OPTIMAL ELIMINATION FOR SPARSE SYMMETRIC SYSTEMS AS A GRAPH PROBLEM*

BY

W. R. SPILLERS AND NORRIS HICKERSON
Columbia University

Abstract. The optimal (requiring the minimum number of multiplications) ordering of a sparse symmetric system of linear algebraic equations to be used with Gaussian elimination is first developed as a graph problem which is then treated using the functional equation techniques of dynamic programming. A simple algorithm is proposed as an alternative to the more lengthy procedures of dynamic programming and this algorithm is shown to be effective for systems whose graphs are "grids".

Introduction. The motivation for this work is the fact that there exists a large class of physical problems which give rise to sparse symmetric linear systems, for which the computational effort required to obtain a solution by elimination is highly dependent upon the ordering of the equations. Here a system of n linear algebraic equations

$$A'x' = b' \tag{1}$$

is called symmetric if the coefficient matrix is symmetric and sparse if A' has a large number of zero elements. In many problems dealing with structures, networks, finite difference formulations, etc., this is precisely the case. Certainly if there are no zero elements, there is no such thing as an optimum procedure in the sense in which the term is used here.

The origins of this work can be traced back to Kron [1] in the work which he calls "Diakoptics" and more recently to the work of Branin [2] and Roth [3]. It has been pointed out (see also [4], [5]) that to solve these sparse systems by first computing the inverse system matrix can be highly inefficient, and that Gaussian elimination, which is in fact a special case of one of Kron's techniques, is apparently the most efficient procedure, excluding special cases such as, e.g., systems which are highly symmetric (systems in which there is much repetition of elements or groups of elements).

There are now digital computer programs available for the automatic analysis of many physical systems. Since the computational effort and therefore the cost is sensitive to the procedure used, it is important to proceed efficiently. In the following, Gaussian elimination is first developed as a graph problem; its functional equation is then treated using the dynamic programming techniques of Bellman [6]; and finally a simple algorithm is discussed which is a computationally attractive alternative to the dynamic programming procedures and which can be easily included in computer programs for automatic analysis.

Gaussian elimination. Given a system in the form of Eq. (1), the question considered here is how to find the solution matrix x' using elimination, so that the computing time, and therefore computing cost, is as small as possible. Following von Neumann, the number of multiplications required will be counted as a measure of the computing time.

*Received April 11, 1966; revised manuscript received July 21, 1967.

Consider the distinct phases, elimination and backsubstitution. Generally, a typical step in the elimination phase consists of using, e.g., the ith row of A' to remove all the nonzero terms except a'_{ij} in the jth column by taking linear combinations of rows. Here, a restrictive form is used in which the ith row is used only to delete terms in the ith column. This implies that the system is well conditioned; it also, however, makes the problem tractable. In the backsubstitution phase, known components of the solution matrix x' are used to compute, e.g., the unknown ith component.

In this restricted form, the numerical procedure is completely specified for a given system once the equations have been ordered. Further, if only the number of multiplications are to be counted, a reduced matrix A whose elements are defined to be

$$\begin{aligned} a_{ij} &= 1 \quad \text{if} \quad a'_{ij} \neq 0 \\ &= 0 \quad \text{if} \quad a'_{ij} = 0 \end{aligned} \tag{2}$$

contains all the information necessary for the problem under discussion. (Note that at this step, zeros which are generated "accidentally" by the elimination scheme are neglected.)

The number of multiplications required may be counted in the following manner. Assume for the moment that the equations are ordered in A in accordance with the order in which elimination is to proceed. Let b_i $(i = 1, \cdots, n)$ be the number of nonzero terms remaining in the ith row *after* the elimination phase has been completed. The number of multiplications for a system with one right-hand side is (using symmetry):

For the ith row—

Step 1	Divide all remaining terms in the ith row by the diagonal term	b_i	mult.
Step 2	Eliminate the terms below the ith row in the ith column	$b_i(b_i - 1)$	mult.
Step 3	Backsubstitution	$b_i - 1$	mult.
Total for the ith row		$(b_i + 1)b_i - 1$	

Total for the system $\phi = \sum_{i=1}^{n} (b_i + 1)b_i - n.$

In general the system will be given some arbitrary initial ordering 1, 2, 3, $\cdots$, n. Let ξ_i $(i = 1, \cdots, n)$ describe a permutation of this ordering. The problem is then to find the ξ_i for which $\phi(\xi_i)$ is a minimum. Since the matrix A may be thought of as purely topological, this is a combinatorial problem concerned only with the topology (connectivity) of the system.

