RESEARCH ARTICLE
doi: 10.2306/scienceasia1513-1874.2014.40.248

Max-out-in pivot rule with cycling prevention for the simplex method

Monsicha Tipawanna, Krung Sinapiromsaran*

Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University, Bangkok 10330 Thailand

*Corresponding author, e-mail: Krung.s@chula.ac.th

Received 5 Mar 2013
Accepted 27 Dec 2013

ABSTRACT: A max-out-in pivot rule is designed to solve a linear programming (LP) problem with a non-zero right-hand side vector. It identifies the maximum of the leaving basic variable before selecting the associated entering nonbasic variable. Our method guarantees convergence after a finite number of iterations. The improvement of our pivot rule over Bland's rule is illustrated by some cycling LP examples. In addition, we report computational results obtained from two sets of LP problems. Among 100 simulated LP problems, the max-out-in pivot rule is significantly better than Bland's rule and Dantzig's rule according to the Wilcoxon signed rank test. Based on these results, we conclude that our method is best suited for degenerate LP problems.

KEYWORDS: linear programming, Bland's rule

INTRODUCTION

Linear programming (LP) is a method to find the optimal points of an objective function subject to linear equality or linear inequality constraints. The system of constraints will form a feasible region. The simplex method was first proposed by George B. Dantzig in 1949 as a solution method for LP problems[1]. This method will start at a corner point corresponding to the initial basis of the feasible region and will then move to an adjacent corner point by increasing the value of a nonbasic variable, called the *entering variable*, and decreasing the value of a basic variable, called the *leaving variable*. The above process will be iterated until an optimal solution is obtained. Performance of the method depends on the number of iterations required to find an optimal solution of a given LP problem.

The rule to select the entering variable and leaving variable is called the *pivot rule*. It directly affects the number of iterations, and will in turn affect the efficiency of the simplex method. The well-known Dantzig's pivot rule selects the entering variable from among all nonbasic variables with the most positively reduced cost for maximization. Although the simplex method is practical for small size LP problems, there are examples of the worst case running time proposed by Klee and Minty[2]. These examples have an exponential growth of the running time based on the number of LP dimensions. In addition, the simplex method with Dantzig's rule cannot prevent cycling in LP[3]. If cycling occurs, the simplex method keeps repeating a degenerate basic feasible solution. Consequently, it may not converge to an optimal solution.

Several researchers have attempted to improve the simplex performance by reducing the number of iterations or the computational time, see Refs. 4–8. Since the pivot rule affects the simplex performance, a lot of research has been carried out by presenting new pivot rules such as Devex rule[9], Steepest-edge rule[10], and a largest-distance pivot rule[11]. However, these rules did not prevent cycling. In order to prevent cycling, Bland suggested a new finite pivot rule, called Bland's rule, in 1977. This rule will choose the entering variable with the smallest index from among all candidates, and the leaving variable will then be determined by the minimum ratio test. Should there be many leaving variables, the one with the smallest index will be chosen. This rule can be proved to prevent the cycling of the simplex method; however, it does not guarantee an improvement of the objective value.

Pivot rules can be categorized into two types: *in-out* and *out-in*. An in-out pivot rule first selects the entering variable from the non-basic variable set based on some criteria in order to improve the objective value, and then chooses the leaving variable from the basic variable set by the minimum ratio test[3]. On the other hand, an out-in pivot rule first selects the leaving variable, and then determines the entering variable that maintains the property of the basis.

In this paper, we propose a new out-in pivot rule called *max-out-in pivot rule* safeguarding with Bland's rule for the simplex method. The distinctive feature of this rule is that it first selects the leaving variable that has the largest right-hand-side value from the current basic variable set. Then it chooses the best corresponding entering variable that gives the smallest positive contribution to the binding constraint of the leaving variable. If the selected basic variable and nonbasic variable cannot be exchanged or there is no corresponding nonbasic candidate, then Bland's rule (safeguarding rule) will be used. We can show that our proposed rule can prevent cycling for LP problems having a non-zero right-hand-side vector. In addition, it improves Bland's rule for some cycling examples, and performs relatively well on the Klee and Minty's problem[2]

PRELIMINARIES

The simplex method

Consider an LP problem in the standard form:

$$\text{Maximize } \mathbf{c}^{\mathrm{T}}\mathbf{x} \text{ subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geqslant \mathbf{0}, \tag{1}$$

where $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{m\times n}$ $(m < n)$, and $\operatorname{rank}(\mathbf{A}) = m$. After some possible rearrangment of the columns of $\mathbf{A}$, we may let

$$\mathbf{A} = \begin{bmatrix}\mathbf{B} & \mathbf{N}\end{bmatrix}$$

where $\mathbf{B}$ is an $m \times m$ invertible matrix and $\mathbf{N}$ is an $m \times (n-m)$ matrix. Here $\mathbf{B}$ is called the basic matrix and $\mathbf{N}$ the associated nonbasic matrix. The set of basic indices and the set of nonbasic indices will be denoted by I_B and I_N, respectively. In this paper, we will assume that $\mathbf{b} \neq \mathbf{0}$. Moreover, the equality constraints can be transformed to have all $b_i \geqslant 0$ where b_i is the ith component of the vector $\mathbf{b}$.

Suppose that a basic feasible solution to the system (1) is

$$\begin{bmatrix}(\mathbf{B}^{-1}\mathbf{b})^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}},$$

and its associate objective value is z_0. Then

$$z_0 = \mathbf{c}_B^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{b}. \tag{2}$$

Let

$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_B^{\mathrm{T}} & \mathbf{x}_N^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$$

be a basic feasible solution to (1), where $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} \geqslant 0$ and $\mathbf{x}_N \geqslant 0$ denote the basic and nonbasic variables for the current basis, respectively. Then we can rewrite the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ as

$$\mathbf{b} = \mathbf{B}\mathbf{x}_B + \mathbf{N}\mathbf{x}_N. \tag{3}$$

Then

$$\mathbf{x}_B = \bar{\mathbf{b}} - \sum_{j\in I_N} (\mathbf{y}_j x_j), \tag{4}$$

where $\bar{\mathbf{b}} = \mathbf{B}^{-1}\mathbf{b}$, $\mathbf{y}_j = \mathbf{B}^{-1}\mathbf{A}_j$, and $\mathbf{A}_j$ denotes the jth column vector of $\mathbf{A}$. Let z denote the objective value and

$$\mathbf{c}^{\mathrm{T}} = \begin{bmatrix}\mathbf{c}_B^{\mathrm{T}} & \mathbf{c}_N^{\mathrm{T}}\end{bmatrix}.$$

From (1), (2), and (3), we have

$$z = z_0 - \sum_{j\in I_N} (z_j - c_j)x_j \tag{5}$$

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

$$J = \{j \mid z_j - c_j < 0, \ \ j \in I_N\} \tag{6}$$

is empty.

