High-Order Lifting

Arne Storjohann
Ontario Research Centre for Computer Algebra
Department of Computer Science
University of Waterloo
Waterloo ON, N2L 3G1, Canada

February 15, 2002

Abstract

The well-known technique of adic-lifting for linear-system solution is studied. Some new methods are developed and applied to get algorithms for the following problems over the ring of univariate polynomials with coefficients from a field: rational system solving, integrality certification and determinant/ Smith-form computation. All algorithms are Las Vegas probabilistic.

1 Introduction

Let K be a field and $A \in \mathsf{K}[x]^{n\times n}$ be nonsingular modulo X. Let $B \in \mathsf{K}[x]^{n\times m}$. Then $A^{-1}B$ admits a unique X-adic series expansion

$$A^{-1}B = C_0 + C_1 X + C_2 X^2 + \cdots + \overbrace{C_h X^h + \cdots + C^{h+k} X^{h+k}}^{HX^h} + \cdots \qquad (1)$$

where each $C_* \in \mathsf{K}[x]^{n\times m}$ has $\deg C_* < \deg X$. This paper presents fast algorithms for computing only parts of the expansion, eg. H as shown (1) for a given h and k. We call this high-order lifting. The algorithms for high-order lifting lead to Las Vegas solutions of many other computational problems. Let $b \in \mathsf{K}[x]^{n\times 1}$ and $B \in \mathsf{K}[x]^{n\times m}$ be given. The three main problems are:

- **Rational system solving** Compute $A^{-1}b$.
- **Integrality certification** Assay if $A^{-1}B$ is integral.
- **Determinant** Compute the determinant/Smith-form of A.

Assuming $\deg b = O(n \deg A)$ and $m = O(n/\lceil \deg B/\deg A \rceil)$, all the problems listed above can be solved in an expected number of $O\tilde{~}(n^{\theta} \deg A)$ field operations from K. Here, θ is the exponent for matrix multiplication (see below for cost model). These complexity bounds improve on previous results.

Consider first rational system solving. The currently best deterministic algorithm is $O\tilde{}(n^3 \deg A)$, see [10]. Our algorithm is based on adic-lifting, see [2, 6]. The algorithm is probabilistic because a small degree polynomial not dividing the determinant of A is required to be chosen randomly. The previosly best Las Vegas complexity, also using adic-lifting, is about $O\tilde{}(n^{2.698} \deg A)$, see [8].

Now consider the integrality certification problem when $B = I_n$. Then A is unimodular precisely when A^{-1} is over $\mathsf{K}[x]$. Unimodularity can be tested by computing $\det A \bmod X$ for a randomly chosen small degree X; this gives a nearly-optimal $O\tilde{}(n^\theta + n^2 \deg A)$ Monte Carlo probabilistic algorithm. The algorithm we give here is deterministic and nearly matches this running time.

The Hermite-form of A (and hence also the determinant) can be computed deterministically in time $O(n^3(\deg A)^2)$, see [9]. The Smith-form can be computed in the same time (Las Vegas) using the preconditioning of [4]. The computation of the determinant has been well studied, especially also in the case of integer matrices. We refer to [5] for a survey; the currently best result for integers extends to polynomials, giving a Las Vegas algorithm with time about $O\tilde{}(n^{2.698} \deg A)$.

Outline of the paper Sections 2 and 3 define some notation and recall some basic facts about X-adic expansions of rational functions, including the recovery of such expansions using adic-lifting.

Section 4 gives our first high-order lifting algorithm: the purpose is to recover $O(\log n)$ coefficients of the X-adic expansion of A^{-1}. This algorithm is used in almost all subsequent sections, including Section 5, which gives an easy algorithm for unimodularity certification.

Section 6 gives an algorithm for rational system solving in the case where $\deg b \leq \deg A$. The idea of the algorithm is to reduce the problem of solving one system up to order k to that of solving two systems up to order $k/2$; this idea is applied recursively $\log k$ times. Section 7 extends the result of the previous section to allow $\deg b = O(n \deg A)$.

Section 8 gives a general algorithm for solving the high-order lifting problem. This is applied in Section 9 to solve the integrality certification problem.

Sections 10, 11 and 12 deal with determinant/Smith-form computation. It is well known that if b is the last row of I_n, then the minimal denominator of bA^{-1} is the last entry h_n of the row Hermite-form of A. Furthermore, if A is suitably preconditioned, then h_n will be the largest entry in the Smith-form of A. Ideas similar to this are used in [1], [3] and [11]. Section 10 gives a method of transforming A to a new matrix B such that $\deg B \leq \deg A$ and $\deg A = h_n \deg B$. Section 11 shows how to recover the trailing m diagonal entries of the Hermite-form of A, where m is chosen according the degree of these entries. Section 12 puts all the pieces together and gives the complete algorithm for determinant/Smith-form.

Finally, Section 13 concludes and mentions something about the integer case.

Model of computation By time we mean the number of required field operations from K on an algebraic RAM; the operations $+$, $-$, $\times$ and "divide by a nonzero" are considered as unit step operations. Let $O(d^{1+\epsilon})$ be the time to multiply degree d polynomials. Let $O(n^{\theta})$ be the time to multiply two $n \times n$ matrices over a commutative ring with identity. We are going to assume that $2 < \theta \leq 3$ and $0 < \epsilon \leq 1$. Sometimes we will make the (emminently reasonable) assumption that $\epsilon \leq \theta - 2$.

2 X-adic representation of polynomials

Let l be nonnegative integer and $X \in \mathsf{K}[x]$ have degree greater than zero. By X-adic expansion of $a \in \mathsf{K}[x]$ we mean to write

$$a = a_0 + a_1 X + a_2 X^2 + \cdots + a_l X^l,$$

$\deg a_* < \deg X$. Note that by "degree" we will always means degree in x. In other words, if $\deg X = d$ and a_l is nonzero, then $dl \leq \deg a < d(l+1)$. The a_* are called the coefficients of the X-adic expansion of a.

The ring $\mathsf{K}[x]$ has the usual arithmetic operations $\{+, -, \times\}$. We define three additional operations Left, Trunc and Inverse and gives some of their properties. These functions will implicitly be defined in terms of a proscribed X. Let $a \in \mathsf{K}[x]$ and k be nonnegative. Suppose the X-adic expansion of a is

$$a = a_0 + a_1 X + a_2 X^2 + \cdots .$$

Then

$$\mathrm{Left}(a, k) = a_k + a_{k+1} X + a_{k+2} X^2 + \cdots \tag{2}$$

and

$$\mathrm{Trunc}(a, k) = a_0 + a_1 X + a_2 X^2 + \cdots + a_{k-1} X^{k-1}. \tag{3}$$

If $a \perp X$, then $\mathrm{Inverse}(a, k)$ denotes the unique $b \in \mathsf{K}[x]$ such that $b = \mathrm{Trunc}(b, k)$ and $\mathrm{Trunc}(ab, k) = \mathrm{Trunc}(ba, k) = 1$.

