# AN INTERACTIVE GRAPHICS PACKAGE FOR THE AUTOMATIC NODE RENUMBERING OF FINITE ELEMENT MATRICES*

Ronald F. Boisvert and William G. Poole, Jr.

College of William and Mary
and
Institute for Computer Applications in Science and Engineering (ICASE)

## SUMMARY

An interactive graphics software package which allows users to display the non-zero structure of large sparse symmetric matrices is described and methods used to implement it as a portable FORTRAN callable subroutine are summarized. In particular, the system permits the display of the resulting matrix after reordering the rows and columns, with the reordering scheme either defined by the user or automatically generated by the program with the aim of reducing matrix bandwidth and profile. Although the primary application of the package has been to the finite element analysis of structures, it is equally well suited to the many other areas of engineering and science which use sparse matrices.

## INTRODUCTION

Frequently, in many areas of application, we must solve the linear algebraic system of equations represented by

$$Ax = b \tag{1}$$

where A is a non-singular n x n symmetric matrix and x and b are n-vectors. Here we assume that n is moderately large (from about one hundred to several thousand) and that the matrix A is sparse; that is, the number of non-zero elements in the matrix is small compared to $n^2$.

In order to describe the non-zero structure of sparse matrices the concepts of bandwidth and profile are helpful. The bandwidth of a matrix A is defined as $b = \max_{a_{ij}\neq 0} |i-j|$, which is simply the radius of the smallest band about the diagonal which includes all non-zero components of the matrix. The

*This paper is a result of work performed, in part, under NASA Grant NGR 47-102-001 while in residence at ICASE, NASA Langley Research Center. The work of the second author also was partially supported by the Office of Naval Research Contract N00014-73-A-0374-0001, NR 044-459. The first author is currently at the Computer Science Department, Purdue University.

profile is defined as $p = \sum_{i=1}^{n} d_i$, where $d_i = i - \min_j \{j : a_{ij} \neq 0\}$, that is, the sum of the distances from the main diagonal of the leftmost non-zero component in each row. The profile is exactly the number of non-zeroes in the lower triangular factor of the decomposition of A whose calculation is typically the first step in solving (1) by Gaussian elimination.

The non-zero structure of large sparse matrices is often used to reduce the otherwise restrictive storage and computational requirements for solving the linear system. Storage schemes for sparse matrices abound (references 1 and 2) with the band scheme being among the simplest to use. In this case, when the bandwidth of A is small, we can eliminate most zero components by simply storing only the diagonal bands of A. Other techniques incur even larger savings by taking advantage of the profile of the matrix. The non-zero structure of the matrix also greatly influences the computation time required to solve (1). The time required to perform Gaussian elimination on a full matrix is proportional to $n^3$ while methods for band matrices typically require computation times proportional to $nb^2$. Once again, this can be reduced further by exploiting profile.

It is clear that for matrices with small bandwidth and profile considerable savings can be realized and, in fact, many codes are currently available for band or profile decomposition. Thus, the engineer wishing to solve (1) would normally pay particular attention to the non-zero structure of the matrix and, whenever possible, would seek that representation of the matrix which minimizes bandwidth and profile. Although band and profile schemes are not always the best way to treat sparse matrices, many of the matrices encountered in real applications, in particular, finite element approximations in structural engineering, are quite appropriate for the use of band or profile schemes.

The sparse matrix graphics package, by allowing the display of a representation of the non-zero structure of the matrix, gives the user a unique visual aid in the evaluation of various methods of storage and solution of the particular problem in question. In addition, by providing a matrix bandwidth and profile reducer as part of the package, the user may visually evaluate the worth of a renumbering of the rows and columns of the matrix which reduces bandwidth and profile to nearly minimal levels.

An example of the use of the graphics program is given in the following section with a description of the full capabilities and implementation of the graphics system following that. In the last two sections the algorithm used for matrix bandwidth and profile reduction is outlined and the results of the author's experience with the program are discussed.