Problems of this kind fit into the very general, but here not very useful, classification of nonlinear programming problems. Bellman [7] has discussed the solution of several related problems, remarking on their deceptively simple appearance and the lack of any systematic theory for their treatment. Motzkin and Straus [8] consider a combinatorial problem which includes the present problem, but offer a solution with restrictions into which it has not yet been possible to fit the present work. A point of general agreement among these workers is that the large number of possibilities obviates solution by trial and error for any but the most trivial examples.

As a final example of related work, in [5] the solution for the problem of minimizing

the average "band width" has been given as the solution of a linear programming problem. It may be noted that not only is a finer measure of computational effort used here, but also that while the computational effort required by the scheme described in [5] obviates its application to any but the most trivial cases, the algorithm proposed here is simple enough to be quite useful.

A graph problem. It was pointed out in the preceding section that optimal elimination is actually a topological problem. In this section this property is exploited by reformulating the problem using notation from graph theory [9] in order to facilitate the visualization of procedures and simplify the proofs.

Some systems, such as networks and skeletal structures, may be thought of as being their own graph. That is, a picture of one of these physical systems, with slight modification, could serve as its graph in the following discussion in spite of the fact that there may be more than one scalar equation associated with each node (see later). Other systems, such as those which arise from difference equations, may have no natural graph. For these systems the following procedure may be used to construct a graph: With each equation in A there is associated a node in the system graph and with each nonzero term a_{ij} $(i = 1, \cdots, n; j = i + 1, \cdots, n)$ there is associated an undirected branch between the ith and jth nodes.

This system graph will be referred to as G. During the elimination procedure it is modified just as the rows of the system matrix are modified; let G^i refer to the graph formed by the elimination of the ith node from G. Let G_i $(i = 1, \cdots, n)$ refer to the set consisting of the ith node together with all nodes adjacent to the ith node. (Two nodes i and j are adjacent if they are both endpoints of the same branch.) Elimination of the ith row from A corresponds to first forming for each node j in the graph

$$\begin{aligned} \bar{G}_j^i &= G_i \cup G_j \qquad j \in G_i \\ &= G_j \qquad\quad j \notin G_i \end{aligned} \qquad (j = 1, \cdots, n) \tag{3}$$

and then deleting the ith node from each $\bar{G}_j^i$ for which $i \in \bar{G}_j^i$ to form G_j^i. The implications of this on the system graph are shown in Fig. 1. Successive eliminations proceed in a straightforward manner; e.g., elimination next of the kth node would produce the sets $G_j^{i,k}$. Introducing the norm $|G_j^{i,\cdots,m}|$ which is defined to be the number of nodes included in the set $G_j^{i,\cdots,m}$, for an ordering $\xi_1, \xi_2, \cdots, \xi_n$ it is possible to identify $b_i = |G_i^{\xi_1,\xi_2,\cdots,\xi_{i-1}}|$ which is in fact one plus the *valence* of the node ξ_i in the graph $G^{\xi_1,\xi_2,\cdots,\xi_{i-1}}$. (The *valence* of a node is defined to be the number of branches having that node for an endpoint.) Since deletion of a vertex I alters the remaining graph only by addition of branches JK if IJ and IK were branches, but not JK, we have clearly the following result.

