INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING, VOL. 14, 837–853 (1979)

# A COMPARISON OF THREE RESEQUENCING ALGORITHMS FOR THE REDUCTION OF MATRIX PROFILE AND WAVEFRONT

GORDON C. EVERSTINE

*Numerical Mechanics Division, David W. Taylor Naval Ship R&D Center, Bethesda, Maryland, U.S.A.*

## SUMMARY

Three widely-used nodal resequencing algorithms were tested and compared for their ability to reduce matrix profile and root-mean-square (rms) wavefront, the latter being the most critical parameter in determining matrix decomposition time in the NASTRAN finite element computer program. The three algorithms are Cuthill–McKee (CM), Gibbs–Poole–Stockmeyer (GPS), and Levy. Results are presented for a diversified collection of 30 test problems ranging in size from 59 to 2680 nodes. It is concluded that GPS is exceptionally fast, and, for the conditions under which the test was made, the algorithm best able to reduce profile and rms wavefront consistently well. An extensive bibliography of resequencing algorithms is included.

## BACKGROUND

Many problems of scientific and engineering interest reduce to the numerical problem of solving a large set of linear algebraic equations such as, in matrix form,

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

where the vector $\mathbf{b}$ and the square matrix $\mathbf{A}$ are known, and the unknown vector $\mathbf{x}$ is sought. In finite element and other applications, $\mathbf{A}$ is also sparsely populated (i.e., it contains far more zeros than non-zeros), since the procedure under which finite element matrices are assembled dictates that the off-diagonal matrix terms coupling any two degrees-of-freedom are zero unless those degrees-of-freedom are common to the same finite element.[1,2] It also follows that the locations of the non-zero elements of the matrix $\mathbf{A}$ depend solely on the ordering of the unknowns. In finite element applications, for example, the ordering of the unknowns corresponds to the selection of grid point (or nodal) numbers for the mesh points. Thus, it is possible with sparse matrices to choose an ordering which results in the non-zeros being located in a way convenient for subsequent matrix operations such as equation solving or eigenvalue extraction. A good ordering is essential to the finite element user since virtually all finite element computer programs contain equation solution and eigenvalue routines which have been expressly written to operate efficiently on matrices possessing small bandwidth, profile, or wavefront. Efficiency is obtained by avoiding arithmetic operations on matrix elements known to be zero. The execution time for a 'band solver,' for example, is $O(NB^2)$ for large $N$ and $B$, where $N$ is the matrix order and $B$ the bandwidth. For a given finite element model, $N$ is fixed, but $B$ depends on the ordering of the unknowns (grid points). Clearly, in this case, it is desirable to reduce $B$ as much as possible.

0029–5981/79/0614–0837$01.00


*Received 12 May 1978*
*Revised 2 August 1978*

Unfortunately, it is often difficult to know how to sequence the nodes to effect a good numbering, particularly for large complicated meshes or those generated automatically on a computer. Even if known, a good sequence may be tedious to implement. A large number of algorithms have therefore been developed[3–51] to automate the assignment of grid point labels, given the connectivity of the mesh. Since it is clearly impractical to check each of the $N!$ possible sequences associated with a given matrix $\mathbf{A}$ of order $N$, each algorithm attempts some presumably rational strategy for arriving quickly at a good, but not necessarily optimal, grid point sequence.

The large number of available algorithms presents the potential user with a difficult choice. Ideally, each developer of a new resequencing scheme would test his method against the competition, and many developers claim to have done so. (Some additional comparisons appear in References 52–61.) However, most comparisons are inadequate because the test problems selected are either too small or too few. Moreover, there is no general agreement among developers on which of their predecessors' algorithms are the most effective and hence the most logical to use for comparison.

Since there is no known way of evaluating resequencing strategies on strictly theoretical grounds, the comparison must be done empirically on a computer. Thus, there appears to be a need for a set of test problems which would be available to any interested researcher. To be most useful, the set should consist of a large number of diversified problems and should include a selection of large problems. A large number is needed to determine which strategy performs well consistently, not on just a few fortuitously chosen problems. (A review of the relevant literature indicates that no one strategy is best for all problems.) On the other hand, large problems are essential since proper grid point sequencing is more important with large problems than with small ones. Of course, the criteria for what is 'large' will vary with the user and will depend on the computer and financial resources available. The author, for example, classifies problems with over 1000 nodes as 'large', but even this rough guideline changes for field problems having only one degree-of-freedom (DOF) per node.

The purposes of this paper are thus two-fold: (i) to present a large set of diversified problems which can be used to evaluate resequencing strategies, and (ii) to show how three widely-used algorithms perform on this collection with regard to the reduction of matrix profile and root-mean-square (rms) wavefront.†

The reasons for selecting rms wavefront for these tests are, first, that it is the parameter of primary interest to users of the widely-used NASTRAN[62] finite element computer program, and, second, that the author is unaware of any previous rms wavefront comparisons. In NASTRAN, the computer CP time $T$ required to perform a symmetric matrix decomposition (triangular factorization) is given approximately by[62]

$$T - \tfrac{1}{2}NW_{\text{rms}}^2\,(T_0) \tag{2}$$

where

$N$ = matrix order
$W_{\text{rms}}$ = rms wavefront of matrix
$T_0$ = machine time constant

For example, on a CDC 6100 computer, $T_0$ is usually taken as $15 \times 10^{-6}$ sec. Only the dominant term in the complete NASTRAN timing equation is given in equation (2).

The primary reason for selecting the three algorithms to compare (Cuthill–McKee (CM),[17]

† See the next section for definitions.

Gibbs–Poole–Stockmeyer (GPS),[26] and Levy[32]) is that these are the algorithms most often run by NASTRAN users. Both CM and GPS are included in the BANDIT program,[55,63,64] a NASTRAN preprocessor developed to reduce, at the user's option, matrix bandwidth, profile, or wavefront. The Levy scheme is in a NASTRAN preprocessor called WAVEFRONT.[34]

Subsequent sections of this paper present precise definitions of the relevant terms, a description of the collection of 30 test problems, a brief description of the three algorithms to be tested and the ground rules of the test, and the test results.

## DEFINITIONS

Although the definitions given here are reasonably standard, at least in finite element circles, uniformity of definitions and notation among the various workers in the field does not yet exist.

Given a symmetric square matrix **A** of order $N$, we define a 'row bandwidth' $b_i$ for row $i$ as the number of columns from the first non-zero in the row to the diagonal, inclusive. Numerically, $b_i$ exceeds by unity the difference between $i$ and the column index of the first non-zero entry of row $i$ of **A**. Then the matrix bandwidth $B$ and profile $P$ are defined as

$$B = \max_{i \leq N} b_i \tag{3}$$

$$P = \sum_{i=1}^{N} b_i \tag{4}$$

Let $c_i$ denote the number of active columns in row $i$. By definition, a column $j$ is active in row $i$ if $j \geq i$ and there is a non-zero entry in that column in any row with index $k \leq i$. The matrix wavefront $W$ is then defined as

$$W = \max_{i \leq N} c_i \tag{5}$$

Sometimes $c_i$ is referred to as the row wavefront for row $i$. Since the matrix **A** is symmetric,

$$P = \sum_{i=1}^{N} b_i = \sum_{i=1}^{N} c_i \tag{6}$$

The wavefront $W$ is sometimes called the maximum wavefront $W_{\max}$ to distinguish it from the average wavefront $W_{\text{avg}}$ and root-mean-square wavefront $W_{\text{rms}}$ defined as

$$W_{\text{avg}} = \frac{1}{N} \sum_{i-1}^{N} c_i = \frac{P}{N} \tag{7}$$

$$W_{\text{rms}} = \sqrt{\left(\frac{1}{N} \sum_{i=1}^{N} c_i^2\right)} \tag{8}$$

From these definitions, it follows that, for a given matrix,

$$W_{\text{avg}} \leq W_{\text{rms}} \leq W_{\max} \leq B \leq N \tag{9}$$

The first two inequalities would be equalities only for uninteresting special cases such as diagonal matrices.

