5 NUMERIC EXAMPLES TO ILLUSTRATE MULTIPLEX ALGORITHM

5.1 INTRODUCTION

In the previous two chapters, the step by step multiplex algorithm for the solution of linear programming problem was discussed. In this chapter a few numerical examples have been worked out to illustrate the working of the multiplex algorithm. The solution of linear programming problems involves the selection of a pair of variables to improve the solution from iteration to iteration. This is the basic philosophy of Dantzig's simplex procedure. The reason for changing only a single vector in the basis at each iteration is that no method has been found to change more than one vector at a time that is less time consuming than changing only a single vector[28]. The decision rule presented in this thesis is a technique for selecting a set of variables to examine the validity of the above statement. This is primarily the reason for naming the technique as multiplex algorithm.
The theoretically best decisive rule for making any such selection should embody the following characteristics:

i. It should be computationally simple,

ii. It should tend to minimize the number of iterations required to obtain an optimal solution and

iii. It should be computationally efficient on the basis of time.

The proposed multiplex algorithm seems to possess the above characteristics.

5.2 CRITERIA FOR SELECTING VARIABLES

For the selection of variables, there are three decision rules depending upon the order of increasing computational complexities. They are:

i. Select the very first variable whose contribution $(z_j - c_j)$ is of proper sign

ii. Select the variable with $(z_j - c_j)$ of proper sign and largest in magnitude

iii. Select that variable which will give the greatest change (in the direction of steepest ascent or descent depending on the objective) in the value of the objective function for the immediate iteration.
In general, selection of variables at random will lead to many more iterations than a more purposeful selection, as has been pointed out by Dantzig[16,17]. Selecting the first acceptable variable is similar to selecting at random, with a potential advantage, if the matrix is logically ordered prior to computation. Improvement in efficiency will be provided by either of the other two rules (ii) or (iii). Since the third rule involves several multiply/divide operations on each variable being considered and since most computers perform multiplication/division much more slowly than other arithmetic operations, this rule has been hitherto condemned in favour of the second as a compromise. It is shown in this thesis that the third decision rule applied in a proper way is more efficient in some of the cases and at least as efficient as the second rule in most of the cases.

5.3 EXPERIENCE IN APPLICATION

Four packages in Fortran were developed to run the problem in the IBM System 370/Model 155-II computer. One package adopted the conventional maximum rate of change criterion to select a variable to enter the basis evicting a variable from the basis. The second
package incorporated the multiplex algorithm which brought into the basis a set of variables, sending out an equal number. The third package is for the conventional bounded variable algorithm and the last package is for the multiplex bounded variable algorithm.

Experience with the multiplex selection procedure in solving practical problems has been rewarding. While solving a steel rolling problem which has 17 constraints and 31 decision variables, it was noticed that the number of iterations (basis changes) required for this problem by the conventional procedure was 26, whereas the multiplex algorithm solved the same problem in 3 passes (16 basis changes). This confirms the observation made by Hadley[28] that a linear programming problem is solvable theoretically in as many iterations as there are constraints; but the simplex procedure is incapable of accomplishing this since variables keep popping in and out of the basis. Another problem had 43 constraints and 372 decision variables. It took 119 iterations (basis changes) by the conventional method whereas the multiplex algorithm took only 90 basis changes.
A third problem had 22 constraints and 96 variables. Although it took 25 iterations (basis changes) in the conventional procedure and 6 passes (22 basis changes) by the multiplex algorithm, the saving in computational effort was appreciable.

A few small problems were also solved and the step by step computations using the multiplex algorithm are shown. It was difficult to log the processing time precisely for text book examples. It was also observed that the same problem took different processing times run at different points in time. This difference may be due to other overheads of the system and therefore the computational efficiency of the multiplex algorithm should not be based on text book examples. Examples are worked out to illustrate the various aspects of the multiplex algorithm.

i. Example 1 : To illustrate the working of the multiplex algorithm in a step by step manner and the three different methods.

ii. Example 2 : To illustrate situations when a problem seeks optimal solution at the end of first pass.
iii. Example 3: To show how the multiplex algorithm drastically cuts down the number of iterations.

iv. Example 4: To show a problem in which the solution converges exactly in m iterations where m is the number of constraints.

v. Example 5: To show how the multiplex algorithm chooses always a set of linearly independent columns.

vi. Example 6: To illustrate how the dependent columns are avoided even though they are selected using 'Θ' matrix.

vii. Example 7: To show the method of working of dual multiplex algorithm (unbounded variables).

viii. Example 8: To illustrate the multiplex algorithm for bounded variables.

ix. Example 9: To illustrate the degraded version of the dual multiplex algorithm (bounded variables).

5.4 NUMERIC EXAMPLES

5.4.1 Example 1

Max \( Z = 10x_1 + 5x_2 + x_3 \)

subject to

\[
\begin{bmatrix}
2 & 1.1 & -1 \\
4 & 6 & 2 \\
-4 & -4 & 3
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\leq
\begin{bmatrix}
15 \\
60 \\
5
\end{bmatrix}
\]

\( x_j \geq 0 \quad j = 1, 2, 3 \)
Method 1
Pass 1:
Step 1: \((z_j - c_j) = [-10, -5, -1]\)
Step 2: \([a] = [B^{-1}P_j]\)
\[
\begin{bmatrix}
2 & 1.1 & -1 \\
4 & 6 & 2 \\
-4 & -4 & 3
\end{bmatrix}
\]
\([B^{-1}P_0]\) = \[
\begin{bmatrix}
15 \\
60 \\
5
\end{bmatrix}
\]
Step 3:

<table>
<thead>
<tr>
<th>(z_j - c_j)</th>
<th>(s_1)</th>
<th>(s_2)</th>
<th>(s_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-10 (x_1)</td>
<td>7.5</td>
<td>15</td>
<td>-</td>
</tr>
<tr>
<td>-5 (x_2)</td>
<td>15</td>
<td>10</td>
<td>-</td>
</tr>
<tr>
<td>-1 (x_3)</td>
<td>-</td>
<td>30</td>
<td>5/3</td>
</tr>
</tbody>
</table>

i. \(x_1\) has the most negative \((z_j - c_j)\) and hence \(x_1\) enters and \(s_1\) leaves

ii. \(x_2\) enters and \(s_2\) leaves

iii. \(x_3\) enters and \(s_3\) leaves.
Step 4: The vector for $x_1$ is

$$\begin{bmatrix}
5 \\
1/2 \\
-2 \\
2
\end{bmatrix}$$

Enter $\eta$ in column 2 of $M^{-1}$

$$i.e. \quad M^{-1} = \begin{bmatrix}
1 & 5 & 0 & 0 \\
0 & 1/2 & 0 & 0 \\
0 & -2 & 1 & 0 \\
0 & 2 & 0 & 1
\end{bmatrix}$$

