This is to certify that the

thesis entitled

Modelling Cylindrical Discontinuities
In Running Shoe Soles Using A
Condensed Finite Element Formulation

presented by

Andrew James Hull

has been accepted towards fulfillment
of the requirements for

M.S. degree in Mechanical Engineering

Clark Radcliffe
Major professor

Clark Radcliffe

Date February 21, 1985

MODELLING CYLINDRICAL DISCONTINUITIES IN RUNNING SHOE SOLES USING A CONDENSED FINITE ELEMENT FORMULATION

By

Andrew James Hull

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Department of Mechanical Engineering

1985

ABSTRACT

MODELLING CYLINDRICAL DISCONTINUITIES IN RUNNING SHOE SOLES USING A CONDENSED FINITE ELEMENT FORMULATION

By

Andrew James Hull

This thesis presents modelling techniques that are useful to analyze the effects of holes in running shoe soles. A two dimensional finite element analysis of a running shoe containing one to three horizontal holes in the heel area was performed using condensed two dimensional elements. The displacement results serve as a basis for the effects of these holes during heel strike. A three dimensional element program that condenses a twenty node solid into a ten node solid and formulates the element stiffness matrix was written. The practical and economical advantages of using condensed formulations were demonstrated.

DEDICATION

To my father

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my major professor, Dr. Clark Radcliffe. His knowledge, guidance, and friendship have made this research possible.

I would like to thank Wolverine World Wide, Inc., for funding this research and Swanson Analysis Systems, Inc., for the use of ANSYS.

Special thanks to my mother, brother, and sister for their constant support they have given me the entire time I have been in school.

I would also like to thank Drs. R.W. Sotus-Little, M.V. Gandhi, and Erik Goodman whose comments and advice were greatly appreciated.

Finally I would like to thank Ken Rowe, Ken Specht, Brian Stermer, and Dan Larabell for their friendship during my academic career.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

a = mapping matrix, x direction

b = mapping matrix, y direction

c = mapping matrix, z direction

C = constraint matrix

K = stiffness matrix

N = shape function

N' = modified shape function

r = natural coordinate position

s = natural coordinate position

t = natural coordinate position

u = generalized position

X = spatial coordinate position

Y = spatial coordinate position

Z = spatial coordinate position

α = generalized displacement

δ = natural coordinate r, s, or t

σ = stress component

τ = shear stress

INTRODUCTION

Shoe designers frequently place holes in running shoe soles. The orientation of these holes is both vertical and horizontal in the shoe sole. The number of holes also ranges considerably: some shoes do not have them, and other shoes incorporate more than 35 holes into the sole design. There are various reasons for including these holes, ranging from weight reduction to aiding in manufacture of the shoe. The effects of these holes has never been fully understood.

This thesis applies a finite element analysis to a two dimensional shoe sole containing one to three holes orientated horizontally in the wedge area. The radius of these holes is also varied to study the effects of different size holes. Due to the number of nodes required, the current finite element program in use does not have the ability to model the three dimensional problem. A three dimensional modelling technique is discussed using nodal condensation to reduce the three dimensional problem to a manageable size.

THE FINITE ELEMENT METHOD

Analytical methods in structural analysis have been studied for many years. Exact solutions are usually not available to problems with complicated geometry and/or boundary conditions. The emergence of high speed computers, however, makes it possible to accurately apply numerical methods to complex problems. A commonly used numerical technique for structural analysis is the finite element method[1,2]. This method involves a discretization of a continuous structure into a number of smaller parts (elements). Equations are first formulated for each element with individual loading and boundary conditions, then a set of equations are assembled modelling the entire system. The resulting equations are then solved, yielding the structural response.

For an elastic body, the stiffness matrix and load vector can be formulated using the finite element method, and the displacements of the body can be determined (Appendices A, B, C, and D). The best method to verify the results of any analytical technique is actual testing. When testing is not feasible, a commonly used convergence test is to reduce the element size and compare the results of a denser grid of smaller elements to the base model. If the differences in responses are small, then the base model probably has enough elements to model the system. This was the verification technique used in this paper, comparing solutions of variously populated grids to insure

analytical convergence.

The major topic of this thesis is the effect of holes in running shoe soles, therefore, a modelling technique was developed to determine the global effects of holes in materials. Consider the three 2 dimensional element test grids shown in Figures 2.1, 2.2, and 2.3. All three were subjected to a single point load on the upper right node. The four left hand nodes were constrained in the x and y direction. The structure was six units long, six units high, and the hole in the middle was one unit in diameter.

There was a choice of two dimensional models based on modelling assumptions. The model assumption of plane strain was used. Plane strain is a specialization of three dimensional linear elastic theory. It represents the situation where the structures geometry and loading are constant in the z direction and the component of displacement perpendicular to the x-y plane is zero. This is discussed in Appendix E. Plane stress, the other two dimensional specialization of three dimensional elastic theory, represents the situation of a very thin, flat structure, whose loading occurs only in the x-y plane of the plate. All three models were developed and analyzed using ANSYS. ANSYS is a general purpose finite element program developed by Swanson Analysis Systems, Inc.

Figures 2.1 and 2.2 illustrate the five and eight node elements used to model the hole in the middle of the solid. Each element bordering the hole had three active nodes on the hole boundary. The midside node on the boundary of the hole was left on to insure a curve was fitted thru this area for the element interpolation. Different

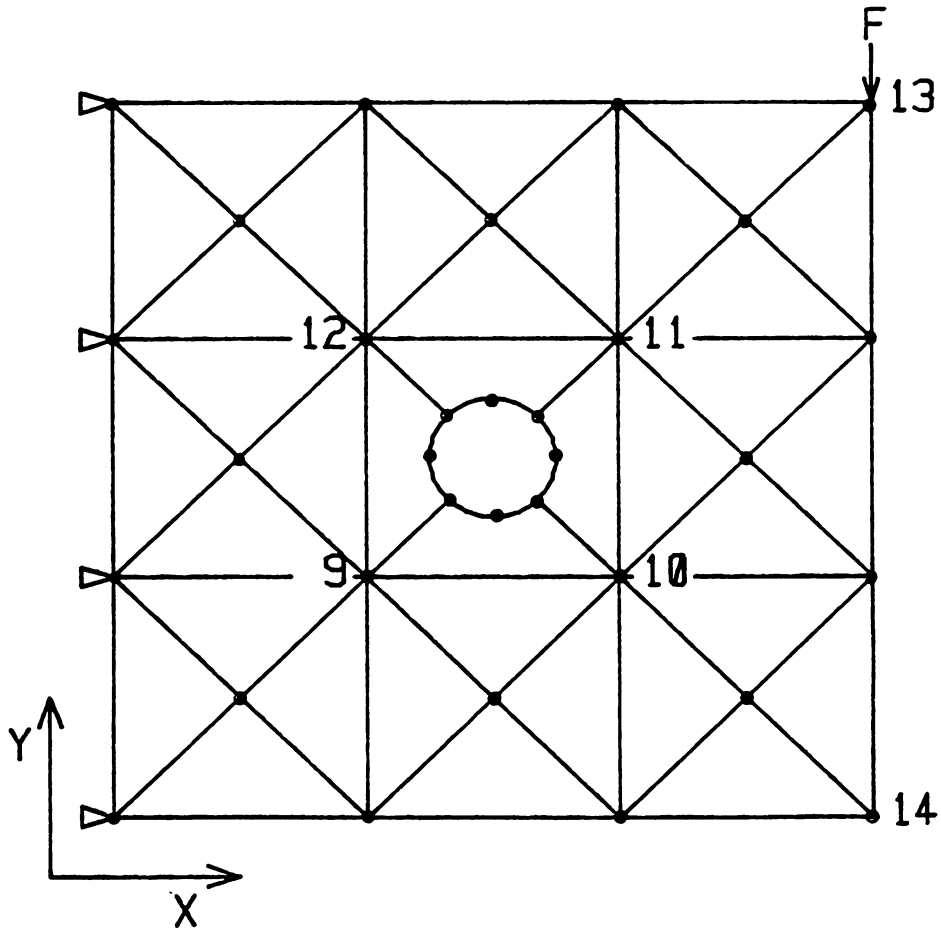


Figure 2.1 Base grid with condensation

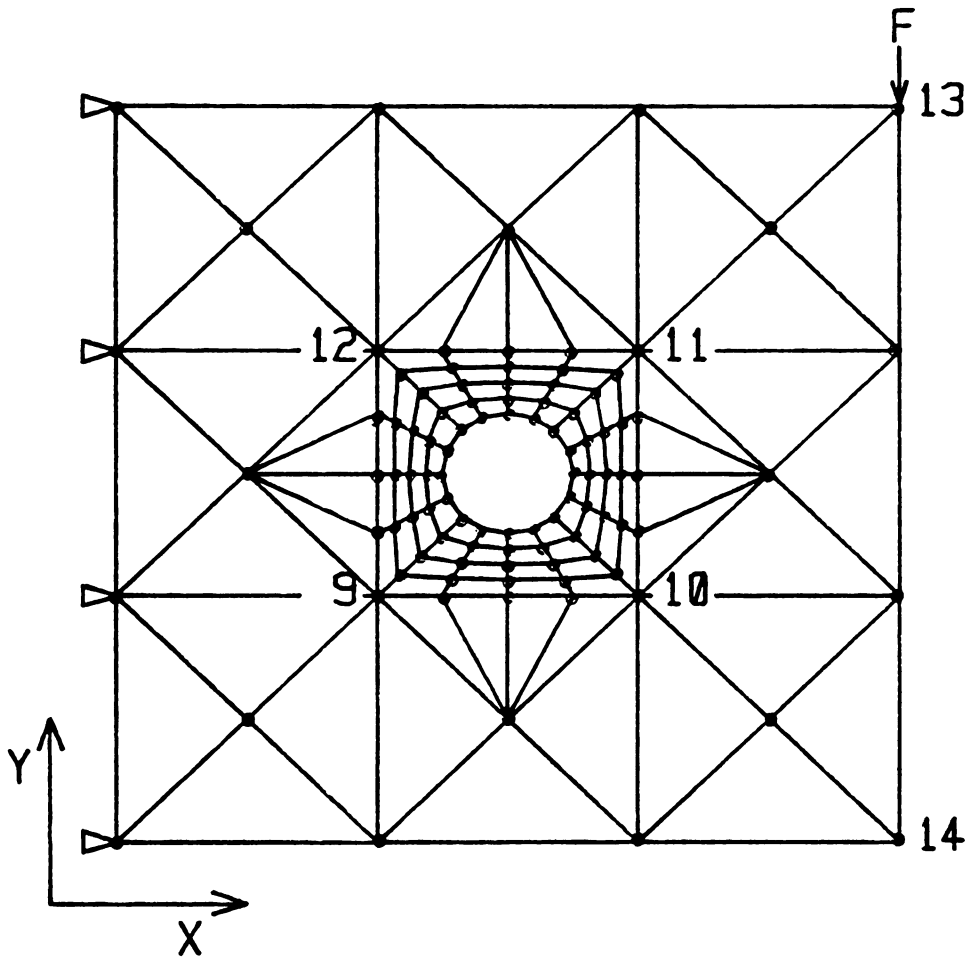


Figure 2.3 Refined grid with linear elements

sources recommend different locations for this node[1,3]. From a theoretical standpoint, the midside node must be located near the middle of the curve between the two end points. When the midside node is too close to the end nodes of the curved side, the Jacobian matrix becomes singular, and has no inverse. The inverse of the Jacobian matrix must exist to insure a unique mapping from spatial coordinates to natural coordinates (Appendix B).

Figure 2.1 was the base model. It consisted of 36 elements and 32 nodes. The four elements in the middle of the model were five node isoparametric solid elements. They each had three nodes condensed out of the element matrices (Appendix B). Figure 2.2 was the second base model, consisting of 40 elements and 40 nodes. The four elements in the middle of the model were eight node isoparametric solid elements. This model was included so that the effects of nodal condensation could be viewed. Figure 2.3 was the refined model. It consisted of 108 elements and 100 nodes. All the elements of Figure 2.3 used linear interpolation. The results were compared in Tables 2.1 and 2.2. The displacements of the base models in Figures 2.1 and 2.2 were compared to the displacements of the refined model in Figure 2.3. Nodes 1 thru 8 were on the circle, and nodes 9 thru 12 were labeled on each figure. Node 13 and 14 were the right corner nodes, node 13 being located directly under the load.

The results from Table 2.1 shows that the local convergence (the displacements on the hole) in the x direction was very poor. The grids in Figures 2.1 and 2.2 would not be adequate if the displacements on the circle were desired. The global convergence (the

Table 2.1 X displacements for Figures 2.1, 2.2, and 2.3

Node number	Figure 2.3 -refined grid a	Figure 2.1 -base grid b	Normalized difference	Figure 2.2 -base grid b	Normalized difference
1	-.0029	-.0931	-2.5	-.0698	-1.9
2	-.0651	-.1863	-3.4	-.1742	-3.1
3	-.1735	-.2198	-1.3	-.2306	-1.6
4	-.2225	-.1816	1.4	-.2101	0.3
5	-.2172	-.1183	2.8	-.1485	1.9
6	-.1961	-.0682	3.6	-.0887	3.0
7	-.1464	-.0713	2.1	-.0940	1.5
8	-.0533	-.0636	-0.3	-.0627	-0.3
9	-.2631	-.3071	-1.2	-.2732	-0.3
10	-.6614	-.6891	-0.8	-.6689	-0.2
11	.2892	.3262	1.0	.2997	0.3
12	.1864	.2239	1.1	.1990	0.4
13	3.5680	3.5370	-0.9	3.5590	-0.3
14	-2.2870	-2.2630	0.7	-2.2810	0.2

Table 2.2 Y displacements for Figures 2.1, 2.2, and 2.3

Node number	Figure 2.3 -refined grid a	Figure 2.1 -base grid b	Normalized difference	Figure 2.2 -base grid b	Normalized difference
1	-1.858	-1.890	0.4	-1.889	0.4
2	-2.486	-2.429	-0.7	-2.455	-0.4
3	-3.104	-2.951	-1.8	-3.005	-1.2
4	-3.378	-3.174	-2.4	-3.232	-1.7
5	-3.112	-2.990	-1.4	-3.018	-1.1
6	-2.455	-2.450	-0.1	-2.431	-0.3
7	-1.822	-1.896	0.9	-1.855	0.4
8	-1.584	-1.682	1.1	-1.673	1.0
9	-1.283	-1.288	0.1	-1.275	-0.1
10	-3.758	-3.635	-1.4	-3.712	-0.5
11	-3.885	-3.745	1.6	-3.842	-0.5
12	-1.200	-1.196	~0.0	-1.196	~0.0
13	-8.590	-8.424	-1.9	-8.532	-0.7
14	-5.578	-5.433	-1.7	-5.525	-0.6

Normalized difference = $[(a_i - b_i)/\max(a)]$ X 100%

displacements at common points not on the hole) in the x direction was fair in Figure 2.1 and good in Figure 2.2. The results from Table 2.2 shows that the local and global convergence were good to excellent for Figures 2.1 and 2.2. For the two dimensional model, the modelling technique in Figure 2.1 was used. For each hole modelled the number of nodes needed using a five node element was eight. For the conventional method of using many linear elements, the number of nodes needed was 68. Since each node had two degrees of freedom (adding two equations per node to the system of equations) the difference in number of equations between the base grid and the refined grid was 128. Furthermore, the formulation time for four 5 node elements was less than the formulation time for 64 four node elements. The use of higher order elements in this case resulted in a time savings on the computer while retaining the numerical accuracy in the the finite element analysis.

TWO DIMENSIONAL ANALYSIS

The first analysis of the running shoe was a model of a Brooks Supervillinova (Figure 3.1). A two dimensional model of the heel/wedge area was developed (Figure 3.2), which consisted of 82 elements and 103 nodes subjected to a plane strain analysis. The shoe in this model was cut with a plane that ran parallel to the longest direction of the shoe and was perpendicular with the ground.

Figure 3.1 Brooks Supervillinova running shoe

The bottom row of elements in Figure 3.2 represented the outsole, the middle three rows of elements represented the midsole, and the top row of elements represented the sockliner. The sockliner was a removable

layer of material placed in the running shoe to increase comfort. The model was constrained in the x and y directions at the nodes marked with triangles. Although there was a tread pattern on the outsole, this area was modelled as a solid since the protrusions occupied over 85% of the volume under the loaded area. The material properties were determined using a Model 1331 Instron Servohydaulic Materials Testing Machine located in the Biomechanics Department at Michigan State University. They are listed below in Table 3.1.

Table 3.1 Material properties of the Brooks Supervillinova

Layer	Young's Modulus	Poisson's Ratio
Outsole	260 psi	.25
Midsole	130 psi	.25
Sockliner	80 psi	.25

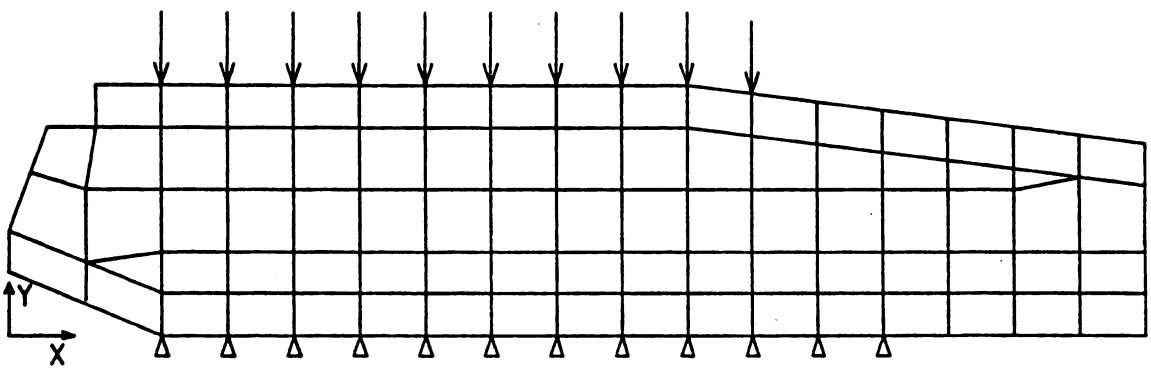


Figure 3.2 Base grid of a Brooks Supervillinova

The section of the size eight and a half shoe modelled (Figure 3.2) was 5.2 inches long and 1.2 inches thick. Shoe designers who include

horizontal holes in their midsole design normally locate them in the heel area of the shoe. The model was developed to investigate the effects of these holes.

The following assumptions were used: 1) All sole materials behaved elastically, 2) the shoe upper provided negligible additional stiffness to the sole 3) there was no slip between the ground and the bottom of the shoe, 4) The runner struck the ground with the heel area of his shoe, 5) Plane strain occured.

The loading for this model was derived from experimental measurements of pressures exerted by running human subjects.

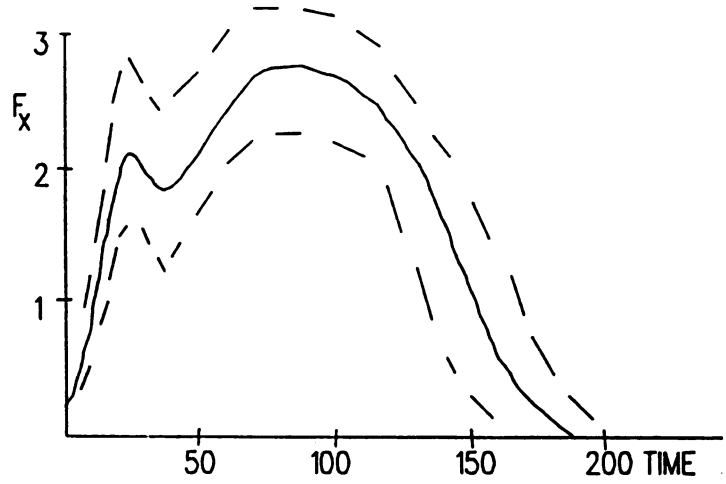


Figure 3.3 Average vertical component of ground reaction force

Figure 3.3[4] was the vertical component of ground reaction force in 12 subjects running at 6.5 minutes per mile. The solid line on the graph was the average value and the dashed lines were the maximum and minimum ranges. Units of force (F_x) were in body weight and the units of time were in milliseconds. The ground reaction force in the x direction was vertical, and a positive force was orientated in an

upward direction. The first maxima of the curve had a value of 2.1 units of body weight at approximately 25 milliseconds. Since this finite element model only analyzed wedge area displacements, the first local maxima was used as it corresponded to heel strike. Using a subject weight of 150 lbs, the reaction force at heel strike in the shoe was (-2.1 X 150)= -315 lbs. Since the width of the shoe was 3 inches, and a plane strain analysis has a unity thickness, the loading for this model was (-315 / 3)= -105 lbs total in the x direction. Distributing the -105 pound load over ten nodes resulted in a force vector at each node equal to (-105 / 10)= -10.5 lbs per node in the x direction.