We now give a summary of the simplex method using Dantzig's rule to solve the LP problem (1).

The simplex algorithm:

```
Initial Step:
    Choose a starting basic feasible solution with the basis $\mathbf{B}$ and the associated nonbasic matrix $\mathbf{N}$.
Main Step:
    Step 1: Determine the entering variable from the nonbasic variables: By Dantzig's rule choose $x_k$ such that
            $z_k - c_k = \min\{z_k - c_k \mid j \in I_N\}$.
    Step 2: If $z_k - c_k \geqslant 0$ then
            $\mathbf{x} = [\mathbf{x}_B^{\mathrm{T}} \ \ \mathbf{x}_N^{\mathrm{T}}]^{\mathrm{T}}$
            is an optimal solution. Stop.
    Step 3: Determine the leaving variable $x_r$ from the basic variables by the minimum ratio test:
            $r = \arg\min \left\{ \frac{\bar{b}_j}{\bar{a}_{jk}} \,\middle|\, j \in \{1, \dots, m\} \right\}$.
    Step 4: Update $\mathbf{B}$ by swapping between the leaving and the entering variable, and go to Step 1.
```

Bland's rule

A basic feasible solution is called *degenerate* if one of its basic variables is equal to zero. In this case the entering nonbasic variable and its corresponding leaving basic variable does not increase in value and the objective value does not change. If a sequence of pivot steps starts from some basic feasible solution and ends at the same basic feasible solution, then we call this situation *cycling*. If cycling occurs, the simplex method with Dantzig's rule may not converge to an optimal solution.

In order to prevent cycling, Robert G. Bland proposed Bland's rule[12]. This rule is defined as follows:

Bland's rule:

Step 1: Select an entering variable x_k such that

$$k = \min\{k \mid k \in J\}.$$

Step 2: Select a leaving variable x_r by the minimum ratio test:

$$r = \arg\min\left\{ \frac{\bar{b}_j}{\bar{a}_{jk}} \;\middle|\; j \in \{1, \dots, m\} \right\}.$$

Among all indices r for which the minimum ratio test results in a tie, select the smallest index.

Note that the difference between the simplex method with Bland's rule and the simplex method with Dantzig's rule is the way to select the entering variable. Although Bland's rule is simple and converges in a finite iteration, this rule does not improve the objective value for each iteration. In the next section, we will present the new pivot rule, which uses Bland's rule as a safeguarding rule.

MAX-OUT-IN PIVOT RULE

Main idea

From (1), we separate the index set of the decision variables into two groups. The first group contains variables that have positive objective costs denoted by $\Gamma^+ = \{i \mid c_i > 0\}$. The remaining group is $\bar{\Gamma}^+$, denoted by $\bar{\Gamma}^+ = \{i \mid c_i \leqslant 0\}$. Consider an LP problem in the following form:

$$\begin{aligned} &\text{Maximize} \quad \sum_{i\in\Gamma^+} x_i - \sum_{j\in\bar{\Gamma}^+} \delta_j x_j \\ &\text{subject to} \quad \mathbf{Ax} = \mathbf{b}, \quad \mathbf{x} \geqslant \mathbf{0}, \end{aligned} \tag{7}$$

where $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{m\times n}$ $(m < n)$, $\operatorname{rank}(\mathbf{A}) = m$, and

$$\delta_i = \begin{cases} 1, & c_i < 0, \\ 0, & c_i = 0. \end{cases}$$

Note that any LP problem can be converted into the system (7) by substituting x_i with $c_i \neq 0$ by $|c_i|\, x_i$. We will next show that $\operatorname{rank}(\mathbf{A})$ remains unchanged under this transformation.

Proposition 1 (Ref. 13) *Let $\mathbf{C}$ be an $m \times m$ matrix over $\mathbb{R}$ and $\mathbf{D}$ be an $n \times n$ matrix over $\mathbb{R}$. If $\mathbf{C}$ and $\mathbf{D}$ are nonsingular and $\mathbf{A}$ is an $m \times n$ matrix over $\mathbb{R}$, then*

$$\operatorname{rank}(\mathbf{CA}) = \operatorname{rank}(\mathbf{A}) = \operatorname{rank}(\mathbf{AD}).$$

That is, rank is unchanged upon left or right multiplication by a nonsingular matrix.

Lemma 1 *Let $\mathbf{A}$ be an $m \times n$ matrix over $\mathbb{R}$ where $m \leqslant n$ and let $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n$ be the column vectors of $\mathbf{A}$. Let $\tilde{\mathbf{A}}$ be an $m \times n$ matrix over $\mathbb{R}$, denoted by $\tilde{\mathbf{A}} = [k_1\mathbf{A}_1, k_2\mathbf{A}_2, \dots, k_n\mathbf{A}_n]$, where $k_i \in \mathbb{R}\backslash\{0\}$ for $i = 1, \dots, n$. Then $\operatorname{rank}(\mathbf{A}) = m$ if and only if $\operatorname{rank}(\tilde{\mathbf{A}}) = m$.*

