INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

Efficient Cauchy-like Computations

by

Mohammad Abu Tabanjeh

A dissertation submitted to the Graduate Faculty in Mathematics in partial fulfillment of the requirements for the degree of Doctor of Philosophy at the City University of New York.

2000

This manuscript has been read and accepted for the Graduate Faculty in Mathematics in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Nov. 19, 1999
Date

Victor Pan
Chair of Examining Committee

Nov. 19, 1999
Date

Executive Officer

Victor Y. Pan Victor Pan

Michael Anshel Michael Anshel

Alexei Myasnikov Alexei Miasnikov

Supervisory Committee

The City University of New York

Abstract

Efficient Cauchy-like Computations

by

Mohammad Abu Tabanjeh

Advisor: Professor Victor Pan

Computation with $n \times n$ dense structured matrices are highly important in sciences, communication and engineering. The *space* and *time* complexity of the computations decreases in comparison with the case of $n \times n$ general matrices, that is, from order of n^2 words of storage space and order of n^i arithmetic operations with $2.37 < i \leq 3$ in the best algorithms, to $O(n)$ words of storage space and to $O(n \log^2 n)$ (and sometimes to $O(n \log n)$) arithmetic operations, with small overhead constants.

The most celebrated classes of structured matrices are Toeplitz and Hankel matrices, but some other classes of structured matrices such as Cauchy and Vandermonde are quite popular too.

We focused our study on Cauchy and Cauchy-like computations, in particular, we present a superfast Cauchy-like linear solver for any nonsingular Cauchy-like linear system of equations. The algorithm also computes short displacement generator for the inverse and determinant of a nonsingular Cauchy and Cauchy-like matrix. The algorithm is an extension of the well known divide-and-conquer (MBA) algorithm, and its every recursive divide-and-conquer step can be reduced to Trummer's celebrated problem, for which we have some generalization of Fast Multipole Algorithm.

Finally, we extend our superfast algorithm to the solution of consistent but possibly singular Cauchy-like linear system over any field of constants.

Acknowledgement

No dissertation is written alone, and mine is no exception. Many are the people who have lent me their expertise, their patience and – very often – their sympathy during the long journey of research and writing.

First among these, I would like to thank my advisor, Professor Victor Pan. His phenomenal mathematical ability has served to inspire me in my research more than I could possibly express. I look forward to continuing in the path he has helped me hew.

I am also deeply indebted to Professors Michael Anshel and Alexei Myasnikov for serving on my committee and for the invaluable assistance they have provided me.

I am also enormously thankful to all the faculty members of the Mathematics Ph.D. Program at the CUNY Graduate School and University Center for their generous support of my work and me. Especially, I would like to recognize the generosity and thoughtful insights provided by Professors Joseph Dodziuk, Burton Randol and Alphonse Vasquez.

Beside the professors, I am grateful to the administrators of the Mathematics and Computer Science Departments, among them Mr. Joseph Driscoll.

Finally, I must acknowledge the love and support of my parents, my family and my friends who have stuck out this often agonizing process with me, never failed to give me support and – when the time got rough – helped divert my attention from Cauchy-like computations.

I thank them all.

Contents

1 Introduction **1**

2 Structured Matrices **4**

2.1 General and Dense Structured Matrices 4

2.2 Definitions and Basic Facts 5

2.3 Structured Matrices: Correlations to Polynomial Computations 10

3 Recursive Factorization (RF) of General Matrices and Extensions **12**

3.1 Introduction to Recursive Factorization of General Matrices 12

3.2 Control over the Precision of the Computation of the RF 14

4 Recursive Factorization of Cauchy-like Matrices **16**

4.1 Recursive Factorization of a Strongly Nonsingular Cauchy-like Matrix: Bounds on the Scaling Ranks. 17

4.2 Computational Complexity Estimates of Cauchy-like CRF 18

4.3 Ensuring Strong Nonsingularity of a Nonsingular Cauchy-like Matrix by Means of Symmetrization 20

5 Superfast Computations with Singular Structured Matrices **22**

5.1 Introduction 22

5.2 Definitions and Basic Facts 24

5.3 Recursive Factorization of a Cauchy-like Matrix 24

5.4 Ensuring strong nonsingularity 24

5.5 Fast Cauchy-like Computations - Singular Case 26

APPENDIX A: Computations with Cauchy Matrices 31

BIBLIOGRAPHY 33

Chapter 1

Introduction

Computations with dense structured matrices are ubiquitous in sciences, communication and engineering. Exploitation of structure enables dramatic acceleration of the computations and major decrease of the memory space constraints but sometimes this also leads to numerical stability problems.

The best known and most celebrated classes of structured matrices are Toeplitz and Hankel matrices, but some other classes of structured matrices, such as ones of Cauchy and Vandermonde types, are also quite popular and are gaining more and more recognition. In particular, Cauchy and Cauchy-like matrices appear in applications to rational interpolation [D74] and rational matrix (tangential) interpolation [GO94], conformal mapping [T86], and numerical solution of integral equations [Rok85], [Re90] and have special structure naturally defined in terms of the associated scaling operators.

Increased interest to Cauchy-like matrix computations is partly due also to the development originated in [P90], [GKO95] and [H95]. Namely, in [P90], it was proposed to apply some transformations among the listed classes of structured matrices (that is, among the matrices of Toeplitz, Hankel, Cauchy and Vandermonde types) as a general tool for improving the known algorithms for computations with such matrices. Some sample transformations of this kind were shown in [P90]. Then [GKO95] and [H95] showed that the transformation from Toeplitz-like to Cauchy-like matrices can be achieved by very simple means, essentially by fast Fourier transform (FFT); this enabled substantial practical improvement of the known Toeplitz and Toeplitz-like solvers by their reduction to Cauchy-like solvers.

Applications to solving Toeplitz and Toeplitz-like linear systems of equations and to rational in-

terpolation give us major motivation for our main topic of designing effective algorithms for computations with Cauchy and Cauchy-like matrices, in particular, for solving linear systems of equations with Cauchy-like coefficient matrices and recursive factorization of Cauchy-like matrices.

A known explicit formula for the inverse of a Cauchy matrix (cf. e.g. [BP94], p.131) produces a good Cauchy solver but this is not enough in the applications to Toeplitz linear solvers. We will show a Cauchy-like linear solver, which extends the well known divide-and-conquer MBA algorithm, proposed in [M74], [M80], [BA80] as a Toeplitz-like linear solver. We will apply recursive factorization to nonsingular Cauchy and Cauchy-like matrices, and hereafter we will call such a factorization *CRF (or complete recursive factorization)*. CRF can be obtained by applying Gauss-Jordan elimination to 2×2 block matrix A and then, recursively, to some input and output blocks of half-size ([St69], [M74], [M80], [BA80]). It turns out that every recursive divide-and-conquer step of the algorithm can be reduced to Trummer's celebrated problem, that is, to the problem of multiplication of a Cauchy matrix C by a vector [PACPS98].

The entire algorithm requires small memory (of order of n rather than n^2, for $n \times n$ matrices) and can be performed very fast (by using order of $n \log^3 n$ arithmetic operations versus n^3 for general matrices). Elaboration of this algorithm and its application to rational interpolation involves advanced mathematical machinery, in particular manipulation with linear operators associated to strucured matrices, randomization techniques and the techniques of computational linear algebra.

We will organize our presentation as follows. In the next chapter, we will recall some definitions and basic properties of structured matrices, in particular, Toeplitz, Hankel, Cauchy and Vandermonde matrices, and matrices having similar structure (we will call them Toeplitz-like, Hankel-like, Cauchy-like and Vandermonde-like). In chapter 3, we will recall the techniques of CRF *(complete recursive factorization)* for a general matrix and its applications to the solution of linear systems and the computation of matrix inverse and determinant. In chapter 4, we will show how the CRF computation is simplified dramatically in the case of Cauchy-like input matrix with immediate application to the solution of Cauchy-like linear systems. The computations are reduced to the solution of Trummer's problem (that is, the problem of multiplication of an $n \times n$ Cauchy matrix C by a vector), which is also well-known as the basis for the solution of several other important problems of scientific and engineering computing [PACLS98]. In chapter 5, we extend the CRF approach to superefast solution of a singular Cauchy-like linear system of equations over any field of constants and, furthermore, to

superfast computation of the rank of a Cauchy-like matrix and a basis for its null space, by using the randomization technique.

Chapter 2

Structured Matrices

2.1 General and Dense Structured Matrices

Computation with dense structured matrices are widely applied in the areas of control, signal processing, solution of PDE's and algebraic computing. The complexity of computations with $n \times n$ dense structured matrices, such as Toeplitz, Hankel, Vandermonde and Cauchy, dramatically decreases in comparison with the case of $n \times n$ general matrices, that is, from the order of n^2 words of storage space and n^ω arithmetic operations (*to which we will refer hereafter as ops*) with $2.37 < \omega \leq 3$ in the best algorithms, to $O(n)$ words of storage space and to $O(n \log^2 n)$ ops (and sometimes to $O(n \log n)$ ops) (see table 1). We refer the readers to [Bun], [K87], [G84], [Rok85], [Tur], [ODR89] and [Ger] on some of the numerous practical and theoretical applications of computations with structured matrices and will focus hereafter on computations with Cauchy-like matrices.

Table 1

	Structured matrices	**General matrix**
Memory	$O(n)$	$O(n^2)$
Matrix $\times$ Vector	$O(n \log n)(FFT)$ $O(n \log^2 n)$	$O(n^2)$
Linear Solver	$O(n \log n)$ $O(n \log^2 n)$ $O(n \log^3 n)$	$O(n^3)$

2.2 Definitions and Basic Facts

Example 2.2.1.

Some examples of structured matrices.

Toeplitz matrix, $T = \left[t_{i-j} \right]$

$$\begin{bmatrix} t_0 & t_{-1} & \cdots & \cdots & t_{-n+1} \\ t_1 & t_0 & t_{-1} & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & t_0 & t_{-1} \\ t_{n-1} & \cdots & \cdots & t_1 & t_0 \end{bmatrix}$$

Hankel matrix, $H = \left[h_{i+j} \right]$.

$$\begin{bmatrix} h_0 & h_1 & h_2 & \cdots & h_{n-1} \\ h_1 & h_2 & & \iddots & h_n \\ h_2 & & \iddots & \iddots & \vdots \\ \vdots & h_{n-1} & h_n & & h_{2n-3} \\ h_{n-1} & h_n & \cdots & h_{2n-3} & h_{2n-2} \end{bmatrix}$$

Vandermonde matrix, $V = \left[x_i^{j-1} \right]$

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{bmatrix}$$

Cauchy matrix, $C = \left[\frac{1}{x_i - y_j} \right]$

$$\begin{bmatrix} \frac{1}{x_1-y_1} & \cdots & \frac{1}{x_1-y_n} \\ \vdots & & \vdots \\ \frac{1}{x_n-y_1} & \cdots & \frac{1}{x_n-y_n} \end{bmatrix}$$

Our goal is the study of Cauchy-like matrix computations, but we recall also the definitions of some other dense structured matrices (in Example 2.2.1, Definitions 2.2.2, 2.2.3 and 2.2.5 and equations (2.2.3)-(2.2.6) (cf. [BP94], [GO94a], [H95])

Definition 2.2.1 *Hereafter we will write* $D(\vec{v}) = diag(v_0, v_1, \cdots, v_{n-1})$ *for* $\vec{v} = (v_0, v_1, \cdots, v_{n-1})^T$ *to denote the diagonal matrix with the diagonal entries* v_0, $v_1, \ldots, v_{n-1}$. W^T *will denote the transpose of a matrix or a vector* W. *Rank* W *is the rank of a matrix* W.

