หลักเกณฑ์การหมุนของวิธีซิมเพล็กซ์ที่เน้นการเพิ่มค่าของตัวแปรไม่พื้นฐาน

นายกิตติพงษ์ จันทร์คง

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาคณิตศาสตร์ประยุกต์และวิทยาการคณนา

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2556

SIMPLEX PIVOT RULE EMPHASIZING INCREMENT OF NONBASIC VARIABLES

Mr. Kittiphong Chankong

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Applied Mathematics and
Computational Science
Department of Mathematics and Computer Science
Faculty of Science
Chulalongkorn University
Academic Year 2013

Thesis Title	SIMPLEX PIVOT RULE EMPHASIZING INCREMENT OF NONBASIC VARIABLES
By	Mr. Kittiphong Chankong
Field of Study	Applied Mathematics and Computational Science
Thesis Advisor	Boonyarit Intiyot, Ph.D.
Thesis Co-advisor	Assistant Professor Krung Sinapiromsaran, Ph.D.

Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

......................................Dean of the Faculty of Science
(Professor Supot Hannongbua, Dr.rer.nat.)

THESIS COMMITTEE

......................................Chairman
(Associate Professor Pornchai Satravaha, Ph.D.)

......................................Thesis Advisor
(Boonyarit Intiyot, Ph.D.)

......................................Thesis Co-advisor
(Assistant Professor Krung Sinapiromsaran, Ph.D.)

......................................Examiner
(Phantipa Thipwiwatpotjana, Ph.D.)

......................................External Examiner
(Associate Professor Peerayuth Charnsethikul, Ph.D.)

กิตติพงษ์ จันทร์คง : หลักเกณฑ์การหมุนของวิธีซิมเพล็กซ์ที่เน้นการเพิ่มค่าของตัวแปรไม่พื้นฐาน. (SIMPLEX PIVOT RULE EMPHASIZING INCREMENT OF NONBASIC VARIABLES) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: อ.ดร. บุญฤทธิ์ อินทิยศ, อ.ที่ปรึกษาวิทยานิพนธ์ร่วม : ผศ.ดร.กรุง สินอภิรมย์สราญ, 64 หน้า.

ขั้นตอนวิธีซิมเพล็กซ์เป็นระเบียบวิธีที่ถูกใช้อย่างแพร่หลายในการแก้ปัญหาการกำหนดการเชิงเส้น ซึ่งนำเสนอครั้งแรกโดย จอร์จ แดนท์ซิก ขั้นตอนสำคัญในขั้นตอนวิธีซิมเพล็กซ์คือการประยุกต์หลักเกณฑ์การหมุนที่เหมาะสมเพื่อที่จะใช้เลือกตัวแปรเข้า หลักเกณฑ์การหมุนที่มีประสิทธิภาพสามารถที่จะนำไปสู่ผลลัพธ์เหมาะสุดของปัญหาการกำหนดการเชิงเส้นด้วยจำนวนการทำซ้ำที่น้อยแต่ไม่จำเป็นที่จะทำให้เวลาในการคำนวณน้อยถ้าแต่ละการทำซ้ำใช้เวลามาก ในปัญหาการหาค่าน้อยสุด หลักเกณฑ์การหมุนแบบแดนท์ซิกเลือกตัวแปรเข้าที่สอดคล้องกับ reduced cost ที่เป็นลบมากที่สุด โดยแนวคิดคือต้องการที่จะปรับปรุงค่าจุดประสงค์ต่อหน่วยของตัวแปรเข้าให้ได้มากที่สุด อย่างไรก็ตามหลักเกณฑ์การหมุนแบบแดนท์ซิกอาจจะเคลื่อนไปสู่จุดมุมของบริเวณผลลัพธ์ที่เป็นไปได้หลายครั้งก่อนที่จะไปถึงผลลัพธ์เหมาะสุด ในวิทยานิพนธ์นี้ นำเสนอหลักเกณฑ์การหมุนของวิธีซิมเพล็กซ์ที่เรียกว่าหลักเกณฑ์การหมุนแบบการเปลี่ยนสัมบูรณ์ ซึ่งสามารถที่จะลดจำนวนการทำซ้ำให้น้อยกว่าหลักเกณฑ์การหมุนแบบแดนท์ซิก แนวคิดคือต้องการปรับปรุงค่าของฟังก์ชันจุดประสงค์ให้มากที่สุดโดยการกันตัวแปรออกที่ทำให้ค่าของฟังก์ชันจุดประสงค์เปลี่ยนไปได้น้อยให้มากสุดเท่าที่เป็นไปได้ วิธีการหมุนแบบเปลี่ยนสัมบูรณ์นำมาทดสอบและเปรียบเทียบประสิทธิภาพกับหลักเกณฑ์การหมุนแบบแดนท์ซิกและหลักเกณฑ์การหมุนแบบอื่นๆ

ภาควิชา	คณิตศาสตร์ และ	ลายมือชื่อนิสิต	....
	วิทยาการคอมพิวเตอร์	ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก	....
สาขาวิชา	คณิตศาสตร์ประยุกต์ และ	ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์ร่วม	....
	วิทยาการคณนา		
ปีการศึกษา	2556		

5571920923 : MAJOR APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE KEYWORDS : LINEAR PROGRAMMING / SIMPLEX ALGORITHM / PIVOT RULE / ABSOLUTE CHANGE PIVOT RULE

KITTIPHONG CHANKONG : SIMPLEX PIVOT RULE EMPHASIZING INCREMENT OF NONBASIC VARIABLES. ADVISOR : BOONYARIT INTIYOT, Ph.D., CO-ADVISOR : ASST. PROF. KRUNG SINAPIROMSARAN, Ph.D., 64pp.

The simplex algorithm, first presented by George B. Dantzig, is a widely used method for solving a linear programming (LP) problem. One of the important steps of the simplex algorithm is applying a pivot rule, the rule to select an entering variable. An effective pivot rule can lead to an optimal solution of an LP problem with a small number of iterations but not necessarily small computational time if each iteration spends a lot of time. In a minimization problem, Dantzig's pivot rule selects an entering variable corresponding to the most negative reduced cost. The concept is to have the maximum improvement in the objective value per unit change of an entering variable. However, in some problems, Dantzig's rule may visit a large number of extreme points before reaching the optimal solution. In this thesis, we propose a pivot rule, called the absolute change pivot rule, that could reduce the number of such iterations over the Dantzig's pivot rule. The idea is to have the maximum improvement in the value of an objective function by trying to block a leaving variable that makes a little change in the objective value as much as possible. This absolute change pivot rule is tested and compared the efficacy with Dantzig's original pivot rule and other pivot rules.

Department	:	Mathematics and	Student's Signature ....
		Computer Science	Advisor's Signature ....
Field of Study	:	Applied Mathematics and	Co-advisor's Signature ....
		Computational Science	
Academic Year	:	2013	

ACKNOWLEDGEMENTS

I would like to express my thanks to my advisor and co-advisor, Dr. Boonyarit Intiyot and Assistant Professor Dr. Krung Sinapiromsaran, for their invaluable help and encouragement throughout the course of this thesis. I am most grateful for their suggestions and teaching, not only research methodologies but also many methodologies in life. Without their support, this thesis could not have been completed.

I also would like to give thanks to my thesis committees, Associate Professor Dr. Pornchai Satravaha and Dr. Phantipa Thipwiwatpotjana, and my thesis external examiner, Associate Professor Dr. Peerayuth Charnsethikul who is the lecturer at Department of Industrial Engineering, Faculty of Engineering, Kasetsart University. Moreover, I would like to thank all lecturers who instructed and taught me for valuable knowledge.

Additionally, I would like to thank those whose names are not mentioned here but greatly inspired and encouraged us until this thesis comes to the end.

Finally, I most gratefully acknowledge my parents and my friends for all their support throughout the period of this thesis. I also most gratefully thank to the Applied Mathematics and Computational Science, Faculty of Science and Graduate school, Chulalongkorn University for financial support to the international conference.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I
INTRODUCTION

Mathematical programming is one of mathematical tools used to optimally allocate limited resources such as money, materials, labor and machines to meet certain constraints. The goal of mathematical programming is to obtain the optimal result, such as maximum profit or minimum cost, under those conditions effectively. Typical example would be solving optimization problems of the limited money, supplies, materials, labor and machines in various industries.

Linear programming (LP) is a special case of mathematical programming. This field of study involves techniques for optimizing a linear objective function subject to finite number of linear equality and inequality constraints. To solve an LP problem, we need to consider the computational complexity that depends on the number of constraints and variables.

In this chapter, we will give some general background of linear programming problems. The topics presented in this chapter are: definition of linear programming, linear programming modeling, solution to linear programming, motivation and problem description of our propose technique, objective of this thesis and the scope of the thesis.

1.1 The Linear Programming Problem

The discussion begins by introducing basic definitions of a linear programming problem.

1.1.1 Definition

The mathematical model of a linear programming problem is to find the value of decision variables $x_1, x_2, \ldots, x_n$ that achieve the minimum of the objective value under a set of constraints. This model is defined as

$$\begin{array}{lrcrcrcl} \text{Minimize} & c_1x_1 & + & c_2\ x_2 & +\cdots+ & c_nx_n & & \\ \text{subject to} & a_{11}x_1 & + & a_{12}\ x_2 & +\cdots+ & a_{1n}x_n & \geq & b_1 \\ & a_{21}x_1 & + & a_{22}\ x_2 & +\cdots+ & a_{2n}x_n & \geq & b_2 \\ & \vdots & + & \vdots & +\cdots+ & \vdots & \geq & \vdots \\ & a_{m1}x_1 & + & a_{m2}\ x_2 & +\cdots+ & a_{mn}x_n & \geq & b_m \\ & x_1, & & x_2, & \ldots, & x_n & \geq & 0. \end{array} \tag{1.1}$$

The i^{th} constraint can be written in the form $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$. The coefficients a_{ij} for $i = 1, 2, ..., m$, $j = 1, 2, ..., n$ are called the **technological coefficients**. These technological coefficients form the constraint matrix $\mathbf{A}$. The decision variables $x_1, x_2, \ldots, x_n$ must be greater than or equal to zero, which are called **nonnegative constraints**. If the decision variable x_j can be negative, x_j is said to be unrestricted in sign.

1.1.2 Maximization and Minimization Problems

The maximization of the objective function can be converted into the minimization of the objective function as follows:

$$\text{Maximize} \sum_{j=1}^{n} c_jx_j = -\text{Minimize} \sum_{j=1}^{n} -c_jx_j.$$

It is easy to change a maximization problem into a minimization problem by multiplying the coefficients of the objective function by -1. After the solution of the minimum linear programming problem is achieved, the optimal objective function is its negative value.

1.1.3 Standard and Canonical Formats

The linear programming problem can be written in two formats. These formats are standard form and canonical form.

Standard Form

A linear programming problem in the **standard form** is a linear program that all restrictions are equalities and all variables are nonnegative. The objective function may be in a minimization form or a maximization form. The values of the right-hand-side vector (b_i) of all constraints are greater than or equal to zero ($b_i \geq 0$).

The standard form of a linear programming problem with n variables and m constraints is defined by:

$$\begin{array}{lrcl}
\text{Minimize (or Maximize)} & c_1x_1 + c_2x_2 + \cdots + c_nx_n & & \\
\text{subject to} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\
& a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = & b_2 \\
& \vdots + \vdots + \cdots + \vdots & = & \vdots \\
& a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & = & b_m \\
& x_1, \quad x_2, \quad \ldots, \quad x_n & \geq & 0.
\end{array} \tag{1.2}$$

Canonical Form

In this form, if a linear program is a minimization problem then all variables are nonnegative and all the constraints are of the type $\geq$. On the other hand, if a linear program is a maximization problem then all variables are nonnegative and all the constraints are of the type $\leq$.

The canonical form of a linear programming problem with n variables and m constrants is defined by:

$$\begin{array}{lrcl}
\text{Minimize} & c_1x_1 + c_2\,x_2 + \cdots + c_nx_n & & \\
\text{subject to} & a_{11}x_1 + a_{12}\,x_2 + \cdots + a_{1n}x_n & \geq & b_1 \\
& a_{21}x_1 + a_{22}\,x_2 + \cdots + a_{2n}x_n & \geq & b_2 \\
& \vdots + \vdots + \cdots + \vdots & \geq & \vdots \\
& a_{m1}x_1 + a_{m2}\,x_2 + \cdots + a_{mn}x_n & \geq & b_m \\
& x_1, \quad x_2, \quad \ldots, \quad x_n & \geq & 0
\end{array} \tag{1.3}$$

or

$$\begin{array}{lcccccccc} \text{Maximize} & c_1x_1 & + & c_2\,x_2 & +\cdots+ & c_nx_n & & \\ \text{subject to} & a_{11}x_1 & + & a_{12}\;x_2 & +\cdots+ & a_{1n}x_n & \leq & b_1 \\ & a_{21}x_1 & + & a_{22}\;x_2 & +\cdots+ & a_{2n}x_n & \leq & b_2 \\ & \vdots & + & \vdots & +\cdots+ & \vdots & \leq & \vdots \\ & a_{m1}x_1 & + & a_{m2}\,x_2 & +\cdots+ & a_{mn}x_n & \leq & b_m \\ & x_1, & & x_2, & \ldots, & x_n & \geq & 0. \end{array} \tag{1.4}$$

Any linear programming problem can be converted to the standard form and canonical form by the following transformations [1]:

1. The maximization problem can be converted to the minimization problem as follows:

$$\begin{array}{lcccc} \text{Maximize} & c_1x_1 & +c_2x_2 & +\cdots+ & c_nx_n \\ \text{Minimize} & -c_1x_1 & -c_2x_2 & -\cdots- & c_nx_n. \end{array}$$

2. The inequality sign ($\leq$ or $\geq$) can be changed to the opposite inequality sign ($\geq$ or $\leq$) by multiplying each inequality by -1 such as

$$a_1x_1 + a_2x_2 \geq b \quad \text{is equivalent to} \quad -a_1x_1 - a_2x_2 \;\leq -b$$

$$\text{or}$$

$$a_1x_1 + a_2x_2 \leq b \quad \text{is equivalent to} \quad -a_1x_1 - a_2x_2 \;\geq -b.$$

3. Each equation can be changed to two inequlities as follows:

$$\begin{array}{l} a_1x_1 + a_2x_2 = b \quad \text{is equivalent to} \\ a_1x_1 + a_2x_2 \leq b \quad \text{and} \quad a_1x_1 + \;\; a_2x_2 \;\geq b. \end{array}$$

4. If the variable x_i can be positive or negative (unrestricted in sign), we define $x_i = x_i' - x_i''$ where $x_i', x_i'' \geq 0$. If $x_i'' \geq x_i'$, x_i is negative. If $x_i' \geq x_i''$, x_i is positive.

5. The inequality constraints can be changed into equality constraints by

- If a constraint is in the form $\leq$, it can be changed to an equation by adding a variable which is greater than or equal to zero on the left-hand-side in order to increase the value on the left-hand-side to be equal to the value on the right-hand-side. For example

$$a_1x_1 + a_2x_2 \leq b \quad \text{is changed to} \quad a_1x_1 + a_2x_2 \ + \ x_3 = b,$$

where $x_3 \geq 0$. x_3 is called a **slack variable**.

- If a constraint is in the form $\geq$, it can be changed to an equation by subtracting the value on the left-hand-side by a variable which is greater than or equal to zero in order to decrease the value on the left-hand-side to be equal to the value on the right-hand-side. For example

$$a_1x_1 + a_2x_2 \geq b \quad \text{is changed to} \quad a_1x_1 + a_2x_2 \ - \ x_3 = b,$$

where $x_3 \geq 0$. x_3 is called a **surplus variable**.

1.1.4 The Matrix Notation of a Linear Programming Problem

A linear programming problem can be written in the matrix notation.
Consider

$$\begin{array}{lrcll} \text{Minimize} & \displaystyle\sum_{j=1}^{n} c_jx_j & & & \\ \text{subject to} & \displaystyle\sum_{j=1}^{n} a_{ij}x_j & = & b_i, & i = 1, 2, \ldots, m \\ & x_j & \geq & 0, & j = 1, 2, \ldots, n. \end{array} \tag{1.5}$$

Letting

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \ \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \ \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}, \ \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}.$$

The given linear programming problem can be written as follows:

$$\begin{aligned} \text{Minimize} \quad & \mathbf{c}^{\mathbf{T}}\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{1.6}$$

Moreover, the j^{th} column of $\mathbf{A}$ is denoted by $\mathbf{A}_j$, so $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n]$.

1.2 Linear Programming Modelling

A real-world problem can be formulated or translated into a set of mathematical function, inequalities and equations that represent the objective function and the set of constraints, respectively. Often, data gathering, problem definition, and problem formulation are the most important information.

