PCNDT - AN ELECTROMAGNETIC FINITE ELEMENT PACKAGE FOR PERSONAL COMPUTERS

Nathan Ida

Electrical Engineering Department
The University of Akron
Akron, Ohio 44325 U.S.A

ABSTRACT

An integrated finite element package for the solution of magnetostatic and eddy current problems, suitable for use on personal computers has been developed. It can handle general field problems but is particularly applicable to nondestructive evaluation modeling problems including moving probe applications. The package consists of an interactive preprocessor, routines for adaptive and non-adaptive mesh generation, bandwidth reduction algorithms, iterative and direct linear solvers and a post-processor. In addition, a simple expert system shell is used in conjunction with the user input. The main emphasis throughout the package is on useful options, simplification of user input and minimization of the time needed to correctly define a geometry for finite element analysis. Extensive graphics editing capabilities are provided.

INTRODUCTION

A finite element analysis begins with discretization of the solution space into finite elements. This stage in an analysis consists of (a) geometry definition, (b) division into finite elements, (c) display of the mesh, (d) introduction of various parameters (i.e. material properties) and (5) adjustments to the mesh. The second step is the assembly of elements and the solution of the resulting system of equation. This part of the analysis requires little user interaction but is the most time and computer resources intensive.

Compared to mainframe computers, personal computers are slow. With a typical clock frequency of 4 to 12 mHz they are considerably slower (sometimes orders of magnitude slower) than the faster mainframes. This presents a unique challenge in the choice of solution methods and underscores the need for efficient solution algorithms. The primary advantage of personal computers is their availability as dedicated systems. A finite element program, running on a PC constitutes in effect a "Finite Element Workstation". It is not difficult to justify such dedicated use of a computer because of the low prices involved both in purchase and in operation.

This work describes an integrated electromagnetic finite element package suitable for use on personal computers. It can solve for problems with thousands of unknowns, for DC or eddy current applications and offers a choice of finite elements, preprocessing capabilities, color graphics display and editing as well as interactivity. The various options are chosen by an expert system program in conjunction with the user input. It is particularly suited for moving probe problems and offers both adaptive and standard solutions.

PREPROCESSOR

The preprocessor used here offers two basic modes. In one mode, an adaptive mesh generator is used. This is suited for general field problems. In its adaptive solution mode, the program uses either first order triangular or quadrilateral elements. For triangular elements, Delaunay triangularization [1-3] in conjunction with complementary finite element methods [3] is used. For quadrilateral elements, a nonconformal method combined with linearly constrained equations [4] is used. The choice between the two methods is a user option while the program defaults to Delaunay triangularization. The use of quadrilateral elements seems to be more efficient in many problems, especially those with noncurved boundaries. Although higher order elements can be used, this is not implemented in this package for the adaptive solution. Examples to these two types of meshes are shown in Fig. 1 and 2.

The second mode uses a standard finite element process. It is used for NDT and other problems where certain layers of elements must be kept constant (i.e. probe movement region). In such cases, the size of the elements in certain areas must be kept constant to allow movement of coils or other materials. Although it is possible to combine an adaptive mesh generator with an area of fixed discretization, this approach was not pursued. It runs contrary to the idea of adaptive solution and, for moving coil problems, adaptive solution is very slow. There are two types of problems that are considered with respect to movement. In one, only sources are moving. Thus, only the right hand side of the system of equations is modified and there is no need for elimination except for the first step. In the second, materials are moving with or without sources. The matrix is therefore modified and requires elimination for each step. In either case the velocity of motion is not considered.

The primary purpose of this package is to allow the use of a variety of finite elements and options to allow for either an adaptive solution or a standard finite element solution when adaptive solution is not possible. The use of adaptive mesh generation has been described elsewhere and will not be repeated here. The following sections describe the second mode of geometry definition.

PROGRAM DESCRIPTION

When invoked, the program generates an artificial grid of points over the screen (fine, medium or coarse) that serves as a drawing pad. The cursor (or a "mouse") is moved on grid points to define a geometry. The use of specified grid points is only used to speed up the geometry definition. If required, the cursor can be moved one pixel at a time for more detailed drawings. The geometry is outlined using any number of straight lines. Later, if necessary, curvatures are introduced on any required segment. The geometry in Fig. 3 is obtained by first drawing (on a "fine" grid) the straight line segments A-B, B-C, C-D, D-A to obtain the outer boundary. Next the lines E-F, G-H, I-J, J-K, K-L, M-N, N-O, O-P, Q-R, S-T, U-V and W-X are drawn. 16 straight lines were necessary for the outline of the geometry. This is only an outline since the segments J-K, N-O, UI-WI, WI-WE, XG-XM and XM-VM will later be changed into curved segments. The user

0018-9464/88/0100-0366$01.00©1988 IEEE

can add, erase or change any lines as long as any internal area is finite and is bounded by a general quadrilateral. Any deviation from this will be flagged and the user given an opportunity to correct it by adding or deleting lines.