Definition 2.2.2 *(See Example 2.2.1) We define Toeplitz, Hankel, Vandermonde and Cauchy matrices* $T = (t_{i,j})$, $H = (h_{i,j})$, $V(\vec{x}) = (v_{i,j})$, $C(\vec{s}, \vec{t}) = (c_{i,j})$, *respectively, where* $t_{i+1,j+1} = t_{i,j}$, $h_{i+1,j-1} = h_{i,j}$, $v_{i,j} = x_i^j$, $c_{i,j} = \frac{1}{s_i - t_j}$, $\vec{s} = (s_i)$, $\vec{t} = (t_i)$, $\vec{x} = (x_i)$, $s_i \neq t_j$, *for all* i *and* j *for which the above values are defined.*

We will define some classes of matrices related to Toeplitz, Hankel, Vandermonde and Cauchy matrices and we will call them *Toeplitz-like, Hankel-like, Vandermonde-like* and *Cauchy-like* matrices.

Definition 2.2.3 *For a fixed field* $\mathbf{F}$ *(say,* $\mathbf{F} = \mathbf{C}$, *the field of complex numbers) and for a pair of* n*-dimensional vectors* $\vec{q} = (q_i)$ *and* $\vec{t} = (t_j)$, $q_i \neq t_j$, $i, j = 0, \ldots, n-1$, *an* $n \times n$ *matrix* $A \in \mathbf{F}^{n \times n}$ *is a Cauchy-like matrix if*

$$F_{[D(\vec{q}),D(\vec{t})]}(A) = D(\vec{q})A - AD(\vec{t}) = GH^T, \tag{2.2.1}$$

$G, H \in \mathbf{F}^{n \times r}$, *and* $r = O(1)$. *The pair of matrices* (G, H^T) *of (2.2.1) is a* $[D(\vec{q}), D(\vec{t})]$*-generator (or a scaling generator) of a length (at most)* r *for* A *(s.g.*$_r(A)$*). The minimum* r *allowing the representation (2.2.1) is equal to rank* $F_{[D(\vec{q}),D(\vec{t})]}(A)$ *and is called the* $[D(\vec{q}), D(\vec{t})]$*-rank (or the scaling rank) of* A.

We will next recall some known properties of Cauchy-like matrices.

Lemma 2.2.1 *[GO94a]. Let* A, $\vec{q}$, $\vec{t}$, $G = [\vec{g}_1, \ldots, \vec{g}_r] = (\vec{u_i}^T)_{i=0}^{n-1}$, $H = [\vec{h}_1, \ldots, \vec{h}_r] = (\vec{v_j}^T)_{j=0}^{n-1}$ *be as in Definition 2.2.3, such that (2.2.1) holds. Then*

$$A = \sum_{m=1}^{r} diag(\vec{g}_m) C(\vec{q}, \vec{t}) diag(\vec{h}_m) = \left(\frac{\vec{u_i}^T \vec{v_j}}{q_i - t_j} \right)_{i,j=0}^{n-1}, \tag{2.2.2}$$

where $C(\vec{q}, \vec{t})$ *is a Cauchy matrix and vice versa (2.2.2) implies (2.2.1).*

It follows from (2.2.2) that (2.2.1) is satisfied if A is of the form $\left(\frac{\vec{u_i}^T\vec{v_j}}{q_i-t_j}\right)_{i,j=0}^{n-1}$, where $\vec{u_i}$ and $\vec{v_j}$ are r-dimensional vectors for $i,j=0,1,\ldots,n-1$. A Cauchy matrix $C(\vec{q},\vec{t})$ and a Loewner matrix $(\frac{r_i-s_j}{q_i-t_j})_{i,j=0}^{n-1}$ are two important special cases of Cauchy-like matrices; they have $[D(\vec{q}),D(\vec{t})]$-*ranks* 1 and 2, respectively.

Lemma 2.2.2 *(see appendix A). Given an $n\times n$ Cauchy matrix A and an n-dimensional vector $\vec{v}$, the product $A\vec{v}$ can be computed in $O(n\log^2 n)$ ops. Consequently, if A is an $n\times n$ Cauchy-like matrix given with an $s.g._r(A)$, then the product $A\vec{v}$ can be computed in $O(nr\log^2 n)$ ops.*

Lemma 2.2.3 *Let $A_i\in\mathbf{F}^{n\times n}$, $i=1,2$, denote two Cauchy-like matrices, such that $F_{[D(\vec{q_i}),D(\vec{q}_{i+1})]}(A_i)=G_iH_i^T$, G_i, $H_i\in\mathbf{F}^{n\times r_i}$, $i=1,2$; $\vec{q_j}\in\mathbf{F}^{n\times 1}$, j=1,2,3, and the vectors $\vec{q_1}$ and $\vec{q_3}$ share no components. Then the matrix $A=A_1A_2$ is a Cauchy-like matrix with $F_{[D(\vec{q_1}),D(\vec{q_3})]}(A)=GH^T$, $G=[G_1,A_1G_2]$, $H=[A_2^TH_1,H_2]$, G, $H\in\mathbf{F}^{n\times r}$, $r=r_1+r_2$. Furthermore, $O(nr_1r_2\log^2 n)$ ops suffice to compute G and H.*

Proof. Observe that

$$F_{[D(\vec{q_1}).D(\vec{q_3})]}(A_1A_2)=G_1H_1^TA_2+A_1G_2H_2^T=GH^T,$$

$G=[G_1,A_1G_2]$, $H=[A_2^TH_1,H_2]$. To deduce the desired complexity bound of $O(nr_1r_2\log^2 n)$ ops for obtaining G and H, apply Lemmas 2.2.1 and 2.2.2.

Corollary 2.2.1 *Lemma 2.2.3 can be easily extended to the computation of the product of k Cauchy-like matrices for any fixed integer $k\geq 2$.*

Lemma 2.2.4 *[H95]. Let A denote an $n\times n$ nonsingular Cauchy-like matrix with*

$$F_{[D(\vec{q}).D(\vec{t})]}(A)=GH^T,\quad \textit{where}\quad G=[\vec{g}_1,\cdots,\vec{g}_r]\in\mathbf{F}^{n\times r}\ \textit{and}\ H=[\vec{h}_1,\cdots,\vec{h}_r]\in\mathbf{F}^{n\times r}.$$

Then A^{-1} is also a Cauchy-like matrix such that $F_{[D(\vec{t}),D(\vec{q})]}(A^{-1})=-UV^T$, where the matrices $U=[\vec{u}_1,\cdots,\vec{u}_r]\in\mathbf{F}^{n\times r}$, $V=[\vec{v}_1,\cdots,\vec{v}_r]\in\mathbf{F}^{n\times r}$ satisfy $AU=G$, $V^TA=H^T$.

Corollary 2.2.2 *Under assumptions of Lemma 2.2.4, rank $F_{[D(\vec{t}),D(\vec{q})]}(A^{-1})\leq r$.*

Lemma 2.2.5 *Let an $n\times n$ Cauchy-like matrix A satisfy (2.2.1) and let $B_{I,J}$ be a $k\times d$ submatrix of A, $1\leq k,d\leq n$. Then $B_{I,J}$ is a Cauchy-like matrix with a $[D(\vec{q}_I),D(\vec{t}_J)]$-generator of a length at most r, where $I=[i_1,\ldots,i_k]$, $J=[j_1,\ldots,j_d]$, $D(\vec{q}_I)=diag(q_{i_1},\ldots,q_{i_k})$, $D(\vec{t}_J)=diag(t_{j_1},\ldots,t_{j_d})$.*

Proof. Deduce from (2.2.1) that

$$B_{I,J} = (\frac{\vec{u}_i^T \vec{v}_j}{q_i - t_j})_{i=i_1,j=j_1}^{i_k,\ j_d}$$

and recall (2.2.2).

Lemma 2.2.6 *Let A and B denote a pair of $n \times n$ Cauchy-like matrices. Let A satisfy (2.2.2) and let*

$$F_{[D(\vec{q}),D(\vec{t})]}(B) = D(\vec{q})B - BD(\vec{t}) = XW^T,$$

$$B = (\frac{\vec{x}_i^T \vec{w}_j}{q_i - t_j})_{i,j=0}^{n-1},$$

where $X^T = [\vec{x}_0, \cdots, \vec{x}_{n-1}] \in \mathbf{F}^{r_1 \times n}$, $W^T = [\vec{w}_0, \cdots, \vec{w}_{n-1}] \in \mathbf{F}^{r_1 \times n}$. Then the matrices $A + B$ and $A - B$ are Cauchy-like matrices associated with a $[D(\vec{q}), D(\vec{t})]$-generator of length at most $r + r_1$.

Proof. We have

$$A - B = (\frac{\vec{z}_i^T \vec{y}_j}{q_i - t_j})_{i,j=0}^{n-1},$$

where $\vec{z}_i^T = (\vec{u}_i^T,\ \ \vec{x}_i^T)$, $\vec{y}_j^T = (\vec{v}_j^T,\ \ -\vec{w}_j^T)$, that is, $\vec{z}_i, \vec{y}_j \in \mathbf{F}^{(r+r_1)\times 1}$, which proves the lemma for the matrix $A - B$ (similar proof applies to the matrix $A + B$).

