EVALUATION OF A THREE-DIMENSIONAL STRESS ANALYSIS PROGRAM

Charles Gregory Pfeifer

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

EVALUATION OF A THREE-DIMENSIONAL
STRESS ANALYSIS PROGRAM

by

Charles Gregory Pfeifer

Thesis Advisor: G. Cantin

September 1972

Approved for public release; distribution unlimited.

Evaluation of a Three-Dimensional
Stress Analysis Program

by

Charles Gregory Pfeifer
Lieutenant, United States Navy
B.S., United States Naval Academy, 1966

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1972

ABSTRACT

The objective of this work was to analyze a computer program using three dimensional quadratic isoparametric finite elements for structural analysis. Three problems with classical solutions were run with various mesh sizes using the computer program being tested. The data computed was then extensively analyzed, and compared with the classical solutions. The analysis of a fourth problem was continued and compared with results obtained in an earlier project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

Before the advent of the finite element technique and the digital computer, the solutions to most non-trivial elasto-static problems were unobtainable. Since that time many finite elements have been devised with corresponding computer programs for their use.

One of the most versatile family of finite elements was developed by Professor Zienkiewicz and his co-workers at the University of Wales, Swansea, U.K. This family of elements, called "ISOPARAMETRIC", has been investigated at the Naval Postgraduate School by Professor G. Cantin, and a stress analysis program called TRISOP was written using a 20 nodal point quadratic element.

Four problems with known solutions will be solved using TRISOP and analyzed in this thesis. From this analysis conclusions will be made as to the effectiveness and accuracy of TRISOP.

II. QUADRATIC ISOPARAMETERIC FINITE ELEMENTS

The three-dimensional element discussed in this chapter is the type of element used in the program analyzed by this author. No attempt will be made to fully describe this type of element, but just to give some insight as to how the element is constructed and designed.

The three-dimensional quadratic isoparameteric finite element, hereafter referred to as the "element", is a cube with all sides two units in length in it undeformed, non-dimensional state. The element has 20 nodal points and is oriented with its geometric center at the origin of its non-dimensionalized coordinates, (ξ,η,ζ), and transferred to global, (X,Y,Z), coordinates as shown in Figure 1.

A. TRANSFORMATIONS

Transformations from global coordinates to non-dimensional coordinates are accomplished using shape functions as shown in equations 1

$$\begin{aligned} x(\xi,\eta,\zeta) &= N_i(\xi,\eta,\zeta)x_i \\ y(\xi,\eta,\zeta) &= N_i(\xi,\eta,\zeta)y_i \\ z(\xi,\eta,\zeta) &= N_i(\xi,\eta,\zeta)z_i \end{aligned} \qquad (1)$$

where $N_i(\xi,\eta,\zeta)$ are shape functions, and x_i,y_i,z_i are coordinates of the nodal points. Similarly displacement transformations are made using equations 2,

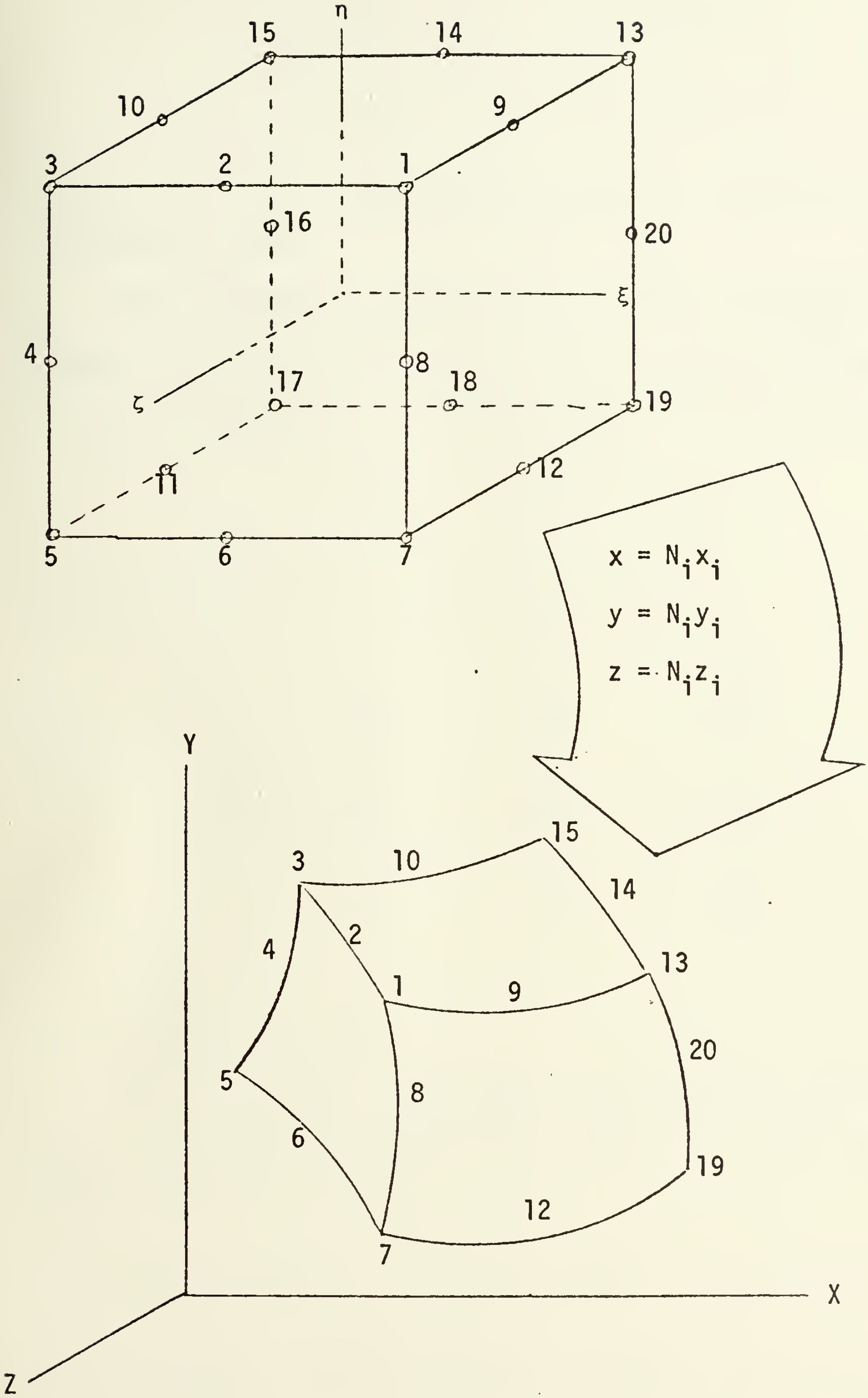


Figure 1. Element Transformation

$$u(\xi,\eta,\zeta) = N_i(\xi,\eta,\zeta)u_i$$
$$v(\xi,\eta,\zeta) = N_i(\xi,\eta,\zeta)v_i \quad (2)$$
$$w(\xi,\eta,\zeta) = N_i(\xi,\eta,\zeta)w_i$$

where u, v, and w are displacements in the global (X,Y,Z) reference frame.

The transformation of a volumetric increment from local (ξ,η,ζ) to the global (X,Y,Z) system of reference is shown in equation 3. The Jacobian, [J] used in the

$$dxdydz = \det[J]\, d\xi d\eta d\zeta \quad (3)$$

transformation is shown in equation 4.

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \quad (4)$$

B. SHAPE FUNCTIONS

For an element with nodes numbered as in Figure 2, the shape functions obtained from Reference 1 are shown in equations 5.

Corner Nodes: 1, 3, 5, 7, 13, 15, 17, and 19

$$N_i = (1/8)(1 + \xi_o)(1 + \eta_o)(1 + \zeta_o)(\xi_o + \eta_o + \zeta_o - 2) \quad (5)$$

where $\xi_o = \xi_i\xi$ and $\xi_i = \pm 1$; similarly for η_o and ζ_o.

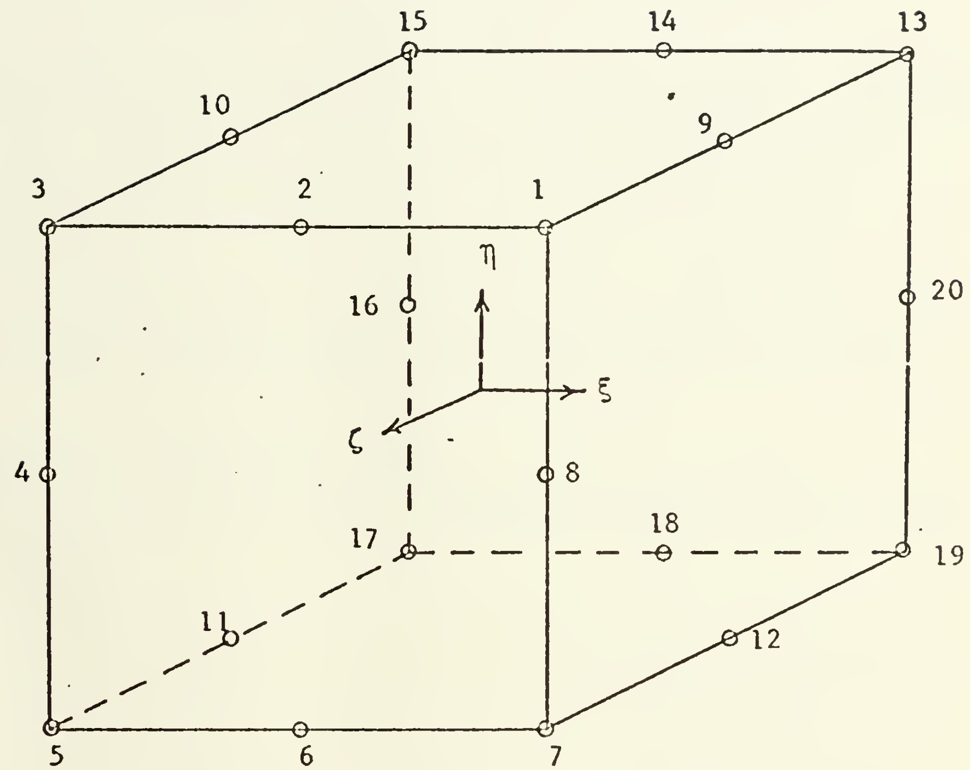


Figure 2. Element Nodal Point Numbering System

Midside Nodes: 2, 6, 14, and 18

$$N_i = (1/4)(1 - \xi^2)(1 + \eta_o)(1 + \zeta_o)$$

Midside Nodes: 4, 8, 16, and 20 (5)

$$N_i = (1/4)(1 - \eta^2)(1 + \xi_o)(1 + \zeta_o)$$

Midside Nodes: 9, 10, 11, and 12

$$N_i = (1/4)(1 - \zeta^2)(1 + \xi_o)(1 + \eta_o)$$

One should take note that if any other nodal numbering system with respect to ξ,η,ζ is used, the nodal point numbers that go with the above shape functions must be adjusted accordingly. It should also be noted that the program under analysis uses the system mentioned above.

C. COMPUTING STRESS AND STRAIN

Nodal point strains are computed using the following relationship.

$$\begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{xy} \\ \gamma_{xz} \\ \gamma_{yz} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (6)$$

where u, v, and w are obtained from equation 2. The derivatives of the displacements in global X, Y, Z coordinates are given by:

$$\begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} & \frac{\partial w}{\partial x} \\ \frac{\partial u}{\partial y} & \frac{\partial v}{\partial y} & \frac{\partial w}{\partial y} \\ \frac{\partial u}{\partial z} & \frac{\partial v}{\partial z} & \frac{\partial w}{\partial z} \end{bmatrix} = [J]^{-1} \begin{bmatrix} \frac{\partial u}{\partial \xi} & \frac{\partial v}{\partial \xi} & \frac{\partial w}{\partial \xi} \\ \frac{\partial u}{\partial \eta} & \frac{\partial v}{\partial \eta} & \frac{\partial w}{\partial \eta} \\ \frac{\partial u}{\partial \zeta} & \frac{\partial v}{\partial \zeta} & \frac{\partial w}{\partial \zeta} \end{bmatrix} \qquad (7)$$

where [J] is the Jacobian matrix (equation 4). The stresses are computed from strains by:

$$\begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{xz} \\ \tau_{yz} \end{bmatrix} = \begin{bmatrix} \lambda+2G & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2G & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2G & 0 & 0 & 0 \\ 0 & 0 & 0 & G & 0 & 0 \\ 0 & 0 & 00 & 0 & G & 0 \\ 0 & 0 & 0 & 0 & 0 & G \end{bmatrix} \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{xy} \\ \gamma_{xz} \\ \gamma_{yz} \end{bmatrix} \qquad (8)$$

in which: $G = \frac{E}{2(1+\nu)}$; $\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}$ (9)

E is Young's modulus, and ν is Poisson's ratio.

III. TRISOP: A QUADRATIC ISOPARAMETRIC FINITE ELEMENT PROGRAM

The stated objective of this work is to evaluate TRISOP, and give some insight into its practical use. Reference 1 contains the theory and methods used in the program, and Figure 3 is a simplified flow diagram.

A. INPUT DATA

The input data required by TRISOP are:

1. The number of elements
2. The total number of nodal points
3. The number of different materials
4. The block size for the large capacity solver
5. The number of nodal points with boundary conditions
6. The number of nodal points with concentrated loads
7. Element connectivity for each element
8. The coordinates of each nodal point
9. Young's Modulus, Poisson's Ratio for each material
10. Concentrated loads
11. Boundary conditions

The element connectivity is a correlation between the overall mesh numbering system and the standard numbering system used for each element.

When large problems are to be solved, preparing the connectivity and coordinate input is not a trivial task. Also, if there is a pressure or gravity load, the computations needed to convert into corresponding nodal point loads can be extremely time consuming. There is, however, a mesh generator program [2] that will solve this problem.

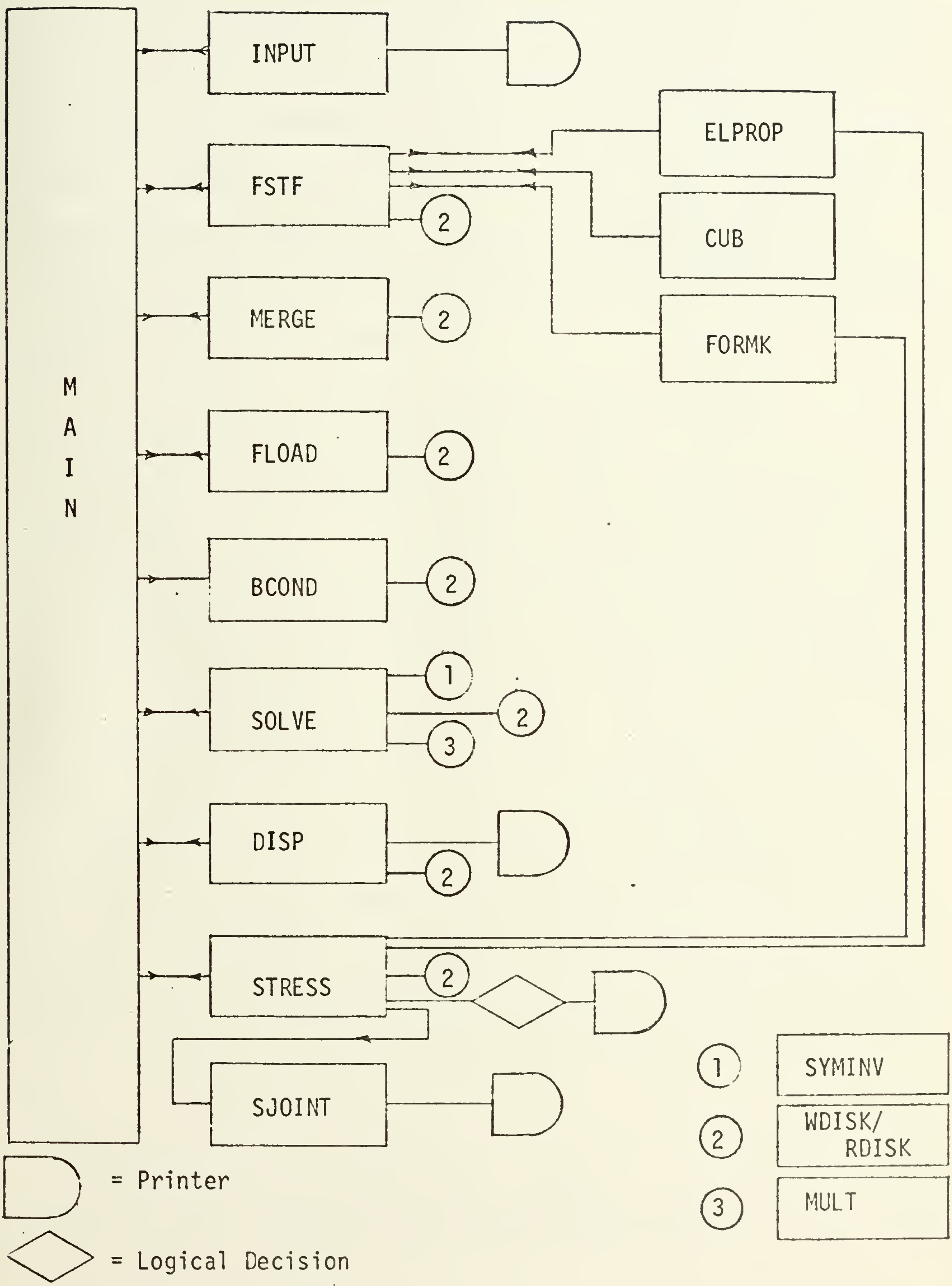


Figure 3. Functional Flow Diagram

With a minimum input, the mesh generator will compute the element connectivity, nodal point coordinates with input in cylindrical or rectangular coordinates, and nodal point loads for pressure or gravity loads. The mesh generator will print the output, draw a two dimensional picture of the mesh and punch data cards for TRISOP. The only other input to TRISOP that can require more than a few cards is the boundary conditions. The mesh generator will not produce boundary condition cards, but they are easily and quickly produced by hand.

B. TECHNIQUES FOR MOST EFFICIENT USE OF TRISOP

TRISOP, in its present form, has a constant core storage requirement. The two variables with the size of the problem are disk storage requirements and running time. To make the most efficient use of TRISOP, it is essential to take advantage of symmetry whenever possible. It is also essential to design the problem mesh in order to reduce the half-band width of the resulting system of equations to a minimum.

The half-band width is a function of the difference between the highest and lowest nodal point number in any element. To obtain the smallest band width one must start numbering on the face having the least number of elements along the side having the smallest number of elements as shown in Figure 4.

TRISOP does not give exact answers to a problem, but will converge asymptotically and monotonically to an exact

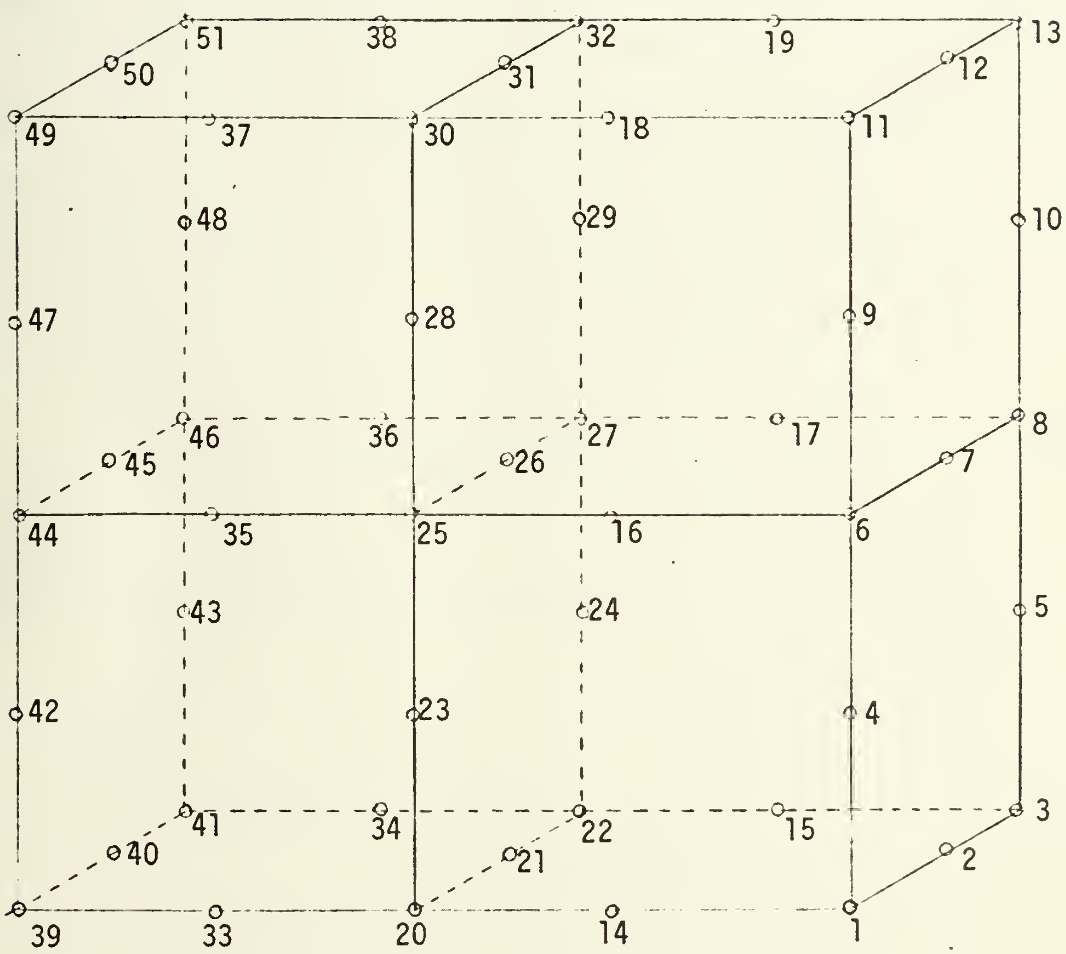


Figure 4. Numbering System for Minimum Half-Band Width

solution with uniform mesh refinement. To insure reliable results to a problem, a convergence study must be made. The convergence technique used here consists of plotting the displacement of a convenient nodal point, versus $1/N^2$ where N is the number of elements in the mesh. If three points plot in a straight line, the extrapolation to the origin is justified. This convergence technique is an adaptation of a technique developed by L. P. Richardson [3].

