Journal of Computational Mathematics, Vol.17, No.3, 1999, 233–242.

A NEW PERTURBATION SIMPLEX ALGORITHM FOR LINEAR PROGRAMMING*[1)]

Ping-qi Pan
(*Department of Applied Mathematics, Southeast University, Nanjing 210096, China*)

Abstract

In this paper, we first propose a perturbation procedure for achieving dual feasibility, which starts with any basis without introducing artificial variables. This procedure and the dual simplex method are then incorporated into a general purpose algorithm; then, a modification of it using a perturbation technique is made in order to handle highly degenerate problems efficiently. Some interesting theoretical results are presented. Nmerical results obtained are reported, which are very encouraging though still preliminary.

Key words: Linear programming, Simplex method, Perturbation, Dual feasibility.

1. Introduction

The dual simplex algorithm[1, 9] and the primal-dual simple algorithm [6] are well-known and efficient simplex variants. However, both of them need an initial dual feasible basis to get started, and therefore can not be directly applied to solving problems that do not have such an explicit basis. A number of schemes have been suggested to achieve dual feasibility [1, 5, 17, 18]. Some of them construct the dual analogues of the artificial-variable techniques, and none of them is as easy to implement computationally as the classical Phase-1 procedure of the primal simplex algorithm. As a result, either the dual or the primal-dual simplex algorithm is usually used only in some special cases in practice.

On the other hand, degeneracy is, in our view, all along a headache for simplex variants, including the dual and the primal-dual simplex algorithms. In practice, degeneracy occurs frequently and degrades their computational performance even thougth hardly leading to cycling. Consequently, various anti-degeneracy techniques of differing flavors have arised since the early days of linear programming. Dantzig (with his students) [4] and Charnce [2] first applied perturbation strategy to resolving degeneracy. Since then methods of this type have been proposed by, among others, Wolfe [19], Benichou, Gauthier, Hentges and Ribiere [3], Harris [8], and Gill, Murray, Saunders and Wright [7].

The distinguished features of our perturbation approach are as follows:

* Received February 27, 1995.
[1)] This work was supported by the National Science Foundation of China. No. 19271038

(a) It applies perturbation to solving the problem itself as well as dealing with degeneracy, via a "partially revised" scheme in a naturally combined manner.

(b) It perturbs the right-hand side or the relative price row only (if necessary), and makes no change to either bounds of variables or pivot rules.

(c) The amount of perturbation can be large.

(d) No additional storage is needed.

(e) It starts from any initial basis without introducing artificial variables.

This paper is organized as follows. In Section 2, we first describe the perturbation procedure for generating a starting point for the dual or the primal-dual simplex algorithms, which starts with any basis without introducing artificial variables. In Section 3, this procedure and the dual simplex method are incorporated into a general two-phase algorithm, which is then modified through perturbation in order to handle not only usual but highly degenerate problems efficiently. Some interesting theorems concerning the perturbation approach are as well given. Finally, in Section 4, computational results are reported, which are very encouraging though still preliminary.

2. Achieving Dual Feasibility

Consider linear programming problem in the standard form:

$$\max\ z = cx \tag{2.1a}$$

$$\text{s.t. } Ax = b \tag{2.1b}$$

$$x \geq 0, \tag{2.1c}$$

where $m < n$, $A \in R^{m\times n}$ with rank $(A) = m$, $b \in R^m$, and c and x are row and column n-vectors, respectively.

Put linear system (2.1b) into the following tableau:

$$\begin{array}{c|c} -c & 0 \\ \hline A & b \end{array} \tag{2.2}$$

Suppose that an initial *simplex* tableau of the preceding presents:

$$\begin{array}{c|c} \bar{c} & \bar{z} \\ \hline \bar{A} & \bar{b} \end{array} \tag{2.3}$$

where $\bar{A} \in R^{m\times n}$, $\bar{b} \in R^m$, and for each $i = 1, \cdots, m$, the j_i-th column $\bar{a}_{j_i}$ of $\bar{A}$ is the identity vector with the i-th component 1. Then, x_{j_i}, $i = 1, \cdots, m$ are the related *basic set* of variables. We denote by J_B the set of indices of basic variables, and take the symbol:

$$\bar{J}_B = \{1, \cdots, n\} \backslash J_B. \tag{2.4}$$

Usually, tableau (2.3) is neither primally nor dually feasible, i.e., the row index set

$$I = \{i | \bar{b}_i < 0\} \tag{2.5}$$

and the index set

$$J = \{j | \bar{c}_j < 0,\ j \in \bar{J}_B\}, \tag{2.6}$$

are both nonempty. We shall describe a procedure for reaching dual feasibility.

