Modification and implementation of two-phase simplex method

Nebojša V. Stojković[1*], Predrag S. Stanimirović[2] and Marko D. Petković[2]

[1] *University of Niš, Faculty of Economics, Trg Kralja Aleksandra 11, 18000 Niš, Serbia.*

[2] *University of Niš, Department of Mathematics, Faculty of Science, Višegradska 33, 18000 Niš, Serbia.*

E-mail: nebojsas@orion.eknfak.ni.ac.yu, pecko@pmf.pmf.ni.ac.yu, dexterofnis@gmail.com

Abstract

We investigate the problem of finding the initial basic feasible solution in the simplex algorithm. Two modifications of the two-phase simplex method are presented. Implementations of the two-phase simplex method and its modifications in the programming package MATHEMATICA and the programming language Visual Basic are written. We report computational results on numerical examples from the Netlib test set.

AMS Subj. Class.: 90C05

Key words: Linear programming, simplex method, basic feasible solution, Visual Basic.

1 Introduction

Consider the linear programming (LP) problem in the standard matrix form:

$$\begin{array}{ll} \text{Maximize} & \mathbf{c}^T\mathbf{x} - d, \\ \text{subject to} & A\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \geq 0, \end{array} \tag{1.1}$$

where $A \in \mathbb{R}^{m\times(m+n)}$ is the full row rank matrix $(\operatorname{rank}(A) = m)$, $\mathbf{c} \in \mathbb{R}^{n+m}$ and the system $A\mathbf{x} = \mathbf{b}$ is defined by $\mathbf{x} \in \mathbb{R}^{m+n}$, $\mathbf{b} \in \mathbb{R}^m$. It is assumed that (i, j)-th

*Corresponding author

entry in A is denoted by a_{ij}, $\mathbf{b} = (b_1, \dots, b_m)^T$, $d \in \mathbb{R}$, $\mathbf{x}^T = (x_1, \dots, x_{n+m})$ and $\mathbf{c}^T = (c_1, \dots, c_{n+m})$. For the LP problem given in the general form

$$\begin{array}{llll} \text{Maximize} & f(x) & = & f(x_1, \dots, x_{n_1}) = \sum_{j=1}^{n_1} c_j x_j - d \\ \text{subject to} & N_i: & & \sum_{j=1}^{n_1} a_{ij} x_j \le b_i, \quad i = 1, \dots, s \\ & J_i: & & \sum_{j=1}^{n_1} a_{ij} x_j = b_i, \quad i = s+1, \dots, m \\ & & & x_j \ge 0, \quad j = 1, \dots, n_1, \end{array}$$

we transform every inequality N_i into the corresponding equality by adding a nonnegative slack variable x_{n_1+i}:

$$N_i: \quad \sum_{j=1}^{n_1} a_{ij} x_j + x_{n_1+i} = b_i, \quad i = 1, \dots, s.$$

In this way we get an equivalent LP problem into the standard form (1.1), where $n = n_1 + s - m$, $c_j = 0$ for $j = n_1 + 1, \dots, n_1 + s$ and

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1,n_1} & 1 & \cdots & 0 \\ \vdots & & \vdots & \vdots & & \vdots \\ a_{s1} & \cdots & a_{s,n_1} & 0 & \cdots & 1 \\ a_{s+1,1} & \cdots & a_{s+1,n_1} & 0 & \cdots & 0 \\ \vdots & & \vdots & \vdots & & \vdots \\ a_{m,1} & \cdots & a_{m,n_1} & 0 & \cdots & 0 \end{bmatrix}.$$

The two-phase simplex method proceeds in two phases, phase I and phase II. Phase I attempts to find an initial basic feasible solution. Once an initial basic feasible solution has been found, phase II is then applied to find an optimal solution. The simplex method iterates through the set of basic solutions (feasible in phase II) of the LP problem (1.1). Each basic solution is characterized by the set of m basic variables $x_{B,1}, \dots, x_{B,m}$. Other n variables are called nonbasic variables and denoted by $x_{N,1}, \dots, x_{N,n}$.