Lemma 2.2.7 *An $n \times n$ Cauchy matrix $C(\vec{q}, \vec{t})$ is well-defined and nonsingular if and only if all the $2n$ components of the vectors $\vec{q}$ and $\vec{t}$ are distinct. Furthermore, every square submatrix of a nonsingular Cauchy matrix is nonsingular.*

Lemma 2.2.8 *([PACPS98]). Let $T(n)$ ops suffice to multiply an $n \times n$ Cauchy matrix by a vector $\vec{v}$. Let A be an $n \times n$ Cauchy-like matrix given with an $s.g._r(A)$. Then the product $A\vec{v}$ can be computed in $(3n + T(n))r$ ops.*

Definition 2.2.4 *Let $\nabla : \mathbf{F}^{m\times n} \mapsto \mathbf{F}^{m\times n}$ be an operator, let $A \in \mathbf{F}^{m\times n}$, and let $G \in \mathbf{F}^{m\times d}$ and $H \in \mathbf{F}^{n\times d}$ denote two matrices such that $\nabla(A) = GH^T$. Then $r = rank(\nabla(A))$, the rank of the matrix $\nabla(A)$, is called the ∇-rank of A, and the pair of the matrices G and H is called a ∇-generator of A of length r.*

Let X and Y be matrices from $\mathbf{C}^{n\times n}$ and refer to the matrix

$$\nabla_{[X,Y]}(A) = A - XAY \tag{2.2.3}$$

as $[X,Y]$-displacement of the matrix $A \in \mathbf{C}^{n\times n}$. Then let us specify displacement operators associated to Toeplitz-type, Hankel-type and Vandermonde-type matrices as follows.

$$\nabla(A) = \nabla_{[Z_\phi, Z_{\frac{1}{\phi}}^T]}(A) = A - Z_\phi A Z_{\frac{1}{\phi}}^T, \tag{2.2.4}$$

$$\nabla(A) = \nabla_{[Z_\phi, Z_\phi^T]}(A) = A - Z_\phi A Z_\phi^T. \tag{2.2.5}$$

$$\nabla(A) = \nabla_{[D(\vec{v}), Z_{\frac{1}{\phi}}^T]}(A) = A - D(\vec{v}) A Z_{\frac{1}{\phi}}^T, \tag{2.2.6}$$

where $\phi \neq 0$ and

$$Z_\phi = \begin{pmatrix} 0 & \dots & 0 & \phi \\ 1 & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 1 & 0 \end{pmatrix}.$$

Definition 2.2.5 *An $m \times n$ matrix that has ∇-rank bounded from above by a constant independent of m and n is called Toeplitz-like if ∇ is the operator defined in (2.2.4), Hankel-like if ∇ is the operator defined in (2.2.5) and Vandermonde-like if ∇ is the operator defined in (2.2.6).*

2.3 Structured Matrices: Correlations to Polynomial Computations

The entries of the matrices of Definition 2.2.2 are related to each other via some operators of displacement and/or scaling (e.g. $t_{i,j}$ and $h_{i,j}$ are invariant in their displacement along the diagonal or antidiagonal directions, respectively). All entries of such an $n \times n$ structured matrix can be expressed via a few parameters (from n to $2n$, versus n^2 for a general $n \times n$ matrix). Such matrices can be multiplied by vectors fast as this task reduces to some basic operations with polynomials. For instance, polynomial product

$$\left(\sum_i u_i x^i\right)\left(\sum_j v_j x^j\right) = \sum_k w_k x^k,$$

Toeplitz-by-vector product $T\vec{v} = \vec{w}$ and Hankel-by-vector product $H\vec{v} = \vec{w}$ can be made equivalent by matching properly the matrices T, H and the vector $\vec{u} = (u_i)$ (see [BP94], pp. 132-133). Likewise, the product

$$V(\vec{x})\vec{p} = \vec{v} \tag{2.3.1}$$

represents the vector $\vec{v} = (v_i)$ of the values of the polynomial $p(x) = \sum_j p_j x^j$ on a node set $\{x_i\}$. Consequently, known fast algorithms for the relevant operations with polynomials also apply to the associated matrix operations and vice versa. In particular (cf. [BP94]), $O(n \log n)$ ops suffice for polynomial and Toeplitz(Hankel)-by-vector multiplication, and $O(n \log^2 n)$ ops suffice for multipoint polynomial evaluation (p.e.) as well as for the equivalent operations with the matrix $V(\vec{x})$ and the vectors $\vec{p}$ and $\vec{v}$ of (2.3.1), assuming the input size $O(n)$ in all cases. Here and hereafter, *"ops"* stand for "arithmetic operations", *"p.e."* for "multipoint polynomial evaluation". In important special case of (2.3.1), $\vec{x} = \vec{w} = (w_n^i)_{i=0}^{n-1}$ is the vector of the n-th roots of 1, $w_n = \exp(2\pi\sqrt{-1}/n)$, $w_n^n = 1$. In this case, we write $F = V(\vec{w})/\sqrt{n}$, and p.e. turns into discrete Fourier transform, takes $O(n \log n)$ ops (due to FFT) and allows numerically stable implementation, in sharp contrast with the case of general $V(\vec{x})$. The numerical stability requirement is practically crucial and motivated the design of fast *approximation algorithms* for the p.e. [PLST93], [PZHY97], which rely on expressing $V(\vec{x})$ of (2.3.1) via Cauchy matrices, e.g. as follows:

$$V(\vec{x}) = \frac{1}{\sqrt{n}}\text{diag}\,(1 - x_i^n)_{i=0}^{n-1}\, C(\vec{x}, \vec{w})\text{diag}\,(w_i)_{i=0}^{n-1}\, F. \tag{2.3.2}$$

Here we use $\vec{x}$ and $\vec{w}$ defined above; here and hereafter, $\text{diag}\,(h_i)_{i=0}^{n-1} = D(\vec{h})$ for $\vec{h} = (h_i)_{i=0}^{n-1}$ denotes the $n \times n$ diagonal matrix with diagonal entries $h_0, \ldots, h_{n-1}$.

Remark 2.3.1 *(2.3.2) reduces the operations of (2.3.1) with $V(\vec{x})$ to ones with $C(\vec{x}, \vec{w})$, which brings us to Trummer's problem (that is, the problem of multiplication of an $n \times n$ Cauchy matrix by a vector) [PACLS98]. Its solution by Multipole Algorithm leads to a p.e. algorithm based on (2.3.2), which is both fast in terms of ops used and (according to experimental tests of [PZHY97]) numerically stable even on the inputs where the known $O(n \log^2 n)$ algorithms fail numerically, due to round-off errors.*

In the Appendix, we will extend the results of this section to yield fast algorithms for Trummer's problem of multiplication of a Cauchy matrix by a vector.

Chapter 3

Recursive Factorization (RF) of General Matrices and Extensions

3.1 Introduction to Recursive Factorization of General Matrices

Definition 3.1.1 *A matrix A is strongly nonsingular if all its leading principal submatrices are nonsingular.*

Definition 3.1.2 *I and O denote the identity and null matrices of appropriate sizes. det A is the determinant of a square matrix A.*

For an $n \times n$ strongly nonsingular matrix A, we have the following identities:

$$A = \begin{pmatrix} I & O \\ EB^{-1} & I \end{pmatrix} \begin{pmatrix} B & O \\ O & S \end{pmatrix} \begin{pmatrix} I & B^{-1}C \\ O & I \end{pmatrix}, \tag{3.1.1}$$

$$A^{-1} = \begin{pmatrix} I & -B^{-1}C \\ O & I \end{pmatrix} \begin{pmatrix} B^{-1} & O \\ O & S^{-1} \end{pmatrix} \begin{pmatrix} I & O \\ -EB^{-1} & I \end{pmatrix}, \tag{3.1.2}$$

where

$$A = \begin{pmatrix} B & C \\ E & G \end{pmatrix}, \qquad S = G - EB^{-1}C. \tag{3.1.3}$$

$S = S(B, A)$ is an $(n-k) \times (n-k)$ matrix, called the *Schur complement* of a $k \times k$ matrix B in A. Factorization (3.1.1) represents block Gauss-Jordan elimination applied to the 2×2 block matrix A of (3.1.3). If the matrix A is strongly nonsingular, then the Schur complement matrix S can be obtained in $n-k$ steps of Gaussian elimination. Using the previous factorization, we would be able to reduce the basic operations with matrix A, that is, matrix-by-vector, matrix-by-matrix multiplication and matrix inversion, to operations with the block matrices. By expanding (3.1.2) we obtain

$$A^{-1} = \begin{pmatrix} B^{-1} + B^{-1}CS^{-1}EB^{-1} & -B^{-1}CS^{-1} \\ -S^{-1}EB^{-1} & S^{-1} \end{pmatrix}. \tag{3.1.4}$$

Lemma 3.1.1 *(cf. [BP94]) If A is strongly nonsingular, so are B and S.*

Lemma 3.1.2 *(cf. [BP94]) Let A be an $n \times n$ strongly nonsingular matrix and let S be defined by (3.1.3). Let A_1 be a leading principal submatrix of S and let S_1 denote the Schur complement of A_1 in S. Then S^{-1} and S_1^{-1} form the respective southeastern blocks of A^{-1}.*

Lemma 3.1.3 *If (3.1.1) holds, then $\det A = (\det B)\det S$.*

Due to Lemma 3.1.1, we may extend the factorization (3.1.1) of A to the submatrix B and to its Schur complement S, and we may recursively continue this factorization process until we arrive at 1×1 matrices ([St69], [M74], [M80], [BA80]). In this process, we descend from A to the matrices B, C, E, and S, and then we similarly descend recursively from B and S to their submatrices and Schur complements. At these descending stages, we only identify the matrices involved in the recursion but do not compute them. For their actual computation, we recursively proceed *bottom up*, that is, we first invert the 1×1 leading principal submatrix A_1 of A, then use A_1^{-1} to compute the Schur complement S_1 of A_1 in the 2×2 leading principal submatrix A_2 of A, then invert the 1×1 matrix S_1 and the 2×2 matrix A_2, in the latter case, based on its factorization of the form (3.1.2), where the inverses A_1^{-1} and S_1^{-1} have already been computed. We recursively continue this lifting process until we arrive at A^{-1}. As a by-product, we compute all the matrices defined in the recursive descending process.

The entire computation will be called the *CRF (or complete recursive factorization)* of A. Besides the inversion of 1×1 matrices, the CRF only requires matrix multiplications and subtractions.

Hereafter, we will always assume *balanced* CRFs, that is, we will balance the factorization (3.1.1) in its first step, such that B is the $\lfloor \frac{n}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor$ submatrix of A, and we will maintain the similar balancing property in all the subsequent recursive steps. The balanced CRF has depth at most $d = \lceil \log_2 n \rceil$.

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

Algorithm 3.1.1. *Recursive triangular factorization and inversion of a strongly nonsingular matrix.*

Input: a strongly nonsingular $n \times n$ matrix A of (2.2.2).

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

Computations:

1. Apply Algorithm 3.1.1 to the matrix B (replacing A as its input) in order to compute the balanced CRF of B (including B^{-1}).

2. Compute the Schur complement $S = G - EB^{-1}C$.

3. Apply Algorithm 3.1.1 to the matrix S (replacing A as its input) to compute the balanced CRF of S (including S^{-1}).

4. Compute A^{-1} from (3.1.2).

Clearly, given A^{-1} and a vector $\vec{b}$, we may immediately compute the vector $\vec{x} = A^{-1}\vec{b}$. If we also seek $\det A$, then it suffices to add the request for computing $\det B$, $\det S$, and $\det A$ at stages 1, 3, and 4, respectively.

CRF was long recognized as an effective tool for the computation of A^{-1}, $\det A$ and the solution $\vec{x} = A^{-1}\vec{b}$ to a linear system $A\vec{x} = \vec{b}$ via matrix multiplication [St69], [AHU74]. [M74], [M80] and [BA80] applied CRF to numerical computation with Toeplitz-like matrices. [K95] and [PZ,a] applied CRF to (possibly singular) Toeplitz-like matrices over any field. We refer the reader also to [PR89], [PR91], [PR93] and [P93] an application of CRF to the sparse matrix computations and path algebra computations.

