2008

TR-2008003: Unified Nearly Optimal Algorithms for Structured Integer Matrices and Polynomials

Victor Y. Pan

Brian Murphy

Rhys E. Rosholt

Unified Nearly Optimal Algorithms for Structured Integer Matrices and Polynomials *†

Victor Y. Pan[1], Brian Murphy, and Rhys E. Rosholt
Department of Mathematics and Computer Science
Lehman College of CUNY, Bronx, NY 10468, USA
victor.pan@lehman.cuny.edu
brian.murphy@lehman.cuny.edu
rhys.rosholt@lehman.cuny.edu
[1] http://comet.lehman.cuny.edu/vpan/

May 1, 2008

Abstract

We seek the solution of banded, Toeplitz, Hankel, Vandermonde, Cauchy and other structured linear systems of equations with integer coefficients. By combining Hensel's symbolic lifting with either divide-and-conquer algorithms or numerical iterative refinement, we unify the solution for all these structures. We yield the solution in nearly optimal randomized Boolean time, which covers both solution and its correctness verification. Our algorithms and nearly optimal time bounds are extended to the computation of the determinant of a structured integer matrix, its rank and a basis for its null space as well as to some fundamental computations with univariate polynomials that have integer coefficients. Furthermore, we allow to perform lifting modulo a properly bounded power of two to implement our algorithms in binary within a fixed computer precision.

2000 Math. Subject Classification: 68W30, 68W20, 65F05, 68Q25

Key Words: Structured matrices, Hensel's lifting, the MBA algorithm, Iterative refinement

*Some results of this paper have been presented at the Annual International Conference on Application of Computer Algebra, Volos, Greece, June 2002; ACM International Symposium on Sympolic and Algebraic Computation, Lille, France, July 2002; and the 5th Annual Conference on Computer Algebra in Scientific Computing, Yalta, Crimea, Ukraine, September 2002

†Supported by NSF Grant CCR 9732206 and PSC CUNY Awards 67297–0036, 68291–0037, 69330–0038, and 69350–0038

1 Introduction

Linear systems of equations with displacement structure of Toeplitz, Hankel, Vandermonde and Cauchy types are omnipresent in scientific and engineering computations and signal and image processing. Due to the structure they can be solved fast, in quadratic rather than cubic arithmetic time [L47], [D59], [T64] or even superfast, in nearly linear arithmetic time [BGY80], [M80], [BA80].

Quite typically, however, application of the superfast algorithms leads to the problems of numerical stability [B85]. Moreover, structured linear systems of some important classes are ill conditioned [GI88], [T94]. This suggests devising fast and superfast symbolic algorithms, whose complexity is usually analyzed under the Boolean (bit-operation) and word operation models [GG03].

Our main goal in this paper is to present such algorithms in a unified way for the general class of linear systems whose coefficient matrices as well as their pre-computed inverses can be multiplied by vectors fast. Besides all listed classes of linear systems with displacement structure, this includes highly important banded linear systems and more generally the rank structured ones, also known as semiseparable, quasiseparable and the linear systems having a low Hankel rank [VVGM05]. In the recent years computations with rank structured matrices took about as much attention of the researchers as computations with matrices having displacement structure.

Our algorithms support the complexity bounds that cover the cost of both randomized solution and its correction verification and are nearly optimal (up to logarithmic factor) under both Boolean and word operation models. Versus the information lower bound of $n^2 \log n$, our algorithms take the order of $n^2 \log^2 n$ bit-operations for a structured linear system of n equations with n unknowns where all coefficients have absolute values in $n^{O(1)}$. The same cost bound covers the computation of the rank and a basis for the null space, and we increase the bound just by logarithmic factor to cover correctness verification and the computation of determinants.

Our algorithms can be called superfast because their nearly quadratic Boolean cost bounds supersede the orders of n^4 and n^3 bit-operations required for the solution of the same task by Gaussian elimination and by the fast algorithms such as Levinson–Durbin's and Trench's, respectively. We know of no other superfast and nearly optimal algorithms that unify the solution of linear systems for all cited classes.

Our work involves many vexing technicalities. The companion paper [PWa] counters degeneration by means of randomization and, like our present study, fills the void in the literature even for Toeplitz matrices.

The algorithms and nearly optimal complexity estimates can be extended to numerous related computational tasks. In Section 6.2 we specify such extensions to Berlekamp–Massey's reconstruction of a linear recurrence from its values and to computing the gcd, lcm, and Padé approximation for univariate polynomials.

Technically we first combine Hensel's lifting in [MC79], [D82] with the MBA divide-and-conquer algorithm, proposed by Morf [M74], [M80] and Bitmead and Anderson [BA80] for numerical solution of Toeplitz-like linear systems. In

Section 7 we support the same nearly optimal overall asymptotic cost bound (and also unified for all structured matrices) by combining Hensel's symbolic lifting with numerical iterative refinement algorithm [GL96, Section 3.5.3]. The combination of such techniques towards a common goal brings our work into the increasingly popular field of *symbolic-numerical computations* [TCS04], [SNC07], [SNC07a], [TCS08]. From that point it is also interesting that symbolic lifting and numerical refinement mimic one another. Both assume a precomputed approximate inverse. In each recursive loop both multiply it by a vector and refine the resulting approximate solution by using accurate residual vector. Each loop adds a fixed number of new correct bits to every component of the solution, either from the left (in lifting) or from the right (in refinement).

Unlike customary lifting modulo a random prime, we allow computations modulo powers of two, which enables more effective implementation, with the CPU time decreasing by twice. In this case we avoid using the MBA algorithm because it tends to degenerate and to fail modulo powers of two.

We extend our generalization to Newton's lifting. Its sequential complexity is higher by logarithmic factor, but it uses much fewer lifting steps, thus enabling parallel acceleration.

We organize our presentation as follows. In the next section and short Appendix A we state some definitions and basic results. We initialize lifting modulo a random prime in Section 3 and modulo a power of a fixed prime in Section 7. In Section 4 we describe Hensel's lifting for linear systems of equations in the rings of integers modulo an integer, possibly a power of two. In Section 5 and Appendix B we cover the reconstruction of the rational solution from the solution modulo a large integer. In Section 6 and Appendix C we estimate the overall Boolean and word complexity of our solution and of its extenions to Berlekamp–Massey's problem and the cited fundamental polynomial computations. In Section 8 we cover Newton's lifting. In Section 9 we review the computation of determinants and its links to lifting. Section 10 is left for a brief discussion. In Section 11 we recall various related works. The presented algorithms have been implemented by the third and mostly the second authors. Otherwise the paper is due to the first author.

2 Definitions and basic facts

We write log for $\log_2$ unless is specified otherwise, $\mathbb{Z}$ for the ring of integers, $\mathbb{Z}_q$ for the ring of integers modulo an integer q, and $\mathbb{Q}$ for the field of rational numbers. "Ops" stand for "arithmetic operations". "$\ll$" means"much less". $\tilde{O}(f(n))$ denotes $O(f(n)(\log\log n)^c)$ for a constant c. We write $a = z \mod q$, for three integers $q > 1$, a, and z, either to denote a unique integer a such that q divides $z - a$ and $0 \le a < q$ or, wherever due to the context this causes no confusion, just to show that q divides $a - z$.

Fact 2.1. *Let $2|a| < m$ for two integers a and $m > 1$. Write $a = a \mod m$ if $2|a \mod m| < m$, write $a = a \mod m - m$ otherwise.*

2.1 General matrices

Definition 2.1. *$M = (m_{i,j})_{i,j=1}^{k,l} \in \mathbb{R}^{k\times l}$ is a $k \times l$ matrix with entries $m_{i,j}$ in a ring $\mathbb{R}$. $\mathbf{v} = (v_i)_{i=1}^{k} \in \mathbb{R}^{k\times 1}$ is a column vector. I is the identity matrix of a proper size. I_l is the $l \times l$ identity matrix. (K, L) is a 1×2 block matrix with the blocks K and L. $D(\mathbf{v}) = \mathrm{diag}(\mathbf{v}) = \mathrm{diag}(v_i)_i$ denotes the diagonal matrix with the diagonal entries $d_{ii} = v_i$ given by the coordinates of the vector $\mathbf{v} = (v_i)_i$. M^T is the transpose of M. $M^{(h)}$ is the $h \times h$ leading principal (that is northwestern) submatrix of M. A matrix M of rank ρ has* generic rank profile *if its submatrices $M^{(k)}$ are nonsingular for $k = 1, \dots, \rho$, that is up to the rank size $\rho \times \rho$. M is* strongly nonsingular *if it is nonsingular and has generic rank profile. A block of a matrix is its submatrix in the intersection of a set of its contiguous rows and a set of its contiguous columns.*

Definition 2.2. *The matrix $M^T M$ for a nonsingular matrix M is symmetric and positive definite. In $\mathbb{Q}^{n\times n}$ such a matrix is strongly nonsingular.*

Definition 2.3. *$\det M$ and $\mathrm{adj}\, M$ are the determinant and the adjoint of a matrix M, respectively.* ($\mathrm{adj}\, M = M^{-1} \det M$ *if M is nonsingular.)*

Definition 2.4. *$|M| = ||M||_\infty = \max_i \sum_j |m_{i,j}|$ is the row norm of a matrix $M = (m_{i,j})_{i,j}$; $\alpha(M) = \max_{i,j} |m_{i,j}|$; $|\mathbf{v}| = \beta(\mathbf{v}) = \max_i |v_i|$ is the maximum norm of a vector $\mathbf{v} = (v_i)_i$.*

Definition 2.5. *$m_S \le 2n^2 - n$ is the minimum number of arithmetic operations sufficient to multiply an $n \times n$ matrix S by a vector.*

Clearly, we can multiply a pair of $n \times n$ matrices by using $2n^3 - n^2$ arithmetic operations. In this paper we usually ignore theoretical asymptotic acceleration and the more minor practical speed up, on which we refer the reader to [P84], [CW90], [K04], [DGP04], and the bibliography therein.

Hadamard's estimate below is known to be sharp in the worst case but is an over-estimate on the average according to [ABM99].

Fact 2.2. *$|\det M| \le \prod_j (\Sigma_i m_{i,j}^2)^{1/2} \le (\alpha(M)\sqrt{n})^n$, $|\det M| \le |M|^n$, $|\mathrm{adj}\, M| \le n\alpha(\mathrm{adj}\, M)$, and so (since the entries of the matrix $\mathrm{adj}\, M$ are the determinants of $(n-1)\times(n-1)$ matrices) we have that $|\mathrm{adj}\, M| \le (\alpha(M)\sqrt{n-1})^{n-1} n$, $|\mathrm{adj}\, M| \le n|M|^{n-1}$ for an $n \times n$ matrix $M = (m_{i,j})_{i,j}$.*

Definition 2.6. *$d_k = d_k(M)$ is the kth determinantal divisor of a matrix $M \in \mathbb{Z}^{n\times n}$ for $k = 1, \dots, n$, that is the greatest common divisor (gcd) of all its $k \times k$ minors (subdeterminants). $s_0 = d_0 = 1$, $s_k = s_k(M) = d_k/d_{k-1}$ are the kth* Smith invariant factors *of M for $k = 1, \dots, n$.*

It is easily deduced (see [N72]) that $s_1, \dots, s_n \in \mathbb{Z}$ and $|\det M| = s_1 \cdots s_n$. Therefore

$$s_n \le |\det M| \le |M|^n. \tag{2.1}$$

Hereafter $\mathbf{b} \ne \mathbf{0}$, $n > 2$, $|M| > 2$, and so $\log n > 1$, $\log |M| > 1$.

Definition 2.7. *For two integers $q > 0$ and $s > 1$, a matrix M in $\mathbb{Z}_{qs}^{n\times n}$ is* factor-q nonsingular *modulo qs if there exists a matrix Q in $\mathbb{Z}_{qs}^{n\times n}$ such that*

$$MQ \bmod (qs) = qI. \tag{2.2}$$

For $q = 1$ equation (2.2) means that $Q = M^{-1} \mod s$, which implies that s and $\det M$ are coprime and which enables us to define Hensel's lifting in Section 4. For $q > 1$ Definition 2.7 specifies the class of matrices Q that support Hensel's lifting even where $\gcd(s, \det M) = b > 1$.

2.2 Polynomial and integer multiplication

Let $m(n)$ field operations be required to multiply two polynomials of degree $n-1$ or less. We have $m(n) \geq 2n-1$ (this is an information lower bound, which follows because there are $2n-1$ output values), $m(n) = O(n \log n)$ over the fields or rings that support FFT, and

$$m(n) \leq c_{class} n^2, \quad m(n) \leq c_k n^{\log 3}, \quad m(n) \leq (c_{ck} n \log n) \log\log n \tag{2.3}$$

over any field, ring with unity, or algebra. Here $\log 3 = 1.58496\ldots$; c_{class}, c_k, and c_{ck} are three constants, $0 < c_{class} < c_k < c_{ck}$, and the above bounds are supported by the classical, Karatsuba's, and Cantor and Kaltofen's algorithms; the practical choice among them depends on the degree n (see [B03] and [GG03]).

An integer modulo q can be represented with the d-bit precision for $d = \lceil \log q \rceil$. An arithmetic operation modulo q, including division by an integer comprime with q, can be performed by using $O(\mu(d))$ bit operations. Here $\mu(d)$ denotes the bit–operation complexity of multiplication of two integers modulo q, $\mu(d) \geq 2d-2$ (an information lower bound),

$$\mu(d) \leq C_{class} d^2, \quad \mu(d) \leq C_k d^{\log 3}, \quad \mu(d) \leq (C_{ss} d \log d) \log\log d, \tag{2.4}$$

C_{class}, C_k, and C_{ss} are three constants, $0 < C_{class} < C_k < C_{ss}$, and the above bounds are supported by the classical algorithm and those of Karutsuba 1963 and Schönhage and Strassen 1971 (see [K98], [B03], [GG03]).

2.3 Padé approximation and related computational tasks

Definition 2.8. Padé approximation. *For two nonnegative integers m and n and a polynomial $t(x) = \sum_{i=0}^{m+n} t_i x^i$, an (m, n) Padé approximation is a pair of coprime polynomials $r(x) = \sum_{i=0}^{m} r_i x^i$ and $v(x) = \sum_{i=0}^{n} v_i x^i$ that satisfy the equation $t(x)v(x) \mod x^{m+n+1} = r(x)$. (We can informally rewrite the latter equation as $t(x) = \frac{r(x)}{v(x)} \mod x^{m+n+1}$. Surely the pair $(r(x), v(x))$ is not unique, but the ratio $r(x)/v(x)$ is known to be unique.)*

Definition 2.9. Berlekamp–Massey Problem. *Given a positive integer s and $2s$ numbers $a_0, a_1, \ldots, a_{2s-1}$, compute the minimum integer $n \leq s$ and n numbers $c_0, c_1, \ldots, c_{n-1}$ such that $a_i = c_{n-1}a_{i-1} + \cdots + c_0 a_{i-n}$ for $i = n, n+1, \ldots, 2s-1$.*

Fact 2.3. *Berlekamp–Massey Problem has a unique solution, given by the degree n and the coefficients $c_0, c_1, \dots, c_{n-1}$ of the minimum span polynomial $c(x) = x^n - \sum_{i=0}^{n-1} c_i x^i$ such that for some polynomial $r(x)$ the pair of polynomials $(r(x), c(x))$ is an $(s-1, s-1)$ Padé approximation to the polynomial $a(x) = \sum_{i=0}^{2s-1} a_i x^i$.*

Proof. See [BGY80]. □

Definition 2.10. *The greatest common divisor* $\gcd(\mathrm{u}, \mathrm{w})$ *of two polynomials $u(x) = \sum_{i=0}^{m} u_i x^i$ and $w(x) = \sum_{i=0}^{n} w_i x^i$ is their common divisor of the largest degree, whereas their least common multiple* $\mathrm{lcm}(\mathrm{u}, \mathrm{w}) = u(x)w(x)/\gcd(\mathrm{u}, \mathrm{w})$ *is their common multiple of the smallest degree. (Monic gcd and monic lcm are unique.)*

We need the following simple fact.

Fact 2.4. *Suppose $t(x) = \sum_{i=0}^{m+n} t_i x^i$, $t(x)w(x) \mod x^{m+n+1} = u(x)$, the pair of polynomials $(r(x), v(x))$ is an (m, n) Padé approximation to the polynomial $t(x)$, and $d(x)$ is a polynomial such that $u(x) = d(x)r(x)$. Then $w(x) = d(x)t(x)$ and $d(x)$ is a* $\gcd(\mathrm{u}, \mathrm{w})$.

