ON THE COMPLEXITY OF COMPUTING DETERMINANTS*
(Extended abstract)

ERICH KALTOFEN[1] and GILLES VILLARD[2]

[1]Department of Mathematics, North Carolina State University
Raleigh, North Carolina 27695-8205
kaltofen@math.ncsu.edu, http://www.kaltofen.net

[2]Laboratoire LIP, École Normale Supérieure Lyon
46, Allée d'Italie, 69364 Lyon Cedex 07, France
Gilles.Villard@ens-lyon.fr, http://www.ens-lyon.fr/~gvillard

1 Introduction

The computation of the determinant of an $n \times n$ matrix A of numbers or polynomials is a challenge for both numerical and symbolic methods. Numerical methods, such as Clarkson's algorithm [10, 7] for the sign of the determinant must deal with conditionedness that determines the number of mantissa bits necessary for obtaining a correct sign. Symbolic algorithms that are based on Chinese remaindering [6, 17, Chapter 5.5] must deal with the fact that the length of the determinant in the worst case grows linearly in the dimension of the matrix. Hence the number of modular operations is n times the number of arithmetic operations in a given algorithm. Hensel lifting combined with rational number recovery [14, 1] has cubic bit complexity in n, but the algorithm can only determine a factor of the determinant, namely the largest invariant factor. If the matrix is similar to a multiple of the identity matrix, the running time is again that of Chinese remaindering.

The techniques developed in [32] for computing the characteristic polynomial of a sparse matrix lead to a speedup for the general, dense determinant problem. For integer matrices, the bit complexity was shown [16] to be $n^{3.5+o(1)}(\log \|A\|)^{2.5+o(1)}$, where $\log \|A\|$ measures the length of the entries in A and the exponent adjustment by "$+o(1)$" captures missing log factors ("soft-O"). The algorithms of [32, 16] are randomized of the Monte Carlo kind—always fast, probably correct—and can be further speeded by a Strassen/Coppersmith-Winograd sub-cubic time matrix multiplication algo-

*This material is based on work supported in part by the National Science Foundation under Grants Nos. CCR-9988177, DMS-9977392 and INT-9726763 (Kaltofen).
To appear in the *Proc. Fifth Asian Symposium on Computer Mathematics ASCM 2001* at Ehime University, Japan Sep 26–28, 2001, World Scientific Publishing Company, Singapore.

rithm. Note that Clarkson's algorithm and its new variants are in the worst case quartic in n.

An entirely different method, based on an algorithm in [33], was first described for dense matrices with polynomial entries [22]. For integer matrices the resulting randomized algorithm is of the Las Vegas kind—always correct, probably fast—and has worst case bit complexity $(n^{3.5} \log \|A\|)^{1+o(1)}$ and again can be speeded with sub-cubic time matrix multiplication. We give a description of this algorithm in Section 2 below. That algorithm was originally put to a different use, namely that of computing the characteristic polynomial and adjoint of a matrix without divisions, counting additions, subtractions, and multiplications in the commutative ring of entries.

By considering a bilinear map with two blocks of vectors rather than a single pair of vectors, Wiedemann's algorithm can be accelerated [11, 23, 30, 31]. This technique can be applied to our fast determinant algorithm, and results in a worst case bit complexity of $(n^{3+1/3} \log \|A\|)^{1+o(1)}$, again based on standard cubic time matrix multiplication. We discuss this modification and its mathematical justification in Section 3. Serendipitously, blocking can be applied to our original 1992 division-free algorithm, and a similar improvement of the division-free complexity of the determinant is obtained (see Section 4), thus changing the status of a problem that has now been open for over 9 years.

In this extended abstract we do not consider the use of fast matrix multiplication algorithms. By using the algorithms in [13, 12] the bit complexity for the determinant of an $n \times n$ matrix with integer entries can be reduced to $O(n^{2.698} (\log \|A\|)^{1+o(1)})$, and the division-free complexity of the determinant and adjoint of a matrix over a commutative ring to $O(n^{2.698})$ ring operations. These exponents have purely mathematical interest and no impact for the computation of a determinant on a computer. We shall also hide the exponents of the $\log n$ and loglog $\|A\|$ factors in the "$+o(1)$" of the exponents. In Section 2 we shall address the impact of those factors on the practicality of our methods. In general, the precise exponents of these logarithms are dependent on the actual computational model, such as multi-tape Turing machine, logarithmic random access machine, hierarchical memory machine, etc.