Richardson's technique was developed for extrapolating the results of central finite difference approximations where the truncation error is of the order (h^2), and h is the finite difference interval. When the truncation error is of the order (h^2) the extrapolated value is found by:

$$X_{extrap} = \frac{x_2 h_1^{\,2} - x_1 h_2^{\,2}}{h_1^{\,2} - h_2^{\,2}} \qquad (10)$$

This formula is plotted in Figure 5.

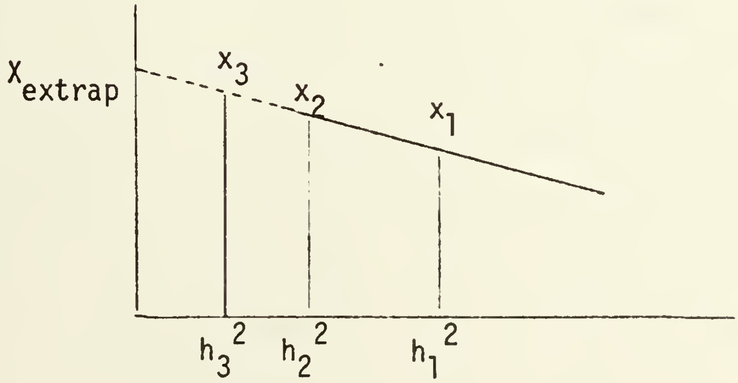


Figure 5. Richardson's Technique

If a third value, x_3, is found with a third interval, h_3, and x_3 plots on the extrapolation line, the extrapolated value is presumed to be exact [3].

There is no guarantee that the error in TRISOP is of order $1/N^2$, but this method of ascertaining convergence was adopted after some numerical experimentation. Since it is assumed that the solution approaches the exact solution asymptotically, if three deflections plot in a straight line it is assumed that extrapolation is valid. If they don't plot in a straight line there is no way of predicting the accuracy of the extrapolated result. This technique has worked quite well with many problems, but a better means of determining convergence is needed.

C. IMPROVEMENTS MADE TO TRISOP

Gaussian integration is used extensively in the computation of the element stiffness. Initially, four Gauss points in each of the three directions of a coordinate system were used in the solution. The subroutine in which this was accomplished was called CUB4. During a visit by Professor O. C. Zienkiewicz this author was told that using two Gauss points in the integration improved the solution. Subroutine CUB2, a two point integration subroutine, was substituted for CUB4 in TRISOP. This change yielded better results with a coarser mesh for all problems tested. The change from CUB4 to CUB2 also reduces significantly the integration CPU time for the calculation of element stiffness.

If an element is deformed into an extreme shape such as the element in Figure 6, the Jacobian becomes singular at points similar to 3, 10, and 15. Since equation 7 which uses the inverse of the Jacobian is used in computing stress and strain values, the computer algorithm fails at points 3, 10, and 15. To eliminate this singularity, TRISOP was modified to displace each node by a small distance away from its actual location at the time the Jacobian is formed for that node. The flow of computations is then uninterrupted, and the results for all other nodes are unaffected. The stresses and strains computed for nodes with a singularity are meaningless and should be eliminated from any further considerations.

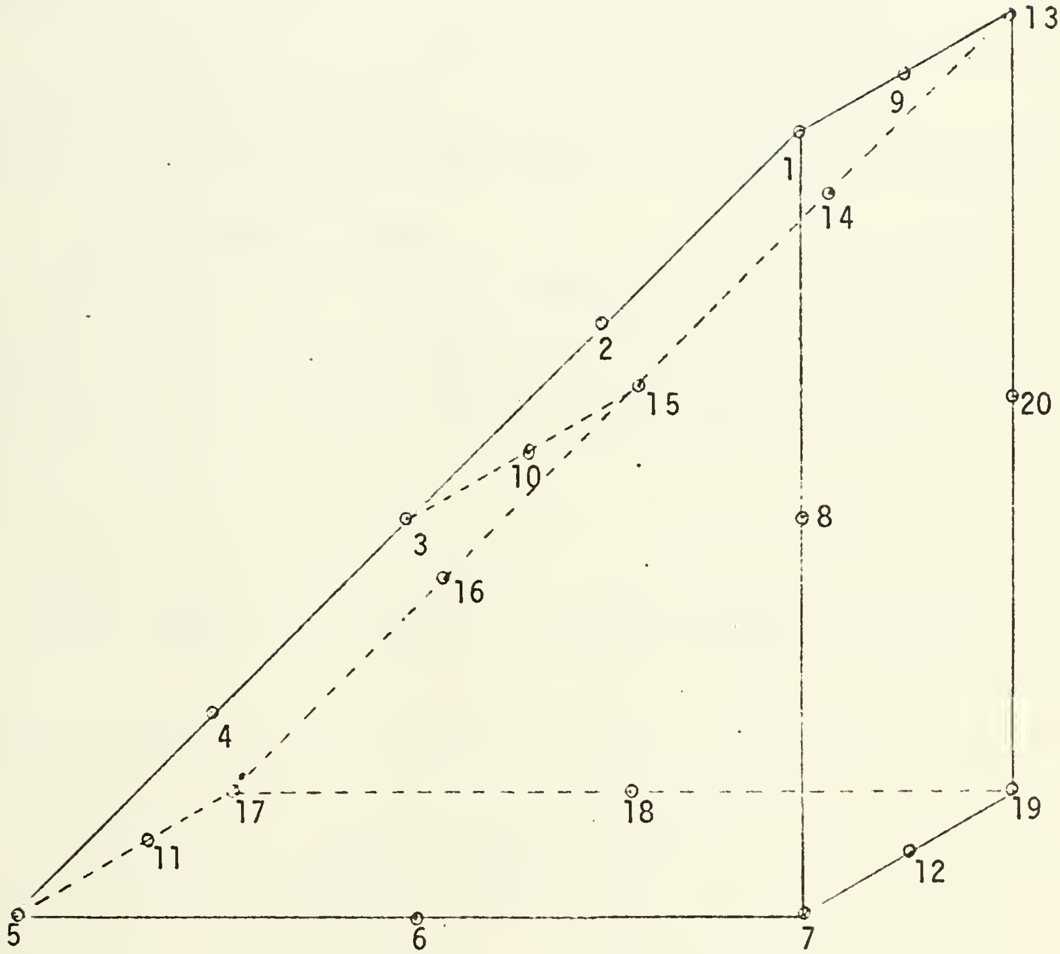


Figure 6. Degenerate Element

IV. SOLUTIONS TO CLASSICAL PROBLEMS USING TRISOP

Four classical problems were considered in the analysis of TRISOP. The four problems were a simply supported beam, a pinched disk, the Boussinesq problem, and a pinched cylinder.

A. SIMPLY SUPPORTED BEAM

The simply supported beam shown in Figure 7 was analyzed.

1. Classical Solution

The classical solution used was an Airy stress function developed using elasticity theory [4]. The Airy stress function for the problem under consideration is shown in equation 11.

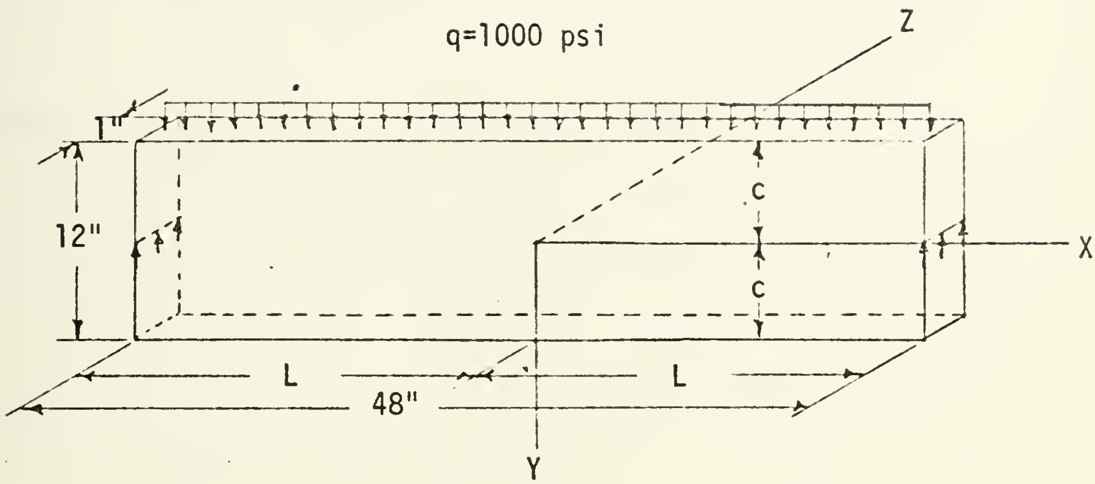
$$\phi = -\frac{q}{4}x^2 + \frac{3q}{8c}x^2y - \frac{q}{8c^3}x^2y^3 + \frac{q}{8c}\left(\frac{L^2}{c^2} - \frac{2}{5}\right)y^3 + \frac{q}{40c^3}y^5 \qquad (11)$$

Equation 11 satisfies equation 12 as is required

$$\frac{\partial^4\phi}{\partial x^4} + 2\frac{\partial^4\phi}{\partial x^2\partial y^2} + \frac{\partial^4\phi}{\partial y^4} = 0 \qquad (12)$$

for a valid Airy stress function in the absence of body forces. Stress values are found from equation 11 using equation 13.

$$\sigma_x = \frac{\partial^2\phi}{\partial y^2} \quad ; \quad \sigma_y = \frac{\partial^2\phi}{\partial x^2} \quad ; \quad \tau_{xy} = -\frac{\partial^2\phi}{\partial x\partial y} \qquad (13)$$



Boundary Conditions

$$(\tau_{xy})_{y=\pm c}=0; \quad (\sigma_y)_{y=+c}=0; \quad (\sigma_y)_{y=-c}=-q$$

At $x = \pm L$:

$$\int_{-c}^{c} \tau_{xy}dy = \pm qL; \quad \int_{-c}^{c} \sigma_x dy = 0; \quad \int_{-c}^{c} \sigma_{xy} dy = 0$$

Figure 7. Simply Supported Beam

Evaluating these relations yields equation 14

$$\sigma_x = \frac{q}{2I}(L^2-x^2)y + \frac{q}{2I}\left(\frac{2}{3}y^3 - \frac{2}{5}c^2y\right)$$

$$\sigma_y = -\frac{q}{2I}\left(\frac{1}{3}y^3 - c^2y + \frac{2}{3}c^3\right) \qquad (14)$$

$$\tau_{xy} = -\frac{q}{2I}(c^2 - y^2)x$$

where $I = \frac{2c^3}{3}$ is the moment of inertia of the beam. Since there are no body forces, such as gravity loading, this is the solution to both the plane stress and plane strain problems. The stress component, sigma x is an exact solution to the problem only if the distributed normal force shown in equation 15 is applied

$$\overline{x} = \pm \frac{3}{4}\frac{q}{c^3}\left(\frac{2}{3}y^3 - \frac{2}{5}c^2y\right) \qquad (15)$$

to each end of the beam. Since this force distribution has zero resultant force, and zero resultant moment, it can be concluded by Saint-Venant's Principle [4] that sigma x is exact at some distance from the ends of the beam. Saint-Venant's Principle assumes that localized forces in static equilibrium will give rise to localized stresses and strains.

The displacement of the center of the beam is found using equation 16. This displacement is greater

$$\delta = \frac{5}{24}\frac{qL^4}{EI}\left[1 + \frac{12}{5}\frac{c^2}{L^2}\left(\frac{4}{5} + \frac{\nu}{2}\right)\right] \qquad (16)$$

than the displacement found using elementary theory because there the assumption is made that cross sections of the beam remain plane during bending.

2. TRISOP Formulation

Considering the problem in Figure 7, there is a plane of symmetry parallel to the XY plane passing through the Z axis at .5 inches. All the points in this plane of symmetry were given a geometrical constraint which set the displacement component in the Z direction equal to zero. Only half of the beam from $0 \leq Z \leq .5$ was considered. For the nodal points in the XZ plane at $X = \pm L$, the displacement component in the Y direction was set equal to zero. Finally, for the nodal points on the YZ plane the displacement component in the X direction was set equal to zero. The distributed load of 1000 psi is modeled using a consistent load vector [5]. This vector dictates that the force on the face of each loaded element be divided as shown in Figure 8.

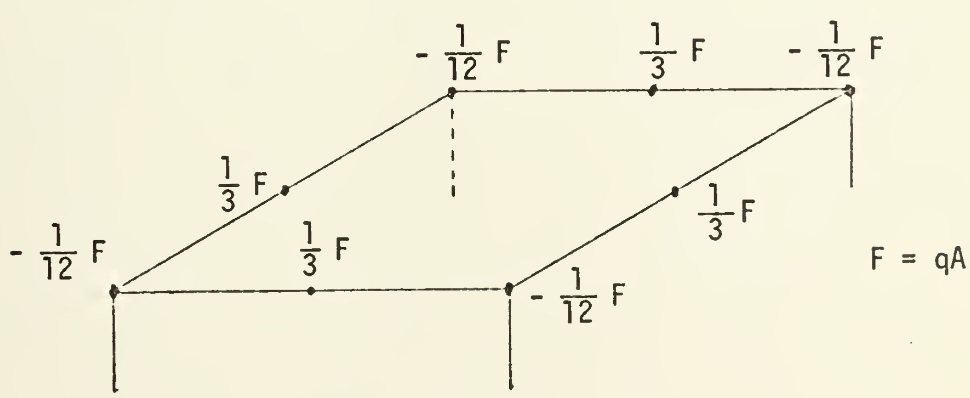


Figure 8. Consistent Load Vector

3. Results

A convergence study of the deflections at the center of the beam is shown in Figure 9 using the meshes shown in Figure 10. The results indicate that extrapolation to a mesh composed of an infinity of elements is justified. The classical value for the deflection at the center of the beam is 0.0180 inches. The extrapolated value given by TRISOP was 0.0188 inches, a value .0008 inches greater than the classical value. The classical solution, being a two dimensional solution, assumes that there is no deflection perpendicular to the XY plane. This makes the classical solution stiffer than the three dimensional solution. Figure 11 shows the displacement component in the Z direction in the XY plane given by TRISOP.

When TRISOP results for sigma x, sigma y, and tau xy were compared with the classical solution using nodes on the mesh face the values given by sigma x were within at least .1 percent of the classical value for all nodal points. On the other hand, the values for sigma y and tau xy were not nearly as good. It was noted, however, that the best results were obtained from interior nodes in the mesh. A 8x3x2 mesh was then analyzed using data from the mid plane parallel to the XY plane. The results from this analysis are shown in Figures 12, 13, and 14. Since the beam is symmetrical about the origin along the X axis, only the left end of the beam, $-L \leq X \leq 0$, is shown.

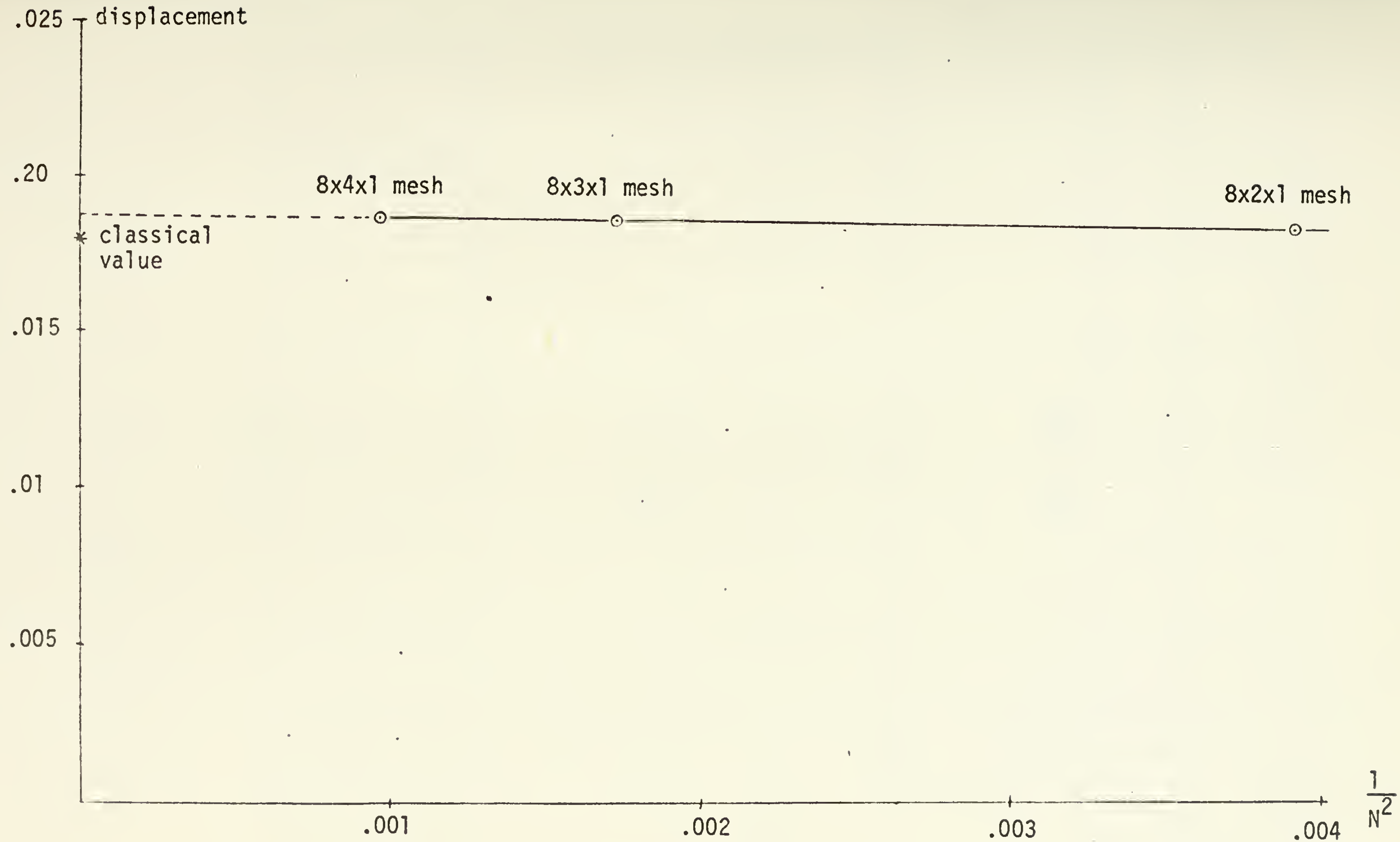
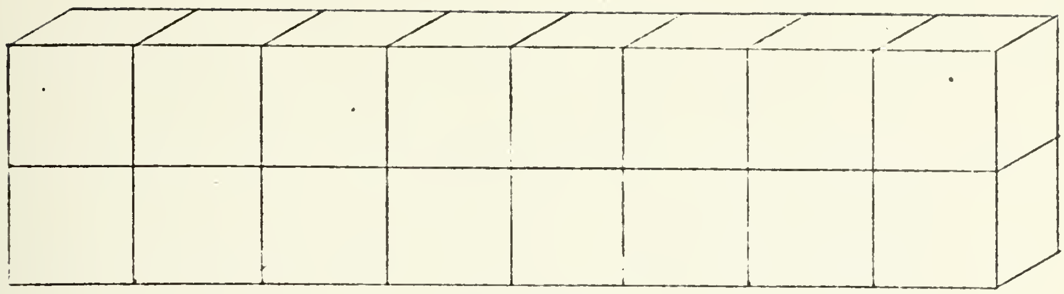
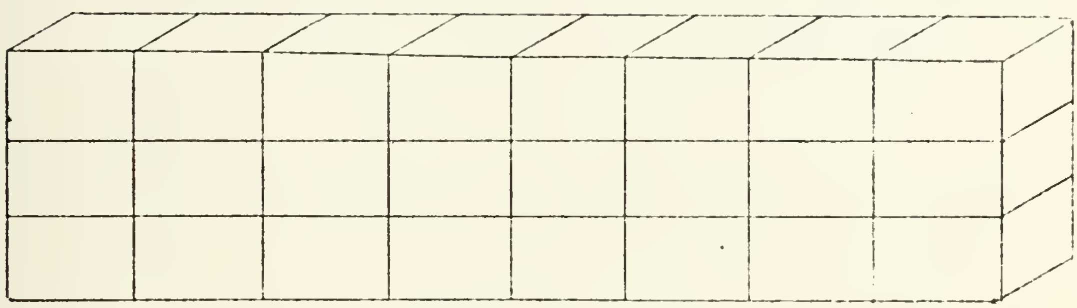