If $\mathbf{b} \ge 0$ and all nonbasic variables $x_{N,1}, \dots, x_{N,n}$ are equal to zero, then $x_{B,1} = b_1, \dots, x_{B,m} = b_m$ is a basic feasible solution. If the condition $\mathbf{b} \ge 0$ is not satisfied, it is necessary to find an initial basic feasible solution or to determine that none exists. There exists a number of strategies for phase I. The classical approach is to associate with the LP problem (1.1) the following expanded problem:

$$\begin{array}{ll} \text{Minimize} & \mathbf{ew}, \\ \text{subject to} & A\mathbf{x} + \mathbf{w} = \mathbf{b}, \\ & \mathbf{x} \ge 0, \ \mathbf{w} \ge 0, \end{array} \tag{1.2}$$

where $\mathbf{e} = (1, \ldots, 1) \in \mathbb{R}^m$ and $\mathbf{w} \in \mathbb{R}^m$ is a vector of artificial variables. It is known that if $(\mathbf{x}^*, \mathbf{w}^*)$ is an optimal solution of (1.2), then a necessary and sufficient condition that (1.1) have a feasible solution is $w_i^* = 0, \quad i = 1, \ldots, m$ [1], [4]. Then no artificial variable is in the final basis, the artificial variables and corresponding columns are eliminated, and a feasible basis for the original LP problem is available. The drawback of this approach is the usage of artificial variables which makes the problem (1.2) larger than (1.1), since the phase I LP is obtained by adding m new artificial variables $w = (w_1, \ldots, w_m)$, one for each of the constraints.

The another variant of the two-phase simplex method is described in [5] and [6] and restated in the second section. In this paper we use this algorithm, because it does not require artificial variables.

This paper is organized as follows. In the second section we consider the transformation of the standard form into the equivalent canonical form and restate known algorithms from [5] and [6]. In the third section we present two new algorithms for obtaining the initial basic feasible solution in the phase I of the two-phase simplex algorithm from [5] and [6]. We provide a new rule for the choice of basic and nonbasic variables, i.e. for choosing the variable entering the base and one leaving the base. Ideally, we want to minimize the total computational effort. However, this is prohibitive. Therefore we aim at optimizing the current simplex step. In this way, we improve the computational efficiency of the simplex algorithm, which is confirmed by the numerical examples reported in the last section.

2 The simplex method

Without loss of generality we assume that the matrix A from (1.1) is of full rank, i.e. that equalities J_i are linearly independent. Otherwise, we apply Gauss-Jordan algorithm for the elimination of redundant equalities. Then we choose m basic variables $x_{B,1}, \ldots, x_{B,m}$, express them as a linear combination of nonbasic variables $x_{N,1}, \ldots, x_{N,n}$ and obtain the canonical form of the problem (1.1). We write this canonical form in the following table

(2.1)

$x_{N,1}$	$x_{N,2}$	$\ldots$	$x_{N,n}$	-1	
a_{11}	a_{12}	$\ldots$	a_{1n}	b_1	$= -x_{B,1}$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
a_{m1}	a_{m2}	$\ldots$	a_{mn}	b_m	$= -x_{B,m}$
c_1	c_2	$\ldots$	c_n	d	$= f$

Coefficients of the transformed matrix A and the transformed vector c are again denoted by a_{ij} and c_j, respectively, without loss of generality.

For the sake of completeness we restate one version of the two-phase maximization simplex algorithm from [5] and [6] for the problem (1.1), represented in the tableau form (2.1).

Within each iteration of the simplex method, exactly one variable goes from nonbasic to basic and exactly one variable goes from basic to nonbasic. The variable that goes from nonbasic to basic is called the entering variable; similarly, the variable that goes from basic to nonbasic is called the leaving variable. Usually there is more than one choice for the entering and the leaving variables. The next algorithm describes the move from the current base to the new base when the leaving-basic and entering-nonbasic variables have been selected.

Algorithm 1. (Interchange a basic variable $x_{B,p}$ and nonbasic variable $x_{N,j}$.)

$$a^1_{ql} = \begin{cases} \frac{1}{a_{pj}}, & q = p,\ l = j \\ \frac{a_{pl}}{a_{pj}}, & q = p,\ l \neq j \\ -\frac{a_{qj}}{a_{pj}}, & q \neq p,\ l = j \\ a_{ql} - \frac{a_{pl}a_{qj}}{a_{pj}}, & q \neq p,\ l \neq j \end{cases}, \qquad b^1_l = \begin{cases} \frac{b_p}{a_{pj}}, & l = p \\ b_l - \frac{b_p}{a_{pj}} a_{lj}, & l \neq p \end{cases},$$

$$c^1_l = \begin{cases} c_l - \frac{c_j a_{pl}}{a_{pj}}, & l \neq j, \\ -\frac{c_j}{a_{pj}}, & l = j \end{cases}, \qquad d^1 = d - \frac{b_p c_j}{a_{pj}}.$$

The next algorithm finds an optimal solution of the LP problem when the condition $b_1, \dots, b_m \geq 0$ is satisfied. This algorithm is called phase II simplex method.

