Prof. OTTO J. M. SMITH
University of California, Berkeley

Linear Programming Using Sparse Elemental Factors

Large scale linear programming needs efficient methods for storage and for matrix operations. In the case of economic models the matrices are sparse and only the non-zero elements need to be stored in a compact vector, with row and columns indices locating the position in the vector. The order in which the basic variables are determined from the simplex method and removed from the nonbasic vector is a poor order from the viewpoint of number of computer operations and size of core storage required. An optimal order minimizes the core storage and the generation of the new non-zero elements. This paper uses a sub-optimal order based on the number of non-zero terms in each row of the basic submatrix of the tableau. The challenge of finding the optimal order in the general case is an open problem, awaiting a skillful innovator.

The tableau of a linear program can be arranged as a matrix A operating on an unknown vector X equals a known vector B. B has n terms and X has m terms, where $m > n$, and

$$AX = B, \quad X \geqslant 0. \tag{1}$$

The matrix A is n by m.

Consider first the case when the first n terms in X are the *basic* variables to be solved for, and the remaining $(m - n)$ terms are the *nonbasic* variables which will be assumed to be zero. The far left square n by n submatrix of A is an identity matrix.

$$\left[I \,\middle|\, A_N \right] \left[\frac{X_B}{X_N} \right] = B. \tag{2}$$

The solution in this case is:

$$X_N = 0 \text{ and } X_B = B. \tag{3}$$

Consider secondly the case when the basic variables are scattered throughout the vector X, and the corresponding columns of A have numbers in them. If the vector X is sorted with the basic variables at the top, then the columns of matrix A will also be correspondingly sorted.

$$\left[\begin{array}{c|c} A_B & A_N \end{array}\right] \left[\frac{X_B}{X_N}\right] = B. \tag{4}$$

Now the solution is again X_N equal to zero and

$$A_B X_B = B \text{ and } X_B = A_B^{-1} B. \tag{5}$$

If instead of sorting, the submatrix A_B is defined as the columns associated with the basic variables, and called the basis matrix, and its inverse used on the unsorted original matrix A, then

$$A_B^{-1} A X = A_B^{-1} B. \tag{6}$$

It is obvious that this has produced the effect of reducing each column to zero except for a single 1.0 in the row corresponding to the number of the column in the basis submatrix A_B.

Starting with the matrix A having this pattern, and with a feasible solution for the subvector X_B, a better solution can be found by a simplex pivoting in which a new column q in the nonbasis submatrix is to be made into a basis column by reducing all of the valued to zero in the column except for a 1.0 in the p row.

The elemental matrix which will accomplish this transformation is the n—by—n square matrix with 1.0 on the diagonal in all but the p row and with the p column equal to

$$\begin{bmatrix} -\frac{a_{1q}}{a_{pq}} \\ \cdots \\ \frac{1}{a_{pq}} \\ \cdots \\ -\frac{a_{nq}}{a_{pq}} \end{bmatrix} \begin{matrix} \\ \\ p \text{ row} \\ \\ \\ \end{matrix} \tag{7}$$

Call this matrix C_{p1}

$$C_{p1} = \begin{bmatrix} 1.0 & 0 & -\frac{-a_{1q}}{a_{pq}} & 0 \\ 0 & 1.0 & \cdots & \\ & & -\frac{1}{a_{pq}} & \\ & & \cdots & \\ 0 & & -\frac{-a_{nq}}{a_{pq}} & 1.0 \end{bmatrix} \begin{matrix} \\ \\ \\ p \text{ row} \\ \\ \\ \end{matrix} \qquad (8)$$

p column

Applying this matrix to the original equations yields:

$$C_{p1} A X = C_{p1} B. \qquad (9)$$

Yet another better solution might be found by pivoting on a new column and row, accomplished by the new elemental matrix C_{p2}:

$$C_{p2} C_{p1} A X = C_{p2} C_{p1} B. \qquad (10)$$

This procedure is repeated untill the optimum set of basic variables is found after the h-th pivot.

$$C_{ph} \ldots C_{p1} A X = C_{ph} \ldots C_{p1} B. \qquad (11)$$

Identifying the subvector X_B of X containing only the last basic variables, then:

$$X_B = C_{ph} \ldots C_{p1} B \qquad (12)$$

$$X_N = 0. \qquad (13)$$

In the process of computing the left side of equations (10 and (11), the order of selecting the columns is important because it influences the number of non-zero elements which appear in ($C_{ph} \ldots C_{p1} A$). There exists an optimal order with minimum "fill-in", and in practice a sub-optimal order should be found and used.

