Journal of Mathematical Extension
Vol. 7, No. 4, (2013), 11-28

Solving Rank One Perturbed Linear Diophantine Systems Using the Hermite Normal Form

M. Khorramizadeh
Shiraz University of Technology

Abstract. We show how we can obtain the general solution of rank one perturbed linear Diophantine systems $(A + uv^T)x = b$ using only information from the application of the Hermite normal form algorithm to the corresponding linear Diophantine system $Ax = b$. The empirical results show that use of the proposed algorithm may result in saving considerable computing time.

AMS Subject Classification: 11D04; 11Y05; 11Y50
Keywords and Phrases: Hermite normal form, free module, linear diophantine system

1. Introduction

Let $A = (a_1, \cdots, a_m)^T$, $a_j \in Z^n$, for $j = 1, \ldots, m$. Consider the following system of linear Diophantine equations:

$$Ax = b, \quad A \in Z^{m\times n},\ x \in Z^n,\ b \in Z^m, \quad m \leqslant n. \tag{1}$$

System (1) can be written as $a_i^T x = b_i$, for $i = 1, \ldots, m$, where a_i^T is the ith row of the coefficient matrix A, and b_i is the ith component of the right hand side vector b. In the following, we assume that $rank(A) = m$. The null space of A is denoted by $N(A)$ and consists of all vectors $x \in R^n$ satisfying $Ax = 0$. If we let $N_z(A) = N(A) \cap Z^n$, then it can be shown that $N_z(A)$ is a free module over Z, with invariant basis number property, meaning that $N_z(A)$ has a linearly independent generating set

Received: March 2013; Accepted: August 2013

and all linearly independent generating sets of $N_z(A)$ have the same cardinality, called the rank of $N_z(A)$. We say that an integer matrix N generates $N_z(A)$, if the the columns of N generate $N_z(A)$. If the columns of N are linearly independent, then N is called a basis for $N_z(A)$. A vector $x_p \in Z^n$ satisfying $a_i^T x_p = b_i$, $1 \leqslant i \leqslant m$, is called a particular solution of (1). Let $v \in Z^n$, $N \in Z^{n,n_1}$, $y \in Z^{n_1}$, $n_1 \in Z$. Note that if v is a particular solution of (1) and N generates $N_z(A)$, then $x = v + Ny$ is the general solution of (1). This means that x_p is a particular solution of (1) if and only if $x_p = v + Ny_p$, for some $y_p \in Z^{n_1}$.

Solving linear Diophantine systems has many applications in mathematics, engineering and computer science such as graph theory, integer programming, design of integrated circuits for radio processing, market split problem, etc. Some algorithms for solving linear Diophantine systems were discussed in Barnette and Pace [6], Blankinship [7, 8], Bradley [9] and Frumkin [18]. The first polynomial time algorithm for computing the Hermite normal form of an integer matrix was given by Kannan and Bachem [22]. Chou and Collins [10] made use of Kannan and Bachem's ideas to present a polynomial time algorithm named LDSMKB for solving systems of linear Diophantine equations. They also developed an algorithm based on Rosser's idea [30], the so-called LDSSBR, and showed numerically that the LDSSBR was more successful than the LDSMKB in controlling the growth of intermediate results. Esmaeili et al. [15] presented a class of algorithms, the so-called EMAS algorithms, for solving systems of linear Diophantine equations based on the ABS algorithms [2, 3]. In 2009, Khorramizadeh and Mahdavi-Amiri [26] proposed a class of algorithms for solving linear Diophantine systems based on an extension of ABS algorithms. Later, Golpar-Raboky and Mahdavi-Amiri [19] used these two classes to propose algorithms for solution of a quadratic Diophantine equation. For a review and discussion of ABS algorithms and their extension for linear Diophantine systems see, [5, 17, 33, 34].

After solving a linear Diophantine system $Ax = b$, a natural mathematical question is whether we could use the information from the solution of the original system to derive an expression for all solutions of the rank one perturbed system. Moreover, changing the coefficient of one variable or deleting a variable or a constraint corresponds to solving a rank one perturbed linear Diophantine system of the form $(A + uv^T)x = b$, which may relate to sensitivity analysis. in integer programming. Not knowing how to use the information obtained after solving $Ax = b$ to obtain the general solution of $(A + uv^T)x = b$, we need to solve a different linear Diophantine system from scratch which might be time consuming and costly. Therefore, algorithms for obtaining the general solution of $(A + uv^T)x = b$ using information obtained after solving $Ax = b$ are worth studying and may result in saving considerable computing time. In 2008, Amini and Mahdavi-Amiri [1] proposed an algorithm for solving rank one perturbed linear Diophantine systems using ABS methods. Khorramizadeh and Mahdavi-Amiri [24] used the Rosser's idea [30] and extended integer ABS algorithms [25] to propose an algorithm for rank one perturbed linear Diophantine systems and showed the algorithm to be more efficient than that of Amini and Mahdavi-Amiri [1]. Algorithms in [1] and [24] are respectively based on the integer ABS and Rosser's approach for solving linear Diophantine systems. However, several efficient algorithms for solving linear Diophantine systems are based on the Hermite normal form [21, 27, 28, 31]. Therefore, here we discuss an algorithm for solving rank one perturbed linear Diophantine systems based on the Hermite normal form algorithm for solving Diophantine systems.