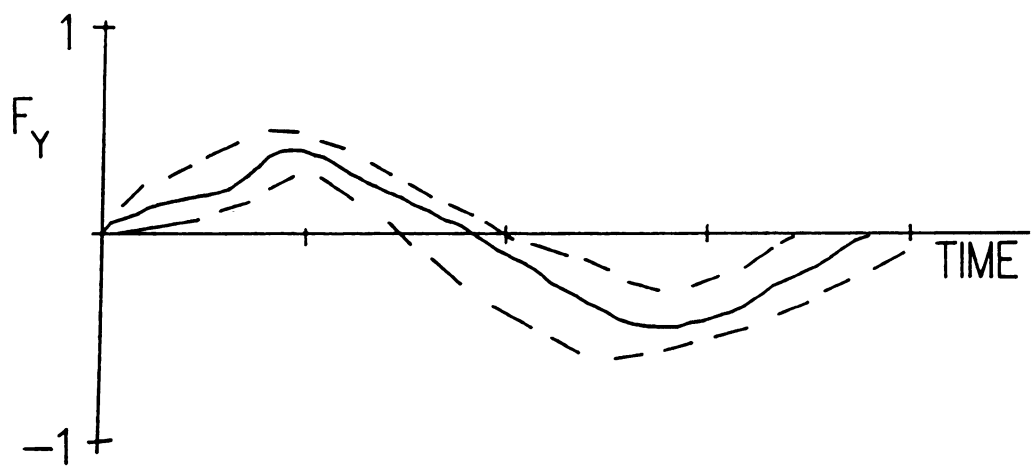


Figure 3.4 Average shear component of ground reaction force

Figure 3.4[4] was the average shear component of the ground reaction force. The ground reaction force (F_y) in the longitudinal y direction was determined using Figure 3.4. Positive force was forward. At 25 milliseconds the average shear component was .2 units of body weight. Using the same method of calculation as above, the average shear force per node was equal to 1.0 lbs in the y direction.

The ten loaded nodes were denoted with vectors in Figure 3.2. For convenience they were called nodes one thru 10, one being the left most node. The analysis was also run using a finer grid of 326 elements and 346 nodes, shown in Figure 3.5. The results are listed in Tables 3.2 and 3.3. With the exception of the left most node, the y direction results were fairly close. The y direction displacements, or the amount of compression of the sole, is the critical response in the design of a shoe. This indicates that the base grid (Figure 3.2) was fine enough to model the load and constraints that were being applied to it. A grid modelling the entire sole was also analyzed, using the constraints and loading conditions determined above. The percent difference for the displacement of all common nodes was less than one percent. This justified ignoring the forward part of the shoe as in Figures 3.2 and 3.5.

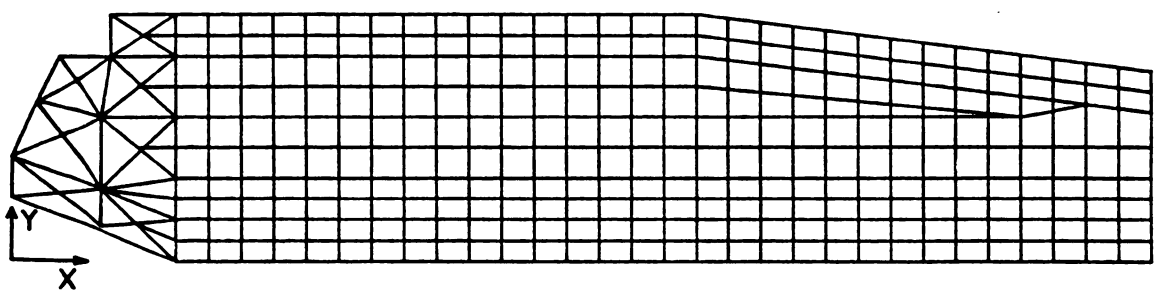


Figure 3.5 Refined grid of a Brooks Supervillinova

Using five node two dimensional elements, the grid in Figure 3.2 was modified to include the effects of holes running across the midsole from the medial to the lateral part of the shoe. The

Table 3.2 Displacements in the x direction

Node number	Displacement -base grid b	Displacement -refined grid a	Normalized difference
1	.01564	.02456	11.6
2	.00482	.02050	20.5
3	.01540	.02712	15.3
4	.02995	.03743	9.8
5	.04409	.04843	5.7
6	.05633	.05842	2.7
7	.06533	.06637	1.4
8	.07061	.07152	1.2
9	.07658	.07643	-0.2
10	.05822	.06437	8.0

Table 3.3 Displacments in the y direction

Node number	Displacement -base grid b	Displacement -refined grid a	Normalized difference
1	-.2992	-.2268	-24.2
2	-.2909	-.2754	-5.2
3	-.2923	-.2836	-2.9
4	-.2867	-.2840	-0.9
5	-.2836	-.2826	-0.3
6	-.2819	-.2815	-0.1
7	-.2807	-.2804	-0.1
8	-.2779	-.2778	~0.0
9	-.2691	-.2716	-0.8
10	-.2427	-.2364	-2.1

Normalized difference = $[(a_i - b_i)/\max(a)]$ X 100%

displacement effects of one, two, and three holes were determined using radius sizes ranging from .05 inches to .30 inches, incremented by .05 inches. All of the holes had their center located at y=.55 inches from the origin shown in Figure 3.2. For the single hole, the location was at x=2.05 inches. For the analysis using two holes, the x locations were at 1.45 and 2.35 inches. The model that included three holes used x locations of 1.15, 2.05, and 2.95 inches. A displacement plot of a typical node is shown in Figure 3.6. The tabular data for the nodal displacements of nodes one thru 10 for all geometry modifications is listed in Appendix F.

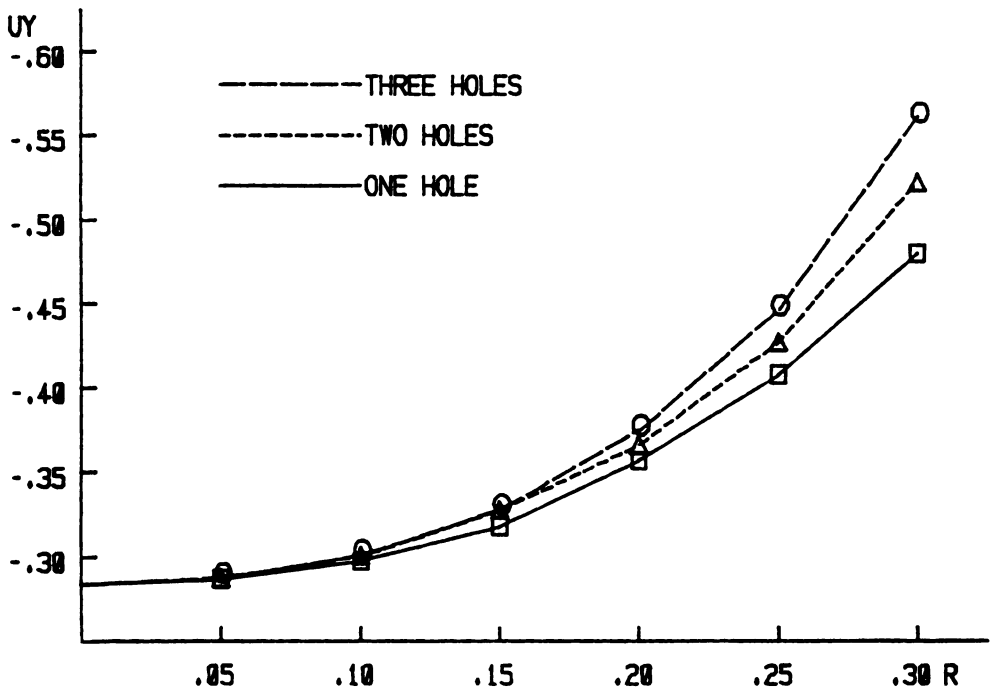


Figure 3.6 Displacements of node 5 versus radius size

A general trend of the data indicated that as hole radius or the number of holes were increased, the absolute value of the displacement

increased (Figure 3.6). For the geometry range tested, the displacement was roughly proportional to the area of the hole (Figure 3.7). The nodal displacements were also sensitive to the location of the holes. Nodes above the holes deformed to a greater extent than the nodes above solid material (Figures 3.8, 3.9, and 3.10).

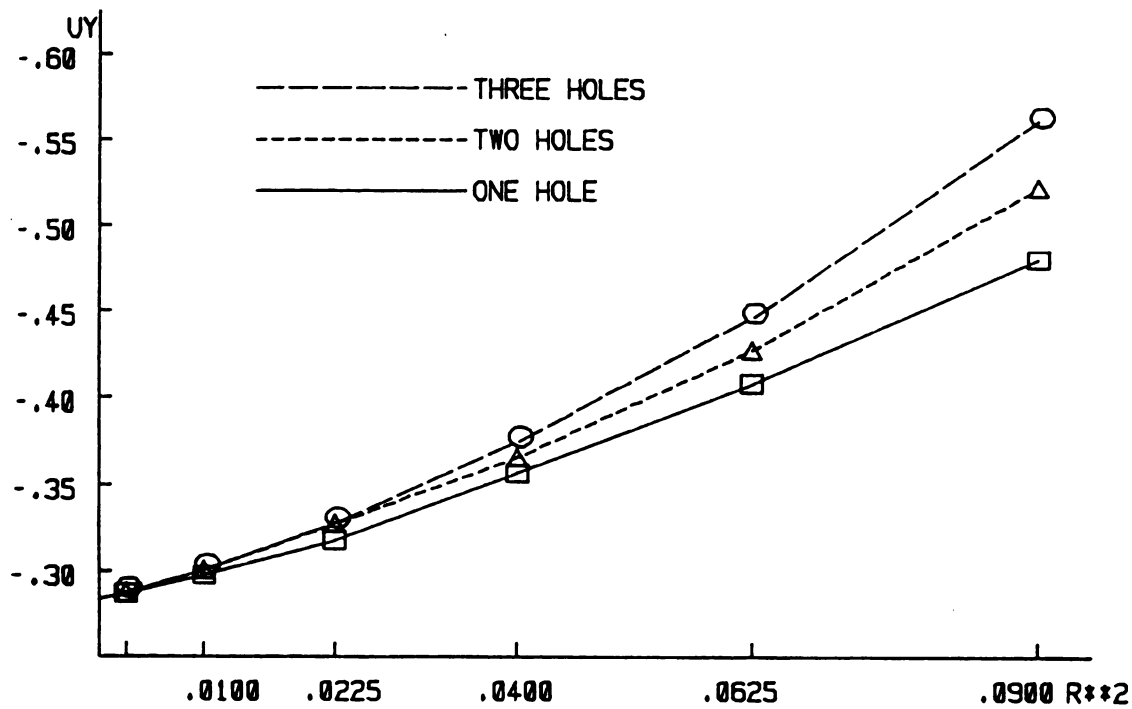


Figure 3.7 Displacement of node 5 versus radius size squared

The main advantage of the two dimensional analysis was rapid solution time, when compared to the three dimensional model, because the grid was simpler and fewer nodes were present in the model. This modelling technique however, was a two dimensional approximation to a three dimensional problem which had finite depth, hence plane strain did not occur. Furthermore, some shoes have the holes running in the

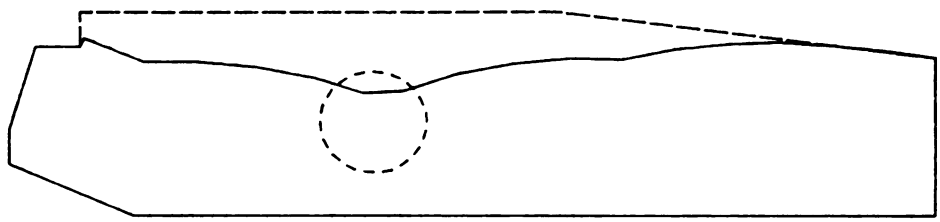


Figure 3.8 Displacement plot of sole with one hole, R=.30

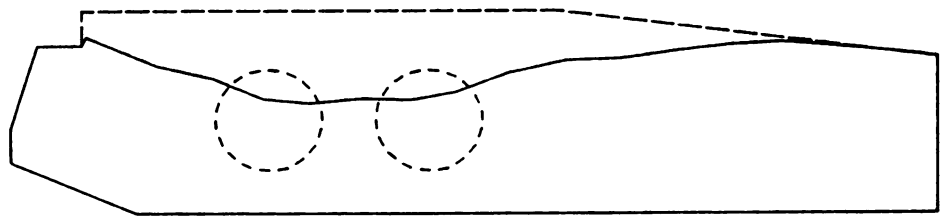


Figure 3.9 Displacement plot of sole with two holes, R=.30

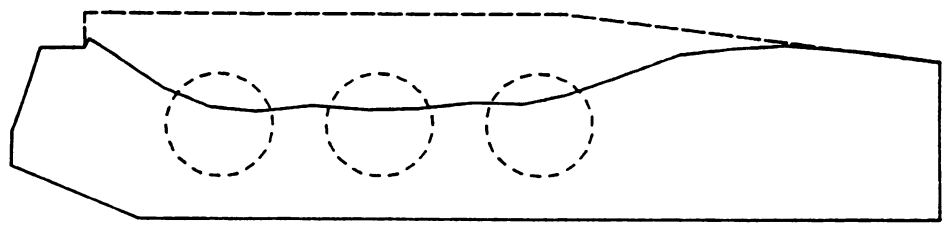
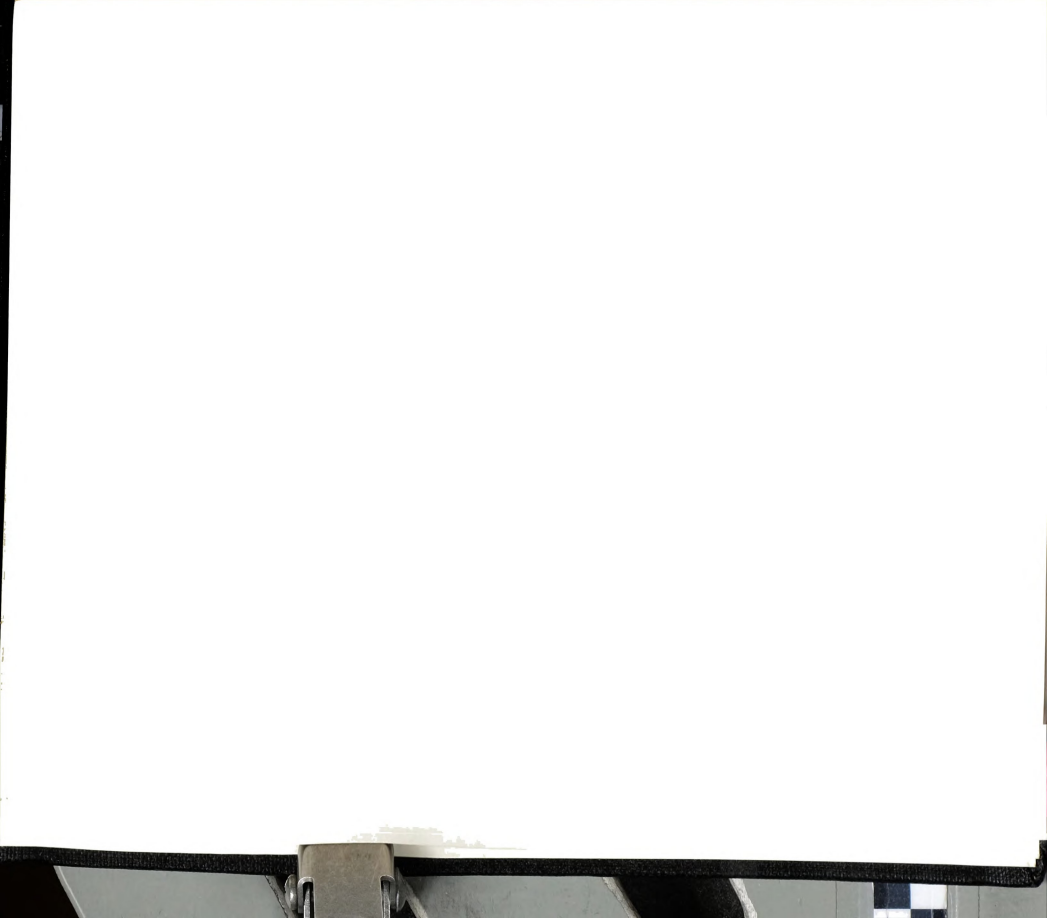


Figure 3.10 Displacement plot of sole with three holes, R=.30



vertical direction which cannot be modelled using a two dimensional approach. The two dimensional approximation agreed with measured values because the out of plane shear force in the z direction was very small and varied widely from one runner to another (Figure 3.11[4]).

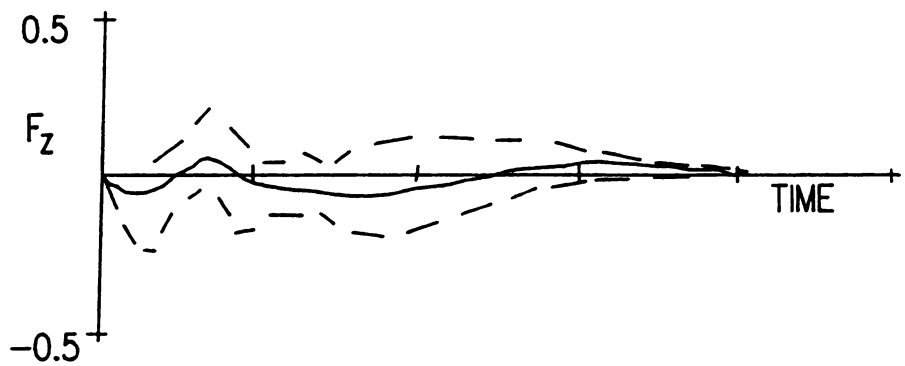


Figure 3.11 Out of plane shear force

THREE DIMENSIONAL ANALYSIS

The three dimensional analysis of structures is more complex than the two dimensional counterpart. The extra degree-of-freedom at each node causes the material stiffness matrix to enlarge from 3X3 to 6X6. The stress components in the finite element formulation changes from

$$\{\sigma\}^T = \{\sigma_x \quad \sigma_y \quad \tau_{xy}\} \tag{4.1a}$$

to

$$\{\sigma\}^T = \{\sigma_x \quad \sigma_y \quad \sigma_z \quad \tau_{xy} \quad \tau_{xz} \quad \tau_{yz}\} \ . \tag{4.1b}$$

The element considerations in the finite element formulation also changes. Because three dimensional curved side elements require midside nodes like their two dimensional counterparts, the definition of a three dimensional curved side element with midside nodes requires 20 nodes instead of the eight present in the two dimensional curved side element. The wave front limitation of the ANSYS educational version is 200 and the connectivity of 20 node three dimensional elements prevent an analysis of a three dimensional model that incorporates vertical holes.

Condensation of the current 20 node solid into a smaller (in terms of number of nodes) element is a technique that can reduce this

problem to a size where an analysis is possible. Quadratic behavior on one side of the condensed element is desirable so the element face can follow a curved surface such as a hole. Curved boundaries cannot be followed by linear elements. The eight midside nodes that do not lie on the hole can be removed through condensation because they do not lie an curved boundaries. Two additional nodes that lie on the hole can be condensed because the holes under consideration are constant radius and do not have a curve in their axial direction. The result is a ten node condensed element that can incorporate a quarter of a hole on one side as a boundary condition. The quadratic displacement behavior along the edges where condensation occurs becomes linear. This is not undesirable since there is almost no loss in accuracy.

Linear three dimensional elements attach to the condensed element in a much more efficient pattern. This is due to element mesh compatibility of the condensed sides of the ten node element with any side of the eight node three dimensional linear element.

The condensation procedure started with a twenty node finite element map to establish the Jacobian matrix. The 20 node element equations that defined the mapping from natural coordinates (r,s,t) to spatial coordinates (X,Y,Z) were

$$\begin{aligned} X = {} & a_1 + a_2 r + a_3 s + a_4 t + a_5 r^2 \\ & + a_6 s^2 + a_7 t^2 + a_8 st + a_9 sr + a_{10} rt \\ & + a_{11} s^2 t + a_{12} s^2 r + a_{13} t^2 s + a_{14} t^2 r + a_{15} r^2 s \\ & + a_{16} r^2 t + a_{17} rst + a_{18} s^2 rt + a_{19} t^2 rs + a_{20} r^2 st \end{aligned} \qquad (4.1a)$$

$$Y = b_1 + b_2 r + b_3 s + b_4 t + b_5 r^2 + b_6 s^2 + b_7 t^2 + b_8 st + b_9 sr + b_{10} rt + b_{11} s^2 t + b_{12} s^2 r + b_{13} t^2 s + b_{14} t^2 r + b_{15} r^2 s + b_{16} r^2 t + b_{17} rst + b_{18} s^2 rt + b_{19} t^2 rs + b_{20} r^2 st \quad (4.1b)$$

$$Z = c_1 + c_2 r + c_3 s + c_4 t + c_5 r^2 + c_6 s^2 + c_7 t^2 + c_8 st + c_9 sr + c_{10} rt + c_{11} s^2 t + c_{12} s^2 r + c_{13} t^2 s + c_{14} t^2 r + c_{15} r^2 s + c_{16} r^2 t + c_{17} rst + c_{18} s^2 rt + c_{19} t^2 rs + c_{20} r^2 st \quad (4.1c)$$

The vectors a, b, and c defined the mapping and were constant for each element defined. They were found by mapping the element back into natural coordinates, which produced three systems of twenty linear equations, the solution being the a, b, and c vectors. This process is similiar to the two dimensional mapping discussed in Appendix B.

