Regulations for Higher Degrees: Copyright

Preamble

Dissemination of knowledge is one of the objects of the University. Therefore Members of the University and others who submit theses/dissertations for higher degrees are expected to relinquish to the University certain rights of reproduction and distribution.

Moreover it is recognised that applicants owe a duty to their Departments of study, the Academic Staff and sponsoring bodies for their respective contributions to the research. Within the limits of these requirements, the author's copyright is safeguarded.

Regulations

1. When submitting a thesis/dissertation for the purposes of a higher degree the applicant shall sign an irrevocable authority in prescribed form appointing the Librarian his attorney with the right to reproduce the thesis/dissertation by photocopy or in microfilm and to distribute copies to those institutions or persons who in the Librarian's opinion require them for academic (as distinct from commercial) purposes.

2. The Librarian in consultation with the appropriate Department of study or sponsoring body shall have the right to refuse to provide copies, or to impose such conditions as he thinks fit on the provision of copies, with the object of safeguarding the applicant's copyright and the interests of the University and the sponsoring body.

3. These Regulations are subject to requirements of any body under whose sponsorship the research project giving rise to the thesis/dissertation is carried on.
FINITE ELEMENT ANALYSIS FOR
AXISYMMETRIC ELASTIC STRESS PROBLEMS
by
JOHN LESLIE WARD, C.Eng, M.I.Mech.E.

A thesis submitted in conformity with
requirements for the Degree of Master of Philosophy

University of Surrey,
Guildford, Surrey,
Great Britain.
This thesis describes the development and the applicability of a finite element sequence of computer programs for the elastic stress analysis of axisymmetric structures.

The following topics are discussed:

(i) the need for an axisymmetric element,
(ii) the mathematical theory outlined in physical terms,
(iii) the philosophy adopted in writing computer programs, and
(iv) an ideal problem used to test the sequence of programs.

Particular attention has been paid to accuracy, minimization of errors and ease of running programs. Thus, in determining the fifteen integrals required to obtain the element stiffnesses, algebraic equations were used in preference to numerical integration and thus avoid both truncation error and round-off error. Care was taken in the formulation of these equations to allow for the possibility of one or more vertices to lie on the axis of symmetry and to allow for two vertices to have the same radius. The latter facility is not available in treatments by other authors. To enable the unsymmetric matrices used in the formation of element stiffness to be inverted; to improve the accuracy of the application of the overall stiffness matrix to the load vector; and in fitting surfaces by the method of least squares, special subroutines were written in a form suitable for partial pivoting. To avoid large ratios of matrix element size, which would prevent inversion on even the largest word length computer, a technique of changing origin and scaling was adopted in fitting surfaces by the method of least squares. The technique of changing origin was also used in the formation of certain of the integrals to obtain a greater degree of accuracy on the digital computers with their inherent round-off error.
Axisymmetric analysis was eased by separating the computation into a sequence of discrete programs. Adopting this procedure (a) reduced the requirements for computer time and space thus minimizing cost and restrictions imposed by normal operating procedures; (b) allowed decisions on restraint and external loading to be deferred until the relevant values were required and available; (c) allowed modifications to be made to any program in the sequence without having a feedback effect on any of the other programs, and, finally, (d) facilitated parametric studies to be made of the effect of varying geometry, material properties, restraints, and loading.

The novel fitting of fourth order surfaces to displacements to obtain strains and stresses, a feature unique to this sequence of programs, produces a high order of accuracy.

The high order of accuracy was illustrated by the problem used to test the sequence of computer programs, where with three quarters of a million matrix elements, the resulting accumulation of round-off error only amounted to 0.001%. There was very good agreement between finite element results and Lame, the general error (except where the loads were applied) being about 0.03%, so that the accumulated round-off error was insignificant even with one thousand finite elements.

In addition to solving small-deflection static-elastic problems, the described sequence of computer programs may be utilised to solve large displacement problems by adopting an iterative procedure and, by taking advantage of the mid-side nodes, small modifications would permit analysis of pseudo dynamic problems.
ACKNOWLEDGEMENTS

I wish to thank Mr R L Latham for his constant support and encouragement.

The work described was carried out under the general supervision of I M Allison, Professor of Experimental Mechanics at the University of Surrey to whom I am grateful for many useful discussions.

I wish to thank Mr T Fuller of the Department of Mathematics at the University of Surrey for helpful discussions and suggestions in connection with the mathematical aspects of this work, in particular his help with the matrix manipulation. I acknowledge the helpful advice from Dr J H Wilkinson of the National Physical Laboratory in connection with inversion of the large stiffness matrix and his proof with regard to singularity.

I acknowledge the help with computers from Mrs Anne Clapp of University of Surrey and Mr Dave Parker of Atlas Computer Laboratory, Chilton.

Finally, without my wife's kind forbearance and encouragement the present work would not have been completed.
### CHAPTER 1 INTRODUCTION

<table>
<thead>
<tr>
<th>Finite Difference and Finite Element Methods</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advantages of Finite Difference Method</td>
<td>3</td>
</tr>
<tr>
<td>Disadvantages of Finite Difference Method</td>
<td>5</td>
</tr>
<tr>
<td>Advantages of Finite Element Method</td>
<td>5</td>
</tr>
<tr>
<td>Disadvantages of Finite Element Method</td>
<td>6</td>
</tr>
<tr>
<td>The need for a particular type of axisymmetric finite element</td>
<td>7</td>
</tr>
<tr>
<td>The requirements for the program</td>
<td>9</td>
</tr>
<tr>
<td>Comparing the application of Finite Element and Photoelastic methods</td>
<td>10</td>
</tr>
</tbody>
</table>

### CHAPTER 2 THEORY OF FINITE ELEMENT

| Definition of Finite Element Method          | 12   |
| Formulation of the spring stiffness equations| 12   |
| Displacement functions and strain           | 14   |
| Determination of stresses from strain       | 15   |
| Determination of spring stiffness matrix    | 17   |
| Original specific features concerning the integral equations | 20   |
| Assembling element stiffnesses and solution of structural problems | 22   |

### NOMENCLATURE

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
</tr>
</tbody>
</table>

### CHAPTER 3 PHILOSOPHY OF COMPUTER PROGRAMS

| Antecedents to program development | 26   |
| Programs dealing with network shape | 27   |
| Computer program Node Tape         | 29   |
| Computer program XY Ord            | 30   |
| Computer program Single line Graph | 31   |
| Programs to obtain displacements   | 32   |
| Computer program Assembly Disc     | 33   |
| Computer program Decomposition     | 41   |
| Computer program Lower Upper Treatment | 44   |
| Feature of the computer program Lower Upper Treatment | 45   |
| Computer program Displacement Check| 47   |
| Computer program Stress Strain     | 50   |

### CHAPTER 4 APPLICATION OF THE PROGRAM TO AN IDEAL PROBLEM

| Choice of test structure | 55   |
| Geometry of structure   | 57   |
| Network of Triangles    | 57   |
| Calculation of radii for equal increments of hoop stress | 59   |
| Table giving radii for constant decrements of hoop stress | 60   |
| Process through the sequence of computer programs | 61   |
| Determination of Loads at nodes | 62   |
| Agreement between Finite Element and Lamé Predictions | 63   |
| Assessment of Results   | 69   |
CHAPTER 5 DISCUSSION

On Theory 74
On the Philosophy of Computer Programs 75
On the Application of the Program to an Ideal Problem 79

CHAPTER 6 CONCLUSIONS

REFERENCES 81
APPENDICES 86
<table>
<thead>
<tr>
<th>No</th>
<th>Description</th>
<th>Page No</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Showing Limits of Sopwith's treatment of screw threads in pressure vessel</td>
<td>2</td>
</tr>
<tr>
<td>1a</td>
<td>External Hoop Strain distribution along cylindrical pressure vessel</td>
<td>4</td>
</tr>
<tr>
<td>1b</td>
<td>Comparison between Higher order and Simple 3 node triangular elements</td>
<td>8</td>
</tr>
<tr>
<td>5.3</td>
<td>2 x 3 Network of triangles</td>
<td>35</td>
</tr>
<tr>
<td>5.9</td>
<td>Pattern of Stiffness Matrix</td>
<td>37</td>
</tr>
<tr>
<td>5.1</td>
<td>Hollow Cylindrical Tube showing position of section</td>
<td>56</td>
</tr>
<tr>
<td>5.2</td>
<td>Triangular Network in section</td>
<td>58</td>
</tr>
<tr>
<td>10.1</td>
<td>Deviation of Finite element results against Lamé straight line</td>
<td>64</td>
</tr>
<tr>
<td>10.2</td>
<td>Contours of differences between Finite Element and Lamé values of Hoop Stress</td>
<td>65</td>
</tr>
<tr>
<td>10.3</td>
<td>Curves of differences against length for various radii</td>
<td>66</td>
</tr>
<tr>
<td>10.4</td>
<td>Curves of differences against length for radii near bore</td>
<td>67</td>
</tr>
<tr>
<td>10.5</td>
<td>Curve of differences against radii</td>
<td>68</td>
</tr>
<tr>
<td>10.6</td>
<td>Curves of radial displacement against length for various radii</td>
<td>70</td>
</tr>
<tr>
<td>10.7</td>
<td>Curves of stress against length for various radii</td>
<td>71</td>
</tr>
<tr>
<td>10.8</td>
<td>Curves of strain against length for various radii</td>
<td>72</td>
</tr>
</tbody>
</table>
### LIST OF APPENDICES

<table>
<thead>
<tr>
<th>Appendix</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a</td>
<td>Theory (section 1) (nomenclature page 83)</td>
<td>86</td>
</tr>
<tr>
<td>1b</td>
<td>Third compatibility equation in cylindrical polar co-ordinates</td>
<td>102</td>
</tr>
<tr>
<td>1c</td>
<td>Test that compatibility is satisfied</td>
<td>103</td>
</tr>
<tr>
<td>1d</td>
<td>Integrals (Theory section 2)</td>
<td>108</td>
</tr>
<tr>
<td>1e</td>
<td>Test of integrals by computer</td>
<td>148</td>
</tr>
<tr>
<td>1f</td>
<td>TO SHOW WHAT HAPPENS WHEN AT LEAST ONE OF THE RADIAL CO-ORDINATES OF THE VERTEX NODES BECOMES ZERO</td>
<td>160</td>
</tr>
<tr>
<td>1g</td>
<td>TO SHOW WHAT HAPPENS WHEN TWO VERTEX NODES OF A TRIANGULAR ELEMENT HAVE THE SAME RADIUS</td>
<td>164</td>
</tr>
<tr>
<td>2a</td>
<td>Assembly Disc</td>
<td>168</td>
</tr>
<tr>
<td>2b</td>
<td>Sequence of subroutines used in the computer program Assembly Disc</td>
<td>183</td>
</tr>
<tr>
<td>3a</td>
<td>Decomposition</td>
<td>188</td>
</tr>
<tr>
<td>3b</td>
<td>Sequence of subroutines used in the computer program Decomposition</td>
<td>200</td>
</tr>
<tr>
<td>4a</td>
<td>Lower-Upper Treatment</td>
<td>204</td>
</tr>
<tr>
<td>4b</td>
<td>Sequence of subroutines used in the computer program Lower-Upper Treatment</td>
<td>205</td>
</tr>
<tr>
<td>5a</td>
<td>Displacement check</td>
<td>209</td>
</tr>
<tr>
<td>5b</td>
<td>Sequence of subroutines used in the computer program Displacement check</td>
<td>210</td>
</tr>
<tr>
<td>6a</td>
<td>DERIVATION OF THE LEAST SQUARES METHOD OF FITTING A CONTINUOUS SURFACE TO A NUMBER OF DISCRETE VALUES AT NODES</td>
<td>212</td>
</tr>
<tr>
<td>6b</td>
<td>THE MATRIX FORMULATION OF THE LEAST SQUARES SOLUTION OF INCONSISTENT LINEAR EQUATIONS</td>
<td>213</td>
</tr>
<tr>
<td>6c</td>
<td>Method of determining best order of fit</td>
<td>214</td>
</tr>
<tr>
<td>6d</td>
<td>DEMONSTRATION THAT 5TH ORDER GIVES A SINGULAR MATRIX (Dr J H Wilkinson)</td>
<td>216</td>
</tr>
<tr>
<td>6e</td>
<td>Stress Strain</td>
<td>222</td>
</tr>
<tr>
<td>6f</td>
<td>Sequence of subroutines used in computer program Stress Strain</td>
<td>228</td>
</tr>
<tr>
<td>7</td>
<td>Derivation of Lame Equations</td>
<td>236</td>
</tr>
<tr>
<td>8</td>
<td>Continuity of Displacements</td>
<td>239</td>
</tr>
<tr>
<td>9</td>
<td>Notes on Computer Programs</td>
<td>273</td>
</tr>
</tbody>
</table>
FIGURES IN APPENDICES

<table>
<thead>
<tr>
<th>No</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Showing a typical six node triangle to indicate the notation used in the theory</td>
<td>87</td>
</tr>
<tr>
<td>3.1 to 3.8</td>
<td>Eight figures indicating the shape and dimensions of triangles used to test integrals by computer</td>
<td>152 to 159</td>
</tr>
<tr>
<td>5.1</td>
<td>Hollow Cylindrical Tube showing position of section</td>
<td>173</td>
</tr>
<tr>
<td>5.2</td>
<td>Triangular network in section</td>
<td>174</td>
</tr>
<tr>
<td>5.3</td>
<td>2 by 3 network of triangles</td>
<td>175</td>
</tr>
<tr>
<td>5.4</td>
<td>Vertex nodes and their ordinates</td>
<td>176</td>
</tr>
<tr>
<td>5.5</td>
<td>Path of continuous line around triangles</td>
<td>177</td>
</tr>
<tr>
<td>5.6</td>
<td>Triangle number</td>
<td>178</td>
</tr>
<tr>
<td>5.7</td>
<td>Node numbers for triangle 1</td>
<td>179</td>
</tr>
<tr>
<td>5.8</td>
<td>Displacement numbers for triangle 1</td>
<td>179</td>
</tr>
<tr>
<td>5.9</td>
<td>Pattern of Stiffness Matrix for three rows of triangles</td>
<td>180</td>
</tr>
<tr>
<td>6.1</td>
<td>Pattern of Stiffness Matrix for four rows of triangles</td>
<td>193</td>
</tr>
<tr>
<td>6.2</td>
<td>6.1 with added zero submatrices for processing</td>
<td>194</td>
</tr>
<tr>
<td>6.3</td>
<td>Block of submatrices taken in fast store for processing</td>
<td>195</td>
</tr>
<tr>
<td>6.4(a)</td>
<td>Lower arrays of submatrices after decomposition</td>
<td>196</td>
</tr>
<tr>
<td>6.4(b)</td>
<td>Upper arrays of submatrices after decomposition</td>
<td>197</td>
</tr>
<tr>
<td>6.5</td>
<td>Movement of last 3 submatrices to first 3 submatrices</td>
<td>198</td>
</tr>
<tr>
<td>6.6</td>
<td>Processing arrays of submatrices</td>
<td>199</td>
</tr>
<tr>
<td>C1</td>
<td>Network of triangles</td>
<td>242</td>
</tr>
<tr>
<td>C1D</td>
<td>Domains of Principal Nodes</td>
<td>243</td>
</tr>
<tr>
<td>C1U7</td>
<td>Radial Displacement Surface for Principal Node 17</td>
<td>244</td>
</tr>
<tr>
<td>C1U9</td>
<td>Radial &quot; &quot; &quot; &quot; &quot; 19</td>
<td>245</td>
</tr>
<tr>
<td>C1U31</td>
<td>&quot; &quot; &quot; &quot; &quot; 31</td>
<td>246</td>
</tr>
<tr>
<td>C1U33</td>
<td>&quot; &quot; &quot; &quot; &quot; 33</td>
<td>247</td>
</tr>
<tr>
<td>C1U45</td>
<td>&quot; &quot; &quot; &quot; &quot; 45</td>
<td>248</td>
</tr>
<tr>
<td>C1U47</td>
<td>&quot; &quot; &quot; &quot; &quot; 47</td>
<td>249</td>
</tr>
<tr>
<td>C1U</td>
<td>Interpenetration of the Six Radial Displacement Surfaces</td>
<td>250</td>
</tr>
<tr>
<td>C1U7.1</td>
<td>Curves of Radial Displacement versus Radius for the Domain of Principal Node 17, showing how far points of radial displacement lie off fitted surface</td>
<td>251</td>
</tr>
<tr>
<td>C1U7.5</td>
<td>Curves of Radial Displacement versus Radius for the Domain of Principal Node 19, showing how far points of displacement lie off fitted surface</td>
<td>252</td>
</tr>
<tr>
<td>C1U9.1</td>
<td>Curves of Radial Displacement versus Radius for the Domain of Principal Node 19, showing how far points of displacement lie off fitted surface</td>
<td>253</td>
</tr>
<tr>
<td>C1U9.5</td>
<td>Curves of Radial Displacement versus Radius for the Domain of Principal Node 19, showing how far points of displacement lie off fitted surface</td>
<td>254</td>
</tr>
<tr>
<td>C1V7</td>
<td>Axial Displacement Surface for Principal Node 17</td>
<td>261</td>
</tr>
<tr>
<td>C1V9</td>
<td>&quot; &quot; &quot; &quot; &quot; 19</td>
<td>262</td>
</tr>
<tr>
<td>C1V31</td>
<td>&quot; &quot; &quot; &quot; &quot; 31</td>
<td>263</td>
</tr>
<tr>
<td>C1V33</td>
<td>&quot; &quot; &quot; &quot; &quot; 33</td>
<td>264</td>
</tr>
<tr>
<td>C1V45</td>
<td>&quot; &quot; &quot; &quot; &quot; 45</td>
<td>265</td>
</tr>
<tr>
<td>C1V47</td>
<td>&quot; &quot; &quot; &quot; &quot; 47</td>
<td>266</td>
</tr>
<tr>
<td>C1V</td>
<td>Interpenetration of the Six Axial Displacement Surfaces</td>
<td>267</td>
</tr>
<tr>
<td>C1V7.1</td>
<td>Curves of Axial Displacement versus Axial Distance for the Domain of Principal Node 17, showing how far points of axial displacement lie off fitted surface</td>
<td>268</td>
</tr>
<tr>
<td>C1V7.5</td>
<td>Curves of Axial Displacement versus Axial Distance for the Domain of Principal Node 19, showing how far points of axial displacement lie off fitted surface</td>
<td>269</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

Since the attempt by Galileo Galilei in 1638 to predict stresses in a beam, experimental and analytical methods have both become more powerful by virtue of increased understanding of structural behaviour and more recently the availability of large computers. Analytical solutions developed during two periods. The first period was at the beginning of the eighteenth century, which gave the Bernoulli-Euler-Young equations for the solution to the bending of an elastic beam. This is a very simple solution which has been found to be generally applicable to a wide range of structural problems in which the elastic behaviour of each element of the structure can be adequately described by simple bending formulae. The second period (in the nineteenth century) yielded the more detailed and exact analysis proposed by Navier, Stokes, Green, Lamé, and Airy. This provided a more elegant and detailed solution for the behaviour of individual elements but it has not found general application in structural analysis owing to the increased complexity of obtaining solutions for large arrays of structural elements.

Although Leonardo Da Vinci conducted experiments on the strength of materials; detailed experimental work on structures really began in 1816 with Sir David Brewster's discovery that glass and other transparent bodies became doubly refractive under stress. This was the beginning of the photoelastic method. In a typical birefringent material two photo-elastic phenomena are observed (by normal observations) in polarised light: the variation of colour from which the magnitude of principal stress difference can be derived; and the dark areas that appear where the principal axes coincide with the polaroid axes. Using the difference of principal stress in conjunction with their directions, the stress distribution through the structure may be determined. Thus, this provides a means of comparing Airy-stress function computations with
the stress values derived from a physical model. In practice comparisons 
are not exact because analytical solution refer to idealised models whilst 
experimental results suffer from errors of observation.

An illustration of these limitations is provided by an investigation of stresses in a cylindrical pressure vessel whose ends were closed by screw threaded plugs (see Figure 1). The maximum stresses occur in the screw threaded region. Both analytical and experimental methods were used to estimate these greatest stresses. The analytical treatment of screw threads proposed by D G Sopwith (1949) used stress function which could not deal with such geometric detail as root fillet radii. It took no account of the local increase in flexibility due to the thread. It did not consider moments or shear loads applied to the thick wall of the nut or bolt; in fact it completely ignores the material outside the length of thread engagement (see Figure 1). The experimental investigation involved testing a photo elastic model which showed the distorted shape indicated in Figure 2. In order to obtain a sufficient number of isochromatic fringes in low stressed areas, the model could be distorted to such an extent, that part of the thread was no longer engaging on the effective diameter in which case the deformation behaviour would become geometrically non linear. Thus neither the purely analytical nor the experimental investigation could be relied upon to give an exact solution to the real stress problem.

**Finite Difference and Finite Element Method**

Only a few practical problems are capable of analysis by solution of the governing differential equations. In both the finite difference method and the finite element method, a system which is varying continuously is replaced by an "equivalent lumped system". The problem is thus changed from solving differential equations to solving a set of simultaneous
equations. So that if a function $f$ is varying continuously over a given domain, it is replaced by approximations to $f$ at discrete points within this domain.

Advantages of the Finite Difference Method

The advantages are:

(a) generally it is not difficult to form the finite difference equations;
(b) its coefficients are simple integers except at the boundaries;
(c) its truncation errors may be calculated;
(d) its matrix is usually symmetric, therefore only the top half needs to be stored, saving computer space;
(e) its matrix is banded, hence there is a further saving of computer space;
(f) its matrix is positive definite, with its associated desirable properties, e.g., it does not require rows or columns to be exchanged to obtain a solution; and
(g) if the finite difference is dealing with the Laplace operator, its matrix is said to possess "property A" which has desirable features.

Disadvantages of Finite Difference Method

The disadvantages are:

(a) the finite difference coefficients are more difficult to form if the boundaries are angled or curved;
(b) the solution of the equations is more involved if the domain is subjected to mixed boundary conditions;
(c) singularities present difficulties;
(d) the position of the boundaries must be defined;
(e) errors increase with a graduated mesh;
(f) there is no criterion to guide the formation of an optimum graduated mesh;
(g) a large number of equations is required; and
(h) the method is generally restricted to one material;

Advantages of the Finite Element Method

The advantages are:

(a) the elements fit the boundaries of the structure;
(b) each element could have different material properties;
(c) its matrix is symmetric;
(d) its matrix is banded; and
(e) its matrix is positive definite.

Disadvantage of the Finite Element Method

The disadvantages of the method are:

(a) if the element stiffness is formed with constant strain it is not consistent with the varying strain field of the structure;
(b) if the element stiffness is formed with varying strain, element boundaries will not necessarily be completely compatible;
(c) the method is not exact and there is no mathematical way of estimating the degree of approximation;
(d) there is no guarantee that on decreasing the element size the results will converge to the correct solution;
(e) the solution of the equations is more involved with mixed boundary conditions;
(f) singularities present difficulties;
(g) the position of boundaries must be defined;
(h) a large variation in the size of the elements results in a large ratio of stiffnesses which can prove troublesome in solving the simultaneous equations;
and (i) a large number of equations is required to represent a complex structure.

The need for a particular type of axisymmetric finite element

The occurrence of screw threads is sufficiently common to warrant the development of a finite element program which would deal with the aspects of loading and stiffness which the theoretical analysis of Sopwith could not accommodate.

For such geometry an axisymmetric element is required, the simplest form of which is of triangular section. The simple three vertex nodded triangular element is not sufficient to deal with screw threads. Whereas some of the mid side nodes of the six-node triangular element could represent the contact points at the effective diameter of the screw thread.

It was this initial consideration which revealed the need for a six noded element; another is illustrated in Figure 1b, which shows a truncated conical shell subjected to edge loads of meridional forces and moments. The right hand side shows the histogram type of stress distribution typical of the three node element and the left hand side shows the continuous curve stress distribution typical of the six node element. These comparisons are made with the same total number of nodes. Simple Bernouli-Euler considerations predict stress resisting applied moments would vary linearly across the section. Thus for the same total number of nodes (and hence approximately the same amount of work in treatment) one would expect the six node element to produce results closer to the truth than those of the three-node element.

The literature search (Circa 1968) did not reveal the existence of a program involving a six-noded-triangular axisymmetric element. A detailed explanation of the formation of the stiffness equations for three node triangular element was given by Zienkiewicz (1967) but this omitted certain terms.
COMPARISON BETWEEN
HIGHER ORDER ELEMENT AND SIMPLE 3 NODE TRIANGULAR ELEMENT

STRESS DISTRIBUTION

CONICAL SHELL

STRESS DISTRIBUTION

DISPLACEMENT SURFACE

RADIAL DISPLACEMENT FUNCTION
(In terms of co-ordinates R & Z)

\[ U = a_1 + a_2 r + a_3 z + a_4 r^2 + a_5 rz + a_6 z^2 \]

RADIAL STRAIN

\[ \text{Err} = \frac{dU}{dr} = a_2 + 2a_4 r + a_5 z \]

STRAIN VARYING THROUGHOUT ELEMENT

\[ \text{Err} = \frac{dU}{dr} = a_2 \]

CONSTANT STRAIN THROUGHOUT TRIANGULAR

FIGURE 1b
Zienkiewicz's equations without these terms would lead to errors when two vertex nodes of a given triangular element have the same radius (see Appendix 1g).

The requirements for the program were that it should be:
(a) flexible in order to allow for different types of material, support fixing and loading;
(b) moderately accurate and reliable; and
(c) include a graphic display to aid interpretation of results.

The appreciation of the need for (a) and (c) stem from experimental experience. The achievement of (b) depends partly on knowledge of numerical analysis to avoid errors, and numerical difficulties (discussed later); but also on experimental experience in structural idealization. Graphical presentation of the deformed shape helps understanding of elastic deformation behaviour.

Several additional objectives were kept in mind whilst writing the program. These were:
(d) ease of program adaptation to other computers. If the program was to be easily run on various computers this necessitated writing the program in basic language (Fortran 2) rather than standard language (Fortran 4) whenever possible. This meant writing each of the 10,000 or so cards specifically.
(e) Data preparation external to the main program. This enabled external geometry to be generated in the same way as for Numerically Controlled Machine Tools, and checks to be made on the loads on the nodes (both radial and axial summations and individual vector summation to check normality to the surface).
(f) Mesh generation of network. The large number of elements ruled out the manual formation of the network and necessitated:
(g) Graphical verification of data prior to running main program. This was performed on the Stromberg-Carlson graphical plotter.

(h) Roundoff error avoidance on conversion from decimal to binary.

(i) Not in core matrix storage. To allow for large numbers of elements entailed matrices so large that they were stored on magnetic tapes. This necessitated developing new procedures for the lower-upper decomposition of the matrices.

(j) Dynamic storage. This enables the available storage space to be used in the most economic ways for different problems and thus minimize the computer time taken for solution.

(k) Step by step solution.

(l) Facility to restart for different conditions of loading and restraint.

(m) Plotting of distorted shape. This was done on the Calcomp graphical plotter.

With the formation of the stiffness matrix for the six node element, 15 integrals had to be evaluated algebraically and a 6 x 6 matrix inverted. It was realised that numerical integration would lead to errors so it was decided to formulate these integrals algebraically and to form these equations so as to minimise roundoff errors in their evaluation on a digital computer. The 6 x 6 matrix which needed to be inverted had some of its largest (matrix) elements off the principal diagonal, with the result that a simple twelve statement algorithm proved unsuitable and a new subroutine had to be written with a consequent increase in the number of statements of more than tenfold.

Perhaps this introduction should be completed by comparing the application of the Finite Element and photoelastic methods to a particular problem.
The investigation using photo elastic models of the pressure vessel mentioned earlier took six months. There is doubt about the validity of results predicted by this investigation because of geometric non-linearity (caused by movement of the contact points at the threads), nevertheless, although it is difficult to identify experimental errors, these would be expected to be in the region of 5\% to 10\%. Provided a suitable program is available the corresponding finite element analysis takes about four months. Thus the cost of the two methods are comparable, ie the finite element is not necessarily cheaper or quicker. It requires knowledge of structural idealisation and where to expect high stresses. The advantages of finite element method are:-

(a) modelling requiring small displacements can be undertaken more satisfactorily;
(b) it enables a parametric study to be undertaken cheaper and quicker than with physical models;

is it allows:-

(i) different types of material to be examined;
(ii) different types of geometry to be investigated;
(iii) different types of restraint to be applied; and
(iv) different types of loads to be applied.

and (c) thermal and body force problems can be studied just as easily as static problems.

These advantages provided the incentive to undertake the very extensive study which is required to develop a new finite element program.
CHAPTER 2

THEORY OF FINITE ELEMENT

Definition of Finite Element Method

The finite element method is used in structural mechanics, fluids, heat flow, and in electrical technology. In fact in any problems involving continua it is used where either the differential equation is unknown or where a solution cannot be obtained for particular boundary conditions. The essence of the method is to model continua as discrete elements of finite size joined at certain points or "nodes".

When the finite element method is applied to structural mechanics certain conditions have to be satisfied. Firstly, all the forces at each node have to be in equilibrium. Secondly, the part of each element touching a node must have the same global displacement under load, otherwise the elements would not continue to meet at that node. Thirdly, the element stiffness must be related to its geometry and material properties.

Finite elements can be thought of as a network of springs joined at their ends. The deflection of an individual spring multiplied by its stiffness gives the force on that spring. Summation of the individual deflections up to a particular node gives the global deflection of that node. Thus the forces at each node and the deflections can be determined. In structural continua, knowing the deflections or rate of change of deflections with distance strains may be obtained. Hence knowing the stress strain relationships the stress distribution can be determined.

Formulation of the spring stiffness equations

The major portion of the work of the theory related to the finite element method is in the formulation of the spring stiffness equations in terms of the element geometry and its material stiffness properties. There are forces applied to nodes which produce displacements at the nodes. The spring stiffnesses
express the relationship between forces and displacements. A force at one node affects the displacements at all the nodes. By Hooke's law the displacements at all the nodes due to a force at a particular node are proportional to that force. The principle of superposition applies if the material is elastic (Southwell 1950), and then the total displacement at any node with all the forces acting is the vector sum of the individual displacements that node would attain due to the forces acting independently. Transposing this, the force at any particular node is given by the sum of the products of coefficients and their associated displacements. This may be written algebraically as:

\[ a_{11} \delta_1 + a_{12} \delta_2 + a_{13} \delta_3 + \cdots + a_{1n} \delta_n = F_i \]

where \( F_i \) is the force at the \( i \)th node, \( n \) is the number of nodes. \( \delta_1, \delta_2, \delta_3, \cdots \delta_n \) are the displacements at the nodes 1, 2, 3, \( \cdots \) \( n \), \( a_{11}, a_{12}, a_{13} \) etc. are the coefficients such that \( a_{ij} \) is the coefficient of the \( j \) node displacement (\( \delta_j \)) relating to the force at the \( i \)-th node. As the force is in pounds and the displacement is in inches, it follows that the coefficients are in units of pounds/inch. These coefficients are the spring stiffness coefficients.

It is the formulation of the equations specifying these spring stiffness coefficients in terms of the element geometry and the material stiffness properties which is required. This is achieved by the following sequence. The displacement function is defined: from this can be derived strains. The constitutive relationships use the material stiffness properties to give the stresses from strains. The strain energy per unit volume is half the product of strains and stresses. Integrating the strain energy per unit volume over the complete volume of the element gives the total strain energy for the element. Dividing the total strain energy by half the displacements gives the forces. Finally dividing the forces by the displacements gives the spring stiffness coefficients.
Displacement functions and strain

The displacement function is defined by the simplest displacement function which can satisfy the displacements at the nodes. For example:— the simplest displacement function which can pass through the three nodal displacements is a plane surface. This displacement surface may be represented by the equation

\[ u = A_1 + A_2 r + A_3 z \]

Whereas, the simplest displacement surface which can be sure to pass through the point displacements at the six nodes of a triangle (ie triangles with mid-side nodes as well as vertex nodes) is defined as a quadratic function of the form:— radial displacement

\[ u = A_1 r^2 + A_2 r + A_3 + A_4 z^2 + A_5 z + A_6 rz \]

and axial displacement

\[ w = A_7 r^2 + A_8 r + A_9 + A_{10} z^2 + A_{11} z + A_{12} rz \]

If the radial displacements were known at each of the six nodes then six simultaneous equations could be formed which could be solved for the coefficients \( A_1 \) to \( A_6 \). Similarly if the axial displacements were known at the six nodes the coefficients \( A_7 \) to \( A_{12} \) could be determined. Hence, knowing the coefficients \( A_1 \) to \( A_{12} \) the radial and axial displacements could be found at any point within the triangle. Also the slopes of those displacement surfaces can be determined at any point. The strains are related to the displacements or to the rate of change of displacement with distance. Because the element and the loading are axisymmetric the hoop strain is a principal strain and so there are no hoop-radial or hoop-axial shear strains. Of the four remaining strains, three strains are direct strains:— hoop \( e_{\theta\theta} \), radial \( e_{rr} \), and axial \( e_{zz} \). These strains may be expressed as functions of displacement as follows:

\[ e_{\theta\theta} = \frac{u}{r} \]
\[ e_{rr} = \frac{2u}{\partial z^2} \]
\[ e_{zz} = \frac{\partial w}{\partial z} \]
\[ e_{rz} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial r} \]

Hence the strains in terms of the displacement coefficients and co-ordinates \((r,z)\) become

\[ e_{\theta\theta} = A_1 r + A_2 + \frac{A_3}{r} + A_4 \frac{z^2}{r} + A_5 \frac{z}{r} + A_6 r z \]
\[ e_{rr} = 2A_1 r + A_2 + A_6 z \]
\[ e_{zz} = 2A_4 z + A_5 z + A_6 r + 2A_7 r + A_8 + A_{12} z \]
\[ e_{rz} = 2A_4 z + A_5 + A_6 r + 2A_7 r + A_8 + A_{12} z \]

**Determination of stresses from strain**

Knowing the strains the stresses need to be derived. The strain in one direction is equal to the stress in that direction divided by Young's Modulus of Elasticity \(E\), minus Poisson's Ratio \(\nu\) times the stresses in the orthogonal directions divided by Young's Modulus of Elasticity. Thus if \(\sigma_{\theta\theta}, \sigma_{rr}, \sigma_{zz}\) are the hoop, the radial, and the axial stresses respectively then the three direct strains may be written:

\[ e_{\theta\theta} = \sigma_{\theta\theta} - \nu \frac{\sigma_{rr}}{E} - \nu \frac{\sigma_{zz}}{E} \]
\[ e_{rr} = \sigma_{rr} - \nu \frac{\sigma_{\theta\theta}}{E} - \nu \frac{\sigma_{zz}}{E} \]
\[ e_{zz} = \sigma_{zz} - \nu \frac{\sigma_{\theta\theta}}{E} - \nu \frac{\sigma_{rr}}{E} \]

This may be written more concisely and conveniently in matrix form as

\[
\begin{pmatrix}
    e_{\theta\theta} \\
    e_{rr} \\
    e_{zz}
\end{pmatrix}
= \begin{pmatrix}
    1 & -\nu & -\nu \\
    -\nu & 1 & -\nu \\
    -\nu & -\nu & 1
\end{pmatrix}
\begin{pmatrix}
    \sigma_{\theta\theta} \\
    \sigma_{rr} \\
    \sigma_{zz}
\end{pmatrix}
\]

Here the strains are still in terms of stresses, so in order to obtain the stresses in terms of strain the matrix equation is transformed into the following...
This may be written as ordinary algebraic equations as

\[
\begin{align*}
\sigma_{\theta\theta} &= \frac{E}{(1 + v)(1 - 2v)} (1 - v)e_{\theta\theta} + ve_{rr} + ve_{zz} \\
\sigma_{rr} &= \frac{E}{(1 + v)(1 - 2v)} ve_{\theta\theta} + (1 - v)e_{rr} + ve_{zz} \\
\sigma_{zz} &= \frac{E}{(1 + v)(1 - 2v)} ve_{\theta\theta} + ve_{rr} + (1 - v)e_{zz}
\end{align*}
\]

So far the relationship between the shear stress and the shear strain has not been included. Shear strain is related to shear stress as follows:

\[
e_{rz} = \frac{\sigma_{rz}}{G}
\]

where \(\sigma_{rz}\) is the radial-axial shear stress and \(G\) is the Modulus of Rigidity.

As \(G = \frac{E}{2(1 + v)}\) the shear relationship may be written \(\sigma_{rz} = \frac{E}{2(1 + v)} e_{rz}\)

which may be included in the matrix equation relating stress in terms of strain in the following manner.

\[
\begin{pmatrix}
\sigma_{\theta\theta} \\
\sigma_{rr} \\
\sigma_{zz} \\
\sigma_{rz}
\end{pmatrix} = \frac{E(1 - v)}{(1 + v)(1 - 2v)} \begin{pmatrix}
1 & \frac{v}{1 - v} & \frac{v}{1 - v} & 0 \\
\frac{v}{1 - v} & 1 & \frac{v}{1 - v} & 0 \\
\frac{v}{1 - v} & \frac{v}{1 - v} & 1 & 0 \\
0 & 0 & 0 & \frac{(1 - 2v)}{2(1 - v)}
\end{pmatrix} \begin{pmatrix}
e_{\theta\theta} \\
e_{rr} \\
e_{zz} \\
e_{rz}
\end{pmatrix}
\]

The strain vector on the right hand side of this equation may be replaced in terms of co-ordinates and displacement coefficients (as shown previously); hence stresses may be expressed in terms of material stiffness properties, co-ordinates and displacement coefficients.
Determination of spring stiffness matrix

To convey the method used to obtain strain-energy and hence the spring stiffness equations, it is necessary to use matrix notation in a symbolic form.

Reverting to the displacement equation, for the six nodes the radial displacements are:

\[
\begin{align*}
\delta(u)_1 &= A_1 r_1^2 + A_2 r_1 + A_3 + A_4 z_1^2 + A_5 z_1 + A_6 r_1 z_1 \\
\delta(u)_2 &= A_1 r_2^2 + A_2 r_2 + A_3 + A_4 z_2^2 + A_5 z_2 + A_6 r_2 z_2 \\
\delta(u)_3 &= A_1 r_3^2 + A_2 r_3 + A_3 + A_4 z_3^2 + A_5 z_3 + A_6 r_3 z_3 \\
\delta(u)_4 &= A_1 r_4^2 + A_2 r_4 + A_3 + A_4 z_4^2 + A_5 z_4 + A_6 r_4 z_4 \\
\delta(u)_5 &= A_1 r_5^2 + A_2 r_5 + A_3 + A_4 z_5^2 + A_5 z_5 + A_6 r_5 z_5 \\
\delta(u)_6 &= A_1 r_6^2 + A_2 r_6 + A_3 + A_4 z_6^2 + A_5 z_6 + A_6 r_6 z_6
\end{align*}
\]

Where \(\delta(u)_i\), \(r_i\), \(z_i\) are the radial displacement, the radial and axial co-ordinates of the \(i\)th node. In matrix form this is

\[
\begin{pmatrix}
\delta(u)_1 \\
\delta(u)_2 \\
\delta(u)_3 \\
\delta(u)_4 \\
\delta(u)_5 \\
\delta(u)_6
\end{pmatrix} =
\begin{pmatrix}
r_1^2 & r_1 & 1 & z_1^2 & z_1 & r_1 z_1 \\
r_2^2 & r_2 & 1 & z_2^2 & z_2 & r_2 z_2 \\
r_3^2 & r_3 & 1 & z_3^2 & z_3 & r_3 z_3 \\
r_4^2 & r_4 & 1 & z_4^2 & z_4 & r_4 z_4 \\
r_5^2 & r_5 & 1 & z_5^2 & z_5 & r_5 z_5 \\
r_6^2 & r_6 & 1 & z_6^2 & z_6 & r_6 z_6
\end{pmatrix}
\begin{pmatrix}
A_1 \\
A_2 \\
A_3 \\
A_4 \\
A_5 \\
A_6
\end{pmatrix}
\]

In matrix symbolic form this becomes

\[
\{\delta_u\} = [\delta A] \{A\}
\]

Similarly the axial displacement in matrix symbolic form is

\[
\{\delta_w\} = [\delta A] \{A\}
\]

combining these two gives
\[
\begin{bmatrix}
\delta_u \\
\delta_w \\
\end{bmatrix} = \begin{bmatrix}
\delta A & 0 \\
6x6 & \delta A \\
0 & 6x6 \\
\end{bmatrix} \begin{bmatrix}
\{A\} \\
1+12 \\
\end{bmatrix}
\]

or
\[
\{\delta\} = [\delta A] \{A\} \\
12x12 \ 1+12
\]

Transposing
\[
\{A\} = [\delta A]^{-1} \{\delta\}
\]

so that
\[
\begin{bmatrix}
u \\
w \\
\end{bmatrix} = [uA] \{A\}
\]

and substituting \([\delta A]^{-1} \{\delta\}\) for \(\{A\}\)
\[
\begin{bmatrix}
u \\
w \\
\end{bmatrix} = [uA] \ [\delta A]^{-1} \{\delta\}
\]

The strains in matrix form are
\[
\begin{bmatrix}
e_{\theta\theta} \\
e_{rr} \\
e_{zz} \\
e_{rz} \\
\end{bmatrix} = \begin{bmatrix}
r & 1/r & z^2/r & z/r & z \\
2r & 1 & z \\
2r & 1 & r & 2r & 1 \\
\end{bmatrix} \begin{bmatrix}
A_1 \\
A_2 \\
A_3 \\
A_4 \\
A_5 \\
A_6 \\
A_7 \\
A_8 \\
A_9 \\
A_{10} \\
A_{11} \\
A_{12} \\
\end{bmatrix}
\]
which in matrix symbolic form are

\[ \{e\} = [eA] \{A\} \]

Substituting \([\delta A]^{-1}\{\delta\}\) for \(\{A\}\)

\[ \{e\} = [eA] [\delta A]^{-1}\{\delta\} \]

The stresses related to the strains by the material stiffness properties in matrix symbolic form are

\[ \{\sigma\} = [\sigma e] \{e\} \]

so that stresses in terms of material stiffness properties, co-ordinates and displacement may be obtained by writing \([eA][\delta A]^{-1}\{\delta\}\) for \(\{e\}\) thus

\[ \{\sigma\} = [\sigma e][eA][\delta A]^{-1}\{\delta\} \]

Let \([e\delta]\) stand for \([eA][\delta A]^{-1}\), and \([\sigma \delta]\) for \([\sigma e][eA][\delta A]^{-1}\) so that strain

\[ \{e\} = [e\delta] \{\delta\} \]

and stress \(\{\sigma\} = [\sigma \delta] \{\delta\} \)

The external work done by the displacement of the nodal point forces is equated to the integral internal strain energy of the product of strain and stress over the volume of the triangular annulus

\[ \frac{1}{2} [\delta]\{q\} = \frac{1}{2} \int [e]\{\sigma\} \, dv \]

where \([\delta]\) is a row vector of the displacements

\[ \{q\} \]

is a column vector of the nodal point forces

\[ \{e\} \]

is a row vector of the strains

\[ \{\sigma\} \]

is a column vector of the stresses

and \(dv\) is the elemental volume

as \(\{e\} = [e\delta] \{\delta\}\), so \([e] = \{e\}^T = [\delta][e\delta]^T\) and \(\{\sigma\} = [\sigma \delta] \{\delta\}\)

so that

\[ [\delta]\{q\} = [\delta] \left( \int [e\delta]^T [\sigma \delta] \, dv \right) \{\delta\} \]

Premultiplying both sides of the equation by the inverse of the displacement vector \([\delta]\), gives the force vector \(\{q\}\) on the left hand side and its equivalent on the right hand side of the equation thus
\[ \{ q \} = \left( \int [e\delta]^T [\sigma\delta] \; dv \right) \{ \delta \} \]

and as the nodal point forces are equal to the products of stiffness and displacements, i.e. \([k]\{\delta\} = \{q\}\]

substituting

\[ [k]\{\delta\} = \left( \int [e\delta]^T [\sigma\delta] \; dv \right) \{ \delta \} \]

Post multiplying both sides of the equation by the inverse of the displacement vector \{\delta\}, gives the stiffness matrix on the left hand side and its equivalent on the right

\[ [k] = \int [e\delta]^T [\sigma\delta] \; dv \]

as

\[ [e\delta]^T = ([eA][\delta A]^{-1})^T = ([\delta A]^{-1})^T [eA]^T \]

and

\[ [\sigma\delta] = [\sigma e] [eA] [\delta A]^{-1} \]

so that

\[ [k] = ([\delta A]^{-1})^T \left( \int [eA]^T [\sigma e] [eA] \; dv \right) [\delta A]^{-1} \]

\([\delta A], [\delta A]^{-1}\) and \(([\delta A]^{-1})^T\) are calculated in the computer program.

The 12 by 12 matrix arising from the product of the matrices \([eA]^T [\sigma e] [eA]\) is formed in the appendix together with the 15 integrals arising from this matrix. These formulations are used in the computer program to derive the element stiffnesses.

**Original Specific features concerning the integral equations**

It is desired to draw attention to three original features in the calculation of the integrals. These concern respectively: improving the accuracy; generalising the formula to enable elements to touch the axis; and allowing sides of elements to be parallel to the axis.