Proof: Assume that $\operatorname{rank}(\mathbf{A}) = m$. To show that $\operatorname{rank}(\tilde{\mathbf{A}}) = m$, let $\mathbf{D}$ be a diagonal matrix over $\mathbb{R}$ and denoted by

$$\mathbf{D} = \begin{bmatrix} k_1 & 0 & \cdots & 0 \\ 0 & k_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & k_n \end{bmatrix},$$

where $k_i \in \mathbb{R}\backslash\{0\}$ for $i = 1, \dots, n$. Then

$$\begin{aligned} \tilde{\mathbf{A}} &= \begin{bmatrix} k_1\mathbf{A}_1 & \cdots & k_n\mathbf{A}_n \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{A}_1 & \cdots & \mathbf{A}_n \end{bmatrix} \begin{bmatrix} k_1 & 0 & \cdots & 0 \\ 0 & k_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & k_n \end{bmatrix} \\ &= \mathbf{AD}. \end{aligned}$$

Since $\mathbf{D}$ is a diagonal matrix and $k_i \neq 0$ for each i, then $\det(\mathbf{D}) \neq 0$. Thus $\mathbf{D}$ is nonsingular. By Proposition 1, $\operatorname{rank}(\mathbf{A}) = \operatorname{rank}(\mathbf{AD})$. Since $\tilde{\mathbf{A}} = \mathbf{AD}$, $\operatorname{rank}(\tilde{\mathbf{A}}) = \operatorname{rank}(\mathbf{AD})$. Thus $\operatorname{rank}(\tilde{\mathbf{A}}) = \operatorname{rank}(\mathbf{A}) = m$.

Conversely, assume that $\operatorname{rank}(\tilde{\mathbf{A}}) = m$. To show that $\operatorname{rank}(\mathbf{A}) = m$, let $\bar{\mathbf{D}}$ be a diagonal matrix over $\mathbb{R}$ and denoted by

$$\bar{\mathbf{D}} = \begin{bmatrix} \frac{1}{k_1} & 0 & \cdots & 0 \\ 0 & \frac{1}{k_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & \frac{1}{k_n} \end{bmatrix},$$

where $k_i \in \mathbb{R}\backslash\{0\}$ for $i = 1, \dots, n$.

As with how we have shown that $\tilde{\mathbf{A}} = \mathbf{AD}$, it can also be shown that $\mathbf{A} = \tilde{\mathbf{A}}\bar{\mathbf{D}}$. Since $\bar{\mathbf{D}}$ is a diagonal matrix and $k_i \neq 0$ for each i, then $\det(\bar{\mathbf{D}}) \neq 0$. Thus $\bar{\mathbf{D}}$ is nonsingular. By Proposition 1, $\operatorname{rank}(\tilde{\mathbf{A}}) = \operatorname{rank}(\tilde{\mathbf{A}}\bar{\mathbf{D}})$. Since $\mathbf{A} = \tilde{\mathbf{A}}\bar{\mathbf{D}}$, $\operatorname{rank}(\mathbf{A}) = \operatorname{rank}(\tilde{\mathbf{A}}\bar{\mathbf{D}})$. Thus $\operatorname{rank}(\mathbf{A}) = \operatorname{rank}(\tilde{\mathbf{A}}) = m$. □

We can convert the system (1) to the system (7) by right multiplying $\mathbf{A}$ by

$$\mathbf{D} = \begin{bmatrix} k_1 & 0 & \cdots & 0 \\ 0 & k_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & k_n \end{bmatrix},$$

where

$$k_i = \begin{cases} 1/|c_i|, & c_i \neq 0, \\ 1, & c_i = 0. \end{cases}$$

Then $\mathbf{D}$ is nonsingular. By Lemma 1, $\operatorname{rank}(\mathbf{A})$ is unchanged under the transformation to the system (7).

Rewrite the system (7) into tableau:

	x_B	x_N	RHS
	$\mathbf{0}^T$	$\mathbf{c}_B^T\mathbf{B}^{-1}\mathbf{N} - \mathbf{c}_N^T$	$\mathbf{c}_B^T\mathbf{B}^{-1}\bar{\mathbf{b}}$
x_B	$\mathbf{I}_m$	$\bar{\mathbf{A}} = \mathbf{B}^{-1}\mathbf{N}$	$\bar{\mathbf{b}}$

where $\mathbf{I}_m$ is an identity matrix of size m, $\bar{\mathbf{A}} = (\bar{a}_{ij}) \in \mathbb{R}^{m\times(n-m)}$, $\bar{\mathbf{b}} = \mathbf{B}^{-1}\mathbf{b} \in \mathbb{R}^m$, $\bar{\mathbf{b}} \geqslant \mathbf{0}$, and $\mathbf{0} \in \mathbb{R}^m$.

From the fact that the leaving variable decreases to zero when it changes to the nonbasic variable, the increment in the value of the entering variable depends on the decrement in the value of the leaving variable. If the value of the leaving variable is large, the increase of the entering variable may be large. Then we should select the leaving variable that has the maximum value from among all basic variables. From the current basic variables of the system (7), we select x_r as

$$x_r = \max\{x_i \mid i \in I_B\}.$$

It is equivalent to selecting

$$r = \arg\max\{\bar{b}_i \mid i \in I_B\}.$$

Consider the binding constraint of x_r:

$$x_r + \sum_{i\in I_N\setminus J} \bar{a}_{ri}x_i + \sum_{j\in J} \bar{a}_{rj}x_j = \bar{b}_r. \tag{8}$$

In order to improve the objective value, we select the entering variable from the set J, and then set the other nonbasic variables to zero.

Let $j \in J$. From (8), if we decrease x_r to zero and fix $x_k = 0$ for $k \in I_N\setminus\{j\}$, if $\bar{a}_{rj} > 0$, we have $x_j = (\bar{b}_r/\bar{a}_{rj}) \geqslant 0$. We need to select the nonbasic variable that allows the maximum increase. We select $x_{\tilde{j}}$ such that

$$\tilde{j} = \arg\min\{\bar{a}_{rj} \mid \bar{a}_{rj} > 0,\ \ j \in J\}.$$

In summary, we select the leaving variable that has the maximum value from among all basic variables. Then we select the corresponding entering variable which its index is in the set J and has the smallest positive contribution to the binding constraint of the selected basic variable.