2 Wiedemann's algorithm with baby steps/giant steps

Already in his original paper Wiedemann proposes a method for computing the determinant of a sparse matrix [33]. The algorithm is based on a sequence of bilinear projections of the powers of the input matrix. For vectors $u, v \in \mathbb{K}^n$, where $\mathbb{K}$ is an arbitrary field, and the input matrix $A \in \mathbb{K}^{n \times n}$ consider the sequence of field elements

$$a_0 = u^{Tr}v,\ a_1 = u^{Tr}Av,\ a_2 = u^{Tr}A^2v,\ a_3 = u^{Tr}A^3v,\ \ldots \tag{1}$$

The minimal polynomial of A, denote by f^A, linearly generates $\{a_i\}_{i=0,1,\ldots}$. By the Berlekamp/Massey algorithm we can compute in $n^{1+o(1)}$ arithmetic operations a minimal linear generator for the sequence $\{a_i\}_{i=0,1,\ldots}$ [5], which must be a factor of f^A. Hence the Berlekamp/Massey algorithm requires at most $2n$ elements of the sequence (1). Wiedemann now randomly perturbs ("preconditions") A and chooses random u and v. Then with high probability the characteristic polynomial of A, $\det(\lambda I - A)$, is equal to the minimal recurrence polynomial of $\{a_i\}_{i=0,1,\ldots}$.

The above approach is originally intended for sparse matrices, where the Krylov sequence Av, $A^2v, \ldots$ can be computed efficiently. In [22] the following baby steps/giant steps approach is introduced for computing (1).

Let $r = \lceil \sqrt{2n}\, \rceil$ and $s = \lceil 2n/r \rceil$.

```
Step 1. For $j = 1, 2, \ldots, r-1$ Do $v^{[j]} \leftarrow A^j v$;
Step 2. $Z \leftarrow A^r$;
Step 3. For $k = 1, 2, \ldots, s$ Do $(u^{[k]})^{Tr} \leftarrow u^{Tr} Z^k$;
Step 4. For $j = 0, 1, \ldots, r-1$ Do
            For $k = 0, 1, \ldots, s$ Do $a_{kr+j} \leftarrow (u^{[k]})^{Tr} v^{[j]}$.
```

We shall analyze the method for $A \in \mathbb{Z}^{n\times n}$ and shall denote by $\|A\|$ the infinity matrix norm: $\|A\| = \max_{1\le i\le n} \sum_{j=1}^{n} |a_{i,j}|$, where $a_{i,j}$ is the integer in row i and column j of A. Hence the maximal bit length of any entry in A,

$$\min_{1\le i,j\le n} \{\beta : |a_{i,j}| < 2^\beta,\ \beta \ge 1\} \le 1 + \log(\|A\| + 1). \tag{2}$$

In order to avoid zero or undefined logarithms, we shall simply define $\|A\| > 1$ whenever it is necessary.

The costly steps in bit complexity in the above method are Steps 2 and Steps 3. The power A^r is computed by binary exponentiation in $n^{3+o(1)}$ integer operations. Since $\|A^r\| \le \|A\|^r$ the lengths of the integers involved in Step 2 are $(\sqrt{n}\,\log\|A\|)^{1+o(1)}$. In Step 3 the lengths can be as much as $(n\,\log\|A\|)^{1+o(1)}$, but now the number of integer operations is less: s matrix-time-vector products yielding a total of $O(n^{2.5})$. Therefore, both steps can be performed in

$$(n^{3.5}\log\|A\|)^{1+o(1)} \text{ bit operations,} \tag{3}$$

and Steps 1 and 4 are dominated by this cost.

The complete determinant algorithm needs a random preconditioner, random projections, and the Berlekamp/Massey algorithm. Already Wiedemann's original preconditioners, based on Beneš permutation networks and columnwise scaling [33, Chapter V], and random projections (see also [23, Section 3]) achieve the overall bit complexity (3), because they increase $\log \|A\|$ only by an additive term of order $(\log n)^{O(1)}$. Note that the arithmetic cost of the Berlekamp/Massey algorithm is essentially linear, on input coefficients of length $(n \log \|A\|)^{1+o(1)}$.

William J. Turner at North Carolina State University has implemented the complete determinant algorithm for (dense) integer matrices in Maple 6. The implementation is based entirely on residue arithmetic and the Chinese remainder algorithm. In Steps 1 and 2, the entries of $v^{[j]}$ and Z are first computed for as many moduli as are necessary for recovery of the actual integer values, that is no more than $(\sqrt{n} \log \|A\|)^{1+o(1)}$. Steps 3, 4, and the Berlekamp/Massey algorithm is done for as many primes as are necessary for the recovery of the determinant. We use Hadamard's inequality to bound the magnitude of the determinant. Hence no more than $(n \log \|A\|)^{1+o(1)}$ moduli are needed. Furthermore, residue base extension is used for obtaining the remainders of the entries of $v^{[j]}$ and Z with respect to the new moduli. Note that the implementation does not compute with arbitrary precision integers until the very last Chinese remainder step for the determinant. Therefore there are no log factors of any practical significance induced by the manipulation of the intermediate long integers. We like to add that, theoretically, the input matrix could be reduced modulo all $(n \log \|A\|)^{1+o(1)}$ primes in $(n^3 \log \|A\|)^{1+o(1)}$ bit operations using Chinese remainder operations based on tree evaluation and FFT-based integer multiplication [2, Section 8.11].

