ON THE USE OF SUM OF UNOCCUPIED ROWS FOR THE SIMPLEX ALGORITHM INITIALIZATION

BY

MISS TANCHANOK PHUMRACHAT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE (MATHEMATICS)
DEPARTMENT OF MATHEMATICS AND STATISTICS
FACULTY OF SCIENCE AND TECHNOLOGY
THAMMASAT UNIVERSITY
ACADEMIC YEAR 2017

ON THE USE OF SUM OF UNOCCUPIED ROWS FOR THE SIMPLEX ALGORITHM INITIALIZATION

BY

MISS TANCHANOK PHUMRACHAT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF MASTER OF SCIENCE (MATHEMATICS)

DEPARTMENT OF MATHEMATICS AND STATISTICS

FACULTY OF SCIENCE AND TECHNOLOGY

THAMMASAT UNIVERSITY

ACADEMIC YEAR 2017

THAMMASAT UNIVERSITY

FACULTY OF SCIENCE AND TECHNOLOGY

THESIS

BY

MISS TANCHANOK PHUMRACHAT

ENTITLED

ON THE USE OF SUM OF UNOCCUPIED ROWS FOR THE SIMPLEX ALGORITHM INITIALIZATION

was approved as partial fulfillment of the requirements for the degree of Master of Science (Mathematics)

on July 19 , 2018

Chairman

(Assistant Professor Krung Sinapiromsaran, Ph.D.)

Member and Advisor

(Aua-aree Boonperm, Ph.D.)

Member

(Assistant Professor Wutiphol Sintunavarat, Ph.D.)

Member

(Adoon Pansuwan, Ph.D.)

Dean

(Associate Professor Somchai Chakhatrakan, Ph.D.)

Thesis Title	ON THE USE OF SUM OF UNOCCUPIED ROWS FOR THE SIMPLEX ALGORITHM INITIALIZATION
Author	Miss Tanchanok Phumrachat
Degree	Master of Science (Mathematics)
Department/Faculty/University	Mathematics and Statistics Faculty of Science and Technology Thammasat University
Thesis Advisor	Aua-aree Boonperm, Ph.D.
Academic Year	2017

ABSTRACT

In this thesis, we propose the algorithm for starting the simplex algorithm without using artificial variables. The proposed algorithm starts by considering the sum of all constraints in the standard form for identifying the solution or constructing the initial basis. If the largest coefficient of the sum of all constraints is negative, then this algorithm can report infeasibility. Otherwise, it starts with an empty basic variable set, then a nonbasic variable is chosen into the basic variable set one by one by considering the sum of unoccupied rows with the minimum ratio until the basic variable set is full or the infeasibility is reported. If the basic variable set is full, then the simplex method can start. This algorithm can guarantee that a basic feasible variable set is found, or infeasibility is reported. Since it obviates the use of artificial variables and it can report the infeasibility immediately for some problems, the computational time of the simplex algorithm can be reduced. From computational results, we found that the average number of iterations solving by the proposed algorithm is less than the average number of iterations solving by Gao's algorithm.

Keywords: Artificial variable, Basic variable set, Linear programming problem, Simplex algorithm

ACKNOWLEDGEMENTS

This thesis would not have been accomplished without the kind assistance by the ones who always supported me.

First, I wish to gratefully thank my thesis advisor, Dr. Aua-aree Boonperm, who supported and encouraged me in this study. She gave useful ideas and inspiration for my thesis. She also gave me several opportunities to experience in both domestic and international presentations.

Second, I would like to sincerely thank the thesis committees consisting of Assistant Professor Dr. Krung Sinapiromsaran, Assistant Professor Dr. Wutiphol Sintunavarat, and Dr. Adoon Pansuwan for giving their useful suggestions, comments and ideas relating to this work.

Third, I would like to thank for the master's degree scholarship from Faculty of Science and Technology, Thammasat University. I wish to thank the Department of Mathematics and Statistics for supporting the fund for presentation in OR-NET 2018, Thailand.

Along my master degree study, I got the kind assistance from nice friends who have been successively supporting me. For my friends who have the same advisor, I would like to thank Miss Panthira Jamrunroj for her advice in designing the Matlab program. I also would like to thank Miss Chanissara Prayonghom for picking up me when I went to study at Thammasat University.

Finally, I wish to gratefully thank my family especially my parents who have continuously and financially contributed to all of my studies.

Miss Tanchanok Phumrachat

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Backgrounds

Linear programming models are mathematical models that attempt to model a real-life situation. It can be viewed as a generalization of solving simultaneous linear equations or inequalities. All linear programming problems consist of three components: constraints, decision variables and the objective function. For constraints, these are conditions which limit the values of variable. The unknowns are called decision variables which the quantities are used for determining. The objective function will be optimized which is determined by the decision variables. The linear programming model is written as follows:

$$\begin{array}{lrcccccccc} \text{min or max} & z & = & c_1x_1 & + & c_2x_2 & + & \cdots & + & c_nx_n \\ \text{subject to} & & & a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ & & & a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ & & & \vdots & & \vdots & & \vdots & & \vdots \\ & & & a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & = & b_m \\ & & & x_1, & & x_2, & & \ldots, & & x_n & \geq & 0, \end{array} \tag{1.1.1}$$

where coefficients $c_1, c_2, \ldots, c_n$ are the **cost coefficients** and $x_1, x_2, \ldots, x_n$ are the **decision variables**. The coefficients a_{ij} for $i = 1, ..., m$ and $j = 1, ..., n$ are called the **technological coefficients**. The linear function $c_1x_1 + c_2x_2 + \cdots + c_nx_n$ is called the **objective function** which is minimized or maximized and denoted by z. The equations and the restriction of decision variables are denoted as the **constraints**.

Denote the following column vectors **c** and **x** of size n, **b** of size m and $m \times n$ matrix **A**,

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \quad \text{and} \quad \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix}$$

Values of the variables $x_1, x_2, \ldots, x_n$ satisfying all constraints are called a **feasible point** or a **feasible solution**. The set of all such points constitutes the **feasible region** or the **feasible space**.

The problem (1.1.1) can be written in the matrix form as follows:

$$\begin{aligned} \text{min or max} \quad & z = \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{Ax} = \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned} \tag{1.1.2}$$

where $\mathbf{c} \in \mathbb{R}^n$ is a column vector of cost coefficients,

$\mathbf{x} \in \mathbb{R}^n$ is a column vector of decision variables,

$\mathbf{b} \in \mathbb{R}^m$ is a column vector of right-hand-side value,

and $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a coefficient matrix of constraints.

The solution of a linear programming problem depends on the constraints and the objective function. There are three possible situations which is infeasible if it has no feasible solution, unbounded if the objective value can be increased along the feasible direction for maximization or be decreased along the feasible direction for minimization, and it has an optimal solution which maximizes or minimizes the objective function. We can use a variety of techniques and algorithms for solving a linear programming problem such as the graphical method, the simplex method, and the interior point method. The graphical method is appropriate for the linear programming problem in two or three dimensions which it can be plotted. For a large linear programming problem that cannot be plotted a graph, we will use the simplex method or the interior point method for solving it. For the simplex method, it was created by Dantzig [1] in 1947 which is the iterative method for a general linear programming problem. It starts

by choosing the matrix $\mathbf{B}$ which is chosen m columns from $\mathbf{A}$. The matrix $\mathbf{B}$ is called the basic matrix or basis which is an $m \times m$ invertible matrix. If $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$, then the basic feasible solution is found and the simplex method can start. However, a matrix $\mathbf{B}$ with $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$ may not be obviously identified. Thus, artificial variables are added for choosing $\mathbf{B} = \mathbf{I}$ which it can guarantee that the problem has a basic feasible solution for $\mathbf{b} \geq \mathbf{0}$. However, artificial variables will be forced to leave from the basis since they are not legitimate variables. Two well-known methods which can drive the artificial variables out of basis are two-phase method and big-M method. After the basic feasible solution is found, it moves to an adjacent basic feasible solution until the optimal solution is found. Although the simplex method is still the dominant procedure for solving a linear programming problem, a weak point for solving the large linear programming problem is that the worst-case running is exponential. Therefore, another method for solving a linear programming problem is invented called the interior point method developed by Karmarkar [2] in 1984. The interior point method starts with an initial interior point, this method moves through the interior of the feasible point along some direction to another interior point until it converges to the optimal boundary point. Nevertheless, the interior point method is slow for solving a linear programming problem with a dense matrix and cannot use post-optimality for analysis [12]. Thus, several researches have been interested in improvement of the simplex algorithm instead of the interior point method.

The simplex method sometimes starts with artificial variables which are introduced for constructing the basic feasible solution. However, a large matrix size needs more computing time when artificial variables are added. By this reason, the research question arises that the simplex method may start without using the artificial variable. Will it be possible to avoid the use of artificial variable? Later, the researchers have been proposing new methods for solving a linear programming problem without using artificial variables.

1.2 Literature reviews

In 1997, Arsham [3] created the artificial-free simplex algorithm for constructing a basic variable set consisting of two phases. In Phase I, it begins with the empty basic variable set, then a nonbasic variable is chosen into the basic variable set one by one using the coefficient of nonbasic variables in the objective function and the minimum ratio. This phase can report that a problem is infeasibility or the basic variable set is full. After Phase I ends, if the basic variable set is full, Phase II is used to find the optimal solution by the ordinary simplex method. The strong points of this method are the avoidance of artificial variables and the protection of cycling for choosing the variables into the basic variable set. However, Enge and Huhn [4] gave a counterexample, in which Arsham's Phase I algorithm declares the infeasibility of a feasible problem. The mistake of Arsham's algorithm occurs when the variables are already chosen into the basic variable set, then other variables with the minimum ratio cannot be replaced them. Thus, sometimes a basic feasible variable set cannot be found.

In 2015, Gao [5] improved the Phase I of Arsham's algorithm in two variants. Both start when Arsham's algorithm stops, and it cannot construct the basic feasible variable set, and the nonbasic variable with minimum ratio is not in the basic variable set. For Variant 1, other nonbasic variables with minimum ratio are allowed to replace the previous variable in the basic variable set. However, Gao's improvement has a mistake in Variant 1. The selection of a nonbasic variable into the basic variable set by considering the absolute of technical coefficients, some problems are reported the infeasibility for the feasible problem. For Variant 2, the summation of coefficient matrix of unoccupied rows which no basic variable is in these rows is considered. It starts by adding a constraint which is the summation of coefficient matrix of all unoccupied rows. Then, the nonbasic variable is chosen into the basic variable set by considering the largest coefficient of the added constraint. Nevertheless, Variant 2 is used when Arsham's algorithm reports that the problem is infeasible and a nonbasic variable with

the minimum ratio is not chosen into the basic variable set. Although both variants give the basic feasible variable set, this algorithm still has disadvantages. First, it can start when Arsham's algorithm ends. It wastes the calculation time by using Arsham's algorithm for checking infeasiblity before Gao's method starts. Second, Gao's algorithm will leave a basic variable which should not be in the basic variable set chosen by Arsham's algorithm to enter the basic variable set before a nonbasic variable with the minimum ratio will replace it. Thus, the number of iterations increase.

Due to Variant 2 of Gao's algorithm, it can identify the infeasibility of a linear programming problem by considering the summation of coefficient matrix of unoccupied rows from Arsham's algorithm. So, the key research question of this study was to use the sum of unoccupied rows which is all constraints for starting the simplex method first. Since the sum of unoccupied rows can identify the solution for some linear programming problems, and it is equivalent to the reduced cost of the initial tableau of Phase I in the two-phase method, in this thesis, we will use the sum of unoccupied rows to identify its solution without calling Arsham's algorithm. Because all constraints of the linear programming problem are a linear system, if we can find a nonnegative solution of the linear system, then it is also the feasible solution of the linear programming problem. In addition, the sum of unoccupied rows is equivalent to the reduced cost in the initial tableau of Phase I in the two-phase method which is the technique for finding a feasible solution. Therefore, we will use the sum of unoccupied rows for starting the simplex algorithm and reporting the infeasibility. If the largest coefficient of the sum of unoccupied rows is negative, then the infeasibility is reported. Otherwise, it starts with the empty basic variable set, then the first nonbasic variable is chosen into the basic variable set by considering the largest coefficient of the sum of unoccupied rows. After there exists a variable in the basic variable set, the next nonbasic variable is chosen into the basic variable set by considering the sum of unoccupied rows with minimum ratio until the basic variable set is full. Next, the simplex method is used for finding the optimal solution. The proposed algorithm is

separated into two algorithms; **nonhomogeneous linear programming algorithm** for a nonzero right-hand side vector and **homogeneous linear programming algorithm** for a zero right-hand-side vector.

The aim of this thesis is to design the algorithm which can reduce the number of iterations or time to solve a linear programming problem with respect to Gao's algorithm. The efficiency of the proposed method is presented by the computational results which tested by randomly generated linear programming problems.

1.3 Overview

From the previous section, our main objective is proposed. Then we are going to describe the contents of the thesis.

In Chapter 2, we first introduce definitions and theorems of vector and matrix in order to lay the basic idea of our thesis. Next, a linear programming problem and its possible solutions are presented. Another problem which is an associated linear programming problem is proposed called the dual problem. Additionally, many methods for solving a linear programming problem such as the simplex method, the two-phase method, Arsham's algorithm, and Gao's algorithm are presented.

In Chapter 3, we describe the sum of unoccupied rows which is the importance idea of our algorithm. Then, the proposed algorithms are separated into two algorithms, the steps of our algorithm and the illustrative examples are presented.

In Chapter 4, we test the efficiency of the proposed algorithm by implementing the proposed algorithm and Gao's algorithm. Their number of iterations and computation time are compared. Then, results of our algorithm will be analyzed.

In the last chapter, the conclusions of our results is presented.

CHAPTER 2

FOUNDATIONS

This chapter devotes to recall some definitions, theorems, and examples of linear algebra and a linear programming problem which are foundations of our proposed algorithm. First, we will start by describing some basic linear algebra followed by a linear programming problem. Then, some recent methods for solving linear programming problems are presented.

2.1 Vectors and matrices

Some basic properties of vectors and matrices is proposed in this section.

Definition 2.1.1. An ***n*-column vector** is a column array of n numbers.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

whose entries are real numbers, called the **components** of $\mathbf{x}$. The set of all n-vectors is denoted by $\mathbb{R}^n$ and is called ***n*-dimensional space**. The vector $\mathbf{x}$ can be written as $(x_1, x_2, \ldots, x_n)$.

Definition 2.1.2. The **zero vector** is a vector with all components equal to zero, denoted by $\mathbf{0}$.

Definition 2.1.3. An ***i*th unit vector** is a vector with all zero components, except for a 1 in the i^{th} position, denoted by $\mathbf{e}_i$.

Definition 2.1.4. Let $\mathbf{a}_1, \mathbf{a}_2 \in \mathbb{R}^n$ be the vectors of n-dimentional space:

$$\mathbf{a}_1 = (a_{11}, a_{21}, ..., a_{n1}) \text{ and } \mathbf{a}_2 = (a_{12}, a_{22}, ..., a_{n2}).$$

Then, the **addition of vectors** $\mathbf{a}_1$ and $\mathbf{a}_2$, denoted by $\mathbf{a}_1 + \mathbf{a}_2$, is the following vector:

$$\mathbf{a}_1 + \mathbf{a}_2 = (a_{11} + a_{12}, a_{21} + a_{22}, ..., a_{n1} + a_{n2}).$$

Definition 2.1.5. Let $\mathbf{a}$ be a vector in $\mathbb{R}^n$ and k be a scalar. Then, the **scalar multiplied** vector is defined as $k\mathbf{a} = (ka_1, ka_2, ..., ka_n)$.

Definition 2.1.6. Any two n-dimentional vectors $\mathbf{a}$ and $\mathbf{b}$ can be multiplied. The result of this multiplication is a real number called the **inner product** of the two vectors. It is defined as follows:

$$\mathbf{ab} = a_1b_1 + a_2b_2 + \cdots + a_nb_n = \sum_{j=1}^{n} a_jb_j.$$

Definition 2.1.7. A nonempty subset V of $\mathbb{R}^n$ is called a **subspace** for $\mathbb{R}^n$ if the following properties are satisfied.

1. If $\mathbf{x}$ and $\mathbf{y}$ are any vectors in V , then $\mathbf{x} + \mathbf{y}$ is in V .
2. If r is any real number and $\mathbf{x}$ is any vector in V, then $r\mathbf{x}$ is in V.

Definition 2.1.8. A vector $\mathbf{v}$ in $\mathbb{R}^n$ is said to be a **linear combination** of vectors $\mathbf{v}_1$, $\mathbf{v}_2$, ..., $\mathbf{v}_k \in \mathbb{R}^n$, if it can be written as

$$\mathbf{v} = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k$$

where $c_1, c_2, ..., c_k$ are real numbers.

Example 2.1.1. Let

$$\mathbf{v} = \begin{bmatrix} 4 \\ 7 \\ 2 \end{bmatrix}, \quad \mathbf{v}_1 = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}, \quad \text{and} \quad \mathbf{v}_2 = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$$

The vector $\mathbf{v}$ is a linear combination of $\mathbf{v}_1$ and $\mathbf{v}_2$, since there are real numbers c_1 and c_2 such that

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 = \mathbf{v}$$

or

$$c_1\begin{bmatrix}1\\2\\-1\end{bmatrix} + c_2\begin{bmatrix}2\\3\\4\end{bmatrix} = \begin{bmatrix}4\\7\\2\end{bmatrix}$$

where $c_1 = 2$ and $c_2 = 1$.

■

Definition 2.1.9. Let $S = \{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k\}$ be a set of vectors in a subspace V of $\mathbb{R}^n$. The set S **spans** V, or V is **spanned by** S, if every vector in V is a linear combination of the vectors in S.

Definition 2.1.10. Let $S = \{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k\}$ be a set of distinct vectors in a subspace V of $\mathbb{R}^n$. The set S is said to be **linearly dependent** if we can find constants $c_1, c_2, ..., c_k$ not all zeroes, such that

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k = \mathbf{0}. \tag{2.1.1}$$

Otherwise, S is said to be **linearly independent**. That is, S is linearly independent if Equation (2.1.1) can be satisfied only with

$$c_1 = c_2 = \ldots = c_k = 0.$$

Example 2.1.2. Let

$$\mathbf{v}_1 = \begin{bmatrix}1\\2\\0\\1\end{bmatrix}, \quad \mathbf{v}_2 = \begin{bmatrix}0\\1\\1\\2\end{bmatrix}, \quad \text{and} \quad \mathbf{v}_3 = \begin{bmatrix}1\\1\\-1\\1\end{bmatrix}$$

Consider

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + c_3\mathbf{v}_3 = \mathbf{0}$$

which yields the linear system

$$\begin{aligned} c_1 \quad\quad + \quad c_3 &= 0 \\ 2c_1 + c_2 + c_3 &= 0 \\ + c_2 - c_3 &= 0 \\ c_1 + 2c_2 + c_3 &= 0 \end{aligned}$$

Since this linear system has only the solution $c_1 = c_2 = c_3 = 0$, we conclude that $S = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ is linearly independent.

■

Definition 2.1.11. A set of vectors $S = \{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k\}$ in a subspace V of $\mathbb{R}^n$ is called a **basis** for V if S spans V and S is linearly independent.

Theorem 2.1.1. If $S = \{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k\}$ is a basis for a subspace V of $\mathbb{R}^n$, then every vector $\mathbf{x}$ in V can be written in one and only one way as a linear combination of the vectors in S.

Since linear algebra is concerned with matrices and vectors, we will introduce some of the basic ideas about matrices as below.

Definition 2.1.12. An $m \times n$ **matrix A** is a rectangular array of mn numbers arranged in m horizontal rows and n vertical columns:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix} \tag{2.1.2}$$

The $\boldsymbol{i}^{\text{th}}$**row** of **A** denoted by $\mathbf{A}_{i:}$ is

$$\mathbf{A}_{i:} = \begin{bmatrix} a_{i1} & a_{i2} & a_{i3} & \cdots & a_{in} \end{bmatrix} \quad (1 \leq i \leq m)$$

formed from the rows of **A** called the **row vector** of **A**.

The $\boldsymbol{j}^{\textbf{th}}$**column** of **A** denoted by $\mathbf{A}_{:j}$ is

$$\mathbf{A}_{:j} = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{bmatrix} \quad (1 \leq j \leq n)$$

formed from the columns of **A** called the **column vector** of A.

The number in the ith row and jth column of **A** is denoted by a_{ij}, and is called the $\mathbf{ij}^{\textbf{th}}$**element** of **A**, or the **(i, j) entry of A**, and we often write (2.1.2) as

$$\mathbf{A} = [a_{ij}].$$

The $m \times n$ matrix **A** is said to be **square of order** n if $m = n$. In this case, the numbers $a_{11}, a_{22}, ..., a_{nn}$ form the **main diagonal elements** of **A**.

We now turn to the definition of several operations on matrices. These operations will enable us to apply to matrices later.

Definition 2.1.13. Two $m \times n$ matrices $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$ are said to be **equal** if $a_{ij} = b_{ij}$ for each choice of i and j, where $1 \leq i \leq m,\ 1 \leq j \leq n$.

Definition 2.1.14. If $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$ are $m \times n$ matrices, then the **sum** of **A** and **B** is the matrix $\mathbf{C} = [c_{ij}]$, defined by

$$c_{ij} = a_{ij} + b_{ij} \quad (1 \leq i \leq m,\ 1 \leq j \leq n).$$

Definition 2.1.15. If $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$ are $m \times n$ matrices, then the **difference** $\mathbf{A} - \mathbf{B}$ is the matrix $\mathbf{D} = [d_{ij}]$, defined by

$$d_{ij} = a_{ij} - b_{ij} \quad (1 \leq i \leq m,\ 1 \leq j \leq n).$$

Proposition 2.1.2. (Properties of Matrix Addition)

1. $\mathbf{A}+\mathbf{B}=\mathbf{B}+\mathbf{A}$ for all $m\times n$ matrices $\mathbf{A}$ and $\mathbf{B}$.

2. $\mathbf{A}+(\mathbf{B}+\mathbf{C})=(\mathbf{A}+\mathbf{B})+\mathbf{C}$ for all $m\times n$ matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$.

