New Algorithms: Linear, Nonlinear, and Integer Programming

Dhananjay P. Mehendale
Sir Parashurambhau College, Tilak Road, Pune-411030,
India
dhananjay.p.mehendale@gmail.com

Abstract

In this paper we propose a new algorithm for linear programming. This new algorithm is based on treating the objective function as a parameter. We form a matrix using coefficients in the system of equations consisting objective equation and equations obtained from inequalities defining constraint by introducing slack/surplus variables. We obtain reduced row echelon form for this matrix containing only one variable, namely, the objective function itself as an unknown parameter. We analyze this matrix in the reduced row echelon form and develop a clear cut method to find the optimal solution for the problem at hand, if and when it exists. We see that the entire optimization process can be developed through the proper analysis of the said matrix in the reduced row echelon form. From the analysis of the said matrix in the reduced row echelon form it will be clear that in order to find optimal solution we may need carrying out certain processes like rearranging of the constraint equations in a particular way and/or performing appropriate elementary row transformations on this matrix in the reduced row echelon form. These operations are mainly aimed at achieving nonnegativity of all the entries in the columns corresponding to nonbasic variables in this matrix or its submatrix obtained by collecting certain rows of this matrix (i.e. submatrix with rows having negative coefficient for parameter *d*, which stands for the objective function as a parameter for maximization problem and submatrix with rows having positive coefficient parameter *d*, again representing the objective function as a parameter for minimization problem). The care is to be taken so that the new matrix arrived at by rearranging the constraint equations and/or by carrying out suitable elementary row transformations must be equivalent to original matrix. This equivalence is in the sense that all the feasible solution sets for the problem variables obtained for different possible values of *d* with original matrix and transformed matrix are same. We then proceed to show that this idea naturally extends to deal with nonlinear and integer programming problems. For nonlinear and integer programming problems we use the technique of Grobner bases (since Grobner basis is an equivalent of reduced row echelon form for a system of nonlinear equations) and the methods of solving linear Diophantine equations (since the integer programming problem demands for optimal integer solution) respectively.

1. Introduction: There are two types of **linear programs** (linear programming problems):

1. Maximize: $C^T x$

   Subject to: $Ax \le b$

   $x \ge 0$

   Or

2. Minimize: $C^T x$

Subject to: $Ax \geq b$

$x \geq 0$

where x is a column vector of size n×1 of unknown variables. We call these variables the **problem** variables

where C is a column vector of size n×1 of profit (for maximization problem) or cost (for minimization problem) coefficients, and C^T is a row vector of size 1×n obtained by matrix transposition of C.

where A is a matrix of constraints coefficients of size m×n.

where b is a column vector of constants of size m×1 representing the boundaries of constraints.

By introducing the appropriate **slack** variables (for maximization problem) and **surplus** variables (for minimization problem), the above mentioned linear programs gets converted into **standard form** as:

Maximize: $C^T x$

Subject to: $Ax + s = b$ (1.1)

$x \geq 0, s \geq 0$

where s is slack variable vector of size m×1.

This is a **maximization problem**.

Or

Minimize: $C^T x$

Subject to: $Ax - s = b$ (1.2)

$x \geq 0, s \geq 0$

where s is surplus variable vector of size m×1.

This is a **minimization problem**.

In geometrical language, the constraints defined by the inequalities form a region in the form of a convex polyhedron, a region bounded by the **constraint planes,** $Ax_i = b_i$, and the coordinate planes. This region is called **feasible region** and it is straightforward to establish that there exists at least one vertex of this polyhedron at which the optimal solution for the problem is situated when the problem at hand is well defined, i.e. neither inconsistent, nor unbounded, nor infeasible. There may be unique optimal solution and sometimes there may be infinitely many optimal solutions, e.g. when one of the constraint planes is parallel to the objective plane we may have a multitude of optimal solutions. The points on an entire plane or an entire edge can constitute the optimal solution set.

These problems are handled most popularly by using the well known **simplex algorithm or some of its variant**. Despite its theoretical exponential complexity the simplex method works quite efficiently for most of the practical problems. However, there are few computational difficulties associated with simplex algorithm. In order to view them in nutshell we begin with stating some common notions and definitions that are prevalent in the literature. A variable x_i is called **basic variable** in a given equation if it appears with unit coefficient in that equation and with zero coefficients in all other equations. A variable which is not basic is called **nonbasic variable**. A sequence of elementary row operations that changes a given system of linear equations into an **equivalent system** (having the same solution set) and in which a given nonbasic variable can be made a basic variable is called a **pivot operation**. An equivalent system containing basic and nonbasic variables obtained by application of suitable elementary row operations is called **canonical system**. At times, the introduction of slack variables for obtaining standard form automatically produces a canonical system, containing at least one basic variable in each equation. Sometimes a sequence of pivot operations is needed to be performed to get a canonical system. The solution obtained from canonical system by setting the nonbasic variables to zero and solving for the basic variables is called **basic solution** and in addition when all the variables have nonnegative values the solution satisfying all the imposed constraints is called a **basic feasible solution**. Simplex method cannot start without an initial basic feasible solution. The process of finding such a solution, which is a necessity in many of practical problems, is called **Phase I** of the simplex algorithm. Simplex method starts its **Phase II** with an initial basic feasible solution in canonical form at hand. Then simplex tests whether this solution is optimal by checking whether all the values of **relative profits** (profits that result due to unit change in the values of nonbasic variables) of all the nonbasic variables are nonpositive. When not optimal, the simplex method obtains an adjacent basic feasible solution by selecting a nonbasic variable having largest relative profit to become basic. Simplex then determines and carries out the exiting of a basic variable, by the so called **minimum ratio rule**, to change it into a nonbasic variable leading to formation of a new canonical system. On this new canonical system the whole procedure is repeated till one arrives at an optimal solution.

The main computational difficulties of the simplex method which may cause the reduction in its computational efficiency are as follows:

1] There can be more than one nonbasic variable with largest value for relative profit and so a **tie** can take place while selecting a nonbasic variable to become basic. The choice at this situation is done arbitrarily and so the choice made at this stage causing largest possible per unit improvement is not necessarily the one that gives largest total improvement in the value of the objective function and so **not necessarily minimizes** the number of simplex iterations.

2] While applying minimum ratio rule it is possible for more than one constraint to give the same least ratio causing a **tie** in the selection of a basic variable to leave for becoming nonbasic. This degeneracy can cause a further complication, namely, the simplex method can go on without any improvement in the objective function and the method may trap into an infinite loop and fail to produce the desired optimal solution. This phenomenon is called **cycling** which enforces modification in the algorithm by introducing some additional time consuming rules that reduce the efficiency of the simplex algorithm.

3] Simplex is not efficient on theoretical grounds basically because it searches adjacent basic feasible solutions only and all other simplex variants which examine nonadjacent solutions as well have not shown any appreciable change in the overall efficiency of these modified simplex algorithms over the original algorithm.

Because of the far great practical importance of the linear programs and other similar problems in the operations research it is a most desired thing to have an algorithm which works in a **single step**, if not, in as few steps as possible. No method has been found which will yield an optimal solution to a linear program in a single step ([1], Page 19). We aim to propose an algorithm for linear programming which aims at fulfilling this requirement in a best possible and novel way.

2. A New Algorithm for Linear Programming: We start with the following equation:

$$C^T x = d \qquad (2.1)$$

where d is an **unknown parameter**, and call it the **objective equation.** The (parametric) plane defined by this equation will be called **objective plane**.

Please note that we are discussing first the **maximization problems**. A similar approach for **minimization problems** will be discussed next.

Given a maximization problem**,** we first construct the combined system of equations containing the objective equation and the equations defined by the constraints imposed by the problem under consideration, combined into a single matrix equation, viz.,

$$\begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & I_{(m\times m)} \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = \begin{bmatrix} d \\ b \end{bmatrix} \qquad (2.2)$$

Let **E** $= \begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & I_{(m\times m)} \end{bmatrix}$**,** and let **F** $= \begin{bmatrix} d \\ b \end{bmatrix}$

Let **[E, F]** denote the augmented matrix obtained by appending the column vector **F** to matrix **E** as a last column. We then find *R*, the **reduced row echelon form** ([2], pages 73-75) of the above augmented matrix **[E, F]**. Thus,

R = rref (**[E, F]**) (2.3)

Note that the augmented matrix **[E, F]** as well as its reduced row echelon form *R* contains **only one parameter**, namely, d and all other entries are **constants**. From *R* we can determine the solution set S for every fixed $d, S = \{\begin{bmatrix} x \\ s \end{bmatrix} / (fixed) d \in reals\}$. The subset of this solution set of vectors $\begin{bmatrix} x \\ s \end{bmatrix}$ which also satisfies the nonnegativity constraints is the set of all feasible solutions for that d**.** It is clear that this subset can be **empty** for a particular choice of d that is made. The maximization problem of linear programming is to determine the unique d which provides a feasible solution and has maximum value for d, i.e., to determine the unique d**, i.e. the unique optimal value for the objective function,** which can be used to obtain an optimal solution. In the case of an **unbounded** linear program there is no upper (lower, in the case of minimization problem) limit for the value of d**,** while in the case of an **infeasible** linear program the set of feasible solutions is empty. The steps that will be executed to determine the optimal solution should also tell by implication when such optimal solution does not exist in the case of an unbounded or infeasible problem.

The general form of the **matrix *R*** representing the reduced row echelon form is

$$R = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_{11} & b_{12} & \cdots & b_{1m} & c_1 d + e_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_{21} & b_{22} & \cdots & b_{2m} & c_2 d + e_2 \\ a_{31} & a_{32} & \cdots & a_{3n} & b_{31} & b_{32} & \cdots & b_{3m} & c_3 d + e_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_{m1} & b_{m2} & \cdots & b_{mm} & c_m d + e_m \\ a_{(m+1)1} & a_{(m+1)2} & \cdots & a_{(m+1)n} & b_{(m+1)1} & b_{(m+1)2} & \cdots & b_{(m+1)n} & c_{(m+1)} d + e_{(m+1)} \end{bmatrix}$$

The first n columns of the above matrix represent the coefficients of the problem variables (i.e. variables defined in the linear program) $x_1, x_2, \cdots, x_n$. The next m columns represent the coefficients of the slack variables $s_1, s_2, \cdots, s_m$ used to convert inequalities into equalities to obtain the standard form of the linear program. The last column represents the transformed right hand side of the equation (2.2) during the process (a suitable sequence of transformations) that is carried out to obtain the reduced row echelon form. Note that the last column of *R* contains the linear form d as a parameter whose optimal value is to be determined such that the nonnegativity constraints remain valid, i.e. $x_i \geq 0, 1 \leq i \leq n$ and $s_j \geq 0, 1 \leq j \leq m$. Among first $(n+m)$ columns of *R* certain first columns correspond to **basic variables** (columns that are unit vectors) and the remaining ones to **nonbasic variables** (columns that are not unit vectors).