Effect of Order on a Sparse Tableau

Let the linear program tableau A be

$$A = \begin{bmatrix} 1 & & & x & x & x \\ & 1 & & & x & \\ & & 1 & & x & x \\ & & & 1 & x & x \end{bmatrix} \qquad (14)$$

In which each x denotes the location of a non-zero element, and all unfilled locations have identically zero elements. The matrix A is the original tableau, and the pivoting operations to be performed have been deter-

mined either by a sequence of simplex iterations carried out in some other program, or by a shrewd guess concerning which elements in x are basic and which are non-basic. We know in this example that the basic elements in x are in the positions 2, 4, 6, and 7.

The two pivot operations to be performed are Row 3, Column 6 and Row 1, Column 7, introducing columns 6 and 7 into the basis matrix and removing columns 1 and 3. The order to be examined is 7, then 6. To pivot on column 7 first, the elemental matrix C has the form:

$$C_{p1} = \begin{bmatrix} x & & & \\ & 1 & & \\ x & & 1 & \\ & & & 1 \end{bmatrix}. \tag{15}$$

And now:

$$C_{p1}A = \begin{bmatrix} x & & & \\ & 1 & & \\ x & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & x & x & x \\ & 1 & & & x & \\ & & 1 & & x & x \\ & & & 1 & x & x \end{bmatrix} = \begin{bmatrix} x & & & & x & x & 1 \\ & 1 & & & & x & \\ x & & 1 & & \overline{|x|} & x & \\ & & & 1 & x & x & \end{bmatrix} \tag{16}$$

where $\overline{|x|}$ is location of a new element introduced.

To pivot on Column 6 next, the elemental matrix C has the form:

$$C_{p2} = \begin{bmatrix} 1 & & x & \\ & 1 & x & \\ & & x & \\ & & x & 1 \end{bmatrix} \tag{17}$$

and we get

$$C_{p2}C_{p1}A = \begin{bmatrix} 1 & & x & \\ & 1 & x & \\ & & x & \\ & & x & 1 \end{bmatrix} \begin{bmatrix} x & & & x & x & 1 \\ & 1 & & & & x \\ x & & 1 & & \overline{|x|} & x \\ & & & 1 & x & x \end{bmatrix} = \begin{bmatrix} x & & x & & x & & 1 \\ \overline{|x|} & 1 & x & & \overline{|x|} & & \\ x & & x & & \overline{|x|} & 1 & \\ \overline{|x|} & & x & 1 & x & & \end{bmatrix}. \tag{18}$$

Basis Vectors

These two operations require the storage of the vectors in the new matrices, and to store C_{p1} and C_{p2} requires 6 terms. The full tableau has had 4 new terms added.

Now compare with the sequence of pivot (3, 6) followed by (1,7).
For column 6:

$$C_{p1} = \begin{bmatrix} 1 & & x & \\ & 1 & x & \\ & & x & \\ & & x & 1 \end{bmatrix} \text{ so} \tag{19}$$

$$C_{p1}A = \begin{bmatrix} 1 & & x & \\ & 1 & x & \\ & & x & \\ & & x & 1 \end{bmatrix} \begin{bmatrix} 1 & & & x & x & x \\ & 1 & & & x & \\ & & 1 & & x & x \\ & & & 1 & x & x \end{bmatrix} = \begin{bmatrix} 1 & & x & & x & \\ & 1 & x & & & \overline{|x|} \\ & & x & & 1 & x \\ & & x & 1 & x & \overline{|x|} \end{bmatrix}. \tag{20}$$

