COMPUTING THE HERMITE NORMAL FORM
ON AN INTEGRAL MATRIX

by

M. Kaminski and A. Paz

Technical Report #417

June 1986

TECHNION — ISRAEL INSTITUTE OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE

HAIFA, ISRAEL
COMPUTING THE HERMITE NORMAL FORM 
ON AN INTEGRAL MATRIX

by

M. Kaminski and A. Paz*

Technical Report #417

June 1986
Introduction.

The following theorem is well known (cf. [11], say).

Theorem 1. (Hermite) Let $M$ be $n \times n$ integral matrix. There exists a unique unimodular matrix $U$ such that the matrix $UM$ is upper triangular, and each off-diagonal element of $UM$ is non-negative and strictly less than the diagonal element in its column.

$UM$ is called the Hermite normal form of $M$.

Transforming an integral matrix into Hermite normal form plays an important role in the study of integral matrices ([11]) and integer programming ([5]). The best algorithm known from the literature for transforming a matrix into Hermite normal form is the T.-W.J Chow and G.E. Collins ([2]) modification of the R. Kannan and A. Bachem algorithm ([9]). For transforming an $n \times n$ integral matrix $M$ into Hermite normal form it requires $O(n^2 \log |M| + n^4)$ arithmetical operations, where $|M|$ denotes the largest, in absolute value, element of $M$. The intermediate results produced by the algorithm are bounded by $n^6 |M|^2$, i.e., the size of the intermediate results is $O(n \log (n|M|))$.

Before R. Kannan and A. Bachem discovered their algorithm, it has been discovered by T.C. Hu ([8]), that in order to avoid a possible problem of large values of intermediate results in computing the Smith normal form, the computation can be performed modulo the determinant of the matrix. In this paper we show that the computation modulo the determinant can be used not only for computing the Smith normal form but also, after some modification, for computing the Hermite normal form. We present an algorithm for computing the Hermite normal form of $M$ in $O(n^3 + n^2 \log |M|)$ arithmetical operations such that the intermediate results of the computation, in absolute value, do not exceed $n^6 |M|^2$. The algorithm for computing Hermite normal form presented in this paper is similar to an elimination procedure of Gauss. Hence our algorithm is much simpler and faster than the algorithm of R. Kannan and A. Bachem, since it avoids computing the permutations of the rows (to make every principal minor non-singular) and using intermediate normalizations. Reducing the number of arithmetical operations from $O(n^2 \log |M| + n^4)$ to $O(n^2 \log |M| + n^3)$ is important for practical applications, since in
practice, for large values of $n$, the size of the elements of $M$ is often substantially less than $n^2$.

The next section contains a description of our algorithm for computing the Hermite normal form. It is followed by an appendix, where we sketch an algorithm for inverting an integral matrix which, based on an elimination procedure of J. Edmonds ([3]).

Computing the Hermite normal form.

Our algorithm is based on a well-known algorithm combined with modular computation.

It is not hard to see that the straightforward computation of Hermite normal form modulo the determinant may fail in some cases, therefore a more delicate technique is required. The technique is based on the following preliminary observation.

We first observe that Theorem 1 can be, easily, generalized as follows.

**Theorem 2.** Let $M$ be an $m \times n$ ($m \geq n$) integral matrix of rank $n$. There exists a unique unimodular matrix $U$ such that the matrix $UM$ is upper triangular, and the matrix consisting of the first $n$ rows of $UM$ is in Hermite normal form.

$UM$ is called the Hermite normal form of $M$.

The following notation will be used in the sequel. Let $M$ be an $n \times n$ non-singular matrix, $d = |\text{det}M|$. A $2n \times n$ matrix $M'$ is obtained by appending to $M$ the $n$ rows of $dI_n$, where $I_n$ denotes the $n \times n$ unit matrix. I.e.,

$$
\begin{bmatrix}
    m_{1,1} & m_{1,2} & \cdots & m_{1,n} \\
    \vdots & \vdots & \ddots & \vdots \\
    \vdots & \vdots & \ddots & \vdots \\
    m_{n,1} & m_{n,2} & \cdots & m_{n,n}
\end{bmatrix}
$$