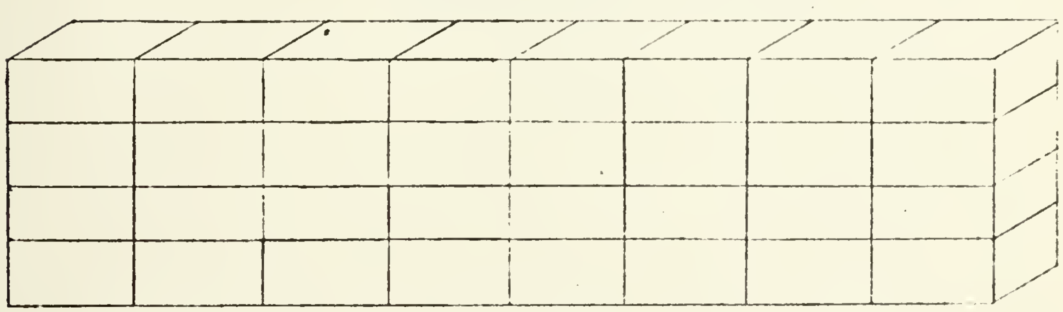
Figure 9. Simply Supported Beam Convergence Study



8x2x1 mesh

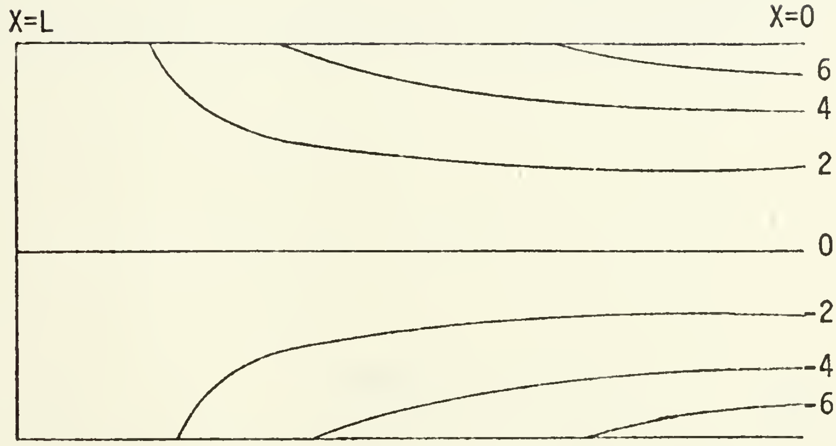


8x3x1 mesh



8x4x1 mesh

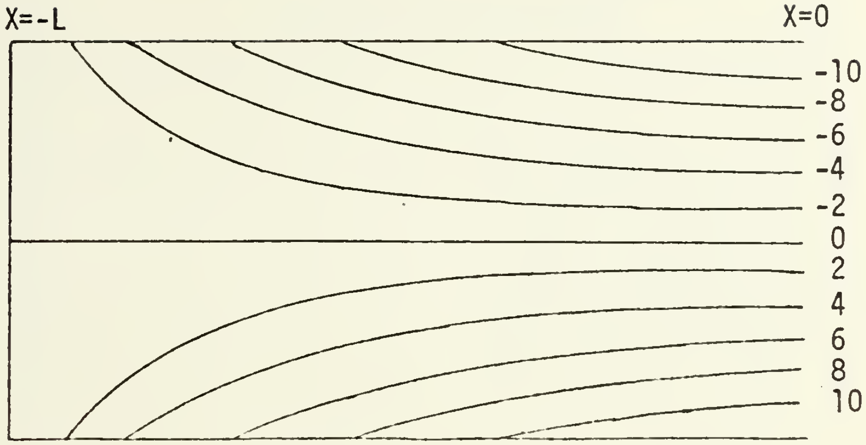
Figure 10. Simply Supported Beam Meshes



Displacements x 10^{-5} on XY Plane of the Simply Supported Beam.

Figure 11

Classical Solution



TRISOP Solution

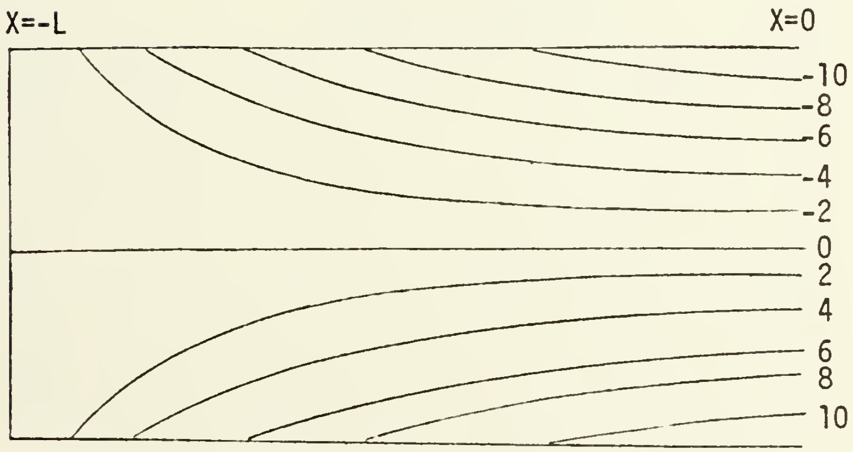
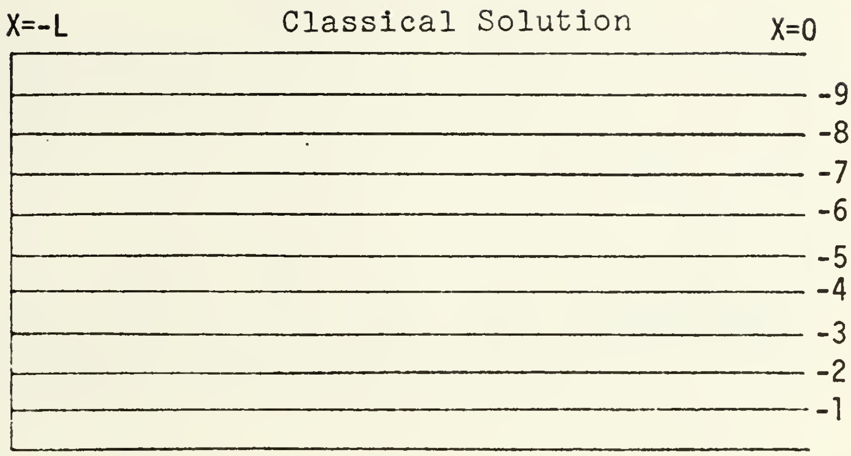


Figure 12. Simply Supported Beam

$\sigma_x \times 10^3$ psi



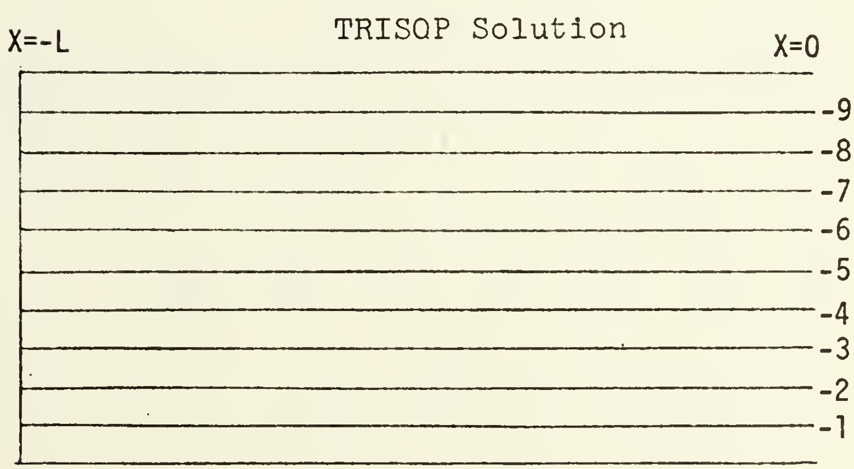
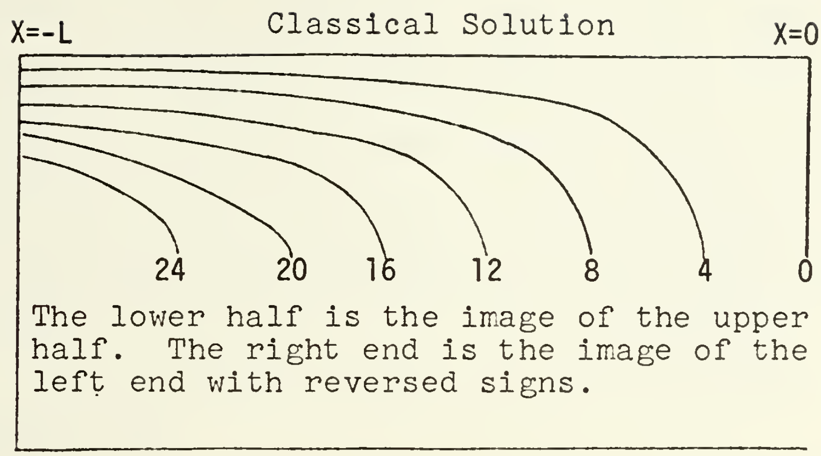


Figure 13. Simply Supported Beam

$\sigma_y \times 10^2$ psi



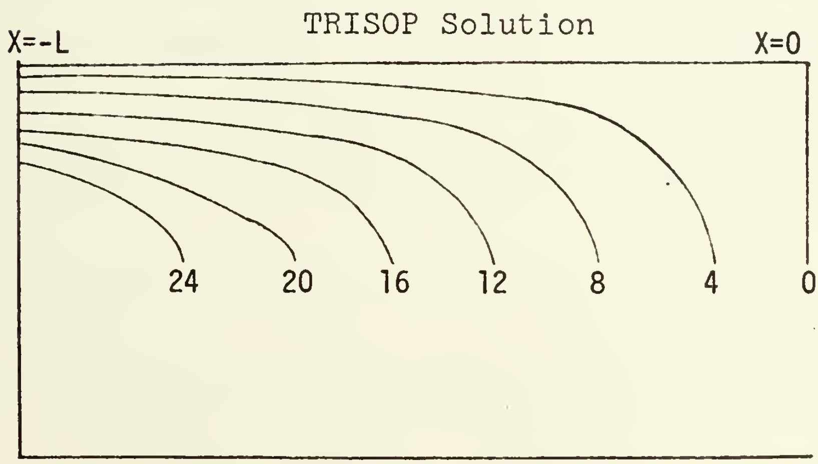


Figure 14. Simply Supported Beam

$\tau_{xy} \times 10^2$ psi

4. Conclusions

The results of this analysis indicate that much more reliable data is obtained from interior nodes. The poor results obtained for sigma y and tau xy on the mesh faces could indicate that the values obtained under a consistent load vector are better in planes where the loaded nodal points have four common elements as in the mid plane of the 8x3x2 mesh.

B. PINCHED DISK

A pinched disk with two equal and opposite forces acting on the diameter as shown in Figure 15 was analyzed.

1. Classical Solution

A classical two dimensional solution was devised using an Airy stress function by H. Hertz, and is discussed in detail in Reference 4. The Airy stress function, equation 17, gives the stress in terms of θ and r for each load as shown in Figure 16. The radial stress, the only

$$\phi = \frac{P}{\pi} r\theta \sin\theta \qquad (17)$$

non zero stress, is shown in Equation 18. Equation 18

$$\sigma_r = \frac{1}{r}\frac{\partial\phi}{\partial r} + \frac{1}{r^2}\frac{\partial^2\phi}{\partial\theta^2} = \frac{2P}{\pi}\frac{\cos\theta}{r} \qquad (18)$$

leads to the presence of an isotropic state of compression of intensity $2P/\pi d$ all around the radial surface of the disk. To free the boundary of this unwanted stress an isotropic tension equal to $2P/\pi d$ is added to the disk.

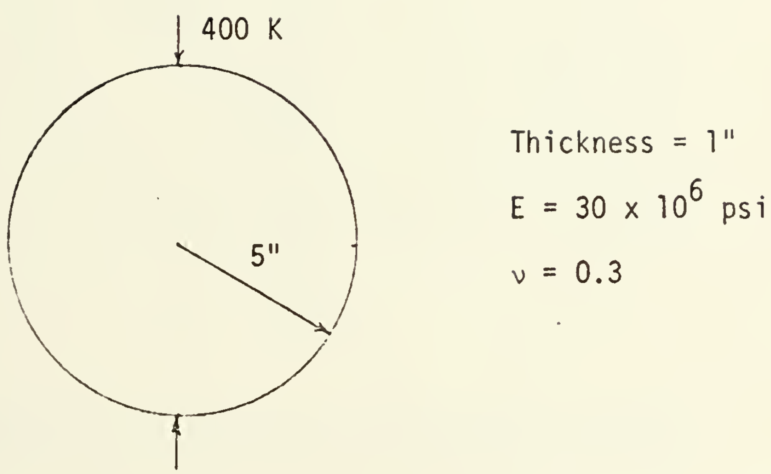


Figure 15. Pinched Disk

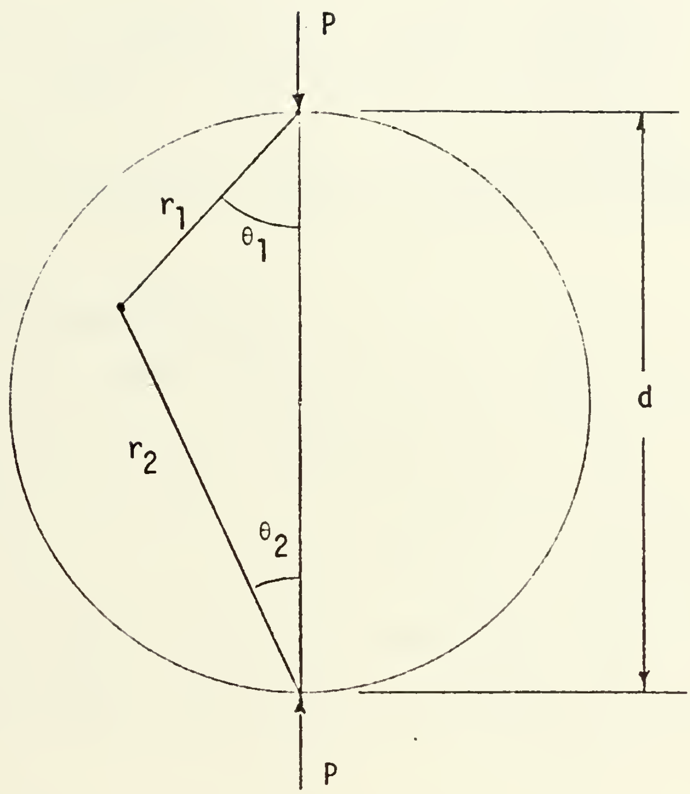


Figure 16. Pinched Disk Radial Stress

After converting sigma r_1 and sigma r_2 into X,Y coordinates, and adding the stresses from each load together, equations 19 result.

$$\sigma_x = -\sigma_{r_1} \sin^2 \theta_1 - \sigma_{r_2} \sin^2 \theta_2 + \frac{2P}{\pi d}$$

$$\sigma_y = -\sigma_{r_1} \cos^2 \theta_1 - \sigma_{r_2} \cos^2 \theta_2 + \frac{2P}{\pi d} \qquad (19)$$

$$\tau_{xy} = -\sigma_{r_1} \sin \theta_1 \cos \theta_1 + \sigma_{r_2} \sin \theta_2 \cos \theta_2$$

2. TRISOP Formulation

The problem was formulated for TRISOP as shown in Figure 17 using half the thickness of the disk. The boundary conditions were w = 0 in the XY plane, u = 0 in the YZ plane, and v = 0 in the XZ plane. Figure 18 shows one face of the 8x8x1 mesh. To model the load, a consistent load vector was again used [5] with values as shown in Figure 17.

3. Results

The deflections on the radial surface of the disk at a distance of 0.975 inches from the load were 0.01130 inches for the 4x4x1 mesh, and 0.01133 inches for the 8x8x1 mesh. Due to the small difference in these two deflections, it was assumed that the solution could be extrapolated with no significant error. The convergence plot is shown in Figure 19.