Max-out-in pivot rule (with Bland's rule safeguarding)

From our main idea, we state our proposed pivot rule, called the max-out-in pivot rule, as following.

Max-out-in pivot rule: If $J \neq \phi$.

Step 1: Select x_r to leave the basic such that

$$r = \arg\max\{\bar{b}_i \mid i \in I_B\}.$$

Step 2: Select $x_{\tilde{j}}$ to enter the basic such that

$$\tilde{j} = \arg\min\{\bar{a}_{rj} \mid \bar{a}_{rj} > 0,\ \ j \in J\}.$$

From the binding constraint of the selected basic variable x_r, if the algorithm hold all nonbasic variables except $x_{\tilde{j}}$ to zero, then

$$x_r + \bar{a}_{r\tilde{j}}x_{\tilde{j}} = \bar{b}_r.$$

If it sets $x_r = 0$, we have $x_{\tilde{j}} = (\bar{b}_r/\bar{a}_{r\tilde{j}}) \geqslant 0$.

The other basic variables are affected by increasing $x_{\tilde{j}}$ as

$$x_k = \bar{b}_k - \bar{a}_{k\tilde{j}}\left(\frac{\bar{b}_r}{\bar{a}_{r\tilde{j}}}\right),\ \ k \in I_B\setminus\{r\}.$$

The necessary condition for the minimum ratio test is

$$\frac{\bar{b}_r}{\bar{a}_{r\tilde{j}}} = \min\left\{\frac{\bar{b}_k}{\bar{a}_{k\tilde{j}}}\ \middle|\ \bar{a}_{k\tilde{j}} > 0,\ \ k = 1,\dots,m\right\}. \tag{9}$$

When it performs the max-out-in pivot rule, there are two possible cases.

Case 1. One basic variable x_r and one nonbasic variable $x_{\tilde{j}}$ can be swapped if $\{j \in J \mid \bar{a}_{rj} > 0\} \neq \phi$ and $\bar{b}_r/\bar{a}_{r\tilde{j}} = \min\{\bar{b}_k/\bar{a}_{k\tilde{j}} \mid \bar{a}_{k\tilde{j}} > 0,\ k = 1,\dots,m\} \geqslant 0$.

Case 2. The max-out-in pivot rule cannot be used.

2.1 If the selected basic variable violates the minimum ratio test, $\exists k$, $\bar{a}_{k\tilde{j}} > 0$, $\bar{b}_k/\bar{a}_{k\tilde{j}} < \bar{b}_r/\bar{a}_{r\tilde{j}}$. Therefore some $\tilde{x}_k$ where $k \in I_B$ may be negative, we apply safe-guarding rule.

2.2 If no corresponding nonbasic variable exists, $\{j \in J \mid \bar{a}_{rj} > 0\} = \phi$, then the max-out-in pivot rule cannot be used; instead it uses the safeguarding rule.

In Case 1, we can use the max-out-in pivot rule to perform the pivot step. However, we cannot perform the pivot step if Case 2 occurs. In order to prevent cycling we apply Bland's rule as the safeguarding rule instead. Note that the other in-out pivot rule can be applied as a safeguarding rule. The detail of this rule is given as follows.

Initial step: Convert the LP problem of system (1) into the system (7) by substituting $x_i/|c_i|$ for x_i if $c_i \neq 0$.

Let $J = \{j \mid z_j - c_j < 0,\ j \in I_N\}$.

If $J \neq \phi$, perform the max-out-in pivot rule. Otherwise, the current solution is optimal. Stop.

Max-out-in pivot rule (with Bland's rule safeguarding):

Step 1: Select index r such that

$$r = \arg\max\{\bar{b}_i \mid i \in I_B\}.$$

Step 2: If $\{j \in J \mid \bar{a}_{rj} > 0\} \neq \phi$. Select $\bar{a}_{r\tilde{j}}$ such that

$$\tilde{j} = \arg\min\{\bar{a}_{rj} \mid \bar{a}_{rj} > 0,\ j \in J\}.$$

Step 3: If $(\bar{b}_r/\bar{a}_{r\tilde{j}}) =$

$$\min\left\{ \frac{\bar{b}_k}{\bar{a}_{k\tilde{j}}} \;\middle|\; \bar{a}_{k\tilde{j}} > 0,\ k = 1, \dots, m \right\},$$

select x_r to leave the basic and $x_{\tilde{j}}$ to enter the basic and go to Step 5. Otherwise, go to Step 4.

Step 4: Perform Bland's Rule to obtain an entering and leaving variable.

Step 5: Update $\mathbf{B}$.

Next we give an example to illustrate the implementation of the proposed method. Let the objective row in the simplex tableau be the first row.

Example 1 Consider the following LP model:

Maximize $x_1 + x_2 + x_3 - x_6$ subject to

$$11x_1 - 2x_2 + 5x_3 + 12x_4 + 9x_5 + 14x_6 - x_7 \leqslant 200$$
$$10x_1 + 5x_2 + 15x_3 + 15x_4 + 10x_5 + 5x_6 + 5x_7 \leqslant 250$$
$$x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geqslant 0.$$

Let x_8 and x_9 be the slack variables associated with the first and the second constraint, respectively. Then the initial simplex tableau for the above model is

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	RHS
z	-1	-1	-1	0	0	1	0	0	0	0
x_8	11	-2	5	12	9	14	-1	1	0	200
x_9	10	**5**	15	15	10	5	5	0	1	250

Since the maximum value among all basic variables is at $x_9 = 250$ corresponding to $r = 2$. Then $\{\bar{a}_{2j} \mid j \in J \text{ and } \bar{a}_{2j} > 0\} = \{\bar{a}_{21}, \bar{a}_{22}, \bar{a}_{23}\} = \{10, 5, 15\}$. $\tilde{j} = 2 = \arg\min\{\bar{a}_{21}, \bar{a}_{22}, \bar{a}_{23}\}$ since $\{b_k/\bar{a}_{k2} \mid \bar{a}_{k2} > 0, \text{ for } k = 1, 2\} = \{b_2/\bar{a}_{22}\} = \{50\}$. From case 1, we select x_9 to be the leaving variable and select x_2 to be the entering variable. After pivoting, the simplex tableau is

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	RHS
z	1	0	2	3	2	2	1	0	1/5	50
x_8	15	0	11	18	13	16	1	1	2/5	300
x_2	2	1	3	3	2	1	1	0	1/5	50

This is the optimal tableau. The optimal solution is $x_2 = 50$, $x_j = 0$ for $j = 1, \dots, 7$ where $j \neq 2$ with the optimal value 50 and the number of iteration is 1. By the simplex method with Bland's rule, the number of iterations is 3.

Next, we show that our rule can prevent cycling for LP problems having non-zero right-hand-side vector, $b_i > 0$ for some i.

Theorem 1 *If an LP problem has $b_i > 0$ for some i, then the max-out-in pivot rule safeguarding with Bland's rule converges in finite iterations.*

Proof: Without loss of generality, we assume that the LP problem is in the form of the system (7) and an initial basic feasible solution is given.

At the current iterate, if the max-out-in pivot rule can be applied as in case 1, then the max-out-in pivot rule selects x_r to be the leaving variable and select $x_{\tilde{j}}$ to be the entering variable. Since $\bar{\mathbf{b}} = \mathbf{B}^{-1}\mathbf{b}$ and $b_i > 0$ for some i, then $\bar{b}_i > 0$ for some i. Then $x_r = \max\{\bar{b}_i \mid i \in I_B\} > 0$. Since $\tilde{j} = \arg\max\{\bar{b}_r/\bar{a}_{rj} \mid z_j - c_j < 0 \text{ and } \bar{a}_{rj} > 0\}$, then $x_{\tilde{j}} = \bar{b}_r/\bar{a}_{r\tilde{j}} > 0$. Let z_0 be the objective. After pivoting, we have $z = z_0 - (z_j - c_j)(\bar{b}_r/\bar{a}_{r\tilde{j}}) > z_0$. Thus the objective value improves. Then the cycling cannot occur.

Otherwise, Bland's rule as a safe-guarding rule is applied repeatedly until the cycling is broken or Case 1 is met which guarantees no cycle. □

The following example is given to illustrate our rule preventing cycling. We note the number of iterations by using Bland's rule to compare with our rule.

Example 2 Klee and Minty example[2]

Maximize $\frac{3}{4}x_1 - 20x_2 + \frac{1}{2}x_3 - 6x_4 + 3$ subject to

$$\frac{1}{4}x_1 - 8x_2 - x_3 + 9x_4 \leqslant 0$$
$$\frac{1}{2}x_1 - 12x_2 - \frac{1}{2}x_3 + 3x_4 \leqslant 0$$
$$x_3 \leqslant 1$$
$$x_1, x_2, x_3, x_4 \geqslant 0.$$

Replacing x_j by $|c_j|\,\tilde{x}_j$ for $j = 1, 2, 3, 4$, we have:

Maximize $\tilde{x}_1 - \tilde{x}_2 + \tilde{x}_3 - \tilde{x}_4 + 3$ subject to

$$\frac{1}{3}\tilde{x}_1 - \frac{2}{5}\tilde{x}_2 - 2\tilde{x}_3 + \frac{3}{2}\tilde{x}_4 \leqslant 0$$
$$\frac{2}{3}\tilde{x}_1 - \frac{3}{5}\tilde{x}_2 - \tilde{x}_3 + \frac{1}{2}\tilde{x}_4 \leqslant 0$$
$$2\tilde{x}_3 \leqslant 1$$
$$\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4 \geqslant 0.$$

Let $\tilde{x}_5$, $\tilde{x}_6$ and $\tilde{x}_7$ be slack variables associated with the first through the third constraints, respectively. Then the initial tableau for the above problem is

	$\tilde{x}_1$	$\tilde{x}_2$	$\tilde{x}_3$	$\tilde{x}_4$	$\tilde{x}_5$	$\tilde{x}_6$	$\tilde{x}_7$	RHS
z	−1	1	−1	1	0	0	0	3
$\tilde{x}_5$	1/3	−2/5	−2	3/2	1	0	0	0
$\tilde{x}_6$	2/3	−3/5	−1	1/2	0	1	0	0
$\tilde{x}_7$	0	0	**2**	0	0	0	1	1

First pivot (Case 1): $\tilde{x}_7$ leaves, and $\tilde{x}_3$ enters the basis.

	$\tilde{x}_1$	$\tilde{x}_2$	$\tilde{x}_3$	$\tilde{x}_4$	$\tilde{x}_5$	$\tilde{x}_6$	$\tilde{x}_7$	RHS
z	−1	1	0	1	0	0	1/2	7/2
$\tilde{x}_5$	1/3	−2/5	0	3/2	1	0	1	1
$\tilde{x}_6$	**2/3**	−3/5	0	1/2	0	1	1/2	1/2
$\tilde{x}_3$	0	0	1	0	0	0	1/2	1/2

Since the maximum value among all basic variables is at $x_5 = 1$ corresponding to $r = 1$. Then $\{\bar{a}_{1j} \mid j \in J \text{ and } \bar{a}_{1j} > 0\} = \{\bar{a}_{11}\} = \{\frac{1}{3}\}$ and $\tilde{j} = 1$. Since $\{b_k/\bar{a}_{k1} \mid \bar{a}_{k1} > 0,\ k = 1,2,3\} = \{b_1/\bar{a}_{11}, b_2/\bar{a}_{21}\} = \{3, \frac{3}{4}\}$ and $b_1/\bar{a}_{11} = 3 \neq \min\{3, \frac{3}{4}\}$. From Case 2, we apply Bland's rule. Then we select x_1 to be the entering variable and select x_6 to be the leaving variable.

Second pivot (Case 2): $\tilde{x}_1$ enters, and $\tilde{x}_6$ leaves the basis.

