6 MULTIPLEX ALGORITHM FOR BOUNDED VARIABLES

6.1 Introduction

Linear Programming has been applied to problems as diverse as scheduling the operations of a multinational oil company to selecting strategies for a defence organisation.

It has also been applied to such problems as inventory control, personnel assignment, compensation planning, traffic and transportation management, location of facilities and blending grain feed mixtures etc. This list merely suggests the range of real life applications, in which some of them may have bounded variables.

6.2 Revised Simplex Method and Bounded Variable Problem

The mathematical model of a bounded variable linear programming problem may be generalised as

\[
\text{Extremize } Z = CX \\
\text{subject to } (A,I)X = P_0 \\
L \leq X \leq U
\]

where

\[ L = \begin{bmatrix} l_1 \\ l_2 \\ \vdots \\ l_{n+m} \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n+m} \end{bmatrix} \]

and \( U \geq L \geq 0 \).
The above model can be transformed into an equivalent LP problem with a mixed set of constraints. But this will lead to further addition of rows and columns which will require quite a bit of computertime and space for solution.

The size of the problem can be reduced considerably by the use of special techniques which will bring the number of constraints to the original set of resource constraints desired in the problem.

The lower bounds on the variables may be accounted for by resorting to a mere translation of the form

\[ X = L + X_{\text{sur}} \]

will introduce a new variable \( X_{\text{sur}} \) which is nonnegative. The real problem occurs only with upper bound variables. A substitution similar to the above may not be possible since there is no guarantee that \( X = U - x_{\text{sla}} \) will remain non-negative. This difficulty is overcome by using a special procedure called bounded variable simplex algorithm.

Considering a problem without the lower bounds

Extremize \( Z = CX \),

subject to

\[ (A, I)X = P_0 \]
\[ X + X_{\text{sla}} = U \]

and \( X \geq 0 \).

(6.2.2)
Instead of introducing the constraints $X+X_{\text{sl}} = U$ in the simplex tableau, their effect will be accounted by modifying suitably the feasibility criterion of the simplex method. The concept behind the modification of the feasibility criterion is based on the premise that not only a negative value for a variable makes it infeasible but also when it exceeds its upper bound. The non-negativity condition is treated exactly as in the conventional simplex method. The upper bound condition requires that a basic variable should become nonbasic at its upper bound. Also, when a nonbasic variable is selected to enter the basis, its entering value should not exceed its upper bound.

6.2.1 New Feasibility Criterion

Simplex method requires that the solution at the end of each iteration should be feasible. Let $x_j$, a non-basic variable enter the basis. Introducing $x_j$ into the basis gives

$$(X_B)^i = (X_B)^* - a_i^j x_j$$  \hspace{1cm} (6.2.3)

where $a_i^j$ is the $i$-th element of $a^j = B^{-1}p_j$ and $p_j$ is the vector of $(A,I)$ corresponding to $x_j$.

As per the conditions stipulated above, $x_j$ remains feasible if

$$0 \leq x_j \leq \mu_j$$  \hspace{1cm} (6.2.4)

and $(X_B)^i$ remains feasible if
0 \leq (x_B^*)_i - \alpha_i^j x_j \leq \mu_i \quad i = 1, 2, \ldots, m \quad (6.2.5)

Since the introduced variable must be non-negative, condition (6.2.4) will be taken care of by checking it for its upper bound. To meet condition (6.2.5) it should satisfy

\[(x_B)_i = (x_B^*)_i - \alpha_i^j x_j \geq 0\]

If \(\alpha_i > 0\), then it may cause \((x_B)_i\) to assume negative value. Let \(\Theta_1\) represent the maximum value of \(x_j\), resulting from this situation. Then

\[\Theta_1 = \min_i \frac{(x_B^*)_i}{\alpha_i^j}, \quad \alpha_i^j > 0 \quad (6.2.6)\]

This is the feasibility condition of the conventional simplex method.

In order to make sure that \((x_B)_i\) will not exceed its upper bound, it is necessary that

\[(x_B)_i = (x_B^*)_i + (-\alpha_i^j)x_j \leq \mu_i\]

This condition can be violated if \(\alpha_i^j\) is negative, Let \(\Theta_2\) represent the maximum value of \(x_j\) resulting from this condition. Then
\( \theta_2 = \min_i \left[ \frac{\mu_i - (x^*_B)_i}{-\alpha_i^j}, \quad \alpha_i^j < 0 \right] \quad (6.2.7) \)

Let \( \theta_0 \) denote the maximum value of \( x_j \) which does not violate any of the above conditions. Then

\[ \theta_0 = \min \left[ \theta_1, \theta_2, \mu_j \right] \quad (6.2.8) \]

It is noticed that an old basic variable \((x_B)_i\) can become non-basic only if the introduction of the entering variable \( x_j \) at level \( \theta_0 \) causes \((x_B)_i\) to be at zero level or at its upper bound.

