Kernels of Integer Matrices via Modular Arithmetic

Johannes Buchmann
Fachbereich Informatik
Technische Universität Darmstadt
Alexanderstr. 10
D-64283 Darmstadt, Germany
buchmann@cdc.informatik.tu-darmstadt.de

Douglas Squirrel
Department of Mathematics
University of California at Berkeley
Berkeley, CA 94720-3840, USA
squirrel@math.berkeley.edu

Abstract

Let Q be an m by n integer matrix of rank e and let $\alpha : \mathbf{Z}^n \to \mathbf{Z}^m$ be the transformation given by $\alpha(x) = Qx$. We give an new algorithm which, like earlier algorithms for the *image* of α, computes the *kernel* of α using modular arithmetic.

1 Introduction

Let Q be an m by n integer matrix. In this paper we present a new algorithm for computing the kernel of Q, i.e. the kernel of the map

$$\alpha : \mathbf{Z}^n \to \mathbf{Z}^m, \quad \alpha : x \mapsto Qx$$

which is a homomorphism of lattices (see Section 2). Our algorithm computes the uniquely determined basis of the kernel of Q which is in Hermite normal form (see Section 2).

The computation of the integer kernel of an integer matrix is necessary for the solution of important problems in computational number theory. It is, for example, a key step in the determination of a system of fundamental units of an algebraic number field (see [1]). There are also applications to group theory, since abelian groups are $\mathbf{Z}$-modules (see for example [9]).

The problem of computing the *image* of α (that is, the HNF-basis of the image of α) has been studied extensively, for example in [4], [10], [14], [15], [11]), [7], [6], [12], and [8]. The first five of these algorithms suffer, to one degree or another, from an explosion in the size of integers used in intermediate stages, a phenomenon known as *entry explosion* which affects many algorithms over $\mathbf{Z}$. The last four of these algorithms use modular arithmetic, with the modulus being any integer multiple of the determinant of the lattice generated by the columns of Q. Therefore, these algorithms avoid entry explosion; we call them *modular image algorithms.*

Any algorithm for the image of α gives as output a matrix Q' which is *equivalent* to Q, that is, such that $Q' = QU$ for some unimodular n by n integer matrix U. One can read off the kernel of α directly from U (see [5]), and so any image algorithm which computes such a U is also a kernel algorithm. Unfortunately, only the nonmodular image algorithms cited above compute such a U, and so no modular kernel algorithm is currently available.

Our new algorithm is such a modular kernel algorithm. That is, it is an algorithm for the kernel of α which is an analogue of the modular HNF-algorithms. It determines the kernel of Q by means of computations modulo the determinant of a submatrix of Q, thereby avoiding entry explosion. More precisely, it proceeds in two steps.

Let e be the rank of Q. Then $f = n - e$ is the dimension of the kernel of Q.

First, a non-singular e by e submatrix Q_1 of Q is computed, and $d = |\det Q_1|$ and $\operatorname{adj} Q_1$ are determined. This is achieved by means of a modification of the chinese remaindering algorithm of Hafner and McCurley [8]. By swapping columns and rows, the matrix Q is transformed into the form

$$Q = \begin{pmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{pmatrix}$$

It is easy to deduce the kernel of the original matrix from the kernel of the transformed matrix. Further, we show by an easy argument that we may assume

$$Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}, \quad d > 0.$$

In the second step, using the kernel of $Q \bmod d$, i.e. computations modulo d, an integer f by f matrix S_2 is determined such that

$$R = \begin{pmatrix} -Q_1^{-1} Q_2 S_2 \\ S_2 \end{pmatrix} = \begin{pmatrix} -(\operatorname{adj} Q_1) Q_2 S_2 / d \\ S_2 \end{pmatrix} \qquad (1)$$

is an integer matrix in Hermite normal form (see Section 2) whose columns form a basis of the kernel of Q. It is easy to compute R from the above formula, or to retain it in a product representation (which may be preferable).