3.2 Control over the Precision of the Computation of the RF

Let us comment on bounding the precision of the computations where the RF is applied to the

computation of $\vec{x} = A^{-1}\vec{b}$, A^{-1} and $\det A$ over the rationals. Towards this goal, we may extend to the Cauchy-like case the algorithm of [K95], which computes RF of Toeplitz-like matrices and assumes all arithmetic operations performed exactly with infinite precision. Such a computation, however, is generally expensive over the rationals because it requires to increase the precision of computing dramatically. A customary ways out of this difficulty is to compute the output modulo a large integer K and then recover the rational output by applying the continued fraction approximation algorithm [Wang], [WGD], [Ab] and [KR].

To recover the output fraction $\frac{y}{x}$ of two integers y and x, one should choose K exceeding $2|x|^2$ and $2|y|^2$. An alternative is to scale the input matrix A to turn all its entries into integers. Then $\det A$ and the entries of $\text{adjA} = A^{-1}\det A$ are integers, not exceeding $N = \|A\|^n$ for any matrix norm, and it is sufficient to compute $((\det A) \bmod K)$ and $((\text{adjA}) \bmod K)$ for $K > 2N$. Then the values $\det A$ and adjA can be recovered immediately.

We may avoid dealing with large numbers by performing computations modulo several smaller primes and then recover all the input values modulo their product by applying the Chinese Remainder algorithm. An alternative is to fix a single prime p, compute the output modulo p, and then lift all inverse matrices involved into the RF and, therefore, lift the entire RF to obtain all values modulo p^{2^k} in k Newton's iterations for matrix inverses (cf. [BP94], Algorithm 3.1 on p.243), each of which essentially amounts to two matrix multiplications.

Chapter 4

Recursive Factorization of Cauchy-like Matrices

As we recalled in chapter 2, the structure of $n \times n$ dense structured matrices (defined by $O(n)$ parameters) enables dramatic acceleration of computations with such matrices. For example, an $n \times n$ Toeplitz matrix $T = [t_{i-j}]$ can be multiplied by a vector over any field of constants by using

$$T_{M_v}(n) = O((n \ \log \ n) \log \log n) \tag{4.1.1}$$

arithmetic operations [BP94] versus $2n^2 - n$ for general $n \times n$ matrices (hereafter, we will keep definitions of the previous chapter, in particular, we will refer to arithmetic operations as *ops*). Furthermore, by extending fast multiplication of structured matrices by a vector, one may compute the RF of structured matrices much faster than in the case of general matrices. In the Toeplitz-like case, this was done by extending the well known divide-and-conquer *MBA algorithm*, proposed in [M74], [M80] and [BA80]. The MBA algorithm rapidly computes the recursive triangular factorization of A, as well as A^{-1}, det A, and the solution $\vec{x} = A^{-1}\vec{b}$ to a linear system $A\vec{x} = \vec{b}$. The algorithm applies to the class of strongly nonsingular Toeplitz-like matrices A and uses a total of $A_{RF}(n) = O(A_{M_v}(n) \log n) = O(n \log^2 n)$ ops, where $A_{M_v}(n) = O(n \log n)$ is the complexity of multiplication of an $n \times n$ Toeplitz matrix A by a vector by means of FFT. The algorithm is called *superfast* because it runs in almost linear time, versus cubic time of Gaussian elimination and quadratic time of some known faster solutions. It became a natural technical challenge to extend such an algorithm for the cited Toeplitz-like matrix computations to other classes of dense structured matrices, in particular, to *Cauchy-like* matrices. While it is clear

that the superfast computation of the CRF of a Cauchy-like matrix should follow the same divide-and-conquer scheme as in the MBA algorithm, that is, one should apply Algorithm 3.1.1 but operate with the scaling generators rather than with matrices, the extension of the MBA algorithm to the case of a Cauchy-like input is not straightforward due to the difference in the structure of the input matrix; in particular, treatment of scaling operators (versus displacement operators of the MBA algorithm) requires distinct techniques. Such an extension will be our goal in this chapter and in Chapter 5. We will follow [PACPS98], [OP98] and [PACP99].

4.1 Recursive Factorization of a Strongly Nonsingular Cauchy-like Matrix: Bounds on the Scaling Ranks.

We use $C_{RF}(n) = O(C_{Mv}(n)\log n)$ ops, where $C_{Mv}(n)$ denotes the complexity of Trummer's problem (that is, the problem of multiplication of an $n \times n$ Cauchy matrix by a vector) which can be solved in $O(n\log^2 n)$ ops [PACLS98]. Obviously, $C_{RF}(n) = O(n\log^3 n)$ ops is the complexity bound for Cauchy-like recursive triangular factorization (CRF) and, consequently, for the related Cauchy-like computations (including matrix inversion, linear system solving and computation of the matrix determinant). The complexity bound is slightly inferior to the Toeplitz-like bound.

Hereafter, we will assume for simplicity that $n = 2^d$ is an integer power of 2. We write $\vec{q} = (q_i)_{i=0}^{n-1}$, $\vec{q}^{\,(1)} = (q_i)_{i=0}^{\frac{n}{2}-1}$, $\vec{q}^{\,(2)} = (q_i)_{i=\frac{n}{2}}^{n-1}$, $\vec{t} = (t_i)_{i=0}^{n-1}$, $\vec{t}^{\,(1)} = (t_i)_{i=0}^{\frac{n}{2}-1}$, $\vec{t}^{\,(2)} = (t_i)_{i=\frac{n}{2}}^{n-1}$. We will start with some auxiliary results.

Lemma 4.1.1 *Let A be an $n\times n$ strongly nonsingular Cauchy-like matrix with $F_{[D(\vec{q}),D(\vec{t})]}(A) = GH^T$, G, $H \in \mathbf{F}^{n\times r}$. Let A, B, C, E, G, satisfy (3.1.3) and S is the Schur complement. Then*

$$rank F_{[D(\vec{t}),\ D(\vec{q})]}(A^{-1}) \le r, \qquad rank F_{[D(\vec{t}^{\,(1)}),\ D(\vec{q}^{\,(1)})]}(B^{-1}) \le r, \tag{4.1.1}$$

$$rank F_{[D(\vec{q}^{\,(2)}),\ D(\vec{t}^{\,(2)})]}(S) \le r, \tag{4.1.2}$$

$$rank F_{[D(\vec{t}^{\,(2)}),\ D(\vec{q}^{\,(2)})]}(S^{-1}) \le r, \tag{4.1.3}$$

$$rank F_{[D(\vec{q}^{(1)}),\ D(\vec{t}^{(2)})]}(C) \le r, \qquad rank F_{[D(\vec{q}^{(2)}),\ D(\vec{t}^{(1)})]}(E) \le r. \tag{4.1.4}$$

Proof. Deduce (4.1.4) from Lemma 2.2.5. By applying Corollary 2.2.2, obtain (4.1.1). Now, due to Lemmas 3.1.2 and 2.2.5, we have (4.1.3). Then we apply Corollary 2.2.2 to the matrix $S = (S^{-1})^{-1}$ and obtain (4.1.2).

Fact 4.1.1 *(cf. Proposition A.6 of [P92b], [P93a]). Given an $s.g._{r^*}(A) = (G, H)$ and the scaling rank r of A, $r < r^* \le n$, one can compute an $s.g._r(A)$ by using $O((r^*)^2 n)$ ops.*

4.2 Computational Complexity Estimates of Cauchy-like CRF

As we mentioned in the introduction to this chapter, the **superfast** extension of Algorithm 3.1.1 to a strongly nonsingular Cauchy-like matrix A goes simply by operating with short scaling generators of all matrices involved, rather than with the entries of these matrices, and the only problem is to control the length of the generators. To solve this problem, we apply Lemma 4.1.1 and Fact 4.1.1. The next theorem supplies the computational complexity estimates of the resulting algorithm, to which we will refer as **Algorithm 4.2.1.**

Theorem 4.2.1 *Let A denote an $n \times n$ strongly nonsingular Cauchy-like matrix with its F-generator of a length r for the operator $F = F_{[D(\vec{q}),D(\vec{t})]}$. Then the respective F-generators of all the matrices encountered in the balanced CRF of A (including an $s.g._r(A^{-1})$) and $\det A$ can be computed in $O(r^2 T(n) \log n) = O(nr^2 \log^3 n)$ ops (for $T(n)$ of Lemma 2.2.8) and can be stored by using $O(nr \log n)$ words of storage space.*

Proof. Let us apply the fast version of Algorithm 3.1.1 to the matrix A of Theorem 4.2.1, that is, instead of slower computations with matrices, let us perform faster computations with their short scaling generators. Let $\phi_r(n)$ ops be involved in computing the balanced CRF of A (including the computation of an $s.g._r(A^{-1})$). Furthermore, let $\sigma_r(n)$ ops be used for computing an $s.g._r(S)$ from given $s.g._r(B^{-1})$, $s.g._r(C)$, $s.g._r(-E)$, and $s.g._r(G)$ (cf.(3.1.3)), and let $\mu_r(n)$ ops be required for computing an $s.g._r(A^{-1})$ from given $s.g._r(B^{-1})$, $s.g._r(C)$, $s.g._r(E)$, and $s.g._r(S^{-1})$ (cf.(3.1.2)). This is summarized in table 2.

Table 2

Input	$s.g._r(A)$	$s.g._r(B^{-1}), s.g._r(C),$ $s.g._r(-E), s.g._r(G)$	$s.g._r(B^{-1}), s.g._r(C),$ $s.g._r(E), s.g._r(S^{-1})$
Output	CRF of A	$s.g._r(S)$	$s.g._r(A^{-1})$
ops	$\phi_r(n)$	$\sigma_r(n)$	$\mu_r(n)$

Let $\phi_r(k)$, $\sigma_r(k)$, and $\mu_r(k)$ denote the similar estimates where the input matrix A is replaced by a strongly nonsingular $k \times k$ matrix W given with an $s.g._r(W)$. For simplicity, let n be even. Then in view of Lemma 4.1.1, the examination of Algorithm 3.1.1 gives us the bound

$$\phi_r(n) \leq 2\phi_r(\frac{n}{2}) + \sigma_r(n) + \mu_r(n). \tag{4.2.1}$$

By expanding (3.1.2), we deduce that

$$A^{-1} = \begin{pmatrix} B^{-1} + B^{-1}CS^{-1}EB^{-1} & -B^{-1}CS^{-1} \\ -S^{-1}EB^{-1} & S^{-1} \end{pmatrix}. \tag{4.2.2}$$

Now we apply Lemmas 2.2.3, 2.2.5, 2.2.6, 4.1.1 and Fact 4.1.1 and deduce that

$$\sigma_r(n) = O(r^2T(n)), \quad \mu_r(n) = O(r^2T(n)). \tag{4.2.3}$$

Substitute (4.2.3) into (4.2.1), recursively extend (4.2.1), and deduce that

$$\phi_r(n) = O(r^2T(n)\log n),$$

with $T(n)$ as in Lemma 2.2.8, which gives us the arithmetic time bound of Theorem 4.2.1. The storage space bound follows similarly when we inspect Algorithm 3.1.1 applied to the matrix A and apply Lemmas 2.2.3, 2.2.5, 2.2.6 and 4.1.1.