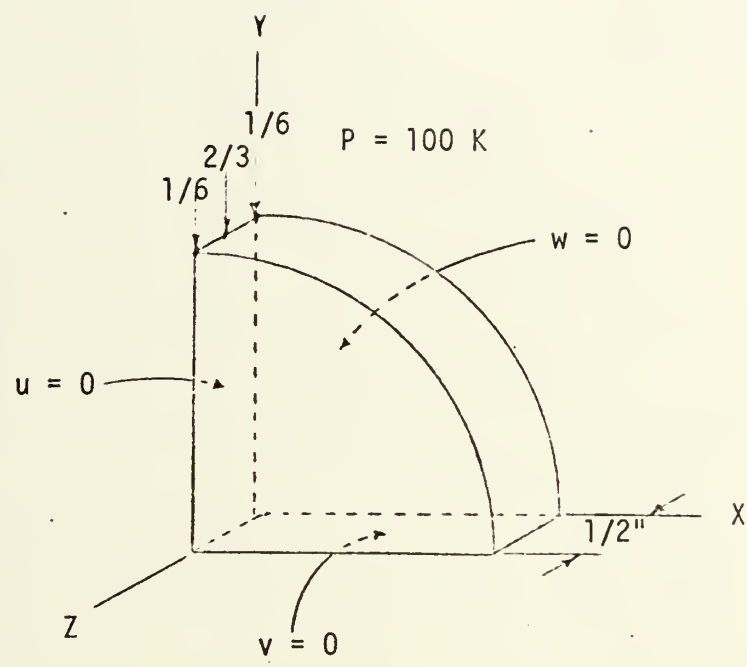


Figure 17. Pinched Disk, TRISOP Formulation

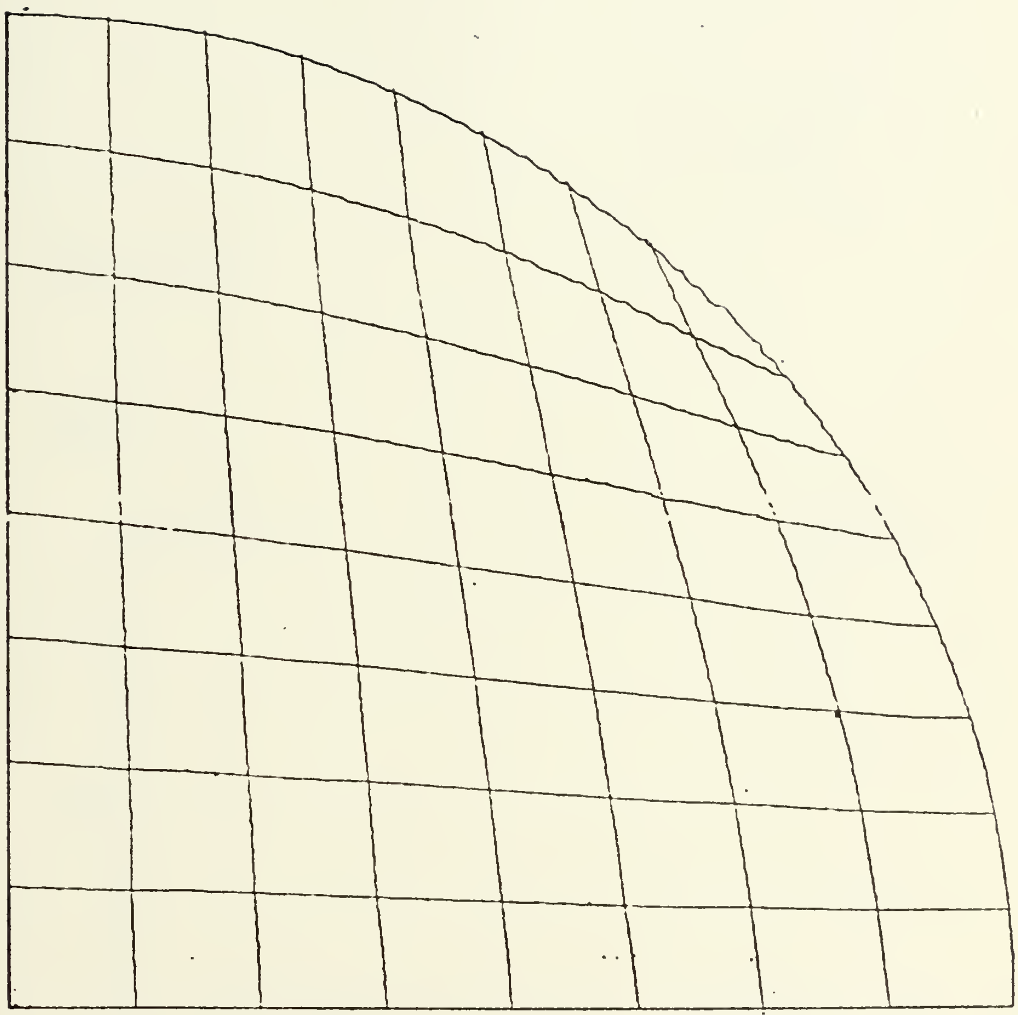


Figure 18. Pinched Disk 8x8x1 Mesh

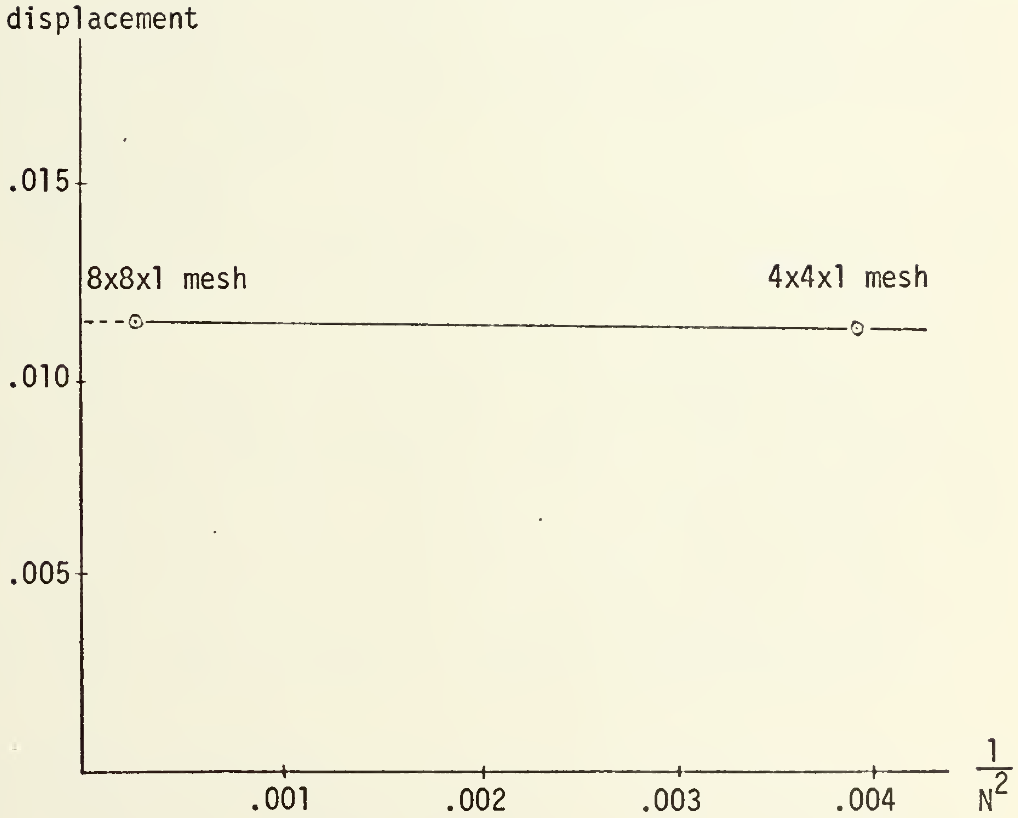


Figure 19. Pinched Disk Convergence Study

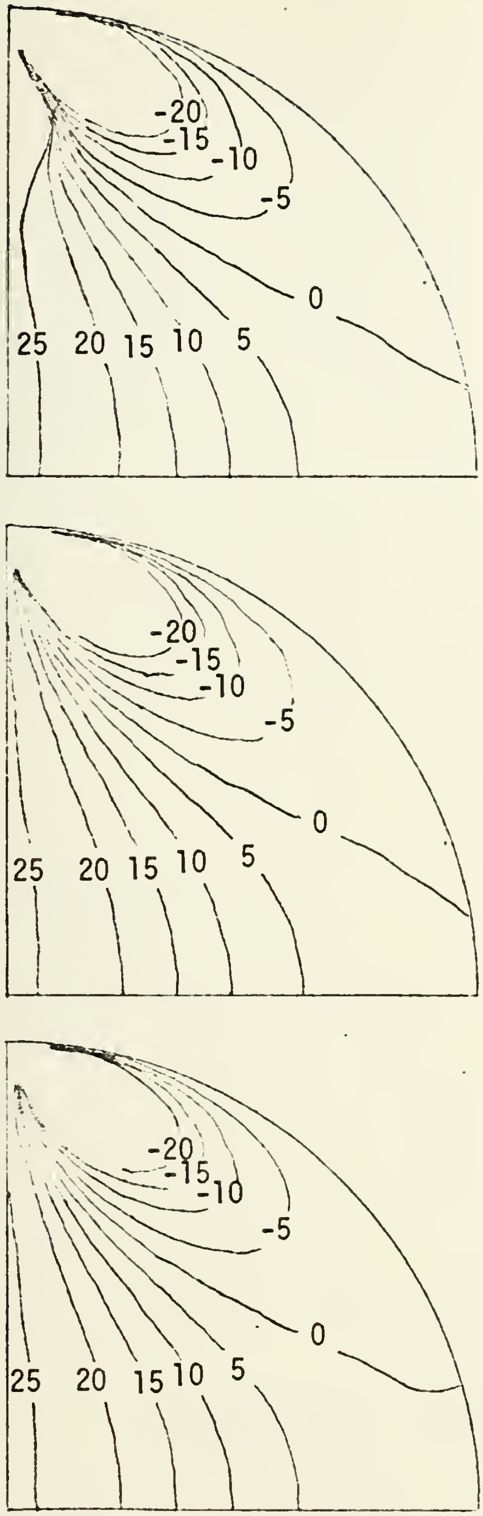
Contour graphs were drawn using data from the 8x8x1 mesh to compare TRISOP data with the classical solution obtained by Carlos Felippa [6]. Two graphs were drawn, one using mid side node data, and one using corner node data with the results shown in Figure 20. The plots for sigma y and tau xy using corner and mid side nodes are shown in Figures 21 and 22. All TRISOP plots shown used data from the constrained (XY) plane. However, the data on the free plane (Z = .5 inches), and the mid plane (Z = .25 inches) was virtually the same as that in the corresponding nodal points in the XY plane.

4. Conclusions

The TRISOP solution agrees with the classical solution for sigma y, tau xy, and the mid side node data for sigma x. The results for sigma x using corner node data indicates a much higher state of stress in the X direction directly under the load. It should be noted, however, that there are only two nodes that have values that are greatly in error, and one of those is directly under the load.

C. BOUSSINESQ PROBLEM

The Boussinesq problem consists of one concentrated load normal to the surface of a semi-infinite solid as shown in Figure 23.

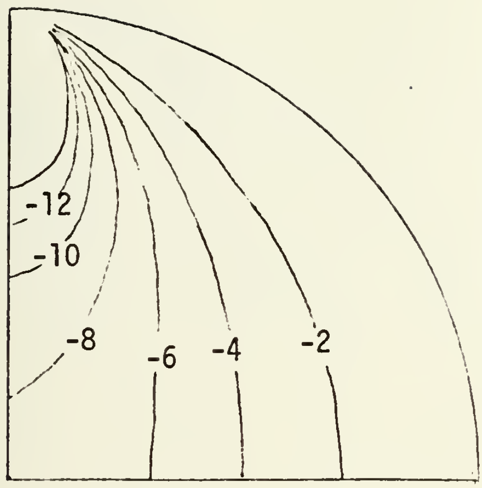


TRISOP Solution
using corner node data

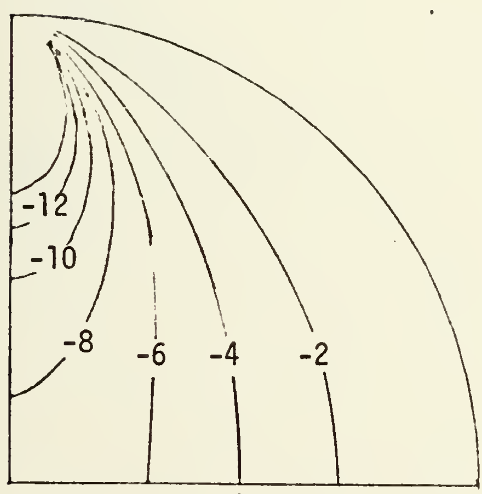
TRISOP Solution
using mid side node data

Classical Solution

Figure 20. Pinched Disk $\sigma_x \times 10^3$ psi

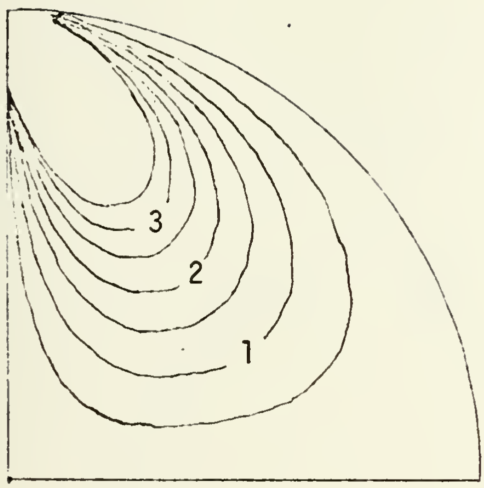


TRISOP Solution

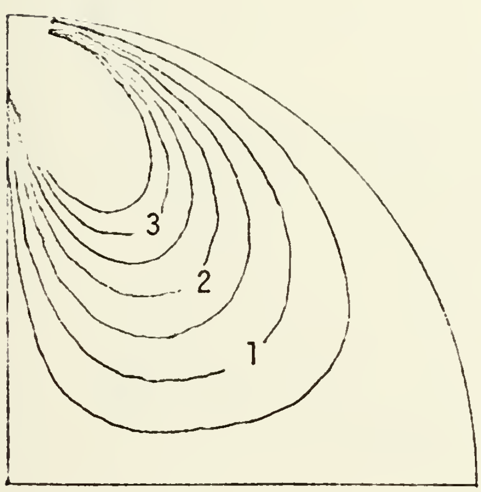


Classical Solution

Figure 21. Pinched Disk $\sigma_y \times 10^4$ psi



TRISOP Solution



Classical Solution

Figure 22. Pinched Disk τ_{xy} x 10^4 psi

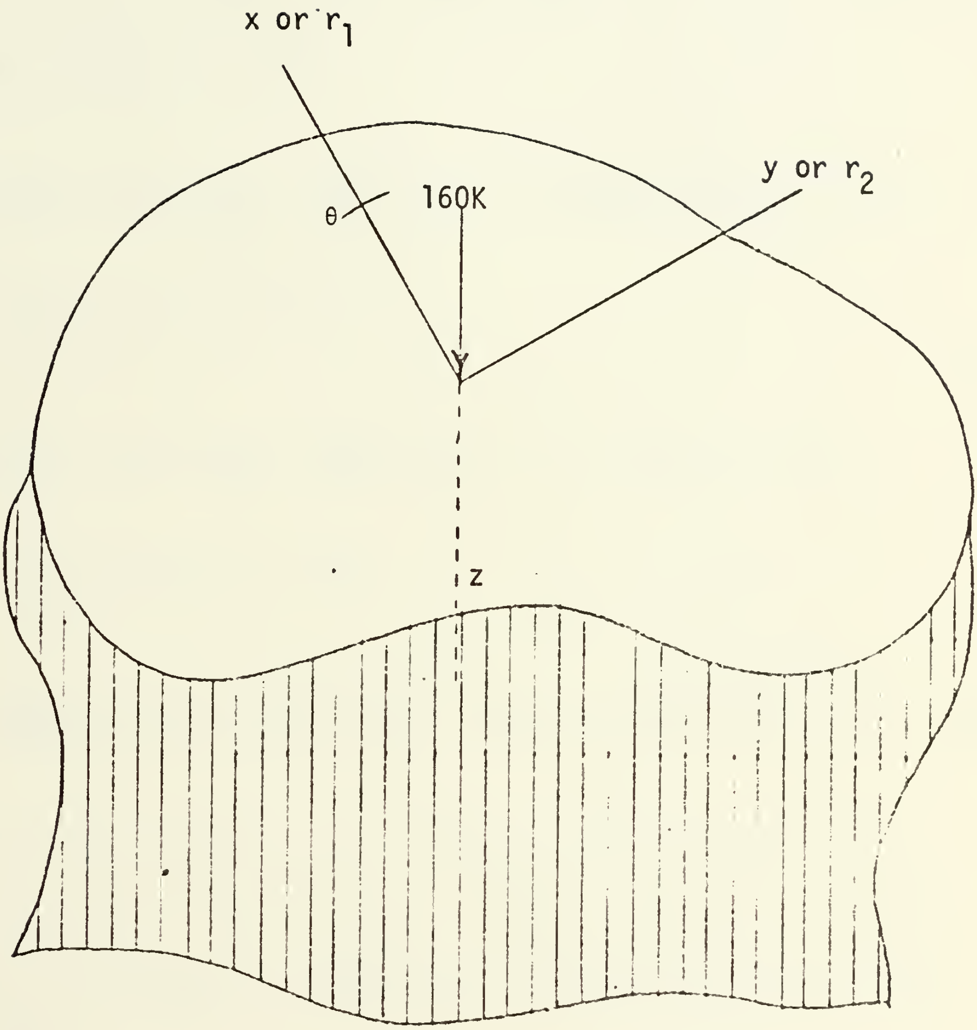


Figure 23. Boussinesq Problem

1. Classical Solution

The classical solution to the problem of a concentrated load normal to the surface of a semi-infinite solid was solved by J. Boussinesq [4] and yields the results shown in equations 20. The displacement in the Z direction is given in equation 21.

$$\sigma_r = \frac{P}{2\pi}\{(1-2\nu)[\frac{1}{r^2} - \frac{z}{r^2}(r^2+z^2)^{-1/2}] - 3r^2z(r^2+z^2)^{-5/2}\}$$

$$\sigma_z = -\frac{3P}{2\pi} z^3(r^2+z^2)^{-5/2} \qquad (20)$$

$$\sigma_\theta = \frac{P}{2\pi}(1-2\nu)\{-\frac{1}{r^2} + \frac{z}{r^2}(r^2+z^2)^{-1/2} + z(r^2+z^2)^{-3/2}\}$$

$$\tau_{rz} = -\frac{3P}{2\pi} rz^2(r^2+z^2)^{-5/2}$$

$$w = \frac{P}{2\pi E}[(1+\nu)z^2(r^2+z^2)^{-3/2} + 2(1-\nu^2)(r^2+z^2)^{-1/2}] \qquad (21)$$

In equations 20 and 21, E is Young's Modulus, and ν is Poisson's ratio. The stresses are in cylindrical coordinates with the axis of symmetry being the line of action of the load.

2. TRISOP Formulation

The TRISOP formulation took advantage of the double symmetry of the problem, and is shown in Figure 24. This figure also shows the dimensions of the 3x3x3 mesh. The 2x2x2, and 4x4x4 meshes were similarly constructed, and Figure 25 shows one face of these meshes to indicate the element dimensions.

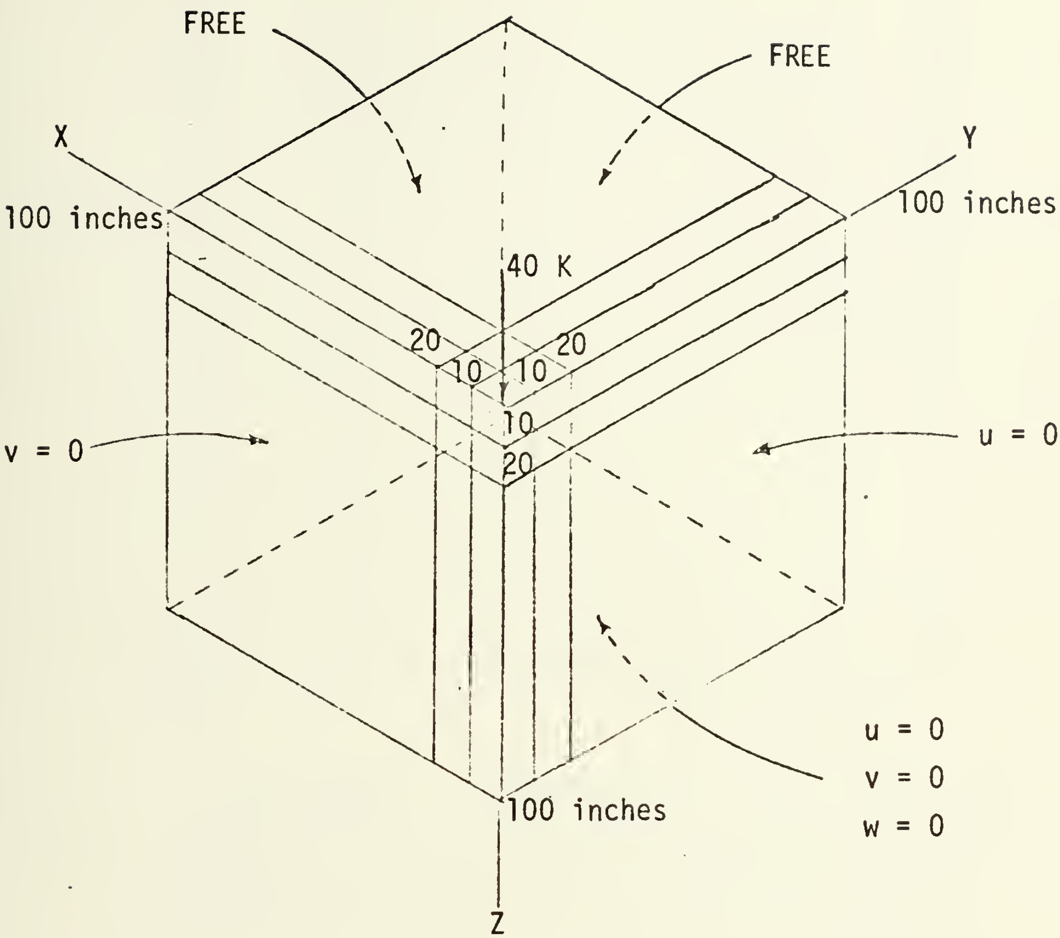


Figure 24. Boussinesq Problem TRISOP Formulation

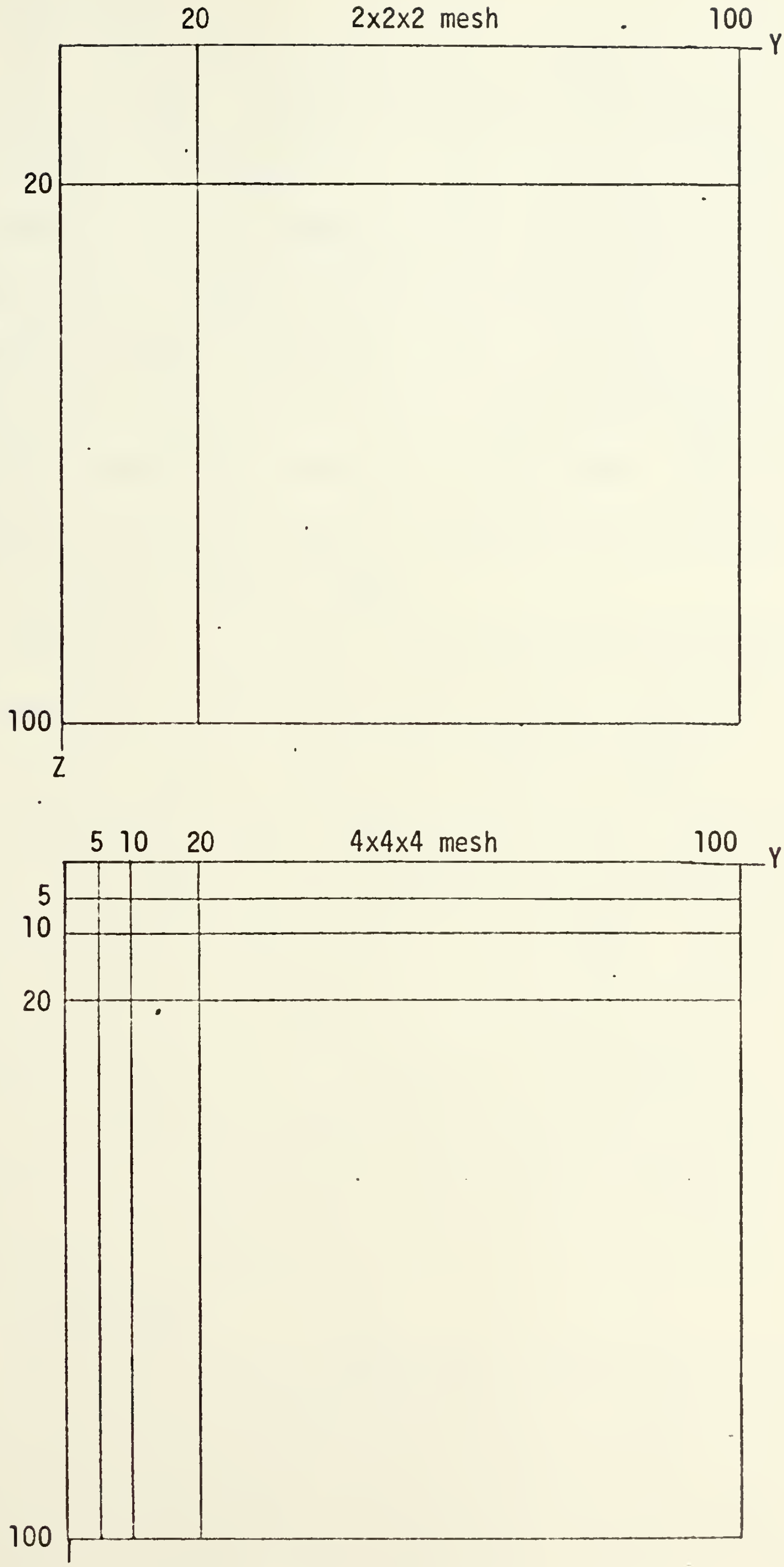


Figure 25. Boussinesq Meshes

3. <u>Results</u>

As evidenced by equation 21, the deflection at the load is not bounded. After examining deflections at other points in the three meshes, it could not be determined if extrapolation to a converged solution would be valid for the meshes used. Figure 26 shows the solution of the deflections in the Z direction from r = 0 to r = 100, at distances of 5, 10, and 20 inches from the surface using nodal point deflections from the 4x4x4 mesh. The TRISOP deflections follow the general contour of the classical solution, but are not as large.

