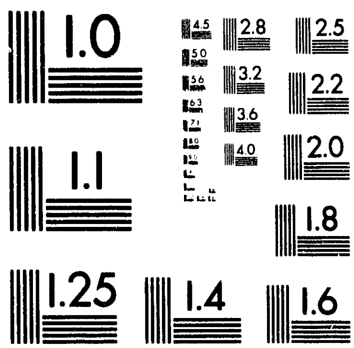
1.0
4.5
2.8
2.5
5.0
3.2
2.2
5.6
6.3
3.6
2.0
4.0
1.1
1.8
1.25
1.4
1.6

1 of 1

SYSTEMS OPTIMIZATION LABORATORY
DEPARTMENT OF OPERATIONS RESEARCH
STANFORD UNIVERSITY
STANFORD, CALIFORNIA 94305-4022

The Simplex Algorithm with a New Primal and Dual Pivot Rule

by
Hsin-Der Chen*, Panos M. Pardalos** and Michael A. Saunders†

TECHNICAL REPORT SOL 93-5

June 1993

*Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611. Partially supported by National Science Foundation Grant DDM-8814075.

**Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611. Partially supported by DIMACS under National Science Foundation Contract STC-91-19999.

†Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, CA 94305-4022. Partially supported by Department of Energy Grant DE-FG03-92ER25117, National Science Foundation Grant DDM-9204208, and Office of Naval Research Grant N00014-90-J-1242.

Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the above sponsors.

Also listed as Operations Research Department Technical Report 93-10. Reproduction in whole or in part is permitted for any purposes of the United States Government. This document has been approved for public release and sale; its distribution is unlimited.

The Simplex Algorithm with a New Primal and Dual Pivot Rule

Hsin-Der CHEN*, Panos M. PARDALOS* and Michael A. SAUNDERS†

Technical Report SOL 93-5
June 1993

Abstract

We present a simplex-type algorithm for linear programming that works with primal-feasible and dual-feasible points associated with bases that differ by only one column.

1. Introduction

Consider the following linear programming problem:

$$(P) \qquad \max\{c^T x \mid Ax = b,\ x \geq 0\},$$

where $A \in R^{m \times n}$ $(m \leq n)$ and the vectors b, c, x have appropriate dimension. Let B be a basis in A, so that $AQ = (\ B \ \ N\ )$ for some permutation Q, and let $Bx_B = b$, $B^T \pi_B = c_B$ and $r = c - A^T \pi$. An iteration of the primal simplex method [Dan63] can be briefly described as:

Given B such that $x_B \geq 0$, if $r \leq 0$, then stop; otherwise, update B.

The dual simplex method is:

Given B such that $r \leq 0$, if $x_B \geq 0$, then stop; otherwise, update B.

The solution in both cases is given by x_B and $x_N = 0$.

1.1. The Aim

We propose an algorithm that maintains two basic solutions for (P). One is primal feasible and the other dual feasible, and the bases differ by only one column. Some preliminary computational results have been obtained (see Section 8), showing promise for the algorithm relative to the primal simplex method.

*Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611. The first author was partially supported by the National Science Foundation Grant DDM-8814075; the second author was partially supported by DIMACS (a National Science Foundation Science and Technology Center, funded under contract STC-91-19999).

†Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, CA 94305-4022. Partially supported by Department of Energy Grant DE-FG03-92ER25117, National Science Foundation Grant DDM-9204208, and Office of Naval Research Grant N00014-90-J-1242.

1.2. Related Work

Many pivot rules are known for simplex-type algorithms. For example, a large number are described in [TZ92]. Under certain circumstances the proposed algorithm is equivalent to Lemke's method [Lem65], as discussed in Section 6, but otherwise the algorithm appears to be new.

2. The Main Feature

For problem (P), let B_P and B_D be bases in the primal and dual simplex methods respectively. Consider the special case in which B_P and B_D differ in exactly one column. Let d in B_D be the column of A that distinguishes B_P and B_D. Also let x_P and x_D be the associated primal-feasible and dual-feasible vertices of (P).

Theorem 2.1. *If x_P and x_D are not both optimal vertices, then either d can enter B_P in the next primal pivot, or d can leave B_D in the next dual pivot.*

Proof. Consider a modified problem (P') in which all variables that are in neither the primal basis nor the dual basis are fixed at zero. It is easy to see that x_P and x_D are also primal-feasible and dual-feasible vertic s of (P'). In this case, the variable corresponding to d is the only nonbasic variable (for B_P).

By definition of optimality, if x_P is not an optimal vertex for (P'), d can enter B_P in order to improve the primal objective value. Otherwise, if x_D is not an optimal vertex for (P'), d can leave B_D in order to improve the dual objective value.