In the course of the paper, we will prove that the above algorithm is correct. We will also prove complexity results as in the following theorem. Let $z = \min\{m, n\}$, let $||Q||$ be the maximum of the absolute values of the entries of Q, and let $L = z \log(z||Q||)$. Arithmetic operations on integers are addition, subtraction, multiplication, division with remainder, and extended gcd. We say that an integer x is of *size* s if the number of bits in its binary expansion is bounded by s.

Theorem 1 *The algorithm given above correctly computes a matrix R whose columns form a basis of the kernel of Q. Further,*

1. *the computation of Q_1, $\operatorname{adj} Q_1$, and $d = |\det Q_1|$ can be accomplished using $O(emnz)$ arithmetic operations on integers of size $O(L)$, and*
2. *the computation of S_2 can be accomplished using $O(en^2)$ arithmetic operations on integers of size $O(\log d)$.*

Note that the smaller d and the entries of $\operatorname{adj} Q$ are, the faster and more space-efficient our algorithm is.

The rest of the paper is organized as follows. Throughout the paper, we let Q be a given m by n integer matrix. In section 2 we set up notation and other preliminaries. In section 3 we describe the computation of the objects Q_1, d, and $\operatorname{adj} Q_1$ and reduce the problem to the case

$$Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}, \quad d > 0.$$

In section 4, we give conditions that are to be satisfied by a matrix S, of which the matrix S_2 will be a submatrix, and we prove that the columns of the matrix R defined by (1) indeed form the HNF-basis of the kernel of Q. Then in section 5, we show how to compute a matrix S meeting the given conditions. In section 6, we prove Theorem 1. Finally, in section 7, we work out an example using our algorithm, and in section 8 we give some timings for an implementation of the algorithm, using various input matrices with small entries.

The authors would like to express their thanks to the members of the LiDIA-Group at the TU-Darmstadt, especially Volker Müller, for helpful conversations. The second author acknowledges support from a National Science Foundation Graduate Fellowship.

2 Preliminaries

Let m and n be positive integers and R a ring. We write $\mathbf{Mat}_{m,n}(R)$ for the set of all matrices with m rows and n columns and entries in R. If $X \in \mathbf{Mat}_{m,n}(R)$ then we write x_{ij} for the entry of X in the ith row and jth column and x_j for the column vector equal to the jth column of X. If x is a vector in R^m then we write $x[i]$ for the ith entry of x, and we define the *last entry function* $\gamma(x)$ by declaring that $\gamma(x)$ is the index of the last nonzero entry in x, i.e. the integer such that

$$x[\gamma(x)] \neq 0, \quad x[\gamma(x)+1] = x[\gamma(x)+2] = \ldots = x[m] = 0.$$

The *ith standard basis vector of* R^n, denoted e_i, is the vector of R^n whose ith entry is 1 and whose other entries are zero.

If R is a ring, then by the terms *kernel* and *image* of $X \in \mathbf{Mat}_{m,n}(R)$ we shall always mean the kernel and image of the homomorphism

$$\mathbf{Z}^n \to \mathbf{Z}^m, \quad v \mapsto Xv.$$

If X is a matrix with entries in $\mathbf{Z}$ then we define $\|X\| = \max\ \{|x_{ij}|\}$. Note that if $X \in \mathbf{Mat}_{m,n}(\mathbf{Z})$ and $Y \in \mathbf{Mat}_{n,p}(\mathbf{Z})$ then $\|XY\| \leq n\|X\|\|Y\|$.

If m is a positive integer and X is a matrix in $\mathbf{Mat}_{m,m}(\mathbf{Z})$ with nonzero determinant, then the *adjoint* of X, denoted $\operatorname{adj} X$, is the unique matrix $Y \in \mathbf{Mat}_{m,m}(\mathbf{Z})$ such that

$$XY = YX = (\det X)I$$

where I is the identity matrix in $\mathbf{Mat}_{m,m}(\mathbf{Z})$. If Y is the adjoint of X then

$$y_{ij} = (-1)^{i+j} d_{ji}$$

where d_{ji} is the determinant of the submatrix X' obtained by removing row j and column i from X. Note that $\operatorname{adj} X = (\det X)X^{-1}$.