To remove midside nodes and condense the element, the shape functions were modified in their natural coordinates. This step differs from most finite element formulations. The shape functions defined by the 20 node element were condensed using[5]

$$\alpha_m = (1/2)(\alpha_{c1} + \alpha_{c2}) \quad (4.2a)$$

where α represents r, s, or t, the subscript m denotes a condensed midside node, and the subscripts c1 and c2 denote the corner nodes that are collinear with node m. Applying this equation at the ten

midside nodes changed the displacement field approximation in the δ direction from

$$\delta(r,s,t) = \sum_{j=1}^{20} N_j(r,s,t)\delta_j \tag{4.2b}$$

to

$$\delta(r,s,t) = \sum_{j=1}^{10} N'_j(r,s,t)\delta_j \quad . \tag{4.2c}$$

where δ is either r, s, or t. The modified shape functions became

$$N'_I = N_I + (N_T/2) + (N_Y/2) \tag{4.2d}$$

$$N'_J = N_J + (N_R/2) + (N_Z/2) \tag{4.2e}$$

$$N'_K = N_K + (N_R/2) + (N_S/2) + (N_A/2) \tag{4.2f}$$

$$N'_L = N_L + (N_S/2) + (N_T/2) + (N_B/2) \tag{4.2g}$$

$$N'_M = N_M + (N_X/2) + (N_Y/2) \tag{4.2h}$$

$$N'_N = N_N + (N_V/2) + (N_Z/2) \tag{4.2i}$$

$$N'_O = N_O + (N_V/2) + (N_W/2) + (N_A/2) \tag{4.2j}$$

$$N'_P = N_P + (N_W/2) + (N_X/2) + (N_B/2) \tag{4.2k}$$

$$N'_Q = N_Q \tag{4.2l}$$

$$N'_U = N_U \quad . \tag{4.2m}$$

This modification eliminated ten nodes per element. Each node had three degrees of freedom, therefore, the condensation technique eliminated thirty equations per element. This technique is discussed in Appendix B and an example of a two dimensional element condensed at

three locations is included.

The derivatives of the shape functions were next multiplied by the inverse of the Jacobian matrix. This step is described in Appendix B. The Jacobian matrix and it's inverse were defined using a twenty node map, while the derivatives of the shape functions were defined using a ten node condensed element. The stiffness matrix was then defined using the equations from Appendix B and numerically integrated using a 3X3X3 grid and the Gauss-Legendre quadrature method (Appendix C).

The element program in Appendix G will build the element stiffness matrix for ten node degraded elements. To insure program accuracy, the stiffness matrix of an ANSYS defined twenty node element was output and this stiffness matrix was modified to incorporate the linear constraints implied by the condensation process. A constraint matrix was defined by

$$\{u\} = [C]\{u\}' \tag{4.3a}$$

where u is the vector containing the x, y, and z displacements of the 60X60 element matrix, u' is the x, y, and z displacements of the 30X30 matrix, and C is the constraint matrix (60X60) that contains the linear displacement constraints on the midside nodes. The constraints were applied to the ANSYS stiffness matrix yielding a new system of equations, as shown in equation (4.3b).

$$[K]\{u\} = [C]^T[K][C]\{u\}' = [K]'\{u\}' \quad . \tag{4.3b}$$

This is best illustrated by the three by three system of equations shown below

$$\begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} = [K]\{u\} \tag{4.4a}$$

which is constrained such that

$$u_1 = u_1' \tag{4.4b}$$

$$u_3 = u_3' \tag{4.4c}$$

$$u_2 = (1/2)(u_1' + u_3') \quad . \tag{4.4d}$$

The constraint matrix becomes

$$[C] = \begin{bmatrix} 1 & 0 & 0 \\ .5 & 0 & .5 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.4e}$$

and assembling this according to equation (4.3b) produces a modified stiffness matrix

$$[K] = \begin{bmatrix} k_{11}+.5k_{21}+.5k_{13}+.25k_{23} & 0 & k_{13}+.5k_{12}+.5k_{33}+.25k_{32} \\ 0 & 0 & 0 \\ k_{31}+.5k_{21}+.5k_{33}+.25k_{23} & 0 & k_{33}+.5k_{32}+.5k_{33}+.25k_{32} \end{bmatrix} . \tag{4.4f}$$

Using this method, the results of the condensed element program were compared to a stiffness matrix multiplied by constraint matrices. The percent difference averaged 0.11% at all nonzero locations in the matrix. This method has the advantage of producing a stiffness matrix to compare to the ten node condensed element stiffness matrix without running an actual finite element analysis.

There is considerable advantages using condensed elements in three dimensional analysis. A running shoe sole such as the Brooks Supervillinova has 37 holes running vertically. The number of equations for this problem without condensation is approximately 36000, which is too large for almost any finite element routine. Since three layers of four elements are needed to model each hole, 544 condensed elements are needed to model this sole. Each condensed element eliminates thirty equations, and the total savings for this type of analysis is 31320 fewer equations. The size of the new problem using condensed elements is approximately 4680 equations, or roughly a magnitude smaller problem. The disadvantage is that the quadratic behavior along the boundaries becomes linear. This, however, is usually not a concern as long as the finite element grid is distributed in a manner that several linear elements fit the displacement curve. Unfortunately ANSYS does not currently have the ability to accept elements input by the user and no actual analysis was performed using this element.

CONCLUSIONS

This thesis has shown that using a condensed finite element formulation can greatly reduce the problem size of three dimensional models. The numerical accuracy using two dimensional condensed formulations was retained, as demonstated by a two dimensional analysis. A program that generates ten node condensed element stiffness matrices is available now to be interfaced with finite element programs that can accept user defined elements. The accuracy of this element was compared to an ANSYS constrained 20 node element, and the percent difference was found to average 0.11% at all nonzero locations. The element has the advantage of eliminating thirty equations every time it is used instead of the base three dimensional element.

The displacement effects of horizontal holes in the sole of a shoe were calculated. The analysis included one to three holes ranging in radius from .05 to .30 inches in the wedge area of the shoe. No attempt was made to determine if these displacements were favorable or detrimental to the runner.

Factors not addressed in this thesis are the dynamic effects of the shoe material, the dynamic effects of the loading, the effect of a ground strike in a location other than the heel area, and the interaction of the shoe upper with the sole. Future topics also include development of a program that uses the three dimensional

condensed element stiffness matrices, assembles them along with the load vectors and linear three dimensional elements, and solves for the displacements of the nodes, resulting in a finite element routine to handle large scale problems.

REFERENCES

REFERENCES

1. Zienkiewicz, O.C., The Finite Element Method, McGraw-Hill Book Company, 1983.

2. Huebner, Kenneth H., The Finite Element Method for Engineers, John Wiley and Sons, 1975.

3. Kohnke, Peter C., ANSYS Engineering Analysis System Theoretical Manual, Swanson Analyis Systems, Inc., 1983.

4. Cavanagh, Peter R., "The Shoe-Ground Interface in Running", American Academy of Orthopaedic Surgeons Symposium on The Foot and Leg in Running Sports, The C.V. Mosby Company, 1982, pp. 30-44.

5. Cook, Robert D., Concepts and Applications of Finite Element Analysis, John Wiley and Sons, 1974.

6. Carnahan, B., Luther, H.A., and Wilkes, J.O, Applied Numerical Methods, John Wiley and Sons, 1969.

APPENDICES

APPENDIX A

FORMULATION OF THE STIFFNESS MATRIX AND LOAD VECTOR FROM THE MINIMUM COMPLEMENTARY ENERGY PRINCIPLE

The stiffness matrix corresponding to an elastic body can be derived in a number of different ways. One way to formulate this matrix is by starting with the complementary energy principle, which corresponds to a compatibility condition. The principle states that the state of stress that satisfies the stress-strain relations in the interior and all prescribed boundary conditions also minimizes the system's complementary energy in an elastic body. If $\pi_c(\sigma_x,\sigma_y,\sigma_z,\tau_{xy},\tau_{xz},\tau_{yz})$ is the complementary energy, $U_c(\sigma_x,\sigma_y,\sigma_z,\tau_{xy},\tau_{xz},\tau_{yz})$ is the complementary stress energy, and V_c is the work done by the applied loads during stress changes, then, according to the minimum complementary energy principle,

$$\delta\pi_c = \delta(U_c - V_c) \tag{A.1}$$

$$= \delta U_c - \delta V_c = 0 \quad . \tag{A.2}$$

To facilitate a finite element formulation, the variation is taken

with respect to the stress components. The complementary stress energy is defined as

$$U_c(\sigma_x,\sigma_y,\sigma_z,\tau_{xy},\tau_{xz},\tau_{yz}) = 1/2\iiint_v \{\sigma\}^T[D]\{\sigma\}dV \qquad (A.3a)$$

where

$$\{\sigma\}^T = \{\sigma_x\ \sigma_y\ \sigma_z\ \tau_{xy}\ \tau_{xz}\ \tau_{yz}\} \qquad (A.3b)$$

in a three dimensional problem. The matrix [D] is the material stiffness matrix that is used to relate the strains to the stresses. In a three dimensional problem it is a 6X6 matrix. In a two dimensional problem it is a 3X3 matrix and has different values depending on if the problem definition is plane strain or plane stress. It is defined by

$$\{\varepsilon\} = [D]\{\sigma\} \qquad (A.3c)$$

where the strain vector

$$\{\varepsilon\}^T = \{\varepsilon_x\ \varepsilon_y\ \varepsilon_z\ \gamma_{xy}\ \gamma_{xz}\ \gamma_{yz}\} \ . \qquad (A.3d)$$

The relationship between the stresses and the strains in equation (A.3c) can also be written as

$$\{\sigma\} = [C]\{\varepsilon\} \qquad (A.3e)$$

where

$$[C] = [D]^{-1} \quad . \tag{A.3f}$$

Including the column vector of initial strains $\{\varepsilon_o\}$ into the complementary stress energy term results in

$$U_c = 1/2\iiint_v \left[\{\sigma\}^T[D]\{\sigma\} - 2\{\sigma\}^T\{\varepsilon_o\} \right] dV \quad . \tag{A.4}$$

The work done by the external forces is

$$V_c = \iiint_v \left[X^* u_o + Y^* v_o + Z^* w_o \right] dV + \iint_s \left[T_x^* u_o + T_y^* v_o + T_z^* w_o \right] dS \tag{A.5a}$$

or written in matrix terms

$$V_c = \iiint_v \{F^*\}^T\{\delta\} dV + \iint_s [T^*]\{\delta\} dS \tag{A.5b}$$

where $\{F^*\}^T$ are the body forces such as gravity, $[T^*]$ are the boundary traction components acting on the surface of the solid, and $\{\delta\}$ is the column matrix of the components of the displacement field. Substituting equations (A.5b) and (A.4) into equation (A.2) results in

$$1/2\iiint_v \left[\{\sigma\}^T[D]\{\sigma\} - 2\{\sigma\}^T\{\varepsilon_o\} \right] dV - \iiint_v \{F^*\}^T\{\delta\} dV - \iint_s [T^*]\{\delta\} dS = \pi_c \tag{A.6}$$

Substituting equation (A.3e) into (A.6) yields

$$1/2\iiint_V\left[[[C]\{\varepsilon\}]^T[D][C]\{\varepsilon\} - 2[[C]\{\varepsilon\}]^T\{\varepsilon_0\}\right]dV$$
$$- \iiint_V\{F^*\}^T\{\delta\}dV - \iint_S[T^*]\{\delta\}dS = \pi_c \quad . \tag{A.7}$$

Now defining the relationship between the strains and the displacements as

$$\{\varepsilon\} = [B]\{\delta\} \tag{A.8}$$

where [B] is a 3NX6 matrix (for a three dimensional solid with N nodes per element) that contains the derivatives of the shape functions. Appendix B discusses the shape functions and their derivatives. Using equation (A.8) in equation (A.7) produces

$$1/2\iiint_V\left[[[B]\{\sigma\}]^T[C]^T[D][C][B]\{\delta\} - 2[[B]\{\delta\}]^T[C]^T\{\varepsilon_0\}\right]dV$$
$$- \iiint_V\{F^*\}\{\delta\}dV - \iint_S[T^*]\{\delta\}dS = \pi_c \quad . \tag{A.9}$$

Since the material properties are isotropic, the material stiffness matrix is symmetric, i.e.,

$$[C] = [C]^T \quad . \tag{A.10}$$

Substituting equations (A.3f) and (A.10) into equation (A.9) will give the following result

$$1/2\iiint_V\left[\{\delta\}^T[B]^T[C][B]\{\delta\} - 2\{\delta\}^T[B]^T[C]\{\varepsilon_0\}\right]dV$$
$$- \iiint_V\{F^*\}^T\{\delta\}dV - \iint_S[T^*]\{\delta\}dS = \pi_c \quad . \tag{A.11}$$

For a single element (e) the discretized functional is

$$\pi^{(e)}(\{\delta\}^{(e)}) = \pi^{(e)}(u_1, u_2, \ldots, u_r, v_1, v_2, \ldots, v_r, w_1, w_2, \ldots, w_r) \qquad (A.12)$$

or in terms of equation (A.11) the functional is

$$\pi^{(e)}(\{\delta\}^{(e)}) = 1/2\iiint_v \Big[\{\delta\}^{T(e)}[B]^{T(e)}[C]^{(e)}\{\delta\}^{(e)} - 2\{\delta\}^{T(e)}[B]^{T(e)}[C]^{(e)}\{\varepsilon_0\}^{(e)}\Big] dV^{(e)} - \iiint_v \{F^*\}^{T(e)}\{\delta\}^{(e)} dV^{(e)} - \iint_s [T^*]^{(e)}\{\delta\}^{(e)} dS^{(e)} \quad . \qquad (A.13)$$

At equilibrium the potential energy of a system is stationary. The discretized system is stationary when the first variation of the discretized functional vanishes, thus

$$\delta\pi(u,v,w) = \sum_{e=1}^{M} \Big[\delta\pi^{(e)}(u,v,w) \Big] = 0 \qquad (A.14a)$$

where

$$\delta\pi^{(e)}(u,v,w) = \sum_{i=1}^{r} \Big[(\partial\pi^{(e)}/\partial u_i)\delta u_i \Big] + \sum_{i=1}^{r} \Big[(\partial\pi^{(e)}/\partial v_i)\delta v_i \Big] + \sum_{i=1}^{r} \Big[(\partial\pi^{(e)}/\partial w_i)\delta w_i \Big] \quad . \qquad (A.14b)$$

The δu_i, δv_i, and the δw_i are independent (they may or may not be zero) therefore, the individial parts of the summation are forced to zero, i.e.,

$$\partial\pi^{(e)}/\partial u_i = \partial\pi^{(e)}/\partial v_i = \partial\pi^{(e)}/\partial w_i = 0 \qquad i=1,2,\ldots,r \qquad (A.15)$$

Since the displacements of the body are approximated by the shape functions, the distributed displacement field in the u direction of an r node element can be expressed as

$$\{u(x,y,z)^{(e)}\} = \{\sum_{i=1}^{r} N_i(x,y,z)u_i\}^{(e)} = \{[N]^T\{u\}\}^{(e)} \quad . \tag{A.16a}$$

The v and w distributed displacements are transformed into discrete displacements in the same manner u is, and the resulting discrete displacement field becomes

$$\{\delta\}^{(e)} = \begin{Bmatrix} u(x,y,z) \\ v(x,y,z) \\ w(x,y,z) \end{Bmatrix}^{(e)} = \begin{Bmatrix} [N^T\{u\}] \\ [N^T\{v\}] \\ [N^T\{w\}] \end{Bmatrix}^{(e)} \tag{A.16b}$$

Using equations (A.14) and (A.16b) equation (A.13) becomes

$$\begin{aligned} \{\partial\pi^{(e)}/\partial\delta\} = \{0\} = &\iiint_V [B]^{T(e)}[C]^{(e)}[B]^{(e)}\{\delta\}^{(e)}dV^{(e)} \\ &- \iiint_V [B]^{T(e)}[C]^{(e)}\{\varepsilon_0\}^{(e)}dV^{(e)} \\ &- \iiint_V [N]\{F^*\}^{(e)}dV^{(e)} - \iint_S [N]\{T^*\}^{(e)}dS^{(e)} \end{aligned} \tag{A.17a}$$

where

$$\{\partial\pi^{(e)}/\partial\delta\}^T = \{\partial\pi^{(e)}/\partial u \quad \partial\pi^{(e)}/\partial v \quad \partial\pi^{(e)}/\partial w\} \quad . \tag{A.17b}$$

Equation (A.17a) is sometimes simplified to the following form

$$[k]^{(e)}\{\delta\}^{(e)} = \{F_0\}^{(e)} + \{F_B\}^{(e)} + \{F_T\}^{(e)} \qquad \text{(A.17c)}$$

where

$[k]^{(e)}$ = stiffness matrix at element e

$\{\delta\}^{(e)}$ = displacement vector at element e

$\{F_0\}^{(e)}$ = initial force vector at element e

$\{F_B\}^{(e)}$ = body force vector at element e

$\{F_T\}^{(e)}$ = surface loading force vector at element e.

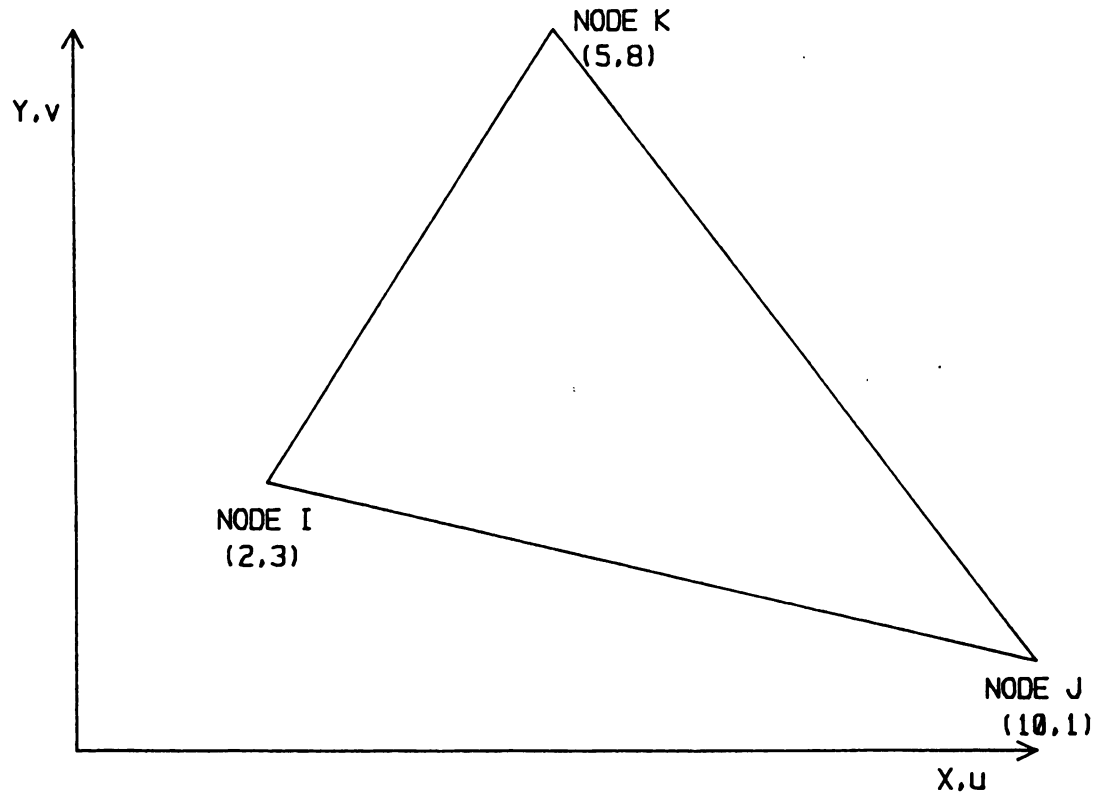


Figure A.1 Two dimensional triangular element

Equation (A.17a) is best illustrated by a two dimensional example shown in Figure A.1. The problem consists of a single triangle

subjected to three loading cases. To formulate the stiffness matrix, the shape functions must first be defined. This is accomplished by using a linear variation of displacements u and v across the element, resulting in two equations.

$$u(x,y) = N_1 u_1 + N_2 u_2 + N_3 u_3 \tag{A.18a}$$

$$v(x,y) = N_1 v_1 + N_2 v_2 + N_3 v_3 \tag{A.18b}$$

In addition to equations (A.18a) and (A.18b), a third condition is imposed, requiring the shape functions to sum to one.

$$N_1 + N_2 + N_3 = 1 \tag{A.18c}$$

Solving equations (A.18a), (A.18b), and (A.18c) for N_1, N_2, and N_3 gives the shape functions in terms of the Cartesian coordinates. They are

$$N_1(x,y) = (1/2\Delta)(a_1 + b_1 x + c_1 y) \tag{A.19a}$$

$$N_2(x,y) = (1/2\Delta)(a_2 + b_2 x + c_2 y) \tag{A.19b}$$

$$N_3(x,y) = (1/2\Delta)(a_3 + b_3 x + c_3 y) \tag{A.19c}$$

where

$$2\Delta = \text{Det}\begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} = 2(\text{Area of triangle } 123) \tag{A.19d}$$

and

$$a_1 = x_2y_3 - x_3y_2, \quad b_1 = y_2 - y_3, \quad c_1 = x_3 - x_2 \tag{A.19e}$$

$$a_2 = x_3y_1 - x_1y_3, \quad b_2 = y_3 - y_1, \quad c_2 = x_1 - x_3 \tag{A.19f}$$

$$a_3 = x_1y_2 - x_2y_1, \quad b_3 = y_1 - y_2, \quad c_3 = x_2 - x_1 \ . \tag{A.19g}$$

Applying the (A.19) equations to the triangle in Figure A.1 results in the following shape functions:

$$N_1 = (1/46)(75 - 7x - 5y) \tag{A.20a}$$

$$N_2 = (1/46)(-1 + 5x - 3y) \tag{A.20b}$$

$$N_3 = (1/46)(-28 + 2x + 8y) \tag{A.20c}$$

The matrix relating the strains to the displacements from equation (A.8) is formed using

$$[B] = \begin{bmatrix} \partial N_1/\partial x & 0 & \partial N_2/\partial x & 0 & \partial N_3/\partial x & 0 \\ 0 & \partial N_1/\partial y & 0 & \partial N_2/\partial y & 0 & \partial N_3/\partial y \\ \partial N_1/\partial y & \partial N_1/\partial x & \partial N_2/\partial y & \partial N_2/\partial x & \partial N_3/\partial y & \partial N_3/\partial x \end{bmatrix} \tag{A.21a}$$

which is

$$[B] = (1/2\Delta) \begin{bmatrix} b_1 & 0 & b_2 & 0 & b_3 & 0 \\ 0 & c_1 & 0 & c_2 & 0 & c_3 \\ c_1 & b_1 & c_2 & b_2 & c_3 & b_3 \end{bmatrix} \tag{A.21b}$$

Applying equation (A.21b) to the element shown in Figure A.1 produces the following [B] matrix

$$[B] = (1/46)\begin{bmatrix} -7 & 0 & 5 & 0 & 2 & 0 \\ 0 & -5 & 0 & -3 & 0 & 8 \\ -5 & -7 & -3 & 5 & 8 & 2 \end{bmatrix} . \qquad \text{(A.21c)}$$

If the element is considered in plane strain, then the material matrix [C] is

$$[C] = E/\{(1+\mu)(1-2\mu)\}\begin{bmatrix} 1-\mu & \mu & 0 \\ \mu & 1-\mu & 0 \\ 0 & 0 & (1-2\mu)/2 \end{bmatrix} \qquad \text{(A.22a)}$$

where E is Young's modulas and μ is Poisson's ratio. If E=1000 and μ=.25 then the material matrix in equation (A.22a) becomes

$$[C] = \begin{bmatrix} 1200 & 400 & 0 \\ 400 & 1200 & 0 \\ 0 & 0 & 400 \end{bmatrix} . \qquad \text{(A.22b)}$$

The stiffness matrix from equation (A.17a) is

$$[k]^{(e)} = \iiint_V [B]^{T(e)}[C]^{(e)}[B]^{(e)} dV^{(e)} \qquad \text{(A.23a)}$$

but since neither the [B] matrix or the [C] matrix in this case are

functions of the volume, the element stiffness matrix can be rewritten as

$$[k]^{(e)} = [B]^{T(e)}[C]^{(e)}[B]^{(e)}\iiint dV$$
$$= [B]^{T(e)}[C]^{(e)}[B]^{(e)}t^{(e)}A^{(e)} \qquad \text{(A.23b)}$$

where $A^{(e)}$ is the element area and $t^{(e)}$ is the element thickness, which usually assumes a value of one for plane strain problems. Using the matrices in equations (A.21c) and (A22.b) in equation (A.23b) and knowing the area of the element is 23 units, the element stiffness matrix for the element shown in Figure A.1 becomes

$$[k]^{(e)} = \begin{bmatrix} 747.8 & 304.4 & -391.3 & -17.4 & -356.5 & -287.0 \\ 304.4 & 539.1 & -17.4 & 43.5 & -287.0 & -582.6 \\ -391.3 & -17.4 & 365.2 & -130.4 & 26.1 & 147.8 \\ -17.4 & 43.5 & -130.4 & 226.1 & 147.8 & -269.6 \\ -356.5 & -287.0 & 26.1 & 147.8 & 330.4 & 139.1 \\ -287.0 & -582.6 & 147.8 & -269.6 & 139.1 & 852.2 \end{bmatrix} . \qquad \text{(A.23c)}$$

The global stiffness matrix of a structure is the sum of the element stiffness matrices, and is found by

$$[K] = \sum_{i=1}^{n} [k]^{(e)} \qquad \text{(A.24)}$$

where $[k]^{(e)}$ is summed with respect to the node numbers. The initial force vector at element (e) is (from equation (A.17a)

$$\{F_0\}^{(e)} = \iiint_V [B]^{T(e)}[C]^{(e)}\{\varepsilon_0\}^{(e)}dV^{(e)} \quad . \tag{A.25a}$$

For this element, [B] and [C] are not functions of the volume, and since the column vector of initial strains is rarely a function of the volume, equation (A.25a) can be expressed as

$$\begin{aligned}\{F_0\}^{(e)} &= [B]^{T(e)}[C]^{(e)}\{\varepsilon_0\}^{(e)}\iiint dV \\ &= [B]^{T(e)}[C]^{(e)}\{\varepsilon_0\}^{(e)}t^{(e)}A^{(e)} \quad .\end{aligned} \tag{A.25b}$$

Using an initial element strain of

$$\begin{aligned}\{\varepsilon_0\}^T &= \{\varepsilon_{x0} \quad \varepsilon_{y0} \quad \gamma_{xy0}\} \\ &= \{.005 \quad .004 \quad .001\}\end{aligned} \tag{A.25c}$$

and the values of [B] and [C] as defined in equations (A.21b) and (A.22b), the initial force vector at element (e) is

$$\{F_0\}^{T(e)} = \{-28.6 \quad -19.8 \quad 17.8 \quad -8.2 \quad 10.8 \quad 28.0\} \quad . \tag{A.25d}$$

The second type of loading that the element can undergo is body force loading. This type of loading is usually gravity. The body force term from equation (A.17a) is

$$\{F_B\}^{(e)} = \iiint_V [N]\{F^*\}^{(e)}dV^{(e)} \tag{A.26a}$$

where

$$[N] = \begin{bmatrix} N_1 & 0 \\ 0 & N_1 \\ N_2 & 0 \\ 0 & N_2 \\ N_3 & 0 \\ 0 & N_3 \end{bmatrix} \qquad \text{(A.26b)}$$

and

$$\{F^*\}^T = \{X^* \quad Y^*\} \quad . \qquad \text{(A.26c)}$$

If the element shown in Figure A.1 has a density of $\rho = .0075$ and gravity $(g = -386.4)$ is acting in the y direction, then the body force is

$$\begin{aligned} \{F^*\}^T &= \{0 \qquad \rho t^{(e)} A^{(e)} g\} \\ &= \{0 \qquad -66.7\} \quad . \end{aligned} \qquad \text{(A.26d)}$$

Equation (A.26a) becomes

$$\{F_B\}^{(e)} = \iint_a \{0 \; N_1(-66.7) \; 0 \; N_2(-66.7) \; 0 \; N_3(-66.7) \; 0\}^T t^{(e)} dA^{(e)} \qquad \text{(A.26e)}$$

and transforming the shape functions back into natural coordinates the integration for a typical entry of (A.26e) is

$$\iint_a N_i C_e t^{(e)} dA^{(e)} = t^{(e)} A^{(e)} C_e / 3 \quad . \qquad \text{(A.26d)}$$

Applying the equation (A.26d) to all the entries of equation (A.26e) results in the following body force vector

$$\{F_B\}^{T(e)} = \{\ 0 \quad -22.2 \quad 0 \quad -22.2 \quad 0 \quad -22.2\}\ . \tag{A.26e}$$

The third force vector to be evaluated is the surface loading force vector which is caused by boundary traction. The boundary traction vector from equation (A.17a) is

$$\{F_T\}^{(e)} = \iint_s [N]\{T^*\}^{(e)} dS\{(e)\}\ . \tag{A.27a}$$

The most common boundary traction term is a point load. If point 2 on the element is loaded with point loads of

$$\{T^*\} = \{-20. \quad 15.\}\ . \tag{A.27b}$$

Then the matrix [N] becomes

$$[N] = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{A.27c}$$

since the shape functions, by definition, have the following property

$$N_i = \begin{cases} 1 \text{ at node i} \\ \\ 0 \text{ at node j} \end{cases} \qquad \text{(A.27d)}$$

the surface loading force vector will be the point loads, i.e,

$$\{F_T\}^{T(e)} = \{0 \quad 0 \quad -20. \quad 15. \quad 0 \quad 0\} \quad . \qquad \text{(A.27e)}$$

If the boundary traction term is a pressure, then it is represented by equivalent point loads on the nodes which share the face the pressure is acting on. Once all of the force terms have been calculated for the element, they are added together.

$$\{F\}^{(e)} = \{F_e\}^{(e)} + \{F_B\}^{(e)} + \{F_T\}^{(e)} \qquad \text{(A.28a)}$$

A global force vector is constructed in the same manner as the global stiffness matrix is, thus

$$\{F\} = \sum_{i=1}^{m} \{F\}^{(e)} \quad . \qquad \text{(A.28b)}$$

In the global force vector, as in the global stiffness matrix, the element vectors are summed with respect to the nodes. The final equation relating the global stiffness matrix and the global force vector is

$$[K]\{\delta\} = \{F\} \qquad \text{(A.29)}$$

where $\{\delta\}$ is the column vector of all displacements of the system.

APPENDIX B

SHAPE FUNCTIONS OF SOLID ISOPARAMETRIC ELEMENTS

The shape functions for all solid isoparametric elements are derived using the same method. For any m node isoparametric solid element, the displacement in any axis direction is approximated by

$$a \simeq \tilde{a} = \sum_{i=1}^{m} N_i a_i = \{N\}^T \{a\} \tag{B.1}$$

where a = actual displacement value

$\tilde{a}$ = approximated displacement value

N_i = shape function of the ith node

a_i = displacement of the ith node in the direction of interest.

The functions N_i, N_j,..... and N_m are chosen to give the appropriate nodal displacements when the coordinates of the corresponding nodes are inserted into equation (B.1). For a three dimensional element, the shape function will have the following property:

$$N_i(r_j, s_j, t_j) = \begin{cases} 1 & \text{for i equal to j} \\ 0 & \text{for i not equal to j.} \end{cases} \tag{B.2a}$$

Furthermore, the shape functions will always sum to unity at any location in the element,

$$N_i(r,s,t) + N_j(r,s,t) + ... + N_m(r,s,t) = 1 \quad . \tag{B.2b}$$

One way to derive the shape function, is by assuming the variation of the shape function is dependent on r, s, and t,

$$N_j(r,s,t) = \sum_{i=1}^{m} b_i r^c s^d t^e \quad . \tag{B.2c}$$

The application of equations (B.2) is best illustrated by the element shown in Figure B.1.

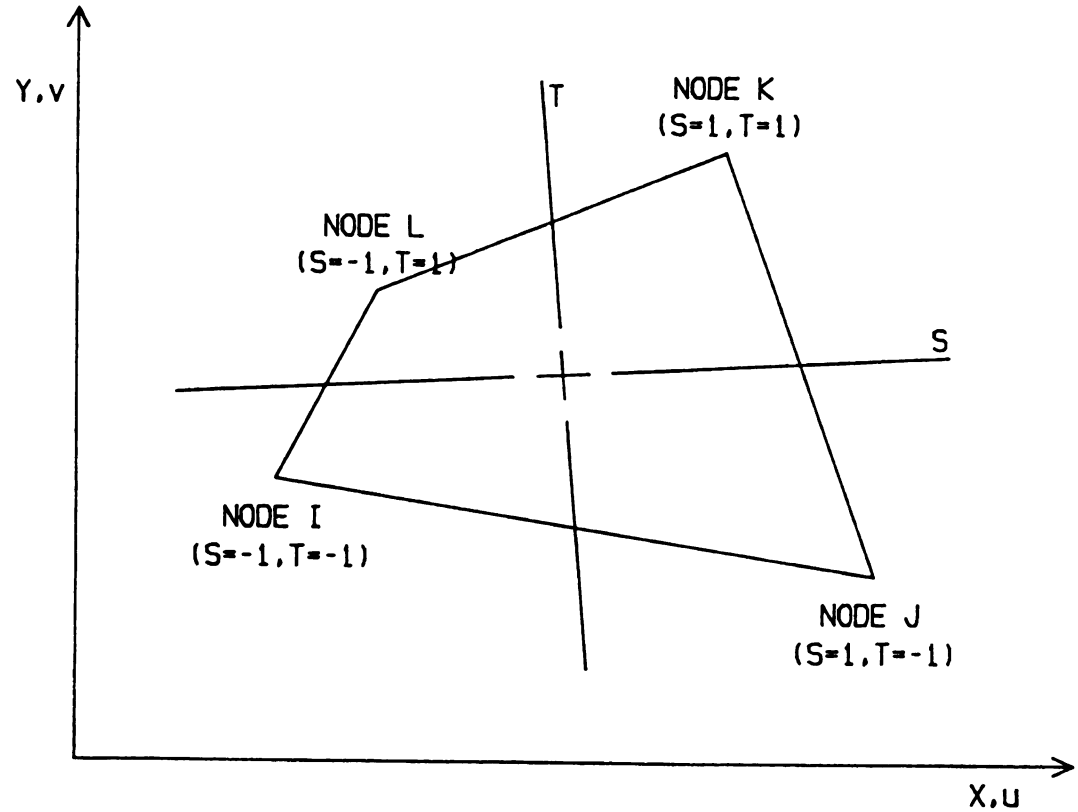


Figure B.1 Four node two dimensional solid element

This element consists of a four node two dimensional solid orientated in it's natural coordinates (all of the examples in this appendix are located in natural coordinates, the transformation back into spatial coordinates will be discussed later). Equation (B.2c) applied to Figure B.1 will produce four equations. Using the smallest values of d and e possible, the equations are

$$N_I = b_1 + b_2 s + b_3 t + b_4 st \qquad (B.3a)$$

$$N_J = b_1 + b_2 s + b_3 t + b_4 st \qquad (B.3b)$$

$$N_K = b_1 + b_2 s + b_3 t + b_4 st \qquad (B.3c)$$

$$N_L = b_1 + b_2 s + b_3 t + b_4 st \qquad (B.3d)$$

and since the nodal values in natural coordinates are either 1 or -1, the values of b_1 thru b_4 can be solved for using equation (B.2a). These values will not necessarily be the same for the different (B.3) equations. Solving for the b's and rearranging the terms, the shape functions become

$$N_I = (1/4)(1-s)(1-t) \qquad (B.3e)$$

$$N_J = (1/4)(1+s)(1-t) \qquad (B.3f)$$

$$N_K = (1/4)(1+s)(1+t) \qquad (B.3g)$$

$$N_L = (1/4)(1-s)(1+t) \; . \qquad (B.3h)$$

A typical shape function for this element is shown graphically in Figure B.2. The next shape function to be considered is an eight node two dimensional isoparametric solid as shown in Figure B.3. The shape

functions are formulated in the same way as before, only this element has eight of them.

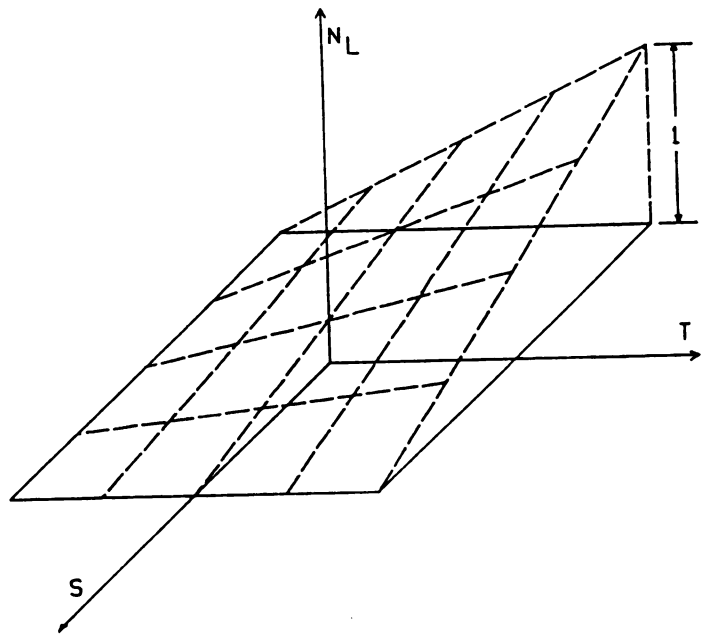


Figure B.2 Shape function of a four node element

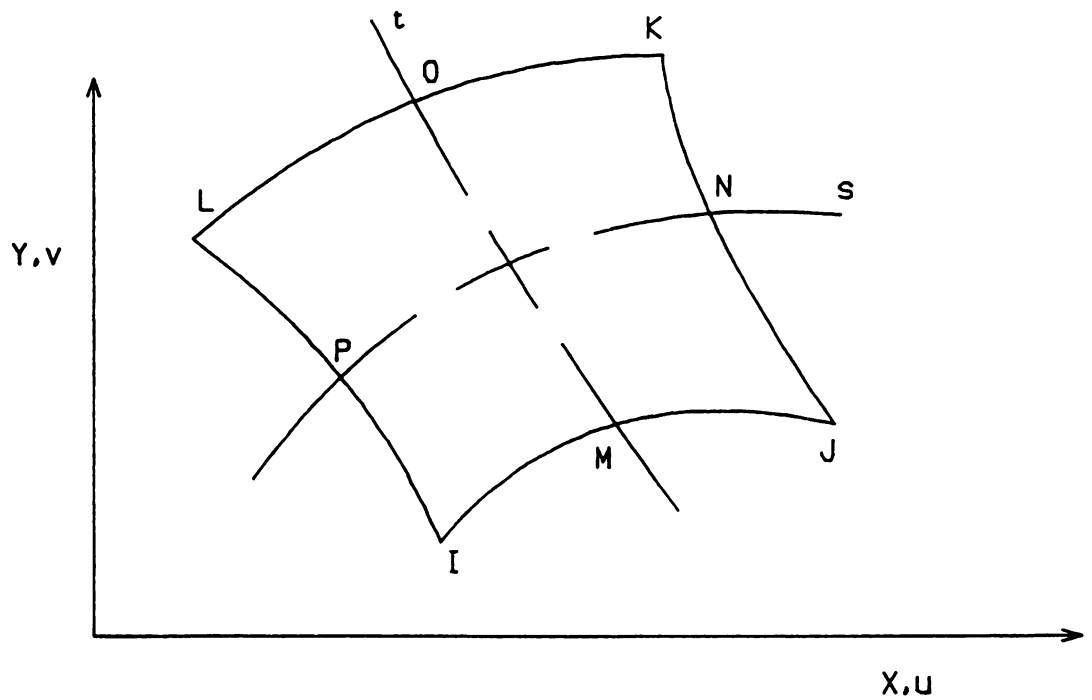


Figure B.3 Eight node two dimensional solid element

The shape functions for the eight node solid are

$$N_I = (1/4)(1-s)(1-t)(-s-t-1) \qquad \text{(B.4a)}$$

$$N_J = (1/4)(1+s)(1-t)(\ s-t-1) \qquad \text{(B.4b)}$$

$$N_K = (1/4)(1+s)(1+t)(\ s+t-1) \qquad \text{(B.4c)}$$

$$N_L = (1/4)(1-s)(1+t)(-s+t-1) \qquad \text{(B.4d)}$$

$$N_M = (1/2)(1-s^2)(1-t) \qquad \text{(B.4e)}$$

$$N_N = (1/2)(1+s)(1-t^2) \qquad \text{(B.4f)}$$

$$N_O = (1/2)(1-s^2)(1+t) \qquad \text{(B.4g)}$$

$$N_P = (1/2)(1-s)(1-t^2) \quad . \qquad \text{(B.4h)}$$

A typical shape function for this element is shown graphically in Figure B.4.

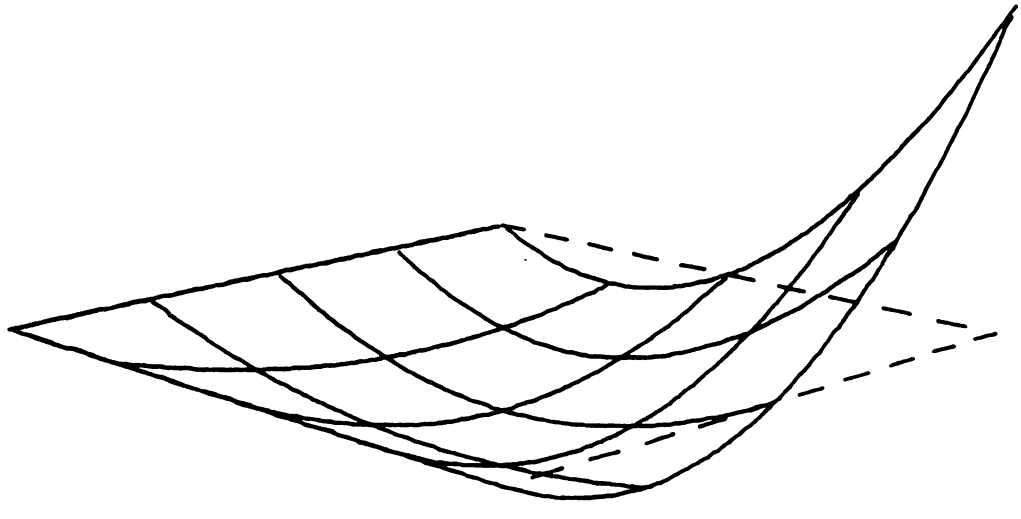


Figure B.4 Shape function for an eight node element

The three dimensional theory for the shape functions is a continuation of the two dimensional theory. Another variable is added to the shape function formulation. An eight node, three dimensional solid is shown in Figure B.5.