We define the degree $d_i$ of node $i$ as the number of other nodes to which it is connected; i.e., more precisely, $d_i$ is the number of non-zero off-diagonal terms in row $i$ of the matrix **A**. (This implies, for example, that diagonally opposite nodes in a quadrilateral finite element are

'connected' to each other.) Hence, the maximum nodal degree $M$ is

$$M = \max_{i \leq N} d_i \tag{10}$$

The number of unique edges $E$ is defined as the number of non-zero off-diagonal terms above the diagonal. Hence, for a symmetric matrix,

$$E = \tfrac{1}{2} \sum_{i=1}^{N} d_i \tag{11}$$

Thus the total number of non-zeros in $\mathbf{A}$ is $2E + N$, and the density $\rho$ of the matrix $\mathbf{A}$ is

$$\rho = (2E + N)/N^2 \tag{12}$$

Note that, in these definitions, the diagonal entries of the matrix $\mathbf{A}$ are included in $b_i$ and $c_i$ (and hence in $B$, $P$, $W_{\max}$, $W_{\text{avg}}$, and $W_{\text{rms}}$). These definitions make it easy to convert the various parameters from one convention (including the diagonal) to the other (not including the diagonal).

Also note that, in this context, the order $N$ of the matrix $\mathbf{A}$ is sometimes taken to be the same as the number of nodes. In general finite element usage, however, each node (grid point) has several degrees-of-freedom (DOF), not just one. For structures having, say, six DOF per node, the actual DOF values of $B$, $W_{\max}$, $W_{\text{avg}}$, or $W_{\text{rms}}$ would be, in the absence of constraints, six times their corresponding grid point values.

*Example*

Definitions (3)–(12) can be illustrated by the following simple example. Figure 1 shows a matrix of order six. In each row and column a line is drawn from the first non-zero to the

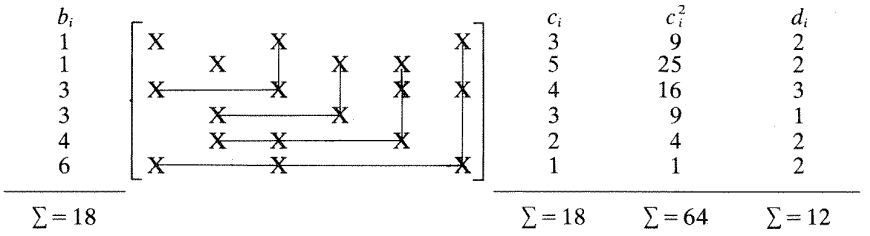



Figure 1. Example illustrating definitions of matrix bandwidth, profile, and wavefront

diagonal. Thus $b_i$ is the number of columns traversed by the solid line in row $i$. Similarly, the number of active columns $c_i$ in row $i$ is the number of vertical lines in row $i$ to the right of and including the diagonal. Thus, from the definitions, $B = 6$, $W_{\max} = 5$, $P = 18$, $W_{\text{avg}} = 3{\cdot}0$, $W_{\text{rms}} = 3{\cdot}3$, $M = 3$, $E = 6$, and $\rho = 50{\cdot}0$ per cent.

## THE TEST PROBLEMS

During the development of the grid point resequencing preprocessor to NASTRAN known as BANDIT,[55,63,64] a collection of test problems was assembled from NASTRAN users represent-

ing various U.S. Navy, Army, Air Force, and NASA laboratories. An early version of this collection was given to Gibbs *et al.* at the College of William and Mary for testing their algorithm.[26,56] However, plots were no longer available for many of those problems. A later version of the collection was used in the author's comparison[55] (for matrix bandwidth reduction) of the Cuthill–McKee (CM) strategy[17] with the Gibbs–Poole–Stockmeyer (GPS)[26] approach. None of the papers cited presented any detailed descriptions or plots of the test problems being used. The collection, now numbering 30, has been improved by adding more large problems and by replacing problems without plots by others with plots. Since the 30 problems have been collected from finite element users in a variety of engineering disciplines and range in size from 59 to 2680 nodes, the collection is probably large enough and diversified enough to provide a good test of nodal resequencing algorithms. (In fact, since the collection gives connectivity information for actual problems, it would probably also be of use to developers of equation solution and eigenvalue extraction routines.)

Complete descriptions of the 30 test problems are given in Table I. All column headings in the table have already been defined except for column 2, 'File no.,' which is the file number on the

Table I. Test problem statistics

| No. of grid points ($N$) | File no. | Max. nodal degree ($M$) | No. of unique edges ($E$) | Matrix density (%) | band-width | profile | Max. wave-front | Avg. wave-front | rms wave-front | |
|---|---|---|---|---|---|---|---|---|---|---|
| 59 | 6 | 5 | 104 | 7·67 | 26 | 464 | 11 | 8· | 8· | 2-D frame |
| 66 | 15 | 5 | 127 | 7·35 | 45 | 640 | 21 | 10· | 11. | Truss |
| 72 | 12 | 4 | 75 | 4·28 | 13 | 244 | 4 | 3· | 3· | Grillage |
| 87 | 7 | 12 | 227 | 7·15 | 64 | 2336 | 43 | 27· | 29· | Tower |
| 162 | 16 | 8 | 510 | 4·50 | 157 | 2806 | 33 | 17· | 19· | Plate w/hole |
| 193 | 17 | 29 | 1650 | 9·38 | 63 | 7953 | 62 | 41· | 44· | Knee prosthesis |
| 198 | 28 | 11 | 597 | 3·55 | 37 | 5817 | 36 | 29· | 31· | Reinforced mast |
| 209 | 10 | 16 | 767 | 3·99 | 185 | 9712 | 71 | 46· | 50· | Console |
| 221 | 30 | 11 | 704 | 3·34 | 188 | 10131 | 77 | 46· | 50· | Hull-tank region |
| 234 | 25 | 9 | 300 | 1·52 | 49 | 1999 | 18 | 9· | 9· | Tower w/platform |
| 245 | 29 | 12 | 608 | 2·43 | 116 | 4179 | 30 | 17· | 18· | Carriage |
| 307 | 20 | 8 | 1108 | 2·68 | 64 | 8132 | 35 | 26· | 27· | Power supply housing |
| 310 | 11 | 10 | 1069 | 2·55 | 29 | 3006 | 16 | 10· | 10· | Hull w/refinement |
| 346 | 5 | 18 | 1440 | 2·69 | 319 | 9054 | 44 | 26· | 27· | Deckhouse |
| 361 | 13 | 8 | 1296 | 2·27 | 51 | 5445 | 25 | 15· | 15· | Cylinder w/cap |
| 419 | 24 | 12 | 1572 | 2·03 | 357 | 40145 | 172 | 96· | 107· | Barge |
| 492 | 26 | 10 | 1332 | 1·30 | 436 | 34282 | 149 | 70· | 80· | Piston shaft |
| 503 | 2 | 24 | 2762 | 2·38 | 453 | 36417 | 126 | 72· | 79· | Baseplate |
| 512 | 8 | 14 | 1495 | 1·34 | 74 | 6530 | 28 | 13· | 15· | Submarine |
| 592 | 19 | 14 | 2256 | 1·46 | 260 | 29397 | 88 | 50· | 55· | CVA bent |
| 607 | 23 | 13 | 2262 | 1·39 | 148 | 30615 | 86 | 50· | 55· | Wankel rotor |
| 758 | 21 | 10 | 2618 | 1·04 | 201 | 23871 | 61 | 31· | 38· | |
| 869 | 14 | 13 | 3208 | 0·96 | 587 | 20397 | 41 | 23· | 25· | |
| 878 | 27 | 9 | 3285 | 0·97 | 520 | 26933 | 40 | 31· | 32· | Plate w/insert |
| 918 | 4 | 12 | 3253 | 0·88 | 840 | 109273 | 194 | 119· | 131· | Beam w/cutouts |
| 992 | 18 | 17 | 7876 | 1·70 | 514 | 263298 | 514 | 265· | 302· | Mirror |
| 1005 | 3 | 26 | 3808 | 0·85 | 852 | 122075 | 228 | 121· | 138· | Baseplate |
| 1007 | 22 | 9 | 3784 | 0·85 | 987 | 26793 | 32 | 27· | 27· | |
| 1242 | 9 | 11 | 4592 | 0·68 | 937 | 111430 | 193 | 90· | 105· | Sea chest |
| 2680 | 1 | 18 | 11173 | 0·35 | 2500 | 590543 | 362 | 220· | 234· | Destroyer |