Example 1.1. A delicatessen company tries to produce two types of products which are types A and B. The details are shown in the Table 1.1. The company makes a contract with a supplier that in each week the supplier will send 15,000 kg of Type 1 material and 20,000 kg of Type 2 material. Any material left by the end of each week must be discarded. The marketing gurantees that they can sell both types of goods at once but the boxing capacity for the type B product is 2,500 boxes/week. What is the maximum profit that the company could profit from both types of product in each week.

Table 1.1: The details about the delicatessen.

		Types of products	
		Type A	Type B
Materials needed to produce one box. (kg.)	Type 1	5	3
	Type 2	4	5
Profit ($)/box		4	4

A thorough understanding of the real-world problem is necessary in order to formulate it correctly. During the formulation stage, an LP problem specialist may discover new insights into the problem that may change the scope of the original problem.

From Example 1.1, let x_1, x_2 be the decision variables which represents the amount of type A and the amount of type B to be produced, respectively.

The linear programming model is defined by:

$$\begin{aligned} \text{Maximize} \quad & 4x_1 + 4x_2 \\ \text{subject to} \quad & 5x_1 + 3x_2 \leq 15000 \\ & 4x_1 + 5x_2 \leq 20000 \\ & \qquad\quad x_2 \leq 2500 \\ & x_1, \quad x_2 \geq 0. \end{aligned} \tag{1.7}$$

The graphical method can be used to solve an LP problem which is introduced briefly below.

1.3 Solution to Linear Programming Problem

The standard definitions involving solution to an LP problem are stated as follows [3]:

Definition 1.2. (Feasible Region).
A **feasible region** is the set of all nonnegative solutions that satisfy all constraints of an LP problem. Given an LP problem in its canonical form (1.3). The feasible region is given by

$$F = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$$

We called $\mathbf{x} \in F$, a **feasible solution**.

Definition 1.3. (Optimal Solution).
Consider an LP problem in the canonical form (1.3). If the feasible region is nonempty, an **optimal solution** is a feasible solution that has the smallest value of the objective function for the minimization problem.

Let $\mathbf{x}^*$ be an optimal solution to the LP problem. Then

$$\mathbf{c}^\mathbf{T}\mathbf{x}^* \leq \mathbf{c}^\mathbf{T}\mathbf{x}, \forall \mathbf{x} \in F.$$

The value of the objective function corresponding to an optimal solution is called the **optimal value**.

Definition 1.4. (Extreme Point).
A point $\mathbf{x}$ in a convex set $\mathbf{X}$ is called an *extreme point* of $\mathbf{X}$, if $\mathbf{x}$ cannot be represented as a strict convex combination of two distinct points in $\mathbf{X}$. In other words, if $\mathbf{x} = \lambda\ \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2$ with $\lambda \in (0,1)$ and $\mathbf{x}_1, \mathbf{x}_2 \in \mathbf{X}$, then $\mathbf{x} = \mathbf{x}_1 = \mathbf{x}_2$.

Definition 1.5. (Infeasible).
A linear program is **infeasible** if it has no feasible solution, i.e. the feasible region is empty.

Definition 1.6. (Unbounded optimal value).
A linear program has an **unbounded optimal value** if the optimal solution is unbounded, i.e. it is either ∞ or $-\infty$. Note that an unbounded feasible region may not yield an unbounded optimal value.

Theorem 1.7. Every linear programming problem must be in one of the following four cases:

1. LP problem has the unique optimal solution.

   This unique optimal solution must be an extreme point.

2. LP problem has alternative optimal solutions.

   If there are two extreme points $\mathbf{x}_1^*$ and $\mathbf{x}_2^*$ being optimal, then any convex combination of $\mathbf{x}_1^*$ and $\mathbf{x}_2^*$ is also optimal.

3. LP problem has an unbounded optimal value.

   For a maximization problem, the feasible region is unbounded and the plane $\mathbf{c}^T\mathbf{x} = z$ can be increased along the unbounded direction of the feasible region. In this case, the objective value is unbounded and no optimal solution exists.

4. LP problem has an empty feasible region.

In this case, the system of equations and/or inequalities defining the feasible region is inconsistent. This means that there is no point satisfying all constraints of the LP problem. Therefore, no optimal solution exists.

Because the linear programming model for Example 1.1 containing only two decision variables ($n = 2$), the **graphical method** can be used to find the optimal solution.

To solve an LP problem by the graphical method, the polyhedron of the feasible solution has to be constructed by all constraints. The optimal solution is the point in the feasible region that achieve the maximum or minimum value of the objective function.

From Example 1.1

$$\begin{aligned} \text{Maximize} \quad & 4x_1 + 4x_2 \\ \text{subject to} \quad & 5x_1 + 3x_2 \leq 15000 \\ & 4x_1 + 5x_2 \leq 20000 \\ & \qquad\quad\; x_2 \leq 2500 \\ & x_1, \quad\; x_2 \geq 0. \end{aligned} \tag{1.8}$$

The two dimensional polyhedron are constructed on x_1x_2-plane. The non-negative constraints indicate that the values of x_1 and x_2 are lying in the 1^{st} quadrant of the plane. Consider the constraint $5x_1 + 3x_2 \leq 15000$, to create the feasible region we have to draw the line $5x_1 + 3x_2 = 15000$. All points lying on the straight line of the equation $5x_1 + 3x_2 = 15000$ and below this line are coresponded to the constraint $5x_1 + 3x_2 \leq 15000$ (see Figure 1.1(a)). Graph of constraints $4x_1 + 5x_2 \leq 20000$ and $x_2 \leq 2500$ can be drawn in the same manner (see Figure 1.1(b)&(c)). From the Figure 1.1(d), the shaded area indicates the **feasible region** and all points in the feasible region are the **feasible solutions**.

Let z be the profit value of the objective function in Example 1.1 i.e. $z = 4x_1+4x_2$. Consider any point in the feasible region that achieves a certain objective value, for example $z = 4000$. In Figure 1.1(d), there are so many points (x_1, x_2) in the feasible region having values greater than 4000 i.e., all the points above

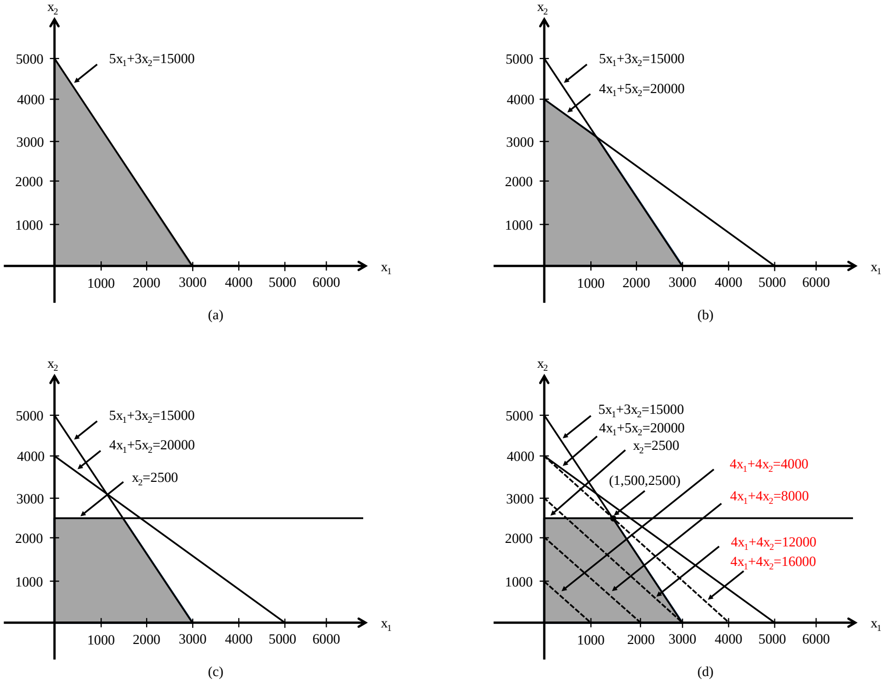

Figure 1.1: Solving LP problem by graphical method.

the line $4x_1 + 4x_2 = 4000$. If the value of z is increased to, for example, $z = 8000$, the graph of equation $4x_1 + 4x_2 = 8000$ will be moved up from and parallel with the line $4x_1 + 4x_2 = 4000$. Thus, the value of z increases. In the figure 1.1(d), the dashed lines are defined as the lines parallel to the objective function. In case of maximization problem, the dashed line will be moved up and parallel in the direction that the objective function value increases (or decreases in case of minimization problem) until the dashed line of the objective function intersect with an endpoint of the feasible region. Hence, this endpoint which is always the corner point of the feasible region is the optimal solution.

In Example 1.1, the optimal solution is the corner point $(x_1, x_2) = (1500, 2500)$ of the feasible region. This point gives the maximum objective value of 16,000. So the delicatessen company should produce 1,500 boxes of Type A and 2,500 boxes of Type B with the maximum profit of \$16,000.

In fact, an optimal solution always occurs at a corner point (or an extreme point) of the feasible region. A corner point of the feasible region is the intersection of two linearly independent contraints. It is easy to find an optimal solution by identifying all corner points of the feasible region for small LP problems and then comparing all the objective values at those points.

Generally, an LP problem contains more than two decision variables and a large number of constraints. So the graphical method is not practical. George B. Dantzig (1947) introduced the simplex method [2, 3], a popular method for solving a linear program. The simplex method is an iterative method to obtain the optimal solution. The details are explained in Chapter 2.

1.4 Motivation and Problem Description

Although the simplex method is a popular method to solve LP problems, in some problems this may not be the best approach to find an optimal solution since there are too many extreme points involved. In 1972, Klee and Minty[4] created collection of LP problems with the worst case running time, i.e., exponential running time, of the simplex method. Nowadays, there are many studies trying to improve the simplex method by reducing the number of the iterations and the solution time. Those techniques include detecting the redundant constraints [5], improving the initial basis [6] and improving a pivot rule [7, 8, 9, 10]. In this thesis, we focus on improving a pivot rule. Pivot rule is an important step of the simplex method for selecting an entering variable. An effective rule can lead to the solution of an LP problem with small number of iterations. Dantzig's original rule is the standard pivot rule, but this rule is efficient only for LP problem with small number of constraints. Moreover Dantzig's rule may take a lot of iterations in some cases. The details of the simplex method and pivot rule are described in Chapter 2 and 3. In this thesis, a new pivot rule is proposed and is called absolute change pivot rule. The concept is to have maximum improvement in objective value by trying to block a leaving variable that make little change in the objective value as much as possible. If such variables can be prevented from leaving the basis, it could make the objective value improved further than using a regular Dantzig's pivot rule and therefore lead to fewer number of iterations.

1.5 The Objective of the Thesis

The objective of this thesis is to design a pivot rule based on the absolute change for the simplex method. The simplex method with this pivot rule is tested with several simulated LP problems and the number of iterations, time, the number of operation, i.e., multiplications and additions are compared with the simplex method using Dantzig's original pivot rule and other rules. Finally, the domain of LP problems that are suitable for this pivot rule is suggested.

1.6 The Scope of the Thesis

In this thesis, if the tableau does not contain zero entries or entries with negative value in the coefficient matrix, this rule is the same as the Dantzig's rule. Therefore, to take advantage of the proposed pivot rule, LP problems with some zero or negative entries that correspond to entering columns in the coefficient matrix will be considered.

1.7 Thesis Overview

This thesis is divided into five chapters. Chapter 1 gives a brief introduction to a linear programming problem, linear programming modelling, solution to linear programming problem, motivation and problem description, the objective of the thesis, the scope of the thesis and thesis overview. Chapter 2 describes the preliminaries of linear programming problem, the simplex method and a pivot rule. Chapter 3 explains the main idea of our pivot rule and applies to the Klee and Minty problem [4]. Chapter 4 deals with computational results by testing and comparing the number of iterations, time, the number of multiplications and additions from this new pivot rule with other pivot rules and the domain of LP problems that are suitable for this pivot rule. The conclusion has been drawn at the end.

CHAPTER II

PRELIMINARIES

In this chapter, the definitions and theorems related to linear programming problems, the simplex method and pivot rules are introduced [1, 2, 3, 11, 12].

In Chapter 1, the graphical method for solving two dimensional LP problems is introduced. George B. Dantzig (1947) [13] presented a method to solve n dimensional LP problems called the **simplex method**, which is described in this chapter.

The discussion of the simplex method is started by introducing the basic feasible solution. Then the simplex method, the simplex method in tableau format, pivot rules and literature reviews about pivot rules are described.

2.1 Basic Feasible Solution

Consider a linear programming (LP) problem in the standard form (4.1), where $\mathbf{A} \in \mathbb{R}^{m \times n}(m < n), \mathbf{b} \in \mathbb{R}^m, \mathbf{c} \in \mathbb{R}^n$ and $rank(\mathbf{A}) = m$:

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

After possibly rearranging the column of $\mathbf{A}$, let $\mathbf{A} = [\mathbf{B}\ \mathbf{N}]$ where $\mathbf{B}$ is an $m \times m$ invertible matrix and $\mathbf{N}$ is $m \times (n-m)$ matrix. Here $\mathbf{B}$ is called the **basic matrix** and $\mathbf{N}$ is called the associated **nonbasic matrix**. Let I_B and I_N be the indices of the variables associated with the columns of matrix $\mathbf{B}$ and $\mathbf{N}$, respectively. Variables associated with the index set I_B are called **basic variables**, denoted by $\mathbf{x_B}$. Variables associated with the index set I_N are called **nonbasic variables**, denoted by $\mathbf{x_N}$. Therefore coefficients in objective function associated with $\mathbf{x_B}$ and $\mathbf{x_N}$ will be denoted by $\mathbf{c_B}$ and $\mathbf{c_N}$, respectively.

Let $\mathbf{x} = [\mathbf{x_B^T}\ \mathbf{x_N^T}]^\mathbf{T}$, $\mathbf{c} = [\mathbf{c_B^T}\ \mathbf{c_N^T}]^\mathbf{T}$, and $\mathbf{A} = [\mathbf{B}\ \mathbf{N}]$. Then the LP problem can be rewritten as follows:

$$\begin{array}{lrrl} \text{Minimize} & \mathbf{c_B^T x_B} & + \mathbf{c_N^T x_N} & \\ \text{subject to} & \mathbf{B x_B} & + \mathbf{N}\ \mathbf{x_N} & = \mathbf{b} \\ & \mathbf{x_B}, & \mathbf{x_N} & \geq \mathbf{0} \end{array} \tag{2.1}$$

The solution $\mathbf{x}$ of the equation $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{x_B} = \mathbf{B}^{-1}\mathbf{b}$ and $\mathbf{x_N} = \mathbf{0}$ is called a **basic solution** of the system. If $\mathbf{x_B} \geq \mathbf{0}$, $\mathbf{x}$ is called a **basic feasible solution** of the system.

Theorem 2.1. If a linear programming problem has a feasible solution, then it has a basic feasible solution.

Theorem 2.2. The set of extreme points of the feasible region corresponds to the set of basic feasible solutions. In other words, extreme points are basic feasible solutions, and vice versa.

Theorem 2.3. If the optimal solution exists (is finite), then the optimal extreme-point solution exists.

2.2 The Simplex Method

The simplex method is an iterative algorithm used to solve LP problems. This method starts with a corner point of the feasible region and moves to the next adjacent corner point that improves the objective value. The simplex method will continue moving in this fashion and stop when the corner point of the feasible region is the optimal solution if one exists.

2.2.1 Algebra of the Simplex Method

Suppose that a basic feasible solution of the system (2.1) is $\begin{bmatrix} \mathbf{x_B} \\ \mathbf{x_N} \end{bmatrix} = \begin{bmatrix} \mathbf{B^{-1}b} \\ \mathbf{0} \end{bmatrix}$. Let z_0 be the objective value of the system (2.1), so z_0 is given by

$$\begin{aligned} z_0 &= \mathbf{c^T x} && (2.2) \\ &= \begin{bmatrix} \mathbf{c_B^T} & \mathbf{c_N^T} \end{bmatrix} \begin{bmatrix} \mathbf{x_B} \\ \mathbf{x_N} \end{bmatrix} && (2.3) \\ &= \begin{bmatrix} \mathbf{c_B^T} & \mathbf{c_N^T} \end{bmatrix} \begin{bmatrix} \mathbf{B^{-1}b} \\ \mathbf{0} \end{bmatrix} && (2.4) \\ &= \mathbf{c_B^T B^{-1} b}. && (2.5) \end{aligned}$$

Now let $\mathbf{x} = \begin{bmatrix} \mathbf{x_B} \\ \mathbf{x_N} \end{bmatrix}$ without setting $\mathbf{x_N} = \mathbf{0}$ denote the set of basic and nonbasic variables for a given basis. Then the feasibility requires that $\mathbf{x_B} \geq \mathbf{0}$ and $\mathbf{x_N} \geq \mathbf{0}$. We denote the j^{th} column of $\mathbf{A}$ by $\mathbf{A}_j$. Then the system $\mathbf{Ax} = \mathbf{b}$ can be rewritten as follows:

$$\begin{aligned} \mathbf{Bx_B} + \mathbf{Nx_N} &= \mathbf{b} && (2.6) \\ \mathbf{x_B} + \mathbf{B^{-1}Nx_N} &= \mathbf{B^{-1}b} && (2.7) \\ \mathbf{x_B} &= \mathbf{B^{-1}b} - \mathbf{B^{-1}Nx_N} && (2.8) \\ &= \mathbf{B}^{-1}\mathbf{b} - \sum_{j \in I_N} \mathbf{B^{-1}A}_j x_j. && (2.9) \end{aligned}$$

Then

$$\mathbf{x_B} = \bar{\mathbf{b}} - \sum_{j \in I_N} (\mathbf{y}_j x_j) \qquad (2.10)$$

where $\bar{\mathbf{b}} = \mathbf{B^{-1}b}$ and $\mathbf{y}_j = \mathbf{B^{-1}A}_j$.

