Asymptotically Fast Computation of Hermite Normal Forms of Integer Matrices

Department of Computer Science
University of Waterloo, Ontario, Canada, N2L 3G1
Technical Report CS-96-04

Arne Storjohann and George Labahn
{astorjoh,glabahn}@daisy.uwaterloo.ca

Abstract

This paper presents a new algorithm for computing the Hermite normal form H of an $A \in \mathbf{Z}^{n\times m}$ of rank m together with a unimodular pre-multiplier matrix U such that $UA = H$. Our algorithm requires $O\tilde{}(m^{\theta-1}n\mathsf{M}(m\log||A||))$ bit operations to produce both H and a candidate for U. Here, $||A|| = \max_{ij}|A_{ij}|$, $M(t)$ bit operations are sufficient to multiply two $\lceil t\rceil$-bit integers, and θ is the exponent for matrix multiplication over rings: two $m\times m$ matrices over a ring R can be multiplied in $O(m^\theta)$ ring operations from R. The previously fastest algorithm of Hafner & McCurley requires $O\tilde{}(m^2n\mathsf{M}(m\log||A||))$ bit operations to produce H, but does not produce a candidate for U. Previous methods require on the order of $O\tilde{}(n^3M(m\log||A||))$ bit operations to produce a candidate for U — our algorithm improves on this significantly in both a theoretical and practical sense.

1 Introduction

A fundamental notion for matrices over rings is *left equivalence*. Two $n\times m$ matrices A and B over a principal ideal ring R are said to be left equivalent if there exists an $n\times n$ unimodular matrix U that satisfies $UA = B$. (A unimodular matrix has determinant a unit in R and hence is invertible.) Any integer matrix A with full column rank can be transformed to an upper triangular matrix T using only elementary row operations. The triangularization T can be made unique by enforcing that diagonal entries be positive and off-diagonal entries be non-negative and reduced in magnitude modulo the diagonal entry in each column. In particular, any $A \in \mathbf{Z}^{n\times m}$ with rank m is left equivalent to a unique upper triangular matrix H. That is, there exists a unimodular matrix U (i.e. $\det(U) \pm 1$) such that

$$UA = H = \begin{bmatrix} h_1 & \bar{h}_{12} & \bar{h}_{13} & & \bar{h}_{1m} \\ & h_2 & \bar{h}_{23} & \cdots & \bar{h}_{2m} \\ & & h_3 & & \bar{h}_{3m} \\ & & & \ddots & \vdots \\ & & & & h_m \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

where h_j is positive for $1 \le j \le m$ and $\bar{h}_{ij}$ satisifes $0 \le \bar{h_{ij}} < h_j$ for $1 \le i,j \le m$. The reduced triangularization H is called the *Hermite normal form* of A and the unimodular U is called a pre-multiplier matrix. The Hermite normal form was first proven to exist by Hermite [5, 1851] for the case of a square nonsingular input matrix. The Hermite normal form is in fact a canonical form for left equivalence over $\mathbf{Z}$ — it always exists and is unique (see, for example, Newman [10, 1972]).

Hafner & McCurley [4, 1991] have given an algorithm that requires $O\tilde{}(m^2n\mathsf{M}(m\log||A||))$ [1] bit operations to compute the Hermite normal form of A. They have also shown how to apply fast matrix multiplication techniques to the problem of triangularizing matrices over principal ideal rings. They show how to apply their result to the case $\mathsf{R} = \mathbf{Z}$ to get an asymptotically fast algorithm for obtaining a unimodular triangularization of an integral input matrix. This results in an algorithm that requires $O\tilde{}(m^{\theta-1}n\mathsf{M}(m\log||A||))$ bit operations to produces an upper triangular T left equivalent to A. Here θ denotes the number such that two $m\times m$ matrices over a ring R can be multiplied in $O(m^\theta)$ ring operations from R. Using standard multiplication $\theta = 3$, while the best known algorithm of Coppersmith & Winograd [2, 1990] allows $\theta = 2.38$. However, Hafner & McCurley were unable to obtain an algorithm to compute the complete Hermite normal form with the improved complexity.

For some applications the complete Hermite normal form of an input matrix is not required; a general triangularization such as produced by Hafner & McCurleys algorithm may be sufficient. The Hermite normal form, however, has some important advantages over a general triangularization. First, the Hermite normal form is a canonical form for left equivalence. To determine whether a second matrix B is left equivalent to A, it is sufficient to compare the Hermite normal forms of A and B. This check for left equivalence is not possible using a general (non unique) triangularization. Secondly, the space required to write down the Hermite normal form H of A will be small compared to that of general triangularization T. Consider the case of a square nonsingular input matrix $A \in \mathbf{Z}^{n\times n}$. The total size of H (the sum of the bit lengths of the individual entries of H) will be on the order of $O\tilde{}(n^2\log||A||)$ bits, or about the same space as required to write down the input matrix. The triangularization T returned by Hafner & McCurleys algorithm will have

[1] To summarize results we use "soft-Oh" notation: for any $f, g : \Re^s \mapsto \Re$, $f = O\tilde{}(g)$ if and only if $f = O(g\cdot\log^c g)$ for some constant $c > 0$.

entries bounded in length by $O^{\sim}(n \log ||A||)$ bits. This leads to a total size bound for T of $O^{\sim}(n^3 \log ||A||)$ bits, or about a factor n larger than H. Since A is nonsingular, there will be a unique unimodular matrix U which satisfies $UA = H$ and $A = HU^{-1}$. Similarly, there exists a unique unimodular P with $PA = T$ and $A = TP^{-1}$. Entries in U and U^{-1} will be bounded in length by $O^{\sim}(n \log ||A||)$ bits and by $O^{\sim}(\log ||A|| + \log n)$ bits respectively. Entries of P and P^{-1}, on the other hand, are bounded in length by $O^{\sim}(n^2 \log ||A||)$ bits and $O^{\sim}(n^3 \log ||A||)$ bits respectively.

In this paper we show how to use a fast matrix multiplication decomposition for reducing off-diagonal entries of an upper triangular matrix T. Combining our result with Hafner & McCurleys triangularization method gives an algorithm for computing the Hermite normal form H of A in $O^{\sim}(m^{\theta-1} n \mathsf{M}(m \log ||A||))$ bit operations.