3. There is a unique $m\times n$ matrix $\mathbf{0}$, called the $m\times n$ **zero matrix**, such that

$$\mathbf{A}+\mathbf{0}=\mathbf{A} \quad \text{for any } m\times n \text{ matrix } \mathbf{A}.$$

4. For each $m\times n$ matrix $\mathbf{A}$, there is a unique matrix, denoted by $-\mathbf{A}$, such that

$$\mathbf{A}+(-\mathbf{A})=\mathbf{0}.$$

   The matrix $-\mathbf{A}$ is called the **negative** of $\mathbf{A}$. The ij^{th} element of $-\mathbf{A}$ is $-a_{ij}$, where $\mathbf{A}=[a_{ij}]$.

Definition 2.1.16. Let $\mathbf{A}=[a_{ij}]$ be an $m\times p$ matrix and $\mathbf{B}=[b_{ij}]$ be an $p\times n$ matrix. The **product** of $\mathbf{A}$ and $\mathbf{B}$ is the $m\times n$ matrix $\mathbf{C}=[c_{ij}]$, defined by

$$c_{ij}=a_{i1}b_{1j}+a_{i2}b_{2j}+\cdots+a_{ip}b_{pj} \qquad (1\leq i\leq m,\ 1\leq j\leq n).$$

Proposition 2.1.3. (Properties of Matrix Multiplication)
Let $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ be matrices of the following sizes $m\times n$, $n\times p$, and $n\times p$. Then the following assertions hold.

1. **A(BC) = (AB)C**.

2. **A(B + C) = AB + AC**.

3. **(A + B)C = AC + BC**.

Definition 2.1.17. Let $\mathbf{A}=[a_{ij}]$ be an $m\times n$ matrix and r be a real number. The **scalar multiple** of $\mathbf{A}$ by r, denoted by $r\mathbf{A}$, is an $m\times n$ matrix $\mathbf{B}=[b_{ij}]$, where $b_{ij}=ra_{ij}$ $(1\leq i\leq m,\ 1\leq j\leq n)$.

Proposition 2.1.4. (Properties of Scalar Multiplication)

Let r and s be real numbers, $\mathbf{A}$ and $\mathbf{B}$ be $m \times n$ and $n \times p$ matrices. Then the following assertions hold.

1. $r(s\mathbf{A}) = (rs)\mathbf{A}$.
2. $(r+s)\mathbf{A} = r\mathbf{A} + s\mathbf{A}$.
3. $r(\mathbf{A}+\mathbf{B}) = r\mathbf{A} + r\mathbf{B}$.
4. $\mathbf{A}(r\mathbf{B}) = r(\mathbf{AB})$.

Definition 2.1.18. Let $\mathbf{A}= [a_{ij}]$ be an $m \times n$ matrix. The $n \times m$ matrix $\mathbf{A^T} = [b_{ij}]$, where

$$b_{ij} = a_{ji} \qquad (1 \leq i \leq m,\ 1 \leq j \leq n),$$

is called the **transpose** of $\mathbf{A}$. Thus, the transpose of $\mathbf{A}$ is obtained by merely interchanging the rows and columns of $\mathbf{A}$.

Proposition 2.1.5. (Properties of the Transpose)

Let r be a scalar, $\mathbf{A}$ and $\mathbf{B}$ be $m \times n$ matrices. Then the following assertions hold.

1. $(\mathbf{A}^T)^T = \mathbf{A}$.
2. $(\mathbf{A}+\mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$.
3. $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$.
4. $(r\mathbf{A})^T = r\mathbf{A}^T$.

Definition 2.1.19. A matrix can be subdivided or partitioned into smaller matrices, is called **partitioned matrix**.

Example 2.1.3. Let

$$\mathbf{A}= \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{21} & a_{22} & a_{23} & a_{24} \end{bmatrix}$$

The matrix $\mathbf{A}$ is partitioned as

$$\left[\begin{array}{cc:cc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ \hdashline a_{31} & a_{32} & a_{33} & a_{34} \\ a_{21} & a_{22} & a_{23} & a_{24} \end{array}\right]$$

then we can define the matrices

$$\mathbf{A_1}=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \mathbf{A_2}=\begin{bmatrix} a_{13} & a_{14} \\ a_{23} & a_{24} \end{bmatrix}$$
$$\mathbf{A}_3=\begin{bmatrix} a_{31} & a_{32} \\ a_{21} & a_{22} \end{bmatrix} \quad \mathbf{A_4}=\begin{bmatrix} a_{33} & a_{34} \\ a_{23} & a_{24} \end{bmatrix}$$

and write $\mathbf{A}$ as

$$\mathbf{A}=\begin{bmatrix} \mathbf{A_1} & \mathbf{A_2} \\ \mathbf{A_3} & \mathbf{A_4} \end{bmatrix}$$

■

Definition 2.1.20. The $n \times n$ matrix $\mathbf{I_n}$, all of whose diagonal elements are 1 and the rest of entries are zero, is called the **identity matrix** of order n. If $\mathbf{A}$ is an $m \times n$ matrix, then

$$\mathbf{I_m A} = \mathbf{A I_n} = \mathbf{A}.$$

Definition 2.1.21. An $n \times n$ matrix $\mathbf{A}$ is called **nonsingular** or **invertible** if there exists an $n \times n$ matrix $\mathbf{B}$ such that

$$\mathbf{AB} = \mathbf{BA} = \mathbf{I_n}.$$

The matrix $\mathbf{B}$ is called the **inverse** of $\mathbf{A}$. If no such matrix $\mathbf{B}$ exists, then $\mathbf{A}$ is called **singular** or **noninvertible**. If the inverse of $\mathbf{A}$ exists, we shall write it as $\mathbf{A}^{-1}$. Thus

$$\mathbf{A A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I_n}.$$

Proposition 2.1.6. (Properties of Inverse)

1. If $\mathbf{A}$ is nonsingular, then $\mathbf{A}^{-1}$ is nonsingular and

$$(\mathbf{A}^{-1})^{-1} = \mathbf{A}.$$

2. If $\mathbf{A}$ and $\mathbf{B}$ are nonsingular, then $\mathbf{AB}$ is nonsingular and

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}.$$

3. If $\mathbf{A}$ is nonsingular, then $\mathbf{A}^T$ is nonsingular and

$$(\mathbf{A}^T)^{-1} = (\mathbf{A}^{-1})^T.$$

2.1.1 Elementary row operations

Since the rows of an augmented matrix correspond to the equations in the associated system, these three operations correspond to the following operations on the rows of the augmented matrix:

1. Interchange two rows.
2. Multiply a row by a nonzero constant.
3. Add a nonzero constant times one row to another row.

These are called **elementary row operations** on a matrix.

Definition 2.1.22. The **elementary row operations** on an $m \times n$ matrix $\mathbf{A} = [a_{ij}]$ are defined as follows.

Type I. Interchange rows r and s of $\mathbf{A}$. That is, the elements $a_{r1}, a_{r2}, \ldots, a_{rn}$ replace the elements $a_{s1}, a_{s2}, \ldots, a_{sn}$ and the elements $a_{s1}, a_{s2}, \ldots, a_{sn}$ replace the elements $a_{r1}, a_{r2}, \ldots, a_{rn}$.

Type II. Multiply row r of $\mathbf{A}$ by $c \neq 0$. That is, the elements $a_{r1}, a_{r2}, ..., a_{rn}$ are replaced by the elements $ca_{r1}, ca_{r2}, ..., ca_{rn}$.

Type III. Add a multiple nonzero constant d of row r of $\mathbf{A}$ to row s of $\mathbf{A}$, writing the result in row s. That is, the elements $a_{s1} + da_{r1}, a_{s2} + da_{r2}, ..., a_{sn} + da_{rn}$ replace the elements $a_{s1}, a_{s2}, ..., a_{sn}$.

2.1.2 Gaussian elimination

In this section, we will present the procedure for solving a linear system. The procedure is based on the idea of performing certain operations on rows of the augmented matrix for the system that simplifies it.

Definition 2.1.23. A matrix is in **reduced row echelon form** if it satisfies the following properties:

1. If a row does not consist entirely of zeros, then the first nonzero number in the row is a 1. We call this a **leading 1**.

2. If there are any rows that consist entirely of zeros, then they are grouped together at the bottom of the matrix.

3. In any two successive rows that do not consist entirely of zeros, the leading 1 in the lower row occurs farther to the right than the leading 1 in the higher row.

4. Each column that contains a leading 1 has zeros everywhere else in that column.

The procedure we have just described for reducing a matrix to reduced row echelon form is called **Gauss-Jordan elimination**.

Theorem 2.1.7. Every $m \times n$ matrix can be transformed to reduced row echelon form by a finite sequence of elementary row operations.

Definition 2.1.24. An $m \times n$ matrix $\mathbf{A}$ is said to be **row equivalent** to an $m \times n$ matrix $\mathbf{B}$ if $\mathbf{B}$ can be obtained from $\mathbf{A}$ by applying a finite sequence of elementary row operations to $\mathbf{A}$.

Example 2.1.4. Let

$$\mathbf{A} = \begin{bmatrix} 3 & -2 & 2 & 5 \\ 1 & 2 & 0 & 3 \\ 4 & 2 & 3 & -4 \end{bmatrix}$$

Adding the first row of $\mathbf{A}$ to the third row of $\mathbf{A}$, we obtain

$$\mathbf{B} = \begin{bmatrix} 3 & -2 & 2 & 5 \\ 1 & 2 & 0 & 3 \\ 7 & 0 & 5 & 1 \end{bmatrix}$$

So, $\mathbf{B}$ is row equivalent to $\mathbf{A}$.

■

Definition 2.1.25. The **rank** of an $m \times n$ matrix $\mathbf{A}$ is the number of nonzero rows in the matrix in reduced row echelon form that is row equivalent to $\mathbf{A}$.

Example 2.1.5. Let

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & -1 & 7 & -2 \\ 0 & 1 & 0 & -3 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Since $\mathbf{A}$ is in reduced row echelon form, the rank of $\mathbf{A}$ is 2.

■

2.1.3 Linear systems

In this section, we describe about the linear system which we will use to construct the proposed algorithm. More generally, a linear system of m equations in n variables that can be expressed in the form of the linear system of m equations in n unknowns stated as follows:

$$\begin{array}{ccccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \\ a_{m1}x_1 & + & a_{m2}x_2 & + & a_{m3}x_3 & + & \cdots & + & a_{mn}x_n & = & b_m \end{array} \tag{2.1.3}$$

We can be written the problem (2.1.3) in the matrix form by letting

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

Then, the equation (2.1.3) can be written as $\mathbf{Ax} = \mathbf{b}$. The matrix $\mathbf{A}$ is called the **coefficient matrix** of the linear system, and the matrix

$$[\mathbf{A} : \mathbf{b}] = \left[\begin{array}{ccccc:c} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} & b_m \end{array}\right]$$

obtained by adjoining $\mathbf{b}$ to $\mathbf{A}$, is called the **augmented matrix**.

Theorem 2.1.8. Let $\mathbf{Ax} = \mathbf{b}$ and $\mathbf{Cx} = \mathbf{d}$ be two linear systems, each consisting of m equations in n unknowns. If the augmented matrices $[\mathbf{A} : \mathbf{b}]$ and $[\mathbf{C} : \mathbf{d}]$ are row equivalent, then both linear systems have no solutions or they have exactly the same solutions.

Definition 2.1.26. (Homogeneous Linear Systems).

The linear system of the following form:

$$\begin{array}{ccccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \cdots & + & a_{1n}x_n & = & 0 \\ a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & + & \cdots & + & a_{2n}x_n & = & 0 \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \\ a_{m1}x_1 & + & a_{m2}x_2 & + & a_{m3}x_3 & + & \cdots & + & a_{mn}x_n & = & 0 \end{array} \tag{2.1.4}$$

called a **homogeneous linear system**.

Observe that a homogeneous system (2.1.4) always has the solution $x_1 = x_2 = \cdots = x_n = 0$, which is called the **trivial solution**. A homogeneous linear system may also have a solution in which not all x_i are zero. Such a solution is called a **nontrivial solution**.

2.2 Linear programming problems

In this section, we describe the general formats of a linear programming problem. The possible solutions of a linear programming problem is presented in this section. Another problem which is corresponded with the linear programming problem is proposed.

2.2.1 Linear programming formats

There are two formats for representing a linear programming problem. That are a standard form and a canonical form.

A linear programming problem is said to be in **standard form** if all constraints are equal type and all variables are nonnegative as the following form:

$$\begin{array}{lrcl} \text{min or max} & z & = & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \mathbf{Ax} & = & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0}. \end{array}$$

A minimization problem is in **canonical form** if all variables are nonnegative and all constraints are of the $\geq$ type as the following form:

$$\begin{array}{lrcl} \min & z & = & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \mathbf{Ax} & \geq & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0}. \end{array}$$

A maximization problem is in **canonical form** if all the variables are nonnegative and all constraints are of the $\leq$ type as the following form.

$$\begin{array}{lrcl} \max & z & = & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \mathbf{Ax} & \leq & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0}. \end{array}$$

2.2.2 Problem manipulations

In many methods, the algorithm was designed to deal with the problem in the standard form. Thus, we have to change the problem to another problem which is a corresponding problem as below.

- **Minimization problem as maximization problem**

Another problem manipulation is to convert a maximization problem into a minimization problem and conversely. Note that over any region,

$$\min \sum_{j=1}^{n} c_j x_j = -\max (-\sum_{j=1}^{n} c_j x_j)$$

Hence, a maximization (minimization) problem can be converted into a minimization (maximization) problem by multiplying the coefficients of the objective function by constant (-1). After the optimization of the new problem is completed, the objective value of the old problem is -1 times the optimal objective value of the new problem.

- **Reversing an inequality constraints**

An inequality with a negative right-hand-side value can be transformed into an inequality with a positive right-hand-side value. To illustrate, consider the *i*th constraint given by

$$\sum_{j=1}^{n} a_{ij}x_j \quad \geq \quad -b_i.$$

If we multiply the inequality by -1, we obtain the inequality

$$-\sum_{j=1}^{n} a_{ij}x_j \quad \leq \quad b_i.$$

- **Changing an inequality to an equality constraints**

An inequality can be transformed into an equation. To illustrate, consider the constraint given by

$$\sum_{j=1}^{n} a_{ij}x_j \quad \geq \quad b_i.$$

This constraint can be put in an equation form by subtracting the nonnegative **surplus variable**, denoted by s_i, which is leading to

$$\sum_{j=1}^{n} a_{ij}x_j \quad - \quad s_i \quad = \quad b_i \quad \text{and} \quad s_i \quad \geq \quad 0.$$

Also the constraint

$$\sum_{j=1}^{n} a_{ij}x_j \quad \leq \quad b_i$$

can be put in an equation form by adding the nonnegative **slack variable**, denoted by s_i, which is leading

$$\sum_{j=1}^{n} a_{ij}x_j \quad + \quad s_i \quad = \quad b_i \quad \text{and} \quad s_i \quad \geq \quad 0.$$

- **Changing an equality to an inequality constraints**

An equality can be transformed into an inequality. To illustrate, consider the constraint given by

$$\sum_{j=1}^{n} a_{ij}x_j \;=\; b_i.$$

This equation can be transformed into the two inequalities

$$\sum_{j=1}^{n} a_{ij}x_j \;\leq\; b_i \quad \text{and} \quad \sum_{j=1}^{n} a_{ij}x_j \;\geq\; b_i.$$

- **Nonnegativity of the variables**

The variable x_j is called **unrestricted** in sign if it can be positive, zero or negative. If a variable x_j is unrestricted in sign, then it can be converted to two new nonnegative variables as follows:

$$x_j \;=\; x_j^{+} - x_j^{-}, \quad \text{where} \quad x_j^{+}, \;\; x_j^{-} \;\geq\; 0.$$

- **Variable bounds**

If $x_j \geq l_j$, then the new variable $x_j^{'} = x_j - l_j$ is automaticcally nonnegative. Also, if a variable x_j is restricted such that $x_j \leq u_j$, where $u_j \leq 0$, then the substitution $x_j^{'} = u_j - x_j$ produces a nonnegative variable $x_j^{'}$.

2.2.3 Solution to linear programming problems

The solution of a linear programming problem depends on the constraints and the objective function. There are three possible outcomes for a linear programming problem which are stated in this section.

Definition 2.2.1 (Feasible solution)**.** A vector $\mathbf{x} \in \mathbb{R}^n$ satisfying the constraints of a linear programming problem is called a **feasible solution**. A feasible solution which maximizes or minimizes the objective function of a linear programming problem is called an **optimal solution**.

Definition 2.2.2 (Unbounded optimal solution)**.** A linear programming problem is unbounded if the value of the objective function increases along the feasible direction of the maximization linear programming problem and if the value of the objective function decreases along the feasible direction of the minimization linear programming problem.

Definition 2.2.3 (Infeasible or empty feasible region)**.** A linear programming problem is **infeasible** if it has no feasible solution.

We can conclude the solution to every problem as the following theorem.

Theorem 2.2.4. Every linear programming problem either:

1. is infeasible,
2. is unbounded,
3. has a unique optimal solution value.

For a unique optimal solution value, it may have multiple optimal solutions having the same objective function value.

2.2.4 Duality

For every linear programming problem, there is another associated linear programming problem. This new linear programming problem satisfies some very important properties. We shall call the original linear programming problem as the **primal** linear programming problem, and we shall call this related linear programming problem as the **dual** linear programming problem. We can write the dual problem as below.

Definition 2.2.5 (Canonical form of duality)**.** Suppose that the primal problem is given in the (canonical) form:

$$\begin{array}{lllll} \text{P:} & \max & \mathbf{c}^T\mathbf{x} & & \\ & \text{subject to} & \mathbf{A}\mathbf{x} & \leq & \mathbf{b} \\ & & \mathbf{x} & \geq & \mathbf{0}. \end{array}$$

Then the dual problem is defined by:

$$\begin{aligned} \text{D:} \quad & \min && \mathbf{b}^T\mathbf{w} \\ & \text{subject to} && \mathbf{A}^T\mathbf{w} \geq \mathbf{c} \\ & && \mathbf{w} \geq \mathbf{0}. \end{aligned}$$

Definition 2.2.6 (Standard form of duality)**.** Suppose that the primal problem is given in the (standard) form:

$$\begin{aligned} \text{P:} \quad & \max && \mathbf{c}^T\mathbf{x} \\ & \text{subject to} && \mathbf{A}\mathbf{x} = \mathbf{b} \\ & && \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Then, the dual problem is defined by:

$$\begin{aligned} \text{D:} \quad & \min && \mathbf{b}^T\mathbf{w} \\ & \text{subject to} && \mathbf{A}^T\mathbf{w} \geq \mathbf{c} \\ & && \mathbf{w} \text{ unrestricted}. \end{aligned}$$

Both the primal and dual problems can be solved by using various methods. But, the well-known methods are the graphical method, the simplex method, or the interior point method. Since we would like to improve the simplex method, it will be described in details in the next section.

2.3 Methods for solving a linear programming problem

In this section, we present many methods for solving a linear programming problem, starting by the simplex algorithm. Then, other methods which improve the simplex method is proposed.

2.3.1 Simplex method

The simplex method is used for solving a linear programming problem which was developed by Dantzig [1]. It is an iterative process which makes the use

of Gauss-Jordan elimination techniques. The goal of the simplex method is to obtain a feasible solution to the objective function satisfying all constraints.

The simplex method starts at a basic feasible solution and moves to the another basic feasible solution until the optimal solution is found. The basic feasible solution can be defined below.

Definition 2.3.1. (Basic Feasible Solution)

Consider a linear programming problem in the standard form:

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned} \tag{2.3.1}$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{c}, \mathbf{x} \in \mathbb{R}^n$.

Suppose that rank ($\mathbf{A}$, $\mathbf{b}$) = rank($\mathbf{A}$) = m. After possibly rearranging the columns of $\mathbf{A}$, let $\mathbf{A}$ = [$\mathbf{B}$, $\mathbf{N}$] where $\mathbf{B}$ is an $m \times m$ invertible matrix and $\mathbf{N}$ is an $m \times (n-m)$ matrix. The solution $\mathbf{x} = \begin{bmatrix} \mathbf{x_B} \\ \mathbf{x_N} \end{bmatrix}$ to the equations $\mathbf{Ax} = \mathbf{b}$, where

$$\mathbf{x_B} = \mathbf{B}^{-1}\mathbf{b} \quad \text{and} \quad \mathbf{x_N} = \mathbf{0}$$

is called a **basic solution** of the system $\mathbf{Ax} = \mathbf{b}$. If $\mathbf{x_B} \geq \mathbf{0}$, then $\mathbf{x}$ is called a **basic feasible solution** of the system (2.3.1). Here $\mathbf{B}$ is called the **basic matrix** or **basis** and $\mathbf{N}$ is called the **nonbasic matrix**. The components of $\mathbf{x_B}$ are called **basic variables** and the components of $\mathbf{x_N}$ are called **nonbasic variables**.