The computations by the algorithm supporting Theorem 4.2.1 can be a little simplified based on the following result (this does not change the asymptotic complexity estimates of the theorem).

Proposition 4.2.1 *([GO94], [OP98]). Let A be a Cauchy-like matrix of Lemma 2.2.4, partitioned into blocks according to (3.1.1). Let (G_0, H_0), (G_0, H_1), (G_1, H_0), (G_1, H_1) and (G_S, H_S) denote*

the induced five scaling generators of the blocks B, C, E, G of A and of the Schur complement S, respectively. Then $G_S = G_1 - EB^{-1}G_0$, $H_S^T = H_1^T - H_0^T B^{-1}C$.

Remark 4.2.1 *In [OP98], Proposition 4.2.1, was extended to the case of Hankel-like matrices H associated with operators $ZH - HZ^T$, Z being a shift matrix. This enabled practical improvement of the MBA Hankel/Toeplitz linear solver.*

Remark 4.2.2 *Scaling generator of a Schur complement S of (3.1.3) has simple explicit expressions via the $s.g._r(A^{-1})$ [OP98], [PACPS98]. Using them simplifies the computation of $s.g._r(S^{-1})$ and similarly at all other stages of the algorithm, though such a simplification decreases the overall asymptotic cost of performing the algorithm only by a constant factor.*

4.3 Ensuring Strong Nonsingularity of a Nonsingular Cauchy-like Matrix by Means of Symmetrization

We have showed how to compute det A and the balanced CRF of A (including an $s.g._r(A^{-1})$), assuming strong nonsingularity of the matrix A. To extend this solution to the case of any nonsingular matrix A, we will seek a strongly nonsingular $n \times n$ matrix X, such that the matrix AX is strongly nonsingular. Then we may apply our machinery to the matrices X and AX or XA, compute $(AX)^{-1} = X^{-1}A^{-1}$ or $(XA)^{-1} = A^{-1}X^{-1}$, det (AX)=det (XA), and det X, and obtain $A^{-1} = X(AX)^{-1} = (XA)^{-1}X$ and det A = det (AX)/det X. Furthermore, if A is singular, then the same algorithm will involve a division by 0 and thus will show us that det $A = 0$. In fact, we will also extend our algorithm to computing the rank of A, in Corollary 5.4.2.

If our computation is performed in the field of real numbers (or in its subfield), then we could have set $X = A^T$. Indeed, the matrix $XA = A^TA$ is positive definite and consequently strongly nonsingular provided that A is nonsingular. Moreover, in this case the condition numbers of all diagonal blocks of the CRF do not exceed the condition number of A (cf. [GL] or [BP], Fact 2.1.4 and page 237). As a by-product, we immediately arrive at a least-squares (normal equations) solution $(A^TA)^{-1}A^T\vec{b}$ to a Cauchy-like linear system $A\vec{x} = \vec{b}$ for a Cauchy-like $m \times n$ rectangular matrix A having full rank n, $n \leq m$. A problem arises, however, when we apply Lemma 2.2.1 to the matrices $W = A^TA$ and $U = AA^T$. Indeed, if we replace A in (2.2.1) by these matrices, then we also ought to replace

$F_{[D(\vec{q}),D(\vec{t})]}$ by $F_{[D(\vec{t}),D(\vec{t})]}$ or $F_{[D(\vec{q}),D(\vec{q})]}$, respectively, and the assumption $q_i \neq t_j$ of Definition 2.2.3 is not extended. To exploit Cauchy-like structure of matrices W associated with the operator $F_{[D(\vec{t}),D(\vec{t})]}$, we may operate with W represented as the product $C^{-1}(\vec{q},\vec{t})Y$, where $Y = C(\vec{q},\vec{t})W$ and $C^{-1}(\vec{q},\vec{t})$ are Cauchy-like matrices, and similarly for U.

For computations in finite fields symmetrization does not work, but in this case we will solve our problem based on a distinct approach to defining the above matrix X, that is, by using randomization (see section 5.4).

Chapter 5

Superfast Computations with Singular Structured Matrices

Algorithm 4.2.1 requires strong nonsingularity of its input matrix. The symmetrization techniques of section 4.3 enables us to ensure this property in the case where the input matrix is nonsingular, and the computations are over the real or complex numbers. These techniques do not work over finite fields and for singular input, but such cases are important, e.g., in the applications to the decoding of algebraic codes. In this chapter, we use randomization to extend the approach to the superfast solution of a singular Cauchy-like linear system of equations over any field of constants and, furthermore, to superfast computation of the rank of a Cauchy-like matrix and a basis for its null space. The algorithms can be extended to similar computations with singular Hankel-like and Vandermonde-like matrices. The applications include rational and polynomial interpolation, Padé approximation and decoding Reed-Solomon and algebraic-geometric codes.

5.1 Introduction

In many cases, in particular in application to signal processing, Padé approximation and sparse multivariate polynomial interpolation, one needs to solve generally singular Toeplitz-like, Vandermonde-like or Cauchy-like linear systems of equations over finite fields.

Kaltofen in [K95] extended the MBA approach to the solution of a singular Toeplitz-like linear system of equations over finite fields. We will show the same thing in the Cauchy-like case. The

resulting algorithms use

$$C_{sing}(n) = O(C_{M_v}(n) \log n) \tag{5.1.1}$$

ops (in the Cauchy-like case). Here,

$$C_{M_v}(n) = O((n \log^2 n) \log \log n) \tag{5.1.2}$$

is the number of ops sufficient to solve *Trummer's problem* of multiplication of an $n \times n$ Cauchy matrix by a vector over any field of constants $\mathbf{F}$ (cf. section 4.4). If $\mathbf{F}$ supports FFT, then the factor $\log \log n$ in (5.1.2) can be removed. Our algorithms sample $4n$ random parameters from a fixed finite set S of cardinality $|S|$ in the Cauchy-like case. In Toeplitz-like case, they sample $2n - 2$ parameters (versus order of $n \log n$ in [K95]). Our algorithms may fail with a probability at most $2\rho(\rho + 1)/|S|$ where ρ is the rank of the input matrix (versus $4n\rho/|S|$ in [K95]), otherwise they produce correct output. Besides being linear solvers, our algorithms (like one of [K95]) can be immediately applied to compute (at the same asymptotic computational cost) the rank of a given matrix and a basis for its null-space and to verify the correctness of the output in all cases.

Our study of Toeplitz-like and Cauchy-like computations and, in particular, our asymptotic complexity estimates (5.1.1), (5.1.2) and (5.1.4) can be easily extended to the study of Hankel-like and Vandermonde-like computations, respectively. The bound (5.1.1) is supported directly by Algorithm 4.2.1, but it can be improved to the bound

$$T_{sing}(n) + O(V_{M_v}(n)) = O((n \log^2 n) \log \log n), \tag{5.1.3}$$

$$T_{sing}(n) = O(T_{M_v}(n) \log n) \tag{5.1.4}$$

(assuming that an $n \times n$ Vandermonde matrix can be multiplied by a vector in $V_{M_v}(n)$ ops), by means of some general techniques proposed in [P90] for the transformation (at the cost $O(V_{M_v}(n))$) of various computational problems for Cauchy-like and Vandermonde-like matrices to the same problems for Toeplitz/Hankel-like matrices. (In fact, [P90] defined such transformations among all the cited classes of structured matrices in all directions.) The difference by logarithmic factor may be not decisive in practice, because of the role of other potential criteria (such as numerical stability and the decrease of

the overhead constants hidden in the above "O" notation), so we show estimates (5.1.1) and (5.1.2) and not just (5.1.3) and (5.1.4). On various applications of Cauchy-like algorithms, see bibliography in [OP98], [PACLS98] and note that by using the cited transition to Toeplitz/Hankel computations we may save logarithmic factor in the complexity estimates versus ones of [OP98].

We will recall some definitions and auxiliary facts and we will recall also the MBA algorithm and the results of [PZ99], [PACPS98], [OP98] on its Cauchy-like extension. We apply randomization to extend the latter results to the case of a nonsingular but not strongly nonsingular input matrix. Finally, we cover the extension of Cauchy-like and Toeplitz-like solvers to the singular case.

5.2 Definitions and Basic Facts

We will use the definitions and basic facts of sections 2.2 and 4.1.

5.3 Recursive Factorization of a Cauchy-like Matrix

We will apply Algorithm 3.1.1 to a Cauchy-like input matrix A or, more generally, to a matrix given by its short scaling generator, and will perform all matrix operations with short scaling generators of the auxiliary matrices involved, thus arriving at Algorithm 4.2.1. It remains to ensure strong nonsingularity of the input matrix A, which will be our next subject.

5.4 Ensuring strong nonsingularity

For a nonsingular real or complex matrix A, the matrices A^TA and AA^T are strongly nonsingular (see section 4.3).

As we mentioned before, in the case of a singular input matrix and for computation in finite fields symmetrization does not generally work, but in this case we will solve the problem based on distinct approach, that is, by using randomization. Namely, having a nonsingular matrix A, we will apply its randomized preconditioning in order to ensure strong nonsingularity of the resulting matrix. For computations in arbitrary field $\mathbf{F}$, we will obtain a desired matrix X of section 4.3 by using random parameters always sampled from a fixed finite set S (in $\mathbf{F}$ or in its extension) independently of each

other and under the uniform probability distribution on S. We will keep using definitions of section 5.2 and rely on the following lemma.

Lemma 5.4.1 *[DL]. Let $p(\mathbf{x}) = p(x_1, x_2, \cdots, x_m)$ be a nonzero m-variate polynomial of a total degree d. Let $\mathbf{S}$ be a finite set in the domain of the definition of $p(\mathbf{x})$, let the random values $x_1^*, \cdots, x_m^*$ be sampled from $\mathbf{S}$, and let $\vec{x}^* = (x_1^*, x_2^*, \cdots, x_m^*)$. Then probability$(p(\vec{x}^*) = 0) \le d/|\mathbf{S}|$.*

Hereafter, we will fix a sufficiently large finite set S from which we will choose all random values that we need. We will choose them from S independently of each other and assuming the uniform probability distribution on S, to be able to apply Lemma 5.4.1. Then application of Lemma 5.4.1 will ensure (with a high probability) that, for an $n \times n$ nonsingular Cauchy-like matrix A given with its $F_{[D(\vec{q}),D(\vec{t})]}$-generator of a length r and for an $n \times n$ matrix X defined by its $F_{[D(\vec{t}),D(\vec{s})]}$-generator with random entries, the matrix AX is strongly nonsingular. Namely, we will arrive at the following result (see an alternative approach in the next section).