Suppose that X is a matrix in $\mathbf{Mat}_{m,m}(\mathbf{Z})$. It follows from Hadamard's inequality (proved in [5], Corollary 2.5.5, for example) that

$$|\det X| \leq (m\|X\|)^m \quad \text{and} \quad \|\operatorname{adj} X\| \leq ((m-1)\|X\|)^{m-1}.$$

A matrix $H \in \mathbf{Mat}_{m,n}(\mathbf{Z})$ is in *Hermite normal form* if there exists an integer $r \leq n$ such that the first r columns of H are 0 and, when $r+1 \leq j < k \leq n$, we have $\gamma(h_j) < \gamma(h_k)$, $h_{\gamma(h_j),j} \geq 1$ and $0 \leq h_{\gamma(h_j),k} < h_{\gamma(h_j),j}$. If X is any matrix in $\mathbf{Mat}_{m,n}(\mathbf{Z})$, then there is a unimodular matrix U in $\mathbf{Mat}_{n,n}(\mathbf{Z})$ such that XU is in Hermite normal form (see [13], Theorem II.2). The matrix U is not uniquely determined but the matrix XU is unique; XU is called the *Hermite normal form of* X.

We say that a matrix $H \in \mathbf{Mat}_{m,n}(\mathbf{Z})$ is in *triangular Hermite normal form* if the following three conditions are met:

1. $m \geq n$,
2. H is in Hermite normal form, and
3. $\gamma(h_i) = (m-n) + i$ for each $i \in \{1, 2, \ldots, n\}$.

A *lattice* is an additive subgroup L of $\mathbf{R}^k$ for some positive integer k which as a point set is discrete; all our lattices will be subsets of the lattice $\mathbf{Z}^k$. The lattice L can be written as $L = \sum_{i=1}^{t} \mathbf{Z}b_i$ with $0 \leq t \leq k$ and linearly independent vectors $b_1, b_2, \ldots, b_t \in L$. The integer t is an invariant of L, called the *dimension* of L. The sequence $(b_1, \ldots, b_t)$ is called a *basis* of L, and the *matrix B associated to this basis* is the k by t integer matrix whose jth column is the vector b_j. A lattice has many bases, but a canonical one exists, namely the basis whose associated matrix is in Hermite normal form; this basis is called the HNF-basis of the lattice.

As usual, if n is an integer then $\lg n$ is the number of bits in the binary representation of n. We use the term "arithmetic operation" to mean one of the following operations on two integers: addition, subtraction, multiplication, division with remainder, and extended gcd.

3 Computing Q_1, d, $\operatorname{adj} Q_1$; reductions

We use a modification of an algorithm of Hafner and McCurley [8] to compute Q_1, d, and $\operatorname{adj} Q_1$. That algorithm computes $e = \operatorname{rank}(Q)$, a nonsingular e by e submatrix Q_1 of Q, and the determinant of Q_1. We sketch the original algorithm, adding to it the computation of the adjoint $\operatorname{adj} Q_1$. Let $z = \min\{m, n\}$. First, the algorithm determines a positive integer h with

$$h = O(z \log(z\|Q\|))$$

such that there is a prime number $p \leq z$ for which the rank of Q modulo p is e, i.e. the rank of Q. For each prime $p \leq h$ the algorithm determines the rank e_p of Q mod p and a submatrix Y_p of Q whose rank mod p is e_p. If q is a prime with $e_q = \max\{e_p : p \leq h\}$ then $e = e_q$ and we set $Q_1 = Y_q$. Then $\det Q_1$ and $\operatorname{adj} Q_1$ can be computed using Gaussian elimination and Chinese remaindering.