computer tape on which the connectivity information for each network is stored. (A copy of this tape is available to interested researchers.) All matrix statistics (bandwidth, profile, etc.) given in Table I refer to the network before resequencing, reflecting the nodal sequences generated by the finite element practitioners. In many cases, these practitioners anticipated using an automatic resequencing computer program and thus made no attempt to start with a good sequence.

Figure 2 shows plots of all 30 test problems, arranged in order of ascending $N$, the number of grid points (nodes). Ten of the 30 problems are two-dimensional structures; the rest are three-dimensional.

## THE RESEQUENCING ALGORITHMS TESTED

The three algorithms tested are Cuthill-McKee (CM),[17] Gibbs–Poole–Stockmeyer (GPS),[26] and Levy.[32] In this section each algorithm is described briefly, with details concerning the specific implementation used. It is recognized that one cannot really evaluate algorithms *per se*, but only specific implementations of algorithms.

### *Cuthill McKee (CM)*[17]

The original version of CM operated generally according to the following procedure: Among the nodes of low degree, select as potential starting nodes those which can root a graph of minimal width. (The term 'starting node' refers to a node which is assigned the label 1 in the new sequence.) For each potential starting node, assign the labels 2–$N$ by numbering those adjacent to new label $I$ (and unnumbered) in order of increasing degree, starting with $I = 1$ and continuing with increasing $I$ until all nodes are sequenced. Of the sequences attempted, select the one having the smallest bandwidth.

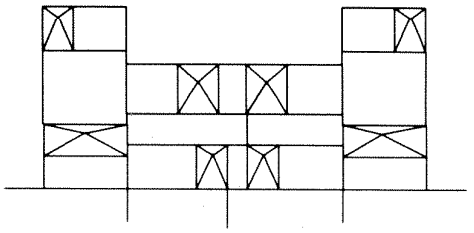
The implementation of CM used in these tests is that appearing in the BANDIT computer program, version 8,[55,63,64] which contains a version of CM differing from the original algorithm in two ways: First, the new sequence obtained is reversed (by setting $I$ to $N+1-I$ for each $I$), since it was observed by George[19] and proved by Liu and Sherman[57] that such a reversal, which preserves matrix bandwidth, will often reduce the matrix profile and never increase it. Second, of all sequences attempted, the one with the smallest rms wavefront is the one selected. (The matrix profile is also computed for these tests using the same sequence.) Except for these two changes, the CM computer code is that originally written by Cuthill and McKee.

The data structure originally used by CM required about $(M+8)N$ words of core storage for the problem-dependent arrays, where $N$ is the number of grid points and $M$ is the maximum nodal degree. In the BANDIT implementation of CM, word packing is used to reduce the storage requirements to $(M/L+8)N$, where $L$, the packing density, is an integer (between 2 and 6, inclusive) which depends on the problem size and the computer being used. On a CDC 6400, for example, the CP time penalty for packing is about 80 $\mu$sec per pack or unpack.

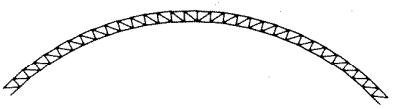
### *Gibbs–Poole–Stockmeyer (GPS)*[26]

The GPS algorithm differs from CM primarily in the selection of starting nodes. In GPS, only one starting node is selected, and it is an endpoint of a pseudo-diameter of the graph associated with the matrix. Thus, the structure need be numbered only once, using a procedure which is similar to the CM numbering algorithm.

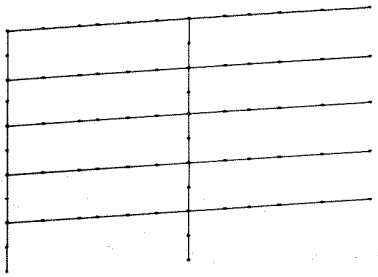
The storage requirements of GPS are identical to those of CM, including the use of integer packing in the BANDIT (version 8) implementation, which is the form of GPS used for the testing. The GPS computer code in BANDIT was written by the developers.[15]