The first feature concerns formulating the equations to minimise round-off errors. In the working of the first ten integrals:- the equation is formed for the integral under each of the three sides independently. These three equations are added, eliminating some terms and giving one equation for the complete triangle. It is convenient for simplicity, in the resulting
equations, to call the radial ordinates of the three vertex nodes, a, b, and c and the axial ordinates x, y, and z. The equations in a, b, c, x, y, and z are expanded. The containment of the expression for the complete triangle in one equation, and the expansion of these equations were attempts to minimise round-off and difference errors. The results of problems indicate that the improvement in accuracy was achieved.

The second feature concerns triangles which have one or more vertex nodes on the axis of symmetry. In the last five integrals which are of the form \( \int \int \frac{r^n}{r} \, dz \, dr \) the exponent n is respectively 0, 1, 2, 3, 4 in turn. These integrals result in equations which contain the term \( \log r^i \), where i is successively 1, 2, or 3 covering the radial ordinates of the three vertex nodes. As each value of \( r^i \) for the three nodes is checked to make sure they are not zero before taking the logarithms, it is more convenient for the last five integrals to be computed under each of the three straight lines independently. If \( r^i = 0 \) then the terms contain \( \log r^i \) are replaced by zero. An explanation of this can be found in the appendix under "To show what happens when at least one of the radial ordinates of the vertex nodes become zero". This enables elements to touch the axis.

The third feature concerns triangular elements which have two vertex nodes of the same radius, i.e., one side is parallel to the axis. In the algebra of the integrals, the vertex nodes were taken as i, j, and k and the radial ordinates \( r_i, r_j, \) and \( r_k \), and the axial ordinates \( z_i, z_j, \) and \( z_k \). The equation for slope and the equation for intercept of the straight line had a denominator of \( r_j - r_i \), where \( r_i \) and \( r_j \) defined the beginning and end vertex nodes of a side of a triangle. The vertices and the sides were taken in clockwise order. To ensure that division by \( (r_j - r_i) \) did not result in infinity \( (r_j - r_i) \) had to be checked for every side to make sure it was not zero. Care has to be taken with the eleventh integral \( \int \int \frac{1}{r} \, dz \, dr \). Under the line i to j the integral became

\[
A_{ij} \log r_j - A_{ij} \log r_i + z_j - z_i \quad \ldots \quad S(11)
\]
Algebraically it follows the z terms which for the successive lines are: \((z_j - z_i), (z_k - z_j),\) and \((z_i - z_k)\) cancel, ie the summation appears to be zero (Zienkiewicz 1967, page 53). If two radial ordinates are the same, ie if \(r_j = r_i\) the result is tantamount to the whole of the equation \(S(11)\) but making \(A_{ij}\) zero. This is explained in the appendix under "To show what happens when two vertex nodes of a triangular element have the same radius". This enables sides of elements to lie parallel to the axis.

Several tests were made of the algebraic equations of the fifteen integrals by runs of a computer program for different geometry and size of triangles. These checks are described in the appendix under "Tests of Integrals by computer". These checks did not fault the equations and provided comparisons of accuracy of these equations on different computers.

Assembling element stiffnesses and solution of structural problems

After the resulting algebraic equations for each of the fifteen integrals have been determined, these equations may be evaluated for given dimensions and material properties, and inserted in the appropriate positions in the 12 x 12 matrix: thus the stiffness matrix for an individual element is formed. The next step is from the individual element stiffness matrices to form the assembly stiffness matrix. Generality is maintained during this process by assuming no restraints are applied. With any linkage of elements the nodes are in equilibrium, in that either the internal forces are balanced between themselves or with the applied loads. It is convenient to make provision for two orthogonal loads (ie radially and axially) to be applied at every node and in the directions where there are no external loads at nodes to take the value of the applied load to be zero. Hence for any individual six-node-element its twelve simultaneous equations may be written with the loads on the right-hand side, and on the left the twelve unknown displacements (each preceded by its stiffness coefficient), ie in the form

\[
a_{i1} \delta_1 + \cdots + a_{i12} \delta_{12} = F_i
\]

(whwere \(i\) is from 1 to 12).
At each common node of adjacent elements their two orthogonal displacements (radial and axial) are common. In either direction at each of these nodes their forces are additive, therefore as their displacements are common it follows that their spring stiffnesses are additive. The effect of this is: in the areas where both the displacement numbers are common and the force numbers are common for two or more structural elements, their matrices overlap, and in these areas where the matrices overlap the matrix elements are algebraically additive. This is illustrated in detail in the section in the appendix on the computer program "Assembly Disc".

To prevent singularity at least one axial restraint is required. No radial restraints need be applied. In the directions where a restraint is applied the displacement is zero. Once the restraint(s) has been applied, the elimination procedure on the left hand side of the simultaneous equations can begin. Details of this are given in the appendix on the computer program "DECOMPOSE".

For each of any number of sets of applied (external) forces the corresponding displacements of all the nodes may be found, (see appendix on computer program "Lower Upper Treatment").

Surfaces are fitted to (separately for the radial and axial) displacements for domains of groups of six triangular elements adjacent to certain nodes. From these surfaces the displacements and the rate of change of displacements are found and hence the strains. Knowing the strains and the material stiffness properties the stresses are determined. How and why this is done is explained in the appendix in the section on the computer program "STRESS-STRAIN".
**NOMENCLATURE**

*F_i* = force in lbs. at *i*th node.

*δ_i* = displacement of the *j*th node.

*a_{ij}* = coefficient of the *j*th node displacement (δ_j) relating to the force at the *i*-th node (F_i).

*r* = radial ordinate of a point.

*z* = axial ordinate of a point.

*u* = radial displacement

*W* = axial displacement.

*A_i* = coefficient of displacement function.

*e_{θθ}* = hoop strain.

*e_{rr}* = radial strain.

*e_{zz}* = axial strain

*e_{rz}* = radial-axial shear strain,

*σ_{θθ}* = hoop stress.

*σ_{rr}* = radial stress.

*σ_{zz}* = axial stress

*σ_{rz}* = radial-axial shear stress.

*E* = Young's modulus of Elasticity.

*v* = Poissons ratio.

*δ_u* = radial displacement at six nodes.

*δW* = axial displacement at six nodes.

*δ* = δ_u and δW.

*{δ_u}* = column vector of radial displacements at six nodes.

*{δW}* = column vector of axial displacement at six nodes.

*{δ}* = column vector of radial and axial displacements at six nodes.

*{A}* = column vector of quadratic displacement function coefficients.

*[δA]* = 6 x 6 matrix relating 6 nodal displacements to 6 6 x 6 coefficients. (functions of r and z).
\([\delta A]\) = 12 \times 12 \) matrix relating 12 nodal displacements to 12 coefficients.

\(\{u\}\) = column vector of radial and axial displacements anywhere within triangle.

\([UA]\) = 12 \times 12 \) matrix relating 12 displacements to 12 coefficients.

\(\{e\}\) = column vector of four strains.

\([eA]\) = 4 \times 12 \) matrix relating strains to displacement coefficients

\(\{\sigma\}\) = column vector of four stresses.

\([\sigma e]\) = 4 \times 4 \) matrix relating stresses to strains

\([e\delta]\) = 4 \times 12 \) matrix relating 4 strains to 12 displacements

\([\sigma\delta]\) = 4 \times 12 \) matrix relating 4 stresses to 12 displacements.

\(\delta\) = row vector of 12 displacements.

\(e\) = row vector of 4 strains.
CHAPTER 5

PHILOSOPHY OF COMPUTER PROGRAMS

Antecedents to program development

The type and sequence of the individual computer programs were patterned on the kind and order of stages adopted in experimental stress analysis; which are as follows:

(1) the shape of the model (a structural idealisation of the prototype) is defined;

(2) its scale is decided;

(3) drawings of the model are prepared;

(4) the material of the model is decided and the model manufactured;

(5) each mode of fixing or supporting the model is specified and implemented;

(6) a number of types of loading (simulating those to which the prototype will be subjected to) are specified in terms of position, direction, and magnitude. These are applied;

(7) after subjecting the model to loading, critical dimensions are measured to ascertain if the general deformations are as expected;

(8) the stresses are deduced: from fringe observations of photoelastic models, from strain readings of strain gauged model;

(9) the determined values of stress are used to verify that they satisfy equilibrium of forces. The displacement functions are examined to substantiate that they are continuous and compatible with each other as regards to slope.

The sequence of computer programs written to parallel the stages of experimental stress analysis are as follows:
(1) Triangle Tape;
(2) Node Tape or Polar-node Tape or Node Tape from Boundary Points;
(3)(a) X-Y ordinates;
(3)(b) Single-line Graph;
(4) Assembly Disc;
(5) Decompose;
(6) Lower-Upper Treatment;
(7) Displacement Check;
(8) Strain-Stress

At stage (3) the geometry and size have been verified. At stage (4) the material is specified: the program Assembly Disc may be run any number of times for different materials for the same geometry without having to re-run the first four programs. At stage (5) the restraints are applied (these depend on how it is desired to support, suspend, and/or restrain the structure): this program may be re-run for each arrangement of restraints without having to re-run any of the preceding programs. At stage (6) the loads are applied: the program Lower-Upper Treatment re-run for each different loading pattern without having to re-run any of the preceding programs. At stage (7) a necessary check of displacements is carried out to ensure they are consistent with the applied loads. This program (Displacement Check) also gives the reactions at the supports or points of restraint. At stage (8) the strains and stresses are determined.

It can be seen from the foregoing sequence that the code is very flexible and will facilitate parametric study.

Programs dealing with Network Shape

The first four programs: Triangle tape; Node Tape; X-Y ord; single-line graph; deal with the geometry of the model. They deal with the mesh
generation and with the graphical verification before the data generated is applied to subsequent programs. In the network generation for a given number of intervals across the thickness, and for a given number of spaces along the meridian, the vertex node numbers of the triangles are the same regardless of the graduation across thickness or along meridian, and also regardless of the geometry of the body. Thus for a given number of spaces across the thickness and spaces along the meridian the vertex node numbers of the triangles can be determined. Therefore they are generated separately from the geometry and graduation of the network (dealt with in Node Tape or its equivalent). In fact the same vertex node numbers of the triangles can be used for several different geometries. These node numbers and geometries are combined in "X-Y ord" prior to the graphical plot in order to save storage space. The philosophy behind the first four types of programs: Triangle Tape; Node Tape (or its equivalent); X-Y ord; single-line graph will now be described.

Computer program "Triangle Tape"

It is desirable to reproduce graphically the network of triangles in order to reveal errors in specification. The nodes need to be plotted in such a way that one continuous line scribes all the sides of the triangles in sequence. It is sufficient to specify the triangles by their three vertex nodes. In order to produce positive integrals it is necessary for the straight lines to proceed around the triangles in a clockwise direction. If the vertices of the triangles lie on a cartesian (or a notionally cartesian) network, then use can be made of this in forming a continuous line.

The node numbers are notionally generated so that they proceed consecutively across the thickness. They are numbered in this way to minimise the bandwidth (R M Blakemore 1968) and hence improve the condition of the assembled stiffness matrix (Birmingham Symposium 1967). The vertex node numbers are generated and stored in a two-dimensional array for the whole network, so that for any pair of co-ordinates of radial index and axial index,
the appropriate vertex node number may be retrieved. They are stored in that way to enable sets of three vertex node numbers to be generated by proceeding around the triangles in a clockwise direction taking alternately the lower then the upper triangles. As each triangle set of vertex nodes numbers are generated they are written on to tape.

This tape is eventually used as part of the input data needed to form the assembled stiffness matrix, but before that it is verified by being part of data used to produce graphical output.

**Computer program Node Tape**

In a cartesian network several nodes have a common radius and several nodes have a common axial length. There are a number of different common radii and these are read into an array in order of increasing magnitude. Similarly, there are a number of different common axial lengths and these are read into another array in order of increasing magnitude. The indices in these two arrays to which any particular common radius or common axial length are assigned correspond to the axial or radial indices respectively.

For every intersection of radial and axial indices the vertex node number is calculated. Each such vertex node number has a radius associated with it, equal to the common radius for that particular radial index and has an axial length associated with it, equal to the common axial length for that particular axial index. As each vertex node number is calculated it is written on to magnetic tape with its corresponding axial and radial co-ordinates.

This tape is eventually used as part of the input data needed to form the assembled stiffness matrix, but before that it is verified by being part of data used to produce graphical output.

**Computer programs "Node Tape Boundary Points" and "Polar Node Tape"**

Two other programs have been written, each of which can run independently in place of or in conjunction with the computer program "Node Tape";
they are "Node Tape Boundary Points" and "Polar Node Tape". "Node Tape Boundary Points" forms a network stretching between nominated points on an inside surface and corresponding points on an outside surface of an axisymmetric walled structure. "Polar Node Tape" is to be used across the wall thickness of a spherical portioned structure. In fact, the notional cartesian co-ordinate system may be used for any transformed cartesian system for instance it may be used for an elliptical-hyperbolic co-ordinate system. But the network is not restricted to orthogonal curvilinear co-ordinate systems, and can thus be used iteratively on distorted shapes.

**Computer program "X-Y ord"**

The two data magnetic tapes: are from the computer program "Triangle Tape" and the other from the program "Node Tape" (or its equivalent) need to be combined in order to obtain the X and Y co-ordinates required for plotting one continuous line graph. In order to save space of the fast store (of the computer), the reading in off these magnetic tapes is done in the following manner. The co-ordinates of the vertex nodes together with their node number, are read notionally proceeding along radial line by radial line until the whole axis is covered; that is the magnetic tape from the program "Node Tape" is completely read into store before continuing. Then for each triangle separately, the three vertex node numbers for that triangle are read (as integers) off the magnetic tape from the program "Triangle Tape", and their X and Y co-ordinates (representing inches) are obtained from the fast store. The triangle number, the three pairs of co-ordinates together with the axial index, are written on to the output magnetic tape from this program. (Then the next three vertex node numbers are read in and so on). Thus, in that way only three pairs of co-ordinates are in store at any one time.
Computer program "Single Line Graph"

This reads the X and Y co-ordinates (in inches), off the magnetic tape from the computer program "X-Y ord", in sets of the three pairs for the three vertex nodes taken in a clockwise order around each triangle. The order of the triangles taken are alternately the lower triangle and the upper triangle: this ensures a continuous line without lines crossing. These are then plotted as one continuous line keeping the same scale for the horizontal and vertical ordinates. There is one version of this program written for the Stromberg Carlson (Data Graphics) 4020 microfilm recorder at the Atlas Laboratory and another version for the Calcomp 663/770 digital drum plotter.
Programs to obtain displacements

When the graphical output from the computer program "Single-line graph" has been obtained and no errors found, the next sequence of four programs: Assembly Disc, Decompose, Lower-Upper Treatment, and Displacement Check may be run. Each of these has a main segment which is effectively a prelude preceding the pseudo main segment. In this prelude from the number of spaces across the thickness and the number of spaces along the meridian the dimensions of the arrays in the pseudo main segment and in other subroutines are determined. (Details of this prelude are given in the appendix).
The computer program "Assembly Disc"

The first computer program in the sequence to obtain the displace­ments is Assembly Disc. This program forms the stiffness matrix for the whole structure. The following notes relate why certain operations (such as special matrix procedure and partitioning of the assembled stiffness matrix) are needed. In an appendix the subroutines used in "Assembly Disc" are briefly described to outline what the program does.

From consideration of the problems which require solving it becomes apparent that the program "Assembly Disc" needed to be written in a particular way and have "tailor-made" features.

The total number of nodes and hence the number of nodal displacements for the structural assembly of finite elements simulating an axisymmetric body is large. The length of the stiffness matrix for the whole body is equal to the total number of nodal displacements; whereas its semi-bandwidth is proportional to the number of spaces across the wall thickness of the axisymmetric body. For example a body having a network with ten spaces across the wall thickness and fifty spaces along the meridian (neither of which is excessive) result in an assembly matrix for the whole body 4242 long, with a semi-bandwidth of 126. The size of this assembly matrix requires consideration be given to the method of treatment to avoid the accumulation of errors. Such a method requires extra space. Some large computers (such as Stretch) have a fast store which will hold 100,000 words; so that a matrix which contains about a million words, together with the extra space required for treatment can not be held in the fast store of current computers. It is evident that auxiliary store is required. Two kinds are available:

(1) the magnetic disc which has rapid access but is fixed to the computer and is limited in the volume of data it will hold, and

(2) the magnetic tape, which although it has slower access, has greater storage capacity and is not fixed to the machine (Golden 1965) In view of
this it was decided (in 1969) to assemble the stiffness matrix for the complete body on disc and to transfer this to tape once it is completely assembled.

To economize and make the most efficient use of space it was decided to limit the data in the fast store to that currently in use. Hence the data for one triangular element is read and everything relevant to that element is calculated and cleared before proceeding to read the data of the next triangular element.

Another problem was to decide what to retain and what to "dump" on to disc. Certain stiffness coefficients of adjacent triangular elements need to be added but stiffness coefficients to which no further additions are to be made would be idly occupying useful current storage space. The problem naturally resolved itself in the following manner. A row of triangular elements across the wall thickness occupying one space along a meridian, has three rows of nodes. The first row of nodes lies along a line (of triangle sides) passing across the wall thickness from inside to the outside. Thus, this row contains vertex and mid-side nodes. The path of the second row of nodes across the wall thickness cuts across the triangles and so this row contains mid-side nodes only. The third row of nodes is similar to the first, in that, it lies along a line (of triangle sides) passing across the wall thickness from inside to outside containing vertex and mid-side nodes. It follows that only the first and third rows have nodes which are common to two bands of triangular elements occupying adjacent meridional spaces. All the nodes are numbered from inside to outside. (Figure 5.3): first, second, third rows consecutively. As there are two displacement numbers for each node (also numbered consecutively) it can be seen that the first third of displacement numbers are common to the previous adjacent band of elements across the thickness (if any previous) and the last third is common to the following band of elements across the thickness.
The matrix relating forces to displacements has the same number of rows (representing forces) as the number of columns (representing displacements). Hence it is a square matrix. If the total width of columns and the total height of rows are each divided into three equal bands, then it is only the first and third of these bands that are concerned with the adjacent space along the meridian. So out of these nine squares of matrix elements, which are formed by dividing both the span of columns and the span of rows into three equal parts, only the first square of the first row of small squares has been concerned with the additions of stiffness coefficients from the previous meridional space (if any) and only the third square of the third row of small squares will be concerned with the addition of the next meridional space if any. Hence the partitioning of the stiffness matrix for the elements assembled for one space along the meridian naturally suggests itself from physical considerations. As mentioned in the notes on the program Assembly Disc, as soon as one of these large square stiffness matrices has been assembled for all the elements occupying one space along a meridian, then the first eight small square partitioned matrices are "dumped" on to disc, the ninth small square of the partitioned matrix is retained in order that the stiffness coefficients for the triangular elements for the next space along the meridian (if any) can be added to it.

The size of these large square stiffness matrices is such that its number of rows and its number of columns are each equal to the total number of nodal displacements of the triangular elements in a space along a meridian. Each large square matrix (has been partitioned into nine small squares) overlaps the preceding and the succeeding large square matrices by the first and ninth small square respectively. Such a chain of large square matrices overlapping each other by a ninth of their area may be visualized (Figure 5.9)
The complete chain is the assembly stiffness matrix of the whole body. Since the semi bandwidth of this chain is equal to the number of columns (or rows) in a large square matrix, it is equal to the number of nodal displacements of triangular elements in one space along a meridian.

Thus the general philosophy of the program "Assembly Disc" (and its influence on the other programs) was formed. Nevertheless, there are some features which are peculiar to Assembly Disc. In the present form of the program, the common material stiffness properties are read in once only before all the triangular elements are dealt with; by altering the position of this read statement: the material properties can be read in for bands of elements or changed for every triangular element if desired.

A matrix which is used in the procedure to find the stiffness matrix for the triangular element is a $12 \times 12$ matrix which relates the twelve general nodal displacements to the twelve displacement coefficients via products of powers of the co-ordinate dimensions of the six nodes. Because the displacement function is of the same form for the radial as for the axial displacements, that is, it is only the two sets of six displacement coefficients which differ, it follows that this $12 \times 12$ matrix can be conveniently split into two identical $6 \times 6$ sub-matrices on the leading diagonal and zeros elsewhere. In so doing the first six rows relate to radial displacements and the second six rows relate to axial displacements. Thus only a $6 \times 6$ matrix needs to be processed instead of a $12 \times 12$. The processing of a $6 \times 6$ matrix takes approximately one eighth of the time of the processing of a $12 \times 12$, requires only one quarter of the space, and will accumulate less errors during the processing than a $12 \times 12$ matrix. Zienkiewicz (1967) dealing with a 3 node triangular element does not need to invert a $6 \times 6$ matrix. It is usually unnecessary to find $A^{-1}$ explicitly since, for example $A^{-1}B$ is the same as to solve $X$ in $AX = B$. But in this instance the
inversion of the 6 x 6 matrix by itself is required for the multiplication chain of a series of matrices in the formulation of the stiffness matrix.

A usual way of inverting a matrix (NPL 1961) is by putting $AX = I$ and then replacing $A$ by $LU$ so that $A^{-1}$ becomes $U^{-1} L^{-1} I$, which has the advantage of forming $L$ and $U$, (which could be useful for solving simultaneous equations), but has the disadvantage of requiring $2 n^2$ storage space. As $L$ and $U$ are not required, and as storage space needs to be used sparingly it was decided to write a routine which would invert and yet only require $n^2$ storage space (in this instance 6 x 6). This routine was based on a 14 line algorithm given by I M Khabaza on 30 January 1964. After several attempts it was found that a restriction of a conditional statement was not only unnecessary but produced erroneous results. This algorithm was corrected by experimenting (with the help of G Mathews 1964 page 48 and McCormick 1964 page 77). Khabaza (November 1967) amended the algorithm, which was essentially the same as before but with the removal of a conditional statement.

The routine written with this amended algorithm operated successfully on a large number of different matrices, but eventually data occurred for a triangular element which caused the routine to produce overflow. Because most of the matrix elements of the 6 x 6 matrix are products of powers of $r_i$ and $z_i$ (the co-ordinates of the $i$th node), if either or both of $r_i$ or $z_i$ approach zero (as is the case if a triangular element has a node near the axis or near the $Z$ origin) then elements on the leading diagonal can become small. As the reduction by the simple algorithm only used elements on the leading diagonal, as these approached zero overflow resulted. In other words dividing by a sufficiently small number the quotient can become too large for the computer to hold in its word length. Also some of the 6 x 6 matrices even which did not contain zero were also badly conditioned, so it was decided to modify the routine to use partial pivoting. This involved searching down a column for the largest element, keeping a record of the row that contained that element, reducing the pivotal row by dividing by that element, and subtracting a multiple of the
pivotal row from all other rows. After all columns had been searched etc, the rows were rearranged. This final routine has been successful on many thousand triangular elements treated, some of which touch the axis.
The Computer Program "DECOMPOSITION"

The second computer program in the sequence to obtain the displacements is the program called "Decomposition". This program decomposes the assembly stiffness matrix for the complete body as a step towards solving the simultaneous equations to give displacements. The input to this program is the overall stiffness matrix which was written onto tape by the previous program. It is to this program that the restraints are applied. With the same input tape it may be run any number of times with different patterns of restraints. The following discusses why the procedures were carried out in particular ways. The process through the computer program in a sequence of sixteen subroutines to show what the program does is outlined in the Appendix.

The stiffness matrix of the whole body contains the coefficients of the unknown displacements, hence these coefficients together with the unknowns form the left hand side of the simultaneous equations; whereas the right hand side is the applied forces. The normal method of solving simultaneous equations is the elimination procedure. This method was regularised into an orderly pattern by Gauss in 1873. In this, by a process of multiplication and subtraction, on both sides of the equations: the first coefficient is eliminated from the second line onwards (to the nth line), the first and second coefficients are eliminated from the third line onwards, and so on until at the nth line all the coefficients except one have been eliminated. Thus in the last line there is only a coefficient times one unknown displacement equal to a known value of applied force: hence this unknown displacement can be determined. In the line before the last there is now one known displacement (multiplied by a coefficient) plus one unknown displacement (multiplied by a coefficient) equated to an applied force: so this unknown displacement can now be resolved. This procedure, called backward substitution, is repeated until all the unknown displacements are resolved. The disadvantage of this method stems from the multiplication and elimination stage, where both sides are dealt with simultaneously. This disadvantage is that it only allows a
solution for one right hand side, that is, only for one set of applied forces.

Various methods have been devised by different people since then in the nineteenth century whereby the multipliers are stored in the places where the coefficients have been eliminated. This allows the right hand side of the equations to be processed at a later time; consequently any number of right hand sides may be processed.

To be able to deal with more than one array of applied forces is the essence of the method used in this sequence of programs; hence it is patterned on this method of storing multipliers. In the current program “Decompose” the overall stiffness matrix is decomposed into a lower portion containing the multipliers and an upper portion containing the remaining coefficients modified after elimination.

Advice from Dr J H Wilkinson (Wednesday 14 May 1969) was that partial pivoting would improve results even for a positive definite matrix. Therefore a method was devised using partial pivoting for a banded matrix in the form of partition blocks, somewhat similar to the matrix element process used by R S Martin 1967 but modified based on advice from Dr J H Wilkinson.

18th March 1970.

As mentioned previously, the input to the computer program “Decompose” is the stiffness matrix assembled for the complete body and the output is written onto two magnetic tapes. On the first tape called LOWTAP is written the locations of the interchanges and the multipliers; and on the second tape called IUPTAP is written the modified coefficients of the upper portion. The reason for the two tapes will be given in the notes on the next computer program in the series, which is called “LOWER-UPPER-TREATMENT”.

The overall stiffness matrix as formed is singular. Physical considerations dictate that at least one axial displacement must be specified. In the program “Decompose”, one or more axial or radial displacements can notionally be made equal to zero. The artifice employed to achieve this
is to zero the relevant complete row (s), (say row Io) with the exception of
the pivot (ie matrix element A (Io, Io)) which is made equal to a large number.
In the Atlas version of the program the large number taken is $10^{20}$. This
was discussed with Dr J H Wilkinson 27 May 1970.
The Computer Program LOWER-UPPER TREATMENT

In the sequence of programs to obtain displacement, the third computer program is called "LOWER-UPPER TREATMENT". The previous program ("DECOMPOSITION") applied the restraints (representing how the body was supported or suspended, or its movement impeded in some way); whereas one of the inputs into this program consists of the forces applied to some of the nodes and the output gives the radial and axial displacements of every node of the network covering the body. These displacements are used in the following computer program to obtain strains and stresses. The sequence of ten of the subroutines used in LOWER-UPPER TREATMENT is given in the appendices.
Features of the Computer Program "LOWER-UPPER TREATMENT"

As mentioned in the notes on the computer program "DECOMPOSE" two magnetic tapes: LOWTAP and IUPTAP, were used for output and these were the two input tapes for "LOWER-UPPER TREATMENT". The reason for two tapes needs some explanation. There are two considerations.

1. The size of the matrices before and after partial decomposition.

There are at least two factors which limit the size of the assembled stiffness matrix and hence the size of the network which can be dealt with. One is the size of the fast store, which limits the semi-band width of the matrix and the other is the amount of data which can be stored on tape, which limits the total number of matrix elements in the banded matrix. In physical terms, as previously shown, the semi-band width is equal to the total number of nodal displacement in one space along a meridian, whereas the length of the matrix is equal to the number of all the nodal displacements of the network covering the complete body.

In a problem which was run the assembled stiffness matrix had a semi-band width of 126 and an overall length of 4242. This matrix was partitioned before being written onto magnetic tape in the form of small square matrices (42 x 42 matrix elements). As there were 50 spaces along a meridian, the number of these small squares written on tape for the assembly matrix was 8 x 50 + 1 ie 401.

Decomposing the stiffness matrix with partial pivoting requires a treatment matrix of 3 x 5 ie 15, of these small square matrices. For each space along a meridian, the treatment matrix is decomposed into:

(a) the lower part which has two columns of small squares:
   the first column three small squares high and the second column two small squares high (ie five small squares) and
(b) an upper part which has two rows of small squares:
   the first row five small squares wide and the second row four small squares wide (ie nine small squares).
It follows that, per span along a meridian, the total number of small squares after decomposition is $5 + 9$ written to tape is about twice as many small squares written to tape, as the 8 per span along the meridian, for the assembly stiffness matrix before decomposition. Therefore, if the assembly stiffness matrix nearly fills a tape, which would be a limit on the length of the network, then to store the small squares after decomposition would require two magnetic tapes.

2. All the re-arrangements (due to partial pivoting) and the multiplications and subtractions of the lower treatment must be completed before the upper treatment of backward substitution can begin.

For these two reasons, it was decided to have one tape for the lower portion and another tape for the upper portion.

Another feature of this program (LOWER-UPPER TREATMENT) is that it economises in space by using the same space for two things at different times. This was based on the fact that once a displacement (of a particular index) was found (by backward substitution) the force (of the same index) was no longer required, hence the space occupied by the force could be overwritten by the displacement. As the space occupied by the array of forces is several thousand long, and the space occupied by the array of displacements is of equal length, this is a worthwhile saving of fast store space.
THE COMPUTER PROGRAM "DISPLACEMENT CHECK"

In any series of arduous calculations it is possible that errors can occur. The purpose of the computer program "Displacement Check" is to enable the analyst to ascertain that the displacements when considered with the stiffness are consistent with the applied loads.

Basically most calculations consist of three parts viz:

1. the data on which the calculation is to be performed;
2. the algorithm which gives the method of calculation;
3. the output results.

Any one or more of these three parts can produce errors.

The input data may be incorrect; so that this should be verified. The input data can be punched onto cards incorrectly. The data can be read into the computer incorrectly.

Because an algorithm has given reliable results to many sets of data, this is no guarantee that it will work correctly on the next set of data. It may not work correctly because a contingency as regards the data has not been anticipated. It may not work because of a change in the computer compiler.

The results may have been obtained correctly within the machine but there may be trouble with the output. For example the results may have been outputted onto magnetic tape either incorrectly or subsequently became a travesty of the original. For example the record may have become ruined by other writes to tape.

It is therefore, a necessary check to pre-multiply the vector of displacements by the assembled stiffness matrix of the whole structure in order to give the vector of loads at the nodes. Except where loads have been applied and where restraint(s) have been enforced, all this vector should be zero.
At displacement indices where loads have been applied (in the computer program "LOWER-UPPER TREATMENT") the output from the computer program "DISPLACEMENT CHECK" should reasonably agree. To explain what is meant by "reasonably agree" consider the example of an actual problem run. In this problem there were 4242 displacement indices, but to these only 101 loads were applied. Ninety nine of these were 845,000 lb each and two were 422,500 lb each. The output from "DISPLACEMENT CHECK" gave these 101 applied loads, the load at single restraint and the remaining 4140 values were effectively zero. In actual fact the maximum deviation from zero was less than 0.1 lb. The applied loads were printed as output from this program (DISPLACEMENT CHECK) as 845000.0 and 422500.0, so the error must have been less than 0.05 lb.

An additional piece of information is given as output from this program is the load(s) at the restraint(s).

Thus this program shows that the displacements are consistent with the stiffness matrix of the structure and the applied loads. It would not reveal any error in the stiffness matrix of the structure. Neither would it indicate any errors in the applied load.

It would not indicate if the structure had been restrained incorrectly or insufficiently. These errors would all stem from the structural idealization of the model.

Even if the structural idealization has been carried out correctly, as mentioned earlier, DISPLACEMENT CHECK is only a necessary check: it is not a sufficient check. It is necessary because if the obtained displacements put into this program do not give the applied loads, then those displacements are not consistent with the stiffness and the loads applied. Alternatively, if the obtained displacements do give the applied loads, they may not be the displacements which are required. In other words, more than one set of displacements may satisfy the applied loads (RALSTON A, 1965). This is not
very likely but it has happened before, that more than one set of solutions satisfy the simultaneous equations within the precision with which the calculations were performed. (ASKENASI 1968) Other checks therefore need to be carried out in addition to "DISPLACEMENT CHECK".

Once displacements give the applied loads, then the next program in the sequence of programs (which is called "STRESS STRAIN") may be run.
The Computer Program Stress-Strain

The purpose of this computer program is to find the value of stresses at "principal nodes". A "principal node" is defined as a vertex node at which six triangular elements meet: hence the majority of vertex nodes are "principal nodes".

Literature search of Finite Element documentation was carried out prior to writing the program stress Strain in 1970, also subsequently (Fullard 1971). This revealed that in existing computer programs more than one set of stresses were calculated at a node. In fact one set of stresses were calculated for each element adjacent to a node. To obtain one set of stresses per node the element sets of stresses were averaged in an arbitrary manner by the people running those programs.

Before writing the present sequence of programs, the author considered displacement functions of existing elements. The displacement function may be visualised as a surface plotted over the area within the boundaries of an element. Visualised in this way, the displacement function of a three-noded triangular-element is a plane surface.

The displacement functions at the junction node of six three-node triangular-elements can be visualised as the intersection of six facets. As most of the strains are dependent on the slopes of the displacement surfaces with respect to distance, and there are six slopes (in either one of the two principal directions) at that point, then there are six values of radial strain, six values of axial strain, and six values of radial-axial shear strain at that point. Because hoop strain is radial displacement divided by radius and there is only one radial displacement at a node then there is only one value of hoop strain at a node. But as stresses are dependent on more than one strain, then there are six values each of hoop, radial, axial, and radial-axial shear stresses at that point.
As the intersection of non-continuous surfaces do not usually give uni-valued slopes, strains or stresses; the author reasoned that, a suitable continuous displacement surface fitted by the method of least squares to the displacements of elements adjacent to a node, would give the best uni-valued slopes, strains and stresses at that node. Suitable surfaces dependent on displacements at nineteen nodes could be expected to give more reliable results, than the plane displacement surfaces of a three-node triangular element which depend on only three displacements.

The question then arose:- which order of polynomial would give a suitable surface. A suitable surface would be one which is consistent with a strain distribution which is true. Photoelastic analysis (carried out by the author in 1962) into regions of stress concentrations indicated that fourth order and sometimes fifth order polynomials were consistent with strain distributions observed. (The method of determining the best order of fit is described in an appendix 6c).

Hoop strains are of the same order as radial displacements; and the other strains (being the differential of the displacements) are one order less than the order of the displacement. So the author decided to attempt fitting fifth order polynomials to the displacements of a number of six-node triangular elements. Because the fifth order polynomial has twenty-one coefficients (and there must be more values of the dependent variable than coefficients for a least-squares fit), eight triangular elements (as indicated) were used, providing twenty five nodal displacements. The attempt was made using double precision on the computer "Stretch" (equivalent to about 29 decimal figures.
and exponents ranging from $10^308$ to $10^4-308$). This was unsuccessful for a Cartesian network but successful for a network based on random numbers.

Dr J H Wilkinson has kindly given a proof (May 71, appendix 6d) which shows that a 5th order Cartesian network produces a singular matrix.

As the program was required to deal with a Cartesian network, a 4th order polynomial was used. So the expression used was

$$U = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 xy + a_6 y^2 + a_7 x^3 + a_8 x^2 y + a_9 xy^2 + a_{10} y^3 + a_{11} x^4 + a_{12} x^3 y + a_{13} x^2 y^2 + a_{14} xy^3 + a_{15} y^4$$

where $a_1$ to $a_{15}$ are the coefficients which are obtained from the least squares fit of $x$ and $y$ co-ordinates (representing $r$ and $z$) with the nineteen displacement values ($U$ or $W$) of the nineteen nodes within the domain of the principal node.

The nineteen nodes comprise the principal node itself and the eighteen nodes in the six triangles surrounding it. These eighteen nodes consist of:--

- the mid-side nodes on the six sides which radiate out from the principal node;
- the mid side node on the six sides (of the six triangles) which face the principal node; and
- the remaining six are the vertices of the hexagon (generally irregular) formed by the six triangles around the principal node.
In one of the stages in fitting a surface by the method of least squares a $19 \times 15$ matrix is formed and this matrix is called $C$. Each row of the matrix $C$ in turn represents one of the nineteen nodes, in that the $x$ and $y$ co-ordinates used in a particular row correspond to the co-ordinates of a particular node. The columns of the matrix correspond to the following fifteen terms:

$$1, x, y, x^2, xy, x^3, y^2, y^3, x^4, y^4, x^2 y, x y^2, x y^3, y^4$$

The equation

$$a = (C^T C)^{-1} C^T U$$

(The derivation is shown in an appendix) is used to determine the array of coefficients "$a$" for the array of displacements "$U$", where $C^T$ stands for the matrix $C$ transposed, $C^T C$ is the product of $C$ transposed and $C$ and $(C^T C)^{-1}$ is the inverse of the matrix $C^T C$.

However, to apply the formula $a = (C^T C)^{-1} C^T U$ to a computer, because of the limitation of wordsize, certain refinements were necessary. One of these was to change the origin of the co-ordinates of the nodes to the principal node. This was suggested by Mr T Fuller (on 27 November 1970) and the reason for it is explained in the following. As some of the columns of the matrix $C$ represent the fourth powers of ordinates, and as ordinate distances may be of say 20 inches, then in the fourth order columns of matrix $C$ they become $20^4$ i.e $16 \times 10^4$, whereas the first column of matrix elements are unity. Consequently the ratio of the matrix elements in $C^T C$ will be in the order of $2.56 \times 10^{10}$ to 1 for a dimension of 20 inches, so that making the principal node the origin avoids this large ratio. Another refinement, was to reduce the $x$ dimensions from the principal node by the greatest $x$ distance of any node and similarly reducing the $y$ dimensions from the principal node by the greatest $y$ distance of any node within the domain of the six triangular elements adjacent to the principal node. In this manner the matrix becomes independent of the dimensions; but what is

53
more important is that by reducing the ratio of the largest to the smallest matrix element, improves the condition of the matrix $C^T C$ (Wilkinson 1969) and so allows it to be inverted without overflow.

The need to invert the $15 \times 15$ matrix $C^T C$ accurately lead to the development of a special partial pivoting subroutine.

The computer program stress-strain has been successfully used on analyses involving thousands of six-node triangular-elements of various shape, size and orientation. (In an appendix the sequence of thirty of the subroutines used in the program stress strain are outlined to show what the computer does).
Choice of test structure

A simple analytical solution exists for the problem of an elastic thick walled circular cylinder subjected to uniform internal pressure. The test structure, illustrated in figures 5.1 and 5.2, was chosen because the analytical and numerical solutions can be compared within any chosen accuracy.

This simple analytical solution yields values of hoop and radial stress in the form:

\[ \sigma_{00} = \frac{pr_1^2}{r_2 - r_1} \left\{ \frac{r_2^2}{r_1^2} + 1 \right\} \]

and

\[ \sigma_{rr} = \frac{-pr_1^2}{r_2 - r_1} \left\{ \frac{r_2^2}{r_1^2} - 1 \right\} \]

with all other stress components being zero, where:

- \( \sigma_{00} \) = hoop stress,
- \( \sigma_{rr} \) = radial stress,
- \( p \) = internal pressure,
- \( r \) = the radial distance from the cylinder axis to any point within the material,
- \( r_1 \) = the internal bore radius,
- \( r_2 \) = the outside radius.

Stresses in an ideal cylindrical tube can be evaluated to more significant figures than can be obtained from experimental observations.

The solution to the problem first appeared in a paper by Lame and Clapeyron in 1833 (Timoshenko 1941). The formulae for stresses arise from the solution to a differential equation which was based on the following postulates:

(i) all stress and displacements are independent of \( \theta \) (axial symmetry);
(ii) all stresses are independent of \( z \), thus \( u \) is independent of \( z \);

(iii) considering plane stress, then \( \sigma_{zz} = 0 \).

It follows that \( u = u(r) \) only,

\[
\begin{align*}
v &= 0, \\
\text{and } w &= w(z).
\end{align*}
\]

By virtue of the equation of equilibrium in the radial direction, and using the strain displacement equations, the radial displacement is expressed in the form of a differential equation

\[
\frac{d^2 u}{dr^2} + \frac{1}{r} \frac{du}{dr} - \frac{u}{r^2} = 0
\]

which has a general solution

\[
u = C_1 r + \frac{C_2}{r}
\]

A detailed derivation of this solution and the resulting stress components is given in Appendix.

Thus, the problem chosen to test the finite element sequence of programs only involved simple calculations and the solution lends itself to a straight line representation which facilitates appraising with simple statistical measures.

Geometry of structure

The bore and outside diameters were based on the pressure vessel shown in Figure 1 and were rounded to 20 inches and 30 inches respectively to give convenient numbers to work with (ie O/D to bore ratio = 1.5). The length was rationalised to 25 inches (the length was thus five times the wall thickness).

Network of triangles

Figure 5.2 illustrates the network which was used. The cylinder was divided into fifty equal \( \frac{1}{2} \) inch sections along the length and ten varying width sections across the radial thickness. Thus there were fifty times ten,
that is five hundred rectangles. Each of these rectangles was divided into two triangles making in all a thousand triangles shown in Figure 5.2. The ten sections across the radial thickness were graduated in width so as to have approximately equal reductions in hoop stress across each rectangle from the bore to the outside diameter. The radial ordinates of the nodes were calculated in the following manner.

**Calculation of radii for equal increments of hoop stress**

From Lame formula

\[
\sigma_{\theta\theta} = \text{constant} \times \left( \frac{r_2}{r_1} \right)^2 + 1
\]

where \( \frac{r_2}{r_1} \) varies from \( \frac{r_2}{r_1} \) to \( \frac{r_2}{r_2} \)

(i.e. in this case from 2.25 to 1).

If \( \sigma_{\theta\theta} \) is to vary in equal increments, \( \left( \frac{r_2}{r} \right)^2 \) must vary in equal increments.

For \( \left( \frac{r_2}{r} \right)^2 \) to go down in ten equal increments from 2.25 to 1, then the increments must equal \( \frac{2.25 - 1.0}{10} = 0.125 \), and this increment is used in the following table to determine values of radii.
Table giving radii for constant decrement of Hoop Stress

<table>
<thead>
<tr>
<th>Space</th>
<th>(\frac{r_2^2}{r})</th>
<th>(\frac{r_2}{r})</th>
<th>(\frac{r}{r_2})</th>
<th>(r) (Theoretical)</th>
<th>(r) (rounded to (\frac{1}{128}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.250</td>
<td>1.5000</td>
<td>0.6667</td>
<td>10.000</td>
<td>10.0</td>
</tr>
<tr>
<td>2</td>
<td>2.125</td>
<td>1.4577</td>
<td>0.6860</td>
<td>10.290</td>
<td>10.2890625</td>
</tr>
<tr>
<td>3</td>
<td>2.000</td>
<td>1.4142</td>
<td>0.7071</td>
<td>10.607</td>
<td>10.609375</td>
</tr>
<tr>
<td>4</td>
<td>1.875</td>
<td>1.3693</td>
<td>0.7303</td>
<td>10.955</td>
<td>10.953125</td>
</tr>
<tr>
<td>5</td>
<td>1.750</td>
<td>1.3229</td>
<td>0.7560</td>
<td>10.340</td>
<td>11.34375</td>
</tr>
<tr>
<td>6</td>
<td>1.625</td>
<td>1.2747</td>
<td>0.7845</td>
<td>11.767</td>
<td>11.765625</td>
</tr>
<tr>
<td>7</td>
<td>1.500</td>
<td>1.2247</td>
<td>0.8165</td>
<td>12.248</td>
<td>12.25</td>
</tr>
<tr>
<td>8</td>
<td>1.375</td>
<td>1.1727</td>
<td>0.8527</td>
<td>12.791</td>
<td>12.7890625</td>
</tr>
<tr>
<td>9</td>
<td>1.250</td>
<td>1.1180</td>
<td>0.8945</td>
<td>13.417</td>
<td>13.4140625</td>
</tr>
<tr>
<td>10</td>
<td>1.125</td>
<td>1.0606</td>
<td>0.9429</td>
<td>14.142</td>
<td>14.140625</td>
</tr>
<tr>
<td>11</td>
<td>1.000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>15.000</td>
<td>15.0</td>
</tr>
</tbody>
</table>

(Adjustment to binary fractions was suggested by Mr T Fuller, 18 February 1972)
Process through the sequence of computer programs

From these eleven constant radial coordinates (shown in column 6 of the table) and the fifty-one axial coordinates (0.0 to 25.0 inches in increments of 0.5 inches), the computer program NODE TAPE calculated the 561 (i.e. 51 x 11) vertex node-numbers and their coordinates, which it then wrote onto another magnetic tape. Whilst the computer program TRIANGLE TAPE wrote onto tape the i, j and k indices of the 1000 triangles. The data from these two magnetic tapes was used by the program X-Y ORD to specify the vertex coordinates taken in clockwise order around each of the 1000 triangles which was written onto a third magnetic tape. The computer program SINGLE LINE GRAPH used this third magnetic tape to produce on the Stromberg Carlson 4020 graphical plotter a diagram as shown in Figure 5.2. Thus at this stage the geometry of the network was verified.