## EXAMPLE OF USE

The following scenario describes a typical use of the system. Suppose a design engineer, seated at a graphics terminal, is using the finite element method for analysing some structure. After generating the elements, the designer numbers the nodes in some order, often an order which is easy to describe, and then generates a table defining the location of the non-zero components in the associated sparse matrix which represents properties of the structure being analysed. Because of interest in the effect of the nodal numbering on the non-zero structure of the matrix, the designer now invokes the sparse matrix graphics program via a subroutine call in the control program, causing the information at the top of Figure 1 to be displayed at the terminal. The design engineer then continues with the interaction shown in Figure 1, informing the program that the first display should be that of the input matrix shown in Figure 2.

Somewhat dissatisfied with his/her own nodal numbering, the designer might then request the program to generate a new numbering which reduces bandwidth and profile, thus producing the display of Figure 3. As a final comparison of the two numberings, the engineer may finally request the lower triangular halves of both matrices to be displayed at the same time, generating the display of Figure 4.

Not only does the design engineer get a nodal numbering which produces a small bandwidth and profile, but he also now has a visual conception of how effective the original numbering was.

## IMPLEMENTATION OF THE DISPLAY SYSTEM

The sparse matrix graphics program is a highly portable FORTRAN program, requiring minimal local computing facilities. As demonstrated in the previous section, the graphics program has the following capabilities:

- Display the original matrix.
- Generate a reordering of the rows and columns which reduces bandwidth and profile, displaying the resultant matrix.
- Display the matrix after reordering the rows and columns according to a scheme input by the user.
- Compare any two of the above, displaying them at the same time on the screen.
- Matrices of any order may be displayed regardless of screen size.
- The size of the matrix display may be varied by the user.
- Either the full matrix or the lower triangular half may be displayed.

The package is invoked by a call to the subroutine SPARSE, with the generated renumbering returned as a parameter if calculated during the interactive session. In this way the package is easily interfaced with existing FORTRAN coded interactive design systems.

The data structure chosen to represent the sparse matrix is the connection table, one frequently used in bandwidth and profile schemes (reference 3). The table has n rows and m columns, where n is the number of rows in the actual sparse matrix and m is the maximum number of off-diagonal non-zero components to be found in any row. The ith row of the table contains the column indices of all off-diagonal non-zero components in the ith row of the actual matrix. These column indices may be in any order. The values of the non-zero components themselves are never needed, only their indices. The renumbering schemes are represented as permutation vectors, whose entries show the order of the old rows and columns in the new matrix. For example, if the reordering vector is p, then the ith row in the original matrix becomes the p(i)-th row in the reordered matrix.

Due to the simplicity of the displays generated, rather minimal local graphics software capabilities are required to support the system. Indeed, any computation facility supporting interactive graphics terminals which provides FORTRAN callable routines to display a symbol and move the beam invisibly can easily implement the package on their system. These capabilities are interfaced with the sparse matrix graphics program via two user supplied routines, one which plots a row of user chosen symbols representing non-zero components at coordinates which are given, and one which simply moves the beam invisibly to a given location.

Displaying large matrices on most screens is a problem; for instance, on a terminal with a resolution of 1024 points by 780 points, a 1000 x 1000 matrix could not be displayed, even if each element were represented by a single point. To overcome this, the graphics program instead displays a related partitioned matrix; specifically, each symbol displayed represents an r x r block of elements of the original matrix in which a non-zero occurs. The blocking factor r is chosen by the program and depends on the order of the original matrix and the display size requested by the user. The resulting display thus yields a visual description of the placement of non-zeroes, even when the actual matrix could not be displayed.

In order for the program to generate the coordinates of the symbols to be displayed, the calling program provides information concerning the dimensions of the screen, the size of the square symbol which the user has chosen to represent non-zeroes, and the size of the displayed characters resulting from ordinary FORTRAN write statements. The program then supplies display coordinates to the user-supplied graphics routines in the same units as the user implicitly defines the screen dimension information.

To run the program, some 7n+2m core storage locations are required, in addition to the storage for the connection table (nm locations) and for the optional user supplied permutation (n locations). Thus the solution of large problems may be restricted by the core memory size. The largest part of these storage requirements is needed to implement the bandwidth and profile reduction subroutine, REDUCE.