In addition, we show how to recover a candidate for a unimodular pre-multiplier matrix U which satisfies $UA = H$. Computing a pre-multiplier is required in such applications as integer programming [6, 1969], solving linear diophantine equations [8, 1989] and computing matrix greatest common divisors [12, 1995]. In the case where A is square and nonsingular, U is unique and can be recovered with no increase in asymptotic complexity by computing $U \leftarrow HA^{-1}$ using standard methods. However, for the rectangular case where A is $n \times m$ with $n > m$, the pre-multiplier matrix U is not unique. Our algorithm recovers both H and a candidate for an $n \times n$ unimodular U in $O^{\sim}(m^{\theta-1} n \mathsf{M}(m \log ||A||))$ bit operations. The previously fastest algorithm for Hermite normal form, which works modulo the determinant of the input matrix to prevent expression swell, has initially been presented for the case of square nonsingular input matrices (see, for example, Domich, Kannan & Trotter [3, 1987] or Iliopolous [9, 1989]). Hafner & McCurley [4, 1991] extend the mod determinant approach and give an algorithm that requires $O^{\sim}(m^2 n \mathsf{M}(m \log ||A||))$ bit operations to produce H, but they don't show how to produce a candidate for U within this time. They suggest the following scheme for producing a U. Permute the rows of A and augment with the $n-m$ identity matrix to get a new $n \times n$ matrix $\bar{A}$ which can be written in block form as

$$\bar{A} = \left[\begin{array}{c|c} A_1 & \\ \hline A_2 & I_{n-m} \end{array}\right]$$

where A_1 is nonsingular. Compute the Hermite normal form $\bar{H}$ of $\bar{A}$ at a cost of $O^{\sim}(n^3 \mathsf{M}(m \log ||A||))$ bit operations, and set $U \leftarrow \bar{H}\bar{A}^{-1}$. The algorithm we give in this paper improves this worst case complexity bound by a factor of about $O(n^2/m^{\theta-1})$ — a significant improvement for rectangular matrices even assuming standard matrix multiplication. Moreover, the matrix U will produced by our algorithm will be "nice". By this we mean that the entries of U will be bounded in length by $O^{\sim}(m \log ||A||)$ bits and U will be sparse, with on the order of only $O^{\sim}(nm)$ nonzero integer entries.

2 Preliminaries and Previous Results

following Hafner & McCurley in [4, 1991], we will express our complexity results in terms of a function $\mathsf{B}(t)$ that bounds the number of bit operations to solve both the extended Euclidean problem with two $\lceil t \rceil$ bit integers and to apply the Chinese remainder algorithm with moduli consisting of all primes less than t. By Theorem 8.20 and 8.21 of Aho, Hopcroft & Ullman [1, 1974] we can take $\mathsf{B}(t) \ll \mathsf{M}(t) \log(t)$ where $\mathsf{M}(t)$ is a monotonic upper bound on the number of bit operations required to multiply two $\lceil t \rceil$ bit integers. The Schönhage & Strassen [11, 1971] integer multiplication algorithm allows $\mathsf{M}(t) \ll t \log t \log\log t$ hence

$$\mathsf{B}(t) \ll t \log^2 t \log\log t.$$

In what follows, we write $\mathrm{MM}(n) = \mathrm{MM}_{\mathsf{R}}(n)$ to mean the number of ring operations required to multiply two $n \times n$ matrices over a ring R, where

$$\mathrm{MM}(n) \ll n^{\theta}. \tag{1}$$

It is well known that the product of the diagonal entries in the Hermite normal form of an $n \times m$ matrix A with rank m is given by the gcd of all $m \times m$ minors of A — in what follows we refer to this quantity as $\det(\mathcal{L}(A))$. We end this section with a result of Hafner & McCurley [4, 1991] that will be required in Section 3 and 4.

Lemma 1 (Hafner & McCurley ([4, 1991])) *There exists a deterministic algorithm that takes as input an $n \times m$ rank m integral matrix A and positive integer d' that is a multiple $\det(\mathcal{L}(A))$, and produces as output an upper triangular matrix T that is left equivalent to A. Entries in T will be bounded in magnitude by d and the running time of the algorithm is $O(mn\mathsf{B}(\log ||A||) + m^{\theta-1} n \log(2n/m)\mathsf{B}(\log d))$ bit operations.*

Hafner & McCurley also show also how to obtain a suitable multiple d of $\det(\mathcal{L}(A))$ which satisfies $d \leq m^{m/2}||A||^m$.

Theorem 2 (Hafner & McCurley [4, 1991]) *There exists a deterministic algorithm that takes as input an $n \times m$ rank m integral matrix A, and produces as output an upper triangular matrix T that is left equivalent to A. Entries in T will be bounded in magnitude by $m^{m/2}||A||^m$, and the running time of the algorithm is $O(m^{\theta-1} n \log(2n/m)\mathsf{B}(m \log m||A||))$ bit operations.*

3 Asymptotically Fast Hermite Normal Form

In this section we give our asymptotically fast algorithm for computing the Hermite normal form of an $A \in \mathbf{Z}^{n \times m}$. First we apply the triangularization algorithm of Theorem 2 to transform A to an upper triangular $T \in \mathbf{Z}^{m \times m}$ having diagonal entries the same as those in the Hermite normal form of A. Our approach is to consider T as a matrix over $\mathbf{Z}_d$, where d is a positive integer multiple of $\det(T)$. In particular, let $\bar{T}$ be the matrix obtained from T by reducing each entry modulo d. In Subsection 3.1 we present an algorithm which computes a unit upper triangular $\bar{U} \in \mathbf{Z}_d^{n \times n}$ which satisfies $\bar{U}\bar{T} = H \bmod d$, where H is in Hermite normal form over $\mathbf{Z}_d$, that is, has offdiagonal entries in each column reduced modulo the diagonal entry in each column. The Hermite normal form of T is simply H, considered now over $\mathbf{Z}$ rather than $\mathbf{Z}_d$. In Subsection 3.2 we make this argument precise and give the complete Hermite normal form algorithm.

3.1 Hermite Normal Form of a Triangular Matrix

In this subsection we work completely over the ring $\mathbf{Z}_d$. In particular, all matrices will be over $\mathbf{Z}_d$, and equations should be taken to hold modulo d. For brevity, we give our complexity results in terms of the number of operations from $\mathbf{Z}_d$ — a single operation from $\mathbf{Z}_d$ has cost $O(\mathsf{B}(\log d))$ bit operations.