Now we show how, once Q_1, d, and $\operatorname{adj} Q_1$ have been computed, we may reduce to the case

$$Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}$$

where $d = \det Q_1 > 0$. Let e be the rank of Q and set $f = n - e$. By swapping columns and rows we transform Q into the form

$$Q = \begin{pmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{pmatrix}$$

with

$$Q_1 \in \mathbf{Mat}_{e,e}(\mathbf{Z}), \quad Q_2 \in \mathbf{Mat}_{e,f}(\mathbf{Z}),$$
$$Q_3 \in \mathbf{Mat}_{m-e,e}(\mathbf{Z}), \quad Q_4 \in \mathbf{Mat}_{m-e,f}(\mathbf{Z}).$$

By swapping at most one more row, we can ensure that $\det Q_1 > 0$. If we know the kernel of the transformed matrix, it is easy to determine the kernel of the original matrix. We have therefore reduced to the case

$$Q = \begin{pmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{pmatrix}, \quad d = \det Q_1 > 0.$$

We further reduce by proving the following result.

Proposition 1 *The kernel of Q is the kernel of $Q' = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}$.*

Proof Clearly the kernel of Q is a subset of the kernel of Q'; we proceed to show the reverse inclusion. Since the rank of Q is e there is a matrix $T \in \mathbf{Mat}_{m-e,e}(\mathbf{Q})$ such that

$$Q = \begin{pmatrix} Q' \\ TQ' \end{pmatrix}$$

If x is in the kernel of Q' then

$$Qx = \begin{pmatrix} Q'x \\ TQ'x \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

so x is in the kernel of Q. □

Thus we can assume, in addition to the condition $d > 0$, that

$$Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix},$$

or in other words that Q is of rank m.

4 R is the kernel of Q

In this section we give conditions which a matrix S is to satisfy; the matrix S_2 will be a submatrix of S. Then we prove that the matrix R given by (1) is indeed the unique matrix in Hermite normal form whose columns generate the kernel of Q. The reductions outlined in the previous section mean that we may assume that Q is of rank m, with Q_1 the submatrix formed by the first m columns of Q, and $d > 0$.

To describe our algorithm we need some notation. Let

$$\Phi = \{x \in \mathbf{Z}^n : Qx \equiv 0 \bmod d\}.$$

Also, for $j \in \{0, 1, 2, \ldots, n\}$ we set

$$\begin{aligned} \Phi_j &= \{x \in \Phi \mid x[j+1] = \ldots = x[n] = 0\}, \\ \phi_j &= \{c \in \mathbf{Z} \mid x[j] = c \text{ for some } x \in \Phi_j\}. \end{aligned}$$

Note that Φ_j is a sublattice of $\mathbf{Z}^n$ and ϕ_j is a $\mathbf{Z}$-ideal for $0 \le j \le n$. Further,

$$\{0\} = \Phi_0 \subset \Phi_1 \subset \ldots \subset \Phi_n = \Phi.$$

We will show below how to construct a matrix $S \in \mathbf{Mat}_{n,f}(\mathbf{Z})$ with the following properties:

1. $QS \equiv 0 \bmod d$,
2. all entries of S lie in $\{0, 1, \ldots, d\}$,
3. S is in triangular Hermite normal form, and
4. the entry $s_{m+j,j}$ of S generates ϕ_{m+j} for each $j = 1, 2, \ldots, f$.

Assume that S is known and write

$$S = \begin{pmatrix} S_1 \\ S_2 \end{pmatrix}, \quad S_1 \in \mathbf{Mat}_{m,f}(\mathbf{Z}), S_2 \in \mathbf{Mat}_{f,f}(\mathbf{Z}).$$

Let

$$R = \begin{pmatrix} -Q_1^{-1}Q_2S_2 \\ S_2 \end{pmatrix} = \begin{pmatrix} -(\operatorname{adj} Q_1)Q_2S_2/d \\ S_2 \end{pmatrix}.$$

