City University of New York (CUNY)
CUNY Academic Works

Computer Science Technical Reports | CUNY Academic Works

2004

# TR-2004013: Toeplitz and Hankel Meet Hensel and Newton: Nearly Optimal Algorithms and Their Practical Acceleration with Saturated Initialization

Victor Y. Pan

Brian Murphy

Rhys E. Rosholt

Xinmao Wang

How does access to this work benefit you? Let us know!

More information about this work at: https://academicworks.cuny.edu/gc_cs_tr/249
Discover additional works at: https://academicworks.cuny.edu

This work is made publicly available by the City University of New York (CUNY).
Contact: AcademicWorks@cuny.edu

# Toeplitz and Hankel Meet Hensel and Newton: Nearly Optimal Algorithms and Their Practical Acceleration with Saturated Initialization *†

Victor Y. Pan, Brian Murphy, Rhys E. Rosholt
Department of Mathematics and Computer Science
Lehman College of CUNY, Bronx, NY 10468, USA
vpan@lehman.cuny.edu
bmurphy@lehman.cuny.edu
rosholt@lehman.cuny.edu
and
Xinmao Wang
Ph.D. Program in Mathematics
Graduate School of CUNY, New York, NY 10016, USA
xwang2@gc.cuny.edu

September 13, 2004

**Abstract**

The classical and intensively studied problem of solving a Toeplitz/ Hankel linear system of equations is omnipresent in computations in sciences, engineering and signal processing. By assuming a nonsingular integer input matrix and relying on Hensel's lifting, we compute the solution faster than with the divide-and-conquer algorithm by Morf 1974/1980 and Bitmead and Anderson 1980 and nearly reach the information lower bound on the bit operation complexity of the solution. Furthermore, we extend lifting to the rings of integers modulo nonprimes, e.g., modulo $2^w$. This allows significant saving of the word operations. We also extend our algorithms and complexity estimates to computations with singular

*The 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 65393–0034 and 66437–0035

Toeplitz/Hankel and Toeplitz/Hankel-like matrices and further to computing the greatest common divisors, least common multiples, resultants, Padé approximations and rational interpolation functions for univariate polynomials.

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

**Key Words:** Toeplitz matrices, Hankel matrices, Solving linear systems, Hensel's lifting, Computations modulo a power of two.

# 1 Introduction

## 1.1 Toeplitz/Hankel computations

Toeplitz, Hankel, and more generally Toeplitz/Hankel-like linear systems of equations are omnipresent in computations in sciences, engineering, and signal and image processing. Solution of such linear systems is required in the shift register synthesis and linear recurrence computation, inverse scattering, adaptive filtering, modelling of stationary and nonstationary processes, numerical computations for Markov chains, solution of PDE's and integral equations, and polynomial root-finding. See the bibliography in Kailath and Sayed (eds.) 1999 [KS99], Pan 2000 [P00, Section 1.1] and Pan 2001 [P01]. The displacement transformations in Pan 1990 [P90] reduce the solution of structured linear systems of the Cauchy and Vandermonde types to the Toeplitz/Hankel-like case and vice versa. Moreover (see Brent et al. 1980 [BGY80], [P96], [P01]), the solution of Toeplitz/Hankel linear systems has been extended to the computation of the polynomial greatest common divisors and least common multiples (hereafter we use the abbreviations gcds and lcms) as well as Padé approximations, rational interpolation functions, and the resultants of univariate polynomials. These problems are central and most intensively studied in computer algebra (see von zur Gathen and Gerhard 2003 [GG03]).

By exploiting the matrix structure one may dramatically decrease the solution cost, from the order of $n^3$ arithmetic operations in Gaussian elimination for a nonsingular Toeplitz/Hankel system of $n$ equations $M\mathbf{x} = \mathbf{b}$ to $O(n^2)$ in the "fast algorithms" (Levinson 1947, Durbin 1959, and Trench 1964) and further, with FFT, to $O(n \log^2 n)$ in the "superfast algorithms" (in [BGY80] and the MBA algorithm by Morf 1974/1980 [M74], [M80] and Bitmead/Anderson 1980 [BA80]). See [KS99], [P01], Pan 2004 [P04], and the bibliography therein on these and other fast and superfast algorithms.

## 1.2 Bounding the precision of computing, CRA and rational number reconstruction

Numerical stability is the Achilles' heel of the Toeplitz/Hankel superfast algorithms (see Bunch 1985 [B85]). Furthermore, all positive definite Hankel

matrices are ill conditioned (see Tyrtyshnikov 1994 [T94]). These observations suggest applying the algebraic computation techniques to simultaneously bound the arithmetic cost and the precision of computing. The most popular approach is to compute the rational solution modulo sufficiently many basic primes of moderate size and to reconstruct it by applying the CRA (*Chinese remainder algorithm*) and the CFAA (continued fraction approximation algorithm). The final stage is the *rational number reconstruction* with the CFAA (see Sections 2.3, 4, and 5). This stage turns into the integer number reconstruction and thus is simplified for the MBA algorithm, which computes the determinant $\det M$ as by-product (see Remark 4.1).

### 1.3 Hensel's lifting versus the MBA algorithm

We rely on Hensel's lifting proposed for solving general linear systems in Moenck and Carter 1979 [MC79] and Dixon 1982 [D82]. As an advantage versus the MBA/CRA approach, we only need a single basic modulus $q$, a power of a random or fixed prime, e.g., $q = 2^w$. Degeneracy occurs only if the input matrix $M$ is singular modulo $q$, whereas the MBA algorithm faces the degeneracy problem whenever the matrix $M$ or any of its leading principal submatrices (that is, northwestern blocks) is singular modulo any of the basic primes. Moreover, lifting is essentially reduced to recursive multiplication by vectors of two matrices $M$ and $Q = M^{-1} \bmod q$, which can in turn be reduced to polynomial multiplication. The MBA algorithm is more involved: it uses divisions and processes various auxiliary matrices of smaller sizes, which means a larger working memory and a more complicated code. The computation time also favors lifting, both in terms of the number of bit operations involved (favors slightly) and under the word operation model [GG03, Section 2.1] (favors significantly). In both cases of lifting and the MBA algorithm we counter degeneration by means of randomization and other special techniques (see our Section 12 and [P04]).

### 1.4 Computational complexity estimates

We estimate both arithmetic complexity and precision of computing at the lifting stage, and this enables us to bound the bit operation and word operation complexity as well. For the final stage of the recovery of the rational solution, we only estimate the bit operation complexity.

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

$$m(n) \leq c_{class} n^2, \quad m(n) \leq c_k n^{\log 3}, \quad m(n) \leq (c_{ck} n \log n) \log\log n \tag{1.1}$$

over any field or ring with unity. Here and hereafter log stands for $\log_2$ unless we specify otherwise, so that $\log 3 = 1.58496\ldots$; $c_{class}, c_k$, 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 Bernstein 2003 [B03], [GG03]).

To each arithmetic operation performed over the integers modulo $q$, that is, with the $d$-bit precision for $d = \lceil \log q \rceil$, we assign the cost of $\mu(d)$ bit operations, where $\mu(d) \geq 2d - 2$ (an information lower bound),

$$\mu(d) \leq C_{class} d^2, \quad \mu(d) \leq C_k d^{\log 3}, \quad \mu(d) \leq (C_{ss} d \log d) \log\log d, \tag{1.2}$$

$C_{class}, C_k$, 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 [B03], [GG03]).

To simplify our exposition of the bit complexity estimates in this section, we assume that all input values are integers in $n^{O(1)}$. (In the paper we detail the estimates without this assumption, except for their display in Table 7.1.) Then the determinants of the $n \times n$ matrices formed by these values are $s$-bit integers where $s$ is typically of the order of $n \log n$ (see, e.g. Abbott et al. 1999 [ABM99]). The $n$ output values are the ratios of such determinants. Therefore, the order of $n^2 \log n$ bits is required for their representation, and at least the order of $n^2 \log n$ bit operations for their computation. The lifting approach nearly reaches this lower bound by using

$$B = O(n\ m(n)\mu(\log n)) \tag{1.3}$$

bit operations (see Theorem 8.1), and in particular $O(nm(n))$ arithmetic operations with $O(\log n)$ bit precision at the lifting stage. The MBA/CRA approach uses the order of $m(n) \log n$ field operations for each basic prime. It requires the order of $n$ primes, each having the length of the order of $\log n$ bits. This means the extra factor of $\log n$ versus (1.3).

We consider two implementations of our lifting algorithm. In one case we use a random basic prime. Its bad choice may cause failure of our algorithm, but bad primes occur with a small bounded probability. Furthermore, our complexity bounds cover the verification of the correctness of the output, that is, our randomized bound (1.3) is of the so called *Las Vegas* type (where we allow no undetected errors). In another implementation, we fix a basic prime for lifting and then again either compute and certify the rational solution or output FAILURE. We also estimate the fraction of all inputs for which the algorithm fails, and if it fails, we show a heuristic recipe for fixing it. According to our analysis of this implementation, the bound (1.3) holds for solving linear systems with the average integer Toeplitz/Hankel input matrix, whereas for the worst case input we need to increase this upper estimate by roughly the factor of $n$.

## 1.5 Lifting in the rings of integers and some technical issues

To save lifting steps and word operations, we initialize lifting with the matrix $M^{-1}$ modulo $q = p^w$ where $p$ is a prime, $w = \lfloor \lambda / \log p \rfloor$, and $\lambda$ is the length of a computer word. We call this policy *saturated initialization*.

Practically, it is more attractive to operate modulo a power of a smaller prime $p$, e.g., $p = 2$. Technically, this means lifting in the rings rather than fields. We could not find any bibliography for this subject, even for lifting for the general linear systems.

In fact, in spite of its simplicity and efficiency, Hensel's lifting has not been studied and even proposed for Toeplitz linear systems, even with a random basic prime $p$. So we had to handle various technicalities such as the optimization of the lifting and initialization parameters, the choice of stopping criteria, practical heuristic recipes for avoiding degeneration, and theoretical and experimental estimates for its likelihood. Addressing these issues has led us to some new techniques and concepts of independent interests such as the concept of factor nonsingularity for the extension of Hensel's lifting to the rings $\mathbb{Z}_q$ where $\det M$ is not necessarily coprime with $q$ (this may be of interest for solving general linear systems as well), the variable diagonal and modular continuation techniques for the initialization of lifting, and random perturbation of the input by adding random matrices of small rank (for avoiding singularities in $\mathbb{Z}_q$). We also supply some probabilistic estimates for degeneracy as well as the results of statistical tests. Various other technicalities were handled when the algorithms were implemented at the Lehman College of the City University of New York by the second and the third authors.

## 1.6 Further extensions

We present lifting for Toeplitz linear systems, but all we need is fast multiplication of the input matrix and its precomputed inverse by a vector. Thus our algorithms apply also to Hankel and, more generally, Toeplitz/Hankel-like linear systems; the cited displacement transformations in [P90] allow further extensions to Cauchy-like, Vandermonde-like, and other structured linear systems of equations. Sparse and structured linear systems is another potential application area.

Our algorithm can be extended to the cases where the input matrices $M$ are nonsingular block matrices with integer blocks or polynomial matrices such that both $M$ and $M^{-1}$ are multiplied by vectors fast.

Furthermore the algorithm works where the input is made up of complex (Gaussian) integers. Scaling enables further extension to allow rational input and, therefore, to allow truncated real and complex input values, although this may lead to the undesirable increase of the magnitude of the input values.

In Section 12 and with more details in [P04], we extend our approach to computing the matrix determinant, rank and a vector from the null space of and to solving a consistent singular linear system. This in turn implies further extensions to the computation of the gcds, lcms, resultants, and Padé approximations as well as rational interpolation functions for univariate polynomials with the coefficients from the above domains. For computing the matrix rank, a basis for the null space, the gcds, lcms, Padé approximations and rational interpolation functions for univariate input polynomials as well as deciding the consistency of a linear system, we only have the so called *Monte Carlo* randomized complexity

estimates, which do not cover the verification of the correctness of the output, that is, in these cases we allow undetected output errors although with a small controlled probability. In particular our output value of the matrix rank never exceeds the actual rank but can be less than it with a small controlled probability. For solving consistent linear systems and computing matrix determinants, univariate polynomial resultants, and vectors from the null spaces of singular matrices, our complexity estimates are still of the Las Vegas type because we may certify the output at a lower cost.

### 1.7 Organization of our paper

We state the computation of some basic definitions and auxiliary results in the next section. We cover Hensel's lifting and Newton's algorithms for linear equations and matrix inversion in the rings $\mathbb{Z}_q$ in Section 3. In Sections 4 and 5, we recover the rational solution from its truncated $q$-adic extension. In Section 6, we estimate the computational complexity of our lifting algorithm. In Section 7 and Theorem 8.1 in Section 8, we summarize our bit operation complexity estimates for both lifting and recovery. We initialize lifting in Sections 8 and 9. In Section 10, we study the degeneration problem; in Section 10.4 we present the results of our experiments on how frequently random integer Toeplitz and general matrices are singular modulo $2^g$. In Section 11, we demonstrate our algorithms with some simple examples. In Section 12 we comment on the extensions of our study. Section 10.4 is due to the fourth author, the implementation of the algorithms to the second and third authors, and all other parts of the paper to the first author.

**Acknowledgements.** Our thanks go to Mark Giesbrecht and Arne Storjohann for the (p)reprints of the papers Eberly et al. 2000 [EGV00], Mulders and Storjohann 2004 [MS04], and Storjohann 2003 [S03], and to Richard Isaac for suggesting a format for the statistical tests reported in Section 10.4.