For solving a linear program **by our way** we proceed with analysis of the *R*. We aim to find that **value** of parameter d which is optimal. To achieve this task we may sometimes need to transform this system further by either rearranging the constraint equations by suitably permuting these equations such that (as far as possible) the values in the columns of our starting matrix $A_{(m \times n)}$ get rearranged in the following way: Either they are rising and then becoming stationary, or falling and then becoming stationary, or falling initially up to certain length of the column vector and then rising again. After this rearrangement to form transformed $A_{(m \times n)}$ we again proceed to form its corresponding **[E, F]** and again find its *R* = rref (**[E, F]**) which will most likely have the desired representation in which columns for nonbasic variables contain nonnegative values. The idea of arrangement of the columns of $A_{(m \times n)}$ as mentioned above is purely heuristic and is based on the favorable outcome observed during applying this idea to linear programming problems while tackling them. We have not tried to discover theoretical reasoning behind achieving favorable form for *R* in most of the cases and have left this problem for the reader to solve and to find out the theoretical reasoning behind getting this favorable form. Now, even after rearrangement of the columns of $A_{(m \times n)}$ as mentioned above if still some negative entries remain present in the columns of *R* corresponding to some nonbasic variables then we carry out suitable elementary row transformations on the obtained *R* = rref (**[E, F]**) so that the columns of coefficients associated with these nonbasic variables become nonnegative. We are doing this because as will be seen below we can then put zero value for these nonbasic variables and can determine the values of all the basic variables and the linear program will then be solved completely. It is easy to check that for a linear program if all the coefficients of parameter d in the last column of *R* are positive then the linear program at hand is unbounded since in such case the parameter d can be increased arbitrarily without violating the nonnegativity constraints on variables x_i, s_j. Also, for a linear program if all the coefficients of some nonbasic variable represented by a column of *R* are nonpositive and are strictly negative in those rows having a negative coefficient to parameter d that appears in the last column of these rows then again the problem belongs to the category of unbounded problems since we can increase the value of d to any high value without violating the nonnegativity constraints for the variables by assigning sufficiently high value to this nonbasic slack variable. Note that the rows of *R* actually offer expressions for basic variables in terms of nonbasic variables and terms of type $c_k d + e_k, k = 1,2,\cdots(m+1)$ containing the parameter d on the right side. The rows with a positive coefficient for the parameter d represent those equations in which the parameter d can be increased arbitrarily without violating the nonnegativity constraints on variables x_i, s_j. So, these equations with a positive coefficient for the parameter d are not implying any upper bound on the maximum possible value of parameter d. However, these rows are useful in certain situations as they are useful to find lower bound on the value of parameter d. The rows with a negative coefficient for the parameter d represent those equations in which the parameter d cannot be increased arbitrarily without violating the nonnegativity constraints on variables x_i, s_j. So, these equations with a negative coefficient for the parameter d are implying an upper bound on the maximum possible value of parameter d and so important ones for maximization problems. Note that actually every row of *R* is offering us a value for parameter d which can be obtained by equating to zero each term of the type $c_k d + e_k, k = 1,2,\cdots(m+1)$. Those values of d that we obtain in this way will be denoted as d^- or d^+ when then value of c_k is negative or positive respectively. We denote by min{d^-}, the minimum value among d^-, and we denote by max{d^+} the maximum value among the d^+. We now proceed to find out the **submatrix of** R, say R_N, made up of all columns of *R* and containing those rows j of R for which the coefficients c_j of the parameter d are negative. Let $c_{i_1}, c_{i_2}, \cdots, c_{i_k}$ coefficient of d in the rows of R which are negative. We collect these rows with negative coefficient for d to form the mentioned submatrix, R_N, of R given below. With this it is clear that coefficients of d in all other rows of R are greater than or equal to zero.

$$R_N = \begin{bmatrix} a_{i_1 1} & a_{i_1 2} & \cdots & a_{i_1 n} & b_{i_1 1} & b_{i_1 2} & \cdots & b_{i_1 m} & c_{i_1} d + e_{i_1} \\ a_{i_2 1} & a_{i_2 2} & \cdots & a_{i_2 n} & b_{i_2 1} & b_{i_2 2} & \cdots & b_{i_2 m} & c_{i_2} d + e_{i_2} \\ a_{i_3 1} & a_{i_3 2} & \cdots & a_{i_3 n} & b_{i_3 1} & b_{i_3 2} & \cdots & b_{i_3 m} & c_{i_3} d + e_{i_3} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ a_{i_k 1} & a_{i_k 2} & \cdots & a_{i_k n} & b_{i_k 1} & b_{i_k 2} & \cdots & b_{i_k m} & c_{i_k} d + e_{i_k} \end{bmatrix}$$

It should be clear to see that if R_N is **empty** (i.e. not containing a single row) then the problem at hand is **unbounded**. Among the first $(n+m)$ columns of R_N first n columns represent the coefficients of problem variables and next m columns represent the coefficients of slack variables. There are certain columns starting from first column and appear in successions which are unit vectors. These columns which are unit vectors correspond to **basic variables**. The columns appearing in successions after these columns and not unit vectors correspond to **nonbasic variables**. As mentioned, among the columns for nonbasic variables those having all entries nonnegative can only lead to decrement in the value of d when a positive value is assigned to them. This is undesirable as we aim maximization of the value of d. So, we can safely set the value of such variables equal to zero. When all columns corresponding to nonbasic variables in R_N are having all entries nonnegative and further if $\min\{d^-\} \geq \max\{d^+\}$ then we can set all nonbasic variables to zero, set $d = \min\{d^-\}$ in every row of R and find the basic feasible solution which will be **optimal**, with $\min\{d^-\}$ as **optimal value** for the objective function at hand. Still further, When all columns corresponding to nonbasic variables in R_N are having all entries nonnegative but $\min\{d^-\} < \max\{d^+\}$ then if $e_k > 0$ then we can still set all nonbasic variables to zero, set $d = \min\{d^-\}$ in every row of R and find the basic feasible solution which will be **optimal**, with $\min\{d^-\}$ as **optimal value** for the objective function at hand, i.e. if value of $e_k > 0$ in the expressions $c_k d + e_k$ in the rows of R other those in R_N that are having value of c_k > 0 then we can proceed on similar lines to find optimal value for d.

In R_N we now proceed to consider those nonbasic variables for which the columns of R_N contain some (at least one) positive values and some negative (at least one) values. In such case when we assign some positive value to such nonbasic variable it leads to decrease in the value of d in those rows in which $c_k > 0$ and increase in the value of d in those rows in which $c_k < 0$. We now need to consider the ways of dealing with this situation. We deal with this situation as follows: In this case, we choose and carry out appropriate and legal elementary row transformations on the matrix R in the reduced row echelon form to achieve nonnegative value for all the entries in the columns corresponding to nonbasic variables in the submatrix R_N of R. The elementary row transformations are chosen to produce new matrix which remains equivalent to original matrix in the sense that the solution set of the matrix equation with original matrix and matrix equation with transformed matrix remain same. Due to this equivalence we can now set all the nonbasic variables in this transformed matrix to zero and obtain with justification $d_{\min} = \min\{d^-\}$ as optimal value for the objective function and obtain basic feasible solution as optimal solution by substitution.

Let us now discuss our new algorithm in steps:

Algorithm 2.1 (Maximization):

1. Express the given problem in standard form:
   Maximize: $C^T x$
   Subject to: $Ax + s = b$
   $x \geq 0, s \geq 0$
2. Construct the augmented matrix [**E F**], where
   $$\mathbf{E} = \begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & I_{(m\times m)} \end{bmatrix}, \text{ and } \mathbf{F} = \begin{bmatrix} d \\ b \end{bmatrix}$$
   and obtain the reduced row echelon form:
   R = rref (**[E, F]**)
3. If there is a row (or rows) of zeroes at the bottom of R in the first n columns and containing a nonzero constant in the last column then declare that the problem is **inconsistent** and stop. Else if the coefficients of d in the last column are all positive or if there exists a column of R corresponding to some nonbasic variable with all entries negative then declare that the problem at hand is **unbounded** and stop.
4. Else if for any value of d one observes that nonnegativity constraint for some variable gets violated by at least one of the variables then declare that the problem at hand is **infeasible** and stop.

5. Else find the submatrix of R, say R_N, made up of those rows of R for which the coefficient of d in the last column is negative.
6. Check whether the columns of R_N corresponding to nonbasic variables are nonnegative. Else, rearrange the constraint equations by suitably permuting these equations such that (as far as possible) the values in the columns of our starting matrix $A_{(m \times n)}$ get rearranged in the following way: Either they are rising and then becoming stationary, or falling and then becoming stationary, or falling initially up to certain length of the column vector and then rising again. After this rearrangement to form transformed $A_{(m \times n)}$ again proceed as is done in step 2 above to form its corresponding augmented matrix **[E, F]** and again find its R = rref (**[E, F]**) which will most likely have the desired representation, i.e. in the new R_N that one will construct from the new R will have columns for nonbasic variables which will be containining nonnegative entries.
7. Solve $c_{i_r} d + e_{i_r} = 0$ for each such a term in the last column of R_N and find the value of $d = d_{i_r}^-$ for $r = 1, 2, \cdots, k$ and find $d_{\min}^- = \min\{d_{i_r}^-\}$. Similarly, solve $c_{i_r} d + e_{i_r} = 0$ for each such a term in the last column for rows of R other than those in R_N and find the values $d = d_{i_r}^+$ for $r = 1, 2, \cdots, k$ and find $d_{\max}^+ = \max\{d_{i_r}^+\}$. Check the columns of R_N corresponding to nonbasic variables. If all these columns contain only nonnegative entries and if $\min\{d^-\} \geq \max\{d^+\}$ then set all nonbasic variables to zero. Substitute $d = d_{\min}$ in the last column of R. Determine the basic feasible solution which will be optimal and stop. Further, if columns corresponding to nonbasic variables contain only nonnegative entries and if $\min\{d^-\} < \max\{d^+\}$ then check whether value of $e_k > 0$ in the expressions $c_k d + e_k$ in these rows of R other those in R_N that are having value of $c_k > 0$. If yes, then set all nonbasic variables to zero. Substitute $d = d_{\min}$ in the last column of R. Determine the basic feasible solution which will be again optimal.
8. Even after proceeding with rearranging columns of our starting matrix $A_{(m \times n)}$ by suitably permuting rows representing constraint equations as is done in step 6, and then proceeding as per step 7 and achieving the validity of $\min\{d^-\} \geq \max\{d^+\}$, if still there remain columns in R_N corresponding to some nonbasic variables containing positive entries in some rows and

negative entries in some other rows then devise and apply suitable elementary row transformations on R such that the columns representing coefficients of nonbasic variables of new transformed matrix R or at least its submatrix R_N corresponding to new transformed R contain only nonnegative entries**.** And step 7 becomes applicable.