1.1 Hermite Normal Form Algorithm

One efficient approach for solving linear Diophantine systems is to use the Hermite normal form algorithm [8, 10, 14, 20]. An $m \times m$ integer matrix H with full row rank is said to be in a Hermite normal form if it is lower triangular nonsingular nonnegative integer matrix with every diagonal element being the largest in its corresponding row [21]. Let A be an $m \times n$ integer matrix and $rank(A) = r$. The goal of the Hermite normal form algorithm is to find a unimodular matrix U such that the integer matrix HNF(A)=AU has the Hermite normal form. An integer square matrix with a determinant of +1 or -1 is called unimodular. Algorithms for computing the Hermite normal form are based on the extended greatest common divisor (GCD) algorithms, i.e., algorithms for computing the GCD of two or more integers. An extended GCD algorithm is applied to an integer vector $a = [a_1, a_2 \dots, a_k]^T$ and computes a unimodular matrix $[p, U]$ such that $a^T[p, U] = [\delta, 0, \dots, 0]$, where δ is the greatest common divisor of $a_1, a_2 \dots, a_k$, $a^T p = \delta$ and U is a basis for the null space of a. Now, assume that A has full row rank and we have a GCD algorithm. Then, we can use the following procedure to compute the Hermite normal form of an integer matrix $A = (a_1, a_2 \dots, a_m)^T$, where a_i is the ith row of A [15, 16, 25]

Algorrithm 1.2. Hermite normal form algorithm.

Step 1. Set $i = 1$, $U = I_n$.

Step 2. Compute $s_i = U^T a_i$ (since A has full row rank, then $s_i \neq 0$). Use the extended GCD algorithm to compute $p^{(i)}$ and $U^{(i)}$ so that $s_i^T[p^{(i)}, U^{(i)}] = [\delta_i, 0, \dots, 0]$.

Step 3. Set $U = UU^{(i)}$.

Step 4. If $i = m$ then stop. Else set $i = i + 1$ and go to Step 2.

Step 5. Let

$$L = \begin{pmatrix} a_1^T p^{(1)} & & & \\ a_2^T p^{(1)} & a_2^T U^{(1)} p^{(2)} & \cdots & \\ \vdots & \vdots & \ddots & \\ a_m^T p^{(1)} & a_m^T U^{(1)} p^{(2)} & \cdots & a_m^T U^{(1)} \cdots U^{(m-1)} p^{(m)} \end{pmatrix}.$$

Step 6. For $i = 2 \dots, m$ do

For $j = 1 \dots, i-1$ do

$$L_j \longleftarrow L_j - \lfloor \frac{L_{ij}}{L_{ii}} \rfloor L_i,$$

where L_j denotes the jth column of L.

According to Algorithm 1, for different choices of the extended GCD algorithm in Step 2, we obtain different algorithms for computing the Hermite normal form. On the other hand, several algorithms have been proposed for computing the greatest common divisor of two or more integers, such as extended Euclidean algorithm [13, 29], Rosser's algorithm [15, 30] and algorithms based on the LLL-reduction method [20]. In 2001, Esmaeili et al. [15] showed that the Hermite normal form of an integer matrix can also be computed implicitly by using the class of ABS methods specialized for solving linear Diophantine systems. Note that we can easily diagonalize the nonzero part of the Hermite normal form of an integer matrix by performing some elementary column operations on its columns [32]. Moreover, the computations can be done so that the diagonal elements of the resulting matrix, i.e., $d_1, d_2 \ldots, d_m$, are positive an $d_1|d_2 \ldots |d_m$. The resulting matrix is called the Smith normal form of the integer matrix.

Now, let C be the submatrix of HNF(A) formed by the r nonzero columns of HNF(A). Then, we have $U = [Q, N]$, $AQ = C$, $AN = 0$ and the columns of N form a basis for the null space of A. The linear Diophantine system $Ax = b$ is equivalent to $AU(U^{-1})x = b$, and hence, $[C|0]z = b$, where $z = U^{-1}x$ is an integer vector if and only if x is an integer vector. Therefore, we can use the Hermite normal form algorithm to reduce an original linear Diophantine system (1) and compute its general solution effectively.

Here, we propose an efficient algorithm for solving rank one perturbed linear Diophantine systems, using information from the solution of the original system by using the Hermite normal form algorithm. We also present some numerical results to illustrate the efficiency of our proposed algorithm.

In Section 2, we propose an efficient algorithm for rank one perturbed linear Diophantine systems. Section 3 is devoted to empirical results. Section 4 gives the concluding remarks.

2. Rank One Perturbed Systems

Consider the following rank one perturbed linear Diophantine system:

$$(A + uv^T)x = b, \quad v, x \in Z^n,\ A \in Z^{m\times n},\ u, b \in Z^m,\ m \leqslant n. \tag{2}$$

Assuming that (1) has integer solutions and A has full row rank. In this section, we show how we can make use of the information obtained after an application of Algorithm 1 for solving the linear Diophantine system (1), to efficiently compute the general solution of the corresponding rank one perturbed linear Diophantine system (2). It can be easily verified that every integer solution of (2) is an integer solution of

$$\begin{aligned} Ax &= b - ut, && (3)\\ v^T x &= t, && (4) \end{aligned}$$

for some $t \in Z$. For $1 \leqslant k \leqslant m$, let $A_k = (a_1, a_2 \dots, a_k)$, $b^k = (b_1, b_2 \dots, b_k)^T$ and $u^k = (u_1, u_2 \dots, u_k)^T$. If we apply Algorithm 1 to A_k^T, then we obtain the unimodular matrix $U_k = [Q_k, N_k]$ so that $A_k^T Q_k = C_k$ and $A_k^T N_k = 0$, where C_k is the nonzero part of the Hermite normal form of A_k^T. Using the unimodularity of U_k, the following theorem can be easily verified.

Theorem 2.1. *For $k = 1, \dots, m$ and $w \in Z^k$, $A_k^T x = w$ has an integer solution if and only if $C_k y = w$ has an integer solution.*

Now, consider the module $S_0^k = \{(y,t) : C_k y + u^k t = 0\} \cap Z^{n+1}$ and the projection $P : Z^{k+1} \to Z$, defined by $P(y,t) = t$. Then, the set of all (y,t) satisfying $C_k y = b^k - u^k t$ is the translated module $S_b^k = S_0^k + b^k$. Therefore, $P(S_b^k)$ is a translated module T^k inside Z. If we assume that $C_k y = b^k$ has an integer solution, then 0 belongs to T^k, and so T^k must be the set of multiples of a particular positive integer q_k. In the following, we first determine q_k, for $k = 1, \dots, m$. Then, we characterize the general solution of (3) and use (4) to characterize the general solution of (2). Let c_{ij}^k be the element in the ith row and jth column of C_k. Now, if we let

$$\begin{aligned} z_1^k &= -(\frac{u_1}{c_{11}^k}), && (5)\\ z_i^k &= -(\frac{u_i + \sum_{j=1}^{i-1} c_{ij}^k z_j^k}{c_{ii}^k}), \quad 2 \leqslant i \leqslant k, && (6) \end{aligned}$$

then it is straightforward to prove the following theorem.

Theorem 2.2. *Let t be integer and $C_k y = b^k$ have an integer solution. Then, for $k = 1, \dots, m$, $C_k y = b^k - u^k t$ have an integer solution if and only if $z_i^k t \in Z$, for $i = 1, \dots, k$, where z_i^k is defined by (5) and (6).*

Let q_k be the least positive integer satisfying $q_k z_i^k \in Z$, for $i = 1, \ldots, k$. Clearly, we have $tz_i^k \in Z$, for $i = 1, \ldots, k$, if and only if $q_k | t$. Therefore, the following corollary immediately follows from Theorem 2.2.

Corollary 2.3. *Let t be integer and $C_k y = b^k$ have an integer solution. Then, for $k = 1, \ldots, m$, $C_k y = b^k - u^k t$ has an integer solution if and only if $q_k | t$.*

Corollary 2.3 asserts that $T^k = \{t' q_k : t' \in Z\}$. From Theorem 2.1 and Corollary 2.3, we have the following result.

Theorem 2.4. *Let t be integer and $A_k^T y = b^k$ have an integer solution. Then, for $k = 1, \ldots, m$, $A_k^T x = b^k - u^k t$ has an integer solution if and only if $t \in T^k$.*

Now, if we let $x_k + N_k y$, with $y \in Z^{(n-k)}$, be the general solution of $A_k^T x = b^k - u^k t$, $z^k = (z_1 \ldots, z_k)^T$ and $g^k = Q_k z^k$, then for $t \in T^k$, the general solution of $A_k^T x = b^k - u^k t$ is $x_k + g^k t + N_k y$, $y \in Z^{(n-k)}$. Specifically, for $k = m$ and $t \in T_m$, the general solution of (2) is $x_m + g^m t + N_m y$, $y \in Z^{(n-m)}$. Note that g^m can be computed, using only the information obtained after an application of the Algorithm 1 to solve $Ax = b$. In the following, for notational simplicity, we let $g = g^m$, $q = q_m$, $x = x_m$ and $N = N_m$.

Theorem 2.5. *Suppose that the linear Diophantine system $(A+uv^T)x = b$ has an integer solution. Then, the following linear Diophantine equation has an integer solution:*

$$[v^T N, (v^T g - 1)q] \begin{bmatrix} y \\ t \end{bmatrix} = -v^T x. \qquad (7)$$

Proof. Suppose that $(A+uv^T)x = b$ has an integer solution and $x_p \in Z^n$ satisfies $(A + uv^T)x_p = b$. Let $t_p = v^T x_p$. Then, $Ax = b - ut_p$ has an integer solution. Since, by assumption, the linear Diophantine system $Ax = b$ has an integer solution, then $Ax = -ut_p$ has an integer solution. Therefore, by Theorem 2.2, we must have $q \mid t_p$, i.e., $t_p = qt'_p$, for some $t'_p \in Z$. Since the general solution of $Ax = b - ut_p$ is $x + gqt'_p + Ny$, $y \in Z^{n-m}$, there exists $y_p \in Z^{n-m}$ such that $x_p = x + gqt'_p + Ny_p$. Since $t_p = v^T x_p$, we have

$$qt'_p = t_p = v^T x_p = v^T x + v^T gqt'_p + v^T N y_p.$$

From the last equality, it follows that the integer vector $[y_p^T, t_p']^T$ satisfies (7). $\square$

Now, if $v^T N = 0$, then (7) becomes $(v^T g - 1)qt = -v^T x$. If $(v^T g - 1) = 0$ and $v^T x \neq 0$, then (7) has no integer solution, and hence, by Theorem 2.5, we conclude that (2) has no integer solution. If $(v^T g - 1) = 0$ and $v^T x = 0$, then (7) holds for every value of y and t. In this case, the general solution of (2) is:

$$x + [qg, N] \begin{bmatrix} t \\ y \end{bmatrix}, \quad t \in Z, \quad y \in Z^{n-m}.$$

If $(v^T g - 1) \neq 0$ and $(v^T g - 1)q \not| v^T x$, then (7) has no integer solution, and by Theorem 2.5, (2) has no integer solution. If $(v^T p - 1) \neq 0$ and $(v^T g - 1)q | v^T x$, then the general solution of (2) is:

$$x + (\frac{v^T x}{(v^T g - 1)q})g + Ny, \quad y \in Z^{n-m}.$$

Now, consider the case $v^T N \neq 0$. In this case, we have the following theorem to characterize the general solution of (2).

Theorem 2.6. *Let $rank(A) = m$ and N be an integer basis for the integer null space of A. If $v^T N \neq 0$, then $rank(A + uv^T) = m$.*

Proof. To prove the theorem, it is sufficient to prove that the rows of $(A + uv^T)$ are linearly independent. First, note that since $a_i^T N = 0$, for $1 \leqslant i \leqslant m$, and $v^T N \neq 0$, then $a_i + u_i v \neq 0$, for $1 \leqslant i \leqslant m$, and thus the rows of $(A + uv^T)$ are nonzero vectors. Now, suppose that d_i, $1 \leqslant i \leqslant m$, not all being zero, be so that $\sum_{i=1}^m d_i(a_i + u_i v) = 0$. Then, we can write

$$\sum_{i=1}^m d_i(a_i + u_i v) = 0 \Rightarrow \sum_{i=1}^m d_i a_i + (\sum_{i=1}^m d_i u_i)v = 0.$$

Since $v^T N \neq 0$, v can not be written as a linear combination of the a_i, $1 \leqslant i \leqslant m$. Therefore, we have $\sum_{i=1}^m d_i u_i = 0$. So, we have $\sum_{i=1}^m d_i a_i = 0$. Since $rank(A) = m$, a_i, $1 \leqslant i \leqslant m$, are linearly independent. So, from $\sum_{i=1}^m d_i a_i = 0$, we conclude that $d_i = 0$, $1 \leqslant i \leqslant m$. Therefore, $a_i + u_i v$, $1 \leqslant i \leqslant m$, are linearly independent and the proof is complete. $\square$

The following theorem characterizes the general solution of (2), when $v^T N \neq 0$ and (7) has an integer solution. Note that if (7) has no integer solution, then (2) has no integer solution, by Theorem 2.5.

Theorem 2.7. *Suppose that* $rank(A) = m$, $v^T N \neq 0$. *The linear Diophantine equation (7) has an integer solution and its general solution, obtained by an application of the Algorithm 1.2, is:*

$$\begin{bmatrix} y \\ t \end{bmatrix} = \begin{bmatrix} v_y \\ v_t \end{bmatrix} + \begin{bmatrix} U_y \\ U_t \end{bmatrix} \begin{bmatrix} \widehat{y} \\ \widehat{t} \end{bmatrix},$$

where $v_y \in Z^{n-m}, v_t \in Z, U_y \in Z^{(n-m)\times(n-m)}, U_t \in Z^{1\times(n-m)}, \widehat{y} \in Z^{(n-m-1)}, \widehat{t} \in Z$. *Then, the general solution of* $(A + uv^T)x = b$ *is* $\overline{x} + \overline{N}y,\ \ y \in Z^{n-m}$, *where* $\overline{N} = NU_y + qgU_t$, $\overline{x} = p + Nv_y + qgv_t$. *Moreover,* $\overline{N}$ *is a basis for* $N_z(A + uv^T)$.

Proof. It suffices to show that x_p is a particular solution of (2) if and only if $\overline{x}_p = \overline{x} + \overline{N}\overline{y}$, for some $y \in Z^{n-m}$. Let x_p be so that $(A + uv^T)x_p = b$. In the proof of Theorem 2.5, we showed that in this case there exist some $y_p \in Z^{n-m}$ and $t'_p \in Z$ so that $x_p = p + gqt'_p + Ny_p$ and $[y_p^T, t'_p]^T$ satisfies (7). So, by definition of general solution, there exist $\widehat{y}_p \in Z^{n-m-1}$ and $\widehat{t}_p \in Z$ so that

$$\begin{bmatrix} y_p \\ t'_p \end{bmatrix} = \begin{bmatrix} v_y \\ v_t \end{bmatrix} + \begin{bmatrix} U_y \\ U_t \end{bmatrix} \begin{bmatrix} \widehat{y}_p \\ \widehat{t}_p \end{bmatrix}.$$

Therefore, we can write

$$\begin{aligned} x_p &= p + [gq, N] \begin{bmatrix} y_p \\ t'_p \end{bmatrix} \\ &= p + Nv_y + qgv_t + (NU_y + qgU_t) \begin{bmatrix} \widehat{y}_p \\ \widehat{t}_p \end{bmatrix} \\ &= \overline{x} + \overline{N} \begin{bmatrix} \widehat{y}_p \\ \widehat{t}_p \end{bmatrix} = \overline{x} + \overline{N}\overline{y}, \end{aligned}$$

where $\overline{y} = [\widehat{y}_p^T, \widehat{t}_p]^T \in Z^{n-m}$. Conversely, suppose that $x_p = \overline{x} + \overline{N}\overline{y}$, for some $\overline{y} \in Z^{n-m}$. We have

$$A\overline{x} = A(p + Nv_y + qgv_t) = Ap + qv_tAg = b - qv_tu.$$

On the other hand, we have

$$[v^TN, (v^Tg-1)q]\begin{bmatrix} v_y \\ v_t \end{bmatrix} = -v^Tp \Rightarrow v^T\overline{x} = qv_t.$$

This shows that $(A+uv^T)\overline{x} = b$. Similarly, it can be easily verified that $(A+uv^T)\overline{N} = 0$. Thus, we have

$$(A+uv^T)x_p = (A+uv^T)(\overline{x}+\overline{N}\overline{y}) = b.$$

It remains to show that $\overline{N}$ is a basis for $N_z(A+uv^T)$. From the above argument, it follows that the columns of $\overline{N}$ generate $N_z(A+uv^T)$. Because, if $x' \in Z^n$ satisfies $(A+uv^T)x' = 0$, then since $\overline{x}+x'$ is a particular solution of (2), there exists $\overline{y} \in Z^{n-m}$ so that $\overline{x}+x' = \overline{x}+\overline{N}\overline{y}$, and hence $x' = \overline{N}\overline{y}$. On the other hand, by Theorem 2.7, we have $rank(A+uv^T) = m$. Therefore, every basis for $N_z(A+uv^T)$ has $n-m$ linearly independent vectors. So, it suffices to show that the columns of $\overline{N}$ are linearly independent. Indeed, since $A(qg) = -qu \neq 0$, then columns of the integer matrix $[N, qg]$ are linearly independent. Clearly, the columns of $[U_y^T, U_t^T]$ are also linearly independent. Therefore, the columns of the integer product matrix $\overline{N} = [NU_y + qgU_t]$ are linearly independent. This completes the proof. □

Remark 2.8. *To control the growth of intermediate results, we can reduce the size of the integer vector qg by using short columns of the null space basis, N. Let n_j be the jth column of N. Then, the following procedure can be used to reduce the size of the integer vector w, by using the columns of N:*

$$For\ i = 1\ to\ n-m\ do\ w = w - \lfloor (w^Tn_j)/(n_j^Tn_j) \rfloor n_j.$$

So, we can write the following algorithm for solving (2), using integer matrices N, Q and C and the integer vector x, which are obtained by an application of the Algorithm 1 to (1).

Algorithm 2.9. Rank One Perturbed Solver Based on Hermite Normal Form.

Step 1. (Compute z) Set $z_1 = -u_1/c_{11}$;

```
    For   $i = 2$ to $m$ do
          begin
          sum=0;
            For   $j = 1$ to $i - 1$
                  $sum = sum + c_{ij}z_j$;
          $z_i = -(u_i + sum)/c_{ii}$
          end;
  Set $z = (z_1, \cdots, z_m)^T$ and $g = Qz$.
```

Step 2. Find the smallest positive integer q such that $qg \in Z^n$ and set $w = qg$.

Step 3. (Reduce the size of w)
For $i = 1$ to $n - m$ do $w = w - \lfloor (w^T n_i)/(n_i^T n_i) \rfloor n_i$.

Step 4. If $v^T N = 0$ then

(a) If $v^T w - q = 0$ and $v^T x \neq 0$ then stop: $\{(A + uv^T)x = b$ has no integer solution$\}$.

(b) If $v^T w - q = 0$ and $v^T x = 0$ then set $\overline{x} = x$, $\overline{N} = [w, N]$ and go to step 6.

(c) If $(v^T w - q) \neq 0$ and $(v^T w - q) \not| v^T p$ then stop $\{(A + uv^T)x = b$ has no integer solution$\}$.

(d) If $(v^T w - q) \neq 0$ and $(v^T w - q) | v^T x$ then set

$$\overline{x} = x + (\frac{v^T x}{(v^T w - q})qg, \ \overline{N} = N,$$

and go to step 6.