## 2 Definitions and basic facts

### 2.1 General matrices

**Definition 2.1.** *$\mathbb{Z}$ is the ring of integers. $\mathbb{Q}$ is the field of rational numbers. $M = (m_{i,j})_{i,j=1}^{k,l}$ is a $k \times l$ matrix with rational or integer entries $m_{i,j}$; $M \in \mathbb{Q}^{k\times l}$ or $M \in \mathbb{Z}^{k\times l}$, respectively. $\mathbf{v} = (v_i)_{i=1}^{k}$ is a vector. $I$ is the identity matrix of a proper size. $I_l$ is the $l \times l$ identity matrix. $M^T$ is the transpose of $M$.*

**Definition 2.2.** $\det M$ *and* $\operatorname{adj} M = (d_{i,j})_{i,j=1}^{k}$ *denote the determinant and the adjoint (adjugate) of a $k \times k$ matrix $M = (m_{i,j})_{i,j=0}^{k-1,k-1}$ where $d_{i,j} = \det M_{i,j}$ and the submatrix $M_{i,j}$ is obtained by deleting the $i$-th row and the $j$-th column of $M$.* $\operatorname{adj} M = M^{-1} \det M$ *if $M$ is nonsingular.*

**Definition 2.3.** *$|M|$ denotes the column norm of a matrix $M = (m_{i,j})_{i,j}$, $|M| = ||M||_1 = \max_j \sum_i |m_{i,j}|$; $|\mathbf{v}| = \sum_i |v_i|$ denotes the $\ell_1$-norm of a vector $\mathbf{v} = (v_i)_i$; $\alpha(M) = \max_{i,j} |m_{i,j}|$, $\beta(\mathbf{v}) = \max_i |v_i|$.*

**Definition 2.4.** *$v_S \le 2n^2 - n$ and $i_S$ are the minimum numbers of arithmetic operations sufficient to multiply a given $n \times n$ matrix $S$ by a vector and to invert it, respectively.*

**Definition 2.5.** *$d_k = d_k(M)$, the $k$-th determinantal divisor of a matrix $M \in \mathbb{Z}^{n\times n}$ for $k = 1, \dots, n$, is the greatest common divisor (gcd) of all its $k \times k$ minors (subdeterminants). We write $s_0 = d_0 = 1$ and define $s_k = s_k(M) = d_k/d_{k-1}$, the $k$-th* Smith invariant factors *of $M$ for $k = 1, \dots, n$.*

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

**Fact 2.1.** *$|\det M| \le \prod_j (\Sigma_i m_{i,j}^2)^{1/2} \le (\alpha(M)\sqrt{n})^n$, $|\det M| \le |M|^n$, $|\operatorname{adj} M| \le (n-1)\alpha(\operatorname{adj} M)$, and so $|\operatorname{adj} M| \le (\alpha(M)\sqrt{n-1})^{n-1}(n-1)$, $|\operatorname{adj} M| \le (n-1)|M|^{n-1}$ for an $n \times n$ matrix $M = (m_{i,j})_{i,j}$.*

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

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

We use the definitions of $m(n)$ and $\mu(n)$ in (1.1) and (1.2) and keep writing log for $\log_2$.

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

**Definition 2.6.** *For two integers $q > 0$ and $s > 1$, a matrix $M$ in $\mathbb{Z}_{qs}^{n\times n}$ is factor-$q$ nonsingular (or just factor-nonsingular) modulo $qs$ if there exists a matrix $Q$ in $\mathbb{Z}^{n\times n}$ such that*

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

*or equivalently if there exists the $s$-adic expansion $qM^{-1} = q\sum_{i=0}^{\infty} Q_i s^i$, $Q_0 = Q$, $Q_i \in \mathbb{Z}_s^{n\times n}$ for all $i$.*

## 2.2 Toeplitz and Hankel matrices

**Definition 2.7.** *$T = (t_{i,j})_{i,j}$ is a Toeplitz matrix if $t_{i,j} = t_{i+1,j+1}$ for every pair of its entries $t_{i,j}$ and $t_{i+1,j+1}$. $Z(\mathbf{v})$ is the lower triangular Toeplitz matrix defined by its first column $\mathbf{v}$. $H = (h_{i,j})_{i,j}$ is a Hankel matrix if $h_{i,j} = h_{i-1,j+1}$ for every pair of its entries $h_{i,j}$ and $h_{i-1,j+1}$. $J = (j_{g,h})_{g,h=0}^{n-1,n-1}$ denotes the unit Hankel (reflection) matrix where $j_{g,n-1-g} = 1$ for $g = 0, \dots, n-1$, $j_{g,h} = 0$ for $h + g \ne n - 1$. (Application of the matrix $J$ reverses any vector $\mathbf{v} = (v_i)_{i=0}^{n-1}$, that is, $J\mathbf{v} = (v_{n-i-1})_{i=0}^{n-1}$, $J^2 = I$.)*

Clearly, $TJ$ and $JT$ are Hankel matrices if $T$ is a Toeplitz matrix, and $HJ$ and $JH$ are Toeplitz matrices if $H$ is a Hankel matrix. Therefore, Toeplitz and

Hankel linear systems are immediately reduced to each other. *We study only the Toeplitz case.*

The next well-known results (see, e.g., [P01, Chapter 2]) reduce multiplication of a Toeplitz matrix $T$ and its inverse by a vector to polynomial multiplication. One may yield similar complexity results by relying on the factor-circulant representations of the matrices $T$ and $T^{-1}$ (see [P01, Section 2.6 and Exercise 2.24c]).

**Theorem 2.1.** *Multiplication of an $m \times n$ Toeplitz matrix $T$ by a vector is a subproblem of multiplication of two polynomials of degrees $m+n-2$ and $n-1$ whose coefficients are given by the entries of the input matrix and vector, respectively. If the matrix $T$ is triangular and $m = n$, then both of these polynomials have degree $n-1$.*

**Corollary 2.1.** *An $n \times n$ Toeplitz matrix $T$ can be multiplied by a vector in $m(k)$ arithmetic operations for $m(n)$ in (1.1) and $k = 3n-3$; the bound decreases to $m(k)$ for $k = 2n-2$ if $T$ is a triangular Toeplitz matrix.*

The next theorem of Heinig 1979 [H79] extends the Gohberg–Semencul formula of 1972.

**Theorem 2.2.** *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 $\mathbf{p} = \mathbf{p}(t_{-n})$ (for a fixed $t_{-n}$) and $\mathbf{v}$ is called a *generator* for $T^{-1}$.

The next theorem is a corollary of Theorems 2.1 and 2.2.

**Theorem 2.3.** *$4m(k)+n$ arithmetic operations for $m(n)$ in (1.1) and $k = 2n-2$ suffice to multiply the matrix $T^{-1}$ by a vector provided that $T$ is a nonsingular Toeplitz matrix and $T^{-1}$ is given with its generator, that is, the vectors $\mathbf{p}$ and $\mathbf{v}$ in Theorem 2.2.*

## 2.3 Rational number reconstruction

**Definition 2.8.** *$\mathbb{Z}_q$ is the ring of integers modulo $q$. $\mathrm{ord}_q(m)$, the order of $q$ in $m$, is the maximal integer $l$ such that $q^l$ divides $m$. $z_q = z \bmod q$ for $z, q \in \mathbb{Z}$, $q > 1$ is a unique integer such that $q$ divides $z - z_q$ and $0 \le z_q < q$. $\nu(y)$ denotes the numerator, and $\delta(y)$ denotes the denominator in the ratio $y = \nu(y)/\delta(y)$ of two coprime integers $\nu(y)$ and $\delta(y)$.*

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

In this paper we recover the rational coordinates $x/y$ of the solution to a linear system of equations where the pairs of coprime integers $|x|$ and $y$ are bounded from above based on Fact 2.1. Further practical gain can rely on heuristics and the early termination techniques (see [ABM99], Dumas et al. 2001 [DSV01], and our Section 5). We choose $l$ and $k$ such that $2|x| < k$ and the solution is unique.

Likewise, by saying *numerical rational roundoff* we refer to the recovery of a unique rational number $x/y$ from three integers $\nu, \delta$, 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 $|\nu| < \delta$. $\bar{\rho}(\log \delta)$ denotes the bit-operation complexity of the recovery.

We may solve both of the recovery problems by applying the extended Euclidean algorithm to the input pairs $(r_0, r_1) = (q, r)$ and $(r_0, r_1) = (\delta, \nu)$. Here $|r_1| < |r_0|$, and we stop for the smallest positive $i$ such that $r_i < k$ in the remainder sequence $r_0, r_1, r_2, \ldots$. The remainder sequence can be complemented by two dual sequences of convergents, denoted $s_0, s_1, s_2, \ldots$ and $t_0, t_1, t_2, \ldots$ in [GG03] (cf. also Schrijver 1986 [S86] and Zippel 1993 [Z93]). For both problems of modular and numerical rational roundoff, the desired rational solution can be readily obtained from the two triples $(r_{i-1}, s_{i-1}, t_{i-1})$ and $(r_i, s_i, t_i)$, each made up of a remainder and two convergents.

Hereafter $\sigma(\log r_0)$ denotes the bit operation complexity of computing these triples for given values of $r_0, r_1$, and $k$. We have

$$\rho(d) \le \sigma(d) + O(d), \quad \bar{\rho}(\bar{d}) \le \sigma(\bar{d}), \tag{2.3}$$

$$\sigma(d) \le cd^2, \quad \sigma(\bar{d}) \le \bar{c}\bar{d}^2 \tag{2.4}$$

where $d = \log l$, $\bar{d} = \log \delta$, and $c$ is a constant [GG03], [S86], [Z93]. Presently the most popular practical choice for the rational number reconstruction is the classical algorithm supporting (2.4), but the asymptotic improvement stated below is potentially competitive for larger values of $d$.

**Theorem 2.4.**

$$\sigma(d) \le C\mu(d) \log d \tag{2.5}$$

*for $\mu(d)$ in (1.2) and a positive constant $C$ exceeding $c$ in (2.4).*

*Proof.* See [B03], Pan and Wang 2002 [PW02], 2003 [WP03], and 2004 [PW04]. □

The recent advance in Monahan 2004 [M04] allows us to recover a unique pair of coprime integers $(\nu(y), \delta(y))$ provided we are given two integers $y$ and $m$ such that $y \bmod m = \nu(y)/\delta(y) \neq 0$ and $2|v(y)|\delta(y) < m$. The pair is unique even if the product of the available upper bounds on $|\nu(y)|$ and $\delta(y)$ exceeds $m/2$.

# 3 Generalized Hensel's and Newton's lifting for linear systems and matrix inverse

## 3.1 Generalized Hensel's lifting

Let us generalize Hensel's lifting algorithm in [MC79], [D82] to perform it in the rings $\mathbb{Z}_{qs}$, for two integers $q > 0$ and $s > 1$. Actually we only need the case where they are the powers of two or another fixed integer $m > 1$, possibly a prime. We assume that $M$ is a factor-$q$ nonsingular matrix in $\mathbb{Z}_{qs}^{n\times n}$ (see Definition 2.6). Then we compute the first $h$ terms in the $s$-adic expansion of the vector $qM^{-1}\mathbf{b} = q\sum_{i=0}^{\infty}\mathbf{u}^{(i)}s^i$ where $\mathbf{u}^{(i)} = Q_i\mathbf{b}$, $i = 0, 1, \ldots$.

**Algorithm 3.1.** *Generalized lifting (see Examples 11.1–11.3).*

INPUT: *a matrix $M \in \mathbb{Z}^{n\times n}$, a vector $\mathbf{b} \in \mathbb{Z}^n$, three positive integers $h$, $q$, and $s$, and a matrix $Q = (qM^{-1}) \bmod (qs)$ satisfying (2.2).*

OUTPUT: *the vector $\mathbf{x}^{(h)} \in \mathbb{Z}^n$ such that $\mathbf{x}^{(h)} = (qM^{-1}\mathbf{b}) \bmod (qs^h)$, that is, such that $M\mathbf{x}^{(h)} = (q\mathbf{b}) \bmod (qs^h)$.*

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

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

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 3.1 and the theorem have appeared in [D82].

**Theorem 3.1.** *For $\mathbf{r}^{(i)}$ and $\mathbf{x}^{(h)}$ in Algorithm 3.1, we have*

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

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

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

$$\beta = \beta(\mathbf{b}) = \max_j |b_j|, \quad \alpha = \alpha(M) = \max_{i,j}|m_{i,j}|, \quad \gamma = 2\alpha n + \beta. \tag{3.1}$$

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

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 $i$.

□

## 3.2 Matrix inversion via generalized Newton's lifting

Let us extend generalized Hensel's lifting to matrix inversion and accelerate it. Recursively compute the matrices

$$X_0 = qM^{-1} \bmod (qs), X_i = X_{i-1}(2qI - MX_{i-1}) \bmod (qs^{2^i}), \tag{3.2}$$

$i = 1, 2, \ldots, h$. Assuming the reduction modulo $qs^{2^i}$, deduce that $qI - MX_i = (qI - MX_{i-1})^2 = (qI - MX_0)^{2^i} = 0$, that is, $qI = MX_i \bmod (qs^{2^i})$. For $q = 1$, this is Newton's lifting for matrix inversion [MC79], which has obvious similarity to Newton's iteration for the inversion of a matrix $M$ numerically [P01, Chapter 6]:

$$X_i = X_{i-1}(2I - MX_{i-1}), \quad i = 1, 2, \ldots. \tag{3.3}$$

The $i$-th step (3.3) squares the residual matrix $I - MX_{i-1}$, thus implying guadratic convergence of the approximations $X_i$ to $M^{-1}$.

**Remark 3.1.** *Striking similarity can be also observed between the algebraic Algorithm 3.1 for Hensel's lifting and the celebrated algorithm for iterative improvement of the numerical solution of linear systems of equations. (Compare Golub and Van Loan 1996 [GL96, Section 3.5.3], Skeel 1980 [S80], Higham 1996 [H96], and our Algorithm 9.2.) This similarity was exploited in Pan 1992 [P92a] and Emiris et al. 1998 [EPY98] to improve the solution algorithm. The improvement relies on performing modular arithmetic with binary rational numbers to avoid computations with the vanishing leading bits of the residuals.*

In $h$ steps, the generalized Newton lifting (3.2) achieves as much as generalized Hensel's in $2^h$ steps, but the precision of computing is roughly doubled in every Newton's step, reaching the level of $(2^h \log s + \log q)$-bit precision in $h$ steps. Every Newton's step (3.2) is essentially reduced to performing $n \times n$ matrix multiplication twice. For Toeplitz matrices $M$, however, we simplify the iteration. Indeed, for a Toeplitz matrix $T = (t_{k-j})_{k,j} = M/q$, $\mathbf{e}_1 = (1, 0, \ldots, 0)^T$ and $\mathbf{t}$ defined in Theorem 2.2, the inverses $X_i = qM^{-1} \bmod (qs^{2^i})$ in $\mathbb{Z}_{qs^{2^i}}$, $i = 0, 1, \ldots$, can be represented with their $n \times 2$ generators $X_i(\mathbf{e}_1, \mathbf{t}) = (X_i\mathbf{e}_1, X_i\mathbf{t})$.