## REDUCING BANDWIDTH AND PROFILE

The algorithm for bandwidth and profile reduction made available by the sparse matrix graphics program is one recently proposed by Gibbs, Poole, and Stockmeyer (reference 4). The actual FORTRAN implementation used, REDUCE, is detailed in reference 5. REDUCE has been found to be considerably faster than all other reduction codes in widespread use, generally superior for bandwidth reduction, and generally as successful as any other for profile reduction.

The algorithm can best be described in terms of the related adjacency graph, G, which has the property that there is an edge in G between vertices $v_i$ and $v_j$ if and only if $a_{ij} \neq 0$ and $i \neq j$.

Step 1: Find the endpoints of a pseudo-diameter of the graph, that is, a pair of vertices that are at nearly maximal distance apart. This is done by a finite, iterative process of determining a vertex that is a maximum distance away from a given vertex.

Step 2: Given the pseudo-diameter endpoints u and v of distance k apart, partition the set of vertices into levels $L_1,L_2,\ldots,L_k$ such that adjacent vertices in G are in the same or adjacent levels and such that $\max_i|L_i|$ is nearly minimized.

Step 3: Number the vertices of G, level by level, beginning at an endpoint of the pseudo-diameter.

A detailed description of the algorithm may be found in reference 4.

## EXPERIENCE

The sparse matrix graphics program was developed on a PRIME 300, a multiprogrammed minicomputer with 64K of core storage at ICASE, NASA Langley Research Center (LRC). The program is interfaced with the University of Michigan Graph-Pack Library and has been run using primarily a Tektronix 4014-1 graphics terminal. The program is also running on the CDC 6600 at LRC under the KRONOS time-sharing operating system, again utilizing the Michigan graphics library, using a Tektronix model 4015-1 display terminal. Earlier batch versions of the program were run on an IBM 360/50 computer with a Calcomp model 765 plotter as well as on a CDC 6400 system with both Calcomp and Varian type plotters.

The package has been tested using most of the finite element matrices used to test subroutine REDUCE in reference 4, as well as on matrices generated in finite element work at LRC. In general, response times for the generation of displays (especially on the minicomputer system) were more than adequate for an interactive design environment. The block matrix scheme for displaying otherwise unrepresentable matrices was found to be most acceptable.

Although very large sparse matrices do not in general appear as sparse under this scheme, a good representation of the distribution of non-zero elements in the matrices is nevertheless provided.

## REFERENCES

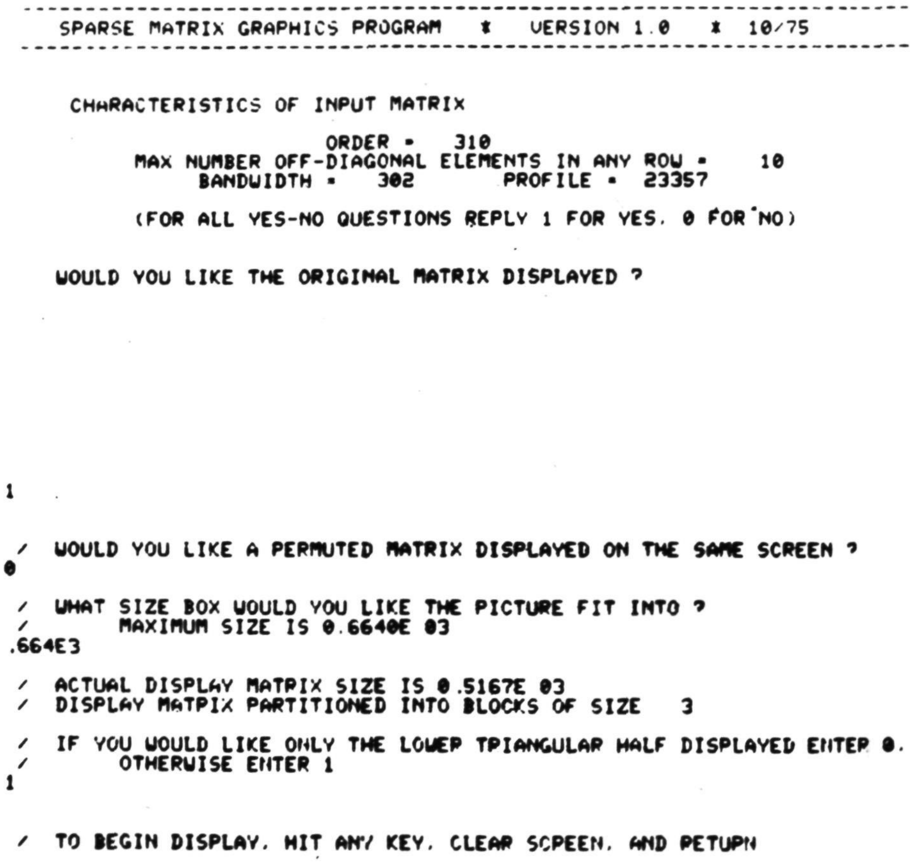
1. Rose, Donald J. and Willoughby, Ralph A. (editors): Sparse Matrices and Their Applications. Plenum Press, New York, 1972.