Algorithm 2.

Step S1A. If $c_1, \dots, c_n \leq 0$, then the basic solution is an optimal solution.

Step S1B. Choose $c_j > 0$ according to the Bland's rule [2].

Step S1C. If $a_{1j}, \dots, a_{mj} \leq 0$, stop the algorithm. Maximum is $+\infty$. Otherwise, go to the next step.

Step S1D. Compute

$$\min_{1 \leq i \leq m} \left\{ \frac{b_i}{a_{ij}} \mid a_{ij} > 0 \right\} = \frac{b_p}{a_{pj}}. \tag{2.2}$$

If the minimum in (2.2) is not unique, make the choice according to the double least-index rule (Bland's rule) [2] to eliminate the cycling. Interchange the basic variable $x_{B,p}$ and nonbasic variable $x_{N,j}$ by applying *Algorithm 1.*

If the condition $b_1, \dots, b_m \geq 0$ is not satisfied, we use the algorithm from [6] and [5] to search for the initial basic feasible solution. In contrast of approach used in [4], it does not use artificial variables, and therefore does not increase the size of the problem. This algorithm is called phase I simplex method, and it is restated here as the following *Algorithm 3.*

Algorithm 3.

Step S2. Select the last $b_i < 0$.

Step S3. If $a_{i1}, \dots, a_{in} \geq 0$ then STOP. LP problem is infeasible.

Step S4. Otherwise, find $a_{ij} < 0$, compute

$$\min_{k>i}\left(\left\{\frac{b_i}{a_{ij}}\right\}\cup\left\{\frac{b_k}{a_{kj}}\mid a_{kj}>0\right\}\right)=\frac{b_p}{a_{pj}}, \tag{2.3}$$

choose $x_{N,j}$ as the entering-nonbasic variable, $x_{B,p}$ as the leaving-basic variable, apply *Algorithm 1* and go to *Step S2*. We also use the Bland's anti-cyclic rule if the minimum in (2.3) is not unique.

3 Modifications

The problem of selecting a leaving-basic variable and corresponding entering-nonbasic variable in the two-phase simplex method is contained in Step $S1D$ of *Algorithm 2* and Step $S4$ of *Algorithm 3*. We observed two drawbacks of Step $S4$. By i we denote the index of the last negative b_i.

1. If $p = i$ for each index $t < i = p$ such that

$$\frac{b_t}{a_{tj}} < \frac{b_p}{a_{pj}}, \quad b_t > 0,\ a_{tj} > 0$$

in the next iteration $x_{B,t}$ becomes negative:

$$x^1_{B,t} = b^1_t = b_t - \frac{b_p}{a_{pj}}a_{tj} < b_t - \frac{b_t}{a_{tj}}a_{tj} = 0.$$

2. If $p>i$, in the next iteration b^1_i is negative:

$$\frac{b_p}{a_{pj}} < \frac{b_i}{a_{ij}} \Rightarrow b^1_i = b_i - \frac{b_p}{a_{pj}}a_{ij} < 0.$$

Although there may exists $b_t<0$, $t < i$ such that

$$\min_{k>t}\left(\left\{\frac{b_t}{a_{tj}},\ a_{tj}<0\right\}\cup\left\{\frac{b_k}{a_{kj}}\mid a_{kj}>0,\ b_k>0\right\}\right)=\frac{b_t}{a_{tj}}.$$

In such case, it is possible to choose a_{tj} for the pivot element and obtain

$$x_{B,t} = b^1_t = \frac{b_t}{a_{tj}} \geq 0.$$

Also, since $\frac{b_t}{a_{tj}} \leq \frac{b_k}{a_{kj}}$, each $b_k > 0$ remains convenient for the next basic feasible solution:

$$x_{B,k} = b^1_k = b_k - \frac{b_t}{a_{tj}}a_{kj} \geq 0.$$

Therefore, it is possible that the choice of entering and leaving variable defined by *Step S4* reduces the number of positive b's after the application of *Algorithm 1.* Our goal is to obviate the observed disadvantages in *Step S4.* For this purpose, we propose a modification of *Step S4*, which gives a better heuristic for the choice of basic and nonbasic variables. That should reduce the number of iterations in the phase I of the two-phase simplex method.