Theorem 2 *The matrix R has integer entries, it is in triangular Hermite normal form, and its columns form a basis of the kernel of Q.*

Proof We show that R has integer entries. We know that $QS \equiv 0 \bmod d$. This means that $Q_1S_1 + Q_2S_2 = dS_3$ with $S_3 \in \mathbf{Mat}_{e,f}(\mathbf{Z})$. Therefore, $Q_1^{-1}Q_2S_2 = dQ_1^{-1}S_3 - S_1$. Since both $dQ_1^{-1} = \operatorname{adj} Q_1$ and S_1 are integer matrices it follows that $Q_1^{-1}Q_2S_2$ is also an integer matrix. Note that R is in triangular Hermite normal form because S is in triangular Hermite normal form and the last f rows of R and S are identical.

Note that

$$\begin{aligned} QR &= \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} -Q_1^{-1}Q_2S_2 \\ S_2 \end{pmatrix} \\ &= -Q_2S_2 + Q_2S_2 = 0 \end{aligned}$$

so the columns of R belong to the kernel of Q.

It remains to be shown that the columns of R form a basis of the kernel of Q. Let T be the unique matrix in Hermite normal form whose columns generate the kernel of Q. Since Q_1 is nonsingular, T must be in triangular Hermite normal form. For $j \in \{0, \cdots, f\}$, let L_j be the lattice generated by the first j columns of R and let L'_j be the lattice generated by the first j columns of T. Clearly, $L_j \subset L'_j$ for each j. We now prove, by induction on j, that $L'_j \subset L_j$. For $j = 0$ the assertion is trivially correct. Suppose that the assertion holds for each $j' < j$. We have $t_j \in \Phi_j$ so $t_{m+j,j} \in \phi_j$. Since $r_{m+j,j}$ generates ϕ_j there must be an integer c such that $t_{m+j,j} = cr_{m+j,j}$. Hence, $t_j - cr_j \in L_{j-1}$. Applying the induction hypothesis, we see that $t_j \in L_j$ and it follows immediately that $L'_j \subset L_j$, completing the induction. Now we know that $L'_j = L_j$ for each j; applying this with $j = n$ shows that the columns of R form a basis of the kernel of Q. □

5 Computation of S

In this section, we show how to compute the matrix S using computations mod d.

We use an algorithm from [3] to compute the matrix $Y \in \mathbf{Mat}_{n,r}(\mathbf{Z})$ in Hermite normal form which satisfies the following conditions:

1. The entries of Y lie in $\{0, 1, \ldots, d-1\}$.

2. The columns of Y together with $de_1, \ldots, de_n$ generate the lattice Φ.

3. For $j \in \{1, \ldots, r\}$ the $\gamma(y_j)$th entry of y_j generates the ideal $\phi_{\gamma(y_j)}$.

We describe the algorithm which we use to produce Y. The proofs can be found in [3]. First, we reduce Q modulo d, obtaining a matrix $\overline{Q} \in \mathbf{Mat}_{m,n}(\mathbf{Z}/d\mathbf{Z})$, and we set a matrix T equal to the identity in $\mathbf{Mat}_{n,n}(\mathbf{Z}/d\mathbf{Z})$. Next, we begin a loop which will process each row of $\overline{Q}$ in turn, starting with the mth. The loop variable i is initialized to m and decreases on each pass until it reaches 1. In the ith pass through the loop, we perform the following two steps.

1. Use Gaussian elimination in $\mathbf{Z}/d\mathbf{Z}$ to zero out all but the last of the entries in the ith row of $\overline{Q}$ (an analogue of the usual extended gcd algorithm allows us to do this); perform all the same column operations on T.

2. Let a be the remaining nonzero element of the ith row; if $ab = 0$ for some nonzero $b \in \mathbf{Z}/d\mathbf{Z}$, then multiply the last column of $\overline{Q}$ by b and multiply the last column of T by b also. If, on the other hand, a is a unit, then delete the last column of $\overline{Q}$ (but *not* the last column of T).

