^T$, $(T_f J)C_f^T - C_f(T_f J) = GH^T J$, *under (3.2), (3.3),*
c) $D^{-1}(\mathbf{t})V_f(\mathbf{t},G,H) - V_f(\mathbf{t},G,H)C_f^T = GH^T$ *under (3.4), and*
d) $D(\mathbf{s})C(\mathbf{s},\mathbf{t},G,H) - C(\mathbf{s},\mathbf{t},G,H)D(\mathbf{t}) = GH^T$ *under (3.5).*
Furthermore, $r_{K,L}(X) = \text{rank } (XK + LX)$ *in all these cases.*
Proof. Parts a), c), d) of Theorem 3.1 easily follow from Theorems 1.1, 2.1 and 3.1 of [GO94] (cf. also [GO92]) and Theorem 2.11.3a) of [BP94]. Part b) follows from part a) and Fact 3.1. □

We immediately obtain a short (K,L)-generators for the matrices $T, H, V(\mathbf{t})$ and $C(\mathbf{s},\mathbf{t})$ of Definition 3.1, and for all scalars f we deduce that $r_{C_f,-C_f}(T) \le 2$, $r_{C_f,-C_f^T}(JT) \le 2$, $r_{C_f^T,-C_f}(TJ) \le 2$, $r_{-C_f^T,D^{-1}(\mathbf{t})}(V(\mathbf{t})) \le 1$, and $r_{-D(\mathbf{t}),D(\mathbf{s})}(C(\mathbf{s},\mathbf{t})) \le 1$. By generalizing these observations, we will say that for all scalars f the matrices T_f of (3.2), (3.3) are *Toeplitz-like*, JT_f and $T_f J$ are *Hankel-like*, $V_f(\mathbf{t},G,H)$ of (3.4) are *Vandermonde-like*, and $C(\mathbf{s},\mathbf{t},G,H)$ of (3.5) are *Cauchy-like* if ℓ is small relatively to n, say if ℓ is bounded by a small fixed constant.

Remark 3.2 *[BP94], [GO94]. The solution to Sylvester's matrix equation (3.1) is unique for a fixed generator (G,H) and each basic pair (K,L) specified above. In the Vandermonde-like and Cauchy-like cases of (3.4), (3.5), there exists such a (unique) solution for any fixed pair of matrices $G,H \in \mathbf{F}^{n\times r}$. In the Toeplitz/Hankel-like case of (3.2), (3.3), there exists a (unique) solution if and only if $\sum_{k=1}^{l} C_f(\mathbf{g})C_{1/f}(\mathbf{h}_k)^T = 0$ for $f \ne 0$ or $\sum_{k=1}^{l} C_f^T(\mathbf{g}_k)\mathbf{h}_k = \sum_{k=1}^{l} C_f(J\mathbf{h}_k)\mathbf{g}_k = \mathbf{0}$ for $f = 0$. Thus, one may take the dual point of view, that is, define the classes of structured matrices by equation (3.1) and then apply Theorem 3.1 as a* recovery theorem, *defining some explicit expression for the solution matrix X satisfying this equation.*

Structured matrices X of (3.2)-(3.5) can be completely represented by the $2\ell n$ entries of their (K,L)-generators (G,H), rather than by their own n^2 entries (in the Toeplitz/Hankel-like case, we need in addition the n entries of the first or last column or row of the matrix). Such a *compressed representation* of a matrix is not unique. Moreover, ℓ may exceed $r_{K,L}(X)$, but this can be repaired by using our *generator compression techniques*:

Fact 3.3 *Let X stand for $T_f, V_f(\mathbf{t},G,H)$ or $C(\mathbf{s},\mathbf{t},G,H)$ of (3.2)-(3.5). Let a (K,L)-generator (G,H) of a length l for a matrix X and the (K,L)-rank of X, $r = r_{K,L}(X)$, be given as an input, together with f and $T_f\mathbf{e}_0$, f and $\mathbf{t}$, or $\mathbf{s}$ and $\mathbf{t}$, respectively. Then it is sufficient to use $O(\ell^2 n)$ ops in order to compute a (K,L)-generator of length r for X.*

Proof. See Proposition A.6 of [P92] or Problem 2.2.11b of [BP94]. □

Remark 3.3 *Theorem 3.1 can be used as a springboard for the definition of the same or closely related classes of structured matrices based on the Sylvester equation (3.1) with* *distinct basic pairs* (K, L). *For instance, the matrices* $C_f - C_e$ *have rank* 1 *for* $e \neq f$, *and this immediately implies close correlation between the classes of* $\{T_f(G, H)\}$ *and* $\{T_e(G, H)\}$ *of Toeplitz-like matrices as well as between* $\{V_f(\mathbf{t}, G, H)\}$ *and* $\{V_e(\mathbf{t}, G, H)\}$ *for distinct* e *and* f, *for a fixed* l, *and for varying* $G, H \in \mathbf{F}^{n \times l}$ *(see Remark 6.1 in section 6). Likewise, parts a) and b) of Theorem 3.1 can be immediately extended to define a* $(C_e^T, -C_f)$*-generator for* T_f; *a* $(C_e, -C_f^T)$*-generator for* JT_f, *and a* $(C_e^T, -C_f)$*-generator for* $T_f J$.

Further variations of the matrix equation (3.1) can be obtained by the transposition of the matrices on its both sides, by its multiplication by a scalar, e.g. by -1, and/or (cf. [GKO95]) by its pre- and post-multiplications by some selected matrices. In this way, we obtain from (3.1) that

$$X^T L^T + K^T X^T = H G^T, \tag{3.6}$$

$$\hat{X}\hat{K} + \hat{L}\hat{X} = \hat{G}\hat{H}^T, \tag{3.7}$$

where $\hat{X} = UXW$, $\hat{K} = W^{-1}KW$, $\hat{L} = ULU^{-1}$, $\hat{G} = UG$, $\hat{H} = H^T W$, and U and W is any fixed pair of nonsingular matrices. (Observe that the spectra of the matrices K and L are preserved in the transition to $\hat{K}$ and $\hat{L}$.) In particular, for $U = W = J$, we obtain

$$(JXJ)(JKJ) + (JLJ)(JXJ) = JGH^T J. \tag{3.8}$$

We may also combine (3.6) and (3.7) together.

In all these variations, the representations (3.2)-(3.5) of a matrix X are immediately extended to the matrices X^T, $\hat{X}$ and $\hat{X}^T$, in particular, *we immediately represent compactly the matrices* UT_fW, $UV_f(\mathbf{t}, G, H)W$, $UV_f^T(\mathbf{t}, G, H)W$, $UC(\mathbf{s}, \mathbf{t}, G, H)W$ *for any fixed pair of nonsingular matrices* U *and* W.

Tables 2-4 summarize some particular classes of structured matrices together with the associated basic matrix pairs (K, L), which we obtained based on Theorem 3.1, equations (3.6) and (3.8) and Fact 3.2 and will use in the next sections. Some other variations of K, L of (3.1) and the respective extensions of Theorem 3.1 can be found in [BP94], pp. 187-188, [KO96], [KO97], [KS99].