Let $R(n) = R_d(n)$ be a bound on the number of operations from $\mathbf{Z}_d$ required to compute, for an upper triangular $T \in \mathbf{Z}_d^{n \times n}$ with nonzero diagonal entries, a unit upper triangular pre-multiplier U such that $UT = H$, with H in Hermite normal form. Our result is the following.

Theorem 3 $R(n) \ll n^\theta$.

We prove Theorem 3 by reducing to a special case for which we require some notation. For n even, let $\mathcal{T}_n$ be the set of $n \times n$ nonsingular upper triangular matrices over $\mathbf{Z}_d$ that can be written in block form as

$$\left[\begin{array}{c|c} I & A \\ \hline & B \end{array}\right]$$

where A and B are $(n/2) \times (n/2)$, B is in Hermite normal form, and the empty block is used to denote the zero matrix. Let $R^*(n) = R_d^*(n)$ be the number of ring operations required to compute, for a $T \in \mathcal{T}_n$, candidates for $(n/2) \times (n/2)$ matrices Q and R over $\mathbf{Z}_d$ such that

$$\left[\begin{array}{c|c} I & -Q \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} I & A \\ \hline & B \end{array}\right] = \left[\begin{array}{c|c} I & R \\ \hline & B \end{array}\right]$$

with the matrix on the right hand side in Hermite normal form.

Lemma 4 $R^*(n) \ll n^\theta$.

Proof By embedding the input matrix into the block matrix $\mathrm{diag}(I_p, T, I_p) \in \mathcal{T}_{n+2p}$, with p a nonnegative integer less than $n/2$, we can assume without loss of generality that n is power of 2. We show how to reduce the problem to four subproblems of half the size which can be combined using matrix multiplication. We claim that

$$\begin{aligned} R^*(n) &\leq 4R^*(n/2) + \mathrm{MM}(n) \\ &\leq 4R^*(n/2) + cn^\theta \qquad (2) \end{aligned}$$

for some absolute constant c. The second line in this inequality follows from (1). To prove (2), we start with a $T \in \mathcal{T}_n$, which, using a block decomposition, we can write as

$$T = \left[\begin{array}{cc|cc} I & & A_1 & A_3 \\ & I & A_2 & A_4 \\ \hline & & B_1 & B_2 \\ & & & B_3 \end{array}\right]$$

where all blocks are of size $(n/4) \times (n/4)$. At a cost of $2R^*(n/2)$, compute $(n/4) \times (n/4)$ matrices Q_1, R_1, Q_2 and R_2 such that

$$\left[\begin{array}{c|c} I & -Q_1 \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} I & A_1 \\ \hline & B_1 \end{array}\right] = \left[\begin{array}{c|c} I & R_1 \\ \hline & B_1 \end{array}\right]$$

and

$$\left[\begin{array}{c|c} I & -Q_2 \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} I & A_2 \\ \hline & B_1 \end{array}\right] = \left[\begin{array}{c|c} I & R_2 \\ \hline & B_1 \end{array}\right]$$

with the matrices on the right hand side in Hermite normal form. At a cost of $\mathrm{MM}(n)$, compute the matrix product

$$\left[\begin{array}{cc|cc} I & & -Q_1 & \\ & I & -Q_2 & \\ \hline & & I & \\ & & & I \end{array}\right] \left[\begin{array}{cc|cc} I & & A_1 & A_3 \\ & I & A_2 & A_4 \\ \hline & & B_1 & B_2 \\ & & & B_3 \end{array}\right]$$

$$= \left[\begin{array}{cc|cc} I & & R_1 & A_3' \\ & I & R_2 & A_4' \\ \hline & & B_1 & B_2 \\ & & & B_3 \end{array}\right]$$

to get the transformed $(n/4) \times (n/4)$ blocks A_3' and A_4'. The last stage is now similar to the first. At a cost of $2R^*(n/2)$, compute $(n/4) \times (n/4)$ matrices Q_3, R_3, Q_4 and R_4 such that

$$\left[\begin{array}{c|c} I & -Q_3 \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} I & A_3' \\ \hline & B_1 \end{array}\right] = \left[\begin{array}{c|c} I & R_3 \\ \hline & B_1 \end{array}\right]$$

and

$$\left[\begin{array}{c|c} I & -Q_4 \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} I & A_4' \\ \hline & B_1 \end{array}\right] = \left[\begin{array}{c|c} I & R_4 \\ \hline & B_1 \end{array}\right]$$

with the matrices on the right hand side in Hermite normal form. We now have

$$\left[\begin{array}{cc|cc} I & & -Q_1 & -Q_3 \\ & I & -Q_2 & -Q_4 \\ \hline & & I & \\ & & & I \end{array}\right] \left[\begin{array}{cc|cc} I & & A_1 & A_3 \\ & I & A_2 & A_4 \\ \hline & & B_1 & B_2 \\ & & & B_3 \end{array}\right]$$

$$= \left[\begin{array}{cc|cc} I & & R_1 & R_3 \\ & I & R_2 & R_4 \\ \hline & & B_1 & B_2 \\ & & & B_3 \end{array}\right]$$

with the premultiplier matrix on the left unit upper triangular and the matrix on the right hand side in Hermite normal form. This shows that

$$R^*(n) \leq 4R^*(n/2) + \mathrm{MM}(n)$$

which verifies (2). Iterate (2) to obtain

$$\begin{aligned} R^*(n) &\leq 4R^*(n/2) + cn^\theta \\ &= 16R^*(n/4) + cn^\theta + 4c(n/2)^\theta \\ &\vdots \\ &= 4^{(\log_2 n)-1} R^*(2) + c \sum_{i=0}^{(\log_2 n)-2} (\frac{4}{2^\theta})^i \\ &\ll n^2 R^*(2) + n^\theta. \end{aligned}$$

The Lemma now follows since the cost of $R^*(2)$ is $O(1)$ operations from $\mathbf{Z}_d$. ■