After the network has been verified, the magnetic tapes from the computer programs NODE TAPE and TRIANGLE TAPE were used as input to the computer program ASSEMBLY DISC which formed the stiffness matrix for the whole structure. This matrix had a semi-bandwidth of 126 and a length of 4242. (The number 4242 is the total number of nodal displacements of this complete structure.) This matrix was partitioned into 50 (which is the number of half inch spaces along the cylinder's 25 inch length) multiplied by 8 plus 1 = 401 sub-matrices. The sub-matrices contained 42 x 42 matrix elements. (That is a total of 707364 matrix elements.) Each of these sub-matrices were written onto magnetic tape together with row and column indices so as to indicate which partition of the complete structure stiffness matrix they were representing.

At this stage any system of restraints in an axial or radial direction could have been applied to any, one or more, of the 2121 nodes. A single node was restrained in an axial direction; this is the minimum that may be applied. The restraint node was at the bore at the datum end.
of the cylinder. This data was supplied on punched cards to the computer program DECOMPONE before it decomposed the stiffness matrix into lower and upper triangular matrices which were written as output to two separate magnetic tapes.

**Determination of loads at nodes**

Using the relationship between hoop stress and internal pressure, the internal pressure was determined so that the hoop stress should not exceed the yield stress; and thence the loads to be applied radially at the internal (bore) nodes were determined to represent this internal pressure. Let:

\[ \sigma_{\theta \theta} = \text{hoop stress} = \text{yield stress of } 140 \times 10^3 \text{ lb/sq in (En25), (0.47\% strain)} \]

\[ p = \text{internal pressure (lb/sq in)}, \]

\[ r_1 = \text{internal radius} = 10 \text{ inches}, \]

\[ r_2 = \text{external radius} = 15 \text{ inches}, \]

then

\[ \frac{\sigma_{\theta \theta}}{p} = \frac{r_2^2 + r_1^2}{r_2^2 - r_1^2} = \frac{15^2 + 10^2}{15^2 - 10^2} = \frac{25^2 + 9^2}{25^2 - 9^2} = \frac{9 + 4}{9 - 4} = \frac{13}{5} = 2.6 \]

if

\[ \sigma_{\theta \theta} = 140 \times 10^3 \text{ lb/sq in}, p = \frac{\sigma_{\theta \theta}}{2.6} = \frac{140 \times 10^3}{2.6} = 53.8 \times 10^3 \text{ lb/sq in} \]

The applied uniform internal pressure was represented by equal radial loads at all the internal (equi-spaced) bore nodes except the two end nodes, which had half the load. As there was 0.25 in. spacing between nodes, which is equivalent to 0.125 in. on either side of nodes, which are at 10 in. internal radius; therefore the pressure area for one typical node = \( 2\pi \times 10 \text{ in.} \times 0.25 \text{ in.} = 5\pi = 15.7 \text{ sq in.} \)

load on a typical node = \( 53.8 \times 10^3 \text{ lb/sq in.} \times 15.7 \text{ sq in.} = 845 \times 10^3 \text{ lb.} \) Then load at end node (half typical load) = \( 422.5 \times 10^3 \text{ lb.} \)

These loads formed the input data for the computer program LOWER-UPPER TREATMENT. There were 99 nodes with the radial load of 845000.0 lb. and the two end nodes with a radial load of 422500.0 lb. The two directions at
the nodes are designated by displacement indices. The even displacement indices indicate the axial direction, and the odd indices the radial direction. The bore displacement indices for this problem were 1, 43, 85 etc. The output for this program was displacements which were written onto magnetic tape.

The magnetic tape from the program LOWER-UPPER TREATMENT provided part of the input data to the program DISPLACEMENT CHECK. These displacements were pre-multiplied by the stiffness matrix which was read off the magnetic tape from ASSEMBLY DISC. The product gave zero load at all the nodes except it gave the radial loads at each of the 101 bore nodes. These were 422500 lb. at each end node and 845000 lb. at the other 99 bore nodes. All the loads at the nodes (zero and otherwise) were within ± 0.1 lb. This was a necessary but not sufficient check.

The results from the check being satisfactory, the displacements were read off the magnetic tape from LOWER-UPPER TREATMENT into the program STRESS-STRAIN. The principal nodes were all the vertex nodes except those at the bore and the 0/D, and those at either end. There were thus (11 - 2) x (51 - 2) = 441 principal nodes. The line-printer output from the program STRESS-STRAIN gave the hoop, radial, axial, and radial-axial shear strains and stresses.

Agreement between Finite Element and Lamé Predictions

There is very good agreement between the Finite Element results and the Lamé theoretical values. This is shown in Figure 10.1 where the values of hoop and radial stress from Finite Element are plotted as points against $1/r^2$ and $-1/r^2$ respectively and the Lamé values as a continuous straight line. It can be seen from this figure that of the points representing the values of hoop and radial stress at the nine radii, only the values of hoop and radial stress near the bore (ie at 10.289 radius) lie off the straight line. Thus the greatest error in the Finite Elements prediction occurs at the principal nodes adjacent to the bore where it overestimates the hoop stress by less
DEVIATION OF FINITE ELEMENT RESULTS AGAINST LAMÉ STRAIGHT LINE

\[ \Theta \text{ INDICATES FINITE ELEMENT RESULTS} \\
\text{--- INDICATES THEORETICAL LAMÉ RESULTS} \]
FIGURE 10.2. DIFFERENCES (IN PARTS PER MILLION) BETWEEN FINITE ELEMENT AND LAMÉ VALUES OF HOOP STRESS
Figure 10.3: Differences (parts per million) between F.E. & Lamé

KEY RADIUS
- O - 14.1406
- □ - 13.4141
- △ - 12.7891
- ● - 12.25
- ▽ - 11.7656
- X - 11.3438

DISTANCE ALONG AXIS
FIGURE 10.5

DIFFERENCES (PER CENT) BETWEEN F.E. & LAME

10.289  10.609  RADIUS  10.953  11  11.344
than 2%. The error in the radial stress (the lesser stress) was 9% at this position. When a straight line was fitted to the remaining points by the method of least squares it was found that it had a product moment correlation coefficient of 0.999996 and a standard deviation of the residuals of less than 210 lb/sq in, whilst the maximum hoop stress was 140,000 lb/sq in.

There were 49 principal nodes adjacent to the bore (at 10.289 inch radius) out of a total of 441 principal nodes. Figure 10.2 shows a contour plot of the extremely small errors in hoop stress outside the area of the nodes adjacent to the bore. These errors are so small over most of the area it is not convenient to plot them as percentages, they are thus plotted as parts per million. Figure 10.2 was constructed from Figure 10.3 which shows curves (one for each radius) of errors in hoop stress against length along the axis. It can be seen from Figure 10.3 that these errors peak at 2" from each end but these peaks do not exceed 300 parts per million. Superimposed on this general U shape curve is a fine ripple which increases from zero at the datum (or origin) end to +10 parts per million at the end farthest from the datum end which is treated later in the numerical analysis.

Assessment of Results

From Figures 10.7 and 10.8 where stresses and strains are plotted along the axis, it can be seen that they do not vary noticeably along the length of the cylinder. This is consistent with Lamé. There are two causes of the slight discrepancies between the Finite Element results and the Lamé values. They are:

(i) the discrete Finite Element does not truly represent the continuum;

(ii) the accumulation of round off error in the computer. This can occur in the integration procedures or in the matrix treatment (Grose 1971).

The largest discrepancy of 2%, which occurs in the vicinity of the applied load is an example of the first cause of discrepancy. Because the
FIGURE 10.6
### Table 10.7

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>RADIUS (INCHES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>⌀</td>
<td>10.2891</td>
</tr>
<tr>
<td>ø</td>
<td>10.6094</td>
</tr>
<tr>
<td>△</td>
<td>10.9531</td>
</tr>
<tr>
<td>+</td>
<td>11.3438</td>
</tr>
<tr>
<td>×</td>
<td>11.7856</td>
</tr>
<tr>
<td>φ</td>
<td>12.2500</td>
</tr>
<tr>
<td>×</td>
<td>12.7891</td>
</tr>
<tr>
<td>N</td>
<td>13.4141</td>
</tr>
<tr>
<td></td>
<td>14.1406</td>
</tr>
</tbody>
</table>

**Figure 10.7**

Distance along axis in inches

Stress in lbf/sq in $\times 10^3$

Axial & Shear

Radial
FIGURE 10.8

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>RADIUS (INCHES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>○</td>
<td>10.2891</td>
</tr>
<tr>
<td>□</td>
<td>10.6034</td>
</tr>
<tr>
<td>△</td>
<td>10.9531</td>
</tr>
<tr>
<td>+</td>
<td>11.3439</td>
</tr>
<tr>
<td>×</td>
<td>11.7655</td>
</tr>
<tr>
<td>○</td>
<td>12.2500</td>
</tr>
<tr>
<td>△</td>
<td>12.7891</td>
</tr>
<tr>
<td>×</td>
<td>13.4141</td>
</tr>
<tr>
<td>□</td>
<td>14.1406</td>
</tr>
</tbody>
</table>

DISTANCE ALONG AXIS IN INCHES

SHEAR

AXIAL

RADIAL
Finite Element was modelling the uniformly distributed internal pressure, equal loads were applied to the equally spaced internal nodes. But radial spring stiffness at the midside nodes is greater than that of the vertex nodes consequently the radial displacement of the vertex nodes is much greater than that of the mid-side nodes: producing a saw tooth effect on the radial displacement (see Figure 10.6). Thus the Finite Elements inner boundary's radial displacement is not uniform as would be a continuum's boundary's radial displacement under uniform pressure.

The next largest discrepancy of 0.03% or 300 parts per million of the peaks at the ends (see Figure 10.3), although small is also probably due to discretisation.

The superimposed fine ripple which increases steadily from zero at the origin end to +10 parts per million at the locations which are treated later in the numerical analysis, is probably an example of the second cause of discrepancy: "accumulation of round off error". The stiffness matrix had a semi-bandwidth of 126 and a length of 4242; round-off error accumulation in a matrix of that size (bearing in mind that the number of operations is proportional to the cube of its size) would not be unexpected. These would be dependent on the condition of the matrix, which in turn is influenced by the graduation of the size of the Finite Elements. The smallness of this ripple may be attributed to grading the elements to have equal increments of hoop stress across them (see page 59) and on the matrix procedures adopted.
A six node element was used in preference to a three node element because the associated displacement function more adequately describes the deformation behaviour of the elastic continuum which is to be represented by the finite element array. The complete quadratic used in the present work is able to cope with more possibilities than a truncated quadratic. Doherty compares two elements, known as Q4CST and QM5 which are used in different editions of Sandia Corporation's SAAS finite element code. Q4CST has a linear displacement function, whereas QM5 has a curtailed quadratic displacement function. The QM5 results are virtually identical to the theoretical results for an infinitely long cylindrical tube subjected to edge moments at one end (Timoshenko S and Woinowsky-Krieger 1959, pages 466-473). Moreover the maximum radial displacement of QM5 (which compares very favourably with the theoretical results) is 2½ times that of Q4CST. This and other similar evidence suggests that solutions involving elements with linear displacement functions have serious shortcomings which render them unsuitable for this type of problem.

Formulation of the stiffness equations used in this finite element program depends on conservative energy considerations and is based on an assumed displacement field. The displacement functions themselves ensure that the compatibility conditions are satisfied within an element. In addition, an appendix 8 illustrates that continuity of displacements and hence compatibility between domains of elements is satisfied to a high degree by applying the method of least squares to the displacements of the nodes of six elements adjacent to a principal node. The product of stiffness coefficients and nodal
displacements equalling the loads applied to nodes ensures that overall equilibrium is satisfied. For the stresses to satisfy equilibrium exactly there should be a minimum of potential energy. Since the displacement field is approximate, the potential energy in the idealised structure is greater than the true minimum, and hence an upper bound to the potential energy of the real structure, and so the values of stresses are generally greater than the true values. As the elements become smaller, the approximation to the true displacement field becomes closer and hence the values of stresses tend towards the true values.

In forming the fifteen integrals required to determine the stiffnesses algebraic equations were used in preference to numerical integration to achieve the maximum accuracy and thus avoid round-off error (Crose 1971). Care was taken in the formulation of these equations to allow for the possibility of one or more vertices to lie on the axis of symmetry and to allow for two vertices to have the same radius. Zienkiewicz (1967) has not made adequate provision for two vertices to have the same radius in his algebraic integration.

ON THE PHILOSOPHY OF COMPUTER PROGRAMS

Large recent multipurpose programs such as NASTRAN (MacNeal 1970) and BRISTOL (Atkin 1972), although more versatile than the present sequence of programs, can only deal with problems as a whole. BRISTOL contains triangular prism elements, which have mid-side nodes as well as vertex nodes. BRISTOL has the disadvantage that it requires a number of these elements (Brebbia Jan 1971, page 2) to form the ring of an axisymmetric model.

Such programs as BRISTOL and NASTRAN are very costly to run; incur waits of about a week; are limited to one aspect of inquiry; and require phantom data if not enough realistic information is available at the initial stage.
The present sequence of separate programs enables the effect of varying any of the physical parameters of geometry, materials, supports and restraint, and loading to be studied without incurring high costs or long waiting time. This is achieved by running each of the sequence of programs separately. (Although they can be run concurrently). The first two programs in the sequence are concerned with the geometry and automatically generate a network to cover a section through the structure. In one of these programs, the node numbers which denote the three vertices of each triangular element are listed in an ordered sequence which is independent of the gradation of element size and boundary profile of the structure. This has advantages in permitting changes to be made in the shape or scale of the structure without modification to subsequent data or analysis procedures in the programs following. Hence geometric non-linearity can be processed in an iterative manner. In the companion program, concerned with network generation, the co-ordinates of each vertex node are evaluated on the basis of profile co-ordinates and specified numbers of internal intervals. The next program in the sequence was written to enable the output from the two network generating programs to be verified visually before the data is used as input to the large main programs which perform numerous calculations. The first of these large main programs assembles the overall stiffness matrix and therefore requires values of the elastic material properties. The network being generated independently of material specification enables the effect of different material properties such as variation of poisson's ratio, on the behaviour of the structure to be examined at this stage. The overall stiffness matrix is far too large to be held
in the computer's fast store which is limited to 100,000 words. Hence an auxiliary store was required. The $2$ million matrix elements required for one test problem almost filled one magnetic tape.

On decomposition into lower and upper matrices performed by the next program in the series, the space requirements increase to such an extent that two magnetic tapes were needed: one each for the lower and upper matrices. The increase in space required was partly the result of processing by partial pivoting which, although theoretically not essential for a positive definite symmetric matrix, is used to obtain a greater degree of accuracy on digital computers with their inherent round-off error (J. H. Wilkinson, May 1969). The overall stiffness matrix is singular and before decomposition requires at least one restraint to make the matrix non-singular. Thus the restraints and supports are not required in the sequence until the following program; the same structure may be investigated for various types of support and/or restraint. The restraints which hold the structure in space must be fixed before the loads are imposed. The next program in the sequence requires data on loads. The loads form the right hand side of the simultaneous equations, and this right side is operated on first by the multipliers from the lower magnetic tape, and then by the modified coefficients on the upper magnetic tape. This process changes the data stored on the right hand vector sequentially from loads to displacements, thus saving storage space. By processing the left hand side of the simultaneous equations first and separately from the right hand, the left hand side has to be processed only once for one system of restraints for any number of right hand sides or systems of loads. Thus any number of systems of loadings can be investigated for each system of supports or restraints; the outputs from this program are values of
displacement. Distributed loading such as uniform external pressure is simulated by a series of equivalent point loads at nodes on the pressurised boundary. The solution of the pressurised cylinder problem considered here indicates that the effects of point loads are distinguishable from the effects of the ideal pressure loading to a depth of three elements. This result suggests that the element array should be graded in relation to the geometric detail of the boundary so as to provide at least three rows of fine elements.

The next program makes a necessary check which is to pre-multiply the displacements by the stiffness matrix giving forces which should have the same values as the applied loads.

The last program in the sequence evaluates the strains and stresses from displacements. Programs by other authors obtain stresses on the basis of some arbitrary means, whereas in this program the stresses were obtained by fitting a fourth order surface to the displacements by the method of least squares and then deriving strains from these surfaces and hence stresses. Previously only plane surfaces have been fitted (Robert D Cook 1968). This method is used to determine the values of stress and strain at the principal nodes which are located at the common vertex of six triangular elements. The stresses at the boundary, where the critical values often occur, may be obtained by extrapolating a plot of the principal node values. In regions of high strain gradient, it is apparent that the accuracy of boundary values predicted by extrapolation will be increased as the pitch of the element array is diminished.

Thus to sum up:-

Separating the computation into a sequence of discrete programs has advantages over running a single large program:-
(a) the requirements for computer time and space are reduced thus minimising the cost and restrictions imposed by normal operating procedures. For example typical waiting time involved in running a large program such as Bristol which is typically one week can be reduced to less than a day;
(b) decisions on restraints and external loading may be deferred until the relevant values are required;
(c) modification to any program of the sequence may be carried out without having a feed back effect on any of the other programs;
(d) parametric studies of the effect of varying: geometry, material, support and restraints, and loading were facilitated; and
(e) enables geometric non-linearity to be treated by iteration.

ON THE APPLICATION OF THE PROGRAM TO AN IDEAL PROBLEM

The thick cylinder under internal pressure loading was chosen as the test structure because an analytical solution (Lamé solution) was available which could be compared with any chosen degree of accuracy to finite element results.

A network was selected to give a reasonable number of intervals (10) along the stress gradient in the radial direction and along the length (50), these being practical numbers which would dispel the impression that the accumulated round-off error would invalidate the results if a large number of elements was used. This impression is created by some of the larger comprehensive programs such as NASTRAN
which limits the number of nodes to several hundred grid points. In this instance there were 2121 nodes and the resulting accumulation of round-off error only amounted to 0.001%. There was very good agreement between finite element results and Lame, the general error (except where the loads were applied) being about 0.03%, so that the accumulated round-off error was insignificant. The smallness of these errors can be attributed in part to the gradation of elements across the stress gradation and the procedures adopted (ie partial pivoting and algebraic integration).
CONCLUSIONS

It has been demonstrated that this finite element sequence of programs

(a) is easy to use;
(b) is readily applicable to any large computer;
(c) produces extremely accurate results in regions of high stress-gradient, and
(d) facilitates parametric analysis.

It is easy to use because it is written from the point of view of the person undertaking the stress analysis. In fact, the division into the separate programs is based on the logical steps used in photoelastic investigation.

The sequence of programs has already been applied to more than one computer, and the ease of application to other machines can be attributed to the fact that the programs were written in simple basic language rather than standard computer language.

The results evaluated in the high stress gradient field of the test problem were generally accurate to within 0.03%. Three main features of the programs were responsible for this high degree of accuracy. These are:

(a) algebraic integration rather than numerical integration was used in forming the stiffness matrix,
(b) a direct method was used in the solution of the linear equations rather than an iterative method, and partial pivoting was used in decomposition and inversion of matrices; and
(c) strains and stresses were obtained from fourth order displacement surfaces fitted by the method of least squares to principal node displacements.

The programs were divided into a sequence so as to facilitate a study of the effect of independent variations in each of the physical
parameters such as geometry, materials, supports and restraints, and loading. This sequential procedure helps to minimise computer cost and time.

The sequence of programs presented here is simple to use; full details concerning the operating procedure are included in the text. But because the results are so dependent on the details of the structural idealisation, the full benefit of the programs can only be gained by those who have a competent knowledge of structural behaviour.
REFERENCES

Ashkenazi, V. 1968 September 20. "Matrix Methods in Civil Engineering Networks". Seminar held in University of Nottingham. More than one set of solutions satisfy the simultaneous equations with precision to which the calculations were performed.

Atkins, W.S. 1972. ASAS stress analysis system (BRISTOL).

Bartlett, Joan. 1962. (Mrs Knock) Program "Polyfit", by Method of Least Squares fits polynomials up to 10th order.


Crose, James G and Jones, Robert M. 1971 June 22. SAAS III Finite element stress analysis of axisymmetric and plane solids with different orthotropic, temperature-dependent material properties in tension and compression. The Aerospace Corporation, San Bernardino, California. AD 729188.


In the third equation on page 180 which reads $\frac{\partial e_{zz}}{\partial z}$ should read $\frac{\partial e_{zz}}{\partial r}$

Fullard, K. 1971 May 26. Applications at CEGB. One day seminar at Birniehill Institute NEL. More than one set of stresses were calculated at a node.

Fuller, T. 1970 November 27. Private communication.

Fuller, T. 1972 February 18. Private communication.


APPENDIX 1a  Theory (Section 1)

The following algebraic treatment, is to obtain the stiffness matrix of a six node axisymmetric triangular element. The integrals are evaluated in the next part. The eight steps in obtaining the stiffness matrix are as follows:

1. Figure 2 is a diagram of a triangular section through a typical axisymmetric annular element indicating the notation of the vertex nodes, showing the radial and axial direction; and the radial and axial displacements.

2. Radial and Axial Displacement function for the domain of the triangle.

3. Boundary conditions specified by displacements at nodes.


5. The three direct strains and the one shear strain in terms of the coefficients of the displacement function, and general ordinates.

6. The four stresses as a product of elastic constants and the four strain values.

7. The four stresses in terms of nodal point displacements.

8. Work done by principle of virtual displacements. That is, the work done by displacement of nodal point forces equated to the strain energy of the product of strain and stress over the volume of the triangular annulus. The first multiplication of both sides of this equation by the inverse of the displacement vector give the force vector on the left hand side and its equivalent on the right hand side. The second (post) multiplication of both sides of this equation by the inverse of the displacement vector, gives the matrix of stiffnesses on the left hand side and its equivalent on the right.
Nomenclature

\[ r = \text{radial ordinate} \]
\[ z = \text{axial ordinate} \]
\[ \delta = \text{displacement} \]
\[ u = \text{radial displacement} \]
\[ w = \text{axial displacement} \]
\[ A_n = \text{Displacement function coefficient, where suffix } n \text{ is from 1 to 12} \]
\[ \{ \} = \text{Column Vector} \]
\[ [ ] = \text{Row Vector} \]
\[ [ ] = \text{Matrix} \]
\[ \{^u\} = \text{Column Vector of displacements, radial at top six, axial at bottom six} \]
\[ \{^w\} = \text{Column Vector of displacement function coefficients} \]
\[ \{^+\} = \text{Column Vector of nodal displacements} \]
\[ \{^u\}_i,j,k = \text{in radial direction} \]
\[ \{^w\}_i,j,k = \text{in axial direction} \]
\[ i,j,k = \text{vertex nodes} \]
\[ l,m,n = \text{mid-side nodes} \]
\[ [^A] = \text{matrix formed from ordinates of nodes, such that if it premultiplies the displacement coefficient vector, the product is the nodal displacement vector} \]
\[ [^A] = \text{matrix formed from ordinates of any points within the triangle, such that if it premultiplies the displacement coefficient vector, the product is the vector of displacements of those points} \]
\[ e_{\theta\theta} = \text{strain in the hoop direction} \]
\[ e_{rr} = \text{strain in the radial direction} \]
\[ e_{zz} = \text{strain in the axial direction} \]
\[ e_{rz} = \text{shear strain in the radial axial plane} \]
\{e\} = the column vector of the four strains

\[eA\] = matrix formed from the ordinates of any points within the triangle, such that if it premultiplies the displacement coefficient vector, the product is the column vector of the four strains

\[e\delta\] = matrix which when it premultiplies the nodal displacement vector produces the column vector of the four strains

D = general determinant

\(\sigma_{\theta \theta}\) = hoop stress

\(\sigma_{rr}\) = radial stress

\(\sigma_{zz}\) = axial stress

\(\sigma_{rz}\) = radial axial shear stress

v = Poisson's Ratio

E = Young's Modulus of Elasticity

\(D\sigma_{\theta \theta}\) = Determinant of hoop stresses, that is \(\frac{D\sigma_{\theta \theta}}{D} = \sigma_{\theta \theta}\)

\{\sigma\} = column vector of the four stresses \(\sigma_{\theta \theta}, \sigma_{rr}, \sigma_{zz}\) and \(\sigma_{rz}\)

\[\sigma e\] = matrix formed from the elastic constants \(v, E\) such that

on premultiplying the strain vector produces the stress vector

\[\sigma \delta\] = matrix which when it premultiplies the nodal displacement vector, it produces the vector of the four stresses

W = the work done

U = the strain energy

q = the nodal point forces

\{q\} = column vector of nodal point forces

dv = elemental volume

\[\delta\] = row vector of nodal point displacements

\[k\] = matrix of stiffness coefficients
(2) Radial and Axial displacements in terms of a complete quadratic in \( r \) and \( z \) ordinates

\[
u = A_1 r^2 + A_2 r + A_3 + A_4 z^2 + A_5 z + A_6 rz\]

\[
w = + A_7 r^2 + A_8 r + A_9 + A_{10} z^2 + A_{11} z + A_{12} rz\]

which may be written in matrix form

\[
\{u\} = \begin{bmatrix}
A_1 \\
\vdots \\
A_{12}
\end{bmatrix}
\begin{bmatrix}
r^2 & r & 1 & z^2 & z & rz \\
r^2 & r & 1 & z^2 & z & rz
\end{bmatrix}
\]

or in symbolic form

\[
\{u\} = \begin{bmatrix} U A \end{bmatrix} \{A\}
\]

(3) Boundary Conditions

at 
\[
z = z_i \text{ and } r = r_i : u = \delta_\alpha \text{ and } w = \delta_\alpha+1
\]
\[
z = z_k \text{ and } r = r_k : u = \delta_{\alpha+2} \text{ and } w = \delta_{\alpha+3}
\]
\[
z = z_j \text{ and } r = r_j : u = \delta_{\alpha+4} \text{ and } w = \delta_{\alpha+5}
\]
\[
z = z_m \text{ and } r = r_m : u = \delta_{\alpha+6} \text{ and } w = \delta_{\alpha+7}
\]
\[
z = z_k \text{ and } r = r_k : u = \delta_{\alpha+8} \text{ and } w = \delta_{\alpha+9}
\]
\[
z = z_n \text{ and } r = r_n : u = \delta_{\alpha+10} \text{ and } w = \delta_{\alpha+11}
\]
Boundary Condition Equations

\[
\{u\} = \begin{bmatrix}
\delta_{a+1} \\
\delta_{a+3} \\
\delta_{a+5} \\
\delta_{a+7} \\
\delta_{a+9} \\
\delta_{a+11}
\end{bmatrix} = \begin{bmatrix}
r_i^2 & r_i & z_i^2 & z_i & r_i z_i \\
r_j^2 & r_j & z_j^2 & z_j & r_j z_j \\
r_m^2 & r_m & z_m^2 & z_m & r_m z_m \\
r_k^2 & r_k & z_k^2 & z_k & r_k z_k \\
r_n^2 & r_n & z_n^2 & z_n & r_n z_n
\end{bmatrix} \begin{bmatrix}
A_1 \\
A_2 \\
A_3 \\
A_4 \\
A_5 \\
A_6
\end{bmatrix}
\]

\[
\{W\} = \begin{bmatrix}
\delta_{a+1} \\
\delta_{a+3} \\
\delta_{a+5} \\
\delta_{a+7} \\
\delta_{a+9} \\
\delta_{a+11}
\end{bmatrix} = \begin{bmatrix}
r_i^2 & r_i & z_i^2 & z_i & r_i z_i \\
r_j^2 & r_j & z_j^2 & z_j & r_j z_j \\
r_m^2 & r_m & z_m^2 & z_m & r_m z_m \\
r_k^2 & r_k & z_k^2 & z_k & r_k z_k \\
r_n^2 & r_n & z_n^2 & z_n & r_n z_n
\end{bmatrix} \begin{bmatrix}
A_7 \\
A_8 \\
A_9 \\
A_{10} \\
A_{11} \\
A_{12}
\end{bmatrix}
\]
In symbolic form

\[
\{ \delta_u \} = \begin{bmatrix} \delta A \end{bmatrix} \{ A \} \quad 6 \times 6 \quad 1+6
\]

\[
\{ \delta_w \} = \begin{bmatrix} \delta A \end{bmatrix} \{ A \} \quad 6 \times 6 \quad 7+12
\]

combining

\[
\begin{bmatrix} \delta u \\ \delta w \end{bmatrix} = \begin{bmatrix} \delta A & 0 \\ 0 & \delta A \end{bmatrix} \begin{bmatrix} A \end{bmatrix}
\]

\[
\{ A \} = \left[ \delta A \right]^{-1} \{ \delta \}
\]

\[
\{ A \} = \begin{bmatrix} \delta A^{-1} \\ 0 \end{bmatrix} \begin{bmatrix} \delta u \\ \delta w \end{bmatrix} \quad \text{and} \quad \{ A \} = \left[ \delta A \right]^{-1} \{ \delta \}
\]

so that

\[
\{ u \} = [UA] \{ A \} = [UA] [\delta A]^{-1} \{ \delta \}
\]

\[
\uparrow \quad \text{displacements of any positions}
\]

\[
\uparrow \quad \text{node point displacements}
\]
Strain in terms of displacement

\( V_r \) = displacement in radial direction
\( V_\theta \) = displacement in theta direction
\( V_Z \) = displacement in axial direction

\( r \) = radial direction
\( \theta \) = theta direction
\( Z \) = axial direction

\( e_{rr} \) = radial strain
\( e_{\theta\theta} \) = hoop strain
\( e_{ZZ} \) = axial strain

\( e_{r\theta} \) = radial-hoop shear strain
\( e_{rZ} \) = radial-axial shear strain
\( e_{\theta Z} \) = hoop-axial shear strain

In general

\[
e_{rr} = \frac{\partial V_r}{\partial r}, \quad e_{\theta\theta} = \frac{V_r}{r} + \frac{1}{r} \frac{\partial V_\theta}{\partial \theta}, \quad e_{ZZ} = \frac{\partial V_Z}{\partial Z}
\]

\[
e_{r\theta} = \frac{1}{2} \frac{1}{r} \left( \frac{\partial V_r}{\partial \theta} + \frac{\partial V_\theta}{\partial r} - \frac{V_\theta}{r} \right)
\]

\[
e_{rZ} = \frac{1}{2} \left( \frac{\partial V_Z}{\partial r} + \frac{\partial V_r}{\partial Z} \right)
\]

\[
e_{\theta Z} = \frac{1}{2} \left( \frac{\partial V_\theta}{\partial Z} + \frac{1}{r} \frac{\partial V_Z}{\partial \theta} \right)
\]
In an axi-symmetric body under the action of axi-symmetric loading, there can be no variation of strains in the theta direction, so there can be neither of the shear strains $e_{r\theta}$, $e_{\theta z}$, also the second term of $e_{\theta \theta}$ disappears. So the strain equations become:

\[ e_{rr} = \frac{\partial u}{\partial r} \, , \, e_{\theta \theta} = \frac{u}{r} \, , \, e_{zz} = \frac{\partial v}{\partial z} \]

\[ e_{rz} = \frac{1}{2} \left( \frac{\partial u}{\partial r} + \frac{\partial v}{\partial z} \right) \]

or if $u = \text{radial displacement}$

$w = \text{axial displacement}$

\[ e_{rr} = \frac{\partial u}{\partial r} \, , \, e_{\theta \theta} = \frac{u}{r} \, , \, e_{zz} = \frac{\partial w}{\partial z} \, , \, e_{rz} = \frac{1}{2} \left( \frac{\partial u}{\partial r} + \frac{\partial w}{\partial z} \right) \]
(5) Strain in terms of displacements in matrix form

This is to obtain the three direct strains and the one shear strain ($e_{rz}$) in terms of the coefficients ($A_1$ to $A_{12}$) of the displacement functions and the general co-ordinates ($r, z$).

$$u = A_1 r^2 + A_2 r + A_3 + A_4 z^2 + A_5 z + A_6 rz$$

$$w = A_7 r^2 + A_8 r + A_9 + A_{10} z^2 + A_{11} z + A_{12} rz$$

$$e_{\theta\theta} = \frac{\partial u}{\partial r} = A_1 r + A_2 + \frac{A_3}{r} + \frac{A_4 z^2}{r} + \frac{A_5 z}{r} + A_6 z$$

$$e_{rr} = \frac{\partial^2 u}{\partial r^2} = 2A_1 r + A_2 + A_6 z$$

$$e_{zz} = \frac{\partial^2 u}{\partial z^2} = 2A_4 z + A_5 + A_6 r + 2A_7 r + A_8 + A_{12} z$$

$$e_{rz} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial r} = 2A_4 z + A_5 + A_6 r + 2A_7 r + A_8 + A_{12} z$$

$$\begin{pmatrix}
  e_{\theta\theta} \\
  e_{rr} \\
  e_{zz} \\
  e_{rz}
\end{pmatrix} =
\begin{pmatrix}
  \frac{u}{r} \\
  \frac{\partial u}{\partial r} \\
  \frac{\partial w}{\partial z} \\
  \frac{\partial u}{\partial z} + \frac{\partial w}{\partial r}
\end{pmatrix} =
\begin{pmatrix}
  r & 1/r & z^2/r & z/r & z \\
  2r & 1 & z \\
  2z & 1 & r \\
  2z & 1 & r & 2r & 1 & z
\end{pmatrix} \times
\begin{pmatrix}
  A_1 \\
  A_2 \\
  A_3 \\
  A_4 \\
  A_5 \\
  A_6 \\
  A_7 \\
  A_8 \\
  A_9 \\
  A_{10} \\
  A_{11} \\
  A_{12}
\end{pmatrix}$
In symbolic form

\[ \{e\} = [eA] \{A\} \]

and substituting for \( \{A\} \)

\[ \{e\} = [eA] \ [A^{-1} \{\delta\}] \]

Let \( [e\delta] \) represent \([eA] \ [A^{-1} \{\delta\}]\)

i.e. \( \{e\} = [e\delta] \{\delta\} \)
To obtain stresses in terms of strain

To obtain the three direct stresses and the shear stress \( \sigma_{rz} \) as a product of the elastic constants and the four strain values

\[
\begin{align*}
Ee_{\theta\theta} &= \sigma_{\theta\theta} - \nu \sigma_{rr} - \nu \sigma_{zz} \\
Ee_{rr} &= -\nu \sigma_{\theta\theta} + \sigma_{rr} - \nu \sigma_{zz} \\
Ee_{zz} &= -\nu \sigma_{\theta\theta} - \nu \sigma_{rr} + \sigma_{zz}
\end{align*}
\]

\[
D = \begin{vmatrix}
1 - \nu & -\nu \\
-\nu & 1 - \nu \\
-\nu & -\nu
\end{vmatrix} = (1 - \nu^3 - \nu^3) - (\nu^2 + \nu^2 + \nu^2)
\]

\[
D = 1 - 2\nu^3 - 3\nu^2 = (1 + \nu)(1 + \nu)(1 - 2\nu)
\]

\[
D\sigma_{\theta\theta} = \begin{vmatrix}
Ee_{\theta\theta} & -\nu & -\nu \\
Ee_{rr} & 1 - \nu & Ee_{zz} & -\nu & 1
\end{vmatrix} = E\begin{vmatrix}
e_{\theta\theta} & -\nu & -\nu \\
e_{rr} & 1 - \nu & e_{zz} & -\nu & 1
\end{vmatrix}
\]

\[
= E\{(e_{\theta\theta} + \nu^2 e_{zz} + \nu^2 e_{rr}) - (-\nu e_{zz} + \nu^2 e_{\theta\theta} - \nu e_{rr})\}
\]

\[
= E\{(1 - \nu^2)e_{\theta\theta} + \nu(1 + \nu)e_{rr} + \nu(1 + \nu)e_{zz}\}
\]

\[
= E\{(1 + \nu)(1 - \nu)e_{\theta\theta} + \nu(1 + \nu)e_{rr} + \nu(1 + \nu)e_{zz}\}
\]

\[
D\sigma_{\theta\theta} = E(1 + \nu)\{(1 - \nu)e_{\theta\theta} + \nu e_{rr} + \nu e_{zz}\}
\]
\[
\sigma_{\theta\theta} = \frac{D\sigma_{\theta\theta}}{D} = \frac{E(1 + \nu)}{(1 + \nu)(1 + \nu)(1 - 2\nu)} \left\{ (1 - \nu)\varepsilon_{\theta\theta} + \nu \varepsilon_{rr} + \nu \varepsilon_{zz} \right\}
\]

\[
\sigma_{\rho\rho} = \frac{E(1 - \nu)}{(1 + \nu)(1 - 2\nu)} \left\{ \frac{\nu}{1 - \nu} \varepsilon_{\theta\theta} + \frac{\nu}{1 - \nu} \varepsilon_{rr} + \frac{\nu}{1 - \nu} \varepsilon_{zz} \right\}
\]

Similarly

\[
\sigma_{\rho\rho} = \frac{E(1 - \nu)}{(1 + \nu)(1 - 2\nu)} \left\{ \frac{\nu}{1 - \nu} \varepsilon_{\theta\theta} + \frac{\nu}{1 - \nu} \varepsilon_{rr} + \frac{\nu}{1 - \nu} \varepsilon_{zz} \right\}
\]

and

\[
\sigma_{zz} = \frac{E(1 - \nu)}{(1 + \nu)(1 - 2\nu)} \left\{ \frac{\nu}{1 - \nu} \varepsilon_{\theta\theta} + \frac{\nu}{1 - \nu} \varepsilon_{rr} + \varepsilon_{zz} \right\}
\]

\[
e_{\rho z} = \frac{\sigma_{\rho z}}{G}, \quad \sigma_{\rho z} = G e_{\rho z} = \frac{E}{2(1 + \nu)} e_{\rho z}
\]

In matrix form

\[
\begin{bmatrix}
\sigma_{\theta\theta} \\
\sigma_{rr} \\
\sigma_{zz} \\
\sigma_{rz}
\end{bmatrix} = \frac{E(1 - \nu)}{(1 + \nu)(1 - 2\nu)} \begin{bmatrix}
1 & \frac{\nu}{1 - \nu} & \frac{\nu}{1 - \nu} & 0 \\
\frac{\nu}{1 - \nu} & 1 & \frac{\nu}{1 - \nu} & 0 \\
\frac{\nu}{1 - \nu} & \frac{\nu}{1 - \nu} & 1 & 0 \\
0 & 0 & 0 & \frac{1 - 2\nu}{2(1 - \nu)}
\end{bmatrix} \times \begin{bmatrix}
\varepsilon_{\theta\theta} \\
\varepsilon_{rr} \\
\varepsilon_{zz} \\
\varepsilon_{\rho z}
\end{bmatrix}
\]
In symbolic form

$$\{\sigma\} = [\sigma e] \{e\}$$

**Stress in terms of displacements**

Substituting $[eA] [\delta A]^{-1} \{\delta\}$ for $\{e\}$

$$\{\sigma\} = [\sigma e] [eA] [\delta A]^{-1} \{\delta\}$$

Let $[\sigma \delta]$ stand for $[\sigma e] [eA] [\delta A]^{-1}$

that is $\{\sigma\} = [\sigma \delta] \{\delta\}$
To obtain stiffness matrix

By the Principle of Virtual Displacements, the work done by nodial point forces is equated to the strain energy of the product of strain and stress over the volume of the triangular annulus

\[ \delta W = \delta U \]

\[ [\delta] \{q\} = \int [e] \{\sigma\} \, dv \]

as \( [e] = [e\delta] \{\delta\} \) so \( [e] = [\delta] \{e\delta\}^T \)

and \( [\sigma] = [\sigma\delta] \{\delta\} \)

so that

\[ [\delta] \{q\} = [\delta] \left( \int [e\delta]^T [\sigma\delta] \, dv \right) \{\delta\} \]

Pre-multiplying both sides of the equation by the inverse of the displacement vector \([\delta]\), gives the force vector \(\{q\}\) on the left hand side and its equivalent on the right hand side of the equation

\[ \{q\} = \left( \int [e\delta]^T [\sigma\delta] \, dv \right) \{\delta\} \]

Post-multiplying both sides of the equation by the inverse of the displacement vector \(\{\delta\}\), gives the stiffness matrix on the left hand side and its equivalent on the right

\[ [k] \{\delta\} = \left( \int [e\delta]^T [\sigma\delta] \, dv \right) \{\delta\} \]

\[ [k] = \int [e\delta]^T [\sigma\delta] \, dv \]
As

\[
[e\delta] = [eA][\delta A]^{-1}
\]

\[
[e\delta]^T = (([\delta A]^{-1})^T [eA]^T
\]

and

\[
[\delta \delta] = [\sigma e] [eA] [\delta A]^{-1}
\]

then

\[
[x] = (([\delta A]^{-1})^T [eA]^T [\sigma e] [eA]) dv ([\delta A]^{-1}
\]

Let \([Ae \sigma A]\) stand for \([eA]^T [\sigma e] [eA]\)

and this is dealt with in the next section.
APPENDIX 1b

Third compatibility equation in cylindrical polar co-ordinates

(Extracted from communication from Mr T Fuller which was received on 14th October 1972)

Differentiation of the strain component formula gives

\[
\frac{\partial e_{\theta z}}{\partial \theta} = \frac{\partial^2 u_\theta}{\partial \theta, \partial z} + \frac{1}{r} \frac{\partial^2 u_z}{\partial \theta^2} \\
\tag{1}
\]

\[
\frac{\partial (r e_{\theta \theta})}{\partial z} = \frac{\partial^2 u_\theta}{\partial z} + \frac{\partial u_r}{\partial z} \\
\tag{2}
\]

\[
\frac{\partial e_{r z}}{\partial z} = \frac{\partial^2 u_r}{\partial z^2} + \frac{\partial^2 u_z}{\partial z, \partial \theta} \\
\tag{3}
\]

\[
\frac{\partial e_{zz}}{\partial r} = \frac{\partial^2 u_z}{\partial r, \partial z} \\
\tag{4}
\]

\[
\frac{\partial e_{\theta z}}{\partial \theta} = \frac{\partial^2 u_z}{\partial \theta^2} \\
\tag{5}
\]

\[
\frac{\partial^2 e_{\theta z}}{\partial z, \partial \theta} = \frac{\partial^3 u_\theta}{\partial \theta, \partial z^2} + \frac{1}{r} \frac{\partial^3 u_z}{\partial \theta, \partial \theta^2} \\
\tag{6} \text{from (1)}
\]

\[
\frac{\partial^2 (r e_{\theta \theta})}{\partial z^2} = \frac{\partial^3 u_\theta}{\partial z^2, \partial \theta} + \frac{\partial^3 u_r}{\partial z^2} \\
\tag{7} \text{from (2)}
\]

\[
\frac{\partial^2 u_r}{\partial z^2} = \frac{\partial e_{r z}}{\partial z} - \frac{\partial^2 u_z}{\partial z, \partial \theta} \\
\tag{8} \text{from (3)}
\]

\[
\frac{\partial^2 u_z}{\partial r, \partial z} = \frac{\partial e_{zz}}{\partial r} \\
\tag{9} \text{from (4)}
\]

\[
\frac{1}{r} \frac{\partial^3 u_z}{\partial \theta^2, \partial z} = \frac{4}{r} \frac{\partial^2 e_{zz}}{\partial \theta^2} \\
\tag{10} \text{from (5)}
\]

Starting with (6) and using (7) to (10) in sequence

\[
\frac{\partial^2 e_{\theta z}}{\partial \theta, \partial z} = \frac{\partial^2 (r e_{\theta \theta})}{\partial z^2} + \left\{ \frac{\partial e_{r z}}{\partial z} - \frac{\partial e_{zz}}{\partial \theta} \right\} + \frac{1}{r} \frac{\partial^2 e_{zz}}{\partial \theta^2}
\]
Test that compatibility is satisfied

The three compatibility equations in cylindrical polar co-ordinates are:

\[
\frac{\partial^2 (r e_\theta)}{\partial r \partial \theta} = r \frac{\partial^2 (r e_\theta)}{\partial r^2} - r \frac{\partial e_r}{\partial r} + \frac{\partial^2 e_r}{\partial \theta^2} \quad \ldots \ldots (1)
\]

\[
\frac{\partial^2 e_{rz}}{\partial r \partial z} = \frac{\partial^2 e_r}{\partial z^2} + \frac{\partial^2 e_{zz}}{\partial r^2} \quad \ldots \ldots (2)
\]

\[
\frac{\partial^2 e_{oz}}{\partial \theta \partial z} = \frac{\partial^2 (r e_\theta)}{\partial z^2} - \left( \frac{\partial e_{rz}}{\partial z} - \frac{\partial e_{zz}}{\partial r} \right) + \frac{1}{r} \frac{\partial^2 e_{zz}}{\partial \theta^2} \quad \ldots \ldots (3)
\]

Considering the first equation, the term to left of the equals must be zero because for an axi-symmetric body with axi-symmetric loading the shear strain \( e_r \) must be zero. The first term right of the equals

\[
r \frac{\partial^2 (r e_\theta)}{\partial r^2} = r \left( \frac{\partial}{\partial r} \left[ e_\theta + \frac{r \partial e_\theta}{\partial r} \right] \right)
\]

\[
= r \left( \frac{\partial}{\partial r} \left[ e_\theta + \frac{r \partial e_\theta}{\partial r} \right] \right) = r \left\{ \frac{\partial e_\theta}{\partial r} + \frac{r \partial e_\theta}{\partial r} \left[ \frac{\partial}{\partial r} \left[ \frac{r \partial e_\theta}{\partial r} \right] \right] \right\}
\]

\[
= r \left( \frac{\partial e_\theta}{\partial r} + \frac{r \partial e_\theta}{\partial r} + \frac{r \partial e_\theta}{\partial r} \cdot \frac{1}{r} \right)
\]

\[
= 2r \frac{\partial e_\theta}{\partial r} + r^2 \frac{\partial^2 e_\theta}{\partial r^2}
\]

inserting this result back into equation (1) but because we are dealing with an axi-symmetric body under axi-symmetric loading the radial strain \( e_r \) cannot vary with \( \theta \); therefore the last term right of the equals disappears. On dividing by \( r^2 \) equation (1) becomes

\[
0 = \frac{\partial^2 e_\theta}{\partial r^2} + 2 \frac{\partial e_\theta}{\partial r} \cdot \frac{1}{r} \frac{\partial e_r}{\partial r} \quad \ldots \ldots (1a)
\]
Considering the third equation, the term to the left of the equals and the last term to the right of the equals must be zero for an axi-symmetric body with axi-symmetric loading. The first term to the right of the equals

\[
\frac{\partial^2 (r e_{66})}{\partial z^2} = \frac{\partial}{\partial z} \left( r \frac{\partial e_{66}}{\partial z} + e_{66} \frac{\partial}{\partial z} r \right)
\]

(and as \( \frac{\partial}{\partial z} r = 0 \))

\[
= \frac{\partial}{\partial z} \left( r \frac{\partial e_{66}}{\partial z} \right) = r \frac{\partial^2 e_{66}}{\partial z^2} + \frac{\partial e_{66}}{\partial z} \cdot r
\]

inserting this result back into equation (3) it becomes

\[
0 = r \frac{\partial^2 e_{66}}{\partial z^2} + \frac{\partial e_{xx}}{\partial z} - \frac{\partial e_{xz}}{\partial z}
\]

\[
\text{.... (3a)}
\]

Thus, the three compatibility equations in cylindrical polar co-ordinates, when related to an axi-symmetric body subjected to axi-symmetric loading, become:

\[
0 = \frac{\partial^2 e_{66}}{\partial r^2} + \frac{1}{r} \frac{\partial e_{66}}{\partial r} - \frac{\partial e_{rr}}{\partial r}
\]

\[
\text{.... (1a)}
\]

\[
\frac{\partial^2 e_{zz}}{\partial r \partial z} = \frac{\partial^2 e_{zz}}{\partial r^2} + \frac{\partial^2 e_{zz}}{\partial r^2}
\]

\[
\text{.... (2)}
\]

\[
0 = r \frac{\partial^2 e_{66}}{\partial z^2} + \frac{\partial e_{xx}}{\partial r} - \frac{\partial e_{xz}}{\partial z}
\]

\[
\text{.... (3a)}
\]

In the section on the theory relating to the Formation of the Stiffness Matrix the three direct strains and the only shear strain are given in terms of the coefficients (\(A_1\) to \(A_{12}\)) of the displacement functions and the general co-ordinates (\(r, z\)).
\[ e_{\theta\theta} = A_1 r + A_2 + \frac{A_3}{r} + \frac{A_4 z^2}{r} + \frac{A_5 z}{r} + A_6 z \]

\[ e_{\rho\rho} = 2A_1 r + A_2 + A_6 z \]

\[ e_{zz} = 2A_4 z + A_5 + A_6 r + 2A_7 r + A_8 + A_{12} z \]

So that

\[ \frac{\partial e_{\theta\theta}}{\partial r} = A_1 - \frac{1}{r^2} \left( A_3 + A_4 z^2 + A_5 z \right) \]

\[ \frac{\partial^2 e_{\theta\theta}}{\partial r^2} = \frac{2}{r^3} \left( A_3 + A_4 z^2 + A_5 z \right) \]

\[ \frac{2}{r} \frac{\partial e_{\theta\theta}}{\partial r} = \frac{2A_1}{r} - \frac{2}{r^3} \left( A_3 + A_4 z^2 + A_5 z \right) \]

\[ \frac{\partial e_{rr}}{\partial r} = 2A_1 \]

\[ \frac{1}{r} \frac{\partial e_{rr}}{\partial r} = \frac{2A_1}{r} \]

Substituting into equation (1a)

\[ \frac{\partial^2 e_{\theta\theta}}{\partial r^2} + \frac{2}{r} \frac{\partial e_{\theta\theta}}{\partial r} - \frac{1}{r} \frac{\partial e_{rr}}{\partial r} \]

\[ = \frac{2}{r^3} \left( A_3 + A_4 z^2 + A_5 z \right) + \frac{2A_1}{r} - \frac{2}{r^3} \left( A_3 + A_4 z^2 + A_5 z \right) - \frac{2A_1}{r} = 0 \]
Considering the second equation

\[
\frac{\partial e_{rz}}{\partial z} = 2A_4 + A_{12} \quad \therefore \frac{\partial^2 e_{rz}}{\partial r \partial z} = 0
\]

\[
\frac{\partial e_{rr}}{\partial z} = A_6 \quad \therefore \frac{\partial^2 e_{rr}}{\partial z^2} = 0
\]

\[
\frac{\partial e_{zz}}{\partial r} = A_{12} \quad \therefore \frac{\partial^2 e_{zz}}{\partial r^2} = 0
\]

So that the second equation

\[\frac{\partial^2 e_{rz}}{\partial r \partial z} + \frac{\partial^2 e_{rr}}{\partial z^2} + \frac{\partial^2 e_{zz}}{\partial r^2} = 0\]

becomes

\[-0 + 0 + 0 = 0\]

Considering the third equation (3a)

\[
\frac{\partial e_{\theta\theta}}{\partial z} = \frac{2A_4}{r} + \frac{A_5}{r} + A_6
\]

\[
\frac{\partial^2 e_{\theta\theta}}{\partial z^2} = \frac{2A_4}{r}
\]

\[
x \frac{\partial^2 e_{\theta\theta}}{\partial z^2} = 2A_4
\]
\[ \frac{\partial e_{zz}}{\partial r} = A_{12} \]

\[ \frac{\partial e_{rz}}{\partial z} = 2A_4 + A_{12} \]

Substituting into the equation (3a)

\[ r \frac{\partial^2 e_{\theta \theta}}{\partial z^2} + \frac{\partial e_{zz}}{\partial r} - \frac{\partial e_{rz}}{\partial z} \]

\[ = 2A_4 + A_{12} - (2A_4 + A_{12}) \]

\[ = 0 \]
Integrals

To evaluate the integral $\int [eA]^T [\sigma e] [eA] dv$, where $[eA]$ is the matrix relating strains to coefficients of the displacement function and $[\sigma e]$ is the matrix relating stress to strains. Firstly the matrix $[eA]$ is premultiplied by the matrix $[\sigma e]$ and the product is called $[\sigma A]$. Then the matrix $[eA]$ is transposed to form $[eA]^T$. Then the matrix $[\sigma A]$ is premultiplied by the matrix $[eA]^T$ and the product is called $[Ae\sigma A]$

$$\int [Ae\sigma A] dv = 2\pi \int [Ae\sigma A] r dr dz,$$

and so $[Ae\sigma A]$ is multiplied by $r$ before carrying out the integration. It is then found within this $12 \times 12$ matrix there are fifteen basic integrals.
\[
\mathbf{[cA]} = \mathbf{[ce]} \mathbf{[eA]}
\]

\[
\frac{E}{(1+v)(1-2v)} \begin{bmatrix}
1-v & v & v & 0 \\
v & 1-v & v & 0 \\
v & v & 1-v & 0 \\
0 & 0 & 0 & \frac{1-2v}{2}
\end{bmatrix} \begin{bmatrix}
r & 1 & 1/r & z^2/r & z/r & z \\
2r & 1 & & & & \\
& & & & & \\
& & & & & \\
2z & 1 & r & 2r & 1 & z
\end{bmatrix}
\]

\[
\begin{bmatrix}
(1+v)r & 1 & (1-v)/r & (1-v)z^2/r & (1-v)z/r & z & 0 & 0 & 0 & 2vz & v & vr \\
(2-v)r & 1 & v/r & vz^2/r & vz/r & z & 0 & 0 & 0 & 2vz & v & vr \\
3vr & 2v & v/r & vz^2/r & vz/r & 2vz & 0 & 0 & 0 & 2(1-v)z & (1-v) & (1-v)r \\
0 & 0 & 0 & (1-2v)z & (1-2v)/2 & (1-2v)r/2 & (1-2v)r & (1-2v)/2 & 0 & 0 & 0 & (1-2v)z/2
\end{bmatrix}
\]
$[\mathbf{eA}]^{-T} =$

\[
\begin{pmatrix}
  r & 2r \\
  1 & 1 \\
  1/r & \\
  z^2/r & 2z \\
  z/r & 1 \\
  z & z \\
  r & z \\
  2z & \\
  1 & \\
  r & z
\end{pmatrix}
\]
\[ [A_{ecA}] = \]

(as follows on the next page)
<table>
<thead>
<tr>
<th>((5-u)r^2)</th>
<th>(3r)</th>
<th>(1+u)</th>
<th>((1+u)r^2)</th>
<th>((1+u)r)</th>
<th>(3rz)</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>6uvr</th>
<th>3vr</th>
<th>3vr^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(3r)</td>
<td>2</td>
<td>(1/r)</td>
<td>(z^2/r)</td>
<td>(z/r)</td>
<td>2z</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4vr</td>
<td>2v</td>
<td>2vr</td>
</tr>
<tr>
<td>(1+u)</td>
<td>(1/r)</td>
<td>((1-u)/r^2)</td>
<td>((1-u)r^2/r^2)</td>
<td>((1-u)r^2/r^2)</td>
<td>(z/r)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2uvr</td>
<td>vr</td>
<td>v</td>
</tr>
<tr>
<td>((1+u)r^2)</td>
<td>(z^2/r)</td>
<td>((1-u)r^2/r^2)</td>
<td>((1-u)r^2/r^2)</td>
<td>((1-u)r^2/r^2)</td>
<td>(z^2/r+2(1-2u)r^2)</td>
<td>(z/r+((1-2u)r^2/2)</td>
<td>((1-2u)v^2/r)</td>
<td>((1-2u)r)</td>
<td>0</td>
<td>2uv^2/r</td>
<td>vz/r</td>
</tr>
<tr>
<td>((1+u)r)</td>
<td>(z/r)</td>
<td>((1-u)r^2/r^2)</td>
<td>((1-u)r^2/r^2)</td>
<td>((1-u)r^2/r^2)</td>
<td>((1-u)r^2+2(1-2u)r^2/2)</td>
<td>((1-2u)v^2/r)</td>
<td>((1-2u)r)</td>
<td>((1-2u)/2)</td>
<td>0</td>
<td>2uv^2/r</td>
<td>vz/r</td>
</tr>
<tr>
<td>3rz</td>
<td>2z</td>
<td>(z/r)</td>
<td>(z^2/r+(1-2u)r^2)</td>
<td>(z^2/r+(1-2u)r^2/2)</td>
<td>((1-2u)v^2/r)</td>
<td>((1-2u)r^2/2)</td>
<td>((1-2u)r)</td>
<td>((1-2u)/2)</td>
<td>0</td>
<td>4uv^2</td>
<td>2vr</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>(2(1-2u)r^2)</td>
<td>((1-2u)r)</td>
<td>((1-2u)r^2)</td>
<td>((1-2u)r^2/2)</td>
<td>((1-2u)r)</td>
<td>((1-2u)/2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>((1-2u)r)</td>
<td>((1-2u)/2)</td>
<td>((1-2u)r/2)</td>
<td>((1-2u)r)</td>
<td>((1-2u)/2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6vr</td>
<td>4uv</td>
<td>2uvr</td>
<td>((2uv+3)/r)</td>
<td>(2uv^2/r)</td>
<td>(4uv^2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4(1-u)r^2</td>
<td>2(1-u)r</td>
<td>2(1-u)r</td>
</tr>
<tr>
<td>3vr</td>
<td>2v</td>
<td>vr</td>
<td>(v^2/r)</td>
<td>(vz/r)</td>
<td>2uv</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2(1-u)r</td>
<td>((1-u)r)</td>
<td>((1-u)r)</td>
</tr>
<tr>
<td>3vr^2</td>
<td>2uvr</td>
<td>((1-u)r^2)</td>
<td>((1-u)r^2)</td>
<td>(z/2)</td>
<td>((1+2u)r^2/2)</td>
<td>((1-2u)r^2)</td>
<td>((1-2u)r^2/2)</td>
<td>((1-2u)r)</td>
<td>((1-2u)r/2)</td>
<td>0</td>
<td>2(1-u)r</td>
</tr>
</tbody>
</table>
\[ r[AeoA] = \]

(as follows on the next page)
<table>
<thead>
<tr>
<th></th>
<th>((5-v)^3)</th>
<th>(3r^2)</th>
<th>((1+v)r)</th>
<th>((1+v)rz^2)</th>
<th>((1+v)rz)</th>
<th>(3r^2z)</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>6vr^2z</th>
<th>3vr^2</th>
<th>3vr^3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(3r^2)</td>
<td>2r</td>
<td>1</td>
<td>(z^2)</td>
<td>(2rz)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2vrz</td>
<td>(2vr)</td>
<td>2vr^2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>((1+v)r)</td>
<td>1</td>
<td>((1-v)/r)</td>
<td>((1-v)z^2/r)</td>
<td>(z)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2vrz</td>
<td>(v)</td>
<td>(vr)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((1+v)rz^2)</td>
<td>(z^2)</td>
<td>((1-v)z^2/r)</td>
<td>((1-v)z^4/r + 2(1-2v)rz^2)</td>
<td>((1-v)z^3/r + (1-2v)rz)</td>
<td>(2(1-2v)r^2z)</td>
<td>(2(1-2v)rz)</td>
<td>0</td>
<td>2vr^3</td>
<td>(vz^2)</td>
<td>((1-v)rz^2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((1+v)rz)</td>
<td>(z)</td>
<td>((1-v)z/r)</td>
<td>((1-v)z^3/r + (1-2v)rz)</td>
<td>((1-v)z^2/r + (1-2v)rz/2)</td>
<td>(z^2 + (1-2v)r^2z/2)</td>
<td>((1-2v)r^2)</td>
<td>((1-2v)r^2/2)</td>
<td>0</td>
<td>2vr^2</td>
<td>(vz)</td>
<td>(rz/2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3r^2z)</td>
<td>(2rz)</td>
<td>(z^2 + (1-2v)r^2z/2)</td>
<td>(2rz^2 + (1-2v)r^3/2)</td>
<td>((1-2v)r^3)</td>
<td>((1-2v)r^2/2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>((1-2v)r^2z)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>(2(1-2v)r^2z)</td>
<td>((1-2v)r^2)</td>
<td>((1-2v)r^3)</td>
<td>((1-2v)r^2/2)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>((1-2v)rz/2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>((1-2v)rz)</td>
<td>((1-2v)r^2/2)</td>
<td>((1-2v)r^2)</td>
<td>(0)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>((1-2v)rz/2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(6vr^2z)</td>
<td>4vrz</td>
<td>2vrz</td>
<td>(2vr^3)</td>
<td>(2vrz)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>(4(1-v)rz^2)</td>
<td>(2(1-v)rz)</td>
<td>(2(1-v)r^2z)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3vr^2)</td>
<td>2vr</td>
<td>(v)</td>
<td>(vz^2)</td>
<td>(vz)</td>
<td>2vrz</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>(2(1-v)rz)</td>
<td>((1-v)r)</td>
<td>((1-v)r^2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3vr^3)</td>
<td>(2vr^2)</td>
<td>(vr)</td>
<td>((1-v)rz^2)</td>
<td>(rz/2)</td>
<td>((1+2v)r^2z/2)</td>
<td>((1-2v)r^2z)</td>
<td>((1-2v)rz/2)</td>
<td>0</td>
<td>(2(1-v)r^2z)</td>
<td>((1-v)r^2)</td>
<td>((1-v)r^3 + (1-2v)rz^2/2)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Looking at the matrix $r[\text{AeA}]$, fifteen separate integrals are required; they are:

$$S(1) = \int\int r\, dz\, dr$$

$$S(2) = \int\int rz\, dz\, dr$$

$$S(3) = \int\int r^2\, dz\, dr$$

$$S(4) = \int\int r^3\, dz\, dr$$

$$S(5) = \int\int rz^2\, dr\, dz$$

$$S(6) = \int\int r^2z\, dz\, dr$$

$$S(7) = \int\int dz\, dr$$

$$S(8) = \int\int z\, dr\, dz$$

$$S(9) = \int\int z^2\, dr\, dz$$

$$S(10) = \int\int z^3\, dr\, dz$$

$$S(11) = \int\int \frac{1}{r}\, dz\, dr$$

$$S(12) = \int\int \frac{z}{r}\, dz\, dr$$

$$S(13) = \int\int \frac{z^2}{r}\, dz\, dr$$

$$S(14) = \int\int \frac{z^3}{r}\, dz\, dr$$

$$S(15) = \int\int \frac{z^4}{r}\, dz\, dr$$

The formula for these integrals are derived after deducing the formula for slope and intercept.
\[ z = A_{ij} + B_{ij} r \]

so that

when \( r = r_j \) then \( z = z_j \)
when \( r = r_i \) then \( z = z_i \)

subtracting

so that slope

so that

\[ A_{ij} + B_{ij} r = z_j \]
\[ A_{ij} + B_{ij} r_i = z_i \]
\[ B_{ij} (r_j - r_i) = z_j - z_i \]
\[ B_{ij} = \frac{z_j - z_i}{r_j - r_i} \]

\[ A_{ij} = \frac{r_j z_i - r_i z_j}{r_j - r_i} \]

Intercept

If \( z_i = z_j = z \), slope \( B_{ij} = 0 \) and intercept \( A_{ij} = z \)

If \( r_i = r_j = r \), slope \( B_{ij} = \tan \frac{\pi}{2} \) and intercept is non-existent
\[ r = C_{ij} + D_{ij}z \]

so that

If \( z = z_j \) then \( r = r_j \)

\[ C_{ij} + D_{ij}z_j = r_j \]

If \( z = z_i \) then \( r = r_i \)

\[ C_{ij} + D_{ij}z_i = r_i \]

Subtracting

\[ D_{ij}(z_j - z_i) = r_j - r_i \]

so slope

\[ D_{ij} = \frac{r_j - r_i}{z_j - z_i} \]

\[ C_{ij} + D_{ij}z_i = r_i \]

\[ C_{ij} + D_{ij}z_j = r_j \]

Determinant

\[ D_{C_{ij}} = \begin{vmatrix} r_i & z_i \\ r_j & z_j \end{vmatrix} = r_iz_j - r_jz_i \]

and determinant

\[ D = \begin{vmatrix} 1 & z_i \\ 1 & z_j \end{vmatrix} = z_j - z_i \]

Intercept

\[ C_{ij} = \frac{r_iz_i - r_jz_j}{z_j - z_i} \]

If \( r_i = r_j = r \), slope \( D_{ij} = 0 \) and intercept \( C_{ij} = r \)

If \( z_i = z_j = z \), slope \( D_{ij} = \tan \frac{\pi}{2} \) and intercept \( C_{ij} \) is non-existent

118
\[
\int_{r=r_i}^{r=r_j} \int_{z=0}^{z=A_i+B_j} r \, dz \, dr = \int_{r=r_i}^{r=r_j} (A_i r + B_j r^2) \, dr = \left[ \frac{A_i r^2}{2} + \frac{B_j r^3}{3} \right]_{r_i}^{r_j}
\]

\[
= \frac{A_i}{2}(r_j^2 - r_i^2) + \frac{B_j}{3}(r_j^3 - r_i^3)
\]

\[
= \frac{A_i}{2}(r_j^2 - r_i^2)(r_j + r_i) + \frac{B_j}{3}(r_j^3 - r_i^3)(r_j + r_i + r_i^2)
\]

\[
= \frac{1}{2(r_j - r_i)} (r_j^3 - r_i^3)(r_j + r_i) + \frac{1}{3(r_j - r_i)} (r_j^3 - r_i^3)(r_j^2 + r_j r_i + r_i^2)
\]

\[
= \frac{1}{2} (r_j^2 z_i - r_i z_i)(r_j + r_i) + \frac{1}{3} (z_j - z_i)(r_j^2 + r_j r_i + r_i^2)
\]

\[
= \frac{1}{2} (r_j^2 z_i + r_i r_j z_i - r_i r_j z_j - r_i^2 z_j) / 2
\]

\[
+ \frac{1}{3} (r_j^2 z_i + r_i r_j z_i + r_i^2 z_i - r_i r_j z_i - r_i^2 z_i) / 3
\]

\[
= \frac{1}{2} (-3r_i^2 z_i + 3r_i r_j z_i - 3r_i r_j z_j + 3r_j^2 z_i)
\]

\[
- 2r_i^2 z_i - 2r_i^2 z_j - 2r_i r_j z_i + 2r_i r_j z_j - 2r_i^2 z_i + 2r_j^2 z_j / 6
\]

\[
= \frac{1}{2} (-2r_i^2 z_i - r_i r_j z_i - r_i r_j z_j + r_i^2 z_i + 2r_j^2 z_j) / 6
\]

\[
\int_{j}^{k} \int_{r}^{z} dr \, dz = \frac{1}{6} \left( -2r_j^2 z_j - r_j r_k z_j - r_j r_k z_j - r_j^2 z_k + 2r_k^2 z_k \right)
\]

\[
\int_{i}^{k} \int_{z}^{r} dr \, dz = \frac{1}{6} \left( -2r_k^2 z_k - r_k r_i z_i - r_k r_i z_i + r_k^2 z_k + 2r_i^2 z_i \right)
\]

\[
\text{Sum} = \frac{1}{6} \left( -r_i^2 z_j + r_i^2 z_k - r_j^2 z_k + r_j^2 z_i - r_j^2 z_i + r_k^2 z_j
\]

\[
+ r_i r_j z_i - r_i r_j z_j + r_j r_k z_j - r_j r_k z_k + r_k r_i z_i - r_k r_i z_k \right) / 6
\]

\[
= \left( r_i^2 (z_k - z_j) + r_j^2 (z_i - z_k) + r_k^2 (z_j - z_i) \right)
\]

\[
- r_j r_k (z_k - z_j) - r_k r_j (z_i - z_k) - r_i r_j (z_j - z_i) / 6
\]

\[
= \left( (r_i^2 - r_j r_k) (z_k - z_j) + (r_j^2 - r_k r_j) (z_i - z_k) + (r_k^2 - r_i r_j) (z_j - z_i) \right) / 6.
\]
Integral S(1) continued

Let \( a = r_i, \ b = r_j, \ c = r_k \) and \( x = z_i, \ y = z_j, \ z = z_k \)

\[
\text{Sum} = \frac{1}{6} \left( (a^2 - bc) (z - y) + (b^2 - ca) (x - z) + (c^2 - ab) (y - x) \right).
\]
Integral $S(2)$:

\[
\begin{align*}
\int_{r=r_j}^{r=r_j} \int_{z=0}^{z=A_{ij}r} r\,dz\,dr &= \int_{r=r_j}^{r=r_j} \frac{r}{2}(A_{ij} + B_{ij}r)^2 \,dr \\
&= \frac{1}{2} \int_{r=r_j}^{r=r_j} (A_{ij}^2 r + 2 A_{ij} B_{ij} r^2 + B_{ij}^2 r^3) \,dr \\
&= \frac{A_{ij}^2}{4} [r^2]_i^j + \frac{A_{ij}}{3} B_{ij} [r^3]_i^j + \frac{B_{ij}^2}{6} [r^4]_i^j \\
&= \frac{A_{ij}^2}{4} (r_j^2 - r_i^2) + \frac{A_{ij}}{3} B_{ij} (r_j^3 - r_i^3) + \frac{B_{ij}^2}{6} (r_j^4 - r_i^4) \\
&= \frac{A_{ij}^2}{4} (r_j - r_i)(r_i + r_j) + \frac{A_{ij}}{3} B_{ij} (r_j - r_i)(r_j^2 + r_i r_j + r_i^2) \\
&\quad + \frac{B_{ij}^2}{6} (r_j - r_i)(r_i + r_j)(r_i^2 + r_j^2) \\
&= \frac{A_{ij}^2}{4} (r_j - r_i)(r_i + r_j) + \frac{A_{ij}}{3} B_{ij} (r_j - r_i)(r_j^2 + r_i r_j + r_i^2) \\
&\quad + \frac{B_{ij}^2}{6} (r_j - r_i)(r_i + r_j)(r_i^2 + r_j^2) \\
&= \frac{(r_j + r_i)}{4} (r_j z_i - r_i z_j)^2 + (r_i^2 + r_i r_j + r_j^2) (r_j z_i - r_i z_j) \frac{(z_i - z_j)}{3} \\
&\quad + (r_i^3 + r_j^2 r_i + r_i r_j^2 + r_j^3) \frac{(z_i - z_j)^2}{8} (r_j - r_i) \\
&\quad \times \left\{ \frac{(r_i + r_j)}{4} (r_j^2 z_i^2 - 2 r_j r_i z_i z_j + r_i^2 z_j^2) + (\frac{r_i^2 + r_i r_j + r_j^2}{3}) (r_j z_i z_j - r_i z_j^2 - r_i z_i^2) + (r_i^3 + r_j^2 r_i + r_i r_j^2 + r_j^3) \frac{z_i^2 - 2 z_i z_j + z_j^2}{8} (r_j - r_i) \right\} \\
&\quad \times (r_j - r_i)^3 \\
&= \frac{1}{8} \left\{ \frac{(r_i + r_j)}{4} (r_j^2 z_i^2 - 2 r_j r_i z_i z_j + r_i^2 z_j^2) + (\frac{r_i^2 + r_i r_j + r_j^2}{3}) (r_j z_i z_j - r_i z_j^2 - r_i z_i^2) + (r_i^3 + r_j^2 r_i + r_i r_j^2 + r_j^3) \frac{z_i^2 - 2 z_i z_j + z_j^2}{8} (r_j - r_i) \right\} \times (r_j - r_i)^3
\end{align*}
\]
Integral $S(2)$ continued

\[
\begin{align*}
&\{ (r_i^3 z_i^2 - 2r_i^2 r_j z_i z_j + r_i^2 r_j z_j^2 + r_i r_j^2 z_i - 2r_i r_j z_i z_j + r_j^2 z_i^2) / 4 \\
+ & \left( r_i^3 z_i z_j - r_i^3 z_j^2 - r_i^2 r_j z_i^2 + 2r_i^2 r_j z_i z_j - r_i^2 r_j z_j^2 - r_i r_j^2 z_i^2 \right) / 3 \\
+ & \left( r_i^3 z_i^2 - 2r_i^3 z_i z_j + r_i^3 z_j^2 + r_i^2 r_j z_i^2 - 2r_i r_j z_i z_j + r_1^2 r_j z_j^2 \right) / 8 \\
+ & r_i r_j^2 z_i z_j + r_i r_j^2 z_i z_j^2 + r_i r_j^2 z_j z_i + r_i r_j^2 z_j z_j^2 \\
= & \left( 6r_i^3 z_i^2 - 12r_i^2 r_j z_i z_j + 6r_i^2 r_j z_j^2 + 6r_i r_j^2 z_i^2 - 12r_i r_j^2 z_i z_j + 6r_j^2 z_i^2 \\
+ & 8r_i^3 z_i z_j - 8r_i^3 z_j^2 - 8r_i^2 r_j^2 z_i - 16r_i^2 r_j^2 z_i z_j - 8r_i^2 r_j^2 z_j - 8r_i r_j^2 z_i^2 \\
+ & 16r_i r_j^2 z_i z_j - 8r_i r_j^2 z_j^2 - 8r_i^2 z_i^2 + 8r_i^2 z_j^2 \\
+ & 3r_i^3 z_i^2 - 6r_i^3 z_i z_j + 3r_i^3 z_j^2 + 3r_i^2 r_j z_i^2 - 6r_i^2 r_j z_i z_j \\
+ & 3r_i^2 r_j z_i^2 + 3r_i r_j^2 z_i^2 - 6r_i r_j^2 z_i z_j + 3r_i r_j^2 z_j^2 + 3r_j^3 z_i^2 \\
- & 6r_j^3 z_i z_j + 3r_j^3 z_j^2 \right) \\
& \times \frac{1}{24(r_j - r_i)} \\
= & \left( 3r_i^3 z_i^2 + 2r_i^3 z_i z_j + r_i^3 z_j^2 - 5r_i^2 r_j z_i^2 - 2r_i r_j^2 z_i z_j + r_j^2 r_j z_j^2 \\
+ & r_i r_j^2 z_i^2 - 2r_i r_j^2 z_i z_j - 5r_i r_j^2 z_j^2 + r_j^2 z_i^2 \\
+ & 2r_j^3 z_i z_j + 3r_j^3 z_j^2 \right) / (24(r_j - r_i))
\end{align*}
\]
Integral S(2) continued

\[ \left\{ z_i^2(3r_i^3 - 5r_i^2 r_j + r_i r_j^2 + r_j^3) + z_i z_j(2r_i^3 - 2r_i^2 r_j - 2r_i r_j^2 + 2r_j^3) + z_j^2(r_i^3 + r_i^2 r_j - 5r_i r_j^2 + 3r_j^3) \right\} / (24(r_j - r_i)) \]

\[ = \left\{ z_i^2(r_j - r_i)(-3r_i^2 + 2r_i r_j + r_j^2) + 2z_i z_j(r_j - r_i)(r_j - r_i) \right\} / (24(r_j - r_i)) \]

\[ r_j \int_{r_i} rz dz dr = \left\{ z_i^2(-3r_i^2 + 2r_i r_j + r_j^2) + 2z_i z_j(r_j - r_i)(r_j - r_i) \right\} / 24 \]

\[ r_j \int_{r_i} r dz dr = \left\{ z_i^2(-3r_i^2 + 2r_i r_j + r_j^2) + 2z_i z_j(r_j - r_i)(r_j - r_i) \right\} / 24 \]

\[ r_k \int_{r_j} rz dz dr = \left\{ z_j^2(-3r_j^2 + 2r_j r_k + r_k^2) + 2z_j z_k(r_k - r_j)(r_k - r_j) \right\} / 24 \]

\[ r_k \int_{r_j} r dz dr = \left\{ z_j^2(-3r_j^2 + 2r_j r_k + r_k^2) + 2z_j z_k(r_k - r_j)(r_k - r_j) \right\} / 24 \]

Summing \[ = \left\{ z_i^2(2r_i r_j + r_j^2 - r_i r_j) + 2z_i z_j(r_j - r_i) \right\} / (24(r_j - r_i)) \]

\[ + z_j^2(2r_j r_k + r_k^2 - r_j r_k) + 2z_j z_k(r_k - r_j) \]

\[ + z_k^2(2r_k r_i + r_i^2 - r_k r_i) + 2z_k z_i(r_i - r_k) \] / 24

Let \( a = r_i, \ b = r_j, \ c = r_k \)

Then sum \[ = \left\{ x^2(2ab + b^2 - c^2 - 2ac) + 2xy(b^2 - a^2) \right\} / 24 \]

\[ + y^2(2bc + c^2 - a^2 - 2ab) + 2yz(c^2 - b^2) \]

\[ + z^2(2ca + a^2 - b^2 - 2bc) + 2zx(a^2 - c^2) \] / 24

\[ = \left\{ x[x(2ab - 2ac + b^2 - c^2) - 2y(a^2 - b^2)] \right\} / 24 \]

\[ + y[y(2bc - 2ab + c^2 - a^2) - 2z(b^2 - c^2)] \]

\[ + z[z(2ca - 2bc + a^2 - b^2) - 2x(c^2 - a^2)] \] / 24

\[ = \left\{ x[x(2a(b-c) + (b+c)(b-c)) - 2y(a+b)(a-b)] \right\} / 24 \]

\[ + y[y(2b(c-a) + (c+a)(c-a)) - 2z(b+c)(b-c)] \]

\[ + z[z(2c(a-b) + (a+b)(a-b)) - 2x(c+a)(c-a)] \] / 24
Integral $S(2)$ continued

\[
\begin{align*}
&= \{x[(b-c)(2a+b+c) - 2y(a+b)(a-b)] \\
&\quad + y[(c-a)(2b+c+a) - 2z(b+c)(b-c)] \\
&\quad + z[(a-b)(2c+a+b) - 2x(c+a)(c-a)]\}/24
\end{align*}
\]
Integral \( S(3) \)

\[
\int_{r=r_1}^{r=r_j} \int_{z=A..+B..r}^{z=r..} r^2 \, dz \, dr = \int_{r=r_1}^{r=r_j} (A_{ij} r^2 + B_{ij} r^3) \, dr = \left[ \frac{A_{ij} r^3}{3} + \frac{B_{ij} r^4}{4} \right]_{r=r_1}^{r=r_j}
\]

\[
= \frac{A_{ij}}{3}(r_j^3-r_1^3) + \frac{B_{ij}}{4}(r_j^4-r_1^4)
\]

\[
= \frac{A_{ij}}{3}(r_j^3-r_1^3)(r_j^2+r_1 r_j+r_1^2) + \frac{B_{ij}}{4}(r_j^4-r_1^4)(r_j^2+r_1^2)
\]

\[
= \frac{(r_j^3z_j^3-r_1^3z_1^3)}{3(r_j^3-r_1^3)} (r_j^2+r_1 r_j+r_1^2) + \frac{(z_j^3-z_1^3)}{4(r_j^2-r_1^2)} (r_j^2+r_1^2)(r_j^2+r_1^2)
\]

\[
= \frac{(r_j^3z_j^3-r_1^3z_1^3)}{3} (r_j^2+r_1 r_j+r_1^2) + \frac{(z_j^3-z_1^3)}{4} (r_j^2+r_1^2)(r_j^2+r_1^2)
\]

\[
= \frac{(-r_1^3z_1 - r_1^2z_j + r_1^2z_j + r_1z_1^2 + r_1z_j^2 + r_1^3z_j)/3}
\]

\[
+ \frac{(-r_1^3z_1 + r_1^3z_j + r_1^2z_j + r_1z_1^2 + r_1z_j^2 + r_1^3z_j)/4}
\]

\[
= \frac{(-4r_1^3z_1 - 4r_1^2z_j + 4r_1^2z_j + 4r_1z_1^2 + 4r_1z_j^2 + 4r_1^3z_1 + 4r_1^3z_j)/1}
\]

\[
= \frac{(-3r_1^3z_1 + 3r_1^3z_j + 3r_1^2z_j + 3r_1z_1^2 - 3r_1z_j^2 + 3r_1^3z_j - 3r_1^3z_j + 3r_1^3z_j)}{1}
\]

\[
= \frac{(-3r_1^3z_1 - r_1^3z_j - r_1^2z_j + r_1^2z_j + r_1z_1^2 + r_1z_j^2 + r_1^3z_1 + 3r_1^3z_j)/12}
\]

\[
\int_j^k r^2 \, dz \, dr = \frac{(-3r_j^3z_j - r_j^2z_j - r_jz_j^2 + r_jz_j^2 + r_jz_j^2 + r_jz_j^2 + r_jz_j^2 + 3r_j^3z_j)/12}{12}
\]

\[
\int_j^i r^2 \, dz \, dr = \frac{(-3r_k^3z_k - r_k^2z_k - r_kz_k^2 + r_kz_k^2 + r_kz_k^2 + r_kz_k^2 + r_kz_k^2 + 3r_k^3z_k)/12}{12}
\]

135
Integral $S(3)$ continued

\[
\text{sum} = \left\{ r_i^3(z_k - z_j) + r_j^3(z_i - z_k) + r_k^3(z_j - z_i) - r_j r_k^2(z_k - z_j) - r_k r_i^2(z_i - z_k) - r_i r_j^2(z_j - z_i) \right\}/12
\]

\[
- \left\{ r_j r_k^2(z_k - z_j) - r_k r_i^2(z_i - z_k) - r_i r_j^2(z_j - z_i) \right\}/12
\]

\[
= \left\{ (r_i^3 - r_j r_k^2)(z_k - z_j) + (r_j^3 - r_k r_i^2)(z_i - z_k) + (r_k^3 - r_i r_j^2)(z_j - z_i) \right\}/12
\]

\[
\times (z_j - z_i)/12
\]

\[
= \left\{ (r_i^3 - r_j r_k^2)(r_j + r_k)(z_k - z_j) + (r_j^3 - r_k r_i^2)(r_k + r_i)(z_i - z_k) + (r_k^3 - r_i r_j^2)(r_i + r_j)(z_j - z_i) \right\}/12
\]

\[
= \left\{ (a^3 - bc(b + c))(z - y) + (b^3 - ca(c + a))(x - z) + (c - ab(a + b))(x - y) \right\}/12
\]
Integral $S(4)$

$$\int_{r=r_i}^{r=r_j} \int_{z=0}^{z=A} r^3 \, dz \, dr = \int_{r=r_i}^{r=r_j} (A_{ij} r^3 + B_{ij} r^4) \, dr$$

$$= \left[ \frac{A_{ij} r^4}{4} + \frac{B_{ij} r^5}{5} \right]_{r_i}^{r_j}$$

$$= \frac{A_{ij}}{4} (r_j^4 - r_i^4) + \frac{B_{ij}}{5} (r_j^5 - r_i^5)$$

$$= \frac{A_{ij}}{4} (r_j - r_i) (r_j + r_i) (r_j^2 + r_i^2) + \frac{B_{ij}}{5} (r_j - r_i) (r_j^4 + r_i^4 + r_j^2 r_i^2 + r_j^3 r_i + r_j r_i^3 + r_j^3 r_i + r_j r_i^3 + r_i^4)$$

$$= \frac{1}{4} (r_j z_j - r_i z_i) (r_j + r_i) (r_j^2 + r_i^2) + \frac{1}{5} (z_j - z_i) (r_j^4 + r_i^4 + r_j^2 r_i^2 + r_j^3 r_i + r_j r_i^3 + r_j^3 r_i + r_j r_i^3 + r_i^4)$$

$$= (-r_i z_j - r_j z_i) + r_j^2 r_i z_j - r_i^2 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^2 r_i z_j + r_i^2 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i) / 4$$

$$= (-r_i z_j - r_j z_i) + r_j^2 r_i z_j - r_i^2 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^2 r_i z_j + r_i^2 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i) / 5$$

$$= (-r_i z_j - r_j z_i) + r_j^2 r_i z_j - r_i^2 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^2 r_i z_j + r_i^2 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i + r_j^3 r_i z_i + r_i^3 r_j z_i) / 20$$

$$|k|^2 = (-r_i z_k - r_k z_i) + r_j^2 r_k z_j + r_i^2 r_k z_i + r_j^3 r_i z_i + r_i^3 r_j z_j + r_j^3 r_i z_i + r_i^3 r_j z_j + r_j^2 r_i z_j + r_i^2 r_j z_i) / 20$$

$$|j|^2 = (-r_i z_k - r_k z_i) + r_j^2 r_k z_j + r_i^2 r_k z_i + r_j^3 r_i z_i + r_i^3 r_j z_j + r_j^3 r_i z_i + r_i^3 r_j z_j + r_j^2 r_i z_j + r_i^2 r_j z_i) / 20$$

$$|i|^2 = (-r_i z_k - r_k z_i) + r_j^2 r_k z_j + r_i^2 r_k z_i + r_j^3 r_i z_i + r_i^3 r_j z_j + r_j^3 r_i z_i + r_i^3 r_j z_j + r_j^2 r_i z_j + r_i^2 r_j z_i) / 20$$
Integral S(4) continued

\[ \text{Sum} = \left\{ \begin{array}{l}
(r_i^4(z_k - z_j) + r_j^4(z_i - z_k) + r_k^4(z_j - z_i) \\
- \frac{3^3}{4}r_k(z_k - z_j) - \frac{3^3}{4}r_i(z_i - z_k) - \frac{3^3}{4}r_j(z_j - z_i) \\
- \frac{2^2}{4}r_k^2(z_k - z_j) - \frac{2^2}{4}r_i^2(z_i - z_k) - \frac{2^2}{4}r_j^2(z_j - z_i) \\
- \frac{2^2}{4}r_k^2(z_k - z_j) - \frac{2^2}{4}r_i^2(z_i - z_k) - \frac{2^2}{4}r_j^2(z_j - z_i) \\
- \frac{2^2}{4}r_k^2(z_k - z_j) - \frac{2^2}{4}r_i^2(z_i - z_k) - \frac{2^2}{4}r_j^2(z_j - z_i) \\
\end{array} \right\}/20
\]

\[ = \left\{ \begin{array}{l}
(r_i^4 - r_j^3r_k - r_j^2r_k^2 - r_j^2r_k^3)(z_k - z_j) \\
+ (r_j^4 - r_k^3r_i - r_k^2r_i^2 - r_k^2r_i^3)(z_i - z_k) \\
+ (r_k^4 - r_i^3r_j - r_i^2r_j^2 - r_i^2r_j^3)(z_j - z_i) \\
\end{array} \right\}/20
\]

\[ = \left\{ \begin{array}{l}
(r_i^4 - r_j^3r_k + r_j^2r_k^2 + r_j^2r_k^3)(z_k - z_j) \\
+ (r_j^4 - r_k^3r_i + r_k^2r_i^2 + r_k^2r_i^3)(z_i - z_k) \\
+ (r_k^4 - r_i^3r_j + r_i^2r_j^2 + r_i^2r_j^3)(z_j - z_i) \\
\end{array} \right\}/20
\]

\[ = \left\{ \begin{array}{l}
(a^4 - bc(b^2 + bc + c^2))(z - y) \\
+ (b^4 - ca(c^2 + ca + a^2))(x - z) \\
+ (c^4 - ab(a^2 + ab + b^2))(y - x) \\
\end{array} \right\}/20
\]
Integral $S(5)$

$$\int_{z=z_i}^{z=z_j} \frac{rz^2 drdz}{r=0} = \frac{1}{2} \int_{z=z_i}^{z=z_j} r^2 (C_{ij} + D_{ij} z)^2 \, dz$$

$$= \frac{1}{2} \int_{z=z_i}^{z=z_j} z^2 (C_{ij}^2 z + D_{ij}^2 z^2) \, dz$$

$$= \frac{1}{2} \int_{z=z_i}^{z=z_j} (C_{ij} z^2 + 2 C_{ij} D_{ij} z^2 + D_{ij}^2 z^2) \, dz$$

$$= \frac{1}{2} \left[ C_{ij}^2 \frac{z^3}{3} + \frac{C_{ij}}{2} D_{ij} z^4 + D_{ij}^2 \frac{z^5}{5} \right]_{z_i}^{z_j}$$

$$= \left[ 10 C_{ij}^2 z^3 + 15 C_{ij} D_{ij} z^4 + 6 D_{ij}^2 z^5 \right]_{z_i}^{z_j} / 60$$

$$= \left[ 10 C_{ij}^2 (z_j^3 - z_i^3) + 15 C_{ij} D_{ij} (z_j^4 - z_i^4) + 6 D_{ij}^2 (z_j^5 - z_i^5) \right] / 60$$

$$= \left\{ 10 \frac{(r_j z_i - r_i z_i)}{(z_j - z_i)^2} (z_j - z_i)^2 (z_j + z_i) \right\}$$

$$+ 15 \frac{(r_j z_i - r_i z_i)}{(z_j - z_i)^2} (r_j - r_i) (z_j - z_i) (z_j + z_i) (z_j^2 + z_i^2)$$

$$+ 6 \frac{(r_j - r_i)}{(z_j - z_i)^2} (z_j - z_i)^2 (z_j^4 + z_j^3 z_i + z_j^2 z_i^2 + z_j z_i^3 + z_i^4) / 60$$

$$= \left\{ 10 (r_j z_i - r_i z_i)^2 (z_j^2 + z_j z_i + z_i^2) \right\}$$

$$+ 15 (r_j z_i - r_i z_i) (r_j - r_i) (z_j + z_i) (z_j^2 + z_i^2)$$

$$+ 6 (r_j - r_i)^2 (z_j^4 + z_j^3 z_i + z_j^2 z_i^2 + z_j z_i^3 + z_i^4) / 60 (z_j - z_i) \right\}$$
Integral S(5) continued

\[ \int \left\{ 10 \left( r_i^2 z_j^2 - 2r_i r_j z_i z_j + r_j^2 z_i^2 \right) \left( z_j^2 + z_i z_i + z_i^2 \right) \right. \]

\[ + 15 \left( -r_j^2 z_i + r_i r_j z_i + r_i r_j z_j - r_i^2 z_j \right) \left( z_j^3 + z_j^2 z_i + z_j z_i^2 + z_i^3 \right) \]

\[ + 6 \left( r_j^2 - 2r_j r_i + r_i^2 \right) \left( z_j^4 + z_j^3 z_i + z_j^2 z_i^2 + z_j z_i^3 + z_i^4 \right) \right\} / \left\{ 60 \left( z_j - z_i \right) \right\} \]
\[
\sum_{j} \left\{ \frac{10(z_j^2 - r_j z_j - x_j^2 + x_j z_j - z_j^2 - 3x_j^2 + 3z_j^2 - 6x_j^2)}{60(z_j - z_i)} \right\} = \frac{10j^2}{60} \left( \frac{z_j^2 - r_j z_j - x_j^2 + x_j z_j - z_j^2 - 3x_j^2 + 3z_j^2 - 6x_j^2)}{60(z_j - z_i)} \right)
\]

Similarly

\[
\sum_{j} \left\{ \frac{10(z_j^2 - r_j z_j - x_j^2 + x_j z_j - z_j^2 - 3x_j^2 + 3z_j^2 - 6x_j^2)}{60(z_j - z_i)} \right\} = \frac{10j^2}{60} \left( \frac{z_j^2 - r_j z_j - x_j^2 + x_j z_j - z_j^2 - 3x_j^2 + 3z_j^2 - 6x_j^2)}{60(z_j - z_i)} \right)
\]