Thus our iteration (3.2) takes the following form,

$$\begin{array}{rcl} X_0(\mathbf{e}_1, \mathbf{t}) & = & qM^{-1}(\mathbf{e}_1, \mathbf{t}) \bmod (qs), \\ X_i(\mathbf{e}_1, \mathbf{t}) & = & X_{i-1}(2qI - MX_{i-1})(\mathbf{e}_1, \mathbf{t}) \bmod (qs^{2^i}), \end{array} \tag{3.4}$$

$i = 1, 2, \ldots$. 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 is still $O(m(n))$ arithmetic operations (see Theorems 2.1 and 2.2), which is much less than the complexity of $n \times n$ matrix multiplication.

Likewise, if $M$ is a Toeplitz matrix, we represent the iterates $X_i$ in (3.3) with their generators and rewrite iteration (3.3) as follows,

$$X_i(\mathbf{e}_1, \mathbf{t}) = X_{i-1}(2I - MX_{i-1})(\mathbf{e}_1, \mathbf{t}). \tag{3.5}$$

# 4 Deterministic recovery of the rational solution

To recover the unique vector $\mathbf{x} = qM^{-1}\mathbf{b}$ from the output vector $\mathbf{x}^{(h)}$ of Algorithm 3.1, we need a sufficiently large $h$. Let us estimate how large.

**Theorem 4.1.** *Let $\mathbf{x} = qM^{-1}\mathbf{b}$ denote a unique solution to the linear system $M\mathbf{x} = q\mathbf{b}$. Assume $\rho(d)$ in (2.3)–(2.5),*

$$d = \lceil \log(2(\alpha\sqrt{n})^{2n-1}n\beta)q \rceil = O(n \log \gamma + \log q), \tag{4.1}$$

*and $\alpha$, $\beta$ and $\gamma$ in (3.1). Suppose that in*

$$h = 1 + \lfloor \log_s(2(\alpha\sqrt{n})^{2n-1}n\beta) \rfloor \tag{4.2}$$

*steps Algorithm 3.1 computes the vector*

$$\mathbf{x}^{(h)} = \sum_{i=0}^{h-1} \mathbf{u}^{(i)}p^i = \mathbf{x} \bmod (qs^h).$$

*Then it is sufficient to perform*

$$B_1 = n\rho(d) \tag{4.3}$$

*bit operations to recover the vector $\mathbf{x}$ from the vector $\mathbf{x}^{(h)}$.*

*Proof.* Suppose that the pairs of 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 $\mathbf{x} = (x_j)_j = qM^{-1}\mathbf{b}$. Fix the smallest integer $k > 2(\alpha\sqrt{n-1})^{n-1}n\beta q$ and choose $h$ in (4.2), such that $s^h > 2(\alpha\sqrt{n})^{2n-1}n\beta$. Deduce from Fact 2.1 that $l = qs^h > 2|\nu_j|\delta_j$ and $2|\nu_j| < k \le qs^h$. Then according to Section 2.3 every component $x_j$ can be uniquely recovered from $qx_j \bmod (qs^h)$. Now Theorem 2.4 supports the claimed bit complexity bound for this recovery. □

**Remark 4.1.** *The reconstruction of the rational solution becomes trivial if we first compute (modulo a prime $p$) the $MBA$ type recursive triangular factorization of the symmetrized matrix $M^TM$ by applying the algorithm in [P04] (cf. [P01, Section 5]), then lift the solution as in [P00], and output it modulo $p^{2h}$ for $h$ in (4.2) where $s = p$. Indeed, as by-product, this produces the*

*values of* $\det(M^TM) = (\det M)^2 \mod p^{2h}$*, from which we obtain* $(\det M)^2$ *and* $|\det M|$*. Now the reconstruction of the vector* $\mathbf{x} = M^{-1}\mathbf{b}$ *becomes trivial because* $\mathbf{y} = |\det M|\mathbf{x}$ *is an integer vector and can be immediately reconstructed from the vector* $\mathbf{y} = |\det M|\mathbf{x} \mod p^{2h}$*. The price is the restriction of the initial computations to the field* $\mathbb{Z}_p$*, some complication of the code, and the increase of the overall arithmetic and bit complexity bounds by the factor in* $O(\log n)$ *versus the Hensel lifting approach.*

# 5 Randomized recovery of the rational solution

For the values $\mu(d)$ in $O(d^{\log 3})$ or $O((d\log d)\log\log d)$ and $\rho(d)$ bounded in (2.3) and (2.5), we may decrease the bit complexity bound in (4.3) by the factor in $O(\log d)$ by using Las Vegas randomization, that is, we allow failure with a probability of at most $\epsilon$ for a fixed positive $\epsilon$ such that $\log(1/\epsilon) = O(\log n)$, but otherwise the output is correct.

The acceleration relies on two observations:

(a) The vector $\mathbf{y} = \delta\mathbf{x}$ is filled with integers provided

$$\delta = \mathrm{lcm}_j\, \delta(x_j), \quad 1 \le j \le n \tag{5.1}$$

(for $\delta(y)$ in Definition 2.8), that is, $\delta$ 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 $M\mathbf{x} = \mathbf{b}$. Due to the integrality of the vector $\mathbf{y}$, its recovery from the vector $\mathbf{y} \bmod (qs^h)$ is immediate if $qs^h > 2\delta|\mathbf{x}| = 2|\mathbf{y}|$. Since $\delta \le s_n(M) \le |\det M| \le (\alpha(M)\sqrt{n})^n$ (see (2.1) and Fact 2.1), it is sufficient to use $h$ of (4.2). Multiplication of the vector $\mathbf{x}$ by $\delta$ requires the order of $n\mu(d)$ bit operations, thus limiting the theoretical gain versus the estimate $B_1 = n\rho(d)$ in (4.3). The practical gain can be significant, however.

(b) Computation of $\delta$ can be accelerated with randomization because $\delta$ is likely to equal the least common multiple of the denominators in a smaller number $K < n$ of random linear combinations $\mathbf{c}_k^T\mathbf{x}$ of the coordinates $x_1, \dots, x_n$, $k = 1, \dots, K$. According to the tests by Victor Shoup and Jean-Guillaume Dumas, one may typically use some selected entries themselves, e.g., the first, the second, etc., instead of random linear combinations of the entries.

The approach can be traced back to Pan 1988 [P88, Section 6]. Its recent studies include [ABM99], Cooperman et al. 1999 [CFG99], [EGV00], and Mulders and Storjohann 2004 [MS04]. Let us specify and briefly analyze generalized Hensel's lifting with randomized recovery.

**Algorithm 5.1.** *Randomized recovery of the rational solution.*

INPUT: *As in Algorithm 3.1 and in addition a positive* $\varepsilon < 1$ *and the vector* $\mathbf{x}^{(h)} = (x_i^{(h)})_{i=1}^n = qM^{-1}\mathbf{b} \bmod (qs^h)$ *for* $h$ *in (4.2).*

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

$$M\mathbf{y} = \delta\mathbf{b}. \tag{5.2}$$

INITIALIZATION: *Compute*

$$K = 2\lceil \log(1/\varepsilon) \rceil, \tag{5.3}$$

$$\eta = 6 + 2n \log(n\alpha), \tag{5.4}$$

$$h = 1 + \lfloor \log_s (2n(\alpha\sqrt{n})^{2n-1}\eta\beta) \rfloor \tag{5.5}$$

*for $\alpha$ and $\beta$ in (3.1). Then sample $K$ pseudo random vectors*

$$\mathbf{c}_k = (c_{jk})_{j=1}^n \in \mathbb{Z}_\eta^n, \quad k = 1, \ldots, K. \tag{5.6}$$

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 $\nu_k$ and $\delta_k$ such that*

$$(\nu_k/\delta_k) \bmod (qs^h) = w_k, 1 \le 2\delta_k|\nu_k| \le qs^h, \quad 2|\nu_k| < qs^h, \quad k = 1, \ldots, K. \tag{5.7}$$

3. *Compute the least common multiple of the denominators*

$$\delta_{lcd} = \operatorname{lcm}_k \delta_k, \quad 1 \le k \le K. \tag{5.8}$$

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 $M\mathbf{y} = \delta_{lcd}\mathbf{b}$, output $\mathbf{y}$ and $\delta = \delta_{lcd}$; otherwise output FAILURE.*

Combining equations (5.4)–(5.6) and Fact 2.1 implies (5.7). Now, correctness of Algorithm 5.1 is implied by the following simple result.

**Theorem 5.1.** *$\delta_{lcd}$ in (5.8) divides $\delta$ in (5.1). Furthermore,*

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

Theorem 5.1 is deduced similarly to Theorem 2.1 in [EGV00] based on (5.3)–(5.8) and the following lemma.

**Lemma 5.1.** *For a prime $p$, integers $K$ in (5.3) and $k$, $1 \le k \le K$, $\delta$ in (5.1), $\eta$ in (5.4), and $\delta_k$ in (5.7), we have* $\text{Probability}(\text{ord}_p(\delta_k) < \text{ord}_p(\delta)) \le \max\{\frac{1}{p}, \frac{1}{\eta}\}$.

*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 $\mathbf{x} = M^{-1}\mathbf{b}$, $l \ge h$, and $a, b, u$, and $v$ are four integers coprime with $p$. Clearly, $\mathrm{ord}_p(\delta_k)$ for $\delta_k$ in (5.7) 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, the probability bound follows. □

Let us estimate the bit complexity of performing Algorithm 5.1 in terms of $d = O(n\log\gamma + \log q)$ in (4.1), $\mu(d)$ in (1.2), $\rho(d)$ in (2.3)–(2.5), and $K$ in (5.3). We need the following auxiliary result.

**Lemma 5.2.** *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 5.1 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 $O(m(n)\mu(\log\gamma)d/\log\gamma)$ for computing the vectors $\delta_{lcd}\mathbf{x}^{(h)}$, $\delta_{lcd}\mathbf{b}$, and $M\mathbf{y}$ at Stage 4, respectively. (The two latter bounds are deduced based on Lemma 5.2.) Summarizing, we obtain the following estimate.

**Theorem 5.2.** *Algorithm 5.1 generates $O(nK)$ random elements in $\mathbb{Z}_\eta$ for $\eta$ in (5.4) and $K = 2\lceil\log(1/\epsilon)\rceil$ in (5.3). It either fails (this occurs with a probability of at most $\epsilon$) or computes the solution $\mathbf{y}, \delta$ to the linear system (5.2). The algorithm involves*

$$B_1 = O(Kn\mu(d) + K\rho(d) + m(n)\mu(\log\gamma)d/\log\gamma) \tag{5.9}$$

*bit operations for $d = O(n\log\gamma + \log q)$ in (4.1), $\rho(d)$ in (2.3)–(2.5), $\gamma$ in (3.1), $m(n)$ in (1.1), and $\mu(d)$ in (1.2); it involves $o(B_1)$ bit operations for generating $O(nK)$ pseudo random elements in $\mathbb{Z}_\eta$.*

# 6 Computational complexity of generalized lifting

**Lemma 6.1.** *Algorithm 3.1 operates with integers in the range $[-2^{d_1}, 2^{d_1})$ where*

$$\lceil\log(qs)\rceil \le d_1 = \lceil\log(\max\{qs, \gamma\})\rceil \tag{6.1}$$

*for $\gamma$ in (3.1).*

*Proof.* The lemma follows from Theorem 3.1 a) and c) since the vectors $\mathbf{u}^{(i)}$ are computed in $\mathbb{Z}_{qs}$. □

Combining (4.2) and Lemma 6.1 implies the following theorem.

**Theorem 6.1.** *Let* $\log n = O(\log \gamma)$, $\log q = O(\log s)$. *Then Algorithm 3.1 uses*

$$A_0 = O((v_M + v_Q)h) \tag{6.2}$$

*arithmetic operations for* $h$ *in (4.2) or (5.5). They are performed with the precision of* $d_1$ *bits and involve*

$$B_0 = O((v_M + v_Q)h\mu(d_1)) \tag{6.3}$$

*bit operations for* $\mu(d)$ *in (1.2),* $\gamma$ *in (3.1),* $d_1$ *in (6.1), and* $v_S$ *in Definition 2.4, so that* $v_M + v_Q$ *is in* $O(n^2)$ *for a general matrix* $M$ *and is in* $O(m(n))$ *for a Toeplitz matrix* $M$.

$A_0$ is the most appropriate complexity measure under the word model provided the precision of computing is bounded by the length $\lambda$ of the computer word. In our case this means the bound

$$d_1 = \lfloor \log \max\{\gamma, qs\} \rfloor < \lambda, \tag{6.4}$$

and typically we have $2^\lambda \geq \gamma$.

Since $A_0$ is inversely proportinal to $\log s$, we seek the maximal $s$ such that

$$\lfloor \log(qs) \rfloor < \lambda. \tag{6.5}$$

We call this policy *saturated initialization*. It allows substantial practical saving of lifting steps and word operations versus the restrictive customary policy where $q = 1$ and $s$ is a prime. Indeed larger primes are harder to handle, whereas for smaller primes $s$ we cannot yield (6.5).

Under (6.5) Algorithm 3.1 performs $A_0$ arithmetic operations with the precision $\lambda$, which are word operations. For $q = 1$ this saves for us the factor of $\lambda / \log p$ word operations versus the unsaturated initialization with a smaller (random) prime $s = p$ as the basis.