It is simple matter to change all $\bar{b}_i (i \in I)$ into some predetermined positive numbers δ_i, by resetting

$$\bar{b}_i := \delta_i, \quad \forall i \in I, \tag{2.7}$$

which amounts to respectively adding quantities

$$\beta_i = \delta_i - \bar{b}_i \qquad i \in I. \tag{2.8}$$

It is evident that the modified tableau, say (2.3) again, is now primally feasible, and the basic technique of the primal simplex method is immediately applicable to solving such a modified problem. Suppose now that variables x_k and x_{j_l} are selected under some rule, e.g., Dantzig's original rule, to enter and to leave the basic set respectively, and that the basis change results in the new simplex tableau below:

$$\begin{array}{c|c} \tilde{c} & \tilde{z} \\ \hline \tilde{A} & \tilde{b} \end{array} \tag{2.9}$$

where

$$\tilde{z} = \bar{z} - (\bar{b}_l / \bar{a}_{lk}) \bar{c}_k \tag{2.10a}$$

$$\tilde{c}_j = \bar{c}_j - (\bar{a}_{lj} / \bar{a}_{lk}) \bar{c}_k, \quad (j = 1, \cdots, n) \tag{2.10b}$$

$$\tilde{b}_i = \begin{cases} \bar{b}_i - (\bar{b}_l / \bar{a}_{lk}) \bar{a}_{ik}, & (i = 1, \cdots, m;\ i \neq l) \\ \bar{b}_l / \bar{a}_{lk}, & (i = l) \end{cases} \tag{2.10c}$$

$$\tilde{a}_{ij} = \begin{cases} \bar{a}_{ij} - (\bar{a}_{lj} / \bar{a}_{lk}) \bar{a}_{ik}, & (i = 1, \cdots, m;\ i \neq l) \\ \bar{a}_{lj} / \bar{a}_{lk} & (i = l) \end{cases} \tag{2.10d}$$

Then, one simplex step has completed. Two tableaus are said to be *equivalent* if one can be obtained from the other in finitely many iteration steps, or in other words, the two canonical systems, represented by them, are equivalent.

We conduct such steps until reaching an optimal tableau, or detecting upper unboundedness. Suppose that the termination occurs at some tableau, say (2.9), which is clearly equivalent to the modified but not the initial tableau. Since the canonical system, represented by the modified tableau (2.3), can be obtained by making the variable transformation $x_{j_i} := x_{j_i} - \beta_i$ for all $i \in I$ on the system, represented by the initial tableau, and the former system is equivalent to the one, represented by (2.9), it can be said that making on (2.9)'s system the inverse transformation: $x_{j_i} := x_{j_i} + \beta_i$ for all $i \in I$ results in a system which is equivalent to that represented by the initial tableau. Therefore, a tableau that is equivalent to the initial one can be obtained from (2.9) by the following operations: for all x_{j_i}, $i \in I$ that are basic, subtracting β_i from the corresponding components of the right-hand side of (2.9), respectively; and for each nonbasic one, subtracting from the right-hand side the vector, obtained by multiplying the corresponding column of the left-hand side of (2.9) by the associated β_i.

However, the preceding scheme may lead to unacceptable loss of correct significant digits, in particular for large problems. This difficulty can be overcome by using a so-called *partially revised* scheme, in which the inverse of the basis is utilized to generate

directly from the the original A just the information required for pivoting without computing and recording the whole new tableau, as is done in the revised simplex method, except $\tilde{z}, \tilde{c}$ and $\tilde{b}$ are still calculated via (2.10a,b,c). Such a trick is advantageous as a sensitivity analysis may restore the desired tableau simply after the termination of the process, as demonstrated below.

