Math 404 Report1
Revised Simplex Method

Abdelateef Khaled Abdelateef 202001344

Department of Engineering, University of Science and Technology at Zewail City, Egypt

Contents

1 Motivation

The Simplex Method is fundamental in addressing linear programming problems and numerous applications in different fields like finance. Still, it exhibits inefficiencies marked by unnecessary row operations and extensive computational demands. This becomes particularly apparent in large-scale problems where the computation and storage of the simplex tableau become impractical, leading to notable accuracy issues.
Recognizing these limitations, the **Revised Simplex Method** emerges as a compelling alternative, introducing significant advancements that enhance Computational Efficiency, Memory Optimization, and Numerical Stability.

1. **Computational Efficiency**: The Revised Simplex Method offers a distinct computational advantage by achieving considerable savings when the fraction of nonzero coefficients falls below a critical threshold. This threshold, specifically 1 - (2m/n), is often met in practical scenarios, leading to substantial reductions in computational workload.

2. **Memory Optimization**: Unlike the Simplex Algorithm, the Revised Simplex Method minimizes data retention between iterations. This not only accommodates scenarios with limited electronic computer memory but also enables the solution of larger problems by circumventing the need to update all entries in the tableau. The method proves particularly advantageous when memory constraints are a limiting factor.

3. **Numerical Stability**: Addressing a notable weakness in the simplex algorithm, the Revised Simplex Method sidesteps the explicit representation of the inverse of the basis in the full tableau. This explicit representation can be numerically unstable. By adopting numerically stable methods to solve specific systems of equations, the Revised Simplex Algorithm ensures a more robust numerical foundation.

Furthermore, The motivation for exploring the Revised Simplex Method lies in its ability to enhance computational efficiency, optimize memory usage, and bolster numerical stability. These attributes make it a promising alternative, especially when faced with large-scale problems or scenarios demanding precision and resource optimization.

2 Theory

The Theory extract from Hamdy Taha References

The LP Problem in standard form :

$$\begin{aligned} &\text{Min } \mathbf{z} = \mathbf{C}^{\mathrm{T}}\mathbf{X} \\ \text{S.t.} \quad &\mathbf{AX} = \mathbf{b} \\ &\mathbf{X} \geq \mathbf{0} \\ &\mathrm{C}, \mathrm{X} \in R^{\mathrm{n}} \\ &\mathbf{b} \in R^{\mathrm{m}} \end{aligned}$$

The system AX = b is written in vector form as below,

$$\sum_{j=1}^{n} \mathbf{P}_{\mathrm{j}} \mathrm{x}_{\mathrm{j}} = \mathbf{b}$$

where The vector $\mathbf{P}_j$ is the jth column of $\mathbf{A}$.
A subset of m vectors forms a basis, $\mathbf{B}$, if, and only if, the selected m vectors are linearly independent. In this case, the matrix $\mathbf{B}$ is nonsingular.
Defining $\mathbf{X}_B$ as an m-vector of the basic variables, then

$$\mathbf{B}\mathbf{X}_B = \mathbf{b}$$

Using the inverse $\mathbf{B}^{-1}$, the associated basic solution is

$$\mathbf{X}_B = \mathbf{B}^{-1}\mathbf{b}$$

If $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$, then $\mathbf{X}_B$ is feasible. The remaining $n-m$ variables are nonbasic at zero level.

2.1 Generalized Simplex Tableau in Matrix Form

the above LP problem can be written as

$$\begin{pmatrix} 1 & -\mathbf{C} \\ \mathbf{0} & \mathbf{A} \end{pmatrix} \begin{pmatrix} \mathbf{Z} \\ \mathbf{X} \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix}$$

Suppose that $\mathbf{B}$ is a feasbible basis for $\mathbf{AX} = \mathbf{b}$, $\mathbf{X} \geq \mathbf{0}$, and $\mathbf{X_B}$ be the corresponding vector of basic variables and $\boldsymbol{C}_B$ its associated objective vector. Given all the nonbasic variables are zero, the solution is then computed as :

$$\begin{pmatrix} \mathrm{Z} \\ \mathbf{X}_{\mathrm{B}} \end{pmatrix} = \begin{pmatrix} 1 & -\mathbf{C}_B \\ \mathbf{0} & \mathbf{B} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{C}_B\mathbf{B}^{-1} \\ \mathbf{0} & \mathbf{B}^{-1} \end{pmatrix} \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix} = \begin{pmatrix} \mathbf{C}_B\mathbf{B}^{-1}\mathbf{b} \\ \mathbf{B}^{-1}\mathbf{b} \end{pmatrix}$$

