МАТЕМАТИЧКИ ВЕСНИК
54 (2002), 163–169

UDK 512.644
оригинални научни рад
research paper

A MODIFICATION OF REVISED SIMPLEX METHOD

Marko D. Petković, Predrag S. Stanimirović and Nebojša V. Stojković

Abstract. We introduce a modification of the revised simplex method, which accelerates the process of finding the first basic feasible solution in the two phases revised simplex method. We report computational results on numerical examples from Netlib test set.

1. Introduction

Consider the linear program in the general form

$$
\begin{aligned}
&\text{Maximize} && f(x) = f((x_N)_1,\dots,(x_N)_{n_1}) = \sum_{i=1}^{n_1} c_i (x_N)_i + d\\
&\text{subject to} && N_i^{(1)}: \ \sum_{j=1}^{n_1} a_{ij}(x_N)_j \le b_i, \quad i=1,\dots,r\\
& && N_i^{(2)}: \ \sum_{j=1}^{n_1} a_{ij}(x_N)_j \ge b_i, \quad i=r+1,\dots,s\\
& && J_i: \ \sum_{j=1}^{n_1} a_{ij}(x_N)_j = b_i, \quad i=s+1,\dots,m\\
& && (x_N)_j \ge 0, \quad j=1,\dots,n_1.
\end{aligned}
\tag{1.1}
$$

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

$$N_i^{(1)}: \quad \sum_{j=1}^{n_1} a_{ij}(x_N)_j + (x_B)_i = b_i, \quad i=1,\dots,r.$$

Also, we transform each inequality of the form $N_i^{(2)}$ (GE constraint) into an equality by subtracting a surplus variable $(x_B)_i$:

$$N_i^{(2)}: \quad \sum_{j=1}^{n_1} a_{ij}(x_N)_j - (x_B)_i = b_i, \quad i=r+1,\dots,s.$$

AMS Subject Classification: 90 C 05

Keywords and phrases: Linear programming, revised simplex method, MATHEMATICA

Communicated at the 5th International Symposium on Mathematical Analysis and its Applications, Niška banja, Yugoslavia, October, 2–6, 2002.

After these transformations we get the equivalent linear program in the standard form

$$\begin{aligned}
\text{Maximize}\quad & c_1x_1+\cdots+c_nx_n+d=c^Tx+d,\ c_i=0, i>n_1\\
\text{subject to}\quad & Ax=b,\\
& b=(b_1,\dots,b_m),\\
& x=((x_N)_1,\dots,(x_N)_n,(x_B)_1,\dots,(x_B)_s),\\
& (x_N)_j\ge 0,\ j=1,\dots,n,\ (x_B)_i\ge 0,\ i=1,\dots,s,
\end{aligned}\tag{1.2}$$

where the real matrix A is of order $m\times n$, for $n=n_1+s$.

The paper is organized as follows. In the second section we describe two-phases revised simplex method which is analogous with corresponding two-phases simplex method from [3], [4], [6] and [7]. In the third section an algorithm is described for finding the first basic feasible solution. In the last section we report some numerical experience with several Netlib test problems.

2. The revised simplex method

By A_i we denote i-th column of A, $i=1,\dots,n$. We suppose that matrix A is of full rank ($\operatorname{rank}(A)=m$), so there exists a base $\{A_{i_k}, k=1,\dots,m\}$ in $\mathbb{R}^m$. The matrix $B=[A_{i_1},\dots,A_{i_m}]$ containing corresponding columns from A is called the basic matrix. Variables $x_{i_1},\dots x_{i_m}$ are called basic variables, and a vector containing basic variables is denoted by x_B. The $m\times(n-m)$ matrix containing columns from A corresponding to nonbasic variables is denoted by N, and the $(n-m)$-vector containing nonbasic variables is denoted by x_N. Columns T_i of Tucker table T are representations of vector A_i in the base B, i.e. $T=B^{-1}N$. Here the objective function is represented as equality $c_1x_1+\cdots+c_nx_n+d=-x_c$ where x_c is a basic variable representing a value of the objective function. After corresponding modifications of the corresponding algorithm from [3], [4], [6], [7], we state the following two phases revised simplex algorithm. In this algorithm the notation A^i means i-th row of the matrix A.

