FILOMAT 17 (2003), 169–176

SEVERAL MODIFICATIONS OF SIMPLEX METHOD

NEBOJŠA V. STOJKOVIĆ, PREDRAG S. STANIMIROVIĆ
AND MARKO D. PETKOVIĆ

ABSTRACT. [1] **We analyze the problem of finding the first basic solution in the two phases simplex algorithm. Also, a modification and several improvements of the simplex method are introduced. We report computational results on numerical examples from Netlib test set.**

1. INTRODUCTION

Consider the linear program

$$
\begin{aligned}
&\text{Maximize} && f(x) = f(x_{N,1},\dots,x_{N,n_1}) = \sum_{i=1}^{n_1} c_i x_{N,i} - d \\
&\text{subject to} && N_i^{(1)}: \quad \sum_{j=1}^{n_1} a_{ij}x_{N,j} \le b_i, \quad i=1,\dots,r \\
& && N_i^{(2)}: \quad \sum_{j=1}^{n_1} a_{ij}x_{N,j} \ge b_i, \quad i=r+1,\dots,s \\
& && J_i: \quad \sum_{j=1}^{n_1} a_{ij}x_{N,j} = b_i, \quad i=s+1,\dots,m \\
& && x_{N,j} \ge 0, \quad j=1,\dots,n_1.
\end{aligned}
\tag{1.1}
$$

Every inequality of the form $N_i^{(1)}$ (LE constraint) we change into the equality by adding a slack variable $x_{B,i}$:

$$N_i^{(1)}: \quad \sum_{j=1}^{n_1} a_{ij}x_{N,j} + x_{B,i} = b_i, \quad i=1,\dots,r.$$

1991 *Mathematics Subject Classification.* 90C05.
Key words and phrases. Linear programming, simplex method, Visual Basic
[1] Presented at the IMC "Filomat 2001", Niš, August 26–30, 2001.

Also, every inequality of the form $N_i^{(2)}$ (GE constraint) we transform into the equality by subtracting a surplus variable $x_{B,i}$:

$$N_i^{(2)}: \quad \sum_{j=1}^{n_1} a_{ij}x_{N,j} - x_{B,i} = b_i, \quad i = r+1, \dots, s.$$

Formally, we add slack variables $x_{B,i}, i = s+1, \dots, m$ with the fixed value zero in every equality constraint. In a such way we get the equivalent linear program into the standard form

(1.2)

$$\begin{array}{ll}
\text{Maximize} & c_1x_{N,1} + \cdots + c_{n_1}x_{N,n_1} - d \\
\text{subject to} & Ax = b, \\
& b = (b_1, \dots, b_m),\ x = (x_{N,1}, \dots, x_{N,n_1}, x_{B,1}, \dots, x_{B,m}), \\
& x_{N,j} \geq 0,\ j = 1, \dots, n, \\
& x_{B,i} \geq 0,\ i = 1, \dots, s,\ x_{B,i} = 0,\ i = s+1, \dots, m,
\end{array}$$

where the matrix A is in $\mathbb{R}^{m\times(n_1+m)}$.

In the second section we consider the transformation of the standard form into the equivalent canonical form and restate known algorithms. In the third section we accelerate the process of finding the first basic solution in the simplex algorithm, improving the choice of basic and nonbasic variables. Also, we introduce several improvements of two phases simplex method. Several numerical examples are reported in the last section.

2. The simplex method

Without loss of generality we assume that the matrix A is of full rank ($\operatorname{rank}(A) = m$), i.e. that equalities in J_i are linearly independent. Otherwise, we apply Gauss-Jordan algorithm for the elimination of redundant equalities. After that, we apply the next algorithm to obtain the canonical form of the problem (1.2).

Algorithm 1.