Table 2, *Toeplitz/Hankel-like matrices and their basic matrix pairs*

X	T_f	T_f^T	JT_f	$T_f J$
K	C_f	$-C_f^T$	C_f	C_f^T
L	$-C_f$	C_f^T	$-C_f^T$	$-C_f$

Table 3, *Vandermonde-like matrices and their basic matrix pairs*

X	$V_f(\mathbf{t}, G, H)$	$V_f^T(\mathbf{t}, G, H)$	$JV_f(\mathbf{t}, G, H)J$	$JV_f^T(\mathbf{t}, G, H)J$
K	$-C_f^T$	$D^{-1}(\mathbf{t})$	$-C_f$	$D^{-1}(J\mathbf{t})$
L	$D^{-1}(\mathbf{t})$	$-C_f$	$D^{-1}(J\mathbf{t})$	$-C_f^T$

Table 4, *Cauchy-like matrices and their basic matrix pairs*

X	$C(\mathbf{s}, \mathbf{t}, G, H)$	$C^T(\mathbf{s}, \mathbf{t}, G, H)$
K	$-D(\mathbf{t})$	$D(\mathbf{s})$
L	$D(\mathbf{s})$	$-D(\mathbf{t})$

Definition 3.5 *The basic matrix pairs* (K, L) *and* (L, K) *are called* dual *and if, in addition,* $K = -L$, *then* self-dual.

Remark 3.4 *All basic matrix pairs* (K, L) *appear in Tables 2-4 together with their dual pairs* (L, K) *(up to scaling by* -1*). The basic matrix pairs* (K, L) *of Table 2 are self-dual.*

Remark 3.5 *For* $K = -L = C_f$ *and* $K = -L = C_f^T$*, the basic operator* $X \to XK + LX$ *maps the matrix* X *to the difference between the two displacements (shifts) of* X *(into two directions orthogonal to each other).* $r_{K,L}(X)$*, the rank of such a difference, is commonly called the* displacement rank *of* X*. The nomenclature is due to the seminal pioneering paper [*KKM79*], where the Toeplitz type structure was first formally introduced (based on the Stein type equations (1.2) with* $K = -M^T = C_0^T$ *and* $K = -M^T = C_0$*). Our Definition 3.4, Theorem 3.1 and Propositions 4.2 and 4.4 of section 4 extend the respecitive fundamental results of [*KKM79*], whereas the concept of generators extends the one of [*CKL-A87*].*

4 Operations with matrices represented by their short (K, L)-generators

Our next goal is to accelerate the computations with structured matrices (versus ones with general matrices) by relying on compact representation of these matrices via their (K, L)-generators and on the following well-known estimates.

Theorem 4.1 *Let us write* $C_{f,v}(n)$*,* $T_v(n)$*,* $H_v(n), V_v(n)$*,* $V_v^T(n)$ *and* $C_v(n)$ *to denote the numbers of ops required to multiply (over a field* $\mathbf{F}$*) an* $n \times n$ f*-circulant, Toeplitz, Hankel, Vandermonde, transposed Vandermonde, and Cauchy matrices, respectively, by a vector. Then we have* $C_{f,v}(n) = O((n \log n) \log \log n)$*,* $T_v(n) = H_v(n) = O((n \log n) \log \log n)$*,* $V_v(n) = V_v^T(n) = O((n \log^2 n) \log \log n)$*,* $C_v(n) = O((n \log^2 n) \log \log n)$*. The factor* $\log \log n$ *can be removed if* $\mathbf{F}$ *supports discrete Fourier transforms at the order of* n *points.*

Proof. $V_v^T(n) = V_v(n)$ by Tellegen's theorem (see [PSD70]). For other estimates, see e.g. [BP94], [GO94]. □

Corollary 4.1 *For three given scalars* e, f*, and* $(1 - ef)^{-1}$*, two matrices* $G, H \in \mathbf{F}^{n \times l}$*, and three vectors* $\mathbf{s}, \mathbf{t}$*, and* $\mathbf{t} - f\mathbf{t}^{n+1}$*, let the matrices* T_f*,* T_0*,* $V_f(\mathbf{t}, G, H)$ *and* $C(\mathbf{s}, \mathbf{t}, G, H)$ *be represented in the form (3.2)-(3.5). Then such matrices can be multiplied by a vector by using at most* $lC_{e,v}(n) + (l+1)C_{f,v}(n) + (l+2)n + 1$*,* $(2l+1)C_{0,v} + nl$*,* $(C_{f,v}(n) + V_v(n) + 2n)l$*, and* $C_v(n)l + (3l - 1)n$ *ops, respectively.*

The next simple results show that the linear combinations, products and inverses of matrices as well as the matrix blocks inherit the matrix structure of the input. In the next section, we will specify and exemplify the applications of these results to the design of superfast algorithms for computations with structured matrices.

Proposition 4.1 *(a generator of a linear combination). Let* $XK + LX = G_X H_X^T$*,* $YK + LY = G_Y H_Y^T$*. Then* $(X + aY)K + L(X + aY) = (G_X, aG_Y)) \begin{pmatrix} H_X \\ H_Y^T \end{pmatrix}$ *for a scalar* a*.*

The next result defines a generator of the matrix product and will enable transformations among various classes of structured matrices.

Proposition 4.2 *(cf. [*P90*]). Let* $-XW + MX = G_X H_X^T, YU + WY = G_Y H_Y^T$*. Then* $(XY)U + M(XY) = (G_X, XG_Y) \begin{pmatrix} H_X^T Y \\ H_Y^T \end{pmatrix}$.

Proof. $(XY)U + M(XY) = X(YU + WY) + (-XW + MX)Y$. □

For convenience, we summarize in Table 5 the correlation among the basic matrix pairs (K, L) for the matrices X, Y, and XY of Proposition 4.2.

Table 5, *Basic matrix pair for the matrix product*

	X	Y	XY
K	$-W$	U	U
L	M	W	M

Proposition 4.3 *(generators of blocks). Let $n = 2m$ be even. Let us write $K = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}$, $L = \begin{pmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{pmatrix}$, $X = \begin{pmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{pmatrix}$, $C_f = \begin{pmatrix} C_0^{(m)} & fU \\ U & C_0^{(m)} \end{pmatrix}$, $G = \begin{pmatrix} G_1 \\ G_2 \end{pmatrix}$, $H = \begin{pmatrix} H_1 \\ H_2 \end{pmatrix}$, $D(\mathbf{v}) = \begin{pmatrix} D_1(\mathbf{v}) & 0 \\ 0 & D_2(\mathbf{v}) \end{pmatrix}$, where $K_{i,j}$, $L_{i,j}$, $X_{i,j}$, and $D_i(\mathbf{v})$, for $i, j = 1, 2$, as well as $C_0^{(m)}$ and U are $m \times m$ matrices, from $\mathbf{F}^{m\times m}$; G_i and H_i, for $i = 1, 2$, are $m \times l$ matrices, from $\mathbf{F}^{m\times l}$; $U = \mathbf{e}_0^{(m)}(\mathbf{e}_{m-1}^{(m)})^T$ is a rank 1 matrix; $\mathbf{e}_0^{(m)}$, $\mathbf{e}_{m-1}^{(m)}$ are unit coordinate vectors, from $\mathbf{F}^{m\times 1}$. Let (3.1) hold. Then $X_{ij}\overline{K}_{jj} + \overline{L}_{ii}X_{ij} = G_iH_j^T + R_{ij}$, $i, j = 1, 2$, where*

