THE STRUCTURE OF A SIMPLE PROGRAM

J.R. Dunn

School of Electronic and Electrical Engineering
University of Birmingham

1.0 INTRODUCTION

This paper presents a suite of fairly elementary programs for Finite Element Analysis; they were developed most fully for axisymmetric structures with anistropic materials and without losses; thus the resonant frequencies and modes of vibration can be studied, but not radiation loading nor the electro-mechanical properties as such. Piezo-electric properties are included only in so far as the mechanical parameters used in the data for the materials are appropriate for the open-circuited or the short-circuited case, the former mainly for hydrophones and the latter for projectors. The driving forces are assumed to be sinusoidal, since it is the dynamic behaviour which is to be analysed, but the static case, principally for hydrophones, is modelled simply by setting the frequency to zero. The main emphasis has been on the use of the programs as an aid in the development of sonar transducers rather than on the development of the programs for their own sake. Furthermore they are being used on relatively unsophisticated micro-computers with limited memory, and for this reason the sub-programs are run independently as separate modules except in so far as the data required for each program is generated by a previous program, and the output passes either directly to hard copy or to a disc file for the next stage. The outline arrangement is generally similar to that described in the preceding paper "Finite Element Principles", and the individual stages or programs are as follows:-

i) Generation of structural data
ii) Calculation of the stiffness matrix for each element
iii) Assembly into the global stiffness matrix
iv) Solution for displacements
v) Graphical output (displacements)
vi) Calculation of stresses
vii) Graphical output (stresses)

The individual sections are discussed in more detail in the next section; parts (iii) and (iv) are combined into one program and parts (vi) and (vii) have been only partially implemented.

These programs were developed in the first place by translating simple program modules in Fortran, mostly taken from ref. (2), firstly into BBC Basic and then into Turbo Pascal as a better computer (IBM compatible) became available. Memory was severely limited, and so they had to be written in very compact forms and with the bare minimum of facilities so that jobs of reasonable complexity could be run. The main limitation is on the fineness of detail in relation to wavelength in the structures which can be analysed, but even so much valuable work has been done. The real message is that one does not need access to a mainframe computer if one is prepared to write one's own software, although commercial software is becoming available which will run on micros. Because the analysis seemed simpler, the programs have been written for constant-strain three-noded triangular elements, initially for plane strain or plane stress, but the fully developed programs are all for axisymmetric structures, since this is the common form for many of the transducers being investigated and programs for general three-dimensional structures would be too complicated at this stage. Some work has been done on programs for six noded triangles; they are running satisfactorily in BBC Basic, but they have not yet been translated into Pascal. The descriptions which follow do not go into great detail in algorithms used, particularly in the calculation of the element stiffness, but the source codes can be made available.

2.0 THE ESSENTIALS OF THE PROGRAMS

2.1 Data generation

The basic data for the structure comprises the mechanical properties of the materials, the co-ordinates of the nodes, the nodes and material for each element (i.e. the element description), the externally applied forces and the nodal fixities (the nodes with prescribed zero displacements). The latter are introduced by adding a very large stiffness into the appropriate self-stiffness terms in the assembled structural matrix, the magnitude of this stiffness being a variable in the data file, although it is the same for all fixed nodes and there is no need to change it at any time; in this way the nominally zero displacements are in fact very small, of the order of at least 10^{20} times less than the average displacement. The frequency is introduced into the assembly and solution phase and so it does not appear in the data file. The units used in the data are millimeters for linear dimensions, Megapascals for stiffnesses, Newtons for applied forces, and gm/cc for densities. Various correcting factors are used within the programs so that these units are self-consistent and the calculated displacements are then in mm.