2.3.2 Algebra of the simplex method

Let $\mathbf{A} = [\mathbf{A}_{:1}, \mathbf{A}_{:2}, ..., \mathbf{A}_{:n}] = [\mathbf{B}, \mathbf{N}]$ where $\mathbf{A}_{:j} \in \mathbb{R}^m$ is a column vector j of matrix $\mathbf{A}$, $\mathbf{B} \in \mathbb{R}^{m\times m}$, $\mathbf{N} \in \mathbb{R}^{m\times(n-m)}$. Assume that $I_\mathbf{B}$, $I_\mathbf{N}$ are the index sets of basic variables and nonbasic variables, respectively. Then, the problem (2.3.1) can be written

as follows:

$$\begin{aligned} \max \quad & z = \mathbf{c}_\mathbf{B}^T\mathbf{x}_\mathbf{B} + \mathbf{c}_\mathbf{N}^T\mathbf{x}_\mathbf{N} \\ \text{subject to} \quad & \mathbf{B}\mathbf{x}_\mathbf{B} + \mathbf{N}\mathbf{x}_\mathbf{N} = \mathbf{b} \\ & \mathbf{x}_\mathbf{B}, \mathbf{x}_\mathbf{N} \geq \mathbf{0}. \end{aligned} \tag{2.3.2}$$

Suppose that we have a basic feasible solution $\mathbf{x} = \begin{bmatrix} \mathbf{x}_\mathbf{B} \\ \mathbf{x}_\mathbf{N} \end{bmatrix} = \begin{bmatrix} \mathbf{B}^{-1}\mathbf{b} \\ \mathbf{0} \end{bmatrix}$ whose objective value z_0 is given by

$$z_0 = \mathbf{c}^T\mathbf{x} = \left[\mathbf{c}_\mathbf{B}^T, \mathbf{c}_\mathbf{N}^T\right] \begin{bmatrix} \mathbf{x}_\mathbf{B} \\ \mathbf{x}_\mathbf{N} \end{bmatrix} = \left[\mathbf{c}_\mathbf{B}^T, \mathbf{c}_\mathbf{N}^T\right] \begin{bmatrix} \mathbf{B}^{-1}\mathbf{b} \\ \mathbf{0} \end{bmatrix} = \mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{b}. \tag{2.3.3}$$

Consider the constraint

$$\begin{aligned} \mathbf{A}\mathbf{x} &= \mathbf{b} \\ \left[\mathbf{B}, \ \mathbf{N}\right] \begin{bmatrix} \mathbf{x}_\mathbf{B} \\ \mathbf{x}_\mathbf{N} \end{bmatrix} &= \mathbf{b} \\ \mathbf{B}\mathbf{x}_\mathbf{B} + \mathbf{N}\mathbf{x}_\mathbf{N} &= \mathbf{b}. \end{aligned}$$

Multiplying the last equation by $\mathbf{B}^{-1}$ and rearranging the terms, we get

$$\mathbf{x}_\mathbf{B} = \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_\mathbf{N}. \tag{2.3.4}$$

Consider the objective function denoted by z :

$$\begin{aligned} z = \mathbf{c}^T\mathbf{x} = \left[\mathbf{c}_\mathbf{B}^T, \mathbf{c}_\mathbf{N}^T\right] \begin{bmatrix} \mathbf{x}_\mathbf{B} \\ \mathbf{x}_\mathbf{N} \end{bmatrix} &= \mathbf{c}_\mathbf{B}^T\mathbf{x}_\mathbf{B} + \mathbf{c}_\mathbf{N}^T\mathbf{x}_\mathbf{N} \\ &= \mathbf{c}_\mathbf{B}^T(\mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_\mathbf{N}) + \mathbf{c}_\mathbf{N}^T\mathbf{x}_\mathbf{N} \\ &= \mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{b} - \mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{N}\mathbf{x}_\mathbf{N} + \mathbf{c}_\mathbf{N}^T\mathbf{x}_\mathbf{N} \\ &= \mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{b} - (\mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{N}\mathbf{x}_\mathbf{N} - \mathbf{c}_\mathbf{N}^T\mathbf{x}_\mathbf{N}) \\ &= \mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{b} - (\mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{N} - \mathbf{c}_\mathbf{N}^T)\mathbf{x}_\mathbf{N} \\ &= \mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{b} - \sum_{j \in I_N} (\mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{A}_{:j} - c_j)x_j \end{aligned}$$

Let $\mathbf{y}_j = \mathbf{B}^{-1}\mathbf{A}_{:j}$, $z_j = \mathbf{c}_\mathbf{B}^T\mathbf{y}_j$ where $j \in I_N$, $\overline{\mathbf{b}} = \mathbf{B}^{-1}\mathbf{b}$ and $z_0 = \mathbf{c}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{b}$. Then, the

problem (2.3.2) can be written as follows:

$$\begin{aligned} \max \quad & z = z_0 - \sum_{j \in I_N} (z_j - c_j) x_j \\ \text{subject to} \quad & \mathbf{x_B} + \sum_{j \in I_N} (\mathbf{y}_j) x_j = \overline{\mathbf{b}} \\ & x_j \geq 0, j \in I_N, \mathbf{x_B} \geq \mathbf{0} \end{aligned} \tag{2.3.5}$$

The key result is the value of $z_j - c_j$ which is called the **reduced cost**. If $z_j - c_j \geq 0$ for all $j \in I_N$, then the objective value decreases when x_j increases. Thus, x_j should not be increased, that is, the current basic feasible solution is optimal. But if there is at least one $z_k - c_k < 0$ for $k \in I_N$, then the objective value increases when x_k increases. Thus, x_k should be increased.

Consider $z = z_0 - (z_k - c_k)x_k$, to determine the increasing value of the entering basic variable x_k before stopping.

Consider the kth column of the constraints

$$\mathbf{x_B} + (\mathbf{y}_k) x_k = \overline{\mathbf{b}}$$

$$\begin{bmatrix} x_{\mathbf{B}_1} \\ x_{\mathbf{B}_2} \\ \vdots \\ x_{\mathbf{B}_r} \\ \vdots \\ x_{\mathbf{B}_m} \end{bmatrix} + \begin{bmatrix} y_{1k} \\ y_{2k} \\ \vdots \\ y_{rk} \\ \vdots \\ y_{mk} \end{bmatrix} x_k = \begin{bmatrix} \overline{\mathrm{b}}_1 \\ \overline{\mathrm{b}}_2 \\ \vdots \\ \overline{\mathrm{b}}_r \\ \vdots \\ \overline{\mathrm{b}}_m \end{bmatrix}$$

We rearrange the terms, we get

$$\begin{bmatrix} x_{\mathbf{B}_1} \\ x_{\mathbf{B}_2} \\ \vdots \\ x_{\mathbf{B}_r} \\ \vdots \\ x_{\mathbf{B}_m} \end{bmatrix} = \begin{bmatrix} \overline{\mathrm{b}}_1 \\ \overline{\mathrm{b}}_2 \\ \vdots \\ \overline{\mathrm{b}}_r \\ \vdots \\ \overline{\mathrm{b}}_m \end{bmatrix} - \begin{bmatrix} y_{1k} \\ y_{2k} \\ \vdots \\ y_{rk} \\ \vdots \\ y_{mk} \end{bmatrix} x_k$$

If $\mathbf{y}_k \leq \mathbf{0}$, then $x_{\mathbf{B}_i}$ increases as x_k increases, and so x_k can be increased to infinity because $x_{\mathbf{B}_i}$ is always nonnegative. Thus, the solution for this case is unbounded.

If $y_{ik} > 0$ for some $i = 1, ..., m$, then $x_{\mathbf{B}_i}$ will decrease as x_k increases. Thus, x_k can be increased as $x_{\mathbf{B}_i} = \overline{b}_i - y_{ik}x_k \geq 0$, that is $x_k \leq \dfrac{\overline{b}_i}{y_{ik}}, \quad y_{ik} > 0.$

For feasibility, we can increase x_k until:

$$x_k = \frac{\overline{b}_r}{y_{rk}} = \underset{1 \leq i \leq m}{\text{minimum}} \left\{ \frac{\overline{b}_i}{y_{ik}} : y_{ik} > 0 \right\}.$$

These calculations are referred to as the **minimum ratio test**.

When we can increase $x_k = \dfrac{\overline{b}_r}{y_{rk}}$, then $x_{\mathbf{B}_i}$ will decrease to zero. Thus, x_k will enter to be a basic variable called an **entering variable** and $x_{\mathbf{B}_i}$ will leave to be a nonbasic variable called a **leaving variable**.

The problem (2.3.5) can be rewritten as follows:

$$\begin{array}{lrcl} \max & z + \displaystyle\sum_{j \in I_N} (z_j - c_j)x_j & = & z_0 \\ \text{subject to} & \mathbf{x_B} + \displaystyle\sum_{j \in I_N} (\mathbf{y}_j)x_j & = & \overline{\mathbf{b}} \\ & x_j \geq 0, \ j \in I_N, \ \mathbf{x_B} \geq \mathbf{0}. & & \end{array} \tag{2.3.6}$$

Here, we think of z as a (basic) variable to be maximize. The objective row will be referred to as row 0 and the remaining rows are rows 1 through m. The right-hand-side column (RHS) will denote the values of the basic variables. The current basic feasible solution is represented by basis $\mathbf{B}$ in the following tableau.

	z	$\mathbf{x_B}$	$\mathbf{x_N}$	RHS
	1	$\mathbf{0}$	$\mathbf{c_B^T B^{-1} N - c_N^T}$	$\mathbf{c_B^T B^{-1} b}$
$\mathbf{x_B}$	0	$\mathbf{I}$	$\mathbf{B^{-1} N}$	$\mathbf{B^{-1} b}$

The steps of the simplex algorithm can be summarized as follows:

The Simplex Method in Tableau Format (Maximization Problem)

Initialization Step

Find an initial basic feasible solution with basis **B**. Form the following initial tableau:

	z	$\mathbf{x_B}$	$\mathbf{x_N}$	RHS
	1	$\mathbf{0}$	$\mathbf{c_B^T B^{-1} N - c_N^T}$	$\mathbf{c_B^T B^{-1} b}$
$\mathbf{x_B}$	0	$\mathbf{I}$	$\mathbf{B^{-1} N}$	$\mathbf{B^{-1} b}$

Main Step

1. Let $z_k - c_k = \min\{z_j - c_j : j \in I_N\}$, where $z_j - c_j = \mathbf{c_B^T B^{-1} N}_{:j} - c_j$.
   (a) If $z_k - c_k \geq 0$, then the current tableau is optimal and ends.
   (b) Otherwise, $z_k - c_k < 0$, examine $\mathbf{y}_k$ in step 2.
2. Examine $\mathbf{y}_k$ (coefficient in the k^{th} column)
   (a) If $\mathbf{y}_k \leq 0$, then the linear programming problem is unbounded and ends.
   (b) Otherwise, That is $\mathbf{y}_k \nleq 0$, determine the index row r (minimum ratio test) as follows:
   $$\frac{\overline{b}_r}{y_{rk}} = \underset{1 \leq i \leq m}{\text{minimum}} \left\{ \frac{\overline{b}_i}{y_{ik}} : y_{ik} > 0 \right\}$$
3. Update the tableau by pivoting at y_{rk}. Update the basic and nonbasic variables where x_k enters the basis and $x_{\mathbf{B}_r}$ leaves the basis. That is, y_{rk} is divided by itself to change to 1 and other entries in the k^{th} column are updated to be 0 by using row operation. Then, repeat the main step.

Example 2.3.1. Consider the following linear programming problem:

$$\begin{array}{lrcrcrcrl} \text{maximize} & z & = & 8x_1 & + & 9x_2 & + & 5x_3 & \\ \text{subject to} & & & x_1 & + & x_2 & + & 2x_3 & \leq 2 \\ & & & 2x_1 & + & 3x_2 & + & 4x_3 & \leq 3 \\ & & & 6x_1 & + & 6x_2 & + & 2x_3 & \leq 8 \\ & & & x_1, & & x_2, & & x_3 & \geq 0. \end{array}$$

We first convert the problem to standard form by adding slack variables x_4, x_5, x_6 as follows:

$$\begin{array}{lrcrcrcrcrcrcl} \text{maximize} & z & = & 8x_1 & + & 9x_2 & + & 5x_3 & & & & & \\ \text{subject to} & & & x_1 & + & x_2 & + & 2x_3 & + & x_4 & & & = 2 \\ & & & 2x_1 & + & 3x_2 & + & 4x_3 & & + x_5 & & & = 3 \\ & & & 6x_1 & + & 6x_2 & + & 2x_3 & & & + & x_6 & = 8 \\ & & & x_1, & & x_2, & & x_3, & & x_4, & & x_5, \quad x_6 & \geq 0. \end{array}$$

We can choose an initial basis as $\mathbf{B} = [\mathbf{A}_{:4}, \mathbf{A}_{:5}, \mathbf{A}_{:6}] = \mathbf{I}_3$ and $\mathbf{b} = \begin{bmatrix} 2 \\ 3 \\ 8 \end{bmatrix} > \mathbf{0}$. Then we have $\mathbf{B}^{-1}\mathbf{b} = \overline{\mathbf{b}} \geq \mathbf{0}$ which is a basic feasible solution. Thus, we can start the simplex method. The initial tableau is as below.

	z	x_1	x_2	x_3	x_4	x_5	x_6	RHS
	1	-8	-9	-5	0	0	0	0
x_4	0	1	1	2	1	0	0	2
x_5	0	2	3	4	0	1	0	3
x_6	0	6	6	2	0	0	1	8

Since $\min\{z_j - c_j : j \in \{1,2,3\}\} = z_2 - c_2 = -9 < 0$, x_2 is the entering variable. Then, we examine $\mathbf{y}_2$ and determine the index row r:

$$\frac{\overline{b}_r}{y_{r2}} = \underset{1 \leq i \leq 3}{\text{minimum}} \left\{ \frac{\overline{b}_1}{y_{12}}, \frac{\overline{b}_2}{y_{22}}, \frac{\overline{b}_3}{y_{32}} \right\} = \min \left\{ \frac{2}{1}, \frac{3}{3}, \frac{8}{6} \right\}.$$

Thus, x_5 is the leaving variable. We update the tableau by pivoting at y_{22} and the basic and nonbasic variables where x_2 enters the basis and x_5 leaves the basis. We get the following tableau.

	z	x_1	x_2	x_3	x_4	x_5	x_6	RHS
	1	-2	0	7	0	3	0	9
x_4	0	$1/3$	0	$2/3$	1	$-1/3$	0	1
x_2	0	$2/3$	1	$4/3$	0	$1/3$	0	1
x_6	0	2	0	-6	0	-2	1	2

Since $\min\{z_j - c_j : j \in \{1,3,5\}\} = z_1 - c_1 = -2 < 0$, x_1 is the entering variable. Then, we examine $\mathbf{y}_1$ and compute the minimum ratio:

$$\frac{\overline{b}_r}{y_{r1}} = \underset{1\le i\le 3}{\text{minimum}}\left\{\frac{\overline{b}_1}{y_{11}}, \frac{\overline{b}_2}{y_{21}}, \frac{\overline{b}_3}{y_{31}}\right\} = \min\left\{\frac{1}{1/3}, \frac{1}{2/3}, \frac{2}{2}\right\}.$$

Thus, x_6 is the leaving variable. We update the tableau by pivoting at y_{31} and the basic and nonbasic variables where x_1 enters the basis and x_6 leaves the basis. We get the following tableau.

	z	x_1	x_2	x_3	x_4	x_5	x_6	RHS
	1	0	0	1	0	1	1	11
x_4	0	0	0	$5/3$	1	0	$-1/6$	$2/3$
x_2	0	0	1	$10/3$	0	1	$-1/3$	$1/3$
x_1	0	1	0	-3	0	-1	$1/2$	1

This is the optimal tableau since $z_k - c_k \ge 0$ for all nonbasic variables. The optimal solution is given by

$$x_1^* = 1, \quad x_2^* = \frac{1}{3}, \quad x_3^* = 0 \quad \text{with} \quad z^* = 11.$$

■

By the previous example, we see that we can pick **B = I** easily and start the simplex method immediately due to the feasibility of the origin point. However, the basic feasible solution which is constructed by **B** may not be easily found. Thus, we will add the **artificial variables** for picking **B = I** to start the simplex method.

Suppose that the constraints are of the following form:

$$\begin{aligned} \mathbf{Ax} &= \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0}, \end{aligned} \tag{2.3.7}$$

where $\mathbf{A}$ is an $m \times n$ matrix and $\mathbf{b}$ is a nonnegative m vector. Furthermore, suppose that $\mathbf{A}$ has no identity submatrix which we have no obvious starting basic feasible solution. In this case, we shall add artificial vector $\mathbf{x}_a$ leading to the system:

$$\begin{aligned} \mathbf{Ax} + \mathbf{x}_a &= \mathbf{b} \\ \mathbf{x},\ \mathbf{x}_a &\geq \mathbf{0}. \end{aligned} \tag{2.3.8}$$

Note that by adding the artificial vector, we created an identity matrix ($\mathbf{I}$) corresponding to the artificial vector. Then, the new constraint matrix is [$\mathbf{A}$, $\mathbf{I}$] and the basic feasible solution of the new system is found immediately, namely, $\mathbf{x}_a = \mathbf{b}$ and $\mathbf{x} = \mathbf{0}$. Therefore, the initial basic feasible solution is $[\mathbf{x}, \mathbf{x}_a]^T = [\mathbf{0}, \mathbf{b}]^T \geq \mathbf{0}$. Now, we have a starting basic feasible solution and the simplex method can be applied.

For a linear programming problem with adding artificial variables, there are many methods for solving it which deals with artificial variables. But, there are two well-known methods, that are, two-phase method and Big-M method.

2.3.3 Two-phase method

There are various methods that can be used to eliminate artificial variables. One of these methods is to minimize the sum of artificial variables, subject to the constraints

$$\begin{aligned} \mathbf{Ax} + \mathbf{x}_a &= \mathbf{b} \\ \mathbf{x},\ \mathbf{x}_a &\geq \mathbf{0}. \end{aligned} \tag{2.3.9}$$

If the original problem has a feasible solution, then the optimal value of this problem is zero, where all artificial variables drop to zero ($\mathbf{x}_a = \mathbf{0}$). Then, other variables enter instead artificial variables. The basis consists of other variables which except the artificial variable. In other words, we get a basic feasible solution for the original system (2.3.7) and the simplex method can start with the original objective function $\mathbf{c}^T\mathbf{x}$. On the other hand, after solving this problem we have a positive artificial variable, then the original problem has no feasible solution. This procedure is called the **two-phase method** which is summarized as follows:

Phase I

In this phase, we reduce artificial variables to zero or conclude that the original problem has no feasible solutions.

The algorithm starts by solving the following linear programming problem starting with the basic feasible solution $\mathbf{x} = \mathbf{0}$ and $\mathbf{x}_a = \mathbf{b}$:

$$\begin{array}{lrcrcl} \min & x_0 & = & \mathbf{1}^T\mathbf{x}_a & & \\ \text{subject to} & \mathbf{A}\mathbf{x} & + & \mathbf{x}_a & = & \mathbf{b} \\ & \mathbf{x}, & & \mathbf{x}_a & \geq & \mathbf{0}. \end{array} \tag{2.3.10}$$

At optimality, if $\mathbf{x}_a^* \neq \mathbf{0}$, then stop; the original problem has no feasible solutions. Otherwise, let the basic and nonbasic variables be $\mathbf{x_B}$ and $\mathbf{x_N}$ and proceed to Phase II.

Phase II

Solve the following linear programming problem starting with the basic feasible solution $\mathbf{x_B} = \mathbf{B}^{-1}\mathbf{b}$ and $\mathbf{x_N} = \mathbf{0}$:

$$\begin{array}{lrcl} \min \text{ or } \max & z & = & \mathbf{c}_\mathbf{B}^T\mathbf{x_B} + \mathbf{c}_\mathbf{N}^T\mathbf{x_N} \\ \text{subject to} & \mathbf{x_B} + \mathbf{B}^{-1}\mathbf{N}\mathbf{x_N} & = & \mathbf{B}^{-1}\mathbf{b} \\ & \mathbf{x_B}, \quad \mathbf{x_N} & \geq & \mathbf{0}. \end{array} \tag{2.3.11}$$

Analysis of the two-phase Method

At the end of Phase I, either $\mathbf{x}_a^* \neq \mathbf{0}$ or $\mathbf{x}_a^* = \mathbf{0}$. These two cases are discussed in details below.

Case A: $\mathbf{x}_a^* \neq \mathbf{0}$

If $\mathbf{x}_a^* \neq \mathbf{0}$, then the original problem has no feasible solution, because if there is an $\mathbf{x} \geq \mathbf{0}$ with $\mathbf{Ax} = \mathbf{b}$, then $\begin{bmatrix} \mathbf{x} \\ \mathbf{0} \end{bmatrix}$ is a feasible solution of the Phase I problem and $\mathbf{0}(\mathbf{x}) + \mathbf{1}(\mathbf{0}) = \mathbf{0} < \mathbf{1}^T\mathbf{x}_a^*$, violating optimality of $\mathbf{x}_a^*$.

Case B: $\mathbf{x}_a^* = \mathbf{0}$

This case is divided into two subcases.

Subcase B1: All artificials are out of the basis

In this subcase, we have identified a basic feasible solution $\mathbf{x} = [\mathbf{x_B}, \mathbf{x_N}]^T$. Simply allow the non-zero basic element (in $\mathbf{x}$) to be $\mathbf{x_B}$ and the remainder of the elements (excluding $\mathbf{x}_a$) are in $\mathbf{x_N}$. Then, we can start Phase II using the basic feasible solution.

Subcase B2: Some artificials are in the basis at zero values

In this subcase, we have identified a degenerate solution to the Phase I problem. We may proceed directly to Phase II, assigning 0 coefficients to artificial variables as long as we ensure that no artificial variable ever becomes positive again.

