A Las Vegas Algorithm for Computing the Smith Form of a Nonsingular Integer Matrix

Stavros Birmpilis
Cheriton School of Computer Science
University of Waterloo
sbirmpil@uwaterloo.ca

George Labahn
Cheriton School of Computer Science
University of Waterloo
glabahn@uwaterloo.ca

Arne Storjohann
Cheriton School of Computer Science
University of Waterloo
astorjoh@uwaterloo.ca

ABSTRACT

We present a Las Vegas randomized algorithm to compute the Smith normal form of a nonsingular integer matrix. For an $A \in \mathbb{Z}^{n\times n}$, the algorithm requires $O(n^3(\log n + \log ||A||)^2(\log n)^2)$ bit operations using standard integer and matrix arithmetic, where $||A|| = \max_{ij} |A_{ij}|$ denotes the largest entry in absolute value. Fast integer and matrix multiplication can also be used, establishing that the Smith form can be computed in about the same number of bit operations as required to multiply two matrices of the same dimension and size of entries as the input matrix.

CCS CONCEPTS

• **Computing methodologies → Linear algebra algorithms**; *Exact arithmetic algorithms*; • **Mathematics of computing →** Probabilistic algorithms.

ACM Reference Format:
Stavros Birmpilis, George Labahn, and Arne Storjohann. 2020. A Las Vegas Algorithm for Computing the Smith Form of a Nonsingular Integer Matrix. In *International Symposium on Symbolic and Algebraic Computation (ISSAC '20), July 20–23, 2020, Kalamata, Greece.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3373207.3404022

1 INTRODUCTION

Any nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ is unimodularly equivalent to a unique diagonal matrix $S = \mathrm{diag}(s_1, s_2, \ldots, s_n)$ in Smith form. Here the diagonal entries, the invariant factors of A, are positive with $s_1 \mid s_2 \mid \cdots \mid s_n$, and unimodularly equivalent means there exist unimodular (with determinant ± 1) matrices $U, V \in \mathbb{Z}^{n\times n}$ such that $UAV = S$.

A natural goal for many computations on integer matrices is to design algorithms that have about the same cost as multiplying together two matrices of the same dimension and size of entries as the input matrix. If ω is a valid exponent for matrix multiplication — two $n \times n$ matrices can be multiplied in $O(n^\omega)$ operations from the domain of entries — then the target complexity is $(n^\omega \log ||A||)^{1+o(1)}$ bit operations, where $||A|| = \max_{ij} |A_{ij}|$ denotes the largest entry in absolute value, and the exponent $1 + o(1)$ indicates some missing $\log n$ and loglog $||A||$ factors. For randomized algorithms, in addition to stating the running time, we will indicate the type. A Monte Carlo type algorithm is allowed to return an incorrect result with probability at most 1/2. A Las Vegas type algorithm is allowed to report failure with probability at most 1/2, and if failure is not reported the output is certified to be correct.

The previously fastest algorithm for Smith form is due to Kaltofen and Villard [10]. They give a Las Vegas algorithm for computing the characteristic polynomial in time $(n^{3.2} \log ||A||)^{1+o(1)}$ assuming $\omega = 3$, and in time $(n^{2.695594} \log ||A||)^{1+o(1)}$ assuming the currently best known upper bound $\omega \leq 2.3728639$ for ω [5] and the best known bound for rectangular matrix multiplication [6]. Using their characteristic polynomial algorithm together with ideas of Giesbrecht [8], they obtain a Monte Carlo algorithm for Smith form with the same running time. In this paper we give a Las Vegas algorithm for Smith form in time $(n^\omega \log ||A||)^{1+o(1)}$.

A Las Vegas algorithm with the target complexity was previously known for the determinant [13]. Like that algorithm, we utilize a "dimension × precision ≤ invariant" compromise. By Hadamard's bound, $|\det A| = s_1 s_2 \cdots s_n \leq \Delta^n$ where $\Delta = n^{1/2}||A||$. Thus, the number of invariant factors with bitlength between $(1/2^i) \times n \log \Delta$ and $(1/2^{i-1}) \times n \log \Delta$ is bounded by 2^i. The determinant algorithm [13] embeds A into a matrix

$$C := \left[\begin{array}{c|ccc} A & & & \\ \hline B_t & I & & \\ \vdots & & \ddots & \\ B_0 & & & I \end{array}\right] \in \mathbb{Z}^{O(n)\times O(n)},$$

where $t = O(\log n)$ and the blocks $B_i \in \mathbb{Z}^{O(2^i)\times n}$ are chosen randomly, and then returns $|\det A| = (\det H_0)(\det H_1)\cdots(\det H_{t+1})$, where

$$H = \left[\begin{array}{c|ccc} H_{t+1} & * & \cdots & * \\ \hline & H_t & \cdots & * \\ & & \ddots & \vdots \\ & & & H_0 \end{array}\right].$$

is the (row) Hermite form of A. With high probability, the part of the determinant captured by H_{i+1}, which has about twice the dimension as H_i, can be computed at a precision about half that required to compute $\det H_i$. We remark that the diagonal blocks H_* are not computed explicitly, and the offdiagonal blocks $*$ of H are avoided entirely.

To obtain the Smith form and not just the determinant, and to facilitate certification of the output, we take here a more structured approach and compute a *Smith massager* for A. This is a tuple of $n \times n$ integer matrices (U, M, T, S) such that

$$B := \left[\begin{array}{c|c} A & \\ \hline & I_n \end{array}\right] \left[\begin{array}{c|c} I_n & \\ \hline U & I_n \end{array}\right] \left[\begin{array}{c|c} I_n & M \\ \hline & T \end{array}\right] \left[\begin{array}{c|c} I_n & \\ \hline & S^{-1} \end{array}\right] \in \mathbb{Z}^{2n\times 2n} \quad (1)$$

ISSAC '20, July 20–23, 2020, Kalamata, Greece
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-7100-1/20/07…$15.00
https://doi.org/10.1145/3373207.3404022

is integral, with T unit upper triangular and S nonsingular and in Smith form. The algorithm succeeds if we compute a *maximal* Smith massager, meaning that S is the Smith form of A. Since (1) implies $(\det B)(\det S) = \det A$, we can conclude from the uniqueness of the Smith form that the massager is maximal if and only if B is unimodular.

Our algorithm for computing the Smith form has three phases. Phase 1 uses a Monte Carlo approach [4, Theorem 2.1] to compute the largest invariant factor s_n of A. Phase 2 iteratively computes a Smith massager of A, together with the massaged matrix B in (1), which will be maximal with probability at least 1/2. Phase 3 uses a known algorithm [11] to assay if B is unimodular.

Phase 2 is the main part of our algorithm. It uses $O(\log n)$ iterations to build a Smith massager that extracts more and more invariant factors from A. The algorithm begins by initializing (U, M, T, S) to be the trivial Smith massager, with $U, M \in 0^{n\times n}$ and $T = S = I_n$. At the start of iteration $i = 0, 1, 2, \ldots$ we assume that the current Smith massager is such that B in (1) has the same Smith form as A but with the largest $2^i - 1 = 2^0 + 2^1 + \cdots + 2^{i-1}$ invariant factors replaced by 1. The goal at iteration i is then to compute and extract the next largest 2^i invariant factors. For example, at iterations $i = 0, 1$ and 2, the largest 1, 2 and 4 invariant factors of the current B are equal to (s_n), (s_{n-1}, s_{n-2}) and $(s_{n-3}, s_{n-4}, s_{n-5}, s_{n-6})$, respectively. Section 3 shows how to recover the largest 2^i invariant factors of B with high probability by computing a projection $B^{-1}J$ for a randomly chosen J with column dimension $O(2^i)$. We exploit the fact that if s is a multiple of the largest invariant factor of B, then the smallest 2^i invariant factors of sB^{-1} correspond to the largest 2^i invariant factors of B. In Sections 4 and 5 we show how to compute a Smith massager that will extract the largest 2^i invariant factors from B, while in Section 6 we show how to combine the partial Smith massagers obtained at each iteration.