The shape drawn in Fig. 3 is now processed. The coordinates of all intersections are calculated in screen coordinates. Each quadrilateral is formed by associating with it the four coordinate pairs at its vertices and its connectivity to neighboring blocks. An additional node is placed at the center of each line segment to form an assembly of 8 node "super-elements" or blocks.

The program continues by asking the user for the number of elements required on each side of the quadrilaterals. Starting with an arbitrary side (internally picked by the program as the first segment processed) the color of the segment is changed and the user asked to enter the number of elements on this side or to accept a default value. Opposite sides in any block must have the same number of divisions. If this segment is A-L, sides Q-K, S-J, U-UI, W-WI and D-I are automatically assigned the same number of elements. This done, the program advances to side A-Q (or L-F). Here, the user input or the default is assigned to sides L-K, QF-F, RH-H, O-P and R-B. Similarly, when the next unassigned side is found the appropriate sides are assigned. By assigning the number of divisions on 10 segments (A-Q, Q-S, S-V, V-W, W-D, A-L, L-F, F-H, H-P and P-B) all the necessary segments have been defined.

Next, the material properties of each block are entered. This is done by the program advancing through all blocks (color of boundary changes to indicate the block) and asking the user to either accept the default material index or specify it. This allows the user to define any material properties necessary for the solution. The user may now proceed to specify boundary conditions on the shape. The program does this by changing the color of each boundary segment and asking the user for acceptance of the default or to enter a value.

Up to this point all coordinates are specified in integer screen coordinates. The user is prompted with the coordinates of the lower left and upper right corners of the screen which he can accept or scale to any value.

Curved boundaries are now treated by moving the middle point on appropriate segments to lie on the actual curve. Fig. 4 shows this for segment J-K. At this stage, the curvature on segment N-O has already been defined and the midpoint on segment J-K marked by placing the cursor on the midpoint. The cursor is now moved to the new desired location and the curved segment created when the new point is marked. In effect, all segments in the geometry are considered to be curved segments, the curvature being defined by the location of the three points of the segment. Figure 5 shows the adjusted geometry after all curved segments and all necessary midpoints have been adjusted.

The middle node on any segment also defines the way the elements are distributed over the segment. Moving the point towards one corner, will create a crowding of the elements towards that corner. The user can do this on any segment, including curved segments and internal segments but he only has to do this for those segments that need adjustment (Fig. 6).

With this, the finite element input is complete. The user may proceed to the mesh generation step or he may enter into a "graphics edit mode" in order to edit the existing file if this becomes necessary. This stage also allows him to edit existing files that have been created in the past or to combine shapes generated separately into a single geometry. Corrections and changes to the geometry drawn can be made at any stage of preprocessing.

OTHER OPTIONS

A variety of additional options are available, mostly designed for convenience. Thus, for example, parts of the geometry can be duplicated to avoid definition of symmetric parts of the object. Menus can be invoked or simple one letter commands used. The program defaults to letter commands because this is faster for the experienced user but, when a mouse is used, menus are more convenient. Other options include choices of color combinations, methods of marking points and lines, editing commands, a help command, display with or without the auxiliary grid, screen dump and plotting routines.

MESH GENERATOR

The mesh generator used in conjunction with this package can handle first and second order elements in the standard mode and first order triangular or quadrilateral elements in adaptive mode. Immediately after the preprocessor is completed, the mesh generator is invoked. No user interaction is necessary other than the specification of the type of finite elements to be used. The mesh can consist of: 3 or 6 node triangular or 4 or 8 node quadrilateral elements. Upon completion of the mesh, the output is saved in a file and the mesh displayed on the screen. Local processing of this data is allowed through simple functions like the ability to zoom on a portion of the mesh. Hard copy output is obtained either by a simple "screen dump" on any dot matrix printer or on a plotter (Fig. 6) if one is available. The output from the mesh generator consists of the following: a) Number of nodal points, number of elements, semibandwidth (suppressed for adaptive solution) and number of boundary nodes, b) coordinate array, c) element array and d) boundary conditions array.