All the above definitions above extend naturally to matrix polynomials. Just replace $a, q \in \mathsf{K}[x]$ with $A, Q \in \mathsf{K}[x]^{n \times m}$. The operation Inverse takes as input a square matrix A which has $\det A \perp X$.

Let $a, \gamma \in \mathsf{K}[x]$ and k be positive. A key property of the $\mathrm{Left}(*, k)$ operation is linearity: $\mathrm{Left}(a + \gamma, k) = \mathrm{Left}(a, k) + \mathrm{Left}(\gamma, k)$. This property gives the following lemma.

Lemma 1. *If* $\deg(\gamma) < \deg(X^k)$ *then* $\mathrm{Left}(a + \gamma, k) = \mathrm{Left}(a, k)$.

The next lemma observes (in essence) that Left and Trunc commute.

Lemma 2. *If* $l \leq k$ *then* $\mathrm{Left}(\mathrm{Trunc}(a, k), l) = \mathrm{Trunc}(\mathrm{Left}(a, l), k - l)$.

Computation with X-adic polynomials

We are working over the $\mathsf{K}[x]$ with the operations $\{+, -, \times, \text{Left}, \text{Trunc}, \text{Inverse}\}$. The cost of these operations will depend essentially on our choice of representation for elements of $\mathsf{K}[x]$.

For $a \in \mathsf{K}[x]$ let k be minimal such that $a = \text{Trunc}(a, k)$. Then a can be stored as a list comprised of the first k coefficients of the X-adic expansion. Let $a, b \in \mathsf{K}[x]$ be nonzero with $\deg a \geq \deg b$. Then the X-adic expansion of $a+b$ or $a-b$ can be computed in $O(1+\min(\deg a, \deg b))$ field operations, that of ab in $O((1+\deg a)(1+\deg b)^{\epsilon})$ field operations, and that of $\text{Inverse}(a, k)$ in $O((k \deg X)^{1+\epsilon})$ field operations. Operations Left, Trunc and multiplication by a power of X are free. For $Y \in \mathsf{K}[x]$, conversion from X-adic to Y-adic representation can be accomplished in $O((1+\deg a)^{1+\epsilon})$ field operations.

3 Adic-lifting for linear system solution

Let $A \in \mathsf{K}[x]$ be nonsingular, $\det A \perp X$. Let the X-adic expansion of A^{-1} be

$$A^{-1} = \overbrace{* + *X + \cdots + *X^{l-1}}^{C} + *\,X^{l} + *X^{l+1} + \cdots .$$

For a given $B \in \mathbb{Z}^{n \times m}$, let the X-adic expansion of $A^{-1}B$ be

$$A^{-1}B = \overbrace{* + *X + \cdots + *X^{k-1}}^{D} + \overbrace{*X^{k} + \cdots + *X^{k+l-1}}^{EX^{k}} + \cdots .$$

Suppose we have C and D. Then we can recover E using:

Theorem 3 (Adic lifting). $E = \text{Trunc}(C\,\text{Left}(B - AD, k), l)$.

We end this section with a fact which will be useful throughout. Let $A \in \mathsf{K}[x]^{n \times n}$ be nonsingular. Then

Fact 4. $|\det A| \leq nd$.

Let $B \in \mathsf{K}[x]^{n \times m}$. THen $(\det A)A^{-1}B$ is over $\mathsf{K}[x]$ and

Fact 5. $\deg(\det A)A^{-1}B \leq \deg B + (n-1)d$.

4 High order components of matrix inverse

Let $A \in \mathsf{K}[x]^{n \times n}$ be nonsingular, $\det A \perp X$. In what follows, let $Z^{(i)} = \text{Inverse}(A, 2^i)$. In this section we show how to recover the high order components of the inverse of A: $E^{(i)} = \text{Trunc}(Z^{(i)}, 2^i - 2)$ for $i = 1, 2, \ldots, k$. To see more clearly what we are computing, write the X-adic expansion of A^{-1} as $C_0 + C_1X + C_2X^2 + \cdots$. Then

$$
\begin{aligned}
Z^{(1)} &= \overbrace{C_0 + C_1X}^{E^{(1)}} \\
Z^{(2)} &= C_0 + C_1X + \overbrace{C_2X^2 + C_3X^3}^{E^{(2)}X^2} \\
Z^{(3)} &= C_0 + C_1X + C_2X^2 + C_3X^3 + C_4X^4 + C_5X^5 + \overbrace{C^6X^6 + C_7X^7}^{E^{(3)}X^6} \\
&\vdots \\
Z^{(k)} &= C_0 + C_1X + C_2X^2 + \cdots + C_{2^k-3}X^{2^k-3} + \overbrace{C_{2^k-2}X^{2^k-2} + C_{2^k-1}X^{2^k-1}}^{E^{(k)}X^{2^k-2}}.
\end{aligned}
$$

Starting with $Z^{(0)}$ we can recover $Z^{(1)}, Z^{(2)}, \ldots, Z^{(k)}$ using k steps of quadratic X-adic lifting. This costs $O((2^k)^{1+\epsilon}n^\theta d^{1+\epsilon})$ field operations, $d = \deg X$. Algorithm `HighOrderComponents` recovers only the high order components $E^{(*)}$ as shown above. The cost estimate of $O(kn^\theta d^{1+\epsilon})$ field operations for the algorithm is easy to derive.

Algorithm 6. `HighOrderComponents`$[X](A, k)$
Input: $A \in \mathsf{K}[x]^{n\times n}$ and $k \geq 2$
Output: $(E^{(1)}, E^{(2)}, \ldots, E^{(k)})$ as shown above
Condition: $X \perp \det A$ and $d = \deg X \geq \deg A$

1. $L := \text{Inverse}(A, 1)$;
   $H := \text{Trunc}(L\,\text{Left}(I - AL, 1), 1)$;
   $E^{(1)} := L + XH$;

2. **for** i **from** 2 **to** k **do**
   $L := \text{Trunc}(\text{Left}(E^{(i-1)}\,\text{Left}(-AL, 1), 1), 1)$;
   $H := \text{Trunc}(\text{Left}(E^{(i-1)}\,\text{Left}(-AH, 1), 1), 1)$;
   $E^{(i)} := L + XH$
   od;
   return $(E^{(1)}, E^{(2)}, \ldots, E^{(k)})$

We now prove that the algorithm is correct. Let (A, X, k) be a valid input tuple. Let $(L^{(i)}, H^{(i)})$ be equal to (L, H) as computed during the loop in phase 2 with index i. Phase 1 computes $(L^{(1)}, H^{(1)}) = (C_0, C_1)$ and $E^{(1)} = C_0 + XC_1$. Using induction on j we now prove that

$$
\begin{aligned}
L^{(j)} &= C_{2^j-2} && (4) \\
H^{(j)} &= C_{2^j-1} && (5) \\
E^{(j)} &= C_{2^j-2} + XC_{2^j-1} && (6)
\end{aligned}
$$

for $j = 1, 2, \ldots, k$. The base case $j = 1$ has already been established. That (6) follows from (4) and (5) is clear.