The complete simplex tableau in matrix form can be derived from the original equations as

$$\begin{pmatrix} 1 & \mathbf{C}_B\mathbf{B}^{-1} \\ \mathbf{0} & \mathbf{B}^{-1} \end{pmatrix} \begin{pmatrix} 1 & -\mathbf{C} \\ \mathbf{0} & \mathbf{A} \end{pmatrix} \begin{pmatrix} z \\ \mathbf{X} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{C}_B\mathbf{B}^{-1} \\ \mathbf{0} & \mathbf{B}^{-1} \end{pmatrix} \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix}$$

Matrix manipulations then yield the following equations:

$$\begin{pmatrix} 1 & \mathbf{C}_B\mathbf{B^{-1}A} - \mathbf{C} \\ \mathbf{0} & \mathbf{B^{-1}A} \end{pmatrix} \begin{pmatrix} z \\ \mathbf{X} \end{pmatrix} = \begin{pmatrix} \mathbf{C}_B\mathbf{B^{-1}b} \\ \mathbf{B^{-1}b} \end{pmatrix}$$

Given the j th vector $\mathbf{P}_j$ of $\mathbf{A}$, the simplex tableau column associated with variable x_j can be written as

Basic	x_j	Solution
z	$\mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_j - c_j$	$\mathbf{C}_B\mathbf{B}^{-1}\mathbf{b}$
$\mathbf{X}_B$	$\mathbf{B}^{-1}\mathbf{P}_j$	$\mathbf{B}^{-1}\mathbf{b}$

We can see that the only variable undergoing modification in the simplex tableau is the matrix $\mathbf{B}^{-1}$. Consequently, the entire set of values can be reconstructed from the initial problem if $\mathbf{B}^{-1}$ is computed. However, the prerequisite for obtaining $\mathbf{B}$ involves defining the fundamental columns of $\mathbf{X}$, which ultimately translates to $\mathbf{X}_{\mathrm{B}}$.

In contrast to the simplex method, this approach offers the advantage of sidestepping roundoff errors and mitigating the demand for excessive memory and computational resources, achieved by computing $\mathbf{B}^{-1}$ directly from the original constraint columns.

2.2 Development of the Optimally and Feasibility condition

Any simplex iteration can be represented by the following equations based on The above tableau :

$$\mathrm{z} + \sum_{j=1}^{\mathrm{B}} (\mathrm{z_j} - \mathrm{c_j})\, \mathrm{x}_i = \mathbf{C}_{\mathrm{B}}\mathbf{B}^{-1}\mathbf{b}$$

$$(\mathbf{X}_{\mathrm{B}})_i + \sum_{j=1}^{n} \left(\mathbf{B}^{-1}\mathbf{P}_j\right)_i \mathbf{x}_j = \left(\mathbf{B}^{1}\mathbf{b}\right)_i$$

Where

$$(\mathrm{z}_j - \mathrm{c}_j) = \mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_j - \mathrm{c}_j$$

Where i represents the element index in the vector j.

2.2.1 Optimality condition

The Minimization Case: the condition is $(z_j - c_j) < 0$. Thus, the entering vector is selected as the nonbasic vector with the most Negative $(z_j - c_j)$.
The Maximization Case: the condition is $(z_j - c_j) > 0$. Thus, the entering vector is selected as the nonbasic vector with the most Positive $(z_j - c_j)$.

2.2.2 Feasibility Condition

Given the entering vector p_j as determined by the optimality condition, the constraint equations reduce to

$$(\mathbf{X}_{\mathrm{B}})_i = \left(\mathbf{B}^1\mathbf{b}\right)_i - \left(\mathbf{B}^{-1}\mathbf{P}_j\right)_i \mathbf{X}_j$$

The purpose is to increase x_j above zero because the other $\mathrm{n} - 1$ variables are zero. The limit to that increase is the following condition:

$$(\mathbf{X}_{\mathrm{B}})_i = \left(\mathbf{B}^1\mathbf{b}\right)_i - \left(\mathbf{B}^{-1}\mathbf{P}_j\right)_i \mathbf{x}_j \geq 0$$