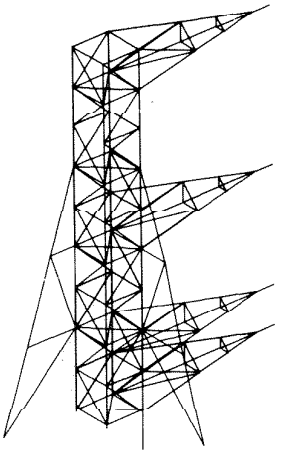
N = 59



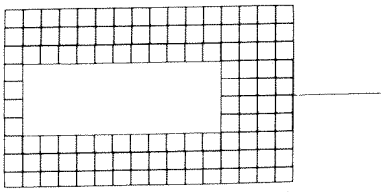
N = 66



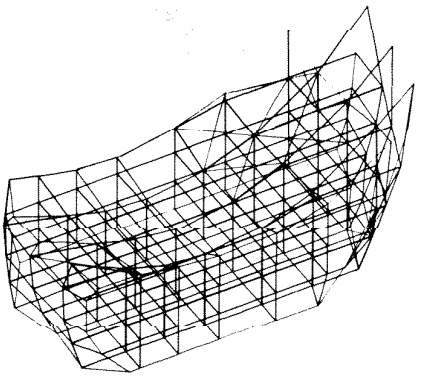
N = 72



N = 87

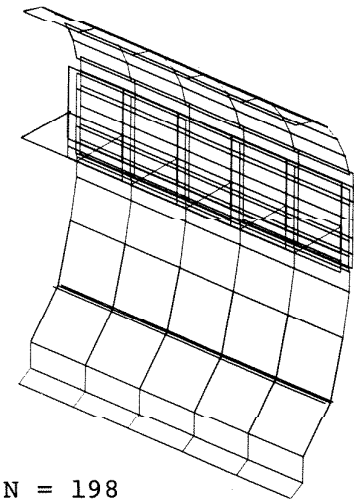


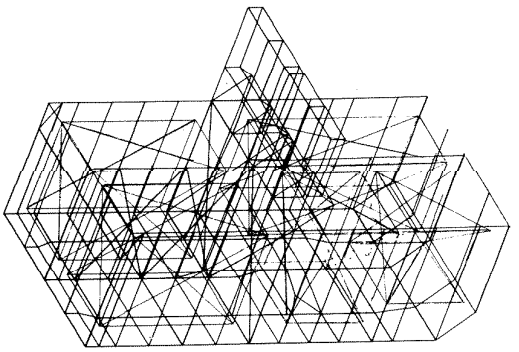
N = 162

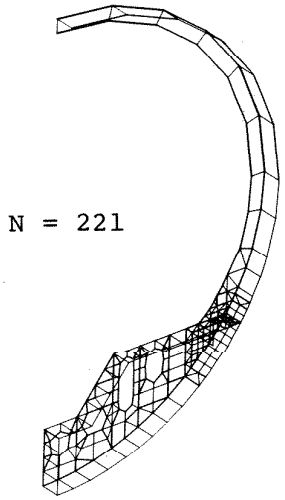


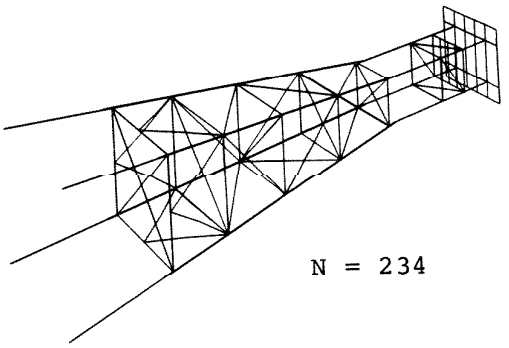
N = 193

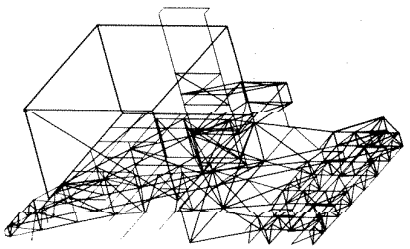
Figure 2. Plots of the test problems



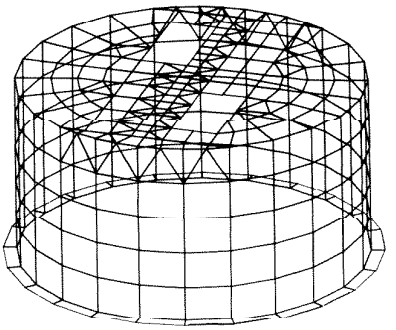
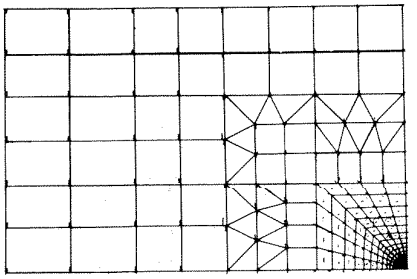
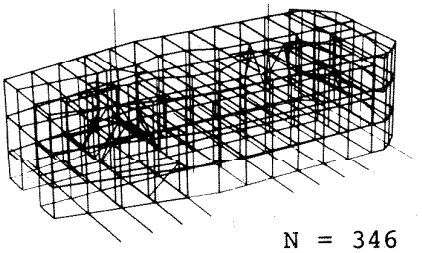