BANDWIDTH REDUCTION

In adaptive mode, no bandwidth reduction is necessary and this step is skipped. In the standard mode, bandwidth reduction using one of two algorithms, can be invoked. The user has no control over this and the program chooses the smallest numbering sequence. If the ICCG routine is used, this step is again not necessary. Once the bandwidth has been reduced or, alternatively, no bandwidth reduction is necessary, the analysis proceeds to the assembly of the system of equations and solution.

FINITE ELEMENT PROGRAM AND SOLUTION

Because of the limited memory available, only a segment of the system of equations is assembled at a given time. The size of the segment is determined by the total memory available. Once the segment is full, the fully assembled equations in the segment are eliminated and written on disk. The rest of the equations, which are not affected by the elimination are shifted to the beginning of the segment and assembly resumes from where it stopped. After elimination, the system of equations is backsubstituted segment by segment and a solution obtained. For moving probe problems, the right hand

side is updated for each probe position and the backsubstitution process repeated without the need for elimination. The second solution method available is the Incomplete Choleski Conjugate Gradient method (ICCG). A Biconjugate method is used for eddy current problems.

POSTPROCESSING

The output from the finite element program consists of the magnetic vector potential at each node of the mesh, for each probe position. From this data other quantities need to be extracted such as impedances fields, eddy current densities, etc. The probe impedances are calculated immediately after a probe position calculation has been completed. Other quantities can be chosen through a menu on the screen. Thus, for example, the flux density at a point in the mesh is found by moving the cursor to that point or by specifying the coordinates at which the field is required. Similarly, cross-sections through the mesh can be calculated and displayed. Output consists of the screen display, a dot matrix print or any other form of output device available such as a plotter.

With this the analysis is complete and the user may chose to terminate the session or to repeat it from any stage of the analysis.

SOME RESULTS

To illustrate the use of the algorithms presented above, as well as some of their limitations some simple results obtained with PCNDT are presented. Fig. 7 shows a flux plot of a coil inside a ferrite pot-core placed over a thick conducting surface. In Fig. 7a, the pot-core is in contact with the surface while in Fig. 7b it is 4mm above the surface. This has been achieved by moving the pot-core, 0.5 mm at a time and calculating the field distribution for each position. The original purpose was to calculate the impedance of the coil and its sensitivity to the distance between coil and surface. A total of 70 positions were calculated. In this case, the conducting surface is actually "moved" by changing the material definition in the geometry and therefore, each position requires a complete elimination and backsubstitution step. The mesh for this problem consists of 2016 quadrilateral elements (1920 nodes). This has a semi-bandwidth of 49 and is typical in size to the type of meshes one would have to deal with for most moving coil problems. Solution time is of the order of 105 minutes on and IBM-AT (8 mHz). The same problem, when solved as a magnetostatic problem takes about 66 minutes. Larger problems with up to 12,000 elements were solved but these usually require an overnight run. The only limitation to the size of problem that can be solved is the amount of disk space available.

The plots in Fig. 7 were drawn on a plotter. The PC allows plotting on the screen but, because of its limited resolution, the plot quality is poorer. Fig. 8 shows a dot-matrix plot of Fig. 7a for a color graphics display (Fig. 8a) and for an enhanced graphics display (Fig. 8b). This type of display is adequate for many applications and is particularly convenient as initial display before plotting.

CONCLUSIONS

The software package described here was developed in an attempt to provide the user with a tool which he can use with confidence without being knowledgeable in either the details of field analysis or the intricacies of the finite element method. Particular consideration was given to error handling and to minimization of user input in an attempt to provide a friendly design environment. Extensive graphics and graphics editing capabilities make for a particularly simple analysis process. The use of a personal computer was found to be ideally suited for such tasks. It allows the user to define the data necessary in a simple fashion, with complete control over the outcome at any stage of the process. The system described can save the user considerable time. The solution times obtained for realistic size problems are larger than the equivalent CPU times on mainframe computers but the total clock time is often better because of the dedicated nature of the system, especially on the faster personal computers.

REFERENCES

[1] D. T. Lee and B. J. Schachter, "Two algorithms for constructing a Delaunay triangulation," International Journal of Computers and Information Science, Vol. 9, No. 3, pp. 219-242, 1980.

[2] D. F. Watson, "Computing the n-dimensional Delaunay tesselation with application to Voronoi polytopes," Computer Journal, Vol. 24, No. 2, pp. 167-172, 1981.

