From the theory of such equations, we find that for large i:

$$Q_i \doteq a\{\tfrac{1}{2}(K + [4 + K^2]^{\frac{1}{2}})\}^i$$

Now it is easily seen that the error in any approximation P_i/Q_i is of the order of $(Q_i)^{-2}$ and therefore, per average cycle, the error should decrease by a factor of $4\{K + (4 + K^2)^{\frac{1}{2}}\}^{-2} \doteq 1/9.1$ or approximately one decimal place.

Daniel Shanks

Naval Ordnance Laboratory
White Oak, Maryland

[1] See for example, D. Teichroew, "Use of continued fractions in high speed computing," *MTAC*, v. 6, 1952, p. 127.

[2] A. Khintchine, "Zur metrische Kettenbruchtheorie," *Compositio Math.*, v. 3, 1936. p. 276–285.

[3] Daniel Shanks, "Note on an absolute constant of Khintchine;" *MTAC*, v. 4, 1950, p. 28.

The Product Form for the Inverse in the Simplex Method

Summary: When a matrix is represented as a product of "elementary" matrices, the matrix, its transpose, its inverse and inverse transpose are readily available for vector multiplication. By an "elementary matrix" is meant one formed from the identity matrix by replacing one column; thus an elementary matrix can be compactly recorded by the subscript of the altered column and the values of the elements in it. In the revised simplex method,[1] both the inverse and inverse transpose of a "basic" matrix are needed; more significant, however, is the fact that each iteration replaces one of the columns of the basis. In the product form of representation, this change can be conveniently effected by multiplying the previous matrix by an elementary matrix; thus, only one additional column of information need be recorded with each iteration. This approach places relatively greater emphasis on "reading" operations than "writing" and thereby reduces computation time. Using the I.B.M. Card Programmed Calculator, a novel feature results: when the inverse matrix is needed at one stage and its transpose at another, this is achieved simply by turning over the deck of cards representing the inverse.

Introduction:

The simplex method is an algorithm for determining values for a set of n non-negative variables which minimizes a linear form subject to m linear restraints.[1,2a,3] It may be characterized briefly as a finite iterative procedure. Each iteration produces a new special solution to the restraint equations involving a subset of m of the variables, only one element of the subset changing on successive iterations; the remaining $n - m$ variables are equated to zero. The vectors of coefficients corresponding to the subset of m variables are linearly independent and constitute a *basis* in m-dimension real vector space. In the original simplex method[2a] (as coded for the SEAC[4] or as found in CHARNES *et al.*[3], it is required that all the coefficient vectors be represented in terms of the latest basis; since the changes of basis are step-wise, a simple recursion relation suffices to alter the representations on each iteration.

The revised simplex method[1,5] differs from the original method in that it uses the same recursion relations to transform only the inverse of the basis for each iteration. It has been introduced to reduce the quantity of writing at each step (which it does in general), and is particularly effective for linear programming models where the original matrix of coefficients is largely composed of zeros, as for example, in the transportation model[2c] or dynamic economic and production models.[2b] If the original method is used, these zeros would be replaced by non-zeros in the successive iterations and this greatly increases the computational effort. On the other hand, the revised method leaves those zeros intact.

One important feature of the simplex method is concerned with the criteria by which one of the vectors in the basis is replaced by a vector not in the basis to form the basis of the next iteration. When the constant terms of the restraint equations are not general, *the choice of the vector to drop from the basis may be ambiguous* and an arbitrary selection (as pointed out in unpublished examples by ALAN HOFFMAN and PHILIP WOLFE) may lead to non-convergence. Several devices exist, however, for perturbing the constant terms so as to avoid this difficulty. The earliest proposal along these lines[2a] consisted in modifying the vector of constant terms by a specially weighted combination of the unit vectors. This approach may be used conveniently both for the revised and original simplex methods[1,5]. With the original simplex method, there is another natural way to form the *pertubation* which consists in adding a weighted linear combination of the column vectors to the vector of constant terms. This was suggested first by ORDEN and developed independently by CHARNES[3].

Although considerable attention has been paid to the above difficulty (called *degeneracy*), *it usually does not lead to non-convergence.* The type of problems in which it can cause non-convergence appear to be exceedingly rare. To date, there have been only two examples and these were artificially constructed for this purpose. Accordingly, the SEAC code and the RAND code use an *arbitrary* selection criteria in case of ambiguity. In these codes, a deliberate decision was made to use a simple code in lieu of a more complex one needed to cover a possible case that may *never* arise in practice.

The present method of using the product form for the representation of the inverse of the matrix, also makes use of this simplification. Again, provision could be made for covering the rare non-convergent case, but again, it does not appear to be worth-while.

We shall now describe a process by which a square non-singular matrix may be expressed as a product of elementary matrices of the form (2) below. This is illustratively seen for the simplex process which involves a step-wise change of basis matrix, that is to say, two successive matrices differ by only one column. Using a notation consistent with[1,5] let $B^{(l-1)} = (P_0, P_{j_1}, \cdots, P_{j_m})$ denote the $(l - 1)^{\text{th}}$ basis. If, in the next basis, P_s is to replace P_{j_r}, then it is easy to show that

$$[B^{(l)}]^{-1} = E_l[B^{(l-1)}]^{-1}, \qquad (1)$$

where E_l and E_l^{-1} are elementary matrices related by

$$(2)\quad E_l = [U_0, \cdots, U_{r-1}, \eta_l, U_{r+1}, \cdots, U_m] = [U_0, \cdots, U_{r-1}, Y_l, U_{r+1}, \cdots, U_m]^{-1},$$

