The Simplex Method of Linear Programming Using LU Decomposition

RICHARD H. BARTELS AND GENE H. GOLUB
Stanford University, Stanford, California*

Standard computer implementations of Dantzig's simplex method for linear programming are based upon forming the inverse of the basic matrix and updating the inverse after every step of the method. These implementations have bad round-off error properties. This paper gives the theoretical background for an implementation which is based upon the LU decomposition, computed with row interchanges, of the basic matrix. The implementation is slow, but has good round-off error behavior. The implementation appears as CACM Algorithm 350.

KEY WORDS AND PHRASES: simplex method, linear programming, LU decomposition, round-off errors, computational stability
CR CATEGORIES: 5.41

1. LU Decomposition

The linear programming problem:

$$\text{maximize } d^T x$$
$$\text{subject to } \begin{cases} Gx = b \\ x \geq 0 \end{cases} \qquad (1)$$

where $d^T = [d_0, \cdots, d_{n-1}]$, $b^T = [b_0, \cdots, b_{m-1}]$, and

$$G = \begin{bmatrix} g_{0,0} & \cdots & g_{0,n-1} \\ \vdots & & \vdots \\ g_{m-1,0} & \cdots & g_{m-1,n-1} \end{bmatrix}$$

are given, with $b \geq 0$, is commonly solved by a two-phase process built upon the simplex method of G. B. Dantzig [2]. Each step of this method requires the solution of three systems of linear equations involving a common matrix of coefficients, the basis matrix P. It is the usual practice to solve these systems by forming P^{-1}, either directly or in factored form as a product of transformations, and then applying it to the right-hand sides of the systems.

The basis matrix of any simplex step differs from that of the preceding step in only one column, so it is easier to update P^{-1} than to invert the new P. While this generally produces satisfactory results, it is vulnerable to computational problems in two respects. First, if P^{-1} is continually updated, computational inaccuracies imposed by limited-precision arithmetic (round-off errors) are allowed to propagate from step to step. Second, the updating, in whatever way it is carried out, is equivalent to premultiplying P^{-1} by a matrix of the form:

$$\begin{array}{c|c|c|c} & & & \\ 1 \;\; 0 & -y_0/y_k & \multirow{2}{*}{O} & \\ \;\; \ddots & \vdots & & \\ 0 \;\; 1 & -y_{k-1}/y_k & & \\ \hline k \quad 0 & 1/y_k & 0 & \\ \hline & -y_{k+1}/y_k & 1 \;\; 0 & \\ \multirow{2}{*}{O} & \vdots & \;\; \ddots & \\ & -y_{m-1}/y_k & 0 \;\; 1 & \\ \hline & k & & \end{array}$$

In the presence of round-off errors this computation can be quite inaccurate if y_k is small relative to the other y_j.

Both of these problems can be eliminated if, instead of P^{-1}, the LU decomposition of P is computed using row interchanges to select pivots. The problem of solving a linear system based upon P is then reduced to that of backsolving two triangular linear systems. The numerical stability of this scheme is generally quite good (see [4]). Advantage can be taken of the similarity between any two successive matrices so as to economize on the computation of the LU decompositions. For details see [1, Secs. 5–6].

Additional accuracy is obtained in the program (see Algorithm 350) by iteratively refining the solution to problem (1), after it has been found, according to the scheme given in [4, p. 121].

The algorithm requires $n^3/6 + O(n^2)$ multiplications per exchange on the average. Thus the algorithm is most effectively used for very ill-conditioned problems or in conjunction with other implementations of the simplex algorithm which produces an initial basis more cheaply. A fast and accurate version of the simplex algorithm is described in [5] but this requires additional storage.

This paper gives the theoretical background of Algorithm 350, "Simplex-Method Procedure Employing LU Decomposition," by the same authors, which appears on pages 275–278.

* Computer Science Department. This project was supported in part by NSF under project GP948 and ONR under project NR 044 211.

2. The Two-Phase Solution Process

The possibility of degeneracy is ignored, as is customary.

PHASE 1 (Find a basic feasible solution.)

$$\text{maximize } -e^T a$$

$$\text{subject to } \begin{cases} Gx + a = b \\ x \geq 0, \qquad a \geq 0, \end{cases} \tag{2}$$

where $a^T = [a_0, \cdots, a_{m-1}]$ is a vector of "artificial variables" and $e^T = [1, \cdots, 1]$ (m components). The solution is obtained by starting with $a = b$ as a basic feasible solution and applying the simplex method. As each a_i becomes nonbasic, it may be dropped from the problem altogether. Since $-e^T a \leq 0$ for all a, problem (2) has a solution. If $max(-e^T a) \neq 0$, then problem (1) has no basic feasible solution. Otherwise, the second phase is entered.

PHASE 2 (Find an optimal basic feasible solution.)