# 7 Summary of the bit operation complexity of lifting and the recovery of the solution

Table 7.1 summarizes the bit complexity estimates in Theorems 4.1, 5.2 and 6.1. To make the estimates more observable, we use the notation "$\tilde{O}$" (which means "$O$" up to the factors in $(\log \log n)^{O(1)}$) and the following simplifying assumptions,

$$\log_s \gamma = O(1), \quad \log(qs) = O(\log n), \quad \log(1/\epsilon) = O(\log n). \tag{7.1}$$

Table 7.1: The bit complexity of lifting (for general and Toeplitz input matrices $M$) and of rational reconstruction (deterministic and randomized), under (5.4), (5.5), (1.1), (1.2), (2.2)–(2.4), and (3.1).

| | |
|---|---|
| Lifting complexity $B_0$ (for a general matrix $M$) | $O(n^3\mu(\log n)) = \tilde{O}(n^3 \log n)$ |
| Lifting complexity $B_0$ (for a Toeplitz matrix $M$) | $O(nm(n)\mu(\log n)) = \tilde{O}(n^2 \log^2 n)$ |
| Reconstruction complexity $B_1$ (deterministic) | $O(n\rho(n \log n)) = \tilde{O}(n^2 \log^3 n)$ |
| Reconstruction complexity $B_1$ (randomized) | $O(n\mu(n \log n) + \rho(n)(\log n)) = \tilde{O}(n^2 \log^2 n)$ |

Here $\epsilon$ is the error probability in the randomized rational reconstruction of the output.

Our bound of $B_0 + B_1 = \tilde{O}(n^2 \log^2 n)$ bit operations on the overall randomized complexity of lifting and rational solution reconstruction is nearly optimal (assuming a Toeplitz input matrix $M$ and equation (7.1)), because the $n$ rational output values $x_1, \dots, x_n$ are represented with $n^2 \log n$ bits.

# 8 Initialization of generalized lifting modulo the power of a larger prime

To complete the lifting algorithm for a linear system $M\mathbf{x} = \mathbf{b}$, it is sufficient to solve the following problem.

**Problem 8.1.** *Initialization of the generalized lifting.*

INPUT: *a nonsingular matrix $M \in \mathbb{Z}^{n \times n}$, a prime $p$, and a positive $\lambda$. ($p$ and/or $M$ are random, $\lambda$ is a fixed upper bound on the length of a computer word.)*

OUTPUT: *either FAILURE or two integers $q > 0$ and $s > 1$, both the powers of $p$ and such that*

$$\log_p(qs) < w + 1 = \lceil \frac{\lambda}{\log p} \rceil, \tag{8.1}$$

*and a matrix $Q$ satisfying (2.2). (Note that (8.1) implies (6.5).)*

Gaussian elimination with pivoting enables us to solve Problem 8.1 for a general matrix $M$. In Section 10 we estimate the FAILURE probability for this computation.

**Algorithm 8.1.** *Initialization via Gaussian elimination.*

COMPUTATIONS: *Fix a prime $p$, compute $w$ in (8.1), and apply Gaussian elimination to invert the matrix $M$. Perform the computations in $\mathbb{Z}_{p^w}$. Apply column pivoting to avoid divisions by the multiples of $p$, that is, at every elimination step interchange the rows to minimize the order of $p$ in the pivot entry (cf. Definition 2.8). If at some step the order exceeds $w$, output FAILURE and stop. Otherwise continue the elimination until $M$ is diagonalized. Then choose $v = \mathrm{ord}_p(s_n)$ for $s_n = s_n(M)$ denoting the Smith leading invariant factor of the matrix $M$ in Definition 2.5, that is, choose $v$ equal to the maximal order in $p$ among all pivot entries (which are the diagonal entries of the output diagonal matrix). Finally fix a positive integer $u \le w - v$ and compute the matrix $Q$ satisfying (2.2) for $q = p^v$ and $s = p^u$.*

The algorithm does not fail if and only if

$$w \ge \mathrm{ord}_p(s_n(M)) \tag{8.2}$$

for $w$ in (8.1). In the next sections we can see that variation of the prime $p$ or the matrix $M$ may help where the algorithm fails.

For general matrices Algorithm 8.1 uses the order of $n^3$ arithmetic operations performed under (8.2). (We ignore the chances for theoretical asymptotic acceleration and minor practical speed up based on fast matrix multiplication; see Kaporin 2004 [K04], Dumas et al. 2004 [DGP04], and the bibliography therein.) If (8.1) holds, they are word operations. This cost bound is also reached or exceeded at the stage of lifting.

To decrease the overall number of word operations involved, we should choose

$$u = w - v. \tag{8.3}$$

In the Toeplitz case, we may replace Algorithm 8.1 by adapting the MBA divide-and-conquer algorithm, which requires only $O(n \log^2 n)$ arithmetic operations. For detailed description and analysis of this algorithm and further bibliography, see [M74], [M80], [BA80], [P01, Chapter 5], and [P04]. In this case the arithmetic, word, and bit complexity bounds at the initialization stage are strongly dominated at the lifting stage.

The MBA algorithm involves divisions, which we cannot perform in $\mathbb{Z}_{p^w}$ for a positive $w$ if $p$ divides the divisors. The algorithm avoids such divisions if and only if the input matrix $M$ is *strongly nonsingular* in the field $\mathbb{Z}_p$, that is, nonsingular (in $\mathbb{Z}_p$) together with all its leading principal submatrices [P01, Chapter 5]. Let us list some relevant results on strong nonsingularity from Section 10 and [P01]:

- A random $n \times n$ integer Toeplitz matrix is likely to be strongly nonsingular modulo any fixed prime $p >> n$ (Theorem 10.4).
- If $M$ is not strongly nonsingular in $\mathbb{Z}_p$ for a random prime $p$ sampled from a large range, then $M$ is unlikely to be strongly nonsingular even in $\mathbb{Z}$ (in virtue of Theorem 10.1).

- The matrices $M^TM$ and $MM^T$ are strongly nonsingular in $\mathbb{Z}$ if $M$ is nonsingular in $\mathbb{Z}$ [P01] and are likely to remain strongly nonsingular in $\mathbb{Z}_p$ for a larger random prime $p$ (in virtue of Theorem 10.1).

Having the matrices $M^TM$ or $MM^T$ inverted, we obtain

$$M^{-1} = (M^TM)^{-1}M^T = M^T(MM^T)^{-1}.$$

$M^TM$ and $MM^T$ are in the class of $n \times n$ Toeplitz-like matrices [P01]. Such matrices generalize $n \times n$ Toeplitz matrices. They can be represented in compact form via their *displacement generators* made up of $O(n)$ parameters and can be multiplied by vectors fast, like the matrix $T^{-1}$ in Theorem 2.2. If they are strongly nonsingular in $\mathbb{Z}_p$, we may adapt the MBA algorithm to invert them in $\mathbb{Z}_p$ by using $O(m(n)\log n)$ field operations. More precisely, we just complement the original MBA algorithm in [M80], [BA80] with the low cost deterministic algorithm in [P01, Section 4.6.2] (cf. Pan 1992 [P92, Proposition A.6]), which compresses the dispacement generators in $\mathbb{Z}_p$ wherever they involve extraneous parameters.

By choosing the basic prime $p$ not very large, we may operate in $\mathbb{Z}_p$ more efficiently. Then the bit precision of $p$ and thus the bit complexity of the initialization is low, versus the bit complexity of lifting. Together with the probabilistic estimates in Theorem 10.1, this implies the following result.

**Theorem 8.1.** *The overall bit operation complexity of solving a nonsingular integer Toeplitz linear system of equations is bounded according to Theorems 4.1, 5.2, and 6.1 and Table 7.1.*

For a lower precision (smaller) prime $p$, the initialization of Hensel's lifting with the matrix $M^{-1} \mod p$ implies performing some extra lifting steps and word operations. We may fix this deficiency by applying a small number of Newton's lifting steps of Section 3.2. Alternatively, we may perform the MBA algorithm in $\mathbb{Z}_{p^w}$ for $w$ in (8.1). As long as the input matrix is strongly nonsingular in $\mathbb{Z}_p$, the same MBA algorithm works provided the pivots in the compression algorithm in [P01, Section 4.6.2] always have the smallest order in $p$.

The MBA initialization, however, is vulnerable to degeneration where the basic prime $p$ is not large, e.g., $p = 2$ (cf. Theorem 10.4). In the next section we propose two initialization algorithms which work in $\mathbb{Z}_{qs}$ for $q = p^w, s = p^v$ (where $p$ can be small) provided $M$ is factor-$q$ nonsingular.

# 9 Initializations of generalized Toeplitz–Hensel's lifting modulo the power of a smaller prime

Let us specify and analyze two algorithms for the initialization of the generalized lifting for factor-$q$ nonsingular Toeplitz linear systems. We first show these

algorithms for solving modulo $qs$ a linear system $M\mathbf{x} = q\mathbf{f}$. The integers $q$ and $s$, both the powers of a fixed prime $p$, are computed in the process of performing the algorithms. We estimate the bit complexity of these algorithms and extend them to inverting the matrix $M$ modulo $qs$. Finally, we compare these algorithms with the MBA initialization in the previous section.

Given a prime $p$, its power $m = p^b$, a matrix $M$, and a vector $\mathbf{f} = (f_i)_i$, both of our algorithms first compute the rational vector $M_0^{-1}\mathbf{f}$ for the matrix $M_0 = aM + mI$ and a fixed integer $a$ ($a$ is coprime with $m$ in our first algorithm, and $a = 1$ in our second algorithm). At the final stage of the algorithms, we extend this to computing the vector $qM^{-1}\mathbf{f} \bmod (qs)$ for appropriate $q$ and $s$, both the powers of $p$.

## 9.1 Step 1: solving a linear system with modular continuation

**Algorithm 9.1.** *Initialization of Toeplitz–Hensel's lifting with modular continuation.*

INPUT: *A nonsingular matrix $M \in \mathbb{Z}^{n\times n}$, a vector $\mathbf{f} \in \mathbb{Z}^n$, a prime $p$, and two integers $m = p^b$ for a positive integer $b$ and $\lambda > 0$, the length of a computer word. (If this length is not bounded, write $\lambda = \infty$.)*

OUTPUT: *FAILURE or two positive integers, $q$ and $s$, both being the powers of $p$ such that $qs < 2^\lambda$, and the vector $\mathbf{y} = (qM^{-1}\mathbf{f}) \bmod (qs)$.*

INITIALIZATION: *Choose an integer $a > 1$ coprime with $p$ and such that*

$$\gamma^+ = \beta(\mathbf{f}) + 2(m + a\alpha(M))n < 2^\lambda. \tag{9.1}$$

*(We assume that the values of $m$ and $a$ are sufficiently small to have this bound.)*

COMPUTATIONS:

1. *Compute the integer $r = m^{-1} \bmod a$ and the matrix $M_0 = mI + aM$; note that $Q = M_0^{-1} \bmod a = rI$.*

2. *Let $\alpha = \alpha(M_0), \beta = \beta(\mathbf{f})$ and choose $h$ in (4.2) or (5.4), (5.5) to support deterministic or randomized recovery of the vector $M_0^{-1}\mathbf{f}$ according to Sections 4 or 5, respectively. Specifically, in the deterministic case we write*

$$h = 1 + \lfloor \log_a(2((a|M| + m)\sqrt{n})^{2n-1} n\beta(\mathbf{f})\rfloor. \tag{9.2}$$

*Apply Algorithm 3.1 for $q = 1$, $M$ replaced by $M_0$, $\mathbf{b}$ by $\mathbf{f}$, and $s$ by $a$ to compute the vector $M_0^{-1}\mathbf{f} \bmod a^h$; recover the rational vector $M_0^{-1}\mathbf{f}$.*

3. *Compute* $d = \max_j \operatorname{ord}_m(\delta((M_0^{-1}\mathbf{f})_j))$. *If* $2d \le b$, *output the integers* $q = p^d$ *and* $s = p^{b-2d} = m/q^2$; *compute and output the vector*

$$\mathbf{y} = (aqM_0^{-1}\mathbf{f}) \bmod (qs) = (qM^{-1}\mathbf{f}) \bmod (qs).$$

*Otherwise output FAILURE.*

Correctness of the algorithm follows because, as soon as we yield the equation $q^2 s = m$ at Stage 3, we have $M_0/q = (m/q)I + (a/q)M = qsI + (a/q)M = (a/q)M \bmod (qs)$, which implies the desired equations

$$M\mathbf{y} = aqMM_0^{-1}\mathbf{f} = q\mathbf{f} \bmod (qs).$$

The bit operation complexity of performing the algorithm is clearly dominated at its Stage 2. The estimates in Theorem 8.1 can be applied for

$$q = 1, s = a, \beta = \beta(\mathbf{f}), \alpha = \alpha(M_0), \gamma = 2\alpha n + \beta, \tag{9.3}$$

so that $\alpha \le \alpha^+ = m + a\alpha(M), \gamma \le \gamma^+$ for $\gamma^+$ in (9.1).

**Theorem 9.1.** *The bit operation complexity of Algorithm 9.1 applied to a Toeplitz matrix* $M$ *is bounded by* $B_0 + B_1$ *for* $B_0$ *and* $B_1$ *in Theorem 8.1 where* $q, s, \alpha, \beta$, *and* $\gamma$ *are defined in (9.3).*

The following properties should guide us in choosing the integers $a$ and $b$.

(a) The larger $a$, the fewer lifting steps at Stage 2 of Algorithm 9.1.

(b) The larger $b$, the more bit operations in Algorithm 9.1.

(c) The larger $a$ and/or $b$, the longer the precision of the computations at Stage 2, but the bound (9.1) is sufficient to keep the precision below $\lambda+1$.

(d) (9.1) holds for a positive integer $b$ if

$$b \le b^+ = \lceil \log_p \Delta \rceil, \Delta = \frac{2^\lambda - 1 - \beta(\mathbf{f})}{2n} - a\alpha(M) > 1. \tag{9.4}$$

(e) If the integer $b^+$ is fixed and we wish to minimize the word complexity, we should apply Algorithm 9.1 for $b = b^+$. If $b^+ \ge 2d$ for $d$ in Stage 2, the algorithm produces the desired output integers $q$ and $s$ and vector $\mathbf{y}$. Otherwise, the algorithm fails, but we may repeat the computations for distinct $a$ and/or $p$.

We can also see two adverse results of increasing the integer $d$:

(f) If $2d$ exceeds $b^+$, then Algorithm 9.1 fails.

(g) The number $h$ of lifting steps defined in (4.2) and (5.5) for $\alpha = \alpha(M_0)$, $\beta = \beta(\mathbf{f})$ (cf. (9.2)) is roughly proportional to $\log_a \alpha(M_0)$ and $\log_a(a\alpha(M) + m)$. Therefore $h$ is roughly proportional to $d/\log a$ if $m = p^b = p^{2d}$ dominates $a\alpha(M)$.

Let us estimate $d$.