The data file is stored on disc as a text file, containing numbers only, so that it can easily be checked and corrected using a simple text editor, with the arrays set out in a logical format for easy reading. This data may be optionally printed out in a similar format but with full annotation as part of the program for calculating the element stiffnesses. A heading section gives the numbers of elements, nodes, materials, nodes with external loads and nodes with constraints, and also the half-bandwidth; these numbers are needed to define the active sizes of the arrays at the start of each program. The half-bandwidth is the parameter which defines the row length in the assembled stiffness matrix, and it is equal to two more than twice the greatest difference in nodal numbers round any element; the factor of two arises from there being two variables at each node, i.e. the radial and longitudinal displacements. In this suite of programs the half bandwidth has to be put in by inspection, but it is of course easy enough to write a simple routine for calculating it, for example during an automatic scheme for working out the element definitions. The fixity stiffness is also given in the heading. In the interests of minimising the memory requirements and the computation time in the assembly and solution phase, the declared half-bandwidth should not be larger than necessary; there is no fundamental reason why it cannot actually be larger, but this would lead to many redundant multiplications by zero. The material properties are given as 4*4 matrices, although by symmetry only 10 of the 16 terms are independent for a general anisotropic material such as the piezo-electric ceramics; if the axes of the anisotropy are aligned with the axes of the basic geometry, then 3 out of the 10 terms are zero, but there are some jobs in which the axes are not so aligned, so these terms are in that case no longer zero. It is more convenient for the properties of isotropic material to be put in the same matrix form, rather than as the conventional parameters Young's Modulus and Poisson's Ratio; in any case the 4*4 matrix would still have to be set up, explicitly or implicitly, for use in the stiffness program.

2.2 Element Stiffness Generation

The essential function of this program is the calculation for each element in isolation of the forces generated at each node by displacements at each node in turn along each axis in turn all other displacements being in effect rigidly set to zero. The ratio of force to displacement is stiffness, and these stiffness coefficients form a 6*6 matrix for each element, which is symmetrical about the leading diagonal and hence there are 21 independent terms out of 36. The calculation of this stiffness matrix follows the theory in the accompanying paper "Finite Element Principles" by B.V. Smith, section 5, with the matrix multiplications carried out explicitly; this simplifies the programming at the expense of an appreciable number of multiplications by zero. Because the stress and strain are assumed to be constant over each element, the integration over the area of the element involves only a trivial multiplication by the area. The stiffness matrices are transferred to disc element by element as a file of records, each record consisting of the element number, included as a marker for monitoring the reading of the file in the next stage of the analysis, the mass associated with each node, which is one third of the mass of the element and which is required for calculating the inertia terms for the dynamic analysis, and the 21 independent stiffness terms. This file is in binary form, for minimum length, since a text file would be unnecessarily long (by a factor of about two and a half). This program also allows an optional print-out of the basic data, so that the results from the later programs (displacements only at present in the Pascal implementation) can be accompanied by the relevant data.

2.3 Assembly and Solution for Displacements

The next stage is the combining of the stiffness matrices of all the elements into a global stiffness matrix which defines the behaviour of the whole structure, and this process is known as the assembly. Each term in this matrix represents the total stiffness between any pair of nodes (including a node paired with itself) in the same way that a term in the element stiffness matrix represents the same within an isolated element; thus each global term is the sum of the stiffnesses for the same pair of nodes for every element which contains that pair. The equivalent fully assembled structural stiffness matrix is square and symmetrical, with a size equal to the total number of variables, i.e. the number of nodes times the number of variables per node (in this case two), but if the numbering of the nodes is organised to the best advantage all the non-zero values are clustered symmetrically about the leading diagonal. Hence this matrix can be assembled and stored in memory as N rows by M columns, where M is the half-bandwidth (hence its importance) and N the number of variables, and the first column contains the terms in the leading diagonal of the complete square matrix; it is by far the largest array which needs to be fully in memory at any time, and hence its size is the governing factor which determines the size of the job which can be handled on a particular computer.