Let $B \in R^{m\times m}$ be the basis, associated with a current simplex tableau, say (2.3), i.e, $B = (a_{j_1}, \cdots, a_{j_m})$, where a_{j_i}, $i = 1, \cdots, m$, are the columns of A, corresponding to the basis variables x_{j_i}, and let B^{-1} be its inverse. Then the well-known formula

$$\tilde{B}^{-1} := B^{-1} + \frac{(e_l - B^{-1}a_k)e_l^T B^{-1}}{e_l^T B^{-1} a_k}, \tag{2.11}$$

can be used to compute the inverse of the new basis, associated with the tableau (2.9), where e_l is the identity column m-vector with the l-th component 1. Thus we can put (2.9) into the *partially revised* tableau

$$\begin{array}{c|c} \tilde{c} & \tilde{z} \\ \hline \tilde{B}^{-1}A & \tilde{b} \end{array} \tag{2.12}$$

where $\tilde{z}, \tilde{c}$ and $\tilde{b}$ are defined by (2.10a,b and c), respectively. Suppose that the iteration process terminates at the tableau (2.12), and that $c_{\tilde{B}}$ is the row m-vector consisting of the price coefficients corresponding to the basic variables. Since any modification of the original right-hand side affects the final value column only, a direct way to restore the desired tableau is to reset $\tilde{z} := c_{\tilde{B}}\tilde{B}^{-1}b$ and $\tilde{b} := \tilde{B}^{-1}b$.

3. The Algorithms

Let us examine what can be said about the restored simplex tableau. If the unrestored tableau, say (2.12), is optimal for the modified problem, the dual feasibility of the original problem is then achieved after the restoration because what has been changed is its right-hand side only. Suppose now that the unrestored (2.12) indicates upper unboundedness of the modified problem, and the last chosen index is k, i.e., we have

$$(\tilde{B}^{-1}a_k)_i \le 0, \qquad \forall i = 1, \cdots, m. \tag{3.1}$$

Clearly, the restored tableau will still indicate the upper unboundedness of the original problem if all the strict inequalities in (3.1) hold. This is always true except when the following conditions

$$(\tilde{B}^{-1}a_k)_{\tilde{i}} = 0 \qquad \text{and} \qquad (\tilde{B}^{-1}b)_{\tilde{i}} < 0 \tag{3.2}$$

are satisfied for some row index $\tilde{i}$. When (3.2) holds, it can still be asserted that there exists no optimal solution, as (3.1) clearly indicates dual infeasibility of the program.

We may incorporate the primal procedure described in Section 2 as phase-1 into a general algorithm, in which either the dual or the primal-dual simplex procedure can be taken as phase-2. Let us describe a model, in which the dual simplex procedure is utilized:

Algorithm 1. Let $B^{-1} \in R^{m\times m}$ be the inverse of an initial basis. Given constants $\delta_i > 0$, $i = 1, \cdots, m$. This algorithm solves linear program (2.1).

1. Set $ia = 0$ and $ib = 0$.
2. Compute

$$\bar{z} = c_B B^{-1} b \tag{3.3a}$$

$$\bar{c} = -c + c_B B^{-1} A \tag{3.3b}$$

$$\bar{b} = B^{-1} b \tag{3.3c}$$

3. If row index set I defined by (2.5) is nonempty, reset according to (2.7) and set $ib = 1$.
4. Determing an index k such that

$$\bar{c}_k = \min\{\bar{c}_j | j \in \bar{J}_B\} \tag{3.4}$$

5. If $\bar{c}_k \geq 0$, then: (i) stop if $ib = 0$; (ii) restore $\bar{z}$ and $\bar{b}$ by (3.3a) ane (3.3c), respectively; (iii) set $ia = 1$, and then go to step 10.
6. Stop if the row index set below is empty:

$$I' = \{i \mid (B^{-1}a_k)_i > 0, \quad i = 1, \cdots, m\} \tag{3.5}$$

7. Determing a row index l such that

$$\bar{b}_l/(B^{-1}a_k)_l = \min\{\bar{b}_i/(B^{-1}a_k)_i | \quad i \in I'\} \tag{3.6}$$