After selecting the leaving variable using the above condition (6.2.8), the changes in the current basic solution can be effected as follows.

Let \((x_B)_r\) be the variable corresponding to \( \theta_0 = \min(\theta_1, \theta_2, \mu_j) \). Then

i. If \( \theta_0 = \theta_1 \), \((x_B)_r\) is dropped from the solution and \( x_j \) is introduced using the regular row operations of the simplex method.

ii. If \( \theta_0 = \theta_2 \), \((x_B)_r\) is dropped and \( x_j \) is introduced then \((x_B)_r\) being non-basic at its upper bound must be substituted out by using

\[ (x_B)_r = u_r - (x_B)_r' \]

iii. If \( \theta_0 = \mu_j \), \( x_j \) is substituted at its upper bound \((\mu_j - x_j)\) which remains nonbasic.
6.2.2 Step by Step Procedure

The various steps involved in implementing the bounded variable algorithm are as follows:

Step 1: If any of the variables has a lower bound, substitute it at its lower bound.

Step 2: Determination of the entering vector $P_j$:

$$ (z_j - c_j) = C_B B^{-1} P_j - c_j $$

$$ = (1, C_B B^{-1}) \begin{bmatrix} -c_j \\ P_j \end{bmatrix} \quad (6.2.9) $$

The most promising vector enters the basis as long as the solution is nonoptimal.

Step 3: Determination of the leaving vector $P_r$: when the entering vector $P_j$ and the current basis matrix is $B_c$, the leaving vector must correspond to

$$ \theta_0 = \min (\theta_1, \theta_2, \mu_j) \quad (6.2.10) $$

where

$$ \theta_1 = \min_k \left[ \frac{B_c^{-1} P}{\alpha_{kj}}, \alpha_{kj} > 0 \right] $$

$$ \theta_2 = \min_k \left[ \frac{\mu_k - (X_B)_k}{-\alpha_{kj}}, \alpha_{kj} < 0 \right] $$

Step 4: Depending on the minimum value the solution is changed as described in Section 6.2.1.

Step 5: Go to step 2.
6.3 *Modified Multiplex Algorithm*

The transpose of the A matrix of the decision variables is used to select a set of linearly independent columns as was done in the case of unbounded linear programming problems.

In addition to the m columns in the 'θ' matrix, there are two more columns due to the lower and upper bounds of the variables. The lower bound column could have been omitted in the 'θ' matrix by substituting out the lower bound values in the linear programming problem. However, the addition of this column in the 'θ' matrix avoids the computations corresponding to the initial transformation. This matrix is used to select a set of linearly independent columns.

6.3.1 *Formation of 'θ' matrix*

The 'θ' matrix formation follows the same steps as explained in Chapter 3. However, by selecting a basis without using 'θ' matrix for the first pass only reduces the computation of 'θ' matrix at the first instant. Instead of selecting the initial basis arbitrarily a heuristic approach by taking the transpose of A matrix described in Chapter 3, is employed.

For the subsequent passes, the 'θ' matrix constructed by taking the transpose of $B^{-1}A$ as per modified multiplex algorithm is used.
Let the \( \Theta \) matrix using the transpose of the constraint matrix, \( A \), be

\[
\begin{bmatrix}
1 & 2 & \ldots & i & \ldots & m & m+1 & m+2 \\
x_1 & a_{11} & a_{21} & \ldots & a_{i1} & \ldots & a_{m1} & 1 & u_1 \\
x_2 & a_{12} & a_{22} & \ldots & a_{i2} & \ldots & a_{m2} & 2 & u_2 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots \\
x_j & a_{1j} & a_{2j} & a_{ij} & a_{mj} & j & u_j \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
x_n & a_{1n} & a_{2n} & a_{in} & a_{mn} & n & u_n
\end{bmatrix}
\]

Where the last two columns contain the lower and upper bound values of the bounded variables respectively.

The selection of a set of entering vectors involves the following steps:

**Step 1:** The \( \Theta \) matrix is scanned row after row and the position of the greatest element in each row is identified. (If there is a tie it is broken arbitrarily).

If the greatest element happens to be \( a_{ij} \) then calculate the intercept \( b_i/a_{ij} \).