We now return to the proof of Theorem 3. By embedding an $n \times n$ upper triangular nonsingular input matrix T into the block diagonal matrix $\mathrm{diag}(I_p, T)$, with p a nonnegative integer bounded by n, we can assume without loss of generality that n is a power of two. We claim that

$$\begin{aligned} R(n) &\leq 2R(n/2) + \mathrm{MM}(n) + R^*(n) \\ &\leq 2R(n/2) + cn^\theta \end{aligned} \tag{3}$$

for some absolute constant c. The second line of the inequality follows from (1) and Lemma 4. To prove (3), we start with an $n \times n$ nonsingular upper triangular matrix T, which, using a block decomposition, we can write as

$$T = \left[\begin{array}{c|c} B_1 & A_2 \\ \hline & B_2 \end{array}\right].$$

At a cost of $2R(n/2)$, compute $(n/2) \times (n/2)$ matrices U_1, H_1, U_2 and H_2 such that $U_1 B_1 = H_1$ and $U_2 B_2 = H_2$ with H_1 and H_2 in Hermite normal form and U_1 and U_2 unit upper triangular. At a cost of $\mathrm{MM}(n)$, compute the matrix product

$$\left[\begin{array}{c|c} H_1 & A_2' \\ \hline & H_2 \end{array}\right] = \left[\begin{array}{c|c} U_1 & \\ \hline & U_2 \end{array}\right] \left[\begin{array}{c|c} B_1 & A_2 \\ \hline & B_2 \end{array}\right].$$

At a cost of $R^*(n)$, compute $(n/2) \times (n/2)$ matrices Q and R such that

$$\left[\begin{array}{c|c} I & Q \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} I & A_2' \\ \hline & H_2 \end{array}\right] = \left[\begin{array}{c|c} I & R_2 \\ \hline & H_2 \end{array}\right].$$

We now have

$$\left[\begin{array}{c|c} U_1 & -Q \\ \hline & U_2 \end{array}\right] \left[\begin{array}{c|c} B_1 & A_2 \\ \hline & B_2 \end{array}\right] = \left[\begin{array}{c|c} H_1 & R_2 \\ \hline & H_2 \end{array}\right]$$

with the premultiplier matrix on the left unit upper triangular and the matrix on the right hand side in Hermite normal form. This shows that

$$R(n) \leq 2R(n/2) + \mathrm{MM}(n) + R^*(n)$$

which verifies (3).

Iterate (3) to obtain

$$\begin{aligned} R(n) &\leq 2R(n/2) + cn^\theta \\ &= 4R(n/4) + cn^\theta(1 + 2(1/2)^\theta) \\ &\vdots \\ &= 2^{(\log_2 n)-1} R(2) + cn^\theta \sum_{i=0}^{(\log_2 n)-1} (2/2^\theta)^i \\ &\ll n^2 R(2) + n^\theta. \end{aligned}$$

The result now follows since the cost of $R(2)$ is $O(1)$ operations from $\mathbf{Z}_d$. ■

3.2 The Hermite Normal Form Algorithm

Theorem 5 *There exists a deterministic algorithm that takes as input an $n \times m$ rank m integral matrix A, and produces as output the Hermite normal form of A. The running time of the algorithm is bounded by $O(m^{\theta-1} n \log(2n/m) \mathsf{B}(m \log m ||A||))$ bit operations.*

Proof Use the algorithm of Theorem 2 to produce an upper triangular T left equivalent to A. As an intermediate step, the algorithm of Theorem 2 computes a positive integer multiple d' of $\det(\mathcal{L}(A))$ with $d' \leq m^{m/2} ||A||^m$. Let $\bar{T}$ be the matrix obtained from A by reducing entries modulo d, where $d = 2d'$, and compute the Hermite normal form H of $\bar{T}$ over $\mathbf{Z}_d$ using the algorithm of Theorem 3. So far, the running time is seen to be bounded by $O(m^{\theta-1} n \mathsf{B}(m \log m ||A||))$ bit operations. The following lemma shows that the Hermite normal form of T, and hence A, is also H, considered now over $\mathbf{Z}$ rather than $\mathbf{Z}_d$. ■

Lemma 6 *Let $\bar{U}, T$ and H be $m \times m$ integral matrices with $\bar{U}$ unit upper triangular, T upper triangular with positive diagonal entries and H in Hermite normal form. If*

$$\bar{U}T = H \bmod d, \tag{4}$$

where d is a positive integer multiple of $2\det(T)$, then H is the Hermite normal form of T.

Proof Since the Hermite normal form is a canonical form for left equivalence, we will be finished if we show their exists a $U \in \mathbf{Z}^{m \times m}$ which is unimodular and satisfies $UT = H$. It follows from (4) that the matrix $\bar{U}T - H$ has all entries divisible by d. Set $U \leftarrow \bar{U} - (1/d)(\bar{U}T - H)(2T^{\mathrm{adj}})$. Then U is integral and

$$\begin{aligned} UT &= (\bar{U} - (1/d)(\bar{U}T - H)(2T^{\mathrm{adj}}))T \\ &= \bar{U}T - (\bar{U}T - H)(2/d)T^{\mathrm{adj}}T \\ &= \bar{U}T - (\bar{U}T - H) \\ &= H. \end{aligned}$$

Since $\bar{U}$ is unit upper triangular and $d = 2\det(T)$ is strictly larger than each diagonal entry of T, the matrix $H = \bar{U}T \bmod d$ will have the same diagonal entries as T. In particular, this means that $\det(H) = \det(T)$ and it follows from the identity $UT = H$ that U is unimodular. ■

4 Asymptotically Fast Pre-multiplier

In this section we consider the problem of computing, for an $n \times m$ rank m integral input matrix A, an $n \times n$ unimodular pre-multiplier matrix U that satisfies $UA = H$. Our approach is based on an algorithm given by Hafner & McCurley [4, 1991] for triangularizing matrices over rings. Combining one of their results, essentially the algorithm of Theorem 2, with the result of Section 3 leads directly to an algorithm that computes an $n \times n$ matrix $\hat{U}$ that satisfies

$$\hat{U}A = H \bmod d \tag{5}$$

for some positive integer multiple d of $\det(\mathcal{L}(A))$. The cost of producing $\hat{U}$ will be bounded by $\tilde{O}(m^{\theta-1} n \mathsf{B}(m \log m ||A||))$ bit operations. Note that this complexity result is almost linear in n even though the output includes the $n \times n$ matrix $\hat{U}$. It turns out the the matrix $\hat{U}$ produced is sparse,

with only $O(nm\log(2n/m))$ nonzero entries. Unfortunately, equation (5) will not hold over $\mathbf{Z}$, nor will $\det(\hat{U}) = \pm 1$.