2.4 Matrices with displacement structure: general properties

In this subsection we define matrices with displacement structure and recall their basic properties.

Definition 2.11. *$\Delta_{A,B}(M) = M - AMB = GH^T$ (resp. $\nabla_{A,B}(M) = AM - MB = GH^T$) is the Sylvester (resp. Stein)* displacement *of an $n \times n$ matrix M where $n \times n$ matrices A and B are* operator matrices *and a pair of $n \times l$ matrices G and H form a* displacement generator *of length l for the matrix M. The rank of the displacement minimizes the length l and is called the* displacement rank *of the matrix M.*

The simple basic results below are from [P01, Theorems 1.3.1, 1.5.1–1.5.6].

Theorem 2.1. *If the matrix A (resp. B) is nonsingular, then $\Delta_{A,B}(M) = A^{-1}\nabla_{A^{-1},B}(M)$ (resp. $\Delta_{A,B}(M) = -\nabla_{A,B^{-1}}(M)B^{-1}$).*

Theorem 2.2. *For matrices A, B, M, and N of compatible sizes, displacement operators $L = \Delta_{A,B}$ and $L = \nabla_{A,B}$, and a scalar a, we have $L(M + aN) = L(M) + aL(N)$, $\Delta_{A,B}(M^T) = (\Delta_{B^T,A^T}(M))^T$, $\nabla_{A,B}(M^T) = -(\nabla_{B^T,A^T}(M))^T$. Furthermore $\nabla_{B,A}(M^{-1}) = -M^{-1}\nabla_{A,B}(M)M^{-1}$ if M is a nonsingular matrix, $\Delta_{B,A}(M^{-1}) = BM^{-1}\Delta_{A,B}(M)B^{-1}M^{-1}$ if the matrices B and M are nonsingular, and $\Delta_{B,A}(M^{-1}) = M^{-1}A^{-1}\Delta_{A,B}(M)M^{-1}A$ if the matrices A and M are nonsingular.*

Theorem 2.3. *For any 5-tuple $\{A, B, C, M, N\}$ of matrices of compatible sizes we have $\nabla_{A,C}(MN) = \nabla_{A,B}(M)N + M\nabla_{B,C}(N)$, $\Delta_{A,C}(MN) = \Delta_{A,B}(M)N + AM\nabla_{B,C}(N)$. Furthermore $\Delta_{A,C}(MN) = \Delta_{A,B}(M)N + AMB\Delta_{B^{-1},C}(N)$ if B is a nonsingular matrix and $\Delta_{A,C}(MN) = \Delta_{A,B}(M)N - AM\Delta_{B,C^{-1}}(N)C$ if C is a nonsingular matrix.*

Theorem 2.4. *Represent the matrices A, B, M, $\nabla_{A,B}(M)$, and $\Delta_{A,B}(M)$ as 2×2 block matrices with blocks $A_{i,j}, B_{i,j}, M_{i,j}, \nabla_{i,j}$, and $\Delta_{i,j}$, respectively, having compatible sizes (for $i, j \in \{0, 1\}$). Then*

$$\nabla_{A_{ii},B_{jj}}(M_{ij}) = \nabla_{ij} - R_{i,j},$$
$$\Delta_{A_{ii},B_{jj}}(M_{ij}) = \Delta_{ij} + S_{i,j},$$

where

$$R_{i,j} = M_{i,1-j}B_{1-j,j} - A_{i,1-i}M_{1-i,j},$$
$$S_{i,j} = A_{i,i}M_{i,1-j}B_{1-j,j} + A_{i,1-i}M_{1-i,j}B_{j,j} + A_{i,1-i}M_{1-i,1-j}B_{1-j,j},$$

for $i, j \in \{0, 1\}$.

Remark 2.1. *The expressions of Theorem 2.4 project the displacement generator of a matrix into those of its blocks with the increase of the length of the generator by at most* $\mathrm{rank}(R_{i,j})$ *or* $\mathrm{rank}(S_{i,j})$*, that is, in both cases at most* $\mathrm{rank}(A_{1-i,j}) + \mathrm{rank}(B_{1-j,j})$*. Hereafter (see Definitions 2.12–2.16) we only deal with diagonal and unit f-circulant operator matrices A and B whose blocks $A_{1-i,i}$ and $B_{1-j,j}$ have ranks zero or one (cf. Remark 2.2).*

In Section 3 for a nonsingular structured matrix M with $dr(M) = r$ we perform operations with short displacement generators to obtain a displacement generator of length r for its inverse. In the process of computing, the length of the generators can grow above the displacement rank, but then we compress the generators to the rank level based on the following results, valid in any field.

Theorem 2.5. *Given a pair of $n \times l$ matrices G and H, it is sufficient to perform $O(l^2 n)$ ops to compute a pair of $n \times r$ matrices $\tilde{G}$ and $\tilde{H}$ such that $\tilde{G}\tilde{H}^T = GH^T$ where $r = \mathrm{rank}(GH^T)$.*

Proof. See [P01, Theorem 4.6.4]). □

Corollary 2.1. *Given a displacement generator of length l for a displacement operator L and an $n \times n$ matrix M with $dr_L(M) = r$, it is sufficient to use $O(l^2 n)$ ops to compute a displacement generator of length r for the same pair of L and M.*

2.5 Most popular matrices with displacement structure

Toeplitz, Hankel, Vandermonde, and Cauchy matrices have displacement ranks one or two for appropriate operator matrices. Larger classes of matrices with

the structures of Toeplitz, Hankel, Vandermonde, and Cauchy types have small displacement ranks r for the same operator matrices. Equivalently the matrices of these classes can be defined (in memory efficient way) as bilinear expressions via the entries of their displacements generators $G = (\mathbf{g}_i)_{i=1}^r$ and $H = (\mathbf{h}_i)_{i=1}^r$ (cf. [GO94] and [P01, Chapter 4]). These are most used matrices with displacement structure (see examples of other important classes in [P01, Examples 4.4.8 and 4.4.9]).

Definition 2.12. *$T = (t_{i,j})_{i,j=1}^n$ is a Toeplitz matrix if $t_{i,j} = t_{i+1,j+1}$ for every pair of its entries $t_{i,j}$ and $t_{i+1,j+1}$. For a scalar f such matrix T is f-circulant if $t_{i,j} = f t_{k,l}$ wherever $k - l + n = i - j > 0$. In this case we write $T = Z_f(\mathbf{t}) = \sum_{h=1}^n t_h Z_f^{h-1}$ where $\mathbf{t} = (t_h)_{h=1}^n$ is the first column of the matrix, $t_h = t_{1,h}$, $h = 1, \dots, n$, and Z_f is the unit f-circulant matrix with the first column $(0, 1, 0, \dots, 0)^T$ and the first row $(0, \dots, 0, f)$. $Z_0(\mathbf{v})$ is the lower triangular Toeplitz matrix with the first column $\mathbf{v}$.*

Definition 2.13. *(Cf. [P01, Example 4.4.1].) T is a Toeplitz-like matrix with displacement rank $dr(T) = dr_{Z_e, Z_f^T}(T) \le r$ if for a pair of scalars e and f, $ef \ne 1$, there exist r pairs of vectors $\mathbf{g}_i$ and $\mathbf{h}_i$, $i = 1, \dots, r$, such that $T = \sum_{i=1}^r Z_e(\mathbf{g}_i) Z_f(\mathbf{h}_i)^T$.*

The latter matrix equation is equivalent to the matrix equation $T - Z_e T Z_f^T = \sum_{i=1}^r \mathbf{g}_i \mathbf{h}_i^T$ [KKM79]. One can easily verify that $dr_{Z_e, Z_f^T}(T) \le 2$ for a Toeplitz matrix T and any pair of scalars e and f and that furthermore

$$|dr_{Z_e, Z_f^T}(T) - dr_{Z_e, Z_g^T}(T)| \le 1 \tag{2.5}$$

for any quadruple (T, e, f, g).)

Definition 2.14. *$J = (j_{g,h})_{g,h=0}^{n-1,n-1}$ is the reflection (or unit Hankel) matrix if $j_{g,n-1-g} = 1$ for $g = 0, \dots, n-1$, $j_{g,h} = 0$ for $h + g \ne n - 1$. ($J\,(v_i)_{i=0}^{n-1} = (v_{n-i-1})_{i=0}^{n-1}$, $J^2 = I$.) $H = (h_{i,j})_{i,j}$ is a Hankel matrix if $h_{i,j} = h_{i-1,j+1}$ for every pair of its entries $h_{i,j}$ and $h_{i-1,j+1}$ or equivalently if $H = TJ$ for a Toeplitz matrix T. H is a Hankel-like matrix if $H = TJ$ for a Toeplitz-like matrix $T = \sum_{i=1}^r Z_e(\mathbf{g}_i) Z_f(\mathbf{h}_i)^T$, so that $H = \sum_{i=1}^r Z_e(\mathbf{g}_i) Z_f(\mathbf{h}_i)^T J$.*

Definition 2.15. *(Cf. [P01, Example 4.4.6b].) $V(\mathbf{t}) = (t_i^{j-1})_{i,j=1}^n$ is a Vandermonde matrix. A matrix V has Vandermonde-like structure and has displacement rank $dr(V) = dr_{D(\mathbf{t}), Z_f}(V) \le r$ if for a vector $\mathbf{t} = (t_j)_{j=1}^n$ and a scalar f such that $t_i^n f \ne 1$ for all i there exist $2r$ vectors $\mathbf{g}_i$ and $\mathbf{h}_i$, $i = 1, \dots, r$, such that $W = \sum_{i=1}^r \operatorname{diag}(\frac{1}{1 - t_i^n})_{i=1}^n \operatorname{diag}(\mathbf{g}_i) V(\mathbf{t}) J Z_f(J \mathbf{h}_i)$ or equivalently $W - \operatorname{diag}(\mathbf{t}) W Z_f = \sum_{i=1}^r \mathbf{g}_i \mathbf{h}_i^T$ for $W = V$ or $V = W^T$. (We have $dr(V(\mathbf{t})) = 1$ for any scalar f.)*

Definition 2.16. *(Cf. [P01, Example 1.4.1].) $C(\mathbf{s}, \mathbf{t}) = (\frac{1}{s_i - t_j})_{i,j}$ is a Cauchy matrix. A matrix C has a Cauchy-like structure and has displacement rank $dr(C) = d_{D(\mathbf{s}), D(\mathbf{t})}(C) \le r$ if for a set of distinct scalars $\{s_i, t_j\}_{i,j}$ there exist $2r$ vectors $\mathbf{g}_i$ and $\mathbf{h}_i$, $i = 1, \dots, r$, such that $C = \sum_{i=1}^r \operatorname{diag}(\mathbf{g}_i) C(\mathbf{s}, \mathbf{t}) \operatorname{diag}(\mathbf{h}_i)$ or equivalently $\operatorname{diag}(\mathbf{s}) C - C \operatorname{diag}(\mathbf{t}) = \sum_{i=1}^r \mathbf{g}_i \mathbf{h}_i^T$. (We have $dr(C(\mathbf{s}, \mathbf{t})) = 1$.)*

Remark 2.2. *(Cf. Remark 2.1.) Observe that the rank of any off-diagonal block is zero for a diagonal matrix and is at most one for any matrix Z_f.*

Remark 2.3. *The classes of matrices with the structures of Toeplitz, Hankel and Vandermonde types above are invariant in the parameters e and f for $ef \neq 1$ that define the associated operator matrices Z_e and Z_f. We can also redefine these classes by applying Theorem 2.1 to replace the displacement operators Δ by ∇ or vice versa unless both operator matrices A and B are singular. (We replace the inequality $ef \neq 1$ by $e \neq f$ in the transition of the operators $\nabla \to \Delta$.)*

Toeplitz, Hankel, Vandermonde and Cauchy matrices can be multiplied by vectors in nearly linear arithmetic time.

Fact 2.5. *(See [P01, Chapters 2 and 3].) We have $m_S = O(m(n))$ if S is an $n \times n$ Toeplitz or Hankel matrix for m_S in Definition 2.5 and $m(n)$ in Section 2.2. $m_S = m_{S^T} = O(m(n) \log n)$ if S is an $n \times n$ Vandermonde or Cauchy matrix.*

The bilinear expressions in Definitions 2.13–2.16 reduce multiplication of a matrix with Toeplitz-like, Hankel-like, Vandermonde-like or Cauchy-like structures essentially to $2r$ multiplications of Toeplitz, Hankel, Vandermonde or Cauchy matrices by $2r$ vectors. Theorems 2.2 and 2.3 extend this property to transposes, inverses, sums, and products of such matrices as well as their blocks. In particular we specify the respective estimates for the aritmetic complexity in Corollary 2.2 below and, in the important case of Toeplitz matrices, in Appendix A.

Definition 2.17. *Suppose we have a nonsingular $n \times n$ matrix M preprocessed with a procedure P that outputs ν parameters $p_1, \dots, p_\nu$. Let $i_M(P)$ be the minimum number of ops required to solve a linear system $M\mathbf{x} = \mathbf{f}$ given a matrix M, a vector $\mathbf{f}$, and the parameters $p_1, \dots, p_\nu$. Write $i_M = \min_P\{i_M(P)\}$ (resp. $i_{M,l} = \min_{P(l)}\{i_M(P)\}$) where the minimum is over all preprocessings P (resp. over all preprocessings $P = P(l)$ that amount to solving at most l linear systems of equations with the matrices M, M^T and M^TM).*

Corollary 2.2. *We have $m_T = m_H = O(lm(n))$, $m_V = O(lm(n) \log n)$, $m_C = O(lm(n) \log n)$, $i_{T,2l} = i_{H,2l} = O(lm(n))$, $i_{V,2l} = O(lm(n) \log n)$, $i_{C,2l} = O(lm(n) \log n)$ where T, H, V, and C stand for $n \times n$ matrices given with their displacement generators of lengths at most l and having structures of Toeplitz, Hankel, Vandermonde, and Cauchy types, respectively.*

The following result reduces the solution of linear systems with the structures of Vandermonde and and Cauchy types to linear systems with Toeplitz-like structures, which enables faster solution in Section 3. The result represents the genral *method of displacement transformation*, due to [P89/90] (cf. [P01, Sections 1.7, 4.8, and 4.9]).

Corollary 2.3. *(Cf. [P89/90].) Given a positive integer r and a displacement generator of a length l for an $n \times n$ matrix M with the structure of Vandermonde (resp. Cauchy) type, one can generate matrices V_1 and/or V_2 with Vandermonde-like structure (defining them with their displacement generators of lengths at most r) and apply $O((l+r)m(n)\log n)$ ops to compute a displacement generator of a length at most $l+r$ (resp. $l+2r$) for a Toeplitz-like matrix V_1M or V_2M (resp. V_1MV_2).*

Proof. The corollary follows from Theorem 2.3 and Corollary 2.2. □

2.6 Banded, rank structured (quasiseparable), and sparse structured matrices

Definition 2.18. *(Cf. [GL96].) A matrix $B = (b_{i,j})_{i,j}$ has a lower bandwidth at most $l = l(B)$ and an upper bandwidth at most $u = u(B)$ and is called an (l, u) banded matrix if $b_{i,j} = 0$ wherever $i > j + l$ or $j > i + u$. Clearly, any $n \times n$ matrix is $(n-1) \times (n-1)$ banded, but an (l, u) banded $n \times n$ matrix B is said to be banded if $l + u \ll n$.*

Theorem 2.6. *a) For an $n \times n$ nonsingular (l, u) banded matrix B we have $m_B = O(nw)$ and $i_B(P) = O(nw)$ for some preprocessing P that involves $O(w^2n)$ ops where $w = l + u + 1$. b) $O(w^2n)$ ops are sufficient to compute $(\det B)^2$.*

Proof. Part a) is a special case of Theorem 2.8 below, but one can also deduce it directly, by applying Gaussian elimination or Block Cyclic reduction to the symmetric positive definite $(2l, 2u)$ banded matrix B^TB and recalling that $B^{-1} = (B^TB)^{-1}B^T$. The latter algorithms also support part b). □

(l, u) banded matrices are a special case of the following class of (l, u) rank structured matrices.