Theorem 5.4.1. *Let A be an $n \times n$ nonsingular matrix satisfying equation (2.2.2). Let X be a matrix satisfying $X = YC(\vec{q}, \vec{s})$, where*

$$Y = \sum_{m=1}^{r} D(\vec{g}_m^*)C(\vec{t}, \vec{q})D(\vec{h}_m^*) \ , \tag{5.4.1}$$

$C(\vec{q}, \vec{s}) = (\frac{1}{q_i - s_j})_{i,j=0}^{n-1}$ is a fixed nonsingular Cauchy matrix, $\vec{q} \in \mathbf{F}^{n\times 1}$, $\vec{t} \in \mathbf{F}^{n\times 1}$, $\vec{s} \in \mathbf{F}^{n\times 1}$, $\vec{q}$ and $\vec{t}$ are as in Lemma 2.2.4, $q_i \neq s_j$, $s_i \neq t_j$ for all i and j, $\vec{g}_m^* \in \mathbf{F}^{n\times 1}$, $\vec{h}_m^* \in \mathbf{F}^{n\times 1}$, $m = 1, \cdots, r$, and the $2nr$ components of the $2r$ latter vectors are random values from a fixed finite set $\mathbf{S}$. Then AX has F-rank at most $2r+1$ and, with a probability at least $1 - n(n+1)/|\mathbf{S}|$, is a strongly nonsingular matrix.

Proof. First consider matrix Y of (5.4.1), where the random vectors $\vec{g}_m^*$ and $\vec{h}_m^*$ are replaced by generic vectors whose components are indeterminates. Recall that the $F_{[D(\vec{t}),D(\vec{q})]}$-rank of A^{-1} is at most r, due to Lemma 2.2.4. Therefore, there exists an assignment of values to the components of the vectors $\vec{g}_m^*$, $\vec{h}_m^*$, for which we have $AY = I$, and then the matrix $AX = C(\vec{q}, \vec{s})$ is strongly nonsingular (cf. Lemma 2.2.7). On the other hand, the determinants of the $k \times k$ leading principal submatrices $(AX)_k$ of AX are polynomials of degrees at most $2k$ in the coordinates of $\vec{g}_m^*$, $\vec{h}_m^*$. Since $AX = C(\vec{q}, \vec{s})$ for a particular assignment, these polynomials are not identically 0 if the components are indeterminates. Therefore, by Lemma 5.4.1, we obtain *probability*$(\det(AX)_k \neq 0, k = 1, \cdots, n) \ge \prod_{k=1}^{n} \frac{1-2k}{|\mathbf{S}|} \ge \frac{1-n(n+1)}{|\mathbf{S}|}$.

Corollary 5.4.1. *Let an $n \times n$ nonsingular Cauchy-like matrix A be given with its F-generator of*

length r for the operator $F = F_{[D(\vec{q}),D(\vec{t})]}$. Then an $F_{[D(\vec{t}),D(\vec{q})]}$-generator of length at most r for A^{-1} can be computed by means of a randomized algorithm using $2nr$ random parameters (sampled from the set $\mathbf{S}$) and $O(nr^2 \log^3 n)$ ops and failing with a probability at most $\frac{n(n+1)}{|\mathbf{S}|}$.

Proof. Let us define X as above. By Theorem 5.4.1, the Cauchy-like matrix AX is strongly nonsingular with a probability at least $1 - n(n+1)/|\mathbf{S}|$, and then, by Theorem 4.2.1, we may compute the matrices $(AX)^{-1}$ and $A^{-1} = X(AX)^{-1}$ by using a total of $O(C_{M_v}(n)r^2 \log n)$ ops. Due to Lemma 5.4.1, we also obtain the desired bounds on the number of random parameters used and on the failure probabilities. Finally, we will decrease the length of the computed F-generator of A^{-1} by applying Fact 4.1.1.

Corollary 5.4.2. *det A and ρ=rank A can be computed by using $2nr$ random parameters and $O(C_{M_v}(n)r^2 \log n)$ ops for a matrix A of (2.2.1), (2.2.2). If $C(\vec{t},\vec{q})$ is a nonsingular Cauchy matrix, then (by Lemmas 2.2.7 and 5.4.1) the matrix X is strongly nonsingular, with a probability at least $1 - n(n+1)/|\mathbf{S}|$, and (by Theorem 4.2.1)* det (AX), det X, *and* det $A = \frac{\det(AX)}{\det X}$ *can be computed at the randomized cost $O(C_{M_v}(n)r^2 \log n)$. Furthermore, with a probability at least $\rho(\rho+1)/|\mathbf{S}|$, $\rho \times \rho$ is the maximum size of a nonsingular leading principal submatrix of AX for the matrix X of Theorem 5.4.1. Such a size is computed as a by-product of application of Algorithm 3.3.1 to AX.*

5.5 Fast Cauchy-like Computations - Singular Case

Studying the solution of a singular Cauchy-like linear system, we will use the next result and definition.

Lemma 5.5.1 *[K95]. Let A be an $n \times n$ matrix of rank ρ with entries from a fixed field $\mathbf{F}$ and suppose that the $\rho \times \rho$ leading principal submatrix A_ρ is nonsingular. Then for a vector $\vec{y}$ with coordinates from the field $\mathbf{F}$ the vector*

$$\vec{x} = \begin{pmatrix} A_\rho^{-1}\vec{b}\,' \\ \vec{0} \end{pmatrix} - \vec{y}$$

is a solution to $A\vec{x} = \vec{b}$, where the vector $\vec{b}\,'$ consists of the first ρ coordinates of $\vec{b} + A\vec{y}$, and $\vec{0}$ denotes the null vector of dimension $n - \rho$.

Definition 5.5.1. *Let A_i be the $i \times i$ leading principal submatrix of A, where $1 \le i \le n$. We say that A has* generic rank profile *if A_j is nonsingular for all j, $1 \le j \le \mathrm{rank}(A)$.*

The next theorem extends the known results from the Toeplitz-like (cf. [BP94], p. 206, or [KS91]) to the Cauchy-like case and can be an alternative to Theorem 5.4.1 where the input matrix is nonsingular

(see Remark 5.5.1).

Theorem 5.5.1. *For an $n \times n$ Cauchy-like matrix A of rank ρ represented by an s.g.-r(A) and satisfying (2.2.1) and (2.2.2), consider the matrix product $\overline{A} = LAM$, where L and M are also Cauchy-like matrices with scaling generators of length 1. Assume the following relations:*

$$F_{[D(\vec{s}),\ D(\vec{q})]}(L) = YZ^T,$$

$$F_{[D(\vec{t}),\ D(\vec{p})]}(M) = XW^T,$$

$$Y^T = [y_1, \cdots, y_n] \in \mathbf{F}^n,\ Z^T = [z_1, \cdots, z_n] \in \mathbf{F}^n,$$

$$X^T = [x_1, \cdots, x_n] \in \mathbf{F}^n,\ W^T = [w_1, \cdots, w_n] \in \mathbf{F}^n,$$

$$L = \left(\frac{y_{i+1}z_{j+1}}{s_i - q_j}\right)_{i,j=0}^{n-1},\quad M = \left(\frac{x_{i+1}w_{j+1}}{t_i - p_j}\right)_{i,j=0}^{n-1},$$

where the entries of the matrices Y, Z, X, W are random samples from a fixed finite subset $\mathbf{S}$ of the field $\mathbf{F}$ and where $\mathbf{S}$ does not contain 0. Let s_i, q_j, p_k be all pairwise distinct for $i, j, k = 0, \cdots, n-1$. Then

(1) L and M are strongly nonsingular matrices and

(2) $\overline{A}$ has generic rank profile with a probability at most $1 - \frac{2\rho(\rho+1)}{|\mathbf{S}|}$.

Proof: Part (1) follows from (2.2.2) and Lemma 2.2.7 since $\mathbf{S}$ does not contain 0. Let us prove part (2). For an $n \times n$ matrix D, denote by $D_{I,J}$ the determinant of the submatrix of D formed by removing from D all rows not contained in the set I and all columns not contained in the set J. First, let Y, Z, X, and W be generic matrices. For $I = [1, 2, \cdots, i]$, $J = [j_1, j_2, \cdots, j_i]$, $K = [k_1, k_2, \cdots, k_i]$, $i = 1, 2, \cdots, \rho$, we have from the Cauchy-Binet formula that

$$\overline{A}_{I,I} = \sum_J \sum_K L_{I,J} A_{J,K} M_{K,I}.$$

Let us prove that

$$\overline{A}_{I,I} \neq 0 \text{ for } i = 1,\ 2,\ \cdots,\ \rho\ . \tag{5.5.1}$$

Observe that, for a fixed pair of $J = [j_1, j_2, \cdots, j_i]$ and $K = [k_1, k_2, \cdots, k_i]$, the determinant $L_{I,J}$ has the unique term

$$a y_1 y_2 \cdots y_i z_{j_1} \cdots z_{j_i},$$

where $a \neq 0$ is a constant. Likewise, $M_{K,I}$ has the unique term

$$b x_{k_1} \cdots x_{k_i} w_1 \cdots w_i,$$

where $b \neq 0$ is a constant. Therefore, $\overline{A}_{I,I} \neq 0$ provided that there exists a pair J, K such that $A_{J,K} \neq 0$. This is true for all $i \leq \rho$, since A has rank ρ, and we arrive at (5.5.1).

Now, we observe that $\overline{A}_{I,I}$ is a polynomial of degree at most $4i$ in the variables y_m, z_m, x_m, w_m, $m = 1, \ldots, n$. Under the random choice of their values, we apply Lemma 5.4.1 and obtain that $probability(\overline{A}_{I,I} \neq 0, i = 1, \cdots, \rho) \geq \prod_{i=1}^{\rho}(1 - 4i/|\mathbf{S}|) \geq 1 - 2\rho(\rho+1)/|\mathbf{S}|$. This proves part (2) of Theorem 5.5.1.

Remark 5.5.1. *If the input Cauchy-like matrix is nonsingular, we may apply Theorem 5.5.1, as an alternative to Theorem 5.4.1. Application of Theorem 5.5.1, rather than Theorem 5.4.1, requires by factor $2/r$ fewer random parameters ($4n$ versus $2nr$) and involves scaling generators of roughly half length ($r+2$ versus $2r+1$), at the small price of doubling the probability of errors ($2n(n+1)/|\mathbf{S}|$ versus $n(n+1)/|\mathbf{S}|$).*

To prove Theorem 5.5.1, we devised a simple algorithm that, for an $n \times n$ Cauchy-like matrix A of rank ρ given with an $s.g._r(A)$, computes a random pair $s.g._1(L)$ and $s.g._1(M)$, where L and M are $n \times n$ matrices such that, with a probability at least $1 - 2\rho(\rho+1)/|\mathbf{S}|$, the matrix $\overline{A} = LAM$ has generic rank profile. Furthermore, based on Lemma 2.2.3, we computed $s.g._{r+2}(\overline{A})$ by using $O(r^2 C_{M_v}(n))$ ops (cf. (5.1.2)). Now, we assume that we have been already given $s.g._1(L)$, $s.g._1(M)$, and $s.g._{r+2}(\overline{A})$ for a pair of nonsingular matrices L and M and an $n \times n$ matrix $\overline{A} = LAM$ having generic rank profile and propose the following algorithm, using $O(r^2 C_{M_v}(n) \log n)$ ops (cf. (5.1.1)).