Since all the data A, b, c, B_P and B_D are the same in (P') and (P), the result also applies to (P). ∎

Initialization is discussed later. To show that the algorithm continues, we must show that the new bases differ by one column in the next iteration.

Let $\bar{d}$ in B_P be the column that is not in B_D. According to the theorem, d may enter B_P in a primal pivot. If the leaving column is $\bar{d}$, then B_P and B_D are identical and the optimal vertex is reached; otherwise, B_P still differs from B_D in one column ($\bar{d}$ in B_P and the leaving column in B_D).

On the other hand, d may leave B_D in a dual pivot. In this case, if the entering column is $\bar{d}$, then we are done; otherwise, B_P and B_D still have $m-1$ columns in common.

The algorithm therefore proceeds in this way: the leaving column in a primal pivot is a candidate leaving column in the next dual pivot, and the entering column in a dual pivot is a candidate entering column in the next primal pivot.

Remarks

- There is no pricing step in the simplex algorithm with this pivot rule. The algorithm may be interpreted as a primal simplex method that uses a related dual simplex procedure to do pricing.

- Since B_D^{-1} is closely related to B_P^{-1}, we don't have to keep both of them. (We can maintain a factorization of just B_P.) Moreover, since there is no pricing step, we don't have to compute shadow prices (π) in the primal method. Similarly, we don't have to compute $B_D^{-1}b$ in the dual method. Therefore, the computational effort is the same as in the ordinary simplex method.

- Suppose that (x_P, x_D) is moved to (x'_P, x'_D) after an iteration (and both B_P and B_D are updated). If the primal step is degenerate, then $x'_P = x_P$; otherwise $c^T x'_P > c^T x_P$. If the dual step is degenerate, then $c^T x'_D = c^T x_D$; otherwise $c^T x'_D < c^T x_D$. Therefore, the duality gap will strictly decrease; that is, $c^T x'_D - c^T x'_P < c^T x_D - c^T x_P$, except in the very special case where both primal and dual degeneracies exist simultaneously.

- If B_P^{-1} or $B_P^{-1}A$ is sparse, as for example in network problems, the algorithm needs to compute only part of $B_P^{-1}b$. Also, since there is no pricing step, only some of the shadow prices have to be computed (to determine the entering column in the dual simplex step).

  In contrast, with minimum-cost network flow problems (for example), although only the flow values in the augmented path need to be updated at each iteration, conventional methods have to compute all the shadow prices (except with partial pricing).

3. An Example

Figure 1 provides a two-dimensional example. The superscript indicates the order in which vertices are visited in the algorithm: $x_P^1 \to x_D^2 \to x_P^3 \to x_D^4 \to x_P^5 \to x_D^6 \to x^*$.

Note that the algorithm updates both primal-feasible and dual-feasible solutions at each iteration. Therefore, it takes three iterations to solve the example. In fact, any primal simplex method starting from x_P^1 will solve this example in exactly three iterations.

4. A Variation

A vertex is adjacent to another vertex if the associated bases differ by one column. Here we suggest a possible variation of the algorithm:

> Given a primal-feasible vertex, move to the adjacent dual-feasible vertex with the minimal dual objective value. Then move to the best adjacent primal-feasible vertex, and so on.

The path of this variation algorithm in the example will be $x_P^1 \to x' \to x^*$.

To implement the variation, we must be able to find the appropriate adjacent vertices. Suppose that B is a basis for (P). Let $\hat{A}$, x_B, π and r be defined by $B\hat{A} = A$, $Bx_B = b$, $B^T\pi = c_B$ and $r = c - A^T\pi$. Let $S_+ = \{j \mid j \text{ is nonbasic and } r_j > 0\}$ and $S_- = \{j \mid j \text{ is nonbasic and } r_j \le 0\}$. If there exists some nonbasic variable j with