Definition 2.19. *(Cf. [EG01], [VVGM05], and the bibliography therein.) A matrix M has a lower rank l (resp. upper rank u) if this is the maximal rank of its submatrices lying below (resp. above) its diagonal. $M = (m_{ij})_{i,j=1}^n$ is an (l, u) rank structured matrix (also called quasiseparable of order (l, u)) if it has a lower rank l and an upper rank u. Such a matrix has a trilinear (l, u) generator for its off-diagonal entries given by the set of $l \times l$ matrices A_h and $u \times u$ matrices B_h, vectors $\mathbf{p}_h$ and $\mathbf{q}_h$ of dimension l, and vectors $\mathbf{s}_h$ and $\mathbf{t}_h$ of dimension u for $h = 2, 3, \dots, n$ such that $m_{ij} = \mathbf{p}_i^T A_{ij} \mathbf{q}_j$, $1 \le j < i \le n$, $m_{ij} = \mathbf{s}_i^T B_{ij} \mathbf{t}_j$, $1 \le i < j \le n$, where $A_{ij} = A_{i-1} \cdots A_{j+1}$ for $i - 1 > j$, $A_{i+1,i} = I_l$, $B_{ij} = B_{j-1} \cdots B_{i+1}$ for $j - 1 > i$, $B_{i,i+1} = I_u$. An (l, u) rank stuctured matrix is (l, u) semiseparable if its every strictly lower (resp. upper) submatrix can be embedded into a matrix of rank l (resp. u).*

Definition 2.20. *An (l, u) rank structured $n \times n$ matrix has a bilinear (l, u) generator given by the vectors $\mathbf{p}_i$, $\mathbf{q}_j$, $\mathbf{s}_i$, and $\mathbf{t}_j$ for all i and j, if it has a trilinear (l, u) generator of Definition 2.19 where $A_h = I_l$ and $B_h = I_u$ for*

all h. In this case the subdiagonal (resp. superdiagonal) part of the matrix $M = (m_{ij})_{i,j}$ (together with its diagonal) can be extended to a matrix of rank l (resp. u). If in addition, $m_{ii} = \mathbf{p}_i^T \mathbf{q}_i$ for all i, then the bilinear (l, u) generator is complete.

E.g., (l, u) banded matrices are (l, u) rank structured (but not (l, u) semiseparable). Generically they do not have bilinear (l, u) generators, but if they are nonsingular, then generically their inverses have complete bilinear (l, u) generators. The inverse of a symmetric positive definite (l, u) banded matrix has a complete bilinear (l, u) generator. A tridiagonal matrix is $(1, 1)$ rank structured, has a trilinear $(1, 1)$ generator where $A_k = B_k = 0$ for $k = 2, \dots, n-1$, but has no bilinear (l, u) generator. If such a matrix is nonsingular and irreducible, then its inverse has complete bilinear (l, u) generator.

The above definitions can be extended to block rank structured matrices [EG01], [EG02].

Theorem 2.7. *(Cf. [EG99], [EG01], [VVGM05].) The inverse of a nonsingular (l, u) rank structured matrix is an (l, u) rank structured matrix.*

Theorem 2.8. *(Cf. [EG99], [EG01], [VVGM05].)*

a) Given an $n \times n$ nonsingular (l, u) rank structured matrix M with $w = l + u + 1$, we can compute its determinant in $O(w^2 n)$ ops, and for some preprocessing P that involves $O(w^2 n)$ ops, we have $m_M = O(nw)$ and $i_M(P) = O(nw)$.

b) $i_{M,l+u} = O(nw)$ if in addition the matrix M is strongly nonsingular.

Next we cover an important class of sparse structured matrices.

Definition 2.21. *(Cf. [LRT79], [PR93], [P93].) Associate the n rows (as well as n columns) of an $n \times n$ matrix S with the set $V(S)$ of vertices of a graph $G(S) = (V(S), E(S))$. Associate the nonzero entries with the set $E(S)$ of edges. Write $|T|$ for the cardinality of a set T, so that $m_S \le 2|E(S)|$ and $|E(S)| < (l(B) + u(B) + 1)n$. A matrix S is sparse if $|E(S)| \ll |V(S)|^2 = n^2$.*

Definition 2.22. *A graph $G = (V, E)$ has an $s(n)$-separator family (with respect to two constants $a > 1$ and n_0) if either $|V| < n_0$ or the deletion of $s(|V|)$ vertices can partition this graph into two disconnected subgraphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ having $s(n)$-separator families with respect to the same constants a and n_0 and such that $|V_i| \le a|V|$ for $i = 1, 2$. (We have $s(n) \le l + u + 1$ and $|E(M)| < (l + u + 1)n$ for the graph $E(B)$ associated with an (l, u) banded matrix B, $s(n) = \sqrt{n}$ and $|E(M)| = O(n)$ for a planar graph, and $s(n) = O(n^h)$, $h = 1 - \frac{1}{d}$, and $|E(M)| = O(dn)$ for a d-dimensional grid graph.)*

Theorem 2.9. *(Cf. [GS92].) Let a symmetric and positive definite matrix M be associated with a graph $G(M) = (V(M), E(M))$ given with an $s(n)$-separator family. One can compute a permutation matrix W and Cholesky factorization of the matrix WMW^T in $O(s(n)^3)$ ops, and for such a preprocessing P we have $i_M(P) = O(|E(M)| + s(n))$ ops.*

Corollary 2.4. *Under the assumptions of Theorem 2.9, one can compute the integer $|\det M|$ in $O(|E(M)| + s(n)^3)$ ops.*

Corollary 2.5. *In the cases of symmetric and positive definite $n \times n$ matrices M associated with planar graphs or 2-dimensional (resp. 3-dimensional) grid graphs (so that $m_M \le 2|E(M)| = O(n)$), there exists preprocessing P by means of an appropriate factorization of the input matrix M that uses $O(n^{1.5})$ (resp. $O(n^2)$) operations and supports the bound $i_M(P) = O(n^2)$.*

2.7 Rational number reconstruction

Definition 2.23. $\mathrm{ord}_q(m)$*, the order of q in m, is the maximal integer l such that q^l divides m.*

Definition 2.24. *$\nu(y)$ is the numerator, and $\delta(y) \ge 1$ is the denominator in the ratio $y = \nu(y)/\delta(y)$ of two coprime integers $\nu(y)$ and $\delta(y)$.*

Modular rational roundoff is the recovery of a rational number x/y from three integers k, l, and $r = (x/y) \bmod l$ provided x and y are coprime unless $r = 0$, l and y are coprime, $|x| < k \le l$, and $0 < y \le l/k$. $\rho(\log l)$ is the bit–operation complexity of this recovery. Clearly, we can write $x = r, y = 1$ if $k > |r|$. The pair (x, y) is unique under the additional assumption that $2|x| < k$ [GG03].

Theorem 2.10. *(Cf. [WP03].) We have*

$$\rho(d) \le cd^2, \rho(d) \le C\mu(d) \log d \tag{2.6}$$

for $\mu(d)$ in (2.4) and two positive constants C and c, $c < C$.

Proof. To support the theorem, it is sufficient to apply the algorithms in any of the papers [PW02], [WP03], [PW04], or [M04]. □

2.8 Las Vegas versus Monte Carlo randomization

Unlike *deterministic algorithms*, which always produce correct output, *randomized algorithms* produce correct output with a probability of at least $1 - \epsilon$ for a fixed tolerance ϵ. The randomized complexity estimates are of the *Las Vegas* type if they cover the cost of the correctness verification, that is if at the estimated cost one either fails with a low probability or outputs the correct solution. The other randomized complexity estimates are of the *Monte Carlo* type. They show the cost bound under which the output can be erroneous, although with a bounded low probability.

2.9 Randomization versus degeneration

Given a nonsingular matrix in $\mathbb{Z}$, what is the probability that it stays such in $\mathbb{Z}_p$ for a random prime p in a fixed large range, e.g. in $(y/20, y]$ for a large integer y? Here is an estimate from [PMRW05], [PWa].

Theorem 2.11. *Suppose that ϵ is a positive number, the matrix $M \in \mathbb{Z}^{n\times n}$ is nonsingular, and a prime p is randomly sampled from the range $(y/20, y]$ under the uniform probability distribution in this range where $y = \frac{n\xi \ln |M|}{\epsilon} \geq 114$, $\xi = \frac{16\ln 114}{16\ln 5.7 - \ln 114} = 16\alpha/(1-\alpha) = 3.278885\ldots$, and $\alpha = \frac{\ln 114}{16\ln 5.7} = 0.17007650\ldots$. Then $P = \text{Probability}((\det M) \bmod p = 0) < \epsilon$.*

3 Computations in the fields with matrices having displacement structure

3.1 Inversion of strongly nonsingular structured matrices

Theorem 3.1. *Assume that a strongly nonsingular $n \times n$ matrix M in a field $\mathbb{F}$ has structure of Toeplitz, Hankel, Vandermonde or Cauchy type (cf. Definitions 2.13–2.16), has a displacement rank r, and is given with its displacement generator of a length l. Then a displacement generator of the minimum length r for the matrix M^{-1} as well as the scalar $\det M$ can be computed by using $O(l^2 n + m_M r \log n)$ field operations.*

Proof. The MBA divide-and-conquer algorithm by Morf 1974 and 1980 and Bitmead and Anderson 1980 [M74], [M80], and [BA80] was proposed for Toeplitz-like matrices. We adapt it to a more general class of matrices with displacement structutre (cf. [OP98], [P01, Chapter 5]). Recall the well-known block triangular factorizations

$$M = \begin{pmatrix} M_{00} & M_{01} \\ M_{10} & M_{11} \end{pmatrix} = \begin{pmatrix} I & 0 \\ M_{10}M_{00}^{-1} & I \end{pmatrix} \begin{pmatrix} M_{00} & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & M_{00}^{-1}M_{01} \\ 0 & I \end{pmatrix}, \quad (3.1)$$

$$M^{-1} = \begin{pmatrix} \tilde{M}_{00} & \tilde{M}_{01} \\ \tilde{M}_{10} & \tilde{M}_{11} \end{pmatrix} = \begin{pmatrix} I & -M_{00}^{-1}M_{01} \\ 0 & I \end{pmatrix} \begin{pmatrix} M_{00}^{-1} & 0 \\ 0 & S^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -M_{10}M_{00}^{-1} & I \end{pmatrix}. \quad (3.2)$$

Here

$$\tilde{M}_{10} = (-S^{-1})(M_{10}M_{00}^{-1}), \quad \tilde{M}_{01} = -(M_{00}^{-1}M_{01})S^{-1}, \quad (3.3)$$

$$\tilde{M}_{00} = M_{00}^{-1} - (M_{00}^{-1}M_{01})\tilde{M}_{10}, \quad \tilde{M}_{11} = S^{-1}, \quad (3.4)$$

the block matrix M_{00} and the $k \times k$ Schur complement

$$S = S(M, M_{00}) = S^{(k)}(M) = M_{11} - (M_{10}M_{00}^{-1})M_{01} \quad (3.5)$$

are strongly nonsingular if so is the matrix M, and the sizes $n_i \times n_i$ of the block matrices M_{ii}, $i = 0, 1$ are assumed to be balanced (say, $n_0 = \lceil n/2 \rceil$, $n_1 = n - n_0$). We obtain and recursively extend factorization (3.1) by applying the block Gauss–Jordan elimination to the matrix M and then recursively to the matrices M_{00} and S until the inversion problem is reduced to the case of one-by-one matrices. Actual computation goes back from the inverses of these one-by-one matrices to the matrix M^{-1}. In this recursive process we can also

recursively factorize the scalar $\det M = (\det M_{00}) \det S$ and output it as by-product.

To yield the claimed complexity bound, we maintatin and exploit the structure of the input matrix M. In particular, we recursively compress the displacement generator of the matrix M and of all computed auxiliary matrices to the level of their displacement ranks and perform all computations with these matrices by operating with their displacement generators (cf. Theorems 2.2–2.5 and Remark 2.1).

Let us examine the associated operator matrices. Assume the pairs of operator matrices (A,B) for an input matrix M and (A_{ii},B_{jj}) for its submatrices M_{ij} for $i,j \in \{0,1\}$. Combine equations (3.3) and (3.4) with Theorems 2.2–2.4 and obtain the pairs of operator matrices (B_{ii},A_{jj}) for the submatrices $\tilde{M}_{ij}$ of M^{-1} for $i,j \in \{0,1\}$ and hence obtain the pair of operator matrices (B,A) for the matrix M^{-1} such that $dr_{A,B}(M) = dr_{B,A}(M^{-1})$ (cf. Theorem 2.2). Likewise we arrive at consistent pairs of operator matrices for every matrix computed in the forward recursive process and for its inverse computed in the respective backward step.

Now we can deduce the complexity bound claimed in Theorem 3.1 in the case of Sylvester displacements by combining Theorems 2.2 and 2.3 and our next result, in which we use Sylvester displacement $\nabla_{A,B}(M)$ (this allows singular operator matrices when we apply Theorem 2.2 for the inverses).

Theorem 3.2. *(Cf. Remark 3.1.) Assume Sylvester displacement $\nabla_{A,B}(M)$. Then*

a) all trailing principal blocks (that is Schur complements) computed in the forward recursive process of the adapted MBA algorithm have displacement ranks at most $r+4$,

b) all leading principal blocks processed in the forward recursive process of the adapted MBA algorithm have displacement ranks at most $r+6$, and

c) all other matrices computed in the forward and backward recursive processes of the adapted MBA algorithm have displacement ranks at most $2r+12$.

Proof. We have $S^{(g)}(S^{(h)}(M)) = S^{(g+h)}(M)$ because the MBA algorithm is a structure preserving variant of the block Gauss–Jordan elimination. Likewise $S^{(g)}(M^{(g+h)}) = (S^{(h)}(M))^{(g)}$. Therefore all trailing principal blocks computed in the MBA forward recursive process are Schur complements in the respective submatrices $M^{(k)}$. Now part a) follows from Theorem 2.4 (applied in the case $i = j$) and Remark 2.1 because the inverse of every Schur complement is a trailing principal (that is, southeastern) block of the inverse of the matrix itself (cf. equation (3.4)) and because $dr(N) = dr(N^{-1})$ (cf. Theorem 2.2).

Part b) follows from part a) and Theorem 2.4.

Let us prove part c). In the first step of the forward recursive MBA process we compute the off-diagonal blocks $M_{00}^{-1}M_{01}$ and $M_{10}M_{00}^{-1}$. In the next steps we compute similar products $\widehat{M}_{00}^{-1}\widehat{M}_{01}$ and $\widehat{M}_{10}\widehat{M}_{00}^{-1}$ where the blocks $\widehat{M}_{ij}$ denote the (i,j)th blocks of the respective principal block computed in the previous step of the forward recursive process. As we have observed in

the proof of part a), such a principal block is the matrix $S^{(k)}(M^{(h)})^{(g)}$ for some integers g, h and k. Its any block is also a block of the matrices M or $S^{(k)}(M^{(h)})$ for some pair of h and k. Now combining part a) with Theorems 2.2–2.4 implies the bound $dr(B) \le 2r + 12$ claimed in part c). Indeed this bound surely covers the factors $M_{00}^{-1}M_{01}$ and $M_{01}M_{00}^{-1}$ of the blocks $\tilde{M}_{10}$ and $\tilde{M}_{01}$ of the inverse M^{-1}, respectively (cf. equations (3.3)), but the same bound is extended to the operands involved in the computation of the northwestern blocks computed in the backward process. Equations (3.4) and the inequalities $dr((M_{00}^{-1}M_{01})\tilde{M}_{00}) \le dr(M_{00}^{-1}M_{01}) + dr(\tilde{M}_{00}))$ support this extension at its final step, and similar relationships support it at the other steps. The stronger upper bound $r + 4$ holds for the southeastern blocks computed in the backward process because they are the inverses of the Schur complements $S^{(k)}(M^{(h)})$ for some integers h and k, and so we can apply Theorems 2.2 and 2.4. □

According to Theorem 3.2, displacement generators of all matrices involved into the MBA process have lengths in $O(r)$. For the final transition from M_{00}^{-1} and S^{-1} to M^{-1} (performed in terms of generators) we use $A = O(rm_M + r^2m(n)) = O(rm_M)$ ops. At the $(i-1)$st preceding level of the recursion we perform similar operations with 2^i matrices of sizes roughly $(n/2^i) \times (n/2^i)$. For a positive constant c and $m_M \ge cn$ this means $O(rm_M)$ ops at each of the $\lceil \log n \rceil$ levels of the MBA backward recursion and thus $O(rm_M \log n)$ ops overall. This completes the proof of Theorem 3.1 under Sylvester displacements. Theorem 2.1 enables extension to Stein displacements $\Delta_{A,B}$. We recall bound (2.5) to treat the cases where $A, B \in \{Z_0, Z_0^T\}$. □

Remark 3.1. *In the case of the Cauchy-like structure, the MBA recursive process involves only diagonal operator matrices, and so the bounds in Theorem 3.2 decrease to r in parts a) and b) and to 2r in part c). We have a smaller decrease in the case of the Vandermonde-like structure, where one half of the operator matrices in the MBA recursive process are diagonal. In the latter case and in the case of Toeplitz-like structure, we can choose the operator matrix B=Z_0^T, to decrease the bounds in Theorem 3.2 because the $(1,0)$th block of this matrix is filled with zeros and thus has rank zero (cf. Remarks 2.1 and 2.2).*

Corollary 2.3 reduces the inversion of matrices with the structures of Vandermonde and Cauchy types to the inversion of Toeplitz-like matrices because $M^{-1} = V_2(V_1MV_2)^{-1}V_1$. This implies the following result (cf. [P89/90]).