Step 5. Solve the following single linear Diophantine equation, by using Algorithm 2.1:

$$[v^T N, (v^T w - q] \begin{bmatrix} y \\ t \end{bmatrix} = -v^T x; \tag{8}$$

If (8) has no integer solution then stop {the Diophantine system $(A + uv^T)x = b$ has no integer solution} else let the general integer solution of (8) be

$$\begin{bmatrix} y \\ t \end{bmatrix} = \begin{bmatrix} v_y \\ v_t \end{bmatrix} + \begin{bmatrix} U_y \\ U_t \end{bmatrix} \begin{bmatrix} \widehat{y} \\ \widehat{t} \end{bmatrix},$$

where $v_y \in Z^{n-m}, v_t \in Z, U_y \in Z^{n-m \times n-m}, U_t \in Z^{1 \times n-m}, \widehat{y} \in Z^{n-m-1}, \widehat{t} \in Z$; Set $\overline{x} + \overline{N} y'$, where

$$\overline{N} = NU_y + wU_t, \quad \overline{x} = x + Nv_y + wv_t.$$

Step 6. Stop { $\overline{x}$ is a particular solution and $\overline{N}$ is a basis for the null space of $(A + uv^T)$}.

3. Numerical Results

Here, we investigate the practical efficiency of the proposed algorithm for solving rank one perturbed linear Diophantine systems, based on the Hermite normal form algorithm.
We applied Algorithm 2. to some randomly generated consistent linear Diophantine systems. To generate consistent random systems, we used the approach proposed by Chou and Collins [10]. We let the bit length of the coefficients of the generated systems to be 40. Then, given the number of variables and the number of constraints, we generated sample linear Diophantine systems until a consistent one was found and applied the algorithms to the generated system.
After generating A, b, u and v, we first solved the linear Diophantine system $Ax = b$, by using the Hermite normal form algorithm proposed by Havas et al. [20] for solving linear diophantine systems. Then, we used the obtained information to solve the corresponding linear Diophantine

system $(A+uv^T)x = b$. To compare our proposed algorithm with the one proposed in [24], we also solved $Ax = b$ by using the Rosser's approach and used the resulting information to compute the general solution of $(A + uv^T)x = b$. To show the efficiency of the proposed algorithm, we also solved $(A + uv^T)x = b$ from scratch by using the the Algorithm 1 and compared the results in tables 1-4.

In these tables, $B1$, $B2$ and $Time$ denote the bit length of the absolute value of the maximum of the components of the particular solution, the bit length of the absolute value of the maximum of the components of the basis and the computing time of the algorithm (in seconds), respectively. Moreover, the data below the column, named *LLL-based* are related to the Algorithm 2 here, the data below the column, named *From scratch* are related to the algorithm solving the system $(A + uv^T)x = b$ from scratch by using the the Algorithm 1 and the data below the column, named *Rosser–based* are related to the algorithm based on Rosser's approach.

The results in these tables show that making use of our proposed algorithm results in saving computing time, significantly. For example, when $n = 70$ and $m = 69$, by solving $(A + uv^T)x = b$ from scratch, we obtained the general solution of the system in 993.453 seconds, while the same particular solution and basis was obtained by our proposed algorithm in 0.859 seconds. When $n - m = 1$, for our generated random systems, the particular solution and the basis obtained by using the proposed algorithm were very close to the ones obtained by solving the rank one perturbed system from scratch. When $n - m > 1$, by solving the rank one perturbed system from scratch, we always obtained particular solutions and bases, with slightly smaller B_1 and B_2. But, the needed computing time for solving from scratch was considerably larger than the one for the proposed algorithm. This shows that the need computing cost for our proposed algorithm is significantly smaller than that of solving the problem from scratch. In tables 1–4, we have also compared our proposed algorithm with the Rosser–based algorithm, proposed in [24]. The numerical results in these tables show that although when $n - m = 1$, the performance of two algorithms was almost the same, but when $n - m > 1$, our proposed algorithm was significantly more efficient than the Rosser–based algorithm. Moreover, in Table 1, when

$n - m = 1$, for both algorithms the values corresponding to $B1$ and $B2$ and the computing times were very close. When $n - m > 1$, the values corresponding to $B1$ and $B2$ of our proposed algorithm were significantly smaller than those of the Rosser–based algorithm.

To compare the computing cost of our proposed algorithm with that of the Rosser–based algorithm, it is sufficient to compare the Rosser–based algorithm with the LLL–based algorithm for solving linear Diophantine systems. In [23], we presented a comparison of these algorithms and conclude that in the sense of controlling the growth of intermediate results ($B1$ and $B2$) and the computing time for small problems, the Rosser–based algorithm is more efficient, while for large problems, the LLL–based algorithm is more efficient. However, for all test problems the bit length of the maximum absolute value of the components of the particular solution and basis, obtained by using the LLL–based algorithm, is significantly less than that of the the Rosser–based algorithm; See [23], for details.