There have been two new elements $\boxed{x}$ introduced in Column 7, which will be the next pivot:

$$C_{p2} = \begin{bmatrix} x & & & \\ \boxed{x} & 1 & & \\ x & & 1 & \\ \boxed{x} & & & 1 \end{bmatrix} \tag{21}$$

$$C_{p2}C_{p1}A = \begin{bmatrix} x & & & \\ \boxed{x} & 1 & & \\ x & & 1 & \\ \boxed{x} & & & 1 \end{bmatrix} \begin{bmatrix} 1 & x & x & x \\ & 1 & x & \boxed{x} \\ & & x & 1 \; x \\ & & x \; 1 \; x & \boxed{x} \end{bmatrix} = \begin{bmatrix} x & x & x & 1 \\ \boxed{x} \; 1 & x & \boxed{x} & \\ x & x & \boxed{x} \; 1 & \\ \boxed{x} & x \; 1 \; x & & \end{bmatrix} \tag{22}$$

and now to store C_{p1} and C_{p2} requires 8 terms instead of 6 while the full tableau has had the same 4 terms added.

Therefore (1,7) followed by (3, 6) is better then the inverse order.

To examine the effect of order, for each column j count the number of row terms that couple into blank positions in each of the other columns k:

R_{jk} = terms added to k column by using the j column,

$$R_{67} = 2; \; R_{76} = 0. \tag{23}$$

Assume that the optimal order is 1, 2, 3,..., n. The number of extra terms introduced intro the C_p matrices that must be remembered are, corresponding to operators C_{pj}:

operator	number of extra terms
C_{p2}	R_{12}
C_{p3}	$R_{13} + R_{23}$
C_{p4}	$R_{14} + R_{24} + R_{34}$
C_{p5}	$R_{15} + R_{25} + R_{35} + R_{45}$
...	...
C_{pn}	$\sum_{j=1}^{n-1} R_{jn}$

(24)

The number of extra terms generated in columns that later must be processed are:

From column 1,	$(R_{12} + R_{13} + R_{14} + R_{11} + ... + R_{1n})$
From column 2,	$R_{23} + R_{24} + R_{22} + ... + R_{2n})$
From column 3,	$(R_{34} + R_{33} + ... + R_{3n})$
From column 4,	$(R_{44} + ... + R_{4n})$
From column $(n-1)$	$R_{(n-1)\,n}$

(25)

The optimal order is that which minimizes

$$min \sum_{j=1}^{n-1} \sum_{k=j+1}^{n} R_{jk}. \tag{26}$$

One method of determining a "good" or "suboptimal" order for selecting the next pivot is to calculate for each column k the sum of:

$$R_k = \sum_{\substack{r \\ \text{all} \\ \text{remaining} \\ \text{columns}}} R_{kr}. \tag{27}$$

Select the minimum R_k and use the corresponding column for the first pivot. Recompute all R_k and again find the minimum to specify the second pivot. Repeat at each iteration.

The effect of order is the same, wheather the matrix operations are performed using elemental factors as in equation (11) or a single matrix step as in equation (6), after rearranging the columns in the specified order.

Note

$$A_B^{-1} = C_{ph} \ldots C_{pl}. \tag{28}$$

It is revealing to examine the effect of order in solving LP problems using an unfactored matrix.

LINEAR PROGRAMMING WITHOUT FACTORIZATION

To restate the problem in slightly more detail, there are inequality constraints of the form:

$$F(x) \leqslant b \tag{29}$$

which are replaced by equalities of the form:

$$F(x) + s = b \tag{30}$$

where s is the amount by which the constraint is not reached. The s values are unknown. In addition, these are many more equality constraints.

When all of the equations are assembled together,

$$\underset{\text{matrix}}{A} \quad \underset{\text{vector}}{X} = \underset{\text{vector}}{B} \tag{31}$$

The s terms can either be at the top or at the bottom of the vector X.

Make a preliminary selection of all of the s terms that one expects to be zero; i.e. all slack variables which are zero. Put these at the bottom of the vector X. Put important non-zero unknowns at the top of vector X. There should be many more unknowns in the vector X that in the known vector B. Let B be of order n, A of n by m, and X of order m, as in equation (1).