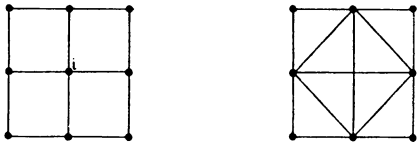

FIG. 1

EFFECT OF DELETING THE iTH NODE

THEOREM 1. *Any graph which is obtained from another graph by deleting nodes in an elimination procedure is independent of the order in which the nodes are eliminated.*

THEOREM 2. *When any two successive nodes in an optimum ordering are adjacent either in the original graph or in the graphs generated by the elimination procedure, the node with the smaller valence occurs first.*

Let 1, 2, 3, $\cdots$, n be an optimum ordering of the nodes. The theorem states that if node $(i + 2) \in G_{i+1}^{1,2,\cdots,i}$, then $|G_{i+1}^{1,2,\cdots,i}| \leq |G_{i+2}^{1,2,\cdots,i}|$. It is sufficient to show this for the first two nodes in the optimum ordering. From Theorem 1 and its discussion, it follows that if $|G_1| > |G_2|$, then the order of these nodes could be interchanged with a decrease in the value of b_1, all other b_i $(i = 2, \cdots, n)$ remaining unchanged. It would then follow that the ordering of the nodes 2, 1, 3, $\cdots$ requires fewer multiplications than the ordering 1, 2, 3, $\cdots$, contradicting the assumption that the latter is optimal.

The functional equation. Let $\mathbf{W(G)}$ be defined to be the number of multiplications required to solve optimally the system whose graph is G; let d_i be one plus the degree of the ith node in graph G; and let $e_i = d_i(d_i + 1) - 1$. ($W(G)$ is, in fact, the minimum value of $\phi(\xi_i)$ and e_i is the number of multiplications required to eliminate the ith row first from the given system whose graph is G.) $W(G)$ satisfies the functional equation

$$W(G) = \min_i [e_i + W(G^i)]. \tag{4}$$

A. *Uniqueness.* Bellman uses the term "policy" to describe a specific permutation; i.e., a certain policy results in a permutation for which it is then possible to evaluate the number of multiplications or work. The optimal policy corresponds to minimum work. It will now be shown that the solution of Eq. (4) is unique while the optimal policy may not be. Let W and Y be two solutions of Eq. (4)

$$W(G) = \min_i [e_i + W(G^i)] \qquad Y(G) = \min_i [e_i + Y(G^i)] \tag{5}$$

and I and J be the respective values of the index i for which the minimums occur. Since it is of no consequence in the proof, it is assumed arbitrarily that $W(G) \geq Y(G)$. Then

$$W(G) = e_I + W(G^I) \leq e_J + W(G^J), \qquad Y(G) = e_J + Y(G^J) \tag{6}$$

and

$$|W(G) - Y(G)| \leq |W(G^J) - Y(G^J)|. \tag{7}$$

These steps may now be repeated for the graph G^J and then the graph $G^{J,K}$, etc., with the difference (Eq. (7)) in the solutions growing larger or remaining the same as each node is eliminated. But by the definition of the elimination procedure, the work required to solve any system whose graph is a single node is just 1. So that

$$W(G^{J,K\cdots}) = Y(G^{J,K,\cdots}) = 1 \tag{8}$$

and therefore

$$W(G) = Y(G). \tag{9}$$

B. *Approximation in Policy Space.* Following Bellman, it is possible to choose any initial policy (method of ordering the nodes) and proceed iteratively to obtain the solution of Eq. (4). Let $W_0(G)$ be the number of multiplications (work) required using this

initial policy on the graph G. Let

$$W_N(G) = \min_i [e_i + W_{N-1}(G^i)] \qquad N = 1, 2, 3, \cdots . \tag{10}$$

The sequence $\{W_N(G)\}$ is clearly monotone-decreasing. First

$$W_0(G) = e_I + W_0(G^I) \tag{11}$$

and

$$W_1(G) = \min_i [e_i + W_0(G^i)] \leq e_I + W_0(G^I) = W_0(G). \tag{12}$$

Then, inductively, assume $W_N(G) \leq W_{N-1}(G)$ from which if

$$W_N(G) = \min_i [e_i + W_{N-1}(G^i)] = e_J + W_{N-1}(G^J) \tag{13}$$

then

$$W_{N+1}(g) = \min_i [e_i + W_N(G^i)] \leqq e_J + W_N(G^J) \leq W_N(G). \tag{14}$$

It follows that

$$\lim_{N\to\infty} W_N(G) = W(G) \tag{15}$$

is a solution of Eq. (4). If the policy chosen produces exact results when applied to a system whose graph has a single node, a detailed examination of the iterative scheme shows that at most $n - 1$ iterations are actually required for the sequence to converge.

C. Monotone-increasing Convergence. An alternative iterative scheme might be used in which the convergence is monotone-increasing. Using Eq. (10) again to iterate but starting with

$$W_0(G) = \min_i [e_i] = e_I \tag{16}$$

then

$$W_1(G) = \min_i [e_i + W_0(G^i)] \geq e_I = W_0(G). \tag{17}$$

Proceeding inductively, if $W_N(G) \geq W_{N-1}(G)$, then

$$W_{N+1}(G) = \min_i [e_i + W_N(G^i)] \geq \min_i [e_i + W_{N-1}(G^i)] = W_N(G). \tag{18}$$

This sequence will next be shown to be bounded. Let $W(G)$ be the solution of Eq. (4). Obviously

$$W_0(G) \leq W(G). \tag{19}$$

By induction, if $W_N(G) \leq W(G)$, then

$$W_{N+1}(G) = \min_i [e_i + W_N(G^i)] \leq \min_i [e_i + W(G^i)] = W(G). \tag{20}$$

The sequence $\{W_N(G)\}$ again converges to the solution of Eq. (4) as $N \to \infty$ and again at most $n - 1$ iterations are actually required.

The proposed algorithm. While the above iterative procedures provide the formal

solution of the problem of the optimal ordering of the nodes, they involve considerable computational effort themselves. If it is necessary to deal with systems containing thousands of nodes, it appears unlikely that they could be of any practical use for the optimization of an elimination scheme in the automatic analysis of linear systems. In this section an algorithm or "policy" is discussed which is simple to automate and which requires almost a minimal amount of calculations to execute. The authors have found this algorithm useful and, in fact, know of no examples for which it does not produce an optimal ordering.

Proposed Algorithm. At each step in the elimination scheme, eliminate the node with the smallest degree.

Note that if it were not for the fact that the graph is modified at each step in the elimination, the execution of this algorithm would be completely trivial.

There are a number of ways to motivate this choice of an algorithm. Thinking in terms of problems such as the "traveling salesman problem" in which it is necessary to take a given number of steps, the proposed algorithm requires that the "smallest" step be taken each time. Flood [10] and Motzkin [11] have noted that this policy can be very effective. Bellman [12] has also discussed a policy as an approximation to the optimal policy for multistage decision processes. Finally, Theorem 2 above provides additional motivation for this algorithm. Certainly, it is difficult to find a more simple algorithm with any relevance.

Square grids. Figure 2 shows a system graph which occurs frequently in physical applications together with a common numbering scheme. This numbering scheme results in a fairly narrow "band width" and is quite convenient to use in computer applications. Figure 3 shows a comparison of the number of multiplications required to solve this system using the indicated ordering, with the number of multiplications required using an ordering which satisfies the proposed algorithm. Note that the number of equations is n^2. As the size of the system increases, the relative efficiency of the ordering scheme proposed here is seen to increase.

For large grids it is possible to follow both of these schemes a few steps in order to attempt to understand their operation. Assume that in the execution of the proposed algorithm the grid is large enough so that it is possible to neglect the edges and assume