Lemma 3.1 *Let the problem* (2.1) *be feasible and let x be the basic infeasible solution with $b_{i_1}, \dots, b_{i_q} < 0$, $q \le m$. Consider the index set $I = \{i_1, \dots, i_q\}$.*
The following statements are valid.

1. *It is possible to produce a new basic solution $x^1 = \{x^1_{B,1}, \dots, x^1_{B,m}\}$ with at most $q-1$ negative coordinates in only one step of the simplex method in the following two cases:*

a) *$q = m$, and*

b) *$q < m$ and there exist $r \in I$ and $s \in \{1, \dots, n\}$ such that*

$$\min_{h \notin I} \left\{ \frac{b_h}{a_{hs}} \mid a_{hs} > 0 \right\} \ge \frac{b_r}{a_{rs}}, \ a_{rs} < 0. \tag{3.1}$$

2. *It is possible to produce a new basic solution $x^1 = \{x^1_{B,1}, \dots, x^1_{B,m}\}$ with exactly q negative coordinates in one step of the simplex method if neither conditions a) nor b) are valid.*

Proof. **1.** a) If $q = m$, for an arbitrary pivot element $a_{js} < 0$ we get a new basic solution with at least one positive coordinate:

$$x^1_{B,j} = b^1_j = \frac{b_j}{a_{js}} > 0.$$

The existence of negative a_{js} is ensured by the assumption that the problem (2.1) is feasible.

b) Now assume that the conditions $q < m$, $r \in I$ and (3.1) are satisfied. Choose a_{rs} for the pivot element and apply *Algorithm 1*. Choose arbitrary $b_k \ge 0$, $k \ne r$.
In the case $a_{ks} < 0$ it is obvious that

$$x^1_{B,k} = b_k - \frac{b_r}{a_{rs}} a_{ks} \ge b_k \ge 0.$$

In the case $a_{ks} > 0$, using $\frac{b_k}{a_{ks}} \ge \frac{b_r}{a_{rs}}$, we conclude immediately

$$x^1_{B,k} = b^1_k = b_k - \frac{b_r}{a_{rs}} a_{ks} \ge b_k - \frac{b_k}{a_{ks}} a_{ks} = 0.$$

On the other hand, for $b_r < 0$ we obtain from *Algorithm 1*

$$b^1_r = \frac{b_r}{a_{rs}} \ge 0.$$

Therefore, all nonnegative b_k remain nonnegative and $b_r < 0$ becomes nonnegative.

2. If neither conditions a) nor b) are valid, let $r \notin I$ and $s \in \{1, \dots, n\}$ be such that

$$\min_{h \notin I} \left\{ \frac{b_h}{a_{hs}} \mid a_{hs} > 0 \right\} = \frac{b_r}{a_{rs}}. \tag{3.2}$$

By choosing a_{rs} as the pivot element and by applying the transformations defined in *Algorithm 1* we obtain the same number of negative elements in the vector b. This fact can be proved similarly as the part **1** b). □

Remark 3.1 *From Lemma 3.1 we get three proper selections of the pivot element in Step S4:*

- *arbitrary* $a_{js} < 0$ *in the case* $q = m$*;*
- *arbitrary* $a_{rs} < 0$ *satisfying* (3.1) *when the conditions* $0 < q < m$, $r \in I$ *are satisfied;*
- *arbitrary* $a_{rs} > 0$ *satisfying* (3.2) *when* $0 < q < m$ *and there is no* $a_{rs} < 0$ *satisfying conditions in the previous case.*

In accordance with Lemma 3.1 and considerations in Remark 3.1, we propose the following improvement of *Algorithm 3*.

Algorithm M1. (Modification of *Algorithm 3*).

Step 1. If $b_1, \dots, b_m \geq 0$ perform *Algorithm 2*. Otherwise continue.

Step 2. Select the first $b_{i_s} < 0$.