Corollary 3.1. *The upper estimates of Theorem 3.1 can be decreased to $O(l^2n + m(n)r^2 \log n)$ field operations for $m(n)$ in (2.3), even for matrices with the structures of Vandermonde and Cauchy types.*

3.2 Inversion of nonsingular structured matrices in $\mathbb{Z}_p$

Corollary 3.2. *Assume a random prime p in the range $(y/20, y]$ for a sufficiently large integer y and a nonsingular matrix $M \in \mathbb{Z}^{n\times n}$ having structure of Toeplitz, Hankel, Vandermonde or Cauchy type, given with its displacement*

generator of a length l, and having a displacement rank r. Then a) the matrix M^TM is expected to be strongly nonsingular in $\mathbb{Z}_p$, and b) if it is strongly nonsingular, then a displacement generator of length r for the matrix $M^{-1} \mod p$ can be computed by using $O(l^2n + r^2m(n)\log n)$ operations in $\mathbb{Z}_p$.

Proof. The matrix M^TM is strongly nonsingular in $\mathbb{Z}$ (cf. Definition 2.2) and is expected to stay such in $\mathbb{Z}_p$ due to Theorem 2.11. This proves part a). Part b) follows from Corollary 3.1 for $\mathbb{F} = \mathbb{Z}_p$ and from the equation $M^{-1} = (M^TM)^{-1}M^T$. □

3.3 Singular matrices with displacement structure

Let us extend our study to the case of singular input matrices M having a rank ρ and a displacement rank r. In this case we seek the inverse of a $\rho \times \rho$ nonsingular submatrix of the matrix M.

Theorem 3.3. *Assume that in a field $\mathbb{F}$ an $n \times n$ matrix M of a rank $\rho < n$, has generic rank profile, has structure of (a) the Toeplitz or Hankel types or (b) Vandermonde or Cauchy types, has a displacement rank r, and is given with a displacement generator of a length $l = O(r)$. Then (i) the rank ρ and (ii) a displacement generator of length r for the matrix $(M^{(\rho)})^{-1}$ can be computed by using $O(rm_{M^{(\rho)}} \log \rho) = O(m(\rho)r^2 \log^{1+\delta} \rho)$ ops in $\mathbb{F}$ where $\delta = 0$ in case (a) and $\delta = 1$ in case (b). (iii) Within the same cost bound one can compute a solution $\mathbf{x}$ to a consistent linear system $M\mathbf{x} = \mathbf{f}$, in $O(m_M)$ additional ops one can verify consistency of the system, and in $O(rm_M)$ additional ops one can compute a shortest displacement generator for a matrix whose columns define a basis for the null space of the matrix M.*

Proof. Apply the adapted MBA algorithm as in the case of strongly nonsingular input matrices until it factorizes the submatrix $M^{(\rho)}$ and computes a shortest displacement generator (of length $r+O(1)$) for the matrix $(M^{(\rho)})^{-1}$. This takes $O(rm_{M^{(\rho)}} \log \rho)$ ops overall. Then the algorithm stops because it is prompted to invert one-by-one matrix filled with the zero.

To solve a consistent nonhomogeneous linear system $M\mathbf{x} = \mathbf{f}$, multiply the matrix $(M^{(\rho)})^{-1}$ by the subvector made up of the first ρ coordinates of the vector $\mathbf{f}$ and append $n - \rho$ zero coordinates to the product to obtain a solution vector $\mathbf{x}$. This stage involves $O(m_{M^{(\rho)}})$ ops.

To verify consistency of the nonhomogeneous linear system, multiply the matrix M by the vector $\mathbf{x}$ and compare the product with the vector $\mathbf{f}$. In fact one only needs to multiply the $n \times (n - \rho)$ southwestern submatrix by the leading subvector of the dimension ρ in the vector $\mathbf{x}$. If $\mathbf{f} = \mathbf{0}$ and if we seek a solution $\mathbf{x} = (x_i)_{i=1}^n$ to the system $M\mathbf{x} = \mathbf{0}$, then we substitute $x_n = 1$ into this system and arrive at a nonhomogeneous linear system with $n-1$ unknowns and equations.

Finally substitute $M_{00} = M^{(\rho)}$ into (3.1) and observe that the columns of the matrix $\begin{pmatrix} (M^{(\rho)})^{-1}M_{01} \\ -I_{n-\rho} \end{pmatrix}$ form a basis for the null space of the matrix M. One

can compute a shortest displacement generator for the matrix $(M^{(\rho)})^{-1}M_{01}$ in $O(rm_M)$ additional ops (cf. Theorem 2.3). □

Theorem 3.4. *I) To extend Theorem 3.3 to the case of input matrices not having generic rank profile it is sufficient to perform $O(m_M)$ additional ops, to generate $2n-2$ random parameters in the field $\mathbb{F}$, and to allow Monte Carlo randomization, that is, to allow erroneous output with a low probability. II) By performing additional $O(rm_M)$ ops one yields Las Vegas randomization, that is, either fails with a low probability or arrives at the correct output.*

Proof. Part I) is implied by the following theorem.

Theorem 3.5. *Let a finite set S of a sufficiently large cardinality $|S|$ lie in a field $\mathbb{F}$ and let a matrix M lie in $\mathbb{F}^{n\times n}$. Define randomized preprocessing $M \leftarrow XMY$ where $X = X_g$, $Y = Y_h$ for $g, h \in \{1, 2\}$, $X_1 = (\frac{x_j}{s_i - t_j})_{i=1}^n$, $Y_1 = (\frac{y_j}{u_i - v_j})_{i=1}^n$, $X_2 = Z_0^T(\mathbf{x})$, $Y_2 = Z_0(\mathbf{y})$, $x_1 = y_1 = 1$, and the other $2n-2$ coordinates of the vectors $\mathbf{x} = (x_i)_{i=1}^n$ and $\mathbf{y} = (y_i)_{i=1}^n$ are randomly sampled from the set S. Then both matrices X_2 and Y_2 are nonsingular and with a probability of at least $(1 - \rho/|S|)^2$ both matrices X_1 and Y_1 are nonsingular. If the matrices X and Y are nonsingular, then with a probability of at least $1-(\rho+1)\rho/|S|$ matrix XMY has generic rank profile (and therefore is strongly nonsingular if the matrix M is nonsingular).*

Proof. See [KS91] on the case $X = X_2$, $Y = Y_2$ and [P01, Corollary 5.6.3] on the case $X = X_1$, $Y = Y_1$. □

We can extend the dispacement structure of the matrix M to the matrix XMY by choosing appropriate matrices $X = X_i$ and $Y = Y_i$ for $i = 1, 2$ to match the operator matrices associated with the matrix M. Then $dr(XMY) \le dr(M) + 2$, the matrix XMY is computed at a low cost (see Theorem 2.3), and so its recursive factorization and a shortest displacement generator for the matrix $M^{-1} = Y(XMY)^{-1}X$ are computed within the cost bounds of Corollary 3.1. This proves part I) of Theorem 3.5.

To prove part II), verify correctness of the rank computation as follows: compute a displacement generator of length $O(r)$ for the Schur complement $S(XMY, (XMY)^{(\rho)})$ of the block $(XMY)^{(\rho)}$ in the matrix XMY (cf. equation (3.5) and Theorems 2.2, 2.3, and 3.3), compress it to the minimum length (cf. Corollary 2.1), and verify that this length is zero. □

Similarly to Corollary 3.1, we refine the estimates of Theorems 3.3 and 3.4 in the case (b).

Corollary 3.3. *In the upper estimates of Theorems 3.3 and 3.4 one can replace the quantity m_M with $rm(n)$, even in case of input matrices with the structures of Vandermonde and Cauchy types provided in the estimate in Theorem 3.3(ii) the matrix $(M^{(\rho)})^{-1}$ is replaced with $V_1(M^{(\rho)})^{-1}V_2$ where V_1 and V_2 are nonsingular Vandermonde or Vandermonde-like matrices (one of them can be the identity matrix).*

Clearly, the results of this subsection can be applied to matrices M in the field $\mathbb{F} = \mathbb{Z}_p^{n\times n}$ for any prime p. Furthermore for a large integer y and a random prime p chosen in the range $(y/20, y]$, a matrix $M \in \mathbb{Z}^{n\times n}$ is likely to keep its rank, displacement rank, and displacement generator in the transition from the ring $\mathbb{Z}$ to the field $\mathbb{Z}_p$ (cf. Theorem 2.11). Therefore the results of this subsection can be extended to structured integer matrices.

4 Hensel's lifting for a linear system of equations

In this section we recall Hensel's lifting algorithm in [MC79], [D82] but present it in a generalized form allowing to run it in the rings $\mathbb{Z}_{qs}$ for any pair of integers $q > 0$ and $s > 1$. The case of $q = 1$ and a prime s covers classical lifting in [MC79], [D82]. If q and s equal the powers of two, we have *binary* Hensel's lifting.

Let us be given a vector $\mathbf{f}$ and black box subroutines for multiplying by a vector a factor-q nonsingular matrix M in $\mathbb{Z}_{qs}^{n\times n}$ (see Definition 2.7) and matrix Q satisfying (2.2). Then the following algorithm computes the first h terms in the vector expansion $M^{-1}\mathbf{f} = \sum_{i=0}^{\infty} \mathbf{u}^{(i)} s^i$, $\mathbf{u}^{(i)} \in \mathbb{Z}_{qs}^n$, $i = 0, 1, \ldots$.

Algorithm 4.1. Hensel's lifting. (See Examples 4.1–4.3 below.)

INPUT: *a matrix $M \in \mathbb{Z}^{n\times n}$, a vector $\mathbf{f} \in \mathbb{Z}^n$, three positive integers h, q, and s, and a matrix $Q \in \mathbb{Z}_{qs}^{n\times n}$ satisfying (2.2).*

OUTPUT: *the vector $\mathbf{x}^{(h)} \in \mathbb{Z}^n$ such that $M\mathbf{x}^{(h)} = (q\mathbf{f}) \bmod (qs^h)$.*

INITIALIZATION: $\mathbf{r}^{(0)} = \mathbf{f}$.

COMPUTATIONS: *for $i = 0, 1, \ldots, h-1$, compute the vectors*

$$\mathbf{u}^{(i)} = Q\mathbf{r}^{(i)} \bmod (qs), \quad \mathbf{r}^{(i+1)} = (q\mathbf{r}^{(i)} - M\mathbf{u}^{(i)})/(qs).$$

Output the vector $\mathbf{x}^{(h)} = \sum_{i=0}^{h-1} \mathbf{u}^{(i)} s^i$.

Example 4.1. $M = \left(\begin{smallmatrix} 2 & 1 \\ 3 & 2 \end{smallmatrix}\right)$, $\mathbf{f} = \binom{3}{4}$. *So* $\mathbf{x} = \binom{2}{-1}$. *By applying Algorithm 4.1 for $q = 1$, $s = 2$, $\mathbf{r}^{(0)} = \mathbf{f}$, we successively compute* $Q = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$, $\mathbf{u}^{(0)} = \binom{0}{1}$, $\mathbf{r}^{(1)} = \binom{1}{1}$, $\mathbf{u}^{(1)} = \binom{1}{1}$, $\mathbf{r}^{(2)} = \binom{-1}{-2}$, $\mathbf{u}^{(2)} = \binom{0}{1}, \ldots$ *to define* $\mathbf{x}^{(3)} = \mathbf{x} \bmod 8 = \binom{0}{1} + 2\binom{1}{1} + 4\binom{0}{1}$.

Example 4.2. $M = \left(\begin{smallmatrix} 4 & 1 \\ 6 & 2 \end{smallmatrix}\right)$, $\mathbf{f} = \binom{3}{4}$. *So* $\mathbf{x} = \binom{1}{-1}$. *By applying Algorithm 4.1 for $q = s = 2$, $\mathbf{r}^{(0)} = \mathbf{f}$, we successively compute* $Q = \left(\begin{smallmatrix} 0 & 1 \\ 2 & 0 \end{smallmatrix}\right)$, $\mathbf{u}^{(0)} = \binom{0}{2}$, $\mathbf{r}^{(1)} = \binom{1}{1}$, $\mathbf{u}^{(1)} = \binom{1}{2}$, $\mathbf{r}^{(2)} = \binom{-1}{-2}$, $\mathbf{u}^{(2)} = \binom{2}{2}, \ldots$. *So, we have* $\mathbf{x}^{(3)} = 2\mathbf{x} \bmod 8 = \binom{0}{2} + 2\binom{1}{2} + 4\binom{2}{2}$, $(M\mathbf{x}^{(h)} - 2\mathbf{f}) \bmod 2^{h+1} = 0$ *for $h = 1, 2, 3$.*

Example 4.3. $M = \left(\begin{smallmatrix} 32 & 2 \\ 48 & 4 \end{smallmatrix}\right)$, $\mathbf{f} = \binom{24}{32}$. *So,* $\mathbf{x} = \binom{1}{-4}$, $s_1(M) = 2$, $s_2(M) = 16$. *We can apply Algorithm 4.1 to* M *and* $\mathbf{b}$ *for* $q = 3, s = 2$.

The following theorem shows correctness of the algorithm (see part b) and bounds the precision of its computations. For $q = 1$ and a prime s, Algorithm 4.1 and the theorem have appeared in [D82].

Theorem 4.1. *For* $\mathbf{r}^{(i)}$ *and* $\mathbf{x}^{(h)}$ *in Algorithm 4.1, we have*

a) $\mathbf{r}^{(i)} \in \mathbb{Z}^n$ *for all* i;

b) $M\mathbf{x}^{(h)} = q\mathbf{f} \bmod (qs^h)$;

c) all components $r_j^{(i)}$ *of all vectors* $\mathbf{r}^{(i)} = (r_j^{(i)})_j$ *satisfy the bounds* $|r_j^{(i)}| \le |f_j|/s^i + \alpha n \frac{qs-1}{q} \sum_{k=1}^{i} s^{-k} < \beta/s^i + \alpha n(qs-1)/(qs-q) < \gamma$ *where* $M = (m_{i,j})_{i,j=1}^n$, $\mathbf{f} = (f_j)_{j=1}^n$,

$$\beta = \beta(\mathbf{f}) = \max_j |f_j|, \quad \alpha = \alpha(M) = \max_{i,j} |m_{i,j}|, \quad \gamma = 2\alpha n + \beta. \tag{4.1}$$

Proof.

a) $(q\mathbf{r}^{(i)} - M\mathbf{u}^{(i)}) \bmod (qs) = (qI - MQ)\mathbf{r}^{(i)} \bmod (qs)$, and the claim follows because $MQ = qI \bmod (qs)$.

b) $M\mathbf{x}^{(h)} = \sum_{i=0}^{h-1} M\mathbf{u}^{(i)} s^i = \sum_{i=0}^{h-1} (q\mathbf{r}^{(i)} - qs\mathbf{r}^{(i+1)})s^i = q\mathbf{f} - qs^h\mathbf{r}^{(h)} = q\mathbf{f} \bmod (qs^h)$.

c) By definition, all components $u_j^{(i)}$ of all vectors $\mathbf{u}^{(i)}$ satisfy $|u_j^{(i)}| \le qs - 1$, and so $qs|r_j^{(i+1)}| \le q|r_j^{(i)}| + \alpha n \max_k |u_k^{(i)}| \le q|r_j^{(i)}| + (qs-1)\alpha n$. The claim now follows by induction on i.

□

Clearly, the arithmetic computational cost of a lifting step is in $m_M + m_Q + O(n)$. Here is an upper bound on the precision of computing.