**Theorem 9.2.** *$d = \max_j \mathrm{ord}_p((\delta(M_0^{-1})_j) = \mathrm{ord}_p(s_n(M_0)) = \mathrm{ord}_p(s_n(M)) \le \mathrm{ord}_p(\det M)$, and so $b \ge 2d$ at Stage 2 of Algorithm 9.1 if $b \ge 2\,\mathrm{ord}_p(\det M)$. The latter bound holds if $b \ge 2\log_p |\det M|$.*

*Proof.* The theorem is easily deduced from the definitions of $M_0$, $\delta(x/y)$, and $s_n(M)$ and from the bounds (2.1) since $a$ and $p$ are coprime. □

Now, in addition to (2.1), recall that for a larger random prime $p$ and/or a random integer matrix $M$, $\mathrm{ord}_p(s_n(M))$ tends to be within a small factor from $\mathrm{ord}_p(\det M)$ (see Theorems 10.3 and 10.7), that is within a small factor from $n\log_p(\sqrt{n}\alpha(M))$. Then, in virtue of Theorem 9.2, the integers $b$ and $d$ should be of the order of $n\log_p(\sqrt{n}\alpha(M))$. This means that for a moderate bound $\lambda$ and a larger integer $n$, Algorithm 9.1 should fail, whereas for a larger $\lambda$, that is, for computations with the extended precision, the number $h$ of lifting steps at Stage 2 of this algorithm should grow by roughly the factor of $n/\log a$ versus the estimates in (4.2) and (5.5). Due to this growth caused by the term $mI = p^b I$ in $M_0$, the arithmetic, word, and bit complexity estimates for the initialization with Algorithm 9.1 should exceed by roughly the factor of $n$ the respective estimates in Theorem 8.1 for the complexity of the subsequent solution of a Toeplitz linear system. We avoid decreasing $h$ by means of increasing the value $\log a$ because of the high price of increasing $\alpha(M_0)$ and $\Delta$ in (9.4).

The above comments apply to the worst case input $p$ and $M$. For a larger random prime $p$ and/or a random Toeplitz matrix $M$, however, the chances for the failure of Algorithm 9.1 dramatically decrease because the integers $\mathrm{ord}_p(\det M)$ and $d$ tend to be in $O(\log_p n)$ according to the estimates in Sections 10.1 and 10.2. If so, we have $\log p^b = O(\log n), \log\alpha(M_0) = O(\log(a\alpha(M)+n))$, and adding the complexity estimates for the initialization with Algorithm 9.1 would not affect our overall asymptotic estimates for solving the Toeplitz linear systems.

## 9.2 Step 1: solving a linear system with variable diagonal

With Algorithm 9.1 we cannot keep the computation of the vector $\mathbf{x} = M^{-1}\mathbf{f}$ in binary form because $a$ and $s$ are coprime and thus cannot both equal the powers of two. Our next algorithm does not have this deficiency and still uses about as many lifting steps and bit operations as Algorithm 9.1. The lifting stage of our second algorithm can be performed numerically with bounded precision. We specify it only for deterministic recovery at Stage 2, but one may immediately extend the recipes of Section 5 for randomized or heuristic acceleration.

**Algorithm 9.2.** *Initialization of Toeplitz–Hensel's lifting by using the variable diagonal technique (cf. [P00]).*

INPUT AND OUTPUT: *as in Algorithm 9.1 and $c > 1$ such that $m \ge c|M|$.*

INITIALIZATION: *Write $\mathbf{z}_0 = \mathbf{0}$, $\mathbf{r}_0 = \mathbf{f}$.*

COMPUTATIONS *(cf. Definition 2.3):*

1. *Compute the matrices* $M_0 = M + mI$ *and* $Q = m^{-1}I$.

2. *Recursively compute the vectors* $\mathbf{z}_{i+1} - \mathbf{z}_i = Q\mathbf{r}_i = m^{-1}\mathbf{r}_i$, $\mathbf{r}_{i+1} = \mathbf{f} - M_0\mathbf{z}_{i+1} = \mathbf{r}_i - M_0Q\mathbf{r}_i = -m^{-1}M\mathbf{r}_i$ *for* $i = 0, 1, \dots, h-1$ *and*

$$h = \lfloor (2n-1)\log_c(|M| + m) + \log_c(2|\mathbf{f}|^2/(c-1)) \rfloor \tag{9.5}$$

*(cf. (9.2)). Then recover the vector* $\mathbf{z} = M_0^{-1}\mathbf{f}$ *from* $\mathbf{z}_h$ *deterministically by using the numerical rational roundoff algorithms in Section 2.3.*

3. *Proceed as in Stage 3 of Algorithm 9.1 for* $a = 1$ *and* $\mathbf{y} = \mathbf{z}$.

Stage 2 can be implemented numerically as the customary residual correction algorithm for iterative improvement of the computed approximations to $\mathbf{z}$ where the initial approximation is given by the scaled identity matrix $Q = m^{-1}I$ (see [S80a], [GL96, Section 3.5.3], [H96]). We employ this algorithm in lieu of Hensel's auxiliary lifting.

We have

$$\begin{gathered} M_0Q - I = QM_0 - I = m^{-1}M, \\ \mathbf{z} - \mathbf{z}_h = M_0^{-1}(\mathbf{f} - M_0\mathbf{z}_h) \\ = M_0^{-1}\mathbf{r}_h, \\ \mathbf{r}_h = -m^{-1}M\mathbf{r}_{h-1} = (-m^{-1}M)^h\mathbf{r}_0 \\ = (-m^{-1}M)^h\mathbf{f}. \end{gathered}$$

Furthermore,

$$|M_0^{-1}| = m^{-1}|(I + M/m)^{-1}| \le m^{-1}\sum_{i=0}^{\infty}(|M|/m)^i,$$

and so

$$|M_0^{-1}| \le \frac{1}{(c-1)m}$$

since $m \ge c|M|$.

Therefore

$$|\mathbf{z} - \mathbf{z}_h| \le (m^{-1}|M|)^h|\mathbf{f}||M_0^{-1}| \le (m^{-1}|M|)^h|\mathbf{f}|/((c-1)m) \le c^{-h}|\mathbf{f}|/((c-1)m) \tag{9.6}$$

for $m \ge 2|M|$.

To ensure correct recovery of the vector $\mathbf{z}$ from $\mathbf{z}_h$ with using the numerical rational roundoff algorithms in Section 2.3, it is sufficient to approximate $\mathbf{z}$ by $\mathbf{z}_h$ within the error norm less than $1/(2|M_0|^{2n-1}|\mathbf{f}|)$. This bound is achieved in Algorithm 9.2 due to (9.5)–(9.6) and the inequality $|M_0| \le |M| + m$.

The analysis in the previous subsection (for $a = 1$) (including Theorems 9.1 and 9.2) is immediately extended. $b^+$ in (9.6) increases since $a = 1$, and the parameter $c$ (rather than $a$) plays the role of the lifting and logarithmic base (cf. (9.6)).

### 9.3 Step 2: extension from system solving to matrix inversion and Newton's acceleration

To initialize lifting, we seek the matrix $Q = (qM^{-1}) \mod (qs)$. For general matrix $M$, this requires $n$ applications of Algorithms 9.1 or 9.2. In the Toeplitz case, we only solve the two linear systems $M\mathbf{x} = q_0\mathbf{t} \mod (q_0 s_0)$ and $M\mathbf{y} = q_1\mathbf{e}_1 \mod (q_1 s_1)$ where $q_0, q_1, s_0$ and $s_1$ denote the respective values of the integer parameters $q$ and $s$ for these two systems and where the two vectors $\mathbf{e}_1 = (1, 0, \dots, 0)^T$ and $\mathbf{t}$ define the generator of the matrix $M^{-1}$ (see Theorem 2.2).

We choose the same basic prime $p$ for both systems and reconcile the choice of $q_0 = q_1$ and $s_0 = s_1$ by computing

$$q = q_0 = q_1 = p^\sigma, \sigma = \max_j \operatorname{ord}_p(\delta(M_0^{-1}(\mathbf{t}, \mathbf{e}_1))_j))$$

and $s = s_0 = s_1 = \frac{m}{q}$ at Stage 3, which is common in Algorithm 9.1 (or 9.2) for both linear systems with $q\mathbf{t}$ and $q\mathbf{e}_1$ on the right-hand sides.

If the precision at the lifting steps in Stage 2 in Algorithms 9.1 or 9.2 is substantially less than $\lambda$, we may accelerate lifting by applying Newton's steps (3.3) or (3.5), respectively.

### 9.4 Comparison with the initialization via the MBA approach

Recall that Theorem 9.1 covers the bit complexity of performing both Algorithms 9.1 and 9.2 and implies that the estimated overall cost of Toeplitz solving increases versus Theorem 8.1 by a factor ranging from a moderate constant for the random average input matrix $M$ to roughly $n$ in the worst case.

The initialization with the algorithm of the MBA type in Section 8 has lower bit complexity than the subsequent stages of Toeplitz solving but requires restriction $q = 1$ and $s = p^w$ for a larger random prime $p$, to counter potential degeneracies coming from the divisions involved in this algorithm. Algorithms 9.1 and 9.2 also have the advantage of involving no auxiliary matrices of smaller sizes.

## 10 Degeneration in the rings $\mathbb{Z}_m$

### 10.1 The probability of degeneration in $\mathbb{Z}_{p^v}$ for a random prime $p$

For a fixed nonsingular matrix $M$, the degeneracy condition (8.2) depends on the prime $p$. Let us assume a random prime $p$, fix its power $v$, and estimate the probability that $p^v$ divides $\det M$, recalling that $s_n(M)$ is a divisor of $\det M$.

We begin with some definitions and basic lemmas. Hereafter $\ln = \log_e$ stands for the natural logarithms (with the base $e = 2.718281\dots$) and $\pi(y)$ denotes the number of primes not exceeding $y$.

**Lemma 10.1.** *(See also (10.4).) If $y > 114$, then $1 < \frac{\pi(y)}{y} \ln y < 1.25$.*

*Proof.* See Rosser and Schoenfeld 1962 [RS62]. □

**Lemma 10.2.** *Let $y \geq 114$, then $\pi(y) - \pi(\frac{y}{20}) > (1/\tilde{\beta})\frac{y}{\ln y}$ for*

$$\tilde{\beta} = \frac{1}{1-\tilde{\alpha}} = 1.2049303\ldots, \quad \tilde{\alpha} = \frac{\ln 114}{16 \ln 5.7} = 0.17007650\ldots. \tag{10.1}$$

*Proof.* By Lemma 10.1, we have $\pi(y) - \pi(\frac{y}{20}) > \frac{y}{\ln y} - \frac{1.25y}{20\ln(y/20)}$. Observe that $\frac{\ln(y/20)}{\ln y}$ is monotone increasing as $y$ grows. So $\frac{1.25}{20\ln(y/20)} \leq \frac{\tilde{\alpha}}{\ln y}$ for $\tilde{\alpha}$ in (10.1) and $y \geq 114$. Combine the above estimates. □

**Lemma 10.3.** *(Cf. Corollary 7.8.2 in [P01].) Let $y$, $v$, $h$, and $k$ be positive integers such that*

$$y \geq 114, \quad 0 < h^{1/k} \leq y/20. \tag{10.2}$$

*Let $p$ be a random prime selected in the range $(y/20, y]$ under the uniform probability distribution. Then* Probability$(h \bmod p^v = 0) < \frac{\tilde{\beta} k \ln y}{vy}$ *for $\tilde{\beta}$ in (10.1).*

*Proof.* Suppose that in the above range there are exactly $l$ distinct primes whose $v$-th powers divide $h$. Then the product of these powers also divides $h$, and therefore we have $h \geq (\frac{y}{20})^{vl}$ because each of the $l$ primes lying in the range $[y/20, y]$ is at least as large as $\frac{y}{20}$. On the other hand, $h \leq (\frac{y}{20})^k$ by assumption. Therefore, $vl \leq k$, that is, $l \leq k/v$. Compare the latter upper bound on $l$ with the lower bound in Lemma 10.2 on the overall number of primes in the range $(\frac{y}{20}, y]$. □

**Theorem 10.1.** *(Cf. Corollary 7.8.3 in [P01].) Fix $\epsilon > 0$. Suppose that $v$ is a positive integer, $M \in \mathbb{Z}^{n\times n}$ is nonsingular, and a prime $p$ is randomly sampled from the range $(y/20, y]$ under the uniform probability distribution in this range where $y = \frac{n\xi \ln |M|}{v\epsilon} \geq 114$ and $\xi = \frac{16\ln 114}{16\ln 5.7 - \ln 114} = 16\tilde{\alpha}\tilde{\beta} = 3.278885\ldots$ for $\tilde{\alpha}$ and $\tilde{\beta}$ in (10.1). Then we have*

$$P = \text{Probability}((\det M) \bmod p^v = 0) < \epsilon. \tag{10.3}$$

*Proof.* Write $h = |\det M|$, $k = \frac{n \ln |M|}{\ln(y/20)}$, so that $h \leq |M|^n$ and $k \ln \frac{y}{20} \geq \ln h$, which implies (10.2). Apply Lemma 10.3 and deduce that

$$P < \frac{\tilde{\beta} k \ln y}{uy} = \frac{\tilde{\beta} n \ln |M|}{v \ln(y/20)} \frac{\ln y}{y} = \frac{v\epsilon\tilde{\beta} n \ln |M|}{vn \ln |M|} \frac{\ln y}{\xi \ln(y/20)} = \frac{\epsilon\tilde{\beta} \ln y}{\delta \ln(y/20)}.$$

Note that

$$\frac{\ln y}{\ln(y/20)} \leq \frac{\ln 114}{\ln 5.7}$$

for $y \geq 114$. Therefore

$$P < \epsilon \frac{\tilde{\beta} \ln 114}{\xi \ln 5.7} = \frac{16\tilde{\alpha}\tilde{\beta}}{\xi}\epsilon = \epsilon.$$

□

To extend the above results to smaller $y$, one may exploit the known extensions of Lemma 10.1, e.g.,

$$1 + \frac{1}{2\ln y} < \pi(y)\frac{\ln y}{y} < 1 + \frac{3}{2\ln y} \tag{10.4}$$

for $y \geq 59$ [GG03, Theorem 18.7]. More refined estimates for $\pi(y)$ can be found in Karatsuba 1990 [K90].