Example 2.3.2. Consider the following linear programming problem:

$$\begin{array}{lrcrcrcl} \max & 3x_1 & + & x_2 & - & 4x_3 & & \\ \text{subject to} & x_1 & + & x_2 & - & x_3 & = & 1 \\ & & & x_2 & & & \geq & 2 \\ & x_1, & & x_2, & & x_3 & \geq & 0. \end{array}$$

Phase I

After introducing the slack variable s_1 and artificial variables x_{a_1}, x_{a_2} the following problem is obtained:

$$\begin{array}{lrcrcrcrcrcl} \min & x_{a_1} & + & x_{a_2} & & & & & & & & \\ \text{subject to} & x_1 & + & x_2 & - & x_3 & & + & x_{a_1} & & = & 1 \\ & & & x_2 & & & - & s_1 & & + & x_{a_2} & = & 2 \\ & x_1, & & x_2, & & x_3, & & s_1, & x_{a_1}, & & x_{a_2} & \geq & 0. \end{array}$$

We can choose an initial basis as $\mathbf{B} = [\mathbf{A}_{:5}, \mathbf{A}_{:6}] = \mathbf{I}_2$ and fill coefficeint matrix into the tableau:

	z	x_1	x_2	x_3	s_1	x_{a_1}	x_{a_2}	RHS
	1	0	0	0	0	−1	−1	0
x_{a_1}	0	1	1	−1	0	1	0	1
x_{a_2}	0	0	1	0	−1	0	1	2

Since x_{a_1} and x_{a_2} are in basis, their reduced cost are zero. Then, the initial tableau is as below.

	z	x_1	x_2	x_3	s_1	x_{a_1}	x_{a_2}	RHS
	1	1	2	−1	−1	0	0	3
x_{a_1}	0	1	1	−1	0	1	0	1
x_{a_2}	0	0	1	0	−1	0	1	2

Since $\max\{z_j - c_j : j \in \{1,2,3,4\}\} = z_2 - c_2 = 2 > 0$, x_2 is the entering variable. Then, we examine $\mathbf{y}_2$ and compute the minimum ratio:

$$\frac{\bar{b}_r}{y_{r2}} = \underset{1 \le i \le 2}{\text{minimum}} \left\{ \frac{\bar{b}_1}{y_{12}}, \frac{\bar{b}_2}{y_{22}} \right\} = \min \left\{ \frac{1}{1}, \frac{2}{1} \right\}.$$

Thus, x_{a_1} is the leaving variable. We update the tableau by pivoting at y_{12} and the basic and nonbasic variables where x_2 enters the basis and x_{a_1} leaves the basis. We get the following tableau.

	z	x_1	x_2	x_3	s_1	x_{a_1}	x_{a_2}	RHS
	1	−1	0	1	−1	−1	0	1
x_2	0	1	1	−1	0	1	0	1
x_{a_2}	0	−1	0	1	−1	−1	1	1

Since $\max\{z_j - c_j : j \in \{1,3,4,5\}\} = z_3 - c_3 = 1 > 0$, x_3 is the entering variable. Then, we examine $\mathbf{y}_3$ and compute the minimum ratio:

$$\frac{\bar{b}_r}{y_{r3}} = \underset{1 \le i \le 2}{\text{minimum}} \left\{ \frac{\bar{b}_2}{y_{23}} \right\} = \min \left\{ \frac{1}{1} \right\}.$$

Thus, x_{a_2} is the leaving variable. We update the tableau by pivoting at y_{23} and the basic and nonbasic variables where x_3 enters the basis and x_{a_2} leaves the basis. We get the following tableau.

	z	x_1	x_2	x_3	s_1	x_{a_1}	x_{a_2}	RHS
	1	0	0	0	0	−1	−1	0
x_2	0	0	1	0	−1	0	1	2
x_3	0	−1	0	1	−1	−1	1	1

Since $x_{a_1}, x_{a_2} = 0$, Phase II starts with the current basic feasible solution.

Phase II

In this phase, the original objective is used. Thus, the reduced cost of starting tableau of Phase II as below:

	z	x_1	x_2	x_3	s_1	RHS
	1	1	0	0	3	-2
x_2	0	0	1	0	-1	2
x_3	0	-1	0	1	-1	1

This is the optimal tableau since $z_k - c_k \geq 0$ for all nonbasic variables. The optimal solution is given by

$$x_1^* = 0, x_2^* = 2, x_3^* = 1, \text{ and } s_1^* = 0 \text{ with } z = -2.$$

■

By the above examples, both the simplex method and the two-phase start if there is a basic feasible solution from choosing the basic matrix. But if we cannot select the basic matrix, then artificial variables are added for picking the basic matrix easily. It can guarantee that the basic feasible solution can be certainly found. However, adding artificial variables will enlarge the size of the computational matrix, and the calculation time is higher. Later, many researchers proposed many methods for finding the basic feasible solution without using artificial variables. These method are proposed in the next section.

2.3.4 Arsham's algorithm

Arsham's algorithm proposed by Arsham [3] in 1997. It is constucted for solving the linear programming problem. This algorithm avoids the use of artificial variables. It consists of two phases. In Phase I, it begins with the empty basic variable set, then a nonbasic variable is chosen into the basic variable set one by one by considering the coefficient of nonbasic variables in the objective function including the minimum ratio. This phase can report that a problem is infeasible or the basic variable

set is full. After Phase I ends, if the basic variable set is full, Phase II is used to find the optimal solution by the ordinary simplex method.

Before the algorithm starts, the linear programming problem must be converted into the following form:

$$\begin{aligned} \max \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathbf{Ax} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{2.3.12}$$

Process of Arsham's algorithm

All coefficients are filled as the following tableau.

BVS	$\mathbf{x}$	RHS	C/R
$-$	c_j	$-$	$-$
?	$\mathbf{A}$	$\mathbf{b}$	

where the column C/R is the column of ratios for finding the minimum ratio.

PHASE 1: BVS AUGMENTATION PHASE

Step 1. Construct the initial tableau which is always considered to have empty basic variable set as the above tableau.

Step 2. Is the basic variable set complete?

- If yes, go to Phase 2.
- Otherwise, continue the pivot column selection in Step 3.

Step 3. A candidate to enter into the basic variable set is a nonbasic variable with the largest c_j.

Step 4. Compute the column ratio C/R . Is the smallest nonnegative C/R in an unoccupied row?

- If yes, the unoccupied row is the pivot row. Continue to enter the variable into basic variable set by performing the Guass-Jordan row operations, and return to Step 2.

- If not, go to Step 5.

Step 5. Is there any other nonbasic variable as a candidate to enter?

- If not, the problem is infeasible.

- Otherwise, choose the next pivot column with the largest c_j and go to Step 4.

PHASE 2: OPTIMALITY (SIMPLEX) PHASE

Step 6. Are all $c_j \leq 0$ in the current tableau?

- If yes, go to step 8.

- Otherwise, go to step 7.

Step 7. Apply the simplex method rules for entering and exiting variables to and form the basic variable set. Go to step 6.

Step 8. This is an optimal tableau.

Next, we shall demonstrate steps of Arsham's algorithm using a below example.

Example 2.3.3. Consider the following linear programming problem:

$$\begin{array}{lrcrcl} \max & x_1 & + & x_2 & & \\ \text{subject to} & x_1 & + & x_2 & \geq & 4 \\ & x_1 & & & \leq & 3 \\ & & & x_2 & \leq & 3 \\ & x_1, & & x_2 & \geq & 0. \end{array}$$

We first convert the problem to the standard form by adding surplus variable s_1 and slack variables s_2 and s_3 as follows.

$$
\begin{array}{lrcrcrcrcrcl}
\max & x_1 & + & x_2 & & & & & & & & \\
\text{subject to} & x_1 & + & x_2 & - & s_1 & & & & & = & 4 \\
 & x_1 & & & & & + & s_2 & & & = & 3 \\
 & & & x_2 & & & & & + & s_3 & = & 3 \\
 & x_1, & & x_2, & & s_1, & & s_2, & & s_3 & \geq & 0.
\end{array}
$$

Phase 1

The initial tableau can be written as follows:

	BVS	x_1	x_2	s_1	s_2	s_3	RHS
c_j	–	1	1	0	0	0	–
	?	1	1	−1	0	0	4
	?	1	0	0	1	0	3
	?	0	1	0	0	1	3

Since c_1 is the largest coefficient, x_1 enters the basic variable set. Then, compute the column ratio C/R as follows:

	BVS	x_1	x_2	s_1	s_2	s_3	RHS	C/R
c_j	–	1	1	0	0	0	–	
	?	1	1	−1	0	0	4	4
	?	1	0	0	1	0	3	3
	?	0	1	0	0	1	3	–

At the second row, there is the minimum C/R. So, we bring x_1 into the basic variable set in the second row. Then, we get the following tableau.

	BVS	x_1	x_2	s_1	s_2	s_3	RHS
c_j	–	0	1	0	−1	0	–
	?	0	1	−1	−1	0	1
	x_1	1	0	0	1	0	3
	?	0	1	0	0	1	3

The next variable to enter the basis is x_2. Performing the iteration, we get the following tableau.

	BVS	x_1	x_2	s_1	s_2	s_3	RHS
c_j	–	0	0	1	0	0	–
	x_2	0	1	−1	−1	0	1
	x_1	1	0	0	1	0	3
	?	0	0	1	1	1	2

Next, the nonbasic variable s_1 is entered into the basic variable set. Performing the iteration, we get the following tableau.

	BVS	x_1	x_2	s_1	s_2	s_3	RHS
c_j	–	0	0	0	−1	−1	–
	x_2	0	1	0	0	1	3
	x_1	1	0	0	1	0	3
	s_1	0	0	1	1	1	2

Since the basic variable set is full, we go to Phase 2.

Phase 2

Now, all c_j are negative. Thus, the current tableau is optimal. Then, the optimal solution is $x_1^* = 3,\ x_2^* = 3$ with $z^* = 6$.

■

Although Arsham's algorithm avoids the use of artificial variables, it makes a mistake which Enge and Huhn [4] gave a counterexample, in which Arsham's phase 1 algorithm declares the infeasibility of a feasible problem. The mistake of Arsham's algorithm is that if the variable is already chosen into the basic variable set, then other variables with minimum ratio cannot be replaced them. Thus, sometimes a basic feasible variable set cannot be found. Then, Arsham's algorithm reports infeasibility for a feasible problem. Next, Arsham's algorithm is improved by Gao [5] in 2015. In the next section, Gao's algorithm is presented.

2.3.5 Gao's algorithm

By the mistake of Arsham's algorithm, Gao [5] improved the Phase I of Arsham's algorithm in two variants called variant 1 and variant 2. Gao's algorithm starts when Arsham's algorithm reports infeasibility and the nonbasic variable with minimum ratio is not in the basic variable set. We first introduce the variant 1. It allows other nonbasic variables with minimum ratio to replace the previous variable in the basic variable set. The selection of a nonbasic variable into the basic variable set by considering the absolute of technical coefficients is still a mistake, since the infeasibility is sometimes reported for a feasible problem. For variant 2, it uses an adding constraint which is the sum of all unoccupied rows after passing Arsham's algorithm. Then, the nonbasic variable is chosen into the basic variable set by considering the largest coefficient of the added constraint. Next, we describe details of both variants as below.

- **Variant 1**

Allowing the nonbasic variable with the smallest nonnegative C/R into the basic variable set which is occupied row.

Example 2.3.4. Consider the following linear programming problem:

$$\begin{array}{lrcrcrcl} \max & 3x_1 & + & x_2 & - & 4x_3 & & \\ \text{subject to} & x_1 & + & x_2 & - & x_3 & = & 1 \\ & & & x_2 & & & \geq & 2 \\ & x_1, & & x_2, & & x_3 & \geq & 0. \end{array}$$

We first convert the problem to standard form by adding surplus variables s_1 obtaining:

$$\begin{array}{lrcrcrcrcl} \max & 3x_1 & + & x_2 & - & 4x_3 & & & \\ \text{subject to} & x_1 & + & x_2 & - & x_3 & & = & 1 \\ & & & x_2 & & & - & s_1 & = & 2 \\ & x_1, & & x_2, & & x_3, & & s_1 & \geq & 0. \end{array}$$

Phase I:

After Arsham's algorithm ends, we get the following tableau.

	BVS	x_1	x_2	x_3	s_1	RHS
c_j	–	0	−2	−1	0	–
	x_1	1	1	−1	0	1
	?	0	1	0	−1	2

Then, the Arsham's algorithm reports infeasibility. At this point, the variants can be used to follow on, instead of declaring the infeasibility of the problem. Next, we choose the nonbasic variable to enter the basic variable set. We examine the nonbasic variables s_1 and x_3, but both candidates cannot enter the basic variable set. Thus, x_2 is the last nonbasic variable for consideration.

	BVS	x_1	x_2	x_3	s_1	RHS	C/R
c_j	–	0	−2	−1	0	–	
	x_1	1	1	−1	0	1	1
	?	0	1	0	−1	2	2

At the first row, there is the minimum C/R. So, x_1 is replaced by x_2 into the basic variable set in the row, we get the following tableau.

	BVS	x_1	x_2	x_3	s_1	RHS
c_j	–	2	0	−3	0	–
	x_2	1	1	−1	0	1
	?	−1	0	1	−1	1

Next, the nonbasic variable x_3 is chosen to enter the basic variable set. After the tableau is updated, we get the following tableau.

	BVS	x_1	x_2	x_3	s_1	RHS
c_j	–	−1	0	0	−3	–
	x_2	0	1	0	−1	2
	x_3	−1	0	1	−1	1

Since the basic variable set is full, we go to Phase 2.

Phase 2:

Now, all c_j are negative. Thus, the current tableau is optimal. Then, the optimal solution is $x_1^* = 0,\ x_2^* = 2,\ x_3^* = 1$ with $z^* = -2$.

■

- **Variant 2**

After Arsham's algorithm ends and reports infeasibility, the sum of all unoccupied rows produces a new constraint, and it is added to the tableau. The following example is illustrated the step of variant 2.

Example 2.3.5. Consider the following linear programming problem:

$$\begin{array}{lrrrrrrr} \max & & x_1 & + & 5x_2 & & \\ \text{subject to} & - & x_1 & + & x_2 & \leq & 1 \\ & & x_1 & - & x_2 & \geq & 1 \\ & & x_1 & - & x_2 & \geq & 2 \\ & & x_1 & - & x_2 & \leq & 4 \\ & & x_1, & & x_2 & \geq & 0. \end{array}$$

We first convert the problem to the standard form by adding slack variables s_1, s_4 and surplus variables s_2, s_3 obtaining:

$$\begin{array}{lrrrrrrrrrrrrr} \max & & x_1 & + & 5x_2 & & & & & & & & & \\ \text{subject to} & - & x_1 & + & x_2 & + & s_1 & & & & & & = & 1 \\ & & x_1 & - & x_2 & & & - & s_2 & & & & = & 1 \\ & & x_1 & - & x_2 & & & & & - & s_3 & & = & 2 \\ & & x_1 & - & x_2 & & & & & & & + \; s_4 & = & 4 \\ & & x_1, & & x_2, & & s_1, & & s_2, & & s_3, & & s_4 \geq & 0. \end{array}$$

Phase I:

After Arsham's algorithm ends, we get the following tableau.

	BVS	x_1	x_2	s_1	s_2	s_3	s_4	RHS
c_j	$-$	6	0	-5	0	0	0	$-$
	x_2	-1	1	1	0	0	0	1
	?	0	0	1	-1	0	0	2
	?	0	0	1	0	-1	0	3
	s_4	0	0	1	0	0	1	5

Then, the Arsham's algorithm reports infeasibility. At this point, the variants can be used to follow on, instead of declaring the infeasibility of the problem. Next, we choose the nonbasic variable to enter into the basic variable set by considering the constraints

associated with unoccupied rows which is the sum of all unoccupied rows denoted by SUM.

	BVS	x_1	x_2	s_1	s_2	s_3	s_4	RHS
c_j	–	6	0	−5	0	0	0	–
	x_2	−1	1	1	0	0	0	1
	?	0	0	1	−1	0	0	2
	?	0	0	1	0	−1	0	3
	s_4	0	0	1	0	0	1	5
SUM	?	0	0	2	−1	−1	0	5

In the added constraint, s_1 has the largest coefficient. Thus, s_1 is chosen to enter the basic variable set.

	BVS	x_1	x_2	s_1	s_2	s_3	s_4	RHS	C/R
c_j	–	6	0	−5	0	0	0	–	
	x_2	−1	1	1	0	0	0	1	1
	?	0	0	1	−1	0	0	2	2
	?	0	0	1	0	−1	0	3	3
	s_4	0	0	1	0	0	1	5	5
SUM	?	0	0	2	−1	−1	0	5	5/2

At the first row, there is the minimum C/R. So, x_2 is replaced by s_1 into the basic variable set in the row, we get the following tableau.

	BVS	x_1	x_2	s_1	s_2	s_3	s_4	RHS
c_j	–	1	5	0	0	0	0	–
	s_1	−1	1	1	0	0	0	1
	?	1	−1	0	−1	0	0	1
	?	1	−1	0	0	−1	0	2
	s_4	1	−1	0	0	0	1	4
SUM	?	2	−2	0	−1	−1	0	3

Next, the nonbasic variable x_1 is chosen to enter the basic variable set. After the tableau is updated, we get the following tableau.

	BVS	x_1	x_2	s_1	s_2	s_3	s_4	RHS
c_j	–	0	6	0	1	0	0	–
	s_1	0	0	1	−1	0	0	2
	x_1	1	−1	0	−1	0	0	1
	?	0	0	0	1	−1	0	1
	s_4	0	0	0	1	0	1	3
SUM	?	0	0	0	1	−1	0	1

Next, the nonbasic variable s_2 is chosen to enter the basic variable set. After the tableau is updated, we get the following tableau.

	BVS	x_1	x_2	s_1	s_2	s_3	s_4	RHS
c_j	–	0	6	0	0	1	0	–
	s_1	0	0	1	0	−1	0	3
	x_1	1	−1	0	0	−1	0	2
	s_2	0	0	0	1	−1	0	1
	s_4	0	0	0	0	1	1	2
SUM	?	0	0	0	0	0	0	0

Now, the added constraint becomes zero and the basic variable set is full. Thus, Phase 2 can start.

Phase 2:

Now, all c_j are not negative. Thus, the current tableau is not optimal. We apply the simplex method rules for entering and exiting variables to the basic variable set. But c_2 is the largest coefficient, x_2 is considered for being the entering variable which there is no the minimum ratio. Consequently, the problem is unbounded.

■

Since variant 2 of Gao's algorithm can identify the infeasibility of a linear programming problem by considering the sum of unoccupied rows from Arsham's algorithm, we will use the sum of unoccupied rows which is all constraints for starting the simplex method first. In the next chapter, we describe the proposed algorithm.

CHAPTER 3

THE PROPOSED ALGORITHM

In this chapter, we present the idea of constructing our algorithm which we use the sum of unoccupied rows for starting the simplex method without using artificial variables. First, the sum of unoccupied rows is described. Then, we separate linear programming problems into different types which are homogeneous and nonhomogeneous linear programming problems. The proposed algorithm for solving both is presented.

3.1 The sum of unoccupied rows

Consider the following linear programming model:

$$\begin{array}{lrcccccccc} \max & z & = & c_1x_1 & + & c_2x_2 & + & \cdots & + & c_nx_n \\ \text{subject to} & & & a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ & & & a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ & & & \vdots & & \vdots & & \vdots & & \vdots & & \\ & & & a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & = & b_m \\ & & & x_1, & & x_2, & & \ldots, & & x_n & \geq & 0, \end{array} \tag{3.1.1}$$

where n is the number of decision variables,

m is the number of constraints,

c_j is the coefficient of objective function for each $j = 1,...,n$,

a_{ij} is the coefficients of constraints for $i = 1,...,m, \quad j = 1,...,n$,

b_i is a nonnegative right-hand-side value for each $i = 1,...,m$,

and x_j is a nonnegative decision variable for each $j = 1,...,n$.

For the problem (3.1.1), we can use the two-phase method for finding a feasible solution by adding artificial variables for constructing the basic variable set when the basic feasible solution cannot be found easily. The two-phase method is separated into two phases. For Phase I, it finds the basic feasible variable set by minimizing of the summation of artificial variables. For Phase II, it uses the simplex method by starting at the basic variable set from Phase I for finding the optimal solution. The two-phase method can conclude as follows:

Phase I: Minimize the summation of artificial variables

$$
\begin{array}{lrcccccccccc}
\min & z_1 & = & x_{a_1} & + & x_{a_2} & + & \cdots & + & x_{a_m} & & \\
\text{subject to} & a_{11}x_1 & + & \cdots & + & a_{1n}x_n & + & x_{a_1} & & & = & b_1 \\
& a_{21}x_1 & + & \cdots & + & a_{2n}x_n & & + & x_{a_2} & & = & b_2 \\
& \vdots & & \vdots & & & & & \ddots & & \vdots & \\
& a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & & & & + \; x_{a_m} & = & b_m \\
& x_1, & & \ldots, & & x_n, & x_{a_1}, & & \ldots, & x_{a_m} & \geq & 0.
\end{array}
$$

Therefore, we have

$$
\mathbf{A}=\begin{bmatrix}
a_{11} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 \\
a_{21} & \cdots & a_{2n} & 0 & 1 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
a_{m1} & \cdots & a_{mn} & 0 & 0 & \cdots & 1
\end{bmatrix}, \quad \mathbf{b}=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad \mathbf{x}=\begin{bmatrix} x_1 \\ \vdots \\ x_n \\ x_{a_1} \\ \vdots \\ x_{a_m} \end{bmatrix} = \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_a \end{bmatrix}, \quad \mathbf{c}=\begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix},
$$

where $\mathbf{0} \in \mathbb{R}^n$ and $\mathbf{1} \in \mathbb{R}^m$.

If we choose $\mathbf{B} = \mathbf{I}$ to be a basis and $\mathbf{N} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$, then

$\mathbf{x_B} = \mathbf{x_a}$, $\mathbf{x_N} = \mathbf{x}$, $\mathbf{c}_B^T = \mathbf{1}^T$, $\mathbf{c}_N^T = \mathbf{0}^T$.

The initial tableau of any basis $\mathbf{B}$ can be computed as the following tableau:

	z_1	$\mathbf{x_B}$	$\mathbf{x_N}$	RHS
	1	$\mathbf{0}$	$\mathbf{c_B^T B^{-1} N - c_N^T}$	$\mathbf{c_B^T B^{-1} b}$
$\mathbf{x_B}$	0	$\mathbf{I}$	$\mathbf{B^{-1} N}$	$\mathbf{B^{-1} b}$

Then, the reduced cost can be computed as follows:

$$\mathbf{c_B^T B^{-1} N - c_N^T} = \mathbf{1^T I^{-1}} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} - \mathbf{0}^T$$

$$= \begin{bmatrix} \sum_{i=1}^{m} a_{i1} & \sum_{i=1}^{m} a_{i2} & \cdots & \sum_{i=1}^{m} a_{in} \end{bmatrix}$$

and the objective value is as follows:

$$\mathbf{c_B^T B^{-1} b} = \mathbf{1^T I^{-1}} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} = \sum_{i=1}^{m} b_i.$$

Thus, the initial tableau becomes

	z_1	x_1	x_2	$\cdots$	x_n	x_{a_1}	x_{a_2}	$\cdots$	x_{a_m}	RHS
z_1	1	$\sum_{i=1}^{m} a_{i1}$	$\sum_{i=1}^{m} a_{i2}$	$\cdots$	$\sum_{i=1}^{m} a_{in}$	0	0	$\cdots$	0	$\sum_{i=1}^{m} b_i$
x_{a_1}	0	a_{11}	a_{12}	$\cdots$	a_{1n}	1	0	$\cdots$	0	b_1
x_{a_2}	0	a_{21}	a_{22}	$\cdots$	a_{2n}	0	1	$\cdots$	0	b_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
x_{a_m}	0	a_{m1}	a_{m2}	$\cdots$	a_{mn}	0	0	$\cdots$	1	b_m

The two-phase method will choose nonbasic variables into the basis and try to leave artificial variables out the basis. Thus, we will not consider columns of all artificial variables and start the algorithm from an empty basic variable set. Then, nonbasic variables are selected to replace the rows which have artificial variables, and the artificial variables in that row are left from the basis. Therefore, we need not add artificial variables, and the size of the problem can be reduced.

Consider the row zero of the initial tableau, we get the equation

$$\left(\sum_{i=1}^{m} a_{i1}\right) x_1 + \left(\sum_{i=1}^{m} a_{i2}\right) x_2 + \cdots + \left(\sum_{i=1}^{m} a_{in}\right) x_n = \sum_{i=1}^{m} b_i \tag{3.1.2}$$

Thus, the equation (3.1.2) is called the **sum of all constraints**.

First, we start by setting the empty basic variable set, and the initial tableau of the proposed algorithm can reduce as the following tableau.

	BVS	x_1	x_2	$\cdots$	x_n	RHS
SUM	$-$	$\sum_{i=1}^{m} a_{i1}$	$\sum_{i=1}^{m} a_{i2}$	$\cdots$	$\sum_{i=1}^{m} a_{in}$	$\sum_{i=1}^{m} b_i$
	?	a_{11}	a_{12}	$\cdots$	a_{1n}	b_1
	?	a_{21}	a_{22}	$\cdots$	a_{2n}	b_2
	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
	?	a_{m1}	a_{m2}	$\cdots$	a_{mn}	b_m

Then, a nonbasic variable is chosen one by one into a row of the BVS column which is **basic variable set**. If a row in the BVS column has a basic variable, then it is called an **occupied row**. Otherwise, it is called an **unoccupied row**. Thus, the equation (3.1.2) can be called a **sum of unoccupied rows** which it can identify the existence of the solution of a linear system. Therefore, we use the equation (3.1.2) for constructing the basic feasible solution or identifying the solution.

Let $\alpha_j = \sum_{i=1}^{m} a_{ij}$ for each $j = 1,...,n$ and $\beta = \sum_{i=1}^{m} b_i$. Then, the equation (3.1.2) can be written below

$$\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = \beta. \tag{3.1.3}$$

By considering the value of β in (3.1.3), we can saperate the linear programming problem in two different types of the problem. If $\beta = 0$, then the problem (3.1.1) is a **homogeneous linear programming problem**. Otherwise, it is a **nonhomogeneous linear programming problem**. We describe the algorithm for solving both in details below.

3.2 Nonhomogeneous linear programming problems

A nonhomogeneous linear programming problem is a problem which has the nonzero right-hand-side vector which it can identify when $\beta > 0$. Its solution can be many possible situations: optimal solution, infeasible or unbounded. We will use the sum of unoccupied rows for finding its solution by considering the largest coefficient in the equation (3.1.3) defined as follows:

$$\alpha_{\max} = \max_{1 \le j \le n} \{\alpha_j\}.$$

Thus, two cases are distinguished to construct the algorithm below.

3.2.1 Case: $\alpha_{\max} \le 0$

Since $\alpha_{\max} \le 0$ and $\beta > 0$, $\alpha_j < 0$ for all $j = 1,...,n$ in (3.1.3) . However, the value of x_j is only positive or zero for all $j = 1,...,n$. Thus, there is no x_j which satisfies (3.1.3). We can conclude the solution as the following lemma.

Lemma 3.2.1. Consider the linear programming model (3.1.1). Let $\alpha_j = \sum_{i=1}^{m} a_{ij}$ for each $j = 1,...,n$, $\alpha_{\max} = \max_{1\le j\le n}\{\alpha_j\}$ and $\beta = \sum_{i=1}^{m} b_i$. If $\alpha_{\max} \le 0$ and $\beta > 0$, then the problem (3.1.1) is infeasible.

Proof. Suppose that $\alpha_{\max} \le 0$ and $\beta > 0$. Assume that the problem is feasible. Thus, there is a solution $x_j \ge 0$ for all $j = 1,...,n$ which $\alpha_1 x_1 + \cdots + \alpha_n x_n = \beta$. Since $\alpha_{\max} \le 0$, $\alpha_j \le 0$ for all $j = 1,...,n$. Since $\alpha_j \le 0$ and $x_j \ge 0$ for all $j = 1,...,n$, $\alpha_j x_j \le 0$ for all $j = 1,...,n$. So, $\alpha_1 x_1 + \cdots + \alpha_n x_n \le 0 \ne \beta > 0$. This is a contradiction. Therefore, the problem is no solution or infeasible. □

3.2.2 Case: $\alpha_{\max} > 0$

In this case, we can choose a nonbasic variable x_k which associate to $\alpha_{\max}$ to enter into the basic variable set since there is at least one coefficient in column k as $a_{ik} > 0$ for some $i = 1,...,m$. Thus, there exists the i^{th} row for calculating the minimum ratio, and the variable x_k is chosen to enter the basic variable set. After the basic variable set is not empty, we would like to use the sum of unoccupied rows to identify the solution or choose the remaining nonbasic variables to enter the basis.

Next, we will use the sum of unoccupied rows to consider the existence of the solution.

Consider the following model:

$$\begin{array}{lllll} \text{LP:} & \max & \mathbf{c}^T\mathbf{x} & & \\ & \text{s.t.} & \mathbf{A}\mathbf{x} & = & \mathbf{b}, \\ & & \mathbf{x} & \ge & \mathbf{0}. \end{array} \tag{3.2.1}$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{c}, \mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{b} \ge \mathbf{0}$ and rank($\mathbf{A}$) = m.

For constructing the basic feasible variable set of the problem (3.2.1), we will use the equation $\boldsymbol{\alpha}^T\mathbf{x} = \beta$ where $\boldsymbol{\alpha}$ is a column vector of coefficient of the sum of

all constraints and β is the sum of all right-hand-side values. Then, the initial tableau of the proposed algorithm can be presented as follows:

	BVS	**x**	RHS
SUM	–	α	β
	?	**A**	**b**

After some nonbasic variables enter the basic variable set, we let $I_{\mathbf{B}}$ be an index set of basic variables in occupied rows which $|I_{\mathbf{B}}| = l, 1 \leq l \leq m$, and $I_{\mathbf{N}}$ be an index set of nonbasic variables which $|I_{\mathbf{N}}| = k = n - l$. Assume that $\mathbf{x_B}$ is the basic variable vector in the occupied rows where $\mathbf{x_B} = [x_{\mathbf{B}_i}],\ i \in I_{\mathbf{B}}$ and $\mathbf{x_N}$ is the nonbasic variable vector where $\mathbf{x_N} = [x_{\mathbf{N}_i}],\ i \in I_{\mathbf{N}}$.

Assume that

$$\mathbf{A} = \left[\begin{array}{c|c} \mathbf{B} & \mathbf{N} \\ \hline \mathbf{A}_{u_{\mathbf{B}}} & \mathbf{A}_{u_{\mathbf{N}}} \end{array}\right],\ \boldsymbol{\alpha}^T = \left[\boldsymbol{\alpha}_{\mathbf{B}}^T \quad \boldsymbol{\alpha}_{\mathbf{N}}^T\right],\ \text{and}\ \mathbf{b} = \begin{bmatrix} \mathbf{b}_o \\ \mathbf{b}_u \end{bmatrix}$$

where $\mathbf{B} \in \mathbb{R}^{l \times l}$ is a basic invertible matrix which each column coressponds to the index set $I_{\mathbf{B}}$ and each row corresponds to occupied rows,

$\mathbf{N} \in \mathbb{R}^{l \times (n-l)}$ is a nonbasic matrix which each column coressponds to the index set $I_{\mathbf{N}}$ and each row corresponds to occupied rows,

$\mathbf{A}_{u_{\mathbf{B}}} \in \mathbb{R}^{(m-l) \times l}$ is a submatrix of $\mathbf{A}$ which each row associates with unoccupied rows and each column associates with $I_{\mathbf{B}}$,

$\mathbf{A}_{u_{\mathbf{N}}} \in \mathbb{R}^{(m-l) \times (n-l)}$ is a submatrix of $\mathbf{A}$ which each row associates with unoccupied rows and each column associates with $I_{\mathbf{N}}$,

$\boldsymbol{\alpha}_{\mathbf{B}}^T$ is a basic row vector of coefficients of the sum of all constaints associated with $I_{\mathbf{B}}$,

$\boldsymbol{\alpha}_{\mathbf{N}}^T$ is a nonbasic row vector of coefficients of the sum of all constaints associated with $I_{\mathbf{N}}$,

$\mathbf{c}_{\mathbf{B}}^T$ is a row vector of $\mathbf{c}^T$ associated with $I_{\mathbf{B}}$,

$\mathbf{c}_{\mathbf{N}}^T$ is a row vector of $\mathbf{c}^T$ associated with $I_{\mathbf{N}}$,

$\mathbf{b}_o$ is a column vector of right-hand-side values in occupied rows,

and $\mathbf{b}_u$ is a column vector of right-hand-side values in unoccupied rows.

So, the problem (3.2.1) can be written as follows:

$$\begin{array}{lrcrcl} \text{LP: max} & \mathbf{c}_{\mathbf{B}}^T\mathbf{x}_{\mathbf{B}} & + & \mathbf{c}_{\mathbf{N}}^T\mathbf{x}_{\mathbf{N}} & & \\ \text{s.t.} & \mathbf{B}\mathbf{x}_{\mathbf{B}} & + & \mathbf{N}\mathbf{x}_{\mathbf{N}} & = & \mathbf{b}_o \\ & \mathbf{A}_{u_{\mathbf{B}}}\mathbf{x}_{\mathbf{B}} & + & \mathbf{A}_{u_{\mathbf{N}}}\mathbf{x}_{\mathbf{N}} & = & \mathbf{b}_u \\ & \mathbf{x}_{\mathbf{B}}, & & \mathbf{x}_{\mathbf{N}} & \geq & \mathbf{0}. \end{array} \tag{3.2.2}$$

So, the tableau of the proposed algorithm when $\mathbf{x}_{\mathbf{B}}$ is chosen to enter the basic variable set is as follows:

	BVS	$\mathbf{x}_{\mathbf{B}}$	$\mathbf{x}_{\mathbf{N}}$	RHS
SUM	$-$	$\alpha_{\mathbf{B}}^T$	$\alpha_{\mathbf{N}}^T$	β
	$\mathbf{x}_{\mathbf{B}}$	$\mathbf{B}$	$\mathbf{N}$	$\mathbf{b}_o$
	?	$\mathbf{A}_{u_{\mathbf{B}}}$	$\mathbf{A}_{u_{\mathbf{N}}}$	$\mathbf{b}_u$

Consider the constraints which is occupied rows:

$$\mathbf{B}\mathbf{x}_{\mathbf{B}} + \mathbf{N}\mathbf{x}_{\mathbf{N}} = \mathbf{b}_o. \tag{3.2.3}$$

Multiply the equation (3.2.3) by $\mathbf{B}^{-1}$, we get

$$\mathbf{x}_{\mathbf{B}} + \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_{\mathbf{N}} = \mathbf{B}^{-1}\mathbf{b}_o \text{ or } \mathbf{x}_{\mathbf{B}} = \mathbf{B}^{-1}\mathbf{b}_o - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_{\mathbf{N}}. \tag{3.2.4}$$

Consider the constraints which is unoccupied rows:

$$\begin{aligned} \mathbf{A}_{u_{\mathbf{B}}}\mathbf{x}_{\mathbf{B}} + \mathbf{A}_{u_{\mathbf{N}}}\mathbf{x}_{\mathbf{N}} &= \mathbf{b}_u \\ \mathbf{A}_{u_{\mathbf{B}}}(\mathbf{B}^{-1}\mathbf{b}_o - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_{\mathbf{N}}) + \mathbf{A}_{u_{\mathbf{N}}}\mathbf{x}_{\mathbf{N}} &= \mathbf{b}_u \\ \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{b}_o - \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{N}\mathbf{x}_{\mathbf{N}} + \mathbf{A}_{u_{\mathbf{N}}}\mathbf{x}_{\mathbf{N}} &= \mathbf{b}_u \\ \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{b}_o - (\mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{N} - \mathbf{A}_{u_{\mathbf{N}}})\mathbf{x}_{\mathbf{N}} &= \mathbf{b}_u \\ (\mathbf{A}_{u_{\mathbf{N}}} - \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{N})\mathbf{x}_{\mathbf{N}} &= \mathbf{b}_u - \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{b}_o. \end{aligned}$$

Consider the equation $\boldsymbol{\alpha}^T\mathbf{x} = \beta$, we get

$$\begin{aligned}
\boldsymbol{\alpha}_{\mathbf{B}}^T\mathbf{x}_{\mathbf{B}} + \boldsymbol{\alpha}_{\mathbf{N}}^T\mathbf{x}_{\mathbf{N}} &= \beta \\
\boldsymbol{\alpha}_{\mathbf{B}}^T(\mathbf{B}^{-1}\mathbf{b}_o - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_{\mathbf{N}}) + \boldsymbol{\alpha}_{\mathbf{N}}^T\mathbf{x}_{\mathbf{N}} &= \beta \\
\boldsymbol{\alpha}_{\mathbf{B}}^T\mathbf{B}^{-1}\mathbf{b}_o - \boldsymbol{\alpha}_{\mathbf{B}}^T\mathbf{B}^{-1}\mathbf{N}\mathbf{x}_{\mathbf{N}} + \boldsymbol{\alpha}_{\mathbf{N}}^T\mathbf{x}_{\mathbf{N}} &= \beta \\
(\boldsymbol{\alpha}_{\mathbf{N}}^T - \boldsymbol{\alpha}_{\mathbf{B}}^T\mathbf{B}^{-1}\mathbf{N})\mathbf{x}_{\mathbf{N}} &= \beta - \alpha_{\mathbf{B}}^T\mathbf{B}^{-1}\mathbf{b}_o.
\end{aligned}$$

So, the problem (3.2.2) can be written as follows:

$$\begin{aligned}
\text{LP:} \quad \text{max} \quad & \mathbf{c}_{\mathbf{B}}^T\mathbf{x}_{\mathbf{B}} + \mathbf{c}_{\mathbf{N}}^T\mathbf{x}_{\mathbf{N}} \\
\text{s.t.} \quad & \mathbf{I}_l\mathbf{x}_{\mathbf{B}} + \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_{\mathbf{N}} = \mathbf{B}^{-1}\mathbf{b}_o \\
& (\mathbf{A}_{u_{\mathbf{N}}} - \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{N})\mathbf{x}_{\mathbf{N}} = \mathbf{b}_u - \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{b}_o \\
& \mathbf{x}_{\mathbf{B}}, \quad \mathbf{x}_{\mathbf{N}} \geq \mathbf{0}.
\end{aligned} \tag{3.2.5}$$

Then, the updated tableau of our algorithm is as follows:

	BVS	$\mathbf{x}_{\mathbf{B}}$	$\mathbf{x}_{\mathbf{N}}$	RHS
SUM	–	$\mathbf{0}$	$\boldsymbol{\alpha}_{\mathbf{N}}^T - \boldsymbol{\alpha}_{\mathbf{B}}^T\mathbf{B}^{-1}\mathbf{N}$	$\beta - \boldsymbol{\alpha}_{\mathbf{B}}^T\mathbf{B}^{-1}\mathbf{b}_o$
	$\mathbf{x}_{\mathbf{B}}$	$\mathbf{I}_l$	$\mathbf{B}^{-1}\mathbf{N}$	$\mathbf{B}^{-1}\mathbf{b}_o$
	?	$\mathbf{0}$	$\mathbf{A}_{u_{\mathbf{N}}} - \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{N}$	$\mathbf{b}_u - \mathbf{A}_{u_{\mathbf{B}}}\mathbf{B}^{-1}\mathbf{b}_o$

where $\mathbf{I}_l$ is an identity $l \times l$ matrix.

By the above tableau, we wolud like to show that $\boldsymbol{\alpha}_{\mathbf{N}}^T - \boldsymbol{\alpha}_{\mathbf{B}}^T\mathbf{B}^{-1}\mathbf{N}$ is the sum of unoccupied rows.

Let

$$\mathbf{B} = \begin{bmatrix} a_{1\mathbf{B}_1} & a_{1\mathbf{B}_2} & \cdots & a_{1\mathbf{B}_l} \\ a_{2\mathbf{B}_1} & a_{2\mathbf{B}_2} & \cdots & a_{2\mathbf{B}_l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{l\mathbf{B}_1} & a_{l\mathbf{B}_2} & \cdots & a_{l\mathbf{B}_l} \end{bmatrix}, \qquad \mathbf{N} = \begin{bmatrix} a_{1\mathbf{N}_1} & a_{1\mathbf{N}_2} & \cdots & a_{1\mathbf{N}_k} \\ a_{2\mathbf{N}_1} & a_{2\mathbf{N}_2} & \cdots & a_{2\mathbf{N}_k} \\ \vdots & \vdots & & \vdots \\ a_{l\mathbf{N}_1} & a_{l\mathbf{N}_2} & \cdots & a_{l\mathbf{N}_k} \end{bmatrix}$$

$$\mathbf{A}_{u_\mathbf{B}} = \begin{bmatrix} a_{l+1\mathbf{B}_1} & a_{l+1\mathbf{B}_2} & \cdots & a_{l+1\mathbf{B}_l} \\ a_{l+2\mathbf{B}_1} & a_{l+2\mathbf{B}_2} & \cdots & a_{l+2\mathbf{B}_l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m\mathbf{B}_1} & a_{m\mathbf{B}_2} & \cdots & a_{m\mathbf{B}_l} \end{bmatrix}, \quad \text{and} \quad \mathbf{A}_{u_\mathbf{N}} = \begin{bmatrix} a_{l+1\mathbf{N}_1} & a_{l+1\mathbf{N}_2} & \cdots & a_{l+1\mathbf{N}_k} \\ a_{l+2\mathbf{N}_1} & a_{l+2\mathbf{N}_2} & \cdots & a_{l+2\mathbf{N}_k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m\mathbf{N}_1} & a_{m\mathbf{N}_2} & \cdots & a_{m\mathbf{N}_k} \end{bmatrix}$$

where $\mathbf{N}_j \in I_\mathbf{N}$ for $j = 1, ..., k$ and $\mathbf{B}_i \in I_\mathbf{B}$ for $i = 1, ..., l$.

Consider $\boldsymbol{\alpha}_\mathbf{N}^T - \boldsymbol{\alpha}_\mathbf{B}^T \mathbf{B}^{-1}\mathbf{N}$

$$= \begin{bmatrix} \alpha_{\mathbf{N}_1} & \alpha_{\mathbf{N}_2} & \cdots & \alpha_{\mathbf{N}_k} \end{bmatrix} - \begin{bmatrix} \alpha_{\mathbf{B}_1} & \alpha_{\mathbf{B}_2} & \cdots & \alpha_{\mathbf{B}_l} \end{bmatrix} \mathbf{B}^{-1}\mathbf{N}$$

$$= \begin{bmatrix} \sum_{i=1}^{m} a_{i\mathbf{N}_1} & \sum_{i=1}^{m} a_{i\mathbf{N}_2} & \cdots & \sum_{i=1}^{m} a_{i\mathbf{N}_k} \end{bmatrix} - \begin{bmatrix} \sum_{i=1}^{m} a_{i\mathbf{B}_1} & \sum_{i=1}^{m} a_{i\mathbf{B}_2} & \cdots & \sum_{i=1}^{m} a_{i\mathbf{B}_l} \end{bmatrix} \mathbf{B}^{-1}\mathbf{N}$$

$$= \mathbf{1}^T \begin{bmatrix} a_{1\mathbf{N}_1} & a_{1\mathbf{N}_2} & \cdots & a_{1\mathbf{N}_k} \\ \vdots & \vdots & & \vdots \\ a_{l\mathbf{N}_1} & a_{l\mathbf{N}_2} & \cdots & a_{l\mathbf{N}_k} \\ a_{l+1\mathbf{N}_1} & a_{l+1\mathbf{N}_2} & \cdots & a_{l+1\mathbf{N}_k} \\ \vdots & \vdots & & \vdots \\ a_{m\mathbf{N}_1} & a_{m\mathbf{N}_2} & \cdots & a_{m\mathbf{N}_k} \end{bmatrix} - \mathbf{1}^T \begin{bmatrix} a_{1\mathbf{B}_1} & a_{1\mathbf{B}_2} & \cdots & a_{1\mathbf{B}_l} \\ \vdots & \vdots & & \vdots \\ a_{l\mathbf{B}_1} & a_{l\mathbf{B}_2} & \cdots & a_{l\mathbf{B}_l} \\ a_{l+1\mathbf{B}_1} & a_{l+1\mathbf{B}_2} & \cdots & a_{l+1\mathbf{B}_l} \\ \vdots & \vdots & & \vdots \\ a_{m\mathbf{B}_1} & a_{m\mathbf{B}_2} & \cdots & a_{m\mathbf{B}_l} \end{bmatrix} \mathbf{B}^{-1}\mathbf{N}$$

$$= \mathbf{1}^T \left(\begin{bmatrix} \mathbf{N} \\ \mathbf{A}_{u_\mathbf{N}} \end{bmatrix} - \begin{bmatrix} \mathbf{B} \\ \mathbf{A}_{u_\mathbf{B}} \end{bmatrix} \mathbf{B}^{-1}\mathbf{N} \right)$$

$$= \mathbf{1}^T \left(\begin{bmatrix} \mathbf{N} \\ \mathbf{A}_{u_\mathbf{N}} \end{bmatrix} - \begin{bmatrix} \mathbf{N} \\ \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{N} \end{bmatrix} \right)$$

$$= \mathbf{1}^T \begin{bmatrix} \mathbf{0} \\ \mathbf{A}_{u_\mathbf{N}} - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{N} \end{bmatrix}.$$

$$= \mathbf{1}^T \left[\mathbf{A}_{u_\mathbf{N}} - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{N}\right], \text{ where } \mathbf{1} \in \mathbb{R}^m.$$

Therefore, $\boldsymbol{\alpha}_\mathbf{N}^T - \boldsymbol{\alpha}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{N}$ is the sum of all rows of $\mathbf{A}_{u_\mathbf{N}} - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{N}$, that is, $\boldsymbol{\alpha}_\mathbf{N}^T - \boldsymbol{\alpha}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{N}$ is the sum of unoccupied rows.

Next, we would like to show that the sum of the right-hand-side values of unoccupied rows is $\beta - \boldsymbol{\alpha}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{b}_o$.

Let

$$\mathbf{b}_o = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_l \end{bmatrix} \quad \text{and} \quad \mathbf{b}_u = \begin{bmatrix} b_{l+1} \\ b_{l+2} \\ \vdots \\ b_m \end{bmatrix}$$

Consider

$$\beta - \boldsymbol{\alpha}_\mathbf{B}^T\mathbf{B}^{-1}\mathbf{b}_o = \sum_{i=1}^{m} b_i - \left[\sum_{i=1}^{m} a_{i\mathbf{B}_1} \quad \sum_{i=1}^{m} a_{i\mathbf{B}_2} \quad \cdots \quad \sum_{i=1}^{m} a_{i\mathbf{B}_l}\right] \mathbf{B}^{-1}\mathbf{b}_o$$

$$= \mathbf{1}^T \begin{bmatrix} b_1 \\ \vdots \\ b_l \\ b_{l+1} \\ \vdots \\ b_m \end{bmatrix} - \mathbf{1}^T \begin{bmatrix} a_{1\mathbf{B}_1} & a_{1\mathbf{B}_2} & \cdots & a_{1\mathbf{B}_l} \\ \vdots & \vdots & & \vdots \\ a_{l\mathbf{B}_1} & a_{l\mathbf{B}_2} & \cdots & a_{l\mathbf{B}_l} \\ a_{l+1\mathbf{B}_1} & a_{l+1\mathbf{B}_2} & \cdots & a_{l+1\mathbf{B}_l} \\ \vdots & \vdots & & \vdots \\ a_{m\mathbf{B}_1} & a_{m\mathbf{B}_2} & \cdots & a_{m\mathbf{B}_l} \end{bmatrix} \mathbf{B}^{-1}\mathbf{b}_o$$

$$
\begin{aligned}
&= \mathbf{1}^T \begin{bmatrix} \mathbf{b}_o \\ \mathbf{b}_u \end{bmatrix} - \mathbf{1}^T \begin{bmatrix} \mathbf{B} \\ \mathbf{A}_{u_\mathbf{B}} \end{bmatrix} \mathbf{B}^{-1}\mathbf{b}_o \\
&= \mathbf{1}^T \left(\begin{bmatrix} \mathbf{b}_o \\ \mathbf{b}_u \end{bmatrix} - \begin{bmatrix} \mathbf{b}_o \\ \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{b}_o \end{bmatrix} \right) \\
&= \mathbf{1}^T \begin{bmatrix} \mathbf{0} \\ \mathbf{b}_u - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{b}_o \end{bmatrix} \\
&= \mathbf{1}^T \left[\mathbf{b}_u - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{b}_o \right].
\end{aligned}
$$

Therefore, $\beta - \boldsymbol{\alpha}_\mathbf{B}^T \mathbf{B}^{-1}\mathbf{b}_o$ is the summation of right-hand-side values of unoccupied rows.

Lemma 3.2.2. Consider the problem (3.2.1). Let $I_\mathbf{B}$ be an index set of basic variables in occupied rows which $|I_\mathbf{B}| = l, 1 \leq l \leq m$, and $I_\mathbf{N}$ be an index set of nonbasic variables which $|I_\mathbf{N}| = k = n - l$, $\mathbf{x_B} = [\mathbf{x}_{\mathbf{B}_i}],\ i \in I_\mathbf{B}$ and $\mathbf{x_N} = [\mathbf{x}_{\mathbf{N}_i}],\ i \in I_\mathbf{N}$. Let

$$
\mathbf{A} = \left[\begin{array}{c|c} \mathbf{B} & \mathbf{N} \\ \hline \mathbf{A}_{u_\mathbf{B}} & \mathbf{A}_{u_\mathbf{N}} \end{array}\right] \text{ and } \mathbf{b} = \begin{bmatrix} \mathbf{b}_o \\ \mathbf{b}_u \end{bmatrix},
$$

where $\mathbf{B} \in \mathbb{R}^{l \times l}$ is a basic invertible matrix which each column coressponds to the index set $I_\mathbf{B}$ and each row corresponds to occupied rows,

$\mathbf{N} \in \mathbb{R}^{l \times (n-l)}$ is a nonbasic matrix which each column coressponds to the index set $I_\mathbf{N}$ and each row corresponds to occupied rows,

$\mathbf{A}_{u_\mathbf{B}} \in \mathbb{R}^{(m-l) \times l}$ is a submatrix of $\mathbf{A}$ which each row associates with unoccupied rows and each column associates with $I_\mathbf{B}$,

$\mathbf{A}_{u_\mathbf{N}} \in \mathbb{R}^{(m-l)\times(n-l)}$ is a submatrix of $\mathbf{A}$ which each row associates with unoccupied rows and each column associates with $I_\mathbf{N}$,

$\mathbf{b}_o$ is a column vector of right-hand-side values in occupied rows,

and $\mathbf{b}_u$ is a column vector of right-hand-side values in unoccupied rows.

Consider the following system:

$$\begin{aligned}(\mathbf{A}_{u_\mathbf{N}} - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{N})\mathbf{x}_\mathbf{N} &= \mathbf{b}_u - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{b}_o \\ \mathbf{x}_\mathbf{N} &\geq \mathbf{0}.\end{aligned} \tag{3.2.6}$$

If the system (3.2.6) is infeasible, then the problem (3.2.1) is infeasible.

Proof. Suppose that the problem LP is feasible. Thus, there exists $\mathbf{x} \geq \mathbf{0}$ and $\mathbf{A}\mathbf{x} = \mathbf{b}$. By rearranging the index of variables $\mathbf{x}$ as in $\mathbf{x} = \begin{bmatrix}\mathbf{x}_\mathbf{B} \\ \mathbf{x}_\mathbf{N}\end{bmatrix}$, we get

$$\begin{aligned}\mathbf{A}\mathbf{x} &= \mathbf{b} \\ \left[\begin{array}{c|c}\mathbf{B} & \mathbf{N} \\ \hline \mathbf{A}_{u_\mathbf{B}} & \mathbf{A}_{u_\mathbf{N}}\end{array}\right]\begin{bmatrix}\mathbf{x}_\mathbf{B} \\ \mathbf{x}_\mathbf{N}\end{bmatrix} &= \begin{bmatrix}\mathbf{b}_o \\ \mathbf{b}_u\end{bmatrix}\end{aligned}$$

Then,

$$\mathbf{B}\mathbf{x}_\mathbf{B} + \mathbf{N}\mathbf{x}_\mathbf{N} = \mathbf{b}_o \quad \text{and} \quad \mathbf{A}_{u_\mathbf{B}}\mathbf{x}_\mathbf{B} + \mathbf{A}_{u_\mathbf{N}}\mathbf{x}_\mathbf{N} = \mathbf{b}_u.$$

Since $\mathbf{x}_\mathbf{B} = \mathbf{B}^{-1}\mathbf{b}_o - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_\mathbf{N}$, $(\mathbf{A}_{u_\mathbf{N}} - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{N})\mathbf{x}_\mathbf{N} = \mathbf{b}_u - \mathbf{A}_{u_\mathbf{B}}\mathbf{B}^{-1}\mathbf{b}_o$.
So, there exists $\mathbf{x}_\mathbf{N}$ which satisfies the system (3.2.6). Thus, the system (3.2.6) is feasible. Therefore, if the system (3.2.6) is infeasible, then the problem LP is also infeasible. □

By the above lemma, if we found that the largest coefficient of sum of unoccupied rows is negative, then we can conclude that the problem with unoccupied rows is infeasible. Therefore, the original problem is infeasible. In other word, if the

basic variable set is not full, then we can conclude that the problem is infeasible as the next lemma.

Lemma 3.2.3. If a basic variable set is not full when the algorithm ends, then the linear programming problem is infeasible.

Proof. Suppose that the basic variable set is not full. Thus, the largest coefficient of sum of unoccupied rows is negative. So, the problem with unoccupied rows is infeasible. By Lemma 3.2.2, we can conclude that the solution of the original problem is infeasible. □

Next, we design our algorithm for solving a nonhomogeneous linear programming problem.

Steps of Nonhomogeous Linear Programming Algorithm

Phase I: find the basic variable set

Step 1 Let $I_{\mathbf{B}} = \varnothing$ and $I_{\mathbf{N}} = \{1, 2, ..., n\}$. Construct an initial tableau and add the following sum of unoccupied rows,

$$\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = \beta$$

in the SUM row.

Step 2 If $\beta > 0$, then let $k = \underset{j \in I_{\mathbf{N}}}{\operatorname{argmax}}\{\alpha_j\}$, where $I_{\mathbf{N}}$ is an index set of non-basic variables.

- If $\alpha_k > 0$, then x_k is chosen into the basic variable set and go to Step 2.1.
- Otherwise, the problem is infeasible and stop.

Step 2.1 Compute the minimum ratio by

$$\frac{\overline{b}_r}{y_{rk}} = \min_{1 \le i \le m} \left\{ \frac{\overline{b}_i}{y_{ik}} : y_{ik} > 0 \right\}.$$

- If the r^{th} row is occupied by $x_{\mathbf{B}_r}$, then $I_{\mathbf{B}} = I_{\mathbf{B}} - \{\mathbf{B}_r\}$ and $I_{\mathbf{N}} = I_{\mathbf{N}} \cup \{\mathbf{B}_r\} - \{k\}$.

- Otherwise, $I_{\mathbf{B}} = I_{\mathbf{B}} \cup \{k\}$ and $I_{\mathbf{N}} = I_{\mathbf{N}} - \{k\}$.

Step 2.2 Bring x_k into the r^{th} row, by using Gauss-Jordan row operations.

Step 2.3 - If $|I_{\mathbf{B}}| = m$, then phase II starts.

- Otherwise, go back to Step 2.

else ($\beta = 0$)

For $i \in U$, where U is an index set of unoccupied rows.

Let $k = \underset{j \in I_{\mathbf{N}}}{\operatorname{argmax}}\{|a_{ij}|\}$, where $I_{\mathbf{N}}$ is a set of index of nonbasic variables.

- If $a_{ik} \neq 0$, then bring x_k into the i^{th} row, by using Gauss-Jordan row operations.

- Otherwise, the i^{th} row is deleted.

Go to Phase II.

Phase II: Solving the original problem by the simplex algorithm

We get the basic feasible solution from Phase I. Then, the the simplex algorithm can start for finding the optimal solution.

Next, we give two illustrative examples for showing the efficiency of our algorithm.

Example 3.2.1. Consider the following linear programming problem:

$$
\begin{array}{lrcrcrcrcr}
\max & & x_1 & + & 5x_2 & + & 2x_3 & + & 4x_3 & \\
\text{subject to} & - & 3x_1 & + & 5x_2 & - & 4x_3 & - & 3x_4 & \geq & 2 \\
 & & x_1 & + & 4x_2 & + & 2x_3 & - & x_4 & \leq & -3 \\
 & - & 2x_1 & + & x_2 & - & 3x_3 & - & 2x_4 & \geq & 4 \\
 & & 3x_1 & - & 6x_2 & + & x_3 & + & x_4 & = & 6 \\
 & & x_1, & & x_2, & & x_3, & & x_4 & \geq & 0.
\end{array}
$$

We first convert the problem to the standard form. Then, we obtain as follows:

$$\begin{array}{lrcrcrcrcrcrcrcl}
\text{max} & & x_1 & + & 5x_2 & + & 2x_3 & + & 4x_3 & & & & & & \\
\text{subject to} & - & 3x_1 & + & 5x_2 & - & 4x_3 & - & 3x_4 & - & s_1 & & & = & 2 \\
& - & x_1 & - & 4x_2 & - & 2x_3 & + & x_4 & & & - & s_2 & = & 3 \\
& - & 2x_1 & + & x_2 & - & 3x_3 & - & 2x_4 & & & & & - \; s_3 & = & 4 \\
& & 3x_1 & - & 6x_2 & + & x_3 & + & x_4 & & & & & = & 6 \\
& & x_1, & & x_2, & & x_3, & & x_4 & & s_1, & & s_2, & s_3 \; \geq & 0.
\end{array}$$

Phase I: The initial tableau of the problem is as below:

	BVS	x_1	x_2	x_3	x_4	s_1	s_2	s_3	RHS
SUM	$-$	-3	-4	-8	-3	-1	-1	-1	15
	?	-3	5	-4	-3	-1	0	0	2
	?	-1	-4	-2	1	0	-1	0	3
	?	-2	1	-3	-2	0	0	-1	4
	?	3	-6	1	1	0	0	0	6

By considering the above tableau, the sum of unoccupied rows is

$$-3x_1 - 4x_2 - 8x_3 - 3x_4 - s_1 - s_2 - s_3 = 15 \tag{3.2.7}$$

We have $\alpha_1 = -3$, $\alpha_2 = -4$, $\alpha_3 = -8$, $\alpha_4 = -3$, $\alpha_5 = \alpha_6 = \alpha_7 = -1$ and $\beta = 15$. Since $\alpha_{\max} = \alpha_5 = -1 \leq 0$ and $\beta = 15 > 0$, the problem is infeasible.

Example 3.2.2. (A counterexample of Arsham's algorithm by Enge and Huhn [4]).
Consider the following linear programming problem:

$$\begin{array}{lrcrcrcl}
\text{max} & 3x_1 & + & x_2 & - & 4x_3 & & \\
\text{subject to} & x_1 & + & x_2 & - & x_3 & = & 1 \\
& & & x_2 & & & \geq & 2 \\
& x_1, & & x_2, & & x_3 & \geq & 0.
\end{array}$$

We first convert the problem to the standard form by substracting surplus variable s_1 obtained as follows:

$$\begin{array}{lrcrcrcrcl} \text{max} & 3x_1 & + & x_2 & - & 4x_3 & & & & \\ \text{subject to} & x_1 & + & x_2 & - & x_3 & & & = & 1 \\ & & & x_2 & & & - & s_1 & = & 2 \\ & x_1, & & x_2, & & x_3, & & s_1 & \geq & 0. \end{array}$$

Phase I

The initial tableau of the problem is as below:

	BVS	x_1	x_2	x_3	s_1	RHS
SUM	–	1	2	−1	−1	3
	?	1	1	−1	0	1
	?	0	1	0	−1	2

By considering the above tableau, the sum of unoccupied rows is

$$x_1 + 2x_2 - x_3 - s_1 = 3. \tag{3.2.8}$$

We have $\alpha_1 = 1$, $\alpha_2 = 2$, $\alpha_3 = -1$, $\alpha_4 = -1$, and $\beta = 3$. Then, $\alpha_{\max} = \alpha_2 = 2 \geq 0$. Thus, x_2 is chosen into the basic variable set. Compute the minimum ratio for choosing the leaving variable, we get the following tableau.

	BVS	x_1	x_2	x_3	s_1	RHS	Ratio
SUM	–	1	2	−1	−1	3	
	?	1	1	−1	0	1	1/1
	?	0	1	0	−1	2	2/1

Next, the nonbasic variable x_2 enters the basic variable set in the first row. Perform Gauss-Jordan row operations, we get the following tableau:

	BVS	x_1	x_2	x_3	s_1	RHS
SUM	–	-1	0	1	-1	1
	x_2	1	1	-1	0	1
	?	-1	0	1	-1	1

Now, the sum of unoccupied rows is updated below

$$-x_1 + x_3 - s_1 = 1. \tag{3.2.9}$$

We have $\alpha_1 = -1$, $\alpha_3 = 1$, $\alpha_4 = -1$, and $\beta = 1$. Then, $\alpha_{\max} = \alpha_3 = 1 \geq 0$. Thus, x_3 is chosen into the basic variable set. The current tableau is updated as follows:

	BVS	x_1	x_2	x_3	s_1	RHS
SUM	–	0	0	0	0	0
	x_2	0	1	0	-1	2
	x_3	-1	0	1	-1	1

Since the number of elements of the basic variable set is equal to the number of all constraints in the original problem, then phase II starts.

Phase II

The initial tableau of Phase II is as follows:

	x_1	x_2	x_3	s_1	RHS
$z_j - c_j$	1	0	0	3	-2
x_2	0	1	0	-1	2
x_3	-1	0	1	-1	1

Since all reduced costs of the current tableau are positive, this is the optimal tableau. The basic feasible variable set is $\{x_2, x_3\}$ and the optimal solution is $x_1^* = 0$, $x_2^* = 2$, $x_3^* = 1$, and $s_1^* = 0$.

From Example 3.2.2, we found that we use 2 iterations while Gao's algorithm uses 3 iterations to solve it.

Afterwards, we will use the sum of unoccupied rows for finding the solution of the homogeneous linear programming problem.

3.3 Homogeneous linear programming problems

Consider the following linear programming model:

$$\begin{array}{lrcccccccc} \max & z & = & c_1x_1 & + & c_2x_2 & + & \cdots & + & c_nx_n \\ \text{subject to} & & & a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & 0 \\ & & & a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & 0 \\ & & & \vdots & & \vdots & & \vdots & & \vdots & & \\ & & & a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & = & 0 \\ & & & x_1, & & x_2, & & \ldots, & & x_n & \geq & 0 \end{array} \tag{3.3.1}$$

is called the **homogeneous linear programming problem** which it has all elements of the right-hand-side being zero ($\beta = 0$). Its solution are only three possible solutions: trivial solution, multiple solutions or unboundedness. If we solve it by the simplex method, then both artificial variables are added and the degeneracy appears. So, we will use the sum of unoccupied rows for finding or identifying the solution of (3.3.1).

Consider the sum of unoccupied rows for the homogeneous linear programming problem

$$\alpha_1x_1 + \alpha_2x_2 + \cdots + \alpha_nx_n = 0, \tag{3.3.2}$$

where $\alpha_j = \sum_{i=1}^{m} a_{ij}$ for each $j = 1, ..., n$.

Let

$$\alpha_{\max} = \max_{1 \leq j \leq n} \{\alpha_j\}.$$

By considering the value of $\alpha_{\max}$, we can distinguish following three cases.

3.3.1 Case: $\alpha_{\max} < 0$

For case $\alpha_{\max} < 0$, that is, $\alpha_j < 0$ for all $j = 1,...,n$ in (3.3.2), but the value of x_j is only positive or zero for all $j = 1,...,n$. Thus, the solution of (3.3.1) is trivial solution.

Lemma 3.3.1. Consider the linear programming model (3.3.1). Let $\alpha_j = \sum_{i=1}^{m} a_{ij}$ for each $j = 1,...,n$, and $\alpha_{\max} = \max_{1 \le j \le n} \{\alpha_j\}$. If $\alpha_{\max} < 0$, then the solution of the problem (3.3.1) is only trivial.

Proof. Suppose that $\alpha_{\max} < 0$. Then, $\alpha_j < 0$ for all $j = 1,...,n$. Assume that the solution is not trivial, there exists at least one x_j which $x_j > 0$ for some $j = 1,...,n$. Then, $\alpha_j x_j < 0$ for some $j = 1,...,n$. That is, $\sum_{j=1}^{n} \alpha_j x_j < 0$. This is a contradiction. The solution of the problem is only trivial. □

3.3.2 Case: $\alpha_{\max} = 0$ and it is a unique value

In this case, we can identify the solution of (3.3.1) by the following lemma.

Lemma 3.3.2. Consider the linear programming model (3.3.1). Let $\alpha_j = \sum_{i=1}^{m} a_{ij}$ for each $j = 1,...,n$, and $\alpha_k = \alpha_{\max} = \max_{1 \le j \le n} \{\alpha_j\}$. If $\alpha_{\max} = 0$ and it is a unique value, then the solution of the problem (3.3.1) is either

1. trivial solution if $c_k < 0$,
2. multiple solutions if $c_k = 0$,
3. unbounded if $c_k > 0$.

Proof. Suppose that $\alpha_k = \alpha_{\max} = 0$ and it is a unique value. Then, the sum of unoccupied rows is $\sum_{j=1, j \ne k}^{n} \alpha_j x_j + 0x_k = 0$. If there exists at least one of x_j which $x_j > 0$

for some $j \in \{1,...,n\} - \{k\}$, then $\sum_{j=1, j\neq k}^{n} \alpha_j x_j + 0x_k \neq 0$. Thus, $x_j = 0$ for all $j \in \{1,...,n\} - \{k\}$. Since $x_k \geq 0$, x_k can be either $x_k = 0$ or $x_k > 0$.

Consider the objective function, $z = c_1x_1 + c_2x_2 + \cdots + c_kx_k + \cdots + c_nx_n$. Since $x_j = 0$ for all $j \in \{1,...,n\} - \{k\}$, the objective function becomes $z = c_kx_k$.

Case $c_k < 0$: If $x_k > 0$, then $z = c_kx_k < 0$. Since the problem is the maximization problem, $x_k = 0$. If $x_k = 0$, then $z = 0$.

Case $c_k = 0$: If $x_k = 0$, then $z = 0$. If $x_k > 0$, then the problem has multiple solutions.