**Step 2:** The first entering vector will be the most promising vector in the set, say the, \( j \)-th vector and the
corresponding row in the 'AT' matrix, be k. This is the entering vector in the place of a vector corresponding to the column in which the maximum element of 'AT' lies in the 'θ' matrix, say, i-th column. The above is true if the calculated intercept lies between the upper and lower bounds in that row.

ii. If the calculated intercept is greater than the upperbound $u_j$ for that row, then the $j$-th vector will enter at its upper bound.

iii. If the calculated intercept is lower than the lower bound $l_j$ for that row, then the $j$-th vector will enter at its lower bound.

Step 3: Examine whether any other row has its maximum in the i-th column. All such rows as well as the k-th row and i-th column are deleted.

Step 4: Steps 1 to 3 are repeated until all the rows are deleted.

A typical θ matrix at the starting pass for a problem, say,

Max $Z = 4x_1 + 4x_2 + 3x_3$

subject to

$$
\begin{bmatrix}
-1 & 2 & 3 \\
0 & -1 & 1 \\
2 & 1 & -1 \\
1 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\leq
\begin{bmatrix}
15 \\
4 \\
6 \\
10
\end{bmatrix}
$$
and

\[ 0 \leq x_1 \leq 8 \]
\[ 0 \leq x_2 \leq 4 \]
\[ 2 \leq x_3 \leq 4 \]

\( \Omega \) matrix

<table>
<thead>
<tr>
<th></th>
<th>( s_1 )</th>
<th>( s_2 )</th>
<th>( s_3 )</th>
<th>( s_3 )</th>
<th>( L )</th>
<th>( U )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-4)</td>
<td>(-4)</td>
<td>(-4)</td>
<td>(2)</td>
<td>(1)</td>
<td>(0)</td>
<td>(8)</td>
</tr>
<tr>
<td>(-3)</td>
<td>(-3)</td>
<td>(-3)</td>
<td>(1)</td>
<td>(2)</td>
<td>(2)</td>
<td>(4)</td>
</tr>
</tbody>
</table>

The maximum in the first row corresponds to \( a_{31} \).

The intercept \( = b_3/a_{31} = 6/2 = 3 \).

and this lies between L.B and U.B and hence \( x_1 \) enters and \( s_3 \) leaves.

For \( x_2 \) and \( x_3 \) the calculated intercepts are \((b_1/a_{12} \text{ and } b_1/a_{13})\) \(15/2\) and \(15/3\). They are greater than the U.Bs and hence \( x_2 \) and \( x_3 \) enter at their upper bounds.

The illustrative examples in the following sections give the step by step procedure.

6.4 Illustrative Examples

Example 1:

Max \( Z = 4x_1 + 4x_2 + 3x_3 \)

such that
Pass 1
Step 1:
\((z_j - c_j)\): \((-4, -4, -3)\)

Step 2
\(B^{-1}p_j = 1\)
\(B^{-1}p_o = p_o\)

Step 3  \(\Theta\) matrix

<table>
<thead>
<tr>
<th></th>
<th>(s_1)</th>
<th>(s_2)</th>
<th>(s_3)</th>
<th>(s_4)</th>
<th>L.B</th>
<th>U.B.</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4 (x_1)</td>
<td>-1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>-4 (x_2)</td>
<td>2</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>-3 (x_3)</td>
<td>3</td>
<td>1</td>
<td>-1</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Intercept for the 1st row = \(6/2 = 3\)
Intercept for 2nd row = \(15/2 = 7.5\)
Intercept for 3rd row = \(15/3 = 5\)

Hence
\(x_1\) enters and \(s_3\) leaves and
\(x_2\) and \(x_3\) enter at their upper bounds.
Step 4 (For $x_1$)

\[
\eta_{\text{old}} = \begin{bmatrix}
-4 \\
-1 \\
2 \\
1
\end{bmatrix}
\]

\[
\eta_{\text{new}} = \begin{bmatrix}
2 \\
1/2 \\
0 \\
1/2 \\
-1/2
\end{bmatrix}
\]

\[
M^{-1} = \begin{bmatrix}
1 & 0 & 0 & 2 & 0 \\
0 & 1 & 0 & 1/2 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1/2 & 0 \\
0 & 0 & 0 & -1/2 & 0
\end{bmatrix}
\]