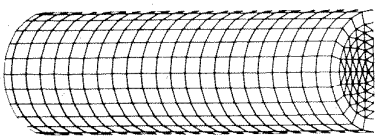

Figure 2 continued



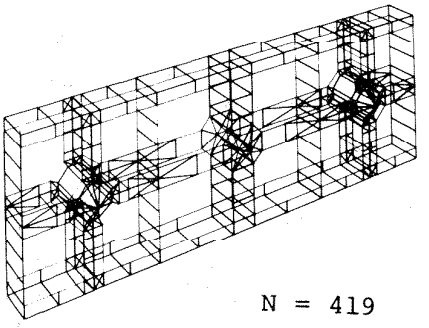
N = 310



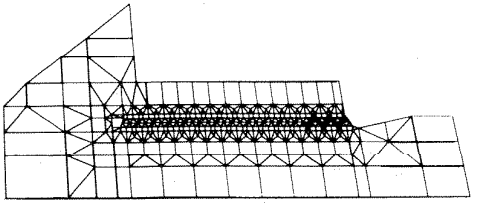
N = 346



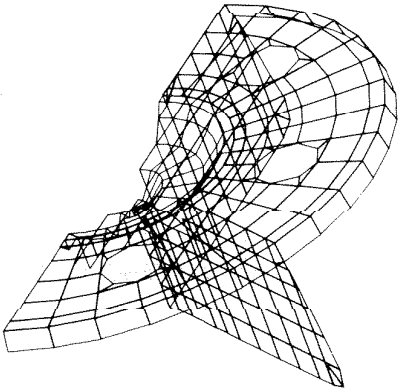
N = 361



N = 419

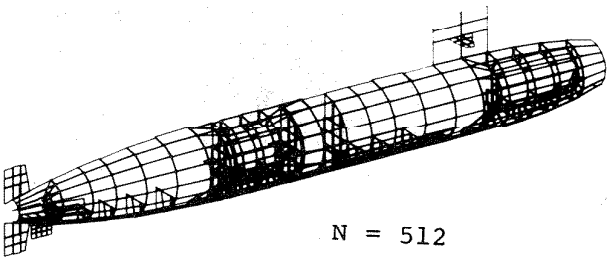


N = 492

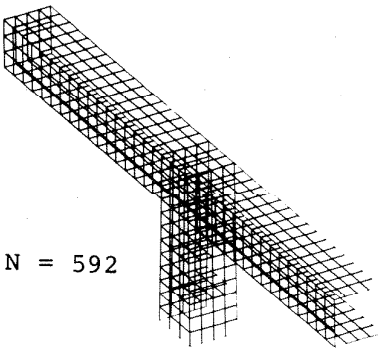


N = 503

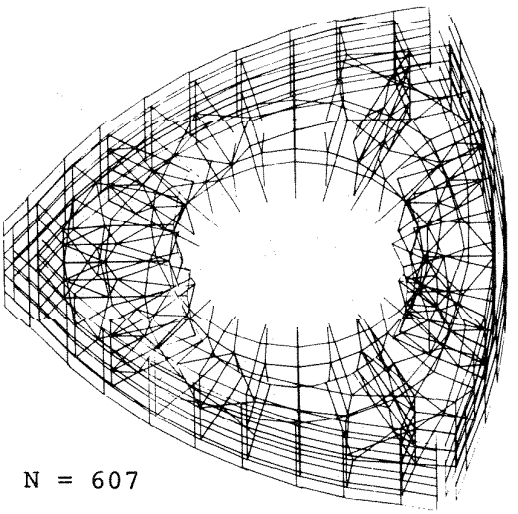
Figure 2 continued



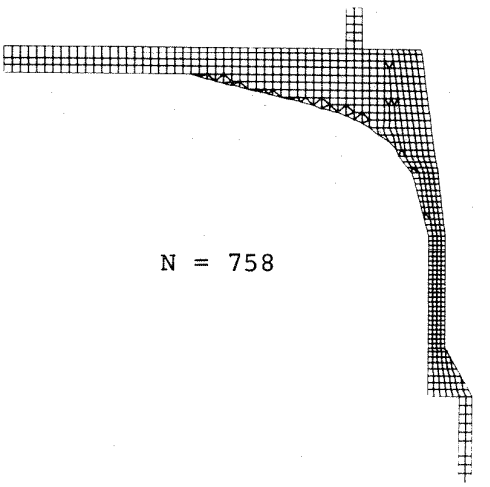
N = 512



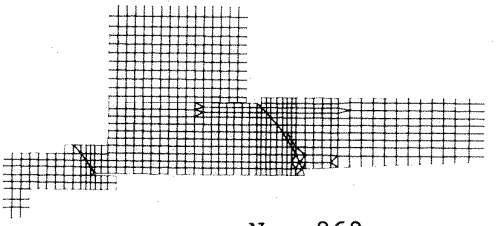
N = 592



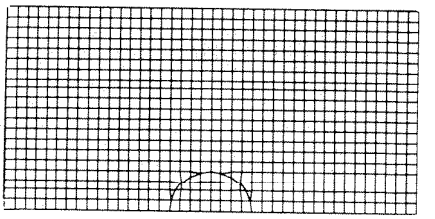
N = 607



N = 758

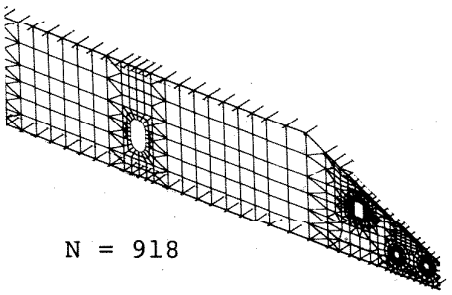


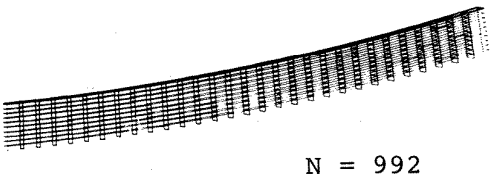
N = 869

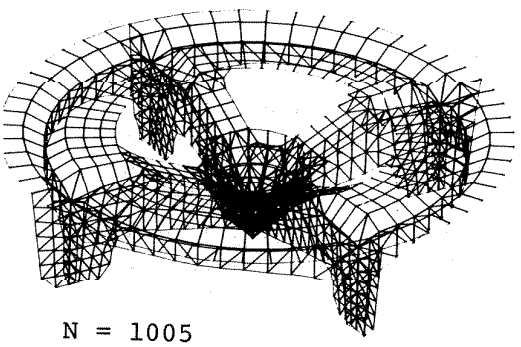


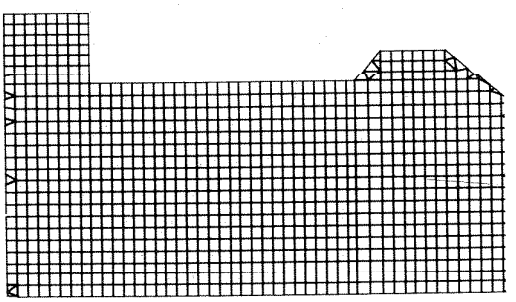
N = 878

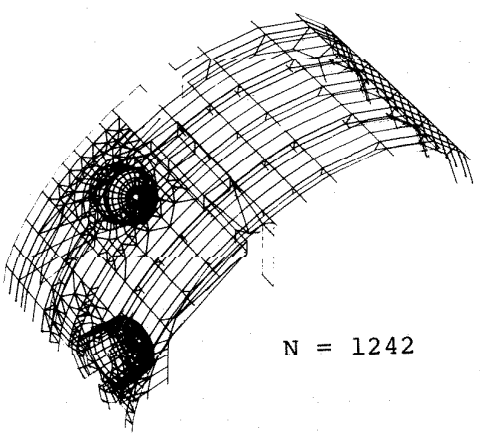
Figure 2 continued



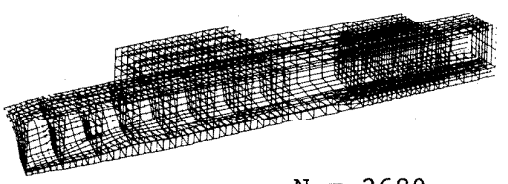




Figure 2 continued

*Levy*[32]

Unlike CM and GPS, which were developed to reduce matrix bandwidth and profile, the Levy algorithm was designed specifically to reduce the maximum matrix wavefront, $W_{max}$. The algorithm operates generally according to the following recursive procedure: Given the first $I$ nodes of a new sequence, the node selected as $I+1$ is the one for which the increase in the row wavefront between rows $I$ and $I+1$ will be minimum. Levy calls this a 'minimum growth' method.

This procedure is followed for one or more trial starting nodes, and the sequence yielding the smallest wavefront $W_{max}$ is selected. The first sequence attempted uses as the starting node either a user-selected node or a node of minimum degree. The latter option was chosen for these tests since it was felt that, for a production mode program, the user ought to be relieved of the burden of specifying a starting node. The second and succeeding sequences attempted by the Levy algorithm select starting nodes randomly. The number of new sequences to be attempted must be specified by the user. After some preliminary experimentation to estimate the speed of the algorithm, it was decided to request ten sequencing attempts for each test problem. Clearly a different number would yield different results.

The implementation used for the tests was that obtained by the author from Levy in 1973, the only change being that the sequence selected as best is the one yielding the smallest rms wavefront $W_{rms}$ or profile $P$. Since the Levy algorithm aborts any resequencing attempt in progress once it determines that it cannot reduce the previous best $W_{max}$, the sequence finally selected will be the one among those carried to completion yielding the smallest $W_{rms}$ or $P$.

The Levy data structure requires $6N + 10E$ words of core storage for the problem-dependent arrays, where $N$ is the number of grid points and $E$ is the number of unique edges. The code was not rewritten to use word packing for the tests.

## TEST RESULTS AND DISCUSSION

The nodes for the 30 test problems were resequenced using the three algorithms described in the preceding section (CM, GPS, and Levy), the objectives being to reduce rms wavefront and profile. All computer runs were made on a CDC 6400 computer under the NOS/BE operating system. The source code was compiled using the FTN compiler, OPT = 1. For reference purposes, a CDC 6400 is about one-third as fast as a CDC 6600.

The results of the tests appear in Tables II and III. In addition to the rms wavefront obtained by each algorithm, Table II also lists, for each algorithm, the CP time expended and the storage requirements for the problem-dependent arrays. In the case of CM and GPS, which use word packing, the worst-case of half-word packing is assumed. The CP times listed do not include basic setup of the arrays.

Table III lists, for each algorithm, the profile results, presented in terms of average wavefront (which equals $P/N$) rather than profile $P$ to facilitate comparisons with rms wavefront results. CP times and storage requirements are unchanged from Table II.