Algorithm 5.5.1. *Computation of the largest nonsingular leading principal inverse.*

Input: vectors $\vec{q} = (q_i)_{i=0}^{n-1}$, $\vec{t} = (t_j)_{j=0}^{n-1}$, $q_i \neq t_j$, $i, j = 0, 1, \cdots, n-1$, and $\vec{g_1}, \cdots, \vec{g}_{r+2}, \vec{h_1}, \cdots, \vec{h}_{r+2}$ such that the next matrix has generic rank profile:

$$\overline{A} = \sum_{m=1}^{r+2} D(\vec{g}_m) C(\vec{q}, \vec{t}) D(\vec{h}_m).$$

Output: An integer $\rho \leq n$ and vectors $\vec{u}_1, \cdots, \vec{u}_{\overline{r}}$, $\vec{v}_1, \cdots, \vec{v}_{\overline{r}}$, $\vec{u}_m, \vec{v}_m \in \mathbf{F}^{n \times 1}$, $m = 1, 2, \cdots, \overline{r}$, $\overline{r} \leq r+2$, such that $\rho = \mathrm{rank}(\overline{A})$ and

$$\overline{A}_{\rho}^{-1} = \sum_{m=1}^{\overline{r}} D(\vec{u}_m) C(\vec{t}, \vec{q}) D(\vec{v}_m).$$

1. Represent $\overline{A}$ as $\overline{A} = \begin{pmatrix} \overline{B} & \overline{C} \\ \overline{E} & \overline{J} \end{pmatrix}$, cf. (3.1.3), where $k = \lceil \frac{n}{2} \rceil$, and the $k \times k$ submatrix $\overline{B}$ of $\overline{A}$ is singular if and only if $k > \rho$ (since $\overline{A}$ has generic rank profile). Apply Algorithm 5.5.1 recursively to the input matrix $\overline{B}$ replacing $\overline{A}$. (Note that we are given an $s.g._r(\overline{B})$.) If $\rho \geq k$, the output of this stage is the desired output of the algorithm. Otherwise, the matrix $\overline{B}$ is nonsingular, and then we obtain $s.g._r(\overline{B}^{-1})$.

2. Apply Algorithm 3.3.1 to compute an $s.g._r(\overline{S})$ for the matrix $\overline{S} = \overline{J} - \overline{E}\,\overline{B}^{-1}\overline{C}$.

3. Apply the algorithm recursively to the Cauchy-like input matrix $\overline{S}$, replacing $\overline{A}$. Output $\rho = \text{rank}(\overline{A}) = k + \text{rank}(\overline{S})$.

4. By using the definitions and the results of section 2.2, compute an $s.g._{2r+4}(\overline{A}_\rho^{-1})$ (see further comments below).

5. Apply Fact 4.1.1, to compute and output $s.g._{r+2}(\overline{A}_\rho^{-1})$.

Let us specify stage 4. Consider the $\rho \times \rho$ leading principal submatrix, $\overline{A}_\rho = \begin{pmatrix} \overline{B} & \overline{G} \\ \overline{D} & \overline{R} \end{pmatrix}$, $\overline{G}$, $\overline{D}^T \in \mathbf{F}^{k \times (\rho-k)}$, $\overline{R} \in \mathbf{F}^{(\rho-k)\times(\rho-k)}$. Write $\hat{S} = \overline{R} - \overline{D}\,\overline{B}^{-1}\overline{G}$. Note that at the preceding stages we have computed $s.g._{r+2}(\overline{G})$, $s.g._{r+2}(\overline{D})$, $s.g._{r+2}(\overline{B}^{-1})$, $s.g._{r+2}(\overline{D}\overline{B}^{-1})$, $s.g._{r+2}(\overline{B}^{-1}\overline{G})$, and $s.g._{r+2}(\hat{S}^{-1})$ (cf. Theorem 4.2.1). Represent $\overline{A}_\rho{}^{-1}$ as follows:

$$\overline{A}_\rho{}^{-1} = \begin{pmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & \overline{S}^{-1} \end{pmatrix},$$

where $B_{1,2} = -\overline{B}^{-1}\overline{G}\,\overline{S}^{-1}$, $B_{2,1} = -\overline{S}^{-1}\overline{D}\,\overline{B}^{-1}$, $B_{1,1} = \overline{B}^{-1} - B_{1,2}\overline{D}\,\overline{B}^{-1}$ (cf. (3.1.4)). Due to Lemma 2.2.5 and Corollary 2.2.2, the matrices $B_{1,1}$, $B_{1,2}$, $B_{2,1}$, and $\overline{S}^{-1}$ have scaling rank at most $r+2$, and we may apply Algorithm 3.1.1 and the results of section 2.2 to compute the respective short scaling generators of these matrices. Let us specify the operators defining these generators. Write $\vec{q}^{\,(1)} = (q_i)_{i=0}^{k-1}$, $\vec{q}^{\,(2)} = (q_i)_{i=k}^{\rho-1}$, $\vec{t}^{\,(1)} = (t_i)_{i=0}^{k-1}$, $\vec{t}^{\,(2)} = (t_i)_{i=0}^{\rho-1}$,
$\vec{q}^{\,(0)} = \begin{pmatrix} \vec{q}^{\,(1)} \\ \vec{q}^{\,(2)} \end{pmatrix}$ and $\vec{t}^{\,(0)} = \begin{pmatrix} \vec{t}^{\,(1)} \\ \vec{t}^{\,(2)} \end{pmatrix}$.

Now obtain that

$$F_{[D(\vec{t}^{(0)}),\, D(\vec{q}^{(0)})]}(\overline{A}_\rho{}^{-1})$$

$$= \begin{pmatrix} D(\vec{t}^{(1)}) & O \\ O & D(\vec{t}^{(2)}) \end{pmatrix} \overline{A}_\rho{}^{-1} - \overline{A}_\rho{}^{-1} \begin{pmatrix} D(\vec{q}^{(1)}) & O \\ O & D(\vec{q}^{(2)}) \end{pmatrix}$$

$$= \begin{pmatrix} F_{[D(\vec{t}^{(1)}),\ D(\vec{q}^{(1)})]}(B_{1,1}) & F_{[D(\vec{t}^{(1)}),\ D(\vec{q}^{(2)})]}(B_{1,2}) \\ F_{[D(\vec{t}^{(2)}),\ D(\vec{q}^{(1)})]}(B_{2,1}) & F_{[D(\vec{t}^{(2)}),\ D(\vec{q}^{(2)})]}(\hat{S}^{-1}) \end{pmatrix},$$

which gives us an $s.g._{2r+4}(\overline{A}_\rho^{-1})$.

To solve a singular Cauchy-like linear system $A\vec{x} = \vec{b}$, first compute a vector $\vec{y}$ that satisfies $LAM\vec{y} = L\vec{b}$ and then recover the vector $\vec{x} = M\vec{y}$ that satisfies $A\vec{x} = \vec{b}$. Since L and M are nonsingular, rank (A) = rank (LAM). As a by-product, the algorithm computes rank $(\overline{A})$, which equals rank (A) with a probability at least $1 - 2\rho(\rho+1)/|\mathbf{S}|$, by Theorem 5.5.1. By using a standard technique (see e.g. [BP94], p.110), the algorithm can be immediately extended (at additional cost $O(rC_{M_v}(n))$) to the computation of a basis for the null space of A.

Finally, we observe that, by using $O(rC_{M_v}(n))$ ops, we may verify whether $A\vec{x} = \vec{b}$, that is, the overall cost bound for the algorithm covers the cost of the verification of its correctness; furthermore, similar property holds for the rank and null space computation by this approach.

APPENDIX A : Computations with Cauchy Matrices

The solution of *Trummer's problem* (that is, the problem of multiplication of a Cauchy matrix C by a vector) is the basis for the solution of several important problems of scientific and engineering. The straightforward algorithm solves Trummer's problem in $O(n^2)$ flops. The fast algorithm uses $O(n \log^2 n)$ flops but has poor numerical stability.

Our algorithms of the chapters 3 and 4 ultimately reduce Cauchy-like computations to multiplications of Cauchy matrices by vectors (Trummer's problem). For the computation of the solution $\vec{x} = C^{-1}(\vec{s}, \vec{t})\vec{v}$ to a nonsingular Cauchy linear system $C(\vec{s}, \vec{t})\vec{x} = \vec{v}$, the reduction is much simplified due to the following formula (cf. e.g. [Gast]):

$$C^{-1}(\vec{s}, \vec{t}) = -diag(\frac{\Gamma_{\vec{s}}(t_i)}{\Gamma'_{\vec{t}}(t_i)})_{i=0}^{n-1} C^T(\vec{s}, \vec{t}) diag(\frac{\Gamma_{\vec{t}}(s_i)}{\Gamma'_{\vec{s}}(s_i)})_{i=0}^{n-1}, \tag{A.1}$$

where $\Gamma_{\vec{x}}(u)$ denote the polynomial

$$\prod_{i=0}^{n-1}(u - x_i) = u^n + \sum_{i=0}^{n-1} r_i u^i,$$

for vector $\vec{x} = [\, x_o, \cdots, x_{n-1}]^T$, and where " $'$ " denotes the derivative. On the other hand, we have the following matrix equation from [GO92] :

$$C(\, \vec{s}, \vec{t}\,) = diag(1/\Gamma_{\vec{t}}\,(s_i))_{i=0}^{n-1} V(\, \vec{s}\,) V^{-1}(\, \vec{t}\,) diag(\Gamma'_{\vec{t}}\,(t_k))_{k=0}^{n-1}, \tag{A.2}$$

where $V(\vec{x})$ is a Vandermonde matrix.

The latter equation reduces the computation of the product $C(\vec{s}, \vec{t}\,)\vec{v}$, for any vector $\vec{v}$, to the computation of the product of the Vandermonde matrix $V(\vec{s})$ by a vector and to the solution of a Vandermonde linear system of n equations. These two operations are equivalent to multipoint polynomial evaluation and to polynomial interpolation, respectively. (Note that the computation of the values of the polynomial $\Gamma_{\vec{t}}\,(s_i)$, for $i = 0, \cdots, n-1$ and for a given vector of the coefficients of this polynomial, is also the problem of multipoint polynomial evaluation.) Since the known fast algorithms (cf. [BP94]) perform the latter operations, as well as the computation of the coefficients of $\Gamma_{\vec{t}}\,(x)$ for a given vector $\vec{t}$, in $O((n \ \log^2 n) \ \log \ \log \ n)$ ops over any field of constants, we arrive at Lemma 2.2.2.