For $i > s$, quadratic X-adic lifting gives

$$\text{Left}(Z^{(i)}, 2^{i-1}) = \text{Trunc}(Z^{(i-1)}\text{Left}(I - AZ^{(i-1)}, 2^{i-1}), 2^{i-1})$$

while the loop computes

$$H^{(i)} = \text{Trunc}(\underbrace{\text{Left}(E^{(i-1)} \overbrace{\text{Left}(-AH^{(i-1)}, 1)}^{R}, 1)}_{S}, 1)).$$

Our goal is to show (4) and (5) hold for $j = i$. It will be sufficient to show that (5) holds since the proof of (4) is analogous. In the proof we will use the following degree estimates, which follow from (5) and (6).

$$\deg(Z^{(j)} - X^{2^j-1}H^{(j)}) < \deg(X^{2^j-1}) \quad (7)$$

$$\deg(Z^{(j)} - X^{2^j-2}E^{(j)}) < \deg(X^{2^j-2}) \quad (8)$$

The next lemma assumes (by induction) that (7) holds for $j = i - 1$.

Lemma 7. $R = (I - AZ^{(i-1)})/X^{2^{i-1}}$.

Proof. Let $a = (I - AZ^{(i-1)})/X^{2^{i-1}}$ and let $\gamma = -A\,\text{Left}(E^{(i-1)}, 1) - Xa$ so that $-A\,\text{Left}(E^{(i-1)}, 1) = Xa + \gamma$. Using (7) for $j = i - 1$ we may derive that $\deg \gamma < \deg A \leq \deg X$. Now use Lemma 1 to conclude that $\text{Left}(Xa + \gamma, 1) = a$. □

At this point we have shown that $S = \text{Left}(E^{(i-1)}R, 1)$ where R is as in Lemma 7. For any nonnegative y we have $S = \text{Left}(X^yE^{(i-1)}R, y + 1)$. Let $y = 2^{i-1} - 2$. The next lemma assumes (by induction) that (8) holds for $j = i - 1$.

Lemma 8. $S = \text{Left}(Z^{(i-1)}R, y + 1)$.

Proof. Let $a = Z^{(i-1)}R$ and $\gamma = X^yE^{(i-1)})R - a$ so that $a + \gamma = X^yE^{(i-1)}R$. Using $\deg(R) < d$ and (7) for $j = i - 1$ gives $\deg \gamma < \deg(X^{y+1})$. Now use Lemma 1. □

Lemmas 7, 8 and 2 now give (5) for $j = i$. The proof that (4) holds for $j = i$ is analogous. This ends the inductive proof of correctness of the algorithm. We have shown:

Proposition 9. *Algorithm* `HighOrderComponents` *is correct. The cost of the algorithm is* $O(kn^\theta d^{1+\epsilon})$ *field operations.*

We remark that typical applications of the algorithm have $k = O(\log n)$.

5 Unimodularity certification

A matrix $A \in \mathsf{K}[x]^{n\times n}$ is said to be unimodular if A is invertible over $\mathsf{K}[x]$. The unimodular matrices are precisely those with determinant a nonzero constant polynomial from $\mathsf{K}[x]$. We present an algorithm to assay if A is unimodular.

Algorithm 10. `UnimodularCertificate`(A)
Input: $A \in \mathsf{K}[x]^{n\times n}$.
Output: True if A is unimodular, false otherwise.

1. **if** $\det(A \bmod x) = 0$ **then return** false **fi**;
   $d := \deg A$;
   $X := x^d$;

2. $k := \lceil \log_2(n+3) \rceil$;
   $(*, *, \ldots, *, H) :=$ `HighOrderComponents`$[X](A, k)$;

3. **if** H is the zero matrix **then**
      return true
   else
      return false
   fi

We now prove that the algorithm is correct. Let k and H be as computed in phase 1. Let $S = \text{Trunc}(A^{-1}, 2^k)$. Then $S = \text{Trunc}(A^{-1}, 2^k - 2) + HX^{2^k-2}$ and $\text{Trunc}(AS, 2^k) = I$. If H is the zero matrix then $\text{Trunc}(AS, 2^k) = AS$, whence $S = A^{-1}$. This shows that a return value of true will always be correct. The paramater k is chosen so that $d(2^k - 2)$ is strictly greater than degrees of numerators in $A^{-1} \in \mathsf{K}(x)^{n\times n}$. Thus, if A^{-1} is over $\mathsf{K}[x]$ then H will be the zero matrix. We have shown:

Proposition 11. *Algorithm* `UnimodularCertificate` *is correct. The cost of the algorithm is in* $O((\log n) n^\theta (\deg A)^{1+\epsilon})$ *field operations.*

6 Series solution — small degree rhs

Let $A \in \mathsf{K}[x]^{n\times n}$ be nonsingular, $\det A \perp X$. Let $b \in \mathsf{K}[x]^{n\times 1}$. We present an algorithm for computing the X-adic expansion of $A^{-1}b$ up to a given order. The algorithm requires both $\deg b$ as well as $\deg A$ to be bounded by d, $d = \deg X$.

Algorithm 12. `SeriesSolutionSmallRHS`$[X](A, b, k)$
Input: $A \in \mathsf{K}[x]^{n\times n}$, $b \in \mathsf{K}[x]^{n\times 1}$, $k \geq 2$
Output: $\text{Trunc}(\text{Inverse}(A, 2^k)\, b, 2^k)$
Condition: $X \perp \det A$ and $d = \deg X \geq \max(\deg A, \deg b)$

1. $(E^{(1)}, E^{(2)}, \ldots, E^{(k-1)}) :=$ `HighOrderComponents`$[X](A, k-1)$;

2. $B := [\ b \mid O\ ]$ where 0 is the $n \times (2^k - 1)$ zero matrix;
for i **from** $k-1$ **by** -1 **to** 1 **do**
  $\bar{B} :=$ the first $2^k - 2^i$ columns of B;
  $\bar{B} := \mathrm{Left}(-A\,\mathrm{Trunc}(\mathrm{Left}(E^{(i)}\bar{B}),1),1),1)$;
  $\bar{B} := [\ O \mid \bar{B}\ ]$ where O is the $n \times 2^i$ zero matrix;
  $B := B + \bar{B}$;
od;
$B := \mathrm{Trunc}(E^{(1)}B, 2)$;

3. # Let $B = [\ d_0 \mid 0 \mid d_2 \mid 0 \mid \cdots \mid d_{2^k-2} \mid 0\ ]$.
$B := d_0 + d_2\,X^2 + \cdots + d_{2^k-2}\,X^{2^k-2}$;
return B

We now prove that the algorithm is correct. Let (A, b, X, k) be valid input tuple. Let $Z^{(i)} = \mathrm{Inverse}(A, 2^i)$ for $i > 0$. Phase 1 computes the high order components of $Z^{(k-1)}$ at a cost of $O(kn^\theta d^{1+\epsilon})$ field operations.