Entering $x_2$ and $x_3$ at their upper bounds new objective function is

\[
\text{Max } Z = 4x_1 + 4(4-x_2^1) + 3(4-x_3^1)
\]

\[
= 4x_1 - 4x_2^1 - 3x_3^1 + 28
\]

such that

\[
\begin{bmatrix}
-1 & -2 & -3 \\
0 & 1 & -1 \\
2 & -1 & 1 \\
1 & 1 & -2
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2^1 \\
x_3^1
\end{bmatrix}
= \begin{bmatrix}
-5 \\
4 \\
6 \\
6
\end{bmatrix}
\]

Pass 2:

Step 1: $(z_1-c_1) = \begin{bmatrix}
1 & 0 & 0 & 2 & 0
\end{bmatrix}
= 0$

$(z_2-c_2) = \begin{bmatrix}
1 & 0 & 0 & 2 & 0
\end{bmatrix}
= 2$
\[(z_3-c_3) = [1 \ 0 \ 0 \ 2 \ 0] \begin{bmatrix} 3 \\ -3 \\ -1 \\ 1 \\ -2 \end{bmatrix} = 5\]

and

\[B^{-1}p_0 = \begin{bmatrix} 1 & 0 & 1/2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & -1/2 & 1 \end{bmatrix} \begin{bmatrix} -5 \\ 4 \\ 6 \\ 6 \end{bmatrix} = \begin{bmatrix} -2 \\ 4 \\ 3 \\ 3 \end{bmatrix}\]

\[x'_2\] enters \(s_1\) leaves.

\[B^{-1}p_2 = \begin{bmatrix} 1 & 0 & 1/2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & -1/2 & 1 \end{bmatrix} \begin{bmatrix} -2 \\ 1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -5/2 \\ 1 \\ -1/2 \\ 3/2 \end{bmatrix}\]

\[\eta_{old} = \begin{bmatrix} 2 \\ -5/2 \\ -1 \\ 3/2 \end{bmatrix}\]

\[\eta_{new} = \begin{bmatrix} 4/5 \\ -2/5 \\ -1/5 \\ -3/5 \end{bmatrix}\]

\[\eta_{new}^{-1} = E_0 M_{current}^{-1}\]

\[\eta_{new}^{-1} = \begin{bmatrix} 1 & 4/5 & 0 & 0 & 0 \\ 0 & -3/4 & 0 & 0 & 0 \\ 0 & 2/5 & 0 & 0 & 0 \\ 0 & -1/5 & 0 & 1 & 0 \\ 0 & 3/5 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 2 & 0 \\ 0 & 1 & 0 & 1/2 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & -1/2 & 1 \end{bmatrix}\]

\[= \begin{bmatrix} 1 & 4/5 & 0 & 12/5 & 0 \\ 0 & -2/5 & 0 & -1/5 & 0 \\ 0 & 2/5 & 1 & 2/5 & 0 \\ 0 & -1/5 & 0 & 2/5 & 0 \\ 0 & 3/5 & 0 & -1/5 & 1 \end{bmatrix}\]
\[ M^{-1} \begin{bmatrix} 0 \\ P_0 \end{bmatrix} = \begin{bmatrix} 1 & 4/5 & 0 & 12/5 & 0 \\ 0 & -2/5 & 0 & -1/5 & 0 \\ 0 & 2/5 & 1 & 1/5 & 0 \\ 0 & -1/5 & 0 & 2/5 & 0 \\ 0 & 3/5 & 0 & -1/5 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -5 \\ 4 \\ 6 \\ 6 \end{bmatrix} \]

\[ x_2^* = 4/5 \]
\[ x_1 = 17/5 \]

solution is
\[ x_2 = x_2^* - x_1^* = 4 - 4/5 = 16/5 \]
\[ x_3 = 4 \]
\[ x_1 = 17/5 \]
\[ Z = 68/5 + 64/5 + 12 = 38 \frac{2}{5} \]

**Example 2**

To illustrate the dual multiplex algorithm (bounded variables) (Hamdy, A. Taha - 'Operations Research - An Introduction').

Maximise \( X_0 = 3x_1 + 5x_2 + 2x_3 \)

subject to
\[
\begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leq \begin{bmatrix} 14 \\ 23 \end{bmatrix}
\]

\[ 0 \leq x_1 \leq 4 \]
\[ 2 \leq x_2 \leq 5 \]
\[ 0 \leq x_3 \leq 5 \]
Pass 1

Step 1: \((z_j - c_j) = (-3, -5, -2)\)

Step 2: 
\[
B^{-1}p_0 = P_0 \\
B^{-1}p_j = A
\]

Step 3:

<table>
<thead>
<tr>
<th></th>
<th>(s_1)</th>
<th>(s_2)</th>
<th>L.B</th>
<th>U.P.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-3 \ x_1)</td>
<td>1</td>
<td>[2]</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>(-5 \ x_2)</td>
<td>2</td>
<td>[4]</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>(-2 \ x_3)</td>
<td>3</td>
<td>[3]</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

\(x_1, x_2, x_3\) enter at their upper bounds.