There now exists an extensive theory on Wiedemann-like preconditioners [9]. For example, it is shown that random column scaling, that is, premultiplication by a diagonal matrix whose entries are uniformly sampled for the set of cardinality $O(n^2)$ is sufficient for non-singular matrices. However, such a preconditioner increases the magnitude of the determinant and slows the algorithm considerably in practice, as William Turner has observed. A preconditioner like the one in [24, Proposition 1], but where the right multiplier is also a unit (lower) triangular (Toeplitz) matrix, can be shown to yield with high probability a matrix with distinct eigenvalues. That preconditioner does not affect the determinant, but its practical efficiency still needs to be tested.

Our algorithm is Las Vegas: unlucky preconditioning or unlucky projections yield minimal generators of degree $< n$ and can be discarded. The infallibility can be extended to singular inputs. First we note that the preconditioners always preserve non-singularity. Thus even if the projections are unlucky and the minimal linear generator for (1) is a proper factor of the minimal polynomial, the minimal generator cannot be divisible by λ. Note that modulo a prime that might actually happen, in which case we know that that prime divides the determinant. Since the preconditioners naturally preserve singularity, the minimum generator of (1) for a singular matrix is with high probability divisible by λ, because it is with high probability equal to the minimum polynomial of the matrix. It is this condition that certifies singularity. In the Chinese remainder setting we encounter a λ factor in the minimum linear generator for so many primes that the Hadamard bound is covered. With bounded probability our algorithm may fail to compute the determinant, namely if the minimal linear generator for (1) is of degree $< n$ but not divisible by λ for almost all primes. We add that if one only needs a certificate for singularity, it is slightly faster to compute a non-zero solution to the linear system $Ax = 0$ [25].

Sub-cubic matrix multiplication algorithms can be employed to improve the theoretical complexity of the above approach. Already in [22] the exponent $(n^{3.188} \log \|A\|)^{1+o(1)}$ is proven. The version of the paper posted at `http://www.math.ncsu.edu/~kaltofen/` has a note added in 1995 that shows how to reduce the exponent to $(n^{3.0281} \log \|A\|)^{1+o(1)}$, which was already below the complexity achieved in [16]. In order to go significantly below this bound we need to employ another technique, which we shall discuss next.

3 Acceleration by blocking

Coppersmith [11] first introduce blocking to the Wiedemann method. In our description we also take into account the interpretation in [30, 31], where the relevant literature from multivariable control theory is cited.

For the "block" vectors $X \in \mathbb{K}^{n\times l}$ and $Y \in \mathbb{K}^{n\times m}$ consider the sequence of $l \times m$ matrices

$$B^{[0]} = X^{Tr}Y,\ B^{[1]} = X^{Tr}AY,\ B^{[2]} = X^{Tr}A^2Y,\ B^{[3]} = X^{Tr}A^3Y,\ \ldots \qquad (4)$$

As in the unblocked Wiedemann method, we seek linear generating polynomials. A vector polynomial $\sum_{i=0}^{d} c^{[i]}\lambda^i$, where $c^{[i]} \in \mathbb{K}^m$, is said to linearly generate the sequence (4) from the right if

$$\forall\, j \geq 0\colon\ \sum_{i=0}^{d} B^{[j+i]}c^{[i]} = \sum_{i=0}^{d} X^{Tr}A^{i+j}Yc^{[i]} = 0^m.$$

For the minimum polynomial of A, $f^A(\lambda)$, and for the μ-th unit vector in $\mathbb{K}^m$, $e^{[\mu]}$, $f^A(\lambda)e^{[\mu]} \in (\mathbb{K}[\lambda])^m = \mathbb{K}^m[\lambda]$ is such a generator because it already generates the Krylov sequence $\{A^i Y^{[\mu]}\}_{i\geq 0}$, where $Y^{[\mu]}$ is the μ-th column of Y. We can now consider the set $W_X^{A,Y}$ of all such right vector generators. This set forms a $\mathbb{K}[\lambda]$-submodule of the $\mathbb{K}[\lambda]$-module $\mathbb{K}[\lambda]^m$ and contains m linearly independent (over the field of rational functions $\mathbb{K}(\lambda)$) elements, namely all $f^A(\lambda)e^{[\mu]}$. Furthermore, the submodule has an ("integral") basis over $\mathbb{K}[\lambda]$, namely any set of m linearly independent generators such that the degree in λ of the determinant of the matrix formed by those basis vector polynomials as columns is minimal. The matrices corresponding to all integral bases clearly are right equivalent with respect to multiplication from the right by any unimodular matrix in $\mathbb{K}[\lambda]^{m\times m}$, whose determinant is by definition of unimodularity a non-zero element in $\mathbb{K}$. Thus we can pick a matrix canonical form for this right equivalence, say the Popov form, and obtain a unique minimum matrix generating polynomial for (4), denoted by $F_X^{A,Y}(\lambda) \in \mathbb{K}^{m\times m}[\lambda] = (\mathbb{K}[\lambda])^{m\times m}$.