If $\left(\mathbf{B}^{-1}\mathbf{P}_j\right)_i > 0$ for at least one i, the nonnegativity condition, $(\mathbf{X}_B)_i \geq 0$ for all i, sets the limit on the maximum increase in the value of the entering variable x_j-namely,

$$x_j = \min_i \left\{ \frac{\left(\mathbf{B}^{-1}\mathbf{b}\right)_i}{\left(\mathbf{B}^{-1}\mathbf{P}_j\right)_i} \mid \left(\mathbf{B}^{-1}\mathbf{P}_j\right)_i > 0 \right\}$$

Suppose that $(\mathbf{X}_B)_k$ is the basic variable that corresponds to the minimum ratio. It then follows that $\mathbf{P}_k$ must be the leaving vector, and its associated (basic) variable must become nonbasic (at zero level) in the next simplex iteration.

3 Algorithm

The Algorithm extract from Hamdy Taha References:

- **Step 0.** Construct a starting basic feasible solution and let $\mathbf{B}$ and $\mathbf{C}_B$ be its associated basis and objective coefficients vector, respectively.
- **Step 1.** Compute the inverse $\mathbf{B}^{-1}$ of the basis $\mathbf{B}$.
- **Step 2.**For each nonbasic vector $\mathbf{P}_j$, compute

$$z_j - c_j = \mathbf{C}_B \mathbf{B}^{-1} \mathbf{P}_j - c_j$$

  If $z_j - c_j \geq 0$ in maximization (≤ 0 in minimization) for all nonbasic vectors, stop; the optimal solution is $\mathbf{X}_B = \mathbf{B}^{-1}\mathbf{b}, z = \mathbf{C}_B \mathbf{X}_B$. Else, determine the entering vector $\mathbf{P}_j$ having the most negative (positive) $z_j - c_j$ in the case of maximization (minimization) among all nonbasic vectors.

- **Step 3.** Compute $\mathbf{B}^{-1}\mathbf{P}_j$. If all the elements of $\mathbf{B}^{-1}\mathbf{P}_j$ are negative or zero, stop; the solution is unbounded. Otherwise, use the ratio test to determine the leaving vector $\mathbf{P}_i$.
- **Step 4.** Form the next basis by replacing the leaving vector $\mathbf{P}_i$ with the entering vector $\mathbf{P}_j$ in the current basis $\mathbf{B}$. Go to step 1 to start a new iteration.

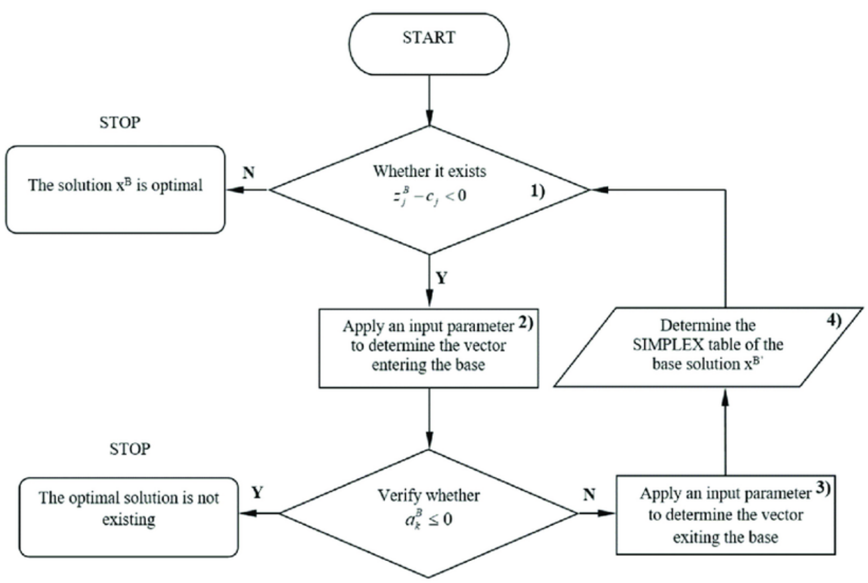

Figure 1: Revised Simplex Flowchart

4 Application

4.1 Example 1:

$$\begin{aligned} \text{Max}\, Z = 2x_1 + 3x_2 \\ \text{s.t.} \quad 2x_1 + x_2 \leq 4 \\ x_1 + 2x_2 \leq 5 \\ x_1, x_2 \geq 0 \end{aligned}$$