Now consider phase 2. The purpose of this phase is to compute all the coefficients of $\mathrm{Trunc}(Z^{(k)}b, 2^k)$. We begin by giving an example when $k = 4$. A formal proof will follow. Let $b_j = (b - A\,\mathrm{Trunc}(\mathrm{Inverse}(A, j)b, j))/X^j$, $j \in \{0, 2, 4, \ldots, 14\}$. Then $b_0 = b$ and we claim that

$$\mathrm{Trunc}(Z^{(4)}b, 16) = \sum_{j=0}^{16/2^i} \mathrm{Trunc}(Z^{(i)}b_{16j/2^i}, 2^i)X^{16j/2^i}$$

for $i = 4, 3, 2, 1$. Our initial problem is to compute the solution to a single linear system up to order 16. At the start of the loop we have

$$B = [\ b_0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0\ ].$$

The i'th column of B may be thought to be implicitly multiplied by X^{i-1}. After the loop completes with index i the matrix B looks like:

$$\left.\begin{array}{c|cccccccc} i=3 & b_0 & & & & b_8 & & & \\ i=2 & b_0 & & b_4 & & b_8 & & b_{12} & \\ i=1 & b_0 & b_2 & b_4 & b_6 & b_8 & b_{10} & b_{12} & b_{14} \end{array}\right].$$

Thus, each pass through the loop doubles the number of systems we need to solve, but halves the order of precision to which we need the solutions. After the loop completes we need to solve 8 systems up to order X^2. The last line of phase 2 does this to compute

$$B = [\ c_0 + c_1X \quad 0 \quad c_2 + c_3X \quad 0 \quad \cdots \quad c_{14} + c_{15}X \quad 0\ ]$$

where $c_0 + c_1X + c_2X^2 + \cdots$ is the X-adic X-adic expansion of $\mathrm{Trunc}(Z^{(4)}b, 16)$.

Now we give a formal proof of the above. At the same time we estimate the complexity in terms of n, k and d. Let $f(i, m)$ be the cost of computing

$\mathrm{Trunc}(Z^{(k)}B, 2^i)$ for a given $B \in \mathsf{K}[x]^{n\times m}$, $\deg B \le d$. For $i > 1$, quadratic X-adic lifting gives the identity

$$\mathrm{Trunc}(Z^{(i)}B, 2^i) = \mathrm{Trunc}(Z^{(i-1)} \left[\begin{array}{c|c} B & \bar{B} \end{array}\right], 2^{i-1}) \left[\begin{array}{c} 1 \\ X^{2^{i-1}} \end{array}\right]$$

where

$$\bar{B} = \mathrm{Left}(B - A\,\mathrm{Trunc}(Z^{(i-1)}B, 2^{i-1})), 2^{i-1}). \tag{9}$$

It is easy to derive that $\deg \bar{B} < \deg A \le d$. Thus we get

$$f(i,m) \le f(i-1, 2m) + \text{ cost of computing } \bar{B}.$$

Consider equation (9) for $\bar{B}$. We may consider $X^{2^i-2}E^{(i)}$ to be an approximation of $Z^{(i)}$. If we expand formula (9) for $\bar{B}$ we get the new formula

Lemma 13. $\bar{B} = \mathrm{Left}(-A\,\mathrm{Trunc}(\mathrm{Left}(E^{(i-1)}B, 1), 1), 1)$

When $i = 1$ we may compute $\bar{B} = \mathrm{Trunc}(Z^{(1)}B, 2)$ directly since $E^{(1)} = Z^{(1)}$. Since the number of nonzero column in B is doubling each time, the last iteration of the loop dominates. The cost is $O((2^k/n)n^\theta d^{1+\epsilon})$ field operations if $2^k > n$. If $2^k \le n$ the cost is is dominated by that of phase 1.

Finally, phase 3 multiplies each column of B by the appropriate power of X and adds all the columns together. Under our cost model this is free.

We have shown:

Proposition 14. *Algorithm* `SeriesSolutionSmallRHS` *is correct. The cost of the algorithm is $O((k + 2^k/n)n^\theta d^{1+\epsilon})$ field operations.*

$A^{-1}b \in \mathsf{K}(x)^{n\times 1}$ can be recovered from $\mathrm{Trunc}(\mathrm{Inverse}(A, 2n), 2n)$ using rational reconstruction. In this case $k = O(\log n)$. The rational reconstruct costs $O(n(nd)^{1+\epsilon})$. If $\theta - 2 \ge \epsilon$ (an emminently reasonable assumption) then the rational reconstruction does not dominate.

Corollary 15. *Let $(A, b, X, *)$ be a valid input tuple to Algorithm 12. Assuming $\epsilon \le \theta - 2$, the vector $A^{-1}b \in \mathsf{K}(x)^{n\times 1}$ can be computed in $O((\log n)n^\theta d^{1+\epsilon})$ field operations.*

7 Series solution

Let $A \in \mathsf{K}[x]^{n\times n}$ be nonsingular, $\det A \perp X$. Let $b \in \mathsf{K}[x]^{n\times m}$. We present an algorithm for computing the X-adic expansion of $A^{-1}b$ up to a given order. The algorithm makes no assumption on the degree of b.

Algorithm 16. `SeriesSolution`$[X](A, b, k)$
Input: $A \in \mathsf{K}[x]^{n\times n}$, $b \in \mathsf{K}[x]^{n\times m}$, $k \ge 2$
Output: $\mathrm{Trunc}(\mathrm{Inverse}(A, 2^k)\,b, 2^k)$
Condition: $X \perp \det A$ and $d = \deg X \ge \deg A$

1. $(E^{(1)}, E^{(2)}, \ldots, E^{(k-1)}) :=$ `HighOrderComponents`$[X](A, k-1)$;

2. # Let the X-adic expansion of b be $b_0 + b_1 X + b_2 X^2 + \cdots$.
   $B := [\ b_0 \mid b_1 \mid \cdots \mid b_{2^k-1}\ ]$;
   for i **from** $k-1$ **by** -1 **to** 1 **do**
   $\quad \bar{B} :=$ the first $m2^k - m2^i$ columns of B;
   $\quad \bar{B} := \mathrm{Left}(-A\,\mathrm{Trunc}(\mathrm{Left}(E^{(i)}\bar{B}), 1), 1), 1)$;
   $\quad \bar{B} := [\ O \mid \bar{B}\ ]$ where O is the $n \times m2^i$ zero matrix;
   $\quad B := B + \bar{B}$;
   od;
   $B := \mathrm{Trunc}(E^{(1)}B, 2)$;

3. # Let $B = [\ d_0 \mid d_1 \mid \cdots \mid d_{2^k-1}\ ]$.
   $B := d_0 + d_1\,X + d_2\,X^2 + \cdots + d_{2^k-2}\,X^{2^k-2} + \mathrm{Trunc}(d_{2^k-1}, 1)\,X^{2^k-1}$;
   return B

We now prove that the algorithm is correct. At the same time we estimate the complexity in terms of n, k and d. Let (A, b, X, k) be valid input tuple. Let $Z^{(i)} = \mathrm{Inverse}(A, 2^i)$ for $i > 0$. Phase 1 computes the high order components of $Z^{(k-1)}$ at a cost of $O(kn^\theta d^{1+\epsilon})$ field operations.