Lemma 4.1. *Algorithm 4.1 operates with integers in the range* $[-2^{d_1}, 2^{d_1})$ *where*

$$d_1 \le \lceil \log(2qs\gamma) \rceil \tag{4.2}$$

for γ *in (4.1).*

Proof. The lemma follows from Theorem 4.1 a) and c) since the vectors $\mathbf{u}^{(i)}$ are computed in $\mathbb{Z}_{qs}$. □

The bit precision of computing in Algorithm 4.1 is at most d_1 and is only $\lceil \log(qs) \rceil$ at the stages of computing the vectors $\mathbf{u}^{(i)}$. Therefore, each lifting step requires $(m_M + O(n))\mu(d_1) + m_Q\mu(\log(qs))$ bit operations. If λ is the length of a computer word and $d_1 < \lambda$, then all arithmetic operations in the algorithm are word operations, that is performed within the computer precision. We save lifting steps and word operations if we choose *saturated initialization*, that is choose q and s that maximize the value d_1 subject to the bound $d_1 < \lambda$.

5 Computing the rational solution of a structured integer linear system of equations

Given a prime p and a structured integer linear system $M\mathbf{x} = \mathbf{f}$, nonsingular in $\mathbb{Z}_p$ (cf. Theorem 2.11 and the end of Section 3), we can apply the algorithms in the previous sections to compute the vectors $\mathbf{x} \mod p$ and then recursively $\mathbf{x} \mod p^i$ for $i = 2, 3, \dots, h$. Finally we can apply the algorithm supporting Theorem 2.10 to reconstruct the rational solution $\mathbf{x}$. Next and in Appendix B we specify the reconstruction techniques, traced back to [P87, Appendix] and [P88] and more recently used in [ABM99], [CFG99], [EGV00], and [MS04].

Theorem 5.1. *Let $\mathbf{x} = qM^{-1}\mathbf{f}$ be a unique solution to the linear system $M\mathbf{x} = q\mathbf{f}$. Assume $\rho(d)$ in (2.6) and α, β and γ in (4.1). Write*

$$d = \lceil \log(2(\alpha\sqrt{n})^{2n-1}n\beta q)\rceil = O(n\log\gamma + \log q), \tag{5.1}$$

$$h = \lfloor \log_s(2(\alpha\sqrt{n})^{2n-1}n\beta)\rfloor. \tag{5.2}$$

Let the vector $\mathbf{x}^{(h)} = \sum_{i=0}^{h-1}\mathbf{u}^{(i)}p^i = \mathbf{x} \mod (qs^h)$ be computed in $h-1$ steps of Algorithm 4.1. Then one can recover the vector $\mathbf{x}$ from the vector $\mathbf{x}^{(h)}$ by performing $B = n\rho(d)$ bit operations.

Proof. Suppose two coprimes $\nu_j = \nu(x_j)$ and $\delta_j = \delta(x_j)$ define the rational components $x_j = \nu_j/\delta_j$ of the vector $\mathbf{x} = (x_j)_j = qM^{-1}\mathbf{f}$. Fix the smallest integer $k > 2(\alpha\sqrt{n-1})^{n-1}n\beta q$. Note that $s^h > 2(\alpha\sqrt{n})^{2n-1}n\beta$ for h in (5.2). Recall that $M^{-1}\det M = \operatorname{adj} M$ and deduce from Fact 2.2 that $l = qs^h > 2|\nu_j|\delta_j$ and $2|\nu_j| < k \le qs^h$. Then according to Section 2.4, every component x_j can be uniquely recovered from $qx_j \mod (qs^h)$ by using $\rho(d)$ bit-operations. □

We can accelerate the reconstruction of the rational solution based on Las Vegas randomization along the following line. Instead of the n rational coordinates x_i of the vector $\mathbf{x} = (x_i)_i$, recover much fewer linear combinations $\nu_j/\delta_j = \sum_i c_i^{(j)}x_i$ of these coordinates with random integers $c_i^{(j)}$, $j = 1, 2, \dots, K$. Then Smith's largest factor $s_n(M)$ (cf. Definition 2.6 and bound (2.1)) is likely to divide the least common multiple δ_{lcm} of the integer denominators $\delta_1, \dots, \delta_K$. If it does, the vector $\delta_{\mathrm{lcm}}\mathbf{x}$ is filled with integers and can be readily reconstructed from its value modulo p^h based on Fact 2.1. In Appendix B we specify this reconstruction and estimate its randomized Las Vegas Boolean cost under a tolerance ϵ on the failure probability.

6 Computational complexity estimates

6.1 Computations with matrices having displacement structure

In this subsection we assume a vector $\mathbf{f} \in \mathbb{Z}^n$ and a matrix $M \in \mathbb{Z}^{n\times n}$ having structure of Toeplitz, Hankel, Vandermonde, or Cauchy type and given with a

Table 6.1: The bit-operation complexity under equations (6.1)–(6.3).

Initialization	$O((r^2 m(n) \log n)\mu(\log n)) = \tilde{O}(r^2 n \log^3 n)$
Lifting	$O(rnm(n)\mu(\log n)) = \tilde{O}(rn^2 \log^2 n)$
Reconstruction (randomized)	$O(n\mu(n \log n) + \rho(n \log n) \log n) = \tilde{O}(n^2 \log^2 n)$

displacement generator of a length r. In Table 6.1 we summarize the estimates for the overall randomized Las Vegas complexity of the exact solution of a nonsingular linear system $M\mathbf{x} = \mathbf{f}$. We involve the expressions $m(n)$, $\mu(d)$ and $\rho(d)$ defined in (2.3)–(2.6) and m_S in Definition 2.5, and for simplicity choose a basic prime p and the tolerance ϵ to the error probability in the randomized rational reconstruction of the output such that

$$\log p = O(\log n), \tag{6.1}$$

$$\log(1/\epsilon) = O(\log n). \tag{6.2}$$

Furthermore we assume that

$$\log_p \gamma = O(1). \tag{6.3}$$

In Appendix C we do not assume equations (6.1)–(6.3) and display more detailed estimates.

Theorem 6.1. *The estimates in Table 6.1 hold provided $r = O(1)$, equations (6.1)–(6.3) hold, and Las Vegas randomization with random parameters involving $O(n \log n)$ bits is allowed. The estimates apply to testing a linear system $M\mathbf{x} = \mathbf{f}$ for consistency and to solving it if it is consistent. The estimates of Table 6.1 for the initialization, lifting and rational solution reconstruction sum to $\tilde{O}(n^2 \log^2 n)$ bit-operations provided $r = O(1)$.*

Theorem 6.2. *Under preprocessing $M \rightarrow XMY$ in Section 3.3, application of Las Vegas (resp. Monte Carlo) randomization with $O(n \log n)$ random bits and performing $\tilde{O}(n^2 \log^3 n)$ (resp. $\tilde{O}(n^2 \log^2 n)$) bit operations are sufficient to compute displacement generators of the minimum length for a nonsingular $\rho \times \rho$ submatrix W of the matrix XMY, for the inverse W^{-1}, and for a matrix whose columns define a basis for the null space of the matrix M. Generating $O(n \log n)$ random bits and performing $\tilde{O}(n^2 \log^3 n)$ (resp. $\tilde{O}(n \log^3 n)$) bit operations are sufficient for randomized Las Vegas (resp. Monte Carlo) computation of the rank ρ of the matrix M.*

Our bit-operation complexity estimates are record low and furthermore are nearly optimal because $n^2 \log n$ bits are required to represent the n rational

coordinates of the vector $\mathbf{x}$, and so computing these values takes at least as many bit operations.

Remark 6.1. *Suppose the integer* $(\det M) \mod p^h \neq 0$ *and the vector* $\mathbf{x} \mod p^h$ *are available. Then we can compute the integer vector* $(\det M)\mathbf{x}$ *by applying Fact 2.1, instead of computing the rational vector* $\mathbf{x} = M^{-1}\mathbf{f}$ *by applying Theorem 2.10. Still this only supports the solution in* $\tilde{O}(n^2 \log^2 n)$ *bit operations overall but avoids randomization at the reconstruction stage. The bit count decreases by logarithmic factor (still using no randomization at the reconstruction stage) in the cases where the output is known to be integral. This occurs, e.g., for Berlekamp–Massey problem (cf. Definition 2.9) and at the final stage of Wiedemann's algorithm [W86], which computes the minimum polynomial, determinant, and Smith's factors of an integer matrix.*

If λ, the length of a computer word, exceeds $\log\lceil 2\gamma p\rceil$, so that lifting and initialization are performed within the computer precision (cf. Lemma 4.1), then the word operation cost of performing these stages is by the factor of λ smaller than the bit–complexity estimates.

6.2 Padé approximation and related computations

Theorem 6.3. *A randomized Las Vegas (resp. Monte Carlo) algorithm for an* (m, n) *Padé approximation for a polynomial* $t(x) = \sum_{i=0}^{m+n} t_i x^i$ *(cf. Definition 2.8) generates* $O(N \log N)$ *random bits and in addition performs* $O(N^2 \log^3 N)$ *(resp.* $O(N^2 \log^2 N)$*) bit operations provided* $N = n + m$ *and equations (6.1)–(6.3) hold for* $\gamma = \max_{i=0}^{N} |t_i|$.

Proof. First recall from [BGY80], [P01, Section 2.11] that the task of computing an (m, n) Padé approximation can be reduced to the solution of the Toeplitz linear system $T\mathbf{v} = -v_0\mathbf{t}$ where $T = (t_{m+i-j})_{i,j=0}^{n-1}$, $\mathbf{v} = (v_{j+1})_{j=0}^{n-1}$, $\mathbf{t} = (t_{j+1})_{j=0}^{n-1}$, $v_0 = 1$ if $\rho = \mathrm{rank}(T) = n$, and $v_0 = 0$ if $\rho = \mathrm{rank}(T) < n$, in which case $\det(T^{(\rho)}) \neq 0$, that is the $\rho \times \rho$ leading principal block of the matrix T is nonsingular. It remains to apply the algorithms supporting Theorem 6.2 to solve this Toeplitz linear systems. □

Theorem 6.3 and Fact 2.3 together imply the following result.

Corollary 6.1. *A randomized Las Vegas (resp. Monte Carlo) algorithm for the Berlekamp–Massey Problem (cf. Definition 2.9), for a positive integer* s *and* $2s$ *numbers* $a_0, a_1, \ldots, a_{2s-1}$*, generates* $O(s \log s)$ *random bits and in addition performs* $O(s^2 \log^3 s)$*(resp.* $O(s^2 \log^2 s)$*) bit operations provided equations (6.1)–(6.3) hold for* $\gamma = \max_{i=0}^{2s-1} |a_i|$.

Theorem 6.4. *Greatest common divisor* $g(x) = \mathrm{gcd}(\mathrm{u}, \mathrm{w})$ *and least common multiple* $\mathrm{lcm}(\mathrm{u}, \mathrm{w})$ *of two polynomials* $u(x) = \sum_{i=0}^{m} u_i x^i$ *and* $w(x) = \sum_{i=0}^{n} w_i x^i$ *(cf. Definition 2.10) can be computed by a randomized Las Vegas (resp. Monte Carlo) algorithm that generates* $O(N \log N)$ *random bits and in addition performs* $\tilde{O}(N^2 \log^3 N)$ *(resp.* $O(N^2 \log^2 N)$*) bit operations provided* $N = n + m$ *and equations (6.1)–(6.3) hold for* $\gamma = \max\{\max_{i=0}^{m} |u_i|, \max_{i=0}^{n} |w_i|\}$.

Proof. We first compute the degree $k = \deg g(x)$ of the polynomal $g(x) = \sum_{i=0}^{k} g_i x^i = \gcd(\mathrm{u}, \mathrm{w})$ by applying the reduction gcd $\rightarrow$ Padé (see Fact 2.4) and then applying our algorithm that supports Theorem 6.3. To avoid the growth of the coefficients and the cost bounds in the transition gcd $\rightarrow$ Padé, we compute $\deg g(x)$ in $\mathbb{Z}_p$ for a reasonably large prime p, so that the bit-cost stays within the claimed bounds and the resulting degree value is likely to withstand the transition to $\mathbb{Z}$.

To yield the Las Vegas estimates, we must verify that the degree indeed remains the same in this transiton. For a fixed candidate value $k = \deg g(x)$ let U_k and W_k denote the Toeplitz matrices of the sizes $(m+n-k) \times (n-k)$ and $(m+n-k) \times (m-k)$, respectively, which are simultaneously upper and lower triangular, that is, are filled with zeros above their upper diagonals and below their lower diagonals. Thus they are defined by their first columns, which are $(u_0, \dots, u_m, 0, \dots, 0)^T$ and $(w_0, \dots, w_n, 0, \dots, 0)^T$, respectively. The $(m+n-k) \times (m+n-2k)$ matrix (U_k, W_k) (called subresultant matrix) has Toeplitz-like structure, has displacement rank at most two, and is known to have rank k if $k = \deg g(x)$. Thus our algorithms supporting Theorem 6.1 enable us to verify the equation $k = \deg g(x)$ within the claimed cost bounds.

Now, having certified the degree k, we wish to compute the gcd. We recall from [BGY80], [P01, Section 2.10] that $g(x) = s(x)u(x) + t(x)w(x)$ where $s(x) = \sum_{i=0}^{n-k} s_i x^i$, $t(x) = \sum_{i=0}^{m-k} t_i x^i$, and the coefficient vectors $\mathbf{s} = (s_i)_{i=0}^{n-k}$ and $\mathbf{t} = (t_i)_{i=0}^{m-k}$ satisfy the subresultant equation $\tilde{U}_k \mathbf{s} + \tilde{W}_k \mathbf{t} = \mathbf{e}_0$ where $\mathbf{e}_0 = (1, 0, \dots, 0)^T$ is the first unit coordinate vector and $\tilde{U}_k$ and $\tilde{W}_k$ are Toeplitz matrices of the sizes $(m+n-2k) \times (n-k)$ and $(m+n-2k) \times (m-k)$, respectively, obtained by deleting the last k rows of the matrices U_k and W_k, respectively. It follows that the coefficient matrix $(\tilde{U}_k, \tilde{W}_k)$ of the above subresultant system is again Toeplitz-like with the displacement rank of at most three. Now the claimed complexity bounds for computing the gcd $g(x) = \gcd(\mathrm{u}, \mathrm{w})$ follow from Theorem 6.2. They are immediately extended to the task of computing the polynomial $\mathrm{lcm}(\mathrm{u}, \mathrm{w}) = u(x)w(x)/\gcd(\mathrm{u}, \mathrm{w})$. □

6.3 Computations with rank structured and sparse structured matrices

In the cases of (l, u) rank structured matrices $M \in \mathbb{Z}^{n \times n}$ (resp. sparse structured matrices $M \in \mathbb{Z}^{n \times n}$ whose graphs have $s(n)$-separator families), we apply Theorem 2.8 (resp. 2.9) to estimate the Boolean cost of the initialization of lifting and performing it. For these two stages (assuming equations (6.2) and (6.3)), we obtain the bounds $\tilde{O}(w^2 n \log n)$ and $\tilde{O}(w n^2 \log n)$ where $w = l + u + 1$ in case a) and $\tilde{O}(s(n)^3 \log n)$ and $s(n) + |E(M)|$ in case b), respectively. They turn into $\tilde{O}(n^2 \log n)$ in both stages where $w = O(1)$ in case a) and $s(n) = O(n^{2/3})$ and $|E(M)| = O(n)$ in case b) (cf. Section 2.6). This matches the cited lower bound on the Boolean cost of computing the vector $\mathbf{x}$, but the cost of the reconstruction of the rational solution raises the overall cost bound to $\tilde{O}(n^2 \log^2 n)$ (see Section 5 and Remark 6.1).

7 Initialization based on iterative refinement

7.1 Introductory comments

Compare our Boolean cost estimates for lifting and its initialization based on factorization algorithms, supported by Theorems 2.6, 2.8, 2.9, and 3.1 and Corollary 3.3. The estimates are much lower for the initialization, except that it tends to degenerate if performed modulo two or a power of two. Instead of factorization, our alternative initialization algorithm in this section uses numerical iterative refinement [GL96, Section 3.5.3], which is numerical counterpart of Hensel's lifting and runs about as fast as lifting. It is effective wherever the bounds m_M and $i_{M,2l}$ (in $O(m_M)$) are small for a reasonably small integer l. In particular it is effective for matrices M with displacement structure and for banded and generic rank structured matrices M. Thus our initialization unifies computations for these two most important classes of structured matrices.