Table 1: $n - m = 1$

	LLL-based			From scratch			Rosser–based		
n	B_1	B_2	Time	B_1	B_2	Time	B_1	B_2	Time
30	1222	1223	0.297	1222	1223	28.26	1223	1227	0.328
35	1431	1435	0.329	1431	1435	49.67	1431	1435	0.375
40	1639	1642	0.375	1640	1641	86.484	1641	1643	0.500
45	1847	1850	0.438	1847	1850	150.047	1847	1850	0.438
50	2056	2059	0.485	2053	2058	212.547	2053	2058	0.750
55	2264	2266	0.516	2270	2271	334.829	2268	2271	1.078
60	2474	2477	0.641	2474	2477	486.406	2474	2477	1.422
65	2688	2689	0.640	2688	2689	699.094	2688	2689	2.078
70	2901	2902	0.859	2898	2900	993.453	2902	2903	3.437

Table 2: $n - m = 5$

	LLL-based			From scratch			Rosser–based		
n	B_1	B_2	Time	B_1	B_2	Time	B_1	B_2	Time
30	262	264	0.234	212	213	22.453	5361	5363	0.344
35	312	313	0.282	254	255	40.36	8404	8406	0.453
40	361	362	0.36	295	296	70.219	12969	12970	0.656
45	412	412	0.39	337	338	116.015	19494	19495	0.969
50	462	464	0.406	378	380	183.5	29875	29880	1.469
55	512	512	0.469	420	422	277.328	44146	44151	2.438
60	563	564	0.531	463	464	413.047	67454	67456	4.031
65	613	615	0.562	505	506	586.86	100385	100387	7.953
70	662	665	0.579	547	548	1647.3	148656	148656	16.235

Table 3: $n - m = 10$

	LLL-based			From scratch			Rosser–based		
n	B_1	B_2	Time	B_1	B_2	Time	B_1	B_2	Time
30	96	122	0.485	86b	87	19	3118	3121	0.359
35	118	142	0.469	106	107	36.453	5309	5311	0.484
40	142	165	0.578	127	129	59.828	8409	8410	0.719
45	164	185	0.609	148	149	102.141	13332	13335	1.110
50	188	206	0.688	169	170	165.406	20492	20495	1.781
55	212	227	0.781	190	190	256.562	31034	31036	3.109
60	234	248	0.828	211	212	365.907	47446	47450	5.718
65	258	269	0.937	232	233	1099.2	71956	71956	11.25
70	280	294	1.062	254	254	2326.84	108007	108012	22.922

Table 4: $n - m = 20$

	LLL-based			From scratch			Rosser–based		
n	B_1	B_2	Time	B_1	B_2	Time	B_1	B_2	Time
30	25	61	1.047	23	23	25.906	807	809	0.265
35	35	70	1.235	33	34	50.14	1682	1683	0.407
40	46	81	1.438	43	44	89	2975	2978	0.609
45	57	92	1.704	54	55	152.048	4987	4989	0.891
50	69	103	1.688	64	65	125.25	8071	8078	1.359
55	79	113	1.813	75	75	400.532	12917	12918	2.375
60	90	123	2.063	85	86	598.907	20010	20010	3.734
65	101	134	2.500	95	96	868.343	30638	30641	6.438
70	112	144	2.406	106	106	608.5	46961	46964	11.563

4. Conclusion

We proposed an efficient algorithm for solving rank one perturbed linear Diophantine systems, based on the Hermite normal form algorithm for solving linear Diophantine systems. Then, by presenting some numerical results, we examined the practical efficiency of the proposed algorithm. The numerical results show that by using the proposed algorithm we may reduce the the computing time significantly.

Acknowledgments
The author thanks the research council of Shiraz University of Technology for supporting this work.

References

[1] K. Aardal, C. A. J. Hurkans, and A. K. Lenstra, Solving a systems of linear Diophantine equations with lower and upper bounds on the variables, *Mathematics of Operations Research*, 3 (2000), 427-442.

[2] J. Abaffy, C. G. Broyden, and E. Spedicato, A class of direct methods for linear equations, *Numerische Mathematik*, 45 (1984), 361-376.

[3] J. Abaffy and E. Spedicato, *ABS Projection Algorithms: Mathematical Techniques for Linear and Nonlinear Equations*, Ellis Horwood, Chichester, 1989.

[4] K. Amini and N. Mahdavi-Amiri, Solving rank one perturbed linear Diophantine systems by the ABS methods, *Optimization Methods and Software*, 21 (5) (2006), 819-831.

[5] K. Amini, N. Mahdavi-Amiri, and M. R. Peyghami, ABS-Type Methods for Solving Full Row Rank Linear Systems Using a New Rank Two Update, *Bulletin of the Australian Mathematical Society*, 70 (1) (2004), 17-34.

[6] S. Barnette and I. S. Pace, Efficient algorithms for linear systems calculations: Part I-Smith form and common divisor of polynomial matrices, *Internat. J. of Systems Sci.,* 5 (1974), 403-411.

[7] W. A. Blankinship, Algorithm 287, matrix triangulation with integer arithmetic [F1], *Comm. ACM.*, 9 (1966), 513.

[8] W. A. Blankinship, Algorithm 288, solution of simultaneous linear Diophantine equations, *Comm. ACM,* 9 (1966), 514.