Before entering $x_2$ check whether it is still promising,

$$(z_2 - c_2) = [1, 5, 0.0, 0.0] = [1.1, 6, -4]$$

Since it is positive, $x_2$ is not promising.

and hence it is left out. To enter $x_3$ find

$$(z_3 - c_3) = [1, 5, 0.0, 0.0] = [-1, 2, 3]$$
Since it is negative it is promising

\[ \eta = \begin{bmatrix} 6 \\ -1/2 \\ -4 \\ 1 \end{bmatrix} \]

Update the \( M^{-1} \) matrix using \( \eta \)

\[ M^{-1} = \begin{bmatrix} 1 & 17 & 0 & 6 \\ 0 & 3/2 & 0 & 1/2 \\ 0 & -10 & 1 & -4 \\ 0 & 2 & 0 & 1 \end{bmatrix} \]

Pass 2:

Step 1: \((z_j - c_j) = [0.0, -10.3, 0.0]\)

Step 2: \(\alpha = [B^{-1}p_j] = \begin{bmatrix} -0.35 \\ 11 \\ -4 \end{bmatrix} \)

\( [B^{-1}p_0] = \begin{bmatrix} 25 \\ -110 \\ 35 \end{bmatrix} \)

Step 3:

<table>
<thead>
<tr>
<th>[ \theta ]</th>
<th>( x_1 )</th>
<th>( s_2 )</th>
<th>( x_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_2 )</td>
<td>-</td>
<td>( \begin{bmatrix} 10 \end{bmatrix} )</td>
<td>-</td>
</tr>
</tbody>
</table>

1. \( x_2 \) enters and \( s_2 \) leaves
Step 4: \( \eta \) vector for \( x_2 \) is

\[
\begin{bmatrix}
10.3 \\
11 \\
0.35 \\
11 \\
1 \\
11 \\
4 \\
11
\end{bmatrix}
\]

Update the \( M^{-1} \) matrix using \( \eta \)

\[
M^{-1} =
\begin{bmatrix}
1 & 7.636 & .936 & 2.254 \\
0 & 1.181 & .0318 & .3727 \\
0 & -.909 & .0909 & -.3636 \\
0 & -1.636 & .3636 & -.4545
\end{bmatrix}
\]

Pass 3

Step 1: \( (z_j - c_j) = [0.0, 0.0, 0.0] \)

No more promising variable.

Step 2: Check the feasibility of the solution

\[
\begin{bmatrix}
21.4865 \\
-10 \\
-5
\end{bmatrix}
\]

The solution is infeasible.
Step 3: Apply the dual simplex algorithm. The leaving variable is $x_2$, which has more negative value i.e. (-10)

\[
\begin{bmatrix}
  x_1 & x_2 & x_3 & s_1 & s_2 & s_3 \\
  - & - & - & 7.636 & - & \boxed{2.254} \\
  - & - & - & -.909 & - & -.3636 \\
\end{bmatrix}
\]

i.e. $x_2$ leaves and $s_3$ enters.

Step 4: The $\eta$ vector is

\[
\begin{bmatrix}
  6.2 \\
  1.025 \\
  -2.75 \\
  -1.25 \\
\end{bmatrix}
\]

updating the $M^{-1}$ matrix using the $\eta$ vector.

\[
M^{-1} =
\begin{bmatrix}
  1 & 2 & 3/2 & 0 \\
  0 & 1/4 & 1/8 & 0 \\
  0 & 5/2 & -1/4 & 1 \\
  0 & -1/2 & 1/4 & 0 \\
\end{bmatrix}
\]

Pass 4: The solution

\[
B^{-1}p_0 =
\begin{bmatrix}
  11.25 \\
  27.5 \\
  7.5 \\
\end{bmatrix}
\]

is feasible and there are no more promising variables. Hence the optimum solution is
Method 2:

Pass 1 is the same as for the method 1.

Pass 2:

Step 1 : \((z_j - c_j) = [0.0, -10.3, 0.0]\)

Step 2 : 

\[
[a] = [B^{-1}p_j] = \begin{bmatrix}
-0.35 \\
11 \\
-4
\end{bmatrix}
\]

\[
[B^{-1}p_o] = \begin{bmatrix}
25 \\
-110 \\
35
\end{bmatrix}
\]

Step 3 : \([\emptyset] = x_2 \begin{bmatrix}
- & - & -
\end{bmatrix}\) since when \((B^{-1}p_j)^t\)_i is positive, \((B^{-1}p_o)^t\)_i is negative. Hence check the solution for feasibility. \(s_2 = -110\), hence it is infeasible. To remove the infeasibility as per the dual simplex algorithm, \(s_2\) is the leaving variable.
Step 4 : \( \eta \) vector for \( s_3 \) is

\[
\begin{bmatrix}
3/2 \\
1/8 \\
-1/4 \\
1/4
\end{bmatrix}
\]

Updating the \( M^{-1} \) matrix using \( \eta \) vector

\[
M^{-1} = \begin{bmatrix}
1 & 2 & 3/2 & 0 \\
0 & 1/4 & 1/8 & 0 \\
0 & 5/2 & -1/4 & 1 \\
0 & -1/2 & 1/4 & 0
\end{bmatrix}
\]

The solution is feasible, i.e.

\[
[B^{-1} P_0] = \begin{bmatrix}
11.25 \\
27.5 \\
7.5
\end{bmatrix}
\]

Pass 3 : \( (z_j - c_j) = [0.0, 6.2, 0.0] \)

No more promising variable is present and the solution is feasible. Hence, the optimum solution is reached.
Method 5:

Pass 1 is the same as in Method 1. At the end of Pass 1, check for feasibility of solution.

\[
[B^{-1}p_o] = \begin{bmatrix}
25 \\
-110 \\
35
\end{bmatrix}
\]

Since the solution is infeasible, the dual simplex algorithm is used to remove infeasibility.