The standard LP form $(Z = -z)$:

$$\begin{aligned} &\text{Min} \quad z = -2x_1 - 3x_2 \\ &\qquad\qquad 2x_1 + x_2 + x_3 \qquad = 4 \\ &\text{s.t.} \qquad\quad x_1 + 2x_2 + \qquad x_4 = 5 \quad \text{(where } x_3 \text{ and } x_4 \text{ are slack)} \\ &\quad x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

Hence,$n = 4$, $m = 2$.

$$\begin{aligned} X &= \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} \\ C &= \begin{bmatrix} -2 & -3 & 0 & 0 \end{bmatrix} \\ A &= \begin{bmatrix} 2 & 1 & 1 & 0 \\ 1 & 2 & 0 & 1 \end{bmatrix} \\ \boldsymbol{b} &= \begin{bmatrix} 4 \\ 5 \end{bmatrix} \end{aligned}$$

4.1.1 Iteration 0

$$\begin{aligned} &\boldsymbol{X}_{B_0} = [x_3 \quad x_4] \\ &\boldsymbol{C}_{B_0} = [0 \quad 0] \\ &B_0 = (P_3, P_4) = \boldsymbol{I} \\ &B_0^{-1} = I \end{aligned}$$

Thus

$$\begin{aligned} &X_{B_0} = B_0^{-1} b = \begin{bmatrix} 4 & 5 \end{bmatrix} \\ &z = C_{B_0} X_{B_0} = 0 \end{aligned}$$

Optimality Condition:

$$\{z_j - c_j\}_{j=1,2} = \boldsymbol{C}_{B_0} \boldsymbol{B}_0^{-1} [\boldsymbol{P}_1 \quad \boldsymbol{P}_2] - [c_1 \quad c_2] = \begin{bmatrix} 2 & 3 \end{bmatrix}$$

Looking for most positive vector, $\mathbf{P_2}$ is the entering vector.

Feasibility Condition:

$$X_{B_0} = \begin{bmatrix} x_3 & x_4 \end{bmatrix}^T$$
$$B_0^{-1}P_2 = \begin{bmatrix} 1 & 2 \end{bmatrix}^T$$

$$x_2 = \min\left\{\frac{4}{1}, \frac{5}{2}\right\} = 2.5$$

Then $\boldsymbol{P}_4$ becomes the leaving vector

4.1.2 Iteration 1

$$\boldsymbol{X}_{B_1} = [x_3 \quad x_2]$$
$$\boldsymbol{C}_{B_1} = [0 \quad -3]$$
$$B_1 = (P_3, P_2) = \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix}$$
$$B_1^{-1} = \begin{bmatrix} 1 & -0.5 \\ 0 & 0.5 \end{bmatrix}$$

Thus

$$X_{B_1} = B_1^{-1}b = \begin{bmatrix} 1 & -0.5 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 4 \\ 5 \end{bmatrix} = \begin{bmatrix} 1.5 \\ 2.5 \end{bmatrix}$$
$$-z = C_{B_1}X_{B_1} = -7.5$$

Optimality Condition:

$$\boldsymbol{C}_{B_1}\boldsymbol{B}_1^{-1} = \begin{bmatrix} 0 & -1.5 \end{bmatrix}$$
$$\{\mathrm{z}_j - c_j\}_{j=1,4} = \boldsymbol{C}_{B_1}\boldsymbol{B}_1^{-1}[\boldsymbol{P}_1 \quad \boldsymbol{P}_4] - [c_1 \quad c_4] = \begin{bmatrix} 0.5 & -1.5 \end{bmatrix}$$

Looking for most positive vector, $\mathbf{P_1}$ is the entering vector.

Feasibility Condition:

$$X_{B_1} = \begin{bmatrix} x_3 & x_2 \end{bmatrix}^T$$
$$B_1^{-1}P_1 = \begin{bmatrix} 1.5 & 0.5 \end{bmatrix}^T$$

$$x_2 = \min\left\{\frac{1.5}{1.5}, \frac{2.5}{0.5}\right\} = 1$$

Then $\boldsymbol{P}_3$ becomes the leaving vector

4.1.3 Iteration 2

$$
\begin{aligned}
\boldsymbol{X}_{B_2} &= [x_1 \quad x_2] \\
\boldsymbol{C}_{B_2} &= [-2 \quad -3] \\
B_2 = (P_1, P_2) &= \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}
\end{aligned}
$$

$$
B_2^{-1} = \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} \end{bmatrix}
$$

Thus

$$
\begin{aligned}
X_{B_2} = B_2^{-1} b &= \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} \end{bmatrix} \begin{bmatrix} 4 \\ 5 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \\
-z = C_{B_2} X_{B_2} &= -8
\end{aligned}
$$

Optimality Condition:

$$
\begin{aligned}
&\boldsymbol{C}_{B_2} \boldsymbol{B}_2^{-1} = \begin{bmatrix} -\frac{1}{3} & -\frac{4}{3} \end{bmatrix} \\
&\{z_j - c_j\}_{j=3,4} = \boldsymbol{C}_{B_2} \boldsymbol{B}_2^{-1} [\boldsymbol{P}_3 \quad \boldsymbol{P}_4] - [c_3 \quad c_4] = \begin{bmatrix} -\frac{1}{3} & -\frac{4}{3} \end{bmatrix}
\end{aligned}
$$

Stop! Optimal solution is $\mathbf{X_{B_2}} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ and $z = 8$

4.1.4 Graphical Method

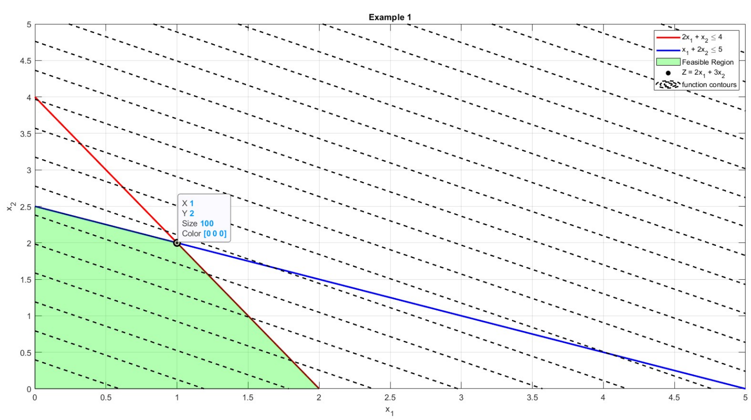

Figure 2: Example 1 using the graphical method

4.2 Example 2:

$$\begin{aligned} \text{Max}\, Z &= 5x_1 + 4x_2 \\ \text{s.t.} \quad 6x_1 + 4x_2 &\le 24 \\ x_1 + 2x_2 &\le 6 \\ x_2 &\le 2 \\ -x_1 + 2x_2 &\le 1 \\ x_1, x_2 &\ge 0 \end{aligned}$$

The standard LP form $(Z = -z)$:

$$\text{Min} \quad z = -5x_1 - 4x_2$$

$$\text{s.t.} \begin{array}{rcccccl} 6x_1 + 4x_2 + x_3 & & & & = 24 \\ x_1 + 2x_2 & +x_4 & & & = 6 \\ -x_1 + x_2 & & +x_5 & & = 1 \\ x_2 & & & +x_6 & = 2 \end{array}$$

$$x_1, x_2, x_3, x_4, x_5, x_6 \ge 0$$

where (x_3, x_4, x_5, x_6 are slack variables)

Hence,$n = 6$, $m = 4$.

$$X = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{bmatrix}$$
$$C = \begin{bmatrix} -5 & -4 & 0 & 0 & 0 & 0 \end{bmatrix}$$
$$A = \begin{bmatrix} 6 & 4 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$
$$\boldsymbol{b} = \begin{bmatrix} 24 \\ 6 \\ 1 \\ 2 \end{bmatrix}$$

4.2.1 Iteration 0

$$\begin{aligned}
&\boldsymbol{X}_{B_0} = [x_3 \quad x_4 \quad x_5 \quad x_6] \\
&\boldsymbol{C}_{B_0} = [0 \quad 0 \quad 0 \quad 0] \\
&B_0 = (P_3, P_4, P_5, P_6) = \boldsymbol{I} \\
&B_0^{-1} = I
\end{aligned}$$

Thus

$$\begin{aligned}
&X_{B_0} = B_0^{-1} b = \begin{bmatrix} 24 & 6 & 1 & 2 \end{bmatrix} \\
&z = C_{B_0} X_{B_0} = 0
\end{aligned}$$

Optimality Condition:

$$\{\mathrm{z}_j - c_j\}_{j=1,2} = \boldsymbol{C}_{B_0} \boldsymbol{B}_0^{-1} [\boldsymbol{P}_1 \quad \boldsymbol{P}_2] - [c_1 \quad c_2] = \begin{bmatrix} 5 & 4 \end{bmatrix}$$