The minimum matrix generating polynomial is computed from the sequence (4) by a block version of the Berlekamp/Massey algorithms [11] or its variants, like by a matrix Padé approximation [4], by a matrix Euclidean algorithm [29], or by a block Toeplitz solver following the classical Levinson-Durbin approach [23]. The latter most easily elucidates the advantage of blocking: the number of sequence elements needed can be much shorter. Let $d = \lceil n/m \rceil$, $\nu = m(d+1)$, $e = \lceil \nu/l \rceil$, and let $\mu = le$. Then the columns in $F_X^{A,Y}(\lambda)$ correspond to solutions of the $\mu \times \nu$ block Toeplitz system

$$\left[\begin{array}{c|c|c|c|c} B^{[d]} & \dots & & B^{[1]} & B^{[0]} \\ \hline B^{[d+1]} & B^{[d]} & & B^{[2]} & B^{[1]} \\ \hline \vdots & & \ddots & & \vdots \\ \hline B^{[d+e+1]} & \dots & & & B^{[e-1]} \end{array}\right] \left[\begin{array}{c} c^{[d]} \\ \hline c^{[d-1]} \\ \hline \vdots \\ \hline c^{[0]} \end{array}\right] = 0^{\mu}. \tag{5}$$

We have $d+e-1 < n/l+n/m+2m/l+1$. That there are m linearly independent solutions follows from rank considerations.

As with the unblocked Wiedemann projections, unlucky projection block vectors X and Y cause a drop in the maximal degree of the minimum matrix generator, which is in Popov form and its degree is to be taken as a vector of column degrees, or equivalently a drop in the maximal rank of the block Toeplitz matrix in (5). It is possible to both characterize the maximal degree vector/maximal rank and to establish conditions under which the block vectors X and Y preserve them. We shall do this in Theorem 1 below.

A relationship between the minimum polynomial $f^A(\lambda)$ and $\det(F_X^{A,Y}(\lambda))$ follows from the theory of realizations of multivariable control theory. The basis is the matrix power series

$$X^{Tr}(I-\lambda A)^{-1}Y = X^{Tr}\Big(\sum_{i\geq 0} A^i\lambda^i\Big)Y = \sum_{i\geq 0} B^{[i]}\lambda^i.$$

For the minimum matrix generator

$$F_X^{A,Y}(\lambda) = C^{[d]}\lambda^d + \cdots + C^{[0]} \in \mathbb{K}^{m\times m}[\lambda]$$

we then have

$$X^{Tr}(I-\lambda A)^{-1}Y(C^{[d]} + \cdots + C^{[0]}\lambda^d) = G(\lambda) \in \mathbb{K}[\lambda]^{m\times m}$$

which yields the matrix Padé approximation

$$X^{Tr}(I-\lambda A)^{-1}Y = G(\lambda)(C^{[d]} + \cdots + C^{[0]}\lambda^d)^{-1}. \tag{6}$$

In control theory, the left side of (6) is called a realization of the rational matrix on the right side of (6). Clearly, the reverse polynomial of $f^A(\lambda)$ is a common denominator of the rational entries of the matrices on both sides. If the least common denominator of the left side matrix in (6) is actually $\det(I-\lambda A)$, then it follows from degree considerations that $\det(F_X^{A,Y}(\lambda)) = \alpha\cdot\det(\lambda I - A)$ for a non-zero element α in $\mathbb{K}$. Our algorithm uses the matrix preconditioners discussed in Section 2 and random projections to achieve this determinantal equality.

We shall make the relationship between $\lambda I - A$ and $F_X^{A,Y}(\lambda)$ more explicit. For a matrix $H(\lambda) \in (\mathbb{K}[\lambda])^{\nu\times\nu}$ we consider the Smith normal form, which is an equivalent diagonal matrix over $\mathbb{K}[\lambda]$ with diagonal elements $s_1(\lambda), \ldots,$ $s_\phi(\lambda)$, $1, \ldots, 1, 0, \ldots, 0$, where s_i are the invariant factors of H, that is, non-constant monic polynomials with the property that s_i is a (trivial or nontrivial) polynomial factor of s_{i-1} for all $2 \leq i \leq \phi$. Because the Smith normal form of the characteristic matrix $\lambda I - A$ corresponds to the canonical forms (Frobenius, Jordan) for similarity to A, the largest invariant factor of $\lambda I - A$, $s_1(\lambda)$, equals the minimum polynomial $f^A(\lambda)$.

Theorem 1 *Let $A \in \mathbb{K}^{n\times n}$, $X \in \mathbb{K}^{n\times l}$, $Y \in \mathbb{K}^{n\times m}$ and let $s_1, \ldots, s_\phi$ denote all invariant factors of $\lambda I - A$. Suppose that $l \geq \min\{m, \phi\}$. Then for all i, the i-th invariant factor of $F_X^{A,Y}(\lambda)$ divides s_i. Furthermore, there exist matrices $U \in \mathbb{K}^{n\times l}$ and $V \in \mathbb{K}^{n\times m}$ such that for all i the i-th invariant factor of $F_U^{A,V}(\lambda)$ is equal to s_i. In the latter case,*

$$\deg_\lambda(\det(F_U^{A,V}(\lambda))) = \deg(s_1) + \cdots + \deg(s_{\min\{m,\phi\}}) \leq n \tag{7}$$

which is also the maximal rank of the block Toeplitz matrix on the left side of (5).