When the loop is complete, the columns of T generate the kernel of $\overline{Q}$. Now let A be the zero matrix in $\mathbf{Mat}_{m,m}(\mathbf{Z}/d\mathbf{Z})$. We apply a similar loop to T. The loop variable is again i and it decreases from m to 1. In the ith pass through the loop, we perform the following three steps.

1. Use Gaussian elimination in $\mathbf{Z}/d\mathbf{Z}$ to zero out all but the last of the entries in the ith row of T.

2. Store the last column of T in the ith column of A; multiply a_i by a suitable element of $\mathbf{Z}/d\mathbf{Z}$ so that the last nonzero entry of a_i is a divisor of d.

3. Let a be the remaining nonzero element of the ith row; if $ab = 0$ for some nonzero $b \in \mathbf{Z}/d\mathbf{Z}$, then multiply the last column of T by b. If, on the other hand, a is a unit, then delete the last column of T.

When the loop is complete, the columns of A generate the same submodule of $(\mathbf{Z}/d\mathbf{Z})^m$ as the columns of the original matrix T, i.e. the kernel of $\overline{Q}$. We now delete all zero columns of A and lift the resulting matrix to $\mathbf{Z}$, using the representatives $\{0, 1, 2, \ldots, d-1\}$; the result is the desired matrix Y.

We now show how to construct S from Y. Construct an upper triangular matrix Z as follows. For $j \in \{1, 2, \ldots, n\}$ the jth column of Z is the column y of Y with $\gamma(y) = j$ if such a column exists. Otherwise it is de_j. Then S is the matrix consisting of the last f columns of Z.

We prove that S has the desired properties. By construction, we have $QS \equiv 0 \bmod d$. Since Y is in Hermite normal form, S is in triangular Hermite normal form. Finally, we must show that the entry $s_{e+j,j}$ generates ϕ_{e+j} for $1 \leq j \leq f$. If the jth column of S is equal to a column of Y this is true because of the corresponding property of Y. Assume that the jth column of S is de_{e+j}. Since the columns of Y together with the vectors de_j, $1 \leq j \leq n$ generate Φ it follows that $\phi_j = d$.

6 Analysis

The correctness of the algorithm given above is obvious from Theorem 2. We complete the proof of Theorem 1 by verifying the time and space bounds given there. Let $N = \max\{m, n\}$, $z = \min\{m, n\}$, $L = \log z\|Q\|$. The analysis of [8] shows that the computation of Q_1, d, and $\operatorname{adj} Q_1$ can be accomplished with $O(emnz)$ arithmetic operations on integers of size $O(L)$ (our addition of the computation of $\operatorname{adj} Q_1$ does not change the bound). The reduction to the case $\operatorname{rank} Q = m$, $d > 0$ involves only row and column swaps, not arithmetic. It takes en arithmetic operations on numbers no larger than $\|Q\|$ to reduce Q modulo d, and the analysis of [3] says that $O(en^2)$ arithmetic operations on numbers no larger than d^2 are required for the remainder of the computation of S outlined above. This proves Theorem 1.

7 Example

We work out the example

$$Q = \begin{pmatrix} 4 & 2 & 1 & 1 \\ 2 & 1 & 1 & 4 \\ 6 & 3 & 2 & 5 \end{pmatrix}.$$

We compute easily that $e = 2$, $f = 2$. Swapping rows and columns and then discarding the last row, we get

$$Q = \begin{pmatrix} 2 & 6 & 3 & 5 \\ 1 & 4 & 2 & 1 \end{pmatrix}.$$

Write $Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix}$ with both Q_1 and Q_2 in $\mathbf{Mat}_{2,2}(\mathbf{Z})$. We easily compute that $d = 2$, so Q_1 is nonsingular, and that

$$\operatorname{adj} Q_1 = \begin{pmatrix} 4 & -6 \\ -1 & 2 \end{pmatrix}.$$

(Of course we could achieve the conditions on Q_1 and d with many other sets of row and column swaps.)