Cost model

Following [7, Section 8.3], cost estimates are given using a function $\mathsf{M}(d)$ that bounds the number of bit operations required to multiply two integers bounded in magnitude by 2^d. We use $\mathsf{B}(d)$ to bound the cost of integer gcd-related computations such as the extended euclidean algorithm. We can always take $\mathsf{B}(d) = O(\mathsf{M}(d) \log d)$. If $\mathsf{M}(d) \in \Omega(d^{1+\epsilon})$ for some $\epsilon > 0$ then $\mathsf{B}(d) \in O(\mathsf{M}(d))$.

As usual, we assume that M is superlinear and subquadratic. We also assume that $\mathsf{M}(ab) \in O(\mathsf{M}(a)\,\mathsf{M}(b))$ for $a, b \geq 1$. We assume that $\omega > 2$, and to simplify cost estimates we make the assumption that $\mathsf{M}(d) \in O(d^{\omega-1})$. This assumption simply stipulates that if fast matrix multiplication techniques are used, then fast integer multiplication techniques should also be used. The assumptions stated in this paragraph apply also to B.

2 COMPUTATIONAL TOOLS

A key step during the iterations of phase 2 of our Smith form algorithm is to compute a projection $B^{-1}J$ for a partially massaged matrix B and a randomly chosen J. More specifically, we will have an $s \in \mathbb{Z}_{>0}$ such that sB^{-1} is integral, and what we need is $\operatorname{Rem}(sB^{-1}J, s)$, that is, the matrix $sB^{-1}J$ with entries reduced modulo s. In this section we show how to compute $\operatorname{Rem}(sB^{-1}J, s)$ within the target complexity by reducing to a deterministic variant of high-order lifting [11, Section 3] for linear system solving.

There are two issues that arise that prevent a direct application of fast linear system solving. First, the massaged matrix B may have some entries with large bitlength, adversely affecting the cost. In Section 2.1 we recall a partial linearization technique that can be used to obtain a matrix with smoothed entries that can be used in lieu of B. Second, entries in $sB^{-1}J$ may have bitlength much larger than the bitlength of s. In Section 2.2 we develop a deterministic variant of integrality certification that allows $\operatorname{Rem}(sB^{-1}J, s)$ to be computed more directly, without computing $sB^{-1}J$ first.

2.1 Partial linearization

The number of bits in the binary representation of a positive integer a is $\lfloor \log_2 a \rfloor + 1$. Any integer a thus satisfies $|a| \leq 2^{\operatorname{length}(a)} - 1$ where

$$\operatorname{length}(a) := \begin{cases} 1 & \text{if } a = 0 \\ \lfloor \log_2 |a| \rfloor + 1 & \text{otherwise} \end{cases}.$$

For v an integer vector or matrix, define $\operatorname{length}(v) := \operatorname{length}(||v||)$.

The cost of high-order lifting [11, Section 3] is sensitive to $\operatorname{length}(A)$. This is an issue because some of the intermediate matrices that we will need to give as input to the high-order lifting algorithm will likely have some rows of large length, even though the average row length is well bounded. In some cases, for an $n \times n$ input matrix A, $\operatorname{length}(A)$ could be about n times as large as the average row length. For the purposes of giving a concrete example, and not considering such an extreme case, consider the input matrix

$$A = \begin{bmatrix} 7 & 4 & 9 & 10 \\ 1 & 1 & 3 & 7 \\ 58538 & 43609 & 77404 & 7995 \\ 72526300 & 20544909 & 66620465 & 80378234 \end{bmatrix}.$$

The lengths of the rows of A are $[4, 3, 17, 27]$. Thus $\operatorname{length}(A) = 27$. But the average length is bounded by $d := \lceil (4 + 3 + 17 + 27)/4 \rceil = 13$. With some adjustment, the partial linearization technique [9, Section 6] developed for polynomial matrices can be applied in the integer setting. Assuming integers are represented in binary, the technique allows to produce from A without computation a new matrix

$$\bar{A} = \begin{bmatrix} 7 & 4 & 9 & 10 & 0 & 0 & 0 \\ 1 & 1 & 3 & 7 & 0 & 0 & 0 \\ 1194 & 2649 & 3676 & 7995 & -8192 & 0 & 0 \\ 2524 & 7565 & 3121 & 6522 & 0 & -8192 & 0 \\ 7 & 5 & 9 & 0 & 1 & 0 & 0 \\ 661 & 2507 & 8132 & 1619 & 0 & 1 & -8192 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

that satisfies $||\bar{A}|| \leq 2^d$ and can be used in lieu of A. The next theorem summarizes the special case of partial linearization that we require.

Theorem 1. *Let $A \in \mathbb{Z}^{n\times n}$ have average row length bounded by $d \in \mathbb{Z}_{\geq 0}$. If the (2^d)-adic expansions of entries of A are available, we can construct without computation from A a new matrix $\bar{A} \in \mathbb{Z}^{\bar{n}\times\bar{n}}$ such that $\bar{n} < 2n$, $||\bar{A}|| \leq 2^d$, $\det \bar{A} = \det A$, and, if A is nonsingular, with the principal $n \times n$ submatrix of $\bar{A}^{-1}$ equal to A^{-1}.*

2.2 Integrality certification

Any rational number can be written as an integer and a proper fraction. For example,

$$\frac{9622976468279041913}{21341} = 450914974381661 + \frac{14512}{21341},$$

where 450914974381661 is the quotient and 14512 is the remainder of the numerator with respect to the denominator. A similar construction replaces the quotient with a truncated p-adic expansion of the fraction, where p should be relatively prime to the denominator. For example,

$$\frac{9622976468279041913}{21341} = 9035820194880943821 - \frac{10453}{21341} \times 2^{64}. \quad (2)$$

In our case, we only require the remainder and not the quotient. Multiplying (2) by 21341 shows that

$$14512 = -10453 \times 2^{64} \bmod 21341.$$

The same idea works for integer matrices. Suppose we are given a nonsingular integer matrix $A \in \mathbb{Z}^{n\times n}$, a $B \in \mathbb{Z}^{n\times m}$ and an $s \in \mathbb{Z}_{>0}$. Then integrality certification [13, Section 11] can test if $sA^{-1}B$ is integral, and, if so, return the matrix $\text{Rem}(sA^{-1}B, s)$. High-order lifting is used to achieve a cost that is sensitive to $\text{length}(s) + \text{length}(B)$, rather than $\text{length}(sA^{-1}B)$. The algorithm in [13] is randomized. Here we show how to solve the integrality certification problem deterministically using some recently developed techniques, provided that $\det A \perp 2$.