We now proceed with some examples:

Example 2.1: Maximize: $x+y$

Subject to: $x+2y\le 4$

$$-x+y\le 1$$

$$4x+2y\le 12$$

$$x,y\ge 0$$

Solution: For this problem we have

$R=$ [1, 0, 0, 0, 1/2, -d+6]
[0, 1, 0, 0, -1/2, -6+2*d]
[0, 0, 1, 0, 1/2, 10-3*d]
[0, 0, 0, 1, 1, 13-3*d]

So, clearly,

R_N = [1, 0, 0, 0, 1/2, -d+6]
[0, 0, 1, 0, 1/2, 10-3*d]
[0, 0, 0, 1, 1, 13-3*d]

For this example the column forming coefficients for nonbasic variable s_3 contains nonegative numbers. So, we set $s_3=0$. Clearly, $d_{\min}$ = 3.3333 = Optimal value for the objective function. Using this value of optimum we have $x=2.66, y=0.66$, $s_1=0, s_2=3, s_3=0$.

Example 2.2: We first consider the duel of the example suggested by E. M. L. Beale [3], which brings into existence the problem of **cycling** for the simplex method, and provide a solution as per the above new method which offers it directly without any cycling phenomenon.

Maximize: $0.75x_1-20x_2+0.5x_3-6x_4$

Subject to: $0.25x_1-8x_2-x_3+9x_4\le 0$

$$0.5x_1-12x_2-0.5x_3+3x_4\le 0$$

$$x_3\le 0$$

$$x_1,\ x_2,\ x_3,\ x_4\ge 0$$

Solution: For this problem we have the following

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & -22/3, & 38/3, & 4/3, & (-14/3)d+4/3 \\ 0, & 1, & 0, & 0, & -7/24, & 11/24, & 1/24, & (-5/24)d+1/2 \\ 0, & 0, & 1, & 0, & 0, & 0, & 1, & 1 \\ 0, & 0, & 0, & 1, & 1/18, & 1/18, & 1/9, & 1/9-(1/18)d \end{bmatrix}$$

So, clearly,

$$R_N = \begin{bmatrix} 1, & 0, & 0, & 0, & -22/3, & 38/3, & 4/3, & (-14/3)d+4/3 \\ 0, & 1, & 0, & 0, & -7/24, & 11/24, & 1/24, & (-5/24)d+1/24 \\ 0, & 0, & 0, & 1, & 1/18, & 1/18, & 1/9, & 1/9-(1/18)d \end{bmatrix}$$

We perform following elementary row transformations on *R*: Let us denote the successive rows of *R* by R(1), R(2), R(3), R(4). We change

(i) R(2) → R(2) + (126/24)*R(4),
(ii) R(1) → R(1) + 132*R(4).
(iii) R(4) → 18*R(4)

This leads to new transformed *R* as follows:

$$R = \begin{bmatrix} 1, & 0, & 0, & 132, & 0, & 20, & 1\ 6, & 16\text{-}12d \\ 0, & 1, & 0, & 21/4, & 0, & 3/4, & 5/8, & 5/8\text{-}(1/2)d \\ 0, & 0, & 1, & 0, & 0, & 0, & 1, & 1 \\ 0, & 0, & 0, & 18, & 1, & 1, & 2, & 2-d \end{bmatrix}$$

In the transformed *R* we have nonnegative columns for all nonbasic variables, which are now those corresponding to x_4, s_2, s_3. So, by setting $x_4 = s_2 = s_3 = 0$ and setting all expressions of type $c_{i_r} d + e_{i_r} = 0$ in the last column we find $d^-_{\min} = \min\{ d^-_{i_r} \}$ = 1.25. Using this value in the last column of the newly obtained transformed *R* we have: x_1= 1.0000, x_2= 0, x_3= 1, x_4 = 0, s_1= 0.7500, s_2= 0, s_3= 0, and the maximum value of d = 1.2500.

Example 2.3: We now consider an **unbounded** problem. The new method directly implies the unbounded nature of the problem through the positivity of the coefficients of d in matrix R for the problem.

Maximize: $-x+3y$

Subject to: $-x-y \le -2$

$$x - 2y \le 0$$

$$-2x + y \le 1$$

$$x, y \ge 0$$

Solution: The following is the matrix R:

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & -3/5, & (1/5)d-3/5 \\ 0, & 1, & 0, & 0, & -1/5, & (2/5)d-1/5 \end{bmatrix}$$

[0, 0, 1, 0, -4/5, (3/5)d-14/5]
[0, 0, 0, 1, 1/5, 1/5+(3/5)d]

Here, all the coefficients of d are positive. So, by setting variable $s_3 = 0$ we can see that we can assign any arbitrarily large value to variable d without violation of nonnegativity constraints for variables. Thus, the problem has an unbounded solution.

Example 2.4: We now consider a problem having an **infeasible starting basis**. We see that new algorithm has no difficulty to deal with it.

Maximize: $3x + 2y$

Subject to: $x + y \leq 4$

$$2x + y \leq 5$$

$$x - 4y \leq -2$$

$$x, y \geq 0$$

Solution: The following is the matrix R:

R = [1, 0, 0, 0, 1/7, (2/7)d-2/7]
[0, 1, 0, 0, -3/14, 3/7+(1/14)d]
[0, 0, 1, 0, 1/14, 27/7-(5/14)d]
[0, 0, 0, 1, -1/14, 36/7-(9/14)d]

R_N = [0, 0, 1, 0, 1/14, 27/7-(5/14)d]
[0, 0, 0, 1, -1/14, 36/7-(9/14)d]

Clearly, min{d^-} ≥ max{d^+}. So, we perform following elementary row transformations on *R*: Let us denote the successive rows of *R* by R(1), R(2), R(3), R(4). We change

(i) R(4) → R(4) + R(3)

This leads to new transformed R_N, R as follows:

R_N = [0, 0, 1, 0, 1/14, (27/7)-(5/14)d]
[0, 0, 1, 1, 0, (63/7) – d]

R = [1, 0, 0, 0, 1/7, (2/7)d-2/7]
[0, 1, 0, 0, -3/14, 3/7+(1/14)d]
[0, 0, 1, 0, 1/14, 27/7-(5/14)d]
[0, 0, 1, 1, 0, (63/7) – d]

The first two columns of R, R_N correspond to basic variables x, y. Since, min{d^-} ≥ max{d^+} and columns corresponding to nonbasic variables s_1, s_3 contain nonnegative entries in R_N, so we set these variables to zero. From last row we have $s_2 = 0$ and $d_{\min} =$ 9. Also from first and second rows, $x = 2.2857$, $y = 1.0714$

Example 2.5: We now consider an **infeasible** problem.

Maximize: $3x + 2y$

Subject to: $2x - y \le -1$

$$-x + 2y \le 0$$

$$x, y \ge 0$$

Solution: The following is the matrix R:

R = [1, 0, 0, -1/4, (1/4)d]
[0, 1, 0, 3/8, (1/8)d]
[0, 0, 1, 7/8, (-3/8)d-1]

Here, the coefficient of d is negative only in the last row and so

R_N = [0, 0, 1, 7/8, (-3/8)d-1].

We perform following elementary row transformations on *R*: Let us denote the successive rows of *R* by R(1), R(2), R(3), R(4). We change

(i) R(1) → (2/7)*R(4) + R(1)

This leads to

R = [1, 0, 2/7, 0, 1/7*d-2/7]
[0, 1, 0, 3/8, (1/8)d]
[0, 0, 1, 7/8, (-3/8)d-1]

and

R_N = [0, 0, 1, 7/8, (-3/8)d-1]

Setting s_1, s_2 equal to zero, we have for consistency of last row *d* = −(8/3) and using this value for *d* we have y = −(1/3). Thus, this problem is **infeasible.**

Remark 2.1: Klee and Minty [4], have constructed an example of a set of linear programs with n variables for which simplex method requires $2^n - 1$ iterations to reach an optimal solution. Theoretic work of Borgwardt [5] and Smale [6] indicates that fortunately the occurrence of problems belonging to the class of Klee and Minty, which don't share the average behavior, is so rare as to be negligible. We now proceed to show that there is no problem of efficiency for new algorithm in dealing with the problems belonging to this class.

Example 2.6: We now consider a problem for which the simplex iterations are **exponential** function of the size of the problem. A problem belonging to the class described by Klee and Minty containing n variables requires $2^n - 1$ simplex steps. We see that the new method doesn't require any special effort

Maximize: $100x_1 + 10x_2 + x_3$

Subject to: $x_1 \le 1$

$$20x_1 + x_2 \le 100$$

$$200x_1 + 20x_2 + x_3 \le 10000$$

$$x_1,\ x_2,\ x_3 \ge 0$$

Solution: The following are the matrices R, R_N:

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & 1/10, & -1/100, & -90+(1/100)d \\ 0, & 1, & 0, & 0, & -1, & 1/5, & 1900-(1/5)d \\ 0, & 0, & 1, & 0, & 0, & -1, & 2d-10000 \\ 0, & 0, & 0, & 1, & -1/10, & 1/100, & 91-(1/100)d \end{bmatrix}$$

$$R_N = \begin{bmatrix} 0, & 1, & 0, & 0, & -1, & 1/5, & 1900-(1/5)d \\ 0, & 0, & 0, & 1, & -1/10, & 1/100, & 91-(1/100)d \end{bmatrix}$$

We perform following elementary row transformations on *R*: Let us denote the successive rows of *R* by R(1), R(2), R(3), R(4). We change

(i) R(2) → 10R(1) +R(2), and
(ii) R(4) → R(1) +R(4)
(iii) R(1) → 10*R(1)

This leads to new transformed R_N, R as follows:

$$R_N = \begin{bmatrix} 0, & 1, & 0, & 0, & 0, & 1/10, & 1000-(1/10)*d \\ 1, & 0, & 0, & 1, & 0, & 0, & 1 \end{bmatrix}$$

$$R = \begin{bmatrix} 10, & 0, & 0, & 0, & 1, & -1/10, & -900+1/10*d \\ 0, & 1, & 0, & 0, & 0, & 1/10, & 1000-(1/10)*d \\ 0, & 0, & 1, & 0, & 0, & -1, & 2*d-10000 \\ 1, & 0, & 0, & 1, & 0, & 0, & 1 \end{bmatrix}$$

Since, min{d^-} ≥ max{d^+} and columns corresponding to nonbasic variables x_1, s_3 contain nonnegative entries in R_N, so we set these variables to zero.

We get easy complete solution as follows:
$x_1 = 0$, $x_2 = 0$, $x_3 = 10000$, $s_1 = 1$, $s_2 = 100$, $s_3 = 0$, and the maximum value of d = 10000.

Example 2.7: We now consider an example for which the reduced row echelon form **contains by itself the basic variables** that are required to be present in the optimal simplex tableau, i.e. the tableau that results at the end of the simplex algorithm, for which only nonpositive entries occur in the bottom row of the tableau representing relative profits. This is understood by the **nonnegativity of entries in the columns of R_N corresponding to nonbasic variables**.