Phase 2 is identical to the corresponding phase in Algorithm 12 except that here we solve $m2^k$ systems in parallel. We need only observe that

$$\begin{aligned} \mathrm{Trunc}(Z^{(k)}b, k) &= \sum_{i=0}^{2^k-1} \mathrm{Trunc}(Z^{(k)}b_i\,X^i, k) \\ &= \sum_{i=0}^{2^k-1} X^i\,\mathrm{Trunc}(Z^{(k)}b_i, k-i) \end{aligned}$$

The cost of phase 2 bounded by $O((km2^k/n)n^\theta d^{1+\epsilon})$ field operations if $m2^k > n$. If $m2^k \le n$ the cost is is dominated by that of phase 1.

Phase 3 multiplies each column of B by the appropriate power of X and adds all the columns together. The cost of this phase is dominated by that of phase 2.

We have shown:

Proposition 17. *Algorithm* `SeriesSolution` *is correct. The cost of the algorithm is $O((k(1 + m2^k/n))n^\theta d^{1+\epsilon})$ field operations.*

Corollary 18. *Let $(A, b, X, *)$ be a valid input tuple to Algorithm 16, with $b \in \mathsf{K}[x]^{n\times 1}$. Assuming $\epsilon \le \theta - 2$ and $\deg b = O(nd)$, the vector $A^{-1}b \in \mathsf{K}(x)^{n\times m}$ can be computed in $O((\log n)n^\theta d^{1+\epsilon})$ field operations.*

8 High order lifting

Let $A \in \mathsf{K}[x]^{n\times n}$ be nonsingular, $\det A \perp X$. Let $b \in \mathsf{K}[x]^{n\times m}$. We present an algorithm to recover a contiguous segment of coefficients from the X-adic expansion of $A^{-1}b$. For example, the algorithm can be used to recover $H = \text{Left}(\text{Trunc}(\text{Inverse}(A, h+2^k)\, b, h+2^k), h)$.

$$A^{-1}b = b_0 + \cdots + b_{h-1}\, X^{h-1} + \overbrace{b_h\, X^h + \cdots + b_{h+2^k-1}\, X^{h+2^k-1}}^{HX^h} + \cdots .$$

If $h = 0$ then we can use Algorithm `SeriesSolution` to recover H. In high order lifting, what is important is that h be larger than some specified bound, say $h > 2^l$ for a given l. The particular value of h is not important, only that $h > 2^l$. The point of the algorithm here is that the complexity depends on 2^k and $\deg b$ but not (essentially) on h. This is important because in typical applications $h \gg 2^k$.

Algorithm 19. `HighOrderLift`$[X](A, b, l, k)$
Input: $A \in \mathsf{K}[x]^{n\times n}$, $b \in \mathsf{K}[x]^{n\times m}$, $l \geq 2$, $k \geq 2$
Output: $\text{Left}(\text{Trunc}(\text{Inverse}(A, h+2^k)\, b, h+2^k), h)$ for some $h > 2^l$.
Condition: $X \perp \det A$ and $d = \deg X \geq \deg A$

1. $\bar{k} :=$ the smallest integer ≥ 2 such $2^{\bar{k}} > \deg b$;
   $H :=$ `SeriesSolution`$[X](A, b, \bar{k})$;
   $H := \text{Left}(-A\,\text{Left}(H, 2^{\bar{k}} - 1), 1)$;

2. $(*, *, \ldots, *, E^{(l)}) :=$ `HighOrderComponents`$[X](A, l)$;
   $H := \text{Left}(-A\,\text{Trunc}(\text{Left}(E^{(l)}H, 1), 1), 1)$;

3. $H :=$ `SeriesSolution`$[X](A, H, k)$;
   return H

Correctness of the algorithm is easy to verify. Since the cost estimate depends on many paramaters, we only give a special case that interests us.

Proposition 20. *Algorithm* `HighOrderLift` *is correct. If* $l = O(\log n)$ *and* $m = O(n/\lceil \deg b/d\rceil + n/\lceil 2^k/d\rceil)$*, then the cost of the algorithm is* $O((\log n) n^\theta d^{1+\epsilon})$ *field operations.*

9 Integrality certification

Let $A \in \mathsf{K}[x]^{n\times n}$ be nonsingular, $\det A \perp X$. Let $B \in \mathsf{K}[x]^{n\times m}$ and $T \in \mathsf{K}[x]^{m\times m}$. We present an algorithm to assay if $A^{-1}BT$ is over $\mathsf{K}[x]$. The algorithm works by computing a high order lift H of $A^{-1}B$.

$$A^{-1}B = B_0 + \cdots + B_{h-1}\, X^{h-1} + \overbrace{B_h\, X^h + \cdots + B_{h+k-1}\, X^{h+k-1}}^{HX^h} + \cdots .$$

Recall that $H = \text{Left}(\text{Trunc}(\text{Inverse}(A, h+k)B, h+k), h)$. Let

$$C = \text{Trunc}(HT, k).$$

Proposition 21. *If h and k are chosen to satisfy $hd > (n-1)d + \deg B + \deg T$ and $kd > d + \deg T$, then C as computed above has $\deg C < (k-1)d$ if and only if $A^{-1}BT$ is integral.*

Proof. Let $S = \mathrm{Trunc}(A^{-1}BT, h+k)$. Then $\mathrm{Trunc}(AS, h+k) = BT$ and

$$S = \overbrace{\mathrm{Trunc}(A^{-1}B, h)T}^{\text{degree} < hd + \deg T} + CX^h. \tag{10}$$

By choice of k we have $hd + \deg T < hd + (k-1)d$. If $\deg CX^h < hd + (k-1)d$ also, then $AS = \mathrm{Trunc}(AS, h+k)$, whence $S = A^{-1}BT$. This shows the "only if". Now for the "if". The paramater h is chosen so that hd is strictly larger than an *a priori* bound on the degrees of numerators in $A^{-1}BT$. Thus, if $A^{-1}BT$ is integral, then it follows from (10) that $\deg C < \deg T$. □

The next two corollaries will be useful later on. The first follows from the proof above and the second is obvious.