Step 1. If $J_i = \emptyset$ (the empty set), perform *Algorithm 4.*

Step 2. Find the first p such that pth constraint is equality and choose the last $a_{pj} \neq 0$ for the pivot element. (If $b_p \neq 0$ and there not exist $a_{pj} \neq 0$, the problem is not feasible; if $b_p = 0$, then we can drop pth constraint.)

Step 3. Applying *Algorithm 2* replace basic variable $x_{B,p} = 0$ and nonbasic variable $x_{N,j}$, and drop jth column (because of $x_{B,p} = 0$).

Step 4. If there exists the next p such that pth constraint is equality then perform *Step 2.* Otherwise, apply *Algorithm 4.* After the substitution $n =$

$n_1 + s - m$, the canonical form of problem (1.2) could be written in the following tableau form

(2.1)

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

where $x_{N,1}, \dots, x_{N,n}$ is the set of nonbasic variables and $x_{B,1}, \dots, x_{B,m}$ are basic variables. Transformed coefficients of the matrix A and the vector c are denoted by a_{ij} and c_j, respectively, without loss of generality.

For the sake of completeness we restate one version of the classical two phases maximization algorithms from [1], [3], [5] and [6] with respect to linear problem (1.1), which is presented in the tableau form (2.1).

Algorithm 2. (Replacing a basis variable $x_{B,p}$ and nonbasis variable $x_{N,j}$.)

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

$$b^1_l = \begin{cases} \frac{b_p}{a_{pj}}, & l = j \\ b_l - \frac{b_p}{a_{pj}} a_{lj}, & l \neq p \end{cases}$$

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

$$d^1 = d - \frac{b_p c_j}{a_{pj}}.$$

Algorithm 3. (Simplex method for basic feasible solution.)

Step S1A. If $c_1, \dots, c_n \leq 0$, then the basic solution is an optimal solution.
Step S1B. Choose an arbitrary $c_j > 0$. (We use the maximal c_j).
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

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

and replace nonbasic and basic variables $x_{N,j}$ and $x_{B,p}$, respectively, applying *Algorithm 2.*

If the condition $b_1, \dots, b_m \geq 0$ is not satisfied, we use the following algorithm to search for the first basic feasible solution from [6]. With respect to analogous algorithms from [1], [3] and [5] this algorithm does not use artificial variables, and does not increase dimensions of the problem.

Algorithm 4. (Find the first basic feasible solution).

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

Step S3. If $a_{i1}, \dots, a_{in} \geq 0$ then STOP. Linear program can not be solved.

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

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

and replace nonbasic and basic variables $x_{N,j}$ and $x_{B,p}$, respectively, using *Algorithm 2*. We use the last $a_{ij} < 0$.

3. Modifications

The problem of the replacement of a basic and a nonbasic variable in the general simplex method is contained in Step $S1D$ and Step $S4$. We observed two drawbacks of Step $S4$.

1. If $p = i$ and if there exists 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:

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

But, 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 this 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 basic feasible solution:

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

For this purpose, we propose a modification of *Step S4*. This modification follows from the following lemma.

Lemma 3.1. *Let the problem* (2.1) *be feasible and let x be the basic infeasible solution with $b_{i_1}, \dots, b_{i_q} < 0$. Consider the set $I = \{i_1, \dots, i_q\}$.*

In the following two cases:

a) *$q = m$, and*

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

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

it is possible to produce the new basic solution $x^1 = \{x^1_{B,1}, \dots, x^1_{B,m}\}$ with at most $q-1$ negative coordinates in only one iterative step of the simplex method, if we choose a_{rs} for the pivot element, i.e. replace nonbasic variable $x_{N,s}$ with the basic variable $x_{B,r}$.

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

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

b) Assume now that the conditions $q < m$ and (3.1) are satisfied. Choose a_{rs} for the pivot element. For $k \neq r$, $k \notin I$ and $a_{ks} < 0$ it is obvious that

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

For $k \neq r$, $k \notin I$ and $a_{ks} > 0$, using $\frac{b_k}{a_{ks}} \geq \frac{b_r}{a_{rs}}$, we conclude immediately

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

Hence, all positive b_k remain positive. Moreover, for $b_r < 0$ we get

$$b^1_r = \frac{b_r}{a_{rs}} \geq 0,$$

which completes the proof. □

In accordance with these considerations, we propose the following improvement of *Algorithm 4*.

Algorithm 5. (Modification of *Algorithm 4*).

Step 1. If $b_1, \dots, b_m \geq 0$ perform *Algorithm 3*. Otherwise, construct the set

$$B = \{b_{i_1}, \dots, b_{i_q}\} = \{b_{i_k} \mid b_{i_k} < 0, \ k = 1, \dots, q\}.$$

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. Linear program is not solvable.

Otherwise, construct the set

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

set $p = 1$ and continue.
Step 4. Compute

$$\min_{1 \le k \le m} \left\{ \frac{b_k}{a_{k,j_p}} \mid a_{k,j_p} > 0,\ b_k > 0 \right\} = \frac{b_h}{a_{h,j_p}}.$$

Step 5. If $\frac{b_{i_s}}{a_{i_s,j_p}} \le \frac{b_h}{a_{h,j_p}}$ then replace nonbasic and basic variables x_{N,j_p} and x_{B,i_s}, else go to *Step 6.*
Step 6. If $p > t$ replace x_{N,j_p} and $x_{B,h}$ and go to *Step 2.* Otherwise, put $p = p + 1$ and go to *Step 3.*

In the sequel we propose an improvement of *Algorithm 2.* Let us observe that in the real problems the matrix A is frequently sparse, so the number of needed nonzero coefficients is relatively small.

Algorithm 6. (The improvement of *Algorithm 2.*)

It is assumed that

$$a_{i,n+1} = b_i,\ i = 1, \dots, m, \quad a_{m+1,j} = c_j,\ j = 1, \dots, n, \quad a_{n+1,n+1} = d.$$

Step 1. Form the sets

$$V = \{a_{pl} | a_{pl} \neq 0,\ l = 1, \dots, n+1\},$$
$$K = \{a_{qj} | a_{qj} \neq 0,\ q = 1, \dots, m+1\}.$$

Step 2. Apply *Algorithm 2* only for a_{ql}, a_{qj} and a_{pl} satisfying $a_{pl} \in V$ and $a_{qj} \in K$.

We also introduce the next improvement of *Algorithm 1.* Instead dropping columns, we will mark these columns. More precisely, we introduce logical sequence *outc* with values $outc(i) = true$ if the ith column is not dropped, and $outc(i) = false$ otherwise. Similarly, *outr* is an indicator for rows.

Algorithm 7. (The improvement of *Algorithm 1.*)
Step 1. Set $outc(p) = true$, $p = 1, \dots, n$, and $outr(j) = true$, $j = 1, \dots, m$.
Step 2. If $J_i = \emptyset$, go to *Step 7.*
Step 3. If the pth constraint is equality, continue.
Step 4. Find $a_{pj} \neq 0$ such that $outc(j) = true$. (If there is not exists $a_{pj} \neq 0$ with $outc(j) = true$ and $b_p \neq 0$, the problem is not feasible; if $b_p = 0$, then we can drop the pth constraint and set $outr(p) = false$.)
Step 5. Put a_{pj} for the pivot element and perform *Algorithm 2.*
Step 6. Go to *Step 2.*
Step 7. Drop all columns and rows with $outc(p) = false$ and $outr(j) = false$, respectively. Set $outc(p) = true$ and $outr(j) = true$ for all p, j and perform *Algorithm 4.*

4. Numerical experience

Example 4.1. We tested the code $MarPlex$ on some *Netlib* test problems. We also compare our results with corresponding results produced by robust code PCx [2].

Problem	PCx	$MarPlex$	*Alg*5	*Alg*4	No5	No4
Adlittle	2.25494963×10^{5}	225494.963162	21	77	368	233
Afiro	-4.64753143×10^{2}	-464.753142	2	17	19	26
Agg	3.59917673×10^{7}	-35991767.286576	38	84	83	151
Agg2	-2.0239251×10^{7}	-20239252.3559776	31	52	223	140
Agg3	1.03121159×10^{7}	10312115.933596	51	141	300	256
Blend	-3.08121498×10^{1}	-30.812150	-	-	439	439
Lotfi	$-2.5264706062\times10^{1}$	-25.264706	111	339	559	771
Sc105	$-5.2202061212\times10^{1}$	-52.202061	-	-	59	59
Sc205	-5.22020612×10^{1}	-52.202061	-	-	172	172
Sc50a	$-6.4575077059\times10^{1}$	-64.575077	-	-	30	30
Sc50b	-7.000000000×10^{1}	-70	-	-	38	38
Scagr25	-1.47534331×10^{7}	-14753433.060769	185	290	695	1404
Scagr7	-2.33138982×10^{6}	-2331389.824330	69	89	125	186
Scorpion	1.87812482×10^{3}	1878.124822	66	118	162	179
Share2b	$-4.1573224074\times10^{2}$	-415.732241	92	123	167	215
Stocfor1	$-4.1131976219\times10^{4}$	-41131.976219	-	-	44	44
LitVera	1.999992×10^{-2}	0	-	-	1	1
Beaconfd	3.359249×10^{4}	33592.4858072	-	-	41	41
Israel	-8.966448×10^{5}	-896644.82	8	23	815	891
Kb2	-1.7499×10^{3}	-1749.9001299062	-	-	72	72
Recipe	-2.66616×10^{2}	-266.6160	-	-	32	32

Table 1.

From the first two columns in Table 1 we can see that our results are in accordance with the results obtained by PCx, in all problems beside the problem $LitVera$, taken from [4]. Note that our result for the problem $LitVera$ is quite correct. On the other side, code PCx achieves the near-optimal value 1.999992×10^{-2} in that case. In the next two columns we give the number of iterations needed for the construction of the first basis feasible solution. After the observation of these columns, it is easy to see that *Algorithm 5* is faster with respect to *Algorithm 4* in all cases. A streak in the corresponding position means that there are no iterative steps before the first basic feasible solution. By $No5$ and $No4$ we denote the complete number of iterations for *Algorithm 5* and *Algorithm 4*, respectively. As we can see, with respect to measures $No5$ and $No4$, *Algorithm 5* is faster with respect to *Algorithm 4* almost in all cases. Note that the maximal dimensions of problems presented in Table are 516×758 (in problems $Agg2$ and $Agg3$).

References

[1] B.D. Bounday, *Basic linear programming,* Edvard Arnold, Baltimore, 1984.

[2] J. Czyzyk, S. Mehrotra and S.J. Wright, *PCx User Guide,* Optimization Thechnology Center, Technical Report 96/01, 1996.

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

[4] V. Kovačević-Vujčić, *Ill conditiones and interior point method,* Technical Report 901-98, Laboratory of Operations Research, Faculty of Organizational Sciences, November 1998.

[5] M. Sakarovitch, *Linear programming,* Springer-Verlag, New York, 1983.

[6] S. Vukadinović, S. Cvejić *Mathematical programming,* Univerzitet u Prištini, Priština, 1996. (In Serbian)

(Stojković) University of Niš, Faculty of Economics,, Trg Kralja Aleksandra 11, 18000 Niš

University of Niš, Faculty of Science and Mathematics,, Višegradska 33, 18000 Niš

E-mail address, Stojković: nebojsas@orion.eknfak.ni.ac.yu

E-mail address, Petković: pecko@pmf.pmf.ni.ac.yu

E-mail address, Stanimirović: dexter_of_nis@yahoo.com