8. Update B^{-1} by (2.11), and $\bar{z}, \bar{c}$ and $\bar{b}$ by (2.10a), (2.10b) and (2.10c), respectively.
9. If $ia = 0$, go to step 4.
10. Determing l such that

$$\bar{b}_l = \min\{\bar{b}_i | i = 1, \cdots, m\} \tag{3.7}$$

11. Stop if $\bar{b}_l \geq 0$.
12. Stop if the index set : $J' = \{j | (B^{-1}a_j)_l < 0,\ j \in \bar{J}_B\}$ is empty.
13. Determing k such that

$$\bar{c}_k/(B^{-1}a_k)_l = \max\{\bar{c}_j/(B^{-1}a_j)_l | j \in J'\} \tag{3.8}$$

14. Go to step 8.

Note: The primal phase-1 procedure consists of steps 2 through 8, and the dual phase-2 steps 8 through 14.

Based of the well-known properties of the primal and the dual simplex methods, and discussions made prior to the above algorithm, we conclude:

Theorem 2. *Assuming primal and dual nondegeneracy, Algorithn* 1 *terminates at either*

(*a*) *step* 5(*i*) *or* 11, *with an optimal solution of* (2.1) *reached; or* (*b*) *step* 6, *indicating upper unboundedness of the program; or* (*c*) *step* 12, *indicating infeasibility of it.*

It is possible to gain more via a sensitivity analysis:

Lemma 3. *Let T_1 and T_2 be two equivalent tableaus and let X be a subset of variables which are basic for both tableaus. For each basic variable in X, let the associated components of two right-hand sides of T_1 and of T_2 correspond to each other. Then adding any same set of real numbers respectively to corresponding components results in two tableaus that are also equivalent.*

Proof. Without loss of generality, let the X be

$$X = \{x_i \mid i = 1, \cdots, r\}, \tag{3.9}$$

where $r \leq m$. Suppose that the tableau T_1 (without the price row) is

$$\begin{array}{cc|c} I_r & D_r & b'_r \\ \hline 0 & D & b' \end{array} \tag{3.10}$$

where I_r is the $r \times r$ identity matrix, 0 is the $(m-r) \times r$ zero matrix, and $D_r \in R^{r \times (n-r)}$, $D \in R^{(m-r) \times (n-r)}$, $b'_r \in R^r$ and $b' \in R^{m-r}$, and that the tableau T_2 is

$$\begin{array}{cc|c} I_r & E_r & b''_r \\ \hline 0 & E & b'' \end{array} \tag{3.11}$$

Denote by B^{-1} the inverse, left-multiplying by which the two sides of T_1 leads to T_2. So, if holds that

$$\begin{pmatrix} b''_r \\ \cdots \\ b'' \end{pmatrix} = B^{-1} \begin{pmatrix} b'_r \\ \cdots \\ b' \end{pmatrix} \tag{3.12}$$

If can be shown that B^{-1} is of the following form:

$$B^{-1} = \left(\begin{array}{c|c} I_r & F_r \\ \hline 0 & F \end{array}\right) \tag{3.13}$$

Denote by $\tilde{T}_1$ the tableau resulting from T_1 by adding some real vector $\Delta b_r \in R^r$ to b'_r. Then left-multiplying by B^{-1} the two sides of $\tilde{T}_1$ yields an equivalent tableau of it. The left-hand side of the resulting tableau is as the same as that of T_2, but the right-hand side is

$$B^{-1} \begin{pmatrix} b'_r + \Delta b_r \\ \cdots\cdots \\ b' \end{pmatrix} = B^{-1} \begin{pmatrix} b'_r \\ \cdots \\ b' \end{pmatrix} + B^{-1} \begin{pmatrix} \Delta b_r \\ \cdots \\ 0 \end{pmatrix} = \begin{pmatrix} b''_r \\ \cdots \\ b'' \end{pmatrix} + \begin{pmatrix} \Delta b_r \\ \cdots \\ 0 \end{pmatrix} \tag{3.14}$$

where 0 is the zero $(m-r)$-vector, and the last equality results from (3.12) and (3.13). Thus, the theorem is proved.