Corollary 22. $\deg C < \deg T$ *if and only if $A^{-1}BT$ is integral.*

Corollary 23. *If $A^{-1}BT$ is integral, then C is invariant of the choice of k.*

In case of integrality, the algorthim returns also C, the *integrality certificate.*

Algorithm 24. `IntegralityCertificate`$[X](A, B, T)$
Input: $A \in \mathsf{K}[x]^{n\times n}$, $B \in \mathsf{K}[x]^{n\times m}$, $T \in \mathsf{K}[x]^{m\times m}$
Output: An integrality certificate if $A^{-1}BT$ is over $\mathsf{K}[x]$, false otherwise.
Condition: $X \perp \det A$ and $d = \deg X \geq \deg A$

1. $l :=$ the smallest integer such that $2^l d > (n-1)d + \deg b + \deg T$;
   $k :=$ the smallest integer such that $2^k d > d + \deg T$;
   $H :=$ `HighOrderLift`$[X](A, B, l, k)$;

2. $C := \mathrm{Trunc}(HT, 2^k)$;
   if $\deg C < \deg T$ **then**
      return (true, C)
   else
      return false
   fi

We get:

Proposition 25. *Algorithm* `IntegralityCertificate` *is correct. If $m = O(n/\lceil \deg B/d \rceil + n/\lceil \deg T/d \rceil)$, and assuming $\epsilon \leq \theta - 2$, the cost of the algorithm is $O((\log n)n^\theta d^{1+\epsilon})$ field operations.*

Worked example

The integrality certification technique described above can be adapted for integer matrices. Let

$$A = \begin{bmatrix} -28 & -11 & -56 & -39 \\ -5 & 42 & -10 & 37 \\ 22 & -44 & -25 & 44 \\ -32 & 3 & 38 & 46 \end{bmatrix}.$$

Let B to be the last two columns of I_2. Let $T = sI_4$ where $s = 3969$. Compute $H = \text{iquo}(\text{mods}(A^{-1}B, 10^{h+k}), 10^h)$ and $C = \text{mods}(HT, 10^k)$ for sufficiently large h and k. For $h = 90$ (overkill) and $k = 8$ we get

$$H = \begin{bmatrix} -12194507 & -23935500 \\ -24086671 & 42529604 \\ -5946082 & 33232552 \\ 24086671 & -42529604 \end{bmatrix} \quad \text{and} \quad C = \begin{bmatrix} 1717 & 500 \\ 2801 & -1724 \\ 542 & -1112 \\ -2801 & 1724 \end{bmatrix}. \tag{11}$$

We conclude that $A^{-1}BT$ is integral since entries in C have substantially fewer than 8 decimal digits. Indeed, the analogue of Corollary 22 guarantees that $||C||_\infty < m||T||_\infty$.

10 Determinant reduction

Let R denote the ring of integers $\mathbb{Z}$ or $\mathsf{K}[x]$. Let $A \in \mathsf{R}^{n\times n}$ be nonsingular. We are concerned with the principal $(n-1)\times(n-1)$ submatrix $\bar{H}$ of the Hermite basis of A. This depends only on the first $n-1$ columns of A and is invariant under a permutation of the rows. Thus, we may assume wlog that

$$A = \left[\begin{array}{c|c} \bar{A} & \\ \hline \bar{a} & 1 \end{array}\right]$$

where $\bar{A}$ nonsingular. Suppose A has Hermite basis

$$\left[\begin{array}{c|c} \bar{H} & * \\ \hline & h_n \end{array}\right].$$

In this section we show how to construct a matrix B, obtained from A by replacing the last column, such that B has Hermite basis

$$\left[\begin{array}{c|c} \bar{H} & \\ \hline & 1 \end{array}\right].$$

Thus, the determinant of B will be a factor of h_n smaller than that of A.

Now we sketch the procedure to construct B and at the same time give an example. Consider the matrix

$$A = \left[\begin{array}{cccc|c} 15 & 17 & 18 & -9 & \\ -14 & 10 & -9 & -3 & \\ 5 & -35 & 7 & 7 & \\ -14 & 5 & 29 & -37 & \\ \hline 31 & -15 & -25 & -19 & 1 \end{array}\right] \quad \text{with HF} \quad \left[\begin{array}{cccc|c} 1 & 0 & 0 & 0 & 5193 \\ & 1 & 0 & 0 & 7144 \\ & & 1 & 1 & 19657 \\ & & & 3 & 13421 \\ \hline & & & & 21619 \end{array}\right]$$

First recover the unique $u \in \mathsf{R}^{1\times n}$ such that $uA = \left[\begin{array}{c|c} 0 & h_n \end{array}\right]$. We get

$$u = \left[\begin{array}{c|c} \bar{u} & u_n \end{array}\right] = \left[\begin{array}{cccc|c} 359110 & 639246 & 335397 & -86830 & 21619 \end{array}\right].$$

Do an extended gcd computation to recover a $v \in \mathsf{R}^{n\times 1}$ such that $uv = 1$. We get

$$v = \left[\begin{array}{c} \bar{v} \\ \hline v_n \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \\ 6933 \\ 26780 \\ \hline 0 \end{array}\right].$$

The matrix obtained from A by replacing the last column with v will have Hermite-form as desired. The problem is that the entries in v may be too large (in general about n times the size of entries in A). We may obtained a reduced v by continuing as follows.

Compute the rational vector $\bar{c} = h_n \bar{A}^{-1}\bar{v}$. Compute an integer vector $\bar{r}$ such that each entry of $\bar{c} + h_n\bar{r}$ has magnitude $< h_n$. Set B equal to

$$B = \left[\begin{array}{c|c} \bar{A} & \bar{v} + \bar{A}\bar{r} \\ \hline \bar{a} & v_n - \bar{u}\bar{A}\bar{r}/u_n \end{array}\right].$$

In this example we get

$$B = \left[\begin{array}{ccccc} 15 & 17 & 18 & -9 & 21 \\ -14 & 10 & -9 & -3 & -9 \\ 5 & -35 & 7 & 7 & -12 \\ -14 & 5 & 29 & -37 & -28 \\ 31 & -15 & -25 & -19 & -9 \end{array}\right] \quad \text{with HF} \quad \left[\begin{array}{ccccc} 1 & 0 & 0 & 0 & \\ & 1 & 0 & 0 & \\ & & 1 & 1 & \\ & & & 3 & \\ & & & & 1 \end{array}\right].$$

In the case that $\mathsf{R} = \mathsf{K}[x]$ the analogous procedure will produce a B that satisfies $\deg B \le \deg A$. We get the following:

Proposition 26. *Let $A \in \mathsf{K}[x]^{n\times n}$ and $X \in \mathsf{K}[x]$ satisfy $\det A \perp X$ and $\deg X \ge \deg A$. If $\epsilon \le \theta - 2$, then the construction of B as described above can be accomplished in $O((\log n)n^\theta d^{1+\epsilon})$ field operations.*

11 Partial determinant computation

Let $A \in \mathsf{K}[x]^{n\times n}$ be nonsingular. Let $X \in \mathsf{K}[x]$ with $X \perp \det A$ and $d = \deg X \geq \deg A$ be given. Let $1 \leq m \leq n$ be given. Throughout this section let $\bar{H}$, $\bar{S}$, $\bar{s}$ and $\bar{L}$ be defined as follows:

- $\bar{H}$ is the trailing $m \times m$ submatrix of the Hermite basis of A.
- $\bar{S}$ is the Smith-form of $\bar{H}$.
- $\bar{s}$ is the largest invariant factor in $\bar{S}$.
- $\bar{L}$ is the last m rows of A^{-1}.