The existence of such U, V establish maximality of the matrix generator for symbolic X and Y, and via the Schwartz/Zippel lemma for random projection matrices. If $\mathbb{K}$ is a small finite field, Wiedmann's analysis has been generalized in [31]. The degree formula (7) already shows the superiority of blocking over our original solution of Section 2. If the number of invariant factors satisfies $\phi \leq m$, we can omit preconditioning of our input matrix A from our algorithm, which we shall—at last—present now.

Let the blocking factors be $l = m = \lceil n^{\sigma} \rceil$ where $\sigma = 1/3$.

Step 1. Precondition A such that with high probability $\det(\lambda I - A) = s_1(\lambda) \cdots s_{\min\{m,\phi\}}$, where $s_1, \ldots, s_\phi$ are the invariant factors of $\lambda I - A$. Note that any of the preconditioners of Section 2 would be sufficient.

Step 2. Select random $X, Y \in S^{n \times m}$, where S is a set of integers of cardinality $n^{O(1)}$, and compute the sequence $B^{[i]} = X^{Tr} A^i Y$ for all $0 \leq i < 2n/m + 3 = O(n^{1-\sigma})$.

Step 3. Compute the minimal matrix generator $F_X^{A,Y}(\lambda)$ for $\{B^{[i]}\}_{i \geq 0}$.

Step 4. Compute the leading and constant coefficients of $\Delta(\lambda) = \det(F_X^{A,Y}(\lambda))$. If $\deg(\Delta) < n$ and $\Delta(0) \neq 0$ then return "failure" else return $\det(A) = \Delta(0)/(\text{leading coefficient of } \Delta)$.

For Step 2 we utilize our baby steps/giant steps technique of Section 2. Let the number of giant steps be $s = \lceil n^{\tau} \rceil$, where $\tau = 1/3$, and let the number of baby steps be $r = \lceil (2n/m + 3)/s \rceil = O(n^{1-\sigma-\tau})$.

Substep 2.1 for $j = 1, 2, \ldots, r-1$ Do $V^{[j]} \leftarrow A^j Y$;

Substep 2.2 $Z \leftarrow A^r$;

Substep 2.3. For $k = 1, 2, \ldots, s$ Do $(U^{[k]})^{Tr} \leftarrow U^{Tr} Z^k$;

Substep 2.4. For $j = 0, 1, \ldots, r-1$ Do
For $k = 0, 1, \ldots, s$ Do $B^{[kr+j]} \leftarrow (U^{[k]})^{Tr} V^{[k]}$.

The substeps above can be carried out either by Chinese remaindering or by arithmetic on arbitrarily long integers using FFT-based integer multiplication. Substep 2.2 costs $n^{3+o(1)}$ arithmetic operations on integers of length $(r \log \|A\|)^{1+o(1)}$. Substep 2.3 costs $O(s\, m\, n^2)$ arithmetic operations on integers of length $(r\, s \log \|A\|)^{1+o(1)}$. Again Substeps 2.1 and 2.4 are dominated by this cost, which is $(n^{1+1/3} \log \|A\|)^{1+o(1)}$.

Steps 3 and 4 in the main algorithm are performed by Chinese remaindering, so that the intermediately computed scalars stay bounded in length. In particular, the size growth analysis in the block Berlekamp/Massey algorithm for fields of characteristic 0 has to our knowledge not been carried out. The Chinese remainder approach introduces a slight complication. Some prime moduli p_i may yield a minimum matrix generating polynomial that trigger the failure condition in Step 4 and thus provide no value for $\det(A) \bmod p_i$, even when the preconditioning and projections were lucky over the integers. It is clear from the above analysis that this can happen only if the rank of the block Toeplitz matrix in (5) is smaller modulo p_i. Since a maximal non-zero minor of the integral block Toeplitz matrix is of length no more than $(n^{1+2/3} \log \|A\|)^{1+o(1)}$, we may choose random prime moduli $p_i = (n \log \|A\|)^{O(1)}$ and avoid such unlucky modular reduction with high probability.

Now Steps 3 and 4 are performed for sufficiently many prime moduli p_i that capture the integral determinant through Chinese remaindering, by Hadamard's bound no more than $(n \log \|A\|)^{1+o(1)}$. For each prime the cost of the block Berlekamp/Massey algorithm is $O(m^3(n/m)^2)$ residue operations [11]. The computation of the leading and constant coefficient of Δ is only $O(m^3)$ residue operations. Overall, the bit complexity of Step 3 is again $(n^{3+1/3} \log \|A\|)^{1+o(1)}$.

Theorem 2 *Our algorithm computes the determinant of any matrix $A \in \mathbb{Z}^{n\times n}$ with $(n^{3+1/3} \log \|A\|)^{1+o(1)}$ bit operations. Our algorithm utilizes $(n^{1+1/3} + n \log \|A\|)^{1+o(1)}$ random bits and either returns the correct determinant or it returns "failure," the latter with probability of no more than* $1/2$.