a)

$$R_{11} = (fX_{11} - X_{12})U + fU(X_{21} - X_{11}),$$
$$R_{12} = f(X_{12} - X_{11})U + fU(X_{22} - X_{12}),$$
$$R_{21} = (fX_{21} - X_{22})U + U(X_{11} - fX_{21}),$$
$$R_{22} = f(X_{22} - X_{21})U + U(X_{12} - fX_{22}),$$

if $X = T_f$, $K = C_f$, $L = -C_f$, $\overline{K}_{jj} = K_{jj} + fU = -\overline{L}_{ii} = -L_{ii} + fU = C_0^{(m)} + fU$,

b)

$$R_{11} = f(X_{12} - X_{11})U^T,\ R_{12} = (X_{11} - fX_{12})U^T,$$
$$R_{21} = f(X_{22} - X_{21})U^T,\ R_{22} = (X_{21} - fX_{22})U^T,$$

if $X = V_f(\mathbf{t}, G, H)$, $K = -C_f^T$, $L = D^{-1}(\mathbf{t})$, $\overline{K}_{jj} = K_{jj} - fU^T = -(C_0^{(m)} + fU)^T, \overline{L}_{ii} = L_{ii} = D_i(\mathbf{t})$,

c)

$$R_{11} = R_{12} = R_{21} = R_{22} = 0,$$

if $X = C(\mathbf{s}, \mathbf{t}, G, H)$, $K = -D(\mathbf{t})$, $L = D(\mathbf{s})$, $\overline{K}_{jj} = K_{jj} = -D_j(\mathbf{t})$, $\overline{L}_{ii} = L_{ii} = D_i(\mathbf{s})$.

Proposition 4.4 *(a generator of the inverse). Let $XK + LX = G_XH_X^T$ and let* $\det X \neq 0$ *in* $\mathbf{F}$. *Then $X^{-1}L + KX^{-1} = (X^{-1}G_X)(H_X^TX^{-1})$.*

Proof. Pre- and post-multiply the matrix equation $XK + LX = G_XH_X^T$ by X^{-1}. □

(Note that the basic matrix pair (K, L) for X is mapped into the dual one for X^{-1}.)

5 Divide-and-conquer algorithm for structured matrices

Suppose that the input matrix X of generalized Algorithm 2.1 is given with its short (K, L)-generator for (K, L) from Tables 2-4. The associated descending process defines the basic matrix pairs for all matrices involved in the CRF. This stage relies on Proposition 4.3 for submatrices, Proposition 4.4 for the matrix inversions, and Propositions 4.1 and 4.2 for matrix additions/subtractions and multiplications. To define the basic matrix pair for a Schur complement (cf. (2.2)), we could have applied Propositions 4.1, 4.2, and 4.4, but it is even simpler to compute the basic matrix pairs successively for X^{-1} (based on Proposition 4.4), its southeastern block S^{-1} (based on Propositions 2.3 and 4.3), and finally, S (by applying Proposition 4.4). Similarly, we proceed for other Schur complements involved. Together with the basic matrix pairs (K, L), the descending process defines the (K, L)-ranks for all matrices of the CRF. (Here and hereafter, we slightly abuse the notation by writing the same letters K, L for the basic matrix pairs of all matrices of the CRF.)

Proposition 5.1 *If the input matrix X of generalized Algorithm 2.1 is given with it (K, L)-generator from Tables 2-4, then in the associated descending process the computed basic matrix pairs (K, L) for all matrices of the resulting CRF stay in Tables 2-4.*

Proof is by recursive application of Propositions 2.3, 2.4, 4.1-4.4. □

In the lifting process of generalized Algorithm 2.1, we compute the $O((m_1+m)l)$ entries of a short (K, L)-generator (of length l) for every $m \times m_1$ matrix Y of the CRF (for K, L defined in the descending process and for $l = r_{K,L}(Y)$) and, in the Toeplitz/Hankel-like case, also the m or m_1 entries of the first (or last) column or row of Y, respectively, so that the output fully defines the matrix by (3.2)-(3.8). The computation relies on equations (2.1)-(2.5), Propositions 4.1-4.3, Corollary 4.1, and Fact 3.3 (applied to every matrix Y computed with its (K, L)-generator of length ℓ where $\ell > r_{K,L}(Y)$). Such a lifting process will be called *compressed computation* of Y. Similarly, we operate with short generators when we represent the matrices F and N of (2.9), e.g. implicitly by a short generator of the matrix $-X_{11}^{-1}X_{12}$. Summarizing and taking into account the bound $O(r)$ of Proposition 5.1, we obtain the following estimates (cf. (2.8) where c is defined by (3.2)-(3.5), Theorem 3.1 and Corollary 4.1).

Theorem 5.1 *Let an $n \times n$ input matrix X of generalized Algorithm 2.1 have generic rank profile and have (K, L)-rank r for (K, L) of Theorem 3.1. Let X be given with its (K, L)-generator of length l, $r \leq \ell \leq n$. Let $C_{f,v}(n)$, $T_v(n)$, $V_v(n)$ and $C_v(n)$ be defined as in Corollary 4.1. Then the compressed computation of a full output set of this algorithm requires $O(n\ell)$ words of memory and*

$$F_{K,L}(n) = O(\ell^2 n + M_{v,K,L}(n) r^2 \log n) \tag{5.1}$$

ops where

$$M_{v,K,L}(n) = C_{f,v}(n) \tag{5.2}$$

for $K = -L = C_f$ and $K = C_f^T$, $L = -C_f$, for any scalar f,

$$M_{v,K,L}(n) = V_v(n) + C_{f,v}(n) \tag{5.3}$$

for $K = -C_f^T$, $L = D^{-1}(\mathbf{t})$ for any scalar f,

$$M_{v,K,L}(n) = C_v(n), \tag{5.4}$$

for $K = -D(\mathbf{t}), L = D(\mathbf{s})$.

Remark 5.1 *Generalized Algorithm 2.1 amounts to recursive application of Propositions 4.1–4.3, Corollary 4.1 and Fact 3.3. As soon as the latter results are extended to the matrix structure defined by a fixed basic pair (K, L), the algorithm, Theorem 5.1 and Corollary 6.1 of the next section can be extended. In particular, based on equations (3.6), (3.8), one may immediately extend the estimates of Theorem 5.1 and Corollary 6.1 to the case of all basic matrix pairs (K, L) from Tables 2-4. We also have an immediate extension to the case of the basic matrix pairs $(C_0 + C_0^T, -C_0 - C_0^T)$ and $(C_0 - C_0^T, C_0^T - C_0)$. Both of these pairs define the class of Toeplitz-like + Hankel-like matrices (see [*BP93*], [*BP94*], pp.185-188).*

6 Transformations among structured matrices and acceleration of Vandermonde-like and Cauchy-like computations

For the Vandermonde-like and Cauchy-like input matrices X, we may further improve the asymptotic cost estimates of (5.1), (5.3), (5.4) to yield the Toeplitz/Hankel-like level of (5.1), (5.2) by applying Proposition 4.2 and Table 5 (cf. [P90]).