Furthermore, we immediately obtain from (*A*.2) that

$$C^{-1}(\vec{s},\vec{t}) = diag(1/\Gamma'_{\vec{t}}(t_i))_{i=0}^{n-1} V(\vec{t}) V^{-1}(\vec{s}) diag(\Gamma_{\vec{t}}(s_i))_{i=0}^{n-1}.$$

Based on this formula, we may deduce the following result.

Fact A.1 [Gast]. *A nonsingular Cauchy linear system of n equations can be solved over any field of constants by using* $O((n\log^2 n)\log\log\ n)$ *ops.*

Alternatively, we may immediately deduce Fact A.1 from the formula (*A*.1), which has an advantage of reducing the solution of a Cauchy linear system to the multiplication of a Cauchy matrix by a vector (Trummer's problem). (Recall that our algorithms of chapters 3 and 4 show a similar reduction of Cauchy-like computations.)

Bibliography

[A63] S. J. Aarseth, Dynamical Evaluation of Clusters of Galaxies-I, *Mon. Not. R. Astron Soc.*, 126, 223–255, 1963.

[A85] A. W. Appel, An Efficient Program for Many-Body Simulation, *SIAM J. on Sci. Stat. Computing*, 6, 85–103, 1985.

[A86] C.R. Anderson, A Method of Local Corrections for Computing the Velocity Field due to a Distribution of Vortex Blobs, *J. Comput. Phys.*, 62, 111–123, 1986.

[Ab] J. Abot, Recovery of Algebraic Numbers from Their p-adic Approximations, *Proc. ACM-SIGSAM Intern. Symp. on Symb. and Algebr. Comp.*, 112-120, ACM Press, New York, 1989.

[AGR88] J. Ambrosiano, L. Greengard, V. Rokhlin, The Fast Multipole Method for Gridless Particle Simulation, *Computer Physics Communications*, 48, 115–125, 1988.

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

[B88] J. Barnes, Encounters of Disk/Halo Galaxies, *The Astrophisical J.*, 331, p. 699, 1988.

[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.

[BH86] J. Barnes, P. Hut, A Hierarchical $O(N \log N)$ Force-Calculation Algorithm, *Nature*, 324, 446–449, 1986.

[BM] A. Borodin and I. Munro, *The computational complexity of algebraic and numerical problems*, Amer. Elsevier, 1975.

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

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

[Bun] J. R. Bunch, Stability of Methods for Solving Toeplitz Systems of Equations, *SIAM J. Sci. Stat. Comput.*, **6, 2**, 349-364, 1985.

[C73] A. J. Chorin, Numerical Study of Slightly Viscous Flaw, *Fluid Mechanics*, 57, 785–796, 1973.

[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.

[CGR88] J. Carier, L. Greengard, V. Rokhlin, A Fast Adaptive Multipole Algorithm for Particle Simulation, *SIAM J. Sci. Stat. Comput.*, 9, 4, 669–686, 1988.

[D74] W.F. Donoghue, *Monotone Matrix Functions and Analytic Continuation*, Springer, Berlin, 1974.

[D83] J. M. Dawson, Particle Simulation of Plasmas, *Rev. Mod. Physics*, 55, 403–447, 1983.

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

[FHR93] T. Fink, G. Heinig, K. Rost, An Inversion Formula and Fast Algorithms for Cauchy-Vandermonde Matrices, *Linear Algebra Appl.*, 183, 179–191, 1993.

[G84] J. von zur Gathen, Parallel Algorithms for Algebraic Problems, *SIAM J. Comput.*, **13, 4**, 802-824, 1984.

[G88] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press, Cambridge, Mass, 1988.

[Gast] N. Gastinel, Inversion d'une Matrice Generalisant la Matrice de Hilbert, *Chiffres* , **3**, 149-152, 1960.

[Ger] A. Gerasoulis, A Fast Algorithm for the Multiplication of Generalized Hilbert Matrices with Vectors, *Math. Comp.*, **50, 181**, 179-188, 1987.

[GKO95] I. Gohberg, T. Kailath, V. Olshevsky, Fast Gausian 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 Univ. Press, Baltimore, Maryland, 1989 (second edition), 1996 (third edition).

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

[GR87] L. Greengard, V. Rokhlin, A Fast Algorithm for Particle Simulation, *J. Comput. Physics*, 73, 2, 325–348, 1987.

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

[GO94] 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.

[GO94a] I. Gohberg, V. Olshevsky, Complexity of Multiplication with Vectors for Structured Matrices, *Linear Algebra Appl.*, **202**, 163-192, 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.

[HE81] R. W. Hockney, J. W. Eastwood, *Computer Simulation Using Particles*, McGraw-Hill, New York, 1981.

[K87] T. Kailath, Signal Processing Applications of Some Moment Problems, *Proc. AMS Symp. in Applied Math.*, **37**, 71-100, 1987.

[K94] E. Kaltofen, Asymptotically Fast Solution of Toeplitz-like Singular Linear Systems, *Proc. ACM-SIGSAM Intern. Symp. on Symbolic and Alg. Comp.*, 297-304, ACM Press, New York, 1994.

[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.

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

[KR] R. Karp, V. Ramachandran, A Survey of Parallel Algorithms for Shared Memory Machines, *Handbook for Theoretical Computer Science* (J. van Leeuwen Editor), 869-941, North Holland, Amstrerdam, 1990.

[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.

[M74] M. Morf, Fast Algorithms for Multivariable Systems, Ph.D. Thesis, Stanford University, Stanford, CA, 1974.

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

[ODR89] S.T. O'Donnell, V. Rokhlin, A Fast Algorithm for Numerical Evaluation of Conformal Mappings, *SIAM J. Sci. Stat. Comput.*, **10, 3,** 475–487, 1989.

[OP98] V. Olshevsky, V. Y. Pan, A Superfast State-Space Algorithm for Tangential Nevanlinna-Pick Interpolation Problem, preprint, 1998.

[OS88] A.M. Odlyzko, A. Schönhage, Fast Algorithms for Multiple Evaluations of the Riemann Zeta Function, *Trans. Am. Math. Soc.*, 309, 2, 797–809, 1988.

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

[P92b] V.Y. Pan, Parametrization of Newton's Iteration for Computations with Structured Matrices and Applications, *Computers & Mathematics (with Applications)*, **24, 3**, 61-75, 1992.

[P93] V.Y. Pan, Decreasing the Displacement Rank of a Matrix, *SIAM J. Matrix Anal. Appl.*, **14, 1**, 118-121, 1993.

[P93a] V.Y. Pan, On Randomized Parallel Computations with General and Toeplitz-like Matrices, manuscript, 1993.

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

[PACPS98] V. Y. Pan, M. Abu Tabanjeh, Z. Chen, S. Providence, A. Sadikou, Transformations of Cauchy Matrices for Trummer's Problem and a Cauchy-like Linear Solver, *Proc. of 5th Annual International Symposium on Solving Irregularly Structured Problems in Parallel (Irregular98)*, (A. Ferreira, J. Rolim, H. Simon, S.-H. Teng Editors), *Lecture Notes in Computer Science*, **1457**, 274-284, Springer, 1998.

[PACP99] V. Y. Pan, M. Abu Tabanjeh, Z. Chen, S. Providence, Superfast Computations with Singular Structrued Matrices over Abstract Fields, *Proc. Second Workshop on Computer Algebra in Scientific Computing (CASC-99)*, (V. G. Ganzha, E. W. Mayr, E. V. Vorontsov, Editors), 323-338, Springer, 1999.

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

[PR89] V. Y. Pan, J. Reif, Fast and Efficient Parallel Solution of Dense Linear Systems, *Computers & Mathematics (with Applications)*, **17, 11**, 1481-1491, 1989.

[PR91] V. Y. Pan, J. Reif, The parallel Computation of the Minimum Cost Paths in Graphs by Stream Contraction, *Information Processing Letters*, **40**, 79-83, 1991.

[PR93] V. Y. Pan, J. Reif, Generalized Compact Multigrid, *Computers & Mathematics (with Applications)*, **25, 9**, 3-5, 1993.

[PZ99] V. Y. Pan, A. Zheng, Fast Cauchy-like and Singular Toeplitz-like Matrix Computations, MSRI Preprint No. 1999-013, *Math. Science Research Institute, Berkeley, California*, 1999.

[PZ96] V. Y. Pan, A. Zheng, Fast Algorithms for Cauchy-like Matrix Computations and an Extension to Singular Toeplitz-like Computations, Preprint, 1996.

[PZ,a] V. Y. Pan, Ailong Zheng, Fast Algorithm for Cauchy-like Matrix Computations and an Extension to Singular Toeplitz-like Computations, Preprint, 1997.

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

[R85] V. Rokhlin, Rapid Solution of Integral Equations of Classical Potential Theory, *J. Comput. Physics*, 60, 187–207, 1985.

[R88] V. Rokhlin, A Fast Algorithm for the Discrete Laplace Transformation, *J. of Complexity*, 4, 12–32, 1988.

[Re90] L. Reichel, A Matrix Problem with Application to Rapid Solution of Integral Equations, *SIAM J. on Scientific and Stat. Computing*, **11**, 263-280, 1990.

[Rok85] F. Rokhlin, Rapid Solution of Integral Equations of Classical Potential Theory, *J. Comput. Physics*, **60**, 187-207, 1985.

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

[St69] V. Strassen, Gaussian Elimination is Not Optimal, *Numer. Math.*, **13**, 354-356, 1969.

[Su] M. Sudan, Decoding of Reed-Solomon Codes Beyond the Error-Correction Bound, *J. of Complexity*, **13**, 180-193, 1997.

[SW] M. A. Shokrollahi, H. Wasserman. Decoding Algebraic-Geometric Codes Beyond the Error-Correction Bound, *Proc. 30th Annual Symp. on Theory of Computing*, 241-248, ACM Press, New York, 1998.

[T86] M. Trummer, An Efficient Implementation of a Conformal Mapping Method Using the Szegö Kernel, *SIAM J. on Numerical Analysis*, **23**, 853-872, 1986.

[Tur] W. Turin, Performance Analysis of Digital Transmission Systems, Computer Society Press, 1990.

[vDR89] L. van Dommelen, E. A. Rundensteiner, Fast Adaptive Summation of Point Forces in the Two-Dimentional Poisson Equation, *J. Comp. Physics*, 83, 126–147, 1989.

[Wang] P. Wang, A p-adic Algorithm for Univariate Partial Fractions, *Proc. 1981 ACM Symp. on Symbolic and Algebraic Comp.*, 212-217, ACM Press, New York, 1981.

[WGD] P. S. Wang, M. J. T. Guy, J. H. Davenport, p-adic Reconstruction of Rational Numbers, *SIGSAM Bulletin*, **16**, 2-3, ACM Press, New York, 1982.

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