The first conclusion to be drawn from Tables II and III is that, for most problems, all three algorithms achieve about the same reduction in rms wavefront and profile. This is, perhaps, somewhat unexpected since CM and GPS were designed primarily to reduce matrix bandwidth, whereas the Levy scheme was designed to reduce matrix wavefront. Of the 30 problems, Levy achieved the best reduction in rms wavefront 13 times, GPS 11 times, and CM 5 times. However, on four occasions ($N = 503$, $N = 607$, $N = 878$, and $N = 918$) Levy did significantly worse than the best achieved; on three occasions ($N = 209$, $N = 245$, and $N = 1242$) GPS did

Table II. Rms wavefront test results

| No. of grid points ($N$) | Rms wavefront | | | | CP time (sec.) | | | Storage (words) | |
|---|---|---|---|---|---|---|---|---|---|
| | Before | After CM | After GPS | After Levy | CM | GPS | Levy | CM & GPS $(M/2+8)N$ | Levy $6N+10E$ |
| 59 | 8.2 | 5·5* | 6·0 | 6.1 | 0·5 | 0·2 | 2·7 | 620 | 1394 |
| 66 | 11·0 | 3·2 | 2·9* | 3·0 | 0·6 | 0·2 | 1·5 | 693 | 1666 |
| 72 | 3·5 | NI | NI | NI | 0·3 | 0·2 | 1·2 | 720 | 1182 |
| 87 | 29·4 | 8·3* | 8·9 | 8·9 | 1·5 | 0·4 | 6·1 | 1218 | 2792 |
| 162 | 19·0 | 10·3 | 10·6 | 8·6* | 2·8 | 0·8 | 13·4 | 1944 | 6072 |
| 193 | 43·8 | 26·0 | 27·1 | 24·7* | 11·9 | 6·6 | 36·2 | 4343 | 17658 |
| 198 | 30·9 | 7·3 | 7·1* | 7·2 | 2·7 | 1·6 | 23·1 | 2673 | 7158 |
| 209 | 50·3 | 19·9 | 24·5 | 18·4* | 6·0 | 1·3 | 37·6 | 3344 | 8924 |
| 221 | 50·4 | 10·2* | 10·4 | 13·3 | 5·7 | 1·5 | 38·0 | 2984 | 8366 |
| 234 | 9·4 | 7·3 | 7·1 | 5·1* | 1·5 | 0·9 | 14·9 | 2925 | 4404 |
| 245 | 18·5 | 17·5 | 18·4 | 13·5* | 4·5 | 1·4 | 26·4 | 3430 | 7550 |
| 307 | 27·4 | NI | NI | 25·7* | 10·7 | 1·9 | 73·7 | 3684 | 12922 |
| 310 | 9·9 | NI | NI | 9·7* | 16·2 | 2·2 | 32·0 | 4030 | 12550 |
| 346 | 27·1 | 22·8 | 24·3 | 21·8* | 18·0 | 2·7 | 61·5 | 5882 | 16476 |
| 361 | 15·4 | 14·3 | 14·2* | 14·3 | 11·3 | 1·8 | 38·7 | 4332 | 15126 |
| 419 | 107·1 | 22·5 | 22·2 | 19·8* | 19·5 | 2·5 | 155·1 | 5866 | 18234 |
| 492 | 79·5 | 14·5 | 13·0 | 10·6* | 13·3 | 2·9 | 145·7 | 6396 | 16272 |
| 503 | 78·6 | 33·1* | 34·6 | 41·9 | 43·3 | 4·2 | 294·3 | 10060 | 30638 |
| 512 | 14·5 | 12·7 | 12·5 | 12·4* | 10·1 | 4·5 | 161·0 | 7680 | 18022 |
| 592 | 55·2 | 25·6 | 20·5* | 21·3 | 56·3 | 5·2 | 133·1 | 8880 | 26112 |
| 607 | 55·4 | 29·2 | 28·9* | 38·0 | 37·6 | 4·0 | 362·5 | 8802 | 26262 |
| 758 | 37·9 | 15·9 | 12·1* | 15·2 | 93·4 | 6·2 | 306·7 | 9854 | 30728 |
| 869 | 25·0 | 20·4 | 20·7 | 19·8* | 132·2 | 10·4 | 450·2 | 12601 | 37294 |
| 878 | 31·9 | 23·7 | 22·9* | NI | 46·0 | 12·2 | 311·2 | 10975 | 38118 |
| 918 | 131·1 | 25·7 | 24·3* | 51·1 | 95·2 | 9·7 | 745·7 | 12852 | 37838 |
| 992 | 302·0 | 35·9 | 34·7* | 38·8 | 141·2 | 34·8 | 801·8 | 16368 | 84712 |
| 1005 | 137·7 | 43·5* | 49·3 | 44·5 | 252·6 | 7·0 | 1010·0 | 21105 | 44110 |
| 1007 | 26·9 | 24·5 | 22·9* | NI | 42·6 | 14·6 | 300·3 | 12588 | 43882 |
| 1242 | 105·2 | 42·9 | 48·6 | 38·7* | 124·2 | 16·9 | 1270·9 | 16767 | 53372 |
| 2680 | 234·4 | 40·4 | 39·9* | # | 342·3 | 23·5 | # | 45560 | 127810 |

* = Greatest reduction, NI = No improvement.
# = Not run.

significantly worse; and on two occasions ($N = 245$ and $N = 592$) CM did significantly worse.

The second, and perhaps most striking, conclusion to be drawn from Table II is that GPS is exceptionally fast. In all cases, CM is second fastest, the Levy algorithm third fastest. The user, of course, has some control over the running time of the Levy program (but not of CM and GPS) through his specification of the number of resequencing attempts.

The third conclusion to be drawn from Table II is that the Levy algorithm, as is, requires considerably more array storage than either CM or GPS, which use the same data structure. In fact, for the Levy program, one problem ($N = 2680$) was too big for a CDC 6400 and could not be run. Clearly, the program could be rewritten to use word packing (as CM and GPS do), but this may be a non-trivial task, since the programmer has to decide which arrays to pack to yield the best compromise between storage and CP time. (Word packing, of course, saves core at the expense of CP time.)

Table III. Profile test results

| No. of grid points ($N$) | Average wavefront ($P/N$) | | | |
|---|---|---|---|---|
| | Before | After CM | After GPS | After Levy |
| 59 | 7·9 | 5·3* | 5·8 | 5·9 |
| 66 | 9·7 | 3·2 | 2·9* | 2·9 |
| 72 | 3·4 | NI | NI | NI |
| 87 | 26·9 | 7·9* | 8·4 | 8·5 |
| 162 | 17·3 | 9·9 | 10·3 | 8·5* |
| 193 | 41·2 | 25·1 | 26·0 | 23·8* |
| 198 | 29·4 | 6·9 | 6·7* | 6·9 |
| 209 | 46·5 | 18·9 | 22·7 | 17·5* |
| 221 | 45·8 | 9·8* | 9·9 | 12·6 |
| 234 | 8·5 | 6·6 | 6·4 | 4·8* |
| 245 | 17·1 | 16·4 | NI | 12·8* |
| 307 | 26·5 | NI | NI | 24·9* |
| 310 | 9·7 | NI | NI | 9·5* |
| 346 | 26·2 | 21·7 | 23·1 | 20·7* |
| 361 | 15·1 | 14·1 | 14·0* | 14·2 |
| 419 | 95·8 | 21·6 | 21·4 | 19·0* |
| 492 | 69·7 | 13·6 | 12·2 | 10·0* |
| 503 | 72·4 | 31·7* | 32·0 | 40·0 |
| 512 | 12·8 | 10·4 | 10·1* | 10·6 |
| 592 | 49·7 | 24·6 | 19·1* | 20·4 |
| 607 | 50·4 | 25·9 | 25·8* | 32·8 |
| 758 | 31·5 | 15·0 | 10·8* | 14·1 |
| 869 | 23·5 | 18·6* | 18·8 | 19·0 |
| 878 | 30·7 | 23·4 | 22·7* | NI |
| 918 | 119·0 | 24·4 | 23·2* | 47·2 |
| 992 | 265·4 | 35·3 | 34·3* | 37·7 |
| 1005 | 121·5 | 40·9* | 42·9 | 43·1 |
| 1007 | 26·6 | 24·0 | 22·6* | NI |
| 1242 | 89·7 | 41·4 | 44·9 | 37·1* |
| 2680 | 220·4 | 39·3 | 38·9* | # |

*=Greatest reduction.
# = Not run.
NI = No improvement.