Pass 2: As per the dual simplex algorithm \( s_2 \) is the leaving variable and \( s_3 \) is the entering variable.

\[
s_2 = \begin{bmatrix}
x_1 & x_2 & x_3 & s_1 & s_2 & s_3 \\
- & - & - & -17/10 & - & -6/4
\end{bmatrix}
\]

Step 4: The \( \eta \) vector for \( s_3 \) is

\[
\eta = \begin{bmatrix}
3/2 \\
1/8 \\
-1/4 \\
1/4
\end{bmatrix}
\]
Pass 3:
Step 1: Updating the \( M^{-1} \) matrix using the \( \eta \) vector

\[
M^{-1} = \begin{bmatrix}
1 & 2 & 3/2 & 0 \\
0 & 1/4 & 1/8 & 0 \\
0 & 5/2 & -1/4 & 1 \\
0 & -1/2 & 1/4 & 0 \\
\end{bmatrix}
\]

The solution \( B^{-1}p_0 = \begin{bmatrix} 11.25 \\ 27.5 \\ 7.5 \end{bmatrix} \) is feasible.

Pass 3:
Step 1: \((z_j - c_j) = [0.0, 6.2, 0.0]\)
No more promising variable is present. Hence the solution is optimal.

5.4.2 Example 2

\[
\text{Max } Z = 4x_1 + 3x_2 - x_3 + 2x_4 + 6x_5
\]
subject to

\[
\begin{bmatrix}
3 & 0 & 1 & 0 & -1 \\
-1 & -1 & 0 & 3 & 0 \\
0 & 1 & 1 & 0 & 1 \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
x_5 \\
\end{bmatrix}
\leq \begin{bmatrix}
5 \\
12 \\
4 \\
\end{bmatrix}
\]

\(x_j \geq 0; \ j = 1, 2, \ldots, 5\)
Pass 1:

Step 1: \((z_j - c_j) = [-4, -3, 1, -2, -6]\)

Step 2: \([a] = [B^{-1}P_j]\)

\[
\begin{bmatrix}
3 & 0 & 1 & 0 & -1 \\
-1 & -1 & 0 & 3 & 0 \\
0 & 1 & 1 & 0 & 1 \\
\end{bmatrix}
\]

\([B^{-1}P_0] = \begin{bmatrix}
5 \\
12 \\
4 \\
\end{bmatrix}\)

Step 3:

<table>
<thead>
<tr>
<th>(z_j - c_j)</th>
<th>(s_1)</th>
<th>(s_2)</th>
<th>(s_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>(x_1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3</td>
<td>(x_2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>(x_3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>(x_4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-6</td>
<td>(x_5)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. \(x_5\) has the most negative \((z_j - c_j)\) and hence \(x_5\) enters and \(s_3\) leaves

ii. \(x_1\) enters and \(s_1\) leaves

iii. \(x_4\) enters and \(s_2\) leaves
Step 4: The \( \eta \) vector for \( x_5 \) is

\[
\eta = \begin{bmatrix}
6 \\
1 \\
0 \\
1
\end{bmatrix}
\]

Updating the \( M^{-1} \) matrix using \( \eta \)

\[
M^{-1} = \begin{bmatrix}
1 & 0 & 0 & 6 \\
0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

The next variable to enter is \( x_1 \). Check whether it is still promising.

\[
(z_1 - a_1) = [1.0, 0.0, 0.0, 6] \begin{bmatrix}
-4 \\
3 \\
-1 \\
0
\end{bmatrix} = [4]
\]

\[
\eta = \begin{bmatrix}
1.33 \\
.33 \\
.33 \\
0.00
\end{bmatrix}
\]
Updating the $M^{-1}$ matrix using $\eta$

$$M^{-1} = \begin{bmatrix}
1 & 1.33 & 0 & 7.33 \\
0 & 0.33 & 0 & 0.33 \\
0 & 0.33 & 1 & 0.33 \\
0 & 0 & 0 & 1.00
\end{bmatrix}$$

The next variable to enter is $x_4$. Check whether it is still promising.

$$(z_4 - c_4) = [1.0, 1.33, 0.0, 7.33] \times \begin{bmatrix}
-2 \\
0 \\
3 \\
0
\end{bmatrix} = [-2]$$

The $\eta$ vector for $x_4$ is

$$\eta = \begin{bmatrix}
0.67 \\
0 \\
0.33 \\
0
\end{bmatrix}$$

Updating the $M^{-1}$ matrix using $\eta$

$$M^{-1} = \begin{bmatrix}
1 & 1.55 & 0.67 & 7.55 \\
0 & 0.33 & 0 & 0.33 \\
0 & 0.11 & 0.33 & 0.11 \\
0 & 0 & 0 & 1.00
\end{bmatrix}$$
Pass 2:

Step 1: \((z_j - c_j) = [0.0, 3.889, 10.1, 0.0, 0.0]\)

Optimality condition is satisfied and the solution is feasible. Hence the optimal solution is

\[
\begin{bmatrix}
  z \\
  x_1 \\
  x_4 \\
  x_5
\end{bmatrix} =
\begin{bmatrix}
  1 & 1.55 & 0.67 & 7.55 \\
  0 & 0.33 & 0 & 0.33 \\
  0 & 0.11 & 0.33 & 0.11 \\
  0 & 0 & 0 & 1.00
\end{bmatrix}
\begin{bmatrix}
  0 \\
  5 \\
  12 \\
  4
\end{bmatrix}
\]

\[
= \begin{bmatrix}
  46 \\
  3 \\
  5 \\
  4
\end{bmatrix}
\]

5.4.3 Example 3 [4]

Max \(Z = 0.4x_1 + 0.28x_2 + 0.32x_3 + 0.72x_4 + 0.64x_5 + 0.6x_6\)

subject to

\[
\begin{bmatrix}
  0.01 & 0.01 & 0.01 & 0.03 & 0.03 & 0.03 \\
  0.02 & 0 & 0 & 0.05 & 0 & 0 \\
  0 & 0 & 0.02 & 0 & 0 & 0.05 \\
  0 & 0 & 0 & 0.03 & 0 & 0 & 0.08
\end{bmatrix}
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  x_4 \\
  x_5 \\
  x_6
\end{bmatrix} \leq \begin{bmatrix}
  850 \\
  750 \\
  100 \\
  900
\end{bmatrix}
\]

\(x_j \geq 0; \quad j = 1, 2, \ldots, 6\)
Pass 1:

Step 1: \((z_j - c_j) = [-0.4, -0.28, -0.32, -0.72, -0.64, -0.6]\)

Step 2: \([a] = [B^{-1}P_j] = A
\)

\([B^{-1}P_0] = P_0\)