The algorithm yields matrix Q that satisfies equation (2.2) for $q = p^u$ and $s = p^v$ equal to the powers of a fixed prime p where the sum $b = u + v$ must exceed the order of p in Smith's largest factor $s_n(M)$ (cf. Definition 2.6). We can ensure this property by choosing $p^b > H$ where H is Hadamard's bound in Fact 2.2 (cf. (2.2)). This bound, however, tends to be overly pessimistic, thus implying excessive overall computational cost bound on the average input. Thus we choose the exponent b dynamically, first testing more moderate values. If this does not work, we output FAILURE and reapply the algorithm for an increased value b.

In view of Section 2.9, a random moderately large prime p is likely to be coprime with $s_n(M)$. If it is, we can choose $q = 1$ and $s = p$ and perform the initialization via numerical iterative refinement as fast as lifting, thus supporting the nearly optimal asympototic time bounds in Section 6.

We describe an algorithm that for a fixed vector $\mathbf{f}$ applies $O((n \log(|M| + p^b) + \log |\mathbf{f}|)m_M)$ ops in iterative refinement to yield the vector $\mathbf{v}$ satifying the equation $\mathbf{v} = qQ\mathbf{f} \mod (qs)$ for Q in (2.2). To initialize lifting for matrices with displacement or rank generators of a length l, we compute a generator for the matrix Q by solving $2l$ linear systems of equations with the coefficient matrices M and M^T and $2l$ right-hand side vectors $\mathbf{f}_i$, thus obtaining $2l$ pairs of basic powers (u_i, v_i) such that $u_i = m/v_i$ for a fixed integer m and for $i = 1, \dots, 2l$. Then we replace these pairs by the single pair (v, u) such that $v = \max_{i=1}^{2l} v_i$, $u = m/v$, $q = p^u$, and $s = p^v$.

7.2 The basic algorithm

Our basic initialization algorithm employs the *numerical rational roundoff* algorithm in [WP03], which recovers a unique rational number x/y from three integers ν, δ, and k provided $1 \le y \le k$, $|x| < k$, $|x|$ and y are coprime unless $x = 0$; $|x/y - \nu/\delta| < 1/(2k^2)$, and $|\nu| < \delta$. The bound of Theorem 2.10 for $d = \delta$ on the bit–operation complexity has been extended to this recovery in [WP03]. The algorithm also uses the following rounding policy.

Definition 7.1. *Represent all components of a vector* $\mathbf{v}$ *as fixed-point numbers with the common fixed point placed in front of the first nonzero digit of the absolutely largest component, so that the fractions of the other components can begin with zeros. Then round all fractions to the closest* t*-digit numbers that all share this fixed point, so that the zeros that follow the fixed point are counted among the* t *digits. The resulting vector* $\tilde{\mathbf{v}}$ *is said to approximate the vector* $\mathbf{v}$ *with the* t*-digit fixed-point precision and with rounding to the closest values.*

The following lemma is immediately verified.

Lemma 7.1. *If a vector* $\tilde{\mathbf{v}}$ *approximates the vector* $\mathbf{v}$ *with the* t*-digit fixed-point precision and with rounding to the closest values, then* $|\tilde{\mathbf{v}} - \mathbf{v}| \le 0.5|\mathbf{v}|/\phi^t$ *assuming* ϕ*-ary digits.*

Algorithm 7.1. *Initialization based on iterative refinement.*

INPUT: *A nonsingular matrix* $M \in \mathbb{Z}^{n\times n}$*, a vector* $\mathbf{f} \in \mathbb{Z}^n$*, a prime* p*, and two positive integers* b *and* t *such that* $m = p^b \ge 2^{t+2}|M|$.

OUTPUT: *either FAILURE if* $s_n(M) \bmod p^v = 0$ *for some* $v \le b$ *(cf. Definition 2.6) or two integers* q *and* s*, both the powers of* p *and such that* $qs = p^b$*, and the vector* $\mathbf{z} = (qM^{-1}\mathbf{f}) \bmod (qs)$.

INITIALIZATION: *Write* $\mathbf{r}_0 = \mathbf{f}$*,* $m = p^b$*,* $M_0 = M + mI$*, and* $Q = I/m$.

COMPUTATIONS:

1. *Write* $\mathbf{w}_i = Q\mathbf{r}_i = \mathbf{r}_i/m$ *and recursively, for* $i = 0, 1, \dots, h-1$ *and*

$$h = \lceil ((2n-1)\log(|M| + m) + \log(2|\mathbf{f}|^2))/t \rceil, \tag{7.1}$$

   compute a) the vectors $\tilde{\mathbf{w}}_i = \mathbf{w}_i + \mathbf{e}_i$ *that approximate the vectors* $\mathbf{w}_i$ *with the* t*-bit fixed-point precision and with rounding to the closest values (cf. Definition 7.1) and b) the vectors* $\mathbf{r}_{i+1} = \mathbf{r}_i - M_0\tilde{\mathbf{w}}_i = \mathbf{r}_i - M\tilde{\mathbf{w}}_i - m\tilde{\mathbf{w}}_i$.

2. *Recover the vector* $\mathbf{z} = M_0^{-1}\mathbf{f}$ *from the vector* $\mathbf{z}_h = \sum_{i=0}^{h-1} \tilde{\mathbf{w}}_i$*, by using the numerical rational roundoff algorithm in [WP03].*

3. *Compute the integer* $v = \max_j \operatorname{ord}_m(\delta((M_0^{-1}\mathbf{f})_j))$*. If* $v > b$*, output the integers* $q = p^v$ *and* $s = p^{b-v} = m/q$*; compute and output the vector*

$$\mathbf{z} = (qM_0^{-1}\mathbf{f}) \bmod (qs) = (qM^{-1}\mathbf{f}) \bmod (qs).$$

   Otherwise output FAILURE.

Stage 1 of Algorithm 7.1 amounts to the extension of the customary numerical algorithm for iterative refinement to produce the output values beyond the double precision.

The algorithm outputs FAILURE if and only if $b \le v$. We have the following estimate.

Theorem 7.1. *$v = \max_j \mathrm{ord}_p((\delta(M_0^{-1})_j) = \mathrm{ord}_p(s_n(M_0)) = \mathrm{ord}_p(s_n(M))$, and so $v \le \mathrm{ord}_p(\det M)$.*

Proof. Combine the definitions of M_0, $\delta(x/y)$, and $s_n(M)$ and bounds (2.1). □

In virtue of Theorems 2.11, the failure of Algorithm 7.1 is unlikely if p is a random prime from a moderately large range (even where $b = 1$). According to [PWa], the failure is also unlikely for a fixed p and random $n \times n$ structured integer matrix M if $p^b \gg n$.

If the algorithm outputs FAILURE, one can apply some heuristic recipes from [PMRW05], [PWa] or reapply the algorithm for an increased value b and/or for p replaced with a distinct basic prime.

7.3 Correctness of the algorithm

W.l.o.g. assume that $M \in \mathbb{Z}_{p^b}^{n\times n}$ and $\mathbf{f} \in \mathbb{Z}_{p^b}^{n}$.

Theorem 7.2. *We have $|\mathbf{z} - \mathbf{z}_h| \le |M_0^{-1}|\ |\mathbf{f}|/2^{th}$.*

Proof. The theorem is a corollary of the two following lemmas.

Lemma 7.2. *We have $\mathbf{z} - \mathbf{z}_h = M_0^{-1}\mathbf{r}_h$.*

Proof. Recall the equations $\mathbf{z}_h = \sum_{i=1}^{h-1} \tilde{\mathbf{w}}_i$ and $M_0\tilde{\mathbf{w}}_i = \mathbf{r}_i - \mathbf{r}_{i+1}$ for all i. By combining them obtain that $M_0\mathbf{z}_h = \mathbf{r}_0 - \mathbf{r}_h = \mathbf{f} - \mathbf{r}_h$. Combine this expression with the equation $\mathbf{z} - \mathbf{z}_h = M_0^{-1}(\mathbf{f} - M_0\mathbf{z}_h)$. □

Lemma 7.3. *We have $|\mathbf{r}_{i+1}| \le |\mathbf{r}_i|/2^t \le |\mathbf{f}|/2^{(i+1)t}$ for all i, $i = 0, 1, \dots$.*

Proof. Recall that $\mathbf{r}_{i+1} = \mathbf{r}_i - M_0\tilde{\mathbf{w}}_i = \mathbf{r}_i - M_0\mathbf{w}_i - M_0\mathbf{e}_i$, $\mathbf{r}_i - M_0\mathbf{w}_i = (I - M_0/m)\mathbf{r}_i = -(M/m)\mathbf{r}_i$, and due to Lemma 7.1 $|\mathbf{e}_i| \le |\mathbf{w}_i|/2^{t+1}$ since we use the binary representation. Now we deduce that $|(M/m)\mathbf{r}_i| \le |\mathbf{r}_i|/2^{t+2}$ and $|M_0\mathbf{e}_i| \le |M + mI|\ |\mathbf{w}_i|/2^{t+1} \le |M + mI|\ |\mathbf{r}_i|/(m2^{t+1}) \le 3|\mathbf{r}_i|/2^{t+2}$. It remains to combine the two latter bounds. □

Numerical rational roundoff ensures correct recovery of the vector $\mathbf{z}$ from $\mathbf{z}_h$ if $|\mathbf{z} - \mathbf{z}_h| < 1/(2|M_0|^{2n-1}|\mathbf{f}|)$. Due to Theorem 7.2, this bound is reached under (7.1). □

7.4 The computational precision and Boolean cost estimates

The complexity of Algorithm 7.1 is estimated in Section 6 for M_0 replacing M and with the precision of the computations adjusted respectively. Let us estimate this precision.

By the definition of the vectors $\tilde{\mathbf{w}}_i$, the binary representations of the components of the vector $\mathbf{r}_{i+1} = \mathbf{r}_i - M_0\tilde{\mathbf{w}}_i$ extend rightward by at most $t + \lceil \log_2 m \rceil$ bits versus the leading bit of the norm $|\mathbf{r}_i|$. At the same time this leading bit itself moves rightward by at least t bits when we move from $\mathbf{r}_i$ to $\mathbf{r}_{i+1}$ because

$|\mathbf{r}_{i+1}| \le |\mathbf{r}_i|/2^t$ (see Lemma 7.3). Thus it is sufficient to use precision of at most $\lceil \log m \rceil$ bits for all components of the vectors $\mathbf{r}_i$ for all i. We increase this precision by at most $\lceil \log_2 |M| \rceil$ and $\lceil \log m \rceil$ when we compute the vectors $M\tilde{\mathbf{w}}_i$ and $m\tilde{\mathbf{w}}_i$, respectively. (In case of binary lifting where $p = 2$ the components of both vectors $\tilde{\mathbf{w}}_i$ and $m\tilde{\mathbf{w}}_i$ are represented with the same precision.) It follows that all asymptotic complexity estimates in Table 6.1 extend to Algorithm 7.1 if $b \log p = O(\log(n\alpha))$. The latter assumption contradicts the bound $b > v$ for v in Theorem 7.1 and for the worst case input matrix M but, according to the analysis and tests in [PWa] (cf. the end of subsection 7.2), not for the average integer matrix M with the displacement structure.

7.5 Extension to computations with singular matrices

As by-product, Algorithm 7.1 determines whether the matrix M is singular in $\mathbb{Z}$. Therefore, for matrices with displacement structure, we can combine the algorithm with the binary search as an alternative to the MBA algorithm for computing the rank ρ and a $\rho \times \rho$ nonsingular submatrix of a preconditioned matrix XMY that has generic rank profile.

8 Matrix inversion via Newton's lifting

Our generalized Newton's lifting algorithm for matrix inversion recursively computes the matrices $X_0, X_1, \dots$ such that

$$MX_0 = qI \bmod (qs), qX_i = X_{i-1}(2qI - MX_{i-1}) \bmod (qs^{2^i}), \tag{8.1}$$

$i = 1, 2, \dots, h$. Deduce that $q^{2^i-1}(qI - MX_i) = (q^{2^{i-1}-1}(qI - MX_{i-1}))^2 = (qI - MX_0)^{2^i} = 0 \mod (qs)^{2^i}$ and therefore $qI - MX_i = 0 \bmod (qs^{2^i})$. For $q = 1$, this is Newton's lifting for matrix inversion [MC79], whose ith step squares the residual matrix $I - MX_{i-1}$, thus implying guadratic convergence of the approximations X_i to M^{-1}. Surely, our initialization recipes for Hensel's lifting can be extended to Newton's.

Every Newton's step (8.1) is essentially reduced to performing $n \times n$ matrix multiplication twice. For general matrices, this operation is expensive, although it is substantially accelerated on multiprocessors.

For matrices M and X_i with consistent displacement or rank structures, the multiplication is performed with their generators and is simplified. For example, in the case of a Toeplitz matrix $T = (t_{k-j})_{k,j} = M/q$, we can represent the inverses $X_i = qM^{-1} \bmod (qs^{2^i})$ in $\mathbb{Z}_{qs^{2^i}}$, $i = 0, 1, \dots$, with their $n \times 2$ generators $X_i(\mathbf{e}_1, \mathbf{t}) = (X_i\mathbf{e}_1, X_i\mathbf{t})$ where $\mathbf{e}_1 = (1, 0, \dots, 0)^T$ and the vector $\mathbf{t}$ is defined in Theorem A.2. Then our iteration (8.1) takes the following form,

$$\begin{aligned} MX_0(\mathbf{e}_1, \mathbf{t}) &= q(\mathbf{e}_1, \mathbf{t}) \bmod (qs), \\ qX_i(\mathbf{e}_1, \mathbf{t}) &= X_{i-1}(2qI - MX_{i-1})(\mathbf{e}_1, \mathbf{t}) \bmod (qs^{2^i}), \end{aligned} \tag{8.2}$$

$i = 1, 2, \ldots$. Its every step is reduced essentially to the multiplication of the matrix M by the $n \times 2$ matrix $X_{i-1}(\mathbf{e}_1, \mathbf{t})$ and of the matrix X_{i-1} by the resulting $n \times 2$ matrix. This takes $O(m(n))$ arithmetic operations (see Theorems A.1 and A.2), which is much less than the complexity of $n \times n$ matrix multiplication.

For $q = 1$ the iteration processes (8.1) and (8.2) are similar to Newton's iteration for numerical inversion of a matrix M [P01, Chapter 6], which takes the forms

$$X_i = X_{i-1}(2I - MX_{i-1}), \quad i = 1, 2, \ldots, \tag{8.3}$$

for general matrix M and

$$X_i(\mathbf{e}_1, \mathbf{t}) = X_{i-1}(2I - MX_{i-1})(\mathbf{e}_1, \mathbf{t}), \quad i = 1, 2, \ldots, \tag{8.4}$$

for Toeplitz matrix M.

In h steps, Newton's lifting computes the solution modulo qs^{2^h}, which the generalized Hensel's lifting yields in 2^h steps, but the precision of computing is roughly doubled in every Newton's step, so that computations with extended precision are generally required already in a smaller number of Newton's steps. If we can parallelize these computations, we yield fast parallel computations, due to dramatic saving of lifting steps, although the overall asymptotic bit–operation cost estimate slightly increases versus Hensel's lifting even in the case where $\mu(d) = O((d \log d) \log \log d)$. Initially, however, one can apply a few steps of Newton's lifting to save word operations wherever the ratio $\lceil \log(2qs\gamma) \rceil / \lambda$ is small.

9 Computing determinants

The most popular numerical and symbolic algorithms for determinants rely on LU or QR factorizations of the input matrix with pivoting (cf. [C92], [S97], [BEPP99], [PY01], and the bibliography therein). Symbolically one computes the determinant in this way with a low precision modulo sufficiently many smaller primes and then reconstruct the integer output by applying the Chinese remainder algorithm. For an $n \times n$ integer matrix M the computation involves $(n^4 \log(n|M|))^{1+o(1)}$ bit operations overall. Some recent randomized symbolic algorithms run asymptotically faster, are technically related to lifting and MBA algorithms, and are at least potentially competitive in practical computations. Furthermore, besides the determinant of an integer matrix, most of the algorithms compute its Smith factors as by-product.