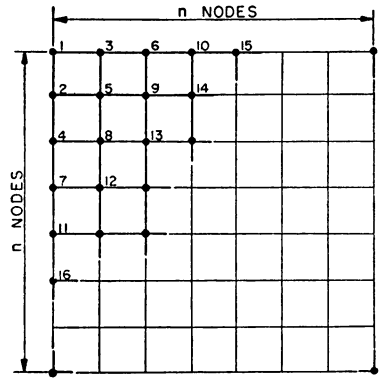

FIG. 2

nxn SQUARE GRID

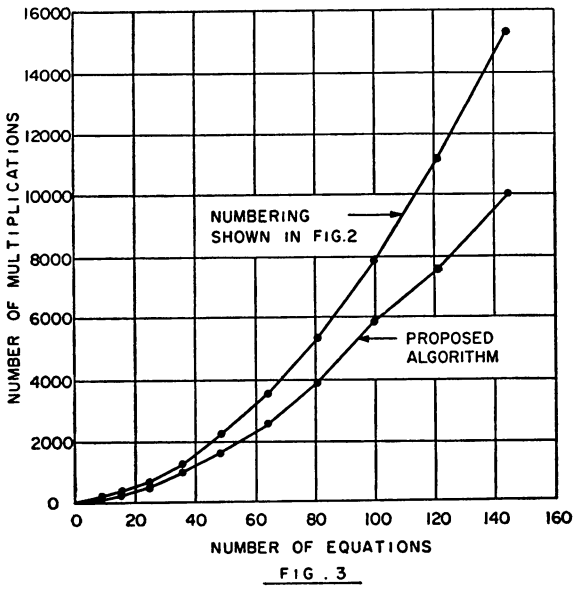

FIG. 3

A COMPARISON FOR SQUARE GRIDS.

that initially each node has the degree 4. It is possible to eliminate $n^2/2$ alternate nodes systematically so that the degree of the node eliminated remains four. Using the common numbering scheme shown in Fig. 2, during the elimination of the first $n^2/2$ nodes, the degree of the node eliminated increases from 3 to $n + 2$. As the elimination process continues it becomes more and more difficult to follow the proposed algorithm, but it can be seen that the next $n^2/8$ nodes eliminated have the degree eight.

A generalization. When the number of variables associated with each node is not one but greater than one, the count of the multiplications must be modified slightly. However, if matrix multiplications are assigned the same work as matrix divisions, the preceding remark can be applied interpreting ϕ as proportional to rather than equal to the total work. Without this simplification the problem appears intractable.

Built-up systems. It is sometimes convenient to separate a system into component parts, such as wings and fuselage in an aircraft, and to deal with each of these parts independently before they are interconnected. The only concern is the effect of this on the use of the proposed algorithm.

Let nodes 1, 2, $\cdots$, s belong to a subgraph $S \subset G$ which is to be considered independently. Under the assumption that G is a connected graph, $G^{1,2,\cdots,s}$ has branches between each node pair in $R = G - S$ which is adjacent to nodes in S. Therefore, if it is required to eliminate the nodes in S first, without so doing, the ordering of the nodes in R can proceed using the proposed algorithm after adding branches between the above node pairs where required.

Acknowledgment. This work was supported by the National Science Foundation.

References

[1] Gabriel Kron, *Diakoptics*, Macdonald, London, 1963. (References to most of Kron's work can be found in this collection.)

[2] Franklin H. Branin, *The relation between Kron's method and the classical methods of network analyses*, IRE WESCON Convention Record, Part 2, 1959, pp. 1–29

[3] J. Paul Roth, *An application of algebraic topology: Kron's method of tearing*, Quart. Appl. Math. **17**, 1 (1959)
[4] W. R. Spillers, *Network techniques applied to structures*, The Matrix and Tensor Quarterly **15**, 2 (1964)
[5] W. R. Spillers, *On diakoptics: tearing an arbitrary system*, Quart. Appl. Math. **23**, 2 (1965)
[6] Richard Bellman, *Dynamic programming*, Princeton Univ. Press, Princeton, N. J., 1957
[7] Richard Bellman, *Combinatorial processes and dynamic programming*, Report P-1284, The RAND Corporation, February 24, 1958
[8] T. S. Motzkin and E. G. Straus, *Some combinatorial extremum problems*, Proc. Amer. Math. Soc. **7**, 1014–1021 (1956)
[9] Oystein Ore, *Theory of graphs*, Amer. Math. Soc. Colloq. Publ., Vol. 38, Amer. Math. Soc., Providence, R. I., 1962
[10] Merrell M. Flood, *The traveling-salesman problem*, J. Operations Research Society of America **4**, 61 (1956)
[11] T. S. Motzkin, *The assignment problem*, Proc. Sympos. Appl. Math. Vol. 6, Amer. Math. Soc, Providence, R. I., 1956
[12] Richard Bellman, *Decision making in the face of uncertainty*—I, Naval Research Logistics Quart. (3) **1**, 230–232 (1954)