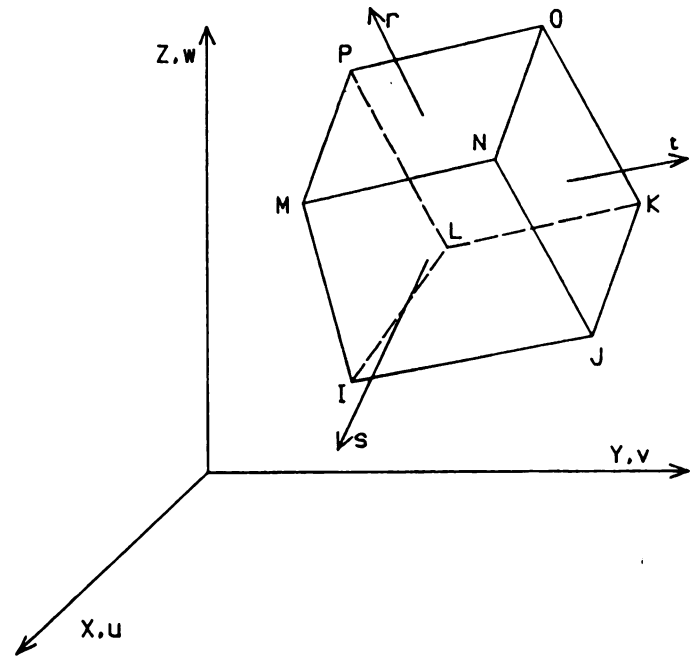


Figure B.5 Eight node three dimensional solid element

The shape functions for the three dimensional eight node solid element are

$$N_I = (1/8)(1-s)(1-t)(1-r) \qquad \text{(B.5a)}$$

$$N_J = (1/8)(1+s)(1-t)(1-r) \qquad \text{(B.5b)}$$

$$N_K = (1/8)(1+s)(1+t)(1-r) \qquad \text{(B.5c)}$$

$$N_L = (1/8)(1-s)(1+t)(1-r) \qquad \text{(B.5d)}$$

$$N_M = (1/8)(1-s)(1-t)(1+r) \qquad \text{(B.5e)}$$

$$N_N = (1/8)(1+s)(1-t)(1+r) \qquad \text{(B.5f)}$$

$$N_O = (1/8)(1+s)(1+t)(1+r) \qquad \text{(B.5g)}$$

$$N_P = (1/8)(1-s)(1+t)(1+r) \quad . \qquad \text{(B.5h)}$$

A twenty node, three dimensional solid is shown in Figure B.6.

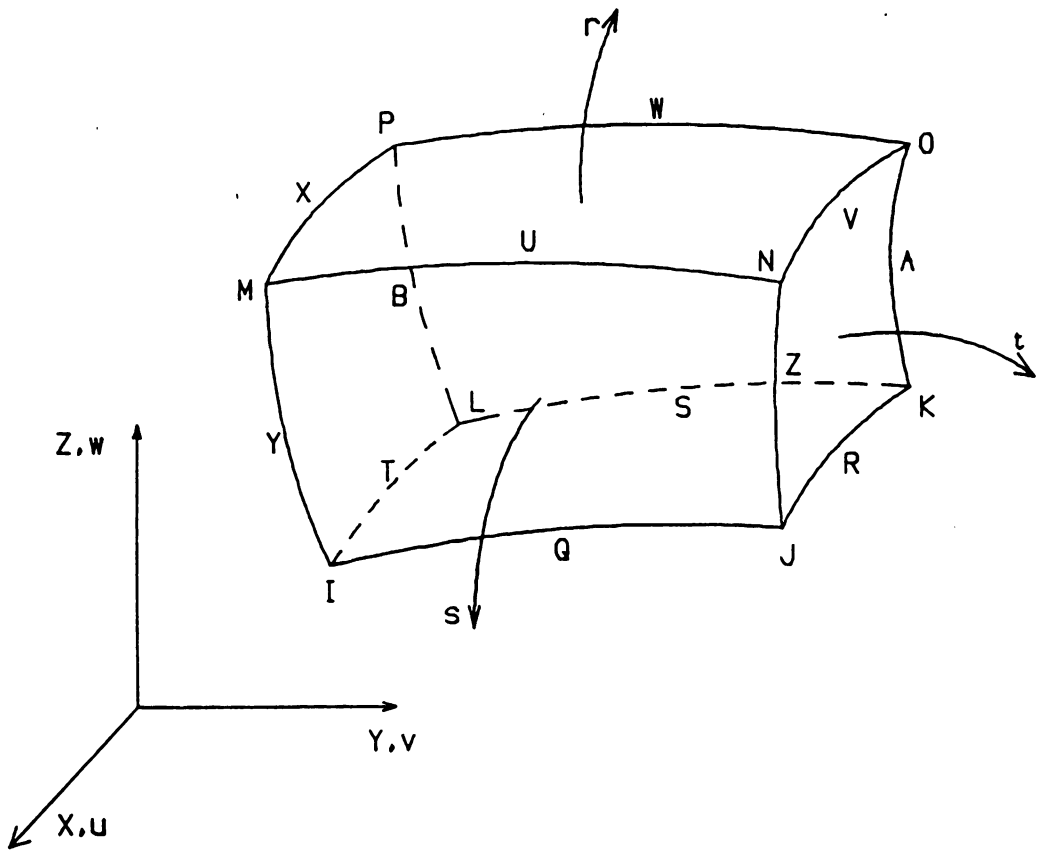


Figure B.6 Twenty node three dimensional solid element

The shape functions for the twenty node three dimensional solid element are

$$N_I = (1/8)(1-s)(1-t)(1-r)(-s-t-r-2) \qquad \text{(B.6a)}$$

$$N_J = (1/8)(1+s)(1-t)(1-r)(\ s-t-r-2) \qquad \text{(B.6b)}$$

$$N_K = (1/8)(1+s)(1+t)(1-r)(\ s+t-r-2) \qquad \text{(B.6c)}$$

$$N_L = (1/8)(1-s)(1+t)(1-r)(-s+t-r-2) \quad \text{(B.6d)}$$

$$N_M = (1/8)(1-s)(1-t)(1+r)(-s-t+r-2) \quad \text{(B.6e)}$$

$$N_N = (1/8)(1+s)(1-t)(1+r)(\ s-t+r-2) \quad \text{(B.6f)}$$

$$N_O = (1/8)(1+s)(1+t)(1+r)(\ s+t+r-2) \quad \text{(B.6g)}$$

$$N_P = (1/8)(1-s)(1+t)(1+r)(-s+t+r-2) \quad \text{(B.6h)}$$

$$N_Q = (1/4)(1-s^2)(1-t)(1-r) \quad \text{(B.6i)}$$

$$N_R = (1/4)(1+s)(1-t^2)(1-r) \quad \text{(B.6k)}$$

$$N_S = (1/4)(1-s^2)(1+t)(1-r) \quad \text{(B.6l)}$$

$$N_T = (1/4)(1-s)(1-t^2)(1-r) \quad \text{(B.6m)}$$

$$N_U = (1/4)(1-s^2)(1-t)(1+r) \quad \text{(B.6n)}$$

$$N_V = (1/4)(1+s)(1-t^2)(1+r) \quad \text{(B.6o)}$$

$$N_W = (1/4)(1-s^2)(1+t)(1+r) \quad \text{(B.6p)}$$

$$N_X = (1/4)(1-s)(1-t^2)(1+r) \quad \text{(B.6q)}$$

$$N_Y = (1/4)(1-s)(1-t)(1-r^2) \quad \text{(B.6r)}$$

$$N_Z = (1/4)(1+s)(1-t)(1-r^2) \quad \text{(B.6s)}$$

$$N_A = (1/4)(1+s)(1+t)(1-r^2) \quad \text{(B.6t)}$$

$$N_B = (1/4)(1-s)(1+t)(1-r^2) \quad . \quad \text{(B.6u)}$$

Condensing the above elements to new elements with less nodes is based on the following equation

$$\alpha_m = (1/2)(\alpha_{c1} + \alpha_{c2}) \quad \text{(B.7a)}$$

where α represents r, s, t, u, v, or w, the subscript m denotes a midside node, and the subscripts c1 and c2 denote the nodes that are collinear with node m. An eight node solid condensed into a five node

solid is shown in Figure B.7.

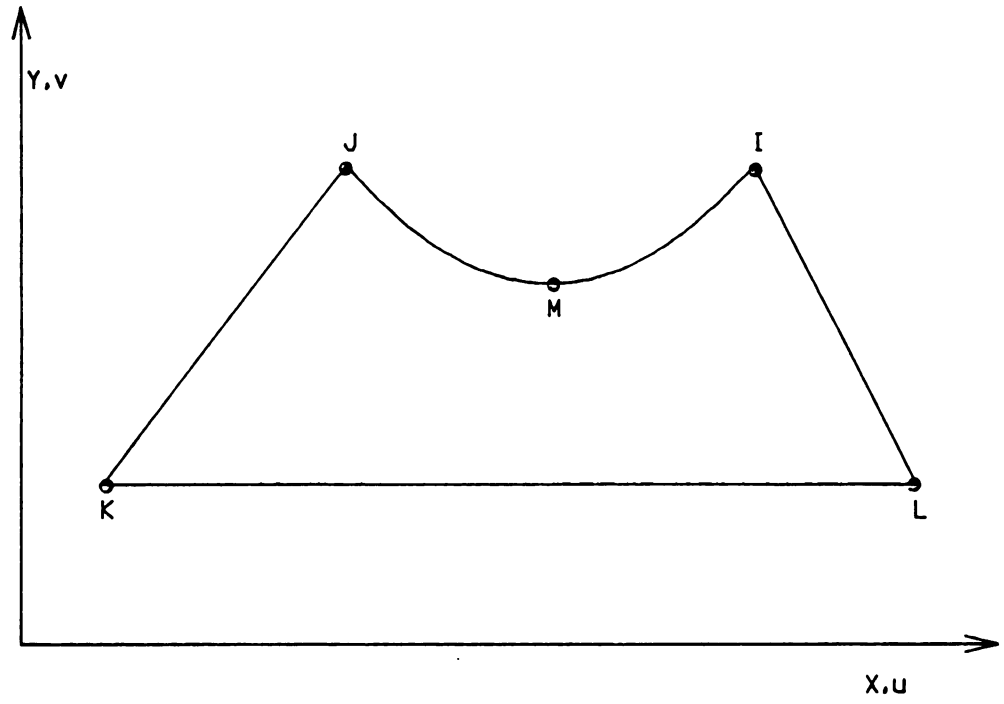


Figure B.7 Five node two dimensional solid element

The displacement in the u direction for the eight node solid is

$$u = N_I u_I + N_J u_J + N_K u_K + N_L u_L + N_M u_M + N_N u_N + N_O u_O + N_P u_P \quad . \qquad (B.7b)$$

The five node element has three condensed sides, which will produce three equations from equation (B.7a) in the u direction.

$$u_M = (1/2)(u_J + u_K) \qquad (B.7c)$$

$$u_O = (1/2)(u_K + u_L) \qquad (B.7d)$$

$$u_P = (1/2)(u_L + u_I) \qquad (B.7e)$$

Inserting equations (B.7c), (B.7d), and (B.7e) into equation (B.7b) and rearranging the terms yields

$$u = \left[N_I + (N_P/2)\right]u_I + \left[N_J + (N_N/2)\right]u_J + \left[N_K + (N_N/2) + (N_O/2)\right]u_K + \left[N_L + (N_O/2) + (N_P/2)\right]u_L + N_M u_M \quad \text{(B.7f)}$$

and the shape functions for the five node element becomes

$$N'_I = N_I + (N_P/2) \quad \text{(B.7g)}$$

$$N'_J = N_J + (N_N/2) \quad \text{(B.7h)}$$

$$N'_K = N_K + (N_N/2) + (N_O/2) \quad \text{(B.7i)}$$

$$N'_L = N_L + (N_O/2) + (N_P/2) \quad \text{(B.7j)}$$

$$N'_M = N_M \ . \quad \text{(B.7k)}$$

The new shape functions derived in the v direction are exactly the same.

If the displacements within an element are known, the strains at any point can be determined. The displacements and strains are related by the equation

$$\{\varepsilon\} = [L]\{a\} \quad \text{(B.8a)}$$

where [L] is a linear operator. Since the displacements are approximated by equation (B.1), equation (B.8a) becomes

$$\{\varepsilon\} \simeq [L][N]\{\tilde{a}\} \quad \text{(B.8b)}$$

where

$$[B] = [L][N] \quad . \tag{B.8c}$$

The entries of matrix [B] are called the strain shape functions. The linear operator [L] varies depending on if the problem is two or three dimensional. In a two dimensional problem, [L] is defined as

$$[L] = \begin{bmatrix} \partial/\partial s & 0 \\ 0 & \partial/\partial t \\ \partial/\partial t & \partial/\partial s \end{bmatrix} \quad . \tag{B.8d}$$

At node i, equation (B.8c) becomes

$$[B_i] = [L]N_i \tag{B.8e}$$

or

$$[B_i] = \begin{bmatrix} \partial N_i/\partial s & 0 \\ 0 & \partial N_i/\partial t \\ \partial N_i/\partial t & \partial N_i/\partial s \end{bmatrix} \quad . \tag{B.8f}$$

For a p node element, the [B] matrix is

$$[B] = \begin{bmatrix} [B_i] & [B_j] & .... & [B_p] \end{bmatrix} \quad . \tag{B.8g}$$

If the finite element formulation is a three dimensional problem, the linear operator [L] is

$$[L] = \begin{bmatrix} \partial/\partial r & 0 & 0 \\ 0 & \partial/\partial s & 0 \\ 0 & 0 & \partial/\partial t \\ \partial/\partial r & \partial/\partial s & 0 \\ 0 & \partial/\partial s & \partial/\partial t \\ \partial/\partial r & 0 & \partial/\partial t \end{bmatrix} . \qquad \text{(B.8h)}$$

At node i, the strain shape function matrix is

$$[B_i] = \begin{bmatrix} \partial N_i/\partial r & 0 & 0 \\ 0 & \partial N_i/\partial s & 0 \\ 0 & 0 & \partial N_i/\partial t \\ \partial N_i/\partial s & \partial N_i/\partial r & 0 \\ 0 & \partial N_i/\partial t & \partial N_i/\partial s \\ \partial N_i/\partial t & 0 & \partial N_i/\partial r \end{bmatrix} . \qquad \text{(B.8i)}$$

For a three dimensional problem, the element [B] matrix assembles according to equation (B.8g).

The above elements have all been defined in natural coordinates. They are frequently called parent elements. Transformation of all isoparametric elements from spatial domain to natural domain occur using the shape functions, i.e., the shape functions defining displacements will also be the same function used to map the element

from a spatial coordinate system to a natural coordinate system. The element shown in Figure B.8 is mapped into the parent element.

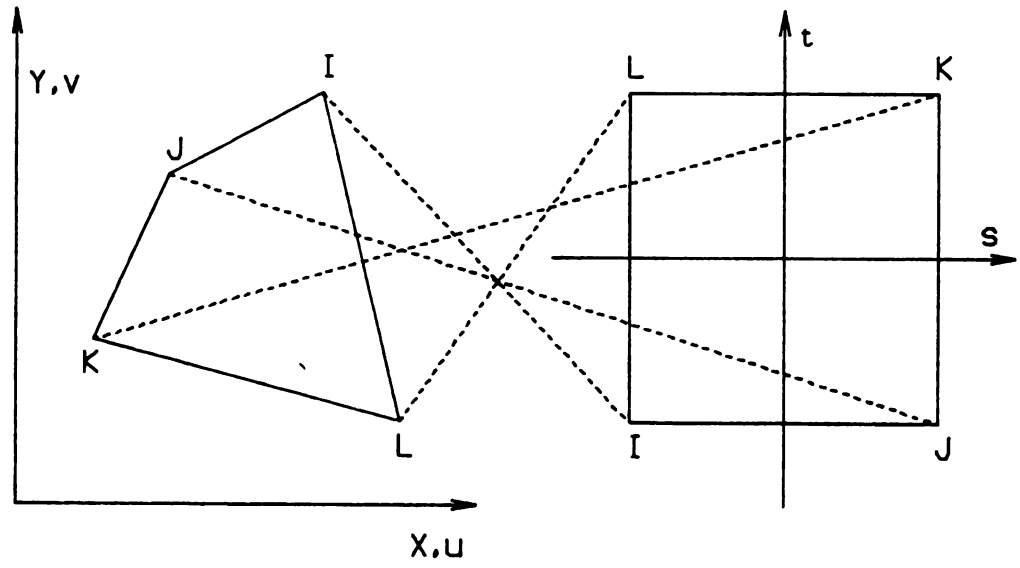


Figure B.8 Four node element mapped into natural coordinates

This usually consists of a change of area, and a rotation. The equations defining this mapping are

$$X = a_1 + a_2 s + a_3 t + a_4 st \qquad \text{(B.9a)}$$

$$Y = b_1 + b_2 s + b_3 t + b_4 st \qquad \text{(B.9b)}$$

with the boundary conditions of

$$X(s,t) = X(-1,-1) = x_I = 4$$

$$= X(\ 1,-1) = x_J = 2$$

$$= X(\ 1,\ 1) = x_K = 1$$

$$= X(-1,\ 1) = x_L = 5$$

$$Y(s,t) = Y(-1,-1) = y_I = 5$$

$$= Y(\ 1,-1) = y_J = 4$$

$$= Y(\ 1,\ 1) = y_K = 2$$

$$= Y(-1,\ 1) = y_L = 1 \quad .$$

Solving equations (B.9a) and (B.9b) results in

$$a_1 = (1/4)(\ x_I + x_J + x_K + x_L) = 12/4 \qquad \text{(B.9c)}$$

$$a_2 = (1/4)(-x_I + x_J + x_K - x_L) = -6/4 \qquad \text{(B.9d)}$$

$$a_3 = (1/4)(-x_I - x_J + x_K + x_L) = \ \ 0 \qquad \text{(B.9e)}$$

$$a_4 = (1/4)(\ x_I - x_J + x_K - x_L) = -2/4 \qquad \text{(B.9f)}$$

$$b_1 = (1/4)(\ y_I + y_J + y_K + y_L) = 12/4 \qquad \text{(B.9g)}$$

$$b_2 = (1/4)(-y_I + y_J + y_K - y_L) = \ \ 0 \qquad \text{(B.9h)}$$

$$b_3 = (1/4)(-y_I - y_J + y_K + y_L) = -6/4 \qquad \text{(B.9i)}$$

$$b_4 = (1/4)(\ y_I - y_J + y_K - y_L) = \ 2/4 \qquad \text{(B.9j)}$$

or

$$X(s,t) = (1/4)(12 - 6s - 2st) \qquad \text{(B.9k)}$$

$$Y(s,t) = (1/4)(12 - 6t + 2st) \quad . \qquad \text{(B.9l)}$$

Once the mapping has been defined, the relationship between the strain shape function in natural coordinates and the strain shape function in

spatial coordinates can be established. Using the rules of partial differentiation, the strain shape functions at node β become

$$\partial N_\beta/\partial s = (\partial N_\beta/\partial x)(\partial x/\partial s) + (\partial N_\beta/\partial y)(\partial y/\partial s) \tag{B.10a}$$

$$\partial N_\beta/\partial t = (\partial N_\beta/\partial x)(\partial x/\partial t) + (\partial N_\beta/\partial y)(\partial y/\partial t) \tag{B.10b}$$

or in matrix form

$$\begin{Bmatrix} \partial N_\beta/\partial s \\ \partial N_\beta/\partial t \end{Bmatrix} = \begin{bmatrix} \partial x/\partial s & \partial y/\partial s \\ \partial x/\partial t & \partial y/\partial t \end{bmatrix} \begin{Bmatrix} \partial N_\beta/\partial x \\ \partial N_\beta/\partial y \end{Bmatrix} . \tag{B.10c}$$

The above matrix equation is also written as

$$\begin{Bmatrix} \partial N_\beta/\partial s \\ \\ \partial N_\beta/\partial t \end{Bmatrix} = \left[J \right] \begin{Bmatrix} \partial N_\beta/\partial x \\ \\ \partial N_\beta/\partial y \end{Bmatrix} \tag{B.10d}$$

where [J] is the Jacobian matrix. To find the spatial derivatives, the Jacobian matrix is inverted,

$$\begin{Bmatrix} \partial N_\beta/\partial x \\ \\ \partial N_\beta/\partial y \end{Bmatrix} = \left[J \right]^{-1} \begin{Bmatrix} \partial N_\beta/\partial s \\ \\ \partial N_\beta/\partial t \end{Bmatrix} . \tag{B.10e}$$

The mapping in three dimensional problems is similar to two dimensional problems, equation (B.10c) becomes

$$\begin{Bmatrix} \partial N_\beta/\partial s \\ \partial N_\beta/\partial t \\ \partial N_\beta/\partial r \end{Bmatrix} = \begin{bmatrix} \partial x/\partial s & \partial y/\partial s & \partial z/\partial s \\ \partial x/\partial t & \partial y/\partial t & \partial z/\partial t \\ \partial x/\partial r & \partial y/\partial r & \partial z/\partial r \end{bmatrix} \begin{Bmatrix} \partial N_\beta/\partial x \\ \partial N_\beta/\partial y \\ \partial N_\beta/\partial z \end{Bmatrix} \qquad \text{(B.10f)}$$

or

$$\begin{Bmatrix} \partial N_\beta/\partial s \\ \partial N_\beta/\partial t \\ \partial N_\beta/\partial r \end{Bmatrix} = \begin{bmatrix} J \end{bmatrix} \begin{Bmatrix} \partial N_\beta/\partial x \\ \partial N_\beta/\partial y \\ \partial N_\beta/\partial z \end{Bmatrix}. \qquad \text{(B.10g)}$$

The Jacobian matrix is inverted as in equation (B.10e), and the result is

$$\begin{Bmatrix} \partial N_\beta/\partial x \\ \partial N_\beta/\partial y \\ \partial N_\beta/\partial z \end{Bmatrix} = \begin{bmatrix} J \end{bmatrix}^{-1} \begin{Bmatrix} \partial N_\beta/\partial s \\ \partial N_\beta/\partial t \\ \partial N_\beta/\partial r \end{Bmatrix} \qquad \text{(B.10h)}$$

where the Jacobian matrix (and it's inverse) are 3X3 in size. With the transformation from spatial coordinates to natural coordinates defined, the element stiffness matrices can be transformed into natural coordinates, the benefit being easier numerical integration of the element stiffness matrix.