As stated in the Introduction, by use of sub-cubic matrix multiplication algorithms the worst case bit complexity of the block algorithm can be brought below cubic complexity in n. We note that taking the n^2 entries of the input matrix modulo n prime residues is already a cubic process in n, so our speedup must proceed differently. For one, we compute those remainders for the integral entries in $B^{[i]}$, which are $O(mn)$ integers of length $(r\, s \log \|A\|)^{1+o(1)}$. Again asymptotically fast tree-like remaindering can be applied on those entries [2]. The exponent that is given in the Introduction, namely 2.697263, requires the use of the Knuth/Schönhage half-GCD algorithm in Step 3, now applied to matrix polynomials. In order for that algorithm to be applicable, the matrix polynomial remainder chain must be normal, that is, none of the leading coefficients must be singular matrices. It can be shown that our randomizations also produce a normal chain with high probability. FFT-based multiplication algorithms for matrix polynomials are described in [8]. Finally, we not only employ the the Coppersmith/Winograd matrix multiplication algorithm but also Coppersmith's fast methods for rectangular matrices [12] (we seem not to need the results in [19]). A Maple 6 worksheet that contains the exponent calculations is posted at `http://www.math.ncsu.edu/~kaltofen/`. With matrix multiplication exponent 2.375477 the optimal value for the blocking factor is $\sigma \approx 0.507$ and for the giant stepping $\tau \approx 0.172$.

4 Improved division-free complexity

Our baby steps/giant steps algorithm with blocking of Section 3 can be employed to improve the division-free complexity of the determinant of [22]. Here we consider a matrix $A \in R^{n\times n}$, where R is a commutative ring with a unit element. At task is to compute the determinant of A by ring additions, subtractions and multiplications. Blocking can improve the number of ring operations from $n^{3.5+o(1)}$ [22] to $n^{3+1/3+o(1)}$, and with subcubic matrix multiplication from $O(n^{3.0281})$ [22, note added in version posted on web] to $O(n^{2.6973})$. Our algorithm combines the blocked determinant algorithm with the elimination of divisions technique of [22]. Our computational model is either a straight-line program/arithmetic circuit or an algebraic random access machine [21]. Further problems are to compute the characteristic polynomial and the adjoint matrix of A.

The main idea of [22] follows [27] and for the input matrix A computes the determinant of the polynomial matrix $L(z) = M + z(A - M)$, where $M \in \mathbb{Z}^{n\times n}$ is an integral matrix whose entries are indepent of the entries in A. For $\Delta(z) = \det(L(z))$ we have $\det(A) = \Delta(1)$. All intermediate elements are represented as polynomials in $R[z]$ or as truncated power series in $R[[z]]$ and the "shift" matrix M determines them in such a manner that whenever a division by a polynomial or truncated power series is performed the constant coefficients are ± 1. For the algorithm in Section 3 we not only pick M but also concrete projection block vectors $X \in \mathbb{Z}^{n\times m}$ and $Y \in \mathbb{Z}^{n\times m}$. No randomization is necessary, as M is a "good" input matrix ($\phi = m$), and X and Y are "good" projections.

The matrices M, X and Y are block versions of the ones constructed in [22]. Suppose that the blocking factor m is a divisor of n, the dimension of A. This we can always arrange by padding A to $\left[\begin{array}{c|c} A & 0 \\ \hline 0 & I \end{array}\right]$. Let $d = n/m$ and let

$$a_i = \binom{i}{\lfloor i/2 \rfloor}, \; c_i = -(-1)^{\lfloor (d-i+1)/2 \rfloor} \binom{\lfloor (d+i)/2 \rfloor}{i},$$

and let

$$C = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & & 0 & 1 \\ c_0 & c_1 & \dots & c_{d-2} & c_{d-1} \end{bmatrix}, \; v = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ \\ a_{d-1} \end{bmatrix}.$$

We show in [22] that for the sequence $a_i = e_1^{Tr} C^i v$, where $e_1^{Tr} = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} \in \mathbb{Z}^{1 \times d}$ is the first d-dimensional unit (row) vector, the Berlekamp/Massey algorithm divides by only ±1. We define

$$M = \left[\begin{array}{c|c|c|c} C & 0 & \dots & 0 \\ \hline 0 & C & \ddots & 0 \\ \hline \vdots & 0 & \ddots & \vdots \\ \hline 0 & \dots & 0 & C \end{array} \right] \in \mathbb{Z}^{n \times n},$$

$$X = \left[\begin{array}{c|c|c|c} e_1 & 0 & \dots & 0 \\ \hline 0 & e_1 & \ddots & 0 \\ \hline \vdots & 0 & \ddots & \vdots \\ \hline 0 & \dots & & e_1 \end{array} \right] \in \mathbb{Z}^{n \times m}, \; Y = \left[\begin{array}{c|c|c|c} v & 0 & \dots & 0 \\ \hline 0 & v & \ddots & 0 \\ \hline \vdots & 0 & \ddots & \vdots \\ \hline 0 & \dots & & v \end{array} \right] \in \mathbb{Z}^{n \times m}.$$

By construction, the algorithm for computing the determinant of Section 3 performed now with the matrices X, M, Y results in a minimum matrix generator

$$F_X^{M,Y}(\lambda) = (\lambda^d - c_{d-1}\lambda^{d-1} - \cdots - c_0) I_m,$$

where I_m is an $m \times m$ identity matrix. Furthermore, this generator can be computed from the sequence of block vectors $B^{[i]} = a_i I_m$ by a matrix Euclidean algorithm (c.f. [15]) in which all leading coefficient matrices are equal to $\pm I_m$.