Our approach is to first use double-plus-one lifting [11, Section 3] to compute a high-order residue $R \in \mathbb{Z}^{n\times n}$ such

$$A^{-1} = D + A^{-1}R \times 2^h \quad (3)$$

for some h such that

$$2^h > 2sn^{n/2}||A||^{n-1}||B||. \quad (4)$$

The matrix D, which will satisfy $||D|| \le (0.6)2^h$ [11, Theorem 5], is not needed and not computed explicitly. If the dimension $\times$ precision compromise

$$m \times (\log s + \log ||B||) \in O(n(\log n + \log ||A||)) \quad (5)$$

holds, then by [11, Theorem 8] such an R can be computed in time

$$O(n^{\omega}\, \mathsf{M}(\log n + \log ||A||) \log n). \quad (6)$$

Now multiply equation (3) on the right by sB to see that

$$sA^{-1}B = sDB + A^{-1}(sRB) \times 2^h. \quad (7)$$

The next step is to use deterministic linear solving [1, Section 3] to compute $\text{Rem}(A^{-1}(sRB), 2^{\ell})$ for some ℓ such that

$$2^{\ell} > 2n||A||(0.6sn||B||). \quad (8)$$

Assuming (5), this can also be done in time (6) [1, Corollary 7].

Adjusting slightly the argument of the proof of [13, Theorem 46] to account for the fact that D in (3) satisfies $||D|| \le (0.6)2^h$ instead of $||D|| \le 2^h$, it can be shown, for the choices of h and ℓ in (4) and (8), respectively, that if C is set to be the matrix equal to $\text{Rem}(A^{-1}(sRB), 2^{\ell})$ but with entries reduced in the symmetric range modulo 2^{ℓ}, then $C = sA^{-1}RB$ (and hence $sA^{-1}RB$ is integral) if and only if $||C|| < 0.6sn||B||$. Considering (7), it then follows that $\text{Rem}(C \times 2^h, s)$ is equal to $\text{Rem}(sA^{-1}B, s)$.

We will need to apply integrality certification with an input matrix A that has skewed row lengths. To maintain a good complexity, we can work with the partial linearization $\bar{A} \in \mathbb{Z}^{\bar{n}\times\bar{n}}$ of Theorem 1. Compute a high-order residue $\bar{R}$ for $\bar{A}$. Let $\bar{B} \in \mathbb{Z}^{\bar{n}\times m}$ be equal to B but augmented with $\bar{n} - n$ zero rows. Then the first n rows of $\text{Rem}(\bar{A}^{-1}(s\bar{R}\bar{B}), 2^{\ell})$ comprise an integrality certificate for $sA^{-1}B$. The running time is as in (6) but with $\log ||A||$ replaced by the average of the lengths of the rows. This gives the following.

Theorem 2. *Let $A \in \mathbb{Z}^{n\times n}$ satisfying $\det A \perp 2$, $s \in \mathbb{Z}_{>0}$, and $B \in \mathbb{Z}/(s)^{n\times m}$ be given. There exists an algorithm that will test if $sA^{-1}B$ is integral, and, if so, return the matrix $\text{Rem}(sA^{-1}B, s)$. If $m \times \log s \in O(n(d + \log n))$, where d is the average of the lengths of the rows of A, then the cost is $O(n^{\omega}\, \mathsf{M}(d + \log n) \log n)$.*

3 LARGEST INVARIANT FACTORS

In this section we show how the largest r invariant factors of a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ can be recovered with high probability by randomly sampling $r + O(\log r)$ vectors from the columns space of A^{-1}. The method assumes we know an $s \in \mathbb{Z}_{>0}$ that is a multiple of the largest invariant factor s_n of A.

Let $U, V \in \mathbb{Z}^{n\times n}$ be unimodular with $S = UAV = \text{diag}(s_1, \ldots, s_n)$ the Smith form of A. Then the *reverse Smith form* of $sA^{-1} \in \mathbb{Z}^{n\times n}$ is equal to $sS^{-1} = \text{diag}(s/s_1, \ldots, s/s_n)$. By reverse Smith form we simply mean that the order of both the rows and the columns is reversed. The smallest invariant factor is thus located in the last row and column. Since the largest invariant factor s/s_1 of the Smith form of sA^{-1} is a divisor of s, the Smith form of sA^{-1} can be computed modulo s over $\mathbb{Z}/(s)$. For convenience, should a diagonal entry in the Smith form over $\mathbb{Z}/(s)$ vanish modulo s, we replace it with s. For example, the reverse Smith form of $\text{diag}(1, 2, 8, 16, 16)$ over $\mathbb{Z}/(16)$ is equal to $\text{diag}(16, 16, 8, 2, 1)$.

To recover only the largest r invariant factors of A, the idea is to choose $J \in \mathbb{Z}/(s)^{n\times r}$ uniformly at random and hope that the submatrix comprised of the last r rows of the reverse Smith form of $sA^{-1}J \in \mathbb{Z}/(s)^{n\times r}$ is equal to $S_1 = \text{diag}(s/s_{n-r+1}, \ldots, s/s_n)$. To ensure a high probability of success, we adjust the recipe slightly by augmenting J with a small number of additional columns k. The main result of this section is:

Theorem 3. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular with Smith form $S = \text{diag}(s_1, \ldots, s_n)$. Let $s \in \mathbb{Z}_{>0}$ be a multiple of s_n. If $J \in \mathbb{Z}/(s)^{n\times(r+k)}$ is chosen uniformly at random for $r \ge 1$ and $k \ge 2$, then, with probability at least $1 - \frac{1}{2^{k-1}}$, the trailing $r \times r$ submatrix of the reverse Smith form of $sA^{-1}J$ over $\mathbb{Z}/(s)$ is equal to $S_1 = \text{diag}(s/s_{n-r+1}, \ldots, s/s_n)$.*

Before we prove Theorem 3, we establish a property of $J \in \mathbb{Z}/(s)^{n\times(r+k)}$ that is sufficient to ensure success. In the following lemma, recall that $U, V \in \mathbb{Z}^{n\times n}$ are unimodular matrices such that $S = UAV$, and thus $sA^{-1} = VsS^{-1}U$.

Lemma 4. *If the $r \times (r + k)$ submatrix comprised of the last r rows of $UJ \in \mathbb{Z}^{n\times(r+k)}$ is right equivalent to $\left[\begin{array}{c|c} 0_{r\times k} & I_r \end{array}\right]$ over $\mathbb{Z}/(s)$, then the trailing $r \times r$ submatrix of the reverse Smith form of $sA^{-1}J$ over $\mathbb{Z}/(s)$ is equal to $S_1 = \text{diag}(s/s_{n-r+1}, \ldots, s/s_n)$.*

Proof. Decompose $sS^{-1} = \text{diag}(S_2, S_1)$ where S_1 is as in the statement of the theorem, and $S_2 = \text{diag}(s/s_1, \ldots, s/s_{n-r})$.