Step 3. If $a_{i_s,1}, \dots, a_{i_s,n} \geq 0$ then STOP. LP problem is infeasible. Otherwise, construct the set

$$Q = \{a_{i_s,j_p} < 0,\ p = 1, \dots, t\},$$

initialize variable p by $p = 1$ and continue.

Step 4. Compute

$$\min_{1\leq h\leq m} \left\{ \frac{b_h}{a_{h,j_p}} \mid a_{h,j_p} > 0,\ b_h > 0 \right\} = \min_{h\notin I} \left\{ \frac{b_h}{a_{h,j_p}} \mid a_{h,j_p} > 0 \right\} = \frac{b_r}{a_{r,j_p}}. \tag{3.3}$$

Step 5. If $\frac{b_{i_s}}{a_{i_s,j_p}} \leq \frac{b_h}{a_{h,j_p}}$ then interchange entering-nonbasic variable x_{N,j_p} and leaving-basic variable x_{B,i_s} (apply *Algorithm 1*) and go to *Step 1.* Otherwise go to *Step 6.*

Step 6. If $p > t$ interchange x_{N,j_p} and $x_{B,r}$ (apply *Algorithm 1*) and go to *Step 1.* Otherwise, put $p = p + 1$ and go to *Step 3.*

If there is no $b_r < 0$ such that the condition (3.1) is valid we choose pivot element according to Remark 3.1 to obtain a solution with the same number of negative b's. To avoid the cycling in this case, we will present an anti-cycling rule for A*lgorithm M1*, which is based on the following result.

Lemma 3.2 *Assume that there is no* $b_r < 0$ *such that the conditions* (3.1) *of Lemma 3.1 are satisfied. After choosing the pivot element according to* (3.2) *we obtain a new base where holds* $0 > b_i^1 \geq b_i$*, for all* $i \in I$*.*

Proof. From *Algorithm 1* we have:

$$b_i^1 = b_i - \frac{b_r}{a_{rs}} a_{is}.$$

According to (3.2) we obtain $\frac{b_r}{a_{rs}} \geq 0$. Now, taking into account $a_{is} < 0$, the conclusion $b_i^1 \geq b_i$ immediately follows.

On the other hand,

$$b_i^1 = a_{is}\left(\frac{b_i}{a_{is}} - \frac{b_r}{a_{rs}}\right) < 0$$

because the condition (3.1) of Lemma 3.1 is not valid for $b_i < 0$. □

Since i_s in Step 2 of *Algorithm M1* is fixed, *Algorithm M1* may cycle only if $b_{i_s}^1 = b_{i_s}$. For that reason, if the minimum in (3.3) is not unique we choose j_p according to the Bland's rule which guarantee that the simplex method always terminates [2], [7] (Theorem 3.3). Therefore, according to Lemma 3.2, after finite number of iterations value of b_{i_s} will start to increase or we will conclude that the problem is infeasible ($a_{i_s,j}$ are positive for all $j = 1, \ldots, n$).

Algorithm M1 chooses one fixed (the first) value $b_{i_s} < 0$ satisfying conditions of Lemma 3.1. But there may exists some other $b_i < 0$ such that conditions of Lemma 3.1 are satisfied, and in the next iteration we can obtain a basic solution with smaller number of negative b's. According to all previous considerations we establish *Algorithm M2*.

Algorithm M2. (Improved version of *Algorithm M1*).

Step 1. If $b_1, \ldots, b_m \geq 0$ perform *Algorithm 2*. Otherwise, construct the set

$$B = \{i_k \,|\, b_{i_k} < 0,\ k = 1, \ldots, q\}.$$

Step 2. Set $s = 1$ and perform the following:

Step 2.1. If $a_{i_s,1}, \ldots, a_{i_s,n} \geq 0$ then STOP. LP problem is infeasible. Otherwise, construct the set

$$Q = \{a_{i_s,j_p} < 0,\ p = 1, \ldots, t\},$$

put $p = 1$ and continue.

Step 2.2. Find the minima:

$$p' = \operatorname{argmin}\left\{\frac{b_k}{a_{k,j_p}} \,|\, b_k < 0, a_{k,j_p} < 0\right\},$$

$$M(j) = \min\left\{\frac{b_k}{a_{k,j_p}} \,|\, b_k > 0, a_{k,j_p} > 0\right\}.$$

If $\frac{b_k}{a_{k,j_p}} \leq M(j_p)$ then choose a_{p',j_p} for the pivot element, apply *Algorithm 1* and go to *Step 1.*

(In the next iteration b_k becomes positive).