if $M = \begin{bmatrix}
    m_{1,1} & m_{1,2} & \cdots & m_{1,n} \\
    \vdots & \vdots & \ddots & \vdots \\
    \vdots & \vdots & \ddots & \vdots \\
    m_{n,1} & m_{n,2} & \cdots & m_{n,n}
\end{bmatrix}$, then $M' = \begin{bmatrix}
    m_{1,1} & m_{1,2} & \cdots & m_{1,n} \\
    \vdots & \vdots & \ddots & \vdots \\
    \vdots & \vdots & \ddots & \vdots \\
    m_{n,1} & m_{n,2} & \cdots & m_{n,n} \\
    d & 0 & \cdots & 0 \\
    0 & d & \cdots & 0 \\
    \vdots & \vdots & \ddots & \vdots \\
    0 & 0 & \cdots & d
\end{bmatrix}$

**Proposition 1.** Let $M$ and $M'$ be as above, and let $N'$ be the Hermite normal form of $M'$. Then a sub-matrix of $N'$ consisting of its first $n$ rows is the Hermite normal form of $M$. 
Proof. For a matrix $A$ let $L(A)$ denote the lattice generated by the rows of $A$. Let $N$ denote the matrix consisting of the first $n$ rows of $N'$. It suffices to show that

(1) \[ L(M) = L(N). \]

Since the matrix $dM^{-1}$ has integral entries and $(dM^{-1})M = dl_n$, all the rows of $M'$ belong to $L(M)$. Hence

(2) \[ L(M) = L(M'). \]

$N'$ has been obtained from $M'$ by a unimodular transformation, therefore

(3) \[ L(M') = L(N'). \]

Finally, the last $n$ rows of $N$ are zero. This implies that

(4) \[ L(N') = L(N). \]

Now (1) follows from (2), (3) and (4). \qed

In view of Proposition 1, instead of computing the Hermite normal form of $M$ we compute the Hermite normal form of $M'$. The algorithm for computing the Hermite normal form of $M'$ is based on the fact that

Reducing an entry modulo $d$ corresponds to the unimodular transformation of subtracting a multiple of an appropriate row of $dl_n$, form the row containing that entry.

Therefore all the arithmetical operations can be performed modulo $d = |\det M|$.

Transforming $M$ into Hermite normal form consists of two major stages:

1. Computing $d = |\det M|$.
2. Transforming $M'$ into Hermite normal form.

Stage 1: Computing $d = |\det M|$.

$d = |\det M|$ can be computed in

(5) \[ O(n^3) \]

arithmetical operations such that the intermediate results of the computation are bounded by $n^a|M|^{2n}$, cf. Appendix.
If $\det M = \pm 1$, then the Hermite normal form of $M$ is the unit matrix $I_n$, and the algorithm terminates here.

Stage 2: Transforming $M'$ into Hermite normal form.

To transform $M'$ into Hermite normal form we use "Gaussian elimination for integral matrices", where all arithmetical operations are performed modulo $d$.

Reducing entries in the first $n$ rows of $M'$ modulo $d$ by unimodular transformations of subtracting multiples of the last $n$ rows, if necessary, we may assume that all the entries in the first $n$ rows of $M'$ are non-negative and less than $d$.

Using unimodular transformations we construct a sequence of matrices $M' = N_0, N_1, \ldots, N_n = N'$ such that for $k = 1, 2, \ldots, n-1$

1. The entries of $N_k$ are non-negative and $|N_k| \leq d$,

2. The last $n-k$ rows of $N_k$ are equal to the last $n-k$ rows of $dI_n$, respectively,

3. The matrix consisting of the first $k$ columns of $N_k$ is in Hermite normal form, and

4. $N_n (= N')$ is the Hermite normal form of $M'$.

I.e., $N_k = (a_{i,j}^k)$ has the following shape:

$$
N_k = \begin{bmatrix}
  a_{1,1}^k & a_{1,2}^k & \ldots & a_{1,k}^k & a_{1,k+1}^k & \ldots & a_{1,n}^k \\
  0 & a_{2,2}^k & \ldots & a_{2,k}^k & a_{2,k+1}^k & \ldots & a_{2,n}^k \\
  \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
  \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & \ldots & a_{k,k}^k & a_{k,k+1}^k & \ldots & a_{k,n}^k \\
  0 & 0 & \ldots & 0 & a_{k+1,k}^k & \ldots & a_{k+1,n}^k \\
  \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
  \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & \ldots & 0 & a_{n+1,k}^k & \ldots & a_{n+1,n}^k \\
  0 & 0 & \ldots & 0 & d & \ldots & 0 \\
  \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
  \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & \ldots & 0 & 0 & \ldots & d 
\end{bmatrix}
$$
\(N_{k+1}\) is constructed from \(N_k\) by annihilating the entries in the \((k+1)st\) column of \(N_k\), from the \((k+2)nd\) entry down to the \((n+k+1)st\) one, using unimodular transformations on the last \(2n - k\) rows of \(N_k\), and then "normalizing" the first \(k\) entries in the \((k+1)st\) column using the \((k+1)st\) row. By Condition 2, the last \(n - k - 1\) entries in the \((k+1)st\) column are already zero and do not have to be annihilated. Therefore the last \(n - k - 1\) rows can be used for reducing modulo \(d\), cf. the observation above. In order to annihilate the \(lth\) entry in \((k+1)st\) column, \(l > k+1\), we proceed in a usual manner.

Since \(\text{rank} M' = n\) and all the transformations are unimodular, \(\text{rank} N_k = n\). Therefore, permuting the rows of \(N_k\), if necessary, we may assume that \(a_{k+1,k+1}^k\neq 0\). Now,

a) Compute the greatest common divisor of \(a_{k+1,k+1}^k\) and \(a_{k+1,k+1}^l\), call it \(r\), and integers \(p\) and \(q\) such that

\[pa_{k+1,k+1}^k + qa_{k+1,k+1}^l = r.\]

b) Define

\[Z = \begin{pmatrix}
p & q \\
-a_{k+1,k+1}^k & a_{k+1,k+1}^l \\
r & r
\end{pmatrix}.\]

Let \(R_{k+1}^l\) and \(R_l^l\) denote the \((k+1)st\) and \(lth\) rows of \(N_k\), respectively. Compute

\[\begin{pmatrix}
R_{k+1}^l \\
R_l^l
\end{pmatrix} = Z \begin{pmatrix}
R_{k+1}^l \\
R_l^l
\end{pmatrix}.\]

Obviously, the first \(k+1\) components of \(R_l^l\) are zero.

Now, substitute \(R_{k+1}^l\) and \(R_l^l\) for \(R_{k+1}^l\) and \(R_l^l\), respectively.

Since

\[\det Z = \frac{pa_{k+1,k+1}^k + qa_{k+1,k+1}^l}{r} = \frac{r}{r} = 1,\]

the above computation corresponds to pre-multiplying by a unimodular matrix.

c) Reduce the last \(n-(k+1)\) components in the \(lth\) and \((k+1)st\) rows of the resultant matrix modulo \(d\).

As it has been mentioned earlier, such a reduction corresponds to subtracting an appropriate multiple of a row of \(dI_n\) (which is a row of \(N_k\)). Therefore the last \(n-k-1\) entries in the \(lth\) and \((k+1)st\) rows of the resultant matrix are less than \(d\). By Condition 1, \(a_{k+1,k+1}^{k+1} \leq d\). Since \(r \leq a_{k+1,k+1}^{k+1}\), the new entries in the \((k+1)st\) column cannot exceed \(d\).
Using the above transformations a manner we can annihilate all the entries from the \((k+2)\)nd down to the \((n+k)\)th in the \((k+1)\)st column. This will ensure \(N_{k+1}\) that satisfies Conditions 1 and 2.

Let \(B_k = (b_{ij}^k)\) denote the matrix computed so far. The first \(k\) entries in the \((k+1)\)st column of \(B_k\) can be normalized as follows.

Let \(S_i\) and \(S_{k+1}\) denote the \(i\)th and \(k+1\)st rows of \(B_k\), respectively. To obtain \(N_{k+1}\) from \(B_k\) we substitute for row \(S_i\) the row \(S_i - \frac{b_{i,k+1}^k}{b_{k+1,k+1}^k} S_{k+1}\), \(i = 1, 2, ..., k\), where \([x]\) denotes the integral part of \(x\). Again, in the above computation all the results are reduced modulo \(d\). After the above normalization, the matrix consisting of the first \(k+1\) columns of \(N_{k+1}\) is in Hermite normal form, and Condition 3 is satisfied. Obviously, if \(k+1 = n\), Condition 4 is satisfied as well.

In order to count the number of arithmetical operations required to perform Stage 2, we first count the number of operations required to obtain matrix \(N_{k+1}\) from \(N_k\).

In Step a) we consecutively compute the greatest common divisor of \((n+k)\)\(-k = n\) integers which are bounded by \(d\). This computation can be performed in

\[
O((\log d + n))
\]
arithmetical operations, cf. [1].

Since the first \(k\) components of \(R_{k+1}\) and \(R_1\) are zero, implementing Step b), which is the computation of \(R_{k+1}'\) and \(R_1'\) requires \(O(n-k)\) arithmetic operations. Hence repeating Step b) \(n-1\) times requires

\[
O((n-1)(n-k))
\]
arithmetical operations.

In Step c), reducing the results obtained in Step b) modulo \(d\) requires \(O(n-k)\) arithmetical operations, and repeating Step c) \(n-1\) times requires

\[
O((n-1)(n-k))
\]
arithmetical operations.
Since, obviously, normalizing an entry can be performed in $O(n-k)$ arithmetical operations, the first $k$ entries in the $(k+1)$st column can be normalized in

$$(9) \quad O(k(n-k))$$

arithmetical operations.

Summing (6), (7), (8) and (9) we obtain that $N_{k+1}$ can be computed from $N_k$ in

$$(10) \quad O(lgd+n^2)$$

arithmetical operations.

Multiplying (10) by $n-1$ we obtain that the matrix $N'$ can be computed from $M'$ in

$$(11) \quad O(nlgd+n^3)$$

arithmetical operations.

Finally, summing (5) and (11), we obtain that $M$ can be transformed into Hermite normal form in

$$(12) \quad O(nlgd+n^3)$$

arithmetical operations, whereas the intermediate results of computation are bounded by $d^2$.

This completes the description of the algorithm.

We can use now the Hadamard inequality to bound $d$:

$$d = |\det M| \leq n^{\frac{1}{2}} |M|^n.$$  

Hence the number of arithmetical operations required by the algorithm is bounded by

$$(13) \quad O(n^2 lg |M| + n^3),$$

whereas the intermediate results of the computation are bounded by $n^n |M|^{\frac{n}{2}}$.

Remark 1. A more careful calculation shows that the number of arithmetical operations is bounded by

$$n^2 \log_{10} |M| + 8n^3.$$  

Remark 2. Of course, in Stage 2 of the algorithm we can start from the matrix $M''$ which is obtained by appending the last $n-1$ rows of $dl_n$ to $M$. In this case the values of the intermediate results computing $N_1$ from $N_0$ are bounded by $\max(d, m_{1,1}, m_{2,1}, \ldots, m_{n,1})$. 

Let $N$ be Hermite normal form of $M$ as computed above. The unimodular matrix $U$ such that $N = UM$ can be computed as follows. Compute first $dM^{-1}$. This can be done in $O(n^3)$ arithmetic operations such that the intermediate results of the computation are bounded by $n^2 |M|^{2n}$ (cf. Appendix). Then compute

$$U = N M^{-1} = \frac{1}{d} (N \cdot (dM^{-1})).$$

Which also can be done in $O(n^3)$ arithmetical operations. It is easy to verify that the intermediate results in the last computation are bounded by $n^{n+1} |M|^{2n}$.

Appendix.

Here we give a sketch of the algorithm for computing $dM^{-1}$, where $d = \det M$, for a non-singular matrix $M$. The computation is based on simple facts of linear algebra and employs the method of J. Edmonds for reducing the values of intermediate results appearing in the computation (cf. [3]). This algorithm has been applied by J. von zur Gathen and M. Sieveking in [6] for solving systems of linear diophantine equations.

To present the algorithm we need the following notation.

Let $A$ and $B$ be $n \times n$ matrices. By $[A, B]$ we denote $n \times 2n$ matrix whose $j$th column is the $j$th column of $A$, if $j \leq n$ and the $(j-n)$th column of $B$, otherwise. I.e., if

$$A = \begin{pmatrix}
    a_{1,1} & \ldots & a_{1,n} \\
    \vdots & & \vdots \\
    a_{n,1} & \ldots & a_{n,n}
\end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix}
    b_{1,1} & \ldots & b_{1,n} \\
    \vdots & & \vdots \\
    b_{n,1} & \ldots & b_{n,n}
\end{pmatrix},$$

then

$$[A, B] = \begin{pmatrix}
    a_{1,1} & a_{1,n} & b_{1,1} & \ldots & b_{1,n} \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    a_{n,1} & a_{n,n} & b_{n,1} & \ldots & b_{n,n}
\end{pmatrix}.$$

We construct a sequence of $n \times 2n$ integral matrices $[M, I_n] = X_0, X_1, \ldots, X_n = [dI_n, dM^{-1}]$, where $I$ denotes the $n \times n$ identity matrix, such that
1. $X_{k+1}$ is obtained from $X_k$ by row operations, i.e., either by multiplying a row by an integer or adding or permuting the rows,

2. Then for $j \leq k$ all of the the entries in the $j$th column, but $(k,k)$-entry, are zero.

3. $|X_k| \leq n^{\frac{2^n}{n}} |M|^n$.

Let $X_k = (x_{ij}^k)$. Conditions 1 and 2 above imply that $X_k$ is has following shape:

$$
X_k = \begin{bmatrix}
  x_{1,1}^k & 0 & \ldots & 0 & x_{1,k+1}^k & \ldots & x_{1,2n}^k \\
  0 & x_{2,2}^k & \ldots & 0 & x_{2,k+1}^k & \ldots & x_{2,2n}^k \\
  \vdots  & \vdots  & \ddots & \vdots & \vdots  & \ddots & \vdots \\
  \vdots  & \vdots  & \ddots & \vdots & \vdots  & \ddots & \vdots \\
  0 & 0 & \ldots & x_{i,k}^k & x_{i,k+1}^k & \ldots & x_{i,2n}^k \\
  \vdots  & \vdots  & \ddots & \vdots & \vdots  & \ddots & \vdots \\
  0 & 0 & \ldots & 0 & x_{n,k+1}^k & \ldots & x_{n,2n}^k
\end{bmatrix}
$$

Assume that $X_k$ has been constructed. Since $\det M \neq 0$, there exists an $i \geq k+1$ such that $x_{k+1,k+1}^k \neq 0$. Fermuting $i$th and $(k+1)$st rows of $X_k$, if necessary, we may assume that $x_{i,k+1}^k \neq 0$.

Let $R_i$ and $R_{k+1}$ denote the $i$th and $(k+1)$st rows of $X_k$, respectively. In order to annihilate $(i,k+1)$-entry, $i \neq k+1$, we substitute for $R_i$ a new row $R'_i$, where

$$
R'_i = \frac{1}{x_{k+1,k+1}^k} (x_{k+1,k+1}^k R_i - x_{i,k+1}^k R_{k+1}).
$$

Obviously the $(k+1)$st component of $R'_i$ is zero. J. Edmonds proved in [3] that all the entries of $R'_i$ are integers.

Proceeding as above we can annihilate all of the entries, but the $(k+1)$st one, in the $(k+1)$st column of $X_k$. The result, obviously, satisfies Conditions 1 and 2 above.

It has been shown by J. Edmonds in [3] that each entry of the resultant matrix is a determinant of a minor of $X_0$. Therefore, by the Hadamard inequality, $|X_k| \leq n^{\frac{2^n}{n}} |M|^n$, and Condition 3 is also satisfied.

Thus $X_{k+1}$ has been constructed.

It is easy to see that the intermediate results in the computation of $X_{k+1}$ from $X_k$ are bounded
Similarly to [3] one can show that $X_n = [dI_n X]$ for some integral matrix $X$.

Since $X_n$ has been obtained from $X_0$ by row operations, there exists an $n \times n$ matrix $Y$ such that

$$YX_0 = X_n$$

or

$$Y[M J_n] = [dI_n X] = [dI_n Y],$$

which implies that $X = dM^{-1}$ and proves that $dM^{-1}$ can be computed by the above algorithm. This algorithm is the only one known from the literature which involves no computation of the greatest common divisor.

**Remark 3.** Of course, $U$ could be obtained by the above algorithm directly starting from $[M, N]$, but this does not decrease the number of arithmetical operations, and the size of intermediate results in such a computation might be $O(n^2 \lg(n|M|))$, whereas the size of intermediate results produced by the algorithm given at the end of the previous section is $O(n \lg(n|M|))$. 
References