We work entirely over $\mathbb{Z}/(s)$. By assumption, we have that

$$UJ \equiv_R \left[\begin{array}{c|c} U_1 & U_2 \\ \hline & I_r \end{array}\right] \tag{9}$$

for $U_1 \in \mathbb{Z}/(s)^{(n-r)\times k}$ and $U_2 \in \mathbb{Z}/(s)^{(n-r)\times r}$. Since all entries in S_2 are divisible by the largest invariant factor s/s_{n-r+1} of S_1, it will be sufficient to show that

$$sA^{-1}J \equiv \left[\begin{array}{c|c} S_2U_1 & \\ \hline & S_1 \end{array}\right].$$

We have

$$\begin{aligned} sA^{-1}J &= V(sS^{-1})UJ & (10)\\ &\equiv_L (sS^{-1})UJ & (11)\\ &\equiv_R (sS^{-1})\left[\begin{array}{c|c} U_1 & U_2 \\ \hline & I_r \end{array}\right] & (12)\\ &= \left[\begin{array}{c|c} S_2U_1 & S_2U_2 \\ \hline & S_1 \end{array}\right] \equiv_L \left[\begin{array}{c|c} S_2U_1 & \\ \hline & S_1 \end{array}\right] & (13) \end{aligned}$$

Here, (10) follows from $S = UAV$, (11) because V is unimodular, and (12) from (9). To obtain (13), we can use a unimodular left transformation to zero out the block S_2U_2 since its entries are all multiples of the diagonal entries in S_1. □

We need two additional technical lemmas before proving the main theorem.

Lemma 5. *If $k \geq 1$, $t \geq 0$ and $0 < x \leq 1/2$, then $\prod_{i=k}^{k+t}(1-x^i) \geq 1 - 2x^k + x^{k+t}$.*

Proof. We will use induction on t. For $t = 0$ the inequality is trivially true. We assume that $\prod_{i=k}^{k+t}(1-x^i) \geq 1 - 2x^k + x^{k+t}$ for fixed t, and we need to show the same for $t \leftarrow t+1$.

$$\begin{aligned} \prod_{i=k}^{k+t+1}\left(1-x^i\right) &= (1-x^{k+t+1})\prod_{i=k}^{k+t}\left(1-x^i\right)\\ &\geq (1-x^{k+t+1})(1-2x^k+x^{k+t})\\ &= 1-2x^k+x^{k+t}-x^{k+t+1}+2x^{2k+t+1}-x^{2(k+t)+1}\\ &= 1-2x^k+x^{k+t+1}\left(1+\frac{1}{x}-2+2x^k-x^{k+t}\right)\\ &\geq 1-2x^k+x^{k+t+1} \end{aligned}$$

In the last step we used that $x \leq 1/2$. □

Lemma 6. *If $k \geq 2$, then $\zeta(k+1) - 1 < 2^{-k}$, where ζ denotes the Riemann zeta function.*

Proof. The lemma inequality is equivalent to:

$$\zeta(k+1)-1 < 2^{-k} \Leftrightarrow \sum_{n=2}^{\infty}\frac{1}{n^{k+1}} < 2^{-k} \Leftrightarrow \sum_{n=2}^{\infty}\left(\frac{2}{n}\right)^{k+1} < 2.$$

Since the left-hand side of the last inequality is a decreasing function on k, it suffices to show the claim for $k = 2$, i.e., $\zeta(3) - 1 < \frac{1}{4}$. □

Proof (of Theorem 3). We start by defining the following event.

FR_p: For a prime p that divides s, the last r rows of the random matrix $J \in (\mathbb{Z}/(s))^{n\times(r+k)}$ have full row rank over $\mathbb{Z}/(p)$.

If the last i rows of J over $\mathbb{Z}/(p)$ are linearly independent, then they span a vector space containing p^i rows. The probability that an additional row avoids that space is $(1 - p^i/p^{r+k})$, and thus

$$\Pr[FR_p] = \prod_{j=k+1}^{r+k}\left(1-\frac{1}{p^j}\right).$$

The above result has already been shown and extensively used in the literature [2, 3]. Furthermore, by applying Lemma 5, we obtain

$$\Pr[\neg FR_p] \leq 2\frac{1}{p^{k+1}}. \tag{14}$$

Next, we define the event described by Lemma 4.

FR_U: For a matrix $U \in \mathbb{Z}^{n\times n}$, the last r rows of the random matrix $UJ \in (\mathbb{Z}/(s))^{n\times(r+k)}$ are right equivalent to $\left[\begin{array}{c|c} 0_{r\times k} & I_r \end{array}\right]$ over $\mathbb{Z}/(s)$.

A matrix J is right equivalent to $\left[\begin{array}{c|c} 0_{r\times k} & I_r \end{array}\right]$ over $\mathbb{Z}/(s)$ if and only if it has full row rank over $\mathbb{Z}/(p)$ for all primes p that divide s. Therefore,

$$\begin{aligned} \Pr[\neg FR_{I_n}] &\leq \sum_{\substack{p\mid s\\ p \text{ prime}}}\Pr[\neg FR_p] & (15)\\ &\leq 2\sum_{p=2}^{\infty}\frac{1}{p^{k+1}} & (16)\\ &= 2(\zeta(k+1)-1) < 2^{1-k}. & (17) \end{aligned}$$

We applied the union bound in (15), equation (14) in (16), and Lemma 6 in (17).

Finally, multiplying matrices from $\mathbb{Z}/(s)^{n\times(r+k)}$ with a unimodular matrix $U \in \mathbb{Z}^{n\times n}$ is an isomorphism back to $(\mathbb{Z}/(s))^{n\times(r+k)}$, which implies that $\Pr[FR_{I_n}] = \Pr[FR_U]$. So, according to Lemma 4, the probability described in Theorem 3 must be at least $\Pr[FR_U] = \Pr[FR_{I_n}] > 1 - \frac{1}{2^{k-1}}$. □

4 PROJECTION BASIS

Throughout this section let $A \in \mathbb{Z}^{n\times n}$ be nonsingular. In Section 3 we showed that the projection $A^{-1}J$, for a well chosen integer matrix J, can reveal the r largest invariant factors of A. In this section we show how these invariant factors can be extracted from A to produce a matrix B that has the same Smith form as A but with the r largest invariant factors replaced by trivial ones.

For any $J \in \mathbb{Z}^{n\times *}$, the set

$$\mathrm{Proj}(A, J) := \{v \in \mathbb{Z}^{1\times n} \mid vA^{-1}J \in \mathbb{Z}^{1\times r}\}$$

forms an integer lattice. A basis of $\mathrm{Proj}(A, J)$ is a matrix $H \in \mathbb{Z}^{n\times n}$ such that the set of all integer linear combinations of rows of H is equal to $\mathrm{Proj}(A, J)$. Bases of $\mathrm{Proj}(A, J)$ are always nonsingular and are unique up to left equivalence. For example, a basis of $\mathrm{Proj}(A, 0_{n\times *})$ is I_n, while a basis of $\mathrm{Proj}(A, I_n)$ is given by A itself.

Lemma 7. *If H is a basis of $\mathrm{Proj}(A, J)$, then AH^{-1} is integral.*

Proof. Since the rows of A belong to $\mathrm{Proj}(A, J)$, there exists a $B \in \mathbb{Z}^{n\times n}$ such that $A = BH$, hence $AH^{-1} = B$ is integral. □

The next two lemmas follow directly from the definition of $\mathrm{Proj}(A, J)$.

Lemma 8. *If $s \in \mathbb{Z}_{>0}$ is such that $sA^{-1}J$ is integral, and $P = \mathrm{Rem}(sA^{-1}J, s)$, then*

$$\mathrm{Proj}(A, J) = \mathrm{Proj}(sI, P) = \{v \in \mathbb{Z}^{1\times n} \mid \mathrm{Rem}(vP, s) = 0\}.$$

Lemma 9. *Let $U \in \mathbb{Z}^{n\times n}$ be unimodular. Then H is a basis of $\mathrm{Proj}(AU^{-1}, J)$ if and only if HU is a basis of $\mathrm{Proj}(A, J)$.*

Our final lemma will be used in the following sections to design an algorithm for computing a Smith massager.