[3] Z. J. Cendes and D. Shenton and H. Shahnasser, "Magnetic field computation using Delaunay Triangulation and complementary finite element methods," IEEE Transactions on Magnetics, Vol. MAG-19, No. 6, pp. 2551-2554, November 1983.

[4] M. Fortin and P. Tanguy, "A non-standard mesh refinement procedure through node labeling," International Journal for Numerical Methods in Engineering, Vol. 20, pp. 1361-1365, 1984.

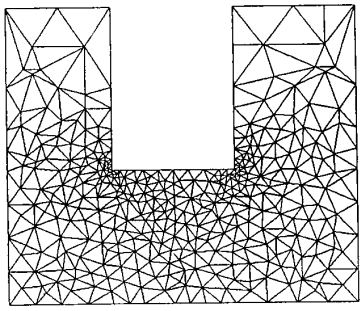

Figure 1. An adaptive mesh using Delaunay triangulation

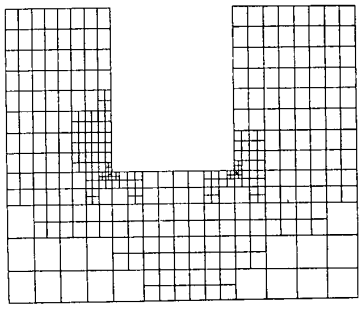

Figure 2. An adaptive mesh using quadrilateral elements.

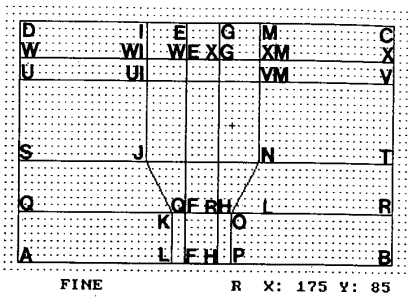

Figure 3. Outline of a simple geometry. A ferrite core over a conducting surface is modeled.

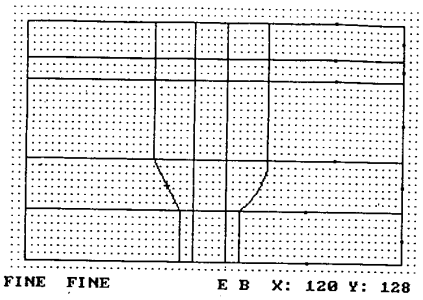

Figure 4. Geometry after adjustment of segment J-K.

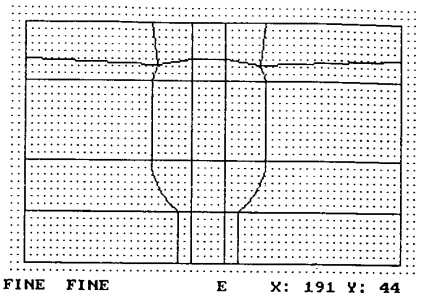

Figure 5. Complete geometry after adjustments, ready for mesh generation

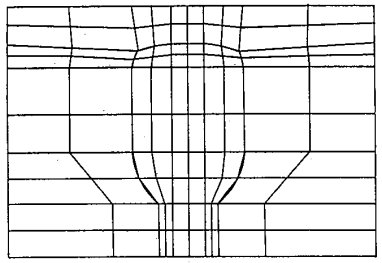

Figure 6. Plot of a simple mesh generated over the geometry in Fig. 5

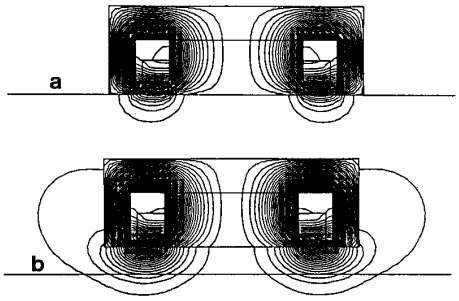

Figure 7. Plotter output for a ferrite pot-core over a conducting surface
a. Pot-core in contact with surface.
b. Pot-core 4mm over the surface.

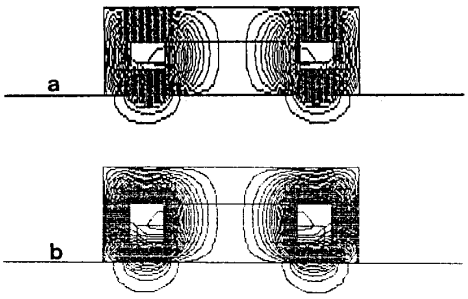

Figure 8. Dot-matrix plot of screen displays of Fig. 7a.
a. Color Graphics Display ()
b. Enhanced Graphics Display ()