Let us extend Theorem 10.1 to any integer $q$ instead of $q = p^v$. We rely on the following observation.

**Lemma 10.4.** *Let $p$ and $q$ be coprime and let $u$, $v$, and $h$ be three positive integers. Then $p^u q^v$ divides $h$ if and only if both $p^u$ and $q^v$ divide $h$.*

**Corollary 10.1.** *Let $p_1, \ldots, p_h$ be $h$ distinct primes sampled randomly and independently in the ranges $(y_i/20, y_i]$, $i = 1, \ldots, h$, respectively, under the uniform probability distribution. Here $y_i = \frac{\xi n}{2u_i\epsilon} \ln |M| \geq 114$ for $\xi$ in Theorem 10.1 and $i = 1, \ldots, h$; the matrix $M \in \mathbb{Z}^{n\times n}$ is nonsingular; $u_1, \ldots, u_h$ are positive integers, and*

$$2h - 2 \leq \frac{y_i}{\overline{\beta}\ln y_i} \tag{10.5}$$

*for $\overline{\beta}$ in Lemma 10.2 and for all $i$. Then we have*

$$P = Probability(p_1^{u_1} \cdots p_h^{u_h} \text{ divides } \det M) \leq \epsilon^h.$$

*Proof.* Corollary 10.1 follows from Lemma 10.4 and Theorem 10.1 for $y = y_i$ and $v = 2u_i$. The primes $p_1, \ldots, p_{i-1}$ are excluded from the range $(y_i/20, y_i]$ for every $i$; this decreases the overall number of primes in this range but less than by twice for $i \leq h$ because of (10.5) and Lemma 10.2. The effect of this decrease on the probability estimates is overweighed by the increase of $v$ from $u_i$ to $2u_i$. □

Corollary 10.1 shows that computing modulo the product of distinct random primes decreases the probability of the degeneration.

**Remark 10.1.** *A random integer matrix $M$ is strongly nonsingular in $\mathbb{R}_q^{n\times n}$ for $q = p^v$ or $q = p_1^{u_1} \cdots p_k^{u_k}$ with a probability which is within the factor of $n$ from the respective bounds in Theorem 10.1 and Corollary 10.1.*

## 10.2 The probability of degeneration for a fixed $p$

Suppose we fix a basic prime $p$ and two integers $q$ and $s$ where $q = p^v$ and $s = p^u$ for a fixed basic prime $p$ (e.g., we choose $p = 2$ wherever our computer environement exploits the advantages of binary computations). Suppose we wish to estimate the probability that our computations for a random integer matrix $M$ can be performed with a precision within the word length $\lambda$. For computations with general matrices we are guided by the following analytic estimate by Brent and McKay 1987 for the proportion of singular matrices in $\mathbb{Z}_{p^u}^{n\times n}$. (They also supply similar estimates in $\mathbb{Z}_q^{n\times n}$ for any integer $q > 1$.)

**Theorem 10.2.** *[BMK87, Corollary 2.2]. Write* $P_k(r) = (1-r)(1-r^2)\cdots(1-r^k)$, $r = 1/p$. *Then the nonsingular matrices make up the fraction* $\frac{P_{n+u-1}(r)}{P_{u-1}(r)}$ *of all matrices in* $\mathbb{Z}_{p^u}^{n\times n}$.

Brent and McKay show specific estimates for their ratios as $n \to \infty$ and $q = 1, \ldots, 16$. Our Table 10.4 in Section 10.4 shows some statistics of nonsingularity of random integer matrices in $\mathbb{Z}_q$, for $n = 5, 10, 50, 100, q = 2^g$, and $g = 0, 1, \ldots, 20$.

They are in reasonable agreement with the analytic estimates in [BMK87].

For Toeplitz versus general matrices $M$, the known analytic estimates and the results of our experiments in Tables 10.1–10.3 in Section 10.4 show a little higher proportion of nonsingular matrices in $\mathbb{Z}_q^{n\times n}$ for the same $n$ and $q$. We have the following estimates in Daykin 1960 and Kaltofen and Lobo 1996 in the case of a prime $q$.

**Theorem 10.3.** *[D60], [KL96]. For any pair of a prime* $p$ *and a positive integer* $n$, *the singular matrices make up a fraction of* $1/p$ *in the space of all Toeplitz matrices in* $\mathbb{Z}_p^{n\times n}$.

We wish to point out a corollary of independent interest.

**Corollary 10.2.** *For any pair of a prime* $p$ *and a positive integer* $n$, *consider the space of the pairs of polynomials* $u(x)$ *and* $v(x)$ *over* $\mathbb{Z}_p$ *such that* $\deg v(x) = n, \deg u(x) < n$. *Then the pairs of coprime polynomials make up a fraction of* $1 - 1/p$ *in this space.*

*Proof.* The corollary follows by combining the latter theorem with Proposition 9.1 on page 159 in the book by Bini and Pan 1994 [BP94]. This proposition defines a bijection map of all pairs $(h, H)$ of $h \in \mathbb{Z}_p$ and nonsingular Hankel matrices $H$ in $\mathbb{Z}_p^{n\times n}$ to all pairs of coprime polynomials $u(x)$ and $v(x)$ over $\mathbb{Z}_p$ where $v(x)$ is monic, $\deg v(x) = n$, and $\deg u(x) < n$. Theorem 10.3 enables us to count the number of pairs $(h, H)$ where $H$ is nonsingular in $\mathbb{Z}_p^{n\times n}$ because of the bijection $J : H \leftrightarrow T = HT$. Now we extend this count to the number of pairs of coprime polynomials $u(x)$ and $v(x)$ over $\mathbb{Z}_p$ and obtain the corollary. □

**Theorem 10.4.** *[KL96, Theorem 5]. For any pair of a prime* $p$ *and a natural* $n$, *the strongly nonsingular matrices (that is, nonsingular with all their leading principal submatrices) make up a fraction of* $(1 - \frac{1}{p})(1 - \frac{p-1}{p^2})^{n-1}$ *in the space of all Toeplitz matrices in* $\mathbb{Z}_p^{n\times n}$.

We know of no extensions of the above analytic estimates to the rings $\mathbb{Z}_q$ for any integer $q > 1$. Our next results may partly fill this void.

**Theorem 10.5.** *The fraction of at least* $1 - n/q$ *Toeplitz matrices in* $\mathbb{Z}_q^{n\times n}$ *are nonsingular.*

*Proof.* There are $q^{2n}$ pairs of univariate polynomials $u, v$ over $\mathbb{Z}_q$ where $\deg u < n$, $\deg v = n$, $v$ is monic. These polynomials are not coprime if and only if their

resultant vanishes in $\mathbb{Z}_q$. In virtue of the celebrated lemma in [DL78] (also in [Z79] and [S80]), this occurs for the fraction of at most $n/q$ pairs because the resultant is a polynomial of degree of at most $n$ in the coefficients of $u$ and $v$. This means at least $(q-n)q^{2n-1}$ pairs of coprime polynomials $u$ and $v$ over $\mathbb{Z}_q$. Due to the bijection on page 159 in [BP94], already cited, we have as many pairs $(h, H)$ in $(\mathbb{Z}_q, \mathbb{Z}_q^{n\times n})$ where $H$ is a nonsingular Hankel matrix. Therefore, there are at least $(q-n)q^{2n-2}$ nonsingular Hankel matrices in $\mathbb{Z}_q^{n\times n}$ among a total of $q^{2n-1}$ Hankel matrices in $\mathbb{Z}_q^{n\times n}$. The bijection $J : H \leftrightarrow T = HJ$ extends this count to Toeplitz matrices. □

**Corollary 10.3.** *The fraction of at least* $1 - \frac{(n+1)n}{2q}$ *Toeplitz matrices in* $\mathbb{Z}_q^{n\times n}$ *are strongly nonsingular.*

*Proof.* There are at most $iq^{2n-2}$ Toeplitz matrices in $\mathbb{Z}_q^{n\times n}$ with singular $i \times i$ leading principal submatrix for $i = 1, \dots, n$, due to Theorem 10.2. This makes up at most $\sum_{i=1}^{n} iq^{2n-2} = q^{2n-2}n(n+1)/2$ submatrices which are not strongly nonsingular in the set of all $q^{2n-1}$ Toeplitz matrices in $\mathbb{Z}_q^{n\times n}$. □

According to the latter results as well as the results of our experimental tests for nonsingularity of random integer Toeplitz and general matrices in $\mathbb{Z}_{q^w}^{n\times n}$ for $q = 2$, $w \le 20$, and $n \le 100$ presented in Section 10.4, the transition to the rings $\mathbb{Z}_{p^w}$ for larger $p^w$ keeps the chances for the degeneration quite remote on the average.

## 10.3 Additive perturbations counter degeneracies

Suppose we have the rare case where, for a fixed triple of $\lambda$, $M$ and $p$, one cannot perform generalized lifting by computing within the word size precision because $(\det M) \bmod p^v = 0$ for all $v \le \lambda$, e.g., wishing to stay with $p = 2$. Suppose we prefer not to change $p$. Should we necessarily give up lifting? Not right away, because we may usually reduce the solution of the linear system $M\mathbf{x} = \mathbf{b}$ to solving a linear system with the coefficient matrix of the form

$$M_i = M - U_i V_i. \tag{10.6}$$

Here $U_i$ in $\mathbb{Z}_a^{n\times i}$ and $V_i$ in $\mathbb{Z}_b^{i\times n}$ are random general (or random Toeplitz) matrices for two integers

$$b \ge 2n^2 \log(n|M|), \quad a \ge 21n^2 b, \tag{10.7}$$

and a relatively small $i = O(1)$.

Namely, we fix two positive integers $i_+$ and $j_+$ and recursively apply our lifting initialization algorithms to the matrices $M_{i,j} = M - U_{i,j}V_{i,j}$ for random matrices $U_{i,j}$ and $V_{i,j}$ for $i = 1$, $j = 1, \dots, j_+$; $i = 2$, $j = 1, \dots, j_+$; $\dots$, and so on, until we either yield the desired initialization for $M_i = M_{i,j}$ and some $i \le i_+$, $j \le j_+$ by computing with the word precision $\lambda$ or reach $i = i_+ + 1$. In the latter case, the algorithm outputs FAILURE. In the former case we compute

the vector $M^{-1}\mathbf{b}$ for a fixed integer vector $\mathbf{b}$ based on (10.6) and the Sherman–Morrison–Woodbury formula [GL96, page 50],

$$M^{-1} = (M_i + U_i V_i)^{-1} = M_i^{-1} - M_i^{-1} U_i (I_i + V_i M_i^{-1} U_i)^{-1} V_i M_i^{-1}. \quad (10.8)$$

The formula holds provided that the matrices $M_i$, $M$, and

$$W_i = I_i + V_i M_i^{-1} U_i$$

are nonsingular for the pair of $n \times i$ matrices $U_i$ and $V_i^T$. We refer to these computations as **Algorithm 10.1**.

We first apply the generalized lifting to the vector $qM_i^{-1}\mathbf{b}$ and to every column of the $n \times i$ matrix $qM_i^{-1}U_i$ to obtain these vector and columns in $\mathbb{Z}_{qs^h}$. Then we compute the vector

$$qM^{-1}\mathbf{b} \bmod (qs^h) = ((I - qM_i^{-1}U_i(qI + qV_iM_i^{-1}U_i)^{-1}V_i)qM_i^{-1}\mathbf{b}) \bmod (qs^h),$$

and reconstruct the rational vector $M^{-1}\mathbf{b}$.

For general matrices $M$, the algorithm is likely to succeed already for reasonably small integers $i_+$ and $j_+$ due to the two following theorems in [EGV00], which relate this likelihood to the choice of the bounds $i_+$ and $j_+$.

**Theorem 10.6.** *[EGV00, Theorem 3.8]. For two positive integers $i$ and $n$, $i < n$, a nonsingular matrix $M$ in $\mathbb{Z}^{n\times n}$, and sufficiently large integers $a$ and $b$ satisfying (10.7), let $U_i = U_{i,j}$ and $V_i^T = V_{i,j}^T$ denote the pairs of random matrices in $\mathbb{Z}_a^{n\times i}$ for $j = 1, 2, \dots, 15$, and in $\mathbb{Z}_b^{n\times i}$ for $j = 16$, and let the matrices $M_i = M_{i,j}$ be defined by (10.6). Then with a probability of at least $1/2$, we have $s_{n-i}(M) = \gcd(s_n(M), \gcd_{j=1}^{16}(s_n(M_{i,j})))$. To increase the probability bound above $1 - \varepsilon$ for a fixed positive $\varepsilon$, it is sufficient to include $j_+$ matrices $M_{i,j}$, $j = 1, \dots, j_+$, for every $i$ and for a sufficiently large $j_+$ in $O(\log(1/\varepsilon))$.*

**Theorem 10.7.** *[EGV00, Theorem 6.2]. For a fixed pair of integers $\lambda > 0$ and $\eta$, let the entries of an $n \times n$ matrix $M$ be independently sampled under the uniform probability distribution in a set of integers $\eta, \eta+1, \dots, \eta+\lambda-1$. Then* Probability$(s_{n-j}(M) > 1) \le \lambda^{-n} + 9(\frac{2}{3})^{j-1} + \frac{n^3}{\lambda^{j-1}}$.

Due to Theorems 10.6 and 10.7 (and also according to the well known statistics), we have with a high probability that

$$\gcd\left(s_n(M), \gcd_{j=1}^{j_+}(s_n(M_{i,j}))\right) = 1$$

for a random $n \times n$ integer matrix $M$, the matrices $M_{i,j}$ defined above, some $i \le i_+$, and reasonably small integers $i_+$ and $j_+$. In fact we just need a weaker property that $g$ is coprime with a fixed prime $p$, and this property has been statistically observed in our experiments with random Toeplitz matrices for $p = 2$ (see the next subsection).

## 10.4 Experimental computations: how frequently is a random integer Toeplitz matrix non-singular modulo a fixed power of two?

In out tests we have randomly generated an $n \times n$ Toeplitz matrix $M = (t_{i-j})_{i,j}$. Its entries $t_{1-n}, \ldots, t_{n-1}$ have been chosen independently of each other under the uniform random distribution on $\mathbb{Z}_q$ for $q = 2^g$ and for a positive integer $g$. The first column in each of Tables 10.1–10.3 shows how frequently in our tests a random $n \times n$ integer Toeplitz matrix $M$ was nonsingular in $\mathbb{Z}_q$.