Maximize: $3x+2y$

Subject to: $-x+2y \le 4$

$$3x+2y \le 14$$
$$x-y \le 3$$
$$x, y \ge 0$$

Solution: The following is the matrix R:

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & 2/5, & (1/5)d+6/5 \\ 0, & 1, & 0, & 0, & -3/5, & -9/5+(1/5)d \\ 0, & 0, & 1, & 0, & 8/5, & 44/5-(1/5)d \\ 0, & 0, & 0, & 1, & 0, & 14-d \end{bmatrix}$$

So, clearly,

$$R_N = \begin{bmatrix} 0, & 0, & 1, & 0, & 8/5, & 44/5-(1/5)d \\ 0, & 0, & 0, & 1, & 0, & 14-d \end{bmatrix}$$

Since, min{ d^- } $\ge$ max{ d^+ } and columns corresponding to nonbasic variable s_3 contain nonnegative entries in R_N , so we set these variables to zero.
Here, the nonbasic variable columns directly contain nonnegative entries leading to decreasing in profit when some positive value is assigned to this variable and so we set this variable to zero which leads to the maximal basic feasible solution: $d=14$, $x=4, y=1, s_1=6, s_2=0, s_3=0$.

We now consider few examples in which **rearrangement of constraint equations** automatically produce **suitable form for reduced row echelon form:**

Example 2.8:. Maximize: $6x+3y$

Subject to: $x+y \le 5$

$$4x+y \le 12$$
$$-x-2y \le -4$$
$$x, y \ge 0$$

For this problem we get following R, R_N

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & 1/3, & 2/9*d-4/3 \\ 0, & 1, & 0, & 0, & -2/3, & 8/3-1/9*d \\ 0, & 0, & 1, & 0, & 1/3, & 11/3-1/9*d \\ 0, & 0, & 0, & 1, & -2/3, & 44/3-7/9*d \end{bmatrix}$$

$$R_N = \begin{bmatrix} 0, & 1, & 0, & 0, & -2/3, & 8/3-1/9*d \\ 0, & 0, & 1, & 0, & 1/3, & 11/3-1/9*d \\ 0, & 0, & 0, & 1, & -2/3, & 44/3-7/9*d \end{bmatrix}$$

Here some entries in the last but one column corresponding to nonbasic variable are mixed type, i.e. entries in the first and third rows of R_N are negative while entry in the second row of R_N is positive. We now rearrange the constraints so that either entries in the new columns are rising and then becoming stationary, or falling and then becoming stationary, or falling initially up to certain length of the column vector and then rising again, as mentioned above. Thus, we just rewrite the problem as:

Maximize: $6x+3y$

Subject to: $x+y\le 5$

$-x-2y\le -4$

$4x+y\le 12$

$x,y\ge 0$

then form new $A_{(m\times n)}$ and again proceed to find reduced row echelon form for new matrix which produce

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & 1/2, & -1/6*d+6 \\ 0, & 1, & 0, & 0, & -1, & -12+2/3*d \\ 0, & 0, & 1, & 0, & 1/2, & 11-1/2*d \\ 0, & 0, & 0, & 1, & -3/2, & -22+7/6*d \end{bmatrix}$$

$$R_N = \begin{bmatrix} 1, & 0, & 0, & 0, & 1/2, & -1/6*d+6 \\ 0, & 0, & 1, & 0, & 1/2, & 11-1/2*d \end{bmatrix}$$

Since, min$\{d^-\}\ge$ max$\{d^+\}$ and columns corresponding to nonbasic variable s_3 contain nonnegative entries in R_N, so we set these variables to zero.
Here, the nonbasic variable columns directly contain nonnegative entries leading to decreasing in profit when some positive value is assigned to this variable and so we set this variable to zero which leads to the maximal basic feasible solution: $d=22$, $x=7/3, y=8/3, s_1=0, s_2=11/3, s_3=0$.

Example 2.9: Maximize: $3x+2y$

Subject to: $x+y\le 4$

$2x+y\le 5$

$x-4y\le -2$

$x,y\ge 0$

For this problem we get following R, R_N

$R = [\quad 1, \quad 0, \quad 0, \quad 0, \quad 1/7, \quad 2/7*d-2/7 \quad]$

[0, 1, 0, 0, -3/14, 3/7+1/14*d]
[0, 0, 1, 0, 1/14, 27/7-5/14*d]
[0, 0, 0, 1, -1/14, 36/7-9/14*d]

R_N = [0, 0, 1, 0, 1/14, 27/7-5/14*d]
[0, 0, 0, 1, -1/14, 36/7-9/14*d]

Here some entries in the last but one column corresponding to nonbasic variable are mixed type, i.e. entries in the first and third rows of R_N are negative while entry in the second row of R_N is positive. We now rearrange the constraints so that either entries in the new columns are rising and then becoming stationary, or falling and then becoming stationary, or falling initially up to certain length of the column vector and then rising again, as mentioned above. Thus, we just rewrite the problem as:

Maximize: $3x+2y$
Subject to: $2x+y\leq 5$
$x-4y\leq -2$
$x+y\leq 4$
$x,y\geq 0$

then we form new $A_{(m\times n)}$ and again proceed to find reduced row echelon form for new matrix which produce

R = [1, 0, 0, 0, -2, d-8]
[0, 1, 0, 0, 3, 12-d]
[0, 0, 1, 0, 1, 9-d]
[0, 0, 0, 1, 14, 54-5*d]

R_N = [0, 1, 0, 0, 3, 12-d]
[0, 0, 1, 0, 1, 9-d]
[0, 0, 0, 1, 14, 54-5*d]

Since, min{ d^- } $\geq$ max{ d^+ } and column corresponding to nonbasic variable s_3 contain nonnegative entries in R_N , so we set this variable to zero.
Here, the nonbasic variable columns directly contain nonnegative entries leading to decreasing in profit when some positive value is assigned to this variable and so we set this variable to zero which leads to the maximal basic feasible solution: $d=9$, $x=1, y=3, s_1=0, s_2=9, s_3=0$.

Example 2.10: Maximize: $4x+3y$
Subject to: $x+3.5y\leq 9$
$2x+y\leq 8$

$$x + y \le 6$$
$$x, y \ge 0$$

For this problem we get following R, R_N

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & -3, & d-18 \\ 0, & 1, & 0, & 0, & 4, & 24-d \\ 0, & 0, & 1, & 0, & -11, & -57+5/2*d \\ 0, & 0, & 0, & 1, & 2, & 20-d \end{bmatrix}$$

$$R_N = \begin{bmatrix} 0, & 1, & 0, & 0, & 4, & 24-d \\ 0, & 0, & 0, & 1, & 2, & 20-d \end{bmatrix}$$

Clearly, though the second last column corresponding to nonbasic variable contains nonnegative entries the inequality min{d^-} ≥ max{d^+} is **invalid!** So, as a first attempt, before starting to carry out elementary row transformation on R to achieve nonnegativity of entries in the columns of R corresponding nonbasic variables, let us first try rearranging the inequalities so that either entries in the new columns are rising and then becoming stationary, or falling and then becoming stationary, or falling initially up to certain length of the column vector and then rising again, as mentioned above. Thus, we just rewrite the problem as:

Maximize: $4x + 3y$

Subject to: $x + 3.5y \le 9$

$$x + y \le 6$$
$$2x + y \le 8$$
$$x, y \ge 0$$

then we form new $A_{(m \times n)}$ and again proceed to find reduced row echelon form for new matrix which produce

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & 3/2, & -1/2*d+12 \\ 0, & 1, & 0, & 0, & -2, & -16+d \\ 0, & 0, & 1, & 0, & 11/2, & 53-3*d \\ 0, & 0, & 0, & 1, & 1/2, & 10-1/2*d \end{bmatrix}$$

$$R_N = \begin{bmatrix} 1, & 0, & 0, & 0, & 3/2, & -1/2*d+12 \\ 0, & 0, & 1, & 0, & 11/2, & 53-3*d \\ 0, & 0, & 0, & 1, & 1/2, & 10-1/2*d \end{bmatrix}$$

For this rearrangement the second last column corresponding to nonbasic variable contains nonnegative entries and also the inequality min{d^-} ≥ max{d^+} is now **valid!** So, we set nonbasic variable s_3 equal to zero which leads to maximal basic feasible solution: $d = 17.667$,

$$x = 3.1665, y = 1.667, s_1 = 0, s_2 = 1.1665, s_3 = 0$$

Example 2.11: Maximize: $3x+2y$

Subject to: $x+y\le 4$

$2x+y\le 5$

$x-4y\le -2$

Solution: For this problem we have

$R=$[1, 0, 0, 0, 1/7, (2/7)d-2/7]
[0, 1, 0, 0, -3/14, 3/7+ (1/14)d]
[0, 0, 1, 0, 1/14, 27/7- (5/14)d]
[0, 0, 0, 1, -1/14, 36/7- (9/14)d]

$R_N=$[0, 0, 1, 0, 1/14, 27/7-(5/14)d]
[0, 0, 0, 1, -1/14, 36/7-(9/14)d]

But, if we permute constraints as:

(I) constraint 1→ constraint 3,
(II) constraint 2→ constraint 1, and
(III) constraint 3→ constraint 2

and form new $A_{(m\times n)}$ and further find out new R, R_N then we get

$R=$[1, 0, 0, 0, -2, d-8]
[0, 1, 0, 0, 3, 12-d]
[0, 0, 1, 0, 1, 9-d]
[0, 0, 0, 1, 14, 54-5d]

$R_N=$ [0, 1, 0, 0, 3, 12-d]
[0, 0, 1, 0, 1, 9-d]
[0, 0, 0, 1, 14, 54-5d]

Since, min{d^-} ≥ max{d^+} and column corresponding to nonbasic variable s_3 contain nonnegative entries in R_N, so we set this variable to zero and we directly get the optimal basic feasible solution from $R =$: $d=d_{\min}=9$, $x=1, y=3$, $s_1=0, s_2=9, s_3=0$.

We now see that we can proceed with in exactly similar way and deal successfully with **minimization linear programming problems**. A problem of minimization goes like:

Minimize: $C^T x$

Subject to: $Ax\ge b$

$x\ge 0$

we first construct the combined system of equations containing the same objective equation used in maximization problem (but this time we want to find **minimum value of parameter d defined in the objective equation)** and the equations defined by the constraints imposed by the problem under consideration, combined into a single matrix equation, viz.,

$$\begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & -I_{(m\times m)} \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = \begin{bmatrix} d \\ b \end{bmatrix} \tag{2.2}$$

Let **E** $= \begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & -I_{(m\times m)} \end{bmatrix}$, and let **F** $= \begin{bmatrix} d \\ b \end{bmatrix}$

Let **[E, F]** denote the augmented matrix obtained by appending the column vector **F** to matrix **E** as a last column. We then find *R*, the **reduced row echelon form** of the above augmented matrix **[E, F]**. Thus,

$$R = \text{rref}\,([\mathbf{E}, \mathbf{F}]) \tag{2.3}$$

Note that the augmented matrix **[E, F]** as well as its reduced row echelon form *R* contains **only one parameter**, namely, d and all other entries are **constants**. From *R* we can determine the solution set S for every fixed $d, S = \{\begin{bmatrix} x \\ s \end{bmatrix} / (fixed)d \in reals\}$. The subset of this solution set of vectors $\begin{bmatrix} x \\ s \end{bmatrix}$ which also satisfies the nonnegativity constraints is the set of all feasible solutions for that d**.** It is clear that this subset can be **empty** for a particular choice of d that is made. The minimization problem of linear programming is to determine the unique d which provides a feasible solution and has minimum value, i.e., to determine the unique d which provides an optimal solution. In the case of an **unbounded** minimization linear program there is no **lower bound** for the value of d**,** while in the case of an **infeasible** linear program the set of feasible solutions is **empty**. The steps that will be executed to determine the optimal solution should also tell by implication when such optimal solution does not exist in the case of an unbounded or infeasible problem.