Step 3: \((z_j - c_j) \begin{array}{c|cccc|c}
  s_1 & s_2 & s_3 & s_4 & (z_j - c_j)x_j \\
-0.4 & x_1 & 85000 & 37500 & -- & -- & -15000 \\
-0.28 & x_2 & 85000 & -- & \underline{5000} & -- & -1400 \\
-0.32 & x_3 & 85000 & -- & -- & \underline{3000} & -9600 \\
-0.72 & x_4 & \underline{85000} & \frac{15000}{3} & -- & -- & -10800 \\
-0.64 & x_5 & \underline{85000} & \frac{2000}{3} & -- & -- & -1280 \\
-0.60 & x_6 & \underline{85000} & \frac{11250}{3} & -- & -- & -6750
\end{array}\)

If \(\bar{c}_jx_j\) criterion is used

i. \(x_1\) enters \(s_2\) leaves

ii. \(x_3\) enters \(s_4\) leaves

iii. \(x_2\) enters \(s_3\) leaves

Step 4: The \(M^{-1}\) matrix, after introducing the variables one after the other, is
Pass 2:

Step 1: 
\[(z_j - c_j) = [0.0, 0.0, 0.0, 0.28, 0.06, 0.253]\]

Optimality condition is satisfied and the solution vector is obtained in a single pass.

\[
\begin{bmatrix}
26000
125
37500
5000
30000
\end{bmatrix}
\]

Had the maximum rate of change criterion been used, it would have taken two passes.

5.4.4 Example 4

Minimize \(Z\)

\[
= 5454x_{11} + 5095x_{12} + 6025x_{13} + 6025x_{14} + 4600x_{15} \\
+ 4125x_{16} + 4125x_{17} + 3958x_{18} + 3980x_{19} + 4175x_{1.10} \\
+ 4430x_{1.11} + 4430x_{1.12} + 10422x_{21} + 9637x_{22} + 9333x_{23} \\
+ 8620x_{24} + 8540x_{25} + 8500x_{26} + 8440x_{27} + 8420x_{28} \\
+ 8380x_{29} + 8310x_{2.10} + 8310x_{2.11} + 8270x_{2.12} + 8270x_{2.13} \\
+ 8270x_{2.14} + 12430x_{34} + 11010x_{35} + 10400x_{36} + 10150x_{37} \\
+ 10150x_{38}
\]
subject to

\[ \frac{x_{11}}{20.7} + \frac{x_{12}}{23.8} + \frac{x_{13}}{40} + \frac{x_{14}}{40} + \frac{x_{15}}{30} + \frac{x_{16}}{40} + \frac{x_{17}}{40} + \frac{x_{18}}{45.3} + \frac{x_{19}}{44.5} + \frac{x_{2.10}}{38.7} + \frac{x_{2.11}}{36} + \frac{x_{2.12}}{36} \leq 600 \]

\[ \frac{x_{21}}{7.2} + \frac{x_{22}}{8.6} + \frac{x_{23}}{9.3} + \frac{x_{24}}{11.5} + \frac{x_{25}}{11.8} + \frac{x_{26}}{12} + \frac{x_{27}}{12.2} + \frac{x_{28}}{12.3} + \frac{x_{29}}{12.5} + \frac{x_{2.10}}{12.8} + \frac{x_{2.11}}{12.8} + \frac{x_{2.12}}{13} \]

\[ \frac{x_{3.13}}{13} + \frac{x_{2.14}}{13} \leq 600 \]

\[ \frac{x_{34}}{9} + \frac{x_{35}}{12} + \frac{x_{36}}{14} + \frac{x_{37}}{15} + \frac{x_{38}}{15} \leq 400 \]

\[ x_{11} + x_{21} = 5000 \]
\[ x_{12} + x_{22} = 2200 \]
\[ x_{13} + x_{23} = 5100 \]
\[ x_{14} + x_{24} + x_{34} = 5000 \]
\[ x_{15} + x_{25} + x_{35} = 4200 \]
\[ x_{16} + x_{26} + x_{36} = 3000 \]
\[ x_{17} + x_{27} + x_{37} = 2000 \]
\[ x_{18} + x_{28} + x_{38} = 1300 \]
\[ x_{19} + x_{29} = 750 \]
\[ x_{1.10} + x_{2.10} = 500 \]
\[ x_{1.11} + x_{2.11} = 500 \]
\[ x_{1.12} + x_{2.12} = 25 \]
\[ x_{2.13} = 100 \]
\[ x_{2.14} = 100 \]
Pass 1:

The set of variables selected to enter the basis and the corresponding variables to leave the basis are obtained from the 'Θ' matrix.

\[
\begin{align*}
    x_{18} & \rightarrow s_{11} \\
    x_{19} & \rightarrow s_{12} \\
    x_{16} & \rightarrow s_{9} \\
    x_{17} & \rightarrow s_{10} \\
    x_{1,10} & \rightarrow s_{13} \\
    x_{1,11} & \rightarrow s_{14} \\
    x_{1,12} & \rightarrow s_{15} \\
    x_{15} & \rightarrow s_{8} \\
    x_{12} & \rightarrow s_{5} \\
    x_{11} & \rightarrow s_{4} \\
    x_{13} & \rightarrow s_{6} \\
    x_{14} & \rightarrow s_{7} \\
    x_{2,13} & \rightarrow s_{16} \\
    x_{2,14} & \rightarrow s_{17} \\
    x_{21} & \rightarrow s_{2} \\
    x_{34} & \rightarrow s_{3}
\end{align*}
\]

But during the formation of $M^{-1}$ matrix, all the variables except $x_{21}$ and $x_{34}$ are entered because they
become nonbasic after the introduction of the other selected variables.

Pass 2:

The optimality condition is satisfied. But the solution is infeasible. Hence using the dual multiplex algorithm, variables $x_{21}$ and $x_{35}$ are selected to enter in place of $s_1$ and $s_2$ respectively.

These two variables are entered and the $M^{-1}$ matrix is updated.