Case $c_k > 0$: If $x_k = 0$, then $z = 0$. If $x_k > 0$, then we can see that z increases when x_k increases. Therefore, the problem is unbounded. □

3.3.3 Case: $\alpha_{\max} = 0$ and it is not a unique value or $\alpha_{\max} > 0$

For this case, we cannot use the sum of unoccupied rows in the original problem for finding the solution directly. Because the degeneracy occurs if the problem is solved by the simplex method. Thus, we will change the original problem to another problem. Associated with each linear programming problem, as (3.1.1), there is another linear programming problem called the dual. The dual problem of (3.1.1) can be written as the following form:

$$
\begin{array}{lccccccccc}
\min & 0y_1 & + & 0y_2 & + & \cdots & + & 0y_m & & \\
\text{subject to} & a_{11}y_1 & + & a_{21}y_2 & + & \cdots & + & a_{m1}y_m & \geq & c_1 \\
& a_{12}y_1 & + & a_{22}y_2 & + & \cdots & + & a_{m2}y_m & \geq & c_2 \\
& \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
& a_{1n}y_1 & + & a_{2n}y_2 & + & \cdots & + & a_{mn}y_m & \geq & c_n, \\
& & & y_1, & & y_2, & \ldots, & y_m & \text{are unrestricted.} &
\end{array}
\tag{3.3.3}
$$

We will find a feasible solution of this problem. For the simplex method, we will convert the problem to the standard form. Since $y_1, y_2, ..., y_m$ are unrestricted,

we can let $y_i = y_i^+ - y_i^-$, where $y_i^+, y_i^- \geq 0$ for $i = 1, ..., m$. Thus, the dual problem in the standard form is as follows:

$$\begin{array}{lccccccc}
\min & 0(y_1^+ - y_1^-) & +\cdots+ & 0(y_m^+ - y_m^-) & & & & \\
\text{subject to} & a_{11}(y_1^+ - y_1^-) & +\cdots+ & a_{m1}(y_m^+ - y_m^-) & - & s_1 & = & c_1 \\
& a_{12}(y_1^+ - y_1^-) & +\cdots+ & a_{m2}(y_m^+ - y_m^-) & - & s_2 & = & c_2 \\
& \vdots & \vdots & & & & \vdots & \\
& a_{1n}(y_1^+ - y_1^-) & +\cdots+ & a_{mn}(y_m^+ - y_m^-) & - & s_n & = & c_n \\
& & y_1^+, y_1^-, \ldots, y_m^+, s_1, \ldots, s_n & & & & \geq & 0.
\end{array} \tag{3.3.4}$$

For some constraints, if there is some $c_j < 0$ for some $j = 1, ..., n$, then we convert it to positive value by multiplying the constraint by -1. In this case, we can recall the algorithm which is used to solve the nonhomogeneous linear programming problem to solve it.

By the equivalent of the primal and dual problem, we use the dual problem for finding the solution of the primal problem (3.3.1). Since the sum of unoccupied rows consideration cannot give the solution directly in this case, the dual problem of (3.3.3) is used for solving the problem (3.3.1). Since the solution of the primal homogeneous linear programming problem is only unbounded or trivial, the solution of the dual problem is only infeasible or optimal. If the dual problem is feasible, then the primal problem is optimal which the optimal value is zero. Otherwise, the primal problem can be concluded that it is unbounded since the dual problem is infeasible.

By the above idea, we can use the sum of unoccupied rows in the dual problem for identifying the solution of the homogeneous linear programming problem as the following steps.

Steps of Homogeneous Linear Programming Algorithm

Phase I: find the basic variable set

Step 1 Compute the sum of unoccupied rows of the problem (3.3.1)

$$\alpha_1 x_1 + \alpha_2 x_2 + \cdots + \alpha_n x_n = 0.$$

Step 2 Let $k = \underset{1 \le j \le n}{\operatorname{argmax}}\{\alpha_j\}$.

If $\alpha_k < 0$, then the solution of the problem (3.3.1) is trivial.

Else if $\alpha_{\max} = 0$ and it is a unique value,

if $c_k < 0$, then the optimal solution is trivial.

Else if $c_k = 0$, then $x_j = 0$ for all $j = 1, ..., n$, $j \neq k$ and

$x_k \geq 0$.

- Otherwise, the optimal solution is unbounded.

Else, the dual is used for finding the solution, and the nonhomogeneous linear programming problem algorithm is used.

- if the dual problem is feasible, then the optimal objective value is zero.

- Otherwise, the original problem is unbounded.

Example 3.3.1. Consider the following linear programming problem:

$$\begin{array}{lrcrcrcl} \max & & & 3x_1 & + & x_2 & + & 2x_3 & \\ \text{subject to} & - & x_1 & - & 3x_2 & + & 2x_3 & = & 0 \\ & - & 3x_1 & + & 2x_2 & - & x_3 & = & 0 \\ & & 4x_1 & + & x_2 & - & x_3 & = & 0 \\ & & x_1, & & x_2, & & x_3 & \geq & 0. \end{array}$$

Since the problem is in the standard form, we get the sum of unoccupied rows

$$0x_1 + 0x_2 + 0x_3 = 0. \tag{3.3.5}$$

We have $\alpha_1 = \alpha_2 = \alpha_3 = 0$, and $\beta = 0$. We see that, the largest coefficients of (3.3.5) is zero which it is not unique. Thus, we cannot identify the solution by considering only

the sum of all constraints of the primal problem. So, the dual of this problem is used for finding the solution. The dual problem is

$$\begin{array}{lrcrcrcl} \min & & & 0y_1 & + & 0y_2 & + & 0y_3 & \\ \text{subject to} & - & & y_1 & - & 3y_2 & + & 4x_3 & \geq 3 \\ & - & & 3y_1 & + & 2y_2 & + & y_3 & \geq 1 \\ & & & 2y_1 & - & y_2 & - & y_3 & \geq 2 \\ & & & & & y_1, & & y_2, \quad y_3 & \text{unrestricted.} \end{array}$$

Since y_1, y_2, y_3 are unrestricted, we let $y_i = y_i^+ - y_i^-$, where $y_i^+, y_i^- \geq 0$ for $i = 1,2,3$. Thus, the standard form of the dual problem can be written as follows:

$$\begin{array}{lrcrcrcrcrcrcl} \min & 0(y_1^+ - y_1^-) & + & 0(y_2^+ - y_2^-) & + & 0(y_3^+ - y_3^-) & & & & & & & \\ \text{subject to} & -(y_1^+ - y_1^-) & - & 3(y_2^+ - y_2^-) & + & 4(y_3^+ - y_3^-) & - & s_1 & & & & = & 3 \\ & -3(y_1^+ - y_1^-) & + & 2(y_2^+ - y_2^-) & + & (y_3^+ - y_3^-) & & & - & s_2 & & = & 1 \\ & 2(y_1^+ - y_1^-) & - & (y_2^+ - y_2^-) & - & (y_3^+ - y_3^-) & & & & & - & s_3 = & 2 \\ & y_1^+, \quad y_1^-, & & y_2^+, \quad y_2^-, & & y_3^+, \quad y_3^-, & & s_1, & & s_2, & & s_3 \geq & 0. \end{array}$$

This is a nonhomogeous linear programming problem. Construct the initial tableau and add the sum of unoccupied rows in the SUM row as the following tableau:

	BVS	y_1^+	y_1^-	y_2^+	y_2^-	y_3^+	y_3^-	s_1	s_2	s_3	RHS
SUM	–	−2	2	−2	2	4	−4	−1	−1	−1	6
	?	−1	1	−3	3	4	−4	−1	0	0	3
	?	−3	3	2	−2	1	−1	0	−1	0	1
	?	2	−2	−1	1	−1	1	0	0	−1	2

Consider the initial tableau, we get the sum of unoccupied rows as

$$-2y_1^+ + 2y_1^- - 2y_2^+ + 2y_2^- + 4y_3^+ - 4y_3^- - s_1 - s_2 - s_3 = 6. \tag{3.3.6}$$

Thus, y_3^+ is chosen into the basic variable set. Consider a row for entering y_3^+ by computing the minimum ratio as the following tableau.

	BVS	y_1^+	y_1^-	y_2^+	y_2^-	y_3^+	y_3^-	s_1	s_2	s_3	RHS	Ratio
SUM	–	−2	2	−2	2	4	−4	−1	−1	−1	6	
	?	−1	1	−3	3	4	−4	−1	0	0	3	3/4
	?	−3	3	2	−2	1	−1	0	−1	0	1	1/1
	?	2	−2	−1	1	−1	1	0	0	−1	2	–

Next, the nonbasic variable y_3^+ is entered into the basic variable set in the first row. Perform Gauss-Jordan row operation, we get the following tableau.

	BVS	y_1^+	y_1^-	y_2^+	y_2^-	y_3^+	y_3^-	s_1	s_2	s_3	RHS
SUM	–	−1	1	1	−1	0	0	0	−1	−1	3
	y_3^+	−0.25	0.25	−0.75	0.75	1	−1	−0.25	0	0	0.75
	?	−2.75	2.75	2.75	−2.75	0	0	0.25	−1	0	0.25
	?	1.75	−1.75	−1.75	1.75	0	0	−0.25	0	−1	2.75

Now, the sum of unoccupied rows is updated as below:

$$-y_1^+ + y_1^- + y_2^+ - y_2^- - s_2 - s_3 = 3. \tag{3.3.7}$$

Thus, y_1^- is chosen into the basic variable set. The tableau is updated as follows:

	BVS	y_1^+	y_1^-	y_2^+	y_2^-	y_3^+	y_3^-	s_1	s_2	s_3	RHS
SUM	–	0	0	0	0	0	0	−0.09	−0.63	−1	2.91
	y_3^+	0	0	−1	1	1	−1	−0.27	0.09	0	0.73
	y_1^-	−1	1	1	−1	0	0	0.09	−0.36	0	0.09
	?	0	0	0	0	0	0	−0.09	−0.63	−1	2.91

Now, the sum of unoccupied row is updated as below:

$$-0.09s_1 - 0.63s_2 - s_3 = 2.91. \tag{3.3.8}$$

Since the largest coefficient of (3.3.8) is zero, the solution of the dual problem is infeasible. By the relationship of the primal and dual problem, we can conclude that the solution of the original problem is unbounded.

■

3.4 The equivalent movement of the proposed algorithm and the two-phase method

From section 3.2, we found that the initial tableau of the proposed algorithm has the same reduced costs with the initial tableau of Phase I in the two-phase method. If the same pivot rule is used for both algorithms, then the movement of basic feasible solutions using the sum of unoccupied row behaves similar to Phase I in the two-phase method. Next, we present the movement of Phase I in the proposed method and the two-phase method as the following example.

Example 3.4.1. Consider the following linear programming problem:

$$\begin{array}{lrrrrrr} \max & - & x_1 & + & 2x_2 & & \\ \text{subject to} & & x_1 & + & x_2 & \geq & 2 \\ & - & x_1 & + & x_2 & \geq & 1 \\ & & & & x_2 & = & 3 \\ & & x_1, & & x_2 & \geq & 0. \end{array}$$

The proposed algorithm:

We first convert the problem to the standard form by substracting surplus variables s_1 and s_2 obtained as follows:

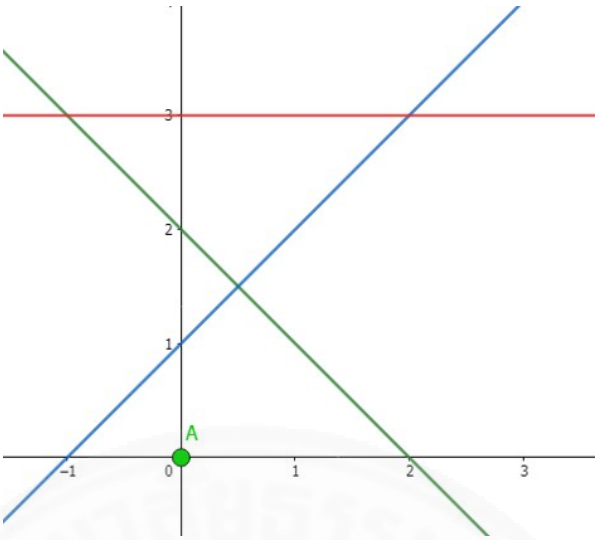

Figure 3.1: The basic variable set is empty

$$
\begin{array}{lrcrcrcrcl}
\max & - & x_1 & + & 2x_2 & & & & & \\
\text{subject to} & & x_1 & + & x_2 & - & s_1 & & = & 2 \\
& - & x_1 & + & x_2 & & & - \; s_2 & = & 1 \\
& & & & x_2 & & & & = & 3 \\
& & x_1, & & x_2, & & s_1, & s_2 & \geq & 0.
\end{array}
$$

Phase I

The initial tableau of the problem is as below. The movement starts at the origin point as Figure 3.1.

	BVS	x_1	x_2	s_1	s_2	RHS
SUM	−	0	3	−1	−1	6
	?	1	1	−1	0	2
	?	−1	1	0	−1	1
	?	0	1	0	0	3

Since $\max\{\alpha_j : j \in \{1,2,3,4\}\} = \alpha_2 = 3 > 0$ and $\beta = 6 > 0$, x_2 is the entering variable. Then, we examine $\mathbf{y}_2$.

$$\frac{\bar{b}_r}{y_{r2}} = \underset{1 \leq i \leq 3}{\text{minimum}} \left\{ \frac{\bar{b}_1}{y_{12}}, \frac{\bar{b}_2}{y_{22}}, \frac{\bar{b}_3}{y_{32}} \right\} = \min \left\{ \frac{2}{1}, \frac{1}{1}, \frac{3}{1} \right\}.$$

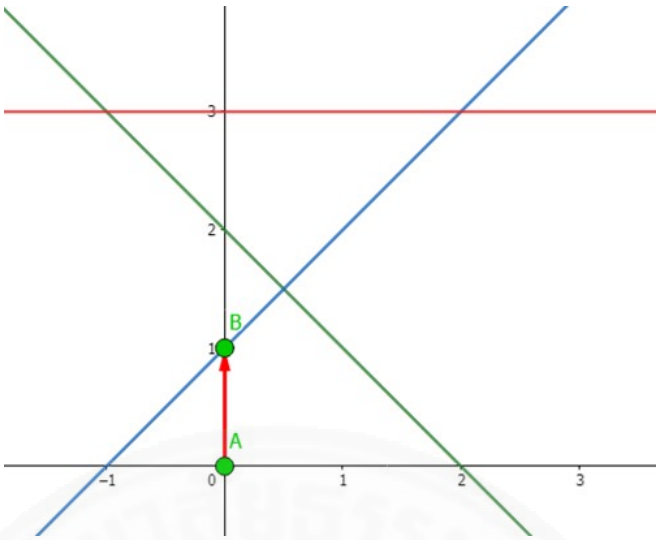

Figure 3.2: The graph after x_2 is entered into the basic variable set

Thus, x_2 is entered into the second row. We update the tableau by pivoting at y_{22}. We get the following tableau and the graph as Figure 3.2.

	BVS	x_1	x_2	s_1	s_2	RHS
SUM	$-$	3	0	-1	2	3
	?	2	0	-1	1	1
	x_2	-1	1	0	-1	1
	?	1	0	0	1	2

Since the basic variable set is not full, the next nonbasic variable is chosen to enter the basic variable set. Consider $\max\{\alpha_j : j \in \{1,3,4\}\} = \alpha_1 = 3 > 0$ and $\beta = 3 > 0$. Thus, x_1 is the entering variable. Then, we examine $\mathbf{y}_1$.

$$\frac{\overline{b}_r}{y_{r1}} = \underset{1\le i\le 3}{\text{minimum}}\left\{\frac{\overline{b}_1}{y_{11}}, \frac{\overline{b}_3}{y_{31}}\right\} = \min\left\{\frac{1}{2}, \frac{2}{1}\right\}.$$

Thus, x_1 is entered into the first row. We update the tableau by pivoting at y_{11}. We get the following tableau and the graph as Figure 3.3.

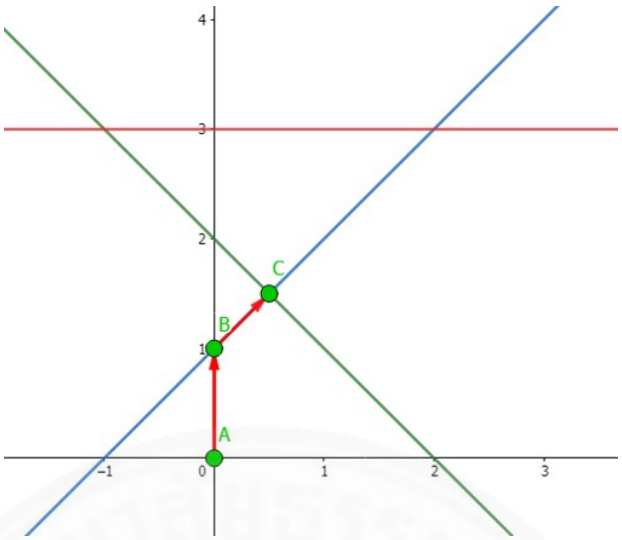

Figure 3.3: The graph after x_1 is entered into the basic variable set

	BVS	x_1	x_2	s_1	s_2	RHS
SUM	–	0	0	1/2	1/2	3/2
	x_1	1	0	−1/2	1/2	1/2
	x_2	0	1	−1/2	−1/2	3/2
	?	0	0	1/2	1/2	3/2

Next, we will choose the nonbasic variable to enter the basic variable set until the basic variable set is full. Consider $\max\{\alpha_j : j \in \{3,4\}\} = \alpha_3 = \frac{1}{2} > 0$ and $\beta = \frac{3}{2} > 0$. Thus, s_1 is the entering variable. Then, we examine $\mathbf{y}_3$

$$\frac{\bar{b}_r}{y_{r3}} = \underset{1\le i\le 3}{\text{minimum}}\left\{\frac{\bar{b}_3}{y_{33}}\right\} = \min\{3\}.$$

Thus, s_1 is entered into the third row. We update the tableau by pivoting at y_{33}. We get the following tableau and the graph as Figure 3.4.

	BVS	x_1	x_2	s_1	s_2	RHS
SUM	–	0	0	0	0	0
	x_1	1	0	0	1	2
	x_2	0	1	0	0	3
	s_1	0	0	1	1	3

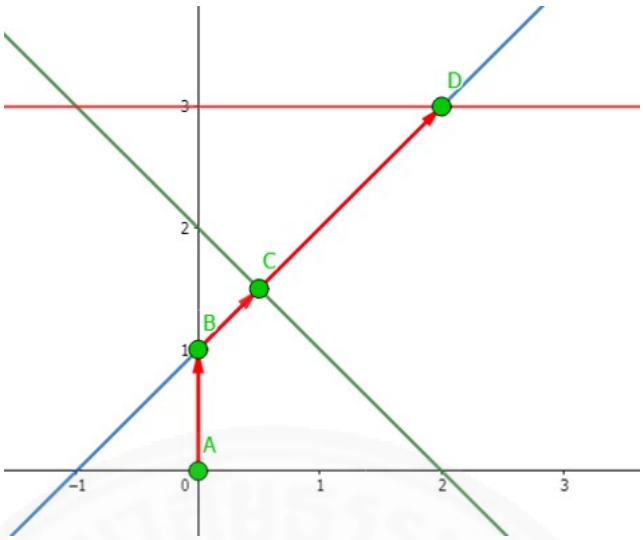

Figure 3.4: The graph after s_1 is entered into the basic variable set

Now, the basic variable set is full. Thus, the Phase I of the proposed algorithm ends.

The two-phase method

We convert the problem to the standard form by substracting surplus variables s_1, s_2 and adding artificial variables $x_{a_1}, x_{a_2}, x_{a_3}$ and the objective function minimize the summation of artificial variables obtained as follows:

Phase I

$$
\begin{array}{lrcrcrcrcrcrcl}
\text{min} & x_{a_1} & + & x_{a_2} & + & x_{a_3} & & & & & & & & \\
\text{subject to} & x_1 & + & x_2 & - & s_1 & & + & x_{a_1} & & & & = & 2 \\
- & x_1 & + & x_2 & & & - & s_2 & & + & x_{a_2} & & = & 1 \\
& & & x_2 & & & & & & & & + & x_{a_3} & = & 3 \\
& x_1, & & x_2, & & s_1, & & s_2 & x_{a_1} & & x_{a_2} & & x_{a_3} & \geq & 0.
\end{array}
$$

We can choose an initial basis as $\mathbf{B} = [\mathbf{A}_{:5}, \mathbf{A}_{:6}, \mathbf{A}_{:7}] = \mathbf{I}_3$ and fill coefficeint matrix into the tableau:

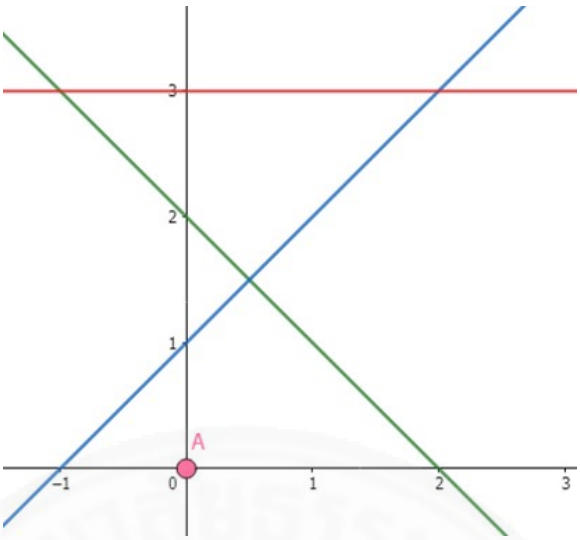

Figure 3.5: All artificial variables are in the basic variable set

	z	x_1	x_2	s_1	s_2	x_{a_1}	x_{a_2}	x_{a_3}	RHS
	1	0	0	0	0	−1	−1	−1	0
x_{a_1}	0	1	1	−1	0	1	0	0	2
x_{a_2}	0	−1	1	0	−1	0	1	0	1
x_{a_3}	0	0	1	0	0	0	0	1	3

Since x_{a_1}, x_{a_2} and x_{a_3} are in basis, their reduced cost are zero. Then, the initial tableau is as below. The movement starts at the origin point as Figure 3.5.

	z	x_1	x_2	s_1	s_2	x_{a_1}	x_{a_2}	x_{a_3}	RHS
	1	0	3	−1	−1	0	0	0	6
x_{a_1}	0	1	1	−1	0	1	0	0	2
x_{a_2}	0	−1	1	0	−1	0	1	0	1
x_{a_3}	0	0	1	0	0	0	0	1	3

Since $\max\{z_j - c_j : j \in \{1,2,3,4,5,6,7\}\} = z_2 - c_2 = 3 > 0$, x_2 is the entering variable. Then, we examine $\mathbf{y}_2$.

$$\frac{\overline{b}_r}{y_{r2}} = \underset{1\le i\le 3}{\text{minimum}}\left\{\frac{\overline{b}_1}{y_{12}}, \frac{\overline{b}_2}{y_{22}}, \frac{\overline{b}_2}{y_{32}}\right\} = \min\left\{\frac{2}{1}, \frac{1}{1}, \frac{3}{1}\right\}.$$

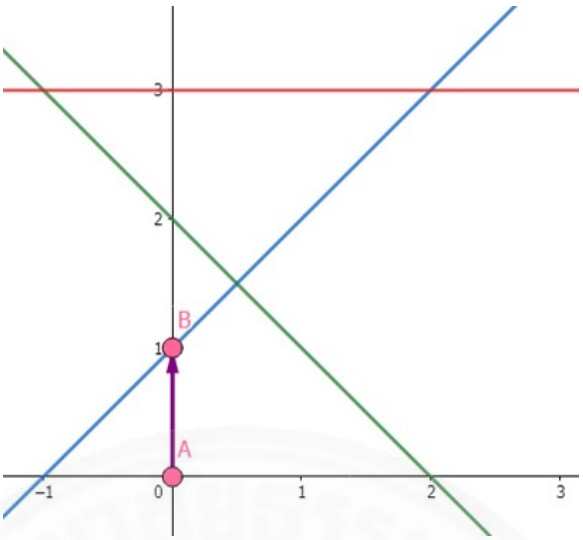

Figure 3.6: The graph after x_2 is entered into the basic variable set

Thus, x_{a_2} is the leaving variable. We update the tableau by pivoting at y_{22}. Then, we get the following tableau and the graph as Figure 3.5.

	z	x_1	x_2	s_1	s_2	x_{a_1}	x_{a_2}	x_{a_3}	RHS
	1	3	0	−1	2	0	−3	0	3
x_{a_1}	0	2	0	−1	1	1	−1	0	1
x_2	0	−1	1	0	−1	0	1	0	1
x_{a_3}	0	1	0	0	1	0	−1	1	2

Since $\max\{z_j - c_j : j \in \{1,3,4,5,6,7\}\} = z_1 - c_1 = 3 > 0$, x_1 is the entering variable. Then, we examine $\mathbf{y}_1$.

$$\frac{\overline{b}_r}{y_{r1}} = \underset{1\le i\le 3}{\text{minimum}}\left\{\frac{\overline{b}_1}{y_{11}}, \frac{\overline{b}_2}{y_{31}}\right\} = \min\left\{\frac{1}{2}, \frac{1}{1}, \frac{2}{1}\right\}.$$

Thus, x_{a_1} is the leaving variable. We update the tableau by pivoting at y_{11}, and the variable x_2 enters the basis and x_{a_1} leaves the basis. We get the following tableau and the graph as Figure 3.6.