Tables II and III indicate that Levy's wavefront reduction performance was generally best for the smaller problems and GPS's was generally best for larger problems. This is probably due to the author's choice of ten sequencing attempts for the Levy algorithm. As the problems get larger, the probability of Levy's selecting a good starting node at random goes down. One can infer that the algorithm's performance would improve if the program were allowed to run longer. However, whether the expenditure of more computer time is justified would be a matter for each user to decide. One issue that enters into such a decision is the number of times a given matrix problem is to be solved. If a given problem is to be solved many times (as, for example, in non-linear analysis), or if many right-hand sides are involved (as, for example, in time-dependent problems), the time spent in sequencing becomes less important.

One might also infer that the performance of the Levy algorithm would improve if trial starting nodes were selected using a strategy such as that used in CM or GPS, rather than at

random.† While this may be true sometimes, it was not true for the test problem on which Levy performed the worst ($N = 918$), because for this problem the first trial starting node selected by Levy (which uses a node of minimum degree for the first attempt) was the same starting node selected by GPS. This same problem ($N = 918$) was also run by Gibbs with his profile algorithm[25] (which is a hybrid of GPS and King,[30] the latter being similar to Levy[32]) with good results. This would indicate that Gibb's modification to the King numbering approach (given a starting node) has a significant effect for some problems.

Overall, GPS's combination of speed and consistency probably rate it the best algorithm of the three for rms wavefront and profile reduction. Previous testing[55,56] has already shown it to be an excellent algorithm for matrix bandwidth reduction, for which it was designed.

Finally, the three algorithms tested were selected because of their heavy use among NASTRAN users. However, it would be interesting to see how other strategies, including Gibbs–King[25] and Snay,[42] would perform on the same data. Both give good results for profile reduction and hence, Tables II and III indicate, would probably also do well in rms wavefront reduction.

## REFERENCES


1. R. H. Gallagher, *Finite Element Analysis: Fundamentals*, Prentice-Hall, Englewood Cliffs, N.J., 1975.
2. O. C. Zienkiewicz, *The Finite Element Method in Engineering Science*, McGraw-Hill, London, 1971.
3. G. Akhras and G. Dhatt, 'An automatic relabelling scheme for minimizing a matrix or network bandwidth', *Int. J. num. Meth. Engng*, **10**, 787–797 (1976).
4. J. E. Akin and R. M. Pardue, 'Element resequencing for frontal solutions', *The Mathematics of Finite Elements and Applications II* (Ed. J. R. Whiteman), Academic Press, New York, 1975, pp. 535–541.
5. F. A. Akyuz and S. Utku, 'An automatic relabelling scheme for bandwidth minimization of stiffness matrices', *AIAA J.* **6**, 728–730 (1968).
6. G. G. Alway and D. W. Martin, 'An algorithm for reducing the bandwidth of a matrix of symmetric configuration', *Comput. J.* **8**, 264–272 (1965).
7. I. Arany, W. F. Smyth and L. Szoda, 'An improved method for reducing the bandwidth of sparse symmetric matrices', *Proc. IFIP Conf.*, North-Holland, Amsterdam, 1246–1250 (1971).
8. R. Baumann, 'Some new aspects of load flow calculation: I—Impedance matrix generation controlled by network topology', *IEEE Trans. Power App. and Syst.* **PAS–85**, 1164–1176 (1966).
9. R. D. Berry, 'An optimal ordering of electronic circuit equations for a sparse matrix solution', *IEEE Trans. Circuit Theory*, **CT 18**, 40–50 (1971).
10. A. Bykat, "A note on an element ordering scheme', *Int. J. num. Meth. Engng*, **11**, 194–198 (1977).
11. J. Carpentier, 'Ordered eliminations', *Proc. Power System Computation Conf.*, London (1963).
12. K. Y. Cheng, 'Minimizing the bandwidth of sparse symmetric matrices', *Computing*, **11**, 103–110 (1973).
13. R. J. Collins, 'Bandwidth reduction by automatic renumbering', *Int. J. num. Meth. Engng*, **6**, 345–356 (1973).
14. W. L. Cook, 'Automated input data preparation for NASTRAN', Goddard Space Flight Center Report X-321-69-237, Greenbelt, Md. (1969).
15. H. L. Crane, Jr., N. E. Gibbs, W. G. Poole, Jr. and P. K. Stockmeyer, 'Algorithm 508: Matrix bandwidth and profile reduction', *ACM Trans. Math. Software*, **2**, 375–377 (1976).
16. J. G. Crose, 'Bandwidth minimization of stiffness matrices', *J. Engng Mech. Div. ASCE*, **97**, 163–167 (1971).
17. E. Cuthill and J. M. McKee, 'Reducing the bandwidth of sparse symmetric matrices', *Proc. 24th Nat. Conf.*, Assn. for Computing Machinery, ACM Pub. P69, New York, 157–172 (1969).
18. H. Edelmann, 'Ordered triangular factorization of matrices', *Proc. Power System Computation Conf.*, Stockholm (1966).
19. A. George, 'Computer implementation of the finite element method', *Ph. D. Dissertation*, Tech. Rep. STAN-CS-71-208, Computer Sci. Dept., Stanford Univ., Stanford, CA, 1971.
20. A. George, 'An efficient band-oriented scheme for solving $n$ by $n$ grid problems', *AFIPS Conf. Proc.* **41**, 1972 Fall Joint Computer Conf., AFIPS Press, Montvale, N. J., pp. 1317-1320.
21. A. George, 'Nested dissection of a regular finite element mesh', *SIAM J. Numer. Anal.* **10**, 345–363 (1973).
22. A. George, 'Numerical experiments using dissection methods to solve $n$ by $n$ grid problems', *SIAM J. Numer. Anal.* **14**, 161–179 (1977).