Figures 27, 28, 29, and 30 show contour graphs of sigma r, sigma θ, sigma z and tau rz using classical and TRISOP data. A block twenty inches on a side is used for these graphs, because the area close to the load is of prime interest. The data used in generating the TRISOP graphs came from interior nodal points in the mesh. Since TRISOP computes results in rectangular coordinates, and the Boussinesq solution is in cylindrical coordinates, it was necessary to transform stress data from the interior nodes to cylindrical coordinates for meaningful comparison with the classical solution.

4. <u>Conclusions</u>

The discrepancy between the classical solution for displacements and the TRISOP solution is caused by the problem formulation used. In the real problem the deflections at Z = 100 inches are not zero. If the boundary

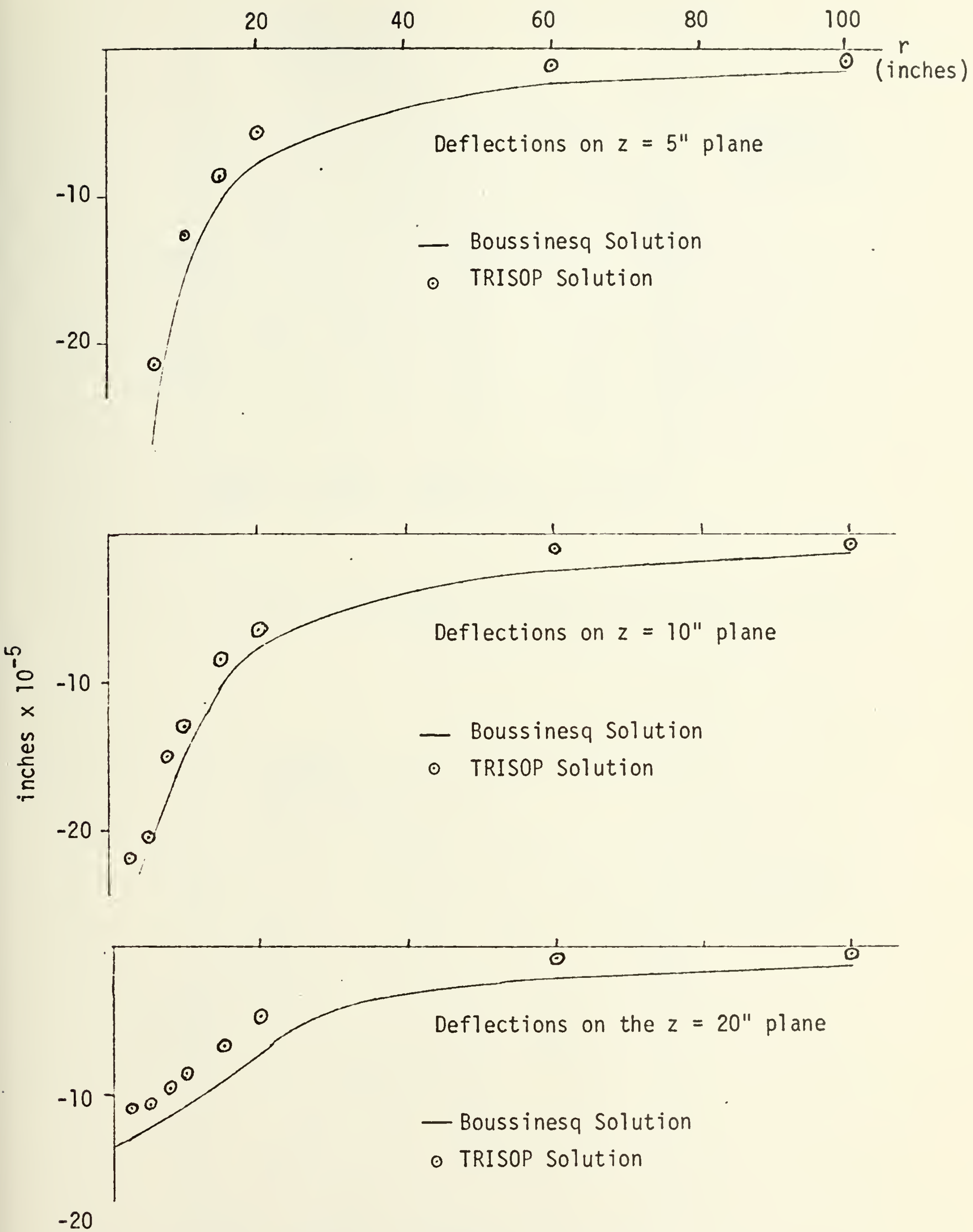
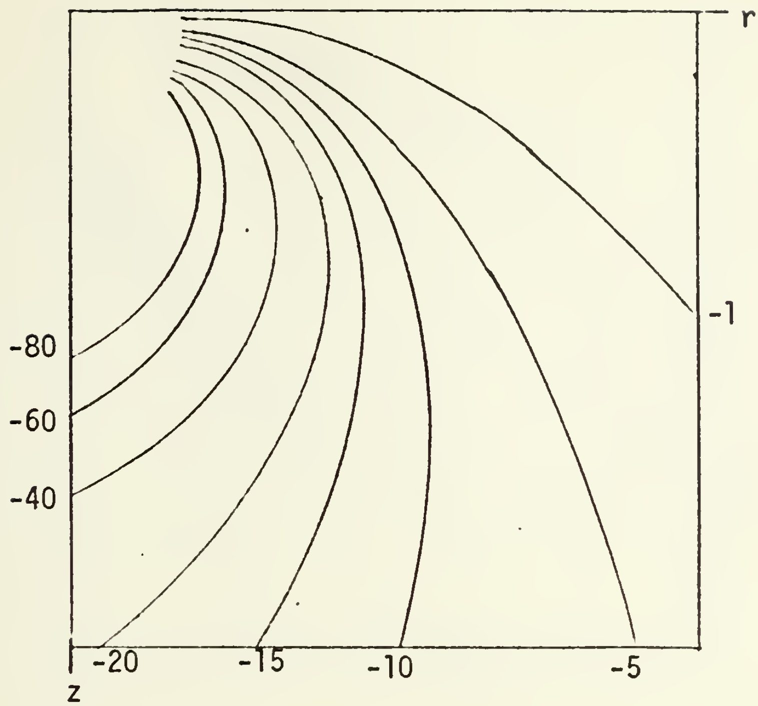
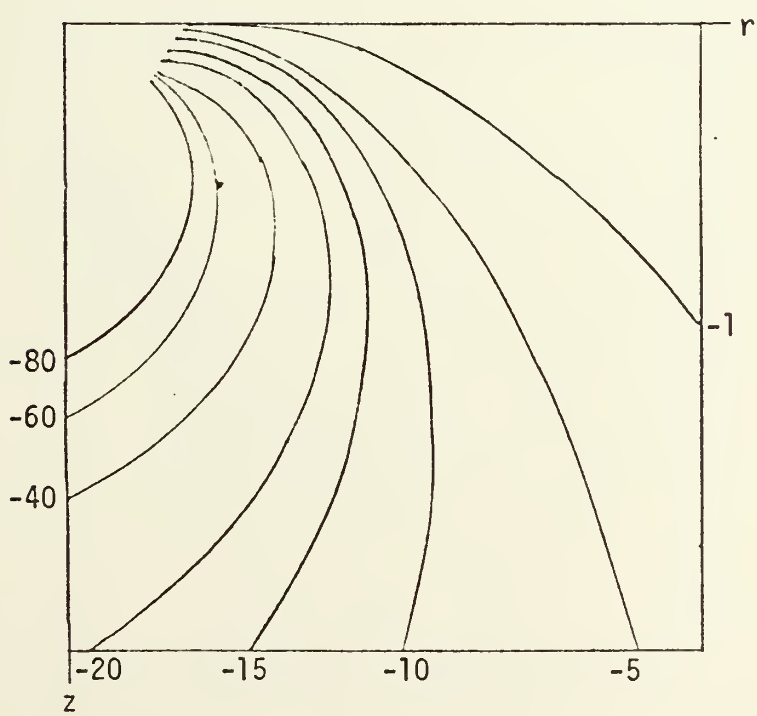


Figure 26. Boussinesq Deflections



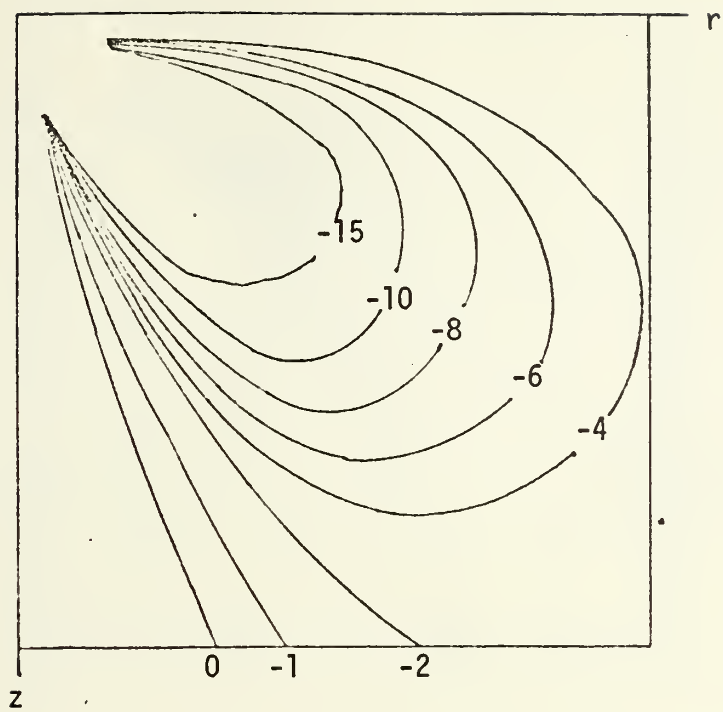
TRISOP Solution

σ_z x 10 psi with r and z from 0 to 20 inches



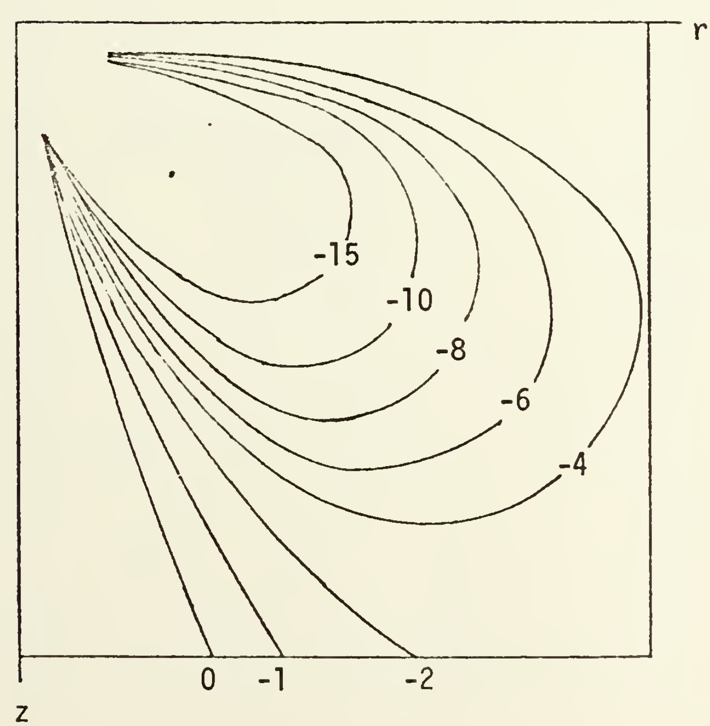
Classical Solution

Figure 27. Boussinesq Problem



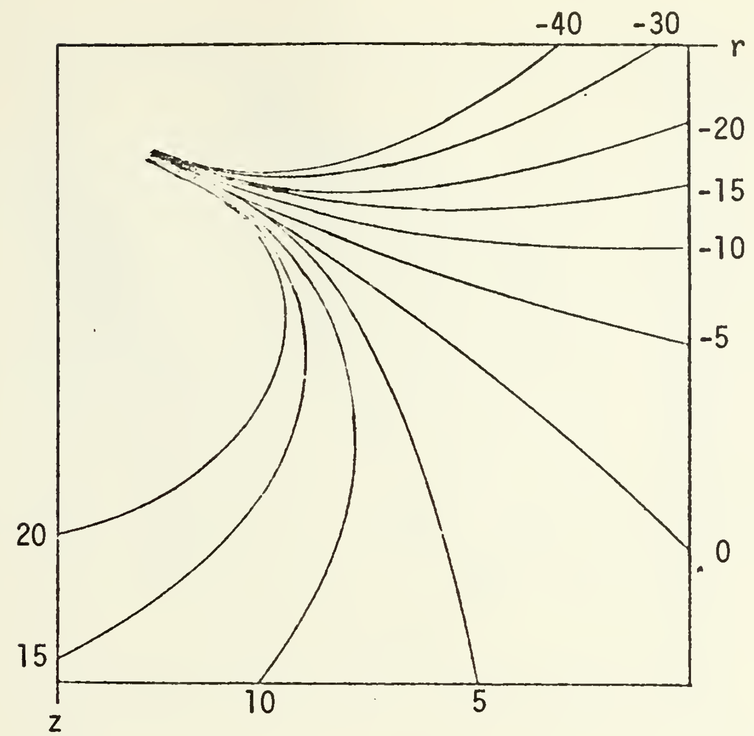
TRISOP Solution

σ_z x 10 psi with r and z from 0 to 20 inches



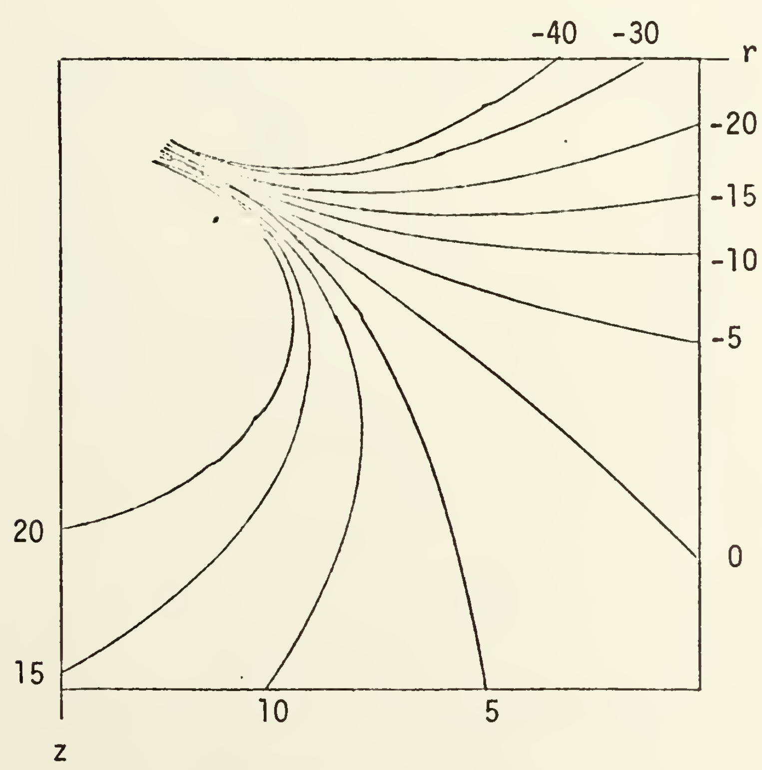
Classical Solution

Figure 28. Boussinesq Problem



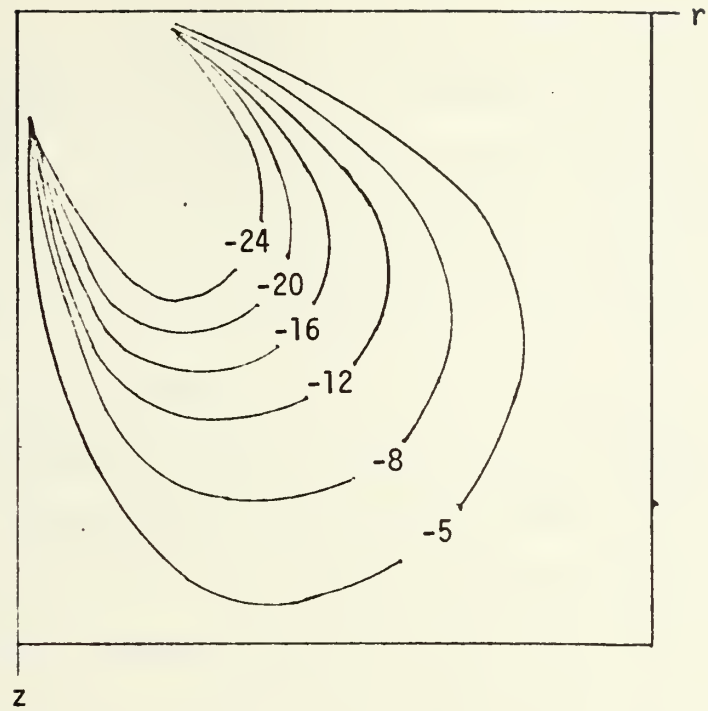
TRISOP Solution

σ_θ psi with r and z from 0 to 20 inches



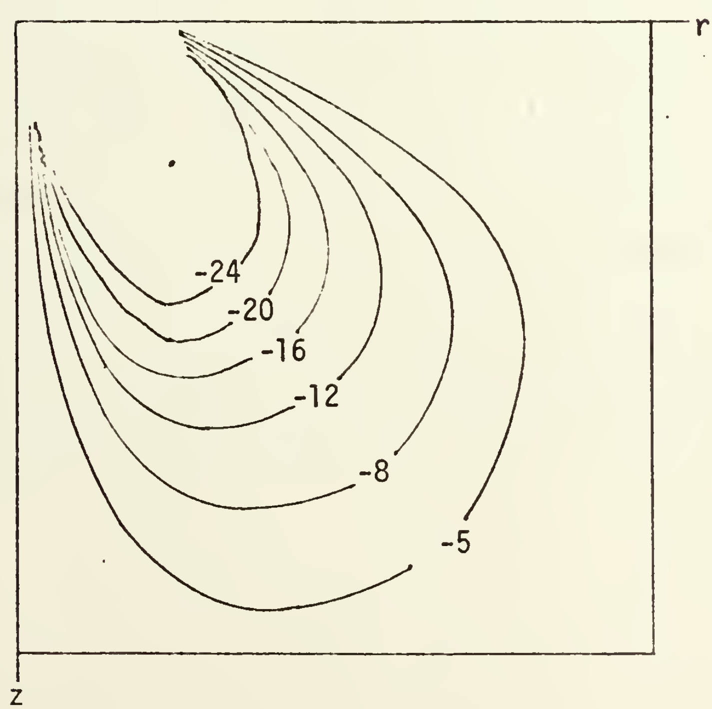
Classical Solution

Figure 29. Boussinesq Problem



TRISOP Solution

τ_{rz} x 10 psi with r and z from 0 to 20 inches



Classical Solution

Figure 30. Boussinesq Problem

conditions at the base of the mesh could have modeled the expected deflections, it is believed that the deflections elsewhere in the mesh would have been nearer to the expected values. The contour graphs indicate that TRISOP generates a solution in close agreement with the classical solution.

D. PINCHED CYLINDER

The pinched cylinder consists of a thin shell cylinder with a concentrated radial load as shown in Figure 31. D. E. Hanson [2] solved this problem using TRISOP with four Gauss point integration, and compared his results to a study mady be G. Cantin [7] using thin shell elements. The objective of this study was to determine if the three dimensional solution would give comparable results. Hanson made runs with meshes up to a 1x10x10, and obtained discouraging results. After discovering that two Gauss point integration might yield better results, Hanson's 1x2x2 and 1x4x4 meshes were rerun using two point integration. Table 1 shows a tabulation of Hanson's and Cantin's results including the reruns using two point integration. Figure 32 is a convergence study of the two meshes run with two point integration. The extrapolated result agrees with the fully converged solution given by Cantin.