Looking for most positive vector, $\mathbf{P_1}$ is the entering vector.
Feasibility Condition:

$$\begin{aligned}
&X_{B_0} = \begin{bmatrix} x_3 & x_4 & x_5 & x_6 \end{bmatrix}^T \\
&B_0^{-1} P_1 = \begin{bmatrix} 6 & 1 & -1 & 0 \end{bmatrix}^T \\
&x_2 = \min\left\{ \frac{24}{6}, \frac{6}{1}, \frac{1}{-1}, \frac{2}{0} \right\} = 4
\end{aligned}$$

Then $\boldsymbol{P}_3$ becomes the leaving vector

4.2.2 Iteration 1

$$\begin{aligned}
&\boldsymbol{X}_{B_1} = [x_1 \quad x_4 \quad x_5 \quad x_6] \\
&\boldsymbol{C}_{B_1} = [-5 \quad 0 \quad 0 \quad 0] \\
&B_1 = (P_1, P_4, P_5, P_6) = \begin{bmatrix} 6 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&B_1^{-1} = \begin{bmatrix} \frac{2}{3} & 0 & 0 & 0 \\ -\frac{2}{3} & 1 & 0 & 0 \\ \frac{2}{3} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}$$

Thus

$$\begin{aligned}
&X_{B_1} = B_1^{-1} b = \begin{bmatrix} 4 & 2 & 5 & 2 \end{bmatrix} \\
&z = C_{B_1} X_{B_1} = -20
\end{aligned}$$

Optimality Condition:

$$\boldsymbol{C}_{B_1}\boldsymbol{B}_1^{-1} = \begin{bmatrix} -\frac{5}{6} & 0 & 0 & 0 \end{bmatrix}$$
$$\{z_j - c_j\}_{j=3,2} = \boldsymbol{C}_{B_1}\boldsymbol{B}_1^{-1}[\boldsymbol{P}_3 \quad \boldsymbol{P}_2] - [c_3 \quad c_2] = \begin{bmatrix} -\frac{5}{6} & \frac{2}{3} \end{bmatrix}$$

Looking for most positive vector, $\mathbf{P_2}$ is the entering vector.
Feasibility Condition:

$$X_{B_1} = \begin{bmatrix} x_1 & x_4 & x_5 & x_6 \end{bmatrix}^T$$
$$B_1^{-1}P_2 = \begin{bmatrix} \frac{2}{3} & \frac{4}{3} & \frac{5}{3} & 1 \end{bmatrix}^T$$

$$x_2 = \min\left\{\frac{4}{\frac{2}{3}}, \frac{2}{\frac{4}{3}}, \frac{5}{\frac{5}{3}}, \frac{2}{1}\right\} = 1.5$$

Then $\boldsymbol{P}_4$ becomes the leaving vector

4.2.3 Iteration 2

$$\boldsymbol{X}_{B_2} = [x_1 \quad x_2 \quad x_5 \quad x_6]$$
$$\boldsymbol{C}_{B_2} = [-5 \quad -4 \quad 0 \quad 0]$$
$$B_2 = (P_1, P_2, P_5, P_6) = \begin{bmatrix} 6 & 4 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ -1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$
$$B_2^{-1} = \begin{bmatrix} \frac{1}{4} & -\frac{1}{2} & 0 & 0 \\ -\frac{1}{8} & \frac{3}{4} & 0 & 0 \\ \frac{3}{8} & -\frac{5}{4} & 1 & 0 \\ \frac{1}{8} & -\frac{3}{4} & 0 & 1 \end{bmatrix}$$

Thus

$$X_{B_2} = B_2^{-1}b = \begin{bmatrix} 3 & 1.5 & 2.5 & 0.5 \end{bmatrix}$$
$$z = C_{B_2}X_{B_2} = -21$$

Optimality Condition:

$$\boldsymbol{C}_{B_2}\boldsymbol{B}_2^{-1} = \begin{bmatrix} -\frac{3}{4} & -\frac{1}{2} & 0 & 0 \end{bmatrix}$$
$$\{z_j - c_j\}_{j=3,4} = \boldsymbol{C}_{B_2}\boldsymbol{B}_2^{-1}[\boldsymbol{P}_3 \quad \boldsymbol{P}_4] - [c_3 \quad c_4] = \begin{bmatrix} -\frac{3}{4} & -\frac{1}{2} \end{bmatrix}$$