The result of this section is a modification of Hafner & McCurley's triangularization algorithm that produces a candidate for an $n \times n$ integral matrix U that satisfies both $UA = H$ and $\det(U) = \pm 1$. The algorithm also requires only $O\tilde{}(m^{\theta-1} n\mathsf{B}(m \log m||A||))$ bit operations. Moreover, U will have entries bounded in length by $O(\log(2n/m)m \log m||A||)$ bits. The major portion of our work will go into ensuring that the size of intermediate integers occuring during the construction of U do not become too large.

Our algorithm has two steps which we present separately in Subsection 4.1 and 4.2. In Subsecton 4.1 we present a fast preconditioning step which ensures that certain $m \times m$ minors of the input matrix are nonsingular — we require this to properly bound the size of intermediate integers during the remainder of the algorithm. In Subsection 4.2 we present our algorithm for a special type of rectangular input matrix that can be decomposed using a block decomposition into $m \times m$ Hermite normal form matrices. Finally, we combine these results in Subsection 4.3 and present the general algorithm.

Before continuing, we present some simple results which will be required later on. The following result establishes some useful bounds on the magnitudes of entries occuring in matrices which arise during intermediate computations.

Lemma 7 *Let U, A and H be $n \times n$ nonsingular integral matrices which satisfy $UA = H$ where H is in Hermite normal form. If $d = |\det(A)|$, then*

$$\|H^{\mathsf{adj}}\| \leq d.$$

If B is any other $n \times n$ integral matrix, then

$$\|HB\| \leq d\|B\|.$$

Lemma 8 *There exists a deterministic algorithm that takes as input a $2m \times m$ matrix*

$$A = \left[\begin{array}{c} H_1 \\ \hline H_2 \end{array}\right],$$

where both H_1 and H_2 are $m \times m$ of rank m and in Hermite normal form, and produces as output the Hermite normal form H of A together with a unimodular matrix U such that $UA = H$. If d bounds both $\det(H_1)$ and $\det(H_2)$ then $||U|| \leq d^2$. The cost of the algorithm is $O(m^{\theta}\mathsf{B}(\log d))$ bit operations.

Proof Compute the Hermite normal form $\bar{H}$ of

$$\bar{A} = \left[\begin{array}{c|c} H_1 & \\ \hline H_2 & I_m \end{array}\right].$$

By Lemma 1 the cost of this is $O(m^{\theta}\mathsf{B}(\log d))$ bit operations. We will compute U as $U \leftarrow \bar{H}\bar{A}^{-1}$. First we establish the bound on $||U||$. Let $d_1 = \det(H_1)$ and $d_2 = \det(H_2)$, then

$$\begin{aligned} \|U\| &= \left\|\bar{H}\bar{A}^{-1}\right\| \\ &\leq d_1 \left\|\bar{A}^{\mathsf{adj}}(1/d_1)\right\| \qquad (6) \\ &\leq \left\|\bar{A}^{\mathsf{adj}}\right\| \\ &= \left\|\left[\begin{array}{c|c} H_1^{\mathsf{adj}} & \\ \hline -H_2 H_1^{\mathsf{adj}} & d_1 I \end{array}\right]\right\| \\ &\leq \max(\|H_1^{\mathsf{adj}}\|, \|H_2 H_1^{\mathsf{adj}}\|, d_1) \\ &\leq \max(d_1, d_2 d_1, d_1) \qquad (7) \\ &\leq d^2. \end{aligned}$$

where lines (6) and (7) follow from Lemma 7. The matrix U can now be computed in $O(m^{\theta}\mathsf{B}(m \log m||A||))$ bit operations using a standard homomorphic imaging scheme. ■

4.1 A Preconditioning Algorithm

Let A be an $n \times m$ rank m integral input matrix. We can write A using a block decomposition as

$$\left[\begin{array}{c} A_1 \\ A_2 \\ \vdots \\ A_l \\ B \end{array}\right] \qquad (8)$$

where each A_i is $m \times m$, $l = \lfloor n/m \rfloor$, and B is $t \times m$ with $t = n - lm$. The purpose of this section is to give a fast algorithm for transforming A to an equivalent matrix, but where A_i is nonsingular for $1 \leq i \leq l$. Our result is the following.

Lemma 9 *There exists a deterministic algorithm that takes as input an $n \times m$ rank m integral matrix A, and returns as output an $n \times n$ permutation matrix P together with a unimodular matrix*

$$R = \left[\begin{array}{ccccccc} I_m & & & & & & \\ R_2 & I_m & & & & & \\ R_3 & & I_m & & & & \\ \vdots & & & \ddots & & & \\ R_l & & & & I_m & & \\ O & & & & & I_t & \end{array}\right]$$

where $l = \lfloor n/m \rfloor$, $t = n - lm$, each R_i is $m \times m$, and the matrix

$$RPA = \left[\begin{array}{c} \bar{A}_1 \\ \bar{A}_2 \\ \vdots \\ \bar{A}_l \\ B \end{array}\right]$$

has each $m \times m$ block $\bar{A}_i$ nonsingular for $1 \leq i \leq l$. The matrix R will satisfy $\|R\| = O(m \log m\|A\|)$, and the running time of the algorithm is $O(m^{\theta-1} n\mathsf{B}(m \log m||A||))$ bit operations.