Whenever the test showed singularity, we repeated the test recursively (up to at most four times), each time adding the outer product of two random vectors to the input matrix. The *(1+i)*-th column of each table, for $i = 0, 1, 2, 3, 4$, shows for how many out of 100,000 samples the results were positive for the matrices $M - U_j V_j^T$, for some $j \le i$ where $U_j, V_j^T \in \mathbb{Z}_q^{n \times l}$, $M \in \mathbb{Z}_q^{n \times n}$, $q = 2^g$. These data should motivate using Algorithm 10.1 for smaller $i_+$ and $j_+$. For comparsion, we include Table 10.4 with similar statistics for general matrices (although without small rank perturbations).

Table 10.1: Number of times the matrix $M + A_i$ for a random $20 \times 20$ Toeplitz matrix $M$ and a random $50 \times 50$ matrix $A$ of rank at most $i$ is nonsingular in the ring $\mathbb{Z}_q$ for $q = 2^g$ out of 100,000 samples

| $g$ \ $i$ | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 1 | 50173 | 59450 | 66672 | 72514 | 77452 |
| 2 | 68814 | 80808 | 87785 | 92256 | 95133 |
| 3 | 82971 | 92311 | 96197 | 98164 | 99136 |
| 4 | 90559 | 96899 | 98862 | 99567 | 99852 |
| 5 | 95079 | 98809 | 99671 | 99907 | 99973 |
| 6 | 97333 | 99557 | 99907 | 99981 | 99997 |
| 7 | 98643 | 99859 | 99973 | 99998 | 100000 |
| 8 | 99302 | 99948 | 99993 | 99999 | 100000 |
| 9 | 99639 | 99983 | 100000 | 100000 | 100000 |
| 10 | 99816 | 99997 | 100000 | 100000 | 100000 |
| 11 | 99903 | 99999 | 100000 | 100000 | 100000 |
| 12 | 99955 | 100000 | 100000 | 100000 | 100000 |

Table 10.2: Number of times the matrix $M + A_i$ for a random $50 \times 50$ Toeplitz matrix $M$ and a random $50 \times 50$ matrix $A$ of rank at most $i$ is nonsingular in the ring $\mathbb{Z}_q$ for $q = 2^g$ out of 100,000 samples

| $g$ \ $i$ | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 1 | 50054 | 59383 | 66661 | 72665 | 77581 |
| 2 | 68781 | 80792 | 87812 | 92341 | 95151 |
| 3 | 82842 | 92263 | 96282 | 98203 | 99139 |
| 4 | 90507 | 96868 | 98877 | 99589 | 99844 |
| 5 | 95132 | 98846 | 99695 | 99915 | 99976 |
| 6 | 97440 | 99597 | 99912 | 99981 | 99994 |
| 7 | 98667 | 99857 | 99972 | 99994 | 99998 |
| 8 | 99315 | 99953 | 99989 | 99997 | 99999 |
| 9 | 99653 | 99985 | 100000 | 100000 | 100000 |
| 10 | 99829 | 99997 | 100000 | 100000 | 100000 |
| 11 | 99917 | 99999 | 100000 | 100000 | 100000 |
| 12 | 99967 | 100000 | 100000 | 100000 | 100000 |

Table 10.3: Number of times the matrix $M + A_i$ for a random $100 \times 100$ Toeplitz matrix $M$ and a random $100 \times 100$ matrix $A$ of rank at most $i$ is nonsingular in $\mathbb{Z}_q$ for $q = 2^g$ out of 100,000 samples

| $g$ \ $i$ | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 1 | 50170 | 59672 | 66652 | 72460 | 77368 |
| 2 | 68969 | 80960 | 87833 | 92188 | 95130 |
| 3 | 82799 | 92261 | 96240 | 98128 | 99122 |
| 4 | 90498 | 96935 | 98884 | 99570 | 99845 |
| 5 | 94975 | 98837 | 99662 | 99893 | 99971 |
| 6 | 97255 | 99547 | 99898 | 99970 | 99991 |
| 7 | 98591 | 99827 | 99966 | 99994 | 99998 |
| 8 | 99249 | 99931 | 99989 | 99998 | 99998 |
| 9 | 99616 | 99976 | 99997 | 100000 | 100000 |
| 10 | 99804 | 99994 | 100000 | 100000 | 100000 |
| 11 | 99898 | 99998 | 100000 | 100000 | 100000 |
| 12 | 99948 | 100000 | 100000 | 100000 | 100000 |

Table 10.4: Number of times a random $n \times n$ general matrix $M$ is nonsingular in the ring $\mathbb{Z}_q$ out of $100,000$ samples for $q = 2^g$

| | $n = 5$ | $n = 10$ | $n = 50$ | $n = 100$ |
|---|---|---|---|---|
| $g = 0$ | 29,986 | 28,781 | 28,940 | 28,781 |
| $g = 1$ | 58,637 | 57,679 | 57,884 | 57,782 |
| $g = 2$ | 77,650 | 76,817 | 77,047 | 77,104 |
| $g = 3$ | 88,399 | 87,916 | 88,000 | 88,080 |
| $g = 4$ | 94,102 | 93,888 | 93,943 | 93,921 |
| $g = 5$ | 97,046 | 96,911 | 96,963 | 96,937 |
| $g = 6$ | 98,519 | 98,414 | 98,483 | 98,452 |
| $g = 7$ | 99,245 | 99,180 | 99,212 | 99,235 |
| $g = 8$ | 99,634 | 99,598 | 99,590 | 99,620 |
| $g = 9$ | 99,820 | 99,791 | 99,783 | 99,806 |
| $g = 10$ | 99,911 | 99,894 | 99,892 | 99,899 |
| $g = 11$ | 99,956 | 99,957 | 99,950 | 99,953 |
| $g = 12$ | 99,977 | 99,977 | 99,978 | 99,980 |
| $g = 13$ | 99,985 | 99,992 | 99,991 | 99,992 |
| $g = 14$ | 99,992 | 99,996 | 99,993 | 99,995 |
| $g = 15$ | 99,993 | 99,997 | 99,996 | 99,998 |
| $g = 16$ | 99,995 | 99,999 | 99,999 | 99,998 |
| $g = 17$ | 99,998 | 99,999 | 99,999 | 99,998 |
| $g = 18$ | 99,999 | 100,000 | 99,999 | 99,999 |
| $g = 19$ | 99,999 | 100,000 | 100,000 | 100,000 |
| $g = 20$ | 99,999 | 100,000 | 100,000 | 100,000 |

### Analysis of the results of the experiments

For fixed $q$ and $n$, we assume that $M$ is singular over $\mathbb{Z}_q$ with a probability $p$. Next we estimate $p$. Let $x$ be a random variable such that

$$x = \begin{cases} 1, & \det M = 0 \bmod q; \\ 0, & \det M \neq 0 \bmod q. \end{cases}$$

Let $x_1, \ldots, x_m$ be the observed values of $x$. By the Central Limit Theorem,

$$\lim_{m\to\infty} \frac{(x_1 + \ldots + x_m) - mp}{\sqrt{mp(1-p)}} = N(0,1)$$

where $N(0,1)$ is the standard normal probability distribution. Therefore, a confidence interval of probability $1-\alpha$ for $p$ is

$$\left(\bar{x} - Z_{\alpha/2}\sqrt{\bar{x}(1-\bar{x})/m},\ \bar{x} + Z_{\alpha/2}\sqrt{\bar{x}(1-\bar{x})/m}\right)$$

where $\bar{x} = \frac{1}{m}(x_1 + \ldots + x_m)$, $Z_\alpha$ is defined by Probability$(N(0,1) > Z_\alpha) = \alpha$.

**Example 10.1.** *For $g = 8, n = 50$, we are "99.9%" sure that*

- *Probability(Toeplitz matrix $M$ is non-singular)* $= 0.993 \pm 0.001$*;*
- *Probability(Toeplitz matrix $M$ is strongly non-singular)* $= 0.731 \pm 0.005$*;*
- *Probability(general matrix $M$ is non-singular)* $= 0.992 \pm 0.001$*;*
- *Probability(general matrix $M$ is strongly non-singular)* $= 0.688 \pm 0.005$*.*

# 11 Demonstration of algorithms with examples

Let us demonstrate the work of Algorithms 3.1 and 10.1 with simple examples.

**Example 11.1.** $M = \left(\begin{smallmatrix} 2 & 1 \\ 3 & 2 \end{smallmatrix}\right)$, $\mathbf{b} = \binom{3}{4}$. *So* $\mathbf{x} = \binom{2}{-1}$. *By applying Algorithm 3.1 for $q = 1$, $s = 2$, $\mathbf{r}^{(0)} = \mathbf{b}$, 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* $\mathbf{x}^{(3)} = 2\mathbf{x} \bmod 8 = \binom{0}{1} + 2\binom{1}{1} + 4\binom{0}{1}$.

**Example 11.2.** $M = \left(\begin{smallmatrix} 4 & 1 \\ 6 & 2 \end{smallmatrix}\right)$, $\mathbf{b} = \binom{3}{4}$, *so* $\mathbf{x} = \binom{1}{-1}$.

*a) By applying Algorithm 3.1 for $q = s = 2$, $\mathbf{r}^{(0)} = \mathbf{b}$, 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* $\mathbf{x}^{(3)} = 2\mathbf{x} \bmod 8 = \binom{0}{2} + 2\binom{1}{2} + 4\binom{2}{2}$, $(M\mathbf{x}^{(h)} - 2\mathbf{b}) \bmod 2^{h+1} = 0$ *for $h = 1, 2, 3$.*

*b) Alternatively, we observe that* $s_2(M) = 2$, $s_1(M) = 1$ *and apply Algorithm 10.1 to* $M_1 = M - U_1V_1$, $U_1 = V_1^T = \binom{1}{1}$, *and* $\mathbf{b} = \binom{3}{4}$, *so that* $M_1 = \left(\begin{smallmatrix} 3 & 0 \\ 5 & 1 \end{smallmatrix}\right)$, $M_1^{-1} = \left(\begin{smallmatrix} 1/3 & 0 \\ -5/3 & 1 \end{smallmatrix}\right)$. *Due to the Sherman–Morrison–Woodbury formula (10.8), this reduces the computation of* $\mathbf{x}$ *to the triple application of Algorithm 3.1 for* $q = 1$, $s = 2$ *with the right-hand-side vectors* $\mathbf{b} = \binom{3}{4}$, $\mathbf{b}^{(1)} = \binom{1}{1}$, *and* $\mathbf{b}^{(2)} = (1/3)\left(\begin{smallmatrix} 1 & 1 \\ 1 & 1 \end{smallmatrix}\right) M_1^{-1}\binom{3}{4}$, *respectively. We obtain* $M_1^{-1}\binom{3}{4} = \binom{1}{-1}$, $M_1^{-1}\binom{1}{1} = \binom{1/3}{-2/3}$. *Therefore,* $\mathbf{b}^{(2)} = \mathbf{0}$, $M_1^{-1}\mathbf{b}^{(2)} = \mathbf{0}$, $M^{-1}\mathbf{b} = M_1^{-1}\mathbf{b} = \binom{1}{-1}$.

**Example 11.3.** $M = \left(\begin{smallmatrix} 32 & 2 \\ 48 & 4 \end{smallmatrix}\right)$, $\mathbf{b} = \binom{24}{32}$. *So,* $\mathbf{x} = \binom{1}{-4}$, $s_2(M) = 32$, $s_1(M) = 2$. *We may*

*a) apply Algorithm 3.1 to* $M$ *and* $\mathbf{b}$ *for* $q = 3, s = 2$, *or*

*b) apply Algorithm 10.1 to* $M_1 = M - U_1V_1$, $U_1 = \binom{1}{1}$, $V_1^T = \binom{2}{2}$, $M_1 = \left(\begin{smallmatrix} 30 & 0 \\ 46 & 2 \end{smallmatrix}\right)$. *For solving the equations* $M_1^{-1}\mathbf{b}^{(i)}, i = 1, 2, 3$ *(cf. Example 11.2 b), apply Algorithm 3.1 for* $q = s = 2$.

# 12 Some extensions and a discussion

Let us further comment on the extensions of our algorithm cited in the introduction.

## 12.1 Extension of the class of structured matrices

Our lifting algorithms can be applied effectively to any nonsingular integer input matrix provided its precomputed inverse and itself can be multiplied by vectors fast. Toeplitz/Hankel-like matrices in Section 8.1 seem to be the most important example. See other examples in [P01, Section 5.7], Chen et al 2002 [CEKSTV02].

The restriction on the integrality of the input excludes Cauchy-like input matrices. Scaling turns them into integer matrices but generally blows up the magnitude of the input entries; then our approach becomes ineffective. As a potential way out, we may apply the displacement transformations in [P90] to transform the Cauchy-like matrices into Toeplitz/Hankel-like matrices and then truncate the real or complex entries and scale the matrices to yield the integrality of the input.

A relevant open problem is the extension of our probabilistic study of degeneration from Toeplitz to Toeplitz-like and other structured matrices.

## 12.2 Extension to computations with singular matrices and polynomials

Our algorithms can be extended to the randomized computation of the rank of a singular matrix $M$, solving a (consistent) singular linear system of equations $M\mathbf{x} = \mathbf{b}$, and finding a vector from (or a generator for a basis for) the

null space of $M$. The solution of these problems enables the extension of our randomized algorithms and complexity estimates to computing the gcds, lcms, Padé approximations, and rational interpolation functions where the input is given by univariate polynomials with integer coefficients [BP94], [P96], [P01].

The listed matrix problems are easily reduced to the inversion of a nonsingular submatrix of $M$ of the largest size (see, e.g. [BP94, Section 2.2]). To find this submatrix, we first preprocess the matrix $M$ so that the resulting matrix $\tilde{M}$ is again an integer Toeplitz/Hankel-like matrix of the same rank $\rho$ as $M$ and is likely to have generic rank profile, that is, to have its $k \times k$ leading principal submatrices $\tilde{M}^{(k)}$ nonsingular for $k = 1, \ldots, \rho$. The randomized complexity estimates for the preprocessing are of Monte Carlo type, that is, they do not cover the verification of the correctness of the output. We recall this preprocessing at the end of the subsection.