The arithmetic cost for executing the block baby steps/giant steps algorithm on the polynomial matrix $L(z) = M + z(A - M)$ is related to the bit complexity of Section 3. Now the intermediate lengths are the degrees in z of the computed polynomials in $R[z]$. Therefore, the matrices $X^{Tr} L(z)^i Y \in R^{m \times m}$ can be computed for all $0 \leq i < d + 3$ in $n^{3+1/3+o(1)}$ ring operations. In the matrix Euclidean algorithm for Step 3 we perform trunctated power series arithmetic module z^{n+1}. The arithmetic cost is $(d^2 m^3 n)^{1+o(1)}$ ring operations for the classical Euclidean algorithm with FFT-based power series arithmetic. For the latter, we employ a division-free FFT-based polynomial multiplication algorithm [8]. Finally, in Step 4 of Section 3 the determinant of $F_X^{L(z),Y}(\lambda)$ is to be computed division-free in trunctated power series arithmetic over $R[z,\lambda]$ mod (z^{n+1}, λ^{n+1}). For this last step we can use our original division-free algorithm [22] and achieve arithmetic complexity $(m^{3.5}n^2)^{1+o(1)}$. We have proven the following theorem.

Theorem 3 *Our algorithm computes the characteristic polynomial of any matrix $A \in R^{n\times n}$ with $(n^{3+1/3})^{1+o(1)}$ ring operations in R. By the results in [3] the same complexity is obtained for the adjoint matrix, which can be symbolically defined as* $\det(A)A^{-1}$.

It is of some mathematical interest to speed our result via subcubic matrix multiplication algorithms. Again, Steps 3 and 4 of the algorithm in Section 3 are at issue. Step 3 is easier than in the integer case, since the matrix polynomial remainder chain is known to be normal at $z = 0$. The number of ring operations for Step 3 shrinks to $(dm^\omega n)^{1+o(1)}$, where ω is the exponent for square matrix multiplication. However, Step 4 creates a problem for the case that we wish to compute the entire characteristic polynomial of A, that is, $\det(F_X^{L(1),Y}(\lambda))$. A preliminary straight-forward implementation of Step 4 obtains a division-free complexity for the characteristic polynomial of $O(n^{2.80652})$ ($\sigma = 0.34$, $\tau = 0.23$).

5 Conclusion

Our methods apply to entry domains other than the integers, like polynomial rings and algebraic number rings. We would like to add that if the entries are polynomials over a finite field, there are different techniques possible [26]. Our determinant algorithm for integer matrices may be extended to a Monte Carlo method for computing the integral Smith normal form of an integral matrix by the techniques described in [18].

The reduction of the bit complexity of an algebraic problem below that of its known algebraic complexity times the bit length of the answer should raise important considerations for the design of generic algorithms with abstract coefficient domains [20] and for algebraic lower bounds for low complexity problems [28]. We demonstrate that the interplay between the algebraic structure of a given problem and the bits of the intermediately computed numbers can lead to a dramatic reduction in the bit complexity of a fundamental mathematical computation task.

Acknowledgement

We thank William J. Turner for his observations made on the practicality of our method, and Mark Giesbrecht for reporting to us the value of the smallest

exponent in [16] prior to its publication.

References

Note: many of the authors' publications cited below are accessible through links in their webpages listed under the title.

1. J. Abbott, M. Bronstein, and T. Mulders. Fast deterministic computation of determinants of dense matrices. In S. Dooley, editor, *ISSAC 99 Proc. 1999 Internat. Symp. Symbolic Algebraic Comput.*, pages 181–188, New York, N. Y., 1999. ACM Press.
2. A. Aho, J. Hopcroft, and J. Ullman. *The Design and Analysis of Algorithms.* Addison and Wesley, Reading, MA, 1974.
3. W. Baur and V. Strassen. The complexity of partial derivatives. *Theoretical Comp. Sci.*, 22:317–330, 1983.
4. B. Beckermann and G. Labahn. A uniform approach for fast computation of matrix-type Padé approximants. *SIAM J. Matrix Anal. Appl.*, 15(3):804–823, July 1994.
5. R. P. Brent, F. G. Gustavson, and D. Y. Y. Yun. Fast solution of Toeplitz systems of equations and computation of Padé approximants. *J. Algorithms*, 1:259–295, 1980.
6. H. Brönnimann, I. Emiris, V. Pan, and S. Pion. Sign determination in residue number systems. *Theoretical Comput. Sci.*, 210(1):173–197, 1999. Special issue on real numbers and computers.
7. H. Brönnimann and M. Yvinec. A complete analysis of Clarkson's algorithm for safe determinant evaluation. Technical Report INRIA-2051, Institut National de Recherche en Informatique et en Automatique, Novembre 1996.
8. D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Inform.*, 28(7):693–701, 1991.
9. L. Chen, W. Eberly, E. Kaltofen, B. D. Saunders, W. J. Turner, and G. Villard. Efficient matrix preconditioners for black box linear algebra. Manuscript submitted for publication, 2000.
10. Kenneth L. Clarkson. Safe and efficient determinant evaluation. In *Proc. 33rd Annual Symp. Foundations of Comp. Sci.*, pages 387–395, Los Alamitos, California, 1992. IEEE Computer Society Press.
11. D. Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comput.*, 62(205):333–350, 1994.
12. D. Coppersmith. Rectangular matrix multiplication revisited. *J. Complexity*, 13:42–49, 1997.