Similarly

\[
\sum_{j} \left\{ \frac{10(z_j^2 - r_j z_j - x_j^2 + x_j z_j - z_j^2 - 3x_j^2 + 3z_j^2 - 6x_j^2)}{60(z_j - z_i)} \right\} = \frac{10j^2}{60} \left( \frac{z_j^2 - r_j z_j - x_j^2 + x_j z_j - z_j^2 - 3x_j^2 + 3z_j^2 - 6x_j^2)}{60(z_j - z_i)} \right)
\]
Summing

\[ r_i^2(z_j^3+2z_j^2z_i+3z_j^2z_i^2+3z_j^3z_i^2-3z_i^3z_i^2z_j^3) + r_k^2 r_j^2 (3z_k^3z_j^2z_j^2_1-3z_j^2z_k^2z_j^3) \]

\[ +r_k^2 (z_j^2+2z_j^2z_k+3z_j^2z_k^2-3z_k^2z_j^2z_j^2) + r_j^2 (3z_j^3z_kz_j^2z_j^2_1-3z_j^2z_k^2z_j^3) \]

\[ +r_k^2 (z_j^3+2z_j^2z_j+3z_j^2z_j^2-3z_j^2z_j^2z_j^2_1) + r_j^2 (3z_j^3z_j^2z_j^2_1-3z_j^2z_j^2z_j^3) \] / 60

\[ = \{r_i^2 (z_j^3-2z_j^2z_k + 3z_j^2z_j^2) + 3z_j^2(z_j-z_i) \} + r_k^2 \{r_j^2 [3(z_k^3-z_j^3)z^3_j_1 + z_kz_j] \} \]

\[ + r_j^2 [z_k^3-2z_k^2z_j + 3z_k^2(z_j-z_i)] \] / 60

\[ = (z_j-z_i) \{r_i^2 r_j^2 [3(z^2_j + z^2_i z^2_j) + z^2_j] - r_k^2 [z^2_j + z^2_i z^2_j + 2z_k^2(z^2_j z^2_i) + 3z^2_i] \} \]

\[ + (z_k-z_j) \{r_j r_k^2 [3(z^2_j + z^2_i z^2_j) + z^2_j] - r_i^2 [z^2_j + z^2_i z^2_j + 2z_k^2(z^2_j z^2_i) + 3z^2_i] \} \]

\[ + (z_i-z_k) \{r_i^2 r_k^2 [3(z^2_i + z^2_j z^2_i) + z^2_i] - r_j^2 [z^2_i + z^2_i z^2_i + 2z_k^2(z^2_i z^2_j) + 3z^2_j] \} / 60 \]

\[ = (y-x) \{ab \ [3(x^2+xy+y^2) + xy] - c^2 [x^2+xy+y^2+2z(x+y) + 3z^2] \} \]

\[ + (z-y) \{bc \ [3(y^2+yz+z^2) + yz] - a^2 [y^2+yz+z^2+2x(y+z) + 3x^2] \} \]

\[ + (x-z) \{ca \ [3(z^2+zx+x^2) + zx] - b^2 [z^2+zx+x^2+2y(z+x) + 3y^2] \} / 60 \]
Integral $S(6)$

$$\int_{r_1}^{r_j} \int_{z=0}^{z=A_{ij}+B_{ij}r} r^2 zdzdr = \frac{1}{2} \int_{r_1}^{r_j} r^2(A_{ij} + B_{ij}r)^2 dr$$

$$= \frac{1}{2} \int_{r_1}^{r_j} r^2(A_{ij}^2 + 2A_{ij}B_{ij}r + B_{ij}^2 r^2) dr$$

$$= \frac{1}{2} \left[ \frac{A_{ij}^2 r^3}{3} + \frac{A_{ij}B_{ij} r^4}{2} + \frac{B_{ij}^2 r^5}{5} \right]_1^{r_j}$$

$$= \left[ 10A_{ij}^2 r^3 + 15 A_{ij} B_{ij} r^4 + 6B_{ij}^2 r^5 \right]_1^{r_j} / 60$$

$$= \{10A_{ij}^2(r_j^3-r_1^3) + 15A_{ij} B_{ij}(r_j^4-r_1^4) + 6B_{ij}^2(r_j^5-r_1^5)\} / 60$$

$$= \left\{ \frac{(r_jz_j-r_1z_1)^2}{(r_j-r_1)^2} (r_j^2+r_jr_1+r_1^2) \right\}$$

$$+ 15 \left\{ \frac{(r_jz_j-r_1z_1)^2}{(r_j-r_1)^2} (z_j-z_1)(r_j^2+r_1^2)(r_jr_1+r_1r_j) \right\}$$

$$+ 6 \left\{ \frac{(z_j-z_1)^2}{(r_j-r_1)^2} (r_j^2-r_1^2)(r_j^3+r_1^3+r_jr_1^2+r_1r_j^2) \right\} / 60(r_j-r_1)$$

$$= \{10(r_jz_j-r_1z_1)^2(r_1^3+r_1r_j+r_j^3) \}

$$+ 15(r_jz_j-r_1z_1)(z_j-z_1)(r_1+r_j)(r_1^2+r_j^2) \}

$$+ 6(z_j-z_1)^2(r_1^4+3r_1^2r_j^2+r_j^4+r_1r_j^3+r_jr_1^3) / 60(r_j-r_1) \}$$

$$= \{10(r_jz_j-r_1z_1)^2(r_1^2+r_1r_j+r_j^2) \}

$$+ 15(z_j-z_1)(r_1z_j-r_1z_1)(r_1^3+r_1r_j+r_j^3) \}

$$+ 6(z_j-z_1)^2(r_1^3+r_1^2r_j^2+r_1r_j^3+r_jr_1^3) / 60(r_j-r_1) \}$$
Summing

\[
\begin{align*}
&\{ z^2_1(r^3_j + 2r^2_ir^2_j + 3r^2_ir^3_j - 3r^2_k r^2_j - 2r^2_k r^2_i - r^3_k) + z^2_1 z_j(3r^3_j + r^2_i r^2_j - r^2_j r^2_i - 3r^3_i) \\
&+ z^2_j(r^3_k + 2r^2_j r^2_k + 3r^2_j r^2_k - r^3_j r^2_i - 2r^3_j r^2_i - r^3_k) + z^2_j z_k(3r^3_k + r^2_j r^2_k - r^2_j r^2_k - 3r^3_j) \\
&+ z^2_k(r^3_i + 2r^2_k r^2_i + 3r^2_k r^2_i - r^3_k r^2_i - 2r^3_k r^2_i - r^3_i) + z^2_k z_i(3r^3_i + r^2_k r^2_i - r^2_i r^2_k - 3r^3_k)\}/60
\end{align*}
\]

\[
= \{ z^2_1(r^3_j - r^3_k) + 2r^2_j(r^3_j - r^3_k) + 3r^2_j(r^3_j - r^3_k) \} + z^2_1 z_j(3(r^3_j - r^3_k) + r^2_i(r^3_j - r^3_k)) \\
+ z^2_j(r^3_k - r^3_i) + 2r^2_j(r^3_k - r^3_i) + 3r^2_j(r^3_k - r^3_i) \} + z^2_j z_k(3(r^3_k - r^3_i) + r^2_j(r^3_k - r^3_i)) \\
+ z^2_k(r^3_i - r^3_j) + 2r^2_k(r^3_i - r^3_j) + 3r^2_k(r^3_i - r^3_j) \} + z^2_k z_i(3(r^3_i - r^3_j) + r^2_k(r^3_i - r^3_j))\}/60
\]

\[
= \{ (r^3_j - r^3_i)\{ z^2_1 z_j(3(r^3_j + r^3_i) - 3r^3_j) \} - z^2_k(r^3_j + r^3_i) - 2r^2_k(r^3_j + r^3_i) + 3r^2_k \} \\
+ (r^3_k - r^3_j)\{ z^2_j z_k(3(r^3_k + r^3_j) - 3r^3_j) \} - z^2_i(r^3_k + r^3_j) - 2r^2_i(r^3_k + r^3_j) + 3r^2_i \} \\
+ (r^3_i - r^3_k)\{ z^2_k z_i(3(r^3_i + r^3_k) - 3r^3_k) \} - z^2_j(r^3_i + r^3_k) - 2r^2_j(r^3_i + r^3_k) + 3r^2_j \}\}/60
\]

\[
= \{ (b-a)\{ xy[3(b^2 + a^2) + ab] - z^2[3(b^2 + a^2) + ab] \} + (c-b)\{ yz[3(c^2 + b^2) + bc] - x^2(c^2 + b^2) + 2a(b+c) + 3a^2] \} \\
+ (a-c)\{ zx[3(a^2 + c^2) + ca] - y^2[a^2 + ca + c^2 + 2b(c+a) + 3b^2] \}\}/60
\]
Integral $S(7)$

\[
\begin{align*}
\int_{r=r_i}^{r=r_j} \int_{z=0}^{z=A_i + B_i r} dz \, dr &= \int_{r=r_i}^{r=r_j} (A_{ij} + B_{ij} r^2) \, dr = \left[ A_{ij} r + B_{ij} \frac{r^3}{3} \right]_{r_i}^{r_j} \\
&= A_{ij} (r_j - r_i) + B_{ij} \frac{(r_j^2 - r_i^2)}{2} \\
&= r_j z_i - r_i z_j + \frac{(z_j - z_i)(r_i + r_j)}{2} \\
&= \frac{(2r_j z_i - 2r_i z_j + r_j z_j + r_i z_j - r_j z_i - r_i z_i)}{2} \\
&= \frac{(r_j z_i - r_i z_i + r_j z_j - r_i z_j)}{2}.
\end{align*}
\]

Similarly

\[
\int_{r=r_k}^{r=r_j} \int_{z=0}^{z= A_j + B_j r} dz \, dr = \frac{(r_k z_j - r_j z_j + r_k z_k - r_j z_k)}{2}.
\]

and

\[
\int_{r=r_i}^{r=r_k} \int_{z=0}^{z=A_i + B_i r} dz \, dr = \frac{(r_i z_k - r_k z_k + r_i z_i - r_k z_i)}{2}.
\]

so that

\[
\Delta f \, dz \, dr = \frac{(r_j z_i - r_i z_i + r_j z_j - r_i z_j)}{2} + \frac{(r_k z_j - r_j z_j + r_k z_k - r_j z_k)}{2} + \frac{(r_i z_k - r_k z_k + r_i z_i - r_k z_i)}{2}.
\]

\[
= \frac{(r_j z_i + r_k z_j + r_i z_k)}{2} - \frac{(r_i z_j + r_j z_k + r_k z_i)}{2}.
\]

\[
= \frac{(r_i (z_k - z_j) + r_j (z_i - z_k) + r_k (z_j - z_i))}{2}.
\]

If $a = r_i$, $b = r_j$, $c = r_k$, $z_i = x$, $z_j = y$, $z_k = z$

then

\[
\Delta f \, dz \, dr = \frac{(a(z-y) + b(x-z) + c(y-x))}{2}.
\]

136
Integral $S(8)$

\[
\int z = z_j \int r = C_i + D_{ij} z \int z = z_j \\
\int z = z_i \int r = 0 \quad z \, dr \, dz = \int z (C_{ij} + D_{ij} z) \, dz \\
\frac{C_{ij} z^2}{2} + D_{ij} \frac{z^3}{3} \int z_i \\
= \left( r_i z_j - r_j z_i \right) \frac{z(j-z_i)}{2(z_j - z_i)} \left( z_j + z_i \right) + \frac{r_i(r_j - r_i)}{3(z_j - z_i)} \left( z_j - z_i \right) \left( z_j^2 + z_i z_j + z_j^2 \right) \\
= \frac{1}{2} \left( r_i z_j + r_i z_i z_j - r_j z_i z_j - r_j^2 \right) + \frac{1}{3} \left( r_i z_j + r_i z_i z_j + r_j^2 - r_j z_i z_j - r_j^2 \right) \\
= (3r_i z_i z_j + 3r_i z_j^2 - 3r_j z_i^2 - 3r_j z_i z_j) \\
- 2r_i z_j z_i z_j - 2r_i z_i z_j^2 + 2r_j z_i z_j + 2r_i z_j^2 - 2r_i z_i^2) / 6 \\
= (r_i z_i z_j + r_i z_i^2 - r_i z_i z_j + 2r_i z_i z_j - 2r_i z_j^2) / 6 \\
\sum_{j}^{k} = (r_i z_j z_k + r_i z_k^2 - r_k z_j z_k + 2r_k z_k^2 - 2r_j z_j^2) / 6 \\
\sum_{i}^{k} = (r_k z_k z_i + r_k z_i^2 - r_i z_k z_i + 2r_i z_i^2 - 2r_k z_k^2) / 6 \\
\Delta = \left( r_i z_i z_j + z_i z_j \right) - r_j (z_i z_j + z_i z_j) \\
\quad + r_i (z_j z_k + z_i z_k) \\
\quad + r_k (z_i z_j + z_k z_i) \\
/ 6 \\
= \left( r_i z_j^2 - r_i z_i^2 + z_i z_j^2 - z_i z_j \right) \\
+ r_j \left( -z_i^2 + z_k^2 - z_i z_j + z_j z_k \right) \\
+ r_k \left( z_i^2 + z_i z_k - z_j z_k - z_j^2 \right) / 6 \\
= \left( (r_i - r_j) (z_i^2 - z_j z_i) + (r_k - r_j) (z_i z_j + z_i z_k) + (r_i - r_k) (z_j^2 - z_i z_k) \right) / 6 \\
\text{NB This needs to be multiplied by - sign for clockwise $ijk$}

let $a = r_i$, $b = r_j$, $c = r_k$ and $x = z_i$, $y = z_j$, $z = z_k$ \\
$S_2 = \{(a-b)(z^2 - xy) + (b-c)(x^2 - yz) + (c-a)(y^2 - xz)/6}$
\[
\int_{z=z_i}^{z=z_j} \int_{r=0}^{r=C_i} z^2 \, dz \, dr = \int_{z=z_i}^{z=z_j} \left( C_{ij} \frac{z^3}{3} + D_{ij} \frac{z^4}{4} \right) \, dz
\]
\[
= \frac{C_{ij}}{3} (z_j^3 - z_i^3) + D_{ij} (z_j^4 - z_i^4)
\]
\[
= \frac{(r_i z_i - r_j z_j)}{3(z_j - z_i)} (z_j^3 - z_i^3) + \frac{(r_j - r_i)}{(z_j - z_i)} (z_j^4 - z_i^4)
\]
\[
= (r_i z_i - r_j z_j) (z_j^2 + z_i z_j + z_i^2)/3 - (r_i - r_j) (z_i + z_j) (z_i^2 + z_j^2)/4
\]
\[
= (r_i z_i^3 + r_j z_j^3 + r_i z_j z_i^2 + r_j z_i z_j^2 - r_j z_i z_j z_i^2 - r_j z_i z_j^2)/3
\]
\[
- (r_i z_i^3 + r_j z_j^3 + r_i z_j z_i^2 + r_j z_i z_j^2 - r_j z_i z_j z_i^2 - r_j z_i z_j^2)/3
\]
\[
= (4r_i z_j^3 + 4r_j z_i z_j^2 + 4r_i z_i z_i^2 - 4r_j z_i z_j^2 - 4r_j z_i^2 z_j - 4r_j z_j^3
\]
\[
- 3r_i z_j^3 - 3r_i z_i z_j^2 - 3r_j z_i z_j^2 + 3r_j z_i z_i^2 + 3r_j z_i z_j^2 + 3r_j z_i^2 z_j + 3r_i z_i^3 + 3r_j z_i^3 - 3r_i z_i^3)/12
\]
\[
= (-3r_i z_i^3 + r_i z_i z_j^2 + r_i z_i z_j^2 + r_i z_j^3 - r_i z_i z_i^2 - r_i z_i z_j z_i^2 - r_i z_i z_j^2 + 3r_i z_j^3)/12
\]
\[
= (-3r_i z_i^3 + 3r_j z_i^3 + (r_i - r_j)(z_i^2 z_j + z_i z_j^2) + r_i z_j - r_j z_i)/12
\]
\[
\sum_{k=1}^{k} (-3r_j z_j^3 + 3r_j z_k + (r_j - r_k)(z_j z_k + z_j z_k^2) + r_j z_k^3 - r_k z_j^3)/12
\]
\[
\sum_{k=1}^{k} (-3r_k z_k^3 + 3r_i z_k^3 + (r_k - r_i)(z_k^2 z_i + z_k z_i^2) + r_k z_i^3 - r_i z_k^3)/12
\]
\[
\sum = \{ (r_j - r_i)(z_k^3 - z_j^3) + (r_k - r_j)(z_i^3 - z_j z_k - z_j^2 z_k)
\]
\[
+ (r_i - r_k)(z_i^3 - z_j z_k + z_j^2 z_k) \}/12
\]
\[
= \{ (r_j - r_i)(z_k - z_i z_j (z_i + z_j)) + (r_k - r_j)(z_i - z_j z_k (z_j + z_k))
\]
\[
+ (r_i - r_k)(z_i - z_k z_j (z_j + z_k)) \}/12
\]
\[
= \{ (b - a)(z^3 - xy (x+y)) + (c - b)(x^3 - yz (y+z))
\]
\[
+ (a - c)(y^3 - zx (z+x)) \}/12
\]
\[ \int_{z_i}^{z_j} \int_{r=0}^{r=C_{ij}+D_{ij}z} z^3 \, dz \, dr = \int_{z_i}^{z_j} \int_{r=0}^{r=C_{ij}+D_{ij}z} r^3 \, C_{ij}^{z_j} \, dz \]

\[ = \int_{z_i}^{z_j} \int_{z_i}^{z_j} z^3 \, (C_{ij}^{z_j} + D_{ij}^{z_j}z) \, dz \]

\[ = \left[ \frac{C_{ij}^{z_j}}{4} z^4 + \frac{D_{ij}^{z_j}}{5} z^5 \right]_{z_i}^{z_j} \]

\[ = \frac{C_{ij}^{z_j}}{4} z^4 + \frac{D_{ij}^{z_j}}{5} z^5 \]

\[ = \frac{(r_i z_j^4 - r_j z_i^4)}{4} \left( (z_j - z_i)(z_j + z_i)(z_j^2 + z_i^2) \right) + \frac{(r_j - r_i)}{5} \left( (z_j - z_i)(z_j^2 + z_i^2 + z_j^3 + z_i^3 + z_j^4 + z_i^4) \right) \]

\[ = \frac{r_i z_j^4 - r_j z_i^4}{4} \frac{(z_j^3 + z_j^2 z_i + z_i^2 + z_j z_i^2 + z_j^2 z_i + z_i^2 + z_j z_i^3 + z_i^3)}{z_j - z_i} \]

\[ + \frac{(r_j - r_i)}{5} \left( (z_j^4 + z_j^3 z_i + z_i^2 + z_j^2 z_i^2 + z_j^3 z_i^2 + z_i^3 + z_j^4 + z_i^4) \right) \]

\[ = \frac{(5 r_i z_j^4 - 5 r_j z_i^4)}{4} \frac{z_j^3 z_i + 5 r_i z_j^2 z_i + 5 r_j z_i^2 z_j + 5 r_i z_j^2 z_i + 5 r_j z_i^2 z_j + 5 r_i z_j^3 z_i + 5 r_j z_i^3 z_j - 5 r_j z_i^4}{z_j - z_i} \]

\[ + \frac{4 r_j z_j^4 - 4 r_i z_i^4}{5} \frac{z_j^3 z_i - 4 r_i z_j^3 z_i + 4 r_j z_i^3 z_j + 4 r_j z_i^3 z_i + 4 r_i z_j^4}{z_j - z_i} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]

\[ = \frac{(4 r_j z_j^4 + r_i z_i^4 - r_j z_i^4 - r_j z_j^3 z_i + r_i z_j z_i^3 + r_j z_j^2 z_i + r_i z_i^2 z_j - r_j z_i z_j^3 + r_i z_j^3 z_i + r_j z_i^3 z_j - r_j z_i^4 - r_j z_i^4)}{20} \]
Sum = \{ z_j^u (r_i - r_k) + z_k^u (r_j - r_i) + z_i^u (r_k - r_j) \\
- z_i^3 z_k (r_i - r_k) - z_j^3 z_i (r_j - r_i) - z_k^3 z_j (r_k - r_j) \\
- z_i^2 z_k^2 (r_i - r_k) - z_j^2 z_i^2 (r_j - r_i) - z_k^2 z_j^2 (r_k - r_j) \\
- z_i z_k^3 (r_i - r_k) - z_j z_i^3 (r_j - r_i) - z_k z_j^3 (r_k - r_j) \}/20 \\
= \{(z_j^u - z_k^3 z_j - z_i z_j^2 - z_j z_i^3)(r_k - r_j) \\
+ (z_j^u - z_i z_k^2 - z_i z_i^2 - z_i z_k^3)(r_i - r_k) \\
+ (z_k^u - z_j z_i^2 - z_k z_i z_j^2 - z_j z_i z_k^2)(r_j - r_i) \}/20 \\
= \{((z_j^u - z_j z_k (z_j^2 + z_j z_k + z_k^2))(r_k - r_j) \\
+ (z_j^u - z_k z_i (z_k^2 + z_k z_i + z_i^2))(r_i - r_k) \\
+ (z_k^u - z_i z_j (z_i^2 + z_i z_j + z_j^2))(r_j - r_i) \}/20 \\
= \{(x^u - yz (y^2 + yz + z^2))(c - b) \\
+ (y^u - zx (z^2 + zx + x^2))(a - c) \\
+ (z^u - xy (x^2 + xy + y^2))(b - a) \}/20
Integral S(11)
\[
\int_{r=r_i}^{r=r_j} \int_{z=A_i}^{z=B_i} \frac{1}{r} \, dz \, dr = \int_{r=r_i}^{r=r_j} \left[ \frac{z}{r} \right]_{z=A_i}^{z=B_i} \, dr
\]
\[
= \int_{r=r_i}^{r=r_j} \frac{1}{r} (A_i \, z - B_i \, r) \, dr = \int_{r=r_i}^{r=r_j} \left( \frac{A_i}{r} \, z - B_i \, \frac{r}{r} \right) \, dr
\]
\[
= A_{ij} \left[ \log r \right]_{r_i}^{r_j} + B_{ij} \left[ r \right]_{r_i}^{r_j}
\]
\[
= A_{ij} (\log r_j - \log r_i) + B_{ij} (r_j - r_i)
\]
\[
= A_{ij} (\log r_j - \log r_i) + \frac{z_j - z_i}{r_j - r_i} (r_j - r_i)
\]
\[
= A_{ij} (\log r_j - \log r_i) + z_j - z_i
\]
\[
= A_{ij} \log r_j - A_{ij} \log r_i + z_j - z_i
\]

Similarly
\[
\int_{r=r_j}^{r=r_k} \int_{z=A_j}^{z=B_j} \frac{1}{r} \, dz \, dr = A_{jk} \log r_k - A_{jk} \log r_j + z_k - z_j
\]
and
\[
\int_{r=r_k}^{r=r_i} \int_{z=A_k}^{z=B_k} \frac{1}{r} \, dz \, dr = A_{ki} \log r_i - A_{ki} \log r_k + z_i - z_k
\]
Adding the three
\[
(A_{ki} - A_{ij}) \log r_i + (A_{ij} - A_{jk}) \log r_j + (A_{jk} - A_{ki}) \log r_k
\]

where \( A_{ij} = \frac{r_i \, z_j - r_j \, z_i}{r_j - r_i} \), \( A_{jk} = \frac{r_k \, z_j - r_j \, z_k}{r_k - r_j} \), and \( A_{ki} = \frac{r_i \, z_k - r_k \, z_i}{r_i - r_k} \).
Integral S(12)

\[
\int_{r=r_i}^{r=r_f} \int_{z=0}^{z=A_{ij}+B_{ij}r} \frac{r}{r} \ dz \ dr = \frac{1}{2} \int_{r=r_i}^{r=r_f} \int_{z=0}^{z=A_{ij}+B_{ij}r} \frac{r}{r} \ dz \ dr
\]

= \frac{1}{2} \int_{r=r_i}^{r=r_f} \left( \frac{A_{ij}}{r} + 2A_{ij}B_{ij} + B_{ij}^2 r \right) \ dr

= \frac{A_{ij}^2}{2} \left[ \log r \right]_{r_i}^{r_f} + A_{ij}B_{ij} \left[ r \right]_{r_i}^{r_f} + \frac{B_{ij}^2}{4} \left[ r^2 \right]_{r_i}^{r_f}

= \frac{A_{ij}}{2} (\log r - \log r_i) + A_{ij}B_{ij}(r_j - r_i) + \frac{B_{ij}^2}{4}(r_j^2 - r_i^2)

r=r_k \ z=A_{jk}+B_{jk}r

\int_{r=r_j}^{r=r_k} \int_{z=0}^{z=A_{jk}+B_{jk}r} \frac{r}{r} \ dz \ dr

= \frac{A_{jk}^2}{2} \log r_k - \frac{A_{jk}^2}{2} \log r_j + A_{jk}B_{jk}(r_k - r_j) + \frac{B_{jk}^2}{4}(r_k^2 - r_j^2)

r=r_i \ z=A_{ki}+B_{ki}r

\int_{r=r_i}^{r=r_k} \int_{z=0}^{z=A_{ki}+B_{ki}r} \frac{r}{r} \ dz \ dr

= \frac{A_{ki}^2}{2} \log r_i - \frac{A_{ki}^2}{2} \log r_k + A_{ki}B_{ki}(r_i - r_k) + \frac{B_{ki}^2}{4}(r_i^2 - r_k^2)

Summing

S(8) = A_{ij}B_{ij}(r_j - r_i) + A_{jk}B_{jk}(r_k - r_j) + A_{ki}B_{ki}(r_i - r_k)

+ \frac{B_{ij}^2}{4}(r_j - r_i)(r_j + r_i) + \frac{B_{jk}^2}{4}(r_k - r_j)(r_k + r_j) + \frac{B_{ki}^2}{4}(r_i - r_k)(r_i + r_k)

+ \left( \frac{A_{ki}}{2} - \frac{A_{ij}}{2} \right) \log r_i + \left( \frac{A_{ij}}{2} - \frac{A_{jk}}{2} \right) \log r_j + \left( \frac{A_{jk}}{2} - \frac{A_{ki}}{2} \right) \log r_k
Integral S(13)
\[ \int_{r=r_i}^{r=r_j} \left( \frac{z}{r} \right)^2 \, dz \, dr = \int_{r=r_i}^{r=r_j} \frac{z^3}{3r} \, dr \]
\[ = \frac{1}{3} \int_{r=r_i}^{r=r_j} \left( A_{ij} + B_{ij} \right) r^3 \, dr = \frac{1}{3} \int_{r=r_i}^{r=r_j} \left( A_{ij} r^3 + 3A_{ij}^2 B_{ij} r + 3A_{ij} B_{ij}^2 r + B_{ij}^3 r^2 \right) \, dr \]
\[ = \frac{1}{3} A_{ij} \left[ \log r \right]_{r=r_i}^{r=r_j} + A_{ij}^2 B_{ij} \left[ r \right]_{r=r_i}^{r=r_j} + \frac{A_{ij}}{2} B_{ij}^2 \left[ r^2 \right]_{r=r_i}^{r=r_j} + \frac{B_{ij}^3}{9} \left[ r^3 \right]_{r=r_i}^{r=r_j} \]
\[ = \frac{A_{ij}^3}{3} \left( \log r_j - \log r_i \right) + A_{ij}^2 B_{ij} \left( r_j - r_i \right) + \frac{A_{ij}}{2} B_{ij}^2 \left( r_j^2 - r_i^2 \right) + \frac{B_{ij}^3}{9} \left( r_j^3 - r_i^3 \right) \]
\[ = \frac{A_{ij}^3}{3} \log r_j - \frac{A_{ij}^3}{3} \log r_i + A_{ij}^2 B_{ij} \left( r_j - r_i \right) + \frac{A_{ij}}{2} B_{ij}^2 \left( r_j^2 - r_i^2 \right) + \frac{B_{ij}^3}{9} \left( r_j^3 - r_i^3 \right) \]
\[ \int_{r=r_k}^{r=r_j} \int_{z=A_{jk} r_k}^{z=A_{jk} r_j} \frac{z^2}{r} \, dz \, dr \]
\[ = \frac{A_{jk}^3}{3} \log r_j - \frac{A_{jk}^3}{3} \log r_i + A_{jk}^2 B_{jk} \left( r_j - r_i \right) + \frac{A_{jk}}{2} B_{jk}^2 \left( r_j^2 - r_i^2 \right) + \frac{B_{jk}^3}{9} \left( r_j^3 - r_i^3 \right) \]
\[ \int_{r=r_k}^{r=r_j} \int_{z=A_{ki} r_k}^{z=A_{ki} r_j} \frac{z^2}{r} \, dz \, dr \]
\[ = \frac{A_{ki}^3}{3} \log r_i - \frac{A_{ki}^3}{3} \log r_j + A_{ki}^2 B_{ki} \left( r_i - r_j \right) + \frac{A_{ki}}{2} B_{ki}^2 \left( r_i^2 - r_j^2 \right) + \frac{B_{ki}^3}{9} \left( r_i^3 - r_j^3 \right) \]
Sum = \[ A_{ij}^2 B_{ij} \left( r_j - r_i \right) + A_{jk}^2 B_{jk} \left( r_j - r_i \right) + A_{ki}^2 B_{ki} \left( r_i - r_j \right) \]
\[ + \frac{A_{ij}}{2} B_{ij}^2 \left( r_j^2 + r_i \right) \left( r_j - r_i \right) + \frac{A_{jk}}{2} B_{jk}^2 \left( r_j + r_i \right) \left( r_j - r_i \right) + \frac{A_{ki}}{2} B_{ki}^2 \left( r_i + r_j \right) \left( r_i - r_j \right) \]
\[ + \frac{B_{ij}}{9} \left( r_j^2 - r_i^2 \right) \left( r_j^2 + r_i \right) + \frac{B_{jk}}{9} \left( r_j^2 - r_i^2 \right) \left( r_j + r_i \right) + \frac{B_{ki}}{9} \left( r_i^2 - r_j^2 \right) \left( r_i + r_j \right) \]
\[ + \left( \frac{A_{ki}}{3} - \frac{A_{ij}}{3} \right) \log r_i + \left( \frac{A_{ij}^3}{3} - \frac{A_{jk}^3}{3} \right) \log r_j + \left( \frac{A_{jk}^3}{3} - \frac{A_{ki}^3}{3} \right) \log r_k \]

145
Integral S(14)

\[ r=r_j \quad z=A_{ij}^0 + B_{ij}^0 \quad r^3 = \frac{z^{1/4}}{4r} \quad z=A_{ij}^0 + B_{ij}^0 \]

\[
\int_{r=r_i}^{r=r_j} \int_{z=0}^{z=\frac{1}{4r}r^3} \frac{z^3}{r} \, dz \, dr = \int_{r=r_i}^{r=r_j} \left[ \frac{z^{1/4}}{4r} \right]_{z=0}^{z=\frac{1}{4r}r^3} \, dr
\]

\[ = \frac{1}{4} \int_{r=r_i}^{r=r_j} \frac{1}{r^3} (A_{ij}^0 r + B_{ij}^0)^4 \, dr \]

\[ = \frac{1}{4} \int_{r=r_i}^{r=r_j} \left( \frac{A_{ij}^0}{r} + 4A_{ij}^2 B_{ij} + 6A_{ij}^2 B_{ij} r^2 + 4A_{ij}^2 B_{ij} r^3 + B_{ij}^4 \right) \, dr \]

\[ = \frac{1}{4} A_{ij}^0 \left[ \log r \right]_{r_i}^{r_j} + A_{ij}^2 B_{ij} \left[ r^2 \right]_{r_i}^{r_j} + \frac{3}{4} A_{ij}^2 B_{ij} \left[ r^2 \right]_{r_i}^{r_j} + \frac{A_{ij}^4}{4} B_{ij} \left[ r^4 \right]_{r_i}^{r_j} \]

\[ = \frac{A_{ij}^4}{4} (\log r_j - \log r_i) + A_{ij}^2 B_{ij} (r_j^2 - r_i^2) + \frac{3}{4} A_{ij}^2 B_{ij} (r_j^2 - r_i^2) + \frac{A_{ij}^4}{4} B_{ij} (r_j^3 - r_i^3) + \frac{B_{ij}^4}{16} (r_j^4 - r_i^4) \]
similarly
\[
\int_{r=r_j}^{r=r_k} \int_{z=0}^{r} \frac{z^3}{r} \, dz \, dr
\]
\[
= A_{jk}^3 \left( \log r_k - \log r_j \right) + A_{ijk}^2 B_{kj}(r_k-r_j) + \frac{3}{4} A_{jk}^2 B_{jk}(r_k^2-r_j^2) + \frac{A_{jk}^3}{3} B_{jk}(r_k^3-r_j^3) + \frac{B_{jk}^4}{16} (r_k^4-r_j^4)
\]
\[
and \quad \int_{r=r_i}^{r=r_k} \int_{z=0}^{r} \frac{z^3}{r} \, dz \, dr
\]
\[
= A_{ki}^4 (\log r_k - \log r_i) + A_{ki}^3 B_{ki}(r_i-r_k) + \frac{3}{4} A_{ki}^2 B_{ki}(r_i^2-r_k^2) + \frac{A_{ki}^3}{3} B_{ki}(r_i^3-r_k^3) + \frac{B_{ki}^4}{16} (r_i^4-r_k^4)
\]
\[
\text{sum} = A_{ij}^5 B_{ij}(r_j-r_i) + A_{jk}^3 B_{jk}(r_k-r_j) + A_{ki}^3 B_{ki}(r_i-r_k)
\]
\[
+ \frac{3}{4} A_{ij}^2 B_{ij}(r_j^2+r_i^2)(r_j-r_i) + \frac{3}{4} A_{jk}^2 B_{jk}(r_k^2+r_j^2)(r_k-r_j) + \frac{3}{4} A_{ki}^2 B_{ki}(r_k^2+r_i^2)(r_k-r_i)
\]
\[
+ \frac{A_{ij}^3}{3} B_{ij}(r_j^2+r_i^2+r_j^2)(r_j-r_i) + \frac{A_{jk}^3}{3} B_{jk}(r_k^2+r_j^2+r_j^2)(r_k-r_j) + \frac{A_{ki}^3}{3} B_{ki}(r_k^2+r_i^2+r_i^2)(r_k-r_i)
\]
\[
+ \frac{B_{ij}^4}{16} (r_j^2+r_i^2)(r_j+r_i)(r_j-r_i) + \frac{B_{ik}^4}{16} (r_k^2+r_i^2)(r_k+r_i)(r_k-r_i) + \frac{B_{ki}^4}{16} (r_i^2+r_k^2)(r_i+r_k)(r_i-r_k)
\]
\[
+ (A_{ki}^2+A_{ij}^2)(A_{ki}+A_{ij})(A_{ki}-A_{ij}) \frac{\log r_i}{4} + (A_{ij}^2+A_{jk}^2)(A_{ij}+A_{jk})(A_{ij}-A_{jk}) \frac{\log r_j}{4} + (A_{jk}^2+A_{ki}^2)(A_{jk}+A_{ki})(A_{jk}-A_{ki}) \frac{\log r_k}{4}
\]
\[
\int_{r=r_1}^{r=r_2} \int_{z=0}^{z=A_j + B_j r} \frac{z^5}{r} \, dz \, dr = \int_{r=r_1}^{r=r_2} \left[ \frac{z^5}{5r} \right]_{z=0}^{z=A_j + B_j r} \, dr
\]

\[
= \frac{1}{5} \int_{r=r_1}^{r=r_2} \frac{1}{r} (A_{ij} + B_{ij} r)^5 \, dr
\]

\[
= \frac{1}{5} \int_{r=r_1}^{r=r_2} \frac{1}{r} \left( \frac{A_{ij}}{r} + 5A_{ij} B_{ij} r + 10A_{ij}^3 B_{ij} r^2 + 10A_{ij} B_{ij}^3 r^3 + 5A_{ij} B_{ij}^4 r^4 + B_{ij}^5 r^5 \right) \, dr
\]

\[
= \frac{1}{5} \left[ \frac{A_{ij}}{r} \log r + 5A_{ij} B_{ij} r + 10A_{ij}^3 B_{ij} r^2 + 10A_{ij} B_{ij}^3 r^3 + 5A_{ij} B_{ij}^4 r^4 + B_{ij}^5 r^5 \right]^{r_2}_{r_1}
\]

\[
= \frac{A_{ij}^5}{5} ( \log r_j - \log r_i ) + A_{ij} B_{ij} (r_j - r_i) + A_{ij}^3 B_{ij} (r_j^2 - r_i^2) + \frac{2}{3} A_{ij}^2 B_{ij}^3 (r_j^3 - r_i^3)
\]

\[
+ \frac{A_{ij} B_{ij}}{4} (r_j^4 - r_i^4) + \frac{B_{ij}^5}{25} (r_j^5 - r_i^5)
\]

\[
k = \frac{A_{ik}^5}{5} ( \log r_k - \log r_i ) + A_{ik} B_{ik} (r_k - r_i) + A_{ik}^3 B_{ik} (r_k^2 - r_i^2) + \frac{2}{3} A_{ik}^2 B_{ik}^3 (r_k^3 - r_i^3)
\]

\[
+ \frac{A_{ik} B_{ik}}{4} (r_k^4 - r_i^4) + \frac{B_{ik}^5}{25} (r_k^5 - r_i^5)
\]

\[
i = \frac{A_{ki}^5}{5} ( \log r_i - \log r_k ) + A_{ki} B_{ki} (r_i - r_k) + A_{ki}^3 B_{ki} (r_i^2 - r_k^2)
\]

\[
+ \frac{2}{3} A_{ki}^2 B_{ki}^3 (r_i^3 - r_k^3) + \frac{A_{ki} B_{ki}}{4} (r_i^4 - r_k^4) + \frac{B_{ki}^5}{25} (r_i^5 - r_k^5)
\]
Sum =
\[ A_{ij}B_{ij}(r_j - r_i) + A_{jk}B_{jk}(r_k - r_j) + A_{ki}B_{ki}(r_i - r_k) \]
\[ + A_{ij}^3B_{ij}^2(r_j - r_i)(r_j + r_i) + A_{jk}^3B_{jk}^2(r_k - r_j)(r_k + r_j) + A_{ki}^3B_{ki}^2(r_i - r_k)(r_i + r_k) \]
\[ + \frac{2}{3}A_{ij}^3B_{ij}3(r_j - r_i)(r_j^2 + r_j^r_i + r_i^2) + \frac{2}{3}A_{jk}^3B_{jk}3(r_k - r_j)(r_k^2 + r_k r_j + r_j^2) + \frac{2}{3}A_{ki}^3B_{ki}3(r_i - r_k)(r_i^2 + r_i r_k + r_k^2) \]
\[ + \frac{A_{ii}^4B_{ii}4(r_j - r_i)(r_j + r_i)(r_j^2 + r_i^2)}{4} - \frac{A_{jk}^4B_{jk}4(r_k - r_j)(r_k + r_j)(r_k^2 + r_j^2)}{4} + \frac{A_{ki}^4B_{ki}4(r_i - r_k)(r_i + r_k)(r_i^2 + r_k^2)}{4} \]
\[ + \frac{B_{ij}^5}{25}(r_j - r_i)(r_j^4 + r_j^3 r_i + r_j^2 r_i^2 + r_j r_i^3 + r_i^4) + \frac{B_{jk}^5}{25}(r_k - r_j)(r_k^4 + r_k^3 r_j + r_k^2 r_j^2 + r_k r_j^3 + r_j^4) + \frac{B_{ki}^5}{25}(r_i - r_k)(r_i^4 + r_i^3 r_k + r_i^2 r_k^2 + r_i r_k^3 + r_k^4) \]
\[ + \frac{(A_{ki}^5 - A_{ij}^5)}{5} \log r_i + \frac{(A_{ij}^5 - A_{jk}^5)}{5} \log r_j + \frac{(A_{jk}^5 - A_{ki}^5)}{5} \log r_k \]
Tests of Integrals by Computer

In a separate section, algebraic formulae integrals of functions of \( r \) and \( z \) over the area of a triangle \( i, j, k \) have been formed. Six of these integrals may be readily calculated by using an equi-momental figure. An equi-momental figure of a triangle is formed by placing a third of its area at the mid point of each of its three sides. The integrals which may be calculated in this way are:

- \( S(7) = \int \int dzdr \) = area of the triangle
- \( S(1) = \int \int r dzdr \) = first moment of triangle about the \( z \) axis (which is the axi symmetric axis)
- \( S(8) = \int \int z dzdr \) = first moment of the triangle about the \( r \) axis (a radial axis)
- \( S(3) = \int \int r^2 dzdr \) = second moment of the triangle about the \( z \) axis
- \( S(9) = \int \int z^2 dzdr \) = second moment of the triangle about the \( r \) axis
- \( S(2) = \int \int r dzdr \) = the product moment of the area of the triangle

The remaining nine integrals however, need not necessarily be able to be checked by using an equi-momental triangle. There was also the question of accuracy. This was particularly important when the size of the triangles were small in comparison with their distances from the axes. That is; the difference in the ordinates of the vertex nodes was very small in comparison with the magnitude of the ordinates.

For this purpose a computer program was written called INTEGRATION CHECK in which the value of each of the fifteen integrals for several triangles was calculated. Not only was the final value of the integral printed out, but also the values of the formula were printed out to eleven or more significant figures. The sum(s) or difference(s), as appropriate, at each stage were printed out in order to assess the accuracy in terms of number of significant figures.
Figures 3.1 to 3.8 illustrate the types of triangles which were used to test the formulae of the integrals. The R, Z co-ordinates of the three vertex nodes of each triangle were read in on punched cards as data to the program INTEGRATION CHECK.

Figures 3.1 and 3.2 are 45° triangles which when joined together would form a square. So that each of the six integrals S(7), S(1), S(6), S(3), S(9) and S(2) of one 45° triangle (figure 3.1) could be added to the corresponding integral of the other 45° triangle (figure 3.2) to form the area or the moments of area of the 5" x 5" square about datum origin.

The triangle illustrated in figure 3.3 was used to test the accuracy of a small triangle at a large distance from the axis. That is the triangle is only about 0.002" wide by about 0.002" high but is 10 inches in a radial direction from the Z axis and is 20 inches in an axial direction from the abscissa. That is, the difference in the ordinates of the vertex nodes was very small in comparison with the distance from the axes. This triangle, however, was almost an equilateral triangle. Whereas the triangle shown in figure 3.4 is much slenderer. That is, it is 0.01" long by about 0.002" thick whilst being at 10 inches radius and 20 inches along the axis. This triangle was convenient for equi-moment calculations. For the integral S(15) \( \int \int \frac{z^2}{r} \, dz \, dr \) the computer program gave 0.00155 whereas by slide rule the \( z^2/r \) times the area of the triangle gave 0.0016 (where z and r are the co-ordinates of the centre of area), but the value of S(15) in the computer program is calculated (and printed out) in three stages by summation process. In the first stage the summation is 343,646.96458; at the second stage it comes to 395,787.62759; and the third (final) stage to 0.00155. So between the second and third stages a negative quantity was added which only differed to the second summation at the ninth significant figure. As the computer on which the program was run was only accurate to 11 significant (decimal) figures; it seems that in this instance the computer value of S(15) can only be accurate to three significant figures.
Figure 3.5 shows a square divided into four triangles. The lower left hand corner is 10 inches from both axes. It is evident from this diagram that the fifth and sixth triangles are reflections of each other (about a 45° line). So that:

\[ S(1) = \iiint r \, dz \, dr, \quad \text{and} \quad S(8) = \iiint z \, dr \, dz, \quad \text{first moment of the triangle} \]

\[ S(3) = \iiint r^2 \, dz \, dr, \quad \text{and} \quad S(9) = \iiint z^2 \, dr \, dz, \quad \text{second moment of the triangle} \]

\[ S(4) = \iiint r^3 \, dz \, dr, \quad \text{and} \quad S(1^2) = \iiint z^3 \, dr \, dz, \quad \text{third moment of the triangle} \]

\[ S(6) = \iiint r^2 \, dz \, dr, \quad \text{and} \quad S(5) = \iiint r z^2 \, dr \, dz \]

should correspond, and they do. Also

\[ S(1) = \iiint dz \, dr, \quad \text{the area of the triangle} \]

\[ S(2) = \iiint r \, dz \, dr, \quad \text{the product moment} \]

all should agree and they do. Triangles 7 and 8 are symmetric about 45° and so the second moments:

\[ \iiint r^2 \, dz \, dr, \quad \iiint r z^2 \, dr \, dz, \quad \iiint s^2 \, dz \, dr, \]

(ie S(3), S(9) and S(2)) should have the same value; they have but S(2) differs at the seventh significant figure. The first ten integrals appear to be consistent within themselves.