Step 2.3. If $p < t$ then put $p = p + 1$ and go to *Step 2.2*, otherwise continue.

Step 2.4. If $s < q$ then put $s = s + 1$ and go to *Step 2.1*, otherwise continue.

Step 3. (Condition (3.1) is not valid)

Step 3.1. Select $j_0 = \text{argmin}\left\{x_{N,l} \mid a_{i_q,l} < 0\right\}$.

Step 3.2. Compute:

$$p'' = \text{argmin}\left\{x_{B,p} \mid \frac{b_p}{a_{p,j_0}} = M(j_0)\right\}.$$

Step 3.3. Choose $a_{p'',j}$ for pivot element, apply *Algorithm 1* and go to *Step 1*.

4 Numerical experience

We implemented all presented algorithms in the package MATHEMATICA and in the programming language Visual Basic. Software MarPlex (written in programming language Visual Basic), is available on:
`http://tesla.pmf.ni.ac.yu/people/dexter/software/marplex.zip`.

Example 4.1 In this example we point out the sensitivity of the algorithm in [6] to the initial ordering of the main constraints. We tested MATHEMATICA implementation on the LP problem

Maximize $-3x_1 - 2x_2$
subject to $-x_1 + 3x_2 \le -1,\ -2x_1 - 10x_2 \le -10,\ 2x_1 + 4x_2 \le 8,\ 3x_1 - 5x_2 \le 6.$

and encountered next problems. By using *Algorithm 1–Algorithm 3* we obtain the maximal value $-17/2$ and the extreme point $x_1 = 5/2, x_2 = 1/2$ in 4 iterations.

If we change the order of the constraints, and consider the same objective function subject to the constraints
$2x_1 + 4x_2 \le 8,\ 3x_1 - 5x_2 \le 6,\ -x_1 + 3x_2 \le -1,\ -2x_1 - 10x_2 \le -10,$
we obtain the same solution in two iterations.

Moreover, in the next configuration of the constraints
$-2x_1 - 10x_2 \le -10,\ 3x_1 - 5x_2 \le 6,\ 2x_1 + 4x_2 \le 8,\ -x_1 + 3x_2 \le -1$
we achieve the optimal solution in three iterative steps.

The source of this problem lies in the mentioned drawback of *Algorithm 3* as well as in the specific choice of basic and nonbasic variables in MATHEMATICA. About the package MATHEMATICA see, for example [8].

On the other hand, using the Algorithm M1 we get the solution in two iterations for all cases.

Example 4.2 We tested the code *MarPlex* on some *Netlib* test problems. For each problem we reserve three rows in the Table 1: the first row contains results corresponding to *Algorithm M2*, the second one corresponds to *Algorithm 3* and the third row corresponds to *Algorithm M1.* Dash in a column means that the implementation of the particular algorithm gives wrong result. The number of iterations for finding an initial basic feasible solution (phase I) are arranged in columns denoted by *Bf.* The number of applications of the phase II and the total number of iterations are given in columns denoted by *Sim.* and *Bf.+Sim.*, respectively. In the last column we place results obtained by code PCx [3]. Let us mention that code PCx is based on primal-dual interior point method.

Name	Bf.	Sim.	Bf.+Sim.	Objective value	PCx
	57	44	101	225494.963162364	
adlittle	77	38	115	225494.96316238	2.25494963e+005
	21	54	76	225494.963162379	
	4	8	12	−464.753142857143	
afiro	17	5	22	−464.753142857143	−4.64753143e+002
	2	9	11	−464.753142857143	
	67	22	89	−35991767.2865765	
agg	84	31	115	−35991767.2865765	−3.59917673e+007
	38	25	63	−35991767.2865765	
	40	69	109	−20239252.3559771	
agg2	52	64	118	−20239252.3559771	−2.02392521e+007
	31	123	154	−20239252.3559771	
	71	77	148	10312115.9293083	
agg3	141	81	222	10312115.7307162	1.03121159e+007
	51	143	194	10312115.9372015	
	273	159	432	−158.628018177046	
bandm	3128	171	3299	–	−1.58628018e+002
	1495	127	1622	–	
	1	33	34	33591.8961121999	
beaconfd	1	33	34	33591.8961121999	3.35924858e+004
	1	33	34	33591.8961121999	
	1	732	733	−30.769485006264	
blend	1	732	733	−30.769485006264	−3.08121498e+001
	1	732	733	−30.769485006264	
	1276	72	1348	1518.50982913344	
brandy	2248	81	2329	–	1.51851054e+003
	624	90	714	1518.50992977114	
	251	120	371	2690.01291380796	
capri	214	138	352	2690.01291380796	2.69001291e+003
	1316	163	1479	2691.57274856721	
	6933	591	7524	2185196.69885648	
czprob	11261	635	11886	2185196.69882955	2.18519682e+006
	6824	648	7472	2185196.69885615	
	215	318	533	−18.7519290765415	
e226	5663	567	6230	–	−1.87519291e+001
	395	364	759	–	