Proof Compute a number z such that $b = \prod_{p \leq z}^{p \text{ prime}} p > m^{m/2}\|A\|^m$. By Hadamard's bound every minor of A is has magnitude less than b. For each prime $p \leq z = O(m \log m||A||)$, compute the rank of $(A \bmod p)$, and let p be the first prime for which A has rank m over $\mathbf{Z}_p$. This can be accomplished in $O(m^{\theta-1} n\mathsf{B}(b))$ bit operations using an algorithm of Ibarra, Moran and Hui [7, 1982]. The algorithm of Ibarra *et al.* [7, 1982] also returns a maximal set of linearly independent rows over $\mathbf{Z}_p$, and this gives the permutation matrix P such that PA can be written as in (8) with A_1 nonsingular over $\mathbf{Z}$ and $\mathbf{Z}_p$. Fix some j with $2 \leq j \leq l$. Our goal is to compute a matrix R_j such that

$\bar{A}_j = A_j + R_j A_1$ is nonsingular. Note that if $(\bar{A}_j \bmod p)$ is nonsingular over $\mathbf{Z}_p$, then $\bar{A}_j$ will be nonsingular over $\mathbf{Z}$. Our approach is to find an R_j with entries between 0 and $p-1$ such that $(\bar{A}_j \bmod p)$ is nonsingular over $\mathbf{Z}_p$. Since A_1 is nonsingular over $\mathbf{Z}_p$, $(\bar{A}_j \bmod p)$ will be nonsingular over $\mathbf{Z}_p$ if and only $(\bar{A}_j A_1^{-1} \bmod p)$ is nonsingular over $\mathbf{Z}_p$. Working mod p, decompose $(A_j A_1^{-1} \bmod p)$ as the sum of a unit upper triangular matrix U_j and lower triangular matrix L_j. Set $R_j = (-L_j \bmod p)$. Then entries in R_j are between 0 and $p-1$ and

$$\begin{aligned} \bar{A}_j A_1^{-1} &\equiv (A_j A_1^{-1} + R_j) \bmod p \\ &\equiv ((U_j + L_j)) - L_j) \bmod p \\ &\equiv U_j \bmod p \end{aligned}$$

whence $(\bar{A}_j A_1^{-1} \bmod p)$ is nonsingular over $\mathbf{Z}_p$. The cost of computing $(A_j A_1^{-1} \bmod p)$ for $2 \le j \le l$ is bounded by $O(m^{\theta-1} n \mathsf{B}(\log p))$ bit operatons. ■

4.2 A Special Case of the Algorithm

In this section we develop our algorithm to compute pre-multiplier matrices for a special class of input matrices. Fix some positive integer paramaters m and d, and let $\mathcal{T}_l = \mathcal{T}_l[m,d]$ be the set of all $lm \times m$ rank m integral matrices which can be written using a block decomposition as

$$\begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_l \end{bmatrix} \tag{9}$$

where each $m \times m$ block A_i is either the zero matrix or is nonsingular and in Hermite normal form, and where d is a positive multiple of $\det(A_i)$ for $1 \le i \le l$. Let $H(l) = H[m,d](l)$ denote a function which bounds the number of bit operations required to compute, for a given input matrix $A \in \mathcal{T}_l[m,d]$, a unimodular matrix U such that $UA = H$, the Hermite normal form of A. Our result is the following.

Theorem 10 $H[m,d](l) \ll l \log(2l) m^\theta \mathsf{B}(\log(2l) \log md)$.

Our proof of Theorem 10 has two parts. First, we prove the existence of a deterministic algorithm that constructs a candidate for U that will both be sparse and have small entries. For the second part, we analyse the complexity of the algorithm.

We need to define some notation. A $pm \times qm$ matrix X can be written using a block decomposition as

$$\begin{bmatrix} \bar{X}_{11} & \bar{X}_{12} & \cdots & \bar{X}_{1q} \\ \bar{X}_{21} & \bar{X}_{22} & & \\ \vdots & & \ddots & \\ \bar{X}_{p1} & & & \bar{X}_{pq} \end{bmatrix} \tag{10}$$

where each block is $m \times m$. We denote the submatrix $\bar{X}_{ij}$ by $\mathsf{block}(X,i,j) = \mathsf{block}_m(X,i,j)$. For $1 \le i \le p$, we also define $\mathsf{L}(X,i) = \mathsf{L}_m(X,i)$ by

$$\mathsf{L}(X,i) := \{j : 1 \le j \le q,\ \mathsf{block}(X,i,j) \text{ is not the zero matrix}\}.$$

The quantity $\sum_{i=1}^{p} |\mathsf{L}(X,i)|$ indicates the degree of sparsity of the matrix X, that is, the number of nonzero $m \times m$ blocks in the decomposition (10).

Lemma 11 *There exists a deterministic algorithm that takes as input an $2^k m \times m$ integral matrix $A \in \mathcal{T}_{2^k}[m,d]$, and produces as output:*

- *the Hermite normal form H of A,*
- *a unimodular pre-multilier U such that $UA = H$,*
- *the lists $\mathsf{L}(U,i)$ for $1 \le i \le 2^k$.*

If the last tm rows of A are zero, then U can be written as $\mathrm{diag}(\bar{U}, I_{tm})$. *Furthermore, $\sum_{i=1}^{2^k} |\mathsf{L}(U,i)| \le 2^k(k+1)$, and if d is a positive integer multiple of* $\det(A_i)$ *for $1 \le i \le 2^k$, then $||U|| \le (md^2)^k$.*

Proof We prove the existence of an algorithm which satisfies the requirements of Lemma 11 by induction on k. For the initial case $k = 0$, set $U \leftarrow I_m$, $H \leftarrow A$ and $L_1 \leftarrow \{1\}$. The bounds on $\sum_{i=1}^{2^k} |L_i|$ and $||U||$ are trivially satsified. Assume the lemma holds for $k = N$. To prove the lemma holds for $k = N+1$, let A be a $2^{N+1} m \times m$ input matrix in $\mathcal{T}_{2^{N+1}}$. Write A in block form as

$$\left[\begin{array}{c} \hat{A}_1 \\ \hline \hat{A}_2 \end{array}\right]$$

where $\hat{A}_1$ and $\hat{A}_2$ are $2^N m \times m$ and in $\mathcal{T}_{2^N}$. By induction, there exists a determinisitic algorithm that computes $2^N m \times 2^N m$ unimodular U_1 and U_2 such that

$$\left[\begin{array}{c|c} U_1 & \\ \hline & U_2 \end{array}\right] \left[\begin{array}{c} \hat{A}_1 \\ \hline \hat{A}_2 \end{array}\right] = \left[\begin{array}{c} H_1 \\ O \\ \vdots \\ O \\ \hline H_2 \\ O \\ \vdots \\ O \end{array}\right]$$

where H_1 and H_2 are $m \times m$ and in Hermite normal form with determinants bounded by d. We also get the lists $\mathsf{L}(U_1,i)$ and $\mathsf{L}(U_2,i)$ for $1 \le i \le 2^N$. Compute a $2m \times 2m$ unimodular matrix

$$\left[\begin{array}{c|c} P & Q \\ \hline R & S \end{array}\right],$$

with each block $m \times m$, and that satisfies

$$\left[\begin{array}{c|c} P & Q \\ \hline R & S \end{array}\right] \left[\begin{array}{c} H_1 \\ \hline H_2 \end{array}\right] = \left[\begin{array}{c} H \\ \hline O \end{array}\right] \tag{11}$$

where H is $m \times m$ and in Hermite normal form.