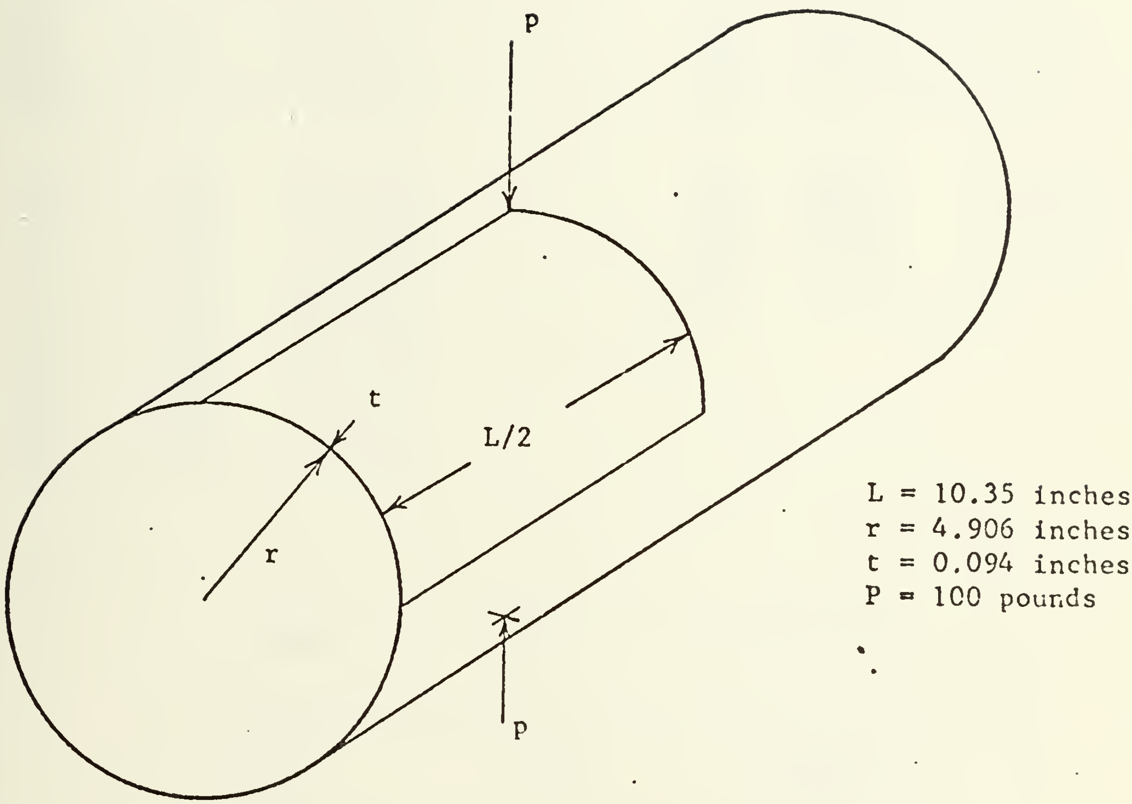


Figure 31. Pinched Cylinder

TABLE I

Displacement of a Pinched Cylinder at the Applied Load

Cantin

No. of Eq.	Mesh	Displac., in.
36	1x2	-0.0921
48	1x3	-0.1072
60	1x4	-0.1099
54	2x2	-0.0931
72	2x4	-0.1085
90	2x4	-0.1113
150	4x4	-0.1126
294	6x6	-0.1137
486	8x8	-0.1139
726	10x10	-0.1139

Hanson

No. of Eq.	Mesh	Displac., in.
153	1x2x2	-0.00177
465	1x4x4	-0.00375
945	1x6x6	-0.03002
1593	1x8x8	-0.08702
2409	1x10x10	-0.10057

Hanson with Two Point Integration

No. of Eq.	Mesh	Displac. in.
153	1x2x2	-0.1043
465	1x4x4	-0.1127

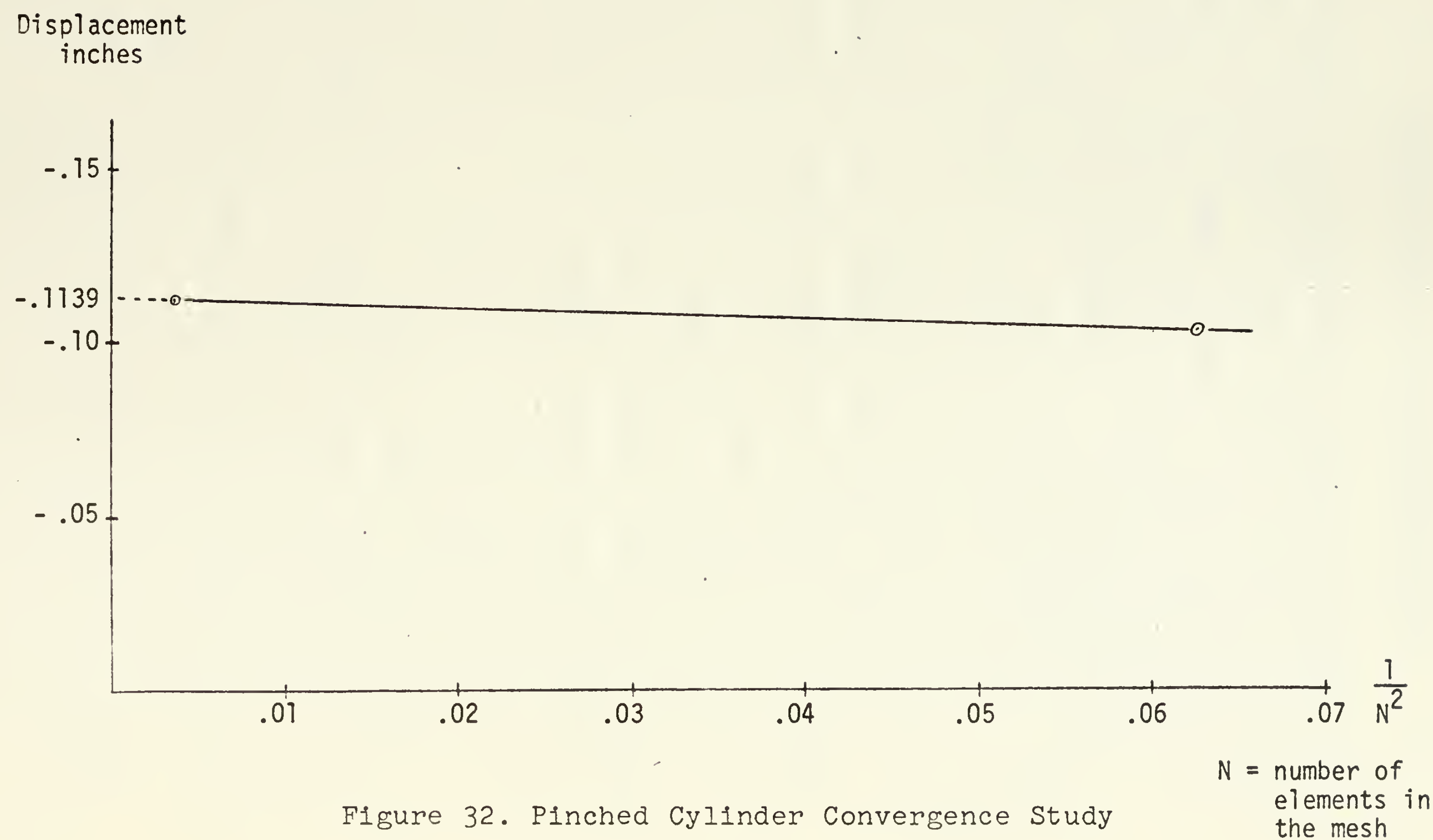


Figure 32. Pinched Cylinder Convergence Study

V. CONCLUSIONS AND RECOMMENDATIONS

The analysis of the problems considered indicates that TRISOP can be expected to give accurate results. However it was found in the simply supported beam and Boussinesq problems that data obtained for external surfaces were inaccurate. It would therefore be wise to avoid using data from surface nodes. If information near the surface of a solid is desired, a possible solution would be to have a very thin plane of elements at the surface, and use the data generated by the elements just below the surface. Another possible solution would be to modify the program to compute stress and strain at points other than nodal points such as the Gauss integration points to see if better results are obtained. However, this would increase the complexity of the program, because the coordinates of the points to be used would have to be specified or calculated and exhibited together with the stress values.

Concurrent with this research LCDR E. Leonidas was making improvements to TRISOP [8] that included reducing the program run time, and generalizing the boundary conditions to make it possible to specify boundary displacements. This last improvement will make it possible to model problems such as the Boussinesq problem more accurately.

TRISOP has the capability of receiving input mesh data in cylindrical coordinates, but this data is converted, and the output is given in rectangular coordinates. For problems

like the Boussinesq problem, output data in cylindrical coordinates would be a great help. TRISOP could be modified to accomplish this, or possibly another similar program could be produced to solve problems in cylindrical and spherical coordinates.

The change from four point to two point integration greatly reduces the computer time needed to solve a problem. This change also seems to give a more accurate solution as shown by the results of the pinched cylinder analysis. The computer time needed to solve large problems, however, can be excessive and is a very real limitation. Table II shows the CPU time for various problems run with both CUB2 and CUB4 on an IBM 360 computer.

The largest shortcoming of TRISOP is that the excessive amount of data produced is very time consuming and tedious to analyze. There are two possible solutions to this problem. The first would be to modify the program so that the output would be placed on contour graphs by the computer. The problem with this solution is that if the graphs do not plot, or graphs other than those asked for are needed after the run, the program would have to be rerun. In the case of a large problem, the cost in computer time could be excessive. What appears to be a far better solution would be to have the output placed on magnetic tape. The data could then be thoroughly analyzed using the computer with a minimum expenditure of computer time.