Pass 3:

The solution is optimal and feasible. The optimal solution is given below.

$$
\begin{align*}
x_{11} & : 791.071 \\
x_{12} & : 2200 \\
x_{13} & : 5000 \\
x_{14} & : 5100 \\
x_{15} & : 376.464 \\
x_{16} & : 2000 \\
x_{17} & : 3000 \\
x_{18} & : 1300 \\
x_{19} & : 750
\end{align*}
$$
5.4.5 Example 5

Min \[ Z = -2.3x_1 - 1.6x_2 - 1.9x_3 - 1.4x_4 \]

subject to

\[
\begin{bmatrix}
20000 & 15000 & 20000 & 15000 \\
1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
\end{bmatrix}
\leq
\begin{bmatrix}
100000 \\
30 \\
20 \\
20 \\
\end{bmatrix}
\]

\[ x_j \geq 0, \quad j = 1, 2, 3, 4. \]

This example describes an application of linear programming to a probability problem [32]. Two types of missiles are available to fire at two types of targets. The number of missiles that would minimize the overall probability of failure is to be determined.
i.e., \( \text{Min } Q = q_1^x q_2^x q_3^x q_4^x \)

This is equivalent to

\[
\text{Min } Z = -2.3x_1 - 1.6x_2 - 1.9x_3 - 1.4x_4
\]

and the constraints being

i. There can be no more than 20 missiles of each type,

ii. No more than 30 missiles altogether can be expended on a given engagement and

iii. The maximum amount that can be spent on a given engagement is $1000000.$

The rank of the A matrix in the above example is 2 and there is an explicit set of linearly dependent columns. The step by step procedure given below explains how the algorithm always selects only a set of linearly independent columns.

Pass 1:

Step 1: \( [z_j - c_j] = [2.3, 1.6, 1.9, 1.4] \)

Step 2: \( [B^{-1}P_j] = A \)

\( [B^{-1}P_0] = P_0 \)
Step 3:

<table>
<thead>
<tr>
<th>$z_j - c_j$</th>
<th>$s_1$</th>
<th>$s_2$</th>
<th>$s_3$</th>
<th>$s_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3 $x_1$</td>
<td>50</td>
<td>30</td>
<td>20</td>
<td>--</td>
</tr>
<tr>
<td>1.6 $x_2$</td>
<td>66.67</td>
<td>30</td>
<td>--</td>
<td>20</td>
</tr>
<tr>
<td>1.9 $x_3$</td>
<td>50</td>
<td>30</td>
<td>20</td>
<td>--</td>
</tr>
<tr>
<td>1.4 $x_4$</td>
<td>66.67</td>
<td>30</td>
<td>--</td>
<td>20</td>
</tr>
</tbody>
</table>

i. $x_1$ enters and $s_3$ leaves the basis.

ii. $x_2$ enters and $s_4$ leaves the basis.

Step 4:

$$M^{-1} = \begin{bmatrix} 1 & 0 & 0 & -2.3 & -1.6 \\ 0 & 1 & 0 & 20000 & -15000 \\ 0 & 0 & 1 & -1 & -1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Pass 2:

Step 1: $(z_j - c_j) = [0.0, 0.0, -0.4, -0.2]$

Optimality is satisfied. Hence the solution is checked for feasibility. The solution vector is

$$\begin{bmatrix} Z \\ X_B \end{bmatrix} = \begin{bmatrix} -78 \\ -300000 \\ -10 \\ 20 \\ 20 \end{bmatrix}$$
Infeasibility is removed using dual simplex procedure. \( s_2 \) is the leaving variable and \( s_4 \) is the entering variable.

\[
\begin{bmatrix}
  x_1 & x_2 & x_3 & x_4 & s_1 & s_2 & s_3 & s_4 \\
\end{bmatrix}
\]

\[
s_2 = \begin{bmatrix}
  -1.6 & -0.7 & 0 \\
  1 & -15000 & -5000 & 0 \\
  0 & 1 & 1 & 1 \\
  0 & 0 & 1 & -1 & 0 \\
\end{bmatrix}
\]

Step 4 :

\[
M^{-1} = \begin{bmatrix}
  1 & 0 & -1.6 & -0.7 & 0 \\
  0 & 1 & -15000 & -5000 & 0 \\
  0 & 0 & 1 & 1 & 1 \\
  0 & 0 & 1 & -1 & 0 \\
\end{bmatrix}
\]

Pass 3 :

Step 1 : \([z_j - c_j] = [0.0, 0.0, -0.4, -0.2]\)

The solution is optimal and feasible and the solution vector is

\[
\begin{bmatrix} Z \\ s_1 \\ s_4 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -62 \\ 150000 \\ 10 \\ 20 \\ 10 \end{bmatrix}
\]

Thus the multiplex algorithm always selects a set of linearly independent columns.
Maximize \( Z = 10x_1 + 8x_2 + 6x_3 + 4x_4 \)

subject to
\[
\begin{bmatrix}
1 & 1 & 2 & 1 \\
2 & -3 & -1 & -1 \\
-1 & 2 & 1 & -1
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{bmatrix}
\leq
\begin{bmatrix}
10 \\
8 \\
8
\end{bmatrix}
\]

In this problem column 3 can be obtained as a linear combination of columns 1 and 2. In other words, column 3 is dependent on columns 1 and 2.

Pass 1:

Step 1 : \([z_j - c_j] = [-10, -8, -6, -4]\)

Step 2 : \([B^{-1}P_j] = A\)

\([B^{-1}P_0] = P_0\)

Step 3 : \(z_j - c_j\)

<table>
<thead>
<tr>
<th></th>
<th>(s_1)</th>
<th>(s_2)</th>
<th>(s_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-10</td>
<td>(x_1)</td>
<td>10</td>
<td>([4])</td>
</tr>
<tr>
<td>-8</td>
<td>(x_2)</td>
<td>10</td>
<td>--</td>
</tr>
<tr>
<td>-6</td>
<td>(x_3)</td>
<td>([5])</td>
<td>--</td>
</tr>
<tr>
<td>-4</td>
<td>(x_4)</td>
<td>([10])</td>
<td>--</td>
</tr>
</tbody>
</table>

i. \(x_1\) enters \(s_2\) leaves

ii. \(x_2\) enters \(s_3\) leaves

iii. \(x_3\) enters \(s_1\) leaves
Although column $x_3$ is a linear combination of $x_1$ and $x_2$, as seen from the A matrix, it is shown below how the dependent column is omitted while forming the $M^{-1}$ matrix.