Lemma 10. *Suppose the Smith form of A is $S = \mathrm{diag}(S_2, S_1)$, where $S_1 \in \mathbb{Z}^{r\times r}$ and $S_2 \in \mathbb{Z}^{(n-r)\times(n-r)}$. If $U \in \mathbb{Z}^{n\times n}$ is unimodular such that $H = \mathrm{diag}(I_{n-r}, S_1)$ is a basis of $\mathrm{Proj}(AU^{-1}, J)$, then the Smith form of $AU^{-1}H^{-1}$ is $\mathrm{diag}(I_r, S_2)$.*

5 MAXIMAL INDEX SMITH MASSAGER

In this section we combine all the results from the previous sections to present a randomized algorithm for the Problem `IndexMassager` shown in Figure 1. We begin with the following definition.

Definition 11 (Index-(m, r) Smith massager). *Let $B \in \mathbb{Z}^{2n\times 2n}$ be nonsingular with the shape*

$$B = \begin{bmatrix} A & & * \\ & I_{n-m} & \\ * & & * \end{bmatrix}.$$

For $m, r \in \mathbb{Z}_{\geq 0}$ such that $m + r \leq n$, an index-(m, r) Smith massager *for B is a tuple $(U, M, T, S) \in (\mathbb{Z}^{r\times n}, \mathbb{Z}^{n\times r}, \mathbb{Z}^{r\times r}, \mathbb{Z}^{r\times r})$ such that the matrix*

$$C := B \begin{bmatrix} I_n & & & \\ & I & & \\ U & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & M & \\ & I & & \\ & & T & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & S^{-1} & \\ & & & I_m \end{bmatrix} \tag{18}$$

is integral, with S nonsingular and in Smith form, and T unit upper triangular. We say that (U, M, T, S) is maximal *for B if S is comprised of the r largest invariant factors of the Smith form of B.*

Notice that when $m = 0$ the matrix B is equal to $\mathrm{diag}(A, I_n)$. When, in addition, $r = n$, an index-(m, r) Smith massager for $\mathrm{diag}(A, I_n)$ corresponds to a Smith massager for A as defined in the introduction.

`IndexMassager`$(B, n, m, r, s, \epsilon)$

Input: B, n, m and r are as in Definition 11. In addition, $s \in \mathbb{Z}_{>0}$ and ϵ is such that $0 < \epsilon < 1$.

Output: An index-(m, r) Smith massager (U, M, T, S) for B with $T = I_r$, entries in U and M reduced modulo s, and S_{rr} a divisor of s.

Note: If s is a positive integer multiple of the largest invariant factor of B, and the last n rows and columns of B^{-1} are integral, then a maximal index-(m, r) Smith massager for B is returned with probability at least $1 - \epsilon$.

Figure 1: Problem `IndexMassager`

In the design of the algorithm we are assuming that s is a multiple of the largest invariant factor of B and that the last n rows and columns of B^{-1} are integral. If, during the course of the algorithm, we detect that either of these conditions is not satisfied then we simply return the trivial index-(m, r) Smith massager $(0_{r\times n}, 0_{n\times r}, I_r, I_r)$ in order to satisfy the output requirements of the problem.

As shown in Section 4, we can "massage" away a block of the largest invariant factors of B by computing a basis of $\mathrm{Proj}(B, J)$ for a well chosen

$$J := \left[\begin{array}{c} J_1 \\ \hline J_2 \end{array}\right] \in \mathbb{Z}^{(n+n)\times r}.$$

Note that under the assumption that the last n columns of B^{-1} are integral, the basis $\mathrm{Proj}(B, J)$ will remain invariant of the choice of entries in the block $J_2 \in \mathbb{Z}^{n\times r}$. For this reason, we set J_2 to be the zero matrix. Entries in J_1 are chosen independently and uniformly at random from $\mathbb{Z}/(s)$.

Next, we use the algorithm supporting Theorem 2 to check if $sB^{-1}J$ is integral, and, if so, compute the projection

$$P := \mathrm{Rem}(sB^{-1}J, s) = \left[\begin{array}{c} P_1 \\ \hline P_2 \end{array}\right] \in \mathbb{Z}/(s)^{(n+n)\times r}.$$

Under the assumption that the last n rows of B^{-1} are integral, we expect P_2 to be the $n \times r$ zero matrix. If $sB^{-1}J$ is determined not to be integral, or P_2 is not the zero matrix, then we abort and return the trivial index-(m, r) massager for B.

At this point, by Lemma 8, we have reduced the problem of computing a basis of $\mathrm{Proj}(B, J)$ to that of computing a basis of $\mathrm{Proj}(sI, P)$. A basis of $\mathrm{Proj}(sI, P)$ can be computed as follows. First, using the Smith form algorithm from [12, Section 7], compute matrices $U \in \mathbb{Z}^{r\times n}$ and $V \in \mathbb{Z}^{r\times r}$, such that $\det V \perp s$ and $D := \mathrm{Rem}(-UP_1V, s)$ is congruent to the reverse Smith form of $P_1 \in \mathbb{Z}^{n\times r}$ over $\mathbb{Z}/(s)$. Then, we have the relations

$$-UP_1V = D \bmod s \ \text{ and } \ P_1V = MD \bmod s,$$

for some integer matrix $M \in \mathbb{Z}^{n\times r}$. We can put those two together and obtain

$$\begin{bmatrix} I_n & & & \\ & I & & \\ -U & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} P_1 \\ \\ \\ \\ \end{bmatrix} V = \begin{bmatrix} MD \\ \\ D \\ \\ \end{bmatrix} \bmod s.$$

Next, we apply a unimodular left transformation to zero out the block MD, and we take right equivalence to omit V.

$$\begin{bmatrix} I_n & & -M & \\ & I & & \\ & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ -U & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} P_1 \\ \\ \\ \\ \end{bmatrix} \equiv_R \begin{bmatrix} \\ \\ D \\ \\ \end{bmatrix} \bmod s \tag{19}$$

Finally, define $S := sD^{-1}$, which will be in regular Smith form, and notice that S is a basis of $\mathrm{Proj}(S, I_r) = \mathrm{Proj}(sI, D)$, which corresponds to the non-zero part of the matrix in the right-hand side of (19). Therefore,

$$\begin{bmatrix} I_n & & & \\ & I & & \\ & & S & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & -M & \\ & I & & \\ & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ -U & & I_r & \\ & & & I_m \end{bmatrix} \tag{20}$$

must be a basis of $\mathrm{Proj}(sI, P)$ according to Lemma 9, since the two matrices containing M and U are unimodular. Postmultiplying B by the inverse of this basis results in an integer matrix according to Lemma 7. That is,

$$C := B \begin{bmatrix} I_n & & & \\ & I & & \\ U & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & M & & \\ & I & & \\ & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & S^{-1} & \\ & & & I_m \end{bmatrix}.$$

Therefore, matrices (U, M, I_r, S) form an index-(m, r) Smith massager in accordance with Definition 11.