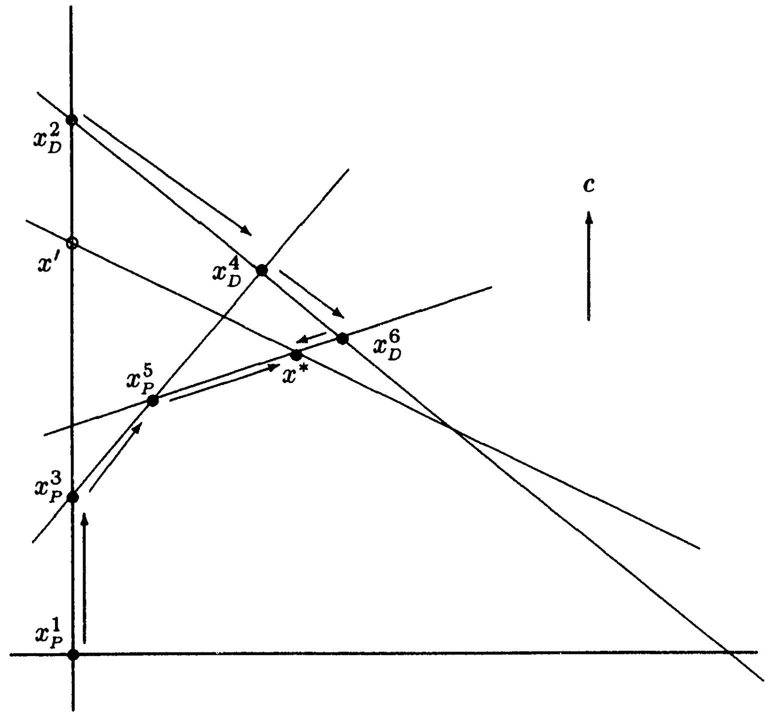

Figure 1: A two-dimensional example.

$r_j < 0$, then for some basic variable i, we may obtain a dual-feasible vertex by deleting variable i from B if the following case holds:

Case 1: $\hat{A}_{ij} > 0 \; \forall j \in S_+$, and

$$\max_j \{ \frac{r_j}{\hat{A}_{ij}} \mid j \in S_+ \} \le \min_j \{ \frac{r_j}{\hat{A}_{ij}} \mid j \in S_-, \hat{A}_{ij} < 0 \}.$$

Let $R_+ = \{i \mid i \text{ is basic and } (x_B)_i > 0\}$ and $R_- = \{i \mid i \text{ is basic and } (x_B)_i \le 0\}$. Similarly, if there exists some basic variable i with $(x_B)_i < 0$, then for some nonbasic variable j, we may obtain a primal-feasible vertex by adding variable j to B if the following case holds:

Case 2: $\hat{A}_{ij} < 0 \; \forall i \in R_+$, and

$$\max_i \{ \frac{(x_B)_i}{\hat{A}_{ij}} \mid i \in R_- \} \le \min_i \{ \frac{(x_B)_i}{\hat{A}_{ij}} \mid i \in R_+, \hat{A}_{ij} > 0 \}.$$

5. Initialization

Now it is time to say something about initialization. Suppose that a nonsingular basis B of (P) is given. If the initial vertex is not primal feasible ($x_B \not\geq 0$), then we may add an artificial variable (column), say j, such that Case 2 is satisfied. For example, $A_{ij} = -1$ for all $i \in R_-$, and $A_{ij} = 0$ otherwise; and c_j is assigned a large negative value. Then a primal-feasible vertex can be obtained by choosing this artificial variable to enter the basis.

A dual-feasible vertex can be obtained in a similar way by adding a redundant constraint that satisfies Case 1 (now we have B_P) and deleting the corresponding slack variable. The new basis is B_D.

6. Relationship to Lemke's Method

There is a similarity between the new algorithm and Lemke's [Lem65]. If the initial nonsingular basis B corresponds to a primal-feasible vertex of (P), then both algorithms are exactly the same (they follow the same path from that vertex to the solution). The two algorithms differ in the following aspects:

- Lemke's method works on a larger dimension matrix (involving A and A^T).
- When B does not correspond to a primal feasible vertex of (P) the new algorithm uses two artificial variables instead of one as in Lemke's.
- In Lemke's method, the primal and dual variable pairs are "almost complementary"; i.e, they are complementary except for one pair of variables, both of which are nonbasic. In the new algorithm, there are two pairs of variables that are not complementary. (In one pair, both variables are basic; in the other pair, both are nonbasic.)
- There is only one path to the optimal solution in Lemke's method, but the new algorithm is flexible (for example the variation mentioned above).

7. Summary of the Algorithm

We now summarize the algorithm. We assume that a nonsingular basis B_P is given.

1. Solve $B_P x_B = b$.
2. If B_P is not primal feasible ($x_B \not\geq 0$), find another initial B_P.
   (a) Add $a_P = -Be$ to A as an artificial column, where e is a columns of 1s. Set the corresponding entry in c to be $-M$, for some large number M.
   (b) Remove the column corresponding to the most negative x_B from B_P.
   (c) Add the artificial column to B_P. Solve $B_P x_B = b$.
3. Solve $B_P^T \pi = c_B$ and compute reduced costs $r = c - A^T\pi$. Define $B_D = B_P$.
4. If B_D is dual feasible ($r \leq 0$), then either x_B and $x_N = 0$ form an optimal solution (if no artificial column was added in Step 2), or the problem is infeasible; stop. Otherwise, find another initial B_D.
   (a) Add an artificial row to A with zeros in basic columns and ones in nonbasic columns. Set the corresponding entry in b to be ∞.

(b) Add a new artificial column $a_D = (0, \ldots, 0, 1)^T$ to A. Set the corresponding entry in c to be 0.

(c) Expand B_P by adding this new column and the appropriate part of the artificial row.