The general form of the matrix *R* representing the reduced row echelon form is similar as previously discussed maximization case:

$$R = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_{11} & b_{12} & \cdots & b_{1m} & c_1 d + e_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_{21} & b_{22} & \cdots & b_{2m} & c_2 d + e_2 \\ a_{31} & a_{32} & \cdots & a_{3n} & b_{31} & b_{32} & \cdots & b_{3m} & c_3 d + e_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_{m1} & b_{m2} & \cdots & b_{mm} & c_m d + e_m \\ a_{(m+1)1} & a_{(m+1)2} & \cdots & a_{(m+1)n} & b_{(m+1)1} & b_{(m+1)2} & \cdots & b_{(m+1)n} & c_{(m+1)} d + e_{(m+1)} \end{bmatrix}$$

The first n columns of the above matrix represent the coefficients of the problem variables (i.e. variables defined in the linear program) $x_1, x_2, \cdots, x_n$. The next m columns represent the coefficients of the surplus variables $s_1, s_2, \cdots, s_m$ used to convert inequalities into equalities to obtain the standard form of the linear program. The last column represents the transformed right hand side of the equation (2.2) during the process (a suitable sequence of transformations) that is carried out to obtain the reduced row echelon form. Note that the last column of *R* contains the linear form d as a parameter whose optimal value is to be determined such that the nonnegativity constraints remain valid, i.e. $x_i \geq 0, 1 \leq i \leq n$ and $s_j \geq 0, 1 \leq j \leq m$. Among first $(n+m)$ columns of *R* certain first columns correspond to **basic variables** (columns that are unit vectors) and the remaining ones to **nonbasic variables** (columns that are not unit vectors). For solving the linear program we need to determine the **values** of all nonbasic variables and the optimal value of d, from which we can determine the values of all the basic variables by substitution and the linear program is thus solved completely. For a linear program if all the coefficients of parameter d in the last column of *R* are **negative** then the linear program at hand is unbounded (since, the parameter d can be decreased arbitrarily without violating the nonnegativity constraints on variables x_i, s_j). For a linear program if all the coefficients of some nonbasic slack variable represented by a column of *R* are nonpositive and are strictly negative in those rows having a positive coefficient to parameter d that appears in the last column of these rows then we can decrease the value of d to any low value without violating the nonnegativity constraints for the variables by assigning sufficiently high value to this nonbasic slack variable and the problem is again belongs to the category of unbounded problems. Note that the rows of *R* actually represent equations with variables $x_i, i = 1,2,\cdots n$ and variables $s_j, j = 1,2,\cdots m$ on left side and expressions of type $c_k d + e_k, k = 1,2,\cdots(m+1)$ containing the variable d on the right side. The rows with a negative coefficient for the parameter d represent those equations in which the parameter d can be decreased arbitrarily without violating the nonnegativity constraints on variables x_i, s_j. So, these equations with a negative coefficient for the parameter d are not implying any lower bound on the minimum possible value of parameter d. However, these rows are useful to know about upper bound on parameter d. The rows with a positive coefficient for the parameter d represent those equations in which the parameter d cannot be decreased arbitrarily without violating the nonnegativity constraints on variables x_i, s_j. So, these equations with a positive coefficient for the parameter d are implying a lower bound on the minimum possible value of parameter d and so important ones in this respect. So, we now proceed to find out the submatrix of *R*, say R_P, made up of all columns of *R* and containing those rows j of *R* for which the coefficients c_j of the parameter d are positive. Let $c_{i_1}, c_{i_2}, \cdots, c_{i_k}$ are all and are only positive real numbers in the rows collected in R_P given below and all other coefficients of d in other rows of *R* are greater than or equal to zero.

$$R_P = \begin{bmatrix} a_{i_1 1} & a_{i_1 2} & \cdots & a_{i_1 n} & b_{i_1 1} & b_{i_1 2} & \cdots & b_{i_1 m} & c_{i_1} d + e_{i_1} \\ a_{i_2 1} & a_{i_2 2} & \cdots & a_{i_2 n} & b_{i_2 1} & b_{i_2 2} & \cdots & b_{i_2 m} & c_{i_2} d + e_{i_2} \\ a_{i_3 1} & a_{i_3 2} & \cdots & a_{i_3 n} & b_{i_3 1} & b_{i_3 2} & \cdots & b_{i_3 m} & c_{i_3} d + e_{i_3} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ a_{i_k 1} & a_{i_k 2} & \cdots & a_{i_k n} & b_{i_k 1} & b_{i_k 2} & \cdots & b_{i_k n} & c_{i_k} d + e_{i_k} \end{bmatrix}$$

If R_P is **empty** (i.e. containing not a single row) then the problem at hand is **unbounded**. There are certain columns starting from first column and appear in successions which are unit vectors. These columns which are unit vectors correspond to **basic variables**. The columns appearing in successions after these columns and not unit vectors correspond to **nonbasic variables**. As mentioned, among the columns of R_P for nonbasic variables those having all entries nonnegative can only lead to increase in the value of d when a positive value is assigned to them. This is undesirable as we aim minimization of the value of d. So, we desire to set the values of such variables equal to zero. When all columns corresponding to nonbasic variables in R_P are having all entries nonnegative and further if the inequality $\min\{d^-\} \geq \max\{d^+\}$ holds then we can set all nonbasic variables to zero and set $d = \max\{d^+\}$ in every row of *R* and find the basic feasible solution which will be **optimal**, with $\max\{d^+\}$ as **optimal value** for the objective function at hand. Still further, When all columns corresponding to nonbasic variables in R_P are having all entries nonnegative but $\min\{d^-\} < \max\{d^+\}$ then if $e_k > 0$ then we can still set all nonbasic variables to zero, set $d = \max\{d^+\}$ in every row of *R* and find the basic feasible solution which will be **optimal**, with max{ d^+ } as **optimal value** for the objective function at hand, i.e. if value of $e_k > 0$ in the expressions $c_k d + e_k$ in the rows of R rather those in R_P that are having value of $c_k > 0$ then we can proceed on similar lines to find optimal value for d.

In R_P we now proceed to consider those nonbasic variables for which the columns of R_P contain some (at least one) positive values and some negative (at least one) values. In such case when we assign some positive value to such nonbasic variable it leads to decrease in the value of d in those rows in which $c_k > 0$ and increase in the value of d in those rows in which $c_k < 0$. We now need to consider the ways of dealing with this situation. We deal with this situation as follows: In this case, we choose and carry out appropriate and legal elementary row transformations on the matrix R in the reduced row echelon form to achieve nonnegative value for all the entries in the columns corresponding to nonbasic variables in the submatrix R_P of R. The elementary row transformations are chosen to produce new matrix which remains equivalent to original matrix in the sense that the solution set of the matrix equation with original matrix and matrix equation with transformed matrix remain same. Due to this equivalence we can now set all the nonbasic variables in this transformed matrix to zero and obtain with justification $d_{\max} =$ max{ d^+ } as optimal value for the objective function and obtain basic feasible solution as optimal solution by substitution.

Algorithm 2.2 (Minimization):

1. Express the given problem in standard form:

   Maximize: $C^T x$

   Subject to: $Ax - s = b$

   $x \geq 0, s \geq 0$

2. Construct the augmented matrix [**E F**], where

   $$\mathbf{E} = \begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & -I_{(m\times m)} \end{bmatrix}, \text{ and } \mathbf{F} = \begin{bmatrix} d \\ b \end{bmatrix}$$

   and obtain the reduced row echelon form:

   R = rref (**[E, F]**)

3. If there is a row (or rows) of zeroes at the bottom of R in the first n columns and containing a nonzero constant in the last column then declare that the problem is **inconsistent** and stop. Else if the coefficients of d in the last column are all positive or if there exists a column of R corresponding to some nonbasic variable with all entries negative then declare that the problem at hand is **unbounded** and stop.

4. Else if for any value of d one observes that nonnegativity constraint for some variable gets violated by at least one of the variables then declare that the problem at hand is **infeasible** and stop.
5. Else find the submatrix of *R*, say R_P, made up of those rows of *R* for which the coefficient of d in the last column is positive.
6. Check whether the columns of R_P corresponding to nonbasic variables are nonnegative. Else, rearrange the constraint equations by suitably permuting these equations such that (as far as possible) the values in the columns of our starting matrix $A_{(m\times n)}$ get rearranged in the following way: Either they are rising and then becoming stationary, or falling and then becoming stationary, or falling initially up to certain length of the column vector and then rising again. After this rearrangement to form transformed $A_{(m\times n)}$ again proceed as is done in step 2 above to form its corresponding augmented matrix **[E, F]** and again find its *R* = rref (**[E, F]**) which will most likely have the desired representation, i.e. in the new R_P that one will construct from the new *R* will have columns for nonbasic variables which will be containing nonnegative entries.
7. Solve $c_{i_r}d + e_{i_r} = 0$ for each such a term in the last column of R_P and find the value of $d = d_{i_r}^-$ for $r = 1,2,\cdots,k$ and find $d_{\min}^- = \min\{d_{i_r}^-\}$. Similarly, solve $c_{i_r}d + e_{i_r} = 0$ for each such a term in the last column for rows of *R* other than those in R_N and find the values $d = d_{i_r}^+$ for $r = 1,2,\cdots,k$ and find $d_{\max}^+ = \max\{d_{i_r}^+\}$. Check the columns of R_P corresponding to nonbasic variables. If all these columns contain only nonnegative entries and if $\min\{d^-\} \geq \max\{d^+\}$ then set all nonbasic variables to zero. Substitute $d = d_{\max}$ in the last column of *R*. Determine the basic feasible solution which will be optimal and stop. Further, if columns corresponding to nonbasic variables contain only nonnegative entries and if $\min\{d^-\} < \max\{d^+\}$ then check whether value of $e_k > 0$ in the expressions $c_k d + e_k$ in these rows of *R* other those in R_P that are having value of $c_k > 0$. If yes, then set all nonbasic variables to zero. Substitute $d = d_{\max}$ in the last column of *R*. Determine the basic feasible solution which will be again optimal.
8. Even after proceeding with rearranging columns of our starting matrix $A_{(m\times n)}$ by suitably permuting rows representing constraint equations as is

done in step 6, and then proceeding as per step 7 and achieving the validity of $\min\{d^-\} \geq \max\{d^+\}$, if still there remain columns in R_P corresponding to some nonbasic variables containing positive entries in some rows and negative entries in some other rows then devise and apply suitable elementary row transformations on R such that the columns representing coefficients of nonbasic variables of new transformed matrix R or at least its submatrix R_P corresponding to new transformed R contain only nonnegative entries **so that** step 7 becomes applicable.