	$\tilde{x}_1$	$\tilde{x}_2$	$\tilde{x}_3$	$\tilde{x}_4$	$\tilde{x}_5$	$\tilde{x}_6$	$\tilde{x}_7$	RHS
z	0	1/10	0	7/4	0	3/2	5/4	17/4
$\tilde{x}_5$	0	−1/10	0	5/4	1	−1/2	3/4	3/4
$\tilde{x}_1$	1	−9/10	0	3/4	0	3/2	3/4	3/4
$\tilde{x}_3$	0	0	1	0	0	0	1/2	1/2

This is the optimal tableau. The optimal solution is $\tilde{x}_1 = \frac{3}{4}$, $\tilde{x}_3 = \frac{1}{2}$ and $\tilde{x}_j = 0$ for $j = 2, 4$. Then we get $x_1 = 1$,$x_3 = 1$ and $x_j = 0$ for $j = 2, 4$ and with the optimal value $\frac{17}{4}$ and the number of iteration is 2. By the simplex method with Bland's rule the number of iterations is 6. For this problem, by the simplex method with Dantzig's pivot rule cycles in 6 iterations[2].

To show the improvement of our pivot rule over Bland's rule, we collect and solve five LP problems (Examples 3–7) that have cycling when they are solved by the simplex method with Dantzig's rule[14]. Then we compare the results with Bland's rule. For each problem, we note the number of iterations that forms a cycle when the problem is solved by the simplex method with Dantzig's rule. Note that each problem has been converted into a maximization problem.

Example 3 Yudin and Gol'shtein[15]

Maximize $x_3 - x_4 + x_5 - x_6$ subject to

$$x_1 + 2x_2 - 3x_4 - 5x_5 + 6x_6 = 0$$
$$x_2 + 6x_3 - 5x_4 - 3x_5 + 2x_6 = 0$$
$$3x_3 + x_4 + 2x_5 + 4x_6 + x_7 = 1$$
$$x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geqslant 0.$$

Solution: $x_1 = \frac{5}{2}$, $x_2 = \frac{3}{2}$, $x_5 = \frac{1}{2}$; Maximum $= \frac{1}{2}$; Cycle $= 6$. Max-out-in pivot rule converges in 1 iteration. Bland's rule converges in 5 iterations.

Example 4 Kuhn example (Balinski and Tucker[16])

Maximize $2x_4 + 3x_5 - x_6 - 12x_7$ subject to

$$x_1 - 2x_4 - 9x_5 + x_6 + 9x_7 = 0$$
$$x_2 + \frac{1}{3}x_4 + x_5 - \frac{1}{3}x_6 - 2x_7 = 0$$
$$x_3 + 2x_4 + 3x_5 - x_6 - 12x_7 = 2$$
$$x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geqslant 0.$$

Solution: $x_1 = 2$, $x_4 = 2$, $x_6 = 2$; Maximum $= 2$; Cycle $= 6$. Max-out-in pivot rule converges in 2 iterations. Bland's rule converges in 2 iterations.

Example 5 Marshall and Suurballe[17]

Maximize $\frac{2}{5}x_5 + \frac{2}{5}x_6 - \frac{9}{5}x_7$ subject to

$$x_1 + \tfrac{3}{5}x_5 - \tfrac{32}{5}x_6 + \tfrac{24}{5}x_7 = 0$$
$$x_2 + \tfrac{1}{5}x_5 - \tfrac{9}{5}x_6 + \tfrac{3}{5}x_7 = 0$$
$$x_3 + \tfrac{2}{5}x_5 - \tfrac{8}{5}x_6 + \tfrac{1}{5}x_7 = 0$$
$$x_4 + x_6 = 1$$
$$x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geqslant 0.$$

Solution: $x_1 = 4$, $x_2 = 1$, $x_5 = 4$, $x_6 = 1$; Maximum $= 2$; Cycle $= 6$. Max-out-in pivot rule converges in 2 iterations. Bland's rule converges in 4 iterations.

Example 6 Beale example[18]

Maximize $\frac{3}{4}x_1 - 150x_2 + \frac{1}{50}x_3 - 6x_4$ subject to

$$\tfrac{1}{4}x_1 - 60x_2 - \tfrac{1}{25}x_3 + 9x_4 + x_5 = 0$$
$$\tfrac{1}{2}x_1 - 90x_2 - \tfrac{1}{50}x_3 + 3x_4 + x_6 = 0$$
$$x_3 + x_7 = 1$$
$$x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geqslant 0.$$

Solution: $x_1 = \frac{1}{25}$, $x_3 = 1$, $x_5 = \frac{3}{100}$; Maximum $= \frac{1}{20}$; Cycle $= 6$. Max-out-in pivot rule converges in 2 iterations. Bland's rule converges in 7 iterations.

The next example shows that our rule may prevent cycling for the case $b_i = 0$ for all i.

Example 7 Sierksma[19]

Maximize $3x_1 - 80x_2 + 2x_3 - 24x_4$ subject to

$$x_1 - 32x_2 - 4x_3 + 36x_4 + x_5 = 0$$
$$x_1 - 24x_2 - x_3 + 6x_4 + x_6 = 0$$
$$x_1, x_2, x_3, x_4, x_5, x_6 \geqslant 0.$$

Solution: Unbounded above; Cycle $= 6$. Max-out-in pivot rule converges in 4 iterations. Bland's rule converges in 4 iterations.

Table 1 shows the number of iterations of the max-out-in pivot rule safeguarding with Bland's rule and Bland's rule for Examples 2–7.

For Examples 2–7, max-out-in pivot rule improves Bland's rule. In addition, Example 7 shows that our rule prevent cycling for an LP problem having a zero right-hand-side vector.

Application to Klee and Minty's problem

In 1972, Klee and Minty showed a collection of LP problems that the simplex method performs the exponential worst-case running time[2]. This collection is called the Klee and Minty's problem, which is stated as the following.

Table 1 Comparison between max-out-in pivot rule and Bland's rule over 6 cycling problems.