APPENDIX C

NUMERICAL INTEGRATION USING THE GAUSS-LEGENDRE QUADRATURE METHOD

The stiffness matrix in most finite element routines is numerically integrated by a technique called the Gauss-Legendre Quadrature method. This method estimates the value of the integral f(x) by approximating the actual function with an nth-degree interpolating polynomial $p_n(x)$ and integrating.

$$\int_a^b f(x)dx = \int_a^b p_n(x)dx + \int_a^b E_n(x)dx \qquad (C.1)$$

The error term is $E_n(x)$ for the numerical integration. The interpolating polynomial is of Lagrangian form, i.e,

$$p_n(x) = f[x_0] + (x-x_0)f[x_1,x_0] + (x-x_0)(x-x_1)f[x_2,x_1,x_0] + \ldots\ldots + (x-x_0)(x-x_1)\ldots\ldots(x-x_{n-1})f[x_n,\ldots,x_1,x_0] \qquad (C.2a)$$

or, expanding the first three terms

$$p_n(x) = f(x_0) + \left[(x-x_0)/(x_0-x_1)\right]f(x_0) + \left[(x-x_0)/(x_1-x_0)\right]f(x_1) + \left[(x-x_0)(x-x_1)/(x_0-x_1)(x_0-x_2)\right]f(x_0) + \left[(x-x_0)(x-x_1)/(x_1-x_0)(x_1-x_2)\right]f(x_1)$$

$$+ \left[(x-x_0)(x-x_1)/(x_2-x_0)(x_2-x_1)\right]f(x_2) + \ldots\ldots\ldots \qquad (C.2b)$$

The terms in equation (C.2b) can be collected and the equation can be rewritten as

$$\begin{aligned} p_n(x) = &\left[\prod_{j=0,j\neq 0}^{n}[(x-x_j)/(x_0-x_j)]\right]f(x_0) \\ &+ \left[\prod_{j=0,j\neq 1}^{n}[(x-x_j)/(x_1-x_j)]\right]f(x_1) \\ &+ \ldots\ldots + \left[\prod_{j=0,j\neq n}^{n}[(x-x_j)/(x_n-x_j)]\right]f(x_n) \quad . \end{aligned} \qquad (C.2c)$$

Equation (C.2c) is sometimes compressed to

$$p_n(x) = \sum_{i=0}^{n}\left[\prod_{j=0,j\neq i}^{n}[(x-x_j)/(x_i-x_j)]\right]f(x_i) \qquad (C.2e)$$

or

$$p_n(x) = \sum_{i=0}^{n}L_i(x)f(x_i) \qquad (C.2f)$$

where

$$L_i(x) = \prod_{j=0,j\neq i}^{n}[(x-x_j)/(x_i-x_j)] \quad . \qquad (C.2g)$$

The error term from equation (C.1) is

$$E_n(x) = \left[\prod_{i=0}^{n}(x-x_i)\right]f[x,x_n,x_{n-1},\ldots,x_0] \qquad (C.3a)$$

or

$$E_n(x) = \left[\prod_{i=0}^{n} (x-x_i) \right] f^{(n+1)}(\eta)/(n+1)! \qquad (C.3b)$$

where η must lie on or within the limits of integration. The value of η, however, is unknown. The next step is to transform the function and the limits of integration into a natural coordinate system. This is accomplished by defining a new coordinate system r as

$$r = [2x-(a+b)]/(b-a) \quad . \qquad (C.4a)$$

The new function in natural coordinates will be

$$F(r) = f(x) = f((1/2)((b-a)r+(a+b)) \quad . \qquad (C.4b)$$

Equation (C.1) in natural coordinates becomes

$$F(r) = \sum_{i=0}^{n} L_i(r) + \left[\prod_{i=0}^{n} (r-r_i) \right] F^{n+1}(\bar{\eta})/(n+1)! \qquad (C.5a)$$

where

$$L_i(r) = \prod_{j=0, j\neq i}^{n} [(r-r_j)/(r_i-r_j)] \qquad (C.5b)$$

and

$$-1 < \bar{\eta} < 1 \quad . \qquad (C.5c)$$

The r_i's are point transformations of the x_i's. Assuming f(x) is a

polynomial of degree 2n+1, then $\sum_{i=0}^{n} L_i(r)F(r_i)$ is a polynomial of degree n at most. Furthermore, $\prod_{i=0}^{n}(r-r_i)$ is a polynomial of degree n+1, thus forcing the term $F^{(n+1)}(\bar{\eta})/(n+1)!$ to a polynomial of degree n. This is represented by

$$F^{(n+1)}(\bar{\eta})/(n+1)! = g_n(r) \tag{C.6}$$

where $g_n(r)$ is a polynomial of degree n. Equation (C.5a) becomes

$$F(r) = \sum_{i=0}^{n} L_i(r)F(r_i) + \left[\prod_{i=0}^{n}(r-r_i)\right]g_n(r) \tag{C.7}$$

Integration of both sides of (C.7) yields

$$\int_{-1}^{1} F(r) = \int_{-1}^{1}\sum_{i=0}^{n} L_i(r)F(r_i)dr + \int_{-1}^{1}\left[\prod_{i=0}^{n}(r-r_i)\right]g_n(r)dr \quad . \tag{C.8}$$

Dropping the error term and taking the summation operator outside of the integral produces

$$\begin{aligned}\int_{-1}^{1} F(r) &\simeq \sum_{i=0}^{n} F(r_i)\int_{-1}^{1} L_i(r)dr \\ &\simeq \sum_{i=0}^{n} w_i F(r_i)\end{aligned} \tag{C.9a}$$

where

$$w_i = \int_{-1}^{1} L_i(r)dr = \int_{-1}^{1}\prod_{j=0,j\neq i}^{n}[(r-r_j)/(r_i-r_j)]dr. \tag{C.9b}$$

The term w_i is frequently called the weighing function. The location

of the numerical integration points is determined by the error function. The error function is

$$\int_{-1}^{1}\left[\prod_{i=0}^{n}(r-r_i)\right]g_n(r)dr \quad . \tag{C.10}$$

The best way to make the error function vanish is to select values of r_i that drive the function to zero. The first step is to expand the two polynomials ($g_n(r)$ and $\prod_{i=1}^{n}(r-r_i)$) into Legendre polynomials. The first part of the error term becomes

$$\prod_{i=0}^{n}(r-r_i) = c_0Q_0(r) + c_1Q_1(r) + \\ .... + c_nQ_n(r) + c_{n+1}Q_{n+1}(r) = \sum_{i=0}^{n+1}c_iQ_i(r) \tag{C.11a}$$

and the second part of the error term becomes

$$g_n(r) = d_0Q_0(r) + d_1Q_1(r) + \ldots + d_nQ_n(r) \\ = \sum_{i=0}^{n}d_iQ_i(r) \quad . \tag{C.11b}$$

Substituting equations (C.11a) and (C.11b) into the error term, equation (C.10) results in

$$\int_{-1}^{1}\left[\sum_{i=0}^{n}\sum_{j=0}^{n}c_id_jQ_i(r)Q_j(r) + c_{n+1}\sum_{i=0}^{n}d_iQ_i(r)Q_{n+1}(r)\right]dr \quad . \tag{C.12}$$

Since the Legendre polynomials have the following orthogonal properties

$$\int_{-1}^{1} Q_n(r)Q_m(r)dr = 0 \qquad n \neq m \tag{C.13a}$$

$$\int_{-1}^{1} Q_n(r)Q_m(r)dr \neq 0 \qquad n = m \tag{C.13b}$$

all terms of equation (C.12) that are of the form

$$c_i d_j \int_{-1}^{1} Q_i(r)Q_j(r)dr \qquad i \neq j \tag{C.13c}$$

will be zero. The error term will then become

$$\int_{-1}^{1} \left[\prod_{i=0}^{n} (r-r_i) \right] g_n(r)dr = \sum_{i=0}^{n} c_i d_i \int_{-1}^{1} [Q_i(r)]^2 dr \tag{C.14}$$

One way to make equation (C.14) equal zero is to assign a zero value to the first n+1 c_i's. The coefficient c_{n+1} is not known, but equation (C.11a) produces

$$\prod_{i=0}^{n} (r-r_i) = c_{n+1} Q_{n+1}(r) \quad . \tag{C.15}$$

In order for the left hand side of equation (C.15) to be zero, r_i must be the roots of the Legendre polynomial. The general recursion formula for Legendre polynomials is

$$Q_n(r) = [(2n-1)/n]rQ_{n-1}(r) - [(n-1)/n]Q_{n-2}(r) \tag{C.16a}$$

with

$$Q_0(r) = 1 \tag{C.16b}$$

$$Q_1(r) = r \quad . \tag{C.16c}$$

Most finite element routines use a three point integration scheme, and the Legendre polynomial corresponding to n=3 is

$$Q_3(r) = (1/2)(5r^3 - 3r) \quad . \tag{C.16d}$$

The roots of equation (C.16d) are

$$r_1 = -0.77459\ 66692\ 41483$$

$$r_2 = 0.00000\ 00000\ 00000$$

$$r_3 = 0.77459\ 66692\ 41483$$

and the weight factors from equation (C.9b) are

$$w_1 = 0.55555\ 55555\ 55556$$

$$w_2 = 0.88888\ 88888\ 88889$$

$$w_3 = 0.55555\ 55555\ 55556 \quad .$$

The transformation from one dimensional functions to three dimensional functions is an extension of one dimensional theory. The function is evaluated in the other two directions in the same manner it was evaluated in the first direction.

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} f(r,s,t)\,dr\,ds\,dt \simeq \sum_{k=1}^{l}\sum_{j=1}^{m}\sum_{i=1}^{n} w_i \bar{w}_j \tilde{w}_k F(r_i, r_j, r_k) \tag{C.17}$$

In equation (C.17) the same number of integration points don't have to be used in each direction, but for most three dimensional elements, an integration grid of 3X3X3 is used. When the stiffness matrix is transformed from spatial to natural coordinates, a constant term is multiplied into the entire stiffness matrix to preserve equality. The term

$$\int_z\int_y\int_x [B(x,y,z)]^T[C][B(x,y,z)]dxdydz \qquad (C.18)$$

will be transformed into natural coordinates as

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} [B(r,s,t)]^T[C][B(r,s,t)]Det[J]drdsdt \qquad (C.19)$$

where Det[J] is the determinant of the Jacobian matrix.

APPENDIX D

WAVE FRONT SOLUTION

The ANSYS finite element program uses a wave front solution procedure. This is a solution technique for a system of simultaneous linear equations derived by the finite element method. The wave front at any point is the number of equations which are active at that instant. The ordering of elements is critical to minimize the wave front, for reasons of efficiency and problem size. The node numbering is arbitrary. The active equations are

$$\sum_{j=1}^{L} K_{kj} u_j = F_k \qquad (D.1)$$

where K_{kj} = stiffness term kj

u_j = nodal displacement j

F_k = nodal force k

k = row number

j = column number

L = number of equations.

The elimination of an equation (i=k) begins by normalizing the first equation in the following manner

$$\sum_{j=1}^{L}(K_{ij}/K_{ii})u_j = (F_i/K_{ii}) \quad . \tag{D.2}$$

If the finite element formulation is correct, the diagonal entry K_{ii} will never be zero. Equation (D.2) is rewritten as

$$\sum_{j=1}^{L}K'_{ij}u_j = F'_i \tag{D.3}$$

where

$$K'_{ij} = K_{ij}/K_{ii} \tag{D.4}$$

and

$$F'_i = F_i/K_{ii} \quad . \tag{D.5}$$

Equation (D.3) is saved for a back substitution solution. The remaining equations are modified such that

$$K'_{kj} = K_{kj} - K_{ki}K'_{ij} \tag{D.6}$$

$$F'_k = F_k - K_{ki}F'_i \tag{D.7}$$

where $k \neq i$. These equations are assembled as

$$\sum_{j=2}^{L}K'_{kj}u_j = F'_k \tag{D.8}$$

where k varies from 2 to L. Once the first row is eliminated, the

other rows are eliminated by repeating the process. An example of this solution technique is shown below, using a three by three system of equations. The equations are

$$\begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \end{Bmatrix} . \tag{D.9}$$

The first equation is normalized so that

$$u_1 + (k_{12}/k_{11})u_2 + (k_{13}/k_{11})u_3 = f_1/k_{11} \tag{D.10a}$$

The normalized components are then

$$k'_{12} = k_{12}/k_{11} \tag{D.10b}$$

$$k'_{13} = k_{13}/k_{11} \tag{D.10c}$$

$$f'_1 = f_1/k_{11} \quad . \tag{D.10d}$$

The second iteration using the wave front solution produces the following components:

$$k'_{22} = [k_{22} - k_{21}(k_{12}/k_{11})] \tag{D.11a}$$

$$k'_{23} = [k_{23} - k_{21}(k_{13}/k_{11})] \tag{D.11b}$$

$$k'_{32} = [k_{32} - k_{31}(k_{12}/k_{11})] \tag{D.11c}$$

$$k'_{33} = [k_{33} - k_{31}(k_{13}/k_{11})] \tag{D.11d}$$

$$f'_2 = [f_2 - k_{21}(f_2/k_{11})] \tag{D.11e}$$

$$f_3' = [f_3 - k_{31}(f_3/k_{11})] \tag{D.11f}$$

which are assembled as

$$\begin{bmatrix} k_{22}' & k_{23}' \\ k_{32}' & k_{33}' \end{bmatrix} \begin{Bmatrix} u_2 \\ u_3 \end{Bmatrix} = \begin{Bmatrix} f_2' \\ f_3' \end{Bmatrix}. \tag{D.11g}$$

The first equation of (D.11g) is modified to become

$$u_2 + \left[[k_{23} - k_{21}(k_{13}/k_{11})]/[k_{22} - k_{21}(k_{12}/k_{11})]\right]u_3 = [f_2 - k_{21}(f_1/k_{11})]/[k_{22} - k_{21}(k_{12}/k_{11})] \tag{D.12a}$$

while the second equation becomes

$$k_{33}'' u_3 = f_3'' \quad . \tag{D.12b}$$

The term u_3 is solved for as

$$u_3 = f_3''/k_3'' \tag{D.12c}$$

where

$$f_3'' = \left[[k_{22} - k_{21}(k_{12}/k_{11})][f_3 - k_{31}(f_1/k_{11})]\right] - \left[[k_{32} - k_{31}(k_{12}/k_{11})][f_2 - k_{21}(f_1/k_{11})]\right] \tag{D.12d}$$

and

$$k''_{33} = \left[[k_{22} - k_{21}(k_{12}/k_{11})][k_{33} - k_{31}(k_{13}/k_{11})]\right]$$
$$- \left[[k_{32} - k_{31}(k_{12}/k_{11})][k_{23} - k_{21}(k_{13}/k_{11})]\right] \qquad \text{(D.12e)}$$

The result is the same that would be obtained using some other method such as Cramer's Rule. The advantage of the wave front technique is the optimization of computer time during the equation solution phase of the finite element analysis.

APPENDIX E

PLANE STRAIN

Plane strain is a specialization of three dimensional linear elastic theory. It represents the situation where the component of displacement normal to the x-y plane is zero. The material stiffness matrix for the three dimensional case is

$$[C] = E/\{(1+\mu)(1-2\mu)\} \begin{bmatrix} 1-\mu & \mu & \mu & 0 & 0 & 0 \\ \mu & 1-\mu & \mu & 0 & 0 & 0 \\ \mu & \mu & 1-\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & (1-2\mu)/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & (1-2\mu)/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & (1-2\mu)/2 \end{bmatrix}. \quad (E.1)$$

Using the fact that there are no out of plane displacements, the material stiffness matrix compresses to

$$[C] = E/\{(1+\mu)(1-2\mu)\} \begin{bmatrix} 1-\mu & \mu & 0 \\ \mu & 1-\mu & 0 \\ 0 & 0 & (1-2\mu)/2 \end{bmatrix} \quad (E.2)$$

where [C] is expressed in Hooke's law as

$$\{\sigma\} = [C]\{\varepsilon\} \tag{E.3a}$$

where

$$\{\sigma\}^T = [\sigma_x \quad \sigma_y \quad \tau_{xy}] \tag{E.3b}$$

and

$$\{\varepsilon\}^T = [\varepsilon_x \quad \varepsilon_y \quad \gamma_{xy}] \quad . \tag{E.3c}$$

APPENDIX F

TWO DIMENSIONAL DISPLACEMENT DATA

The following tables correspond to the two dimensional model of the wedge area from Chapter 3. They are the displacements for the loaded nodes of the Brooks Supervillinova including the geometry modifications of various size holes.