In [P87, Appendix] and [P88] it was proposed to compute the determinant $\det M$ of an $n \times n$ general integer matrix M by solving the linear system $M\mathbf{y} = \mathbf{v}$ for a random integer vector $\mathbf{v}$ modulo a random prime and then to employ Hensel's lifting for the solution. This approach was resurrected and advanced in [ABM99] and [EGV00] to support the randomized Monte Carlo computation of the determinant by using $(n^d \log |M|)^{1+o(1)}$ bit operations where $d = 3$ for the average input matrix M and $d = 7/2$ for the worst case input matrix.

The Las Vegas exponent $d = 10/3$ for the worst case input was obtained in [KV01, Theorem 2] by adapting block Wiedemann algorithm in [C94] (cf. [W86]), which computes $\det(XMY)$ as the z-free term of the characteristic polynomial $\det(XMY - zI)$ for a pair of random preprocessors X and Y whose determinants are readily available. Then the algorithm outputs $\det M = \frac{\det(XMY)}{(\det X)\det Y}$. The characteristic polynomial is obtained from the block Krylov sequence of the matrix XMY. The final stage of the algorithm is the solution of an auxiliary block Toeplitz/Hankel linear system of equations. The papers [P02, Theorem 5.1] and [P04a] incorporated the MBA algorithm and lifting at this stage, which immediately decreased the exponent d to $16/5$ and also removed the last major obstacle for asymptotically faster and practically promising implementation of the algorithm.

A. Storjohann in [S05] applied his novel technique of high order Newton's lifting to yield his record Las Vegas exponent $d = 3$ for the worst case input.

One can decrease the above Las Vegas exponents $d = 16/5$ to $d \approx 2.7$ and $d = 3$ to $d \approx 2.376$ by incorporating the asymptotically fast algorithms in [CW90] for $n \times n$ matrix multiplication, although this decrease is practically invalid due to the huge overhead constant hidden in the notation $\tilde{O}$.

For computing the determinants of (l, u) banded or more generally (l, u) rank structured matrices M (resp. matrices M given with their displacement generators of length r) in $\mathbb{Z}^{n\times n}$, one can obtain dramatic acceleration based on Theorems 2.6b, 2.8a, and 3.1. By combining these algorithms with the symmetrization $M \to M^T M$, we compute $\det M$ in $O(nw^2)$ for $w = l + u + 1$ (resp. $O(r^2 n \log^2 n)$) ops. We apply this computation in $\mathbb{Z}_{p_i}$ for n sufficiently large random primes p_i in $(n\gamma)^{O(1)}$ for $i = 1, \dots, n$ and expect to output the values $(\det M) \mod p_i$ for all i. Then we apply the Chinese remainder algorithm to combine these values to compute $\det M$ by using $O(w^2 n^2 \mu(\log n))$ (resp. $O(r^2 n^2 \log^2 n \mu(\log n))$ bit operations, which turns into the upper bound $\tilde{O}(n^2 \log n)$ for $w = O(1)$ (resp. $\tilde{O}(n^2 \log^3 n)$ for $r = O(1)$). Likewise using $O(s(n)^3 + |E(M)|\mu(\log n)$ bit operations is sufficient to compute $|\det M|$ for a symmetric positive definite matrix M in $\mathbb{Z}^{n\times n}$ provided its associated graph is given with an $s(n)$-separator family. In this case $\text{sign}(\det M)$ can be recovered from $|\det M|$ and the readily computable value $(\det M) \mod p$ for an appropriate smaller prime p (cf. [EGV00]).

10 Concluding remarks

Our lifting-based exact computations are nearly optimal and are unified for all most popular classes of structured matrices. Besides theoretical importance of such unification and optimization, the algorithms are valuable for some important classes of inputs for which numerical computations with double precision cannot produce output with the required accuracy [B85], [GI88], [T94].

Our combination of symbolic lifting with numerical iterative refinement in Section 7 can be viewed as an example of successful symbolic-numerical algorithms (cf. [TCS04], [SNC07], [SNC07a]). Searching for further examples of

this kind one can try to devise effective symbolic counterparts to various other iterative numerical algorithms for linear systems of equations (cf. [EPY98]).

11 Related works

Earlier papers [KKM79], [M74], [M80], [BA80] introduced and effectively employed Toeplitz-like displacement structure. The subsequent huge literature can be partly traced via the survey [KS99] and the books [HR84], [BP94], and [P01] (also see the bibliography in [GO94], [OP98], [P00]). Displacement transformation techniques implying Corollaries 2.3 and 3.1 are due to [P89/90]. Unification of the MBA numerical algorithm for various displacement structures was proposed in [OP98] (cf. also [P01, Chapter 5 and Section 4.6]). On the huge bibliography for rank structured (quasiseparable) and semiseparable matrices see [VVGM05]. The immense bibliography on banded matrices can be partly traced via [GL96, page 160]. On sparse matrices with small separator families see [LRT79], [GL81], [P93]. [MC79], [D82] were the first papers on Hensel's and Newton's lifting for solving integer linear systems of equations and integer matrix inversion. Nearly optimal exact solution of Toeplitz and Toeplitz-like linear system of equations based on the lifting and MBA algorithms was first sketched in the proceedings paper [PW02] with the focus on fast reconstruction of a rational number from its modular and numerical approximations. The papers [WP03] and [M04] cover the latter topic in some detail and include earlier bibliography. The proceedings paper [P02] and the report [PMRW05] supply more details on Toeplitz-like solving via lifting and introduced generalized lifting. The latter report and [PWa] study how to counter the degeneration in these algorithms. We are aware about no earlier studies of the Boolean complexity of the solution of structured linear systems, but in the case of general linear systems Hensel's lifting in [D82] supports the solution in $\tilde{O}(n^3 \log \gamma)$ bit-operations (versus the order of $n^4 \log \gamma$ in Gaussian elimination). This was improved to the record theoretical randomized Boolean cost in $\tilde{O}(n^{2.376} \log \gamma)$ by A. Storjohann in [S05] based on his high order Newton's lifting and on fast matrix multiplication in [CW90] (the latter tool implied a huge overhead constant).

Both MBA algorithm and lifting are not efficient where the structure of the input matrix is not readily extended to its inverse, which is the case for various sparse unstructured matrices as well as for the multivariate polynomial resultants. In this case the current best algorithms for a linear system of equations as well as determinant (under the Las Vegas randomized bit–operation complexity model) rely on the block Wiedemann algorithm. They reach the complexity bound $n^{2.5} \log \gamma$ up to polylogarithmic factors in n and $\log \gamma$ provided the input matrix (but not necessarily its preprocessed inverse) can be multiplied by a vector in $\mathbb{Z}_p$ in $\tilde{O}(n)$ bit-operations where $\log p = O(\log n)$ [EGG06], [EGG07] (cf. also [EP03/05]). Theoretically this is inferior to the cited bound $\tilde{O}(n^{2.376} \log \gamma)$ in [S05], but realistically is superior due to the immense overhead constant in the latter bound.

Finally we refer the reader to Section 9 and the beginning of Section 5 for

the bibliography on the Boolean complexity of computing determinants and on reconstructing the rational solution, respectively.

Appendix

A Multiplication of Toeplitz matrices and their inverses by vectors

Theorem A.1. *Multiplication of an $n \times n$ Toeplitz matrix T by a vector is a subproblem of multiplication of two polynomials of degrees $2n-2$ and $n-1$ whose coefficients are given by the entries of the input matrix and vector, respectively, that is $m_T \le m(3n-3)$ for m_T and $m(n)$ in Sections 2.1 and 2.2. If the Toeplitz matrix T is triangular and $m = n$, then both of these polynomials have degree $n-1$, that is in this case $m_T \le m(2n-2)$.*

Proof. See, e.g., [P01, pages 27–28]. □

The next theorem in [H79] (and also in [HR84]) extends the Gohberg–Semencul formula of 1972.

Theorem A.2. *Let $T = (t_{i,j})_{i,j=0}^{n-1}$ be a nonsingular Toeplitz matrix, let t_{-n} be any scalar (e.g., $t_{-n} = 0$), and write $t_{i-j} = t_{i,j}$ for $i, j = 0, \dots, n-1$; $p_n = -1$, $\mathbf{t} = (t_{i-n})_{i=0}^{n-1}$, $\mathbf{p} = (p_i)_{i=0}^{n-1} = T^{-1}\mathbf{t}$, $\mathbf{q} = (p_{n-i})_{i=0}^{n-1}$, $\mathbf{v} = T^{-1}\mathbf{e}_1$, $\mathbf{e_1}^T = (1, 0, \dots, 0)^T$, $\mathbf{u} = ZJ\mathbf{v}$. Then $T^{-1} = Z(\mathbf{p})Z^T(\mathbf{u}) - Z(\mathbf{v})Z^T(\mathbf{q})$.*

Hereafter the $n \times 2$ matrix $(\mathbf{v}, \mathbf{p})$ for the above vectors $\mathbf{v}$ and $\mathbf{p} = \mathbf{p}(T, t_{-n})$ is called a *generator* for T^{-1}. The next theorem is a corollary of Theorems A.1 and A.2.

Theorem A.3. *$i_T = m_{T^{-1}} \le 4m(2n-2) + n$ for i_S and m_S in Definition 2.5 and a nonsingular Toeplitz matrix T provided the matrix T^{-1} is given with its generator, that is, the vectors $\mathbf{v}$ and $\mathbf{p}$ in Theorem A.2.*

B Randomized recovery of the rational solution

By using the Las Vegas randomization, we decrease the bound in Theorem 5.1 by the factor of $\log d$ provided $\mu(d) = O(d^{\log_2 3})$ or $\mu(d) = O((d \log d) \log\log d)$ and $\rho(d)$ is bounded in (2.6). Empirical evidence shows further progress with some heuristics. Write

$$\delta = \mathrm{lcm}_j\, \delta(x_j), \quad 1 \le j \le n \tag{B.1}$$

(for $\delta(y)$ in Definition 2.24), that is δ is the least common multiple of the denominators in all rational coordinates x_j of the solution $\mathbf{x} = (x_j)_j$ to the system $M\mathbf{x} = q\mathbf{b}$.

Algorithm B.1. *Randomized recovery of the rational solution.*

INPUT: *The same as in Algorithm 4.1 and in addition a positive* $\varepsilon < 1$, *an integer* $h = 1 + \lfloor \log_s(2n(\alpha\sqrt{n})^{2n-1}\eta\beta) \rfloor$ *of equation (5.2), and the vector* $\mathbf{x}^{(h)} = (x_i^{(h)})_{i=1}^n = qM^{-1}\mathbf{f} \bmod (qs^h)$.

OUTPUT: *FAILURE with a probability of at most* ε *or a positive integer* δ *and an integer vector* $\mathbf{y}$ *such that*

$$M\mathbf{y} = \delta q\mathbf{f}. \tag{B.2}$$

INITIALIZATION: *Compute*

$$K = 2\lceil \log(1/\varepsilon) \rceil, \tag{B.3}$$

$$\eta = \lceil 6 + 2n \log(n\alpha) \rceil \tag{B.4}$$

for α *and* β *in (4.1). Then sample* K *pseudo random vectors*

$$\mathbf{c}_k = (c_{jk})_{j=1}^n \in \mathbb{Z}_\eta^n, \quad k = 1, \ldots, K. \tag{B.5}$$

COMPUTATIONS:

1. *Compute the* K *integers* $w_k = \mathbf{c}_k^T \mathbf{x}^{(h)} = \sum_{j=1}^n c_{jk} x_j^{(h)}, \quad k = 1, \ldots, K.$
2. *Recover a unique set of the pairs of coprime integers* ν_k *and* δ_k *such that*

$$(\nu_k/\delta_k) \bmod (qs^h) = w_k, \ 1 \le 2\delta_k|\nu_k| \le qs^h, \quad 2|\nu_k| < qs^h, \tag{B.6}$$

   for $k = 1, \ldots, K$.
3. *Compute the least common multiple of the denominators*

$$\delta_{lcd} = \operatorname{lcm}_k \delta_k, \quad 1 \le k \le K. \tag{B.7}$$

4. *Compute the integer vector* $\mathbf{y} = (y_j)_{j=1}^n$ *such that* $\mathbf{y} \bmod (qs^h) = \delta_{lcd}\mathbf{x}^{(h)}$ *and* $2|y_j| < qs^h$ *for all* j. *If* $M\mathbf{y} = q\delta_{lcd}\mathbf{f}$, *output* $\mathbf{y}$ *and* $\delta = \delta_{lcd}$; *otherwise output FAILURE.*

Combining equations (5.2), (B.4), and (B.5) with Fact 2.2 implies (B.6). Now, correctness of Algorithm B.1 is implied by the following simple result.

Theorem B.1. δ_{lcd} *in (B.7) divides* δ *in (B.1). Furthermore,*

$$\text{Probability}(\delta_{lcd} \ne \delta) \le \varepsilon.$$

Theorem B.1 is deduced similarly to Theorem 2.1 in [EGV00] based on equations (5.2), (B.3)–(B.7), and the following lemma.

Lemma B.1. *For a prime* p, *integers* K *in (B.3),* k *such that* $1 \le k \le K$, δ *in (B.1),* η *in (B.4), and* δ_k *in (B.6), we have* $\text{Probability}(\text{ord}_p(\delta_k) < \text{ord}_p(\delta))$ *equal to* $1/\eta$ *for* $p \ge \eta$ *and to* $\lfloor \eta/p \rfloor/\eta \le 1/p$ *for* $p \le \eta$.

Proof. Let $l = \mathrm{ord}_p(\delta) = \max_j \mathrm{ord}_p(\delta(x_j))$ for $1 \le j \le n$. W.l.o.g., let $l = \mathrm{ord}_p(\delta(x_1))$ and let c denote the first coordinate of the vector $\mathbf{c} = \mathbf{c}_k$. Then we have

$$\mathbf{c}^T\mathbf{x} = \frac{cu}{ap^l} - \frac{v}{p^h b} = \frac{cub - avp^{l-h}}{abp^l}$$

where $\mathbf{x} = M^{-1}\mathbf{f}$, $l \ge h$, and a, b, u, and v are four integers coprime with p. Clearly, $\mathrm{ord}_p(\delta_k)$ for δ_k in (B.6) never exceeds l; it equals l if and only if $cub - avp^{l-h}$ is coprime with p. Since ub is coprime with p and since c is random in $\mathbb{Z}_\eta$, the probability bound follows. □

B.1 The bit-complexity of randomized reconstruction of rational solution

Let us first estimate the bit complexity of performing Algorithm B.1 in terms of $d = O(n \log\gamma + \log q)$ in (5.1), m_S in Definition 2.5, $\mu(d)$ in (2.4), $\rho(d)$ in (2.6), and K in (B.3). We need the following auxiliary result.

Lemma B.2. *Let j and k be positive integer parameters, $j \to \infty$. Then $O(\mu(j)k)$ bit operations are sufficient to multiply two positive integers u and v such that $u < 2^j$ and $v < 2^{j+k}$.*

Proof. Represent v as $\sum_{i=0}^{k-1} v_i 2^{ij}, 0 \le v_i < 2^j$ for all i. Compute the products $w_i = uv_i$ for $i = 0, 1, \ldots, k-1$. This takes $O(\mu(j)k)$ bit operations. Now compute the sum $uv = \sum_{i=0}^{k-1} w_i 2^{ij}$. This takes $O(jk)$ bit operations. □

Algorithm B.1 involves $O(Kn\mu(d))$ bit operations at Stage 1; $O(K\rho(d))$ at Stage 2; $O(K\mu(d)\log d)$ at Stage 3, and $O(n\mu(d))$, $O(n\mu(\log\beta)d/\log\beta)$, and $O(m_M\mu(\log\gamma)d/\log\gamma)$ for computing the vectors $\delta_{lcd}\mathbf{x}^{(h)}$, $q\delta_{lcd}\mathbf{f}$, and $M\mathbf{y}$ at Stage 4, respectively. (The two latter bounds are deduced based on Lemma B.2.) Summarizing, we obtain the following estimates.

Theorem B.2. *Algorithm B.1 generates nK random elements in $\mathbb{Z}_\eta$ for η in (B.4) and $K = 2\lceil\log(1/\epsilon)\rceil$ in (B.3). It either fails (this occurs with a probability of at most ϵ) or computes the scalar δ of equation (B.1) and the solution $\mathbf{y}$ to linear system (B.2). The algorithm involves*

$$B_1 = O(Kn\mu(d) + K\rho(d) + m_M\mu(\log\gamma)d/\log\gamma)$$

bit operations for $d = O(n\log\gamma + \log q)$ in (5.1), $\rho(d)$ in (2.6), γ in (4.1), m_S in Definition 2.5, and $\mu(d)$ in (2.4); it involves $o(B_1)$ bit operations for generating nK pseudo random elements in $\mathbb{Z}_\eta$.