Let z be the objective value, we get

$$\begin{aligned}
z &= \mathbf{c^T x} && (2.11)\\
&= \begin{bmatrix} \mathbf{c_B^T} & \mathbf{c_N^T} \end{bmatrix} \begin{bmatrix} \mathbf{x_B} \\ \\ \mathbf{x_N} \end{bmatrix} && (2.12)\\
&= \mathbf{c_B^T x_B} + \mathbf{c_N^T x_N} && (2.13)\\
&= \mathbf{c_B^T} \left(\mathbf{B^{-1} b} - \sum_{j \in I_N} \mathbf{B^{-1} A}_j x_j \right) + \sum_{j \in I_N} c_j x_j && (2.14)\\
&= \mathbf{c_B^T B^{-1} b} - \mathbf{c_B^T} \sum_{j \in I_N} \mathbf{B^{-1} A}_j x_j + \sum_{j \in I_N} c_j x_j && (2.15)\\
&= \mathbf{c_B^T B^{-1} b} - \sum_{j \in I_N} \left(\mathbf{c_B^T B^{-1} A}_j x_j - c_j x_j \right) && (2.16)\\
&= \mathbf{c_B^T B^{-1} b} - \sum_{j \in I_N} \left(\mathbf{c_B^T B^{-1} A}_j - c_j \right) x_j && (2.17)\\
&= z_0 - \sum_{j \in I_N} (z_j - c_j) x_j && (2.18)
\end{aligned}$$

where $z_j = \mathbf{c_B^T B^{-1} A}_j$ for each nonbasic variable. The negative reduced cost is obtained by $z_j - c_j$. The key result exhibits that the optimal solution is achieved if the index set

$$J = \{j | z_j - c_j > 0, j \in I_N\} \qquad (2.19)$$

is empty. The key result now simply says that if $(z_j - c_j) \leq 0$ for all $j \in I_N$, then the current basic feasible solution is optimal. From the definition we can write $\bar{c}_j = z_j - c_j$ for all j.

Definition 2.4. The subspace that contains only the nonbasic variables is referred to as a **reduced space**. The components of the objective row in a reduced space are called **reduced costs**, donoted be $\bar{\mathbf{c}}$:

$$\bar{\mathbf{c}}^{\mathbf{T}} = (\bar{\mathbf{c}}_{\mathbf{B}}^{\mathbf{T}}, \bar{\mathbf{c}}_{\mathbf{N}}^{\mathbf{T}}) = (\mathbf{0^T}, \mathbf{c_B^T B^{-1} N} - \mathbf{c_N^T}) \qquad (2.20)$$

Note that the cost vector associated with the set of **basic variables** is a **null vector 0**.

The simplex method consists of the following three crucial steps:

1. Initialization step: Find an initial basic feasible solution. Consider a linear program in the following standard form:

$$
\begin{array}{lcccccccccl}
\text{Minimize} & c_1x_1 & + & c_2\,x_2 & +\cdots+ & c_nx_n & & & & & \\
\text{subject to} & a_{11}x_1 & + & a_{12}\,x_2 & +\cdots+ & a_{1n}x_n & +\,x_{n+1} & & & = & b_1 \\
 & a_{21}x_1 & + & a_{22}\,x_2 & +\cdots+ & a_{2n}x_n & & +\,x_{n+2} & & = & b_2 \\
 & \vdots & & \vdots & & \vdots & & & & & \vdots \\
 & a_{m1}x_1 & + & a_{m2}\,x_2 & +\cdots+ & a_{mn}x_n & & & +\,x_{n+m} & = & b_m \\
 & x_1, & & x_2, & \ldots, & x_n, & x_{n+1}, & \ldots, & x_{n+m} & \geq & 0.
\end{array}
\tag{2.21}
$$

The simplex method starts with a corner point of the feasible region. From the LP problem (2.21), the point $(x_1, x_2, \ldots, x_n) = (0, 0, \ldots, 0)$ are the starting basic feasible solution so $x_1, x_2, \ldots, x_n$ is the nonbasic variables and the auxiliary variables $x_{n+1}, x_{n+2}, \ldots, x_{n+m}$ are the basic variables. Since the values 0 are assigned to all nonbasic variables, thus the result can be immediately obtained as $x_{n+1} = b_1, x_{n+2} = b_2, \ldots, x_{n+m} = b_m$ and hence the basic feasible solution is $(x_1, x_2, \ldots, x_n, x_{n+1}, x_{n+2}, \ldots, x_{n+m}) = (0, 0, \ldots, 0, b_1, b_2, \ldots, b_m)$.

2. Iterative step: Find an adjacent basic feasible solution that improves the objective value. The iterative step contains 3 steps that are determining an entering variable (a nonbasic variable to enter the basis), determining the leaving variable (a basic variable to leave the basis) and pivoting on the pivot element. The details will be described in the next section.

3. Optimality test: In this step, the optimality of the current basic feasible solution is tested. In a minimization problem, the optimality condition is satisfied if all reduced costs are nonnegative.

2.2.2 The Simplex Algorithm

Consider the algorithm of the simplex method for solving the linear programming problem of the system (2.21).

Initialization Step : Choose the starting basic feasible solution with the basis $\mathbf{B}$ and the associated nonbasic $\mathbf{N}$.

Main Step :

1. Determine the entering variable from the nonbasic variables: By Dantzig's rule, choose x_k where $z_k - c_k = \max\{z_j - c_j \mid j \in I_N\}$.
2. If $z_k - c_k \leq 0$, then $\begin{bmatrix} \mathbf{x_B} \\ \mathbf{x_N} \end{bmatrix}$ is an optimal solution. Stop the algorithm.
3. Determine the leaving variable from the basic variables by the minimum ratio test.
4. Perform the pivot operation using the entering and the leaving variable, and go to Step 1.

2.3 The Simplex Method in Tableau Format

To make the simplex method easier to handle, the algebra of the simplex method can be transformed into row operations in the tableau format. Suppose that the starting basic feasible solution $\mathbf{x}$ with basis $\mathbf{B}$ is given. The linear programming problem can be represented as follows:

$$\begin{aligned} \text{Minimize} \quad & z \\ \text{subject to} \quad & z - \mathbf{c_B^T x_B} - \mathbf{c_N^T x_N} = 0 && (2.22) \\ & \mathbf{B x_B} + \mathbf{N x_N} \quad = \mathbf{b} && (2.23) \\ & \mathbf{x_B}, \mathbf{x_N} \quad \geq \mathbf{0}. \end{aligned}$$

From (2.23), we have

$$\mathbf{x_B} + \mathbf{B^{-1} N x_N} = \mathbf{B^{-1} b}. \tag{2.24}$$

Multiplying (2.24) by $\mathbf{c_B^T}$, we get

$$\mathbf{c_B^T x_B} + \mathbf{c_B^T B^{-1} N x_N} = \mathbf{c_B^T B^{-1} b}. \tag{2.25}$$

Adding (2.25) to (2.22), we get

$$z + (\mathbf{c_B^T B^{-1} N} - \mathbf{c_N^T})\mathbf{x_N} = \mathbf{c_B^T B^{-1} b}. \tag{2.26}$$

Curently, $\mathbf{x_N} = \mathbf{0}$, and from equation (2.24) and (2.26) we get $\mathbf{x_B} = \mathbf{B^{-1}b}$ and $z = \mathbf{c_B^T B^{-1} b}$. The current basic feasible solution with basis $\mathbf{B}$ can be comveniently represented in the following tableau.

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

From the above tableau format, it can be expanded into the simplex tableau in Table 2.1. The simplex tableau gives the value of the objective function $z = \mathbf{c_B^T B^{-1} b}$, the basic variables $\mathbf{x_B} = \mathbf{B^{-1}b}$ and the objective row $\mathbf{c_B^T B^{-1} N} - \mathbf{c_N^T}$, which consists of the $\bar{c}_j = z_j - c_j$ value for nonbasic variable x_j. Thus, row zero tells us if $z_j - c_j \leq 0$, $\forall j$, the current basis yields an optimal solution. Otherwise, the objective can be improved by increasing the value of a nonbasic variable with $z_j - c_j > 0$. If x_k for some $k \in I_N$ increases, then the vector $\mathbf{y}_k = \mathbf{B}^{\mathbf{-1}}\mathbf{A}_k$, which is stored in the x_k column from rows 1 through m, will determine how much x_k can be increased. If $\mathbf{y}_k \leq 0$, then the problem is unbounded since x_k can be increased indefinitely without being blocked. Conversely, if $\mathbf{y}_k \nleq 0$, that is, if the vector $\mathbf{y}_k$ has at least one positive component, then there exists one of the current basic variables that blocks the increase in x_k by dropping to zero first. The minimum ratio test determines the blocking variable.

2.4 Pivot Rule

The important step for solving a linear programming problem with the simplex method is to select an entering variable for each pivot operation. This method of making such a selection is known as a **pivot rule**. Pivot rule aims to improve the objective value. The best pivot rule would move along the path with the smallest number of visited corner points from the starting solution to the optimal solution.

Table 2.1: The initial simplex tableau before pivoting.

Basic		$\mathbf{x_B}$						$\mathbf{x_N}$				RHS
Variable	z	x_{B_1}	...	x_{B_r}	...	x_{B_m}	...	x_j	...	x_k	...	
z	1	0	...	0	...	0	...	$\bar{c}_j$	...	$\bar{c}_k$	...	$\mathbf{c_B^T B^{-1} b}$
x_{B_1}	0	1	...	0	...	0	...	y_{1j}	...	y_{1k}	...	$\bar{b}_1$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_{B_r}	0	0	...	1	...	0	...	y_{rj}	...	y_{rk}	...	$\bar{b}_r$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_{B_m}	0	0	...	0	...	1	...	y_{mj}	...	y_{mk}	...	$\bar{b}_m$

If an entering variable and a leaving variable are chosen, a new basic solution can be obtained by performing the **pivot operation**.

2.4.1 Pivot Operation

In terms of the geometric motivation of the simplex method, the pivot operation is equivalent to moving from a basic feasible solution to an adjacent basic feasible solution. The pivot operation takes place after a pivot element is selected. The column and the row containing the pivot element are called the pivot column and the pivot row, respectively. A pivot element must be positive. If a nonbasis variable x_k is selected in the column k as a pivot column, the variable corresponding to the pivot column enters the set of basic variable is called the **entering variable**. If a basic variable x_{B_r} in the row r is selected as the pivot row, it is called the **leaving variable**. The steps of the pivot operation are described as follows. First, multiply all elements in the row r by the reciprocal of the pivot element in order to change this element to 1. Then, all entries in the pivot column k except the pivot element have to change to 0 by some row operations. Then the pivot column k becomes one of the columns of the identity matrix. So the entering variable is now a basic variable and the leaving variable now becomes a nonbasic variable.

2.4.2 Entering Variable

An entering variable x_k selected from a nonbasic variable is a variable that corresponds to the pivot column k to enter the set of basic variables since the entering variable will be, in general, increased from 0 to a positive value without exceeding the resource availability on the right-hand-side of the equation. If x_k is increased by one unit, then $x_{B_1}, x_{B_2}, \ldots, x_{B_m}$ will be decreased by $y_{1k}, y_{2k}, \ldots, y_{mk}$ units respectively. (If $y_{ik} < 0$, then x_{B_i} will be increased.)

It is well known that Dantzig's pivot rule [13] is originally the first rule for selecting the entering variable. Dantzig's pivot rule picks a nonbasic variable which provides the most improvement in the objective function to be an entering variable. Thus if $J \neq \varnothing$, the entering variable x_k based on the Dantzig's pivot rule is selected by the most per-unit negative reduced cost as follows:

$$k = \operatorname{argmax}\{z_j - c_j \mid j \in J\}. \tag{2.27}$$

Remark 2.5. For python programming, argmax stands for argument of the maximum, which returns the index of the first maximum element in the set. In a similar fashion, argmin stands for argument of the minimum, which returns the index of the first minimum element in the set.

2.4.3 Leaving Variable

After the pivot column has been chosen, the pivot row will be determined as the result of the feasibility requirement of the solution. Since the entering variable (x_k) will be increased from 0 to a positive value without exceeding the resource availability on the right-hand-side of the equation, the nonbasic variable x_k cannot be indefinitely increased (unless we have an unbounded problem). A basic variable x_{B_r} that first drops to 0 becomes a leaving variable and blocks the further increase of x_k.

A leaving variable corresponds to only one of positive entries in the pivot column k. If all entries in pivot column k is not positive then the problem is unbounded. Otherwise, given a pivot column x_k, the pivot row r is determined by

$$r = \operatorname{argmin} \left\{ \frac{\bar{b}_j}{y_{jk}} \middle| \; y_{jk} > 0, j \in \{1, ..., m\} \right\}$$

This is called the **minimum ratio test**.

To update the simplex tableau 2.1, the pivot operation is performed as follows [2, 11]:

1. Divide row r by y_{rk}. ($y_{rk} > 0$.)

2. For $i = 1, 2, \ldots, m$ and $i \neq r$, update the ith row by adding to it $-y_{ik}$ times the new r^{th} row.

3. Update row zero by adding to its $\bar{c}_k$ times the new r^{th} row.

After the pivot operation is performed, the simplex tableau after pivoting is obtained by following tableau.

Table 2.2: The simplex tableau after pivoting.

Basic		$\mathbf{x_B}$					$\mathbf{x_N}$					RHS
Variable	z	x_{B_1}	...	x_{B_r}	...	x_{B_m}	...	x_j	...	x_k	...	
z	1	0	...	$-\frac{\bar{c}_k}{y_{rk}}$	...	0	...	$\bar{c}_j - \frac{y_{rj}}{y_{rk}}\bar{c}_k$	...	0	...	$\bar{b}_0 - \frac{\bar{b}_r}{y_{rk}}\bar{c}_k$
x_{B_1}	0	1	...	$-\frac{y_{1k}}{y_{rk}}$	...	0	...	$y_{1j} - \frac{y_{rj}}{y_{rk}}y_{1k}$	...	0	...	$\bar{b}_1 - \frac{\bar{b}_r}{y_{rk}}y_{1k}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_k	0	0	...	$\frac{1}{y_{rk}}$	...	0	...	$\frac{y_{rj}}{y_{rk}}$	...	1	...	$\frac{\bar{b}_r}{y_{rk}}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_{B_m}	0	0	...	$-\frac{y_{mk}}{y_{rk}}$	...	1	...	$y_{mj} - \frac{y_{rj}}{y_{rk}}y_{mk}$	...	0	...	$\bar{b}_m - \frac{\bar{b}_r}{y_{rk}}y_{mk}$

2.5 Literature Review on Pivot Rules

An effective pivot rule can lead to the solution of LP with small number of iterations. Dantzig's original rule is the standard pivot rule but this rule is efficient only for LP with a small number of constraints. Moreover Dantzig's rule may take a lot of iterations in some case. Klee and Minty [4] exhibited the worst case running time of the simplex algorithm using Dantzig's pivot rule. To avoid this weakness, there are many studies trying to improve the simplex algorithm via the pivot rule by reducing the number of iterations and the solution time. In 1977, Forrest and Goldfarb [7] presented a pivot rule that reduces the number of iterations which was called "steepest-edge rule". Later, other rules followed such as Devex rule by Harris [8] and the largest-distance pivot rule by Pan [9].

The following are pivot rules used to select an entering index in the simplex algorithm for solving LP problems.

1. **Steepest-edge rule** [7]

   The set of edge directions can be written as:

$$\mathbf{d}_j = \begin{bmatrix} -\mathbf{B}^{-1}\mathbf{N} \\ \mathbf{I} \end{bmatrix} e_{j-m}, \quad j \in I_N, \tag{2.28}$$

   where $\mathbf{I} \in \mathbb{R}^{(n-m)\times(n-m)}$ and e_i is the unit $(n-m)$ vector. Note that $\bar{c}_j = \mathbf{c}^{\mathbf{T}}\mathbf{d}_j$. When J is nonempty, the steepest-edge rule chooses an entering index q such that :

$$\frac{\bar{c}_q}{\|\mathbf{d}_q\|_2} = \min\left\{ \frac{\bar{c}_j}{\|\mathbf{d}_j\|_2} \;\middle|\; j \in J \right\} < 0. \tag{2.29}$$

   If x_q replaces x_p in the basis, then **recurrence formulas** of the edge directions are easily derived:

$$\bar{\mathbf{d}}_p = -\left(\frac{1}{\alpha_q}\right)\mathbf{d}_q, \tag{2.30}$$

$$\bar{\mathbf{d}}_j = \mathbf{d}_j - \left(\frac{\alpha_j}{\alpha_q}\right)\mathbf{d}_q, \; j \in I_N, j \neq p, \tag{2.31}$$

   where $\alpha_q = \mathbf{w}^{\mathbf{T}}\mathbf{A}_q, \mathbf{wB} = \mathbf{c}_{\mathbf{B}}^{\mathbf{T}}$.

2. **Devex rule** [8]

   This rule uses an approximate steepest-edge rule, in which the norms $\|\mathbf{d}_j\|_2$ of the edge directions are replaced by approximated weights w_j. Initially, a so called **"reference framework"** is set to the current set of nonbasic indices, and all weights w_j are set to one for all j in the set. In other iterations, it uses weights w_j to approximate the norms of the subvectors $\hat{\mathbf{d}}_j$ consisting of only those components of the edge direction $\mathbf{d}_j$ associated with the reference framework.

The weights w_j are updated as follows:

$$\bar{w_p} = \max\left\{1, \frac{\|\hat{\mathbf{d}}_q\|_2}{\alpha_q}\right\}, \tag{2.32}$$

$$\bar{w_j} = \max\left\{w_j, \left|\frac{\alpha_j}{\alpha_q}\right| \|\hat{\mathbf{d}}_q\|_2\right\}, \quad j \in I_N, \quad j \neq p, \tag{2.33}$$

where $\alpha_j = \mathbf{e}_p^{\mathbf{T}}\mathbf{B}^{\mathbf{-1}}\mathbf{A}_j$.

3. **Largest-distance pivot rule** [9]

   If J is nonempty, select nonbasic index q to become basic such that:

$$\hat{c}_q := \frac{\bar{c}_q}{\|\mathbf{A}_q\|} = \max\left\{\frac{\bar{c}_j}{\|\mathbf{A}_j\|} \;\middle|\; j \in J\right\}, \tag{2.34}$$

   where $\|\mathbf{A}_j\|$ denotes some norm of $\mathbf{A}_j$.

CHAPTER III
ABSOLUTE CHANGE PIVOT RULE FOR THE SIMPLEX ALGORITHM

Although the simplex algorithm with Dantzig's pivot rule is popular for solving LP problems, in some problems it may not take the best path to the optimal solution. In this chapter the pivot rule that could reduce the number of iterations of LP problem is proposed. The idea is to have the maximum improvement in the objective value by trying to block a leaving variable that makes a little change in the objective value as much as possible. If such variables can be prevented to leave the basis, it could make the objective function value improved further than using a regular Dantzig's pivot rule and therefore lead to fewer number of iterations.

3.1 The Concept of Absolute Change Pivot Rule

To select the entering variable, Dantzig's pivot rule considers only the most negative reduced cost. In some problems, it may not be an effective rule to choose the entering variable. Thus there is a need to develop an efficient pivot rule to identify an entering variable. It is possible that we can select an entering variable that can improve the objective function value as much as possible. Hence we propose a pivot rule to improve the objective value by trying to avoid all leaving variables that cause small change in the objective function. We call it **absolute change pivot rule**.

First, the row with the minimum right-hand-side value will be considered. The motivation behind this is that, given an entering variable, the basic variable associated with this row will have a tendency to become zero first and, as the result, tends to block the increase of the objective value. By preventing this variable from leaving the basis, the value of the entering variable can be increased further. To prevent that, we look for an entering variable that has zero or negative value in that row so that the minimum ratio is not applicable for that row. If there is more than one candidate for such entering variable, we look for the row with the next minimum right-hand-side and repeat the process until we have only one candidate or until we cannot find a row with zero or negative value anymore. If we still end up with more than one entering candidate, we select the one with the most negative reduced cost. In summary, this rule heuristically selects the entering variable that can move the farthest.

3.1.1 Geometric Motivation of the Simplex Method with Absolute Change Pivot Rule

To make it easier to understand the step of an algorithm for linear programming problem, the concept of absolute change pivot rule is introduced in terms of geometry through the following example.

Example 3.1. Consider the following problem:

$$\begin{array}{lrcl} \text{Minimize} & -10x_1 - x_2, & & \\ \text{subject to} & x_1 & \leq & 1, \\ & 20x_1 + x_2 & \leq & 100, \\ & x_1, \quad x_2, & \geq & 0. \end{array}$$

Figure 3.1 shows a feasible region in 2 dimensions of Example 3.1. The simplex method starts at the origin and moves in a feasible direction to an adjacent extreme point along the x_1 axis or x_2 axis. The objective function changes either at the rate of change $\frac{\partial z}{\partial x_1} = -(z_1 - c_1) = \bar{c_1} < 0$ when holding $x_2 = 0$ and increasing x_1 along the x_1 axis or at the rate of change $\frac{\partial z}{\partial x_2} = -(z_2 - c_2) = \bar{c_2} < 0$ when holding $x_1 = 0$ and increasing x_2 along the x_2 axis.

Herein $\bar{c_1} = -10$ and $\bar{c_2} = -1$. Since the most negative reduce cost is $\bar{c_1}$, Dantzig's rule select x_1 as the entering variable, which corresponds to moving along the x_1 axis to visit the position number 1 in Figure 3.1. The pivot rule is

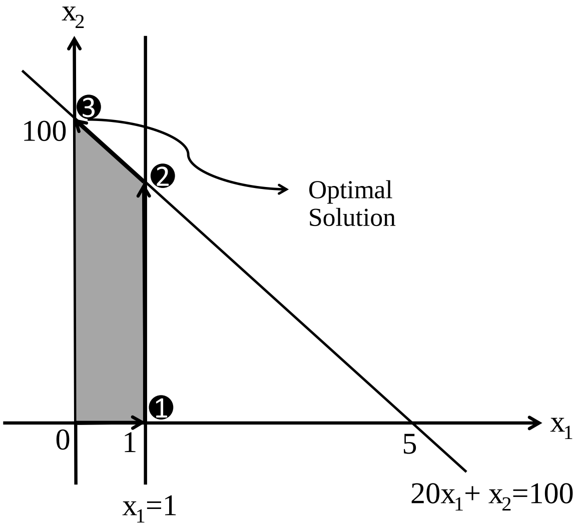

Figure 3.1: The feasible region of Example 3.1.

then repeated, which results in the basic feasible solution moving from the position number 1 to the position number 2 and finally achieving the optimal solution at the position number 3.

As mentioned before, in some problem the simplex method with Dantzig's pivot rule may not give the best path to achieve the optimal solution since this rule may visit a large number of extreme points before reaching the optimal solution. The simplex method with the absolute change pivot rule does not consider moving along the direction with the most rate of change. This rule tries to prevent basic variable with small value from leaving the basis. For Example 3.1, this corresponds to the slack variable in the first constraint. Hence, the slack variable for the second constraint leaves the basis in stead. Thus this rule causes the algorithm to move from the origin to the position number 1 as depicted in Figure 3.2 along x_2 axis and achieve the optimal solution in one step.

3.1.2 The Simplex Algorithm with the Absolute Change Pivot Rule

The simplex algorithm with the absolute change pivot rule starts from a basic feasible solution with the basis $[\mathbf{B}\ \mathbf{N}]$.

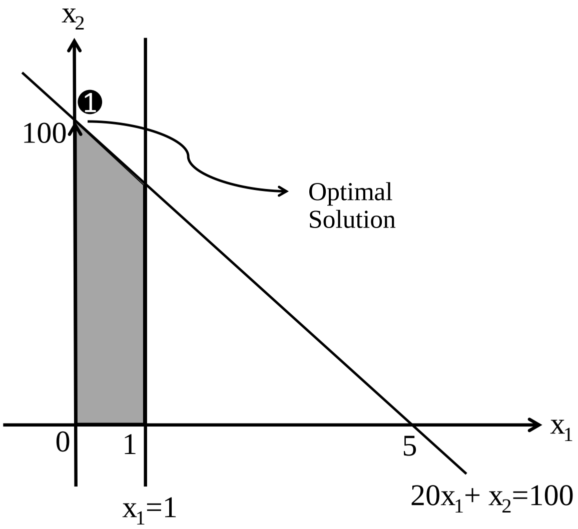

Figure 3.2: The feasible region of Example 3.1.

1. If $z_j - c_j \leq 0$ for all $j \in I_N$, then $\begin{bmatrix} \mathbf{x_B} \\ \mathbf{x_N} \end{bmatrix}$ is the optimal solution. Stop the algorithm.

2. Determine the entering variable by using absolute change pivot rule:

   i. Set $CI = \{1, \ldots, m\}$. Let $J = \{j \mid z_j - c_j > 0, j \in I_N\}$

   ii. Select index r such that $r = \text{argmin}\{\bar{b}_i \mid i \in CI\}$

   iii. Let $\hat{J} = \{j \in J \mid y_{rj} \leq 0\}$.
   If $\hat{J} \neq \varnothing$, let $J = \hat{J}$.

   - If $|J| = 1$, go to (iv). Else, remove r from CI and go to (ii).

   iv. Else, choose x_k by $z_k - c_k = \max\{z_j - c_j \mid j \in J\}$

3. Determine the leaving variable from the basic variables by the minimum ratio test.

4. Perform the pivot operation using the entering and the leaving variables, and go to Step 1.

This algorithm is different from the simplex algorithm with Dantzig's pivot rule at step 2, which is the step where an entering variable is determined. This step can also be described in the following flowchart.

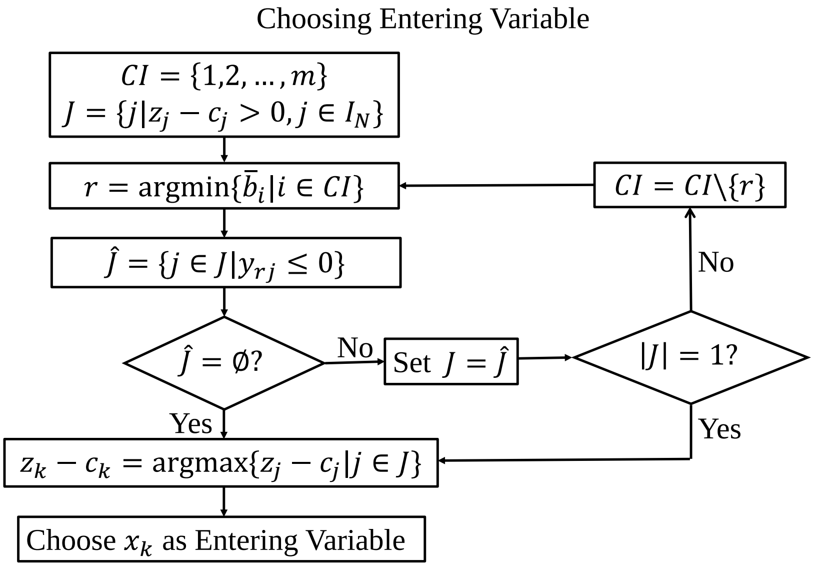
Choosing Entering Variable
$CI = \{1,2,\dots,m\}$
$J = \{j|z_j - c_j > 0, j \in I_N\}$
$r = \text{argmin}\{\bar{b}_i|i \in CI\}$
$CI = CI\setminus\{r\}$
$\hat{J} = \{j \in J|y_{rj} \leq 0\}$
No
$\hat{J} = \emptyset?$
No
Set $J = \hat{J}$
$|J| = 1?$
Yes
Yes
$z_k - c_k = \text{argmax}\{z_j - c_j|j \in J\}$
Choose x_k as Entering Variable

Figure 3.3: Flowchart of the step 2 for identifying an entering variable.

3.2 Illustration of the Method

The proposed pivot rule is demonstrated with two examples: Klee and Minty [4] problem and a randomly generated linear programming problem.

In 1972, Klee and Minty showed a collection of LP problems that cause the worst-case running time for the simplex method using Dantzig's pivot rule. The collection of LP problems is given by

$$\begin{aligned} &\text{Minimize} \quad -\sum_{j=1}^{n} 10^{n-j} x_i, \\ &\text{subject to} \quad 2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \quad \leq \; 100^{i-1}, \quad i = 1,\dots,n, \\ &\qquad\qquad\quad x_i \geq 0, \quad i = 1,\dots,n. \end{aligned} \tag{3.1}$$

The simplex method with the Dantzig's pivot rule requires $2^n - 1$ iterations to solve Klee and Minty problem. For example, if we let $n = 4$, the simplex method using Dantzig's pivot rule visits 15 corner points before reaching the optimal solution.

The following examples (3.2 and 3.4) are presented to show the efficiency of the proposed pivot rule. To illustrate the proposed pivot rule geometrically, Klee and Minty problem with $n = 3$ is used.

Example 3.2. Consider the following problem:

$$\begin{array}{lrcl}
\text{Minimize} & -100x_1 - 10x_2 - x_3, & & \\
\text{subject to} & x_1 \qquad\qquad & \leq & 1, \\
& 20x_1 + \quad x_2 \qquad & \leq & 100, \\
& 200x_1 + 20x_2 + x_3 & \leq & 10000, \\
& x_1, \quad x_2, \ x_3 & \geq & 0.
\end{array}$$

Figure 3.4 shows the feasible region of Klee and Minty problem with $n = 3$. If we follow the simplex method with Dantzig's pivot rule, this rule moves from the origin to the position numbers 1,2,...,6, consecutively and achieves the optimal solution at the position number 7, with the total of seven iterations. We can see in Figure 3.5 that the simplex method with absolute change pivot rule moves from the origin to the position number 1, with only one iteration.

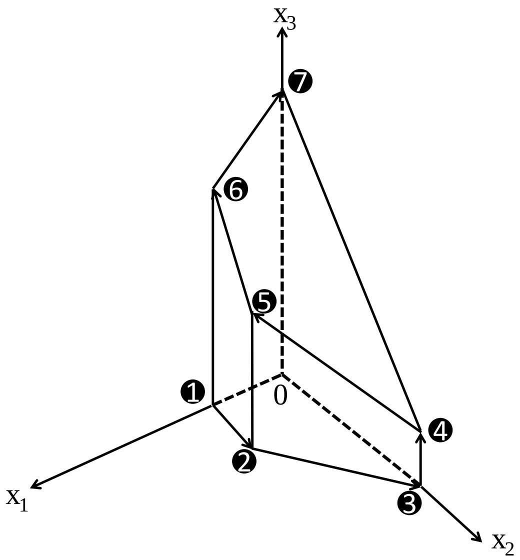

Figure 3.4: The feasible region of Klee and Minty problem with $n = 3$.

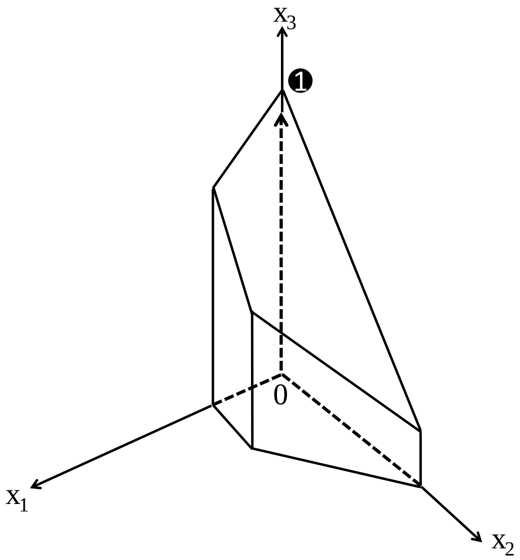

Figure 3.5: The feasible region of Klee and Minty problem with $n = 3$.

Example 3.2 can be written in the tableau format, where x_4, x_5, x_6 are the slack variables, as follows:

	z	x_1	x_2	x_3	x_4	x_5	x_6	RHS
z	1	100	10	1	0	0	0	0
x_4	0	1	0	0	1	0	0	1
x_5	0	20	1	0	0	1	0	100
x_6	0	200	20	1	0	0	1	10000

Since Example 3.2 is the minimization problem, we can see from the above tableau that x_1, x_2 and x_3 can be a candidate for the entering variable since the negative reduce cost is greater than zero. If we follow the simplex algorithm with absolute change pivot rule we can see that right-hand-side entries are already sorted from the smallest to the largest values. Consider the first row (x_4), since the element in the third and fourth column are zero then x_2 and x_3 can be entering variables. x_1 is not a candidate because the element in the column of x_1 is positive. Since we still have two candidates, the second row has to be considered. The second row has zero value at the third column so x_3 is a candidate while x_2 is no longer a candidate since its entry is positive. As the result of this pivot rule, the entring variable is x_3. From the minimum ratio test we get x_6 is the leaving variable. After pivot operation, the optimal solution is obtained which is shown in the tableau below.

	z	x_1	x_2	x_3	x_4	x_5	x_6	RHS
z	1	-100	-10	0	0	0	-1	-10000
x_4	0	1	0	0	1	0	0	1
x_5	0	20	1	0	0	1	0	100
x_3	0	200	20	1	0	0	1	10000

For this example the optimal solution is $x_1 = 0$, $x_2 = 0$ and $x_3 = 10000$. The optimal value is -10000 within one iteration.

Theorem 3.3. The simplex method using absolute change pivot rule achieves the optimal solution within one iteration for any Klee and Minty problem.

Proof. Consider a Klee and Minty problem in the following standard form:

$$\begin{array}{llllllll}
\text{Minimize} & -10^{n-1}x_1 & - \quad 10^{n-2}\, x_2 & -\cdots- \; x_n, & & & & \\
\text{subject to} & x_1 & & & + \; x_{n+1} & & & = \; 1, \\
 & 20x_1 & + \quad x_2 & & & + \; x_{n+2} & & = \; 100, \\
 & \vdots & \quad \vdots & & & & & \vdots \\
 & 2\times 10^{n-1}x_1 & + \; 2\times 10^{n-2}\, x_2 & +\cdots+ \; x_n & & & + \; x_{2n} & = \; 10^{n-1}, \\
 & x_1, & \quad x_2, & \ldots, x_n, & x_{n+1}, & \ldots, & x_{2n} & \geq \; 0,
\end{array} \tag{3.2}$$

where $x_{n+1}, x_{n+2}, \ldots, x_{2n}$ are the slack variables associated with the n^{th} constraints. The initial simplex tableau of problem (3.2) is:

	z	x_1	x_2	$\cdots$	x_n	x_{n+1}	$\cdots$	x_{2n}	RHS
z	1	10^{n-1}	10^{n-2}	$\cdots$	1	0	$\cdots$	0	0
x_{n+1}	0	1	0	$\cdots$	0	1	$\cdots$	0	1
x_{n+2}	0	20	1	$\cdots$	0	0	$\cdots$	0	100
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_{2n}	0	$2 \times 10^{n-1}$	$2 \times 10^{n-2}$	$\cdots$	1	0	$\cdots$	1	10^{n-1}

The right-hand-side entries are sorted from the smallest to the largest values. By the simplex method using the absolute change pivot rule, x_n is chosen to be an entering variable and x_{2n} is a leaving variable by the minimum ratio test. After the first iteration, the optimal solution is obtained which is shown in the tableau below.

	z	x_1	x_2	$\cdots$	x_n	x_{n+1}	$\cdots$	x_{2n}	RHS
z	1	-10^{n-1}	-10^{n-2}	$\cdots$	0	0	$\cdots$	-1	-10^{n-1}
x_{n+1}	0	1	0	$\cdots$	0	1	$\cdots$	0	1
x_{n+2}	0	20	1	$\cdots$	0	0	$\cdots$	0	100
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_n	0	$2 \times 10^{n-1}$	$2 \times 10^{n-2}$	$\cdots$	1	0	$\cdots$	1	10^{n-1}

Tableau above is the optimal tableau for Klee and Minty problem with any size n. The optimal solution is $x_1 = 0, x_2 = 0, \ldots, x_{n-1} = 0$ and $x_n = 10^{n-1}$. The optimal objective value is -10^{n-1} and is achieved within one iteration. □

Example 3.4. Consider the following generated linear programming problem :

$$\begin{array}{rrcl} \text{Minimize} & -50x_1- \ 2x_2-46x_3-40x_4-15x_5, & & \\ \text{subject to} & 15x_1- \ 3x_2+22x_3+ \ 3x_4- \ 4x_5 & \leq & 1467, \\ & 17x_1+11x_2+23x_3+19x_4-28x_5 & \leq & 1733, \\ & 10x_1-18x_2+21x_3+28x_4+ \ 6x_5 & \leq & 1758, \\ & -49x_1+ \ 6x_2+36x_3+34x_4- \ 2x_5 & \leq & 606, \\ & -33x_1+25x_2+48x_3-14x_4+12x_5 & \leq & 1365, \\ & x_1, \quad x_2, \quad x_3, \quad x_4, \quad x_5 & \geq & 0. \end{array}$$

Let $x_6, x_7, x_8, x_9, x_{10}$ be slack variables associated with constraints. The initial simplex tableau is :

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	50	2	46	40	15	0	0	0	0	0	0
x_6	0	15	-3	22	3	-4	1	0	0	0	0	1467
x_7	0	17	11	23	19	-28	0	1	0	0	0	1733
x_8	0	10	-18	21	28	6	0	0	1	0	0	1758
x_9	0	-49	6	36	34	-2	0	0	0	1	0	606
x_{10}	0	-33	25	48	-14	12	0	0	0	0	1	1365

From the above initial simplex tableau, we can see that x_1, x_2, x_3, x_4 and x_5 can be a candidate for the entering variable since the negative reduce cost is greater than zero. If we follow the simplex algorithm with the absolute change pivot rule, it looks for the row with the minimum right-hand-side first. Since row x_9 has the minimum right-hand-side, we look for zero or negative value in that row, which are the entries in the column x_1 and x_5. So x_1 and x_5 are the candidates for the entering variable. Because there is more than one candidate to be an entering variable, we have to look for the row with the next minimum right-hand-side, which is row x_{10}. Since the zero or negative entries in this row are in the columns of x_1 and x_4, the only candidate left is x_1. Hence, after the first iteration we get x_1

is the entering variable and, with minimum ratio test, x_6 is the leaving variable. After pivoting, the simplex tableau becomes

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	0	12	$-27\frac{1}{3}$	30	$28\frac{1}{3}$	$-3\frac{1}{3}$	0	0	0	0	-4890
x_1	0	1	$-\frac{1}{5}$	$1\frac{7}{15}$	$\frac{1}{5}$	$-\frac{4}{15}$	$\frac{1}{15}$	0	0	0	0	$97\frac{4}{5}$
x_7	0	0	$14\frac{2}{5}$	$-1\frac{14}{15}$	$15\frac{3}{5}$	$-23\frac{7}{15}$	$-1\frac{2}{15}$	1	0	0	0	$70\frac{2}{5}$
x_8	0	0	-16	$6\frac{1}{3}$	26	$8\frac{2}{3}$	$-\frac{2}{3}$	0	1	0	0	780
x_9	0	0	$-3\frac{4}{5}$	$107\frac{13}{15}$	$45\frac{4}{5}$	$-15\frac{1}{15}$	$3\frac{4}{15}$	0	0	1	0	$5398\frac{1}{5}$
x_{10}	0	0	$18\frac{2}{5}$	$96\frac{2}{5}$	$-7\frac{2}{5}$	$3\frac{1}{5}$	$2\frac{1}{5}$	0	0	0	1	$4589\frac{2}{5}$

In a similar fashion, after the second iteration, x_5 is the entering variable and x_8 is the leaving variable. The result simplex tableau becomes

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	0	$64\frac{4}{13}$	$-48\frac{1}{26}$	-55	0	$-1\frac{2}{13}$	0	$-3\frac{7}{26}$	0	0	-7440
x_1	0	1	$-\frac{9}{13}$	$1\frac{43}{65}$	1	0	$\frac{3}{65}$	0	$\frac{2}{65}$	0	0	$121\frac{4}{5}$
x_7	0	0	$-28\frac{12}{13}$	$15\frac{14}{65}$	86	0	$-2\frac{61}{65}$	1	$2\frac{46}{65}$	0	0	$2182\frac{2}{5}$
x_5	0	0	$-1\frac{11}{13}$	$\frac{19}{26}$	3	1	$-\frac{1}{13}$	0	$\frac{3}{26}$	0	0	90
x_9	0	0	$-31\frac{8}{13}$	$118\frac{57}{65}$	89	0	$2\frac{7}{65}$	0	$1\frac{48}{65}$	1	0	$6754\frac{1}{5}$
x_{10}	0	0	$24\frac{4}{13}$	$94\frac{4}{65}$	-17	0	$2\frac{29}{65}$	0	$\frac{24}{65}$	0	1	$4301\frac{2}{5}$

In the last iteration of this example, x_2 is the entering variable and x_{10} is the leaving variable. The last simplex tableau is

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	0	0	$-296\frac{23}{26}$	$-10\frac{2}{79}$	0	$-7\frac{5}{8}$	0	$-2\frac{19}{65}$	0	$-2\frac{51}{79}$	$-18819\frac{32}{49}$
x_1	0	1	0	$4\frac{16}{47}$	$\frac{49}{95}$	0	$\frac{11}{95}$	0	$\frac{2}{99}$	0	$\frac{1}{35}$	$244\frac{4}{13}$
x_7	0	0	0	$127\frac{3}{22}$	$-65\frac{61}{79}$	0	$-\frac{1}{36}$	1	$2\frac{11}{41}$	0	$1\frac{15}{79}$	$7300\frac{12}{23}$
x_5	0	0	0	$7\frac{7}{8}$	$1\frac{56}{79}$	1	$\frac{5}{46}$	0	$\frac{2}{23}$	0	$\frac{6}{79}$	$416\frac{42}{61}$
x_9	0	0	0	$241\frac{21}{97}$	$66\frac{8}{9}$	0	$5\frac{24}{83}$	0	$1\frac{8}{31}$	1	$1\frac{3}{10}$	$12348\frac{49}{66}$
x_2	0	0	1	$3\frac{20}{23}$	$-\frac{7}{10}$	0	$\frac{1}{10}$	0	$-\frac{1}{66}$	0	$\frac{3}{73}$	$176\frac{22}{23}$

This simplex tableau is the optimal tableau with the optimal solution $x_1 = 244\frac{4}{13}$, $x_2 = 7300\frac{12}{23}$, $x_3 = 0$, $x_4 = 0$ and $x_5 = 416\frac{42}{61}$. The optimal value is $-18819\frac{32}{49}$ and the number of iterations is three while the simplex method with Dantzig's pivot rule uses five iterations to achieve the optimal solution as follows:

Consider the initial simplex tableau of the Example 3.4:

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	50	2	46	40	15	0	0	0	0	0	0
x_6	0	15	-3	22	3	-4	1	0	0	0	0	1467
x_7	0	17	11	23	19	-28	0	1	0	0	0	1733
x_8	0	10	-18	21	28	6	0	0	1	0	0	1758
x_9	0	-49	6	36	34	-2	0	0	0	1	0	606
x_{10}	0	-33	25	48	-14	12	0	0	0	0	1	1365

From the initial simplex tableau above if we follow the simplex algorithm with Dantzig's pivot rule, after the first iteration we get x_1 as the entering variable and x_6 as the leaving variable. After pivoting, the simplex tableau becomes

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	0	12	$-27\frac{1}{3}$	30	$28\frac{1}{3}$	$-3\frac{1}{3}$	0	0	0	0	-4890
x_1	0	1	$-\frac{1}{5}$	$1\frac{7}{15}$	$\frac{1}{5}$	$-\frac{4}{15}$	$\frac{1}{15}$	0	0	0	0	$97\frac{4}{5}$
x_7	0	0	$14\frac{2}{5}$	$-1\frac{14}{15}$	$15\frac{3}{5}$	$-23\frac{7}{15}$	$-1\frac{2}{15}$	1	0	0	0	$70\frac{2}{5}$
x_8	0	0	-16	$6\frac{1}{3}$	26	$8\frac{2}{3}$	$-\frac{2}{3}$	0	1	0	0	780
x_9	0	0	$-3\frac{4}{5}$	$107\frac{13}{15}$	$45\frac{4}{5}$	$-15\frac{1}{15}$	$3\frac{4}{15}$	0	0	1	0	$5398\frac{1}{5}$
x_{10}	0	0	$18\frac{2}{5}$	$96\frac{2}{5}$	$-7\frac{2}{5}$	$3\frac{1}{5}$	$2\frac{1}{5}$	0	0	0	1	$4589\frac{2}{5}$

After the second iteration, x_4 is the entering variable and x_7 is the leaving variable. The result simplex tableau becomes

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	0	$-15\frac{9}{13}$	$-23\frac{8}{13}$	0	$73\frac{6}{13}$	$-1\frac{2}{13}$	$-1\frac{12}{13}$	0	0	0	$-5025\frac{5}{13}$
x_1	0	1	$-\frac{5}{13}$	$1\frac{29}{59}$	0	$\frac{1}{29}$	$\frac{3}{37}$	$-\frac{1}{78}$	0	0	0	$96\frac{35}{39}$
x_4	0	0	$\frac{12}{13}$	$-\frac{1}{8}$	1	$-1\frac{1}{2}$	$-\frac{4}{55}$	$\frac{5}{78}$	0	0	0	$4\frac{20}{39}$
x_8	0	0	-40	$9\frac{5}{9}$	0	$47\frac{7}{9}$	$1\frac{2}{9}$	$-1\frac{2}{3}$	1	0	0	$662\frac{2}{3}$
x_9	0	0	$-44\frac{3}{13}$	$113\frac{23}{78}$	0	$50\frac{32}{39}$	$6\frac{35}{78}$	$-2\frac{21}{26}$	0	1	0	$5200\frac{7}{13}$
x_{10}	0	0	$25\frac{3}{13}$	$95\frac{14}{29}$	0	$-7\frac{41}{44}$	$1\frac{51}{77}$	$\frac{37}{78}$	0	0	1	$4622\frac{31}{39}$

In the third iteration, x_5 is the entering variable and x_8 is the leaving variable. The simplex tableau becomes

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	0	$45\frac{47}{58}$	$-38\frac{4}{13}$	0	0	$-3\frac{1}{30}$	$\frac{55}{86}$	$-1\frac{43}{80}$	0	0	$-6044\frac{11}{26}$
x_1	0	1	$-\frac{21}{59}$	$1\frac{47}{97}$	0	0	$\frac{2}{25}$	$-\frac{1}{86}$	0	0	0	$96\frac{11}{26}$
x_4	0	0	$-\frac{1}{3}$	$\frac{3}{17}$	1	0	$-\frac{3}{88}$	$\frac{1}{86}$	$\frac{1}{32}$	0	0	$25\frac{26}{69}$
x_5	0	0	$-\frac{36}{43}$	$\frac{1}{5}$	0	1	$\frac{1}{39}$	$-\frac{3}{86}$	$\frac{1}{48}$	0	0	$13\frac{20}{23}$
x_9	0	0	$-1\frac{41}{60}$	$103\frac{3}{23}$	0	0	$5\frac{11}{74}$	$-1\frac{3}{86}$	$-1\frac{3}{47}$	1	0	$4495\frac{2}{3}$
x_{10}	0	0	$18\frac{49}{83}$	$97\frac{2}{29}$	0	0	$1\frac{45}{52}$	$\frac{17}{86}$	$\frac{1}{6}$	0	1	$4732\frac{70}{87}$

Next iteration, x_7 is the entering variable and x_4 is the leaving variable. The simplex tableau is

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	0	$64\frac{4}{13}$	$-48\frac{1}{26}$	-55	0	$-1\frac{2}{13}$	0	$-3\frac{7}{26}$	0	0	-7440
x_1	0	1	$-\frac{9}{13}$	$1\frac{43}{65}$	1	0	$\frac{3}{65}$	0	$\frac{2}{65}$	0	0	$121\frac{4}{5}$
x_7	0	0	$-28\frac{12}{13}$	$15\frac{14}{65}$	86	0	$-2\frac{61}{65}$	1	$2\frac{46}{65}$	0	0	$2182\frac{2}{5}$
x_5	0	0	$-1\frac{11}{13}$	$\frac{19}{26}$	3	1	$-\frac{1}{13}$	0	$\frac{3}{26}$	0	0	90
x_9	0	0	$-31\frac{8}{13}$	$118\frac{57}{65}$	89	0	$2\frac{7}{65}$	0	$1\frac{48}{65}$	1	0	$6754\frac{1}{5}$
x_{10}	0	0	$24\frac{4}{13}$	$94\frac{4}{65}$	-17	0	$2\frac{29}{65}$	0	$-\frac{24}{65}$	0	1	$4301\frac{2}{5}$

In the last iteration, x_2 is the entering variable and x_{10} is the leaving variable. The simplex tableau is

	z	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	RHS
z	1	0	0	$-296\frac{23}{26}$	$-10\frac{2}{79}$	0	$-7\frac{5}{8}$	0	$-2\frac{19}{65}$	0	$-2\frac{51}{79}$	$-18819\frac{32}{49}$
x_1	0	1	0	$4\frac{16}{47}$	$\frac{49}{95}$	0	$\frac{11}{95}$	0	$\frac{2}{99}$	0	$\frac{1}{35}$	$244\frac{4}{13}$
x_7	0	0	0	$127\frac{3}{22}$	$65\frac{61}{79}$	0	$-\frac{1}{36}$	1	$2\frac{11}{41}$	0	$1\frac{15}{79}$	$7300\frac{12}{23}$
x_5	0	0	0	$7\frac{7}{8}$	$1\frac{56}{79}$	1	$\frac{5}{46}$	0	$\frac{2}{23}$	0	$\frac{6}{79}$	$416\frac{42}{61}$
x_9	0	0	0	$241\frac{21}{97}$	$66\frac{8}{9}$	0	$5\frac{24}{83}$	0	$1\frac{8}{31}$	1	$1\frac{3}{10}$	$12348\frac{49}{66}$
x_2	0	0	1	$3\frac{20}{23}$	$-\frac{7}{10}$	0	$\frac{1}{10}$	0	$-\frac{1}{66}$	0	$\frac{3}{73}$	$176\frac{22}{23}$

This simplex tableau is the optimal tableau with optimal solution $x_1 = 244\frac{4}{13}$, $x_2 = 176\frac{22}{23}$, $x_3 = 0$, $x_4 = 0$ and $x_5 = 416\frac{42}{61}$. The optimal value is $-18819\frac{32}{49}$ and the number of iterations is five.