Table F.1 Y direction displacements of model with one hole

Node	R=0.05	R=0.10	R=0.15	R=0.20	R=0.25	R=0.30
1	-.2990	-.2984	-.2973	-.2957	-.2937	-.2914
2	-.2909	-.2911	-.2917	-.2919	-.2933	-.2953
3	-.2930	-.2952	-.2997	-.3042	-.3128	-.3246
4	-.2888	-.2954	-.3086	-.3269	-.3542	-.3916
5	-.2870	-.2977	-.3180	-.3569	-.4075	-.4798
6	-.2851	-.2953	-.3148	-.3515	-.3994	-.4679
7	-.2825	-.2882	-.2995	-.3155	-.3386	-.3701
8	-.2785	-.2804	-.2840	-.2876	-.2944	-.3036
9	-.2692	-.2696	-.2704	-.2709	-.2725	-.2748
10	-.2427	-.2427	-.2425	-.2420	-.2417	-.2414

Table F.2 Y direction displacements of model with two holes

Node	R=0.05	R=0.10	R=0.15	R=0.20	R=0.25	R=0.30
1	-.2997	-.3013	-.3049	-.3088	-.3170	-.3297
2	-.2929	-.2994	-.3124	-.3321	-.3622	-.4068
3	-.2959	-.3075	-.3301	-.3736	-.4336	-.5253
4	-.2909	-.3043	-.3308	-.3771	-.4445	-.5502
5	-.2877	-.3008	-.3274	-.3659	-.4268	-.5219

Table F.2 (cont'd)

6	-.2858	-.2988	-.3239	-.3685	-.4328	-.5324
7	-.2840	-.2946	-.3152	-.3529	-.4050	-.4829
8	-.2797	-.2853	-.2964	-.3122	-.3360	-.3694
9	-.2696	-.2713	-.2745	-.2777	-.2842	-.2932
10	-.2427	-.2429	-.2432	-.2431	-.2440	-.2453

Table F.3 Y direction displacements of model with three holes

Node	R=0.05	R=0.10	R=0.15	R=0.20	R=0.25	R=0.30
1	-.3014	-.3084	-.3234	-.3480	-.3840	-.4383
2	-.2947	-.3066	-.3302	-.3784	-.4437	-.5450
3	-.2966	-.3103	-.3378	-.3878	-.4594	-.5726
4	-.2908	-.3043	-.3317	-.3718	-.4356	-.5365
5	-.2877	-.3012	-.3282	-.3749	-.4460	-.5605
6	-.2859	-.2993	-.3261	-.3711	-.4405	-.5526
7	-.2847	-.2978	-.3241	-.3619	-.4225	-.5183
8	-.2819	-.2947	-.3192	-.3648	-.4288	-.5281
9	-.2723	-.2825	-.3019	-.3403	-.3913	-.4680
10	-.2442	-.2490	-.2588	-.2730	-.2941	-.3237

Table F.4 X direction displacements of model with one hole

Node	R=0.05	R=0.10	R=0.15	R=0.20	R=0.25	R=0.30
1	.01655	.01983	.02560	.03178	.04197	.05457
2	.00590	.00972	.01669	.02463	.03737	.05351
3	.01703	.02252	.03221	.04480	.06294	.08629
4	.03164	.03718	.04820	.06638	.08989	.1210
5	.04502	.04811	.05307	.06277	.07507	.09260
6	.05578	.05409	.05202	.04549	.03880	.02901
7	.06417	.06053	.05306	.03978	.02387	.00318
8	.06960	.06639	.06091	.05330	.04330	.03089
9	.07606	.07430	.07114	.06761	.06206	.05532
10	.05793	.05687	.05521	.05373	.05114	.04833

Table F.5 X direction displacements of model with two holes

Node	R=0.05	R=0.10	R=0.15	R=0.20	R=0.25	R=0.30
1	.01776	.02535	.03837	.05591	.08309	.1211
2	.00711	.01506	.03013	.05396	.08804	.1368

Table F.5 (cont'd)

3	.01718	.02341	.03411	.05170	.07827	.1191
4	.03084	.03413	.04083	.04626	.05869	.07768
5	.04440	.04566	.04801	.05145	.05790	.06804
6	.05612	.05554	.05411	.05454	.05425	.05432
7	.06423	.06065	.05499	.04387	.02942	.00783
8	.06914	.06441	.05509	.03947	.01942	-.00773
9	.07536	.07135	.06463	.05521	.04230	.02579
10	.05765	.05564	.05217	.04836	.04206	.03441

Table F.6 X direction displacements of model with three holes

Node	R=0.05	R=0.10	R=0.15	R=0.20	R=0.25	R=0.30
1	.01760	.02484	.03794	.05867	.08965	.1348
2	.00627	.01185	.02061	.03472	.05818	.09553
3	.01617	.01953	.02573	.02892	.04087	.06010
4	.03041	.03259	.03617	.04011	.05092	.06977
5	.04452	.04641	.04947	.05532	.06650	.08599
6	.05667	.05812	.06125	.06195	.06608	.07233
7	.06549	.06617	.06755	.06961	.07253	.07652
8	.07036	.06952	.06762	.06792	.06613	.06388
9	.07551	.07193	.06694	.05549	.04140	.01922
10	.05702	.05311	.04546	.03253	.01640	-.00431