It should now be clear why the half-bandwidth should be minimised by careful numbering of the nodes in the original structure; on the other hand the numbering of the elements has no constraints. The data used in this program are the element definitions, giving the nodes associated with each element, the nodal fixities, the external loads, and the element stiffness file. The basic data except for the nodal co-ordinates is held in memory, but the stiffness data for each element is taken in turn from disc storage and the terms added into the appropriate locations in the overall structural stiffness matrix. This matrix is then modified in two ways; firstly the high stiffness is added to the terms in the first column which correspond to the zero nodal displacements (to be strictly accurate these are very, very small, much smaller than any other displacements), and secondly the inertial terms are subtracted from the terms in the same column. Each inertial term is the radian frequency squared time the mass associated with each node in each element, so that the number of inertial terms modifying each nodal variable is equal to the number of elements meeting at that node.

The external forces are included as terms in a column matrix, the length of which is equal to the number of variables; for nodes without external forces, the great majority in most cases, the values are of course zero. This column matrix represents the right hand side of a set of simultaneous equations, the unknowns being the nodal variables and the full N*N matrix in non-redundant form representing the coefficients on the left hand side.

The next stage is the solution of this set of equations to determine the nodal displacements. The solution is carried out by Gaussian reduction and back-substitution, and therefore the full assembly has to be carried out before the solution phase begins. The essential mechanism of the Gaussian reduction is that in the full N*N matrix all terms below the leading diagonal are progressively reduced to zero by using each equation in turn to eliminate in the equation below it those terms which occur before the leading diagonal term in each row. The details of the programming are in fact fairly simple, involving only three nested loops. At the end of this phase the last equation is reduced to one term on the left hand side, and hence the last unknown can be found immediately. This value can then be substituted into the previous equation, so that a solution for the next to last variable can be found, and the process, the Back Substitution phase, continues back up the set of equations. This also is a simple process, with only two nested loops. The alternative method of solution, the Frontal method, allows much bigger jobs to be handled by combining assembly and reduction, and transferring intermediate results continuously to disc; in this method the numbering of the elements is much more important than the numbering of the nodes. The programming is more complicated, as there is a lot of re-ordering of the nodes, but the active memory required is substantially reduced. The displacements are transferred to disc storage as a text file, which may therefore be easily examined on an editor, and they may be printed out if required; the disc file includes the frequency as a header. If a search is being made for a resonance frequency, i.e. a frequency at which the displacements would go to infinity in the absence of dissipation, it is usually not necessary to print out full sets of results for every frequency, but only to examine the displacement at a selected node, and this is just as conveniently done from the text file.

This program can be rerun as necessary with different frequencies without the need to re-enter the basic data, although the element stiffness file has to be re-read each time, since it is too long to remain in memory.

2.4 Graphical Output

The rest of the Pascal programs which are running at the present time produce a hard copy output of the nodal displacements. The inputs required for these programs are the element descriptions, the nodal co-ordinates and the displacements. Two versions have been written showing the pattern of the displacements in different ways. One version shows the distorted elements in an exaggerated form, the relative scale of the displacements being selectable; zero displacement therefore shows the element mesh as defined by the basic data. The other form plots the vector displacement of each node and of the centroid of each element as a short arrow with a length proportional either to the displacement or to the square root of it. The effect of the latter scaling is to reduce the range of displacements on the plot, and this is useful if there is a wide range to be displayed. The displacement at the centroid is simply the mean of the displacements at the three nodes associated with each element, and this is justified on the grounds that the formulation of the stiffness matrix is based on constant strain and hence linear variation of displacement with position over each element. The scale of the basic mesh is automatically calculated within the program so that it fits on A4 paper to the best advantage, including the orientation. The graphics commands are written for a "Sweet P" plotter, but they could easily be rewritten to suit a Hewlett Packard plotter.