In this chapter the concept of the absolute change pivot rule is proposed. Then we describe geometric motivation of the simplex method with absolute change pivot rule, the simplex algorithm with the absolute change pivot rule and illustrate the performance of the simplex method with this pivot rule by using Klee and Minty problem with $n = 3$ and a randomly generated linear programming problem. In the next chapter the effectiveness of the simplex method with this pivot rule is tested by solving the randomly generated LP problems and is compared the number of iterations and time with other pivot rules.

CHAPTER IV
EXPERIMENTS AND ANALYSIS

In this chapter, the absolute change pivot rule (ACP) was tested with randomly generated linear programming problems of various sizes. the number of iterations, time, the number of operations, i.e., multiplications and additions of this pivot rule are compared with Dantzig's pivot rule (DZP), steepest-edge pivot rule (SEP), Devex rules (DVP) and the largest-distance pivot rule (LDP).

The programming language used was Python. All codes were run under an Oracle VM VirtualBox version 4.3.4r91027 by software Sage [14] version 5.7 with base memory 512 MB. The computer system processor is Intel(R) Core(TM)i7-3770K CPU @3.50GHz, 8.00 GB of memory, and 64-bit Window 7 Operating System. The time used in the experiments is measured in seconds.

4.1 Problem Generation

All randomly generated linear programming problems are minimization problems and are generated according to the following specifications: The cost vector $\mathbf{c}$ is generated with $c_i \in [-10, 10]$. The matrix $\mathbf{A}$ is generated with $a_{ij} \in [-10, 10]$. To guarantee a feasible problem, a feasible solution $\mathbf{x}$ is generated with $x_i \in [0, 10]$ and then the right-hand-side $\mathbf{b}$ is calculated by $\mathbf{b} = \mathbf{Ax}$. All constraints are in the form $\leq$.

In this chapter, m and n represent the number of constraints and the number of decision variables, respectively. The LP problems are in the following form:

$$\begin{aligned} \text{Minimize} \quad & \mathbf{c^T x} \\ \text{subject to} \quad & \mathbf{Ax} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{4.1}$$

The size of matrix $\mathbf{A}$, $m \times n$, is used to indicate the size of the problem. Note that these m and n are defined differently from m and n in the Section 2.1 since the problems are in different forms.

The sizes of problems are varied from 10×10 through 650×650. For each size, we generate 50 problems and find the mean results for each method.

4.2 Comparison

The absolute change pivot rule is compared with Dantzig's original pivot rule and other pivot rules. The performance measures used for comparison is the number of iterations (pivot) and time. Moreover, the comparison of the operations, i.e., multiplications and additions, are shown in the next section.

Table 4.1 and 4.2 show the comparison of the average number of iterations and average time from solving LP problems by the simplex algorithm with DZP, LDP, SEP, DVP and ACP with problem sizes from 10×10 to 120×120. Moreover, these tables also show standard deviation for average number of iterations and average time for each problem size.

Table 4.1: The average number of iterations ± standard deviation ($\mu \pm \sigma$) from solving LP problems by the simplex method with DZP, LDP, SEP, DVP and ACP with problem sizes from 10×10 to 120×120.

No.	Problem size		Average no. of iterations ± Standard deviation ($\mu \pm \sigma$)				
	m	n	DZP	LDP	SEP	DVP	ACP
1	10	10	7.33±3.42	6.23±2.47	10.86±4.83	10.16±4.19	7.58±3.58
2	20	20	17.94±5.45	16.60±4.68	31.58±8.63	32.48±8.16	20.06±6.38
3	30	30	37.74±9.09	30.79±6.80	73.11±18.80	65.49±13.98	38.57±9.68
4	40	40	51.90±15.61	44.51±11.31	117.69±29.45	103.37±24.67	53.96±12.89
5	50	50	79.98±22.02	65.60±13.73	167.88±28.83	153.20±29.15	78.38±12.29
6	60	60	113.14±24.12	86.52±15.11	225.04±40.07	206.76±29.89	102.50±16.96
7	70	70	141.46±35.94	109.30±14.54	298.68±44.95	267.64±45.44	131.38±21.48
8	80	80	184.66±40.82	136.82±16.74	385.82±56.01	337.94±49.91	161.02±18.32
9	90	90	232.72±41.30	167.56±21.60	477.92±58.55	413.12±62.61	196.20±25.17
10	100	100	280.30±51.98	199.28±25.46	539.82±81.95	483.64±60.95	223.84±31.31
11	110	110	331.56±63.00	231.78±34.65	641.80±83.49	565.68±62.07	258.96±34.50
12	120	120	409.94±75.21	277.44±34.13	739.56±74.55	662.24±67.32	292.58±35.45

Remark 4.1. No unbounded problem in Table 4.1.

Table 4.2: The average time ± standard deviation ($\mu \pm \sigma$) from solving LP problems by the simplex method with DZP, LDP, SEP, DVP and ACP with problem sizes from 10 × 10 to 120 × 120.

No.	Problem size		Average time (sec) ± Standard deviation				
	m	n	DZP	LDP	SEP	DVP	ACP
1	10	10	0.00±0.01	0.01±0.01	0.06±0.06	0.03±0.02	0.01±0.01
2	20	20	0.01±0.01	0.03±0.01	0.41±0.11	0.29±0.08	0.04±0.01
3	30	30	0.04±0.01	0.09±0.03	1.90±0.47	1.16±0.24	0.12±0.04
4	40	40	0.08±0.02	0.16±0.04	5.32±1.16	3.18±0.63	0.24±0.06
5	50	50	0.14±0.04	0.29±0.06	11.14±1.93	7.07±1.52	0.43±0.07
6	60	60	0.23±0.06	0.46±0.10	21.11±3.82	13.38±1.99	0.69±0.12
7	70	70	0.34±0.10	0.67±0.10	37.92±5.88	23.33±3.98	1.05±0.17
8	80	80	0.48±0.14	0.90±0.13	59.97±9.70	35.91±6.00	1.40±0.19
9	90	90	0.62±0.12	1.15±0.15	85.97±10.61	50.61±7.70	1.81±0.24
10	100	100	0.86±0.18	1.58±0.21	123.62±18.98	75.78± 9.79	2.43±0.36
11	110	110	1.09±0.21	2.02±0.31	174.50±23.55	105.59±12.04	3.14±0.46
12	120	120	1.44±0.26	2.55±0.30	231.61±23.34	142.64±14.62	3.81±0.47

Since DVP and SEP give a large number of iterations (see Table 4.1), they take a lot of solution time (see Table 4.2) for solving LP problems for each problem sizes. Hence, we reduce our comparison to only DZP, LDP and ACP for larger LP problems.

Table 4.3 and 4.4 show the comparison of the average number of iterations and average time with their standard deviations from solving LP problems by the simplex algorithm with DZP, LDP and ACP with problem size from 150 × 150 to 650 × 650. While generating the problems (where A is a square matrix), we encountered a few unbounded problems, which are discarded when found and then replaced by a new randomly generated bounded LP problem . So there is no unbounded problem in both tables.

Table 4.3: The average number of iterations ± standard deviation ($\mu \pm \sigma$) from solving LP problems by the simplex method with DZP, LDP and ACP with problem sizes from 150 × 150 to 650 × 650.

No.	Problem size		Average no. of iterations ± Standard deviation ($\mu \pm \sigma$)		
	m	n	DZP	LDP	ACP
1	150	150	607.96±104.86	400.34±55.82	405.20±45.64
2	200	200	1089.32±146.76	663.48±63.89	638.00±55.47
3	250	250	1581.46±217.37	1008.28±82.21	873.46±65.77
4	300	300	2383.50±297.00	1374.60±102.76	1120.60±74.05
5	350	350	3216.44±365.56	1810.70±138.23	1431.16±112.21
6	400	400	4218.00±517.35	2336.02±128.25	1782.74±137.22
7	450	450	5349.88±610.45	2930.36±164.92	2067.66±146.08
8	500	500	6774.70±673.83	3581.94±233.35	2457.86±168.58
9	550	550	7976.80±748.22	4260.74±188.36	2792.06±170.39
10	600	600	9378.76±932.69	5004.62±222.11	3213.22±141.35
11	650	650	11280.76±1021.6	5906.76±287.62	3644.34±215.71

Remark 4.2. No unbounded problem in Table 4.3.

Table 4.4: The average time ± standard deviation ($\mu \pm \sigma$) from solving LP problems by the simplex method with DZP, LDP and ACP with problem sizes from 150×150 to 650×650.

No.	Problem size		Average time (sec) ± Standard deviation		
	m	n	DZP	LDP	ACP
1	150	150	2.63±0.47	4.51±0.65	6.69±0.84
2	200	200	6.46±0.89	10.29±0.97	15.49±1.38
3	250	250	11.90±1.62	19.86±1.63	28.84±2.27
4	300	300	25.87±2.97	37.25±3.66	58.13±4.29
5	350	350	42.67±5.23	60.06±5.12	97.12±7.65
6	400	400	74.03±10.14	97.90±8.02	184.39±15.20
7	450	450	107.50±12.35	143.83±10.48	263.51±19.93
8	500	500	112.91±13.36	158.04±15.39	258.98±26.01
9	550	550	139.71±13.78	196.67±10.20	321.17±20.18
10	600	600	338.74±43.94	492.25±389.95	698.14±73.73
11	650	650	483.35±582.66	511.74±40.56	859.09±67.48

Tables 4.5 and 4.6 show the similar results to the previous tables but the matrix **A** is not square. In particular, $m < n$. The problem sizes are varied from 150×200 to 300×400 in these tables. Unfortunately, once the matrix **A** is no longer a square matrix, the randomly generated problems become unbounded more frequently, especially when the number of constraints is much less than the number of the decision variables. In many cases, unbounded problems occurs 100% as seen Table 4.5. Hence the average number of iterations in this case are the average number of iterations from solving an LP problem until unboundedness is determined or an optimal solution is found.

Table 4.5: The average number of iterations ± standard deviation ($\mu \pm \sigma$) from solving LP problems by the simplex method with DZP, LDP and ACP with problem sizes from 150 × 200 to 300 × 400.

No.	Problem size		Average no. of iterations (sec) ± Standard deviation			%Unbounded
	m	n	DZP	LDP	ACP	
1	150	200	911.84 ± 119.80	523.66 ± 44.46	543.30 ± 61.13	0
2	150	250	775.16 ± 294.49	562.48 ± 96.58	530.94 ± 120.56	62
3	150	300	354.66 ± 78.87	451.30 ± 66.76	394.88 ± 69.06	100
4	150	350	280.68 ± 41.47	405.98 ± 52.57	324.84 ± 47.11	100
5	150	400	256.98 ± 31.06	386.10 ± 39.23	305.18 ± 38.64	100
6	150	450	250.76 ± 29.52	367.54 ± 46.82	285.50 ± 37.20	100
7	150	500	230.58 ± 28.78	343.12 ± 28.71	264.92 ± 39.72	100
8	200	250	1395.42 ± 185.27	818.52 ± 52.43	770.96 ± 51.82	0
9	200	300	1489.28 ± 371.00	939.08 ± 109.11	871.96 ± 126.44	38
10	200	350	880.88 ± 408.19	855.30 ± 127.48	700.38 ± 158.06	94
11	200	400	506.58 ± 133.01	723.06 ± 89.37	556.18 ± 86.76	100
12	200	450	429.64 ± 52.35	681.08 ± 68.85	502.46 ± 61.03	100
13	200	500	395.89 ± 42.40	656.60 ± 67.27	476.40 ± 60.68	100
14	300	350	2742.15 ± 337.57	1560.00 ± 93.30	1312.47 ± 94.28	0
15	300	400	3315.11 ± 462.93	1790.65 ± 136.09	1528.36 ± 118.60	0

Table 4.6: The average time ± standard deviation ($\mu \pm \sigma$) from solving LP problems by the simplex method with DZP, LDP and ACP with problem sizes from 150 × 200 to 300 × 400.

No.	Problem size		Average time (sec) ± Standard deviation		
	m	n	DZP	LDP	ACP
1	150	200	4.69 ± 1.26	7.62 ± 1.29	12.71 ± 2.05
2	150	250	3.62 ± 1.36	8.37 ± 1.30	13.77 ± 2.36
3	150	300	1.56 ± 0.35	7.07 ± 1.05	12.10 ± 1.91
4	150	350	6.17 ± 9.60	15.63 ± 3.49	34.71 ± 14.31
5	150	400	5.76 ± 2.27	17.78 ± 4.09	42.01 ± 8.47
6	150	450	4.61 ± 0.57	15.16 ± 2.42	34.55 ± 5.77
7	150	500	4.80 ± 0.87	16.77 ± 3.41	40.55 ± 7.68
8	200	250	23.39 ± 3.31	26.63 ± 3.16	78.82 ± 5.56
9	200	300	10.22 ± 2.59	19.82 ± 2.50	33.54 ± 3.93
10	200	350	5.95 ± 2.69	19.48 ± 2.86	32.45 ± 5.89
11	200	400	4.21 ± 1.17	21.17 ± 8.79	32.93 ± 4.49
12	200	450	4.17 ± 0.91	22.09 ± 3.47	38.40 ± 7.35
13	200	500	4.00 ± 0.76	21.79 ± 2.68	36.78 ± 4.53
14	300	350	26.07 ± 3.45	40.67 ± 3.22	65.62 ± 5.66
15	300	400	30.56 ± 4.27	47.75 ± 3.63	81.89 ± 6.46

Tables 4.7 and 4.8 show the similar results to the previous tables but the matrix **A** is not square with $m > n$. The problem sizes are varied from 200 × 150 to 400 × 300. As before, any problem found to be unbounded will be discarded and a new ramdomly generated problem will replace it. Therefore, Tables 4.7 and 4.8 contains no unbounded problem.

Table 4.7: The average number of iterations ± standard deviation ($\mu \pm \sigma$) from solving LP problems by the simplex method with DZP, LDP and ACP with problem sizes from 200 × 150 to 400 × 300.

No.	Problem size		Average no. of iterations (sec) ± Standard deviation		
	m	n	DZP	LDP	ACP
1	200	150	727.84 ± 115.35	500.66 ± 70.07	476.68 ± 50.92
2	250	150	780.70 ± 118.77	600.88 ± 74.47	519.16 ± 49.45
3	300	150	834.78 ± 155.51	691.96 ± 76.73	561.60 ± 71.71
4	350	150	842.68 ± 117.47	752.26 ± 82.70	578.38 ± 65.61
5	400	150	884.36 ± 137.96	808.40 ± 114.91	610.02 ± 73.38
6	450	150	912.96 ± 141.60	889.30 ± 111.90	633.00 ± 76.79
7	500	150	946.14 ± 122.25	956.60 ± 129.73	637.44 ± 81.98
8	250	200	1275.00 ± 177.22	790.26 ± 87.73	724.06 ± 71.17
9	300	200	1340.72 ± 190.75	931.28 ± 94.99	757.96 ± 71.51
10	350	200	1385.08 ± 181.39	1033.18 ± 104.27	797.86 ± 76.44
11	400	200	1496.00 ± 208.32	1156.80 ± 107.87	860.18 ± 76.66
12	450	200	1505.20 ± 226.51	1217.14 ± 136.42	863.06 ± 75.94
13	500	200	1617.66 ± 229.22	1357.00 ± 128.52	910.92 ± 92.23
14	350	300	2889.70 ± 345.09	1809.00 ± 149.56	1312.36 ± 86.49
15	400	300	3033.82 ± 402.34	1945.10 ± 164.92	1391.50 ± 108.36

Remark 4.3. No unbounded problem in Table 4.7.

Table 4.8: The average time ± standard deviation ($\mu \pm \sigma$) from solving LP problems by the simplex method with DZP, LDP and ACP with problem sizes from 200×150 to 400×300.