We now consider few examples for minimization problems:

Example 2.12: This example for minimization is like Example 2.6 for maximization in which the reduced row echelon form **contains by itself the basic variables** that are required to be present in the optimal simplex tableau, i.e. the tableau that results at the end of the simplex algorithm, for which only nonnegative entries occur in the bottom row of the tableau representing relative costs. This is understood by the nonnegativity of entries in the columns of R_P corresponding to nonbasic variables.

$$\text{Minimize: } -3x_1 + x_2 + x_3$$

$$\text{Subject to: } -x_1 + 2x_2 - x_3 \geq -11$$

$$-4x_1 + x_2 + 2x_3 \geq 3$$

$$2x_1 - x_3 \geq -1$$

$$-2x_1 + x_3 \geq 1$$

$$x_1, x_2, x_3 \geq 0$$

Solution: For this problem we have

$$R = \begin{bmatrix} 1, & 0, & 0, & 0, & -1, & 0, & 1, & -d+2 \\ 0, & 1, & 0, & 0, & -1, & 0, & 2, & 1 \\ 0, & 0, & 1, & 0, & -2, & 0, & 1, & 5-2d \\ 0, & 0, & 0, & 1, & 1, & 0, & 2, & 6+3d \\ 0, & 0, & 0, & 0, & 0, & 1, & 1, & 0 \end{bmatrix}$$

It is clear from separately equating each entry in the last column of *R* that the expected inequality, $\min\{d^-\} \geq \max\{d^+\}$, holds good. Also,

$$R_P = [\ 0, \ 0, \ 0, \ 1, \ 1, \ 0, \ 2, \ 6+3d]$$

Since all the entries in the columns corresponding to nonbasic variables in R_P are positive so we put $s_1, s_2, s_3, s_4 = 0$. Also, we put $d = \max(d^+) = -2$. By further substitutions, we have $x_1 = 4, x_2 = 1, x_3 = 9$.

Example 2.13: We now consider a minimization primal linear program for which **neither the primal nor the dual has a feasible solution**.
Minimize: $x-2y$
Subject to: $x-y \geq 2$
$-x+y \geq -1$
$x, y \geq 0$
Solution: For this problem we have

$R =$ [1, 0, 0, 2, 2-d]
[0, 1, 0, 1, 1-d]
[0, 0, 1, 1, -1]

Here, R_P is an empty matrix. So, there is no lower limit on the value of d. But, from the last row of R it is clear that (for any nonnegative value of nonbasic variable, s_2) the value of s_1 is negative and so the problem is thus **infeasible.** Similarly, if we consider the following dual, viz,

Maximize: $2x-y$
Subject to: $x-y \leq 1$
$-x+y \leq -2$
$x, y \geq 0$

then we have
$R =$ [1, 0, 0, 1, -2+d]
[0, 1, 0, 2, -4+d]
[0, 0, 1, 1, -1]

$R_N =$[1, 0, 0, 1, -2+d]
[0, 1, 0, 2, -4+d]
which implies a zero value for s_2 and $d = 2$. But again from the last row of R it is clear that (even for any nonnegative value of nonbasic variable, s_2) the value of y is negative and so the problem is **infeasible.**

Example 2.14: We now consider a minimization linear program for which the **primal is unbounded and the dual is infeasible**.
Minimize: $-x-y$
Subject to: $x-y \geq 5$
$x-y \geq -5$
$x, y \geq 0$
Solution: For this problem we have
$R =$ [1, 0, 0, -1/2, -5/2-(1/2)d]
[0, 1, 0, 1/2, (-1/2)d+5/2]

[0, 0, 1, -1, -10]

Clearly, R_p is empty so there is no bound to the value of d on the lower side and by giving value ≥ 10 to nonbasic variable s_2 we can avoid negativity of s_1, so the problem is unbounded. Also, let us apply following elementary transformations on *R*: Let us denote the successive rows of *R* by R(1), R(2), R(3), R(4). We change

(i) R(1) → R(1) +R(2), and
(ii) R(3) → 2R(2) +R(3)

$R =$ [1, 1, 0, 0, -d]
[0, 1, 0, 1/2, (-1/2)d+5/2]
[0, 1, 1, 0, -d -5]

So, by setting $s_2 = 0,\ d \leq -5$ we can check that this problem in **unbounded below.**

Now, if we consider the following dual, viz,

Maximize: $5x - 5y$

Subject to: $x + y \leq -1$

$$-x - y \leq -1$$

$$x, y \geq 0$$

We have

$R =$ [1, 0, 0, -1/2, 1/2+(1/10)d]
[0, 1, 0, -1/2, 1/2-(1/10)d]
[0, 0, 1, 1, -2]

$R_N =$ [0, 1, 0, -1/2, 1/2-(1/10)d]

Again from the last row of R it is clear that even for any nonnegative value of nonbasic variable, s_2 the value of s_1 is negative and so the problem is **infeasible.**

Example 2.15: Minimize: $x_1 + 4x_2 + 3x_4$

Subject to: $x_1 + 2x_2 - x_3 + x_4 \geq 3$

$$-2x_1 - x_2 + 4x_3 + x_1 \geq 2$$

$$x_1, x_2, x_3 \geq 0$$

We have following R, R_P for this problem:

$R =$ [1, 0, 0, -1, -16, -4, -7*d+56]
[0, 1, 0, 1, 4, 1, -14+2*d]
[0, 0, 1, 0, -7, -2, 25-3*d]

$R_P =$ [0, 1, 0, 1, 4, 1, -14+2*d]

Since it is clear from separately equating each entry in the last column of R that the expected inequality, $\min\{d^{-}\} \geq \max\{d^{+}\}$, holds good. Also, since all the entries in the columns corresponding to nonbasic variables in R_P are positive so we put $x_4, s_1, s_2 = 0$. Also, we put $d = \max(d^{+}) = 7$. By further substitutions, we have $x_1 = 7, x_2 = 0, x_3 = 4$.

Remark 2.1: It is clear from the discussion made so far that our aim should be to achieve the situation in which **all** the columns corresponding to nonbasic variables in R_N, R_P contain nonnegative entries. This situation corresponds to directly having the possession of maximal/minimal basic feasible solution. By achieving this one gets directly the optimal basic feasible solution by simply setting all the nonbasic variables to zero and finding the basic solution. This is the situation in a sense of having directly the optimal simplex tableau for which one sets all the nonbasic variables to zero as they in fact lead to decrement/increment in the otherwise maximal/minimal value of d for maximization/minimization linear programming problems under consideration. Thus, we can find the solution of any maximization/minimization linear program by properly analyzing the R, R_N, R_P matrices, and taking the relevant actions as per the outcomes of the analysis.

Remark 2.2: Instead of shuffling rows of matrix A, b to form new **E, F** without changing the content of the problem, we can also achieve the same effect of rearrangement of constraint by simply shuffling rows of identity matrix $I_{m \times m}$ in **E** and proceed with formation of new **E, F** without changing the content of the problem and achieving same effect in the sense that the corresponding reduced row echelon form will automatically produce same effect.

Remark 2.3: The nonnegativity of the entries that are present in the columns of nonbasic variables of the concerned reduced row echelon form R_N or R_P is in effect similar to obtain the optimal simplex tableau, i.e. the tableau at which the simplex algorithm terminates and where the basic feasible solution represents the optimal solution.

3. **A New Algorithm for Nonlinear Programming:** We now proceed show that we can deal with **nonlinear programs** (nonlinear constrained optimization problems) using the same above given technique used to deal with linear programs. The algorithms developed by **Bruno Buchberger** which transformed the abstract notion of **Grobner basis** into a fundamental tool in computational algebra will be utilized. The technique of Grobner bases is essentially a **version** of reduced row echelon form (used above to handle the linear programs made up of linear polynomials) for **higher degree** polynomials [7]. A typical nonlinear program can be stated as follows:
   Maximize/Minimize: $f(x)$
   Subject to: $h_j(x) = 0, j = 1,2,\cdots,m$

$$g_j(x) \geq 0, j = m+1, m+2, \cdots, p$$
$$x_k \geq 0, k = 1,2,\cdots,n$$

Given a nonlinear optimization problem we first construct the following nonlinear system of equations:

$$f(x) - d = 0 \tag{3.1}$$
$$h_j(x) = 0, j = 1,2,\cdots,m \tag{3.2}$$
$$g_j(x) + s_j = 0, j = m+1, m+2, \cdots, p \tag{3.3}$$

where d is the unknown parameter whose optimal value is to be determined subject to nonnegativity conditions on problem variables and slack variables. For this to achieve we first transform the system of equations into an equivalent system of equations bearing the same solution set such that the system is easier to solve. We have seen so far that the effective way to deal with linear programs is to obtain the reduced row echelon form for the combined system of equations incorporating objective equation and constraint equations. We will see that for the nonlinear case the effective way to deal with is to obtain the equivalent of reduced row echelon form, namely, the **Grobner basis representation** for this system of equations (3.1)-(3.3). We then set up the equations obtained by equating the partial derivatives of d with respect to problem variables x_i and slack variables s_i to zero and utilize the standard theory and methods used in **calculus**. We demonstrate the essence of this method by solving certain examples. where d is the unknown parameter whose optimal value is to be determined subject to nonnegativity conditions on problem variables and slack variables. For this to achieve we first transform the system of equations into an equivalent system of equations bearing the same solution set such that the system is easier to solve. We have seen so far that the effective way to deal with linear programs is to obtain the reduced row echelon form for the combined system of equations incorporating objective equation and constraint equations. We will see that for the nonlinear case the effective way to deal with is to obtain the equivalent of reduced row echelon form for the set of polynomials, namely, the **Grobner basis representation** for this system of equations (3.1)-(3.3). We then set up the equations obtained by equating the partial derivatives of d with respect to problem variables x_i and slack variables s_i to zero and utilize the standard theory and methods used in **calculus**. We demonstrate the essence of this method by solving an example:
These examples are taken from [8], [9]. These examples sufficiently illustrate the power of this new method of using powerful technique of Grobner basis to successfully and efficiently deal with nonlinear programming problems.