Theorem 12. *If $r \times \log s \in O(n(d + \log n))$, where d is the average of the lengths of the rows of B, and $\epsilon = \frac{1}{8r}$, then Problem* `IndexMassager` *can be solved in time $O(n^{\omega}\, \mathsf{B}(d + \log n) \log n)$.*

Proof. The correctness of the algorithm follows directly from the preceding discussion. The proposed massager fits the description of Definition 11. We achieve the probabilistic result, for $\epsilon = \frac{1}{8r}$, by exploiting Theorem 3. Instead of working with the projection $sB^{-1}J \in \mathbb{Z}^{2n\times r}$, we augment J with $k := \log_2 r + 4$ columns. After the Smith form computation, we keep only the last r rows of U, the last r columns of M, and the r largest invariant factors of S. This massager will be maximal with probability at least $1 - \frac{1}{2^{k-1}} = 1 - \frac{1}{8r}$.

Finally, regarding the running time, the algorithm consists of only two computational parts. The first is to test if $sB^{-1}J$ is integral, and, if so, compute P. By Theorem 2 this can be done in time $O(n^{\omega}\, \mathsf{M}(d + \log n) \log n)$. The second is the reverse Smith form computation: by [12, Corollary 7.17] which can be done in time $O(n^{\omega}\, \mathsf{B}(d + \log n) \log n)$, after simplifying the cost estimate using our assumptions on B. ☐

5.1 Reduced index Smith massager

In this subsection, we introduce the notion of the reduced Smith massager, which keeps the overall size of the matrices well bounded.

Denote by $U \operatorname{rmod} S$ the matrix obtained from U by reducing entries in row i modulo S_{ii}, $1 \le i \le r$. Similarly, we denote by $M \operatorname{cmod} S$ the matrix obtained from M by reducing entries in column j modulo S_{jj}, $1 \le j \le r$.

Definition 13 (Reduced index-(m, r) Smith massager). *Let (U, M, T, S) be an index-(m, r) Smith massager for $B \in \mathbb{Z}^{2n\times 2n}$ as in Definition 11. We say that (U, M, T, S) is* reduced *if $U = U \operatorname{rmod} S$, $M = M \operatorname{cmod} S$ and $T = ((T - I_r) \operatorname{cmod} S) + I_r$*

Lemma 14. *Suppose (U, M, T, S) is an index-(m, r) Smith massager for $B \in \mathbb{Z}^{2n\times 2n}$ as in Definition 11. Let $U' = U \operatorname{rmod} S$ and $M' = M \operatorname{cmod} S$. Let T' be the matrix obtained from $-U'M' \operatorname{cmod} S$ except with diagonal entry T'_{ii} reset to 1 when $S_{ii} = 1$, $1 \le i \le r$. Then (U, M', T, S) and (U, M, T', S) are index-(m, r) massagers for B, and (U', M', T', S) is a reduced index-(m, r) Smith massager for B.*

Proof. Without loss of generality, in order to simplify the presentation, we consider the case of an index-$(0, m)$ Smith massager. By multiplying together the first three matrices in (21) we obtain

$$B = \begin{bmatrix} A & & & AM \\ & I & & \\ & & I_r & \\ U & & & (T + UM) \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & I_r & \\ & & & S^{-1} \end{bmatrix}.$$

Note that the property that AMS^{-1} is integral is equivalent to $AM \operatorname{cmod} S$ being the zero matrix. But then $A(M \operatorname{cmod} S) \operatorname{cmod} S$ is also the zero matrix. This shows that (U, M', T, S) is an index massager. A similar argument shows that (U, M, T', S) is an index massager. By the definition of T',

$$(T' + (U \operatorname{rmod} S)(M \operatorname{cmod} S)) \operatorname{cmod} S$$

is also the zero matrix. Since T is unit upper triangular and also $(T + UM) \operatorname{cmod} S$ is the zero matrix, we have that $-UM \operatorname{cmod} S$ is unit upper triangular, except that the i'th diagonal entry will be zero for $S_{ii} = 1$. Using the property that $S_{11} \mid S_{22} \mid \cdots \mid S_{mm}$ it follows that T' is also unit upper triangular. ☐

6 MAXIMAL SMITH MASSAGER

In this section we develop a randomized algorithm for computing a Smith massager for a nonsingular $A \in \mathbb{Z}^{n\times n}$. Section 6.1 gives a subroutine for combining an index-$(0, m)$ and index-(m, r) Smith massager to obtain an index-$(0, m + r)$ Smith massager. The algorithm is given in Section 6.2 with a proof of correctness and running time given in Sections 6.3 and 6.4, respectively.

6.1 Combining index massagers

We show how an index-$(0, n)$ Smith massager for $\operatorname{diag}(A, I_n)$ can be computed in a block iterative fashion. Suppose we have an index-$(0, m)$ Smith massager (U, M, T, S) for $\operatorname{diag}(A, I_n)$. Then

$$B := \begin{bmatrix} A & & & \\ & I & & \\ & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & I_r & \\ U & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & M \\ & I & & \\ & & I_r & \\ & & & T \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & I_r & \\ & & & S^{-1} \end{bmatrix} \quad (21)$$

is integral. Let (U', M', T', S') be an index-(m, r) Smith massager for B. Then

$$C := B \begin{bmatrix} I_n & & & \\ & I & & \\ U' & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & M' & & \\ & I & & \\ & & T' & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & S'^{-1} & \\ & & & I_m \end{bmatrix} \quad (22)$$

is integral. A direct computation shows that the product of the first trio of matrices post-multiplying $\operatorname{diag}(A, I_n)$ in (21) with the second trio of matrices post-multiplying B in (22) is equal to

$$\begin{bmatrix} I_n & & & \\ & I & & \\ U' & & I_r & \\ U & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & M' & M \\ & I & & \\ & & T' & -U'M \\ & & & T \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & S'^{-1} & \\ & & & S^{-1} \end{bmatrix}. \quad (23)$$

Thus, the result of post-multiplying $\operatorname{diag}(A, I_n)$ by the combined trio in (23) is integral. The next result follows as a result of the above discussion and as a corollary of Lemma 14.

Theorem 15. *Let (U, M, T, S) be a reduced index-$(0, m)$ Smith massager for $\operatorname{diag}(A, I_n)$, and let (U', M', T', S') be a reduced index-(m, r) Smith massager for the matrix B in (21). If S'_{rr} is a divisor of S_{11}, then a reduced index-$(0, m + r)$ Smith massager for $\operatorname{diag}(A, I_n)$ is given by (U'', M'', T'', S'') where*

$$U'' = \left[\begin{array}{c} U' \\ \hline U \end{array}\right],\ M'' = \left[\begin{array}{c|c} M' & M \end{array}\right],\ S'' = \left[\begin{array}{c|c} S' & \\ \hline & S \end{array}\right],$$

and

$$T'' = \left[\begin{array}{c|c} T' & -U'M \operatorname{cmod} S \\ \hline & T \end{array}\right].$$

6.2 Maximal Smith massager algorithm

Algorithm SmithMassager(A) is shown in Figure 2. For convenience, assume for the moment that n is equal to one less than a power of two. Phase 2 of the algorithm initializes $B := \mathrm{diag}(A, I_n)$ and $(U, M, T, S) \in (\mathbb{Z}^{0\times n}, \mathbb{Z}^{n\times 0}, \mathbb{Z}^{0\times 0}, \mathbb{Z}^{0\times 0})$ to be the trivial index-$(0, 0)$ Smith massager, and then uses $\log_2(n + 1)$ applications of Theorem 15 to update (U, M, T, S) to be an index-$(0, n)$ Smith massager for $\mathrm{diag}(A, I_n)$. The technique of Lemma 14 is used to keep the intermediate index massagers reduced. At the beginning of iteration i of the for-loop, (U, M, T, S) is a reduced index-$(0, m)$ Smith massager where $m = 2^{i-1} - 1$. Iteration i then updates (U, M, T, S) to be a reduced index-$(0, m + r)$ Smith massager where $r = 2^{i-1}$.

At the end of phase 2, the algorithm has computed a Smith massager (U, M, T, S) for A. It remains only to assay if (U, M, T, S) is maximal. This is done by checking that the massaged matrix B is unimodular.

```
SmithMassager(A)
Input: Nonsingular $A \in \mathbb{Z}^{n\times n}$ with $\det A \perp 2$.
Output: A reduced maximal Smith massager for $A$ or FAIL.
Note: FAIL will be returned with probability less than $1/2$.
  (1) [Compute the largest invariant factor of $A$]
      $s :=$ the largest invariant factor $s_n$ of $A$
      # $s$ may be a proper divisor of $s_n$ with probability $\le 1/4$.
  (2) [Compute an index-$(0, n)$ Smith massager for $\mathrm{diag}(A, I_n)$]
      $(U, M, T, S) \in (\mathbb{Z}^{0\times n}, \mathbb{Z}^{n\times 0}, \mathbb{Z}^{0\times 0}, \mathbb{Z}^{0\times 0})$
      $B := \mathrm{diag}(A, I_n)$
      for $i = 1$ to $\lceil \log_2(n + 1)\rceil$ do
        $m := 2^{i-1} - 1$
        $r := \min(2^{i-1}, n - m)$
        if $i > 1$ then $s := S_{11}$
    (a) [Compute an index-$(m, r)$ massager of $B$ and reduce]
        $(U', M', I, S') :=$ IndexMassager$(B, m, r, s, 2^{-(i+2)})$
        $U', M' := U' \,\mathrm{rmod}\, S', M' \,\mathrm{cmod}\, S'$
        $T' := -U'M' \,\mathrm{cmod}\, S'$, 0 diagonal entries replaced by 1
    (b) [Augment massager and reduce]
        $U, M, S := \left[\begin{array}{c} U' \\ \hline U \end{array}\right], \left[\begin{array}{c|c} M' & M \end{array}\right], \left[\begin{array}{c|c} S' & \\ \hline & S \end{array}\right]$
        $T := \left[\begin{array}{c|c} T' & -U'M \,\mathrm{cmod}\, S \\ \hline & T \end{array}\right]$
    (c) [Apply massager]
        $B := \begin{bmatrix} A & & AMS^{-1} \\ & I & \\ U & & (T + UM)S^{-1} \end{bmatrix}$
  (3) [Certify that $(U, M, T, S)$ is maximal]
      if $|\det B| = 1$ then return $(U, M, T, S)$
      else return FAIL
```

Figure 2: Algorithm SmithMassager

6.3 Correctness

We begin with two lemmas regarding properties of the massaged matrix B in phase 2(c) of the algorithm. Lemma 16 is a corollary of Lemma 10.

Lemma 16. *If (U, M, T, S) is a maximal index-$(0, m)$ Smith massager for $\mathrm{diag}(A, I_n)$ with Smith form $\mathrm{diag}(I_n, S', S)$, then the Smith form of the massaged matrix $B \in \mathbb{Z}^{2n\times 2n}$ as in (21) is $\mathrm{diag}(I_n, I_m, S')$.*

Lemma 17. *If (U, M, T, S) is a maximal index-$(0, m)$ Smith massager for $\mathrm{diag}(A, I_n)$ and $B \in \mathbb{Z}^{2n\times 2n}$ the massaged matrix as in (21), then the last n rows and columns of B^{-1} are integral.*

Proof. Notice that the augmenting operation of Theorem 15 can also be reversed to separate a Smith massager. We will use induction on m. The base case, for $m = 0$, holds vacuously. Next, assume that the statement of the lemma holds for a maximal index-$(0, m)$ Smith massager (U, M, T, S). This means that: (1) the largest invariant factor of the massaged matrix B is s_{n-m} according to Lemma 16, and: (2) the last n rows of B^{-1} are integral according to the induction hypothesis. Now, let (U', M', T', S') be a maximal index-$(m, 1)$ Smith massager for B. The product of the trio of matrices defined by (U, M, T, S) and the product defined by (U', M', T', S') correspond to a trio of matrices defined by a maximal index-$(0, m + 1)$ Smith massager. The inverse of the massaged matrix C will be

$$\begin{bmatrix} I_n & & & \\ & I & & \\ & & s_{n-m} & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & -M' & \\ & I & & \\ & & 1 & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ -U' & & 1 & \\ & & & I_m \end{bmatrix} B^{-1}.$$

We see that the largest invariant factor of the product of the last three matrices is still s_{n-m}. In addition, the row of the product that is multiplied with s_{n-m}, is the only one from the last n rows of B^{-1} to which elements from the non-integral part of B^{-1} are added. Of course, multiplying with the matrix's largest invariant factor ensures that the last n rows of C^{-1} remain integral.

Finally, the last n columns are necessarily integral since they are the product of integral parts. □

Theorem 18. *Algorithm SmithMassager shown in Figure 2 is correct. The algorithm returns FAIL with probability less than $1/2$.*

Proof. The correctness of the algorithm is certified by the unimodularity check in phase 3. Regarding the probability of success, we define the following events.

- E_0: In phase 1, s is not the largest invariant factor s_n of A.
- E_i: At iteration $i = 1, \ldots, \lceil \log_2(n + 1)\rceil$ of phase 2, massager (U, M, T, S) is not maximal.

In other words, in order to prove the theorem, it is enough to show that $\Pr[E_{\lceil \log_2(n+1)\rceil}] < 1/2$. From the specification of phase 1, the routine IndexMassager, and Lemma 17, we obtain that

$$\Pr[E_0] \le \frac{1}{4} \quad \text{and} \quad \Pr[E_i | \neg E_{i-1}] \le 2^{-(i+2)}.$$

Furthermore,

$$\begin{aligned} \Pr[E_i] &= \Pr[E_i|\neg E_{i-1}] \Pr[\neg E_{i-1}] + \Pr[E_i|E_{i-1}] \Pr[E_{i-1}] \\ &\le \Pr[E_i|\neg E_{i-1}] \cdot 1 + 1 \cdot \Pr[E_{i-1}] \\ &\le 2^{-(i+2)} + \Pr[E_{i-1}]. \end{aligned}$$

So, Algorithm SmithMassager returns FAIL with probability less than

$$\Pr[E_{\lceil \log_2(n+1)\rceil}] \le \sum_{i=1}^{\lceil \log_2(n+1)\rceil} 2^{-(i+2)} + \Pr[E_0] < \sum_{i=1}^{\infty} \frac{1}{2^{i+2}} + \frac{1}{4} = \frac{1}{2}.$$

□

6.4 Complexity

We begin by bounding the cost of phases 2(b) and 2(c). Lemma 19 presents a subroutine that computes matrix B in phase 2(c), and Lemma 20 a subroutine that realizes the construction of Theorem 15 in phase 2(b).

Lemma 19. *There exists a procedure that takes a reduced index-$(0, m)$ Smith massager (U, M, T, S) for $\operatorname{diag}(A, I_n)$, and returns a matrix B as in (21). The running time of the procedure is $O(n^\omega \mathsf{M}(\log n + \log ||A||)$.*

Proof. It is enough to prove the claim for $m = n$. We have that

$$B = \begin{bmatrix} A & AMS^{-1} \\ U & (T + UM)S^{-1} \end{bmatrix}.$$

The cost-dominating operation is the product $UM \in \mathbb{Z}^{n\times n}$.