If the solution to problem (2) contains only components of x as basic variables, these become the initial basic variables for solving problem (1) by the simplex method. If artificial variables, say $a_{j_0}, \cdots, a_{j_l}$, remain basic but at zero level in the solution of (2), an equivalent problem to (1) is solved:

$$\text{maximize } d^T x$$

$$\text{subject to } \begin{cases} \left[\begin{array}{c|c|c|c|c} G & e_{j_0} & \cdots & e_{j_l} & 0 \\ \hline 0 & 1 & \cdots & 1 & 1 \end{array}\right] \left[\begin{array}{c} x \\ \hline a_{j_0} \\ \hline \vdots \\ \hline a_{j_l} \\ \hline s \end{array}\right] = \left[\begin{array}{c} b \\ \hline 0 \end{array}\right] \\ x \geq 0, s \geq 0, a_{j_0} \geq 0, \cdots, a_{j_l} \geq 0 \end{cases} \tag{3}$$

where e_k represents the kth column of the identity matrix (column numbering begun at zero). The additional variable s, together with the additional constraint

$$s + \sum_{i=0}^{l} a_{j_i} = 0,$$

holds the artificial variables at zero level throughout further computation. The simplex method is applied to problem (3) with s and the basic variables from problem (2) as the initial set of basic variables. (For an example of this two-phase process in operation see [3, Sec. 7.7].)

3. Some Computational Details

Phase 1 is skipped if a basic feasible set of variables for problem (1) is known a priori. If a partial set of basic variables is known for problem (1), the computation of phase 1 can be restructured to take this into account. To be precise, let κ ($1 \leq \kappa \leq m - 1$) columns of G (without loss of generality these can be made $g_{n-\kappa}, \cdots, g_{n-1}$, the last columns of G) be used to construct the following matrix:

$$P = \left[\begin{array}{c|c|c|c|c} e_0 & \cdots & e_{m-\kappa-1} & g_{n-\kappa} & \cdots & g_{n-1} \end{array}\right]$$

Suppose that $P^{-1}b \geq 0$. (This situation arises if the variables are numbered so that $x_{n-\kappa}, \cdots, x_{n-1}$ are the slack variables and the constraints are ordered so that $g_{n-\kappa} = e_{m-\kappa}, \cdots, g_{n-1} = e_{m-1}$, whereby P becomes the identity matrix.) Then problem (2) can be replaced by the problem:

$$\text{maximize } -\tilde{e}^T \tilde{a}$$

$$\text{subject to } \begin{cases} \left[\begin{array}{c|c|c} G^{(1)} & \begin{array}{c} \tilde{I} \\ \hline 0 \end{array} & G^{(2)} \end{array}\right] \left[\begin{array}{c} x^{(1)} \\ \hline \tilde{a} \\ \hline x^{(2)} \end{array}\right] = b \\ x^{(1)} \geq 0, \qquad x^{(2)} \geq 0, \qquad \tilde{a} \geq 0 \end{cases} \tag{4}$$

where $x^{(1)} = [x_0, \cdots, x_{n-\kappa-1}]$, $x^{(2)} = [x_{n-\kappa}, \cdots, x_{n-1}]$, $G^{(1)}$ and $G^{(2)}$ are appropriate submatrices of G, $\tilde{a} = [a_0, \cdots, a_{m-\kappa-1}]$, $\tilde{e} = [1, \cdots, 1]$ ($m - \kappa$ components), and $\tilde{I}$ is the $(m - \kappa)$-order identity matrix. The simplex method is begun with $a_0, \cdots, a_{m-\kappa-1}, x_{n-\kappa}, \cdots, x_{n-1}$ as basic variables. Since there are fewer artificial variables to be driven to zero in problem (4) than in problem (2), we expect problem (4) to require on the average less work to solve.

A reduction of computation beyond that described in [1] can be arranged for the LU decompositions calculated in phase 1. Each time an artificial variable becomes nonbasic, a row and column interchange can be applied to the basis matrix so that it always has the partitioned form

$$P = \left[\begin{array}{c|c} I & R \\ \hline O & S \end{array}\right]$$

where I is the identity matrix of appropriate order, and R and S are composed of components of G. The LU decomposition of P is known when the decomposition $S = \mathfrak{LU}$ has been computed; viz.

$$P = \left[\begin{array}{c|c} I & O \\ \hline O & \mathfrak{L} \end{array}\right] \left[\begin{array}{c|c} I & R \\ \hline O & \mathfrak{U} \end{array}\right]$$

The triangular systems of equations induced by this format are solved in a particularly efficient manner by Algorithm 350.