Select the top n terms of X and call them the basic variables X_B, as in equation (4).

$$\left[\, A_B \,\middle|\, A_N \,\right] \cdot \left[\frac{X_B}{X_N}\right] = B. \tag{32}$$

For the first solution, let $X_N = 0$ and $X_B = A_B^{-1}B$.
The original equations may be now written:

$$\left[\begin{array}{c|c} I & A_B^{-1} A_N \end{array}\right] \quad \left[\begin{array}{c} X_B \\ \hline X_N \end{array}\right] = A_B^{-1} B. \tag{33}$$

Note that this is similar to equation (2), but not exactly the same.

In the event that some columns of A_B have only a single term then the corresponding terms in X_B are already basic variables. Put these at the top. Then D is a diagonal matrix and the structure of the tableau and equation is:

$$\left[\begin{array}{c|c|c} D & 0 & A_c \\ \hline 0 & A_B & A_N \end{array}\right] \left[\begin{array}{c} X_D \\ \hline X_B \\ \hline X_N \end{array}\right] = B = \left[\begin{array}{c} V_1 \\ \hline V_2 \end{array}\right] \tag{34}$$

where:

$$V_1 = D\,X_D + A_c X_N \tag{35}$$

$$V_2 = A_B X_B + A_N X_N \tag{36}$$

$$D^{-1}\,V_1 = X_D + D^{-1}\,A_c X_N \tag{37}$$

$$A_B^{-1}\,V_2 = X_B + A_B^{-1}\,A_N X_N. \tag{38}$$

Using the matrix operator:

$$\begin{bmatrix} D^{-1} & 0 \\ 0 & A_B^{-1} \end{bmatrix}$$

Operating to the right on equation (33) yields:

$$\left[\begin{array}{c|c|cc} I & 0 & D^{-1} & A_c \\ \hline 0 & I & A_B^{-1} & A_M \end{array}\right] \begin{bmatrix} X_D \\ X_B \\ X_N \end{bmatrix} = \begin{bmatrix} D^{-1} & V_1 \\ A_B^{-1} & V_2 \end{bmatrix} \tag{39}$$

Therefore the Basic Solution is seen to be:

$$\begin{aligned} X_D &= D^{-1}\,V_1 \\ X_B &= A_B^{-1}\,V_2 \\ X_N &= 0 \end{aligned} \tag{40}$$

V_2 should be optimally ordered so that $A_B^{-1}\,V_2$ needs the fewest number of terms in its solution.

The Cost Function

Let a row C contain the weighting parameters c_k that enter into the scalar cost function f.

$$C\,X = f. \tag{41}$$

In the simplex method, the magnitudes and signs of c_k are examined to select the corresponding x_k which if reduced to zero would make the largest improvement (reduction) in the scalar cost function f. After several interations of the simplex method, we will have the list of the new basic variables, and the list of the non-basic variables.

Let X_B be the vector of all the original basic variables that stay basic. Let X_N be the vector of all of the original nonbasic variables that stay non-basic.

Let subscript C denote "change".

Let X_{CB} be the vector of variables changed to basic from original non-basic.

Let X_{CN} be the vector of variables changed to non-basic from basic.

The old tableau will be sorted for the vector X assembled in the order X_B, X_{CN}, X_N and X_{CB}.

This sorting will produce the following equation of restrictions in which the identity matrix is still on the left side of the tableau, and the terms in the right side vector corresponding to the unchanged basic variables X_B are V_1 and the terms corresponding to the changed basic variables X_{CB} are V_2:

$$\left[\begin{array}{c|c|cc} I & 0 & M_1 & M_3 \\ \hline 0 & I & M_2 & M_4 \end{array}\right] \quad \left[\begin{array}{c} X_B \\ \hline X_{CN} \\ X_N \\ \hline X_{CB} \end{array}\right] = \left[\begin{array}{c} V_1 \\ \hline V_2 \end{array}\right]. \tag{42}$$

The Old feasible solution was:

$$X_B = V_1, \; X_N = 0, \tag{43}$$
$$X_{CN} = V_2 \; X_{CB} = 0.$$

The New feasible solution which has a less scalar cost f, is:

$$X_{CN} = 0, \; X_{CB} = 0 \tag{44}$$

$$V_1 = X_B + M_1 X_N + M_3 \; X_{CB} \tag{45}$$

$$V_2 = X_{CN} + M_2 X_N + M_4 X_{CB}. \tag{46}$$

Operate from the left on equation (41) with the matrix:

$$\begin{bmatrix} I & 0 \\ 0 & M_4^{-1} \end{bmatrix}$$

to produce the matrix equation (46).

$$\left[\begin{array}{c|c|cc|c} I & 0 & M_1 & & M_3 \\ \hline 0 & M_4^{-1} & M_4^{-1} & M_2 & I \end{array}\right] \quad \left[\begin{array}{c} X_B \\ X_{CN} \\ X_N \\ X_{CB} \end{array}\right] = \left[\begin{array}{ccc} V_1 & & \\ M_4^{-1} & & V_2 \end{array}\right] \tag{47}$$

$$M_4^{-1} V_2 = M_4^{-1} X_{CN} + M_4^{-1} M_2 X_N + X_{CB}. \tag{48}$$

Now

$$X_N = 0$$

$$X_{CN} = 0 \tag{49}$$

$$X_B = V_1$$

$$X_{CB} = M_4^{-1} V_2$$

Rearranging Columns

$$\left[\begin{array}{c|c|c|c} I & M_3 & M_1 & 0 \\ \hline 0 & I & M_4^{-1} M_2 & M_4^{-1} \end{array}\right] \left[\begin{array}{c} X_B \\ \hline X_{CB} \\ X_N \\ X_{CN} \end{array}\right] = \left[\begin{array}{c} V_1 \\ M_4^{-1} V_2 \end{array}\right]. \tag{50}$$

The Rows are changed for optimal ordering, before each inverse matrix operation. Note that the solution requires the equivalent of M_4^{-1}, which is the inverse of a large sparse matrix which is very sensitive to the order of the operations. The equation to be solved is:

$$M_4 \; X_{CB} = V_2 \tag{51}$$

an preferably should be done by factoring M_4 or by upper triangularizing and back substitution.

MATRIX FACTORIZATION TECHNIQUE

Define the following elemental sparse matrices:

$$U_j = \begin{bmatrix} 1 & & & & 0 \\ & 1 & x & x & x \\ & & 1 & & 0 \\ 0 & & & 1 & \\ & & & & 1 \end{bmatrix} \leftarrow j. \tag{52}$$

This has nonzero elements only in the upper triangle and only in the j row, and ones on the diagonal.

$$L_k = \begin{bmatrix} 1 & & & & 0 \\ 0 & 1 & & & \\ & x & 1 & & \\ & x & 0 & 1 & \\ & x & & & 1 \end{bmatrix} \tag{53}$$

$\uparrow$
k

This has nonzero elements only in the lower triangle and only in the k column and ones on the diagonal.

$$C_h = \begin{bmatrix} 1 & & & & \\ & 1 & x & & \\ & & 1 & & \\ & & x & 1 & \\ & & x & & 1 \\ & & h & & \end{bmatrix} \tag{54}$$

This has nonzero elements only in the k column, and ones on the diagonal.

The inverse are:

$$\begin{aligned} U_j^{-1} &= 2\ I - U_j \\ L_h^{-1} &= 2\ I - L_h \\ C_k^{-1} &= 2\ I - C_k. \end{aligned} \tag{55}$$

Each inverse equals the original matrix with a change of sign for all off-diagonal elements.

An elemental matrix can be factored out of any matrix. Given the matrix A:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & \\ \cdots & & & \\ a_{n1} & & & a_{nn} \end{bmatrix}. \tag{56}$$

All but the diagonal term of the first column can be a zero. Generate the lower elemental matrix:

$$L_1 = \begin{bmatrix} 1 & & & \\ -\frac{a_{21}}{a_{11}} & 1 & & \\ -\frac{a_{31}}{a_{11}} & & 1 & \\ \cdots & & & \\ -\frac{a_{n1}}{a_{11}} & & & 1 \end{bmatrix}. \tag{57}$$

Then:

$$L_1 A = \begin{bmatrix} a_{11} & a_{12} & \cdots & \\ 0 & a_{22} - \dfrac{a_{21}\, a_{12}}{a_{11}} & a_{23} - \dfrac{a_{21}\, a_{13}}{a_{11}} & \\ 0 & a_{32} - \dfrac{a_{31}\, a_{12}}{a_{11}} & \cdots & \\ \cdots & \cdots & & \left(a_{nn} - \dfrac{a_{n1}\, a_{1n}}{a_{11}}\right) \end{bmatrix}. \quad (58)$$

Generate the upper elemental matrix:

$$U_1 = \begin{bmatrix} 1 & -\dfrac{a_{12}}{a_{11}} & -\dfrac{a_{13}}{a_{11}} & \cdots & -\dfrac{a_{1n}}{a_{11}} \\ & 1 & & & \\ & & 1 & & \\ & & & \cdots & \\ & & & & 1 \end{bmatrix} \quad (59)$$

Then:

$$L_1 A U_1 = \begin{bmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} - \dfrac{a_{21}\, a_{12}}{a_{11}} & a_{23} - \dfrac{a_{21}\, a_{13}}{a_{11}} & & \\ 0 & a_{32} - \dfrac{a_{31}\, a_{12}}{a_{11}} & & & \\ \cdots & \cdots & & & \cdots \\ 0 & & & & a_{nn} - \dfrac{a_{n1}\, a_{1n}}{a_{11}} \end{bmatrix} = A^{(1)}. \quad (60)$$

To Reduce the 2nd column below the diagonal and the 2nd row above the diagonal to zero, generate the lower and upper elemental matrices L_2 and U_2:

$$L_2 = \begin{bmatrix} 1 & & & \\ 0 & 1 & & \\ & -\dfrac{a_{32}^{(1)}}{a_{22}^{(1)}} & & \\ & \cdots & & \\ 0 & -\dfrac{a_{n2}^{(1)}}{a_{22}^{(1)}} & & \end{bmatrix} \quad (61)$$

$$U_2 = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ & 1 & -\dfrac{a_{23}^{(1)}}{a_{22}^{(1)}} & \cdots & -\dfrac{a_{2n}^{(1)}}{a_{22}^{(1)}} \\ & \cdots & & & \end{bmatrix}. \quad (62)$$

Then:

$$L_2 L_1 A U_1 U_2 = A^{(2)}. \tag{63}$$

Continue until

$$L_n \ldots L_2 L_1 A U_1 U_2 \ldots U_n = A^{(n)} \tag{64}$$

$A^{(n)}$ is Diagonal and Known:

$$D = A^{(n)}. \tag{65}$$

The matrix A has now been factored into:

$$A = L_1^{-1} L_2^{1} \ldots L_n^{-1} D U_n^{-1} \ldots U_2^{-1} U_1^{-1}. \tag{66}$$

A Linear Equation can be solved as follows:

Let:

$$AX = B \tag{67}$$

where A and B are Known and X is unknown.

Then:

$$X = A^{-1} B = U_1 U_2 \ldots U_n D^{-1} L_n \ldots L_2 L_1 B \tag{68}$$

This involves only multiplications by elemental matrices, which preserves sparsity. This is the method that should be used to solve equation (50).

The optimal order is important in both factorizing and upper triangularizing. To illustrate this, consider a matrix with elements in only the first row and column as bellow:

$$\begin{bmatrix} x & x & x & x & x & x \\ x & x & & & & \\ x & & x & & & \\ x & & & x & & \\ x & & & & x & \\ x & & & & & x \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{bmatrix} \tag{69}$$

When the first column is reduced to zero bellow the diagonal by either factoring or upper triangularizing, the matrix is completely filled in. This yields:

$$\begin{bmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{bmatrix}. \tag{70}$$

Completing the upper triangularizing will yield eventually:

$$\begin{bmatrix} x & x & x & x & x & x \\ & x & x & x & x & x \\ & & x & x & x & x \\ & & & x & x & x \\ & & & & x & x \\ & & & & & x \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{bmatrix}. \tag{71}$$

The cost of this order in additional storage locations required is 5 in this case, and $n^2 - 3n + 2$ in the general case.

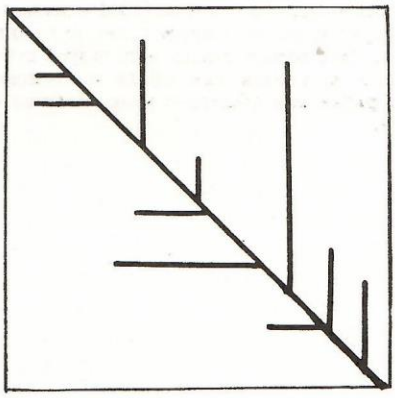
When we reverse the order of rows and columns, the original matrix has the pattern:

$$\begin{bmatrix} x & & & & & x \\ & x & & & & x \\ & & x & & & x \\ & & & x & & x \\ & & & & x & x \\ x & x & x & x & x & x \end{bmatrix} \begin{bmatrix} \\ X \\ \\ \\ \\ \\ \end{bmatrix} = \begin{bmatrix} \\ B \\ \\ \\ \\ \\ \end{bmatrix}. \tag{72}$$

When this is the upper triangularized, there is on fill-in, and the final matrix has the pattern:

$$\begin{bmatrix} x & & & & & x \\ & x & & & & x \\ & & x & & & x \\ & & & x & & x \\ & & & & x & x \\ & & & & & x \end{bmatrix} \begin{bmatrix} \\ X \\ \\ \\ \\ \\ \end{bmatrix} = \begin{bmatrix} \\ B' \\ \\ \\ \\ \\ \end{bmatrix} \tag{73}$$

No additional storage locations are necessary, and this order was optimal (defined by no fill-in). An optimal order with no fill-in is a "tree" (in the graph theory sense) with its "root" in the lower right corner of the matrix, and all matrix elements being branches connected to the "trunk" (the diagonal) in contiguous vertical columns above the diagonal and horizontal rows to the left of the diagonal like in the upper picture:



There exists a topological analog of this matrix, in which "towns" are the "nodes" or the elements of vector X, and one-way roads are the directed branches, which are the elements of the matrix. The process of factoring or upper- triangularizing is the process of removing nodes in order, and adding oneway- road links between the remaining nodes so that no communication route is broken. The optimal order is that for which the minimum number of links needs to be added.

LA PROGRAMMATION LINEAIRE UTILISANT DES FACTEURS ELEMENTAIRES DISPERSES

La programmation linéaire de grandes dimensions nécessite des méthodes efficaces pour la mémorisation et pour les opérations avec des matrices. Au cas des modèles économiques, les matrices renferment nombre d'éléments non nuls, mais seuls les éléments non nuls doivent être ordonnés en un vecteur compact, avec des indices de ligne et de colonne, pour localiser leur positions dans le vecteur.

L'ordre dans lequel les variables de la base sont déterminées au moyen de la méthode simplex est un ordre non efficace au point de vue du nombre des opérations exécutées au calculateur et de la grandeur de la mémoire centrale nécessaire. Un ordre optimal fait minimiser la mémoire occupée et l'opération d'engendrement des nouveaux éléments non nuls. Ce travail utilise un ordre sous-optimal, basé sur le nombre des termes non nuls de chaque ligne de la sous-matrice de base du tableau. Trouver un ordre optimal pour le cas général constitue un problème qui attend un innovateur averti.

ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ С ПРИМЕНЕНИЕМ ЭЛЕМЕНТАРНЫХ РАССЕЯННЫХ ФАКТОРОВ

Линейное программирование больших размеров требует эффективных методов для запоминания и для матричных операций. В случае экономических моделей матрицы содержат много нулевых элементов, но только ненулевые элементы должны быть расположены в компактный вектор с индексами линий и колонн, локализующими их позиции в векторе.

Порядок, в котором переменные основы определяются по методу симплекс, не является эффективным в отношении числа операций на ЭВМ и величины необходимой центральной памяти. Оптимальный порядок минимизирует занятую память и операцию порождения новых ненулевых элементов. Настоящая статья использует субоптимальный порядок, основанный на числе ненулевых элементов каждой линии основной субматрицы таблицы. Изыскание оптимального порядка для общего случая является открытым вопросом, ожидающим умелого новатора.

12. XII. 1973