No.	Problem size		Average time (sec) ± Standard deviation		
	m	n	DZP	LDP	ACP
1	200	150	4.30 ± 0.66	7.00 ± 0.98	9.74 ± 1.09
2	250	150	5.89 ± 0.96	10.14 ± 1.24	12.90 ± 1.27
3	300	150	7.82 ± 1.48	14.10 ± 1.69	16.62 ± 2.16
4	350	150	10.68 ± 1.79	19.48 ± 2.30	22.69 ± 2.91
5	400	150	12.74 ± 1.99	23.49 ± 3.38	26.97 ± 3.30
6	450	150	16.17 ± 3.05	30.70 ± 4.83	34.02 ± 6.04
7	500	150	16.58 ± 2.25	33.27 ± 4.62	33.34 ± 4.68
8	250	200	19.10 ± 9.92	44.41 ± 158.49	78.15 ± 44.19
9	300	200	19.78 ± 2.81	27.29 ± 3.29	94.37 ± 14.29
10	350	200	25.68 ± 3.70	35.28 ± 3.82	143.99 ± 18.24
11	400	200	34.53 ± 5.31	47.63 ± 4.81	230.20 ± 31.49
12	450	200	36.81 ± 6.29	54.39 ± 7.11	258.49 ± 34.66
13	500	200	62.08 ± 27.86	86.34 ± 28.42	514.53 ± 234.80
14	350	300	74.91 ± 10.45	85.30 ± 11.57	589.00 ± 64.59
15	400	300	90.44 ± 11.14	97.63 ± 10.52	766.30 ± 76.19

Table 4.9 shows the number of iterations and time from solving Klee and Minty problem with n varied from $n = 2, \ldots, 20$.

Table 4.9: The number of iterations and time from solving Klee and Minty problem by the simplex method with DZP, LDP and ACP with $n = 2, 3, \ldots, 20$.

No.	Problem size (n)	Iterations			Time (Sec)		
		DZP	LDP	ACP	DZP	LDP	ACP
1	2	3	1	1	0.0009	0.0005	0.0004
2	3	7	1	1	0.0011	0.0005	0.0001
3	4	15	1	1	0.0028	0.0006	0.0007
4	5	31	1	1	0.0124	0.0087	0.0013
5	6	63	1	1	0.0228	0.0054	0.0009
6	7	127	1	1	0.0515	0.0013	0.0013
7	8	255	1	1	0.0781	0.0011	0.0020
8	9	511	1	1	0.1990	0.0015	0.0019
9	10	1023	1	1	0.4440	0.0020	0.0025
10	11	2047	1	1	0.9280	0.0014	0.0030
11	12	4095	1	1	1.9967	0.0018	0.0030
12	13	8191	1	1	3.8323	0.0020	0.0036
13	14	16383	1	1	8.7590	0.0019	0.0036
14	15	32767	1	1	17.6465	0.0020	0.0046
15	16	65535	1	1	34.2084	0.0021	0.0048
16	17	131071	1	1	71.9518	0.0026	0.0051
17	18	262143	1	1	153.9014	0.0022	0.0063
18	19	524287	1	1	337.4665	0.0034	0.0069
19	20	1048575	1	1	724.2283	0.0054	0.0075

4.3 Analysis

In this section we discuss the performance analysis of ACP comparing to other pivot rules. Then we introduce the domain of problems that are suitable for applying the simplex method with ACP. Lastly, we analyze the operations involved for each pivot rule.

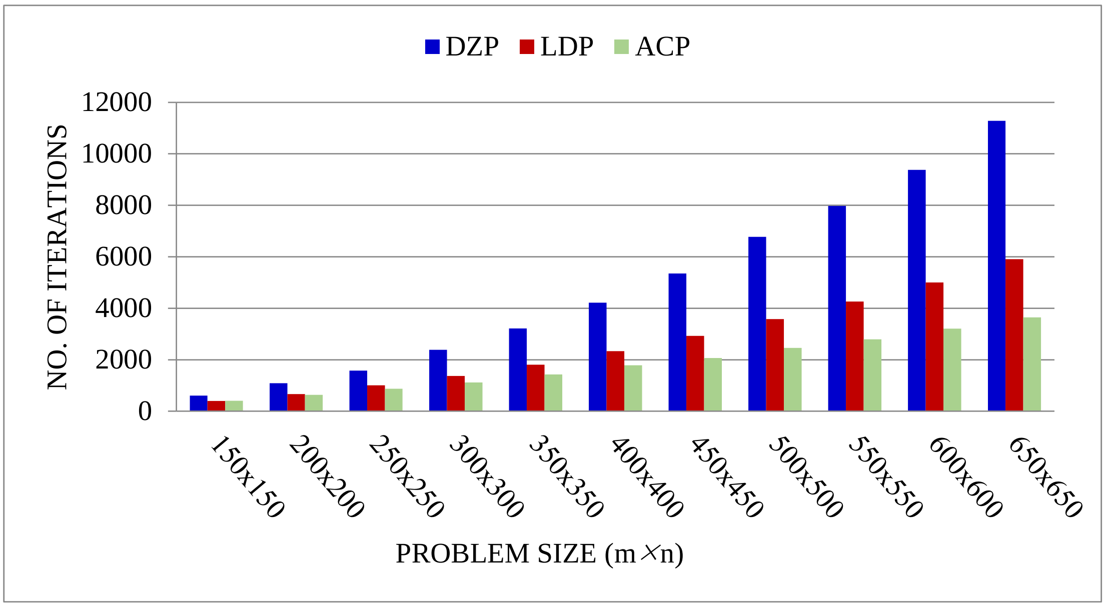

Figure 4.1: Comparison between the average number of iterations from solving LP problems by the simplex algorithm with DZP, LDP and ACP when **A** is a square matrix.

4.3.1 Analysis of Numerical Results

We analyze the numerical results by comparing the number of iterations from solving LP problems by the simplex method with ACP and the simplex method with DZP, LDP, SEP and DVP. Table 4.1 shows the average number of iterations of these pivot rule for small problems. The details in this table indicates the number of iterations from solving LP problems using DVP and SEP are much greater than DZP, LDP and ACP in most cases and on average. The average number of iterations from LDP is minimum comparing with other pivot rules. Table 4.2 shows the average time for solving LP problem from these pivot rules. In this table, SEP and DVP spend a long of time to solve LP problem while DZP uses minimum average time for solving LP problems. The next minimum average time pivot rules are LDP and ACP, in that order.

Table 4.3 shows the average number of iterations from DZP, LDP and ACP for larger problems. From this table ACP pivot rule uses less average number of

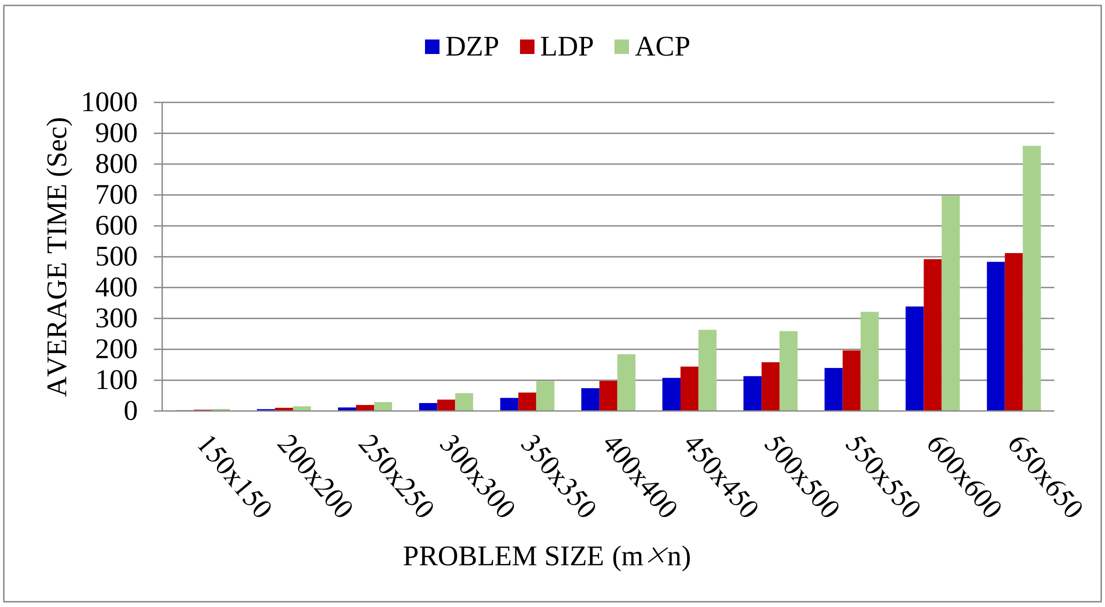

Figure 4.2: Comparison between the average time from solving LP problems by the simplex algorithm with DZP, LDP and ACP when **A** is a square matrix.

iterations than DZP in every case and less than the one from LDP in every case except the first one. Moreover, Figure 4.1 shows the details of Table 4.3. In this figure, the number of iterations between ACP, DZP and LDP is not significantly different in small problems while in larger-scale problems ACP achieves a better performance.

Table 4.4 indicates the average time from DZP, LDP and ACP. We found that DZP uses the minimum time for solving LP problem with several sizes since DZP only chooses the maximum negative reduced cost to be an entering variable while LDP has to compute the norm of the column vector and finds the minimum ratio between the current negative reduced cost and the norm of the column vector. ACP spent largest time for solving LP when compared with DZP and LDP since there are many steps involved in each iteration. First, ACP looks for the row with the minimum right-hand-side. Then ACP looks for an entering variable that has zero or negative value in its columns so that the minimum ratio is not applicable for those rows. If there is more than one candidate for such entering variable, ACP looks for the row with the next minimum right-hand-side and repeat the process until we have only one candidate or until we cannot find a row with zero or negative value.

In Table 4.5, problem sizes of LP problems are varied by $m < n$ from 150×200 to 300×400. The average number of iterations in Table 4.5 are the average number of iterations from solving LP problem until we found unbounded problem or optimal solution. The average number of iterations from ACP is less than one from DZP and LDP in some problem size. Figure 4.3 shows the details of Table 4.5 in terms of barchart. The details about average time used to solve LP problems in Table 4.5 is indicated in Table 4.6 and is indicated in terms of barchart in Figure 4.4.

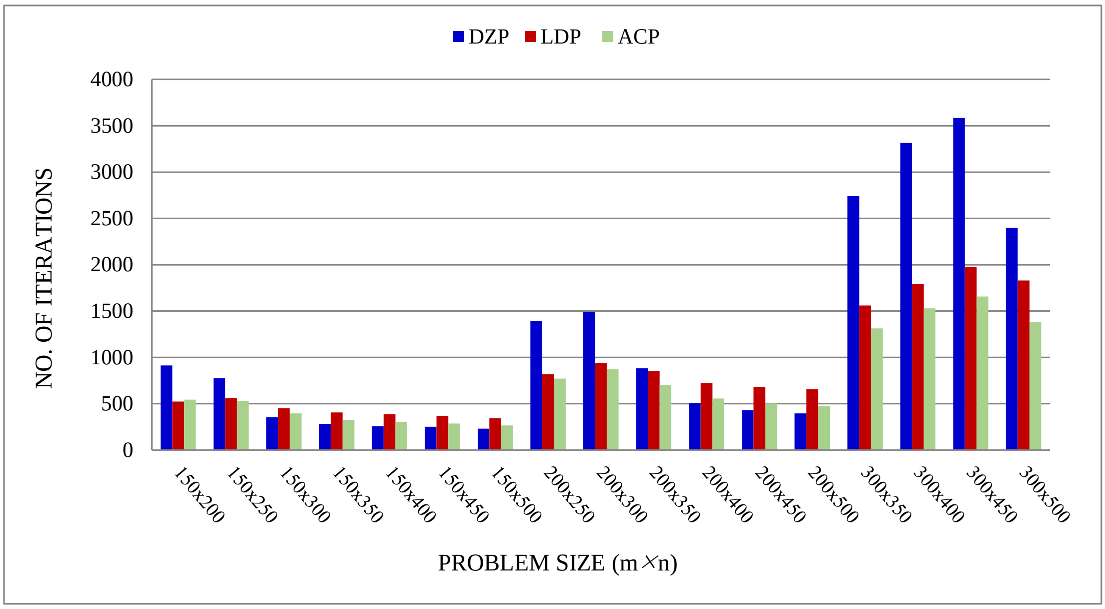

Figure 4.3: Comparison between the average number of iterations from solving LP problems by the simplex algorithm with DZP, LDP and ACP when $m < n$.

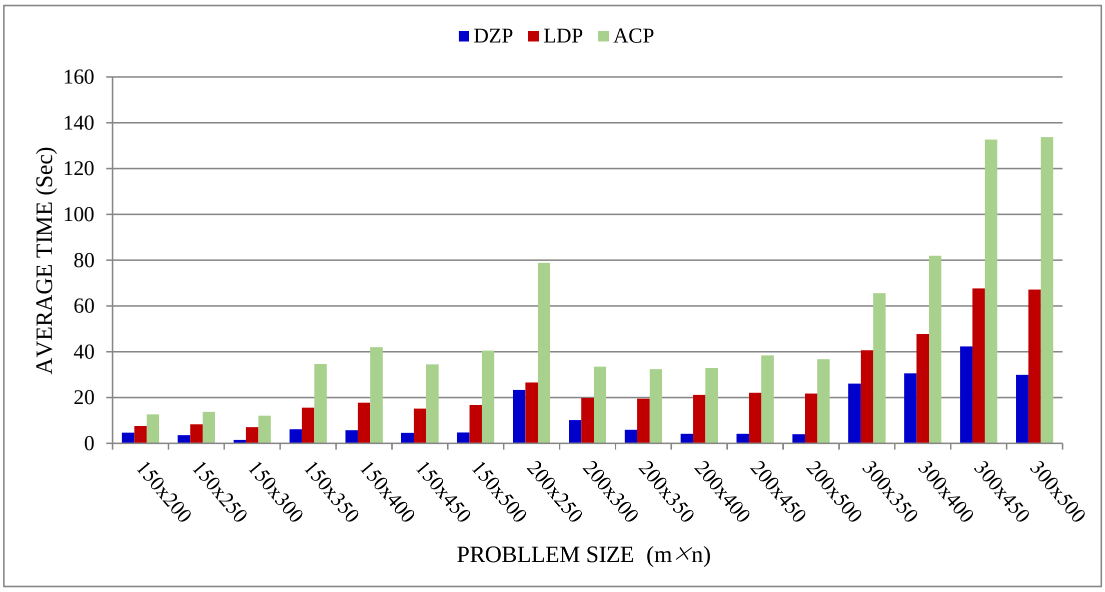

Figure 4.4: Comparison between the average time from solving LP problems by the simplex algorithm with DZP, LDP and ACP when $m < n$.

Table 4.7 and 4.8 also show average number of iterations and time similar to previous tables. In these tables, problem sizes of LP problems are varied by $m > n$ from 200×150 to 400×300. In every problem size in Table 4.7, ACP uses minimum average number of iterations for solving LP problems. In contrast, ACP took the maximum average time in every problem size (see Table 4.8). Figure 4.5 and 4.6 show the details of Table 4.7 and Table 4.8, respectively.

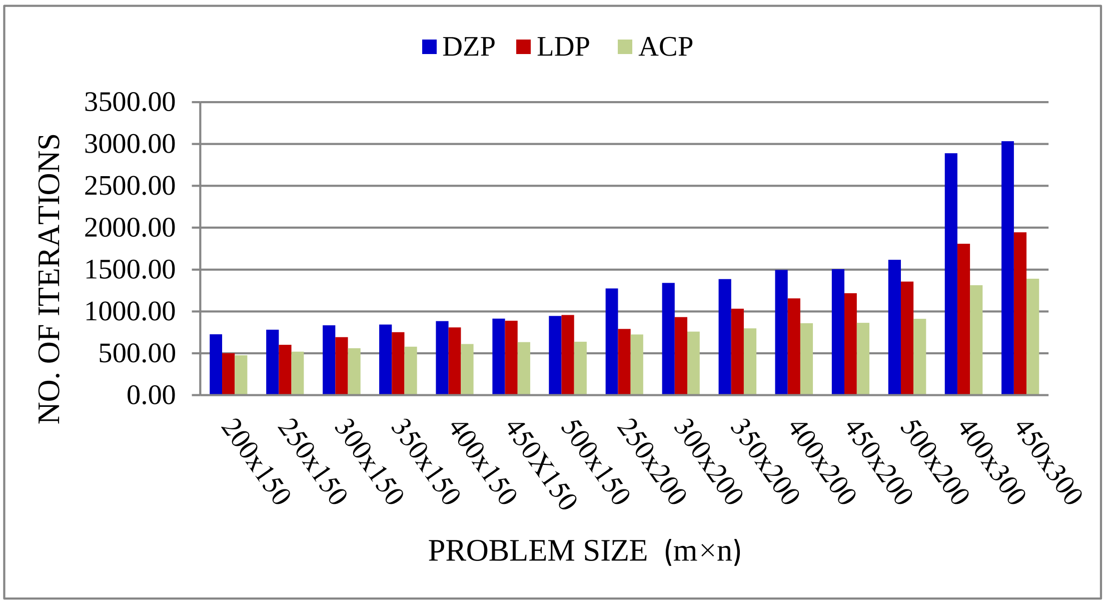

Figure 4.5: Comparison between the average number of iterations from solving LP problems by the simplex algorithm with DZP, LDP and ACP when $m > n$.