[9] G. H. Bradley, Algorithms for Hermite and Smith normal matrices and linear Diophantine equations, *Math. Comp.,* 25 (1971), 897-907.

[10] T. J. Chou and E. E. Collins, Algorithms for the solutions of systems of linear Diophantine equations, *SIAM J. Comput.,* 11 (1982), 686-708.

[11] E. Contejean and H. Devie, An efficient algorithm for solving systems of Diophantine equations, *Information and Computation*, 1 (1994), 143-172.

[12] M. Clausen and A. Fortenbacher, Efficient solution of Diophantine equations, *Journal of Symbolic Computation*, 8 (1&2) (1989), 201-216.

[13] H. Cohen, *A Course in Computational Number Theory,* Vol. 138 of graduate texts in mathematics, Springer, Heidelberg, 1993.

[14] B. M. M. de Weger, A solving exponential Diophantine equations using lattice basis reduction algorithms, *Journal of Number Theory*, 26 (1987), 325-367.

[15] H. Esmaeili, N. Mahdavi-Amiri, and E. Spedicato, A class of ABS algorithms for Diophantine linear systems, *Numerische Mathematik*, 90 (2001), 101-115.

[16] H. Esmaeili, N. Mahdavi-Amiri, and E. Spedicato, Explicit ABS solution of a class of linear inequality systems and LP problems, *Bulletin of the Iranian Mathematical Society,* 30 (2) (2004), 21-38.

[17] H. Esmaeili, N. Mahdavi-Amiri, and E. Spedicato, Generating the integer null space and conditions for determination of an integer basis using the ABS algorithms, *Bulletin of the Iranian Mathematical Society*, 27 (1) (2001), 1-18.

[18] M. A. Frumkin, Polynomial time algorithms in the theory of linear diophantine equations, *M. Karpinski, ed., Fundamentals of Computation Theory, Lecture Notes in Computer Science*, Vol. 56, Springer, New York, 1977, 386-392.

[19] E. Golpar-Raboky and N. Mahdavi-Amiri, Diophantine quadratic equation and smith normal form using scaled extended integer Abaffy-Broyden-Spedicato algorithms, *Journal of Optimization Theory and Applications*, 152 (1) (2012), 75-96.

[20] G. Havas, B. S. Majewski, and K. R. Mathews, Extended GCD and Hermite normal form algorithms via lattice basis reduction, *Experimental Mathematics*, 7 (2) (1998), 125-135.

[21] C. Hermite, Sur I'introduction des variables continues dans la thérie des nombers, *J. Reine Angew. Math.,* 41 (1851), 191-216.

[22] R. Kannan and A. Bachem, Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix, *SIAM J. Comput.*, 8 (1979), 499-507.

[23] M. Khorramizadeh, Numerical experiments with the LLL-based hermite normal form algorithm for solving linear diophantine systems, *Int. J. Contemp. Math. Sciences,* 7 (13) (2012), 599-613.

[24] M. Khorramizadeh and N. Mahdavi-Amiri, An efficient algorithm for solving rank one perturbed linear Diophantine systems using Rossers approach, *4OR-A Quarterly Journal of Operations Research*, 9 (2011), 159-173.

[25] M. Khorramizadeh and N. Mahdavi-Amiri, On solving linear Diophantine systems using generalized Rosser's algorithm, *Bulletin of the Iranian Mathematical Society*, 34 (2) (2008), 1-25.

[26] M. Khorramizadeh and N. Mahdavi-Amiri, Integer extended ABS algorithms and possible control of intermediate results for linear Diophantine systems, *4OR-A Quarterly Journal of Operations Research*, 7 (2) (2009), 145-167.

[27] M. Pohst, *Computational Algebraic Number Theory*, Birkhauser, Boston, 1993.

[28] M. Newman, *Integral Matrices*, Academic press, New York, 1972.

[29] K. H. Rosen, *Elementary Number Theory and Its Applications, (4th ed.)*, Addison Wesley, 2000.

[30] J. B. Rosser, A note on the linear Diophantine equation, *Amer. Math. Monthly*, 48 (1941), 662-666.

[31] A. Schrijver, *Theory of Linear and Integer Programming*, John Wiley and Sons, 1986.

[32] H. J. S. Smith, On systems of linear indeterminate equations and congruences, *Philos. Trans. Roy. Soc. London, Ser. A.*, 151 (1861), 293-326.

[33] E. Spedicato, E. Bodon, Z. Xia, and N. Mahdavi-Amiri, ABS methods for continuous and integer linear equations and optimization, *Central European Journal of Operations Research*, 18 (1) (2010), 73-95.

[34] E. Spedicato, E. Bodon, Z. Xia, and N. Mahdavi-Amiri, ABS methods and ABSPACK for linear systems and optimization: A review, *4OR-A Quarterly Journal of Operations Research*, 1 (1) (2003), 51-56.

Mostafa Khorramizadeh
Department of Mathematics
Assistant Professor of Mathematics
Shiraz University of Technology
P.O. Box 71555-313
Shiraz, Iran
E-mail: m.khorrami@sutech.ac.ir