Indeed, let $Y = V_f(\mathbf{t}, G, H)$ (cf. (3.4)) and write $U = -C_f^T$, $W = D^{-1}(\mathbf{t})$, $X = -V^T(\mathbf{t})$, $M = C_f$, $YU + WY = G_Y H_Y^T$; $G_Y, H_Y \in \mathbf{F}^{n \times l}$. Then we have $MX - XW = V^T(\mathbf{t})D^{-1}(\mathbf{t}) - C_f V^T(\mathbf{t}) = \mathbf{e}_0(\mathbf{t}^{-1} - f\mathbf{t}^{n-1})^T$, and application of Proposition 4.2 yields $-(XY)C_f^T + C_f(XY) = (XY)U + M(XY) = (XG_Y)H_Y^T + \mathbf{e}_0((\mathbf{t}^{-1} - f\mathbf{t}^{n-1})^T Y)$, that is, we obtain a $(-C_f^T, C_f)$-generator of length $\ell + 1$ for the matrix $-V_f^T(\mathbf{t})Y$. By (3.4) and Corollary 4.1, this computation requires multiplication of $X = -V^T(\mathbf{t})$ by G_Y and of the vector $(\mathbf{t}^{-1} - f\mathbf{t}^{n-1})^T$ by Y, that is, $(V_v^T(n) + 2n - 1)\ell = O((\ell n \log^2 n) \log\log n)$ ops (not counting $O(n \log n)$ ops for the computation of the vector $\mathbf{t}^{-1} - f\mathbf{t}^{n-1}$).

Similarly, we compute a $(C_f^T, -C_f)$-generator of length $l+2$ for the matrix $V^T(\mathbf{s^{-1}})YV(\mathbf{t}^{-1})$ provided that a matrix $Y = C(\mathbf{s}, \mathbf{t}, G, H)$ of (3.5) is given with its $(-D(\mathbf{t}), D(\mathbf{s}))$-generator of length l. The computation uses $(V_v^T(n) + V_v(n) + O(n))l = O(nl \log^2 n) \log\log n)$ ops. By Theorem 5.1, we may compute a full output set of generalized Algorithm 2.1 applied to the respective Hankel-like input matrices, $X = -V_f^T(\mathbf{t})V_f(\mathbf{t}, G, H)$ or $X = V^T(\mathbf{s}^{-1})C(\mathbf{s}, \mathbf{t}, G, H)V(\mathbf{t}^{-1})$, at the cost bounded according to (5.1), (5.2). Then we may obtain easily the partial output sets (cf. Definition 2.2) of the same algorithm for the matrices $V_f(\mathbf{t}, G, H)$ and $C(\mathbf{s}, \mathbf{t}, G, H)$ at the cost dominated by (5.1), (5.2). More precisely, we obtain

Corollary 6.1 *Under the assumptions of Theorem 5.1, generalized Algorithm 2.1 applied to compressed computation of a partial output set of the matrix X requires $O(n\ell)$ words of memory and $V_v(n) + F_{K,L}(n)$ ops, for $F_{K,L}(n)$ bounded by (5.1), (5.2), that is, $F_{K,L}(n) = O(nl^2 + (nr^2 \log^2 n) \log\log n)$ over any field of constants.*

The above transformations among various classes of structured matrices use *Vandermonde multipliers*, as this was proposed in [P90]. In the special case where $\mathbf{s}$ and/or $\mathbf{t}$ are the (scaled) vectors of roots of 1, the Vandermonde multipliers turn into the matrices of (scaled) Fourier transforms (*Fourier multipliers*). In this special case the transformations are simplified [H95], [GKO95]. In particular, next, we will simplify the above maps to Toeplitz/Hankel-like matrices for some important subclasses of the input classes of Vandermonde-like and Cauchy-like matrices in the case where the field $\mathbf{F}$ supports the n-point FFT. We will use the following definition and auxiliary results.

Definition 6.1 *Write* $\mathbf{e} = (g^i)_{i=0}^{n-1}$, $\mathbf{f} = (h^i)_{i=0}^{n-1}$ *provided that* $g^n = e, h^n = f$; $\mathbf{w} = (\omega^i)_{i=0}^{n-1}$ *where* ω *is a primitive* n*-th root of* 1, $\omega^n = 1$, $\omega^s \neq 1$ *for* $s = 1, \ldots, n-1$; $\Omega = (\omega^{ij})_{i,j=0}^{n-1}$. *(*$\Omega$ *denotes the matrix of the* n*-point discrete Fourier transform, DFT. In the complex field* $\mathbf{C}$*, we may choose* $\omega = exp(2\pi\sqrt{-1}/n)$*, and we have* $\Omega^H\Omega = I$*.)*

Proposition 6.1 *(cf. [CPW74]). Let* $f \neq 0$, $h^n = f$. *Then*

$$C_f = U_f^{-1} D(g\mathbf{w}) U_f \tag{6.1}$$

where

$$U_f = \Omega D(\mathbf{f}). \tag{6.2}$$

Proposition 6.2 *Let* $J\mathbf{t} = h\mathbf{w}$, $f = h^n \neq 0$, $X = JV_f(\mathbf{t}, G, H)J$,

$$XC_f - D^{-1}(J\mathbf{t})X = G_X H_X^T \tag{6.3}$$

(cf. Table 3). Then we have (cf. Theorem 3.1a) that $T_f C_f - C_f T_f = GH^T$,

$$G = U_f^{-1} G_X, H = H_X, T_f = U_f^{-1} X, \tag{6.4}$$

for U_f *of (6.2).*

Proof. Pre-multiply (6.3) by U_f^{-1} and substitute equations (6.1), (6.2) and (6.4). □

Proposition 6.3 *(cf. [P90], [H95], [GKO95]). Let* $\mathbf{s} = g\mathbf{w}, \mathbf{t} = h\mathbf{w}$, $e = g^n \neq 0$, $f = h^n \neq 0$, $Y = C(\mathbf{s}, \mathbf{t}, G, H)$. *Define* U_e *and* U_f *by (6.2). Let*

$$YD(\mathbf{t}) - D(\mathbf{s})Y = G_Y H_Y^T \tag{6.5}$$

(cf. Table 4). Then we have (cf. Theorem 3.1a) that $\tilde{T}C_e - C_f\tilde{T} = GH^T$,

$$G = U_f^{-1} G_Y, H^T = H_Y^T U_e, \tilde{T} = U_f^{-1} Y U_e. \tag{6.6}$$

Proof. Pre-multiply (6.5) by U_f^{-1}, post-multiply by U_e, and substitute equations (6.1), (6.2), (6.6), and $U_e = \Omega D(\mathbf{e})$, which is a variation of (6.2). □