Stop! Optimal solution is $\mathbf{X_{B_2}} = \begin{bmatrix} 3 \\ 1.5 \\ 2.5 \\ 0.5 \end{bmatrix}$ and $z = 21$

4.2.4 Graphical Method

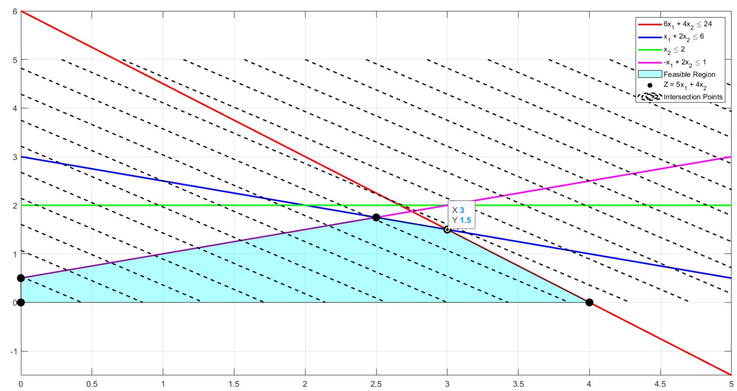

Figure 3: Example 2 using the graphical method

5 Implementation

5.1 Matlab Revised Simplex Function Implementation

```
% The revised_simplex function takes four input parameters:
%   f: Coefficients of the objective function to be maximized or minimized
%   A: Coefficient matrix for the system of constraints
%   b: Right-hand side vector of the constraints
%   obj: String specifying the optimization type ("max" for maximization, "min" for minimization)

% The function internally converts the maximization problem into a minimization problem,
% adds slack variables to handle inequalities, and initializes the basic feasible solution.
% The function prints the optimal solution vector X_B and the optimal objective function value z.
% Ensure that the input follows the standard form of linear programming problems.
% The function also handles unbounded problems and reports if the problem is unbounded.

% Output:
% The function will display the optimal solution vector X_B and the optimal objective function value z.

function revised_simplex(f, A, b,obj)
    [m, n] = size(A);
    if obj=="max"
        c=-f;
    elseif obj == "min"
        c=f;
    end
    % Add slack variables
    A = [A, eye(m)];
    c = [c, zeros(1, m)];
    % Initial basic feasible solution
    [m, n] = size(A);
    non_basic_index= 1:(n - m);
    basic_index=(n - m + 1):n;
    C_B = c(:, basic_index);
    B = A(:, basic_index);
    P=A(:, non_basic_index);
    C=c(:, non_basic_index);
    %the solution in each iteration
    invB =inv(B);
    X_B = invB*b;
    z = C_B * X_B;
    while true
        % Optimality condition
        reduced_costs = C_B * invB * P - C;
        if all(reduced_costs <= 0)
            fprintf('Optimal solution found:\n');
            fprintf('X_B = %s\n', mat2str(X_B));
            fprintf('z = %f\n', -z);
            break;
```

```
        end
        % Entering vector
        [~, entering_index] = max(reduced_costs);
        real_entering_index = non_basic_index(entering_index);
        %Feasibility Condition
        P_j = invB * A(:, real_entering_index);
        if all(P_j <= 0)
            fprintf('Problem is unbounded.\n');
            break;
        end
        % Ratio test to find leaving vector
        ratios = X_B ./ P_j;
        [~, leaving_index] = min(ratios(ratios>0));
        real_leaving_index = basic_index(leaving_index);
        % Update basis
        index_to_update_basic_index = find(basic_index ==
    real_leaving_index);
        index_to_update_non_basic_index  = find(non_basic_index ==
    real_entering_index);
        non_basic_index(index_to_update_non_basic_index)=
    real_leaving_index;
        basic_index(index_to_update_basic_index)=
    real_entering_index;
        C_B = c(:, basic_index);
        B = A(:, basic_index);
        P=A(:, non_basic_index);
        C=c(:, non_basic_index);
        invB=inv(B);
        X_B = invB*b;
        z = C_B * X_B;
    end
end
```

5.2 Matlab Test Cases Scenario For Validation

```
clear
clc
%%
% Example 1
f = [2 3];
A = [2 1; 1 2];
b = [4; 5];
disp("                        Example 1                        ")
revised_simplex(f, A, b,"max");
disp("-------------------------------------------------------")
%%
% Example 2
f = [5 4];
A = [6 4; 1 2;-1 1;0 1];
b = [24 ;6;1;2];
disp("                        Example 2                        ")
revised_simplex(f, A, b,'max');
disp("-------------------------------------------------------")
```

6 Comparison with MATLAB built-in function

The Comparison was carried out between MATLAB's function linprog that used the traditional simplex way to evaluate and My code above in section 5.