Example 3.1: Maximize: $-x_1^2 + 4x_1 + 2x_2$

Subject to: $x_1 + x_2 \leq 4$

$$2x_1 + x_2 \leq 5$$
$$-x_1 + 4x_2 \geq 2$$

Solution: We build the following system of equations:

$$-x_1^2+4x_1+2x_2-d=0$$
$$x_1+x_2+s_1-4=0$$
$$2x_1+x_2+s_2-5=0$$
$$-x_1+4x_2-s_3-2=0$$

such that: $x_1,x_2,s_1,s_2,s_3 \geq 0$

We now transform the nonlinear/linear polynomials on the left hand side of the above equations by obtaining Grobner basis for them as follows:

$$486-81d-18s_2-16s_2^2+36s_3-8s_2s_3-s_3^2=0 \quad (3.1.1)$$
$$9-9s_1+5s_2-s_3=0 \quad (3.1.2)$$
$$-9+s_2-2s_3+9x_2=0 \quad (3.1.3)$$
$$-18+4s_2+s_3+9x_1=0 \quad (3.1.4)$$

Setting $\frac{\partial d}{\partial s_2}=0$ and $\frac{\partial d}{\partial s_3}=0$ we get equations:

$$32s_2+8s_3=-18$$
$$8s_2+2s_3=36$$

a rank deficient system. Note that for maximization of d if we set $\frac{\partial d}{\partial s_2}=0$ we get the value of s_2 that maximizes d, namely, $s_2=-(18/32)-(8s_3/32)$, a negative value for any nonnegative value of s_3. So, we set $s_2=0$. Similarly, for maximization of d if we set $\frac{\partial d}{\partial s_3}=0$ we get the value of s_3 that maximizes d, namely, $s_3=18-4s_2(=18)$, setting $s_2=0$. But, by setting $s_2=0$ in the second equation above the largest possible value for s_3 that one can have (is obtained by setting $s_1=0$ and it) is **9**, when $s_2=0$. Thus, setting $s_2=0, s_3=9$ in the first equation we get d = **9**. From third and fourth equation we get $x_2=3, x_1=1$.

Example 3.2: Maximize: $-8x_1^2-16x_2^2+24x_1+56x_2$

Subject to: $x_1+x_2 \leq 4$

$$2x_1+x_2 \leq 5$$
$$-x_1+4x_2 \geq 2$$
$$x_1,x_2 \geq 0$$

Solution: We build the following system of equations:

$$-8x_1^2-16x_2^2+24x_1+56x_2-d=0$$

$$x_1 + x_2 + s_1 - 4 = 0$$
$$2x_1 + x_2 + s_2 - 5 = 0$$
$$-x_1 + 4x_2 - s_3 - 2 = 0$$

We now transform the nonlinear/linear polynomials on the left hand side of the above equations by obtaining Grobner basis for them as follows:

$$504 - 9d + 8s_2 - 16s_2^2 + 56s_3 - 8s_3^2 = 0 \qquad (3.2.1)$$
$$9 - 9s_1 + 5s_2 - s_3 = 0 \qquad (3.2.2)$$
$$-9 + s_2 - 2s_3 + 9x_2 = 0 \qquad (3.2.3)$$
$$-18 + 4s_2 + s_3 + 9x_1 = 0 \qquad (3.2.4)$$

from first equation (3.2.1), in order to maximize d, we determine the values of s_2, s_3 as follows:

If we set $\frac{\partial d}{\partial s_2} = 0$ we get the value of s_2 that maximizes d, namely, $s_2 = \frac{1}{4}$.

Similarly, if we set $\frac{\partial d}{\partial s_3} = 0$ we get the value of s_3 that maximizes d, namely, $s_3 = \frac{7}{2}$. Putting these values of s_2, s_3 in the first and second equation we get respectively the maximum value of $d = 67$ and the value of $s_1 = \frac{3}{4}$. Using further these values in the third and fourth equation we get $x_1 = 1.5, x_2 = 1.75$.

Example 3.3: Minimize: $(x_1 - 3)^2 + (x_2 - 4)^2$

Subject to: $2x_1 + x_2 = 3$

Solution: We form the objective equation and constraint equations as is done in the above examples and then find the Grobner basis which yields:

$$5x_1^2 - d - 2x_1 + 10 = 0$$
$$2x_1 + x_2 - 3 = 0$$

Setting $\frac{\partial d}{\partial x_1} = 0$ we get the value of x_1 that minimizes d, namely, $x_1 = 0.2$.

This yields $d = 9.8$ and $x_2 = 2.6$

Example 3.4: Minimize: $x_1^2 - x_2$

Subject to: $x_1 + x_2 = 6$

$x_1 \geq 1$

$$x_1^2 + x_2^2 \leq 26$$

Solution: We form the objective equation and constraint equations as is done in the above examples and then find the Grobner basis which yields:

$$\begin{aligned} 2d - 14s_1 - s_2 + 8 &= 0 \\ x_2 + s_1 - 5 &= 0 \\ x_1 - s_1 - 1 &= 0 \\ -s_2 - 8s_1 + 2s_1^2 &= 0 \end{aligned}$$

For minimizing d we should set the values of s_1, s_2 equal to zero (as they have signs opposite to d) which yields $d = -4$. From other equations we get $x_1 = 1, x_2 = 5$.

Example 3.5: Minimize: $-6x_1 + 2x_1^2 - 2x_1x_2 + 2x_2^2$

Subject to: $x_1 + x_2 \leq 2$

Solution: We form the objective equation and constraint equations as is done in the above examples and then find the Grobner basis which yields:

$$6x_2^2 - d + 6x_2s_1 - 6x_2 + 2s_1^2 - 2s_1 - 4 = 0 \quad (3.5.1)$$

$$x_1 + x_2 + s_1 - 2 = 0 \quad (3.5.2)$$

Setting $\frac{\partial d}{\partial x_2} = 0$ and $\frac{\partial d}{\partial s_1} = 0$ we get equations:

$$\begin{aligned} 12x_2 + 6s_1 &= 6 \\ 6x_2 + 4s_1 &= 2 \end{aligned}$$

There solution gives $s_1 = -1$, which is forbidden so we first set $s_1 = 0$ in the initial equations (3.5.1) and (3.5.2) and again set $\frac{\partial d}{\partial x_2} = 0$ which yields the value of x_2 that minimizes d, namely, $x_2 = \frac{1}{2}$. This in turn produce $x_1 = \frac{3}{2}$ and $d = -\frac{11}{2}$.

4. **A New Algorithm for Integer Programming:** We now proceed to deal with **integer programs** (integer programming problems) using the same above given technique used to deal with linear programs. The essential difference in this case is that we need to obtain integer optimal solution. A typical integer program is just like a linear program having a linear objective function to be optimized and the optimal solution to be determined should satisfy linear constraints, and nonnegativity constraints, and in addition, imposed **integrality of values for certain variable**. When all problem variables are required to be integers the problem is called **pure integer** program, when only certain variables are needed

to be integers while certain others can take nonintegral values the problem is called **mixed integer** program. When all variables can only take 0 or 1 values the problem is called **pure (0-1)** integer program. When only certain variables are needed to satisfy take 0 or 1 value the problem is called **mixed (0-1)** integer program. This additional integrality condition on the problem variables makes it extremely difficult to solve. The optimal solution obtained by relaxing the integrality conditions and by treating it as a linear program is called **LP-relaxation** solution. There are two main **exact methods** to solve integer programs: The **branch and bound** method and the **cutting plane** method but unfortunately they both have **exponential** time complexity. We will be propossing **Two New Methods** to deal with integer programs.

Two types of integer programming problems are:

1. Maximize: $C^T x$

   Subject to: $Ax \le b$

   $x \ge 0$, and integers.

   Or

2. Minimize: $C^T x$

   Subject to: $Ax \ge b$

   $x \ge 0$, and integers.

Where x is a column vector of size n×1 of unknowns.

Where C is a column vector of size n×1 of profit (for maximization problem) or cost (for minimization problem) coefficients, and C^T is a row vector of size 1×n obtained by matrix transposition of C.

Where A is a matrix of constraints coefficients of size m×n.

Where b is a column vector of constants of size m×1 representing the boundaries of constraints.

By introducing the appropriate slack variables (for maximization problem) and surplus variables (for minimization problem), the above mentioned linear programs gets converted into **standard form** as:

Maximize: $C^T x$

Subject to: $Ax + s = b$ (4.1)

$x \ge 0, s \ge 0$ and integers.

Where s is slack variable vector of size m×1.

This is a **maximization problem**.

Or

Minimize: $C^T x$

Subject to: $Ax - s = b$ (4.2)

$x \ge 0, s \ge 0$ and integers.

Where s is surplus variable vector of size m×1.

This is a **minimization problem**.

We begin (as done previously) with the following equation:

$$C^T x = d \tag{4.3}$$

where d is an **unknown parameter**, and call it **objective equation.** The (parametric) plane defined by this equation will be called **objective plane**. Let C^T be a row vector of size 1×n and made up of integer components $c_1, c_2, \cdots, c_n$, not all zero. It is clear that the objective equation will have integer solutions if and only if gcd (greatest common divisor) of $c_1, c_2, \cdots, c_n$ divides d . We discuss first the **maximization problem**. A similar approach for **minimization problem** can be developed on similar lines.

Given a maximization problem**,** we first construct the combined system of equations containing the objective equation and the equations defined by the constraints imposed by the problem under consideration, combined into a single matrix equation, viz.,

$$\begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & I_{(m\times m)} \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = \begin{bmatrix} d \\ b \end{bmatrix} \tag{4.4}$$

and obtain **LP-relaxation solution**. This LP-relaxation solution provides the upper bound that must be satisfied by the optimal integer solution. Then we proceed to form and solve a system of **Diophantine equations** as follows: In order to solve this system as Diophantine system of equations we use the standard technique given in ([10], pages 212-224). First by appending new variables $u_1, u_2, \cdots, u_{(m+n)}$ and carrying out appropriate **row and column transformations** discussed in ([10], pages 217, 221) we obtain the **parametric solutions** for the system. Thus, we start with the following **table**:

$$\begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} & d \\ A_{(m\times n)} & I_{(m\times m)} & b \\ & & \\ I_{((m+n)\times(m+n))} & & \end{bmatrix} \tag{4.5}$$

and transform the system of equations into an equivalent system that is diagonal. Thus, we have the following **parametric solution**:

$u_k = d$ (for some k)

$u_{i_r} = h_{i_r}$ (where h_{i_r} are constants for $r = 1$ to n , $i_r \neq k$), and

$x_i = \sum_{r=1}^{n} \alpha_{ij_r} u_{j_r} + \delta_i$ (where α_{ij_r}, δ_i are constants.)

$s_i = \sum_{r=1}^{n} \beta_{ij_r} u_{j_r} + \eta_i$ (where β_{ij_r}, η_i are constants.)

Important Points:

(1) The parametric solution is in terms of variables u_k , u_{i_r} , x_i , and s_i out of these variables x_i , and s_i must satisfy nonnegativity and integrality for their values. Nonnegativity is not imposed on u_k , u_{i_r} but some of them are forced to satisfy integrality because of their built-in relation with variables x_i , and s_i on which requirement of fulfilling integrality condition in the statement of the problem.