† In fact, subsequent to the writing of this paper, a new version of WAVEFRONT (called WAVELEVEL) became available which uses the GPS scheme for starting node selection.[65]

23. A. George and J. W. H. Liu, 'An automatic partitioning and solution scheme for solving large sparse positive definite systems of linear algebraic equations', CS–75–17, Dept. of Computer Sci., Univ. of Waterloo, Waterloo, Ontario (1975).
24. A. George and J. W. H. Liu, 'Algorithms for matrix partitioning and the numerical solution of finite element systems', *SIAM J. Numer. Anal.* **15**, 297–327 (1978).
25. N. E. Gibbs, 'Algorithm 509: A hybrid profile reduction algorithm', *ACM Trans. Math. Software*, **2**, 378–387 (1976).
26. N. E. Gibbs, W. G. Poole, Jr. and P. K. Stockmeyer, 'An algorithm for reducing the bandwidth and profile of a sparse matrix', *SIAM J. Numer. Anal.* **13**, 236–250 (1976).
27. H. R. Grooms, 'Algorithm for matrix bandwidth reduction', *J. Struct. Div. ASCE*, **98**, 203–214 (1972).
28. J. R. Hanson, 'A new procedure for topologically controlled eliminations', *Proc. Fourth Power System Computation Conf.*, Grenoble (1972).
29. J. G. Jewell, 'Ordering systems of equations', *J. SIAM*, **9**, 55–71 (1961).
30. I. P. King, 'An automatic reordering scheme for simultaneous equations derived from network systems', *Int. J. num. Meth. Engng*, **2**, 523–533 (1970).
31. P. F. Lemieux and P. E. Brunelle, 'Relabeling algorithm to obtain a band-matrix from a sparse one', Tech. Rep. PFL–3–72, Dept. Civil Engng, Univ. of Sherbrooke, Quebec, Canada (1972).
32. R. Levy, 'Resequencing of the structural stiffness matrix to improve computational efficiency', *Jet Propulsion Laboratory Quart. Tech. Review*, **1**, 61–70 (1971).
33. R. Levy and S. Wall, 'Savings in NASTRAN decomposition time by sequencing to reduce active columns', *NASTRAN: Users' Experiences*, NASA TM X–2378, 627–631 (1971).
34. R. Levy, 'Structural stiffness matrix wavefront resequencing program (WAVEFRONT)', JPL Tech. Report 32–1526, **XIV**, 50–55 (1972).
35. C. C. Lin, 'Bandwidth reduction for simultaneous equations in matrix analysis', Babcock and Wilcox Company Report TP-535, Lynchburg, Virginia (1974).
36. E. C. Ogbuobiri, W. F. Tinney and J. W. Walker, 'Sparsity-directed decomposition for Gaussian elimination on matrices', *IEEE Trans. Power App. and Syst.* **PAS–89**, 141–150 (1970).
37. E. Roberts, Jr., 'Relabeling of finite-element meshes using a random process', NASA TM X–2660, Lewis Research Center, Cleveland, Ohio (1972).
38. J. S. N. Rodrigues, 'Node numbering optimization in structural analysis', *J. Struct. Div. ASCE*, **101**, 361–376 (1975).
39. R. Rosen, 'Matrix bandwidth minimization', *Proc. 23rd Nat. Conf.*, Assn. for Computing Machinery, Brandon/Systems Press, Princeton, N. J., 585–595 (1968).
40. N. Sato and W. F. Tinney, 'Techniques for exploiting the sparsity of the network admittance matrix', *IEEE Trans. Power App. and Syst.* **82**, 944–950 (1963).
41. M. Silverberg, 'Near-optimal ordering of electronic circuit equations', *IEEE Trans. computers*, **C–17**, 1173–1174 (1968).
42. R. A. Snay, 'Reducing the profile of sparse symmetric matrices', *Bull. Geod.* **50**, 341–352 (1976).
43. W. R. Spillers, 'On Diakoptics: Tearing an arbitrary system', *Q. Appl. Math.* **23**, 188–190 (1965).
44. W. R. Spillers, 'Techniques for analysis of large structures', *J. Struct. Div. ASCE*, **94**, 2521–2534 (1968).
45. W. R. Spillers and N. Hickerson, 'Optimal elimination for sparse symmetric systems as a graph problem', *Q. Appl. Math.* **26**, 425–432 (1968).
46. R. P. Tewarson, 'Row-column permutation of sparse matrices', *Computer J.* **10**, 300–305 (1967).
47. W. F. Tinney, 'Optimal ordering for sparsely coupled subnetworks', Bonneville Power Admin. Report, Portland, Oregon (1968).
48. W. F. Tinney and W. S. Meyer, 'Solution of large sparse systems by ordered triangular factorization', *IEEE Trans. Automatic Control*, **AC–18**, 333–346 (1973).
49. W. F. Tinney, W. L. Powell and J. W. Walker, 'Programming of sparsity-directed ordering schemes', *Proc. Power System Computation Conf.*, Cambridge, England (1975).
50. W. F. Tinney and J. W. Walker, 'Direct solutions of sparse network equations by optimally ordered triangular factorization', *Proc. IEEE*, **55**, 1801–1809 (1967).
51. P. T. R. Wang, 'Bandwidth minimization, reducibility, decomposition, and triangularization of sparse matrices', *Ph.D. Dissertation*, Dept. of Computer and Information Sciences, Ohio State Univ., Columbus, 1973.
52. J. H. Bolstad, G. K. Leaf, A. J. Lindeman and H. G. Kaper, 'An empirical investigation of reordering and data management for finite element systems of equations', ANL–8056, Argonne Nat. Lab., Argonne, Illinois, 1973.
53. E. Cuthill, 'Several strategies for reducing the bandwidth of matrices', in *Sparse Matrices and Their Applications* (Ed. D. J. Rose and W. A. Willoughby), Plenum Press, New York, 1972, 157–166.
54. H. W. Dommel and W. F. Tinney, 'Optimal Power Flow Solutions', *IEEE Trans. Power App. Syst.* **PAS–87**, 1866–1876 (1968).
55. G. C. Everstine, 'Recent improvements to BANDIT', *NASTRAN: Users' Experiences*, NASA TM X–3278, 511–521 (1975).
56. N. E. Gibbs, W. G. Poole, Jr. and P. K. Stockmeyer, 'A comparison of several bandwidth and profile reduction algorithms', *ACM Trans. Math. Software*, **2**, 322–330 (1976).

57 W.-H. Liu and A. H. Sherman, 'Comparative Analysis of the Cuthill–McKee and the reverse Cuthill-McKee ordering algorithms for sparse matrices', *SIAM J. Numer. Anal.* **13**, 198–213 (1976).
58. W. S. Meyer and V. D. Albertson, 'Loss formula computation by optimal ordering techniques which exploit the sparsity of the network admittance matrix', *1969 Midwest Power Symp.*, Univ. of Minnesota, Minneapolis (1969).
59. B. Stott and E. Hobson, 'Solution of large power-system networks by ordered elimination: A comparison of ordering schemes', *Proc. IEE*, **118** (1971).
60. W. F. Tinney and C. E. Hart, 'Power flow solution by Newton's method', *IEEE Trans. Power App. and Syst.* **PAS–86**, 1449–1460 (1967).
61. W. F. Tinney, W. L. Powell and N.M. Peterson, 'Sparsity oriented network reduction', *Proc. Power Industry Computer Applications Conf.* (Minneapolis, Minn.), 384–390 (1973).
62. 'The NASTRAN theoretical manual', NASA SP–221 (03), Washington, D.C. (1976).
63. G. C. Everstine, 'The BANDIT computer program for the reduction of matrix bandwidth for NASTRAN', Rept. 3827, Naval Ship Research and Development Center, Bethesda, Md. (1972).
64. G. C. Everstine, 'BANDIT Users' guide', TM–184–77–03, David W. Taylor Naval Ship Research and Development Center, Bethesda, MD. (1977).
65. *Computer Program Abstracts* **10**, 15 July, 1978.