(d) Solve $B_P^T \pi = c_B$ and compute reduced costs $r = c - A^T \pi$.

(e) Let d be the column with the most positive element of r.

(f) Let $B_D = B_P$ except that a_D is replaced by d.

5. Repeat until $d = a_D$.

(a) Solve $B_P x_B = b$ and $B_P y = d$.

(b) Determine the index $p = \arg\min\{(x_B)_i / y_i \mid y_i > 0\}$.

(c) Let the pth column of B_P be a_l.

(d) If $a_l = a_D$, the problem is unbounded; stop.

(e) Let $B_D = B_P$ except that a_D is replaced by d.

(f) Solve $B_D^T \pi = c_B$ and $B_D^T v = e_p$.

(g) Compute $r = c - A^T \pi$ and $w = A^T v$ (the pth row of $B_D^{-1} A$).

(h) Determine the index $s = \arg\min\{r_j / w_j \mid w_j < 0\}$.

(i) Replace column a_l by d in B_P.

(j) Let $d = a_s$.

6. If a_P is in B_P, the problem is infeasible; stop.

7. Otherwise, find the optimal solution.

(a) Remove a_D and the corresponding row from B_P.

(b) Solve $B_P x_B = b$ and set $x_N = 0$.

In practice, we could use y in 5(a) to update x_B, and w in 5(g) to update r. The main work per iteration is therefore solving $B_P y = d$ and $B_D^T v = e_p$ and forming $w = A^T v$. (Only nonbasic entries are needed.) This is analogous to one iteration of the primal simplex method with full pricing.

Two forms of partial pricing are also possible. First, columns with the most negative r_j could be temporarily ignored (keeping $x_j = 0$). The bases B_P will remain primal feasible. When the columns are reconsidered, we have to check that the current basis B_D is dual feasible. If so, the algorithm continues; otherwise, a new dual-feasible vertex must be constructed as in 4(a)–4(f).

Alternatively, rows with the most positive $(x_B)_i$ could be temporarily ignored. When they are reconsidered, we have to check that B_P is primal feasible. If not, a new vertex must be constructed as in 2(a)–2(c).

8. Experimental Results

The algorithms (original and variational) have been coded in MATLAB [MLB87] and compared with the simplex method with least-reduced-cost pivot rule (also coded in MATLAB). Full pricing was used in all methods. Some random test problems were generated with the following properties. All constraints (except for non-negativity) are tangent to a unit ball. None of the constraints is redundant. The center of the unit ball is either at the origin or at a point within two units of the origin.

The initial vertex was taken to be the origin. The position of the center therefore determines whether the initial vertex is (primal) feasible or not.

Table 1 lists results for 20 test problems with varying dimensions and center. The iteration numbers for the three algorithms are shown for each problem. Note that for the least-reduced-cost simplex method (LP) and our original algorithm (PDO), the iteration number is the number of primal vertices visited. For our variational algorithm (PDV), it is the number of primal *and* dual vertices visited. Hence, the iteration numbers reflect the relative computation times for the three algorithms.

m	n	density	LP	PDO	PDV	LP	PDO	PDV
			Origin feasible			Origin infeasible		
50	25	0.3	38	37	63	36	26	49
100	25	0.3	41	42	66	32	26	41
150	25	0.3	29	24	30	53	54	95
200	25	0.3	51	41	85	38	38	72
100	50	0.1	51	40	62	60	45	56
150	50	0.1	102	81	124	114	92	111
200	50	0.1	103	96	80	118	87	105
150	100	0.1	260	179	341	292	207	407
200	100	0.1	261	166	311	397	311	494
200	150	0.1	454	308	516	430	264	536

LP: Simplex method with least-reduced-cost rule
PDO: Primal and dual algorithm (original)
PDV: Primal and dual algorithm (variational)

Table 1: Iteration numbers for three pivot rules.

9. Conclusions

The preliminary computational tests suggest that our original primal and dual algorithm (PDO) performs increasingly well as the problem size increases, compared to the traditional simplex method (LP). They also suggest that PDO is generally better than the variational algorithm (PDV), though other test problems may reveal different performances.

We believe that further computational tests are justified for both new algorithms.

References

[Dan63] G. B. Dantzig (1963). *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ.

[Lem65] C. E. Lemke (1965). Bimatrix equilibrium points and mathematical programming, *Management Science* 11, 681–689.

[MLB87] C. Moler, J. Little and S. Bangert (1987). PRO-MATLAB User's Guide, The MathWorks, Inc., Sherborn, MA.

[TZ92] T. Terlaky and S. Zhang (1992). *A survey of pivot rules for linear programming*, Research Memorandum No. 480, Faculty of Economics, University of Groningen, The Netherlands.

DATE
FILMED
10/13/93
END