6.1 Example 1

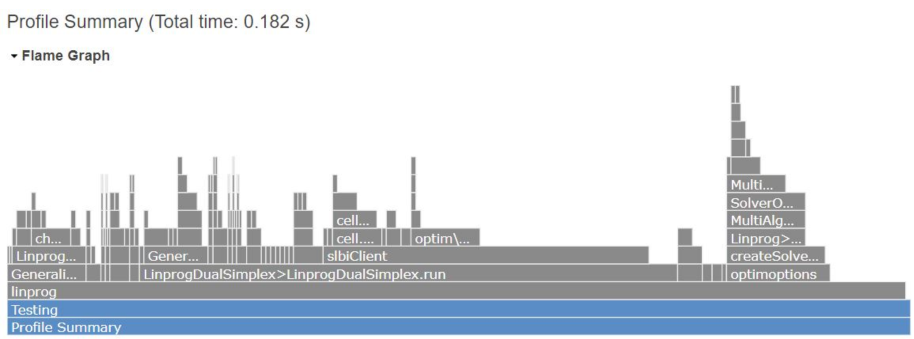

Figure 4: linprog Compiling Time

Figure 5: My code Compiling Time

Example 1 demonstrates a significant improvement in compilation time with my code, reducing it from **0.182 s** (using linprog) to just **0.009 s**.
This notable enhancement in efficiency underscores the effectiveness of the proposed approach over the traditional simplex method.

6.2 Example 2

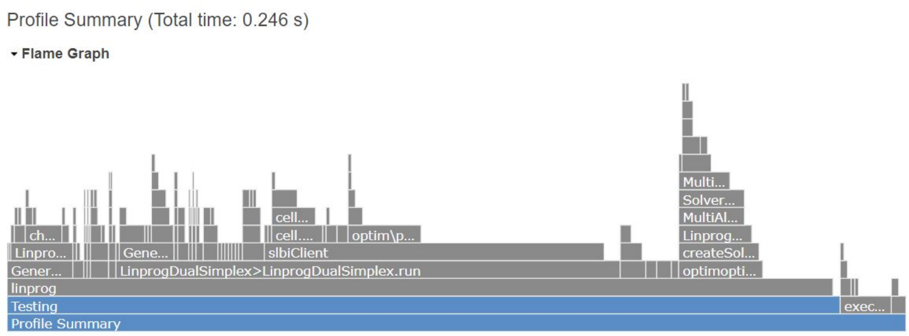

Figure 6: linprog Compiling Time

Figure 7: My code Compiling Time

In Example 2, the compilation time is further reduced with my code, clocking in at **0.013 s** compared to linprog's **0.246 s**.
This consistent trend in improved performance highlights the efficiency gains achieved by the proposed method across various instances, reinforcing its superiority in optimizing compilation times.

7 conclusion

In conclusion, the Revised Simplex Method emerges as a crucial tool in cost and profit analysis, demonstrating its superiority in handling large-scale problems compared to traditional approaches. It presents a compelling alternative with notable advancements in Computational Efficiency, Memory Optimization, and Numerical Stability, particularly evident in addressing large-scale problems.

8 References

1. Taha, H. A. (2017). Operations Research: An Introduction, Global Edition (10th ed.). Pearson.

2. Bhunia, A., Sahoo, L., Shaikh, A. (2019). Revised Simplex Method. In: Advanced Optimization and Operations Research. Springer Optimization and Its Applications, vol 153. Springer, Singapore.

3. Singiresu S. Rao, Engineering Optimization: Theory and Practice, 4 th ed., John Wiley Sons, 2009.

4. Jorge Nocedal and S.J. Wright, Numerical Optimization, Springer, 2006.

5. Fuksa, Dariusz. (2021). A Method for Assessing the Impact of Changes in Demand for Coal on the Structure of Coal Grades Produced by Mines. Energies. 14. 7111. 10.3390/en14217111.