Step 4:

$$\eta_1 = \begin{bmatrix} 5 \\ -1/2 \\ 1/2 \\ 1/2 \end{bmatrix}$$

Hence

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

$$M^{-1} \begin{bmatrix} -c_2 \\ P_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 5 & 0 \\ 0 & 1 & -1/2 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} -8 \\ 1 \\ -3 \\ 2 \end{bmatrix} = \begin{bmatrix} -23 \\ 5/2 \\ -3/2 \\ 1/2 \end{bmatrix}$$

$$\eta_2 = \begin{bmatrix} 46 \\ -5 \\ 3 \\ 2 \end{bmatrix}$$

Updating the $M^{-1}$ matrix,

$$M^{-1} = \begin{bmatrix} 1 & 0 & 28 & 46 \\ 0 & 1 & -3 & -5 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 1 & 2 \end{bmatrix}$$
\[
M^{-1} \begin{bmatrix}
-o_3 \\
P_3
\end{bmatrix} = \begin{bmatrix}
12 \\
0 \\
1 \\
1
\end{bmatrix}
\]

\(x_3\) becomes nonpromising. Even if it had been promising, the pivot element becomes zero. Hence, the dependent column eventhough may be selected in the 'O' matrix will be left out while forming the \(M^{-1}\) matrix because it will turn out to be nonpromising or the pivot element will become zero.

Pass 2:

Step 1: \((z_j - c_j) = [0.0, 0.0, 12, -78]\)

Step 2:
\[
B^{-1}P_j = \begin{bmatrix}
9 \\
-5 \\
-3
\end{bmatrix}
\]
\[
B^{-1}P_o = \begin{bmatrix}
-54 \\
40 \\
24
\end{bmatrix}
\]

Step 3:
\[
x_4 \begin{bmatrix}
s_1 \\
x_1 \\
x_2 \\
x_4
\end{bmatrix} = \begin{bmatrix}
-- \\
-- \\
--
\end{bmatrix}
\]
No more promising variable which will increase
the objective function exists and hence the
solution is checked for feasibility.
Infeasibility is removed by invoking the dual
simplex procedure. \( s_1 \) is the leaving variable
and \( s_3 \) is the entering variable.

\[
\begin{bmatrix}
 x_1 & x_2 & x_3 & x_4 & s_1 & s_2 & s_3 \\
 s_1 & - & - & - & - & -9.33 & -9.2
\end{bmatrix}
\]

Step 4:

\[
M^{-1} = \begin{bmatrix}
 1 & 46/5 & 2/5 & 0 \\
 0 & 2/5 & -1/5 & 0 \\
 0 & 3/5 & 1/5 & 0 \\
 0 & -1/5 & 3/5 & 1
\end{bmatrix}
\]

Pass 3:

Step 1: \((z_j - c_j) = [0.0, 0.0, 12, 4.8]\)

The solution is optimal and feasible and the
solution vector is

\[
\begin{bmatrix}
 z \\
 s_3 \\
 x_1 \\
 x_2
\end{bmatrix} = \begin{bmatrix}
 476/5 \\
 54/5 \\
 38/5 \\
 12/5
\end{bmatrix}
\]
5.4.7 Example 7

Maximize \( Z = -x_1 - 2x_2 - 4x_3 \)
subject to
\[
\begin{bmatrix}
2 & -1 & 1 \\
1 & 2 & -2 \\
-1 & 3 & 4
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\leq
\begin{bmatrix}
5 \\
10 \\
8
\end{bmatrix}
\]

The constraints can be rewritten as
\[
\begin{bmatrix}
-2 & 1 & -1 \\
-1 & -2 & 2 \\
+1 & -3 & -4
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\leq
\begin{bmatrix}
-5 \\
-10 \\
-8
\end{bmatrix}
\]

Pass 1:

Step 1: \((z_j - c_j) = [1, 2, 4]\)

Optimality condition is satisfied. But the solution is infeasible.

Step 2: \(B^{-1}p_j = A\)
\(B^{-1}p_0 = p_0\)

Step 3:

| \hline
| Entering variable | \(x_1\) | \(x_2\) | \(x_3\) |
| \hline
| Leaving variable | \(-5\) | \(-10\) | \(-8\) |
| \hline
| \(s_1\) | \(1/2\) | \(-4\) |
| \(s_2\) | \(1\) | \(4/3\) |
| \(s_3\) | \(-2/3\) | \(1\) |
| \hline

i. \(s_2\) leaves \(x_1\) enters

ii. \(s_3\) leaves \(x_2\) enters.
Step 4:

\[ \eta = \begin{bmatrix} 1 \\ -2 \\ -1 \\ 1 \end{bmatrix} \quad \text{Hence } M^{-1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & -2 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \]

\[ M^{-1} \begin{bmatrix} -a_2 \\ P_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & -2 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ -2 \\ -3 \end{bmatrix} = \begin{bmatrix} 0 \\ 5 \\ 2 \\ -5 \end{bmatrix} \]

\[ B^{-1}p_0 = \begin{bmatrix} 15 \\ 10 \\ -18 \end{bmatrix} \]

\( (B^{-1}p_0) \) in row 3 is negative and also the pivot element is negative. Hence

\[ \eta = \begin{bmatrix} 0 \\ 1 \\ 2/5 \\ -1/5 \end{bmatrix} , \quad M^{-1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & -1 & 1 \\ 0 & 0 & -3/5 & 2/5 \\ 0 & 0 & -1/5 & -1/5 \end{bmatrix} \]
Pass 2:

Step 1: \((z_j - c_j) \quad [0.0, 0.0, 6]\)

Step 2:

\[
B^{-1}p_0 = \begin{bmatrix} -3 \\ 14/5 \\ 18/5 \end{bmatrix}
\]

\(s_1\) is infeasible

\[
M^{-1} \begin{bmatrix} -c_1 \\ p_1 \end{bmatrix} = \begin{bmatrix} 6 \\ -7 \\ -14/5 \\ 2/5 \end{bmatrix}
\]

Step 3:

\[
x_3 \quad s_2 \quad s_3
\]

\(s_1\) leaves and \(x_3\) enters

Step 4:

\[
n := \begin{bmatrix} 6/7 \\ -1/7 \\ -2/5 \\ 2/35 \end{bmatrix}; \quad M^{-1} = \begin{bmatrix} 1 & 6/7 & 1/7 & 6/7 \\ 0 & -1/7 & 1/7 & -1/7 \\ 0 & -2/5 & -1/5 & 0 \\ 0 & 2/35 & -9/35 & -1/7 \end{bmatrix}
\]
Pass 3:

Step 1: \((z_j - c_j)\) \([-,-, 6/7, 1/3, 6/7]\)

Step 2: \(B^{-1}p_0 = \begin{bmatrix} 3/7 \\ 4 \\ 24/7 \end{bmatrix}\)

Solution is feasible and optimal.