It is not hard to verify that (using the notation of section 5)

$$Y = \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}, \quad Z = \begin{pmatrix} 2 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$S = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 2 & 1 \\ 0 & 1 \end{pmatrix}, \quad S_2 = \begin{pmatrix} 2 & 1 \\ 0 & 1 \end{pmatrix}.$$

A simple application of the formula of Theorem 2 then gives

$$R = \begin{pmatrix} 0 & -7 \\ -1 & 1 \\ 2 & 1 \\ 0 & 1 \end{pmatrix}$$

which indeed is the HNF-basis for the kernel of Q as modified in the first step. To recover the kernel of the original Q, we swap rows in a manner consistent with the swaps of columns used earlier; the result is

$$\begin{pmatrix} -1 & 1 \\ 2 & 1 \\ 0 & -7 \\ 0 & 1 \end{pmatrix}.$$

8 Timings

In this section we give a brief indication of the behavior of an implementation of the algorithm using the LiDIA [2] number theory library. We report the CPU time required to run the implementation on a SPARC Ultra for matrices of various sizes. In each case the entries of the matrix were randomly selected from the set $\{0, 1, 2, \ldots, 10\}$. (Matrices with such small entries are those for which our algorithm is most likely to be practical; larger entries, of course, produce larger d's.)

Dimensions	Time required
50x51	4.82 s
50x75	8.31 s
80x81	32.72 s
80x120	56.96 s
100x101	1 m 26.06 s
100x150	2 m 46.76 s
130x131	4 m 37.01 s
130x200	9 m 22.44 s
150x151	9 m 2.06 s
150x200	14 m 9.29 s
180x181	21 m 4.15 s
180x240	33 m 27.81 s
200x201	35 m 16.23 s
200x250	50 m 0.64 s
250x251	1 h 38 m 2.85 s
250x325	2 h 31 m 58.89 s
300x301	3 h 54 m 48.17 s

References

[1] Buchmann, J. A subexponential algorithm for the determination of class groups and regulators of algebraic number fields. In *Séminaire de Théorie des Nombres* (1988–89), pp. 27–41.

[2] Buchmann, J., et al. LiDIA—a library for computational number theory. Web address: http://www.informatik.th-darmstadt.de/TI/LiDIA/.

[3] Buchmann, J., and Neis, S. Algorithms for linear algebra problems over principal ideal rings. Tech. Rep. TI-7/96, Fachbereich Informatik, Technische Universität Darmstadt, 1996.

[4] Chou, T.-W. J., and Collins, G. Algorithms for the solution of systems of linear Diophantine equations. *SIAM Journal on Computing 11* (1982), 687–708.

[5] Cohen, H. *A Course in Computational Algebraic Number Theory.* Springer-Verlag, 1995.

[6] Domich, P. D. Residual Hermite normal form computations. *ACM Transactions on Mathematical Software 15* (1989), 275–286.

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

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

[9] Havas, G., Holt, D., and Rees, S. Recognizing badly presented **Z**-modules. *Linear Algebra and its Applications 192* (1993), 137–163.

[10] Havas, G., and Majewski, B. Hermite normal form computation for integer matrices. *Congressus Numerantium 105* (1994), 87–96.

[11] Havas, G., Majewski, B., and Matthews, K. Extended GCD and Hermite normal form algorithms via lattice basis reduction. *Experimental Mathematics 7* (1998), 125–136.

[12] Iliopoulos, C. S. 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 on Computing 18* (1989), 658–669.

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

[14] Storjohann, A. A fast+practical+deterministic algorithm for triangularizing integer matrices. Tech. Rep. 255, Department of Computer Science, ETH Zürich, 1996.

[15] Storjohann, A., and Labahn, G. Asymptotically fast computation of Hermite normal forms of integer matrices. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC '96)* (1996), Y. N. Lakshman, Ed., ACM Press, pp. 259–266.