Figure 3.6 shows eight triangles which were used to test (a) those integrals which were independent of radius didn't vary with variation of radial position

(ie S(8) = \iiint z \, dr \, dz, \quad S(9) = \iiint z^2 \, dr \, dz, \quad \text{and} \quad S(10) = \iiint z^3 \, dr \, dz); \quad \text{and (b) those integrals which were independent with axial position didn't vary with variation of axial position.} \]

Finally figure 3.7 shows a 5" x 5" x 45° triangle (whose lower left vertex is 10" from both axes) which has been divided up into a hundred \( \frac{1}{2} \)" x \( \frac{1}{2} \)" x 45° triangles. In figure 3.8 the centroids of these hundred triangles are shown in units of sixtieths (so as to input in whole numbers). The results of the formula of the fifteen integrals were compared with the appropriate
the area of $\frac{1}{2}'' \times \frac{1}{2}'' \times 45^\circ$ triangle, (where $r$ and $z$ are the centroid coordinates) for the hundred triangles. There was insignificant difference. Also each of the fifteen integrals were summated separately for the hundred triangles and compared with the $5'' \times 5'' \times 45^\circ$ triangle; again there was no significant difference. The $\frac{1}{2}'' \times \frac{1}{2}'' \times 45^\circ$ triangle No. 57 has the same centroid as the $5'' \times 5'' \times 45^\circ$ triangle. Ignoring the difference in scale the fifteen integrals varied by a maximum of $4\%$. 
Figure 3.1
**Figure 3.3**

Diagram showing points labeled i, j, and k with distances as follows:
- Point i to j: 20.0000
- Point j to k: 20.0000
- Point k to j: 10.0019
- Point j to i: 10.0000
- Point i to j: 10.0008
- Point j to k: 20.0018

Directional arrows indicate the direction of the points and distances.
Figure 3.5
FIGURE 3.6
FIGURE 3.7
APPENDIX 1f

To show what happens when at least one of the radial ordinates of the vertex nodes become zero

In the integrals $S(11)$, $S(12)$, $S(13)$, $S(14)$, and $S(15)$ occurs the expression:

$$\frac{1}{n} \left[ A_{ki}^n - A_{ij}^n \right] \log r_i$$

In $S(11)$, $n = 1$; in $S(12)$, $N = 2$; in $S(13)$, $n = 3$; in $S(14)$, $n = 4$; and in $S(15)$, $n = 5$. It is required to determine the value of this expression as $r_i$ approaches zero.

$$A_{ki} = \frac{(r_i z_k - r_k z_i)}{(r_i - r_k)} \quad A_{ij} = \frac{(r_j z_i - r_i z_j)}{(r_j - r_i)}$$

so that

$$\left\{ \frac{(r_i z_k - r_k z_i)^n}{(r_i - r_k)} \right\} - \left\{ \frac{(r_j z_i - r_i z_j)^n}{(r_j - r_i)} \right\}$$

$$= \left\{ \frac{r_i^* z_i - r_k z_i}{r_i - r_k} \right\} + \left\{ \frac{r_i^* z_k - r_i z_i}{r_i - r_k} \right\} - \left\{ \frac{r_j^* z_i - r_i z_i}{r_j - r_i} \right\} + \left\{ \frac{r_i^* z_i - r_i z_i}{r_j - r_i} \right\}$$

$$= \left\{ z_i - r_i \left( \frac{z_i - z_k}{r_i - r_k} \right) \right\} - \left\{ z_i - r_i \left( \frac{z_i - z_i}{r_j - r_i} \right) \right\}$$

(* The mathematical artifice of adding and subtracting $r_i z_i$)
\[ z_i - n \sum_{j=1}^{n} \frac{z_i - z_j}{r_i - r_j} \cdot \frac{z_i - z_j}{r_i - r_j} = z_i - n \sum_{j=1}^{n} \frac{z_i - z_j}{r_i - r_j} \cdot \frac{z_i - z_j}{r_i - r_j} \]

Therefore, \( \frac{1}{n} (A x_i - A_{ij}) \log r_i \)

which \( \to 0 \) as \( r_i \to 0 \)
Alternatively

Let \( Q = \frac{1}{A_{1ij}} = \frac{(r_i - r_k)(r_j - r_i)}{((r_i - z_k)(r_k - r_i)(r_j - r_2) - (r_j - z_k)(r_k - z_j)(r_i - r_k))}
\]

\[= \frac{-r_i^2 + (r_j + r_k)(r_i - r_1)(r_k - r_1) - r_i^2}{r_j^2 + (r_j - z_k)(r_k - z_j) + r_k(z_j - z_i) + r_j(z_k - z_i)} \]

\[= \frac{-r_i^2 + (r_j - r_k)(r_i - r_1)(r_k - r_1)}{(z_j - z_k)r_i^2 + (r_j(z_k - z_j) + r_k(z_j - z_i))r_i} \]

\[
d\frac{Q}{dr_i} = \frac{(z_j - z_k)r_i^2 + (r_j(z_k - z_j) + r_k(z_j - z_i))r_i}{(z_j - z_k)r_i^2 + (r_j(z_k - z_j) + r_k(z_j - z_i))r_i} \]

\[
r_i^2 \frac{dQ}{dr_i} = \frac{(z_j - z_k)r_i^2 + (r_j(z_k - z_j) + r_k(z_j - z_i))r_i}{(z_j - z_k)r_i^2 + (r_j(z_k - z_j) + r_k(z_j - z_i))r_i} \]

\[
\text{as } r_i \to 0, \quad r_i^2 \frac{dQ}{dr_i} = \frac{r_j(z_k - z_j) + r_k(z_j - z_i)}{(z_j - z_k)_i^2 + (r_j(z_k - z_j) + r_k(z_j - z_i))} = \frac{r_jr_k}{(z_j - z_k)_i} = k
\]

\[
as A_{1ij} \to 0, \quad A_{1ij} = \frac{(z_i - z_k)(r_i - r_k)}{((r_i - z_k)(r_k - z_j)(r_j - r_2) - (r_j - z_k)(r_k - z_j)(r_i - r_k))}
\]

\[= \left\{ z_i - r_1 \frac{(z_j - z_k)}{r_j - r_2} \right\} \left\{ z_i - r_1 \frac{(z_j - z_k)}{r_j - r_1} \right\} = -r_1 \left\{ \frac{z_i - z_k}{r_j - r_1} - \frac{z_i - z_j}{r_j - r_1} \right\} = \frac{z_i - z_j}{q}
\]

\[
\frac{1}{Qr_i} = \frac{(z_i - z_j)(z_j - z_i)}{(r_k - r_1)(r_k - r_1)}\]

so that \( \frac{Qr_i}{r_i^2} \frac{dQ}{dr_i} = \frac{Q}{r_i} \frac{dQ}{dr_i} = -1 \) as \( r_i \to 0 \)
so that, as $r_1 \to 0$

$$Q = - r_1 \frac{dq}{dr_1}$$

As $(A_{ki} - A_{ij}) \log r_1 = \frac{\log r_1}{Q}$

in the limit as $r_1 \to 0$

$$\frac{\log r_1}{Q} \cdot \frac{\log r_1}{-r_1 \frac{dq}{dr_1}} = \frac{r_1 \log r_1}{-r_1^2 \frac{dq}{dr_1}} = \frac{r_1 \log r_1}{-k}$$

$$= \frac{\log r_1}{-k/r_1}, \text{ on differentiating numerator & denominator } \text{(by } d e l \text{ Hopital's rule)}$$

$$\frac{d}{dr_1} (\log r_1) = \frac{1}{r_1} \Rightarrow \frac{r_1^2}{2k} \Rightarrow \frac{r_1}{2k}$$

which as $r_1 \to 0$, $\frac{r_1}{2k} \to 0$

therefore

$$(A_{ki} - A_{ij}) \log r_1 \to 0 \text{ as } r_1 \to 0$$

Physically

Regarding $r_1 \to 0$, the radial ordinate of a vertex node is approaching the axis. The $\int_0^1 \frac{dz}{r}$ which gives the function of $\log r_1$, comes from the hoop strain $= \frac{u}{r}$ (where $u$ is the radial displacement and $r$ is the radius). But the radial displacement $u$ on the axis is zero (as there is no body movement), therefore displacement function must be zero and all integrals arising from this must be zero also.
To show what happens when two vertex nodes of a triangular element have the same radius (e.g. \(r_k + r_j\))

\[
S(11) = A_{ij} (\log r_j - \log r_i) + B_{ij} (r_j - r_i)
+ A_{jk} (\log r_k - \log r_j) + B_{jk} (r_k - r_j)
+ A_{ki} (\log r_i - \log r_k) + B_{ki} (r_i - r_k)
\]

\[
= A_{ij} \log \frac{r_j}{r_i} + \frac{z_j - z_i}{r_j - r_i} (r_j - r_i)
+ A_{jk} \log \frac{r_k}{r_j} + \frac{z_k - z_j}{r_k - r_j} (r_k - r_j)
+ A_{ki} \log \frac{r_i}{r_k} + \frac{z_i - z_k}{r_i - r_k} (r_i - r_k)
\]

\[
= A_{ij} \log \frac{r_j}{r_i} + z_j - z_i
+ A_{jk} \log \frac{r_k}{r_j} + z_k - z_j
+ A_{ki} \log \frac{r_i}{r_k} + z_i - z_k
\]

(cancelling the + and - z values)

\[
= A_{ij} \log \frac{r_j}{r_i} + A_{jk} \log \frac{r_k}{r_j} + A_{ki} \log \frac{r_i}{r_k}
\]
(Rearranging and changing sign of $A_{ki}$)

$$= A_{ij} \log \frac{r_j}{r_i} - A_{ki} \log \frac{r_k}{r_i} + A_{jk} \log \frac{r_k}{r_j}$$

(subtracting and adding $A_{ki} \log \frac{r_j}{r_i}$ w.r.t ist two terms)

$$= A_{ij} \log \frac{r_j}{r_i} - A_{ki} \log \frac{r_j}{r_i} - A_{ki} \log \frac{r_k}{r_i} + A_{ki} \log \frac{r_j}{r_i} + A_{jk} \log \frac{r_k}{r_j}$$

$$= (A_{ij} - A_{ki}) \log \frac{r_j}{r_i} - A_{ki} \log \frac{r_k}{r_i} + A_{jk} \log \frac{r_k}{r_j}$$

these terms are finite as $r_i + r_j$.

The remaining terms are (taking last product first)

(assume $r_k > r_j$)

$$= A_{jk} \log \frac{r_k}{r_j} + (A_{ij} - A_{ki}) \log \frac{r_j}{r_i}$$

$$= \frac{r_k z_i - r_j z_k}{r_k - r_j} \log \left(1 + \frac{r_k - r_j}{r_j}ight) + m$$

(Where $m$ stands for $(A_{ij} - A_{ki}) \log \frac{r_j}{r_i}$ and

$$\frac{r_k}{r_j} = \frac{r_j + r_k - r_j}{r_j} = \frac{r_j}{r_j} + \frac{r_k - r_j}{r_j} = 1 + \frac{r_k - r_j}{r_j}$$

165
Note \[ \log (1 + x) + x \] as \[ x \to 0 \]

therefore as \( r_k + r_j \) the equation becomes

\[
\frac{r_k z_j - r_j z_k}{r_k - r_j} \cdot \frac{r_k - r_j}{r_j} + m
\]

\[
= \frac{r_k z_j - r_j z_k}{r_j}
\]

\[
= \frac{r_k}{r_j} \cdot z_j - z_k + m
\]

(and as \( r_k + r_j \)

\[
= \frac{r_k}{r_j} + 1
\]

\[
= z_j - z_k + m
\]

(subtracting and adding \( z_i \) between \( z_j \) and \( z_k \))

\[
= z_j - z_i + z_i - z_k + m
\]

(dividing and multiplying the first pair by \( (r_j - r_i) \) and the second pair by \( (r_i - r_k) \))

\[
= \frac{(z_j - z_i)}{(r_j - r_i)} \times (r_j - r_i) + \frac{(z_i - z_k)}{(r_i - r_k)} \times (r_i - r_k) + m
\]
= B_{ij} (r_j - r_i) + B_{ki} (r_i - r_k) + m

(inserting the value of m, but at the beginning)

= (A_{ij} - A_{ki}) \log \frac{r_i}{r_j} + B_{ij} (r_j - r_i) + B_{ki} (r_i - r_k)

(putting the intercept (A) and the slope (B) of the same suffix together on the same line)

= A_{ij} (\log r_j - \log r_i) + B_{ij} (r_j - r_i)

+ A_{ki} (\log r_i - \log r_j) + B_{ki} (r_i - r_k)

as r_k + r_j then - \log r_j (which occurs on the second line) may be written as - \log r_k

= A_{ij} (\log r_j - \log r_i) + B_{ij} (r_j - r_i)

+ A_{ki} (\log r_i - \log r_k) + B_{ki} (r_i - r_k)

This is tantamount to taking the initial equation of S(11) but making both A_{jk} = 0 and B_{jk} = 0 when r_k + r_j.
APPENDIX 2a

ASSEMBLY DISC

The purpose of these notes is to describe briefly the background to the computer program ASSEMBLY DISC. This program assembles the complete stiffness matrix on disc and then transfers it to tape as a form of output. This tape will be the input to another program and the ultimate output will be displacements, strains and stresses.

Consider a portion of a hollow cylinder shown in figure 5.1, an axial-radial section of this is shown in figure 5.2; which, instead of being cross-batched, is completely divided into triangles. Each triangle is a section through an annular ring which is a single element. Thus the finite-elements are axi-symmetric. Each element is joined at its nodes to the adjacent elements as shown in figure 5.3. The nodes are numbered so as to make the bandwidth of the assembled stiffness matrix as small as possible.

In order to obtain positive area these triangles have to be defined by their boundary in a clockwise direction. Three vertices are sufficient to define a triangle. Thus the ordinates of nodes shown in figure 5.4 may be used to define the triangles in figure 5.3. The ordinates of the nodes shown in figure 5.4 are given in table 5.1.

<table>
<thead>
<tr>
<th>Node</th>
<th>Z</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0</td>
<td>10.0</td>
</tr>
<tr>
<td>3</td>
<td>0.0</td>
<td>15.0</td>
</tr>
<tr>
<td>5</td>
<td>0.0</td>
<td>20.0</td>
</tr>
<tr>
<td>11</td>
<td>5.0</td>
<td>10.0</td>
</tr>
<tr>
<td>13</td>
<td>5.0</td>
<td>15.0</td>
</tr>
<tr>
<td>15</td>
<td>5.0</td>
<td>20.0</td>
</tr>
<tr>
<td>21</td>
<td>10.0</td>
<td>10.0</td>
</tr>
<tr>
<td>23</td>
<td>10.0</td>
<td>15.0</td>
</tr>
<tr>
<td>25</td>
<td>10.0</td>
<td>20.0</td>
</tr>
<tr>
<td>31</td>
<td>20.0</td>
<td>10.0</td>
</tr>
<tr>
<td>33</td>
<td>20.0</td>
<td>15.0</td>
</tr>
<tr>
<td>35</td>
<td>20.0</td>
<td>20.0</td>
</tr>
</tbody>
</table>
Input data of ordinates, nodes and triangle numbers, is verified by a computer program which draws triangles as graphical output. How this is done is illustrated diagrammatically in figure 5.5; a single continuous line scribes around each triangle in a clockwise direction. This is achieved by numbering the triangles as in figure 5.6; in each rectangle the smaller triangle number is given to the lower triangle and the larger triangle number to the upper triangle.

TABLE 5.2

<table>
<thead>
<tr>
<th>Triangle Number</th>
<th>Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>13</td>
</tr>
<tr>
<td>8</td>
<td>13</td>
</tr>
<tr>
<td>9</td>
<td>21</td>
</tr>
<tr>
<td>10</td>
<td>21</td>
</tr>
<tr>
<td>11</td>
<td>23</td>
</tr>
<tr>
<td>12</td>
<td>23</td>
</tr>
</tbody>
</table>
Table 5.2 gives the node numbers in a clockwise direction corresponding to figure 5.6 and facilitating the continuous line shown in figure 5.5.

Figure 5.7 shows the first triangle on its own with its six node numbers. Notice that the mid-side node number is the mean of the two adjacent vertex node numbers. Figure 5.8 again shows triangle number 1, but with its twelve displacement numbers, of which the radial displacement numbers are odd, and the axial displacement numbers are even. The displacement numbers of any node are related to that node's number; such that the axial displacement number is twice the node number, and the radial displacement number is one less than the axial displacement number. There are forces at each node which act in the same direction as indicated by the displacement numbers. A displacement of any magnitude, sense or direction, may be represented by two orthogonal displacements in that plane. It follows that a force of any magnitude, sense or direction, may be represented by two orthogonal force components in that plane. Any variation of the orthogonal force components will affect all the orthogonal displacement components. Thus Table 5.3 is formed which connects all the force components with all the displacement components. The ratio of force to displacement is known as stiffness thus Table 5.3 is the form of a table of stiffnesses and is called an "element-stiffness-matrix", (each x representing a number, the magnitude of the stiffness). The derivation of the element-stiffness-matrix is described in a previous chapter by the author.

The triangular-element behaves as a spring and if springs act together their stiffnesses are additive. In fact all of the triangular-elements for a row of two rectangles added together produce a 30 x 30 matrix, which may be partitioned into three equal widths by three equal heights; each
### Table 5.5

Each X represents a stiffness value.

| Forces | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|--------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 1      | X | X | X | X | X | X |   | X | X | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 2      | X | X | X | X | X | X |   | X | X | X |   | X |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 3      | X | X | X | X | X | X |   | X | X | X |   | X | X |   |   |   |   |   |   |   |   |   |   |   |   | X |   |   |
| 4      | X | X | X | X | X | X |   | X | X | X |   | X | X | X |   |   |   |   |   |   |   |   |   | X |   |   | X |   |
| 5      | X | X | X | X | X | X |   | X | X | X |   | X | X | X | X |   |   |   |   |   |   |   | X |   |   | X |   |
| 6      | X | X | X | X | X | X |   | X | X | X |   | X | X | X | X | X |   |   |   |   |   | X |   |   | X |   |
| 7      |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 8      |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 9      |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 10     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 11     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 12     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 13     | X | X | X | X | X | X |   | X | X | X |   | X | X | X | X | X |   |   |   |   |   |   |   |   |   | X |   |   |
| 14     | X | X | X | X | X | X |   | X | X | X |   | X | X | X | X | X | X |   |   |   |   |   |   |   | X |   |   |   |
| 15     | X | X | X | X | X | X |   | X | X | X |   | X | X | X | X | X | X | X |   |   |   |   | X |   |   | X |   |
| 16     | X | X | X | X | X | X |   | X | X | X |   | X | X | X | X | X | X | X | X |   | X |   | X |   | X |   |
| 17     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 18     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 19     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 20     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 21     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 22     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 23     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 24     |     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 25     | X | X | X | X | X | X |   | X | X | X |   | X | X | X | X | X | X | X | X | X | X | X | X | X | X |   |
| 26     | X | X | X | X | X | X |   | X | X | X |   | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |
band corresponding to a line of displacement numbers.

Figure 5.9 shows what the combined stiffness matrix for three rows of physical rectangles looks like. There is one large square for each row and each of these is partitioned into three bands for the three lines of displacement numbers per row of physical rectangles, forming nine small squares. Thus the larger squares overlap by one small square which represents the common line of displacement numbers. It is this type of banded partition matrix which is formed by the program ASSEMBLY DISC, the details of which follows.
FIGURE 5.1

AXIAL LENGTH

RADIUS
Program title: ASSEMBLY DISC

Purpose: to assemble axysymmetric stiffness matrix on Disc and transfer the assembled matrix onto magnetic tape.

Input data: First the maximum number of physical rectangles across the radial thickness and the maximum number of rows of rectangles along the axial length are the two numbers read off the first card. They are the number of spaces between the axial lines and the radial lines respectively. The program has been run for 10 and 50 respectively.

Next the matrix used for assembling the triangles per row is zeroed (CALL ZERASY).

Then all the vertex nodes and their ordinates are read off tape (CALL MODORD). This involves reading off a card the number of nodes to be read off the magnetic tape and then reading these nodes with their ordinates off the magnetic tape.

Next the physical properties of the material, that is, Young's Modulus of Elasticity and Poisson's Ratio are read in and printed out.

From a card the number of triangles to be read in, is read and printed out. Then a loop is started to go round that number of times. The first thing that happens in this loop is the triangle number and its node numbers are read in from magnetic tape (CALL TRIAIN) together with the axial index (IAXIAL) and an index to indicate a newline (NEWLIN).

Nota Bene Nodes and their ordinates are read before triangle and its nodes, because in that way there only needs to be provision for storing one triangle at a time, thus saving storage.

Procedure for each Triangle

(a) If the newline index is zero.

The ordinates of the three vertex are formed in the form of a $2 \times 3$ matrix $T$ (CALL FORMTE) and then the stiffness matrix $QD$ for the triangle is formed (CALL STIFMA). The stiffness matrix is formed in the form of
(6 RADIAL + 6 AXIAL) x (6 RADIAL + 6 AXIAL) and this is re-arranged so that odd numbers are radial and even numbers are axial (CALL REARRA).

The six nodial numbers are completed (CALL COMSIX) and the twelve displacement numbers formed (CALL FORM12). At that stage the stiffness matrix for the triangular element is added to the assembly matrix for the row of triangles (CALL ADDELE).

This process continues until the row of triangles is completed, that is, just before starting a newline.

(b) If the newline index is 1

This indicates that the triangle is the first triangle of a newline. Before anything is done about this triangle, the assembly matrix for the previous row (except for a portion relating to the common line of nodes) is transferred to disc (CALL SQTODES). The portion of the assembly matrix remaining, is transferred to the appropriate position ready to be added into the assembly matrix for the next row. When all this is done, the stiffness of the first triangle of the new row is calculated as before when the index was zero.

(c) After the last triangle of the last row has been added to the assembly matrix of that row, there is no newline index to release the assembly matrix to disc; so that after the loop the assembly matrix of the last row is sent to disc (CALL SQTODES and CALL LASLB).

Disc to Tape

All the data on disc is transferred to tape (CALL DSTOTA).
Sequence of subroutines used in "Assembly Disc"

(1) ZERASY. The assembly matrix is zeroed.

(2) NODORD. All the vertex nodes with their corresponding pairs of Z & R co-ordinates are read in off of the magnetic tape which was output from the computer program "Node Tape" (or its equivalent).

The material's stiffness properties: Young's Modules of Elasticity and Poisson's Ratio are read in off a punched card.

(3) TRIAIN. A triangle number, with its three vertex node numbers, axial (or meridian) index, and newline index are read in from the magnetic tape which was output from the computer program "Triangle Tape". The newline index is 1 when a new line of triangles across the thickness (ie a new space along the meridian) was begun otherwise the newline index is zero.

(4) FORMTE. The three vertex node numbers are used to refer to the node list (read in previously) to obtain the R & Z co-ordinates of each of the vertex nodes of the triangle, taken in clockwise direction around the triangle.

(5) STIFKA. The formation of the $12 \times 12$ triangle element stiffness matrix is due in the following manner.

(6) FORMAD. The $6 \times 6$ matrix is formed which if multiplied by the appropriate six coefficients would produce the radial displacements or the axial displacements of the six nodes. To form this $6 \times 6$ matrix the mid-side node's co-ordinates were taken as the arithmetic mean of the co-ordinates of the vertex nodes at the ends of that side. Thus the six pairs of co-ordinates are $(r_i, Z_i)$ where $i$ goes from 1 to 6; and the elements for each row are of the form:

$$r_i^2, r_i, 1, Z_i^2, Z_i, r_i Z_i$$

hence the six rows are formed. This matrix is known as $\delta A$.

(7) INVFPV. The $6 \times 6$ matrix $\delta A$ is inverted and is used to form the $12 \times 12$ matrix thus:

$$
\begin{pmatrix}
\delta A^{-1} & 1 & 0 \\
\delta A^{-1} & 1 & 0 \\
0 & \delta A^{-1} & 1 \\
\end{pmatrix}
$$

This matrix is called $AD$. 183
FORMSA. Of the stiffness expression

\[ [k] = \left( [\delta A]^{-1} \right)^T \left( \int [\delta e]^T [\delta e] \, dv \right) \left( [\delta A]^{-1} \right) \]

the central part:

\[ \int [\delta e]^T [\sigma e] \, [\delta e] \, dv \]

is to be formed.

This 12 x 12 matrix is called SA.

INTI15. In order to do this fifteen integrals are calculated.

TEN1ST. The first ten of these integrals are calculated using expressions where the radial ordinates of the vertices are represented by a, b, and c and the axial ordinates of the vertices by x, y and z. So that the first ten integrals are taken over the area of the whole triangle.

LAST 5. The last five integrals (for reasons explained in the section on the theory related to finite element) are worked out as integrals for the area under the three straight lines and summed as they proceed around the triangle in a clockwise direction.

SLOG. The log of each vertex radius, as there is a possibility of the radius being zero, is calculated separately when the fifteen integrals for the triangle have been formed; the 12 x 12 matrix SA is formed whose matrix elements are composed of equations in terms of these integrals.

FORMQD. This matrix SA is post-multiplied by the matrix AD forming a matrix SD. The transpose of the matrix AD is formed. The matrix SD is pre-multiplied by the transpose of matrix AD forming the matrix QD. That is

\[(QD) = (AD)^T (SA) (AD)\]

which is isomorphic to

\[ [k] = \left( [\delta A]^{-1} \right)^T \left( \int [\delta e]^T [\sigma e] [\delta e] \, dv \right) \left( [\delta A]^{-1} \right) \]

REARRA. The 12 x 12 stiffness matrix (QD) is now in the form of (6 radial, 6 axial) by (6 radial, 6 axial) and this needs to be rearranged so that the odd numbers are radial and the even numbers are axial.

COMSIX. Also the three mid side node numbers are needed, (they are the arithmetic mean of the vertex node numbers at the ends of the side) in order to complete the six node numbers.
FORM12. Each node has two displacement indices: the odd indices representing the radial displacements (or forces) and the even indices representing the even displacements (or forces). The twelve displacement numbers are formed from the six node numbers of the triangle.

ADDELE. Each triangle has a 12 by 12 stiffness matrix, but one row of triangles across the wall thickness filling one meridian space form a stiffness matrix of size (NDSPRL by NDSPRL) where:

\[ \text{NDSPRL} = 2 \times 3 \times (\text{twice the number of spaces across the thickness} + 1) \]

number of nodes across the thickness
number of nodes in a meridian space
number of displacement numbers within one meridian space

Hence the largest matrix of the core store is NDSPRL by NDSPRL in size, this is the assembly matrix. The elements of the 12 x 12 stiffness matrix for the triangle have to be added into this assembly stiffness matrix for one meridian space at the appropriate positions. The appropriate row and column locations are determined by the two displacement numbers representing the row and column in the 12 x 12 stiffness matrix of the triangular element.

When the stiffness for that triangle has been added, the flow of computation is then returned to step (3) where the data for the next triangle is read. Unless there is a new position along the meridian, the procedure goes through as before.

SQTOL3. If there is a new position along the meridian (other than the first position) then some of the assembly stiffness matrix for the unit meridian band width is written to disc. The assembly stiffness matrix for the band is a square matrix whose length of side is divisible by three. Thus this matrix may be divided into nine small square matrices. The first eight of these small squares, in the order: the first row of three squares; the second row of three squares; the third row of first two squares are all written to disc together with their axial index. The ninth small square is stored temporarily. The assembly matrix for the meridian unit length is zeroed.
Then what was the ninth small square matrix is placed in the location of the first small square. The assembly matrix for the band is then ready to start on the new meridian band. The data for the first triangle of this new band is already read in.

(19)  LASEMB. At the end of the last band the eight small squares are written on to disc as before and the ninth square is also written on to disc.

(20)  DSTOTA. The small squares are read from disc with their row, column, and axial indices, and written to the output magnetic tape. The output magnetic tape for this program now contains the stiffness matrix for the whole structure.
List of Subroutines used in Assembly Disc

1. ZERASY
2. NODORD
3. TRIAIN
4. FORMTE
5. STIFMA
6. FORMAD
7. INVPPV
8. FORMSA
9. INTI15
10. TEN1ST
11. LAST5
12. SLOG
13. FORMQD
14. REARRA
15. COMSIX
16. FORM12
17. ADDLE
18. SQTODS
19. LASSEMB
20. DSTOTA
21. PNSMSQ
22. PRMAHD
23. PNTASP
24. PLOCK
25. PRIDOZ
DECOMPOSITION

A set of simultaneous equations states the relationship between forces and displacements. Thus the coefficients of these equations are the stiffness influence coefficients; the right hand side of the equations are the forces applied at the nodes (in a radial or axial directions) and the displacements (in a radial or axial directions) are the unknowns to be determined.

The stiffness coefficients form a banded matrix of which physical considerations suggest the partitioning shown in figure 6.1. These notes outline the method used in dealing with decomposition of such a matrix into its upper and lower factors. Decomposition is used to solve the simultaneous equations for displacements.

In describing decomposition with partial pivoting the banded partitioned matrix shown in figure 6.1 is of the smallest size to allow for generality.

INPUT TAPE

The matrix indicated by figure 6.1 is generated by the program which has been called ASSEMBLY DISC and stored on magnetic tape. Each small square formed by the partitioning is called SMSQ and this is written on to the tape as a whole as follows. First is written the row number and the number of columns of SMSQ in that row, then each of the SMSQs is written as a whole preceded by its row and column number. This is indicated in table 6.1 where each line represents a separate write to tape.
<table>
<thead>
<tr>
<th>Row</th>
<th>Number of Columns</th>
<th>Row</th>
<th>Column</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>7</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>9</td>
<td>4</td>
</tr>
</tbody>
</table>

**TAPE TO DISC**

The tape is read and transferred to disc with the additional zero small squares as indicated in Figure 6.2. This allows for any rearrangement of the (matrix) element rows within a section. Thus consider the first three
rows of small squares. Any element row from within the third row of small squares could be interchanged with any element row from within the first or second row of small squares. Since the third row of small squares was initially five squares long, the first two rows of small squares have two additional small squares of zero elements to accommodate the interchange.

**TREATMENT MATRIX**

Decomposition is carried out in a matrix called TREATM which is three small squares high by five small squares long as shown in Figure 6.3a. This matrix is decomposed into upper and lower factors along the leading diagonal as far as the end of the second column and second row of small squares as shown in Figure 6.3b. Each element row is exchanged with the row with the largest element in the pivotal column, this exchange is limited to: from the pivot to the right. The elements to the left of the pivot are not interchanged. An integer vector called LOG with a subscript corresponding to the pivotal column number and whose value is equal to the position of the row which had the largest element in that column, was used to record interchanges.

The remaining elements in the pivotal column are the ratio of the element which was in that position to the pivotal element. These elements, which are less than unity (or equal to unity), can be considered as multiplying factors. In fact, this factor from a particular row, is multiplied by the elements in the pivotal row to the right of the pivot and subtracted from the corresponding elements in that factor's row.

Thus in the columns below the pivot are the multiplying factors; this is the lower factor of the matrix; and the pivot and to the right of the pivot is the upper factor.

**WRITING TO OUTPUT TAPES**

Two magnetic tapes are used, one for the lower factor and one for the upper factor, because they are used independently in the next stage.
The lower factor is written in column of small squares on to its tape (Figure 6.4a) and the upper factor is written in rows of small squares on its tape (as shown in Figure 6.4b). In general there are three rows of small squares in the first column and two in the second column of the lower portion; whereas on the upper tape there are five columns in the first row and four in the second. Thus both include the small squares containing the pivots.

The operation of the lower factor and interchanges prior to the lower factor; therefore the vector recording the row interchanges is written only on the lower tape and before the lower factor.

**TRANSFERRING PARTS OF MATRIX**

When the treated part of the matrix TREATM, (see Figure 6.3b) comprising the first two columns and first two rows of small squares, have been written to tape; then remaining part (which does not yet contain pivots), ie the last three small squares of the last row, is transferred to become the first three squares of the first row (as shown in Figure 6.5). This is done by subroutine TRANSF.

**READING FROM DISC**

When the first three small squares are in position in matrix TREATM then the next dozen small square are read from disc. The successive dozens are shown in Figure 6.6. The reading of the next dozen small squares is done by the subroutine NEXTDOZ. The reading of the very first three small squares is by the subroutine FIR3SQ.

**GENERAL PROCEDURE**

Recapitulating, first the input magnetic tape containing the assembly matrix (as in Figure 6.1) is read and transferred to disc with auxiliary zeros (Figure 6.2).

Next the first three small squares are read off disc into the treatment matrix. Then the cycle is begun as follows:
Read the next dozen small squares from tape (see Figure 6.6) into the treatment matrix (Figure 6.3a) and the lower and upper factors written on to the output tapes (Figures 6.4'). The remaining end portion of the treatment matrix is transferred to the beginning (Figure 6.5) and then the next dozen squares are read and so on.
FIGURE 6.1
APPENDIX 3b

The Sequence of Subroutines used in Computer Program "DECOMPOSITION"

1. **TAPDIS**
   The input magnetic tape to this program is the output from the program "Assembly Disc" and contains a series of small square partitioned matrices. These small square matrices are read, first three small squares, then five small squares each time adding two extra small squares and writing all these to magnetic disc for easy access.

2. **ZESMSQ**
   Of the small squares which are added, all the matrix elements (rows and columns) are zeroed.

3. **FIR3SQ**
   After all the magnetic tape is read and put on to disc (with the additions) then the first three small square partitioned matrices are read from disc and put in a larger treatment matrix called TREATM which is equal in size to three by five partitioned matrices (that is equivalent to fifteen small squares). These first three small squares are placed in the first three positions of the treatment matrix.

4. **ZETRET**
   The treatment matrix has all its elements (rows and columns) made equal to zero before

5. **ADTRET**
   Adding the small square matrices to the appropriate positions.

6. **REDSTA**
   The number of displacement positions to be restrained and the indices of those positions are read in from cards.
   The following sequence of steps is repeated a number of times:

7. **NEXDOZ**
   The next dozen small square partitioned matrices are read from disc and put into the large treatment matrix thus filling it up.

200
8. **RESTRA**

   The restraints are applied where appropriate,

9. **ZEROWI**

   This includes zeroing a whole row (index $l_0$) except putting $10^{20}$ in at $A(l_0,l_0)$.

10. **LUDEFV**

    After the restraint(s) are applied, lower-upper decomposition of the treatment matrix (with partial pivoting) may begin. The partial pivoting involves interchanging the row with the largest element in the pivotal column to become the pivot and keeping a record of the location of row interchanges in the integer array called LCC. For the $J$th pivotal column the following procedure is adopted:

    (a) find the largest matrix element in column $J$ located at row $L$
    (b) interchange the remaining part of row $L$ and row $J$ from the $J$th element to the $N_J$th element (where $N_J$ is the number of columns)
    (c) a record of index $L$ is kept in LCC ($J$)
    (d) a multiple of the remaining part of the pivotal row from the $J + 1$th element is subtracted from each of the remaining rows ie $J + 1$ to $N_I$ (where $N_I$ is the number of rows) such that it would have eliminated all the elements in the $J$th column had they been included in the subtraction; but instead the multiples are stored in the $J$th column.

    This process (a to d) is repeated for each column, ie each value of $J$, in turn from 1 to $N_{JS}$ (where $N_{JS}$ stands for the number of columns searched. $N_{JS}$ is usually equal to twice the width of the small square partitioned matrix.

11. **WRTAP**

    After decomposition of the current treatment matrix, there are writes to the magnetic tape as follows: the array LCC containing a record of integers indicating the indices of row interchanges are written onto the first magnetic
tape called LOWTAP. (The next write to this is indicated in the next paragraph).

12. **WTRETJ**

The lower portion of the treatment matrix, (which contains the multipliers) is written on to the first magnetic tape LOWTAP in the form of columns of small partitioned square matrices.

13. **WTRETJ**

The upper portion of the treatment matrix is written onto the second magnetic tape called IUPTAP in the form of rows of small partitioned square matrices.

14. **ZEORES**

After the lower and upper portions of the treatment matrix have been written onto tape, the restraint element A (Io, Io) which was previously made equal to $10^{120}$, needs to be zeroed if it occurs within two row widths of the small partitioned squares. The reason for this is that on a particular compiler on Atlas it would not over write this figure.

15. **TRANSF**

Also, after the data has been written onto tapes, the last three small partitioned square matrices of the treatment TREATM are transferred into the position of the first three.

This process is then returned to step (7) where the next dozen small square partitioned matrices are read from disc into TREATM and the process up to and including step (15) is repeated for all the chain. Then:

16. **WTAPEN**

The last small partitioned square matrix is written onto both the magnetic tape LOWTAP (for multiples) and the magnetic tape IUPTAP (for the upper portion).
LIST OF SUBROUTINES USED IN DECOMPOSE

(Except printing subroutines etc)

<table>
<thead>
<tr>
<th>DECOMP</th>
<th>SUBROUTINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>TAPDIS</td>
</tr>
<tr>
<td>2.</td>
<td>ZESMSQ</td>
</tr>
<tr>
<td>3.</td>
<td>FIR3SQ</td>
</tr>
<tr>
<td>4.</td>
<td>ZETRET</td>
</tr>
<tr>
<td>5.</td>
<td>ADTRET</td>
</tr>
<tr>
<td>6.</td>
<td>REDSTA</td>
</tr>
<tr>
<td>7.</td>
<td>NEXDOZ</td>
</tr>
<tr>
<td>8.</td>
<td>RESTRA</td>
</tr>
<tr>
<td>9.</td>
<td>ZEROWI</td>
</tr>
<tr>
<td>10.</td>
<td>LUDEPV</td>
</tr>
<tr>
<td>11.</td>
<td>WTOTAP</td>
</tr>
<tr>
<td>12.</td>
<td>WTRTJ</td>
</tr>
<tr>
<td>13.</td>
<td>WTREI</td>
</tr>
<tr>
<td>14.</td>
<td>ZEÖRES</td>
</tr>
<tr>
<td>15.</td>
<td>TRANSF</td>
</tr>
<tr>
<td>16.</td>
<td>WTAPEN</td>
</tr>
</tbody>
</table>
APPENDIX 4

LOWER UPPER TREATMENT

The decomposition is performed in a separate computer program to that of lower and upper treatment. They are in separate programs not merely from storage considerations, but because the decomposed matrix can operate on any number of right-hand side vectors B; which is equivalent to all possible loading conditions for the same structure.

The procedure with the Lower Upper Treatment is that the whole of the lower tape is processed with its operations. The lower factors are preceded on tape by the LOG vector which determines row interchange.

INPUT OF LOADING VECTOR

The vector B is the right-hand side of the simultaneous equations. Initially the whole of the vector B is given zero values. The non-zero values of B are read in off cards and overwrite the zero values.

LOWER TAPE

One stage of the lower-tape is read. A portion of the vector B is taken; rearranged as specified by vector LOG, then operated on by the lower factors. The next portion of the lower-tape is read and operates on the next portion of the vector B and so on, until the end of the lower-tape.

UPPER TAPE

After the lower treatment has been completed then the upper treatment is started. The upper tape is read from the end backwards, so that the last portion of the upper-factors act on the last portion of the B vector, starting at the end and in the form of backward substitution.

When this is complete the vector B contains the orthogonal displacements off all the nodes.

OUTPUT TAPE

The nodal displacements are written on magnetic tape. This tape is first verified by the program DISPLACEMENT CHECK and is then input to the program which calculates strains and stresses.

204
SEQUENCE OF SUBROUTINES USED IN COMPUTER PROGRAM LOWER-UPPER TREATMENT

The program obtains displacements $X$ from loads $B$ by interchanges, operating with the lower factors and the upper portion of the decomposed bonded matrix acting on $B$, in the following manner.

1. **ZEROVE**
   The whole of the vector $B$, whose length is equal to the total number of possible displacements (radial and axial) of all the nodes in the network of finite elements covering the body, is zeroed.

2. **READVE**
   The magnitude of each applied load (radial or axial in direction, positive or negative in sense) together with its position (indicated by an index equivalent to the displacement index) is read in. Note that the program checks to see if a load has been read in previously for that same position. If so the program stops. There is one card for each load and position, the order of these cards being unimportant. In fact in test runs, to test this point, the cards were deliberately shuffled. The program continued to read the load cards, keeping a count of the number of loads read in, until a zero in the load position index was reached. The zero index card must be the last card.

3. **PRNVEC**
   The numbers, positions, and the magnitudes of the loads are printed out for the whole array was read in.

   The following sequence from (4) to (8) for interchanging and lower treatment for each index along the meridian. (This index being equivalent to the number or index of the large square matrix, (described in the section on "Assembly Disc") which has been reached.)

4. **RELOTA**
   One of the arrays LOC which gives the location of the row interchanges, and a group of the small square partition matrices are read from the first magnetic tape LOWTAP.
5. **ADRET**

These small square partitioned matrices are added into the appropriate position of the treatment matrix (called **TREATM**) whose height is twice the height of a small square matrix and whose width is five times the width of a small square matrix.

6. **PUTTOY**

An appropriate portion of the whole force vector is transferred to **Y**.

7. **LOWTRI**

Inside a subroutine, **Y** is called **B** and for the column index **J** (where **J** has integer values from 1 to **NJ**) the following two steps are taken:

(a) **K**, which is the index stored in the **J**th position of the location array **LOC**, is read and then array elements **B(J)** and **B(K)** are interchanged, so that **B(K)** becomes **B(J)** and **B(J)** becomes **B(K)**;

(b) having done this interchange then for each row index **I** an appropriate multiple **A(I, J)** of **B(J)** is subtracted from **B(I)**. Those two steps ((a) and (b)) are repeated for each value of **J** from 1 to **NJ**.

8. **PUTTOB**

The treated portion of the force vector **B** is transferred back into the appropriate position of the whole force vector **B**.

This lower treatment process is repeated until the whole of the force vector has been treated in this way for all the large square indices.

Now the following is the upper treatment for each index along the meridian.

9. **RUJTA**

A group of small square partitioned matrices representing part of the upper-half of the banded matrix is read from the magnetic tape called **TUPTAP**, ready for the backward substitution. This is done by reading off the last row first time through and thereafter two previous rows of small square matrices.
This subroutine is particularly concerned with reading the data from the correct position off the magnetic tape.

These small square matrices are added into the correct position of the treatment matrix TREATM.

10. **UPFTRI**

The next step is to find the displacements $X$ such that $UX = Y$.

Starting from the Nth row in steps back to the Kth row it proceeds as follows; as the last row (i.e. the Nth row) has only one coefficient $(A(N,N))$ this may be divided into the force $B(N)$ to give the Nth displacement which is then stored in $B(N)$. Then in each row (Index I) after that, the products of these displacements and their respective coefficients are subtracted from the force $(B(I))$ relevant to that row. After completing the subtraction for that row, the remaining force is divided by the coefficient $(A(I,I))$ on the diagonal (the only coefficient without a known displacement) and this gives the displacement for that row, this being stored in $B(I)$.

Then the treated portion of the vector $B$ (which is now displacements) is transferred back into the whole vector $B$ which started as forces.

This process from (9) is repeated until the whole of the vector $B$ has been treated in this way. It now contains all the displacements (radial and axial) for all the nodes of the network covering the body.

This displacement vector is now written on to the magnetic tape called IDISPL.
LIST OF PRINCIPAL SUBROUTINES USED IN LOWER-UPPER TREATMENT

(Except printing subroutine etc)

LOWPT
1. ZEROVE
2. READVE
3. PRNVEC
4. RELOTA
5. ADTRET
6. PUBTOY
7. LOWTRI
8. PUTTON
9. REIUTA
10. UPTRI
APPENDIX 3

DISPLACEMENT CHECK

The stiffness matrix $A$ (which was formed in the program ASSEMBLY DISC) having been restrained from at least one movement (at least one node) was decomposed into lower and upper parts (in program DECOMPOSE). Consider the equation $A \mathbf{x} = \mathbf{b}$, where $\mathbf{b}$ is the vector of loads applied at various nodes (which are read into program LOWER-UPPER TREATMENT), this equation is solved (by LOWER-UPPER TREATMENT) for the vector of displacements $\mathbf{x}$.

During the processing of solving the simultaneous equations $A \mathbf{x} = \mathbf{b}$ for $\mathbf{x}$, errors could occur; for example, round off errors will occur and their magnitude will be dependant on the word length of the machine, and on the large number of operations (proportional to the cube of the band width of the matrix). Other errors may occur dependent on the computer and the particular compiler used on that machine.

It is therefore necessary to premultiply the displacement vector $\mathbf{x}$ by the stiffness matrix $A$, to obtain the force vector $\mathbf{b}$ and to compare these forces so obtained with those applied (feed into program LOWER-UPPER TREATMENT). The program which does this is called DISPLACEMENT CHECK. This program reads the magnetic tape (stream 14) which contains the stiffness matrix $A$ from the program ASSEMBLY DISC; and the magnetic tape (stream 17) which contains the displacement vector from the program LOWER-UPPER TREATMENT. $A$ is a banded matrix. The program DISPLACEMENT CHECK takes portions of $A$ and corresponding portions of $\mathbf{x}$ (these portions being equivalent to a physical row of triangles) and multiplies them together to obtain the portion of force vector $\mathbf{b}$, until the whole of the force vector $\mathbf{b}$ is obtained.

This is a necessary check (which is advocated by Mr C Snell of Nottingham University) but it is not a sufficient check. It would not for example reveal any error in the stiffness matrix $A$; also more than one vector $\mathbf{x}$ can satisfy the equation $A \mathbf{x} = \mathbf{b}$. 