This section gives a novel algorithm to compute $\bar{S}$. The next two lemmas follow easily from the uniqueness of $\bar{H}$ and $\bar{s}$. For sundry such facts about lattices, see [12, Chapter 5].

Lemma 27. *Let $T \in \mathsf{K}[x]^{m\times m}$ be such that $T\bar{L}$ is integral. Then $T = \bar{T}\bar{H}$ for some $\bar{T} \in \mathsf{K}[x]^{m\times m}$.*

Lemma 28. *Let $s \in \mathsf{K}[x]$ be such that $s\bar{L}$ is integral. Then $s = \bar{t}\bar{s}$ for some $\bar{t} \in \mathsf{K}[x]$.*

Let B be the last m rows of I_n. Let C be the integrality certificate produced by calling $\texttt{IntegralityCertificate}(A^t, B^t, X, sI_m)$, except that C be transposed. Then $\deg C < \deg s$ (Lemma 22) and can be produced in time $O\tilde{~}(n^\omega d)$ field operations if $m = O(nd/\deg s)$ (Proposition 25).

Proposition 29. *$\bar{S}$ is the Smith-form of sD^{-1}, where D is the principal $m \times m$ submatrix of the Smith form of $\left[\begin{array}{c|c} C & sI_m \end{array}\right]$.*

Proof. Let H be the high order lift computed in the algorithm, except that H be transposed. The algorithm computes $C = \mathrm{Trunc}(sH, k)$ for some choice of k, $kd > d + \deg s$. We have $sI_m = \bar{T}\bar{H}$ for some $\bar{T}$ over $\mathsf{K}[x]$ (Lemma 27), so

$$C = \mathrm{Trunc}(\bar{T}\overbrace{\mathrm{Trunc}(\bar{H}H, k)}^{\bar{C}}, k). \tag{12}$$

Since $\bar{H}A^{-1}B$ is integral $\deg \bar{C} < \deg \bar{H}$ (Corollary 22). Since $\bar{T}$ and $\bar{C}$ are over $\mathsf{K}[x]$, we may conclude from (12) that $C = \bar{T}\bar{C}$ (Corollary 23).

We have established that

$$\left[\begin{array}{c|c} C & sI_m \end{array}\right] = \bar{T}\left[\begin{array}{c|c} \bar{C} & \bar{H} \end{array}\right].$$

Let G be the column Hermite-form of $\left[\begin{array}{c|c} \bar{C} & \bar{H} \end{array}\right]$. If $G = I_m$ then the D in the statement of the proposition is equal to the Smith-form of $\bar{T}$. Using $s\bar{T}^{-1} = \bar{H}$ it is easy to derive that sD^{-1} is equivalent to $\bar{H}$. Thus, we will be finished if we can show that $G = I_m$.

To arrive at a contradiction, suppose $G \neq I_m$. Both $G^{-1}\bar{C}$ and $G^{-1}\bar{H}$ are necessarily integral with degrees bounded by $\deg \bar{C}$ and $\deg \bar{H}$ respectively. But then $\deg \mathrm{Trunc}((G^{-1}\bar{H})H, k) \leq \deg \bar{C}$ which shows $(G^{-1}\bar{H})A^{-1}B$ is integral (Proposition 21). Lemma 27 now gives a contradiction. □

The Smith-form in Proposition 29 can be computed in $O(nm^{\theta-1}(\deg s)^{1+\epsilon})$ field operations (Lemma 7.14 in [12]).

Proposition 30. *Let A, X, s and m as described above be given. If $m = O(n/\lceil \deg s/d \rceil)$, and assuming $\epsilon \le \theta - 2$, then $\bar{S}$ as described above can be computed in $O((\log n)n^{\theta}d^{1+\epsilon})$ field operations.*

Worked example

The ideas described above carry over to the case of integer matrices. The matrix

$$A = \begin{bmatrix} -28 & -5 & 22 & -32 \\ -11 & 42 & -44 & 3 \\ -56 & -10 & -25 & 38 \\ -39 & 37 & 44 & 46 \end{bmatrix} \quad \text{has HF} \quad H = \left[\begin{array}{cc|cc} 1 & 220 & 0 & 379 \\ & 1231 & 2 & 670 \\ \hline & & 3 & 3792 \\ & & & 3969 \end{array}\right]$$

and

$$\bar{H} = \begin{bmatrix} 3 & 3792 \\ 0 & 3969 \end{bmatrix} \quad \text{has Smith form} \quad \bar{S} = \begin{bmatrix} 3 & \\ & 3969 \end{bmatrix}.$$

Let $s = 3969$ and C be the integrality certificate shown in (11), except transposed. Then the principal 2×2 submatrix of the Smith form of

$$\left[\begin{array}{cccc|cc} 1717 & 2801 & 542 & -2801 & 3969 & \\ 500 & -1724 & -1112 & 1724 & & 3969 \end{array}\right] \quad \text{is} \quad D = \begin{bmatrix} 1 & \\ & 1323 \end{bmatrix}.$$

Note that $sD^{-1} = \bar{S}$.

12 Determinant computation

Suppose $A \in \mathsf{R}^{n\times n}$ is nonsingular with $\det \perp X$ and $\deg X \ge \deg A$. We present an algorithm to compute the determinant of A. We are going to assume that A satisfies some rather strong conditions. First, assume wlog (up to augmentation with an identity matrix) that $n = 2^{k+1} - 1$ for some k. Decompose the Hermite basis H of C as

$$H = \begin{bmatrix} H_k & * & * & * \\ & \ddots & \vdots & \vdots \\ & & H_1 & * \\ & & & H_0 \end{bmatrix}$$

where H_i has dimension $2^i \times 2^i$. Let S_i be the Smith-form of H_i. The algorithm requires that

- (C1) $\mathrm{diag}(S_k, S_{k-1}, \ldots, S_1, S_0)$ is in Smith-form.
- (C2) The Hermite basis of $A[1 \ldots m+1, 1 \ldots m]$ is equal to the Hermite basis of $A[*, 1 \ldots m]$, $m = 2^k + 2^{k-1} + \cdots + 2^i$, $i = k, k-1, \ldots, 1$.