Problem Name	Iteration number	
	Max-out-in	Bland's rule
Klee-Minty	2	6
Yudin and Gol'shtein	1	5
Kuhn example	2	24
Marshall and Suurballe	2	4
Beale example	2	7
Sierksma	4	4
Total	13	28

Klee and Minty's problem:

Maximize $\sum_{j=1}^{n} 10^{n-j} x_i$ subject to

$$2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \leqslant 100^{i-1}, \qquad i = 1, \dots, n$$
$$x_i \geqslant 0, \qquad i = 1, \dots, n.$$

The simplex method with Dantzig's pivot rule requires $2^n - 1$ iterations to solve Klee and Minty's problem[2]. However, the max-out-in pivot rule requires only one iteration for any n. This is a significantly improvement. We show the case for $n = 4$ by the next example.

Example 8 Consider the following problem:

Maximize $1000x_1 + 100x_2 + 10x_3 + x_4$ subject to

$$x_1 \leqslant 1$$
$$20x_1 + x_2 \leqslant 10^2$$
$$200x_1 + 20x_2 + x_3 \leqslant 10^4$$
$$2000x_1 + 200x_2 + 20x_3 + x_4 \leqslant 10^6$$
$$x_1, x_2, x_3, x_4 \geqslant 0.$$

Replacing x_j by $|c_j|\,\tilde{x}_j$ for $j = 1, 2, 3, 4$, we have:

Maximize $\tilde{x}_1 + \tilde{x}_2 + \tilde{x}_3 + \tilde{x}_4$ subject to

$$\tfrac{1}{1000}\tilde{x}_1 \leqslant 1$$
$$\tfrac{2}{100}\tilde{x}_1 + \tfrac{1}{100}\tilde{x}_2 \leqslant 10^2$$
$$\tfrac{2}{10}\tilde{x}_1 + \tfrac{2}{10}\tilde{x}_2 + \tfrac{1}{10}\tilde{x}_3 \leqslant 10^4$$
$$2\tilde{x}_1 + 2\tilde{x}_2 + 2\tilde{x}_3 + \tilde{x}_4 \leqslant 10^6$$
$$\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4 \geqslant 0.$$

Let $\tilde{x}_5$, $\tilde{x}_6$, $\tilde{x}_7$ and $\tilde{x}_9$ be the slack variables associated with the first to the forth constraint, respectively. Then the initial tableau for the above problem is

	$\tilde{x}_1$	$\tilde{x}_2$	$\tilde{x}_3$	$\tilde{x}_4$	$\tilde{x}_5$	$\tilde{x}_6$	$\tilde{x}_7$	$\tilde{x}_4$	RHS
z	-1	-1	-1	-1	0	0	0	0	0
$\tilde{x}_5$	0.001	0	0	0	1	0	0	0	1
$\tilde{x}_6$	0.02	0.01	0	0	0	1	0	0	10^2
$\tilde{x}_7$	0.2	0.1	0.1	0	0	0	1	0	10^4
$\tilde{x}_8$	2	2	2	**1**	0	0	1	0	10^6

First pivot: $\tilde{x}_8$ leaves, $\tilde{x}_4$ enters the basis.

	$\tilde{x}_1$	$\tilde{x}_2$	$\tilde{x}_3$	$\tilde{x}_4$	$\tilde{x}_5$	$\tilde{x}_6$	$\tilde{x}_7$	$\tilde{x}_4$	RHS
z	1	1	1	0	0	0	0	0	10^6
$\tilde{x}_5$	0.001	0	0	0	1	0	0	0	1
$\tilde{x}_6$	0.02	0.01	0	0	0	1	0	0	10^2
$\tilde{x}_7$	0.2	0.1	0.1	0	0	0	1	0	10^4
$\tilde{x}_4$	2	2	2	1	0	0	1	0	10^6

This is the optimal tableau. The optimal solution is $\tilde{x}_4 = 10^6$, $\tilde{x}_j = 0$ for $j = 1, 2, 3$. Then we get $x_4 = 10^6$, $x_j = 0$ for $j = 1, 2, 3$ and $j \neq 4$ with the optimal value 10^6 and the number of iteration is 1.

For this problem, our rule takes only 1 iteration. By the simplex method with Dantzig's pivot rule, the number of iterations is 15 [2].

COMPUTATIONAL EXPERIMENTS

In this section, we show the numerical runs and the computational results that show the efficiency of our rule in randomly generated LP problems. We randomly generated two sets, Set1 and Set2, of LP problems. Set1 contains only maximization problems. All coefficients of the vector **c** are 1. a_{ij} is between -19 and 19. The vector **b** is calculated by $\mathbf{b} = \mathbf{A}\mathbf{x}^*$ where $\mathbf{x}^*$ is the vector with its component lying between -19 and 19. All of the constraints are of the type less than or equal to. Set2 is generated similarly, except that each coefficients of the vector **c** is either 0 or 1. The following three codes were tested: Dantzig: uses the simplex method with Dantzig pivot rule; Bland: uses the simplex method with Bland pivot rule; Max-out-in: uses the simplex method with max-out-in pivot rule safeguarding with Bland rule.

The simplex code is implemented using Python[20]. Dantzig, Bland, max-out-in pivot rules are implemented as functions in Python. We generate 50 LP problems with $\mathbf{A} \in \mathbb{R}^{5\times 10}$ for Set1 and Set2. They are executed by the same simplex code with three different pivot rules. The results from our experiments are shown in Fig. 1. The number of iterations from the simplex method with Bland rule subtracting with the number of iterations from the simplex method with max-out-in pivot rule are plotted in Fig. 1. The positive value of a bar indicates the larger iterations of the simplex method with Bland rule comparing to our method. Most LP problems show that our method needs less number of iterations than that of the simplex method with Bland rule. Table 2 summarizes the results from Fig. 1.

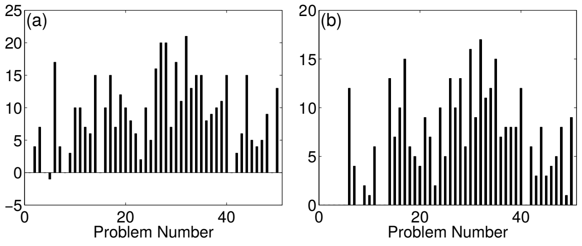

Fig. 1 The subtraction of the number of iterations between using Bland's rule by max-out-in pivot rule; (a) Set1, (b) Set2.