Remark 1: By Lemma 8, the cost of producing P,Q,R and S is bounded by $O(m^\theta \mathsf{B}(\log d))$ bit operations.

Embed P,Q,R and S into the $2^{N+1} m \times 2^{N+1} m$ identity matrix and compute U as

$$U = \left[\begin{array}{ccc|ccc} P & & & Q & & \\ & I & & & & \\ & & \ddots & & & \\ & & I & & & \\ \hline R & & & S & & \\ & & & & I & \\ & & & & & \ddots \\ & & & & & I \end{array}\right] \left[\begin{array}{c|c} U_1 & \\ \hline & U_2 \end{array}\right] \tag{12}$$

so that U is unimodular and satisfies

$$UA = \begin{bmatrix} H \\ O \\ \vdots \\ O \end{bmatrix}.$$

By the induction hypothesis, $\|U_1\|, \|U_2\| \leq (md^2)^N$, and by Lemma 7, entries in P, Q, R and S are bounded by d^2. It follows that $\|U\| \leq m \cdot d^2 \cdot (md^2)^N = (md^2)^{N+1}$ as required.

> Remark 2: The matrix multiplication in (12) requires 2^N multiplications of pairs of $2m \times 2m$ blocks. Since $\|U\| \leq (md^2)^{N+1}$, entries in U are bounded in length by $\log \|U\| = O(N \log md)$ bits — this bounds as well the length of entries in the matrices P, Q, R, S and U_1, U_2. This leads to a bound of $O(2^N \mathsf{MM}(m)\mathsf{B}(N \log md))$ bit operations to accomplish the multiplication in (12).

For convenience, define $b = 2^N$. By (12), we have $\mathsf{L}(U, i) = \mathsf{L}(U_1, i)$ for $2 \leq i \leq b$. Similarly, for $2 \leq i \leq b$, we can obtain $\mathsf{L}(U, b+i)$ from the list $\mathsf{L}(U_2, i)$ by adding 2^N to each entry. The lists $\mathsf{L}(U, 1)$ and $\mathsf{L}(U, b+1)$ can be computed by examining each of the blocks $\mathsf{block}(U, 1, j)$ and $\mathsf{block}(U, b+1, j)$ for $1 \leq j \leq 2^{N+1}$.

> Remark 3: The cost of examining each of the blocks $\mathsf{block}(U, 1, j)$ and $\mathsf{block}(U, b+1, j)$ for $1 \leq j \leq 2b$ is bounded by $O(bm \log d)$ bit operations, and this bounds the total cost of computing $\mathsf{L}(U, i)$ for $1 \leq i \leq 2b$.

It remains to bound the quantity $\sum_{i=1}^{2b} |\mathsf{L}(U, i)|$. It follows from the construction of U in (12) that both $|\mathsf{L}(U, 1)|$ and $|\mathsf{L}(U, b+1)|$ are bounded by $|\mathsf{L}(U_1, 1)| + |\mathsf{L}(U_2, 1)|$. We get

$$\begin{aligned}
&\sum_{i=1}^{2b} |\mathsf{L}(U, i)| \\
&= |\mathsf{L}(U,1)| + |\mathsf{L}(U,b+1)| + \sum_{i=2}^{b} (|\mathsf{L}(U,i)| + |\mathsf{L}(U,b+i)|) \\
&= 2(|\mathsf{L}(U_1,1)| + |\mathsf{L}(U_2,1)|) + \sum_{i=2}^{b} (|\mathsf{L}(U_1,i)| + |\mathsf{L}(U_2,i)|) \\
&\leq |\mathsf{L}(U_1,1)| + |\mathsf{L}(U_2,1)| + \sum_{i=1}^{b} (|\mathsf{L}(U_1,i)| + |\mathsf{L}(U_2,i)|) \\
&\leq 2^N + 2^N + 2\,2^N(N+1) \\
&= 2^{N+1} + 2^{N+1}(N+1) \\
&= 2^{N+1}(N+2)
\end{aligned}$$

as required. ∎

We now return to the proof of Theorem 10. We claim that for $H(l) = H[m, d](l)$ we have

$$H(l) \leq 2H(l/2) + clm^\theta \mathsf{B}(\log_2(2l) \log md) \qquad (13)$$

for some absolute constant c. Let A be in $\mathcal{T}_l$ with Hermite normal form H. By augmenting A with at most $(l-1)m$ rows of zeros, we may assume that $l = 2^{N+1}$ for some $N \in \mathbf{Z}$. Rewrite (13) with $l = 2^{N+1}$ to get

$$H(2^{N+1}) \leq 2H(2^N) + c2^{N+1} m^\theta \mathsf{B}((N+2)) \log md) \qquad (14)$$

The algorithm presented in Lemma 11 reduces the problem of computing a unimodular U with UA in Hermite normal form to two subproblems in $\mathcal{T}_{2N}$ and some combining steps. Thus, to prove (14), it is sufficient to show that the cost of the combining steps is bounded by $O(2^{N+1} m^\theta \mathsf{B}((N+2) \log md)$ bit operations — but this follows from Remarks 1,2 and 3. Iterate (14) to get

$$\begin{aligned}
&H(2^{N+1}) \\
&\leq 2H(2^N) + c2^{N+1} m^\theta \mathsf{B}((N+2) \log md) \\
&\leq 4H(2^{N-1}) + cm^\theta \mathsf{B}((N+2) \log md)(2^{N+1} + 2 \cdot 2^N) \\
&\vdots \\
&\leq 2^{N+1} H(0) + cm^\theta \mathsf{B}((N+2) \log md) \sum_{i=0}^{N+2} 2^i 2^{N+2-i} \\
&\ll 2^N N m^\theta \mathsf{B}(N \log md)
\end{aligned}$$

and we have our result since $l = 2^{N+1}$. ∎

4.3 The Premultiplier Algorithm

Theorem 12 *There exists a deterministic algorithm that takes as input an $n \times m$ rank m integral matrix A, and produces as output the Hermite normal form H of A together with a unimodular pre-multiplier matrix U that satisfies $UA = H$. The running time of the algorithm is bounded by $O(m^{\theta-1} n \log(2n/m) \mathsf{B}(\log(2n/m) m \log m \|A\|))$ bit operations.*