Remark 6.1 *The equation* $\tilde{T}C_e - C_f\tilde{T} = GH^T$ *defines a* $(C_e, -C_f)$*-generator* (G, H) *for* $\tilde{T}$*. Equivalently, we have* $\tilde{T}C_f - C_f\tilde{T} = \tilde{G}\tilde{H}^T$ *where* $\tilde{G} = (G, \tilde{T}\mathbf{e}^{(0)})$, $\tilde{H} = (H, (f-e)\mathbf{e}^{n-1})$. *By Theorem 3.1a) for* $\tilde{T}, \tilde{G}, \tilde{H}$ *replacing* T_f, G, H*, respectively, we conclude that* $(\tilde{G}, \tilde{H})$ *is a* $(C_f, -C_f)$*-generator for* $\tilde{T}$*, whose length increases at most by 1 versus the* $(C_e, -C_f)$*-generator* (G, H)*. Proposition 6.3 enables us to avoid even such a minor increase of the length. Indeed, by Theorem 3.1 d), we express* $Y = C(\mathbf{s}, \mathbf{t}, G, H)$ *by applying (3.5) and then obtain a* (C_e, C_f)*-generator of the* same *length* l *for the matrix* $\tilde{T}$*, by applying (6.6).*

By extending Propositions 6.2 and 6.3, we may transform the (K, L)-generators of all the Vandermonde-like and Cauchy-like matrices associated with basic matrix pairs (K, L) of Tables 3 and 4 (at the cost of 1 or 2 diagonal scalings and performing 1 or 2 DFTs) into generators of the same length for Toeplitz/Hankel-like matrices provided that K and L are of the form $\pm C_f, \pm C_f^T$ and/or $\pm D(g\mathbf{w})$ for two scalars, g and $f = g^n \neq 0$. The first (or last) column or row of every resulting matrix can be also computed easily. (All the above computations are further simplified slightly in the case where $g = f = 1, D(\mathbf{f}) = I$, and $U_f = \Omega$.)

Remark 6.2 *One may post-multiply (6.3) by U_f^{-1} (cf (6.2)) and then substitute (6.1) to map the Vandermonde-like matrix X of (6.3) into the Cauchy-like matrix XU_f^{-1} satisfying the matrix equation $(XU_f^{-1})D(g\mathbf{w}) - D^{-1}(J\mathbf{t})(XU_f^{-1}) = G_X(H_X^T U_f^{-1})$. Similarly, we may map every matrix pair of Table 3 into one of Table 4 and also any Toeplitz/Hankel-like matrix into a closely related matrix from either of Tables 3 and 4.* [GKO95] *uses such maps to improve Toeplitz-like matrix computations with pivoting.*

7 Transformations of a general matrix X into a matrix with generic rank profile and an extended randomized algorithm

To relax the generic rank profile assumption for the input, we may apply our computations with pivoting, but pivoting generally destroys matrix structure. As an exception, we may preserve Cauchy-like structure in Gaussian but not block Gaussian elimination, with pivoting, that is, where we eliminate the variables one by one, rather than by blocks, but in this case the best resulting algorithms involve order of n^2 ops [GKO95].

On the other hand, for nonsingular real or complex matrices X, symmetrization ensures strong nonsingularity and preserves all the structures of Tables 2-4. (Exception is the Cauchy-like complex case where symmetrization generally destroys the structure. In the real Cauchy-like case, one should use implicit representation of X^TX or XX^T by the pair of generators of X^T and X in order to satisfy the assumption of Definition 3.1 that $s_i \neq t_i$ for all i.)

Next, we will apply randomization to transform (over any field $\mathbf{F}$) a general (unstructued or structured and possibly singular) matrix X into one having generic rank profile. We will use the following basic result.

Theorem 7.1 [DL78] *(cf. also [Z79], [S80]). Let $p(\mathbf{x}) = p(x_1, x_2, \cdots, x_m)$ be a non-zero m-variate polynomial of a total degree d. Let $\mathbf{S}$ be a finite set of cardinality $|\mathbf{S}|$ in the domain of the definition of $p(\mathbf{x})$, let the random values $x_1^*, \cdots, x_m^*$ be sampled from $\mathbf{S}$, that is, chosen from $\mathbf{S}$ independently of each other under the uniform probability distribution on $\mathbf{S}$, and let $\mathbf{x}^* = (x_1^*, x_2^*, \cdots, x_m^*)$. Then probability$(p(\mathbf{x}^*) = 0) \leq d/|\mathbf{S}|$.*

We also have

Theorem 7.2 *(cf. [M82],* [C841]*). An $n \times n$ Cauchy matrix $C(\mathbf{s}, \mathbf{t})$ is nonsingular if and only if all the 2n components of the vectors $\mathbf{s}$ and $\mathbf{t}$ are distinct. Every square submatrix of a nonsingular Cauchy matrix is nonsingular.*

Next, for a given matrix X, we will define some pairs of random structured matrices (*preconditioners*) A and B to yield generic rank profile for the matrix

$$\tilde{X} = AXB. \tag{7.1}$$

Our next result generalizes one of [KS91] for triangular Toeplitz preconditioners A_2, B_2 (cf. also unstructured preconditioners used in the pioneering paper [BGH82]).

Theorem 7.3 *Let us be given a matrix $X \in \mathbf{F}^{n\times n}$, and six vectors $\mathbf{p}, \mathbf{q}, \mathbf{u}, \mathbf{v}$, $\mathbf{y} = (y_j)$, $\mathbf{z} = (z_j) \in \mathbf{F}^{n\times 1}$, where each of the two pairs of vectors $\mathbf{p}, \mathbf{q}$ and $\mathbf{u}, \mathbf{v}$ is filled with 2n* *distinct scalars, $y_0 = z_0 = 1$, and y_j, z_j are indeterminates for $j > 0$. Let $\mathbf{1} = (1)_{i=0}^{n-1} = \sum_{i=0}^{n-1} \mathbf{e}_i$. Then the matrix $\tilde{X}$ of (7.1) has generic rank profile if $(A, B) = (A_\alpha, B_\beta)$, $(\alpha, \beta) \in \{(1,1), (2,2), (1,2), (2,1)\}$ where $A_1 = C(\mathbf{p}, \mathbf{q}, \mathbf{1}, \mathbf{y})$, $B_1 = C(\mathbf{u}, \mathbf{v}, \mathbf{z}, \mathbf{1})$, $A_2 = C_0^T(\mathbf{y})$, $B_2 = C_0(\mathbf{z})$.*

Proof. For an $n \times n$ matrix M, denote by $M_{I,J}$ the determinant of the submatrix of M formed by all entries lying simultaneously in the rows indexed by the set I and in the columns indexed by the set J. Let $\rho = \text{rank } X$. For the sets $I = \{1, 2, \dots, i\}$, $J = \{j_1, j_2, \dots, j_i\}$, $K = \{k_1, k_2, \dots, k_i\}$, $i = 1, 2, \dots, \rho$, we have from the Cauchy-Binet formula

$$\tilde{X}_{I,I} = \sum\nolimits_J \sum\nolimits_K A_{I,J} X_{J,K} B_{K,I},$$

where the summation is in all sets J and K, each made of i distinct indices.