Step 4:

\[
M^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad P_0 = \begin{bmatrix} -6 \\ -14 \end{bmatrix}
\]

Solution is infeasible and the problem becomes

Maximize 
\[
Z = -3x_1^I - 5x_2^I - 2x_3^I + 43
\]

subject to

\[
\begin{bmatrix} -1 & -2 & -2 \\ -2 & -4 & -3 \end{bmatrix} \begin{bmatrix} x_1^I \\ x_2^I \\ x_3^I \end{bmatrix} \leq \begin{bmatrix} -6 \\ -14 \end{bmatrix}
\]

Applying the degraded version of the dual multiplex bounded variable algorithm, \(s_2\) leaves

\[
\begin{bmatrix} x_1^I \\ x_2^I \\ x_3^I \end{bmatrix} \leftarrow \begin{bmatrix} 3/2 \\ 5/4 \\ 2/4 \end{bmatrix}
\]

\[
\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}
\]

\[
\begin{bmatrix} -6 \\ -14 \end{bmatrix}
\]

\[
\begin{bmatrix} -1 & -2 & -2 \\ -2 & -4 & -3 \end{bmatrix}
\]
i. $x_3^i$ enters and the value $x_3^i = \frac{-14}{3} = 4.67$ which is greater than its upper bound. Hence $x_3^i$ enters at its upper bound.

ii. $x_2^i$ enters and $x_2^i = \frac{14}{4} = 3.5$ which is greater than its upper bound. Hence $x_2^i$ enters at its upper bound.

iii. $x_1^i$ enters at its upper bound.

Step 5:

1. $x_3^i$ enters at its upper bound $P_0 = \begin{bmatrix} 0 \\ -5 \end{bmatrix}$
2. $x_2^i$ enters at its upper bound $P_0 = \begin{bmatrix} 6 \\ 7 \end{bmatrix}$

$s_2$ becomes positive. Hence $X_2^i$ is allowed to enter at an intermediate level.

$$M^{-1} = \begin{bmatrix} 1 & 0 & 5/4 \\ 0 & 1 & -1/2 \\ 0 & 0 & -1/4 \end{bmatrix}$$

Pass 2:

Step 1: $(z_j - c_j) = [0.5, 0.0, 7/4]$

Step 2: $B^{-1}p_0 = \begin{bmatrix} s_1 \\ x_2^i \end{bmatrix} = \begin{bmatrix} 2.5 \\ 1.75 \end{bmatrix}$

solution is feasible and optimal.

Hence

$$x_1 = 4$$
$$x_2 = x_2 - x_2^i = 4 - 1.25 = 3.75$$
$$x_3 = x_3 - x_3^i = 3.5 - 0$$
$$Z = 3 + 4 + 5 	imes 3.75 = 30.75$$
Table 6.1 Comparison of Simplex bounded and Modified multiplex bounded Algorithms.

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>No of Variables</th>
<th>No of Constraints</th>
<th>Objective</th>
<th>Simplex bounded</th>
<th>Modified multiplex bounded</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>412</td>
<td>100</td>
<td>Max.</td>
<td>313.03</td>
<td>22.32</td>
</tr>
<tr>
<td>2</td>
<td>462</td>
<td>150</td>
<td>Max.</td>
<td>373.20</td>
<td>30.83</td>
</tr>
<tr>
<td>3</td>
<td>700</td>
<td>100</td>
<td>Max.</td>
<td>487.00</td>
<td>43.03</td>
</tr>
</tbody>
</table>

TIME IN SEC.
6.5 Computational Results

Modified multiplex bounded algorithm has been found to be far superior to the revised simplex bounded algorithm, as may be witnessed from Table 6.1. This shows that for a given problem, the computing time required per constraint of the modified multiplex bounded algorithm is much less than that of the simplex.

The modified multiplex version for bounded problems has also yielded a 12 times faster rate of convergence as compared to that in the simplex version.

6.6 Conclusions

A modified multiplex algorithm for solving bounded variable problem has been suggested. To select more number of variables while trying to remove infeasibility dual multiplex algorithm has been used. Few bounded variable problems have given encouraging results. Some numerical examples are also given in support of this modified algorithm in this chapter.