Name	Bf.	Sim.	Bf.+Sim.	Objective value	PCx
etamacro	191	257	448	−755.715233369051	−7.55715223e+002
	185	176	361	−755.715233352295	
	162	215	377	−755.715233346024	
finnis	141	375	516	172791.065595611	1.72791066e+005
	276	308	584	172791.065595611	
	809	225	1034	172791.03306592	
fit1d	1	834	835	−9146.3780989634	−9.14637809e+003
	1	834	835	−9146.3780989634	
	1	834	835	−9146.3780989634	
ganges	1	420	421	−109585.736129308	−1.09585736e+005
	1	420	421	−109585.736129308	
	1	420	421	−109585.736129308	
gfrd-pnc	229	305	534	6902235.99954881	6.90223600e+006
	240	337	577	6902235.99954882	
	126	311	437	6902235.99954882	
grow15	1	879	880	−106870942.285325	−1.06870941e+008
	1	879	880	−106870942.285325	
	1	879	880	−106870942.285325	
grow22	1	3569	3570	−160871482.230788	−1.60834336e+008
	1	3569	3570	−160871482.230788	
	1	3569	3570	−160871482.230788	
grow7	1	240	241	−47787811.8605706	−4.77878118e+007
	1	240	241	−47787811.8605706	
	1	240	241	−47787811.8605706	
israel	2	157	159	−896644.821863043	−8.96644817e+005
	2	157	159	−896644.821863043	
	2	157	159	−896644.821863043	
kb2	1	50	51	−1749.9001299062	−1.74990013e+003
	1	50	51	−1749.9001299062	
	1	50	51	−1749.9001299062	
lotfi	76	128	204	−25.2647060618762	−2.52647061e+001
	339	158	397	−25.2647060618632	
	111	137	248	−25.2647060618773	
recipe	9	30	39	−266.616	−2.66616000e+002
	8	29	37	−266.616	
	9	28	37	−266.616	
sc105	1	56	57	−52.2020612117073	−5.22020612e+001
	1	56	57	−52.2020612117073	
	1	56	57	−52.2020612117073	
sc205	1	135	136	−52.2020612117073	−5.22020612e+001
	1	135	136	−52.2020612117073	
	1	135	136	−52.2020612117073	
sc50a	1	26	27	−64.5750770585645	−6.45750771e+001
	1	26	27	−64.5750770585645	
	1	26	27	−64.5750770585645	
sc50b	1	29	28	−70	−7.00000000e+001
	1	29	28	−70	
	1	29	28	−70	