Proof The algorithm has four steps. First, use the algorithm developed in Subsection 4.1 to compute an $n \times n$ permutation matrix P and an $n \times n$ unimodular matrix

$$R = \begin{bmatrix} I_m & & & & & \\ R_2 & I_m & & & & \\ R_2 & & I_m & & & \\ \vdots & & & \ddots & & \\ R_l & & & & I_m & \\ O & & & & & I_t \end{bmatrix}$$

such that

$$\bar{A} = RPA = \begin{bmatrix} \bar{A}_1 \\ \bar{A}_2 \\ \vdots \\ \bar{A}_l \\ B \end{bmatrix}$$

where each $m \times m$ submatrix $\bar{A}_i$ is nonsingular, $l = \lfloor n/m \rfloor$, and B is $t \times m$ where $t = n - lm$. By Lemma 9, the cost of this will bounded by $O(m^{\theta-1} n \mathsf{B}(m \log m \|A\|))$ bit operations. Secondly, compute an $(m+t) \times m$ unimodular matrix U_l such that

$$U_l \left[\begin{array}{c|c} \bar{A}_l & \\ \hline B & I_t \end{array}\right] = \begin{bmatrix} H_l \\ O \end{bmatrix}$$

with H_l $m \times m$ and in Hermite normal form. Thirdly, compute matrices $U_1, U_2, \ldots, U_{l-1}$, each $m \times m$ and unimodular, and satisfying

$$\begin{bmatrix} U_1 & & & & \\ & U_2 & & & \\ & & \ddots & & \\ & & & U_{l-1} & \\ & & & & U_l \end{bmatrix} \bar{A} = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_l \\ O \end{bmatrix}$$

with each H_i an $m \times m$ matrix in Hermite normal form for $1 \leq i \leq l$. By Theorem 2, the cost of computing U_i and H_i for $1 \leq i \leq l$ is bounded by $O(lm^\theta \mathsf{B}(m \log m \|\bar{A}\|)$ bit operations. Compute the quantity $d = \text{lcm}(\det(H_1), \det(H_2), \ldots, \det(H_l))$. By Hadamards inequality on determinants, we will have $\log d = O(m \log m \|\bar{A}\|)$. Fourthly, use the algorithm described in Subsection 4.2 to compute an $lm \times lm$ unimodular S satisfying

$$S \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_l \end{bmatrix} = \begin{bmatrix} H \\ O \end{bmatrix}$$

where H is $m \times m$ and in Hermite normal form. By Theorem (10), the cost of this will be $O(m^{\theta-1} n \log(2n/m) \mathsf{B}(\log(2n/m) m \log m \|\bar{A}\|))$ bit operations, which bounds the total cost up until this point. Finally, set

$$U = \begin{bmatrix} S & \\ & I_t \end{bmatrix} \begin{bmatrix} U_1 & & & & \\ & U_2 & & & \\ & & \ddots & & \\ & & & U_{l-1} & \\ & & & & U_l \end{bmatrix} RP \qquad (15)$$

so that U is unimodular with UA the Hermite normal form of A. It remains to establish a bound on the cost of the matrix multiplications in (15). By Lemma 11, we have $2 \sum_{i=1}^{k} |\mathsf{L}_m(S, i)| \leq 2((2n/m) \log(4n/m))$, which, because of the special structure of R and the second matrix in the product (15), bounds the number of pairs of $m \times m$ matrix blocks which need to be multiplied. By the bounds established on the magnitudes of entries in S, $U_1, U_2, \ldots, U_l$, we have

$$\log \|U\| = O(\log(2n/m) m \log m \|\bar{A}\|),$$

leading to a cost for the multiplications in (15) of $O(m^{\theta-1} n \log(2n/m) \mathsf{B}(\log(2n/m) m \log m \|\bar{A}\|))$ bit operations. By Lemma 9, the entries in R will be bounded in magnitude by $cm \log m ||A||$ bits for some absolute constant c. This leads to the bound $\|\bar{A}\| \leq cm^2 \|A\| \log m \|A\|$. The result now follows by noting that $m \log m \|\bar{A}\| = O(m \log m \|A\|)$. ■

References

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.

[2] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9:251–280, 1990.

[3] P. D. Domich, R. Kannan, and L. E. Trotter, Jr. Hermite normal form computation using modulo determinant arithmetic. *Mathematics of Operations Research*, 12(1):50–59, Feb. 1987.

[4] J. L. Hafner and K. S. McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM Journal of Computing*, 20(6):1068–1083, Dec. 1991.

[5] C. Hermite. Sur l'introduction des variables continues dans la théorie des nombres. *J. Reine Angew. Math.*, 41:191–216, 1851.

[6] T. C. Hu. *Integer Programming and Network Flows*. Addison-Wesley, Reading, MA, 1969.

[7] O. Ibarra, S. Moran, and R. Hui. A generalization of the fast LUP matrix decomposition algorithm and applications. *SIAM Journal of Computing*, 3:45–56, 1982.

[8] C. S. Iliopoulos. Worst-case complexity bounds on algorithms for computing the canonical structure of infinite abelian groups and solving systems of linear diophantine equations. *SIAM Journal of Computing*, 18(4):670–678, Aug. 1989.

[9] C. S. Iliopoulos. Worst-case complexity bounds on algorithms for computing the canonical structure of finite abelian groups and the Hermite and Smith normal forms of an integer matrix. *SIAM Journal of Computing*, 18(4):658–669, Aug. 1989.

[10] M. Newman. *Integral Matrices*. Academic Press, 1972.

[11] A. Schönhage and V. Strassen. Schnelle Multiplikation grosser Zahlen. *Computing*, pages 281–292, 1971.

[12] A. Storjohann and G. Labahn. Preconditioning of rectangular polynomial matrices for efficient Hermite normal form computation. In *Proceedings of ISSAC'95*, pages 119–125, Montreal, Canada, 1995.