2.5 Post Processing

There is no means at present of calculating resonant frequencies directly, but they are found by a combination of interpolation and rerunning the assembly/solution phase with different frequencies. Since at resonance with a loss-free structure the displacements go to infinity, it is best to consider the reciprocal of the displacement at a chosen node, usually one of the driven ones. This reciprocal goes through zero with a change of sign as the frequency is varied through resonance, the variation being linear with frequency close to resonance. Below resonance the sign of the displacement must be the same as that of the driving force at that point (consider what happens at one end of a bar, the other end of which is fixed, as the frequency is raised from zero), and the slope of the displacement vs. frequency must always be positive. Thus every resonance must be followed by an anti-resonance (zero displacement), and this behaviour is exactly analogous to the behaviour of electrical networks containing pure reactances only (Foster's reactance theorem). Thus by considering the sign of the displacement it is possible to estimate where the nearest resonance is, although in many cases in the analysis of transducers one usually has a good idea what the resonance should be, and successive recalculations and interpolations should take one to the resonance without difficulty.

Programs have been developed for calculating and plotting the stresses in the elements, both direct (radial, longitudinal and circumferential linear stresses, and radial/longitudinal shear stress) and as principal stresses (two magnitudes and an angle in the radial/longitudinal plane, and the circumferential stress), but these have not yet been translated from BBC Basic to Turbo Pascal. The graphical output plots the principal stresses as short lines proportional to the stress and oriented at the correct angle, with the circumferential stress indicated by a square of proportional size, these symbols being centred in each element.

3.0 LIMITATIONS ON THE PRESENT PROGRAMS

With a given computer the final limiting factor is the size of the fully assembled structural stiffness matrix. As written originally in Turbo Pascal there was a limit on the biggest array of 64 K, i.e. about 10,000 floating point numbers, but some additional software (source code) was obtained which enabled bigger jobs to be done by paging the big array into sub-arrays, each with a 64 Kbyte limit, and so the computer memory became the limiting factor. The largest job which has been done had more than 400 nodes and was limited by the computer memory of 256 Kbytes. Another problem which could arise with large jobs is the occurrence of round-off errors in the solution phase, and the use of a maths co-processor is then highly desirable.

4.0 PRACTICAL EXAMPLES

A variety of structures has been analysed, both by these Pascal programs and by their Basic predecessors, ranging from simple cylindrical structures with a single material to complicated devices using piezo-electric materials with the poling axis not aligned with any of the structural axes. Two examples are discussed here, a cylindrical block and a somewhat complicated sonar transducer. Because of the long length of the data files only the size and complexity of the jobs are indicated, and selected outputs are shown graphically.

The first example is the analysis of a cylindrical block of a single material, for which it was required to find the resonant frequencies and the vibrational patterns for the lowest four or five longitudinal modes; the particular point of interest was whether the higher resonant frequencies were sensibly independent of diameter and hence whether they were a proper guide to the bulk velocity of sound in the material. The block was 74 mm in diameter and 78 mm long; it was assumed to be driven by a uniform pressure over a diameter of 59 mm, i.e. well within the outside diameter. The second example was the design of a 'tonpilz' type of transducer with an extra mechanical coupling section at the head end, by which means a good acoustic response should be obtained over an octave bandwidth (27 to 54 kHz); the basic design had been done on a lumped element/equivalent circuit basis, but there were considerable difficulties in the mechanical design of the outer head so that it was light but stiff enough to behave effectively as a rigid piston. The diameter of the head was 20 mm, i.e. about half a wavelength in water at 38 kHz, and the overall length 36 mm.

The general data for these structures was as follows:

	Cylinder	Transducer
Materials	1	4
Elements	840	247
Nodes	465	171
External Forces	12	6
Fixities	31	9
Half bandwidth	32	24

The finer division into elements of the cylinder was necessary in order to achieve an acceptable accuracy for the higher resonances and it was permitted by the improved software which overcame the 64 K limit in Turbo Pascal; because of the simplicity of the structure the nodal co-ordinates and the element descriptions were generated easily by simple pre-processor programs instead of their having to be entered manually with a text editor, a very tedious exercise with such a large number of elements and nodes. The same pre-processing was used to generate parts of the mesh for the transducer, but this program was too simple to cope with those areas where the nodes do not form a rectangular grid.