Theorem 4. *Assume that the original linear program* (2.1) *has a dually nondegenerate optimal tableau. If only those components of the right-hand side of the initial tableau are modified in step* 3*, for which the associated basic variables are also basic for the optimal tableau, then, achieving optimality, Algorithm* 1 *terminates at step* 11 *immediately after the execution of the primal phase*-1 *if primal nondegeneracy is assured.*

Proof. Let T_2 be a dually nondegenerate optimal tableau of (2.1) and let β_i be positive numbers, defined by (2.8). Denote by T_1 the initial tableau, and $\tilde{T}_1$ the modified one from T_1 by adding respective numbers β_i to the components of the right-hand side, associated with those variables which is also basic for the optimal tableau. According to Lemma 3, adding the same set of positive numbers to the corresponding components of the right-hand side of T_2 must result in $\tilde{T}_1$'s equivalent tableau, say $\tilde{T}_2$, which is optimal for the modified program since the values of according components are strictly increased by $\beta_i's$. Thus, under the primal nondegeneracy assumption, phase-1 must terminate at some optimal tableau of the modified program, say $\hat{T}_2$; and under the dual nondegeneracy assumption on T_2 and hence $\tilde{T}_2$, the two basic solutions, related to $\hat{T}_2$ and to $\tilde{T}_2$, are equal. Therefore, subtracting the positive numbers from corresponding components of the value column of $\hat{T}_2$ results in a nonnegative right-hand side again, an equivalence to that restored in step 5(ii). It is thus evident from Lemma 3 that an optimal tableau of (2.1) presents, and hence the algorithm terminates at step 11 immediately.

Geometrically, the effect of the modifications can be viewed as the violated nonnegative restrictions being relaxed. Obviously, if all the relaxed restrictions are not active (or binding) at an optimal solution, no change to optimal solution will happen. So, in this case adding any positive quantities to those components of the initial right-hand side that are associated with optimally basic variables does not interfere with our purpose. On the other hand, it is seen from (3.14) that the modifications usually interfere if some of the associated variables are optimally nonbasic; nevertheless, such interference may not be serious if the added β_i are small:

Remark 5. Assume that primal phase-1 procedure of Algorithm 1 produces an optimal tableau for the modified program. If this tableau is primally nondegenerate and the modifications are small enough, then the algorithm terminates at step 11 immediately.

It is clear however that quantities β_i are essentially uncontrollable, and can be very large. So, the dual phase-2 procedure is needed in the general purpose algorithm.

Like the classical primal and dual simplex algorithms, Algorithm 1 has no defences against degeneracy. Therefore, it might be better to modify it to handle not only usual but highly degenerate problems. Obviously, there is no reason to relax the violated restrictions only: why do not relax as well the nearly-violated restrictions, including those corresponding to degenerate basic variables? Why do not deal with the dual procedure similarly? This leads to the anti-degenerate variant of Algorithm 1 below:

Algorithm 6. Let B^{-1} be the same as in Algorithm 1. Given $\delta_i > 0$, $i = 1, \cdots, n$ and a small positive number $\varepsilon < \min\{\delta_i | i = 1, \cdots, n\}$. This algorithm solves linear program (2.1).

1. Set $ia = 0$, $ib = 0$ and $ic = 0$.

2. Compute $\bar{z}, \bar{c}$ and $\bar{b}$ by (3.3).

3. If index set $I = \{i | \bar{b}_i < \varepsilon, i = 1, \cdots, m\}$ is nonempty, reset $\bar{b}_i = \delta_i$, $\forall i \in I$ and set $ib = 1$.

4. Determing index k by (3.4).

5. If $\bar{c}_k \geq 0$, then: (i) stop if $ib = 0$; (ii) restore $\bar{z}$ and $\bar{b}$ by (3.3a) and (3.3c),

respectively; (iii) set $ia = 1, ib = 0$, and then go to step 10.

6. Stop if I', defined by (3.5), is empty.

7. Determing l by (3.6).

8. Update B^{-1} by (2.11), and $\bar{z}, \bar{c}$ and $\bar{b}$ by (2.10a), (2.10b) and (2.10c), respectively.

9. If $ia = 0$, go to step 3.

10. Determine row index l by (3.7).

11. If $\bar{b}_l \geq 0$, then (i) stop if $ic = 0$; (ii) restore $\bar{z}$ and $\bar{c}$ by (3.3a) and (3.3b), respectively; (iii) set $ia = 0, ic = 0$, and then go to step 3.