C Detailed computational complexity estimates

Theorem C.1. *Assume a prime p, a vector $\mathbf{f} \in \mathbb{Z}^n$, and a nonsingular matrix $M \in \mathbb{Z}^{n\times n}$ with the structure of Toeplitz, Hankel, Vandermonde, or Cauchy*

type, having a displacement rank r and given with a displacement generator of a length $l = O(r)$, and write $Q = M^{-1} \mod p$. Then we can compute the rational solution $\mathbf{x}$ to the linear system $M\mathbf{x} = \mathbf{f}$ by using single random parameter p and at the Las Vegas randomized bit–operation cost within the following bounds:

i) $O(r^2 m(n)\mu(\log p)\log n)$ *at the initialization stage,*

ii) $O((m_Q\mu(\log p) + m_M\mu(\log(\gamma np)))h)$ *at the lifting stage;*

iii) $O(n\rho(d))$ *at the stage of Las Vegas randomized reconstruction of rational solution, which involves* $n\lceil\log\frac{1}{\varepsilon}\rceil\lceil\log(6 + 2n\log(n\alpha))\rceil$ *random bits and may fail with a probability of at most* $\varepsilon > 0$.

Here $m(n)$, $\mu(d)$ and $\rho(d)$ are defined in (2.3)–(2.6), m_S in Definition 2.5, γ and α in (4.1), $d = O(n\log\gamma)$ in (5.1) for $q = 1$, and $h = O(n\log_p(\gamma n))$ in (5.2) for $s = p$.

Proof. See [PMRW05]. □

References

[ABM99] J. Abbott, M. Bronstein, T. Mulders. Fast Deterministic Computation of the Determinants of Dense Matrices, *Proc. Intern. Symp. Symbolic and Algebraic Comput. (ISSAC'99)*, 197–204, ACM Press, New York, 1999.

[B85] J. R. Bunch, Stability of Methods for Solving Toeplitz Systems of Equations, *SIAM J. Sci. Stat. Comput.*, **6(2)**, 349–364, 1985.

[B03] D. J. Bernstein, Fast Multiplication and Its Applications, to be printed in *Algorithmic Number Theory* (edited by Joe Buhler, Peter Stevenhagen), **44**, Mathematical Sciences Research Institute Publications, Cambridge University Press. Available from http://cr.yp.to/papers.html

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

[BEPP99] H. Brönnimann, I. Z. Emiris, V. Y. Pan, S. Pion, Sign Determination in Residue Number Systems, *Theoretical Computer Science*, **210(1)**, 173–197, 1999.

[BGP03/05] D. A. Bini, L. Gemignani, V. Y. Pan, Fast and Stable QR Eigenvalue Algorithms for Generalized Companion Matrices and Secular Equation, *Numerische Math.*, **3**, 373–408, 2005. (Also Tech. Report 1470, *Dept. Math., Univ. Pisa*, Italy, July 2003.)

[BGY80] R. P. Brent, F. G. Gustavson, D. Y. Y. Yun, Fast Solution of Toeplitz Systems of Equations and Computation of Padé Approximations, *J. Algorithms*, **1**, 259–295, 1980.

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

[C92] K. L. Clarkson, Safe and Effective Determinant Evaluation, *Proc. 33rd Ann. IEEE Symp. on Foundations of Computer Science*, 387–395, IEEE Computer Soc. Press, Los Alamitos, CA, 1992.

[C94] D. Coppersmith, Solving Homogeneous Linear Equations over *GF(2)* via Block Wiedemann Algorithm, *Math. of Computation*, **62(205)**, 333–350, 1994.

[CFG99] G. Cooperman, S. Feisel, J. von zur Gathen, G. Havas, GCD of Many Integers, *Computing and Combinatorics, Lecture Notes in Computer Science*, **1627**, 310–317, Springer, Berlin, 1999.

[CK91] D. G. Cantor, E. Kaltofen, On Fast Multiplication of Polynomials over Arbitrary Rings, *Acta Informatica*, **28(7)**, 697–701, 1991.

[CW90] D. Coppersmith, S. Winograd, Matrix Multiplicaton via Arithmetic Progressions. *J. Symbolic Comput.*, **9(3)**, 251–280, 1990.

[D59] J. Durbin, The Fitting of Time-Series Models, *Review of International Statistical Institute,* **28**, 229–249, 1959.

[D82] J. D. Dixon, Exact Solution of Linear Equations Using p-adic Expansions, *Numerische Math.*, **40**, 137–141, 1982.

[DGP04] J.-G. Dumas, P. Giorgi, C. Pernet, FFPack: Finite Field Linear Algebra Package, *Proc. Intern. Symp. on Algebraic and Symbolic Comput. (ISSAC'04)*, 119–126, ACM Press, New York, 2004.

[EG99] Y. Eidelman, I. Gohberg, Linear Comlexity Inversion Algorithms for a Class of Structured Matrices, *Integral Equations and Operator Theory*, **35**, 28–52, Birkhäuser, Basel, 1999.

[EG01] Y. Eidelman, I. Gohberg, Fast Inversion Algorithms for a Class of Block Structured Matrices, *Contemp. Math.*, **281**, 17–38, 2001.

[EG02] Y. Eidelman, I. Gohberg, A Modification of the Dewilde–Van der Veen Method for Inversion Finite Structured Matrices, *Linear Algebra and Its Applications*, **343–344**, 419–450, 2002.

[EGG06] W. Eberly, M. Giesbrecht, P. Giorgi, A. Storjohann, G. Villard, Solving Sparse Linear systems, *Proc. Intern. Symp. Symb. Algebraic Comput. (ISSAC'06)*, 63-70, ACM Press, New York, 2006.

[EGG07] W. Eberly, M. Giesbrecht, P. Giorgi, A. Storjohann, G. Villard, Faster Inversion and other Black Box Matrix Computation Using Efficient Block Projections, *Proc. Intern. Symp. Symb. Algebraic Comput. (ISSAC'07)*, 143-150, ACM Press, New York, 2007.

[EGV00] W. Eberly, M. Giesbrecht, G. Villard, On Computing the Determinant and Smith Form of an Integer Matrix, *Proc. 41st Annual Symp. on Foundations of Computer Science (FOCS'2000)*, 675–685, IEEE Computer Society Press, Los Alamitos, CA, 2000.

[EP03/05] I. Z. Emiris, V. Y. Pan, Improved Algorithms for Computing Determinants and Resultants, *J. of Complexity*, **21(1)**, 43–71, 2005. Proceedings version in *Proc. of the 6th International Workshop on Computer Algebra in Scientific Computing (CASC '03)*, edited by E. W. Mayr, V. G. Ganzha, and E. V. Vorozhtzov, 81–94, Technische Univ. München, Germany, 2003.

[EPY98] I. Z. Emiris, V. Y. Pan, Y. Yu, Modular Arithmetic for Linear Algebra Computations in the Real Field, *J. of Symbolic Computation*, **21**, 1–17, 1998.

[GG03] J. von zur Gathen, J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, Cambridge, UK, 2003 (second edition).

[GI88] W. Gautschi, G. Inglese, Lower Bounds for the Condition Number of Vandermonde Matrices, *Numerische Math.*, **52**, 241–250, 1988.

[GL81] A. George, J. W.–H. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice–Hall, Englewood Cliffs, NJ, 1981.

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

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

[GS92] J. R. Gilbert, R. Schreiber, Highly Parallel Sparse Cholesky Factorization, *SIAM J. Scientific Computing*, **13**, 1151–1172, 1992.

[H79] G. Heinig, Beitrage zur spektraltheorie von Operatorbuschen und zur algebraischen Theorie von Toeplitzmatrizen, Dissertation **B**, *TH Karl-Marx-Stadt*, 1979.

[HR84] G. Heinig, K. Rost, *Algebraic Methods for Toeplitz-like Matrices and Operators, Operator Theory*, **13**, Birkhäuser, 1984.

[K98] D. E. Knuth, *The Art of Computer Programming*, **2**: *Seminumerical Algorithms*, Addison-Wesley, Reading, Massachusetts, 1998.

[K04] I. Kaporin, The Aggregation and Cancellation Techniques As a Practical Tool for Faster Matrix Multiplication, *Theoretical Computer Science*, **315**, 469–510, 2004.

[KKM79] T. Kailath, S. Y. Kung, M. Morf, Displacement Ranks of Matrices and Linear Equations, *Journal Math. Analysis and Appls*, **68(2)**, 395–407, 1979.

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

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

[KV01] E. Kaltofen, G. Villard, On the Complexity of Computing Determinants, *Proc. 5th Asian Symp. Computer Math. (ASCM 2001)*, (K. Shirayanagi and K. Yokoyama, editors), *Lecture Notes Series on Comput.*, **9**, 13–27, World Scientific, Singapore, 2001.

[KV04] E. Kaltofen, G. Villard, Computing the Sign or the Value of the Determinant of an Integer Matrix, a Complexity Survey, *J. Computational Applied Math.*, **162(1)**, 133–146, 2004.

[L47] N. Levinson, The Wiener RMS (Root-Mean-Square) Error Criterion in the Filter Design and Prediction, *Journal of Mathematical Physics*, **25**, 261–278, 1947.

[LRT79] R. J. Lipton, D. Rose, R. E. Tarjan, Generalized Nested Dissection, *SIAM J. on Numerical Analysis*, **16(2)**, 346–358, 1979.

[M74] M. Morf, Fast Algorithms for Multivariable Systems, Ph.D. Thesis, *Dept. Electrical Engineering, Stanford Univ.,* CA, 1974.

[M80] M. Morf, Doubling Algorithms for Toeplitz and Related Equations, *Proceedings of IEEE International Conference on ASSP*, 954–959, IEEE Press, Piscataway, New Jersey, 1980.

[M04] M. Monagan, Maximal Quotient Rational Reconstruction: an Almost Optimal Algorithm for Rational Reconstruction, *Proc. Intern. Symp. Algebraic and Symb. Comp. (ISSAC'04)*, 243–249, ACM Press, New York, 2004.

[MC79] R. T. Moenck, J. H. Carter, Approximate Algorithms to Derive Exact Solutions to Systems of Linear Equations, *Proceedings of EUROSAM, Lecture Notes in Computer Science*, **72**, 63–73, Springer, Berlin, 1979.

[MS04] T. Mulders, A. Storjohann, Certified Dense Linear System Solving, *J. of Symbolic Computation*, **37(4)**, 485-510, 2004.

[N72] M. Newman, *Integral Matrices*, Academic Press, New York, 1972.

[OP98] V. Olshevsky, V. Y. Pan, A Unified Superfast Algorithm for Boundary Rational Tangential Interpolation Problem and for Inversion and Factorization of Dense Structured Matrices, *Proc. 39th Ann. IEEE Symp. Foundation of Comp. Science (FOCS 98)*, 192–201, IEEE Computer Soc. Press, Los Alamitos, CA, 1998.

[P84] V. Y. Pan, How Can We Speed up Matrix Multiplication?, *SIAM Review*, **26(3)**, 393–415, 1984.

[P87] V. Y. Pan, Complexity of Parallel Matrix Computations, *Theoretical Computer Science*, **54**, 65–85, 1987.

[P88] V. Y. Pan, Computing the Determinant and the Characteristic Polynomials of a Matrix via Solving Linear Systems of Equations, *Information Processing Letters*, **28**, 71–75, 1988.

[P89/90] V. Y. Pan, On Computations with Dense Structured Matrices, *Math. of Computation*, **55(191)**, 179–190, 1990. Proceedings version in *Proc. ISSAC89*, 34-42, ACM Press, New York, 1989.

[P93] V. Y. Pan, Parallel Solution of Sparse Linear and Path Systems, in *Synthesis of Parallel Algorithms* (J.H. Reif, editor), Chapter 14, pp. 621–678, Morgan Kaufmann publishers, San Mateo, California, 1993.

[P96] V. Y. Pan, Parallel Computation of Polynomial GCD and Some Related Parallel Computations over Abstract Fields, *Theoretical Computer Science*, **162(2)**, 173–223, 1996.

[P00] V. Y. Pan, Parallel Complexity of Computations with General and Toeplitz-like Matrices Filled with Integers and Extensions, *SIAM J. Comput.*, **30(4)**, 1080–1125, 2000.

[P01] V. Y. Pan, *Structured Matrices and Polynomials: Unified Superfast Algorithms*, Birkhäuser/Springer, Boston/New York, 2001.

[P02] V. Y. Pan, Randomized Acceleration of Fundamental Matrix Computations, *Proc. Symp. on Theoretical Aspects of Computer Science (STACS)*, *Lect. Notes in Comput. Sci.*, **2285**, 215–226, Springer, Heidelberg, Germany, 2002.

[P04a] V. Y. Pan, On Theoretical and Practical Acceleration of Randomized Computation of the Determinant of an Integer Matrix, *Zapiski Nauchnykh Seminarov POMI* (in English), **316**, 163–187, St. Petersburg, Russia, 2004.

[PMRW05] V. Y. Pan, B. Murphy, R. E. Rosholt, X. Wang, Toeplitz and Hankel Meet Hensel and Newton Modulo a Power of Two, Technical Report 2005008, *PhD Program in Computer Science, The Graduate Center, CUNY*, New York, 2005.

[PR93] V. Y. Pan, J. Reif, Fast and Efficient Parallel Solution of Sparse Linear Systems, *SIAM J. on Computing*, **22(6)**, 1227–1250, 1993.

[PW02] V. Y. Pan, X. Wang, Acceleration of Euclidean Algorithm and Extensions, *Proc. Intern. Symp. Symbolic and Algebraic Computation (ISSAC'02)*, 207–213, ACM Press, New York, 2002.

[PW03] V. Y. Pan, X. Wang, Inversion of Displacement Operators, *SIAM J. on Matrix Analysis and Applications*, **24**, **3**, 660–677, 2003.

[PW04] V. Y. Pan, X. Wang, On Rational Number Reconstruction and Approximation, *SIAM J. on Computing*, **33(2)**, 502–503, 2004.

[PWa] V. Y. Pan, X. Wang, Degeneration of Integer Matrices Modulo an Integer, preprint, 2006 (submitted).

[PY01] V. Y. Pan, Y. Yu, Certification of Numerical Computation of the Sign of the Determinant of a Matrix, *Algorithmica*, **30**, 708–729, 2001.

[S97] J. Shevchuk, Adaptive Precision Floating–Point Arithmetic and Fast Robust Geometric Predicates, *Discrete and Computational Geometry*, **18**, 305–363, 1997. Available at www.cs.cmu.edu/ quake/robust.html

[S05] A. Storjohann, The Shifted Number System for Fast Linear Algebra on Integer Matrices, *J. of Complexity*, **21(4)**, 609–650, 2005.

[SNC07] *Symbolic-Numeric Computation*, (Dongming Wang and Li-Hong Zhi, editors), Birkhäuser, Basel/Boston, 2007.

[SNC07a] *Proceedings of the 3rd Intern. Workshop on Symbolic-Numeric Computation (SNC2007)*, July 2007, London, Ontario, Canada (J. Verschelde and S. Watt, eds.), ACM Press, New York, 2007.

[T64] W. F. Trench, An Algorithm for Inversion of Finite Toeplitz Matrices, *J. of SIAM*, **12**, 515–4522, 1964.

[T94] E. E. Tyrtyshnikov, How Bad Are Hankel Matrices? *Numerische Mathematik*, **67(2)**, 261–269, 1994.

[TCS04] *Theoretical Computer Science*, Special Issue on Algebraic and Numerical Algorithms (I. Z. Emiris, B. Mourrain, V. Y. Pan, editors), **315**, **2**–**3**, 307–672, 2004.

[TCS08] *Theoretical Computer Science*, Special Issue on Symbolic–Numerical Algorithms (D. A. Bini, V. Y. Pan, and J. Verschelde, editors), (in press).

[VVGM05] R. Vandebril, M. Van Barel, G. Golub, N. Mastronardi, A Bibliography on Semiseparable Matrices, *Calcolo*, **42(3–4)**, 249–270, 2005.

[W86] D. Wiedemann, Solving Sparse Linear Equations over Finite Fields, *IEEE Trans. Inf. Theory*, **IT-32**, 54–62, 1986.

[WP03] X. Wang, V. Y. Pan, Acceleration of Euclidean Algorithm and Rational Number Reconstruction, *SIAM J. on Computing*, **32(2)**, 548–556, 2003.