\[
\begin{bmatrix}
Z \\
x_3 \\
x_1 \\
x_2
\end{bmatrix} = \begin{bmatrix}
-88/7 \\
2/7 \\
4 \\
24/7
\end{bmatrix}
\]

5.4.8 Example 8[37]

Maximize \[Z = 4x_1 + 4x_2 + 3x_3\]

subject to

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

\[0 \leq x_1 \leq 8\]

\[0 \leq x_2 \leq 4\]

\[2 \leq x_3 \leq 4\]
Pass 1:

Step 1: \((z_j - c_j) = [-4, -4, -3]\)

Step 2: \(E^{-1} P_j = A\)
\(E^{-1} P_0 = P_0\)

Step 3:

<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)</td>
<td>(x_1)</td>
<td>-</td>
<td>-</td>
<td>[3]</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>(-4)</td>
<td>(x_2)</td>
<td>7.5</td>
<td>-</td>
<td>6</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>(-3)</td>
<td>(x_3)</td>
<td>5</td>
<td>4</td>
<td>-</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

i. \(x_1\) enters \(s_3\) leaves

ii. \(x_2\) and \(x_3\) enter at their upper bound.

Step 4:
\[\eta = \begin{bmatrix} 2 \\ 1/2 \\ 0 \\ 1/2 \\ -1/2 \end{bmatrix} \quad \text{and} \quad M^{-1} = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 \\ 0 & 1 & 0 & 1/2 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & -1/2 & 1 \end{bmatrix}\]

Entering the variables \(x_2\) and \(x_3\) at their upper bound.
New objective function is \( Z = 4x_1 - 4x_2 - 3x_3 \)
subject to
\[
\begin{bmatrix}
-1 & -2 & -3 \\
0 & 1 & -1 \\
2 & -1 & 1 \\
1 & 1 & -2
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\leq
\begin{bmatrix}
-5 \\
4 \\
6 \\
6
\end{bmatrix}
\]

Solution is
\[
x_2 = x_2 - x_3 = 4 - \frac{4}{5} = \frac{16}{5}
\]
\[
x_3 = 4
\]
\[
x_1 = \frac{17}{5}
\]
\[
Z = \frac{68}{5} + \frac{64}{5} + 12 = \frac{38}{5}
\]

5.4.9 Example 9[29]

Maximize \( x_0 = 3x_1 + 5x_2 + 2x_3 \)
subject to
\[
\begin{bmatrix}
1 & 2 & 3 \\
2 & 4 & 3
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\leq
\begin{bmatrix}
14 \\
23
\end{bmatrix}
\]
\[0 \leq x_1 \leq 4\]
\[2 \leq x_2 \leq 5\]
\[0 \leq x_3 \leq 3\]

**Pass 1:**

**Step 1:** \((z_i - c_i) = [-3, -5, -2]\)

**Step 2:** \(B^{-1}P_0 = P_0\)
\[B^{-1}P_j = A\]

**Step 3:**

<table>
<thead>
<tr>
<th>(s_1)</th>
<th>(s_2)</th>
<th>LB</th>
<th>UB</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3 (x_1)</td>
<td>14</td>
<td>11.5</td>
<td>0</td>
</tr>
<tr>
<td>-5 (x_2)</td>
<td>7</td>
<td>5.75</td>
<td>2</td>
</tr>
<tr>
<td>-2 (x_3)</td>
<td>7</td>
<td>7.67</td>
<td>0</td>
</tr>
</tbody>
</table>

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

**Step 4:**

\[P^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\]
\[P_0 = \begin{bmatrix} -6 \\ -14 \end{bmatrix}\]
Solution is infeasible and the problem becomes

Maximize \( Z = -3x'_1 - 5x'_2 - 2x'_3 + 43 \)

subject to

\[
\begin{bmatrix}
-1 & -2 & -2 \\
-2 & -4 & -3
\end{bmatrix}
\begin{bmatrix}
x'_1 \\
x'_2 \\
x'_3
\end{bmatrix}
\leq
\begin{bmatrix}
-6 \\
-14
\end{bmatrix}
\]

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

\[
s_2
\begin{bmatrix}
3/2 & 5/4 & 2/4
\end{bmatrix}
\]

i. \( x'_3 \) enters and the value \( x'_3 = \frac{-14}{3} = 4.67 \)

which is greater than its upper bound.

Hence \( x'_3 \) enters at its upper bound.

ii. \( x'_2 \) enters and \( x'_2 = \frac{14}{4} = 3.5 \) which is greater than its upper bound. Hence \( x'_2 \) enters at its upper bound.

iii. \( x'_1 \) enters at its upper bound.

Step 5: i. \( x'_3 \) enters at its upper bound

\[
P_0 = \begin{bmatrix} 0 \\ -5 \end{bmatrix}
\]
ii. $x'_2$ enters at its upper bound

$$ P_0 = \begin{bmatrix} 6 \\ 7 \end{bmatrix} $$

$s_2$ becomes positive. Hence, $x'_2$ 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: $F^{-1}P_0 = \begin{bmatrix} a_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.5 \\ 1.25 \end{bmatrix}$

Solution is feasible and optimal. Hence

$x_1 = 4$
$x_2 = x_2 - x'_2 = 4 - 1.25 = 3.75$
$x_3 = x_3 - x'_3 = 3 - 3 = 0$
$z = 3 \times 4 + 5 \times 3.75$
$= 30.75$
5.5 BASIS FOR COMPARISON

Algorithms, in general, may have different computational efficiencies. Researchers many a times have a tendency to compare two or more algorithms based on the total number of iterations to find the optimal solution[19,40]. The authors experience has been totally different in this respect. Comparisons based on number of iterations and time for a few typical problems are furnished in Table 5.1. It may be observed that the computational savings obtained on the basis of the number of iterations is invariably greater than that obtained based on time. It is more rational to base the comparison either on time or on the number of multiply/divide operations performed than on number of iterations/passes. For, the number of iterations/passes is algorithm dependent whereas time or arithmetic operation is independent of algorithm employed. For any given problem, the multiplex algorithm has always been superior to the simplex on the basis of the number of multiply/divide operation as furnished in Table 5.2. However, the multiply/divide operations and the processing time of the multiplex algorithm do neither bear any proportion nor in consonance as may be witnessed from Tables 5.1 and 5.2. Table 5.3 gives a comparison of the processing times for bounded variable simplex and multiplex algorithms. The discrepancy between the processing time
## Table 5.1 Comparison of Simplex and Multiplex Algorithms