12. Stop if J', defined by (3.8), is empty.

13. If the index set $J = \{j | \bar{c}_j < \epsilon, j \in J'\}$ is nonempty, then reset $\bar{c}_j = \delta_j, \forall j \in J$ and set $ic = 1$.

14. Determing k by (3.8).

15. Set $ia = 1$, and go to step 8.

Comparing between Algorithms 1 and 6, we conclude that all the results stated previously in this section hold as well for the latter.

Moreover, Algorithm 6 has important features. It is noted that the modified tableau produced in step 3 is now guaranteed to be primally nondegenerate; and before taking on the dual simplex steps, after phase-1, the relative price row is modified so as the resulting tableau is also dually nondegenerate. There can also be an analogue of Lemma 3, associated with the latter modifications:

Lemma 7. *Let T_1 and T_2 be two equivalent tableaus and let X be a subset of variables that are nonbasic for both tableaus. Then adding any same set of real numbers to corresponding relative price coefficients results in two tableaus which are also equivalent.*

Proof. Suppose that N is the submatrix consisting of the columns of A, corresponding to the modified relative price coefficients, B is the basis related to T_1 and $\bar{B}$ the basis related to T_2. Then the validity of the statement comes from the forms of the corresponding relative price coefficients of the two tableaus, $c_N - c_B B^{-1} N$ and $c_N - c_{\bar{B}} \bar{B}^{-1} N$, where c_N is the row vector consisting of price coefficients corresponding to the columns of N.

In conjunction with Algorithm 6, we state an analogue to Theorem 4:

Theorem 8. *Assume that the original linear program* (2.1) *has a primally nondegenerate optimal tableau and that the end tableau of primal phase-1 is restored at step* 5(*ii*). *If, before carrying on dual phase-*2, *only those relative price coefficients of the restored tableau are modified for which the associated nonbasic variables are also nonbasic for the optimal tableau, Algorithm* 6 *terminates immediately at step* 5(*i*) *after the execution of dual phase-*2, *achieving optimality, if dual nondegeneracy is assured throughout.*

Clearly, the situation may become different if some of modified coefficients are associated with optimally basic variables. But a statement, similar to Remark 5, can also be made here; that is, the same thing holds if the added δ_i are small enough and the end tableau of phase-2 is dually nondegenerate. However, in contrast to the case of the former modifications, where added quantities are uncontrollable, the δ_i being added here is allowed to be small, fortunately. This may be explained geometrically: the

latter modifications amount to perturbing the gradient vector of the original objective function so that the edges at the reached vertex are all *strictly* decreasing ones.

As a safeguarding strategy, we designed the algorithm go back to its primal phase again if optimality of (2.1) is not yet achieved after the dual simplex steps taken, and modify all encountered degenerate entries, not only in the first but also in subsequent circles throughout.

We conclude this section with the following:

Remark 9. Algorithm 6 terminates if primal and dual degeneracy occur only finitely many times.

4. Computational Results

In order to gain an insight into the performance of the proposed approach, Algorithm 6 was coded into a FORTRAN 77 program, where the predetermined numbers used in Algorithm 6 were $\delta_j = 10^{-1}$, $\forall j = 1, \cdots, n$, and $\varepsilon = 10^{-3}$. We used Algorithm 6.5 of [11] to determine an initial basis.

Tested linear programming problems fall into 4 groups. The first consists of 62 problems with only inequality constraints and of up to 22 decision variables and constraints. The second includes 23 randomly produced problems with from 23 up to 80 decision variables and constraints. The third are 4 larger sparse problems. To see what will happen with Klee-Minty problems, the fourth group involves two such ones (see, for example, [16]).

In TABLE 1, in terms of number of pivot steps required, numerical results obtained are summarized, and compared with the revised two-phase simplex algorithm using Dantzig's original rule. Four problems of group 3 are designated by P1, P2, P3 and P4, and Klee-Minty problems by KM1 and KM2, respectively.