Representative graphical outputs are shown in figs. 1 and 2; in both cases one half (radial) of the axisymmetric structure is shown. Fig. 1 shows the vibrational pattern for the cylinder at 27870 Hz, close to its third overtone, fig. 1a showing the distorted mesh and fig. 1b the vector displacements with square root scaling. The latter indicates clearly that the resonance involves much more than simple longitudinal motion, there being significant circular wave motion around clearly defined points of zero motion, which is not obvious in the plot of the distorted mesh. Fig. 2 shows the mainly longitudinal vibration of the sonar transducer at a frequency close to a resonance near the top end of its intended range of operation; there is some indication of the outer head bending, but this should not be too serious a problem in use. This bending is visible in the distorted mesh of fig. 2a, but it is a little clearer in the vector plot of fig. 2b, also with square root scaling, where it can be seen that the outer corner deflects a little inwards towards the centre line.

5.0 REFERENCES

The following books were consulted in developing the programs:

(1) O.C. Zienkiewicz, The finite element method. 3rd edition. McGraw Hill, Maidenhead, 1977.

(2) Y.K. Cheung & M.F. Yeo, A practival introduction to finite element analysis. Pitman, London, 1979.

(3) B.M. Irons & S. Ahmed, Techniques of finite elements. Ellis Horwood, Chichester, 1980.