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Problem</th>
<th>Number of variables</th>
<th>Number of constraints</th>
<th>Objective</th>
<th>Time in Basis changes, Iterations</th>
<th>CPU Time</th>
<th>Basis changes</th>
<th>Passes/Iterations</th>
<th>CPU Time</th>
<th>Basis changes</th>
<th>Passes/Iterations</th>
<th>Percent saving</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Max</td>
<td>31</td>
<td>17</td>
<td>1.22</td>
<td>1.18</td>
<td>1.15</td>
<td>1.04</td>
<td>1.04</td>
<td>1.02</td>
<td>1.02</td>
<td>1.02</td>
<td>16.5</td>
</tr>
<tr>
<td>2</td>
<td>Min</td>
<td>31</td>
<td>17</td>
<td>1.18</td>
<td>1.11</td>
<td>1.07</td>
<td>1.02</td>
<td>1.02</td>
<td>1.08</td>
<td>1.02</td>
<td>1.08</td>
<td>13.5</td>
</tr>
<tr>
<td>3</td>
<td>Min</td>
<td>96</td>
<td>22</td>
<td>2.68</td>
<td>2.06</td>
<td>2.09</td>
<td>2.09</td>
<td>2.23</td>
<td>2.21</td>
<td>2.13</td>
<td>2.13</td>
<td>23.1</td>
</tr>
<tr>
<td>4</td>
<td>Max</td>
<td>372</td>
<td>43</td>
<td>30.83</td>
<td>30.64</td>
<td>32.01</td>
<td>18.26</td>
<td>21.58</td>
<td>34.26</td>
<td>17.2</td>
<td>17.2</td>
<td>44</td>
</tr>
<tr>
<td>5</td>
<td>Min</td>
<td>372</td>
<td>43</td>
<td>30.67</td>
<td>18.77</td>
<td>15.77</td>
<td>17.18</td>
<td>17.65</td>
<td>15.97</td>
<td>16.8</td>
<td>16.8</td>
<td>48</td>
</tr>
<tr>
<td>6</td>
<td>Min</td>
<td>928</td>
<td>74</td>
<td>30.67</td>
<td>18.77</td>
<td>15.77</td>
<td>17.18</td>
<td>17.65</td>
<td>15.97</td>
<td>16.8</td>
<td>16.8</td>
<td>48</td>
</tr>
</tbody>
</table>

CPU Time = CPU Time

Method 1 = Method 1

Method 2 = Method 2

Method 3 = Method 3

Percent saving = Percent saving
### Table 5.2 Comparison of simplex and multiplex algorithms on the basis of multiply/divide operations

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>No. of consts.</th>
<th>No. of vars.</th>
<th>Objective</th>
<th>Percentage sparsity</th>
<th>Simplex</th>
<th>Multiplex Criterion I</th>
<th>Multiplex Criterion II</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>6</td>
<td>Max.</td>
<td>50</td>
<td>147</td>
<td>114</td>
<td>63</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>16</td>
<td>Max.</td>
<td>25</td>
<td>1075</td>
<td>265</td>
<td>290</td>
</tr>
<tr>
<td>3</td>
<td>17</td>
<td>31</td>
<td>Max.</td>
<td>11.8</td>
<td>4725</td>
<td>1468</td>
<td>1686</td>
</tr>
<tr>
<td>4</td>
<td>17</td>
<td>31</td>
<td>Min.</td>
<td>11.8</td>
<td>5177</td>
<td>1299</td>
<td>1426</td>
</tr>
<tr>
<td>5</td>
<td>18</td>
<td>9</td>
<td>Max.</td>
<td>55.6</td>
<td>1259</td>
<td>775</td>
<td>637</td>
</tr>
<tr>
<td>6</td>
<td>43</td>
<td>372</td>
<td>Max.</td>
<td>4.65</td>
<td>106394</td>
<td>77761</td>
<td>39460</td>
</tr>
</tbody>
</table>
Table 5.3 Comparison of simplex bounded algorithm and multiplex bounded algorithm

<table>
<thead>
<tr>
<th>Sl. No</th>
<th>Problem</th>
<th>Time in Secs.</th>
<th>Percent saving</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of Variables</td>
<td>Number of Constraints</td>
<td>Objective</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>4</td>
<td>Max.</td>
</tr>
<tr>
<td>2</td>
<td>42</td>
<td>11</td>
<td>Max.</td>
</tr>
<tr>
<td>3</td>
<td>350</td>
<td>32</td>
<td>Max.</td>
</tr>
</tbody>
</table>
and the number of multiply/divide operations in the multiplex algorithm may be due to such phenomena as 'Cycle stealing', 'Swapping', 'Page fault', 'thrashing' etc., which occur in a multiprogramming mode of operation. It is very difficult to make a sharp contrast in such an environment. Another interesting observation was that the same problem gave different processing times processed at different points in time. Though the multiplex algorithm does not prove to be so efficient on the basis of processing time compared to the arithmetic operation, it enjoys uniformly the reduction in the number of multiply/divide operations. It is the author's contention that any comparison made on the basis of the number of iterations is more an illusion and is misleading and as such it should not be chosen as a basis at all. This is one of the important conclusions arrived at in this dissertation.

Of equal or greater importance is the increase in digital accuracy that can be enjoyed when a reduction in the number of iterations is achieved. Rounding error is a common plague of matrix algebra codes. Troubles with arithmetic overflow or underflow have been greatly reduced through the use of the multiplex algorithm.
FIG. 5.1. PROCESSING TIME VERSUS NUMBER OF CONSTRAINTS

SIMPLEX

MULTIPLEX

PROCESSING TIME IN SECS

NUMBER OF CONSTRAINTS

0 10 20 30 40 50 60 70 80

0 20 40 60 80 100 120 140 160
Graphs connecting the number of resource constraints and the time required to solve the linear programming problems listed in Table 5.1 by the simplex and the multiplex algorithms are shown in Figure 5.1. It may be observed from the graphs drawn using the computed results that the slope of the curve for the simplex algorithm is steeper than that of the multiplex for any range of constraints. This shows that for a given problem, the computing time required per constraint of the multiplex algorithm is less than that of the simplex.

5.6 CONCLUSION

The various aspects of the multiplex algorithm and multiplex bounded algorithm are illustrated in this chapter with some numeric examples. To show the advantage of this method, it is compared with the revised simplex algorithm based on the processing time and multiply/divide operations. In the next chapter, few of the computing experiences are explained and illustrated with some examples.