Next we compute $\rho$ and invert $\tilde{M}^{(\rho)}$. If the preprocessed matrix $\tilde{M}$ has generic rank profile, we may compute its rank $\rho$ by applying the binary search. In each search step we apply our lifting algorithm to invert the matrix $\tilde{M}^{(k)}$ for the current $k$ or to solve the linear system $\tilde{M}^{(k)}\mathbf{x} = \mathbf{b}^{(k)}$ where $\mathbf{b}^{(k)}$ is a random vector. If we fail, $\tilde{M}^{(k)}$ is likely to be a singular matrix and we decrease $k$; otherwise $\tilde{M}^{(k)}$ is likely to be nonsingular and we increase $k$. Guided by our study in Section 10, we select a random prime $p$ and $a$ positive integer $w$ and perform our computations in $\mathbb{Z}_{p^w}$. As soon as we compute the rank $\rho$, we apply our lifting algorithm to invert the matrix $\tilde{M}^{(\rho)}$.

Alternatively, we may apply the MBA algorithm to the matrix $\tilde{M}$ to yield both $\rho$ and $(\tilde{M}^{\rho})^{-1}$; then again we perform our computations in $\mathbb{Z}_{p^w}$ choosing a larger random prime $p$ to decrease the chances for degeneration. Finally we lift the matrix $(\tilde{M}^{(\rho)})^{-1}$ from $\mathbb{Z}_{p^w}$ to the desired level.

Both approaches use by the factor of $\log \rho$ more word operations than we need for solving a nonsingular Toeplitz linear system $M\mathbf{x} = \mathbf{b}$. In both approaches, by choosing $w = 1$ and sampling a random prime $p$ from a moderately large range, we may both make degeneration unlikely and yield $\rho$ at a low randomized bit operation cost.

As we mentioned in the introduction, our randomized complexity bounds are of Las Vegas type for system solving and computing a vector from the null space, but not so for the computation of the rank and a null space basis, testing the consistency of a linear system, and the cited problems of polynomial computations; thus for the latter problems our complexity estimates are of the Monte Carlo type.

Let us next specify the preprocessing policy due to Kaltofen and Saunders 1991 [KS91]. We assume a Toeplitz/Hankel-like input matrix $M$ and compute the matrix $\tilde{M} = UML$ where $U^T$ and $L$ are two unit lower triangular Toeplitz matrices, each defined by the $n-1$ entries in its first column. We sample these $2n-2$ entries at random (independently of each other) from a set $S$ of $|S|$ distinct integers (say, $S = \{x, x = 1-|S|/2, \ldots, |S|/2\}$ for even $|S|$) and assume the uniform probability distribution on $S$. Clearly, rank $\tilde{M}$=rank $M$, $\tilde{M}$ is still an integer and Toeplitz/Hankel-like matrix, and our computational problems

for the input $M$ are reduced to those for $\tilde{M}$. In virtue of a theorem in [KS91], based on the cited lemma in [DL78], the matrix $\tilde{M}$ has generic rank profile with a probability of at least $1 - \rho^2/|S|$.

## 12.3 Extension to computing the determinant and Smith's factors

The MBA algorithm outputs the determinant of an input matrix $M$ as by-product and also certifies its correctness at a low cost. Since the resultant of two univariate polynomials is a Toeplitz-like matrix, we yield the resultant as a special case. The computation can be performed with a lower precision modulo several primes, and the determinant can be recovered with the CRA. Alternatively, one may first perform the MBA algorithm in $\mathbb{Z}_p$ for a random moderately large prime $p$ and then lift the entire construction and easily yield the integer determinant from its value in $\mathbb{Z}_{p^w}$ for a sufficiently large $w$ (compare [P00]). The application of the MBA algorithm implies the increase of the overall complexity bounds by the factor of $\log n$ versus our solution of nonsingular linear system with lifting. To avoid this increase and also to compute the Smith factors of $M$, let us recall the randomized Monte Carlo approach proposed in [EGV00] for general input matrices and adapt it to the Toeplitz/Hankel-like case.

In [EGV00], computing Smith's factors and the determinant of a general integer matrix $M$ is reduced to solving a small number of linear systems $M\mathbf{x}_k = \mathbf{b}_k$ for random vectors $\mathbf{b}_k$. The reduction is immediately extended to a Toeplitz/ Hankel-like matrix $M$. Here are the resulting bit cost estimates.

**Theorem 12.1.** *Allow output errors with a probability of at most $\nu > 0$, and also allow an additional factor of $\log(1/\nu)$ in all asymptotic estimates in Theorems 8.1 for the numbers of random bits and bit operations. Then the resulting (increased) estimates apply to the computation of Smith's leading factor $s_n(M)$ of an $n \times n$ integer Toeplitz/Hankel-like matrix $M$; the estimates do not including the correctness verification cost. Up to increasing the bit operation complexity bounds by the factor of $k$ and sampling $O(kn \log n)$ additional random bits, the same bounds cover the computation of the next $k$ distinct leading Smith's factors of $M$; with the $l$-fold increase, the bit operation cost bounds of Theorem 8.1 cover the computation of all Smith's factors of $M$ and $\det M$ (without correction verification) where $l$ is the overall number of distinct Smith's factors, $l \le \sqrt{\log\det |M|} \le \sqrt{n \log |M|}$ for every matrix $M \in \mathbb{Z}^{n\times n}$.*

The theorem is supported by the algorithm in [EGV00] complemented by the smaller complexity bounds for solving Toeplitz (rather than general) linear systems, given by Theorem 8.1. We recall a basic lemma in [EGV00].

**Lemma 12.1.** *Let $\mathbf{b}$ be a random vector in $\mathbb{Z}^n$. Then $\delta$ in (5.2) divides $s_n = s_n(M)$, and furthermore, for any prime $p$, we have*

$$\text{Probability}(\text{ord}_p(\delta) < \text{ord}_p(s_n)) \le \max\{1/\eta, 1/p\}.$$

*Proof.* The lemma follows from Theorem 2 in [ABM99], but here is a simple direct proof. We have $x_i = \sum_j (-1)^{i+j} d_{i,j} b_j / \det M$, $s_n = |(\det M)/d|$, $d = \gcd(d_{i,j})_{i,j}$ for $d_{i,j}$ in Definition 2.1, and $\mathbf{b} = (b_j)_{j=1}^n$. Write $h_{i,j} = \mathrm{ord}_p(d_{i,j})$, $h = \mathrm{ord}_p(d) = \min_{i,j} d_{i,j}$. We have $h = \mathrm{ord}_p(d_{u,v})$ for some $u, v$; w.l.o.g., let $u = v = 0$. Furthermore, write $\bar{d}_{i,j} = d_{i,j}/d$ for all $i$ and $j$. Then it follows that $s_n x_0^{(k)} = \bar{d}_{0,0} b_0^{(k)} + r$, where $r = \sum_{j=1}^{n-1} (-1)^j \bar{d}_{0,j} b_j^{(k)} \in \mathbb{Z}$. It remains to recall that $\mathrm{ord}_p(\bar{d}_{0,0}) = 0$ and $b_0^{(k)}$ is randomly sampled from $\mathbb{Z}_\eta$, independently of $\bar{d}_{0,0}$. □

## 12.4 The extensions of the domain for the input values

The extensions of the input domain to the matrices with complex (Gaussian) integer, rational, or polynomial entries, and to block matrices have already been cited in the introduction.

## 12.5 Further topics

Many aspects of the cited extensions must be elaborated upon, e.g., theorems on and statistics of degeneration. Theoretical support for our statistical data in Section 10.4 is another interesting technical challenge. Even more important is to refine our codes for our algorithms in the rings $\mathbb{Z}_{qs}$ for $s = 2^u$ and $q = 2^v$ and to experiment with the parameters involved, e.g., $a$ and $m$ in our initialization algorithms in Sections 9.1 and 9.2. The next important direction is the extension of these codes to computing polynomial gcd, lcm, etc. and their experimental comparison with the alternative computations in $\mathbb{Z}_p$ for larger random primes $p$.

Should we expect to see a further asymptotic decrease of our bit complexity estimates? The factor of $m(n)$ in them comes from our basic operation of Toeplitz matrix-by-vector multiplication or equivalently polynomial multiplication. It is unlikely that any efficient algebraic computation scheme for our tasks could dispense with this operation. (Try to imagine such a scheme, e.g., for polynomial gcd.) This informal argument suggests that improvement of our bit complexity bounds by the factor of $m(n)/n$ is unlikely. Our basic operation can be viewed as multiplication of polynomials with bounded integer coefficients, and therefore the binary segmentation technique of Fischer and Paterson 1974 [FP74] (cf. [BP94, Section 3.9]) could yield theoretical acceleration by the factor of $(\log\log n) \log\log\log n$. The resulting complexity bound in $O(n\mu(n \log n))$, however, is not practically attractive unless $n$ is huge. Indeed the overhead constant $C_{ss}$ is large, whereas with $C_{class}$ and $C_k$ in (1.2) the overall bit complexity bounds become as large as $n^\alpha$ for $\alpha > 2.5$.

# References


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

[B85] J. R. Bunch, Stability of Methods for Solving Toeplitz Systems of Equations, *SIAM J. of Scientific and Statistical Computing*, **6(2)**, 349–364, 1985.

[B03] D. J. Bernstein, Fast Multiplication and Its Applications, preprint, 2003. Availabe 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.

[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.

[BMK87] R. P. Brent, B. D. McKay, Determinants and Ranks of Random Matrices over $\mathbb{Z}_m$, *Discrete Math.*, **66**, 35–49, 1987.

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

[CEKSTV02] L. Chen, W. Eberly, E. Kaltofen, B.D. Saunders, W. J. Turner, G. Villard, Efficient Matrix Preconditioners for Black Box Linear Algebra, *Linear Algebra and Its Applications*, **343–344**, 119–146, 2002.

[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. of Symbolic Computation*, **9(3)**, 251–280, 1990.

[D60] D. E. Daykin, Distribution of Bordered Persymmetric Matrices in a Finite Field. *J. Reine und Angewandte Math.,* **203**, 47–54, 1960.

[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. International Symp. on Algebraic and Symbolic Computation (ISSAC'04)*, 119–126, ACM Press, New York, 2004.

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

[DSV01] J.-G. Dumas, B. D. Saunders, G. Villard, On Efficient Sparse Integer Matrix Smith Form Computations, *J. of Symbolic Computation*, **32**, 71–99, 2001.

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

[EPY98] I. Z. Emiris, V. Y. Pan, Y. Yu, Modular Arithmetic for Linear Algebra Computations in the Real Field, *J. of Symbolic Computation*, **26**, 71–87, 1998.

[FP74] M. J. Fischer, M. S. Paterson, String Matching and Other Problems, *SIAM-AMS Proceedings*, **7**, 113–125, 1974.

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

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

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

[H96] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM Publications, Philadelphia, PA, 1996.

[K90] A. A. Karatsuba, The Distribution of Prime Numbers, *Russian Math. Surveys*, **45**, 99–171, 1990.

[K98] D. E. Knuth, *The Art of Computer Programming. Vol. 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.

[KL96] E. Kaltofen, A. Lobo, On Rank Properties of Toeplitz Matrices over Finite Fields, *Proceedings of International Symposium on Symbolic and Algebraic Computation (ISSAC'96)*, 241–249, ACM Press, New York, 1996.

[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 Publications, Philadelphia, PA, 1999.

[M74] M. Morf, Fast Algorithms for Multivariable Systems, Ph.D. Thesis, *Department of Electrical Engineering, Stanford University,* Stanford, 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. Monahan, Maximal Quotient Rational Reconstruction: an Almost Optimal Algorithm for Rational Reconstruction, *Proc. International Symp. on Algebraic and Symbolic Computation (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.

[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.

[P90] V. Y. Pan, On Computations with Dense Structured Matrices, *Mathematics of Computation*, **55(191)**, 179–190, 1990.

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

[P92a] V. Y. Pan, Can We Utilize the Cancellation of the Most Significant Digits?, Tech. Report TR–92–061, *The International Computer Science Institute*, Berkeley, California, 1992.

[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, Can We Optimize Toeplitz/Hankel Computations? *Proc. of the Fifth International Workshop on Computer Algebra in Scientific Computing (CASC'02)*, Yalta, Crimea, Sept. 2002 (E. W. Mayr, V. G. Ganzha, E. V. Vorozhtzov, Editors), 253–264, *Technische Universität München*, Germany, 2002.

[P02a] V. Y. Pan, Nearly Optimal Toeplitz/Hankel Computations, Technical Reports TR 2002 001 and 2002 017, *Ph.D. Program in Computer Science, The Graduate Center of the City University of New York*, New York, 2002.

[P04] V. Y. Pan, Superfast Algorithms for Singular Integer Toeplitz/ Hankel-like Matrices, Technical Reports 2002 002, 2003 004 and 2004, *Ph.D. Program in Computer Science, The Graduate Center of the City University of New York*, New York, 2002/2003/2004.

[PW02] V. Y. Pan, X. Wang, Acceleration of Euclidean Algorithm and Extensions, *Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC'02)*, (Teo Mora editor), 207–213, ACM Press, New York, 2002.

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

[RS62] J. B. Rosser, L. Schoenfeld, Approximate Formulas of Some Functions of Prime Numbers, *Illinois J. of Math.*, **6**, 64–94, 1962.

[S80] R. D. Skeel, Iterative Refinement Implies Numerical Stability for Gaussian Elimination, *Math. of Computation*, **35**, 817–832, 1980.

[S80a] J. T. Schwartz, Fast Probabilistic Algorithms for Verification of Polynomial Identities, *Journal of ACM*, **27(4)**, 701–717, 1980.

[S86] A. Schrijver, *Theory of Linear and Integer Programming*, Wiley, New York, 1986.

[S03] A. Storjohann, High Order Lifting and Integrality Certificaiton, *J. of Symbolic Computation*, **36 (3–4)**, 613–648, 2003.

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

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

[Z79] R. E. Zippel, Probabilistic Algorithms for Sparse Polynomials, *Proceedings of EUROSAM'79, Lecture Notes in Computer Science*, **72**, 216–226, Springer, Berlin, 1979.

[Z93] R. Zippel, *Effective Polynomial Computation*, Kluwer, Boston, 1993.