where U_i is a unit vector with unity in the $(i + 1)^{st}$ component, η_l is a vector whose components η_{il} are related to components y_{il} of Y_l by

$$(3)\quad \begin{aligned} \eta_{il} &= - y_{il}/y_{rl}, \qquad i \neq r \\ \eta_{rl} &= 1/y_{rl}, \end{aligned}$$

where it is necessary that $y_{rl} \neq 0$ and Y_l is defined by

$$(4)\quad Y_l = [B^{(l-1)}]^{-1}P_s.$$

Successive applications of (1) for $l = k, k - 1, k - 2, \cdots, 1$ yield

$$(5)\quad (B^{(k)})^{-1} = E_k E_{k-1} \cdots E_1 [B^{(0)}]^{-1},$$

where $B^{(0)}$ is the initial basis. It is usually easy to arrange that the *initial basis* $B^{(0)}$ *be the identity matrix so that* $B^{(0)}$ may be dropped from (5).

Consider the problem of computing a row vector β_0, defined by

$$(6)\quad \beta_0 = \alpha B^{-1} = \alpha E_k E_{k-1} \cdots E_1,$$

where α is a given row vector (actually a unit vector[1,5]). Such a vector is required by the revised simplex method as the first step in determining the vector P_s to introduce into the basis. It is clear that β_0 can be obtained by successive transformations on row vectors, i.e., forming $(\alpha)E_k$, $(\alpha E_k)E_{k-1}$, $\cdots$, etc. However, when a row vector $A = (a_0, a_1, \cdots, a_m)$ is transformed into a row vector $B = (b_0, b_1, \cdots, b_m)$ by multiplying A on the right by an elementary matrix E_l one obtains simply

$$(7)\quad \begin{aligned} b_i &= a_i, \qquad\qquad i \neq r_l \\ b_{r_l} &= \sum_0^m \eta_{il} a_i, \end{aligned}$$

where, because r may be different for different l, we have set $r = r_l$.

Consider next the problem of computing Y by relation (4).

$$(8)\quad Y = B^{-1}P_s = E_k E_{k-1} \cdots E_1 P_s.$$

It is clear that Y can be determined by successive transformations on column vectors, i.e., forming $E_1(P_s), \cdots$, etc. However, when a column vector $C = \{c_0, c_1, \cdots, c_m\}$ is transformed into a column vector $D = \{d_0, d_1, \cdots, d_m\}$ by multiplying C on the left by a matrix of the special form E, one obtains simply

$$(9)\quad \begin{aligned} d_i &= c_i + \eta_{il} c_{r_l}, \qquad i \neq r_l \\ d_{r_l} &= \qquad \eta_{r_l l} c_{r_l}. \end{aligned}$$

From (7) and (9) it is clear that the only essential information contained in E_l is the set of values η_{il} and the index r_l. Note further that in (8), the successive E_l are used with *increasing* l and it follows from (9) that it *is necessary* to know r_l *before* using the η_{il}. On the other hand, in (6), the E_l are used in *decreasing* sequence of l but from (7) it *is not necessary* to know r_l until *after* the η_{il} have been used. The perfect complementarity of the preceding two sentences, together with the fact that $\sum_0^m \eta_{il}a_i$ can obviously be computed starting with $i = m$ as well as with $i = 0$, makes it clear that (6) may be computed using the information in the reverse order of that used in (8).

Let L_l denote the ordered set of 'words' of information

$$L_l = \{r_l; \eta_{0l}, \eta_{1l}, \cdots, \eta_{ml}\}. \tag{10}$$

Then each change of a column of B will produce a new L_{l+1} which may be stored in consecutive order to the previously computed $L_1, L_2, \cdots, L_l$.

On the CPC, by punching two sets of instructions on each card—one being, in form, the reflection, in the vertical center line, of the other (with appropriate adjustments for difference in algorithms (7) and (9))—the transpose use of the inverse may be accomplished by simply turning the cards over using the vertical center line of the card for the axis.

George B. Dantzig
Wm. Orchard-Hays

Rand Corp.
Santa Monica, Calif.

[1] G. B. Dantzig, Alex Orden & Philip Wolfe, *The Generalized Simplex Method.* Rand P-392-1 dated August 4, 1953.

[2] T. C. Koopmans, Ed., *Activity Analysis of Production and Allocation.* New York, 1951.
(a) George B. Dantzig, *Maximization of a Linear Function of Variables Subject to Linear Inequalities.* P. 339–347.
(b) *The Programming of Interdependent Activities: Mathematical Model.* P. 19–32.
(c) *Application of the Simplex Method to a Transportation Problem.* P. 359–373.
(d) T. C. Koopmans & S. Reiter, *A Model of Transportation.* P. 222–259.

[3] A. Charnes, W. W. Cooper & A. Henderson, *An Introduction to Linear Programming.* New York, 1953.

[4] A. Hoffman, M. Mannos, D. Sokolowsky & N. Wiegmann, "Computational experience in solving linear programs," Soc. Industrial and Applied Math., *Jn.*, v. 1, 1953, p. 17–33.

[5] G. B. Dantzig, *Computational Algorithm of the Simplex Method.* Rand P–394, April 10, 1953.

On Modified Divided Differences II

[Continued from *MTAC*, v. 8, p. 1–11]

Errors of Type (c). A question that presents itself is the extent to which errors of Type c will mask an isolated error. It will be desirable to approach the problem from a statistical standpoint, and to introduce the simplifying assumptions that the errors of Type (c) behave like round-off errors, subject to the following restrictions: