LITERATURE CITED

Duffy, G. J., and I. A. Furzer, "Mass Transfer on a Single Sieve Plate Column Operated with Periodic Cycling," *AIChE J.*, **24**, 588 (1973).
Furzer, I. A., and G. J. Duffy, "Generalized Theory of Periodically Operated Plate Columns," Joint Symposium on Distillation, University of Sydney/University of N.S.W., Australia (May, 1974).
———, "Periodic Cycling of Plate Columns: Discrete Residence Time Distribution," *AIChE J.*, **22**, No. 6, 1118 (1976).
———, "Microprocessor System for Plate Column Control," *IEEE. IECI*, **25**, No. 2, 145 (1978).
Gerster, J. A., and H. M. Scull, "Performance of Tray Columns Operated in the Cycling Mode," *AIChE J.*, **16**, 108 (1970).
Hausen, H., "Zur Definition des Austauschgrades von Rektifzierboden," *Chemie-Ingr-Tech.*, **25**, 595 (1953).
Holland, C. D., *Multicomponent Distillation*, Prentice-Hall, Englewood Cliffs, N.J. (1963).
Horn, F. J. M., and R. A. May, "Effect of Mixing on Periodic Countercurrent Processes," *Ind. Eng. Chem. Fundamentals*, **1**, 349 (1968).
Krishna, R., "A Film Model Analysis of Non-equimolar Distillation of Multicomponent Mixtures," *Chem. Eng. Sci.*, **32**, 1197 (1977).
———, and G. L. Standard, "A Multicomponent Film Model Incorporating a General Matrix Method of Solution to the Maxwell-Stefan Equations," *AIChE J.*, **22**, 383 (1976).
Murphree, E. V., "Rectifying Column Calculations," *Ind. Eng. Chem.*, **17**, 747 (1925).
Sargent, R. W. H., and B. A. Murtagh, "The Design of Plate Distillation Columns for Multicomponent Mixtures," *Trans. Inst. Chem. Engrs.*, **47**, T85 (1969).
Schrodt, V. N., J. T. Sommerfeld, O. R. Martin, P. E. Parisot, and H. H. Chien, "Plant-scale Study of Controlled Cyclic Distillation," *Chem. Eng. Sci.*, **22**, 759 (1967).
Standardt, G., "Studies in Distillation-V. Generalized Definition of a Theoretical Plate or Stage of Contacting Equipment," *ibid.*, **20**, 611 (1965).
———, "Comparison of Murphree-type Efficiencies with Vaporisation Efficiencies," *ibid.*, **26**, 985 (1971).

Manuscript received June 9, 1978; revision received January 24, and accepted February 6, 1979.

Maintaining Sparsity In Process Design Calculations

MARK A. STADTHERR

Chemical Engineering Department
University of Illinois
Urbana, Illinois 61801

Sparse linear systems in bordered block-triangular form can be efficiently solved by block-triangular splitting, a new technique that provides a drastic reduction in the fill-in (loss of sparsity) that occurs in the direct solution of such systems. This technique is applicable in the iterative solution of the large systems of nonlinear equations that arise in the equation oriented approach to process simulation and design calculations.

SCOPE

In most programs for computer aided chemical process design, the computational work is organized into modules, each of which performs the computations for one sort of unit operation. In such programs, the computational modules are usually simulation oriented; that is, given the variables describing the input streams and the equipment, they calculate the variables describing the output streams. However, if a design problem is to be solved, it may be necessary to calculate an input variable or equipment parameter given an output variable. Design oriented modules can be provided, but because of the large number of possible design problems, an inordinately large number of such modules would have to be provided. In practice, a few design oriented modules are typically provided to handle the most common design problems. In other cases, the design problem must be solved by an iterative or controlled simulation, in which the process is repeatedly simulated until the desired values for the design variables are obtained. While this is reasonably effective for small processes of average complexity, it is much less so when applied to the design of the large, complex processes commonly found today.

In principle, this difficulty can be resolved by eliminating the computational modules and performing all computations simultaneously. The potential advantages of this global (or equation oriented) approach to the design problem are well recognized, but acceptance of this approach has been slow, primarily because of the difficulties involved in solving extremely large sets of nonlinear equations simultaneously. For solving such systems, Newton-Raphson or related techniques are typically most effective. These techniques require the periodic solution of a set of linear equations, which in this case will be extremely large and sparse. In solving large, sparse linear systems, a major difficulty is the loss of sparsity that occurs when the conventional elimination techniques are used. This fill-in may lead to excessive storage requirements and unacceptably long computational times. Thus, the acceptance of the global approach to computer aided chemical process design depends to a great extent on the development of efficient techniques for maintaining sparsity. The work described here is directed largely at the development and application of such techniques.

CONCLUSIONS AND SIGNIFICANCE

Large sparse linear equation systems, such as those that typically arise in the equation oriented approach to chemical process design calculations, can be solved by a new technique called block-triangular splitting. This technique involves splitting the coefficient matrix into the sum of a block-triangular matrix and one or more matrices of low rank. Solution by block-triangular splitting is computationally efficient and provides a drastic reduction in fill-in compared to solution by other techniques. For instance, in an example derived from the design of a triple-effect evaporator, a nearly 90% reduction in fill-in can be obtained by using the new technique. It is hoped that the development of this technique will facilitate the application of the global approach to process simulation and design.

BACKGROUND

Consider the problem of solving the linear system $A\mathbf{x} = \mathbf{b}$, where $A = [a_{ij}]$ is an $n \times n$ nonsingular matrix, $\mathbf{x}$ is an n dimensional vector of unknowns, and $\mathbf{b}$ is an n dimensional constant vector. The usual solution procedure involves the factorization $A = LU$, where $L = [l_{ij}]$ is lower triangular and $U = [u_{ij}]$ is upper triangular. In these terms, the inverse can be represented as $A^{-1} = U^{-1}L^{-1}$. This form is computationally convenient, since the inverse of an $n \times n$ triangular matrix can be factored into a product of n triangular matrices that differ from the identity matrix in only one column. That is, $L^{-1} = L_nL_{n-1} \ldots L_1$ and $U^{-1} = U_1U_2 \ldots U_n$, where $L_k = I_n + (\mathbf{e}_k - \mathbf{l}_k)\mathbf{e}_k^T/l_{kk}$ and $U_k = I_n + (\mathbf{e}_k - \mathbf{u}_k)\mathbf{e}_k^T/u_{kk}$. Here, $\mathbf{l}_k$, $\mathbf{u}_k$, and $\mathbf{e}_k$ are vectors identical to the kth columns of L, U, and I_n, respectively. This is commonly known as the elimination form of the inverse (EFI).

The elements of L and U are usually found using Gaussian elimination or some variation thereof. If conventional Gaussian elimination is used, $l_{ij} = a_{ij}^{(j)}$ for $i \geq j$, $u_{ij} = a_{ij}^{(i+1)}$ for $i < j$, and $u_{ii} = 1$, where $A^{(k)} = [a_{ij}^{(k)}]$ is the $n \times n$ matrix remaining after the $(k-1)$th step of the elimination $[A = A^{(1)}]$.

During Gaussian elimination, it is usually necessary to perform pivoting, that is, to reorder the rows and/or columns of A in order to place desirable elements in the pivot positions $a_{ii}^{(i)}$, $i = 1, \ldots, n$. This may be done to maintain numerical stability or, as discussed below, to maintain sparsity, though these may be mutually exclusive aims. Thus, the factorization obtained is actually $PAQ = LU$, where P and Q are the appropriate row and column permutation matrices. For simplicity, P and Q will subsequently be omitted.

If A is large and sparse, the number of nontrivial nonzeros in L and U may greatly exceed the number of nonzeros in A. This loss of sparsity, or fill-in, may lead to excessive storage requirements and unacceptably long computational times. For this reason, iterative methods have been widely used when dealing with large sparse systems. Such methods are very effective in many cases, but in other cases, convergence is slow if there is convergence at all within a reasonable time. Moreover, selection of an efficient iteration technique is not always straightforward, nor is the selection of other factors such as acceleration parameters and closure tolerances. Thus, expensive and time consuming experimentation may be necessary to select an effective solution procedure.

In recent years, much progress has been made toward reducing the fill-in associated with the direct solution methods. Much of this work has been reviewed in a monograph by Tewarson (1973) and in a recent survey by Duff (1977). These techniques can be divided into two general categories: those that involve pivoting and those that reorder A prior to elimination into forms for which fill-in can be limited to certain areas of the matrix. The pivoting techniques can be subdivided into local strategies, in which both columns and rows are reordered during elimination, and a priori strategies, in which columns are reordered before elimination and rows during elimination or vice versa.

The most widely used pivoting technique is the local strategy of Markowitz (1957). At the kth stage of Gaussian elimination, this strategy selects as the pivot the element $a_{pq}^{(k)}$, $p \geq k$, $q \geq k$, where p and q minimize $[r_p^{(k)} - 1][c_q^{(k)} - 1]$. Here, $r_i^{(k)}$ and $c_j^{(k)}$ are the numbers of nonzero elements in the last $n - k + 1$ positions of the ith row and jth column of $A^{(k)}$, respectively. Although not locally optimal, in the sense that the fill-in at each stage is not minimized, this strategy produces overall results nearly as good as and sometimes better than more expensive locally optimal strategies. This is the case because minimizing fill-in locally does not imply that the total fill-in is minimized.

Techniques that reorder A prior to elimination usually involve permuting A to block-triangular form (BTF) or bordered block-triangular form (BBTF). These forms can be so factored that fill-in is limited to the diagonal blocks of a BTF and to the diagonal blocks and one border of a BBTF.

There is no clear-cut advantage to either of the two general approaches to maintaining sparsity. The pivoting techniques seem to be better suited to general purpose use, since it is not always possible to permute a matrix to BTF or to BBTF with small borders. On the other hand, when these permutations are possible, as is often the case in chemical engineering design problems, savings can be appreciable.

A new sparsity preservation technique, which will be called block-triangular splitting, is introduced below. This technique involves the use of BTF and BBTF matrices, which are now considered in more detail.

Block-Triangular Form

Let A be in BTF; that is

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1N} \\ & A_{22} & \cdots & A_{2N} \\ & & \ddots & \vdots \\ & & & A_{NN} \end{bmatrix}$$

where A_{ii} is a square matrix of order m_i. A matrix that can be permuted to BTF is reducible; one that cannot is irreducible. For a given reducible matrix, the BTF is unique at least in the sense that N and the particular rows and columns represented in each block are fixed. There are several techniques available for permuting to BTF; for general purpose use the algorithm of Tarjan (1972) appears to be most effective.

A system in BTF is readily solved by block-back substitution. Thus, in factored form the inverse can be represented as $A^{-1} = \overline{A}_1\overline{A}_2 \ldots \overline{A}_N$, where $\overline{A}_k = I_n + (\overline{E}_k - \overline{F}_k)A_{kk}^{-1}\overline{E}_k^T$. Here, $\overline{E}_k$ and $\overline{F}_k$ are $n \times m_k$ matrices identical to the $(1 + \Sigma_{i=1}^{k-1} m_i)$th to the $(\Sigma_{i=1}^{k} m_i)$th columns of I_n and A, respectively. Note that since only the diagonal blocks of A need be inverted, fill-in is limited to these areas.

Bordered Block-Triangular Form

If A is irreducile, it can be permuted to BBTF:

$$A = \left[\begin{array}{c|c} M & R \\ \hline S & T \end{array}\right]$$

where M is $m \times m$ and block triangular, R and S are nonzero borders, and T is $t \times t$ and may be either zero or nonzero. A system in BBTF may be solved by block elimination, factoring A as

$$A = \left[\begin{array}{c|c} M & 0 \\ \hline S & I \end{array}\right]\left[\begin{array}{c|c} I & M^{-1}R \\ \hline 0 & T - SM^{-1}R \end{array}\right]$$

Since both factors are block triangular, the system is now readily solved by block substitution, as described above. In this process, fill-in occurs in the diagonal blocks of M as it is inverted, in R and T as they are replaced with $M^{-1}R$ and $T - SM^{-1}R$, and finally in the filled-in T as $T - SM^{-1}R$ is solved during block substitution. This procedure is described algorithmically by Tewarson (1973) for the case in which M is block upper triangular. It may also be regarded as an application to BBTF of George's (1974) so-called F_2 factorization. It should be noted that if M is simply triangular, this solution procedure is equivalent to ordinary Gaussian elimination, using the diagonal elements of M as pivots in the usual order if M is lower triangular and in reverse order if M is upper triangular.

Permuting a matrix to BBTF is often called tearing. However, it should be noted that this term has a more general significance, tearing to BBTF being only a special case. Because of the fill-in pattern discussed above, in tearing a matrix to BBTF it is desirable to minimize the number of tears t while keeping the size of the diagonal blocks in M small. However, numerical considerations may dictate the use of a BBTF with larger borders.

A variety of techniques for tearing to BBTF are surveyed by Hlavacek (1977) and Duff (1977). Although sparsity preservation was not usually a consideration in the development of these techniques, most aim to minimize t and thus produce a BBTF with desirable fill-in characteristics. Many of these algorithms, for instance, those given by Christensen and Rudd (1969), Barkley and Motard (1972), and Kevorkian and Snoek (1973) require the a priori assignment of an output set (that is, the permutation of A to produce a zero free diagonal). Since the minimum number of tears obtained depends on the initial choice of output set, to obtain an absolute minimum would require looking at all output sets, the number of which may be quite large. Furthermore, in the BBTF with the absolute minimum of tears, T may have one or more rows or columns with no nonzero elements. Such a BBTF could not be found if output set assignment is required. Thus, these techniques are useful primarily when an output set is fixed by other considerations. For instance, in modular chemical process simulators, the output set is determined by the nature of the modules. Also, the output set may be fixed by numerical considerations (Kevorkian and Snoek, 1973; Westerberg and Edie, 1971*a*).

If the absolute minimum number of tears is desired, techniques such as those given by Christensen (1970), Westerberg and Edie (1971*b*), and Stadtherr et al. (1974), which lead directly to the absolute minimum, are more useful. With few exceptions, the algorithms using these techniques are capable only of tearing to bordered triangular form. Using Stadtherr's algorithm, a BBTF with small (2×2 or 3×3) blocks on the diagonal of M can be efficiently produced. Also, this algorithm is capable of disallowing certain elements as candidates for the diagonal of M. This feature is useful in preventing the selection of a BBTF with poor numerical characteristics.

It has been suggested (Duff, 1977) that a system in BBTF could be solved by first splitting the coefficient matrix into the sum of a block-triangular matrix and a matrix of low rank and then applying Householder's formula for the inverse of such a sum. The obvious splitting is to let the low rank matrix simply comprise one of the borders; for instance, $A = A' + E_t[S|0]$ where E_t is an $n \times t$ matrix identical to the last t columns of I_n and

$$A' = \left[\begin{array}{c|c} M & R \\ \hline 0 & T \end{array}\right]$$

Applying Householder's formula now requires the inversion of A'. However, since one or more rows or columns of T may contain no nonzeros, inverting A' may not be possible. Thus, this simple splitting is not a practical technique for solving systems in BBTF.

We now describe block-triangular splitting, a technique that splits a matrix in BBTF in such a way that the aforementioned difficulty is avoided. Moreover, a system so split can be readily solved without resorting to Householder's formula. Using this technique, fill-in can be limited to the diagonal blocks of A, thus eliminating the off-block diagonal fill-in inherent in conventional methods.

BLOCK-TRIANGULAR SPLITTING

Consider the $n \times n$ bordered block-triangular system

$$A\mathbf{x} = \left[\begin{array}{c|c} M & R \\ \hline S & T \end{array}\right]\mathbf{x} = \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} \end{array}\right] = \mathbf{b} \qquad (1)$$

where $\mathbf{c}$ is $m \times 1$, and $\mathbf{d}$ is $t \times 1$. The goal is to reduce A to BTF so it can be solved by back substitution. This can be done by subtracting from both sides of Equation (1) the column vector $E_t[S|SM^{-1}R]\mathbf{x}$. Thus we have $(A - E_t[S|SM^{-1}R])\mathbf{x} = \mathbf{b} - E_t[S|SM^{-1}R]\mathbf{x}$. The right-hand side can be rearranged by noting that from Equation (1), $[M|R]\mathbf{x} = \mathbf{c}$, from which it follows $[S|SM^{-1}R]\mathbf{x} = SM^{-1}\mathbf{c}$. Thus, the original system has been reduced to the block-triangular system $A^*\mathbf{x} = \mathbf{b}^*$, where

$$A^* = \left[\begin{array}{c|c} M & R \\ \hline 0 & T - SM^{-1}R \end{array}\right]$$

and

$$\mathbf{b}^* = \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} - SM^{-1}\mathbf{c} \end{array}\right]$$

This amounts to splitting A into the sum of a block-triangular matrix A^* and a matrix of low rank $E_t[S|SM^{-1}R]$; that is, $A = A^* + E_t[S|SM^{-1}R]$. We call this process block-triangular splitting (BTS). Note that in performing BTS, fill-in occurs only in the diagonal blocks of M as it is inverted and in T as it is replaced with $T - SM^{-1}R$. This may also be viewed in terms of implicit back solution (George, 1974). The solution is now readily obtained by computing $\mathbf{b}^*$ and inverting A^*. Since M has already been inverted, only $T - SM^{-1}R$ must be inverted to invert A^*. Thus, the overall fill-in is limited to the diagonal blocks of A. We now discuss the implementation of BTS and show that in addition to being efficient in terms of fill-in, it is also efficient in terms of number of computations required.

Implementation

Consider first the computation of $W = M^{-1}R$. This is equivalent to solving the systems $M\mathbf{w}_i = \mathbf{r}_i$, $i = 1, 2, \ldots, t$, where $\mathbf{w}_i$ and $\mathbf{r}_i$ are identical to the ith columns of W and R, respectively. Except for the fact that M is block triangular and presumably sparse, this would, in most cases, require a prohibitively large number of computations.

Since we must solve the same system for a number of right-hand sides, it is convenient to first obtain the EFI of M. For the simple case in which M is triangular, this requires $|M|$ operations, where $|M|$ denotes the number of nonzero elements in M. Computing $\mathbf{w}_i$ requires $|\mathbf{r}_i|$ operations to divide the elements of $\mathbf{r}_i$ by the appropriate pivots, and for triangular M, at most $|M| - m$ operations to perform the back substitution. (As is customary, in counting operations we include only multiplications and divisions.) We emphasize at most since in many cases not all the elements of $\mathbf{w}_i$ need be calculated. This is the case when there are trailing zeros in $\mathbf{r}_i$. For instance, if the last k elements of $\mathbf{r}_i$ are zero, then the last k elements of $\mathbf{w}_i$ can immediately be set to zero without calculation. Thus, computation of W requires at most $|R| + |M| + t(|M| - m)$ operations. Similar considerations apply to the calculation of $M^{-1}\mathbf{c}$.

Next consider the computation of SW. This requires at most $t|S|$ operations. Even fewer operations may be required if there are trailing zeros in the $\mathbf{r}_i$ and thus in the $\mathbf{w}_i$. In fact, if the number of trailing zeros in one of the $\mathbf{w}_i$ plus the number of leading zeros in one of the rows of S is greater than or equal to m, the corresponding element of SW can immediately be set to zero without calculation. Similar considerations apply in computing $SM^{-1}\mathbf{c}$.

At this point, the block-triangular system $A^*\mathbf{x} = \mathbf{b}^*$ can be formed without additional multiplicative operations and is readily solved by back substitution.

The computational efficiency of this technique is evident in the fact that as long as t is small relative to n, there will be no $0(n^2)$ or $0(n^3)$ computational processes required. This is the case, since for a sparse matrix, $|M|$, $|R|$, and $|S|$ will all be $0(n)$. The operation counts on which this comment is based were made for triangular M. However, the conclusion is the same for block-triangular M, provided the diagonal blocks are relatively small. This discussion is culminated in the following algorithm.

Algorithm

1. Compute the EFI of M.
2. Let $i = 1$.
3. Compute $\mathbf{w}_i = M^{-1}\mathbf{r}_i$.
4. Compute $S\mathbf{w}_i$ and $\mathbf{t}_i - S\mathbf{w}_i$, where $\mathbf{t}_i$ is the ith column of T.
5. If $i = t$, proceed to step 6; if not, increase i by one and return to step 3.
6. Compute $M^{-1}\mathbf{c}$, $SM^{-1}\mathbf{c}$, and $\mathbf{d} - SM^{-1}\mathbf{c}$.
7. Solve $A^*\mathbf{x} = \mathbf{b}^*$ by block back substitution.

Note that by computing W, SW, and $T - SW$ a column at a time, we economize temporary storage requirements, since the entire W need not be stored.

RANK-ONE BTS

A system in BBTF may also be solved by performing a series of t rank-one splittings rather than the single rank-t splitting described above. Rank-one BTS provides further reduction in fill-in but usually at the expense of some increase in operation count.

Consider the bordered block-triangular system

$$A_1\mathbf{x} = \left[\begin{array}{c|c} M_1 & R_1 \\ \hline S_1 & T_1 \end{array}\right]\mathbf{x} = \left[\begin{array}{c} \mathbf{c}_1 \\ \hline \mathbf{d}_1 \end{array}\right] = \mathbf{b}_1$$

where M_1 is $m \times m$, T_1 is $t \times t$, $\mathbf{c}_1$ is $m \times 1$, and $\mathbf{d}_1$ is $t \times 1$. Again, the goal is to reduce the system to BTF. As a first step, we eliminate the nonzero elements of the first row of S_1. This is done by subtracting from both sides of Equation (1) the vector $\mathbf{e}_{m+1}\mathbf{v}_1^T\mathbf{x}$, where $\mathbf{v}_1^T = [\mathbf{s}^T_{1,1}|\mathbf{s}^T_{1,1}M_1^{-1}R_1]$ and $\mathbf{s}^T_{1,1}$ is a row vector identical to the first row of S_1. Thus we have $(A_1 - \mathbf{e}_{m+1}\mathbf{v}_1^T)\mathbf{x} = \mathbf{b}_1 - \mathbf{e}_{m+1}\mathbf{v}_1^T\mathbf{x}$. Rearranging the right-hand side as before, we obtain the new system $A_2\mathbf{x} = \mathbf{b}_2$, where $A_2 = A_1 - \mathbf{e}_{m+1}\mathbf{v}_1^T$ and $\mathbf{b}_2 = \mathbf{b}_1 - \mathbf{e}_{m+1}\mathbf{s}^T_{1,1}M_1^{-1}\mathbf{c}_1$.

Note that A_2 is identical to A_1 except in the $(m + 1)$th row, whose first m elements are now zero. Thus, A_2 is in BBTF but with the order of the block-triangular part increased by one; that is

$$A_2\mathbf{x} = \left[\begin{array}{c|c} M_2 & R_2 \\ \hline S_2 & T_2 \end{array}\right]\mathbf{x} = \left[\begin{array}{c} \mathbf{c}_2 \\ \hline \mathbf{d}_2 \end{array}\right] = \mathbf{b}_2$$

where M_2 is $(m + 1) \times (m + 1)$, T_2 is $(t - 1) \times (t - 1)$, $\mathbf{c}_2$ is $(m + 1) \times 1$, and $\mathbf{d}_2$ is $(t - 1) \times 1$. Also note that fill-in can occur only in the diagonal blocks of M_1 as it is inverted and in the first row of T_1 as we subtract $\mathbf{s}^T_{1,1}M_1^{-1}R_1$.

Since A_2 is in BBTF, we can now proceed as above, eliminating the nonzero entries in the first row of S_2. This step produces the system $A_3\mathbf{x} = \mathbf{b}_3$, where $A_3 = A_2 - \mathbf{e}_{m+2}\mathbf{v}_2^T$ and $\mathbf{b}_3 = \mathbf{b}_2 - \mathbf{e}_{m+2}\mathbf{s}^T_{2,1}M_2^{-1}\mathbf{c}_2$. Here, $\mathbf{v}_2^T = [\mathbf{s}^T_{2,1}|\mathbf{s}^T_{2,1}M_2^{-1}R_2]$, and $\mathbf{s}^T_{2,1}$ is a row vector identical to the first row of S_2. A_3 is in BBTF with M_3 of order $m + 2$. Additional fill-in can occur only in the first row of T_2. Note that M_2^{-1} is readily obtained, since the first m columns of M_2 contain the same elements as M_1, which has already been inverted.

Finally, after t such steps we arrive at the block triangular system $A_{t+1}\mathbf{x} = \mathbf{b}_{t+1}$, where $A_{t+1} = A_t - \mathbf{e}_{m+t}\mathbf{v}_t^T = A_1 - \Sigma^t_{i=1}\, \mathbf{e}_{m+i}\mathbf{v}_i^T$ and $\mathbf{b}_{t+1} = \mathbf{b}_1 - \Sigma^t_{i=1}\, \mathbf{e}_{m+i}\mathbf{s}^T_{i,1}M_i^{-1}\mathbf{c}$. This amounts to splitting A_1 into the sum of a block-triangular matrix A_{t+1} and t rank-one matrices $\mathbf{e}_{m+i}\mathbf{v}_i^T$, $i = 1, \ldots, t$.

Note that in rank-one BTS the fill-in in T is limited to the area on and above the diagonal, while in rank-t BTS the entire T is subject to fill-in. Usually, however, this reduction in fill-in is obtained at the expense of an increase in operation count, as discussed below.

Rank-one BTS may be implemented in much the same fashion as rank-t BTS. From the standpoint of computational efficiency, the major difference is the number of $\mathbf{w}$ vectors that must be computed. The first step requires computing the $m \times t$ matrix $W_1 = M_1^{-1}R_1$, the second step the $(m + 1) \times (t - 1)$ matrix $W_2 = M_2^{-1}R_2$, etc. Thus a total of $\Sigma^t_{i=1}\, i = t(t + 1)/2$ columns must be computed. On the other hand, rank-t BTS requires the computation of only t such vectors.

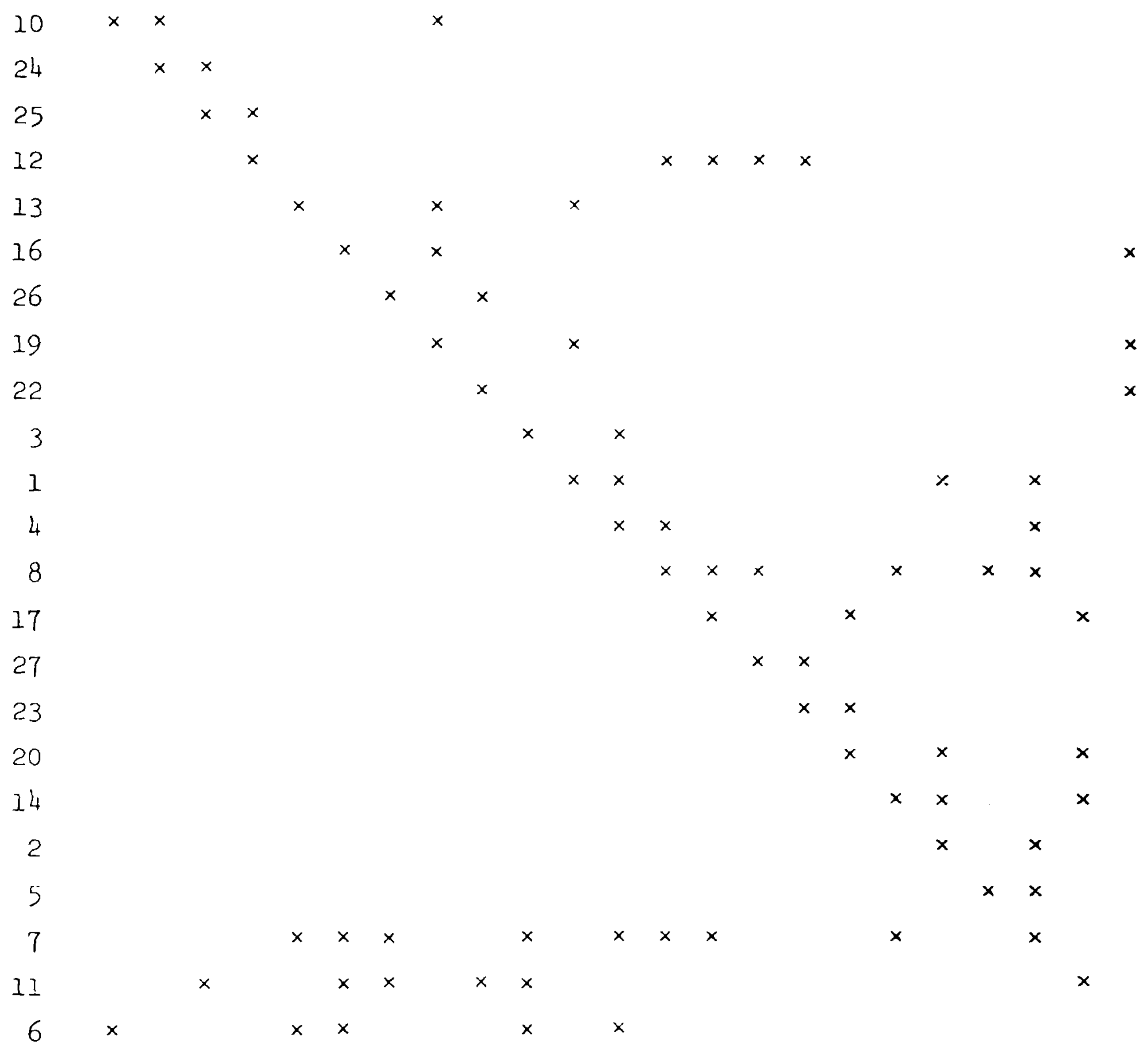

Fig. 1. Occurrence matrix for triple-effect evaporator example discussed in text. Equations and variables are numbered as in Ledet and Himmelblau (1970).

This increase may be balanced, partially at least, by a decrease in the number of elements of the W_i that need actually be calculated. As noted above, trailing zeros in the columns of R_i mean that the corresponding elements of W_i can be set immediately to zero and require no calculation. However, in rank-one BTS the presence of leading zeros in $\mathbf{s}^T_{i,1}$ also affects the calculation of W_i. For instance, if the first k elements of $\mathbf{s}^T_{i,1}$ are zero, then the first k rows of W_i need not be calculated, since W_i is needed only to compute $\mathbf{s}^T_{i,1}W_i$, and in doing so the elements in the first k rows of W_i will just be multiplied by zero. In fact, if the number of leading zeros in $\mathbf{s}^T_{i,1}$ plus the number of trailing zeros in one of the columns of R_i is greater than or equal to $m + i - 1$, then the corresponding column of W_i need not be calculated at all.

Algorithm

To perform rank-one BTS:

1. Let $i = 1$.
2. Compute the EFI of M_i.
3. Let $j = 1$.
4. Compute $\mathbf{w}_{i,j} = M_i^{-1}\mathbf{r}_{i,j}$, where $\mathbf{w}_{i,j}$ and $\mathbf{r}_{i,j}$ are the jth columns of W_i and R_i, respectively. Do not compute the first k elements of $\mathbf{w}_{i,j}$, where k is the number of leading zeros in $\mathbf{s}^T_{i,1}$.
5. Compute $\mathbf{s}^T_{i,1}\mathbf{w}_{i,j}$ and $\mathbf{e}_1{}^T T_i \mathbf{e}_j - \mathbf{s}^T_{i,1}\mathbf{w}_{i,j}$.
6. If $j = t - i + 1$, proceed to step 7; if not, increase j by one and return to step 4.
7. Compute $M_i^{-1}\mathbf{c}_i$, $\mathbf{s}^T_{i,1}M_i^{-1}\mathbf{c}_i$, and $\mathbf{e}_1{}^T\mathbf{d}_i - \mathbf{s}^T_{i,1}M_i^{-1}\mathbf{c}_i$.
8. If $i = t$, proceed to step 9; if not, increase i by one and return to step 2.
9. Solve $A_{t+1}\mathbf{x} = \mathbf{b}_{t+1}$ by back substitution.

EXAMPLE

To demonstrate the savings in fill-in and number of computations that can be realized using BTS, we present a simple example. Consider the bordered block-triangular system given as an occurrence matrix in Figure 1. This system arises in the design of a triple-effect evaporator, as discussed by Ledet and Himmelblau (1970) and later by Stadtherr and Scriven (1974). The particular BBTF used here was obtained using an algorithm designed to increase the likelihood of trailing zeros in the columns of R and leading zeros in the rows of S, which, as discussed above, enhance the computational efficiency of BTS. This algorithm will be described elsewhere. Applying Gaussian elimination to compute the EFI of A for this system requires 207 operations and causes thirty-six fill-ins. For a full **b**, computing **x** from this EFI then requires 113 operations. These figures are obtained using the ordering shown in Figure 1. If the ordering obtained using the Markowitz criterion is used, one can compute the EFI in 193 operations with thirty-one fill-ins and **x** for a full **b** in 108 operations.

Applying rank-t BTS, the EFI of A^* can be computed in 172 operations while causing only seven fill-ins. For a full **b**, computing $\mathbf{b}^*$ and then **x** from the EFI of A^* requires 110 operations. Thus, in this case, by using rank-t BTS we obtain a drastic reduction in fill-in and a modest savings in operation count. It should be pointed out that whether there will be any reduction in number of operations is system dependent, though in any case rank-t BTS can be regarded computationally efficient as long as t is relatively small. For virtually all systems in BBTF, however, a dramatic reduction in fill-in can be obtained.

Applying rank-one BTS, the EFI of A_{t+1} can be obtained in 223 operations while causing just four fill-ins. For a full **b**, to compute $\mathbf{b}_{t+1}$ and then **x** from the EFI of A_{t+1} requires 162 operations. Thus, a nearly 90% reduction in fill-in is obtained but at the expense of an increase in operation count. These results suggest that at installations where storage is at a premium, rank-one BTS may be preferred, while in other situations rank-t BTS may be the choice.

CONCLUDING REMARKS

Instead of using a single rank-t splitting or a series of t rank-one splittings, as described above, one can also use a series of splittings of other rank. For example, a system with $t = 5$ might be solved by using a rank-three splitting followed by a rank-two splitting, or by two rank-two splittings and a rank-one splitting. This permits a reduction in fill-in compared to rank-t BTS and an increase in computational efficiency compared to rank-one BTS. For instance, in the triple-effect evaporator example given above, the system may be solved by a rank-two splitting followed by a rank-one splitting. The EFI of the block-triangular matrix so obtained can be calculated in 185 operations with five fill-ins, and for a full right-hand side the solution can be obtained from this EFI in 140 operations.

Compared to elimination performed on a system in BBTF, BTS would appear to offer no additional numerical difficulties. In both cases it is important to prevent undesirable elements or blocks from becoming pivots in M. This can be accomplished at the time the system is torn by using a tearing algorithm such as Stadtherr et al.'s (1974) which is capable of disallowing certain elements or blocks as candidates for the diagonal of M. However, there is no readily apparent means of foreseeing the numerical properties of $T - SM^{-1}R$ at the time the system is torn. Thus, a difficulty that may arise in solution strategies employing matrices in BBTF is that $T - SM^{-1}R$ is not well-behaved numerically and is difficult to invert with precision. Though serious problems of this nature are unlikely, if they occur it may be necessary to retear the system to produce a different BBTF. Of course this difficulty could be avoided if pivoting techniques rather than a priori permutation to BBTF are used to maintain sparsity.

ACKNOWLEDGMENT

This work was supported in part by a DuPont Young Faculty Grant.

NOTATION

A	=	$[a_{ij}]$ = coefficient matrix
$A^{(k)}$	=	$[a_{ij}^{(k)}] = n \times n$ coefficient matrix remaining after $(k-1)$th step of Gaussian elimination
A_k	=	coefficient matrix after $(k-1)$th step of rank-one BTS
$\overline{A}_k$	=	factor of matrix in BTF
A_{ij}	=	block in matrix in BTF
A^*	=	coefficient matrix after rank-t BTS
A'	=	$A - E_t[S\|0]$
$\mathbf{b}$	=	right-hand side vector
$\mathbf{b}_k$	=	right-hand side vector after $(k-1)$th step of rank-one BTS
$\mathbf{b}^*$	=	right-hand side vector after rank-t BTS
$\mathbf{c}$	=	upper right-hand side vector
$\mathbf{c}_k$	=	upper right-hand side vector after $(k-1)$th step of rank-one BTS
$c_j^{(k)}$	=	number of nonzeros in last $n - k + 1$ positions of jth column of $A^{(k)}$
$\mathbf{d}$	=	lower right-hand side vector
$\mathbf{d}_k$	=	lower right-hand side vector after $(k-1)$th step of rank-one BTS
$\mathbf{e}_k$	=	unit vector
E_t	=	$n \times t$ matrix identical to last t columns of I_n
$\overline{E}_k, \overline{F}_k$	=	$n \times m_k$ matrices identical to the $(1 + \Sigma_{i=1}^{k-1} m_i)$th to the $(\Sigma_{i=1}^{k} m_i)$th columns of I_n and A, respectively
I_n	=	$n \times n$ identity matrix
I	=	identity matrix
L	=	$[l_{ij}]$ = lower triangular factor of A
L_k	=	factor of L
$\mathbf{l}_k$	=	kth column of L
m	=	order of M and M_1
m_k	=	order of A_{kk}
M	=	BTF part of BBTF
M_k	=	BTF part of BBTF after $(k-1)$th step of rank-one BTF
n	=	order of A
N	=	number of diagonal blocks in BTF
P, Q	=	permutation matrices
R	=	upper-right border in BBTF
R_k	=	upper-right border in BBTF after $(k-1)$th step of rank-one BTS
$\mathbf{r}_k$	=	kth column of R
$\mathbf{r}_{k,j}$	=	jth column of R_k
r_j^k	=	number of nonzeros in last $n - k + 1$ positions of jth row of $A^{(k)}$
S	=	lower left border in BBTF
S_k	=	lower left border in BBTF after $(k-1)$th step of rank-one BTS
$\mathbf{s}^T_{k,1}$	=	first row of S_k
t	=	order of T and T_1
T	=	lower right border in BBTF
T_k	=	lower right border in BBTF after $(k-1)$th step of rank-one BTS
$\mathbf{t}_k$	=	kth column of T
U	=	upper triangular factor of A
U_k	=	factor of U
$\mathbf{u}_k$	=	kth column of U
$\mathbf{v}_k^T$	=	$[s^T_{k,1}\|s^T_{k,1}M_k^{-1}R_k]$

$W \;= M^{-1}R$
$W_k \;= M_k{}^{-1}R_k$
$\mathrm{w}_k \;=$ kth column of W
$\mathrm{w}_{k,j} \;=$ jth column of W_k
$\mathrm{x} \;=$ unknown vector

Superscript

$T \;=$ transpose (row vector)

LITERATURE CITED

Barkley, R. W., and R. L. Motard, "Decomposition of Nets," *Chem. Eng. J.*, 3, 265 (1972).
Christensen, J. H., "The Structuring of Process Optimization," *AIChE J.*, 16, 177 (1970).
———, and D. F. Rudd, "Structuring Design Computations," *ibid.*, 15, 94 (1969).
Duff, I. S., "A Survey of Sparse Matrix Research," *Proc. IEEE*, 65, 500 (1977).
George, A., "On Block Elimination for Sparse Linear Systems," *SIAM J. Numer. Anal.*, 11, 585 (1974).
Hlavacek, V., "Analysis of a Complex Plant—Steady State and Transient Behavior," *Comput. Chem. Eng.*, 1, 75 (1977).
Kevorkian, A. K., and J. Snoek, "Decomposition of Large Scale Systems," in *Decomposition of Large Scale Problems*, Proc. NATO Conf. at Cambridge, July 1972, D. M. Himmelblau, ed., pp. 491-515, North Holland, Amsterdam (1973).
Ledet, W. P., and D. M. Himmelblau, "Decomposition Procedures for the Solving of Large Scale Systems," *Adv. Chem. Eng.*, 8, 185 (1970).
Markowitz, H. M., "The Elimination Form of the Inverse and its Application to Linear Programming," *Management Sci.*, 3, 255 (1957).
Stadtherr, M. A., W. A. Gifford, and L. E. Scriven, "Efficient Solution of Sparse Sets of Design Equations," *Chem. Eng. Sci.*, 29, 1025 (1974).
Stadtherr, M. A., and L. E. Scriven, "Applications of a New Algorithm for Scientific and Design Calculations," *Lat. Am. J. Chem. Eng. Appl. Chem.*, 4, 47 (1974).
Tarjan, R., "Depth First Search and Linear Graph Algorithms," *SIAM J. Comput.*, 1, 146 (1972).
Tewarson, R. P., *Sparse Matrices*, Academic Press, New York (1973).
Westerberg, A. W., and F. C. Edie, "Computer-Aided Design. I. Enhancing Convergence Properties by the Choice of Output Variable Assignments in the Solution of Sparse Equation Sets," *Chem. Eng. J.*, 2, 9 (1971*a*).
———, "Computer-Aided Design. II. An Approach to Convergence and Tearing in the Solution of Sparse Equation Sets," *ibid.*, 17 (1971*b*).

Manuscript received August 14, 1978; revision received March 2, and accepted March 23, 1979.

Characteristics of Macromolecular Gel Layer Formed on Ultrafiltration Tubular Membrane

SHIN-ICHI NAKAO
TSUYOSHI NOMURA
and
SHOJI KIMURA

Institute of Industrial Science
University of Tokyo
7-22-1 Roppongi, Minato-ku
Tokyo, Japan 106

Characteristics of the gel layer were investigated by direct measurement of its concentration treating polyvinylalcohol and ovalbumin aqueous solutions, using cellulose acetate tubular ultrafiltration membranes. The concentration of the gel layer was not constant but a function of bulk concentration and feed velocity. The mass transfer coefficient obtained agreed with Deissler correlation. There was a definite correlation between the resistance of the gel layer and its concentration.

SCOPE

Ultrafiltration is now not only a laboratory curiosity, but it is also an important industrial process. The recent development of many asymmetric high flux membranes has made practical applications possible in a large number of fields. Porter and Michaels (1971*a*, *b*, *c*, *d*, 1972) have reviewed the applications of ultrafiltration in the areas of concentration, recovery, and so on. In the field of wastewater treatment, many applications of ultrafiltration have been reported.

The development of high flux membranes, however, has brought with it the problem of a gel layer formation of macromolecules on the membrane surface. This layer has such a large permeation resistance that the product flux decreases considerably.

Ultrafiltration fluxes have been treated theoretically using the gel polarization model, which deals with the boundary layer between the bulk solution and the gel layer on a membrane surface. But there are two questionable points in this model. One is the assumption that the concentration of the gel layer (C_g) is constant. If C_g is, in fact the gelling concentration, it should change only with the kind of materials used, not with the types of apparatus or modules, or with the experimental conditions. But there have been no experimental results to support this assumption, and the value of C_g has never been measured directly by the previous experiments. The assumption that C_g is constant is quite doubtful and needs experimental verification.

The other problem has to do with the mass transfer coefficient (k). The gel polarization model itself does not specify what k should be used. It is not certain whether the usual k can be applied to ultrafiltration, where the Schmidt number is very large. Moreover, k of macromolecules is considered to change with the bulk concentration when the viscosity and diffusivity also change.