Table 1. Numerical Results

Problem	Algorithm 6	Classical	Ratio: C/A
Total for Group 1	240	800	3.33
Total for Group 2	1164	3209	2.76
$P1 : m = 27,\ n = 51$	9	27	3.00
$P2 : m = 28, n = 56$	4	38	9.50
$P3 : m = 55, n = 137$	46	170	3.70
$P4 : m = 56, n = 138$	67	172	2.57
Total for Group 3	126	407	3.23
$KM1 : n = 8$	5	255	51.00
$KM2 : n = 10$	5	1023	204.60

The preceding shows the correctness of the analysis made in Section 3, and points clearly to the excellence of our algorithm on these tested problems. We stress that it outperformed the classical simplex algorithm for each of these problems. It turns out that numerical results are not very sensitive to the predetermined amount of perturbation.

Such a good performance of Algorithm 6 might be partially due to its anti-degeneracy feature. Stalling phenomenon is not observed in the solution process. Our experience is that after modifications at step 3 or 13 in the first circle of the two phases, only a little

primal or dual degeneracy, if any, will occur in subsequent steps; all the problems are solved in the first circle. However, the situation may be different with highly degenerate and/or large problems, and further computational tests are expected.

Finally, seeing that the proposed perturbation simplex method has some very atractive features theoretically, and performs favorably in solving linear programming problems of such sizes as those tested, we conclude that it is certainly promising, and deserves further investigation.

References

[1] E.M.L. Beale, An alternative method for linear programming, in *Proceedings of Cambridge Philosophical Society,* **50** (1954), 513–523.

[2] A. Charnes, Optimality and degeneracy in linear programming, *Econometrica,* **20** (1952), 160–170.

[3] M. Benichou, J.M. Gauthier, G. Hentges, G. Ribiere, The effcient solution of large-scale linear programming problems, *Mathematical Programming,* **13** (1977), 280–322.

[4] G.B. Dantzig, Application of the simplex method to a transportation problem, in T.C. Koopman (ed.), Activity Analysis of Production and Allocation, John Wiley & Sons, Inc., New York, (1951), 359–373.

[5] G.B. Dantzig, The Dual Simplex Algorithm, RAND Report RM-1270, The RAND Corporation, Santa Monica, CA, 1954.

[6] G.B. Dantzig, L.R. Ford, D.R. Fulkerson, A Primal-Dual Algorithm, RAND Report RM-1709, The RAND Corporation, Santa Monica, CA, 1956.

[7] P.E. Gill, Walter Murray, M.A. Saunders, M.H. Wrighr, A practical anti-cycling procedure for linearly constrained optimization, *Mathematical Programming,* **45** (1989), 437–474.

[8] P.M.J. Harris, Pivot selection methods of the Devex LP code, *Mathematical Programming Study,* **4** (1975), 30–57.

[9] C.E. Lemke, The dual method of solving the linear programming problem, *Naval Research Logistics Quaterly,* **1** (1954), 36–47.

[10] P.-Q. Pan, Practical finite pivot rules for the simplex method, *OR Spektru,* **12** (1990), 219–225.

[11] P.-Q. Pan, A simplex-like method with bisection for linear programming, *Optimization,* **22**:5 (1991), 717–743.

[12] P.-Q. Pan, A variant of the dual pivot rule in linear programming, *Journal of Information & Optimization Sciences,* **15**:3 (1994), 405–413.

[13] P.-Q. Pan, The most-obtuse-angle row pivot rule for achieving dual feasibility: a computational study, *European Journal of Operations Research,* **101** (1997), 167–176.

[14] P.-Q. Pan, A dual projective simplex method for linear programming, *Computers and Mathematics with Applications,* **36** : 6 (1998), 119–135.

[15] P.-Q. Pan, A basis-dificiency-allowing variation of the simplex method, *Computers and Mathematics with Applications,* **36** : 3 (1998), 33–53.

[16] A. Schrijver, The Theory of Linear and Integer Programming, John Wiley & Sons, Chichester, 1986.

[17] D.I. Steinberg, On finding an initial solution for the dual simplex algorithm, *Decision Sciences,* **7**:1 (1976).

[18] S. Vajda, Mathematical Programming, Addson-Wesley Publishing Company Inc., Reading, MA, 1961.

[19] P. Wolfe, A technique for resolving degeneracy in linear programming, *Journal of the Society for SIAM,* **11** (1963), 205–211.