Algorithm 1. (Revised simplex method for basic feasible solution)

Step S1A. Reconstruct the vector $b_T=B^{-1}b$, Tucker table $T=B^{-1}N$ and the objective function $c_T=T^{n+1}$.

Step S1B. If $(c_T)_1,\dots,(c_T)_n\le 0$, then the basic solution is an optimal solution.

Step S1C. Choose an arbitrary $(c_T)_j>0$.

Step S1D. If $t_{1j},\dots,t_{mj}\le 0$, stop the algorithm. Maximum is $+\infty$.

Otherwise, go to the next step.

Step S1E. Compute

$$\min_{1\le i\le m}\left\{\frac{(b_T)_i}{t_{ij}}\ \middle|\ a_{ij}>0\right\}=\frac{(b_T)_p}{t_{pj}},$$

replace the nonbasic and basic variables $(x_N)_j$ and $(x_B)_p$, respectively, and go to Step S1A.

If the condition $(b_T)_1, \dots, (b_T)_m \geq 0$ is not satisfied, we use the following algorithm to search for the first basic feasible solution. Algorithm in this version is not found in the literature.

ALGORITHM 2. (Find the first basic feasible solution).

Step S1. Reconstruct the vector $b_T = B^{-1}b$ and Tucker table $T = B^{-1}N$.

Step S2. Select the last $(b_T)_i < 0$.

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

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

$$\min_{k>i} \left(\left\{ \frac{(b_T)_i}{t_{ij}} \right\} \cup \left\{ \frac{(b_T)_k}{t_{kj}} \;\middle|\; t_{kj} > 0 \right\} \right) = \frac{(b_T)_p}{t_{pj}},$$

replace the nonbasic variable $(x_N)_j$ and the basic variable $(x_B)_p$ and go to Step S1. We use the last $t_{ij} < 0$.

3. Modification of algorithm for finding first basic solution

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

1. If $p = i$ and if there exists an index $y < i = p$ such that

$$\frac{(b_T)_y}{t_{yj}} < \frac{(b_T)_p}{t_{pj}}, \quad (b_T)_y > 0,\, t_{yj} > 0,$$

in the next iteration $(x_B)_y$ becomes negative:

$$(x_B)_y^1 = (b_T)_y^1 = (b_T)_y - \frac{(b_T)_p}{t_{pj}} t_{yj} < (b_T)_y - \frac{(b_T)_y}{t_{yj}} t_{yj} = 0.$$

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

$$\frac{(b_T)_p}{t_{pj}} < \frac{(b_T)_i}{t_{ij}} \Rightarrow (b_T)_i^1 = (b_T)_i - \frac{(b_T)_p}{t_{pj}} t_{ij} < 0.$$

We can find solution of this problem in two ways.

For fixed $(b_T)_i$ try to find $t_{ij} < 0$ such that

$$\min_{k} \left(\left\{ \frac{(b_T)_i}{t_{ij}} \;\middle|\; t_{ij} < 0 \right\} \cup \left\{ \frac{(b_T)_k}{t_{ij}} \;\middle|\; t_{kj} > 0,\ (b_T)_k > 0 \right\} \right) = \frac{(b_T)_i}{t_{ij}}. \tag{3.1}$$

In this case, it is possible to choose t_{ij} for the pivot element and obtain a new solution with a smaller number of negative coordinates $(b_T)_i$ in B_T.

This way is suitable for the basic simplex but not for the revised simplex because we need to reconstruct columns corresponding to all $t_{ij} < 0$.

In the second one, the main idea is to find a minimum

$$\frac{(b_T)_p}{t_{pj}} = \min \left\{ \frac{(b_T)_k}{t_{kj}} \;\middle|\; (b_T)_k < 0, t_{kj} < 0, k = 1, \dots, m \right\}$$

and if relation (3.1) is valid then choosing $(b_T)_p > 0$ as a pivot element we obtain a new solution with a smaller number of negative coordinates $(b_T)_i$.

In order to do that, we propose a modification of *Step S4*. This modification follows from the following lemma.

LEMMA. *Let the problem* (1.2) *be feasible and let x be the basic infeasible solution with q negative coordinates. Then there exists $t_{ij} < 0$. Also, in the following two cases:*

a) $q = m$,

b) $q < m$ *and*

$$Neg = \left\{ \frac{(b_T)_k}{t_{kj}} \;\middle|\; (b_T)_k < 0, t_{kj} < 0, k = 1, \dots, m \right\},$$

$$Pos = \left\{ \frac{(b_T)_k}{t_{kj}} \;\middle|\; (b_T)_k > 0, t_{kj} > 0, k = 1, \dots, m \right\},$$

$$\min\left(Neg \cup Pos\right) = \frac{(b_T)_r}{t_{rj}} \in Neg \tag{3.2}$$

it is possible to produce the new basic solution with at most $q-1$ negative coordinates in only one iterative step of the revised simplex method, if we choose t_{rj} for the pivot element, i.e. if we replace nonbasic variable $(x_N)_j$ by the basic variable $(x_B)_r$.

Proof. Consider an arbitrary $(b_T)_i < 0$. If the condition $t_{ij} > 0$ is satisfied for all $j = 1, \dots, n$, then

$$-(x_B)_i = t_{i1}(x_N)_1 + \cdots + t_{im}(x_N)_m - (b_T)_i > 0.$$

Hence, for $(x_N)_1 > 0, \dots, (x_N)_n > 0$ we have $(x_B)_i < 0$. That is a contradiction because problem (1.2) is feasible.

We now prove the second part of theorem.

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

$$(x_B)_i^1 = (b_T)_i^1 = \frac{(b_T)_i}{t_{ij}} > 0.$$

b) Assume now that the conditions $q < m$ and (3.2) are satisfied. Choose t_{rj} for the pivot element.

For $(b_T)_k > 0$ and $t_{kj} < 0$ it is obvious that

$$(b_T)_k^1 = (b_T)_k - \frac{(b_T)_r}{t_{rj}} t_{kj} > (b_T)_k \geq 0.$$

For $(b_T)_k > 0$ and $t_{kj} > 0$, using $\frac{(b_T)_k}{t_{kj}} \geq \frac{(b_T)_r}{t_{rj}}$, we conclude immediately

$$(b_T)_k^1 = (b_T)_k - \frac{(b_T)_r}{t_{rj}} t_{kj} \geq 0.$$

Hence, all positive $(b_T)_k$, remain positive. Moreover, for $(b_T)_r < 0$ we get

$$(b_T)_r^1 = \frac{(b_T)_r}{t_{rj}} \geq 0,$$

which completes the proof. ∎

REMARK 3.1. Let the condition (3.1) is not satisfied, $\min(Neg \cup Pos) = \frac{(b_T)_r}{t_{rj}} \in Neg$. If we choose t_{rj} for the pivot element we will also have

$$(b_T)_k^1 = (b_T)_k - \frac{(b_T)_r}{t_{kj}} t_{rj} \geq 0,$$

for $(b_T)_k > 0$ and t_{rj} both positive or negative. But for negative $(b_T)_k > 0$ we can prove similarly that

$$(b_T)_k^1 = (b_T)_k - \frac{(b_T)_r}{t_{kj}} t_{rj} < 0,$$

for both t_{rj} positive or negative. Finally new solution has the same number of negative coordinates q.

ALGORITHM 3. (Modification of *Algorithm 2*).

Step SM1. Let B is a basic and N is a non-basic matrix. Reconstruct vector $b_T = B^{-1}b$.

Step SM2. Construct the set

$$B = \{(b_i)_1, \dots, (b_i)_q\} = \{(b_i)_k \mid (b_i)_k < 0,\ k = 1, \dots, q\}.$$

Step SM3. Select $(b_i)_s < 0$.

Step SM4. Reconstruct i_s-th row. If $t_{i_s,1}, \dots, t_{i_s,n} \geq 0$ then STOP. Linear program is not solvable.

Otherwise, choose first $t_{i_s,j} \leq 0$.

Step SM5. Reconstruct j-th column T_j. Compute

$$\min_{1 \leq i \leq n} \left\{ \frac{(b_T)_i}{t_{ij}} \,\middle|\, (b_T)_i t_{ij} > 0, i = 1, \dots, m \right\} = \frac{(b_T)_p}{t_{pj}},$$

replace nonbasic and basic variables $(x_N)_j$ and $(x_B)_p$ and go to Step SM2.

4. Implementation and numerical experience

In the previous algorithms the reconstruction of the Tucker table is based on the matrix inversion. For large numerical examples this method is too slow and numerically unstable. Moreover, the matrix inversion should be avoided, since we require only few rows and columns of Tucker table. In the sequel we will show how to reconstruct a simple row or column of Tucker table using linear equation solver. It is well known that it is much more efficient to solve a linear system than to invert a matrix [2]. Reconstruction of i-th column T_i is based on the application of the `MATHEMATICA` function *LinearSolve* to solve the system $BT_i = A_i$ with respect to unknown variables in the vector T_i. To find a row T^i we need to know $(B^{-1})^i$, so:

$$T^i = (B^{-1})^i N.$$

Vector $(B^{-1})^i$ can be found from $B^{-1}B = I$, which implies $(B^{-1})^i B = I^i$. If we apply the transpose to the left- and right-hand side we get: $B^T((B^{-1})^i)^T = (I^i)^T$. This method is incorporated in the following algorithm :

ALGORITHM 4. (Reconstruction of row T^i of Tucker table).

Step 1. Let B be a basic, N a non-basic, and I the identity matrix. Solve the system

$$B^T((B^{-1})^i)^T = (I^i)^T$$

with respect to unknown variables contained into the vector $((B^{-1})^i)^T$.

Step 2. Compute and return $T^i = (B^{-1})^i N$.

The revised simplex method and the modification are implemented in the program called RevMarPlex, written in programming language `MATHEMATICA` [8], and tested on some Netlib test problems. Results are compared with robust LP solver PCx.

From the columns two and three in Table 1 we can see that our results are in accordance with the results obtained by PCx. In the columns four and five of Table 1 we give the number of iterations needed for the construction of the first basis feasible solution, using *Algorithm 3* and *Algorithm 2*, respectively.

Table 1.

After observing these columns, it is easy to see that *Algorithm 3* is faster than *Algorithm 2* in all cases. In some of them, there were no iterative steps before the first basic feasible solution. Table cells corresponding to these cases are marked by a streak.

Also, RevMarPlex solved a class of extremely ill-conditioned test examples, taken from [1], [5], which PCx is unable to solve (it ends with UNKNOWN status). These test problems are avaiable on the web page `www.psmtmath.s5.com`. Results are presented in Table 2. Let us observe that numbers of iterations of Algorithm 2 and Algorithm 3 were the same in all cases.

Table 2.

Program $RevMarPlex$ (by means of `MATHEMATICA` function $LinearSolve$) indicates that matrices of these examples are ill conditioned (with the condition number between 10^{15} and 10^{20}). Nevertheless $RevMarPlex$ solved these examples giving solutions close to optimal.

REFERENCES

[**1**] M.D. Ašić and V.V. Kovačević-Vujčić, *Ill-conditionedness and interior-point methods*, Univ. Beograd Publ. Elektrotehn. Fak., **11** (2000), 53–58.

[**2**] M.A. Bhatti, *Practical optimization with MATHEMATICA applications*, Springer Verlag Telos, 2000.

[**3**] B.D. Bounday, *Basic Linear Programming*, Edvard Arnold, Baltimore, 1984.

[**4**] J.P. Ignizio, *Linear Programming in Single-multiple-objective Systems*, Englewood Cliffs: Prentice Hall, 1982.

[**5**] V.V. Kovačević-Vujčić, and M.D. Ašić, *Stabilization of interior-point methods for linear programming*, Computational Optimization and Applications, **14** (1999), 1–16.

[**6**] M. Sakarovitch, *Linear Programming,* Springer-Verlag, New York, 1983.

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

[**8**] S. Wolfram, *Mathematica Book, Version 3.0*, Wolfram Media and Cambridge University Press, 1996.

(received 12.12.2002)

M. D. Petković, P. S. Stanimirović, University of Niš, Department of Mathematics, Faculty of Science, Višegradska 33, 18000 Niš, Yugoslavia

N. V. Stojković, University of Niš, Faculty of Economics, Trg Kralja Aleksandra 11, 18000 Niš, Yugoslavia

E-mail: dexter_of_nis@inmail.sk, peckois@ptt.yu, nebojsas@orion.eknfak.ni.ac.yu