209
APPENDIX 5b

Subroutines used in the computer program "DISPLACEMENT CHECK"

Most of the subroutines used in this program are identical with those (with the same name) in the program "DECOMPOSE", the two exceptions being the subroutine CEQDIS (which is effectively the main segment) and the subroutine MULTAX. Certain features of these two will be mentioned here.

CEQDIS  this program is to multiply the assembled stiffness of the structure times the displacements to obtain the nodal loads. It reads in the whole vector of displacements of the magnetic tape called IDISPL. It causes pieces of the stiffness matrix to be read off the magnetic tape called IASSEM and placed in a treatment matrix which is three small partitioned squares high by five small partitioned squares wide. It causes parts of the displacement vector to be taken to be used in the subroutine MULTAX. It specifies the integers to be used in that subroutine; which are:

ni  which is always equal to three times the height of the small partitioned square matrix,
is = 1  the first time round and thereafter
is = 1  + height of the small partitioned square matrix,
nj = five times the width of the small partitioned square matrix until at the last large square matrix,
nj = three times the width of the small partitioned square matrix.

MULTAX.  This subroutine is to multiply part of the stiffness matrix A by part of the displacement vector X to obtain the load vector Y

For  i = is , ni
\[ y_i = \sum_{j}^{nj} a_{ij} x_j \]
Sequence of titles of subroutines used in computer program "DISPLACEMENT CHECK"

(Excluding the printing subroutines)

CHQDIS

TAPODIS

ZESMSQ

FIR3SQ

ZETRET

ADTRET

NEXDOZ

(ADTRET)

PUETCY

MULTAX

PUYTOB

TRANSF
APPENDIX 6a

Derivation of the Least Square method of fitting a continuous surface to a number of discrete values at nodes

Nomenclature

Let

\( u = \) a column vector of displacements
\( a = \) is a column vector consisting of the coefficients of the displacement
\( C = \) coefficient matrix consisting of powers and products of \( r \) and \( z \)
\( e = \) is a column vector consisting of differences between displacements at nodes and those predicated by displacement function

Condition

\[ u = C \cdot a + e \quad \text{subject to } e^T \cdot e \text{ being a minimum} \]

so that \( e = C \cdot a - u \), Let \( B = C \cdot a \)

\[ (e)^T = (B - u)^T = B^T - u^T = (C \cdot a)^T - u^T = e^T \cdot C^T - u^T \]

\[ \text{(is } e^T \cdot C^T = (Ca)^T ) \]

\[ e^T \cdot e = (a^T \cdot C^T - u^T) (Ca - u) \]

Multiplying out

\[ e^T \cdot e = a^T \cdot C^T \cdot C \cdot a - a^T \cdot C^T \cdot u - u^T \cdot Ca + u^T \cdot u \]

as \( (a^T \cdot C^T \cdot u)^T = u^T \cdot Ca \) but is a scalar, is a single number

so that \( e^T \cdot e = a^T \cdot C^T \cdot C \cdot a - 2a^T \cdot C^T \cdot u + u^T \cdot u \)

differentiating \( \frac{\partial e^T \cdot e}{\partial a} = 2a \cdot C^T \cdot Ca - 2a \cdot C^T \cdot u = 0 \) for maximum or minimum

\[ \therefore C^T \cdot Ca = C^T \cdot u \]

\[ a = (C^T \cdot C)^{-1} \cdot C^T \cdot u \]
The matrix formulation of the least squares solution of inconsistent linear equations (from correspondence from T. Fuller on 28th September, 1970)

Column of residuals = $Ax - b$ where

$A$ is $m \times n$, $m > n$ or at least rank ($A$) + rank ($A:b$)

Sum of squares of residuals

$= (Ax - b)^T (Ax - b)$

$= x^T A^T A x - x^T A^T b - b^T A x + b^T b$

$= x^T (A^T A) (A^T A)^{-1} A^T A x - x^T (A^T A) (A^T A)^{-1} A^T b - b^T A (A^T A)^{-1} (A^T A) x + b^T A (A^T A)^{-1} A^T b + b^T b$

$= [x^T A^T A - b^T A] (A^T A)^{-1} [A^T A x - A^T b] + b^T b - b^T A (A^T A)^{-1} A^T b$

provided $A^T A$ is non-singular

$= x^T b x + b^T [I - A (A^T A)^{-1} A^T] b$

$= b^T [I - A (A^T A)^{-1} A^T] b$

provided $B = (A^T A)^{-1}$ is positive definite, and the minimum is attained for $A^T A x = A^T b$ (normal equations). This avoids differentiating w.r.t. as column or row.

It turns out that $A^T A$ is non-singular and positive definite provided rank ($A$) = $n$ (<m), i.e. provided the 'equations' are independent.
Method of determining best order of fit

Analysis of strain distributions occurring in the region of high strain gradient were carried out by the author in 1962 in order to determine which order of polynomial best fitted the photoelastic observations.

The example of strain gradient considered was that occurring near the boundary of a circular hole in a tension specimen of rectangular section. The tension specimen from which observations were taken, had a circular hole (centrally placed) whose diameter was equal to half the width of the specimen. The highest strain gradient occurs on a plane lying across the centre of the hole normal to the axis of the specimen. Observations of isochromatic fringe orders \( F \) were taken at incremental distances \( x \) from the boundary of the hole, proceeding away from the hole along a straight line normal to the axis. Thus points representing fringe order \( F \) may be plotted (ordinates) against the abscissa \( x \). "\( x \)" equals zero at the surface of the hole. A curve of the form:

\[
F_{o} = a_{0} + a_{1}x + a_{2}x^{2} + a_{3}x^{3} + \ldots + a_{n}x^{n}
\]

may be fitted to the results. The difference between \( F \) and \( F_{o} \) at any point is called a residual. A computer program called "Polyfit" (written by Mrs Joan Knock (nee Bartlett)) was used to fit polynomials of 2nd order to 10th order by the method of least squares to the observations of \( x \) and \( F \).

To facilitate interpretation of the results from "Polyfit" data was simulated. A particular order polynomial was taken, for example:-- using the expression

\[
y = 12.5 - 6.25x + 1.25x^{2} - 0.125x^{3} + 0.00625x^{4} - 0.000125x^{5}
\]

for values of \( x \) from 0.1 to 0.9 in 0.1 increments the values of \( y \) were calculated. To these values of \( y \), numbers representing the errors in observation were added or subtracted, forming simulated fringe values. These error numbers
varied in a binomially distributed random manner having a standard deviation of 0.0316 fringes. These simulated fringe values \( F \), together with their corresponding \( x \) values, were used as data for the program "Polyfit" which was run for the orders from 2 to 7. This process was repeated starting with polynomial expressions of various orders, for each polynomial adding separately five sets of simulated random errors. For every set of these simulated \( x \) and \( y \) values the results from "Polyfit" were examined and the standard Deviation (SD) of the residuals was plotted against the order of the polynomial fit on log-linear graph paper. The SD was plotted on the log scale and the order of fit on the linear scale of the log-linear graph paper. These graphs showed that with increasing order the SD of the residuals generally decreased. The exception occurred at the order of polynomial from which the results were simulated. In the region of that order the decrease in standard deviation was reduced or even reversed. In other words these graphs looked like cooling graphs of temperature against time where a "plateau" (point of inflexion or stationary value) occurs when there is a change of state. In the "SD against order" graph the centre of this plateau indicated the correct order of fit in these simulated results.

When photo-elastic observations from the tensile specimens were used as input data for the program Polyfit and the standard deviations of the residuals plotted against order as in the simulated cases it was found that the centre of the "plateau" occurred at the 4th or 5th order. Thus suggesting that the 4th or 5th order was the best fit.
Demonstration that 5th order 25 points give a singular Matrix, and that 4th order 23 points is non-singular. (From Dr J H Wilkinson FRS, 14 May 1971)

(Assuming equal intervals in y)

1. The problem in this. Suppose our matrix is an \((m \times n)\) matrix with \(n < m\). If rank = \(n\) then \(C^T C\) is non-singular. If rank < \(n\) then \(C^T C\) is singular. We want to show that with degree 5 and 25 points for which we have \(m = 25\), \(n = 21\) that the rank is < 21. It is easy to prove but the notation is tricky.

Remarks (i) We can alter the ordering of row and columns of \(C\) to suit our convenience (ii) We can subtract one row of \(C\) from another without altering the rank. We shall do this many times.

To illustrate it, consider the case of 16 points and degree 2. The matrix with suitable ordering of rows and columns is:--
where $X(i,j)$ is on $(i,j)$ matrix of the obvious type.

Subtracting the appropriate rows the matrix is of the same rank as

<table>
<thead>
<tr>
<th>$X(4,3)$</th>
<th>$y_1 \times (4,2)$</th>
<th>$y_1^2 \times (4,1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$(y_2-y_1) \times (4,2)$</td>
<td>$(y_2^2-y_1^2) \times (4,1)$</td>
</tr>
<tr>
<td>0</td>
<td>$(y_3-y_2) \times (4,2)$</td>
<td>$(y_3^2-y_2^2) \times (4,1)$</td>
</tr>
<tr>
<td>0</td>
<td>$(y_4-y_3) \times (4,2)$</td>
<td>$(y_4^2-y_3^2) \times (4,1)$</td>
</tr>
</tbody>
</table>

Subtracting rows again and remembering that second differences of $y_{\frac{1}{2}}^2$ are zero we get:-
and continuing we get

<table>
<thead>
<tr>
<th>$x_{(4,3)}$</th>
<th>$y_1 x_{(4,2)}$</th>
<th>$y_1^2 x_{(4,1)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$(y_2 - y_1) x_{(4,2)}$</td>
<td>$(y_2^2 - y_1^2) x_{(4,1)}$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>$(y_3^2 - 2y_2^2 + y_1^2) x_{(4,1)}$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>$(y_4^2 - 2y_3^2 + y_2^2) x_{(4,1)}$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

since 3rd differences of the $y_1^2$ are zero

In the same way dealing with the first four rows of this we reduce them to

<table>
<thead>
<tr>
<th>$x_{(4,3)}$</th>
<th>$y_1 x_{(4,2)}$</th>
<th>$y_1^2 x_{(4,1)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$(y_2 - y_1) x_{(4,2)}$</td>
<td>$(y_2^2 - y_1^2) x_{(4,1)}$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>$(y_3^2 - 2y_2^2 + y_1^2) x_{(4,1)}$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

(we use only the fact that the $k$ th differences of the $x_{1}^{k-1}$ are zero)

and dealing with the next 4 rows we reduce them to

<table>
<thead>
<tr>
<th>0 0 0</th>
<th>X X</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>0 X</td>
<td>0</td>
</tr>
<tr>
<td>0 0 0</td>
<td>0 0</td>
<td>0</td>
</tr>
<tr>
<td>0 0 0</td>
<td>0 0</td>
<td>0</td>
</tr>
</tbody>
</table>
and the next 4 rows become

\[
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 & X \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

while the last 4 rows are already zero

The rank is \(3 + 2 + 1 = 6\) and hence \(C^T C\) is non singular
In the case, 25 points and 5th degree the final matrix is of the form

<table>
<thead>
<tr>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

In the case, 25 points and 5th degree the final matrix is of the form
The rank of this matrix is $5 + 5 + 4 + 3 + 2$ (from the first five rows, next five, next four etc). Rank is therefore 19 and not 21. \( C^T C \) is 'doubly' singular.

If you do the same thing for 25 point and degree 4 you will find you have 25 x 15 matrix and rank is 15. \( C^T C \) is non-singular.

Notice that if the \( x_i, y_i \) are random we do not get the vanishing differences and the null matrices do not arise.
APPENDIX 6e

STRAIN STRESS

The purpose of this program is to find the values of stresses at the "principal" nodes. A principal node is defined as a vertex node at which six triangles meet. In programs of other authors, at such a point, there would be six different sets of stresses i.e. one set in each of the six triangles and they use some averaging procedure; such an averaging procedure is not used in this program. In place of this, a superior procedure is used which maintains continuity of displacements, strains and stresses.

The first subroutine of this program (FORMBE) forms the $4 \times 4$ constitutive matrix $SE$ which is used much later in the program. The constitutive matrix is a matrix which when post-multiplied by the four strains, forms the four stresses. These four strains and the four stresses are: the hoop, the radial, the axial, and the radial-axial shear. The elastic constants: Young's Modulus and Poisson's Ratio are read in on a card into the program. The two Lame constants $\lambda$ and $\mu$ (= $G$ the Modulus of Rigidity) are evaluated in this subroutine to simplify the form of the matrix. This matrix is used for every principal node later in the program.

From a magnetic tape (stream 10) containing data which is produced by the program NODETAPE, each node with its radial and axial co-ordinates is read in and stored by subroutine READAZ. Next the magnetic tape (stream 17), from program LOWER-UPPER TREATMENT which contains the displacement for each displacement-number (there are two displacement numbers for each node) is read by subroutine REDISP. (This also reads from the magnetic tape: the number of spaces per radial width, the number of spaces per axial length, the number of radial lines of nodes along the axis, and the total number of nodes).

Next in the program the subroutine SPLITS separates the displacements into the radial ones ($U$) and the axial ones ($W$). These two vectors $U$ and $W$, which are both functions of $r$ and $s$ (the radial and axial co-ordinates) are dealt with separately in the remainder of the program.
Having the two sets of displacements, a continuous function which describes them in terms of $r$ and $z$ is required, which will define them over the area of concern. That area is the area of the six triangles meeting at the principal node. Subroutine FORNOD defines the node-number of the principal node and gives the node-numbers of the remaining eighteen nodes on the six triangles surrounding it. These eighteen nodes consist of: the mid-side nodes on the six sides which radiate from the principal node; the mid-side node on the sides (of the six triangles) which face the principal node; and the remaining six are the vertices of the hexagon (probably irregular) formed by the six triangles around the principal node.

The next subroutine COLLECT forms arrays of length 19 for the radial ordinates $X(19)$, the axial ordinates $Y(19)$, the radial displacements $U(19)$, and the axial displacements $W(19)$, which are appropriate to the nineteen nodes of the six triangles.

The following series of subroutines are first to fit a surface to the radial displacement (ie $U$ in terms of $X$ and $Y$) and then to fit a surface to the axial displacement (ie $W$ in terms of $X$ and $Y$) both by Method of Least Squares which is derived in a separate section of this paper.
Amongst these subroutines are FORMCE which generates the matrix \( C \), which relates the displacement vector \( U \) via its ordinates to the coefficients such that the matrix equation \( U = C^T \alpha + E \), (where \( E \) is the error vector, and \( C \) is a 19 x 15 matrix). This is based on the equation, that the calculated value of 
\[
U = a_1 + a_2x + a_3y + a_4x^2 + a_5xy + a_6y^2 + a_7x^3 + a_8x^2y + a_9xy^2 + a_{10}y^3 + a_{11}x^4 + a_{12}x^3y + a_{13}x^2y^2 + a_{14}xy^3 + a_{15}y^4
\]
where \( a_1 \) to \( a_{15} \) are the fifteen coefficients which are obtained from the Least Squares Fit of the nineteen pairs of the \( x \) and \( y \) co-ordinates with the nineteen displacement values \( U \) (or \( W \)), of the nineteen nodes within the domain of the principal node. (The above equation is a fourth order polynomial equation for fitting a surface \( U \) in terms of position indicated by \( x \) and \( y \)).

The subroutine TRANSC transposes the matrix \( C \) to form \( C^T \). The subroutine FRMCTC post-multiplies \( C^T \) by \( C \) to form \( C^T C \). (\( C^T C \) is a 15 x 15 matrix). The subroutine RECIPR inverts the subroutine \( C^T C \) to form \( (C^T C)^{-1} \). The subroutine CTICICT post-multiplies the (15 x 15) matrix \( (C^T C)^{-1} \) by the (15 x 19) matrix \( C^T \) to form the matrix \( (C^T C)^{-1} C^T \). The subroutine COEFFPA in conjunction with the subroutine OBTANA post-multiplies the (15 x 19) matrix \( (C^T C)^{-1} C^T \) by the displacement vector (19 values) \( U \) (or \( W \) as the case may be) and so obtain the vector of coefficients (15 values) = \( \alpha \).

Although the above outlines the general order of the subroutines there are certain refinements that have been found necessary to obtain results. The most important of these is:— before carrying out the operations, detailed in the previous paragraph, is to change the origin of the co-ordinates of the nodes to
Those of the principal node and to scale their distances from the principal node in units of a module. A module is defined as the distance of the nearest node to the principal node. There is a different module in the $x$ to the $y$ direction. This procedure (change of origin and scaling) is called by some people normalizing. It is performed by the subroutine ORTOPR. This altered the ratio of the largest element to the smallest element in the matrix $C^{-1}$ by an order of $10^{20}$ and so permitted inversion which otherwise might have been difficult. The change back of the $x$ and $y$ ordinates to their initial origin (one end of the centreline of the axi-symmetric body) and initial scale is performed by the subroutine BAGSTA.

As well as changing back the axis of the $x$ and $y$ values to their initial origin and scale; the values of the Least Squared fitted coefficients have to be changed back to their initial origin (one end of the centreline of the axi-symmetric body) and scale. This is done by subroutine CHNGEA.

Having obtained the 15 coefficients $A_U$ associated with the radial displacements $U$, and the 15 coefficients $A_W$ associated the axial displacements $W$ (both about their initial origin and scale), these two vectors are joined together (by subroutine FORMWA) to form a vector of 30 coefficients.

Using the radial $(R)$ and the axial $(Z)$ co-ordinates of the principal node, the matrix $D$ ($4 \times 30$) is evolved (by subroutine DISTRA) so that the vector of strains $= D \times A$, where $A$ is the vector of the 30 coefficients obtained by the Least Squares fit to the 4th order polynomial surface; and where

- Strain (1) = Hoop Strain $E(\theta, \theta) = U/R$
- Strain (2) = Radial Strain $E(R, R) = \frac{\partial U}{\partial R}$
- Strain (3) = Axial Strain $E(Z, Z) = \frac{\partial W}{\partial R}$
- Strain (4) = Radial-Axial Strain $E(R, Z) = \frac{\partial U}{\partial Z} + \frac{\partial W}{\partial R}$

The subroutine STRANA post-multiplying the matrix $D$ by the coefficient vector $A$ to obtain the four strains.
The subroutine STREPS post-multiplies the constitutive matrix SE 
\((4 \times 4)\) (mentioned at the beginning of this section) by the vector STRAIN \((4)\),
to obtain the vector STRESS \((4)\) which is

- **Hoop Stress** \(S(\theta, \theta)\)
- **Radial Stress** \(S(R,R)\)
- **Axial Stress** \(S(Z,Z)\)
- **Radial-Axial Stress** \(S(R,Z)\)

This procedure from the subroutine FORNOD onwards is repeated for
each principal node in the network of the structure.

Some explanation is perhaps required on why a 4th order polynomial
was used. Photo-elastic experiments in regions of stress concentrations
indicated that when the number of isochromatics \(P\) are plotted against (one
dimension) distance \(x\), that a fourth and sometimes a fifth order polynomial
was required to fit the observations. As the isochromatics were proportional
to the difference in principal stress, and as some strains are the differentials
of displacements, it was expected that a fifth or higher order would be required
when fitting a surface of displacements \(u\) against position of \(x\) and \(y\) (two
dimensions).

The method of deciding the fit of the number of isochromatics \(P\)
against \(x\), was to fit by method of least squares, all the orders from 1st order
to 10th order, and then to plot a graph of the standard deviation of the
residuals against the order of the fit. The residuals are the difference between
the observed value and the calculated value using the polynomial expression.
Generally this graph shows that with increasing order, the standard deviation of
the residuals decrease; this continues to the tenth order but at the correct
order (say 4th or 5th) this drop in standard deviation of residuals is reduced
(or even reversed). In other words this graph looks like a cooling graph, where
there is a plateau (point of inflexion) where there is a change of state. In
the SD against order graph the centre of this plateau indicates the correct
order of fit.
Returning to the finite element least squares, a fifth order was attempted using double precision on the computer STRETCH (equivalent to about 29 decimal significant figures and exponents ranging from $10^{+308}$ to $10^{-308}$) but the attempt was unsuccessful for a cartesian network but successful for a network based on random numbers.

In an appendix Dr. J. H. Wilkinson has kindly given a proof that a 5th order cartesian network produces a singular matrix.
The Sequence of Subroutines used in the Computer Program "STRESS-STRAIN"

1. STESTA
   This is really the main segment. The purpose of this program is to find stresses at principal nodes. (Principal nodes are vertex nodes where six triangles meet).

2. CONSTI
   This subroutine is to form the matrix which gives the constitutive relationship between stress $\sigma$ and strain $\varepsilon$, in the form
   
   \[ \{\sigma\} = [SE] \{\varepsilon\} \]

3. FORMSE
   This subroutine reads in Young's Modulus $E$ and Poisson's ratio $\nu$ to form the matrix (in terms of the first Lamé constant $\lambda$ and the second Lamé constant $G$ the modulus of rigidity) which when post multiplied by the axisymmetric strains will give the axisymmetric stresses.

4. REA
   This subroutine reads from a magnetic tape (NODTAP) the node numbers of the vertices and their respective radial and axial co-ordinates. It then calculates the node number of each mid-side and its respective radial and axial co-ordinates.

5. REDISP
   This subroutine reads the displacements (alternately radial and axial) from a magnetic tape (IDISPL).

6. SPLTDS
   This subroutine separates the radial and the axial displacements so that the radial displacement is stored in an array $U$ and the axial displacement is stored in an array $W$. 
7. **FORNOD**

For each position along the meridian (except the beginning and the end) and for each position across the wall thickness (except the bore and the 0/D) a principal node number is calculated by this subroutine. It also calculates for each principal node, eighteen node numbers within the domain of the principal node, thus making a total within an array of nineteen node numbers. The intention of forming a set of node numbers is to extract co-ordinates and displacements of a set of nodes around and including a principal node and then by method of least squares, to fit a surface to the displacements of that region in order to find the strains and stresses at that principal node.

8. **COLECT**

This subroutine extracts from the large arrays, the radial and axial co-ordinates and the radial and axial displacements for the nineteen nodes within a domain, in preparation for the least squares calculation.

9. **AINESS**

This subroutine calculates the four strains and four stresses for each principal node. First the coefficients of the surfaces are obtained, then the strains and then the stresses.

10. **FORMEA**

This subroutine uses a series of seventeen subroutines to obtain thirty coefficients: fifteen coefficients for the radial displacement surface, and fifteen coefficients for the axial displacement surface.

11. **FMCICT**

This subroutine uses nine subroutines to form a matrix $CT\left[C^TC\right]^{-1}$ which depends only on the co-ordinates of the nineteen nodes.

12. **ORTOPR**

This subroutine is to change the origin of the nineteen co-ordinates by making the principal node the origin, and to alter the scale of the ordinates by dividing by the largest distance from the new origin.
13. **FORMCE**

This subroutine generates the matrix \([C]\) which relates the displacements \([U]\) via their co-ordinates to the coefficients \([A]\), such that

\[
[U] = [C] [A] \times [E]
\]

where \([E]\) = a column array of errors.

Each of the nineteen rows of matrix \([C]\) is for a different pair of \((x,y)\) co-ordinates. For each of these rows, the fifteen columns are:-

1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3, x^4, x^3y, x^2y^2, xy^3, y^4

respectively. Thus the matrix \([C]\) is formed.

14. **BACSTA**

This subroutine is to return the nineteen pairs of co-ordinates back to their original dimensions and initial origin.

15. **TRANS C**

This subroutine transposes the matrix \([C]\) into \([CT]\), that is

\[
[CT] = [C]^T
\]

16. **FRMCTC**

This subroutine forms matrix \([CTC]\) by postmultiplying the matrix \([CT]\) with the matrix \([C]\)

\[
[CTC] = [CT]^T [C]
\]

17. **EQUIVA**

This subroutine is only used for checking purposes and so is not generally used. It is used to make the matrix \([D]\) equal to the matrix \([CTC]\), because in the next subroutine the matrix \([CTC]\) will be modified. (This subroutine is used in conjunction with subroutine MULMAT).
18. **RECIPR**
This subroutine inverts the 15 x 15 matrix \([CTC]\) by a special method using partial pivoting. It uses double precision. It forms:
\[
( [c]^T [c] )^{-1}
\]

19. **MULMAT**
This subroutine (used in conjunction with subroutine EQUIVA) is only used for checking purposes and so is not generally used. It forms the product:
\[
[F] = ([c]^T [c])^{-1} \times D
= ([c]^T [c])^{-1} \times ([c]^T [c])
\]
which should be a unit matrix.

20. **CTCICT**
This subroutine premultiplies the matrix \([CT]\) by the inverted matrix \([CTC]\) that is:
\[
[CTC] = ([c]^T [c])^{-1} [c]^T
\]
The resulting matrix is now in the form which if it premultiplies any set of displacements for the nineteen nodes (whose co-ordinates have been used in its formation) the product will be the fifteen coefficients necessary to define a full fourth order surface (i.e., the polynomial is not curtailed in any way). Therefore the same matrix is used both for radial displacements and for the axial displacements.

21. **COEFFA**
This subroutine uses a series of five subroutines to obtain the fifteen coefficients \(\{A\}\), to verify them and to change their origin and dimensions.

22. **OBTANA**
This subroutine premultiplies the nineteen nodal displacements \(\{U\}\) by the matrix \(( [c]^T [c] )^{-1} [c]^T\) to obtain fifteen coefficients \(\{A\}\) which will be used later to obtain strains and thence stresses.
23. **OBTNUC**

This subroutine is to calculate the "best fit" values (on the basis of least squares) of the nineteen nodal displacements \{UC\}. This is done by premultiplying the fifteen coefficients \{A\} by the 19 x 15 matrix \(C\) thus

\[
\{UC\} = [C] \{A\}.
\]

24. **DIFFER**

This subroutine is to find the difference between the nineteen nodal displacements from the six triangles within the domain of the principal node and the displacements calculated for those nodes as heights of the least square fitted surface. (That surface has been fitted to the very displacements whose difference is now being sought). The smaller that difference the better the fit.

25. ** XBARS**

This subroutine calculates the mean difference (which would be expected to approach zero) and the standard deviation of the differences calculated in **DIFFER**. It uses Bessels correction in calculating the standard deviation.

26. **CHNGEA**

The fifteen coefficients \{A\} obtained, so far, have been based on distances from the principal node and in units of a convenient module. These now have to be changed back so that they are consistent with the original units of measurement and to their true origin. This is done in this subroutine.

27. **FORMWA**

To obtain the fifteen coefficients \{AU\} for the radial displacements surface the subroutine **COEFFPA** was used, and it was used again to obtain the fifteen coefficients \{AW\} for the axial displacements surface. In the present subroutine **FORMWA** the two sets of coefficients \{AU\} and \{AW\} are combined to form one set of 30 coefficients \{A\}. 
This subroutine uses the following two subroutines ((DISTRA & STRANA) to calculate the strains at the principal node (its co-ordinates being R & Z).

29. DISTRA

This subroutine is to obtain a 4 x 30 matrix $[D]$ such that when this matrix premultiplies the column vector $\{A\}$ (of length 30) the four strains $\{e\}$, hoop, radial, axial, and radial-axial are obtained. That is:

$$[D]\{A\} = \{e\}$$

In order to do that, use is made of two matrices $[B]$ and $[C]$:

- $[B]$ which is 3 x 15 and is related to the radial displacements $U$; and
- $[C]$ which is 2 x 15 and is related to the axial displacements $W$.

The fifteen values of $B(1,j)$ are related to the hoop strain.
The fifteen values of $B(2,j)$ are related to the radial strain.
The fifteen values of $C(1,j)$ are related to the axial strain.
The fifteen values of $B(3,j)$ are related to $\partial W/\partial z$.
The fifteen values of $C(2,j)$ are related to $\partial W/\partial r$.

Advantage is made at this stage of the fact that $\partial / \partial z$ of the products of powers of $r$ and $z$ (before the coefficients $\{A\}$ are involved) are independent of $\{U\}$ or $\{W\}$. The same thing applies to $\partial / \partial r$. Thus, the number of calculations which need to be performed are minimised. $[D]$ is built up from $[B]$ and $[C]$.

30. STRANA

This subroutine is to obtain the four components of axisymmetric strain viz

$$e_{00}, e_{rr}, e_{zz}, e_{rz}$$

This is done by premultiplying the thirty coefficients $\{A\}$ by the matrix $[D]$ obtained in the previous subroutine

$$\{e\} = [D]\{A\}$$
This subroutine is to obtain stress from strain. The four axisymmetric stress components

\[ \sigma_0, \sigma_{rr}, \sigma_{zz}, \sigma_{rz} \]

are obtained from the four axisymmetric strains

\[ e_0, e_{rr}, e_{zz}, e_{rz} \]

by premultiplying the vector of strains by the matrix which represents the elastic constitutive relationships i.e.

\[ \{\sigma\} = [SE] \{e\} \]

where SE is the 4 x 4 constitutive matrix obtained by the subroutine FORMSE.
List of the titles of 31 of the 41 subroutines used in program "STRESS-STRAIN"
(Excluding printing and punching subroutines)

<table>
<thead>
<tr>
<th>Subroutine</th>
</tr>
</thead>
<tbody>
<tr>
<td>STESTA</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>19</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>21</td>
</tr>
<tr>
<td>22</td>
</tr>
<tr>
<td>23</td>
</tr>
<tr>
<td>24</td>
</tr>
<tr>
<td>25</td>
</tr>
<tr>
<td>26</td>
</tr>
<tr>
<td>27</td>
</tr>
<tr>
<td>28</td>
</tr>
<tr>
<td>29</td>
</tr>
<tr>
<td>30</td>
</tr>
<tr>
<td>31</td>
</tr>
</tbody>
</table>
APPENDIX 7

Derivation of Lame Equations

Let \( \sigma_{\theta\theta} \) = the hoop stress
\( \sigma_{rr} \) = the stress in the radial direction, (which is equal to the pressure at the bore),
\( \sigma_{zz} \) = the axial stress,

and let \( e_{\theta\theta} \) = the hoop strain,
\( e_{rr} \) = the radial strain,
\( e_{zz} \) = the axial strain,

then
\[
Ee_{\theta\theta} = \sigma_{\theta\theta} - \nu(\sigma_{rr} + \sigma_{zz}),
\]
\[
Ee_{rr} = \sigma_{rr} - \nu(\sigma_{\theta\theta} + \sigma_{zz}),
\]
\[
Ee_{zz} = \sigma_{zz} - \nu(\sigma_{\theta\theta} + \sigma_{rr}),
\]

where \( E \) = Young's Modulus of Elasticity
\( \nu \) = Poisson's Ratio.

For plane stress considerations:

axial stress \( \sigma_{zz} = 0 \)

then
\[
e_{zz} = -\frac{\nu}{E} (\sigma_{\theta\theta} + \sigma_{rr})
\]

and
\[
Ee_{\theta\theta} = \sigma_{\theta\theta} - \nu.\sigma_{rr}
\]
\[
Ee_{rr} = -\nu.\sigma_{\theta\theta} + \sigma_{rr}
\]

which may be written:
\[
E \begin{pmatrix} e_{\theta\theta} \\ e_{rr} \end{pmatrix} = \begin{pmatrix} 1 & -\nu \\ -\nu & 1 \end{pmatrix} \begin{pmatrix} \sigma_{\theta\theta} \\ \sigma_{rr} \end{pmatrix}
\]

Multiplying both sides by the inverse:
\[
\begin{pmatrix} \sigma_{\theta\theta} \\ \sigma_{rr} \end{pmatrix} = \frac{E}{1-\nu^2} \begin{pmatrix} 1 & \nu \\ \nu & 1 \end{pmatrix} \begin{pmatrix} e_{\theta\theta} \\ e_{rr} \end{pmatrix}
\]
so that:

\[ \sigma_{\theta\theta} = \frac{E}{1 - \nu^2} \left\{ \varepsilon_{\theta\theta} + \nu \varepsilon_{rr} \right\} \]

and

\[ \sigma_{rr} = \frac{E}{1 - \nu^2} \left\{ \nu \varepsilon_{\theta\theta} + \varepsilon_{rr} \right\} \]

As \( \varepsilon_{\theta\theta} = \frac{u}{r} \) and \( \varepsilon_{rr} = \frac{du}{dr} \)

\[ \sigma_{\theta\theta} = \frac{E}{1 - \nu^2} \left\{ \frac{u}{r} + \nu \frac{du}{dr} \right\} \]

and

\[ \sigma_{rr} = \frac{E}{1 - \nu^2} \left\{ \nu \frac{u}{r} + \frac{du}{dr} \right\} \]

\[ \sigma_{\theta\theta} - \sigma_{rr} = \frac{E}{1 - \nu^2} \left\{ (1 - \nu) \frac{u}{r} + (\nu - 1) \frac{du}{dr} \right\} \]

\[ \sigma_{\theta\theta} - \sigma_{rr} = \frac{E}{1 - \nu^2} \left\{ (1 - \nu) \left( \frac{u}{r} - \frac{du}{dr} \right) \right\} \]

As

\[ \frac{d}{dr} \left( \frac{u}{r} \right) = \frac{r \frac{du}{dr} - u \frac{dr}{dr}}{r^2} = \frac{r \frac{du}{dr} - u}{r^2} \]

it follows that

\[ \frac{d\sigma_{rr}}{dr} = \frac{E}{1 - \nu^2} \left\{ \nu \left( \frac{du}{dr} - \frac{u}{r} \right) + \frac{d^2 u}{dr^2} \right\} \]

From radial equilibrium (because of symmetry there are no shearing stresses):-

\[ \sigma_{\theta\theta} - \sigma_{rr} - \frac{r \cdot d\sigma_{rr}}{dr} = 0 \]

that is:

\[ \frac{E}{1 - \nu^2} \left\{ (1 - \nu) \left( \frac{u}{r} - \frac{du}{dr} \right) + \nu \left( \frac{du}{dr} - \frac{u}{r} \right) - r \cdot \frac{d^2 u}{dr^2} \right\} \]

\[ = \frac{E}{1 - \nu^2} \left( \frac{u}{r} - \frac{du}{dr} - r \cdot \frac{d^2 u}{dr^2} \right) = 0 \]

\[ \therefore \frac{d^2 u}{dr^2} + \frac{1}{r} \cdot \frac{du}{dr} - \frac{u}{r^2} = 0 \]

287
The general solution of this equation is

\[ u = C_1 r + \frac{C_2}{r} \]

which yields strains and stresses in the form:

\[ e_{\theta \theta} = \frac{u}{r} = C_1 + \frac{C_2}{r^2} \]

\[ e_{rr} = \frac{du}{dr} = C_1 - \frac{C_2}{r^2} \]

\[ \sigma_{\theta \theta} = \frac{E}{1 - \nu^2} \left[ (1 - \nu)C_1 + \frac{(1 - \nu)C_2}{r^2} \right] \]

\[ \sigma_{rr} = \frac{E}{1 - \nu^2} \left[ (1 - \nu)C_1 - \frac{(1 - \nu)C_2}{r^2} \right] \]

ie

\[ \sigma_{\theta \theta} = A + \frac{B}{r^2} \]

\[ \sigma_{rr} = A - \frac{B}{r^2} \]

where \( A = \frac{E}{1 + \nu} C_1 \), and \( B = \frac{E}{1 + \nu} C_2 \).

At the bore radius \( r = r_1 \) and \( \sigma_{rr} = -p = A - \frac{B}{r_1^2} \)

At the outside radius \( r = r_2 \) and \( \sigma_{rr} = 0 = A - \frac{B}{r_2^2} \)

\[ -p = -B \left( \frac{1}{r_1^2} - \frac{1}{r_2^2} \right) \]

\[ B = \frac{pr_1^2 r_2^2}{r_2^2 - r_1^2} \]

\[ A = \frac{pr_1^2}{r_2^2 - r_1^2} \]

Hoop stress

\[ \sigma_{\theta \theta} = \frac{pr_1^2}{r_2^2 - r_1^2} \left( \frac{r_2^2}{r_1^2} + 1 \right) \] \[ \sigma_{rr} = \frac{pr_1^2}{r_2^2 - r_1^2} \left( \frac{r_2^2}{r_1^2} - 1 \right) \]
Continuity of Displacements

As mentioned in the section on the computer program "STRAIN-STRESS", the principal node is defined as the vertex node where six triangles meet. The domain of the principal node is the region of those six triangles. In the program "STRAIN-STRESS", a 4th order polynomial, which defines displacement in terms of powers of the radial and axial co-ordinates, is fitted to the point displacements of the nineteen nodes within that domain. This is done for both the radial and axial displacements. The displacement may be considered as the height of a "surface" at any point whose radial and axial co-ordinates are $R$ and $Z$ respectively. The plan of such a "surface" is limited to the region of the domain being considered.

It is desirable that these displacement surfaces have continuity, in certain respects, one with another. At any point $(R, Z)$, let the radial displacement be $U$ and the axial displacement be $W$. As the hoop-strain = $U/R$, so the surfaces of radial displacement should have the same height $U$ where they meet or overlap. As the radial strain = $dU/dR$, so the surfaces of radial displacement should have the same slope in the radial direction where they meet or overlap. As the axial strain = $dW/dZ$, so the surfaces of axial displacement should have the same slope in the axial direction where they meet or overlap. These are the conditions of continuity of displacement which are examined in the following.

A thick cylindrical tube of 15.0 inches outside radius and 10.0 inches bore radius, 12.0 inches long, open ended, of Young's Modulus $E = 30,000,000$ lb/sq in. and Poisson's Ratio = 0.3, subjected to an internal pressure of 106.1 lb/sq in. was considered.

The cross section of the wall of this thick cylindrical tube was divided into a network of triangles as indicated in Figure C1 for the purposes of finite element calculations. As the axial pitch $s$ of the nodes was 1.5 inches, the area on which the pressure acts per "non-end" internal node =

$$2 \pi R s = 2 \pi \times 10.0'' \times 1.5'' = 94.25 \text{ sq in.}$$

Thus the total radial thrust on each...
of the seven non-end internal nodes (numbers 8, 15, 22, 29, 36, 43 and 50 on Figure C1) = 106,1 lb/sq in. x 94.25 sq in. = 10,000 lb thrust and half this amount of radial thrust at the two end internal nodes (numbers 1 and 57) is 5,000 lb thrust.

Using the computer programs up to the Lower-Upper Treatment program, where an axial restraint was applied to node 1; the individual radial and axial displacements were found for each of the sixty-three nodes shown on Figure C1.

By Method of Least Squares the computer program "STRAIN-STRESS", fitted a "surface" to the nineteen node point displacements of each of the six domains of the principal nodes, which are shown in Figure C1D, in turn. This process was applied to both the radial and the axial displacements.

The radial displacement surfaces for the domains of the six principal nodes are shown individually in figures CU17, CU19, CU31, CU33, CU45 and CU47. They are shown combined in figure CU. It can be seen, in figure CU, that the hoop-strain condition of continuity is satisfied, in that the height U (representing the radial displacement) does not alter appreciably in going from one domain to the next. It can also be seen from figure CU that the radial-strain condition of continuity is also satisfied; in that the slope in the radial direction (du/dr) does not alter appreciably in going from one domain to the next.

The five individual curves (representing a radial section through the surface) for the domain of principal node 17 are shown in the five figures CU17.1, CU17.2, CU17.3, CU17.4 and CU17.5. Similarly, for the domain of principal node 19 the curves are shown in figures CU19.1, CU19.2, CU19.3, CU19.4 and CU19.5. This procedure of numbering the curves is also used for principal nodes 31, 33, 45 and 47. On each of these figures the individual node point displacements are shown as well as the curve representing a section through the
surface at constant axial distance. It can be seen from these figures that the radial displacement surfaces are a good fit to the individual node point radial displacements.

Similarly, the axial displacement surfaces for the six domains of the principal nodes are shown individually in figures CW17, CW19, CW31, CW33, CW45 and CW47. They are shown combined in figure CW; where it can be seen that the axial strain condition of continuity is satisfied, in that the slope in the axial direction ($\partial W/\partial Z$) does not alter appreciably in going from one domain to the next. There are five individual curves (representing axial sections through the surface) for each principal node domain which are shown on five separate figures. For example the curves for the domain of principal node 17 are shown on figures CW17.1, CW17.2, CW17.3, CW17.4 and CW17.5. The same procedure of numbering the figures is used for the domains of principal nodes 19, 31, 33, 45, and 47. On each of these figures the individual node point displacements are shown as well as the curve representing a section through the surface at constant radial distance. It can be seen from these figures that the axial-displacement surfaces are a good fit to the individual node point axial displacements.

These figures demonstrate:

(a) the surfaces are a good fit to node point displacements and

(b) continuity is maintained between surfaces of adjacent domains.
FIGURE C.1. NETWORK OF TRIANGLES
FIGURE C.1D. DOMAINS OF PRINCIPAL NODES
SURFACE ASSEMBLY OF CURVES
POINTS WITH INDIVIDUAL CURVES
SHOWN ON FIGURES CU3I-1-CU31-5

FIGURE CU31
AXIAL DISTANCE

z = 2.0 INCHES
SURFACE = ASSEMBLY OF CURVES
POINTS WITH INDIVIDUAL CURVES
SHOWN ON FIGURES CW7.1 - CW7.5

SURFACE OF DOMAIN OF PRINCIPAL NODE

R = 11.979
R = 12.345
R = 11.976
R = 10.900

W

Z

K

0.9
5.1
0.3
5.4
SURFACE = ASSEMBLY OF CURVES
POINTS WITH INDIVIDUAL CURVES
SHOWN ON FIGURES CW 33.1 - CW 33.5
SURFACE = ASSEMBLY OF CURVES
POINTS WITH INDIVIDUAL CURVES
SHOWN ON FIGURES CW.45.1 - CW.45.5

SURFACE OF DOMAIN OF PRINCIPAL NODE 45

FIGURE CW.45
SURFACE ASSEMBLY OF CURVES.
POINTS WITH INDIVIDUAL CURVES
SHOWN ON FIGURES CW47.1-CW47.5
ASSEMBLY OF SURFACES
(Separate surfaces shown in
figures CW17, CW19, CW31, CW33, CW45, CW47)
Figure CW17.2 Axial displacement v axial distance for domain of principal node 17.
Figure CW17.3  Axial Displacement v Axial Distance for Domain of Principal Node 17
Figure CW 17.6 Axial displacement vs. axial distance for domain of principal mode 17.
APPENDIX 9

OPERATION NOTES ON COMPUTER PROGRAMS

The notes following are written only to aid a person running the programs, written briefly without going into detail about how and why the programs were written in that way.

The notes are on the following computer programs:

- NODE TAPE
- TRIANGLE TAPE
- XY ORD
- SINGLE LINE GRAPH
- ASSEMBLY DISC
- DECOMPOSE
- LOWER UPPER TREATMENT
- DISPLACEMENT CHECK
- STRESS STRAIN
The axisymmetric element is of triangular section. Thus a section through the object being analysed appears as a network of triangles. Each triangle is defined by the node numbers of its three vertices. To minimise the bandwidth of the stiffness matrix, node numbers must proceed across the narrowest width of the object. The node numbers of the vertices are such that the mid-side node has a number equal to the mean of its two adjacent vertex node numbers. There are thus six node numbers associated with each triangle, that is: three node numbers for the vertices, and three node numbers for the mid-sides.

Forming nodial numbers for Cartesian Network

In a cartesian network several nodes have a common radius, and several nodes have a common axial length. Each common radius in turn is indexed in order of size. Each common axial length in turn is indexed in order of size. Using the axial and radial indices, the nodial numbers are generated and assigned their radius and axial length.

Tracing Triangles

It is desirable to reproduce graphically the network of triangles (this will show errors in specification). The nodes need to be plotted in such a way that one continuous line scribes, in clockwise order, all the sides of all the triangles in sequence.

If the vertices form an orthogonal network, use can be made of this in forming the continuous. In order to do this use is made of the meridional and radial indices, but it is superfluous to specify the triangles by these indices, as they are specified by their vertex nodial numbers and this is sufficient.
**NODE TAPE**

This program generates the node numbers and assigns to each node number its respective radius and axial distance.

The output of this program goes to magnetic tape (stream 10): each write to tape gives a node number and the axial and radial co-ordinates of that node.

The input data required for this program is:

1. The number of radii which remain constant over the axial length,
2. the numbers of axial lengths which remain constant over the range of radii,
3. the values of the constant radii,
4. the values of the constant axial lengths.