If any of these conditions are not satisfied, the algorithm will detect this and report failure. Given (C2) holds for a given m, we will assume wlog (up to a permutation of the first m rows) that $A[1\ldots m, 1\ldots m]$ is nonsingular.

The algorithm proceeds in stages for $i = 0, 1, 2, \ldots, k$. Stage $i = 0$ is to compute S_0. Now fix some i, $i > 0$. Let $m = 2^k + 2^{k-1} + \ldots + 2^i$. Let B be the matrix constructed from $A[1\ldots m+1, 1\ldots m+1]$ using the algorithm supporting Proposition 26. Let $C = A[m+2\ldots n, 1\ldots m]$.

Lemma 31. *(C2) holds for i if and only if* CB^{-1} *is integral.*

In case $\mathsf{R} = \mathsf{K}[x]$, the integrality of CB^{-1} can be assayed using Algorithm 24. Assume henceforth that CB^{-1} is integral. (If it isn't, report failure and terminate.)

At this point we have constructed an $(m+1) \times (m+1)$ matrix B which has Hermite-form equal to

$$\left[\begin{array}{c|c|c} * & * & \\ \hline & H_i & \\ \hline & & 1 \end{array}\right].$$

Our goal now is to recover the Smith-form of H_i. Let s be the smallest invariant factor in S_{i-1}, computed during the (previous) stage $i-1$. Finally, use Proposition 30 to either determine that $\mathrm{diag}(S_i, S_{i-1})$ is not in Smith-form or to recover S_{i-1}. Since $\mathrm{diag}(S_{i-1}, S_{i-2}, \ldots, S_0)$ is in Smith-form, we must have $\deg s \leq nd/m$. The cost is given by Proposition 30. Summing over all stages gives

Proposition 32. *Let* $A \in \mathsf{K}[x]^{n\times n}$ *be nonsingular. Let* $X \in \mathsf{K}[x]$ *with* $X \perp \det A$ *and* $\deg X \geq \deg A$ *be given. The algorithm described above will assay if* A *satisfies conditions* $(C1)$ *and* $(C2)$. *If so, the algorithm will produce* $\mathrm{diag}(S_k, S_{k-1}, \ldots, S_0)$. *Assuming* $\epsilon \leq \theta - 2$, *the cost of the algorithm is* $O((\log n)^2 n^\theta d^{1+\epsilon})$ *field operations.*

Consider assaying the integrality of $B^{-1}A[1\ldots m+1, *]$ at each stage.

Corollary 33. *Let* $\mathrm{diag}(S_k, S_{k-1}, \ldots, S_0)$ *be as in Proposition 32. Then we can assay that* $\mathrm{diag}(S_k, S_{k-1}, \ldots, S_0)$ *is the Smith-form of* A *in the same time.*

13 Conclusions

The algorithms in sections 4—11 are deterministic but require as input a small degree X such that $X \perp \det A$. See [7, Proof of Theorem 29] for a method of finding such an X randomly.

The algorithms in Section 12 requires that A satisfy some conditions. These are easy to achieve using the preconditioning technique as shown in [4]. Choose (nonsingular) matrices U and V uniformly and randomly from $S^{n\times n}$, S a subset of K with $\#S \geq 4dn^4$. Then UAV will satisfy all required conditions with probability at least 1/2 (see [4, Algorithm 3.2] and [13, Algorithm REDUCE]). If $\#\mathsf{K}$ is too small, work over an algebraic extension field.

Arguably the most important contribution of this paper is the idea of using high-order lifting to certify integrality. Without this technique, many of the algorithms we propose would be Monte Carlo instead of Las Vegas. The algorithm we have proposed for integrality certification and determinant/Smith-form computation are of immediate practical interest, especially because they certify the output.

The main task remaining is to extend the results here to the case of integer matrices. The reader may have already noticed that the key ideas in Section 5 and 9—12 carry over easily. The main difficulties to be solved are:

- Achieve a suitable preconditioning for the input matrix of the Smith-form computation.
- Get analogous versions of the lifting algorithms in Sections 4, 7 and 8.

To solve the first difficulty the results in [3] and [8] should prove useful. The idea is to modify the determinant algorithm to allow considerably weaker conditions than those outlined at the beginning of Section 12.

The crux of the second difficulty is that the absolute value norm over $\mathbb{Z}$, unlike the degree norm over $\mathsf{K}[x]$, is Archimedean; because integer addition has carries, the analogue of Lemma 1 does not hold. One solution to this is to do computation in a shifted-adic number system. We will present this in a future paper.

References

[1] J. Abbott, M. Bronstein, and T. Mulders. Fast deterministic computation of determinants of dense matrices. In S. Dooley, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '99*, pages 197–204. ACM Press, 1999.

[2] J. D. Dixon. Exact solution of linear equations using p-adic expansions. *Numer. Math.*, 40:137–141, 1982.

[3] W. Eberly, M. Giesbrecht, and G. Villard. Computing the Smith form of a dense integer matrix. In *Proc. 31st Ann. IEEE Symp. Foundations of Computer Science*, 2000.

[4] E. Kaltofen, M. S. Krishnamoorthy, and B. D. Saunders. Parallel algorithms for matrix normal forms. *Linear Algebra and its Applications*, 136:189–208, 1990.

[5] E. Kaltofen and G. Villard. Computing the sign or the value of the determinant of an integer matrix, a complexity survey. 2002. Submitted to the special issue on Congrès International Algèbre Linéaire et Arithmétique: Calcul Numérique, Symbolique et Parallèle, held in Rabat, Morocco, May 2001, 17 pages.

[6] R. T. Moenck and J. H. Carter. *Approximate algorithms to derive exact solutions to systems of linear equations.*, pages 65–72. Springer-Verlag, Berlin-Heidelberg-New York, 1979.

[7] T. Mulders and A. Storjohann. Diophantine linear system solving. In S. Dooley, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '99*, pages 281–288. ACM Press, 1999.

[8] T. Mulders and A. Storjohann. Certified diophantine dense linear system solving. Technical Report 355, Departement Informatik, ETH Zürich, Dec. 2000.

[9] T. Mulders and A. Storjohann. On lattice reduction for polynomial matrices. Technical Report 356, Departement Informatik, ETH Zürich, Dec. 2000.

[10] T. Mulders and A. Storjohann. Rational solutions of singular linear systems. In C. Traverso, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '00*, pages 242–249. ACM Press, 2000.

[11] V. Pan. Computing the determinant and the charactersitic polynomial of a matrix via solving linear systems of equations. *Inf. Proc. Letters*, 28:71–75, 1988.

[12] A. Storjohann. *Algorithms for Matrix Canonical Forms.* PhD thesis, ETH – Swiss Federal Institute of Technology, 2000.

[13] A. Storjohann and G. Labahn. Preconditioning of rectangular polynomial matrices for efficient Hermite normal form computation. In A. H. M. Levelt, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '95*, pages 119–125. ACM Press, 1995.