Name	Bf.	Sim.	Bf.+Sim.	Objective value	PCx
scagr7	81	41	122	−2331389.82433099	−2.33138982e+006
	90	35	125	−2331389.82433097	
	69	26	95	−2331389.82433098	
scfxm1	1133	97	1220	18417.3255500362	1.84167590e+004
	2478	200	2878	–	
	311	123	434	18416.7590283489	
scorpion	90	38	128	1878.12482273811	1.87812482e+003
	114	37	151	1878.12482273811	
	70	70	140	1878.12482273811	
sctap1	320	8	328	1412.25	1.41225000e+003
	496	57	553	1412.24999999998	
	131	137	268	1412.25	
sctap2	739	195	934	1724.80714285713	1.72480714e+003
	739	195	934	1724.80714285713	
	739	195	934	1724.80714285713	
sctap3	469	252	721	1424	1.42400000e+003
	618	247	865	1424	
	369	909	1278	1424	
seba	79	32	111	15711.6000000006	1.57116000e+004
	90	40	130	15711.5999999923	
	–	–	–	–	
share1b	89	65	154	−76589.3185791853	−7.65893186e+004
	366	69	435	−76589.3224159041	
	368	63	431	−76589.3185791526	
share2b	135	38	173	−415.73224074142	−4.15732241e+002
	123	46	169	−415.732240741419	
	92	25	117	−415.732240741416	
shell	41	276	317	1208825346	1.20882535e+009
	55	279	334	1208825346	
	78	278	356	1208825346	
ship04l	8	251	259	1793324.53797036	1.79332454e+006
	450	124	574	1793324.53797036	
	100	374	474	1793324.53797035	
ship04s	14	172	186	1798714.70044539	1.79871471e+006
	116	185	301	1798714.70044539	
	57	194	251	1798714.70044539	
ship08l	320	530	850	1909055.21138913	1.90905521e+006
	461	364	825	1909055.21138913	
	144	631	775	1909055.21138913	
ship08s	54	258	312	1920098.21053462	1.92009821e+006
	169	239	408	1920098.21053462	
	67	272	339	1920098.21053462	
ship12l	49	1019	1068	1470187.91932926	1.47018797e+006
	938	1908	2846	1470187.91932926	
	232	1711	1943	1470187.91932926	
ship12s	55	439	494	1489236.13440613	1.48923613e+006
	429	556	985	1489236.13440613	
	166	486	632	1489236.13440613	

Name	Bf.	Sim.	Bf.+Sim.	Objective value	PCx
sierra	75	326	401	15394362.1836319	1.53943622e+007
	65	325	490	15381546.3836319	
	82	310	392	15394362.1836319	
stair	2450	44	2494	−251.26695098074	−2.51266951e+002
	686	33	719	−251.266951192317	
	11066	168	11234	−	
standata	21	138	159	1257.6995	1.25769951e+003
	76	98	174	1257.6995	
	146	116	262	1257.69949999999	
standmps	131	109	240	1406.0175	1.40601750e+003
	260	155	415	1406.01749999996	
	752	72	824	1406.0175	
stocfor1	1	17	18	−41131.9762194364	−4.11319762e+004
	1	17	18	−41131.9762194364	
	1	17	18	−41131.9762194364	
vtp.base	69	55	124	129831.462637412	1.29831463e+005
	179	71	250	129831.462461362	
	430	47	477	129831.464051472	

Table 1.

5 Conclusion

We described two improvements of the algorithm for finding the initial basic feasible solution of the conventional simplex algorithm from [5], [6]. Both algorithms as well as the conventional two-phase simplex algorithm are implemented, tested and compared for each other. From Table 1 it is evident that *Algorithm M2* gives the best results, in general. Also *Algorithm M1* is better with respect to *Algorithm 3*, in the most cases. This agrees with our theoretical considerations. Summarizing the results arranged in Table 1, we see that *Algorithm M2* gives a minimal number of iterations in 19 problems, *Algorithm M1* in 12, and *Algorithm 3* in 4 test problems. In the rest 15 problems all methods give the same number of iterations.

Acknowledgement. We are grateful to Professor James Strayer for helpful conversations.

References

[1] D. Bertsimas, J.N. Tsitsiklis, *Introduction to linear optimization*, Athena Scientific, Belmont, Massachusetts, 1997.

[2] R.G. Bland, *New finite pivoting rules for the simplex method*, Mathematics of Operations Research **2** (1977) 103–107.

[3] J. Czyzyk, S. Mehrotra and S.J. Wright, *PCx User Guide*, Optimization Thechnology Center, Technical Report 96/01 (1996) 1–21.

[4] J.P. Ignizio, *Linear programming in single-multiple-objective systems,* Englewood Cliffs: Prentice Hall, 1982.

[5] E. Nering and A. Tucker, *Linear Programs and Related Problems*, Academic Press, New York, 1993.

[6] J. Strayer, *Linear Programming and Its Applications*, Springer-Verlag 1989.

[7] R.J. Vanderbei, *Linear Programming: Foundations and Extensions*, Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08544, 2001.

[8] S. Wolfram, *The Mathematica Book, 4th ed.*, Wolfram Media/Cambridge University Press, 1999.