TABLE II

COMPUTER TIMES FOR TEST PROBLEMS RUN

Simply Supported Beam	FSTF (sec)	Merge (sec)	Solve (sec)	Overall Time (min, sec)	
1x2x8 CUB4	310.38	120.77	381.27	13	53.58
1x3x8 CUB4	462.31	174.73	539.26	20	06.58
1x4x8 CUB4	619.38	353.28	1116.13	35	23.07
2x3x8 CUB4	897.62	507.29	1357.63	47	44.98
1x2x8 CUB2	48.49	118.35	386.38	9	33.73
1x3x8 CUB2	72.01	174.58	534.60	13	31.32
1x4x8 CUB2	94.47	347.45	1112.54	26	29.13
2x3x8 CUB2	135.66	507.26	1357.67	35	01.59
Pinched Disk					
1x4x4 CUB4	315.05	175.82	522.73	17	09.66
1x5x5 CUB4	470.96	285.88	835.18	26	57.49
1x6x6 CUB4	675.46	566.05	1694.84	49	25.42
1x8x8 CUB4	1195.18	1433.38	4217.03	115	33.68
1x4x4 CUB2	47.38	173.84	502.34	12	19.66
1x5x5 CUB2	72.47	288.26	832.60	20	18.46
1x6x6 CUB2	102.60	566.32	1688.59	39	44.81
1x8x8 CUB2	179.07	1433.12	4216.28	98	34.08
Boussinesq Problem					
2x2x2 CUB2	25.19	90.43	249.43	6	12.41
3x3x3 CUB2	76.61	432.22	1062.09	26	29.91
4x4x4 CUB2	178.31	1853.68	4309.12	107	07.95
Pinched Cylinder					
1x4x4 CUB4	317.60	152.09	646.10	18	53.85
1x4x4 CUB2	44.16	147.75	497.40	12	42.00

```
C              TRISOP:  A THREE DIMENSIONAL FINITE ELEMENT PROGRAM
C        *****************************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
     1,NCLD,NPBC,NSB
      COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
      COMMON /B1/ ELDAT(10,3),CLOAD(99,3)
      COMMON /B2/ COORD(850,3),COREL(20,3),ELAST(6,6)
   10 DO 100 I=1,850
      DO 100 J=1,3
  100 COORD(I,J)=0.0D0
      DO 200 I=1,200
      DO 200 J=1,20
  200 NCON(I,J)=0
      CALL SETIME
      CLOCK=ITIME(0)*0.01
      CALL INPUT
      CALL GETIME(IET)
      CPUTM=IET*0.000026
      CLOCK=ITIME(0)*0.01-CLOCK
      WRITE(6,8000) CLOCK,CPUTM
      CALL SETIME
      CLOCK=ITIME(0)*0.01
      CALL FSTF
      CALL GETIME(IET)
      CPUTM=IET*0.000026
      WRITE(6,1000)
 1000 FORMAT(5X,'***** FSTF ****')
      CLOCK=ITIME(0)*0.01-CLOCK
      WRITE(6,8000) CLOCK,CPUTM
      CALL SETIME
      CLOCK=ITIME(0)*0.01
      CALL MERGE
      CALL GETIME(IET)
      CPUTM=IET*0.000026
      WRITE(6,2000)
 2000 FORMAT(5X,'***** MERGE ****')
      CLOCK=ITIME(0)*0.01-CLOCK
      WRITE(6,8000) CLOCK,CPUTM
      CALL SETIME
      CLOCK=ITIME(0)*0.01
      CALL FLOAD
      CALL GETIME(IET)
      CPUTM=IET*0.000026
      WRITE(6,3000)
 3000 FORMAT(5X,'***** FLOAD *****')
      CLOCK=ITIME(0)*0.01-CLOCK
```

```
      WRITE(6,8000) CLOCK,CPUTM
      CALL SETIME
      CLOCK=ITIME(0)*0.01
      CALL BCOND
      CALL GETIME(IET)
      CPUTM=IET*0.000026
      WRITE(6,3500)
 3500 FORMAT(5X,'***** BCOND *****')
      CLOCK=ITIME(0)*0.01-CLOCK
      WRITE(6,8000) CLOCK,CPUTM
      CALL SETIME
      CLOCK=ITIME(0)*0.01
      CALL SOLVE
      CALL GETIME(IET)
      CPUTM=IET*0.000026
      WRITE(6,4000)
 4000 FORMAT(5X,'***** SOLVE *****')
      CLOCK=ITIME(0)*0.01-CLOCK
      WRITE(6,8000) CLOCK,CPUTM
      CALL SETIME
      CLOCK=ITIME(0)*0.01
      CALL DISP
      CALL GETIME(IET)
      CPUTM=IET*0.000026
      WRITE(6,5000)
 5000 FORMAT(5X,'*****  DISP *****')
      CLOCK=ITIME(0)*0.01-CLOCK
      WRITE(6,8000) CLOCK,CPUTM
      CALL SETIME
      CLOCK=ITIME(0)*0.01
      CALL STRESS(1)
      CALL GETIME(IET)
      CPUTM=IET*0.000026
      WRITE(6,6000)
 6000 FORMAT(5X,'***** STRESS *****')
      CLOCK=ITIME(0)*0.01-CLOCK
      WRITE(6,8000) CLOCK,CPUTM
 8000 FORMAT(5X,7H TIME =,1F7.2,8H SECONDS,6H(CPU =,1F7.2,9H SECONDS)//)
      GO TO 10
      END

      SUBROUTINE INPUT
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
     1,NCLD,NPBC,NSB
      COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
      COMMON /B1/ ELDAT(10,3),CLOAD(99,3)
```

```
      COMMON /B2/ COORD(850,3),COREL(20,3),ELAST(6,6)
      DIMENSION TITLE (10),FMT (10)
      PI=3.14159265359
      READ(5,1000) TITLE
 1000 FORMAT(10A8)
      WRITE(6,1000) TITLE
      READ(5,2000) NEL,NDPT,NMAT,NS,NPBC,NCLD
 2000 FORMAT(6I10)
      IF(NEL.LT.0) STOP
      WRITE(6,3000) NEL,NDPT,NMAT,NS,NPBC,NCLD
 3000 FORMAT(//,
     15X,'NEL....TOTAL NUMBER OF ELEMENTS.........................',I3/,
     A5X,'        (MAXIMUM IS 200)',/,
     25X,'NDPT...TOTAL NUMBER OF NODAL POINTS...................',I3/,
     A5X,'        (MAXIMUM IS 850)',/,
     35X,'NMAT...NUMBER OF DIFFERENT MATERIALS..................',I3/,
     A5X,'        (MAXIMUM IS  10)',/,
     55X,'NS.....BLOCK SIZE FOR THE LARGE CAPACITY SOLVER.......',I3/,
     65X,'NPBC...NUMBER OF NODAL POINTS WITH BOUND. COND........',I3/,
     A5X,'        (MAXIMUM IS 200)',/,
     75X,'NCLD...NUMBER OF NODAL POINTS WITH CONC. LOAD.........',I3/,
     A5X,'        (MAXIMUM IS  50)',/)
      IF(NEL.GT.100) STOP
      IF(NDPT.GT.850) STOP
      IF(NPBC.GT.500) STOP
      IF(NCLD.GT.99) STOP
      NPEL=20
      NDF=3
      NEQ=NDPT*NDF
      NCOUNT=NS*NS
      NST=NPEL*NDF
      NSTF=NST*NST
      NSB=6*NST
      NN=(NEQ+NS-1)/NS
      WRITE(6,1100)
 1100 FORMAT(//)
      READ(5,1000) TITLE
      READ(5,1000) FMT
      WRITE(6,1000) TITLE
      WRITE(6,1100)
      DO 100 I=1,NEL
      READ(5,FMT) (NCON(I,J),J=1,21)
  100 WRITE(6,FMT) (NCON(I,J),J=1,21)
      NBAND=0
      DO 200 I=1,NEL
      NPELM=NPEL-1
      DO 200 J=1,NPELM
      JP=J+1
```

```
      DO 200 K=JP,NPEL
  200 NBAND=MAXO(NBAND,IABS(NCON(I,J)-NCON(I,K)))
      NBAND=(NBAND+1)*NDF
      MM=(NBAND+2*(NS-1))/NS
       WRITE(6,4000) NEQ,NBAND,NN,MM,NCOUNT,NSTF
 4000 FORMAT(//,
     15X,'NEQ....TOTAL NUMBER OF EQUATIONS........................',I5/,
     25X,'NBAND..HALF-BAND WIDTH OF THE SYSTEM....................',I5/,
     35X,'NN.....NUMBER OF BLOCKS PER COLUMN......................',I5/,
     45X,'MM.....NUMBER OF BLOCKS PER ROW.........................',I5/,
     55X,'NCOUNT.NUMBER OF COEFFICIENTS PER BLOCKS...............',I5/,
     65X,'NSTF...NUMBER OF COEFFICIENTS PER ELEMENT..............',I5/)
      WRITE(6,1100)
      READ (5,1000) TITLE
      WRITE(6,1000) TITLE
      WRITE(6,1100)
      READ(5,1000) FMT
      DO 300 I=1,NDPT
      READ (5,FMT) IEL,(COORD(IEL,J),J=1,NDF),KND
      IF(KND.EQ.0) GO TO 301
      PHI=PI*COORD(IEL,2)/180.0D0
      X=COORD(IEL,1)*DCOS(PHI)
      Y=COORD(IEL,1)*DSIN(PHI)
      COORD(IEL,1)=X
      COORD(IEL,2)=Y
  301 CONTINUE
  300 WRITE(6,FMT) IEL,(COORD(IEL,J),J=1,NDF),KND
      WRITE(6,1100)
      READ(5,1000) TITLE
      WRITE(6,1000)TITLE
      WRITE(6,1100)
      READ(5,1000) FMT
      DO 400 I=1,NMAT
      READ(5,FMT) N,(ELDAT(N,J),J=1,3)
C    ELDAT(I,1) IS YOUNG'S MODULUS
C    ELDAT(I,2) IS POISSON'S RATIO
C    ELDAT(I,3) IS THE COEFIICIENT OF THERMAL EXPANSION
  400 WRITE(6,FMT) N,(ELDAT(N,J),J=1,3)
      WRITE(6,1100)
      READ(5,1000) TITLE
      WRITE(6,1000)TITLE
      WRITE(6,1100)
      READ(5,1000) FMT
      DO 500 I=1,NCLD
      READ (5,FMT)(NCL(I),(CLOAD(I,J),J=1,NDF))
  500 WRITE(6,FMT)(NCL(I),(CLOAD(I,J),J=1,NDF))
      READ(5,1000) TITLE
      WRITE(6,1000)
```

```
      WRITE(6,1000) TITLE
      WRITE(6,1100)
      NDFP=NDF+1
      READ(5,1000) FMT
      DO 600 I=1,NPBC
      READ(5,FMT) (NBC(I,J),J=1,NDFP)
  600 WRITE(6,FMT)(NBC(I,J),J=1,NDFP)
      RETURN
      END


      SUBROUTINE FSTF
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
     1,NCLD,NPBC,NSB
      COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
      COMMON /B1/ ELDAT(10,3),CLOAD(99,3)
      COMMON /B2/ COORD(850,3),COREL(20,3),ELAST(6,6)
      COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
     1RB3(60)
      DIMENSION STK(3600),AK(3600),B(360)
      EQUIVALENCE (AK1(1,1),STK(1)),(AK2(1,1),AK(1)),(AK3(1,1),B(1))
      N2=-1
      DO 300 I=1,NEL
      DO 100 J=1,NPEL
      J1=NCON(I,J)
      DO 100 K=1,NDF
  100 COREL(J,K)=COORD(J1,K)
      N=NCON(I,21)
      IF(N.EQ.N2) GO TO 200
      CALL ELPROP(N)
      N2=N
  200 CONTINUE
      CALL CUB2
      CALL WDISK1(I,STK,NSTF)
  300 CONTINUE
      RETURN
      END


      SUBROUTINE ELPROP(I)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /B1/ ELDAT(10,3),CLOAD(99,3)
      COMMON /B2/ COORD(850,3),COREL(20,3),ELAST(6,6)
      DO 200 L=1,6
      DO 200 J=1,6
  200 ELAST(L,J)=0.0D0
      E=ELDAT(I,1)
```

```
      PR=ELDAT(I,2)
      ER=E/((1.0D0+PR)*(1.0D0-2.0D0*PR))
      EK=ER*(1.0D0-PR)
      EL=ER*PR
      ELAST(1,1)=EK
      ELAST(1,2)=EL
      ELAST(1,3)=EL
      ELAST(2,1)=EL
      ELAST(2,2)=EK
      ELAST(2,3)=EL
      ELAST(3,1)=EL
      ELAST(3,2)=EL
      ELAST(3,3)=EK
      ELAST(4,4)=ER*(1.0D0-2.0D0*PR)/2.0D0
      ELAST(5,5)=ELAST(4,4)
      ELAST(6,6)=ELAST(4,4)
      RETURN
      END


      SUBROUTINE CUB2
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,N   ,NSTF
     1,NCLD,NPBC,NSB
      COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
     1RB3(60)
      DIMENSION   STK(60,60),AK(60,60),B(6,60)
      EQUIVALENCE (STK(1,1),AK1(1,1)),(AK(1,1),AK2(1,1)),(B(1,1),
     1AK3(1,1))
      DIMENSION XI(2)
      DATA XI/0.5773502691896258,-0.5773502691896258/
      DO 100 I=1,N
      DO 100 J=1,N
  100 STK(I,J)=0.0D0
      DO 200 I=1,2
      X=XI(I)
      DO 200 J=1,2
      Y=XI(J)
      DO 200 K=1,2
      Z=XI(K)
      CALL FORMK (X,Y,Z,1)
      DO 200 L=1,N
      DO 200 M=L,N
  200 STK(L,M)=STK(L,M)+ AK(L,M)
      DO 300 I=1,N
      DO 300 J=I,N
  300 STK(J,I)=STK(I,J)
      RETURN
```

```
      END

      SUBROUTINE FORMK(X,Y,Z,INDIC)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPT ,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,N   ,NSTF
     1,NCLD,NPBC,NSB
      COMMON /B2/ CCCCC(850,3),COORD(20,3),ELAST(6,6)
      COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
     1RB3(60)
      DIMENSION    STK(60,60),AK(60,60),B(6,60)
      EQUIVALENCE (STK(1,1),AK1(1,1)),(AK(1,1),AK2(1,1)),(B(1,1),
     1AK3(1,1))
      DIMENSION AJ(3,3),AJIN(3,3),DNX(3,20),W1(3,20),       B1(60,6),
     1          CORDG(20,3)
      DATA CORDG/1.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,0.0D0,1.0D0,1.0D0,
     11.0D0,-1.0D0,-1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,0.0D0,
     21.0D0,1.0D0,1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,0.0D0,
     31.0D0,1.0D0,-1.0D0,-1.0D0,1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,
     4-1.0D0,0.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,
     50.0D0,0.0D0,0.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,
     6-1.0D0,-1.0D0/
      DF(C,D,E,X1,Y1,Z1)=X1*(1.0D0+D*Y1)*(1.0D0+E*Z1)*(2.0D0*C*X1+D*Y1+
     1E*Z1-1.0D0)/8.0D0
      D2(C,D,E,    Y1,Z1)=-C*(1.0D0+D*Y1)*(1.0D0+E*Z1)/2.0D0
      D4(C,  E,    Y1,Z1)=(1.0D0-C*C)*Y1*(1.0D0+E*Z1)/4.0D0
      DO 10 I=1,6
      DO 10 J=1,N
   10 B(I,J)=0.0D0
      DO 100 I=1,2
      II=12*(I-1)+1
      IT=II+6
      DO 100 J=II,IT,2
      X1=CORDG(J,1)
      Y1=CORDG(J,2)
      Z1=CORDG(J,3)
      W1(1,J)=DF(X,Y,Z,X1,Y1,Z1)
      W1(2,J)=DF(Y,Z,X,Y1,Z1,X1)
  100 W1(3,J)=DF(Z,X,Y,Z1,X1,Y1)
C    MID POINTS X1=0.0
      DO 200 K=1,2
      II=(K-1)*12+2
      IT=II+4
      DO 200 L=II,IT,4
      X1=CORDG(L,1)
      Y1=CORDG(L,2)
      Z1=CORDG(L,3)
      W1(1,L)=D2(X,Y,Z,    Y1,Z1)
```

```
      W1(2,L)=D4(X,   Z,    Y1,Z1)
  200 W1(3,L)=D4(X,   Y,    Z1,Y1)
      DO 300 K=1,2
      II=(K-1)*12+4
      IT=II+4
      DO 300 L=II,IT,4
      X1=CORDG(L,1)
      Y1=CORDG(L,2)
      Z1=CORDG(L,3)
      W1(1,L)=D4(Y,   Z,    X1,Z1)
      W1(2,L)=D2(Y,Z,X,     Z1,X1)
  300 W1(3,L)=D4(Y,   X,    Z1,X1)
      DO 400 L=9,12
      X1=CORDG(L,1)
      Y1=CORDG(L,2)
      Z1=CORDG(L,3)
      W1(1,L)=D4(Z,   Y,    X1,Y1)
      W1(2,L)=D4(Z,   X,    Y1,X1)
  400 W1(3,L)=D2(Z,X,Y,     X1,Y1)
      DO 500 I=1,3
      DO 500 J=1,3
      AJ(I,J)=0.0D0
      DO 500 K=1,NPT
  500 AJ(I,J)=AJ(I,J)+W1(I,K)*COORD(K,J)
      DTJ=AJ(1,1)*AJ(2,2)*AJ(3,3)+AJ(1,2)*AJ(2,3)*AJ(3,1)+AJ(1,3)*AJ(2,1
     1)*AJ(3,2)-AJ(3,1)*AJ(2,2)*AJ(1,3)-AJ(3,2)*AJ(2,3)*AJ(1,1)-AJ(3,3)*
     2AJ(2,1)*AJ(1,2)
      AJIN(1,1)= (AJ(2,2)*AJ(3,3)-AJ(3,2)*AJ(2,3))/DTJ
      AJIN(2,1)=-(AJ(2,1)*AJ(3,3)-AJ(3,1)*AJ(2,3))/DTJ
      AJIN(3,1)= (AJ(2,1)*AJ(3,2)-AJ(3,1)*AJ(2,2))/DTJ
      AJIN(1,2)=-(AJ(1,2)*AJ(3,3)-AJ(3,2)*AJ(1,3))/DTJ
      AJIN(2,2)= (AJ(1,1)*AJ(3,3)-AJ(3,1)*AJ(1,3))/DTJ
      AJIN(3,2)=-(AJ(1,1)*AJ(3,2)-AJ(3,1)*AJ(1,2))/DTJ
      AJIN(1,3)= (AJ(1,2)*AJ(2,3)-AJ(2,2)*AJ(1,3))/DTJ
      AJIN(2,3)=-(AJ(1,1)*AJ(2,3)-AJ(2,1)*AJ(1,3))/DTJ
      AJIN(3,3)= (AJ(1,1)*AJ(2,2)-AJ(2,1)*AJ(1,2))/DTJ
      DO 600 I=1,3
      DO 600 J=1,NPT
      DNX(I,J)=0.0D0
      DO 600 K=1,3
  600 DNX(I,J)=DNX(I,J)+AJIN(I,K)*W1(K,J)
      DO 700 I=1,NPT
      I1=3*I-2
      I2=I1+1
      I3=I2+1
      B(1,I1)=DNX(1,I)
      B(2,I2)=DNX(2,I)
      B(3,I3)=DNX(3,I)
```

```
      B(4,I1)=DNX(2,I)
      B(4,I2)=DNX(1,I)
      B(5,I2)=DNX(3,I)
      B(5,I3)=DNX(2,I)
      B(6,I3)=DNX(1,I)
  700 B(6,I1)=DNX(3,I)
      IF(INDIC.EQ.2) RETURN
      DO 800 I=1,N
      DO 800 J=1,6
      B1(I,J)=0.0D0
      DO 800 K=1,6
  800 B1(I,J)=B1(I,J)+B(K,I)*ELAST(K,J)
      DO 900 I=1,N
      DO 900 J=I,N
      AK(I,J)=0.0D0
      DO 900 K=1,6
  900 AK(I,J)=AK(I,J)+B1(I,K)*B(K,J)
      DO 1000 I=1,N
      DO 1000 J=I,N
      AK(I,J)=AK(I,J)*DTJ
 1000 AK(J,I)=AK(I,J)
      RETURN
      END


      SUBROUTINE MERGE
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
     1,NCLD,NPBC,NSB
      COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
      COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
     1RB3(60)
      DIMENSION A(60,60),S(60,60),B(3600),T(3600)
      EQUIVALENCE (AK1(1,1),A(1,1),B(1)),(AK2(1,1),S(1,1),T(1))
      NTRK(I,J)=(I-1)*(MM+2)+J-I+1
      ZRO=0.0D0
      DO 400 I=1,NN
      DO 400 J=1,MM
      DO 100 I1=1,NS
      DO 100 J1=1,NS
  100 A(I1,J1)=ZRO
      DO 300 K=1,NEL
      DO 200 L=1,NPEL
  200 LM(L)= NDF*NCON(K,L)-NDF
      LMIN=5000
      LMAX=0
      DO 210 I2=1,NPEL
      LMIN=MIN0(LMIN,LM(I2))
```

```
210 LMAX=MAX0(LMAX,LM(I2))
    LMIN=LMIN+1
    LMAX=LMAX+1
    NSI=NS*I
    NSIM=NSI-NS
    IF(LMIN.GT.NSI) GO TO 300
    IF(LMAX.LE.NSIM) GO TO 300
    CALL RDISK1(K,T,NSTF)
    DO 290 II=1,NPEL
    DO 280 JJ=1,NPEL
    DO 270 KK=1,NDF
    III=LM(II)+KK-NS*(I-1)
    IF((III.GT.NS).OR.(III.LT.1)) GO TO 270
    KKK=NDF*II-NDF+KK
    DO 260 LL=1,NDF
    JJJ=LM(JJ)+LL-NS*(J+I-2)
    IF((JJJ.GT.NS).OR.(JJJ.LT.1)) GO TO 260
    LLL=NDF*JJ-NDF+LL
    A(III,JJJ)=A(III,JJJ)+S(KKK,LLL)
260 CONTINUE
270 CONTINUE
280 CONTINUE
290 CONTINUE
300 CONTINUE
    NTK=NTRK(I,J)
    CALL WRDISK(NTK,B,NCOUNT)
400 CONTINUE
    IF(NN*NS.EQ.NEQ) GO TO 600
    NTK=NTRK(NN,1)
    CALL RDDISK(NTK,B,NCOUNT)
    DO 500 I=1,NS
    IF(DABS(A(I,I)).LT.1.0D-14)  A(I,I)=1.0D0
500 CONTINUE
    CALL WRDISK(NTK,B,NCOUNT)
600 CONTINUE
    RETURN
    END


    SUBROUTINE FLOAD
    IMPLICIT REAL*8(A-H,O-Z)
    COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
   1,NCLD,NPBC,NSB
    COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
    COMMON /B1/ ELDAT(10,3),CLOAD(99,3)
    COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
   1RB3(60)
    DIMENSION   A(60,60),S(60,60),R1(60),R2(60)
```

```
      EQUIVALENCE(A(1,1),AK1(1,1)),(S(1,1),AK2(1,1)),(R1(1),RB1(1)),
     1(R2(1),RB2(1))
      ZRO=0.0D0
      DO 1100 I=1,NN
      DO 100 J=1,NS
  100 R1(J)=ZRO
      IL=(I-1)*NS+1
      IH=IL+NS-1
      DO 1000 K=1,NCLD
      INCH=NCL(K)*NDF
      INCL=INCH-NDF+1
      IF((INCH.LE.IH).AND.(INCL.GE.IL)) GO TO 200
      GO TO 400
  200 I1=INCL-IL+1
      I2=I1+NDF-1
      IC=0
      DO 300 L=I1,I2
      IC=IC+1
  300 R1(L)=CLOAD(K,IC)
      GO TO 1000
  400 IF(((INCL.LE.IH).AND.(INCL.GT.IL)).AND.(INCH.GT.IH)) GO TO 500
      GO TO 700
  500 I1=INCL-IL+1
      IC=0
      DO 600 M=I1,NS
      IC=IC+1
  600 R1(M)=CLOAD(K,IC)
  700 IF(((INCH.LT.IH).AND.(INCH.GE.IL)).AND.(INCL.LT.IL)) GO TO 800
      GO TO 1000
  800 I1=INCL-IL+1
      I2=I1+NDF-1
      IC=NDF-I2
      DO 900 N=1,I2
      IC=IC+1
  900 R1(N)=CLOAD(K,IC)
 1000 CONTINUE
      NTK=I*(MM+1)
      CALL WRDISK(NTK,R1,NST)
 1100 CONTINUE
      RETURN
      END

      SUBROUTINE BCOND
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
     1,NCLD,NPBC,NSB
      COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
```

```
      COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
     1RB3(60)
      DIMENSION AKW(3600)
      EQUIVALENCE (AKW(1),AK1(1,1))
      NTRK(I,J)=(I-1)*(MM+2)+J-I+1
      ABIGN=1.0D20
      DO 100 I=1,NPBC
  100 NBC(I,1)=NDF*NBC(I,1)-NDF
      DO 900 I=1,NN
      NTK=NTRK(I,1)
      CALL RDDISK(NTK,AKW,NCOUNT)
      DO 800 J=1,NPBC
      IBL=NBC(J,1)+1-NS*(I-1)
      IBH=IBL+NDF-1
      IF(((IBL.LT.1).AND.(IBH.LT.1)).OR.((IBL.GT.NS).AND.(IBH.GT.NS)))
     1 GO TO 800
      IF((IBL.GE.1).AND.(IBH.LE.NS)) GO TO 200
       GO TO 300
  200 IC=1
      DO 250 K=IBL,IBH
       IC=IC+1
      IF(NBC(J,IC).NE.1) GO TO 250
      AK1(K,K)=AK1(K,K)+ABIGN
  250 CONTINUE
      GO TO 800
  300 IF(((IBL.GT.1).AND.(IBL.LE.NS)).AND.(IBH.GT.NS)) GO TO 400
      GO TO 600
  400 IC=1
      DO 500 K=IBL,NS
      IC=IC+1
      IF(NBC(J,IC).NE.1) GO TO 500
      AK1(K,K)=AK1(K,K)+ABIGN
  500 CONTINUE
      GO TO 800
  600 IC=NDF-IBH+1
      DO 700 K=1,IBH
      IC=IC+1
       IF(NBC(J,IC).NE.1) GO TO 700
      AK1(K,K)=AK1(K,K)+ABIGN
  700 CONTINUE
  800 CONTINUE
      CALL WRDISK(NTK,AKW,NCOUNT)
  900 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SOLVE
      IMPLICIT REAL*8(A-H,O-Z)
C*****************************************************************************
C
C                        CODED BY: GILLES CANTIN, DECEMBER 1969
C                        TESTED ON AN IBM 360/67
C
C        NN    IS THE NUMBER OF BLOCKS PER COLUMN
C        MM    IS THE NUMBER OF BLOCKS PER ROW
C        NS    IS THE SIZE OF ONE BLOCK
C        NSIZ  IS EQUAL TO NS*NS
C              IN THIS VERSION NS IS 50, TO MODIFY, THE DIMENSION
C              STATEMENT MUST BE REPUNCHED ACCORDING TO CARD ZZZZZZZZ
C              WITH THE PARAMETERS NSIZ AND NS REPLACED BY NUMBERS
C     DIMENSION AK1(NSIZ),AK2(NSIZ),AK3(NSIZ),RB1(NS),RB2(NS),RB3(NS)
C
C*****************************************************************************
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
     1,NCLD,NPBC,NSB
      COMMON /B3/ AT1(60,60),AT2(60,60),AT3(60,60),RT1(60),RT2(60),
     1RT3(60)
      DIMENSION AK1(3600),AK2(3600),AK3(3600),RB1(60),RB2(60),RB3(60)
      EQUIVALENCE (AK1(1),AT1(1,1)),(AK2(1),AT2(1,1)),(AK3(1),AT3(1,1)),
     1(RB1(1),RT1(1)),(RB2(1),RT2(1)),(RB3(1),RT3(1))
      NTRK(I,J)=(I-1)*(MM+2)+J-I+1
      NCOUNT = NS*NS
      KMM=MM
      N=0
  100 N=N+1
C
C        REDUCE BLOCK ROW  "N"
C
C        1.-  REDUCE R.H.S.
C
      NTK=NTRK(N,1)
      NTR=N*(MM+1)
      CALL RDDISK(NTK,AK2,NCOUNT)
      CALL SYMINV(AK2,NS,RB1,RB2,IFLG)
      IF(IFLG.EQ.1) GO TO 600
      IF(IFLG.EQ.2) WRITE(6,2000) N
 2000 FORMAT(5X,'BLOCK (',I3,', 1) IS NEARLY SINGULAR')
      CALL RDDISK(NTR,RB1,NS)
      CALL MULT(AK2,RB1,RB2,NS,NS,1)
      CALL WRDISK(NTR,RB2,NS)
C
C        2.-  CHECK FOR LAST ROW OF BLOCKS
C
      IF(N.EQ.NN) GO TO 300
```

```
C
C          3.-  REDUCE BLOCKS IN ROW "N"
C
      IF(N.GT.(NN-MM+1)) KMM=KMM-1
      DO 200 K=2,KMM
      NTK=NTRK(N,K)
      CALL RDDISK(NTK,AK1,NCOUNT)
      CALL MULT(AK2,AK1,AK3,NS,NS,NS)
      CALL WRDISK(NTK,AK3,NCOUNT)
      DO 150 I=1,NS
      II=(I-1)*NS
      DO 150 J=1,NS
      IL=II+J
      IR=I+(J-1)*NS
  150 AK3(IL)=AK1(IR)
      NTK=NTRK(NN,K)
      CALL WRDISK(NTK,AK3,NCOUNT)
  200 CONTINUE
C
C          4.-  REDUCE REMAINING ROWS OF BLOCKS
C
      DO 260 L=2,KMM
      I=N+L-1
      IF(I.GT.NN) GO TO 260
      J=0
      NTK=NTRK(NN,L)
      CALL RDDISK (NTK,AK2,NCOUNT)
      DO 250 K=L,KMM
      J=J+1
      NTK=NTRK(N,K)
      CALL RDDISK(NTK,AK1,NCOUNT)
      CALL MULT(AK2,AK1,AK3,NS,NS,NS)
      NTK=NTRK(I,J)
      CALL RDDISK(NTK,AK1,NCOUNT)
      DO 210 I1=1,NCOUNT
  210 AK1(I1)=AK1(I1)-AK3(I1)
      CALL WRDISK(NTK,AK1,NCOUNT)
  250 CONTINUE
      CALL MULT(AK2,RB2,RB3,NS,NS,1)
      NTR=I*(MM+1)
      CALL RDDISK(NTR,RB1,NS)
      DO 255 I1=1,NS
  255 RB1(I1)=RB1(I1)-RB3(I1)
      CALL WRDISK(NTR,RB1,NS)
  260 CONTINUE
      GO TO 100
C
C          BACK SUBSTITUTION
```

```
C
  300 N=N-1
C
C        1.-  CHECK FOR FIRST ROW OF BLOCKS
C
      IF (N.EQ.0) GO TO 500
C
C        2.- CALCULATE BLOCKS OF UNKNOWNS
C
      NT1=N*(MM+1)
      CALL RDDISK(NT1,RB3,NS)
      DO 400 K=2,KMM
      L=N+K-1
      IF(L.GT.NN) GO TO 400
      NTK=NTRK(N,K)
      CALL RDDISK(NTK,AK1,NCOUNT)
      NTR=L*(MM+1)
      CALL RDDISK(NTR,RB1,NS)
      CALL MULT(AK1,RB1,RB2,NS,NS,1)
      DO 310 K1=1,NS
  310 RB3(K1)=RB3(K1)-RB2(K1)
  400 CONTINUE
      CALL WRDISK(NT1,RB3,NS)
      KMM=KMM+1
      IF(KMM.GT.MM) KMM=MM
      GO TO 300
  500 RETURN
  600 WRITE(6,1000) N
 1000 FORMAT(5X,'BLOCK (',I3,', 1) IS SINGULAR')
      STOP
      END

      SUBROUTINE MULT(A,B,C,NRA,NCA,NCB)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION A(1),B(1),C(1)
      ZRO=0.0D0
      DO 100 I=1,NRA
      DO 100 J=1,NCB
      IC=I+(J-1)*NRA
      C(IC)=ZRO
      DO 100 K=1,NCA
      IA=I+(K-1)*NRA
      IB=K+(J-1)*NCA
      C(IC)=C(IC)+A(IA)*B(IB)
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SYMINV(A,N,B,C,IFLG)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION A(1),B(1),C(1)
C***********************************************************************
C
C        SYMMETRICAL MATRIX INVERSION
C
C***********************************************************************
      IFLG=0
      ZRO=0.0D0
      ABIGN=1.0D15
      TRACE=ZRO
      DO 5 I=1,N
      IAD=(I-1)*N+I
      IF(A(IAD).GE.ABIGN ) GO TO 5
      TRACE=TRACE+DABS(A(IAD))
    5 CONTINUE
      APZRO=TRACE*1.0D-12
      IF(APZRO.GT.1.0D-20) APZRO=1.0D-20
      LP=N
      DO 10  I=2,LP
      DO 10  J=I,LP
      ICOL=J+LP*(I-2)
      IROW=I+(J-1)*LP-1
      IF(A(ICOL).EQ.A(IROW)) GO TO 10
      A(ICOL)=0.5D0*(A(ICOL)+A(IROW))
      A(IROW)=A(ICOL)
   10 CONTINUE
      DO 25 I=1,N
      B(I)=ZRO
      II=(I-1)*N
      DO 20 J=1,N
      JJ=II+J
      IF(A( JJ).GE.ABIGN ) GO TO 20
      B(I)=B(I)+DABS(A(JJ))
   20 CONTINUE
   25 CONTINUE
      ANR=ZRO
      DO 30 I=1,N
   30 ANR=DMAX1(ANR,B(I))
      DO 145 I=1,N
      NR=(I-1)*N
C
      DO 100 J=1,N
      K=NR+J
  100 B(J)=A(K)
```

```
      D=B(I)
      IF(D.EQ.ZRO) GO TO 180
      IF(DABS(D).LT.APZRO) IFLG=2
      DO 110 J=1,N
  110 C(J)=-B(J)/D
C
      L=1
      DO 130 J=1,N
      M=L
      DO 120 K=J,N
      DB=DABS(B(J))
      IF(DB.LE.1.OD-40) GO TO 115
      DC=DABS(C(K))
      IF(DC.LE.1.OD-40) GO TO 115
      IF((DB.LE.APZRO).AND.(DC.LE.APZRO)) GO TO 115
      A(L)=A(L)+B(J)*C(K)
  115 CONTINUE
      A(M)=A(L)
      M=M+N
  120 L=L+1
  130 L=L+J
C
      C(I)=-1.0D0/D
      M=I
      DO 140 J=1,N
      K=NR+J
      A(K)=C(J)
      A(M)=C(J)
  140 M=M+N
  145 CONTINUE
C
      NS=N*N
      DO 150 J=1,NS
  150 A(J)=-A(J)
C
      DO 165 I=1,N
      B(I)=ZRO
      II=(I-1)*N
      DO 160 J=1,N
      JJ=II+J
      IF(A(JJ).EQ.1.0D0) GO TO 160
      B(I)=B(I)+DABS(A(JJ))
  160 CONTINUE
  165 CONTINUE
      AINR=ZRO
      DO 170 I=1,N
  170 AINR=DMAX1(AINR,B(I))
      IF(AINR.LT.1.0D-15) AINR=1.0D0/ANR
```

```
      CNBR=ANR*AINR
      WRITE(6,2000) CNBR,ANR,AINR
 2000 FORMAT(5X,'CONDITION NUMBER ',5X,1PD25.16,5X,2(1PD25.16))
      RETURN
  180 IFLG=1
      RETURN
      END


      SUBROUTINE DISP
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
     1,NCLD,NPBC,NSB
      COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
      COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
     1RB3(60)
      DIMENSION DAT(3000),PAGE(60,3),REACT(850,3)
      EQUIVALENCE (AK1(1,1),DAT(1)),(AK3(1,1),PAGE(1,1)),
     1(AK3(1,4),REACT(1,1))
      REWIND 11
      ABIGN=1.0D20
      ZRO=0.0D0
      DO 10 I=1,NPBC
      DO 10 J=1,NDF
   10 REACT(I,J)=ZRO
      DO 200 I=1,NN
      II=(I-1)*NS+1
      IJ=II+NS-1
      NTK=I*(MM+1)
      CALL RDDISK(NTK,RB1,NS)
      DO 100 J=II,IJ
      IM=J-(I-1)*NS
  100 DAT(J)=RB1(IM)
  200 CONTINUE
      WRITE(11) (DAT (I),I=1,NEQ)
      END FILE 11
      DO 250 I=1,NPBC
      I1=NBC(I,1)+1
      I2=I1+1
      I3=I2+1
      IF(NBC(I,2).NE.1) GO TO 220
      REACT(I,1)=DAT(I1)*ABIGN
      DAT(I1)=ZRO
  220 IF(NBC(I,3).NE.1) GO TO 240
      REACT(I,2)=DAT(I2)*ABIGN
      DAT(I2)=ZRO
  240 IF(NBC(I,4).NE.1) GO TO 250
      REACT(I,3)=DAT(I3)*ABIGN
```

```
      DAT(I3)=ZRO
  250 CONTINUE
      ICT=0
      NPAGE=(NDPT+59)/60
      NLM=NDPT-(NPAGE-1)*60
      DO 600 I=1,NPAGE
      WRITE(6,1000)
      NLINE=60
      IF(I.EQ.NPAGE) NLINE=NLM
      DO 300 J=1,60
      DO 300 K=1,NDF
  300 PAGE(J,K)=0.0D0
      IX=60*NDF*(I-1)-NDF
      DO 400 J=1,NDF
      IY=IX+J
      DO 400 K=1,NLINE
      IY=IY+NDF
  400 PAGE(K,J)=DAT(IY)
      DO 500 J=1,NLINE
      ICT=ICT+1
  500 WRITE(6,2000) ICT,(PAGE(J,K),K=1,NDF)
  600 CONTINUE
      NPAGE=(NPBC+59)/60
      NLM=NPBC-(NPAGE-1)*60
      DO 800 I=1,NPAGE
      WRITE(6,1100)
      NLINE=60
      IF(I.EQ.NPAGE) NLINE=NLM
      IX=(I-1)*60
      DO 700 J=1,NLINE
      J1=J+IX
      DO 700 K=1,NDF
      ICT=(NBC(J1,1)/NDF)+1
  700 WRITE(6,2000) ICT,(REACT(J1,K),K=1,NDF)
  800 CONTINUE
      SUMX=ZRO
      SUMY=ZRO
      SUMZ=ZRO
      DO 850 I=1,NPBC
      SUMX=SUMX+REACT(I,1)
      SUMY=SUMY+REACT(I,2)
  850 SUMZ=SUMZ+REACT(I,3)
      WRITE(6,3000) SUMX,SUMY,SUMZ
      RETURN
 1000 FORMAT(1H1,//,5X,'  D-I-S-P-L-A-C-E-M-E-N-T-S   ',
     1           //,5X,5H N.PT,12X,3H  X,23X,2H Y,23X,2H Z,//)
 1100 FORMAT(1H1,//,5X,'  R-E-A-C-T-I-O-N-S  ',
     1           //,5X,5H N.PT,12X,3H  X,23X,2H Y,23X,2H Z,//)
```

```
2000 FORMAT(5X,I5,3(1PD25.16))
2001 FORMAT (I4,3(1PD25.16))
3000 FORMAT(///,' EQUILIBRIUM CHECK ',/,10X,3(1PD25.16))
     END


     SUBROUTINE STRESS(KELP)
     IMPLICIT REAL*8(A-H,O-Z)
     COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
    1,NCLD,NPBC,NSB
     COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
     COMMON /B1/ ELDAT(10,3),CLOAD(99,3)
     COMMON /B2/ COORD(850,3),COREL(20,3),ELAST(6,6)
     COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
    1RB3(60)
     DIMENSION CORDG(20,3)
     DATA CORDG/1.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,0.0D0,1.0D0,1.0D0,
    11.0D0,-1.0D0,-1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,0.0D0,
    21.0D0,1.0D0,1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,0.0D0,
    31.0D0,1.0D0,-1.0D0,-1.0D0,1.0D0,1.0D0,1.0D0,0.0D0,-1.0D0,-1.0D0,
    4-1.0D0,0.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,1.0D0,
    50.0D0,0.0D0,0.0D0,0.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,-1.0D0,
    6-1.0D0,-1.0D0/
     DIMENSION SNELM(20,6),UEL(60),UJNT(3000),SSELM(20,6),SSNN(6),
    1B(6,60),SSSS(6)
     EQUIVALENCE (AK3(1),B(1)),(AK3(361),SNELM(1)),(AK3(481),SSELM(1)),
    1(AK3(601),SSNN(1)),(AK3(607),SSSS(1)),(AK1(1),UJNT(1))
     REWIND 12
     REWIND 13
     DO 500 I=1,NEL
     REWIND 11
     READ (11) (UJNT(J),J=1,NEQ)
     WRITE(6,1000) I
     DO 300 J=1,NPEL
     LM(J)=NCON(I,J)
     JJ=LM(J)
     JNT=3*(JJ-1)
     JEL=3*(J-1)
     DO 250 K=1,NDF
     COREL(J,K)=COORD(JJ,K)
     J1=JNT+K
     J2=JEL+K
 250 UEL(J2)=UJNT(J1)
 300 CONTINUE
     N=NCON(I,21)
     CALL ELPROP(N)
     DO 400 J=1,NPEL
     X=CORDG(J,1)*(1.0D0-1.0D-10)
```

```
      Y=CORDG(J,2)*(1.0D0-1.0D-10)
      Z=CORDG(J,3)*(1.0D0-1.0D-10)
      CALL FORMK(X,Y,Z,2)
      DO 310 J1=1,6
      SSNN(J1)=0.0D0
      DO 310 J2=1,60
  310 SSNN(J1)=SSNN(J1)+B(J1,J2)*UEL(J2)
      DO 320 J1=1,6
      SSSS(J1)=0.0D0
      DO 320 J2=1,6
  320 SSSS(J1)=SSSS(J1)+ELAST(J1,J2)*SSNN(J2)
      WRITE(12) (SSNN(L),L=1,6)
      WRITE(13) (SSSS(M),M=1,6)
      DO 350 K=1,6
      SNELM(J,K)=SSNN(K)
  350 SSELM(J,K)=SSSS(K)
  400 CONTINUE
      IF(KELP.EQ.0) GO TO 500
      DO 450 K=1,NPEL
      WRITE(6,2000)  LM(K),(SNELM(K,L),L=1,6)
      WRITE(6,2000)  LM(K),(SSELM(K,L),L=1,6)
  450 WRITE(6,3000)
  500 CONTINUE
      END FILE 12
      END FILE 13
      CALL SJOINT
      RETURN
 1000 FORMAT(1H1,//' S-T-R-A-I-N-S / S-T-R-E-S-S-E-S FOR ELEMENT ',I4,
     1//,2X,'JOINT',6X,
     2'SNX/SSX',8X,'SNY/SSY',8X,'SNZ/SSZ',8X,'SNXY/SSXY',6X,'SNYZ/SSYZ',
     36X,'SNZX/SSZX',//)
 2000 FORMAT(2X,I4,4X,6(1PD15.6))
 3000 FORMAT(1H )
      END


      SUBROUTINE SJOINT
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /NB1/NEL,NDPT,NPEL,NDF,NEQ,NBAND,NN,MM,NS,NCOUNT,NST,NSTF
     1,NCLD,NPBC,NSB
      COMMON /NB2/ NCON(100,21),NCL(99),LM(20),NBC(500,4)
      COMMON /B3/ AK1(60,60),AK2(60,60),AK3(60,60),RB1(60),RB2(60),
     1RB3(60)
      DIMENSION SSJNT(1000,6),COUNT(1000),SSSS(6)
      EQUIVALENCE (AK1(1,1),SSJNT(1,1)),(AK3(1,1),COUNT(1)),
     1(RB1(1),SSSS(1))
      DO 100 I=1,NDPT
      COUNT(I)=0.0D0
```

```
      DO 100 J=1,6
  100 SSJNT(I,J)=0.0D0
      REWIND 12
      DO 270 I=1,NEL
      DO 260 J=1,NPEL
      JJ=NCON(I,J)
      READ(12) (SSSS(L),L=1,6)
      COUNT(JJ)=CCUNT(JJ)+1.0D0
      DO 250 K=1,6
  250 SSJNT(JJ,K)=SSJNT(JJ,K)+SSSS(K)
  260 CONTINUE
  270 CONTINUE
      WRITE(6,1000)
      NSX=60
      DO 410 I=1,NDPT
      SCNT=COUNT(I)
      DO 300 J=1,6
  300 SSJNT(I,J)=SSJNT(I,J)/SCNT
      IF(I.NE.NSX) GO TO 400
      WRITE(6,1000)
      NSX=NSX+60
  400 WRITE(6,2000) I,(SSJNT(I,K),K=1,6)
  410 CONTINUE
      DO 450 I=1,NDPT
      DO 450 J=1,6
  450 SSJNT(I,J)=0.0D0
      REWIND 13
      DO 520 I=1,NEL
      DO 510 J=1,NPEL
      JJ=NCON(I,J)
      READ(13) (SSSS(L),L=1,6)
      DO 500 K=1,6
  500 SSJNT(JJ,K)=SSJNT(JJ,K)+SSSS(K)
  510 CONTINUE
  520 CONTINUE
      WRITE(6,3000)
      NSX=60
      DO 610 I=1,NDPT
      SCNT=COUNT(I)
      DO 550 J=1,6
  550 SSJNT(I,J)=SSJNT(I,J)/SCNT
      IF(I.NE.NSX) GO TO 600
      WRITE(6,3000)
      NSX=NSX+60
  600 WRITE(6,2000) I,(SSJNT(I,K),K=1,6)
  610 CONTINUE
      RETURN
 1000 FORMAT(1H1,//,' A-V-E-R-A-G-E  S-T-R-A-I-N-S  AT  THE  JOINTS ',
```

```
     1//,2X,'JOINT',
     29X,'SNX',12X,'SNY',12X,'SNZ',12X,'SNXY',11X,'SNYZ',11X,'SNZX',//)
2000 FORMAT(2X,I4,4X,6(1PD15.6))
3000 FORMAT(1H1,//,' A-V-E-R-A-G-E  S-T-R-E-S-S-E-S  AT  THE  JOINTS ',
     1//,2X,'JOINT',
     29X,'SSX',12X,'SSY',12X,'SSZ',12X,'SSXY',11X,'SSYZ',11X,'SSZX',//)
4000 FORMAT(I4,6(1PD12.5))
     END


     SUBROUTINE WRDISK(NTRACK,A,NCT)
     INTEGER LAST/0/,LASTT/0/
     REAL*8 NAME(4)/'WRDISK','WRDISK1','RDDISK','RDDISK1'/
     DIMENSION A(1)
     DATA NRL/60/,NRCM/12480/,NRL1/60/,NRCM1/3840/
     REAL*8 A
      DEFINE FILE 7(12480,480,E,I),8(3840,480,E,II)
1000 FORMAT (60A8)
1100 FORMAT (60A8)
     NRCMM=NRCM-1
     NRCD= NRCM/NRL
     IF(NTRACK .GT. NRCD ) GO TO 900
     IF(NTRACK .LT. 0) NTRACK =(LAST+NRL-1)/NRL
     N = NTRACK-1
C  THE MAXIMUM NUMBER OF WORDS IN A OR B MUST BE .LE. NRL*60
     N=N*NRL+1
     IF(NCT.GT.NRL ) GO TO 50
     WRITE(7'N,1000)        (A(J),J=1,NCT)
     IF (LAST .LT. I) LAST = I
     IF(LAST.GT. NRCMM)LAST=0
     RETURN
  50 JI = NRL+1
     WRITE(7'N,1000)       (A(J),J=1,NRL)
  75 JE = JI + NRL - 1
     IF ( JE .GE. NCT) GO TO 100
     WRITE(7'I,1000)        (A(J),J=JI,JE)
     JI = JI+NRL
     GO TO 75
 100 WRITE(7'I,1000)       (A(J),J=JI,NCT)
     IF(I.GT.LAST) LAST=I
     IF(LAST.GT. NRCMM)LAST=0
     RETURN
     ENTRY  WDISK1(NTRACK,A1,NCT)
     DIMENSION A1(1)
     REAL*8 A1
      DEFINE FILE 7(12480,480,E,I),8(3840,480,E,II)
     NRCMM1=NRCM1-1
     NRCD1=NRCM1/NRL1
```

```
      IF(NTRACK .GT. NRCD1) GO TO 905
      IF(NTRACK .LT. 0) NTRACK =(LASTT+NRL1-1)/NRL1
      N = NTRACK-1
      N=N*NRL1+1
      IF (NCT.GT.NRL1)GO TO 350
      WRITE(8'N,1100)         (A1(J),J=1,NCT)
      IF(LASTT.LT.II) LASTT=II
      IF(LASTT.GT.NRCMM1) LASTT=0
      RETURN
  350 JI=NRL1+1
      WRITE(8'N,1100)         (A1(J),J=1,NRL1)
  375 JE = JI + NRL1- 1
      IF ( JE .GE. NCT) GO TO 300
      WRITE(8'II,1100)         (A1(J),J=JI,JE)
      JI = JI+NRL1
      GO TO 375
  300 WRITE(8'II,1100)         (A1(J),J=JI,NCT)
      IF(II.GT.LASTT) LASTT=II
      IF(LASTT.GT.NRCMM1) LASTT=0
      RETURN
      ENTRY RDDISK(NTRACK,B,NCT)
      REAL*8 B
      DIMENSION B(1)
       DEFINE FILE 7(12480,480,E,I),8(3840,480,E,II)
      IF(NTRACK .GT.NRCD ) GO TO 910
      N=NTRACK-1
      N=N*NRL+1
      IF (NCT .GT.NRL)GO TO 150
      READ(7'N,1000,ERR=500) (B(J),J=1,NCT)
      RETURN
  150 READ(7'N,1000,ERR=500) (B(J),J=1,NRL)
      JI = NRL + 1
  175 JE = JI + NRL - 1
      IF (JE .GE. NCT) GO TO 200
      READ(7'I,1000,ERR=500) (B(J),J=JI,JE)
      JI = JI + NRL
      GO TO 175
  200 READ(7'I ,1000,ERR=500) (B(J),J=JI,NCT)
      RETURN
      ENTRY  RDISK1(NTRACK,B1,NCT)
      REAL*8 B1
      DIMENSION B1(1)
       DEFINE FILE 7(12480,480,E,I),8(3840,480,E,II)
      IF(NTRACK .GT.NRCD1) GO TO 915
      N=NTRACK-1
      N=N*NRL1+1
      IF (NCT.GT.NRL1)GO TO 450
      READ(8'N,1100,ERR=500) (B1(J),J=1,NCT)
```

```
      RETURN
  450 READ(8'N,1100,ERR=500)(B1(J),J=1, NRL1)
      JI = NRL1+ 1
  475 JE = JI + NRL1- 1
      IF (JE .GE. NCT) GO TO 400
      READ(8'II,1100,ERR=500) (B1(J),J=JI,JE)
      JI = JI + NRL1
      GO TO 475
  400 READ(8'II,1100,ERR=500) (B1(J),J=JI,NCT)
      RETURN
  500 WRITE(6,2000)
 2000 FORMAT(' A MACHINE ERROR WAS MADE DURING THE READ OR WRITE DISK ')
      STOP
  900 K=1
      GO TO 920
  905 K=2
      GO TO 920
  910 K=3
      GO TO 920
  915 K=4
  920 WRITE(6,1920)NAME(K),NRCD,NTRACK
 1920 FORMAT('0 ERROR IN CALL OF ',A6,', NTRACK TOO LARGE,(MUST BE .LT.
     1THAN ',I5,/,'NTRACK = ',I5)
      STOP
      END
```