APPENDIX G

THREE DIMENSIONAL CONDENSED ELEMENT ROUTINE

This appendix contains the computer program to generate condensed three dimensional element stiffness matrices. The program was written and run on a PRIME 750 computer.

```
C
C
C--This is the driver program, STIF10NOD, that determines the
C   stiffness matrix to a condensed ten node 3D isoprametric
C   element.
C
C--REAL variables
C
C  X(20) - Matrix of X values of element - There are 20 since
C          a condensed 10 node element maps 20 nodes.
C  Y(20) - Matrix of Y values of element
C  Z(20) - Matrix of Z values of element
C
C  A(20) - Mapping coefficient from  RST * A = X
C  B(20) - Mapping coefficient from  RST * B = Y
C  C(20) - Mapping coefficient from  RST * C = Z
C  RST(20,20) - Mapping matrix (see above)
C
C  WORK(21,21) - Work matrix
C
C  MU - Poissons ratio
C  E - Youngs modulas
C
C
C--INTEGER variables
C
C  ITERM - The terminal location
C  ITYPE - The terminal type
C  IRATE - The terminal baud rate
C  ICODE - Error flag for subroutine call
C
C
C
      REAL X(20),Y(20),Z(20),A(20),B(20),C(20)
      REAL WORK(21,21),RST(20,20),MU
      CHARACTER*1 IANS
C
C
C--Determine the terminal type and baud rate
C
      CALL TERM (ITERM,ITYPE,IRATE,ICODE)
      IF (ICODE.NE.0) IRATE = 4800
C
      N = 20
      NP1 = 21
C
C--Input the data and material properties
C
 100  CALL INPUT (X,Y,Z,E,MU)
C
C--Find the location of the condensed nodes, assuming
C   they lie halfway between the endpoints in all
C   three directions
```

```
C
      CALL LOCATE (X,Y,Z)
C
C--Build the coefficient matrix RST where
C   the mapping is dependent on  RST * A = X
C   RST * B = Y , RST * C = Z
C
      CALL ASMBL (RST)
C
C
C--Solve for A, B, and C from above.
C   This uses the linear equation solver
C   in the Math Library on the Prime.
C   The calling sequence is (A,X,RST,WORK,
C   N,NP1,IERR).  The last variable, IERR,
C   is an error flag that indicates if
C   the matrix RST is singular.
C
      CALL LINEQ (A,X,RST,WORK,N,NP1,IERR)
      CALL LINEQ (B,Y,RST,WORK,N,NP1,IERR)
      CALL LINEQ (C,Z,RST,WORK,N,NP1,IERR)
C
C--Check the error messgage.
C
      IF (IERR.EQ.1) WRITE (1,110)
 110  FORMAT (/,1X,'---> MATRIX IS SINGULAR <---')
C
C--Give the user the opportunity to view the
C   element.  This is done by mapping the element
C   from natural coordinates to spatial coordin.
C
 120  WRITE (1,130)
 130  FORMAT (/,1X,'Would you like to see the element? ')
      READ (1,140) IANS
 140  FORMAT (A1)
      IF (IANS.EQ.'Y') CALL DRELE (A,B,C,IRATE)
      IF (IANS.EQ.'Y') GOTO 150
      IF (IANS.EQ.'N') GOTO 150
      WRITE (1,900) IANS
      GOTO 120
C
C--Let the user integrate the element
C
 150  WRITE (1,160)
 160  FORMAT (/,1X,'Would you like to integrate this element? ')
      READ (1,140) IANS
      IF (IANS.EQ.'Y') CALL INTEG (A,B,C,E,MU)
      IF (IANS.EQ.'Y') GOTO 170
      IF (IANS.EQ.'N') GOTO 170
      WRITE (1,900) IANS
      GOTO 150
C
C--Let the user input another element if they wish.
```

```
C
 170   WRITE (1,180)
 180   FORMAT (/,1x,'Would you like to input another element? ')
       READ (1,140) IANS
       IF (IANS.EQ.'Y') GOTO 100
       IF (IANS.EQ.'N') GOTO 190
       WRITE (1,900) IANS
       GOTO 170
C
 190   CONTINUE
C
 900   FORMAT (/,1X,A1,' is not a legal response.  The only',
     1 ' legal responses are Y(Yes) and N(No).')
c
       CALL EXIT
       END
C
C
C---This subroutine, INPUT, will read in element data
C   and material data concerning the ten node degraded
C   element.
C
       SUBROUTINE INPUT (X,Y,Z,E,MU)
       CHARACTER*1 IANS
       REAL X(20),Y(20),Z(20),MU
C
 100   WRITE (1,110)
 110   FORMAT (/,1X,'Enter Youngs modulas: ')
       READ (1,*,ERR=120) E
       GOTO 130
 120   WRITE (1,900)
       GOTO 100
C
 130   WRITE (1,140)
 140   FORMAT (/,1X,'Enter Poissons ratio: ')
       READ (1,*,ERR=150) MU
       GOTO 160
 150   WRITE (1,900)
       GOTO 130
C
 160   DO 210 J = 1, 10, 1
       IF (J.EQ. 1) IANS = 'I'
       IF (J.EQ. 2) IANS = 'J'
       IF (J.EQ. 3) IANS = 'K'
       IF (J.EQ. 4) IANS = 'L'
       IF (J.EQ. 5) IANS = 'M'
       IF (J.EQ. 6) IANS = 'N'
       IF (J.EQ. 7) IANS = 'O'
       IF (J.EQ. 8) IANS = 'P'
       IF (J.EQ. 9) IANS = 'Q'
       IF (J.EQ.10) IANS = 'R'
 170   WRITE (1,180) IANS,IANS,IANS
 180   FORMAT (/,1X,'Enter X(',A1,'), Y(',A1,'), and Z(',A1,'): ')
```

```
      READ (1,*,ERR=190) X(J),Y(J),Z(J)
      GOTO 200
 190  WRITE (1,900)
      GOTO 170
 200  CONTINUE
 210  CONTINUE
C
 220  WRITE (1,230)
 230  FORMAT (/,1X,'The following options are available: ')
      WRITE (1,240)
 240  FORMAT (/,7X,'I - Input new element and material data')
      WRITE (1,250)
 250  FORMAT (7X,'V - View current element and material data')
      WRITE (1,260)
 260  FORMAT (7X,'E - Exit input mode')
      WRITE (1,270)
 270  FORMAT (/,1X,'Which option would you like? ')
      READ (1,280) IANS
 280  FORMAT (A1)
      IF (IANS.EQ.'I') GOTO 100
      IF (IANS.EQ.'V') GOTO 290
      IF (IANS.EQ.'E') GOTO 990
      WRITE (1,910) IANS
      GOTO 220
C
 290  WRITE (1,300) E
 300  FORMAT (/,1X,'Youngs modulas = ',F12.3)
      WRITE (1,310) MU
 310  FORMAT (1X,'Poissons ratio = ',F8.5)
      WRITE (1,320) X(1),Y(1),Z(1)
 320  FORMAT (/,1X,'X( 1)= ',F12.3,4X,'Y( 1)= ',F12.3,4X,
     1 'Z( 1)= ',F12.3)
      DO 340 J = 2, 10, 1
      WRITE (1,330) J,X(J),J,Y(J),J,Z(J)
 330  FORMAT (1X,'X(',I2,')= ',F12.3,4X,'Y(',I2,')= ',F12.3,4X,
     1 'Z(',I2,')= ',F12.3)
 340  CONTINUE
      GOTO 220
C
 900  FORMAT (/,1X,'Error - Data incompatible with program - ',
     1 'Try again')
 910  FORMAT (/,1X,A1,' is not a legal response.  Try again.')
C
 990  CONTINUE
      RETURN
      END
C
C--This subroutine, LOCATE, determines the location
C   of the condensed nodes.  Their location is important
C   to insure complete mapping.
C
      SUBROUTINE LOCATE (X,Y,Z)
C
```

```
      DIMENSION X(20),Y(20),Z(20)
C
C--First move the R node to it's proper
C   twenty node location
C
      X(13) = X(10)
      Y(13) = Y(10)
      Z(13) = Z(10)
C
C--Node between J and K
C
      X(10) = (X(2) + X(3)) / 2.0
      Y(10) = (Y(2) + Y(3)) / 2.0
      Z(10) = (Z(2) + Z(3)) / 2.0
C
C--Node between K and L
C
      X(11) = (X(3) + X(4)) / 2.0
      Y(11) = (Y(3) + Y(4)) / 2.0
      Z(11) = (Z(3) + Z(4)) / 2.0
C
C--Node between L and I
C
      X(12) = (X(4) + X(1)) / 2.0
      Y(12) = (Y(4) + Y(1)) / 2.0
      Z(12) = (Z(4) + Z(1)) / 2.0
C
C--Node between N and O
C
      X(14) = (X(6) + X(7)) / 2.0
      Y(14) = (Y(6) + Y(7)) / 2.0
      Z(14) = (Z(6) + Z(7)) / 2.0
C
C--Node between O and P
C
      X(15) = (X(7) + X(8)) / 2.0
      Y(15) = (Y(7) + Y(8)) / 2.0
      Z(15) = (Z(7) + Z(8)) / 2.0
C
C--Node between P and M
C
      X(16) = (X(8) + X(5)) / 2.0
      Y(16) = (Y(8) + Y(5)) / 2.0
      Z(16) = (Z(8) + Z(5)) / 2.0
C
C--Node between I and M
C
      X(17) = (X(1) + X(5)) / 2.0
      Y(17) = (Y(1) + Y(5)) / 2.0
      Z(17) = (Z(1) + Z(5)) / 2.0
C
C--Node between J and N
C
```

```
      X(18) = (X(2) + X(6)) / 2.0
      Y(18) = (Y(2) + Y(6)) / 2.0
      Z(18) = (Z(2) + Z(6)) / 2.0
C
C--Node between K and O
C
      X(19) = (X(3) + X(7)) / 2.0
      Y(19) = (Y(3) + Y(7)) / 2.0
      Z(19) = (Z(3) + Z(7)) / 2.0
C
C--Node between L and P
C
      X(20) = (X(4) + X(8)) / 2.0
      Y(20) = (Y(4) + Y(8)) / 2.0
      Z(20) = (Z(4) + Z(8)) / 2.0
C
C
      RETURN
      END
C
C--This subroutine, ASMBL, assembles the coefficient
C   matrix for the mapping of the function between
C   natural coordinates and spatial coordinates.
C   It is built on a twenty point map and passed back to
C   the driver program.
C
      SUBROUTINE ASMBL (RST)
C
      REAL RST(20,20),R(20),S(20),T(20)
C
C--Matrices R, S, and T are the nodal positions
C   in natural coordinates.
C
      DATA (S(I),I=1,20)/-1.0,1.0,1.0,-1.0,-1.0,1.0,1.0,-1.0,
     1 0.0,1.0,0.0,-1.0,0.0,1.0,0.0,-1.0,-1.0,1.0,1.0,-1.0/
      DATA (T(I),I=1,20)/-1.0,-1.0,1.0,1.0,-1.0,-1.0,1.0,1.0,
     1 -1.0,0.0,1.0,0.0,-1.0,0.0,1.0,0.0,-1.0,-1.0,1.0,1.0/
      DATA (R(I),I=1,20)/-1.0,-1.0,-1.0,-1.0,1.0,1.0,1.0,1.0,
     1 -1.0,-1.0,-1.0,-1.0,1.0,1.0,1.0,1.0,0.0,0.0,0.0,0.0/
C
      DO 100 J = 1, 20, 1
C
      RST(J, 1) = 1.0
      RST(J, 2) = R(J)
      RST(J, 3) = S(J)
      RST(J, 4) = T(J)
      RST(J, 5) = R(J)**2
      RST(J, 6) = S(J)**2
      RST(J, 7) = T(J)**2
      RST(J, 8) = S(J)*T(J)
      RST(J, 9) = S(J)*R(J)
      RST(J,10) = R(J)*T(J)
C
```

```
      RST(J,11) = S(J)**2 * T(J)
      RST(J,12) = S(J)**2 * R(J)
      RST(J,13) = T(J)**2 * S(J)
      RST(J,14) = T(J)**2 * R(J)
      RST(J,15) = R(J)**2 * S(J)
      RST(J,16) = R(J)**2 * T(J)
      RST(J,17) = R(J) * S(J) * T(J)
      RST(J,18) = S(J)**2 * R(J) * T(J)
      RST(J,19) = T(J)**2 * R(J) * S(J)
      RST(J,20) = R(J)**2 * S(J) * T(J)
 100  CONTINUE
C
      RETURN
      END
C
C
C--This subroutine, DRELE, draws the degraded twenty
C   node axis by using the mapping vectors A, B, and
C   C and remapping the element back from natural
C   coordinates to spatial coordinates.
C
      SUBROUTINE DRELE (A,B,C,IRATE)
C
      REAL A(20),B(20),C(20)
      CHARACTER*1 IANS
C
C--Set the initial scale value, the initial viewpoint
C   and the initial reference point.
C
      SCALE = 2.
      XVP = 10.
      YVP = 10.
      ZVP = 10.
      XR = 0.
      YR = 0.
      ZR = 0.
C
      CALL RECOVR
      CALL INITT3 (IRATE/10)
 100  CALL ANMODE
      WRITE (1,110)
 110  FORMAT (/,1X,'Current graphic displays set at: ')
      WRITE (1,120) XVP,YVP,ZVP
 120  FORMAT (5X,'Viewpoint set at (x,y,z): ',3F7.2)
      WRITE (1,130) XR,YR,ZR
 130  FORMAT (5X,'Reference point set at (x,y,z): ',3F7.2)
      WRITE (1,140) SCALE
 140  FORMAT (5X,'Scaling factor set at: ',F7.2)
C
 145  WRITE (1,150)
 150  FORMAT (/,1X,'The following options are available: ')
      WRITE (1,160)
 160  FORMAT (/,5X,'D - Draw the element')
```

```
      WRITE (1,170)
 170  FORMAT (5X,'V - Change the viewpoint')
      WRITE (1,180)
 180  FORMAT (5X,'R - Change the reference point')
      WRITE (1,190)
 190  FORMAT (5X,'S - Change the scaling factor')
      WRITE (1,200)
 200  FORMAT (5X,'P - View the graphic parameters')
      WRITE (1,210)
 210  FORMAT (5X,'E - Exit the element draw mode')
      WRITE (1,220)
 220  FORMAT (/,1X,'Which option would you like? ')
      READ (1,230) IANS
 230  FORMAT (A1)
      IF (IANS.EQ.'D') GOTO 310
      IF (IANS.EQ.'V') GOTO 250
      IF (IANS.EQ.'R') GOTO 270
      IF (IANS.EQ.'S') GOTO 290
      IF (IANS.EQ.'P') GOTO 100
      IF (IANS.EQ.'E') GOTO 990
      WRITE (1,240) IANS
 240  FORMAT (/,1X,A1,' is not a legal answer. Try again.')
      GOTO 145
C
 250  WRITE (1,260)
 260  FORMAT (/,1X,'Enter the new viewpoint (x,y,z): ')
      READ (1,*,ERR=250) XVP,YVP,ZVP
      GOTO 145
C
 270  WRITE (1,280)
 280  FORMAT (/,1X,'Enter the new reference point (x,y,z): ')
      READ (1,*,ERR=270) XR,YR,ZR
      GOTO 145
C
 290  WRITE (1,300)
 300  FORMAT (/,1X,'Enter the new scaling factor: ')
      READ (1,*,ERR=290) SCALE
      GOTO 145
C
 310  CALL RECOVR
      CALL DWINDO (-10.23,10.23,-7.8,7.8)
      CALL CARTVP (XVP,YVP,ZVP)
      CALL LOOKAT (XR,YR,ZR)
      CALL ZUP
      CALL MAGNFY (SCALE)
C
C--Draw the bottom of the element
C   and then the top
C
      DO 360 K = 1, 2, 1
C
      IF (K.EQ.1) R = -1.
      IF (K.EQ.2) R =  1.
```

```
C
      T = -1.
      DO 320 S = -1., 1., .02
      CALL FCT (X,Y,Z,A,B,C,R,S,T)
      IF (S.EQ.-1.) CALL MOVEA3 (X,Y,Z)
      CALL DRAWA3 (X,Y,Z)
 320  CONTINUE
      S = 1.
      DO 330 T = -1.,  1.,  .02
      CALL FCT (X,Y,Z,A,B,C,R,S,T)
      CALL DRAWA3 (X,Y,Z)
 330  CONTINUE
      T =  1.
      DO 340 S = 1., -1., -.02
      CALL FCT (X,Y,Z,A,B,C,R,S,T)
      CALL DRAWA3 (X,Y,Z)
 340  CONTINUE
      S = -1.
      DO 350 T =  1., -1., -.02
      CALL FCT (X,Y,Z,A,B,C,R,S,T)
      CALL DRAWA3 (X,Y,Z)
 350  CONTINUE
C
 360  CONTINUE
C
C--Connect the top and bottom
C
      DO 380 K = 1, 4, 1
C
      IF (K.EQ.1) S = -1.
      IF (K.EQ.1) T = -1.
      IF (K.EQ.2) S =  1.
      IF (K.EQ.2) T = -1.
      IF (K.EQ.3) S =  1.
      IF (K.EQ.3) T =  1.
      IF (K.EQ.4) S = -1.
      IF (K.EQ.4) T =  1.
C
      DO 370 R = -1., 1., .02
      CALL FCT (X,Y,Z,A,B,C,R,S,T)
      IF (R.EQ.-1.) CALL MOVEA3 (X,Y,Z)
      CALL DRAWA3 (X,Y,Z)
 370  CONTINUE
C
 380  CONTINUE
      CALL ANMODE
C
      GOTO 145
C
 990  CONTINUE
      RETURN
      END
C
```

```
C
C--This subroutine, FCT, maps points in natural
C   coordinates back into spatial coordinates.
C
      SUBROUTINE FCT (X,Y,Z,A,B,C,R,S,T)
C
      REAL A(20),B(20),C(20)
C
      X = A(1) + A(2)*R + A(3)*S + A(4)*T + A(5)*R*R
     1 + A(6)*S*S + A(7)*T*T + A(8)*S*T + A(9)*S*R +
     1 A(10)*R*T + A(11)*S*S*T + A(12)*S*S*R + A(13)*S*T*T
     1 + A(14)*R*T*T + A(15)*R*R*S + A(16)*R*R*T +
     1 A(17)*R*S*T + A(18)*S*S*R*T + A(19)*R*S*T*T +
     1 A(20)*R*R*S*T
C
      Y = B(1) + B(2)*R + B(3)*S + B(4)*T + B(5)*R*R
     1 + B(6)*S*S + B(7)*T*T + B(8)*S*T + B(9)*S*R +
     1 B(10)*R*T + B(11)*S*S*T + B(12)*S*S*R + B(13)*S*T*T
     1 + B(14)*R*T*T + B(15)*R*R*S + B(16)*R*R*T +
     1 B(17)*R*S*T + B(18)*S*S*R*T + B(19)*R*S*T*T +
     1 B(20)*R*R*S*T
C
      Z = C(1) + C(2)*R + C(3)*S + C(4)*T + C(5)*R*R
     1 + C(6)*S*S + C(7)*T*T + C(8)*S*T + C(9)*S*R +
     1 C(10)*R*T + C(11)*S*S*T + C(12)*S*S*R + C(13)*S*T*T
     1 + C(14)*R*T*T + C(15)*R*R*S + C(16)*R*R*T +
     1 C(17)*R*S*T + C(18)*S*S*R*T + C(19)*R*S*T*T +
     1 C(20)*R*R*S*T
C
      RETURN
      END
C
C
C--This subroutine, INTEG, integrates the element
C   stiffness matrix and builds the matrix.
C   The integration scheme used is a 3X3X3 Gauss
C   Legendre quadrature method
C
C--Variable identification
C
C--REAL variables
C
C  A(20) - Mapping coefficients - X direction
C  B(20) - Mapping coefficients - Y direction
C  C(20) - Mapping coefficients - Z direction
C
C  R(3) - R natural coordinate integration points
C  S(3) - S natural coordinate integration points
C  T(3) - T natural coordinate integration points
C  WR(3) - Weight functions, r direction
C  WS(3) - Weight functions, s direction
C  WT(3) - Weight functions, t direction
C
```

```
C  DSFRST(3,10) - Derivative of the shape functions with resp to RST
C  DSFXYZ(3,10) - Derivative of the shape functions with resp to XYZ
C
C  BMAT(6,30) - The B matrix used to form the stiffness matrix
C  BTMAT(30,6) - The transpose of the B matrix
C  CMAT(6,6) - The material matrix used to form the stiffness matrix
C
C  INVJAC(3,3) - The inverse of the Jacobian matrix
C
C  PART(30,30) - A partial sum of the stiffness matrix
C  STIF(30,30) - The element stiffness matrix
C
C
      SUBROUTINE INTEG (A,B,C,E,MU)
C
      REAL A(20),B(20),C(20),MU
      REAL R(3),S(3),T(3),WR(3),WS(3),WT(3)
      REAL DSFRST(3,10),DSFXYZ(3,10)
      REAL BMAT(6,30),BTMAT(30,6),CMAT(6,6)
      REAL INVJAC(3,3)
      REAL STIF(30,30),PART(30,30),WORK(30,6)
C
      DATA (R(I),I=1,3)/-.7745966692,0.0,.7745966692/
      DATA (S(I),I=1,3)/-.7745966692,0.0,.7745966692/
      DATA (T(I),I=1,3)/-.7745966692,0.0,.7745966692/
      DATA (WR(I),I=1,3)/.5555555556,.8888888889,.5555555556/
      DATA (WS(I),I=1,3)/.5555555556,.8888888889,.5555555556/
      DATA (WT(I),I=1,3)/.5555555556,.8888888889,.5555555556/
      DATA STIF/900*0.0/
      DATA PART/900*0.0/
      DATA BTMAT/180*0.0/
      DATA BMAT/180*0.0/
      DATA CMAT/36*0.0/
C
C--Start the integration loop
C
      DO 330 I = 1, 3, 1
      DO 320 J = 1, 3, 1
      DO 310 K = 1, 3, 1
C
C--Get the derivatives of the shape functions
C   in thier natural coordinates
C
      CALL DERSF (DSFRST,R(I),S(K),T(J))
C
C--Determine the Jacobain matrix inverse, the Jacobian
C   matrix determinant, for the numerical values of
C   R, S, and T.
C
      CALL JACOB (A,B,C,R(I),S(K),T(J),DETJAC,INVJAC)
C
C--Transform the derivatives of the shape functions
C   into XYZ spatial coordinates.
```

```
C   This involves using the matrix multplication
C   in the PRIME math library, called MMLT
C
       CALL MMLT (DSFXYZ,INVJAC,DSFRST,3,3,10)
C
C--Get the B matrix from DSFXYZ for the stiffness matrix
C   Also find the transpose of the B matrix.  These two
C   matrices are stored in BMAT and BTMAT
C
       CALL MAKEB (DSFXYZ,BMAT,BTMAT)
C
C
C--Get the material matrix, this is called inside
C   the integration loop because some matrix routines
C   destroy the product matrices.
C
       CALL MAKEC (CMAT,E,MU)
C
C--Build the stiffness matrix parts
C
       CALL MMLT (WORK,BTMAT,CMAT,30,6,6)
       CALL MMLT (PART,WORK,BMAT,30,6,30)
C
C--Multiply the weight factors and DETJAC by the
C   individual summation part of the stiffness matrix
C
       CALL MSCL (PART,PART,30,30,(WR(I)*WS(K)*WT(J)))
       CALL MSCL (PART,PART,30,30,DETJAC)
C
C--Sum the partial stiffness matrix to the entire term
C
       CALL MADD (STIF,STIF,PART,30,30)
C
C
 310   CONTINUE
 320   CONTINUE
 330   CONTINUE
C
C--Output the results to a file named OUTPUT
C
       OPEN (14,FILE='OUTPUT')
       DO 510 I = 1, 30, 1
       DO 500 J = 1, 30, 6
       WRITE (14,*) STIF(I,J),STIF(I,J+1),STIF(I,J+2),STIF(I,J+3),
     1 STIF(I,J+4),STIF(I,J+5)
 500   CONTINUE
 510   CONTINUE
C
       CLOSE (14)
C
       RETURN
       END
C
```

```
C--This subroutine, MAKEC, assembles the material
C   matrix in a location called CMAT.
C
      SUBROUTINE MAKEC (CMAT,E,MU)
C
      REAL CMAT(6,6),MU
C
      CMAT(1,1) = 1 - MU
      CMAT(1,2) = MU
      CMAT(1,3) = MU
C
      CMAT(2,1) = MU
      CMAT(2,2) = 1 - MU
      CMAT(2,3) = MU
C
      CMAT(3,1) = MU
      CMAT(3,2) = MU
      CMAT(3,3) = 1 - MU
C
      CMAT(4,4) = (1 - (2.0*MU)) / 2.0
      CMAT(5,5) = (1 - (2.0*MU)) / 2.0
      CMAT(6,6) = (1 - (2.0*MU)) / 2.0
C
      DO 110 I = 1, 6, 1
      DO 100 J = 1, 6, 1
      CMAT(I,J) = ( E / ((1+MU)*(1-(2.0*MU)))) * CMAT(I,J)
 100  CONTINUE
 110  CONTINUE
C
      RETURN
      END
C
C--This subroutine, DERSF, calculates the
C   derivatives of the shape functions in
C   natural coordinates.
C
C
      SUBROUTINE DERSF (DSFRST,R,S,T)
C
      REAL DSFRST(3,10)
C
      DNDSI = .125 * (1-T) * (1-R) * (2.0*S + T + R + 1.0)
      DNDTI = .125 * (1-S) * (1-R) * (S + 2.0*T + R + 1.0)
      DNDRI = .125 * (1-S) * (1-T) * (S + T + 2.0*R + 1.0)
C
      DNDSJ = .125 * (1-T) * (1-R) * (2.0*S - T - R - 1.0)
      DNDTJ = .125 * (1-R) * (1+S) * (2.0*T - S + R + 1.0)
      DNDRJ = .125 * (1+S) * (1-T) * (T - S + 2.0*R + 1.0)
C
      DNDSK = .125 * (1+T) * (1-R) * (2.0*S + T - R - 1.0)
      DNDTK = .125 * (1+S) * (1-R) * (S + 2.0*T - R - 1.0)
      DNDRK = .125 * (1+S) * (1+T) * (2.0*R - S - T + 1.0)
C
```

```
      DNDSL = .125 * (1+T) * (1-R) * (2.0*S - T + R + 1.0)
      DNDTL = .125 * (1-S) * (1-R) * (2.0*T - S - R - 1.0)
      DNDRL = .125 * (1-S) * (1+T) * (S - T + 2.0*R + 1.0)
C
      DNDSM = .125 * (1-T) * (1+R) * (2.0*S + T - R + 1.0)
      DNDTM = .125 * (1-S) * (1+R) * (S + 2.0*T - R + 1.0)
      DNDRM = .125 * (1-S) * (1-T) * (2.0*R - S - T - 1.0)
C
      DNDSN = .125 * (1-T) * (1+R) * (2.0*S - T + R - 1.0)
      DNDTN = .125 * (1+S) * (1+R) * (2.0*T - S - R + 1.0)
      DNDRN = .125 * (1+S) * (1-T) * (S - T + 2.0*R - 1.0)
C
      DNDSO = .125 * (1+T) * (1+R) * (2.0*S + T + R - 1.0)
      DNDTO = .125 * (1+S) * (1+R) * (S + 2.0*T + R - 1.0)
      DNDRO = .125 * (1+S) * (1+T) * (S + T + 2.0*R - 1.0)
C
      DNDSP = .125 * (1+T) * (1+R) * (2.0*S - T - R + 1.0)
      DNDTP = .125 * (1-S) * (1+R) * (2.0*T - S + R - 1.0)
      DNDRP = .125 * (1-S) * (1+T) * (T - S + 2.0*R - 1.0)
C
      DNDSQ = .25 * (1-T) * (1-R) * (-2.0)*S
      DNDTQ = .25 * (1-R) * (1-(S**2)) * (-1.0)
      DNDRQ = .25 * (1-T) * (1-(S**2)) * (-1.0)
C
      DNDSR = .25 * (1-R) * (1-(T**2))
      DNDTR = .25 * (1+S) * (1-R) * (-2.0)*T
      DNDRR = .25 * (1+S) * (1-(T**2)) * (-1.0)
C
      DNDSS = .25 * (1+T) * (1-R) * (-2.0*S)
      DNDTS = .25 * (1-R) * (1-(S**2))
      DNDRS = .25 * (1+T) * (1-(S**2)) * (-1.0)
C
      DNDST = .25 * (1-R) * (1-(T**2)) * (-1.0)
      DNDTT = .25 * (1-S) * (1-R) * (-2.0)*T
      DNDRT = .25 * (1-S) * (1-(T**2)) * (-1.0)
C
      DNDSU = .25 * (1-T) * (1+R) * (-2.0)*S
      DNDTU = .25 * (1+R) * (1-(S**2)) * (-1.0)
      DNDRU = .25 * (1-T) * (1-(S**2))
C
      DNDSV = .25 * (1+R) * (1-(T**2))
      DNDTV = .25 * (1+S) * (1+R) * (-2.0)*T
      DNDRV = .25 * (1+S) * (1-(T**2))
C
      DNDSW = .25 * (1+T) * (1+R) * (-2.0)*S
      DNDTW = .25 * (1+R) * (1-(S**2))
      DNDRW = .25 * (1+T) * (1-(S**2))
C
      DNDSX = .25 * (1+R) * (1-(T**2)) * (-1.0)
      DNDTX = .25 * (1-S) * (1+R) * (-2.0)*T
      DNDRX = .25 * (1-S) * (1-(T**2))
C
      DNDSY = .25 * (1-T) * (1-(R**2)) * (-1.0)
```

```
      DNDTY = .25 * (1-S) * (1-(R**2)) * (-1.0)
      DNDRY = .25 * (1-S) * (1-T) * (-2.0)*R
C
      DNDSZ = .25 * (1-T) * (1-(R**2))
      DNDTZ = .25 * (1+S) * (1-(R**2)) * (-1.0)
      DNDRZ = .25 * (1+S) * (1-T) * (-2.0)*R
C
      DNDSA = .25 * (1+T) * (1-(R**2))
      DNDTA = .25 * (1+S) * (1-(R**2))
      DNDRA = .25 * (1+S) * (1+T) * (-2.0)*R
C
      DNDSB = .25 * (1+T) * (1-(R**2)) * (-1.0)
      DNDTB = .25 * (1-S) * (1-(R**2))
      DNDRB = .25 * (1-S) * (1+T) * (-2.0)*R
C
C--Load the derivatives into the matrix DSFRST
C
      DSFRST(1, 1) = DNDSI + (DNDST/2.0) + (DNDSY/2.0)
      DSFRST(2, 1) = DNDTI + (DNDTT/2.0) + (DNDTY/2.0)
      DSFRST(3, 1) = DNDRI + (DNDRT/2.0) + (DNDRY/2.0)
      DSFRST(1, 2) = DNDSJ + (DNDSR/2.0) + (DNDSZ/2.0)
      DSFRST(2, 2) = DNDTJ + (DNDTR/2.0) + (DNDTZ/2.0)
      DSFRST(3, 2) = DNDRJ + (DNDRR/2.0) + (DNDRZ/2.0)
      DSFRST(1, 3) = DNDSK + (DNDSR/2.0) + (DNDSS/2.0) + (DNDSA/2.0)
      DSFRST(2, 3) = DNDTK + (DNDTR/2.0) + (DNDTS/2.0) + (DNDTA/2.0)
      DSFRST(3, 3) = DNDRK + (DNDRR/2.0) + (DNDRS/2.0) + (DNDRA/2.0)
      DSFRST(1, 4) = DNDSL + (DNDSS/2.0) + (DNDST/2.0) + (DNDSB/2.0)
      DSFRST(2, 4) = DNDTL + (DNDTS/2.0) + (DNDTT/2.0) + (DNDTB/2.0)
      DSFRST(3, 4) = DNDRL + (DNDRS/2.0) + (DNDRT/2.0) + (DNDRB/2.0)
      DSFRST(1, 5) = DNDSM + (DNDSX/2.0) + (DNDSY/2.0)
      DSFRST(2, 5) = DNDTM + (DNDTX/2.0) + (DNDTY/2.0)
      DSFRST(3, 5) = DNDRM + (DNDRX/2.0) + (DNDRY/2.0)
C
      DSFRST(1, 6) = DNDSN + (DNDSV/2.0) + (DNDSZ/2.0)
      DSFRST(2, 6) = DNDTN + (DNDTV/2.0) + (DNDTZ/2.0)
      DSFRST(3, 6) = DNDRN + (DNDRV/2.0) + (DNDRZ/2.0)
      DSFRST(1, 7) = DNDSO + (DNDSV/2.0) + (DNDSW/2.0) + (DNDSA/2.0)
      DSFRST(2, 7) = DNDTO + (DNDTV/2.0) + (DNDTW/2.0) + (DNDTA/2.0)
      DSFRST(3, 7) = DNDRO + (DNDRV/2.0) + (DNDRW/2.0) + (DNDRA/2.0)
      DSFRST(1, 8) = DNDSP + (DNDSW/2.0) + (DNDSX/2.0) + (DNDSB/2.0)
      DSFRST(2, 8) = DNDTP + (DNDTW/2.0) + (DNDTX/2.0) + (DNDTB/2.0)
      DSFRST(3, 8) = DNDRP + (DNDRW/2.0) + (DNDRX/2.0) + (DNDRB/2.0)
      DSFRST(1, 9) = DNDSQ
      DSFRST(2, 9) = DNDTQ
      DSFRST(3, 9) = DNDRQ
      DSFRST(1,10) = DNDSU
      DSFRST(2,10) = DNDTU
      DSFRST(3,10) = DNDRU
C
C
      RETURN
      END
C
```

```
C
C--This subroutine, JACOB, determines the Jacobian
C   matrix of the twenty node map.  It also finds
C   the determinant of the Jacobian and it's inverse.
C
      SUBROUTINE JACOB (A,B,C,R,S,T,DETJAC,INVJAC)
C
      REAL A(20),B(20),C(20),INVJAC(3,3),JAC(3,3),WORK(4,6)
C
      DXDS = A(3) + A(6)*2.0*S + A(8)*T + A(9)*R +
     1 A(11)*2.0*S*T + A(12)*2.0*S*R + A(13)*T**2 + A(15)*R**2 +
     1 A(17)*R*T + A(18)*2.0*R*S*T + A(19)*R*T**2 + A(20)*T*R**2
      DXDT = A(4) + A(7)*2.0*T + A(8)*S + A(10)*R +
     1 A(11)*S**2 + A(13)*2.0*S*T + A(14)*2.0*R*T + A(16)*R**2 +
     1 A(17)*R*S + A(18)*R*S**2 + A(19)*2.0*R*S*T + A(20)*S*R**2
      DXDR = A(2) + A(5)*2.0*R + A(9)*S + A(10)*T +
     1 A(12)*S**2 + A(14)*T**2 + A(15)*2.0*R*S + A(16)*2.0*R*T +
     1 A(17)*S*T + A(18)*T*S**2 + A(19)*S*T**2 + A(20)*R*S*T
C
      DYDS = B(3) + B(6)*2.0*S + B(8)*T + B(9)*R +
     1 B(11)*2.0*S*T + B(12)*2.0*S*R + B(13)*T**2 + B(15)*R**2 +
     1 B(17)*R*T + B(18)*2.0*R*S*T + B(19)*R*T**2 + B(20)*T*R**2
      DYDT = B(4) + B(7)*2.0*T + B(8)*S + B(10)*R +
     1 B(11)*S**2 + B(13)*2.0*S*T + B(14)*2.0*R*T + B(16)*R**2 +
     1 B(17)*R*S + B(18)*R*S**2 + B(19)*2.0*R*S*T + B(20)*S*R**2
      DYDR = B(2) + B(5)*2.0*R + B(9)*S + B(10)*T +
     1 B(12)*S**2 + B(14)*T**2 + B(15)*2.0*R*S + B(16)*2.0*R*T +
     1 B(17)*S*T + B(18)*T*S**2 + B(19)*S*T**2 + B(20)*R*S*T
C
      DZDS = C(3) + C(6)*2.0*S + C(8)*T + C(9)*R +
     1 C(11)*2.0*S*T + C(12)*2.0*S*R + C(13)*T**2 + C(15)*R**2 +
     1 C(17)*R*T + C(18)*2.0*R*S*T + C(19)*R*T**2 + C(20)*T*R**2
      DZDT = C(4) + C(7)*2.0*T + C(8)*S + C(10)*R +
     1 C(11)*S**2 + C(13)*2.0*S*T + C(14)*2.0*R*T + C(16)*R**2 +
     1 C(17)*R*S + C(18)*R*S**2 + C(19)*2.0*R*S*T + C(20)*S*R**2
      DZDR = C(2) + C(5)*2.0*R + C(9)*S + C(10)*T +
     1 C(12)*S**2 + C(14)*T**2 + C(15)*2.0*R*S + C(16)*2.0*R*T +
     1 C(17)*S*T + C(18)*T*S**2 + C(19)*S*T**2 + C(20)*R*S*T
C
C--Assemble the Jacobian matrix
C
      JAC(1,1) = DXDS
      JAC(1,2) = DYDS
      JAC(1,3) = DZDS
C
      JAC(2,1) = DXDT
      JAC(2,2) = DYDT
      JAC(2,3) = DZDT
C
      JAC(3,1) = DXDR
      JAC(3,2) = DYDR
      JAC(3,3) = DZDR
C
```

```
C--Use the matrix inversion routine on the PRIME.
C   The calling sequence is
C   call minv (OUTPUT,INPUT,ISIZE,WORK,ISIZE+1,
C   ISIZE+ISIZE,ERROR FLAG)
C
C
      N = 3
      NP1 = 4
      NPN = 6
C
      CALL MINV (INVJAC,JAC,N,WORK,NP1,NPN,IERR)
C
C--Solve for the Determinant of the Jacobian
C
      DETJAC = (JAC(1,1)*JAC(2,2)*JAC(3,3)) +
     1         (JAC(1,2)*JAC(2,3)*JAC(3,1)) +
     1         (JAC(2,1)*JAC(3,2)*JAC(1,3)) -
     1         (JAC(3,1)*JAC(2,2)*JAC(1,3)) -
     1         (JAC(2,1)*JAC(1,2)*JAC(3,3)) -
     1         (JAC(3,2)*JAC(2,3)*JAC(1,1))
C
      IF (IERR.NE.0) WRITE (1,100)
 100  FORMAT (/,1X,'---> JACOBIAN IS SINGULAR <---')
C
      RETURN
      END
C
C
C--This subroutine, MAKEB, builds the B matrix
C   (called BMAT) in the stiffness matrix.
C   The transpose of B (called BTMAT) is also
C   assembled and passed back in INTEG.
C
      SUBROUTINE MAKEB (DSFXYZ,BMAT,BTMAT)
C
      REAL DSFXYZ(3,10),BMAT(6,30),BTMAT(30,6)
C
C--Build the B matrix from the matrix DSFXYZ
C
      K = 1
      DO 100 I = 1, 30, 3
      BMAT(1,  I) = DSFXYZ(1,K)
      BMAT(2,I+1) = DSFXYZ(2,K)
      BMAT(3,I+2) = DSFXYZ(3,K)
      BMAT(4,  I) = DSFXYZ(2,K)
      BMAT(4,I+1) = DSFXYZ(1,K)
      BMAT(5,I+1) = DSFXYZ(3,K)
      BMAT(5,I+2) = DSFXYZ(2,K)
      BMAT(6,  I) = DSFXYZ(3,K)
      BMAT(6,I+2) = DSFXYZ(1,K)
      K = K + 1
 100  CONTINUE
C
```

```
C--Find the transpose of the B matrix and place it
C    in the location BTMAT
C
       DO 120 I = 1, 30, 1
       DO 110 J = 1,  6, 1
       BTMAT(I,J) = BMAT(J,I)
 110   CONTINUE
 120   CONTINUE
C
       RETURN
       END
```