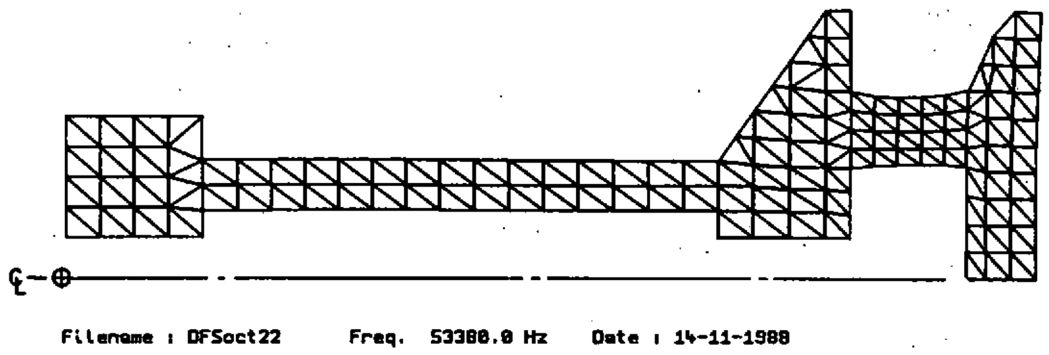

Fig. 2a "Tonpilz" transducer: displaced mesh

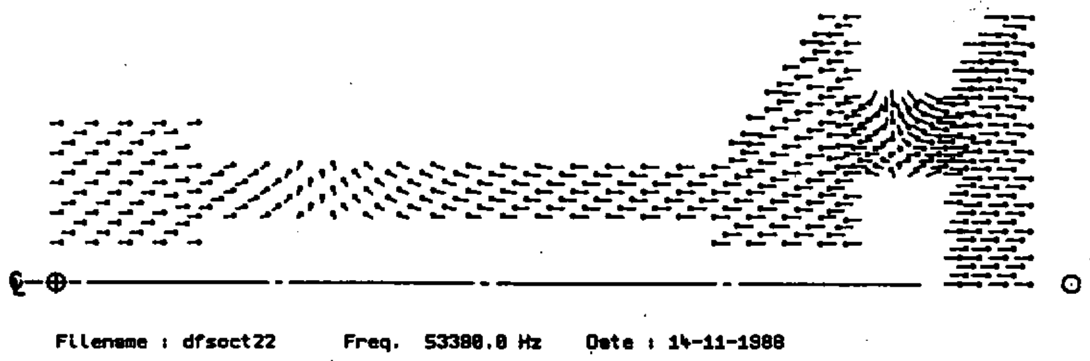

Fig. 2b "Tonpilz" transducer: nodal displacements

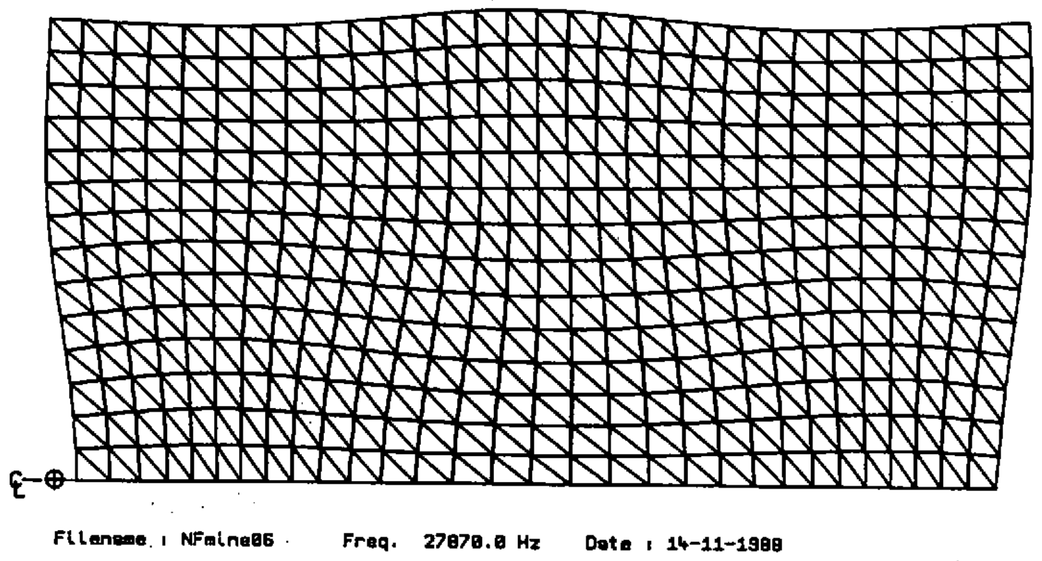

Fig. 1a Uniform cylinder : displaced mesh

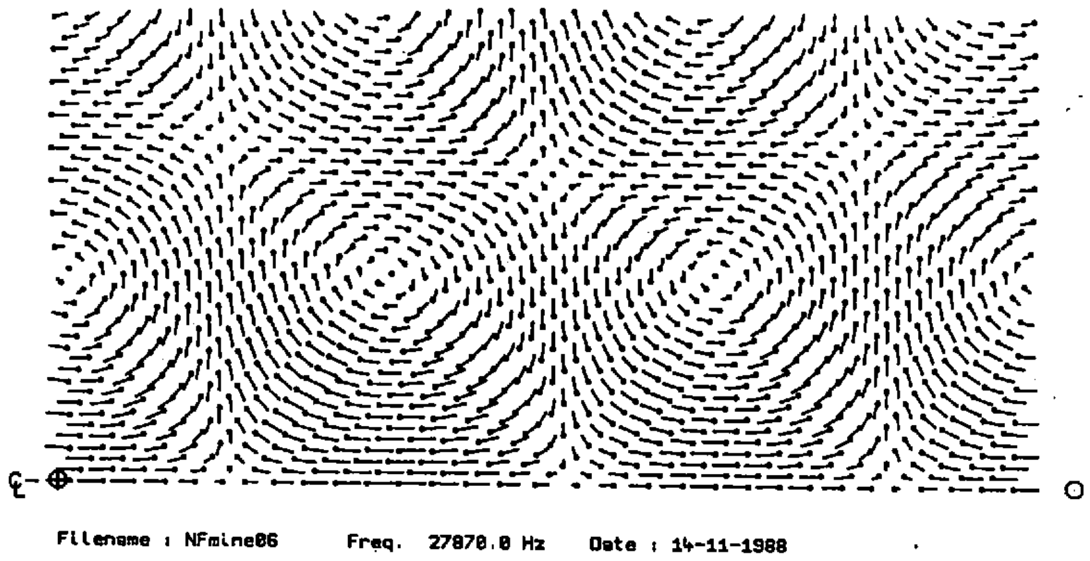

Fig. 1b Uniform cylinder : nodal displacements