LIST OF REFERENCES

1. Cantin, G., "Three Dimensional Finite Element Studies," (Naval Postgraduate School Report in preparation).

2. Hanson, D. E., Three-Dimensional Elasto-static Problems Using Isoparametric Finite Elements, Master's Thesis, Naval Postgraduate School, Monterey, California, 1971

3. Crandall, S. H., Engineering Analysis, McGraw-Hill, 1956.

4. Timoshenko, S. P., Goodier, J. N., Theory of Elasticity, 3rd ed., McGraw-Hill, 1970.

5. Zienkiewicz, O. C., The Finite Element Method in Engineering Science, McGraw-Hill, 1971.

6. Felippa, C. A., Refined Finite Element Analysis of Linear and Non Linear Two-Dimensional Structures, Ph.D. Thesis, University of California, Berkeley, 1966.

7. Cantin, G., "Rigid Body Motions in Curved Finite Elements, AIAA Journal, Vol. 8, No. 7, pp. 1252-1255, July 1970.

8. Leonidas, E., A General Purpose Three-Dimensional Stress Analysis Program, Master's Thesis, Naval Postgraduate School, Monterey, California, 1971.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Professor Gilles Cantin, Code 59Ci Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	5
4.	LT. Charles G. Pfeifer 243 Turnpike Avenue Portsmouth, Rhode Island 02871	1
5.	Professor Robert E. Newton, Code 59Ne Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
6.	Asst. Professor David Salinas, Code 59Zc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
7.	Professor R. H. Nunn, Code 59Nu Chairman, Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
8.	LCDR Emmanuel Leonidas, HN Hellenic Navy Command Athens, Greece	1

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

REPORT TITLE

Evaluation of a Three-Dimensional Stress Analysis Program

DESCRIPTIVE NOTES (Type of report and inclusive dates)

Master's Thesis; September 1972

AUTHOR(S) (First name, middle initial, last name)

Charles Gregory Pfeifer

REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
September 1972	90	8

8a. CONTRACT OR GRANT NO. b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

The objective of this work was to analyze a computer program using three dimensional quadratic isoparametric finite elements for structural analysis. Three problems with classical solutions were run with various mesh sizes using the computer program being tested. The data computed was then extensively analyzed, and compared with the classical solutions. The analysis of a fourth problem was continued and compared with results obtained in an earlier project.

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Stress Analysis						
Isoparametric Finite Element						

Thesis
P4615
c.1

138064

Pfeifer

Evaluation of three-
dimensional stress
analysis program.