Table 2 Summary of the comparison between max-out-in pivot rule and Bland's rule.

Problem set	Number of iterations[*]		Average improvement
	Bland's rule	Max-out-in	
Set1	12.3 ± 4.8	3.6 ± 4.4	70%
Set2	10.0 ± 4.2	3.4 ± 3.6	66%

[*] Mean ± SD.

Next, the number of iterations from the simplex method with Dantzig's rule subtracting with the number of iterations from the simplex method with max-out-in pivot rule are plotted in Fig. 2. More negative bars appearing in the graph indicates that the Bland rule is inferior than Dantzig's rule. However, our method still maintains a larger number of positive differences as shown in Fig. 2 and Table 3.

To verify that the max-out-in pivot rule improved Bland's rule and Dantzig's rule over Set1 and Set2 problems, we used the Wilcoxon signed-rank test[21] with $\alpha = 0.05$.

As the Wilcoxon signed-rank test showed, our pivot rule is statistically faster than both Bland's rule and Dantzig's rule (Table 4).

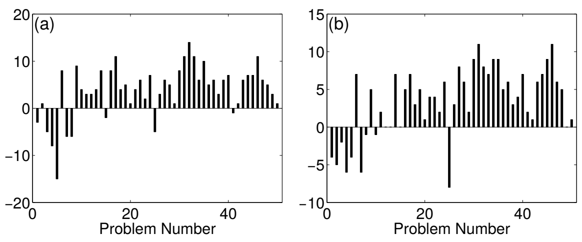

Fig. 2 The subtraction of the number of iterations between using Dantzig's rule by max-out-in pivot rule; (a) Set1, (b) Set2.

Table 3 Summary of the comparison between max-out-in pivot rule and Dantzig's rule.

Problem set	Number of iterations[*]		Average improvement
	Dantzig's rule	Max-out-in	
Set1	7.3 ± 2.7	3.6 ± 4.4	51%
Set2	6.7 ± 2.4	3.4 ± 3.6	50%

[*] Mean $\pm$ SD.

Table 4 The Wilcoxon signed-rank test compared max-out-in pivot rule with Bland's rule and Dantzig's rule.

Problem set		Median of difference	p-value
Set1	Bland's rule	8.5	4.1×10^{-9}
	Dantzig's rule	5	2.1×10^{-5}
Set2	Bland's rule	6.5	1.2×10^{-8}
	Dantzig's rule	4	1.8×10^{-5}

CONCLUSIONS

The objective of this paper is to propose a new pivot rule called the max-out-in pivot rule safeguarding with Bland's rule for the simplex method. The key features of this rule are that the maximum basic variable is selected to leave the basis, and the corresponding non-basic variable which allowed the maximum increase in the objective value is selected to enter the basis. This new rule can prevent cycling for an LP problem having a non-zero right-hand-side vector. According to our test problems, our rule is statistically better than Bland's rule. In addition, our rule performs relatively well on Klee and Minty problems[2].

For future work, we will test our algorithm in large-scale problems. Moreover, we plan to experiment our rule with other safeguarding rules.

Acknowledgements: The research was partially supported by the 90th Anniversary of Chulalongkorn University Fund (Ratchadaphiseksomphot Endowment Fund).

REFERENCES

1. Dantzig G (1963) *Linear Programming and Extensions*, Princeton Univ Press, Princeton, NJ.
2. Klee V, Minty G (1972) How good is the simplex algorithm? In: Shisha O (ed) *Inequalities III*, Academic Press, New York, pp 158–72.
3. Bazara M, Jarvis J, Sherali H (1990) *Linear Programming and Network Flows*, 3rd edn, John Whiley, NewYork.
4. Pan PQ (1990) Practical finite pivoting rules for the simplex method. *OR Spektrum* **12**, 219–25.
5. Vieira H Jr, Lins MPE (2005) An improved initial basis for the simplex algorithm. *Comput Oper Res* **32**, 1983–93.
6. Corley HW, Rosenberger J, Yeh WC, Sung TK (2006) The cosine simplex algorithm. *Int J Adv Manuf Tech* **27**, 1047–50.
7. Hu JF (2007) A note on "an improved initial basis for simplex algorithm". *Comput Oper Res* **34**, 3397–401.
8. Arsham H (2007) A computationally stable solution algorithm for linear programs. *Appl Math Comput* **188**, 1549–61.
9. Harris PMJ (1973) Pivot selection methods of the Devex LP code. *Math Program* **5**, 1–28.
10. Forrest JJ, Goldfarb D (1992) Steepest-edge simplex algorithms for linear programming. *Math Program* **57**, 341–74.
11. Pan PQ (2008) A largest-distance pivot rule for the simplex algorithm. *Eur J Oper Res* **187**, 393–402.
12. Bland RG (1977) New finite pivoting rules for the simplex method. *Math Oper Res* **2**, 103–7.
13. Horn R, Johnson C (1985) *Matrix Analysis*, Cambridge Univ Press.
14. Gass SI, Vinjamuri S (2004) Cycling in linear programming problems. *Comput Oper Res* **31**, 303–11.
15. Yudin D, Gol'shtein E (1965) *Linear Programming*. Israel Program of Scientific Traslations, Jerusalem.
16. Balinski ML, Tucker AW (1997) Duality theory of linear programs: a constructive approach with applications. *SIAM Rev* **11**, 347–77.
17. Marshall K, Suurballe J (1969) A note on cycling in the simplex method. *Nav Res Logist Q* **12**, 121–37.
18. Gass S (1985) *Linear Programming: Methods and Applications*, 5th edn, McGraw-Hill Book Company, New York.
19. Sierksma G (1969) *Linear and Integer Programming*, 2nd edn, Marcel Dekker, Inc., New York.
20. Lutz M (2010) *Programming Python*, 24th edn, O'Reilly Media.
21. Wilcoxon F (1945) Individual comparisons by ranking methods. *Biometrics Bull* **1**, 80–3.