(2) These Diophantine equations, representing parametric solution for x_i , and s_i , produce the following set of inequalities through imposed nonnegativity of variables x_i , and s_i . Thus,

(i) Nonnegativity of x_i produce following inequalities:

$$0 \leq \sum_{r=1}^{n} \alpha_{ij_r} u_{j_r} + \delta_i$$

(ii) Nonnegativity of x_i produce following inequalities:

$$0 \leq \sum_{r=1}^{n} \beta_{ij_r} u_{j_r} + \eta_i$$

(3) Our **aim** is to find **nonnegative integer values** for x_i such that **the objective function as a parameter, d , has optimal value.**

First Method (Maximization):

1) We first solve the underlying linear programming problem and find the related LP relaxation solution for the problem.

2) We rewrite the equations giving parametric solution such that in all equations the coefficients of parameter d has negative value, i.e. we keep those equations as they are for which the coefficient of parameter d have negative value and multiply those other equations by (−1) in which coefficients of parameter d are positive.

3) We rearrange further this new set of equations having negative value for parameter d such that all the variables x_i , s_i , u_i are on the left side while terms like $c_i d + e_i$ are on the right side of these equations.

4) From these equations we construct a table of rows and columns such that the number of rows are equal to number of equations (for x_i, and s_i), and number of columns equal to number of variables x_i, s_i, u_i and d.
5) We use this table to find the least upper bounds satisfied by various variables using the values of coefficients for the se variables in the table constructed in earlier step and using the obtained bound in the first step by LP relaxation solution on the value of parameter d.
6) We carry out search for the optimal value of parameter d over the range of allowed values and determine the optimal solution by comparison of the feasible values generated by searching over the finite set of values for variables offered by the least upper bound on the values of these variables.

We now proceed to illustrate the procedure by examples from [11]:

Example 4.1: Maximize: $-x_1 + 10x_2$

Subject to: $-x_1 + 5x_2 \le 25$

$$2x_1 + x_2 \le 24$$

$x_1, x_2 \ge 0$, and integers.

Solution: We first find the LP-relaxation optimal value, which is **58.636** for this problem. And the complete optimal solution is

(x_1, x_2, s_1, s_2) = (8.6364, 6.7273, 0, 0)

Thus, the upper limit for optimal value for integer program, say $d_{opt.}$, can be 58.

Starting with the table (4.5) mentioned above and carrying out the appropriate row-column transformations we get the following parametric solution:

$$u_1 = -d$$
$$u_3 = 25$$
$$u_4 = 24$$
$$x_1 = -d + 10u_2$$
$$x_2 = u_2$$
$$s_1 = -d + 5u_2 + 25$$
$$s_2 = 2d - 21u_2 + 24$$

After rewriting these equations as per the step2 and rearranging them as per step 3 we have:

$$x_1 - 10u_2 = -d$$
$$x_2 - u_2 = 0$$
$$s_1 - 5u_2 = -d + 25$$
$$-s_2 - 21u_2 = -2d - 24$$

From these equations we get the following table as per step 4:

x_1	x_2	s_1	s_2	u_1	u_2	$c_i d + e_i$
1	0	0	0	0	-10	$-d$
1	0	0	0	0	-1	0
0	0	1	0	0	-5	$-d+25$
0	0	0	-1	0	-21	$-2d-24$

Using the upper limit on the optimal value, we have $d = d_{opt.} = 58$. In order to maximize the value of parameter d the value of variable u_2 should be increased as much as possible. But, from the last row of table above we see that the maximum value that u_2 can take (to maintain nonnegativity of s_2) is **6**., so we put u_2 = 6 and see that the optimal solution for the integer program is
$d = 55, s_1 = 0, s_2 = 8, x_1 = 5, x_2 = u_2 = 6$.

Example 4.2: Maximize: $3x_1 + 2x_2 + 3x_3 + 4x_2 + x_5$

Subject to: $4x_1 + 3x_2 - 2x_3 + 2x_2 - x_5 \le 12$

$$2x_1 + 3x_2 + x_3 + 3x_2 + x_5 \le 15$$
$$3x_1 + 2x_2 + x_3 + 2x_2 + 5x_5 \le 20$$
$$2x_1 + 4x_2 + x_3 + 6x_2 + x_5 \le 25$$
$$x_3 \le 3$$

All $x_1, x_2, \cdots, x_5 \ge 0$, and integers.

Solution: As per step 1 in the above algorithm for integer programs, we first find the LP-relaxation optimal value, which is **26.20** for this problem. And the complete optimal solution is

($x_1, x_2, x_3, x_4, x_5, s_1, s_2, s_3, s_4$) = (4.0723, 0, 3.0041, 1.1146, 0.5100, 0, 0, 0, 6.6536)

As per step 2, we find the parametric solution equations, which are:

$$u_5 = d$$
$$u_6 = 12$$
$$u_7 = 15$$
$$u_8 = 20$$
$$u_9 = 25$$
$$u_{10} = 3$$
$$x_1 = u_1$$
$$x_2 = u_2$$
$$x_3 = u_3$$

$$x_4 = u_4$$
$$x_5 = -3x_1 - 2x_2 - 3x_3 - 4x_4 + d$$
$$s_1 = -7x_1 - 5x_2 - x_3 - 6x_4 + d + 12$$
$$s_2 = x_1 - x_2 + 2x_3 + x_4 - d + 15$$
$$s_3 = 12x_1 + 8x_2 + 14x_3 + 18x_4 - 5d + 20$$
$$s_4 = x_1 - 2x_2 + 2x_3 - 2x_4 - d + 25$$
$$s_5 = -x_3 + 3$$

After rewriting these equations as per the step2 and rearranging them as per step 3 we get the following table as per step 4:

x_1	x_2	x_3	x_4	x_5	s_1	s_2	s_3	s_4	s_5	$c_i d + e_i$
-3	-2	-3	-4	-1	0	0	0	0	0	$-d$
-7	-5	-1	-6	0	-1	0	0	0	0	$-d-12$
-12	-8	-14	-18	0	0	0	1	0	0	$-5d+20$
-1	1	-2	1	0	0	1	0	0	0	$-d+15$
-1	2	-2	2	0	0	0	0	1	0	$-d+25$
0	0	1	0	0	0	0	0	0	1	3

Using the LP relaxation bound on parameter d we can easily determine the least upper bound on variables as follows:

$x_1 \leq 5, x_2 \leq 2, x_3 \leq 4, x_4 \leq 6, s_5 \leq 3$.

As per step 6, we try and find the following integral solution:

$(x_1, x_2, x_3, x_4, x_5, s_1, s_2, s_3, s_4 s_5)$ = (3, 0, 3, 2, 0, 2, 0, 4, 4,0), which produces the value of d = 26, which is optimal!

Second Method (Maximization):

1) As is done in previous method we first solve the underlying linear programming problem and find the related LP relaxation solution for the problem.
2) As is done in previous method in step 3, we rearrange this new set of equations such that all the variables x_i, s_i, u_i are on the left side while terms like $c_i d + e_i$ are on the right side of these equations.
3) Now we process the system of equations to eliminate the parameter d from all equations and arrive at a set of Diophantine Equations.
4) We then form **dual problem of given linear programming problem** and proceed identically to form equation (4.4), table (4.5) and obtain parametric solution for the dual problem and as is done in previous method in step 3, we rearrange this new set of equations such that all the variables x_i, s_i, u_i are

on the left side while terms like $c_i d + e_i$ are on the right side of these equations for dual problem.

5) Now we process this system of equations for dual problem to eliminate the parameter d from all equations and arrive at a set of Diophantine Equations.

6) We now solve this combined system of primal and dual problem and find out nonnegative integral solution which produce value **closest** to LP relaxation solution $= C^T x = b^T w$.

Example 4.3: We consider here the same example above (Example 4.2). After rewriting the equations as per step 2 of the second method we get these equations in tabular form as follows, where last column represents the right hand side of these equations:

x_1	x_2	x_3	x_4	x_5	s_1	s_2	s_3	s_4	s_5	$c_i d + e_i$
3	2	3	4	1	0	0	0	0	0	d
7	5	1	6	0	1	0	0	0	0	$d+12$
-12	-8	-14	-18	0	0	0	1	0	0	$-5d+20$
-1	1	-2	1	0	0	1	0	0	0	$-d+15$
-1	2	-2	2	0	0	0	0	1	0	$-d+25$
0	0	1	0	0	0	0	0	0	1	3

Carrying out the following elementary row transformations, viz,
(i) R(3) → R(1)+R(3)
(ii) R(4) → 5R(1) + R(4)
(iii) R(5) → R(1) + R(5)

rows R(3), R(4), R(5) transform to

x_1	x_2	x_3	x_4	x_5	s_1	s_2	s_3	s_4	s_5	$c_i d + e_i$
2	3	1	3	1	0	1	0	0	0	15
3	2	1	2	5	0	0	1	0	0	20
2	4	1	6	1	0	0	0	1	0	25

It is easy to check that the integral solution obtained above, viz, $(x_1, x_2, x_3, x_4, x_5, s_1, s_2, s_3, s_4 s_5)$ = (3, 0, 3, 2, 0, 2, 0, 4, 4,0) satisfy these equations!

5. Conclusion: Condensing of the linear form (to be optimized) into a new parameter and developing the appropriate equations containing it is a useful idea. This idea is useful not only for linear programs but also for nonlinear as well as integer programs and provides new effective ways to deal with these problems.

Acknowledgements

The author is very thankful to Dr. M. R. Modak, Dr. P. S. Joag for useful discussions.

References

1. Hadley G., Linear Programming, Narosa Publishing House, New Delhi 110 017, Eighth Reprint, 1997.
2. Goldberg Jack L., Matrix Theory with Applications, McGraw-Hill, International Editions, 1992.
3. E. M. L. Beale, Cycling in the Dual Simplex Algorithm, Naval Research Logistics Quarterly, 2, pages 269-275, 1955.
4. Klee V., Minty G., How Good is the Simplex Algorithm?, Inequalities III, O. Shisha (ed.), Academic Press, pages 159-175, New York, 1972.
5. Borgwardt K. H., The Average Number of Pivot Steps required by the Simplex-Method is Polynomial, Zeitschrift fur Operations Research, 26, pages 157-177, 1982.
6. Smale S., On the Average Speed of the Simplex Method, Mathematical Programming, 27, pages 241-262, 1983.
7. William W. Adams, Philippe Loustaunau, An Introduction to Grobner Bases, Graduate Studies in Mathematics, Volume 3, American mathematical Society.
8. Ravindran A., Phillips D. T., Solberg J. J., Operations Research Principles and Practice, John Wiley & Sons, Pvt. Ltd., 2000.
9. Williams H. P., Model Solving in Mathematical Programming, John Wiley & Sons, Pvt. Ltd., 1993.
10. Niven I., Zuckerman H., Montgomery H., An Introduction to the Theory of Numbers, John Wiley & Sons, Pvt. Ltd., 1991.
11. Wolfe C. S., Linear Programming with Basic and Fortran, Reston Publishing Company, Inc., A Prentice-Hall Company, 1985.