Recall that $(\det S) \mid (\det A)$, and that the entries in row i of matrix U and in column i of matrix M are reduced modulo S_{ii}. Thus, for an $X := 2^h \geq \lceil n^{1/2}||A||\rceil$, the matrices U and M can be written as their X-adic expansions (of length n),

$$U = U_0 + \cdots + U_{n-1}X^{n-1} \text{ and } M = M_0 + \cdots + M_{n-1}X^{n-1},$$

with U_i and M_i reduced modulo X. This gives the following expression for their product: $UM = \sum_{i,j=0}^{n-1} U_i M_j X^{i+j}$. Of course, it is impossible to perform n^2 matrix multiplications within our target complexity. Instead, we observe that since the rows or columns of both matrices are reduced modulo the invariant factors of A, then the higher-order coefficients of the expansion must be sparser.

According to [1, Lemma 17], if we remove the top zero rows of each $U_i \in \mathbb{Z}/(X)^{n\times n}$ to obtain a $\bar{U}_i \in \mathbb{Z}/(X)^{*\times n}$, then the matrix

$$\bar{U} := \left[\begin{array}{c} \bar{U}_0 \\ \hline \vdots \\ \hline \bar{U}_{n-1} \end{array}\right]$$

has at most $2n$ rows. The same holds for the number of columns of the matrix

$$\bar{M} := \left[\begin{array}{c|c|c} \bar{M}_0 & \cdots & \bar{M}_{n-1} \end{array}\right],$$

where each $\bar{M}_i \in \mathbb{Z}/(X)^{n\times *}$ is produced by removing the leading zero columns from each component of the X-adic expansion of M. We can multiply $\bar{U}$ and $\bar{M}$ in $O(n^\omega \mathsf{M}(\log n + \log ||A||)$ and obtain the (at most) $2n \times 2n$ matrix

$$\begin{bmatrix} \bar{U}_0\bar{M}_0 & \cdots & \bar{U}_0\bar{M}_{n-1} \\ \vdots & \ddots & \vdots \\ \bar{U}_{n-1}\bar{M}_0 & \cdots & \bar{U}_{n-1}\bar{M}_{n-1} \end{bmatrix}. \tag{24}$$

The above matrix contains the result of all the $U_i M_j$ products, as they are equal to $\bar{U}_i\bar{M}_j$ along with some additional zero rows and columns. Finally, after multiplying each $\bar{U}_i\bar{M}_j$ with X^{i+j}, we compute UM by adding all the products together, while taking into account their additional zero rows and columns. □

Lemma 20. *The reduced index-$(0, m + r)$ massager of Theorem 15 can be computed in time $O(n^\omega \mathsf{M}(\log n + \log ||A||))$.*

Proof. The only nontrivial computation of Theorem 15 is the product $U'M \in \mathbb{Z}^{r\times m}$, and the required complexity can be achieved by following the same technique as in Lemma 19. □

Theorem 21. *The running time of the Algorithm* `SmithMassager` *shown in Figure 2 is $O(n^\omega \mathsf{B}(\log n + \log ||A||) (\log n)^2)$.*

Proof. Phase 1 is done in time $O(n^\omega \mathsf{B}(\log n + \log ||A||) \log n)$ using the Monte Carlo approach of [4, Theorem 2.1] combined with fast linear system solving [1, Corollary 7] and rational number reconstruction. Phase 2 consists of $O(\log n)$ iterations of the `IndexMassager` algorithm. Since matrix U is always reduced row modulo S, the average of the lengths of the rows of U, and consequently of B, is $O(\log n + \log ||A||)$. Hence, phase 2 requires time $O(n^\omega \mathsf{B}(\log n + \log ||A||)(\log n)^2)$. Finally, according to [11, Section 4], the unimodularity check in phase 3 can be performed in time $O(n^\omega \mathsf{M}(\log n + \log ||A||) \log n)$. □

7 AN ALGORITHM FOR SMITH FORM

Given a nonsingular input matrix $A \in \mathbb{Z}^{n\times n}$, we first compute [1] the 2-Smith form $S_{\text{even}} = \operatorname{diag}(2^{e_1}, \ldots, 2^{e_n})$ of A, where 2^{e_i} is the largest power of 2 that divides the i-th invariant factor of A, together with A_{odd} such that $\det A_{\text{odd}} \perp 2$, $||A_{\text{odd}}|| \leq n||A||$, and $A_{\text{odd}}S_{\text{even}} \equiv_R A$. Then, we compute the Smith form S_{odd} of A_{odd} with Algorithm `SmithMassager` and return $S_{\text{odd}}S_{\text{even}}$.

We remark that the algorithms in [1] were analysed under a more restrictive cost model than the one used in this paper. Replacing the subroutine in [1, Section 6] with the integer analogue of the algorithm supporting [9, Theorem 7] allows A_{odd} and S_{even} to be computed in $O(n^\omega \mathsf{M}(\log n + \log ||A||) (\log n)^2)$ bit operations.

Theorem 22. *There exists a Las Vegas probabilistic algorithm that computes the Smith form of a nonsingular $A \in \mathbb{Z}^{n\times n}$ using $O(n^\omega \mathsf{B}(\log n + \log ||A||) (\log n)^2)$ bit operations.*

REFERENCES

[1] S. Birmpilis, G. Labahn, and A. Storjohann. Deterministic reduction of integer nonsingular linear system solving to matrix multiplication. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'19*. ACM Press, New York, 2019.

[2] J. Blömer, R. Karp, and E. Welzl. The rank of sparse random matrices over finite fields. *Random Structures and Algorithms*, 10(4):407–419, July 1997.

[3] C. Cooper. On the distribution of rank of a random matrix over a finite field. *Random Structures and Algorithms*, 17(3-4):197–212, oct 2000.

[4] W. Eberly, M. Giesbrecht, and G. Villard. Computing the determinant and Smith form of an integer matrix. In *Proc. 31st Ann. IEEE Symp. Foundations of Computer Science*, pages 675–685, 2000.

[5] F. L. Gall. Powers of tensors and fast matrix multiplication. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'14*. ACM Press, New York, 2014.

[6] F. L. Gall and F. Urrutia. Improved rectangular matrix multiplication using powers of the Coppersmith-Winograd tensor. In A. Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1029–1046. SIAM, 2018.

[7] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, 3rd edition, 2013.

[8] M. Giesbrecht. Fast computation of the Smith form of a sparse integer matrix. *Computational Complexity*, 10(1):41–69, 11 2001.

[9] S. Gupta, S. Sarkar, A. Storjohann, and J. Valeriote. Triangular x-basis decompositions and derandomization of linear algebra algorithms over $\mathsf{K}[x]$. *Journal of Symbolic Computation*, 47(4), 2012. Festschrift for the 60th Birthday of Joachim von zur Gathen.

[10] E. Kaltofen and G. Villard. On the complexity of computing determinants. *Computational Complexity*, 13(3-4):91–130, 2004.

[11] C. Pauderis and A. Storjohann. Deterministic unimodularity certification. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'12*, page 281–288. ACM Press, New York, 2012.

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

[13] A. Storjohann. The shifted number system for fast linear algebra on integer matrices. *Journal of Complexity*, 21(4):609–650, 2005. Festschrift for the 70th Birthday of Arnold Schönhage.