4. Example Application

Let V be a random variable restricted to a finite set of values $v_0, \cdots, v_{n-1}$. Let p_i be the probability that V has value v_i. If certain moments

$$\mu_k = \sum_{j=0}^{n-1} v_j{}^k p_j, \qquad k \in \{k_0, \cdots, k_{m-1}\}$$

are known together with the values v_j, estimates of the

p_i and of unknown moments μ_l can be made via linear programming; viz.

$$\alpha_i \leq p_i \leq \beta_i \quad \text{and} \quad \gamma_l \leq \mu_l \leq \delta_l ,$$

where

$$\alpha_i = -\max(-x_i), \qquad \beta_i = \max(x_i)$$

$$\gamma_l = -\max(-\sum_{j=0}^{n-1} v_j{}^l x_j), \qquad \delta_l = \max(\sum_{j=0}^{n-1} v_j{}^l x_j),$$

all subject to the constraints

$$x_0 \geq 0, \cdots, x_{n-1} \geq 0, \qquad \sum_{j=0}^{n-1} x_j = 1,$$

$$\sum_{j=0}^{n-1} v_j{}^k x_j = \mu_k , \qquad k \in \{k_0 , \cdots , k_{m-1}\}, \quad k \neq 0.$$

The basis matrices encountered in solving these problems are submatrices of a Vandermonde matrix, making them somewhat poorly conditioned; so it is important that the simplex method be carried out accurately. Only the objective function differs from one estimation problem to the next; so the solution of any one of the problems would provide an initial basis for the other problems. Phase 1 would need to be carried out only once.

Thus, for example, to compute upper bounds for $p_0 , \cdots , p_{n-1}$ from given values $v_0 , \cdots , v_{n-1}$ and given moments $\mu_0 , \cdots , \mu_{m-1}$, one would set up arrays $G[i, j] \equiv v_j{}^i$ and $b[i] \equiv \mu_i$ and execute

```
kappa := 0; for i := 0 step 1 until n − 1 do d[i] := 0;
for i := 0 step 1 until n − 1 do
begin d[i] := 1;
  if i ≥ 1 then d[i − 1] := 0;
  linprog(m, n, kappa, G, b, d, x, z, ind, infeas, unbdd, sing);
  upper bound[i] := z;
end;
```

As an illustration, the preceding was run on Stanford University's B5500 computer with the data $\mu_0 = 1, \mu_1 = 3, \mu_2 = 10.5, \mu_3 = 40.5$ and $v_i = i$ for $i = 0, \cdots, 6$. The results are listed below.

Computed upper bound	*Actual probability*
.083333333325	.015625
.19999999998	.09375
.50000000001	.234375
.83333333335	.3125
.49999999999	.234375
.20000000001	.09375
.083333333334	.015625

Lower bounds for the p_i computed from the above data in a similar fashion were all zero. Computed bounds for μ_4 through μ_{10} were as follows.

Computed lower bound	*Actual moment*	*Computed upper bound*
165.5	168.0	175.5
700.5	738.0	850.5
3035.5	3396.75	4495.5
13390.5	16251.75	25150.5
59965.5	80335.5	145435.5
272200.5	408280.5	856210.5
1251135.5	2124764.25	5088055.5

RECEIVED SEPTEMBER, 1967; REVISED NOVEMBER, 1968

REFERENCES

1. BARTELS, R. H., AND GOLUB, G. H. Stable numerical methods for obtaining the Chebyshev solution to an overdetermined system of equations. *Comm. ACM 11*, 6 (June 1968), 401–406.
2. DANTZIG, G. B. *Linear Programming and Extensions*. Princeton U. Press, Princeton, N.J., 1963.
3. HADLEY, G. *Linear Programming*. Addison-Wesley, Reading, Mass., 1962.
4. WILKINSON, J. H. *Rounding Errors in Algebraic Processes*. Prentice-Hall Series in Automatic Computation, Englewood Cliffs, N.J., 1963.
5. BARTELS, RICHARD H. A numerical investigation of the simplex method. Tech. Rep. No. CS 104, Comput. Sci. Dep., Stanford U., Stanford, Calif., July 1968.

Rough and Ready Error Estimates in Gaussian Integration of Analytic Functions

PHILIP RABINOWITZ
The Weizmann Institute of Science, Rehovoth, Israel*

KEY WORDS AND PHRASES: numerical integration, analytic functions, error estimates, Gaussian integration, tabulated error coefficients, computable error coefficients, Cauchy integral formula, Chebyshev polynomials
CR CATEGORIES: 5.11, 5.16

* Department of Applied Mathematics

Two expressions are derived for use in estimating the error in the numerical integration of analytic functions in terms of the maximum absolute value of the function in an appropriate region of regularity. These expressions are then specialized to the case of Gaussian integration rules, and the resulting error estimates are compared with those obtained by the use of tables of error coefficients.

Davis and Rabinowitz [1] presented a method for estimating the error committed in integrating an analytic function by an arbitrary integration rule

$$I(f) = \sum_{i=1}^{n} w_i f(x_i).$$

If the error is denoted by $E(f) = \int_{-1}^{1} f(x)\, dx - I(f)$, we