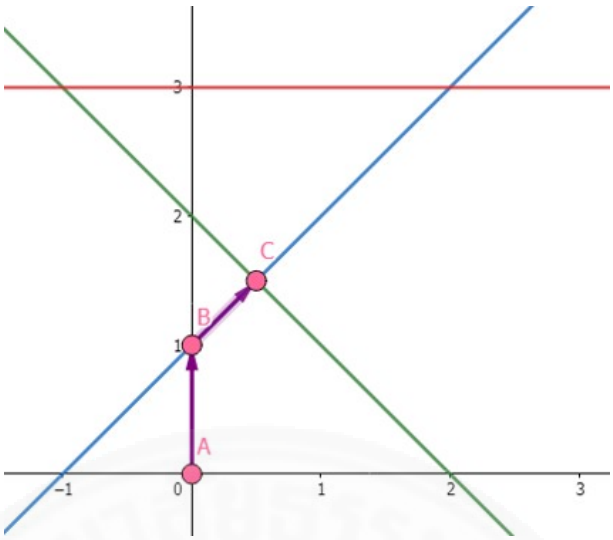

Figure 3.7: The graph after x_1 is entered into the basic variable set

	z	x_1	x_2	s_1	s_2	x_{a_1}	x_{a_2}	x_{a_3}	RHS
	1	0	0	1/2	1/2	−3/2	−3/2	0	3/2
x_1	0	1	0	−1/2	1/2	1/2	−1/2	0	1/2
x_2	0	0	1	−1/2	−1/2	1/2	1/2	0	3/2
x_{a_3}	0	0	0	1/2	1/2	−1/2	−1/2	1	3/2

Since $\max\{z_j - c_j : j \in \{3,4,5,6,7\}\} = z_3 - c_3 = \frac{1}{2} > 0$, s_1 is the entering variable. Then, we examine $\mathbf{y}_3$.

$$\frac{\bar{b}_r}{y_{r3}} = \underset{1\le i\le 3}{\text{minimum}}\left\{\frac{\bar{b}_3}{y_{33}}\right\} = \min\{3\}.$$

Thus, x_{a_3} is the leaving variable. We update the tableau by pivoting at y_{33} where s_1 enters the basis and x_{a_3} leaves the basis. We get the following tableau and the graph as Figure 3.7.

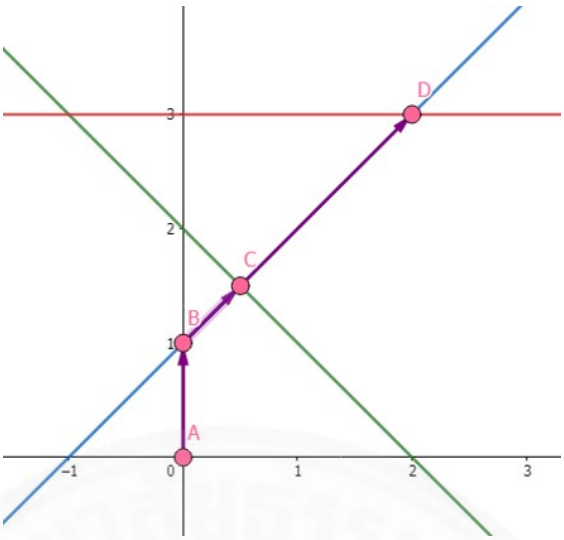

Figure 3.8: The graph after s_1 is entered into the basic variable set

	z	x_1	x_2	s_1	s_2	x_{a_1}	x_{a_2}	x_{a_3}	RHS
	1	0	0	0	0	−1	−1	−1	0
x_1	0	1	0	0	1	0	−1	1	2
x_2	0	0	1	0	0	0	0	1	3
s_1	0	0	0	1	1	−1	−1	2	3

Now, the summation of artificial variables is zero. Thus, Phase I of the two-phase method ends.

By comparison of both algorithms, we found that the movement of the proposed algorithm behaves similar to Phase I in the two-phase method.

CHAPTER 4

COMPUTATIONAL RESULTS

In this chapter, we do further computational study to test the efficiency of the proposed algorithm comparing to the Gao's algorithm. It is given some indication of the performance of the two algorithms on the 100 randomly tested problems in each size of matrices which the value of elements in matrix **A** and **c** are in [-9,9] and each element in **b** is in [-9,9]. The proposed algorithm and the Gao's algorithm were implemented by MATLAB R2014a and run on an Intel(R) Core(TM) i5-2410M 2.30 GHz and 4.00 GB of RAM. The computational results consisting of the average number of iterations and the average computational time are reported as the following table.

TABLE 4.1: The average number of iterations and computational time for $m \times n$ matrices

Size of Matrices	The average number of iterations		The average computational time	
	LC algorithm	Gao's algorithm	LC algorithm	Gao's algorithm
10×20	**12.49**	17.28	**0.01554**	0.08008
10×40	**16.41**	22.00	**0.01614**	0.08391
10×60	**16.86**	22.12	**0.01702**	0.08746
20×40	**27.38**	39.46	**0.02035**	0.09917
20×60	**37.14**	46.15	**0.02307**	0.10614

From TABLE 4.1, the average number of iterations of the proposed algorithm is less than the average number of iterations of Gao's algorithm for all sizes of tested problem. Moreover, the average computational time of the proposed algorithm is less than Gao's algorithm in all sizes of matrices.

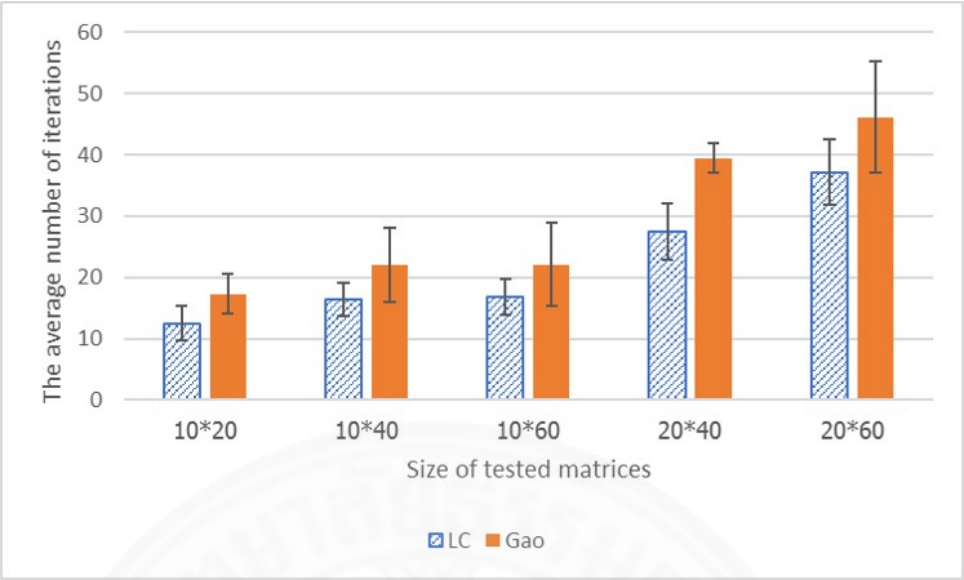

Figure 4.1: The average number of iterations for each size of problems

By observing the standard deviations in Figure 4.1, we found that Gao's algorithm has the high standard deviations for 10×40, 10×60, and 20×60 matrices, it means that the number of iterations is spread out. However, it is low for 10×20 and 20×40 matrices which can indicate that the number of iterations tends to be close to its mean while the proposed algorithm has the low standard deviations for every size of tested matrices. That means the number of iterations of the proposed algorithm tends to be close to its mean.

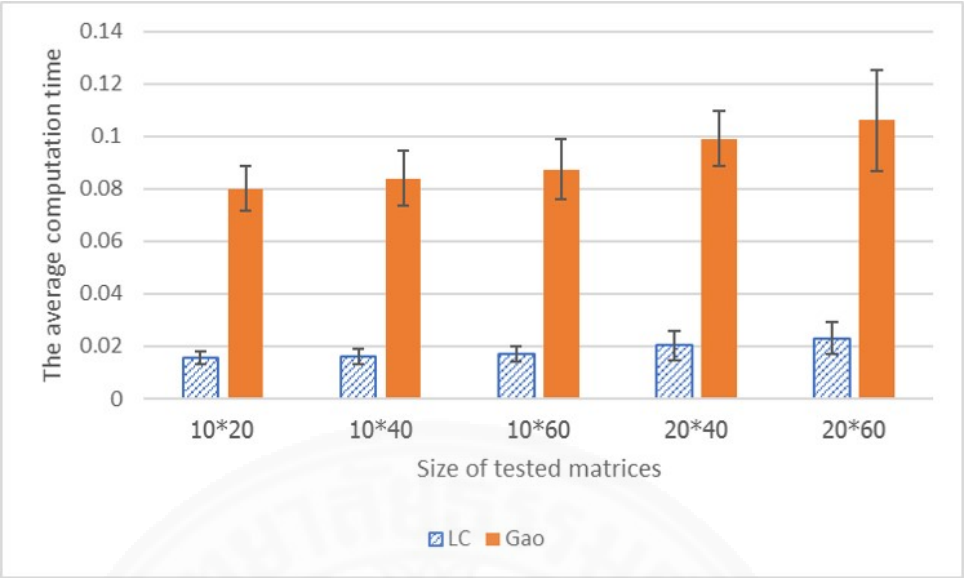

Figure 4.2: The average computational time for each size of problems

From Figure 4.2, we found that the average computational time of both algorithm is more different for every size of tested matrices. The average computational time of the proposed algorithm is less than the average computational time of Gao's algorithm for all sizes of tested problems.

By observing the standard deviations in Figure 4.2, we found that Gao's algorithm has the high standard deviations for 10×60 matrices, it means that the computational time is spread out. While, for 10×20, 10×40, 20×20, 20×40, and 20×60 matrices, Gao's algorithm has the low standard deviations which can indicate that the computational time tends to be close to the mean. However, the proposed algorithm has the low standard deviations for every size of tested matrices which can indicate that the computational time tends to be close to its mean.

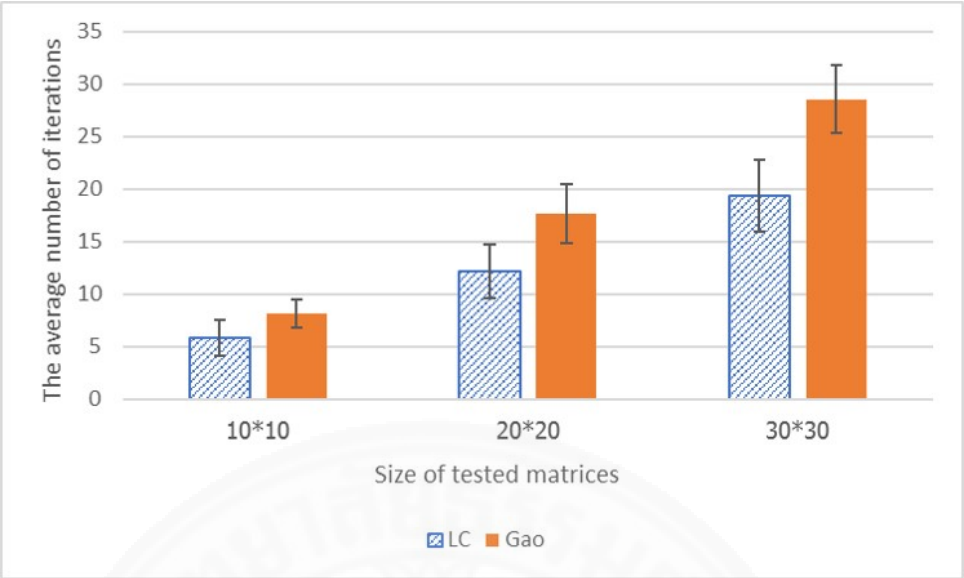

Figure 4.3: The average computational time for each size of problems

TABLE 4.2: The average number of iterations and computational time for $n \times n$ matrices

Size of Matrices	The average number of iterations		The average computational time	
	LC algorithm	Gao's algorithm	LC algorithm	Gao's algorithm
10×10	**5.83**	8.20	**0.01378**	0.07415
20×20	**12.23**	17.64	**0.01612**	0.08208
30×30	**19.32**	28.52	**0.01980**	0.09607

From TABLE 4.2, the average number of iterations of the proposed algorithm is less than the average number of iterations of Gao's algorithm for all sizes of tested problem. Moreover, the average computational time of the proposed algorithm is less than Gao's algorithm in all sizes of matrices.

By observing the standard deviations in Figure 4.3, we found that both algorithm has the low standard deviations for all sizes of tested matrices, it means the number of iterations tends to be close to their mean.

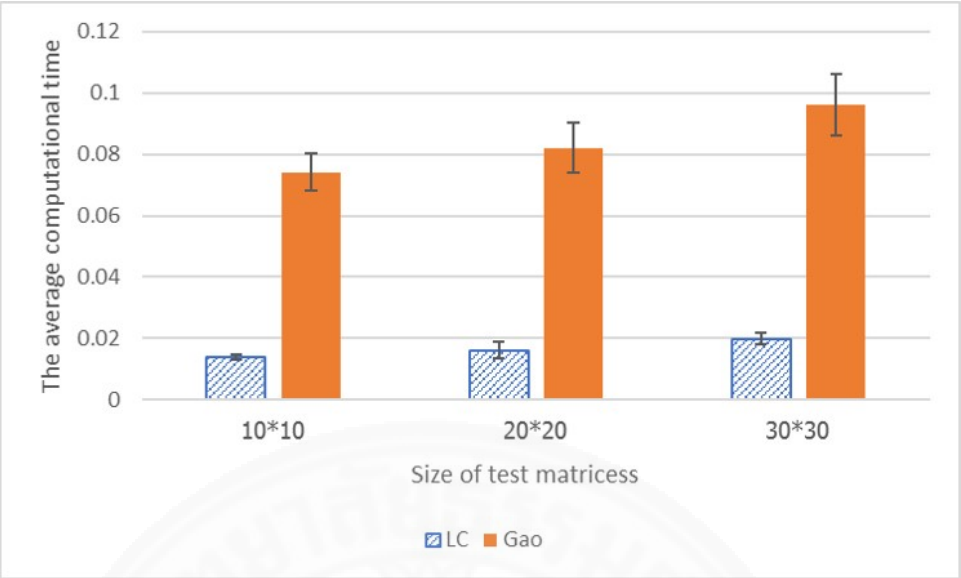

Figure 4.4: The average computational time for each size of problems

By observing the standard deviations in Figure 4.4, we found that both algorithm has the low standard deviations for all sizes of tested matrices, it means the computational time tends to be close to their mean.

By all figures, we see that both the average number of iterations and computational time of the proposed algorithm are less than Gao's algorithm in all size of matrices. Moreover, the difference of the average computational time is more when the size of matrices is bigger. That is, the average computational time of the proposed algorithm is less than Gao's algorithm.

CHAPTER 5

CONCLUSIONS

In this thesis, we propose the algorithm for starting the simplex algorithm without using artificial variables. First, we use the sum of all constraints for starting the simplex method since it can be used for identifying the existence of the solution to a linear programming problem. By considering the sum of all constraints, if its largest coefficient is negative, then the infeasibility is reported immediately. Otherwise, the proposed method starts with an empty basic variable set. Then, a nonbasic variable with the largest coefficient is chosen to be a basic variable and is added into the row which has the minimum ratio and Gauss-Jordan row operations are performed. If the row has a basic variable, then it is called an occupied row. Otherwise, it is called an unoccupied row. Next, we choose other nonbasic variables one by one into the basic variable set by considering the sum of unoccupied rows until the basic variable set is full or the infeasibility is reported. In each iteration, the proposed algorithm can detect infeasibility by considering the largest coefficient of the sum of unoccupied rows which is negative. Thus, the proposed algorithm has more efficiency for checking an infeasible problem.

By observing the movement of basic feasible solutions in two-dimension problem, we found that graphs of the proposed algorithm behaves similar to Phase I in the two-phase method. Moreover, if the reduced costs of the initial tableau of the proposed algorithm is the same the initial tableau of Phase I in the two-phase method, then the graph in each iteration is also identical.

From the computational results, we found that both the average number of iterations and the computational time of the proposed algorithm are less than Gao's algorithm in all sizes of tested problems. Because Gao's algorithm needs to perform Arsham's algorithm first and Arsham's algorithm reported infeasibility. Therefore, by

the results, the proposed algorithm can reduce both the number of iterations and the computational time.

Although the average number of iterations and the computational time of the proposed algorithm are less than Gao's algorithm, we can develope the efficiency of the proposed algorithm by changing the pivot rule. Since we consider the largest positive coefficient of the sum of unoccupied rows to choose an entering variable which is similar to Dantzig's rule, we can use other pivot rules to choose an entering variable from all nonbasic variables which have the positive coefficient of the sum of unoccupied rows. According to these results, the number of iterations and the computational time might be reduced.

REFERENCES

[1] G. Dantzig, Programming in a linear structure, Comptroller, USAF, Washington D.C., (1948).

[2] N. K. Karmarkar, A new polynomial-time algorithm for linear programming, in: Proceedings of the 16th Annual ACM Symposium on Theory of Computing, (1984), 302-311.

[3] H. Arsham, An Artificial-Free Simplex-Type Algorithm for General LP Models, Mathematical and Computer Modelling, 25(1997), 107-123.

[4] A. Enge, P. Huhn, A Counterexample to H. Arsham's "Initialization of the Simplex Algorithm: an Artificial-Free Approach", SIAM Review, 40(1998), 1-6.

[5] P. Gao, Improvement and its computer implementation of an artificial-free simplex-tupe algorithm by Arsham, Applied Mathematics and Computation, 263(2015), 410-415.

[6] B. Kolman, R. Beck, Elementary Linear Programming with Applications, Academic Press, (2008).

[7] G. Dantzig, Linear Programming and tensions, Princeton University Press, New Jersey, (1968).

[8] H. Arsham, G. Cimperman, N. Damij, T. Damij, J. Grad, A Computer Implementation of the Push-and-Pull Algorithm and Its Computational Comparison With LP Simplex method, Applied Mathematics and Computation, 170(2005), 36-63.

[9] H. Eiselt, C. Sandblom, Linear Programming and Its Applications, Berlin New York: Springer, (2007).

[10] M. S. Bazaraa, J. Javis, H. D. Sherail, Linear Programming and Network Flows, 2nd edn. New York : John Wiley, (1990).

[11] T. Hu, B. Kahng, Linear Integer Programming Made Easy, Switzerland: Springer, (2016).

[12] W. Yeh, H. W. Corley, A simple direct cosine simplex algorithm, Applied Mathematics and Computation, 214(2009), 178-186.

BIOGRAPHY

Name	Miss Tanchanok Phumrachat
Date of Birth	January 24, 1994
Educational Attainment	Academic Year 2015: Bachelor of Science (Mathematics), Thammasat University, Thailand Academic Year 2017: Master of Science (Mathematics), Thammasat University, Thailand
Scholarships	2016: Graduate Scholarship from the Faculty of Science and Technology (Thammasat University)

Publications

1. T. Phumrachat, A. Boonperm, Constructing an Initial Basis for an Artificial-free Simplex Algorithm Based on a Linear Combination of All Constraints Consideration, Proceedings of The Operations Research Network of Thailand (OR-NET 2018), (2018), 288-294.

2. T. Phumrachat, A. Boonperm, A Simple Technique for Identifying a Solution to a Homogeneous Linear Programming Problem, Proceedings of 10th International Conference on Advences in Science, Engineering and Technology (ICASET-18), (2018), 27-32.

Oral Presentations

1. T. Phumrachat, A. Boonperm, Constructing an Initial Basis for an Artificial-free Simplex Algorithm Based on a Linear Combination of All Constraints Consideration, in Operations Research Network of Thailand (OR-NET 2018), April 23 - 24, 2018, Chonburi, Thailand.

2. T. Phumrachat, A. Boonperm, A Simple Technique for Identifying a Solution to a Homogeneous Linear Programming Problem, in 10th International Conference

on Advences in Science, Engineering and Technology (ICASET-18), June 20 - 21, 2018, Paris, France.