Let us prove that

$$\tilde{X}_{I,I} \neq 0 \text{ for } i = 1,\ 2,\ \dots,\ \rho\ . \tag{7.2}$$

At first assume that $A = A_1$, $B = B_1$ and then observe that, for a fixed pair of $J = [j_1, j_2, \dots, j_i]$ and $K = [k_1, k_2, \dots, k_i]$, we have $A_{I,J} = a y_{j_1} \cdots y_{j_i}$ where the monomial $y_{j_1} y_{j_2} \cdots y_{j_i}$ uniquely identifies the set J and where $a = (C(\mathbf{p}, \mathbf{q}))_{I,J}, a \neq 0$ by the virtue of Theorem 7.2. Likewise, $B_{K,I} = b z_{k_1} \cdots z_{k_i}$ where the monomial $z_{k_1} \cdots z_{k_i}$ identifies the set K and $b = (C(\mathbf{u}, \mathbf{v}))_{K,I} \neq 0$. Therefore, for distinct pairs (J, K), the terms $A_{I,J}\ X_{J,K} B_{K,I}$ cannot cancel each other. Consequently, $\tilde{X}_{I,I} \neq 0$ provided that there exists a pair (J, K) such that $X_{J,K} \neq 0$. This is true for all $i \leq \rho$ since X has rank ρ, and we arrive at (7.2).

If $A = A_2$, $B = B_2$, then we define lexicographic order for the variables: $1 < y_1 < y_2 < \dots < y_{n-1}$ and $1 < z_1 < z_2 < \dots < z_{n-1}$, and observe that for each set J the determinant $A_{I,J}$ is uniquely defined by its unique lowest order monomial. Furthermore, this monomial does not appear in other determinants $A_{I,J}$ for the same I. Similar property holds for $B_{K,I}$ for all sets K, and again (7.2) follows. The same arguments apply to the pairs $A = A_1$, $B = B_2$ and $A = A_2$, $B = B_1$. □

Corollary 7.1 *Under the assumptions of Theorem 7.3, let the values of the $2n-2$ variables, y_j and z_j, $j = 1, \dots, n-1$, be randomly sampled from a fixed finite set $\mathbf{S}$ of cardinality $|\mathbf{S}|$. Then the matrix $\tilde{X} = AXB$ of rank ρ has generic rank profile with a probability at least $1 - (\rho + 1)\rho/|\mathbf{S}|$.*

Proof. $\det \tilde{X}$ is a polynomial in the variables $y_1, z_1, \dots, y_{n-1}, z_{n-1}$ of a total degree at most $2k$; for $k \leq \rho$ it does not vanish identically in these variables by Theorem 7.3. By Theorem 7.1, it may vanish with a probability at most $2k/|\mathbf{S}|$ under the random sampling of the variables. Therefore, the probability that neither of $\det \tilde{X}^{(k)}$ vanishes under the random sampling is at least $\prod_{k=1}^{\rho}(1 - 2k/|\mathbf{S}|) > 1 - (\rho + 1)/|\mathbf{S}|$. □

Corollary 7.1 can be combined with generalized Algorithm 2.1 as follows.

Algorithm 7.1. *Randomized computation of a partial output set for a general matrix.*

Input: a field $\mathbf{F}$, a pair $(\alpha, \beta), \alpha, \beta \in \{1, 2\}$, an $n \times n$ matrix $X \in \mathbf{F}^{n \times n}$, and a vector $\mathbf{b} \in \mathbf{F}^{n \times 1}$.

Output: a partial output set of generalized Algorithm 2.1 applied to X.

Computations:

1. Fix a finite set $\mathbf{S}$ of non-zero elements of $\mathbf{F}$ (cf. Remark 7.3 at the end of the section) or its extension and randomly sample from $\mathbf{S}$ the $2n-2$ elements y_i, z_i, $i = 1, \dots, n-1$, defining two matrices, $A = A_\alpha$ and $B = B_\beta$ of Theorem 7.3.

2. Compute the matrix $\tilde{X} = AXB$.

3. Apply generalized Algorithm 2.1 to the matrix $\tilde{X}$, which in particular outputs $\rho = \text{rank } \tilde{X}$.

4. Compute the matrices F and N of (2.9) for X replaced by $\tilde{X}$. Verify whether the matrix $\tilde{X}N$ (formed by the $n - \rho$ last columns of the matrix $\tilde{X}F$) is a null matrix. If "not", output FAILURE, which indicates that the randomization failed to insure the generic rank profile property for X.

5. Otherwise compute the matrix BN whose columns form a basis for the null space of X.

6. If the linear system $\tilde{X}\mathbf{w} = A\mathbf{b}$ has no solution $\mathbf{w}$, then also the system $X\mathbf{y} = \mathbf{b}$ has no solution $\mathbf{y}$. In this case output INCONSISTENT. Otherwise compute a solution $\mathbf{w}$ to the former system and then a solution $\mathbf{y} = B\mathbf{w}$ to the latter one.

7. If $\rho = n$, compute $X^{-1} = B\tilde{X}^{-1}A$ and $\det X = (\det \tilde{X})/((\det A) \det B)$.

Correctness of Algorithm 7.1 is easily verified based on (2.9), (2.10) and (7.1).

The computational cost (in the case of a general matrix X) is clearly dominated by the cost of the application of generalized Algorithm 2.1 (we ignore the cost of generation of random parameters).

Remark 7.1 *One may re-apply Algorithm 7.1 in the case if FAILURE is output. In ν applications, the probability of outputing FAILURE ν times is at most $((\rho + 1)\rho/|\mathbf{S}|)^\nu$.*

Remark 7.2 *To increase $|\mathbf{S}|$, one may need to extend $\mathbf{F}$ if $\mathbf{F}$ contains only a few elements. Opeating in the extension $\mathbf{F}[x]/q(x)$ for an irreducible polynomial $q(x)$ of degree d increases the cardinality of $\mathbf{F}$ by factor 2^d at the price of increasing the time cost of our computations by factor $O(d \log d)$.*

Remark 7.3 *The restriction that the set $\mathbf{S}$ should not contain 0 is needed to insure non-singularity of the matrices A and B except for the case where $A = A_2$, $B = B_2$. In the latter case we may let $\mathbf{S}$ contain 0.*

Remark 7.4 *Instead of A_2 and B_2, we may choose $A_3 = C_f^T(\mathbf{y})$, $B_3 = C_f(\mathbf{z})$, for an indeterminate f. Then, clearly, Theorem 7.3 is extended. If the value f is randomly sampled from the set $\mathbf{S}$, the proof of Corollary 7.1 is also extended, except that the degree of* $\det \tilde{X}$ *(as a polynomial in $f, y_1, z_1, \ldots, y_{n-1}, z_{n-1}$) is doubled, and so is the failure probability. Such a minor deficiency can be weighted versus the minor acceleration (by roughly factor 2) of the complexity of the computations with triangular Toeplitz matrices $A_2 = C_0^T(\mathbf{y})$, $B_2 = C_0^T(\mathbf{z})$ in the transition to the f-circulant matrices $A_3 = C_f^T(\mathbf{y})$, $B_3 = C_f(\mathbf{z})$ for $f \neq 0$.*

Remark 7.5 *The results of this section rely on using* Cauchy-like multipliers *A_1, B_1 and* Toeplitz multipliers *A_2, B_2, A_3, B_3. Vandermonde multipliers of section 6 could be also used, but then we would have dealt with polynomials of degree as high as $(\rho + 1)\rho n/2$, with the resulting increase by factor n of the failure probability bound.*

8 Randomization for a structured input matrix X

Let us next show appropriate choices of the matrices A and B of (7.1) that preserve the structure of some sample matrices from Tables 2-4. (The extension to all other matrices of these tables is straightforward.)