13. D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symbolic Comput.*, 9(3):251–280, 1990. Special issue on complexity theory.
14. J. Dixon. Exact solution of linear equations using p-adic expansions. *Numer. Math.*, 40(1):137–141, 1982.
15. J. L. Dornstetter. On the equivalence between Berlekamp's and Euclid's algorithms. *IEEE Trans. Inf. Theory*, IT-33(3):428–431, 1987.
16. W. Eberly, M. Giesbrecht, and Gilles Villard. On computing the determinant and Smith form of an integer matrix. In *Proc. 41stAnnual Symp. Foundations of Comp. Sci.*, Los Alamitos, California, 2000. IEEE Computer Society Press.
17. J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, Cambridge, New York, Melbourne, 1999.
18. M. Giesbrecht. Fast computation of the Smith form of a sparse integer matrix. *Computational Complexity*, 2001. to appear.
19. Xiaohan Huang and Victor Y. Pan. Fast rectangular matrix multiplication and applications. *J. Complexity*, 14:257–299, 1998.
20. R. D. Jenks, R. S. Sutor, and S. M. Watt. Scratchpad II: An abstract datatype system for mathematical computation. In J. R. Rice, editor, *Mathematical Aspects of Scientific Software*, volume 14 of *The IMA Volumes in Mathematics and its Application*, pages 157–182. Springer Verlag, New York, 1988.
21. E. Kaltofen. Greatest common divisors of polynomials given by straight-line programs. *J. ACM*, 35(1):231–264, 1988.
22. E. Kaltofen. On computing determinants of matrices without divisions. In P. S. Wang, editor, *Internat. Symp. Symbolic Algebraic Comput. 92*, pages 342–349, New York, N. Y., 1992. ACM Press.
23. E. Kaltofen. Analysis of Coppersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems. *Math. Comput.*, 64(210):777–806, 1995.
24. E. Kaltofen and V. Pan. Processor-efficient parallel solution of linear systems II: the positive characteristic and singular cases. In *Proc. 33rd Annual Symp. Foundations of Comp. Sci.*, pages 714–723, Los Alamitos, California, 1992. IEEE Computer Society Press.
25. T. Mulders and A. Storjohann. Certified dense linear system solving. Manuscript available from `http://www.scl.csd.uwo.ca/~storjoha/`, 2001.
26. T. Mulders and A. Storjohann. On lattice reduction for polynomial matrices. Manuscript available from `http://www.scl.csd.uwo.ca/~storjoha/`, 2001.

27. V. Strassen. Vermeidung von Divisionen. *J. reine u. angew. Math.*, 264:182–202, 1973. In German.
28. V. Strassen. Algebraic complexity theory. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Algorithms and Complexity*, volume A, pages 633–672. Elsevier Science Publ., Amsterdam, 1990.
29. E. Thomé. Fast computation of linear generators for matrix sequences and application to the block Wiedemann algorithm. In B. Mourrain, editor, *Proc. 2001 Internat. Symp. Symbolic Algebraic Comput.*, New York, N. Y., 2001. ACM Press. to appear.
30. G. Villard. Further analysis of Coppersmith's block Wiedemann algorithm for the solution of sparse linear systems. In W. Küchlin, editor, *ISSAC 97 Proc. 1997 Internat. Symp. Symbolic Algebraic Comput.*, pages 32–39, New York, N. Y., 1997. ACM Press.
31. G. Villard. A study of Coppersmith's block Wiedemann algorithm using matrix polynomials. Rapport de Recherche 975 IM, Institut d'Informatique et de Mathématiques Appliquées de Grenoble, `www.imag.fr`, April 1997.
32. Gilles Villard. Computing the Frobenius normal form of a sparse matrix. In V. G. Ganzha, E. W. Mayr, and E. V. Vorozhtsov, editors, *CASC 2000 Proc. the Third International Workshop on Computer Algebra in Scientific Computing*, pages 395–407. Springer Verlag, 2000.
33. D. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inf. Theory*, IT-32:54–62, 1986.