2. Tewarson, Reginald P.: Sparse Matrices. Academic Press, New York, 1973.

3. Everstine, G. C.: The BANDIT Computer Program for the Reduction of Matrix Bandwidth for NASTRAN. NSRDC Report 3827, 1972.

4. Gibbs, N. E., Poole, W. G., Jr. and Stockmeyer, P. K.: An Algorithm for Reducing the Bandwidth and Profile of a Sparse Matrix. ICASE Report, July 22, 1974. To appear in SIAM Journal on Numerical Analysis, volume 13, 1976.

5. Crane, H. L., Gibbs, N. E., Poole, W. G., Jr. and Stockmeyer, P. K.: Matrix Bandwidth and Profile Reduction. ICASE Report 75-9, April 14, 1975. To appear in ACM Transactions on Mathematical Software, volume 2, 1976.



```
------------------------------------------------------------------------
   SPARSE MATRIX GRAPHICS PROGRAM   *   VERSION 1.0   *  10/75
------------------------------------------------------------------------

     CHARACTERISTICS OF INPUT MATRIX

                          ORDER =   310
          MAX NUMBER OFF-DIAGONAL ELEMENTS IN ANY ROW =     10
               BANDWIDTH =   302        PROFILE =  23357

          (FOR ALL YES-NO QUESTIONS REPLY 1 FOR YES, 0 FOR NO)


    WOULD YOU LIKE THE ORIGINAL MATRIX DISPLAYED ?









1


/   WOULD YOU LIKE A PERMUTED MATRIX DISPLAYED ON THE SAME SCREEN ?
0

 /  WHAT SIZE BOX WOULD YOU LIKE THE PICTURE FIT INTO ?
 /       MAXIMUM SIZE IS 0.6640E 03
.664E3

 /  ACTUAL DISPLAY MATPIX SIZE IS 0.5167E 03
 /  DISPLAY MATPIX PARTITIONED INTO BLOCKS OF SIZE   3

 /  IF YOU WOULD LIKE ONLY THE LOWEP TPIANGULAR HALF DISPLAYED ENTEP 0.
 /       OTHERWISE ENTER 1
1


 /  TO BEGIN DISPLAY, HIT AN/ KEY, CLEAR SCPEEN, AND PETUPN
```

Figure 1.- Typical interaction with sparse matrix graphics program.

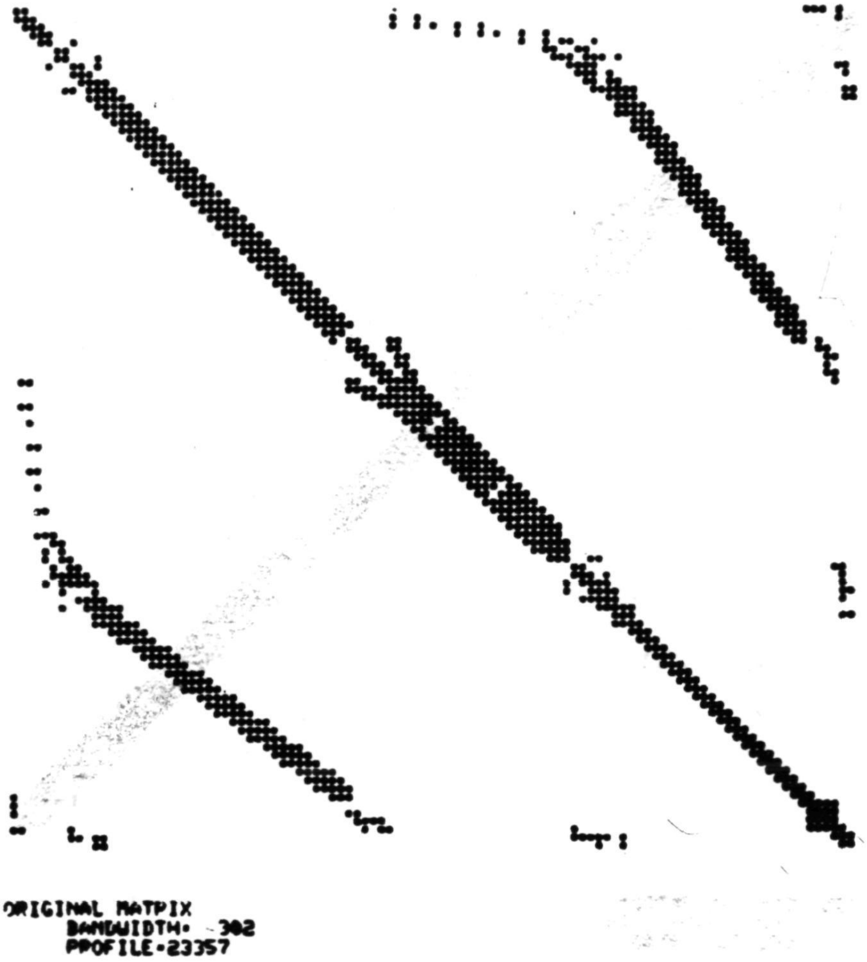
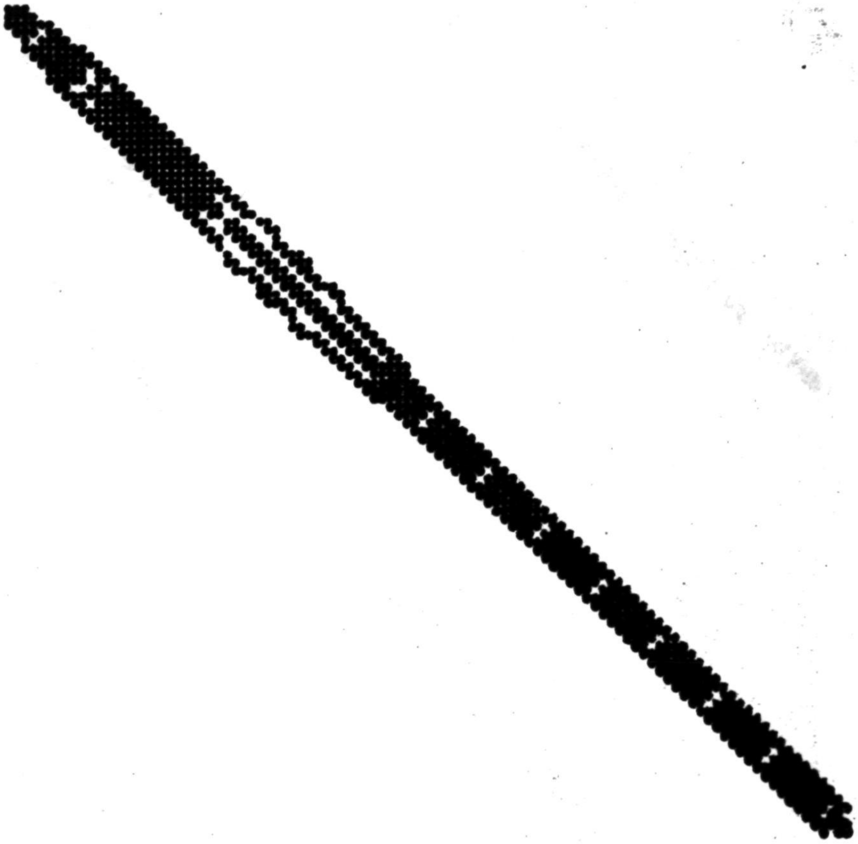


Figure 2.- Display of input matrix.



```
GPS PERMUTED MATRIX
      BANDWIDTH=    14
      PROFILE= 2786
```

Figure 3.- Display of matrix after renumbering.

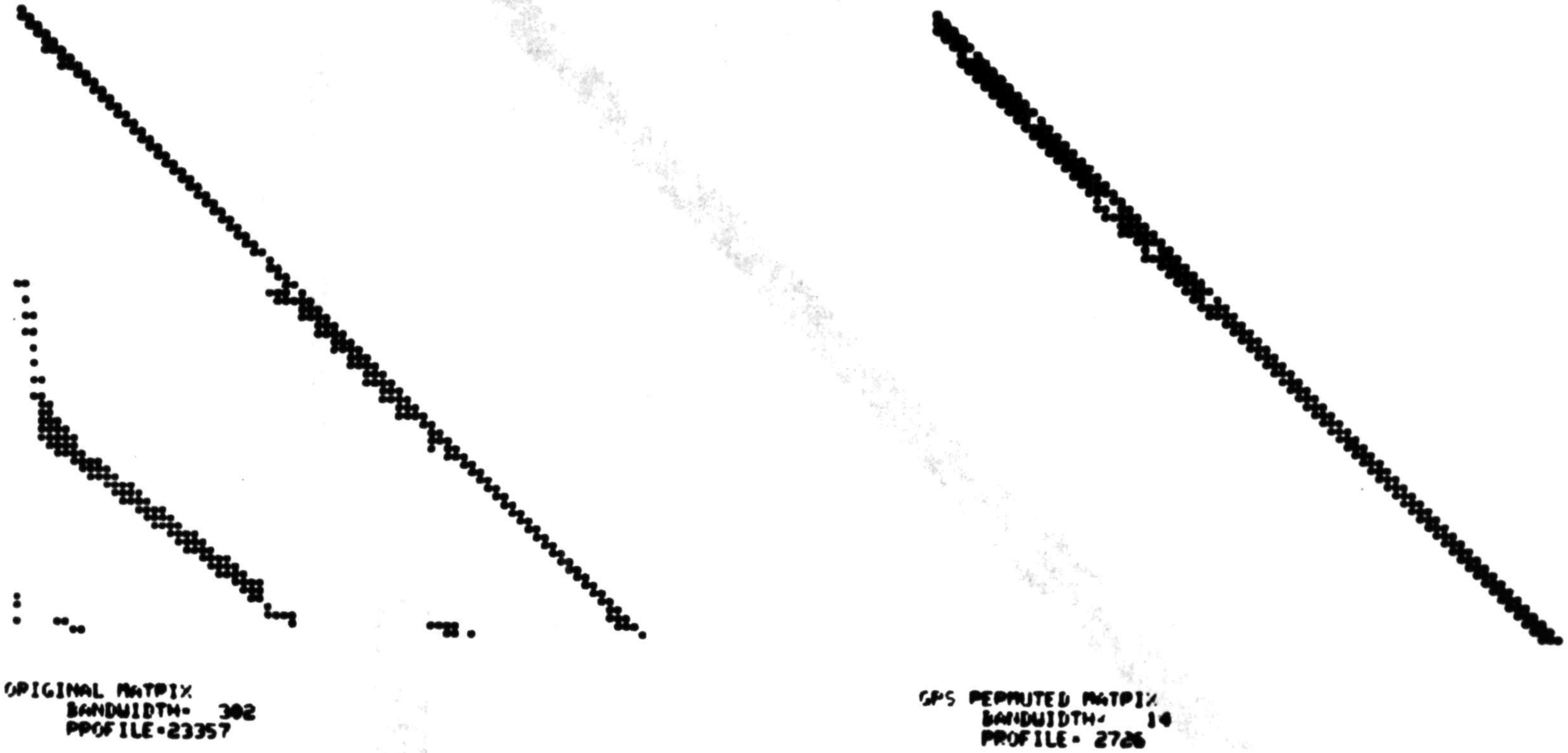


Figure 4.- Display of lower triangular halves of matrix before and after renumbering.