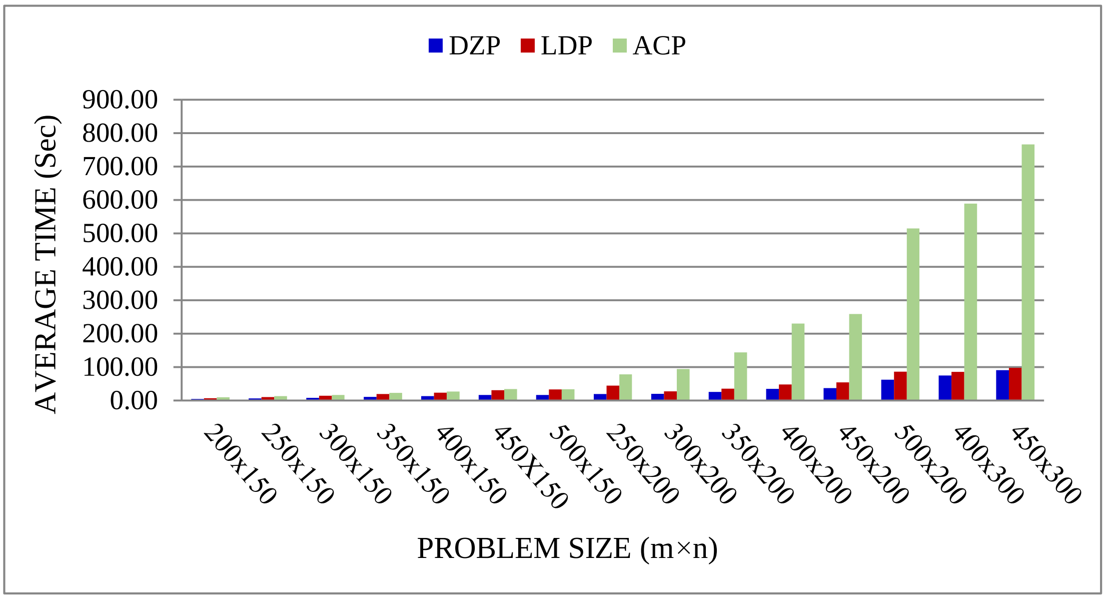

Figure 4.6: Comparison between the average time from solving LP problems by the simplex algorithm with DZP, LDP and ACP when $m > n$.

Even if ACP took the maximum average time in every problem (see Table 4.2, 4.4, 4.6 and 4.8), the average time in Table 4.4 tends to get closed to the average time from DZP as the problem gets bigger (see Table 4.10).

Table 4.10: The average time ratio between ACP and DZP from Table 4.4

No.	Problem size		Average time ratio
	m	n	ACP:DZP
1	150	150	2.54
2	200	200	2.40
3	250	250	2.42
4	300	300	2.25
5	350	350	2.28
6	400	400	2.49
7	450	450	2.45
8	500	500	2.29
9	550	550	2.30
10	600	600	2.06
11	650	650	1.78

Table 4.10 shows average time ratio between ACP and DZP from Table 4.4. In this table, the average time ratio tends to decrease (obviously seen Figure 4.7) when the problem size is increased. This show a promise that, when the problem size gets bigger the average time ratio will decrease until the average times from ACP and DZP are not significantly different.

We can see from the Figure 4.3 and Figure 4.5 that, when the matrix **A** is not square, the number of variables (columns) has more effect on the number of iterations than the number of constraints (rows).

Table 4.9 indicates results from solving Klee and Minty problem with several sizes of n. For all n, ACP and LDP achieve the optimal solution only one iteration while DZP used $2^n - 1$ iterations. The results are consistent with Theorem 3.3.

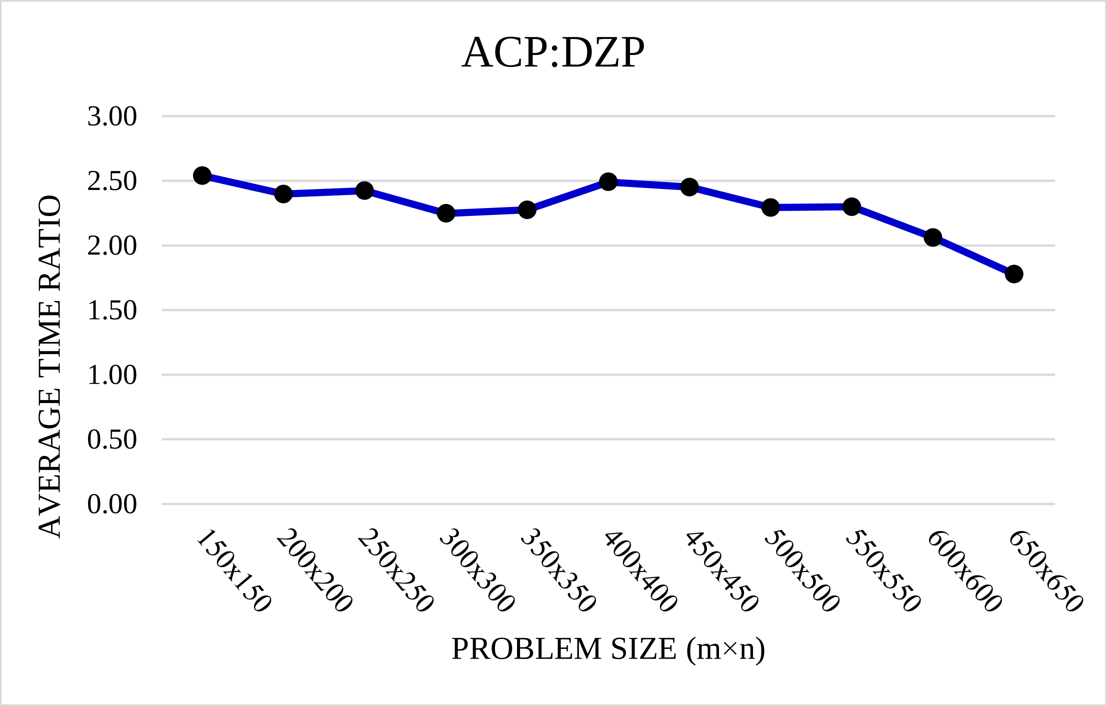

Figure 4.7: Average time ratio between ACP and DZP from Table 4.4.

4.3.2 Domain of Problems

The domain of problems that is suitable for applying the simplex algorithm with absolute change pivot rule are the problems which contain zero entries or entries with negative values that correspond to nonbasic columns in the coefficient matrix. This is because we want to prevent a leaving variable that causes small change in the objective function. If the coefficient matrix does not contain zero or negative entries, this rule is simply the Dantzig's rule.

4.3.3 Comparison of the Number of Operations

Suppose there are m constraints and n decision variables in a standard form of LP problem (4.1). If we transform the algebra of the simplex method into row operations in the tableau format, the number of rows are $m + 1$ (the number of constraints plus one row of objective function). The number of columns are the number of decision variables plus one, which is the column of right-hand-side (**b**). The total is $n + 1$ columns. The number of multiplications elements to update per iteration are $(m + 1)(n - m + 1)$. This is because the basic variable column are identity matrix and each nonbasic variable column is updated at each pivot operation, even if this variable never enters the basis, to make the computer implementations effectively we operate the pivot operation only on the column of nonbasic variables that are $n - m + 1$ column. The number of additions/ subtractions per iteration are $m(n - m + 1)$. We can say that the total number of operations per iteration is of order $O(mn)$. The number of operations from this simplex method with the absolute change pivot rule is not different from the number of operations from Dantzig's pivot rules.

For the simplex method with the steepest-edge pivot rule, Devex rule and the largest-distance pivot rule, before pivoting these rules calculate several terms such as norm of directions, norm of column vectors and the quotient between cost vector and norm of directions or norm of column vectors. These step is call **pricing step**. The details of the number of operations of the simplex method with steepest-edge pivot rule, Devex rule and the largest-distance pivot rule are shown in the Table 4.11, 4.12 and 4.13.

Table 4.11: Pricing operations from steepest-edge pivot rule

TERM	OPERATION	
	Multiplication	Addition
1. $\bar{c}_j = c^T d_j,\ j \in I_N$	$n(n - m)$	$n(n - m)$
2. $\|d_j\|_2,\ j \in I_N$	$n(n - m)$	$n(n - m)$
3. $\frac{\bar{c}_q}{\|d_q\|_2} = \min \left\{ \frac{\bar{c}_j}{\|d_j\|_2} \middle| j \in J \right\}$	$n - m$	-
Total	$(n - m)(2n + 1)$	$2n(n - m)$

After the pricing step, these pivot rules give the entering variable and operates the pivot operation. In the pivot step the number of multiplications and additions are the same as the pivot operation from the simplex method with Dantzig's pivot rule, which are $(m + 1)(n - m + 1)$ and $m(n - m + 1)$, respectively.

Table 4.12: Pricing operations from Devex rule

TERM	OPERATION	
	Multiplication	Addition
1. $\bar{c}_j = c^T d_j,\ j \in I_N$	$n(n-m)$	$n(n-m)$
2. $\lVert \hat{d}_j \rVert_2,\ j \in I_N$	$n(n-m)$	$n(n-m)$
3. $\bar{w}_p = \max\left\{1, \frac{1}{\alpha_q} \lVert \hat{d}_q \rVert_2\right\}$	1	-
4. $\bar{w}_j = \max\left\{w_j, \left\lvert \frac{\alpha_j}{\alpha_q} \right\rvert \lVert \hat{d}_q \rVert_2, j \in I_N, j \neq p\right\}$	n	-
5. $\frac{\bar{c}_q}{w_q} = \min\left\{\frac{\bar{c}_j}{w_j} \,\middle\vert\, j \in I_N\right\}$	$n-m$	-
Total	$(n-m)(2n+1)+n+1$	$2n(n-m)$

Table 4.13: Pricing operations from the largest-distance pivot rule

TERM	OPERATION	
	Multiplication	Addition
1. $\lVert A_j \rVert$	mn	mn
2. $\frac{\bar{c}_q}{\lVert A_q \rVert} = \max\left\{\frac{\bar{c}_j}{\lVert A_j \rVert} \,\middle\vert\, j \in J\right\}$	$n-m$	-
Total	$mn+n-m$	mn

Table 4.15 shows the comparison the total number of multiplications and additions from the simplex method with Dantzig's pivot rule, the steepest-edge pivot rule, Devex rule, the largest-distance pivot rule and absolute change pivot rule. From this table, we can see that the number of operations from Dantzig's pivot rule and the absolute change pivot rule are the same and not different from the steepest-edge pivot rule, Devex rule and the largest-distance pivot rule in terms of time complexity.

Table 4.14: Comparison of the number of operations.

METHOD		OPERATION	
		PIVOTING	PRICING
DZP	Multiplication	$(m+1)(n-m+1)$	
	Addition	$m(n-m+1)$	
SEP	Multiplication	$(m+1)(n-m+1)$	$(n-m)(2n+1)$
	Addition	$m(n-m+1)$	$2n(n-m)$
DVP	Multiplication	$(m+1)(n-m+1)$	$(n-m)(2n+1)+n+1$
	Addition	$m(n-m+1)$	$2n(n-m)$
LDP	Multiplication	$(m+1)(n-m+1)$	$mn+n-m$
	Addition	$m(n-m+1)$	mn
ACP	Multiplication	$(m+1)(n-m+1)$	
	Addition	$m(n-m+1)$	

Table 4.15: Total comparison of the number of operations.

METHOD	OPERATION	TOTAL
DZP	Multiplication	$m(n-m)+n+1$
	Addition	$m(n-m+1)$
SEP	Multiplication	$(n-m)(m+2n+1)+n+1$
	Addition	$m(n-m+1)+2n(n-m)$
DVP	Multiplication	$(n-m)(m+2n+1)+2n+2$
	Addition	$m(n-m+1)+2n(n-m)$
LDP	Multiplication	$m(2n-m-1)+2n+1$
	Addition	$m(2n-m+1)$
ACP	Multiplication	$m(n-m)+n+1$
	Addition	$m(n-m+1)$

CHAPTER V
SUMMARY OF RESULTS

In this thesis, we proposed a pivot rule called the absolute change pivot rule. The idea of this rule is to have the maximum improvement in the objective value in each iteration. The simplex method with ACP is tested with several simulated LP problems and the number of iterations, time, the number of multiplications and additions from the simplex method with this pivot rule are compared with the simplex method using Dantzig's original pivot rule and other rules. The results show that the proposed algorithm can reduce the number of iterations over the Dantzig's pivot rule and other rules, especially for large problems.

Table 4.3, 4.7 and 4.9 offer a summary of the average number of iterations of each method. We conclude that the simplex algorithm using the absolute change pivot rule is very fast for solving linear programming problems in terms of the number of iterations. But this pivot rule takes more time than other rules, i.e., DZP and LDP, as shown in Table 4.4, 4.6, 4.8 and 4.9. In addition, Figure 4.2, 4.4 and 4.6 show that the average time for solving LP problems by the simplex method with ACP is much worse as the problem get larger comparing with DZP and LDP. The reason of this is described in Chapter 4. Although, the average time from ACP is much worse than DZP and LDP, Table 4.10 indicates that the average time ratio between ACP and DZP tends to decrease (obviously seen Figure 4.7) when the problem size is increased.

The domain of problems that are suitable for applying the simplex algorithm with the absolute change pivot rule are problems which contain zero entries or entries with negative value that correspond to nonbasic columns in the coefficient matrix. Moreover, the number of operations from this simplex method with the absolute change pivot rule is of order $O(mn)$. It is not different from the number of operations from Dantzig's pivot rule, SEP, DVP and LDP in term of time complexity.

Finally, absolute change pivot rule performs very well on Klee and Minty problems.

REFERENCES

[1] Trakantalerngsak S.:*Operations Research 1*, Silpakorn University printing house, 2nd ed, Nakorn Pathom, 2550.

[2] Bazara M., Jarvis J., Sherali H.: *Linear programming and network flows*, John Whiley & Sons, 2nd ed, New York, 1990.

[3] Gass S.I.:*Linear programming*, McGraw-Hill, 5th ed, New York, 1994.

[4] Klee V. and Minty G.J.: *How good is the simplex algorithm? in inequalities*, Academic Press, New York, (1972), 159-175.

[5] Paulraj S., Chellappan C. and Natesan T.R. A heuristic approach for identification of redundant constraints in linear programming model. International Journal of Computer Mathematics Vol.83, Nos.8-9 (September 2006):675-683.

[6] Junior H.V. and Lins M.P.E. An improved initial basis for the simplex algorithm. Computer & Operations Research 32(2005):1983-1993.

[7] Forrest J., Goldfarb D.: A practical steepest-edge simplex algorithm for linear programming, *Mathematical Programming* **57**, (1992), 341-374.

[8] Harris P.M.J.: Pivot selection methods of the Devex LP code, *Mathematical Programming* **5**, (1973), 1-28.

[9] Pan P.-Q., A largest-distance pivot rule for the simplex algorithm, *European Journal of Operational research* **187**, (2008), 393-402.

[10] Tipawanna M. and Sinapiromsaran K.: Max-out-in pivot rule with cycling prevention for the simplex method, 2nd International Conference on Mathematical Modeling in Physical Sciences 2013 (IC-MSQUARE 2013), 1-5 Sept. 2013, Prague, Czech Republic, On-line in Journal of Physics:Conference Series **490**, (2014), pp 1-4.

[11] Chen D.-S., Batson R. G., Dang Y.:*Applied lnteger Programming : Modeling and Solution*, John Wiley & Sons,INC., New Jersey, 2010.

[12] Wu N., Coppins R.:*Linear Programming and Extensions*, McGraw-Hill, USA, 1981.

[13] George G.B.:*Linear Programming and Extensions*, Princeton University Press, Princetion, 1963.

[14] Stein W. and Joyner D.: SAGE: System for Algebra and Geometry Experimentation, *ACM SIGSAM Bulletin*, **2**, 39th ed, (2005), 61-64.

BIOGRAPHY

Name	Kittiphong Chankong
Date of Birth	9 June 1989
Place of Birth	Samut Songkhram, Thailand
Education	B.Sc.(Applied Mathematics), (First Class Honors) Silpakorn University, 2011
Publication	K. Chankong, B. Intiyot, K. Sinapiromsaran, *Absolute Change Pivot Rule for the Simplex Algorithm*, Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2014, IMECS 2014, 12-14 March, 2014, Hong Kong, pp1209-1213.