a) For $X = T_f$, $X = JT_f$, and $X = T_f J$ (cf. (3.2), (3.3)) and also for $X = T_f + JT_e$, we may choose $A = A_2$, $B = B_2$ or $A = A_3$, $B = B_3$ (cf. Remark 7.4), thus preserving the Toeplitz/Hankel-like structure of X in the transition to $\tilde{X}$ (cf. Proposition 4.2 and Tables 2 and 5).

b) For $X = C(\mathbf{s}, \mathbf{t}, G, H)$ (cf. (3.5) and Tables 4 and 5), we choose $A = A_1$, $B = B_1$, $\mathbf{q} = \mathbf{s}$, $\mathbf{u} = \mathbf{t}$ and arrive at $\tilde{X} = C(\mathbf{p}, \mathbf{v}, \tilde{G}, \tilde{H})$ where we may choose any pair $(\mathbf{p}, \mathbf{v})$.

c) For $X = V_f^T(\mathbf{t}, G, H)$ (cf. (3.4) and Tables 3 and 5), we choose $A = A_2$, $B = B_1$, with $\mathbf{u} = -\mathbf{t}^{-1}$ and arrive at the matrix $\tilde{X} = V_f^T(-\mathbf{v}^{-1}, \tilde{G}, \tilde{H})$, for $\tilde{G}, \tilde{H}$ defined by Proposition 4.2, where we may choose any vector $\mathbf{v}^{-1}$. Likewise, for $X = V_f(\mathbf{t}, G, H)$, we choose $A = A_1$, $B = B_2$ with $\mathbf{q} = \mathbf{t}^{-1}$ and arrive at the matrix $X = V_f(-\mathbf{v}, \overline{G}, \overline{H})$, for $\overline{G}, \overline{H}$ defined by Proposition 4.2 where we may choose any vector $\mathbf{v}$.

Clearly, in all cases a)-c) above, the matrix $\tilde{X}$ belongs to the same class of structured matrices of Tables 2-4 as X does. We also deduce from Proposition 4.2 that the length of the generator for X increases by at most 2 in the above sample transitions to $\tilde{X}$. It follows immediately that the estimates of Theorem 5.1 and Corollary 6.1 apply to the randomized computational cost of performing Algorithm 7.1 too.

Moreover, in the case of Vandermonde-like and Cauchy-like computations over the fields $\mathbf{F}$ that support FFT at $n = 2^h$ points for an integer h, we may achieve the cost level (5.1), (5.2) in the cases b) and c) above as a by-product of randomization, by choosing $\mathbf{p} = g\mathbf{w}$ and $\mathbf{v} = h\mathbf{w}$ for any nonzero scalars g and h such that the vectors $g\mathbf{w}$ and $h\mathbf{w}$ have $2n$ distinct components and by applying Propositions 6.2 and 6.3 to the matrix $\tilde{X}$ and to the matrices of smaller size involved in the computations by generalized Algorithm 2.1 applied to $\tilde{X}$. (By Propositions 4.1-4.4, the structure allowing application of Propositions 6.2 and 6.3 is preserved for the latter matrices of smaller size.)

9 Simplified expressions for the generators of Schur complements

The computations of short generators of Schur complements in the generalized Algorithm 2.1 can be a little simplified based on the next result, which generalizes one of [GO94a], [OP98] (where it was assumed that $K_{21} = L_{12} = 0$).

Proposition 9.1 *Under the assumptions of Proposition 4.3, let X_{11} be a nonsingular matrix. Then the Schur complement S of X_{11} in X satisfies the following matrix equations:*

$$SK_{22} + L_{22}S = (G_2 - X_{21}X_{11}^{-1}G_1)(H_2^T - H_1^T X_{11}^{-1} X_{12}) + R,$$

$$R = SK_{21}X_{11}^{-1}X_{12} + X_{21}X_{11}^{-1}L_{12}S. \tag{9.1}$$

Proof. Recall Proposition 2.3 and the definitions of Proposition 4.3 and also write $\overline{X} = X^{-1} = \begin{pmatrix} \overline{X}_{11} & \overline{X}_{12} \\ \overline{X}_{21} & S^{-1} \end{pmatrix}$.

Deduce from Proposition 4.4 that $K\overline{X} + \overline{X}L = \overline{X}GH^T\overline{X}$. Deduce from the latter equation that $K_{22}S^{-1} + S^{-1}L_{22} + K_{21}\overline{X}_{12} + \overline{X}_{21}L_{12} = (\overline{X}_{21}G_1 + S^{-1}G_2)(H_1^T\overline{X}_{12} + H_2^TS^{-1})$ and consequently $SK_{22} + L_{22}S = (S\overline{X}_{21}G_1 + G_2)(H_1^T\overline{X}_{12}S + H_2^T) - SK_{21}\overline{X}_{12}S - S\overline{X}_{21}L_{12}S$. Substitute the expressions from (2.5) for $\overline{X}_{12}S$ and $S\overline{X}_{21}$ and obtain Proposition 9.1. $\square$

By (9.1), we have rank $R \leq$ rank $K_{21}+$ rank L_{12}. The expressions of Propositions 9.1 for a (K, L)-generator of S are the simplest where $K_{21} = L_{12} = 0$, which implies that $R = 0$ (cf. (9.1)). This is the case where $K = -L^T = C_0^T$; $K = -C_0^T$, $L = D^{-1}(\mathbf{t})$, and $K = -D(\mathbf{t})$, $L = D(\mathbf{s})$, which corresponde to the Hankel-like, Vandermonde-like and Cauchy-like structures, respectively. In the Hankel-like case, this implies dealing with C_f^T for $f = 0$, that is, we must operate with triangular Toeplitz matrices, which means a minor slowdown (roughly by factor 2, [BP94], pp. 133-135) versus the same operations with f-circulant matrices involved in the case where $f \neq 0$. The cost of such a slowdown can be weighted versus the decrease of the cost of handling non-rull matrices R of (9.1) when we shift from $f \neq 0$ when we shift from $f \neq 0$ to $f = 0$.

References

[AHU74] A. V. Aho, J. E. Hopcroft, J. D. Ullman, *The Design and Analysis of Computer Algorithms,* Addison-Wesley, Reading, Massachusetts, 1974.

[BA80] R.R. Bitmead, B.D.O. Anderson, Asymptotically Fast Solution of Toeplitz and Related Systems of Linear Equations, *Linear Algebra Appl.*, **34**, 103-116, 1980.

[BGH82] A. Borodin, J. von zur Gathen, J. Hopcroft, Fast Parallel Matrix and GCD Computation, *Information and Control*, **52, 3**, 241–256, 1982.

[BP93] D. Bini, V. Y. Pan, Improved Parallel Computations with Toeplitz-like and Hankel-like Matrices, *Linear Algebra Appl.*, **188/189**, 3-29, 1993.

[BP94] D. Bini, V.Y. Pan, *Polynomial and Matrix Computations, Volume 1: Fundamental Algorithms*, Birkhäuser, Boston, 1994.

[C841] A. L. Cauchy, Mémorie sur les Fonctions Alternées et sur les Somme Alternées, *Exercises d' Analyse et de Phys. Math.*, **II**, 151-159, 1841.

[CKL-A87] J. Chun, T. Kailath, H. Lev-Ari, Fast Parallel Algorithm for QR-factorization of Structured Matrices, *SIAM J. Sci. Stat. Comput.*, **8**, **6**, 899-913, 1987.

[CPW74] R. E. Cline, R. J. Plemmons, G. Worm, Generalized Inverses of Certain Toeplitz Matrices, *Linear Algebra Appl.*, **8**, 25–33, 1974.

[DL78] R.A. Demillo, R.J. Lipton, A Probabilistic Remark on Algebraic Program Testing, *Information Process. Letters*, **7, 4**, 193-195, 1978.

[GKK86] I. Gohberg, T. Kailath, I. Koltracht, Efficient Solution of Linear Systems of Equations with Recursive Structure, *Linear Algebra Appl.*, 80, 81-113, 1986.

[GKKL87] I. Gohberg, T. Kailath, I. Koltracht, P. Lancaster, Linear Complexity Parallel Algorithms for Linear Systems of Equations with Recursive Structure, *Linear Algebra Appl.*, **88/89**, 271-315, 1987.

[GKO95] I. Gohberg, T. Kailath, V. Olshevsky, Fast Gaussian Elimination with Partial Pivoting for Matrices with Displacement Structure, *Math. of Computation*, **64**, 1557-1576, 1995.

[GL96] G. H. Golub, C. F. Van Loan, *Matrix Computations,* Johns Hopkins University Press, Baltimore, Maryland, 1996 (third edition).

[GO92] I. Gohberg, V. Olshevsky, Circulants, Displacements and Decompositions of Matrices, *Integral Equations and Operator Theory*, **15, 5**, 730–743, 1992.

[GO94] I. Gohberg, V. Olshevsky, Complexity of Multiplication with Vectors for Structured Matrices, *Linear Algebra Appl.*, **202**, 163-192, 1994.

[GO94a] I. Gohberg, V. Olshevsky, Fast state space algorithms for matrix Nehari and Nehari-Takagi interpolation problems, *Integral Equations and Operator Theory*, **20, 1**, 44-83, 1994.

[H95] G. Heinig, Inversion of Generalized Cauchy Matrices and the Other Classes of Structured Matrices, *Linear Algebra for Signal Processing, IMA Volume in Math. and Its Applications*, **69**, 95-114, Springer, 1995.

[Ho70] A.S. Householder, *The Numerical Treatment of a Single Nonlinear Equation*, McGraw-Hill, New York, 1970.

[IMH82] O. H. Ibarra, S. Moran, R. Hui, A Generalization of the Fast LUP Matrix Decomposition Algorithm and Applications, *J. of Algorithms,* **3**, 45–56, 1982.

[K95] E. Kaltofen, Analysis of Coppersmith's Block Wiedemann Algorithm for the Parallel Solution of Sparse Linear Systems, *Mathematics of Computation*, **64, 210**, 777-806, 1995.

[K,a] I. Kaporin, A Practical Algorithm for Faster Matrix Multiplication, *Numerical Linear Algebra with Applications*, submitted.

[KKM79] T. Kailath, S.Y. Kung, M. Morf, Displacement Ranks of Matrices and Linear Equations, *J. Math. Anal. Appl.*, **68, 2**, 395-407, 1979.

[KO96] T. Kailath, V. Olshevsky, Displacement Structure Approach to Discrete Transform Based Preconditioners of G. Strang Type and of T. Chan Type, *Calcolo*, **33**, 191-208, 1996.

[KO97] T. Kailath, V. Olshevsky, Displacement Structure Approach to Polynomial Vandermonde and Related Matrices, *Linear Algebra and Its Applications*, **261**, 49-90, 1997.

[KS91] E. Kaltofen, B. D. Saunders, On Wiedemann's Method for Solving Sparse Linear Systems, *Proc. AAECC-5, Lecture Notes in Computer Science,* **536**, 29-38, Springer, Berlin, 1991.

[KS95] T. Kailath, A. Sayed, Displacement Structure: Theory and Applications, *SIAM Review*, **37**, 297-386, 1995.

[KS99] T. Kailath, A. Sayed (editors), *Fast Reliable Algorithms for Matrices with Structure*, SIAM Publications, Philadelphia, 1999.

[KVM78] T. Kailath, A. Viera, M. Morf, Inverses of Toeplitz Operators, Innovations, and Orthogonal Polynomials, *SIAM Rev.*, **20**, 1, 106-119, 1978.

[M80] M. Morf, Doubling Algorithms for Toeplitz and Related Equations, *Proc. IEEE Intern. Conf. on ASSP*, 954-959, IEEE Comp. Soc. Press, 1980.

[M82] L. Mirsky, *An Introduction to Linear Algebra*, Dover, New York, 1992.

[OP98] V. Olshevsky, V. Y. Pan, A Unified Superfast Algorithm for Boundary Rational Tangential Interpolation Problem, *Proc. 39th Ann. IEEE Symp. Foundations of Comp. Sci.,* 192-201, IEEE Comp. Soc. Press, 1998.

[OP99] V. Olshevsky, V. Y. Pan, Polynomial and Rational Evaluation and Interpolation (with Structured Matrices), *Proc. 26th Ann. Intern. Colloq. on Automata, Languages, Programming (ICALP'99), Lecture Notes in Computer Science*, Springer, 1999.

[OS99] V. Olshevsky, M. A. Shokrollahi, A Displacement Approach to Efficient Decoding of Algebraic-Geometric Codes, *Proc. 31st Ann. Symp. on Theory of Computing*, 235-244, ACM Press, New York, May 1999.

[P90] V.Y. Pan, On Computations with Dense Structured Matrices, *Math. of Computation*, **55, 191**, 179-190, 1990.

[P92] V.Y. Pan, Parametrization of Newton's Iteration for Computations with Structured Matrices and Applications, *Computers and Math. (with Applics.)*, **24, 3**, 61–75, 1992.

[PACLS98] V. Y. Pan, M. AbuTabanjeh, Z. Chen, E. Landowne, A. Sadikou, New Transformations of Cauchy Matrices and Trummer's Problem, *Computers and Math. (with Applics.)*, **35, 12**, 1-5, 1998.

[PLST93] V. Y. Pan, A. Sadikou, E. Landowne, O. Tiga, A New Approach to Fast Polynomial Interpolation and Multipoint Evaluation, *Computers & Math. (with Applications),* **25, 9**, 25-30, 1993.

[PSD70] P. Penfield Jr., R. Spencer, S. Duinker, *Tellegen's Theorem and Electrical Networks,* MIT Press, Cambridge, Massachusetts, 1970.

[PZHY97] V. Y. Pan, A. Zheng, X. Huang, Y. Yu, Fast Multipoint Polynomial Evaluation and Interpolation via Computations with Structured Matrices, *Annals of Numerical Math.*, **4**, 483–510, January 1997.

[S69] V. Strassen, Gaussian Elimination Is Not Optimal, *Numer. Math.*, **13**, 354-356, 1969.

[S80] J.T. Schwartz, Fast Probabilistic Algorithms for Verification of Polynomial Identities, *J. of ACM*, **27, 4**, 701-717, 1980.

[Z79] R.Z. Zippel, Probabilistic Algorithm for Sparse Polynominals, *Proc. EUROSAM 79, Lecture Notes in Computer Science*, **72**, 216-226, Springer, Berlin, 1979.