Also the following data is required:

5. the triangle number ending the previous section,
6. an integer indicating the beginning radial location,
7. an integer indicating the beginning axial location,
8. an integer indicating the end radial location, and
9. an integer indicating the end axial location
Program Title  NODE TAPE

Input Magnetic tapes stream numbers:  None

Output Magnetic tape stream numbers:  10

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Cards of This Type</th>
<th>Information or data being read in on cards into computer</th>
<th>Format Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>only one</td>
<td>Integer indicating the number of radii which remain constant along the axis</td>
<td>I2</td>
</tr>
<tr>
<td>2</td>
<td>only one</td>
<td>Integer indicating the number of axial lengths which remain constant along the radial width.</td>
<td>I2</td>
</tr>
<tr>
<td>3</td>
<td>one or more</td>
<td>The values of radii which are constant along the length (Floating point) smallest to largest</td>
<td>8F10.4</td>
</tr>
<tr>
<td>4</td>
<td>one or more</td>
<td>The values of axial length which are constant through the thickness, smallest to largest</td>
<td>8F10.4</td>
</tr>
<tr>
<td>5</td>
<td>only one</td>
<td>The triangle number ending the previous section. Integer indicating the beginning radius location &quot; &quot; &quot; &quot; axial &quot; &quot; &quot; &quot; &quot; &quot; end radius &quot; &quot; &quot; &quot; axial &quot; in that order</td>
<td>5I5</td>
</tr>
</tbody>
</table>
PROGRAM TRIANGLE TAPE

Some errors of specification may be revealed by drawing the network of triangles. Three vertex nodes are sufficient to specify a triangle. Using only three vertex nodes in a clockwise order for each triangle in sequence it is required to draw the network by a continuous line. Each write to magnetic tape (stream 11) specifies a triangle number and the number of its three vertex nodes, in such a manner as to satisfy the continuous line condition. The input data to this program is entered by means of punched cards. On the first data card the number of common radii in width is specified. On the second card is specified the triangle number ending previous section, an integer representing the beginning radial location, an integer representing the beginning axial location, an integer representing the end radial location and an integer representing the end axial location.
Program Title **TRIANGLE TAPE**

Input Magnetic tapes stream Numbers: None

Output Magnetic tape stream Numbers: 11

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Cards of This Type</th>
<th>Information or data being read in on cards into computer</th>
<th>Format Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>only one</td>
<td>Total number of common radii</td>
<td>I2</td>
</tr>
<tr>
<td>2</td>
<td>only one</td>
<td>The triangle number ending previous section Integer indicating the beginning radial location &quot; axial &quot; &quot; end radial &quot; &quot; axial &quot;</td>
<td>5I5</td>
</tr>
</tbody>
</table>

In that order
This program uses the three node numbers for each triangle number from the program "Triangle Tape", with the radial and axial ordinates for each node number from the program "Node Tape": to output for each triangle its three pairs of radial and axial ordinates in clockwise order. Thus the input data to this program is from "Node Tape" on magnetic tape stream 10, and from "Triangle Tape" on magnetic tape stream 11. The card input data to this program consists of two cards: the first gives the number of nodes to be read in, and the second the number of triangles to be read in. The output from this program is on magnetic tape stream 12.
Program Title X-Y ORD

Input Magnetic tapes stream Numbers: 10 & 11
Output Magnetic tape stream Numbers: 12

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Card of This Type</th>
<th>Information or data being read in on cards into computer</th>
<th>Format Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>one only</td>
<td>The total number of nodes to be read in (Integer)</td>
<td>I5</td>
</tr>
<tr>
<td>2</td>
<td>one only</td>
<td>The total number of triangles to be read in (Integer)</td>
<td>I5</td>
</tr>
</tbody>
</table>
The Stromberg Carlson graph plotter (SG4020) is used by this program to draw a diagram of the network and so verify part of the input data. The magnetic tape (stream 12) input is from the program XY ord. (This gives the ordinates of the vertices of the triangles). The card input is a single card which gives the total number of triangles.
Program Title  **SINGLE LINE GRAPH**

Input Magnetic tapes stream Numbers:  12
Output Magnetic tape stream Numbers:  None

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Cards of This Type</th>
<th>Information or data being read in on cards into computer</th>
<th>Format Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>one only</td>
<td>The total number of triangles to be read in</td>
<td>15</td>
</tr>
</tbody>
</table>
ASSEMBLY DISC

This program assembles the axisymmetric stiffness on Disc and transfers the assembled matrix onto magnetic tape.

Firstly the maximum number of physical squares (or rectangles) across the radial thickness and the maximum number of rows of physical squares (or rectangles) along the axial length are read in off card (and printed out). These are the number of spaces between the axial lines and the number of spaces between the radial lines respectively. They have been run at 10 and 50. A number is read off the second card specifying the number of nodes to be read in off magnetic tape. Then these node with their ordinates are read off magnetic tape (stream 10). The third card specifies the Young's Modulus and the Poisson's Ratio of the material. The fourth card gives the number of triangles to be read in off magnetic tape (stream 11). In each read off tape: the triangle number is read in together with its three vertex nodes. As each triangle is read in its individual stiffness matrix is calculated and added to the stiffness matrix of the assembly which is stored on disc (stream 20). When this is complete it is transferred to magnetic tape (stream 14) as output from this program.
**Program Title**: ASSEMBLY DISC

**Input Magnetic tapes stream Numbers**: 10 & 11, Magnetic disc stream 20

**Output Magnetic tape stream Numbers**: 14

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Card of This Type</th>
<th>Information or data being read in on cards into computer</th>
<th>Format Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>only one</td>
<td>An integer indicating the number of spaces between the axial lines across the radial thickness and an integer indicating the number of spaces between the radial lines along the axis</td>
<td>E10,3, F16,4</td>
</tr>
<tr>
<td>2</td>
<td>only one</td>
<td>An integer indicating the number of nodes to be read off magnetic tape</td>
<td>E5</td>
</tr>
<tr>
<td>3</td>
<td>only one</td>
<td>Youngs modulus and Poissons Ratio</td>
<td>E10,3, F16,4</td>
</tr>
<tr>
<td>4</td>
<td>only one</td>
<td>An integer indicating the number of triangles to be read in.</td>
<td>E5</td>
</tr>
</tbody>
</table>
DECOMPOSE

This program is to decompose the assembly stiffness matrix into lower and upper matrices using partial pivoting having first applied the restraints. The first card is the same as the first card of ASSEMBLY DISC. The second card indicates how many restraints. The remaining cards, (at least one) each gives the individual number of the displacement position to be restrained (up to 16 per card). The input magnetic tape (stream 14) is from ASSEMBLY DISC and the output of the lower and upper matrix is on magnetic tapes (streams 16 and 15 respectively).
Program Title: **DECOMPOSE**

Input Magnetic tapes stream Numbers: 14

Output Magnetic tape stream Numbers: 15 & 16

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Card of This Type</th>
<th>Information or data being read in on cards into computer</th>
<th>Format Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>only one</td>
<td>An integer indicating the number of spaces between the axial lines across the radial thickness and an integer indicating the number of spaces between the radial lines along the axis</td>
<td>2I2</td>
</tr>
<tr>
<td>2</td>
<td>only one</td>
<td>An integer indicating the number of displacement positions to be restrained</td>
<td>I2</td>
</tr>
<tr>
<td>3</td>
<td>one or more</td>
<td>Integer(s) indicating the individual number(s) of the displacement position(s) to be restrained</td>
<td>16I5</td>
</tr>
</tbody>
</table>
LOWER UPPER TREATMENT

This program is to obtain displacements $X$ from loads $B$ by interchanges: operating with lower factors and upper portion of decomposed partition banded matrix acting on $B$. The input cards are of two types: the first is the same as the first card of ASSEMBLY DISC; the second type of card has an integer to represent the individual number of displacement position and the load at that position. There are a number of the second type of card. There are two input magnetic tapes (streams 16 and 15) with the lower and upper matrices on. The output on magnetic tape (stream 17), is the vector of displacements. A magnetic disc (stream 20) is used during the processing.
Program Title  LOWER UPPER TREATMENT

Input Magnetic tapes stream Numbers:  15 & 16
Output Magnetic tape stream Numbers:  17

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Cards of This Type</th>
<th>Information or data being read in on cards into computer</th>
<th>Format Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>only one</td>
<td>An integer indicating the number of spaces between the axial lines across the radial thickness and an integer indicating the number of spaces between the radial lines along the axis</td>
<td>2I2</td>
</tr>
<tr>
<td>2</td>
<td>many</td>
<td>An integer to represent the identification number of the displacement position and the load at that position</td>
<td>I5,5X F10.4</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>Terminating card with 0</td>
<td>I5</td>
</tr>
</tbody>
</table>
DISPLACEMENT CHECK

This program multiplies the stiffness matrix times the displacements (obtained in LOWER UPPER TREATMENT) to obtain the forces at the nodes. These forces (and their position) should correspond to those applied in LOWER UPPER TREATMENT. The first card is the same as the first card of ASSEMBLY DISC. The stiffness Matrix comes via magnetic tape (stream 14) from ASSEMBLY DISC and the displacement vector via magnetic tape (stream 17) LOWER UPPER TREATMENT. A magnetic disc (stream 20) is used during the calculations. The force for each displacement number is printed out.
Program Title  DISPLACEMENT CHECK

Input Magnetic tapes stream Numbers: 14 & 17 and Magnetic Disc Stream 20
Output Magnetic tape stream Numbers: None

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Cards of This Type</th>
<th>Information or data being read in on cards into computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>only one</td>
<td>An integer indicating the number of spaces between the axial lines across the radial thickness and an integer indicating the number of spaces between radial line along the axis</td>
</tr>
</tbody>
</table>
This program is to find the stresses at the principal nodes. The first card is the same as the first card for ASSEMBLY DISC. The second card gives Young's Modulus and Poisson's Ratio. The third card gives:

the location at the beginning of the axis

end

beginning

radius

end

The magnetic tape (stream 10) from Node Tape gives nodes and their ordinates; whilst magnetic tape (stream 17) from LOWER UPPER TREATMENT gives the displacements.

Stresses and strains calculated for principal nodes are printed out.
Program Title  **STRESS STRAIN**

Input Magnetic tapes stream Numbers: 10 & 17

Output Magnetic tape stream Numbers

<table>
<thead>
<tr>
<th>Type of Card</th>
<th>Number of Cards of This Type</th>
<th>Information or data being read in on cards into computer</th>
<th>Format Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>only one</td>
<td>An integer indicating the number of spaces between the axial lines across the radial thickness and an integer indicating the number of spaces between radial lines along the axis.</td>
<td>E10.3, F10.3</td>
</tr>
<tr>
<td>2</td>
<td>only one</td>
<td>Young's Modulus and Poisson</td>
<td>E10.3, F10.3</td>
</tr>
<tr>
<td>3</td>
<td>only one</td>
<td>Location at the beginning of the axis</td>
<td>5X,4I5</td>
</tr>
</tbody>
</table>
Lineprinter listing of Fortran instructions in each of the sequence of computer programs for Finite Element Analysis of Axisymmetric Elastic Stress Problems

<table>
<thead>
<tr>
<th>List of Programs</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>NODE TAPE</td>
<td>1</td>
</tr>
<tr>
<td>TRIANGLE TAPE</td>
<td>3</td>
</tr>
<tr>
<td>X Y ORD</td>
<td>6</td>
</tr>
<tr>
<td>SINGLE LINE GRAPH</td>
<td>8</td>
</tr>
<tr>
<td>ASSEMBLY DISC</td>
<td>11</td>
</tr>
<tr>
<td>DECOMPOSE</td>
<td>30</td>
</tr>
<tr>
<td>LOWER UPPER TREATMENT</td>
<td>50</td>
</tr>
<tr>
<td>DISPLACEMENT CHECK</td>
<td>63</td>
</tr>
<tr>
<td>STRAIN STRESS</td>
<td>76</td>
</tr>
</tbody>
</table>
GENERATION OF NODE NUMBERS (WITH ORDINATES) FOR CARTESIAN NETWORK

FORMING NODE NUMBERS FOR CARTESIAN NETWORK

IN A CARTESIAN NETWORK SEVERAL NODES HAVE A COMMON RADIUS, AND
SEVERAL NODES HAVE COMMON AXIAL LENGTH.

EACH COMMON RADIUS IN TURN IS INDEXED IN ORDER OF SIZE,
EACH COMMON AXIAL LENGTH IN TURN IS INDEXED IN ORDER OF SIZE

USING THE AXIAL AND RADIAL INDICES, THE NODE NUMBERS ARE
GENERATED AND ASSIGNED THEIR RADIUS AND AXIAL LENGTH

NOMENCLATURE

RADIUS(IRADIU) = RADIUS WHICH IS COMMON ALONG AXIS
AXIALF(IAXIAL) = AXIAL LENGTH WHICH IS COMMON FOR SEVERAL RADII
R(NODE) = RADIUS FOR A PARTICULAR NODE
Z(NODE) = AXIAL LENGTH FOR A PARTICULAR NODE
N RADWI = NUMBER OF DISTINCT RADII IN THICKNESS I/R AND O/R
NAXILE = NUMBER OF DISTINCT AXIAL LENGTHS (INCLUDING ZERO)
IAXIAL = INDIVIDUAL AXIAL INDEX
IRADIU = INDIVIDUAL RADIAL INDEX
NODE = NODE NUMBER

LOBERA = RADIAL LOCATION AT BEGINING
LOBFAX = AXIAL LOCATION AT BEGINING
LOENA = RADIAL LOCATION AT END
LOENAX = AXIAL LOCATION AT END

DIMENSION RADIUS(11), AXIALF(1), R(212), Z(212)

READ (1, 96) N RADWI
WRITE(2, 10) N RADWI
READ (1, 96) NAXILE
WRITE(2, 11) NAXILE
READ (1, 97) (RADIUS(IRADIU), IRADIU=1, N RADWI)
WRITE(2, 20) (RADIUS(IRADIU), IRADIU=1, N RADWI)
READ (1, 97) (AXIALF(IAXIAL), IAXIAL=1, NAXILE)
WRITE(2, 21) (AXIALF(IAXIAL), IAXIAL=1, NAXILE)
READ (1, 110) NO TRI, LOBERA, LOBFAX, LOENA, LOENAX
WRITE(2, 111) NO TRI, LOBERA, LOBFAX, LOENA, LOENAX
**FORMAT**

110 **FORMAT** *(5I5)**
111 **FORMAT** *(1HI,1X,4H)*TRIANGLE-**NUMBER ; ENDING PREVIOUS SECTIONS = *,15//**25X**NODE **054**

1 , 24H**BEGINNING RADIUS LOCATION , 15,14X, 22H**BEGINNING AXIS LOCATION , 15//**NODE **055**

?27X, 2H**ENDING RADIUS LOCATION , 15,14X; 20H**ENDING AXIS LOCATION , 15//**NODE **056**

C C

WRITE(2,98)

DO 100 IAXIAL = LOBEAX , LOENAX
DO 101 IRADIUS = LOBERA , LOENRA

MODE = 2*(IAXIAL-1)*(2*NRADWI-1) + 2*IRADIUS - 1

R(NODE) = RADIUS(IRADIUS)

Z(NODE) = AXIALE(IAXIAL)

WRITE(2,93) NODE,Z(NODE),R(NODE)

WRITE( 10 ) NODE,Z(NODE),R(NODE)

100 CONTINUE

CONTINUE

C C

**FORMAT** *(12)**

**FORMAT** *(8F10.4)**

**FORMAT** *(1H1//32X,4H)*NODE,19X,7H(Z(NODE)),19X,7H(R(NODE))/

**FORMAT** *(//10X,60H)*NUMBER OF RADII IN WIDTH (INCLUDING BORE AND OUTSIDE)

MODE **078**

10 **FORMAT** *(//10X,60H)*NUMBER OF AXIAL LENGTHS (INCLUDING ZERO) = ,13//

10 **FORMAT** *(//10X,60H)*NUMBER OF AXIAL LENGTHS (INCLUDING ZERO) = ,13//

10 **FORMAT** *(//10X,60H)*NUMBER OF AXIAL LENGTHS (INCLUDING ZERO) = ,13//

STOP

FND

* ENTER

DATA

G002 DATA

11 51


13.4140625 14.140625 15.0

0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5

4.0 4.5 5.0 5.5 6.0 6.5 7.0 7.5

8.0 8.5 9.0 9.5 10.0 10.5 11.0 11.5

12.0 12.5 13.0 13.5 14.0 14.5 15.0 15.5

16.0 16.5 17.0 17.5 18.0 18.5 19.0 19.5

20.0 20.5 21.0 21.5 22.0 22.5 23.0 23.5

24.0 24.5 25.0

0 1 1 11 51
IT IS DESIRABLE TO REPRODUCE GRAPHICALLY THE NETWORK OF TRIANGLES (THIS WILL SHOW ERROR IN SPECIFICATION)

THE NODES NEED TO BE PLOTTED IN SUCH A WAY THAT ONE CONTINUOUS LINE Scribes ALL THE SIDES OF ALL THE TRIANGLES IN SEQUENCE.

IF THE VERTICES FORM A CARTESIAN NETWORK, USE CAN BE MADE OF THIS IN FORMING THE CONTINUOUS LINE.

IN ORDER TO DO THIS, USE IS MADE OF THE AXIAL AND RADIAL INDICES.

BUT IT IS SUFFICIENT TO SPECIFY TRIANGLES BY THEIR NODIAL NUMBERS.

NOMENCLATURE

MAXTRI = MAXIMUM TRIANGLE NUMBER OR INDEX
MAXRAD = MAXIMUM RADIUS NUMBER OR INDEX
MAXAXI = MAXIMUM AXIAL NUMBER OR INDEX
LOBERA = RADIAL LOCATION AT BEGINING
LOBFAX = AXIAL LOCATION AT BEGINING
LOENRA = RADIAL LOCATION AT END
LOENAX = AXIAL LOCATION AT END
LOCATN(IAXIAL,IRADIUS)
NRADWI = NUMBER OF DISTINCT RADI (INCLUDING I/R AND O/R)
DIMENSION LOCATN(101,21),NODTRI(3).
READ (1,96)NRADWI
3 WRITE(2,10)NRADWI
10 FORMAT(12)
WRITE(2,10)NRADWI
96 FORMAT(12)
WRITE(2,10)NRADWI
96 FORMAT(12)
WRITE(2,10)NRADWI
10 FORMAT(1/10X,6D10D NUMBER OF RADI IN WIDTH (INCLUDING BORE AND OUTSIDE RADIUS),13//)
READ (1,110) NO TRI,LOBERA,LOBEAX,LOENRA,LOENAX
110 FORMAT(515)
WRITE(2,111) NO TRI,LOBERA,LOBEAX,LOENRA,LOENAX
111 FORMAT(1H1,1X,41HTRIANGLE-NUMBER ENDING PREVIOUS SECTION =,15//25X)
1,24HBEGINING RADIUS LOCATION,15,14X,22HBEGINING AXIS LOCATION,15//21512
227X,22HENDING RADIUS LOCATION,15,14X,20HENDING AXIS LOCATION,15//21512
REWIND 11
DO 100 IAXIAL = LOBEAX, LOENAX
DO 101 IRADIU = LOPERA, LOENPA
NODE = 2*(IAXIAL-1)*(2*IRADWI-1) + 2*IRADIU - 1
LOCATN(IAXIAL,IRADIU) = NODE
101 CONTINUE
100 CONTINUE
WRITE(2,175)
175 FORMAT(1H1/10X,8HTRIANGLE,44X,15HNODIAL NUMBERS/11X,6HNUMBER,22X)
12H1, 27X, 2HJ, 27X, 2HK, //)
IN TRI = NO TRI
NODTRI = NO TRI + 1
IRADIU = LOBERA
TAXIAL = LOBEAX
WRITE(2,172)IAXIAL
NEWLIN=0
120 CONTINUE
C FORM LOWER TRIANGLE
---------------
C
IN TRI = IN TRI + 1
NODTRI(1) = LOCATN(IAXIAL, IRADIU )
NODTRI(2) = LOCATN(IAXIAL+1, IRADIU+1)
NODTRI(3) = LOCATN(IAXIAL , IRADIU+1)
WRITE(2,171)IN TRI, (NODTRI(IVERT), IVERT=1,3)
171 FORMAT(12X, I4, 22X, I4, 25X, I4, 25X, I4)/
WRITE(11 )IN TRI, (NODTRI(IVERT), IVERT=1,3), IAXIAL, NEWLIN
NEWLIN=0
C FORM UPPER TRIANGLE
---------------
C
IN TRI = IN TRI + 1
NODTRI(1) = LOCATN(IAXIAL, IRADIU )
NODTRI(2) = LOCATN(IAXIAL+1, IRADIU+1)
NODTRI(3) = LOCATN(IAXIAL+1, IRADIU+1)
WRITE(2,171)IN TRI, (NODTRI(IVERT), IVERT=1,3)
WRITE( 11 )IN TRI, (NODTRI(IVERT), IVERT=1,3), IAXIAL, NEWLIN
C INCREMENT RADIUS INDEX
---------------
C
IRADIU = IRADIU + 1
C TEST, IF AT END RADIUS
---------------
C
IF(IRADIU-LOFNRA)120,150,150
C 150 CONTINUE
C INCREMENT AXIAL INDEX
---------------
C
IAXIAL = IAXIAL + 1
NEWLIN=1
C TEST, IF AT END AXIS
---------------
C
IF(IAXIAL-LOENAX)160,170,170
C
160 CONTINUE
IRADIU = LOBERA
WRITE(2,172)AXIAL
172 FORMAT(/9X,14AXIAL INDEX = ,I2/)
GO TO 120
170 CONTINUE
MAXTRI = IN TRI
MAXRAD = IRADIU
MAXAXI = IAXIAL
WRITE(2,173)MAXTRI,MAXRAD,MAXAXI
173 FORMAT(1H1/30X,7HMAXTRI = ,I4,14X,7HMAXRAD = ,I4,14X,7HMAXAXI = ,I4///)
END
REWIND 11
STOP
END
*ENTER
DATA
G003DATA
JOB VLA4700A

JOHN WARD  X-Y ORD GRAPH

COMPUTING 2 MINUTES
STORE 72 BLOCKS

OUTPUT
0 LINO PRINTER 1000 LINES
2 LINO PRINTER 2000 LINES
INPUT 1 GOOD DATA
TAPE 10 N0121 VLA47 NODE*INHIBIT
TAPE 11 N0137 VLA47 TRIANGLE*INHIBIT
TAPE 12 N0183 VLA47 X-Y ORD*PERMIT

COMPILER HARTMAN

ENDTRAM

C

X-Y ORNATURATES FOR GRAPH

------------------------

TO OBTAIN X AND Y ORNATURATES TO PLOT ONE CONTINUOUS LINE GRAPH

NOMENCLATURE

------------------------

MAXN0D = MAXIMUM NODE NUMBER
NTRIR = NUMBER OF TRIANGLES TO BE READ IN
INODE = INDIVIDUAL NODE
ITRIAG = INDIVIDUAL TRIANGLE
IN TRI = INDEX OF TRIANGLE
IVERT = INDIVIDUAL VERTEX
IXIAL = AXIAL INDEX
T(J,JVERT) = 2 \times 3 MATRIX = TWC DIRECTION BY THREE VERTICES
NODTRI(IVERT) = A VECTOR CONTAINING THE NODE NUMBERS OF INDIVIDUAL VERTICES
IPOINT = POINT INDEX
R(NODE) = RADIAL ORDINATE OF A NODE
Z(NODE) = AXIAL ORDINATE OF A NODE
NPNO(IPOINT) = NODE NUMBER OF I-TH POINT

DIMENSION Z(2121), R(2121), NODTRI(3), T(2, 3), NPNO(3000), X(20, 500),
Y(20, 500), NPINT(20)
REAIIWIND 10
REWIND 11

READING FROM CARD THE NUMBER OF NODES TO BE READ IN

2A FORMAT(15)
READ (1, 28) MAXNOD
WRITE (2, 29) MAXNOD

2A FORMAT(11H1/43X, 36H THE NUMBER OF NODES TO BE READ IN = / , 15/)

READING FROM TAPE 10 AND WRITING OUT THE NODES AND THEIR ORDINATES

WRITE (2, 93)

98 FORMAT(11H1/32X, 4HNODE, 19X, 7H2(NODE), 19X, 7HR(NODE))/
DO 100 NODE = 1, MAXNOD
READ (10) NODE, (Z(NODE), R(NODE))
WRITE (2, 93) NODE, (Z(NODE), R(NODE))
100 CONTINUE

93 FORMAT(33X, I13, 16X, F10.4, 16X, F10.4)

READING FROM A CARD THE NUMBER OF TRIANGLES TO BE READ IN
READ (1, 20) NTRIRD
WRITE (2, 30) NTRIRD
30 FORMAT (1H1/41X, 39H THE NUMBER OF TRIANGLES TO BE READ IN =, I5/)

READING FROM TAPE 11 * WRITING OUT TRIANGLE INDEX AND ITS NODES.

XMIN = 100
XMAX = 0.0
YMIN = 100
YMAX = 0.0
ICURVE = 1
IPINT = 0
WRITE (2, 173)
173 FORMAT (1H1/3X, 11H TRIANGLE NO, 8X, 2HI., 5X, 2HR., 8X, 2HZ., 11X, 2HIJ., 5X,
12HR., 8X, 2HZ., 11X, 2HK., 5X, 2HR., 9X, 2HZ., 5X, 12H AXIAL NUMBER //)
DO 104 ITRIAG = 1, NTRIRD
READ (11) IN TRI, (NODTRI (IVERT), IVERT = 1, 3), IAXIAL
DO 103 IVERT = 1, 3
NODE = NODTRI (IVERT)
T (1, IVERT) = RNODE
T (2, IVERT) = Z (NODE)
103 CONTINUE
WRITE (12) IN TRI, (T (I, J), I = 1, 2), J = 1, 3), IAXIAL
WRITE (2, 174) IN TRI, (NODTRI (IVERT), (T (I, IVERT), I = 1, 2), IVERT = 1, 3),
1 IAXIAL
174 FORMAT (14, I10, 2F10.3, I10, 2F10.3, I10, 2F10.3, I7//)
104 CONTINUE

STOP
END

*ENTER

DATA
G0040 DATA
SINGLE LINE GRAPH

DIMENSION X(3000), Y(3000)
READ(1, 28) NTRIRD

28 FORMAT(15)
WRITE(2, 30) NTRIRD

30 FORMAT(1H1/41X, 39HTHE NUMBER OF TRIANGLES TO BE READ IN, I5/)

I1 = 1
I3 = 3

DO 104 ITRIAG = 1, NTRIRD
READ(10) IN TRI, (X(I), Y(I), I1, I3), IAXIAL
WRITE(2, 401) IN TRI, (X(I), Y(I), I1, I3), IAXIAL

401 FORMAT(1X, I15, 6F15.3, I15/)
I1 = I1 + 3
I3 = I3 + 3

104 CONTINUE

N = S - NTRIRD
CALL SECTON (N, X, Y)
STOP
END
SUBROUTINE SECTION (N, X, Y)

DIMENSION X(3000), Y(3000)

NN1 = N - 1

CALL IDST

XSCO = 200.0
YGCO = 900.0
XSCR = 480.0
YSR = 800.0

CALL SCFOR(1, XSCO, YSCO, XSCR, YSCR)

XMAX = 15.0

CALL SCFOR(2, 0, 0, XMAX, YMAX)

CALL SCFOR(3, 920.0, 2.0, 3.0, 0.0)

CALL SCFOR(4, 5.0, 0.0, 0.0, 3.0)

CALL SCFOR(5, 190.0, 2.0, 3.0, 0.0)

CALL SCFOR(6, 5.0, 0.0, 0.0, 3.0)

CALL MOVETP(320, 950)

WRITE(14, 99)

99 FORMAT(8HG2RADIUS)

WRITE(14, 98)

98 FORMAT(2HG)

CALL MOVETP(160, 250)

WRITE(14, 100)

100 FORMAT(14HG2AXIAL LENGTH)

DO 10 I = 1, NN1

X1 = X(I)

X2 = X(I + 1)
Y1 = Y(I)
Y2 = Y(I+1)
CALL SCFOR(8, X1, Y1, X2, Y2).
10 CONTINUE
CALL IDEND
RETURN
END
JOB VLA47012 JOHN WARD ASSEMBLY DISC
STORE 140 BLOCKS
COMPUTING 60 MINUTES
EXECUTION 80 MINUTES
OUTPUT 0 LINEPRINTER 2400 LINES
OUTPUT 2 LINEPRINTER 15000 LINES
INPUT 1 G012DATA
TAPE COMMON 20
TAPE 10 N0121 VLA47 NODE*INHIBIT
TAPE 11 N0137 VLA47 TRIANG*INHIBIT
TAPE 14 N1301 ASSEMBLY*PERMIT
*RUN SOURCE
*FORTRAN SOURCE

C NOMENCLATURE
C
C INPUT PARAMETERS
C
C NPHSQR=NUMBER OF ROWS OF PHYSICAL SQUARES ALONG RADIAL LENGTH
C NPHSQA=NUMBER OF ROWS OF PHYSICAL SQUARES ALONG AXIAL LENGTH
C
C DERIVED PARAMETERS
C
C NUMSQS=NUMBER OF SQUARES (TOTAL)
C NUMTRI=NUMBER OF TRIANGLES
C NTVENO=NUMBER OF VERTEX NODES TO SPECIFY TRIANGLES
C NPHSQR=NPHSQR*NPHSQA
C
C NUMTRI=2*NUMSQS
C NTYRN=3*NUMTRI
C
C NTYN=TOTAL NUMBER OF DISTINCT VERTEX NODES
C NTYRN=TOTAL NUMBER OF DISTINCT VERTEX NODES
C MPHSQR=TOTAL NODES PER RADIAL WIDTH
C ANX=TOTAL NODES PER AXIAL LENGTH
C NUMR=TOTAL NUMBER OF NODES
C
C LASMAT=NUMBER OF ROWS OF LARGE ASSEMBLY MATRIX
C NMPRL=NUMBER OF DISPLACEMENTS PER RADIAL LENGTH
C USE=ASSEMBLY (PARTIAL) OF ELEMENTS IN ONE ROW (PHYSICAL)
C
C NASME=SIZE OF ASSEMBLY
C NTREAT=WIDTH OF TREATMENT MATRIX
C
10 FORMAT(2I2)
READ (2,10)NPHSQR,NPHSQA
WRITE(2,20)NPHSQR,NPHSQA
20 FORMAT(1H1/2X,52HNUMBER OF ROWS OF PHYSICAL SQUARES PER RADIAL WIDTH=,I4)
1TH=I2,2X,52HNUMBER OF ROWS OF PHYSICAL SQUARES PER AXIAL LENGTH=,I4)
21 FORMAT(10H0,2H TOTAL NUMBER OF RECTANGLES =,I4)
22 FORMAT(1H1,20HNUMBER OF TRIANGLES=,I4)
23 FORMAT(1H1,58H NUMBER OF VERTEX NODES REQUIRED TO SPECIFY ALL TRIANGLES =,I5)

C
**FORTRAN SOURCE**

SUBROUTINE ASMDIS(NPHSOR, NPHSQA, NTYRNO, NASSEM, NDSPRL, LASMAT)

DIMENSION T(2,3), NODSIX(6), RAD(6), AXI(6), NTWELV(12), DIMENS(12),
QD(12,12), REFASGN(12,12), NODTRI(3),
Z(NTYRNO), P(NTYRNO), ASSEMBY(NASSEM, NASSEM), MSGQ(NDSPRL, NDSPRL),
ASDI(2,3)

C NDSPRL=4*NPHSOR+2
C KOUNT=0
C NASSEM=3*NDSPRL

CALL ZERASY(ASEMBY, NASSEM)

END

NRADWI=NPHSOR+1
WRITE(2,24)NRADWI
24 FORMAT(1H,35)NUMBER OF DISTINCT RADII IN WIDTH =,I2)
C NAXILM=NPHSQA+1
WRITE(2,25)NAXILM
25 FORMAT(1H,33)NUMBER OF DISTINCT AXIAL LENGTHS =,I3)
C NDVRNO=NAXILM*NRADWI
WRITE(2,26)NDVRNO
26 FORMAT(1H,33)NUMBER OF DISTINCT VERTEX NODES =,I4)
C N0PRWI=2*NPHSOR+1
WRITE(2,27)N0PRWI
27 FORMAT(1H,40)TOTAL NUMBER OF NODES PER RADIAL WIDTH =,I2)
C N0PALN=2*NPHSOR+1
WRITE(2,28)N0PALN
28 FORMAT(1H,40)TOTAL NUMBER OF NODES PER AXIAL LENGTH =,I4)
C NUMN0D=N0PRWI*N0PALN
WRITE(2,29)NUMN0D
29 FORMAT(1H,23)WHOLE NUMBER OF NODES =,I5)
C LASMAT=2*NUMN0D
WRITE(2,30)LASMAT
30 FORMAT(1H,41)NUMBER OF ROWS OF LARGE ASSEMBLY MATRIX =,I5)
C NDSPRL=2*N0PRWI
WRITE(2,31)NDSPRL
31 FORMAT(1H,43)NUMBER OF DISPLACEMENTS PER RADIAL LENGTH =,I2)
C NASSEM=3*NDSPRL
WRITE(2,32)NASSEM
32 FORMAT(1H,33)SIZE OF THE ROW ASSEMBLY MATRIX =,I3)
C NTREAT=5*NDSPRL
WRITE(2,33)NTREAT
33 FORMAT(1H,27)WIDTH OF TREATMENT MATRIX =,I3)
C CALL ASMQISINPHSOR,NPHSQA,NTYRNO,NASSEM,NDSPRL,LASMAT)
C STOP
READING IN THE VERTICES NODES AND THEIR ORDINATES

CALL NODORD(R,Z,MTVRNO)

READING IN MATERIAL PROPERTIES

144 FORMAT(E10.3,F10.4)
READ (1,144)E,ENU
WRITE(2,146)E,ENU
146 FORMAT(10X,30X,YOUNG'S MODULUS OF ELASTICITY =,PE10.3,
110X,POISSON'S RATIO =,OP10.4)

READ FROM A CARD THE NUMBER OF TRIANGLES TO BE READ IN

28 FORMAT(I5)
READ (1,28)NTRIRD
WRITE(2,30)NTRIRD
30 FORMAT(1HI/41X,34X,THE NUMBER OF TRIANGLES TO BE READ IN =,I5/)

ISMSQ = 1
DO 104 ITRIAG=1,NTRIRD

READING IN A TRIANGLE INDEX AND ITS VERTEX NODES

CALL TRIAIN(IN TRI,NODTRI,IAXIAL,NEWLIN)

IF(NEWLIN)800,801,822
800 WRITE(2,799)
799 FORMAT(1HI/50X,22X,NEWLINE INDEX NEGATIVE/)
STOP

822 IAXIAL = IAXIAL - 1
CALL SORTDS(ASEMBY,MASSEM,NDSPRIL,ISMSQ,KOUNT,IAXIAL)
IAXIAL = IAXIAL + 1
R01 CONTINUE

FORMING T

CALL FORMTE(NODTRI,R,Z,MTVRNO,NODSIX,T,IN TRI)
CALL STIFMA(T,E,ENU,QD)
WRITE(2,14)
14 FORMAT(1HI/5X,9HMATRIX QD/)
CALL PPDQD(QD)
PUTTING QD INTO REARMG

THE STIFFNESS MATRIX(6 RADII,6 AXIAL)*(6 RADII,6 AXIAL), NEEDS TO
BE RE-ARRANGED SO THAT ODD NUMBERS ARE RADIAL AND EVEN NUMBERS
AXIAL.

ALSO THE SIX NODIAL NUMBER NEED TO BE COMPLETED AND THE TWELVE
DISPLACEMENT NUMBERS FORMED

CALL RFARRA(QD,REARGN)
CALL CONSIX(NODSIX,NTVNO)
CALL FORM12(NODSIX,NTWELV,NTVNO,DIMS)
CALL ADDELE(NTWELV,NTVNO,AXIAL,ASSEMBY,ASSEM,REARGN)
WRITE(2,532) (NTWELV(K),DIMENS(K)),K=1,12)
532 FORMAT(1X,12(I3,F7.3))

CONTINUE

CALL SQTNOS(ASEMBY,ASSEM,NDSPRL,ISMSQ,K0UNT,AXIAL)
CALL LASEMB(ASEMBY,ASSEM,NDSPRL,ISMSQ,K0UNT,AXIAL)
CALL OSTOTA(K0UNT,NDSPRL)
WRITE(2,999)
999 FORMAT(33X,14HEND OF PROGRAM/)
RETURN
END

*FORTRAN SOURCE
SUBROUTINE ZERASY(ASEMBY,ASSEM)
DIMENSION ASMBY(ASSEM,ASSEM)

MAXIMUM NUMBER OF DISPLACEMENT NUMBERS=MXDISN

MXDISN=ASSEM
DO 701 I=1,MAXDISN
DO 702 J=1,MAXDISN
ASEMBY(I,J)=0.0
701 CONTINUE
702 CONTINUE
RETURN
END

*FORTRAN SOURCE
SUBROUTINE NORDN(R,Z,NTVNO)
DIMENSION Z(NTVNO),R(NTVNO)

INODE = INDIVIDUAL NODE
R(NODE) = RADIAL ORDINATE OF NODE
Z(NODE) = AXIAL ORDINATE OF NODE

INPUT NODE AND ORDINATES

READING FROM A CARD THE NUMBER OF NODES TO BE READ IN

MAXNOD = NUMBER OF NODES TO BE READ IN

28 FORMAT(15)
READ (1,28) MAXNOD
WRITE(2,29) MAXNOD
29 FORMAT(1H1//43X,36HTHE NUMBER OF NODES TO BE READ IN = 15)
SUBROUTINE FORTRAN 10 and writing out the nodes and their ordinates.

1. WRITING THE NODES AND THEIR ORDINATES.
   
   2. FORMAT(//3X,4HNODE,19X,7HIZ(NODE),19X,7HR(NODE)//)
   3. DO 100 INODE=1,MAXNOD
   4. READ(10) NODE,(Z(NODE),R(NODE))
   5. WRITE(2,93) NODE,(Z(NODE),R(NODE))
   6. CONTINUE
   7. RETURN
   8. END

*FORTRAN SOURCE

SUBROUTINE FORMTE(NODTRI,R,Z,NTVRNO,NODSIX,T,IN TRI)
DIMENSION NODTRI(3),Z(NTVRNO),R(NTVRNO),NODSIX(6),T(2,3)

FORMING TRIND:

NODTRI(IVERT) = THE THREE NODE NUMBERS PER TRIANGLE, ONE PER VERTEX
NODSIX(I) = THE SIX NODES OF THE TRIANGLE

DO 3 IVERT = 1,3
   I = 2 * IVERT - 1
   NODE = NODTRI(IVERT)
   NODSIX(I) = NODE
   T(1,IVERT) = P(NODE)
   T(2,IVERT) = Z(NODE)
3 CONTINUE

WRITE(2,8)IN TRI
8 FORMAT(1H,17HTRIANGLE NUMBER = ,I5)
RETURN
END

*FORTRAN SOURCE

SUBROUTINE TRIAIN(IN TRI,NODTRI,IAXIAL,NEWLIN)
DIMENSION NODTRI(3)

NOMENCLATURE

IN TRI = INDEX OF TRIANGLE
IVEPT = INDIVIDUAL VERTEX
NODTRI(IVERT) = A VECTOR CONTAINING THE NODE NUMBERS OF INDIVIDUAL
               VERTICES
IAXIAL = AXIAL INDEX
NEWLIN = 1 FOR NEWLINE

READ FROM TAPE 11 AND WRITING OUT TRIANGLE INDEX AND ITS NODES.

READ(11) IN TRI,(NODTRI(IVERT),IVERT=1,3),IAXIAL,NEWLIN
WRITE(2,174) IN TRI,(NODTRI(IVERT),IVERT=1,3),IAXIAL,NEWLIN
174 FORMAT(23X,14,11X,14,11X,14,11X,14,11X,12,11X,12)
RETURN
END

*FORTRAN SOURCE

SUBROUTINE REARRA(QD,REARGN)
DIMENSION QD(12,12),RFARGN(12,12)

OD = STIFFNESS MATRIX AS EVOLVED

REARGN = STIFFNESS MATRIX WITH ODD NUMBERS RADIAL AND EVEN AXIAL

DO 42 I=1,6
DO 41 J=1,6
REARGN(2*I-1,2*J-1) = QD(I , J )
REARGN(2*I-1,2*J ) = QD(I , J+6)
REARGN(2*I ,2*J-1) = QD(I+6,J )
REARGN(2*I ,2*J ) = QD(I+6,J+6)

41 CONTINUE
42 CONTINUE

WRITE(2,15)
15 FORMAT(1H1//52X,17HMATRIX REARRANGED/) CALL PRID07(REARGN)
RETURN

*FORTRAN SOURCE

DIMENSION NODSIX(6),NTWELV(12),DIMENS(12),R(NTWNO),Z(NTWNO)

FORMING TWELVE LOCATING NUMBERS

EACH NODE HAS TWO NUMBERS, I.E. NTWELV(ODD) AND NTWELV(EVEN).

THE ODD NUMBERS ARE THE RADIAL DIMENSION AND THE

EVEN NUMBERS ARE THE AXIAL DIMENSION.

NTWELVE IS A VECTOR OF LENGTH TWELVE, CONTAINING THE SPACIAL

NUMBERS, OF A TRIANGLE.

DO 360 I=1,6
NODE = NODSIX(I)
NTWELV(2*I-1)=2*NODE-1
DIMENS(2*I-1)=R(NODE)
NTWELV(2*I )=2*NODE
DIMENS(2*I )=Z(NODE)

360 CONTINUE

RETURN

*FORTRAN SOURCE

DIMENSION ASMBY,NASSEM,NOSPRL,ISMSQ,KOUNT,IAXIAL

ASMBY = ASSEMBLY, NASSEM = NASSFM, NDSPRL = ISMSQ, KOUNT = IAXIAL

SQTOD = SMALL SQUARED TO DISC

PURPOSE OF THIS PROGRAM IS TO TAKE SMALL SQUARES FROM THE STIFFNESS MATRIX FOR A PHYSICAL ROW AND PUT THEM ON DISC /3 THEN 3 THEN 21

SQTOD 1
SQTOD 2
SQTOD 3
SQTOD 4
SQTOD 5
SQTOD 6
SQTOD 7
SQTOD 8
SQTOD 9
SQTOD 10
SQTOD 11
ASEMBY(I,J) = SUB-ASSEMBLY CONTAINING THE STIFFNESS ELEMENTS FOR ARROWS
NDSPRL = NUMBER OF DISPLACEMENTS PER RADIAL LINE OF PHYSICAL SQUARED
KOUNT = COUNT OF WRITE STATEMENTS ON TO DISC
SMSQ(I,J) = SMALL SQUARE = PARTITIONED MATRIX
JR = (I,J) = TEMPORARY STORE, BEFORE TRANSFER BTM-RT TO TOP LEFT.
JSMSQ = COLUMN INDEX OF SMSQ GLOBALLY
JSMSQ = BEGINNING VALUE OF JSMSQ
ISS = INDEX(1, 2 OR 3) OF SMALL SQUARE FOR ROWS = IROW
JSS = INDEX(1, 2 OR 3) OF SMALL SQUARE FOR COLUMNS = JCOL
IB = ROW INDEX WITHIN SMALL SQUARE (SPACIAL) BEGINNING
IE = ROW INDEX WITHIN SMALL SQUARE (SPACIAL) ENDING
IA = ROW INDEX WITHIN SMALL SQUARE (SPACIAL) ANY
JB = COLUMN INDEX WITHIN SMALL SQUARE (SPACIAL) BEGINNING
JE = COLUMN INDEX WITHIN SMALL SQUARE (SPACIAL) ENDING
JA = COLUMN INDEX WITHIN SMALL SQUARE (SPACIAL) ANY
I = ROW INDEX WITHIN SMALL SQUARE (LOCAL)
J = COLUMN INDEX WITHIN SMALL SQUARE (LOCAL)
NASSEM = SIZE OF ASEMBY
JSS+1 = JSMSQ
DO 804 ISS=1,3
JSMSQ=JSMSQ+1
IB=(ISS-1)*NDSPRL+1
IF= ISS * NDSPRL
DO 807 JSS=1,3
JB=(JSS-1)*NDSPRL+1
JE= JSS * NDSPRL
I=0
DO 810 IA=IB,IE
I=I+1
J=0
DO 812 JA=JB,JE
J=J+1
SMSQ(I,J)=ASEMBY(IA,JA)
812 CONTINUE
810 CONTINUE
WRITE(2,993)
STOP
807 CONTINUE
I=0
DO 816 IA=IB,IE
J=J+1
816 CONTINUE
815 CONTINUE
ZERO ASSEMBLY MATRIX

NASSEM = N * N
DO 104 I = 1, NASSEM
DO 105 J = 1, NASSEM
ASSEMBLY(I, J) = 0.0

ADD STORE TO TOP RIGHT HAND CORNER OF ASSEMBLY MATRIX

DO 817 I = 1, NDSPL
DO 818 J = 1, NDSPL
ASSEMBLY(I, J) = STORE(I, J)

FORMAT(1H1, 31H MISTAKE IN SOTOS, ISS*JS, GT, 97)

SUBROUTINE ASSEMBLY(ASSEMBLY, NASSEM, NDSPL, ISNSQ, KOUT, IAXIAL)

DIMENSION ASSEMBLY(NASSEM, NASSEM), SMSQ(NDSPL, NDSPL)

JSNSQ = ISNSQ
N = NDSPL
DO 36 I = 1, N
DO 27 J = 1, N
SMSQ(I, J) = ASSEMBLY(I, J)

CALL PBLQCKCSMSQ, IFC, ILC, IR, ILR, NDSPL

RETURN
END

SUBROUTINE SOTOS(KOUT, NDSPL)

DIMENSION SMSQ(NDSPL, NDSPL)

THIS PROGRAM IS TO READ FROM DISC AND TO PRINT ON TO TAPE

IFC = 1
ILC = NDSPL
IFR = 1
ILR = NDSPL
REWIND 20
WRITE(14) KOUT
DO 100 I = 1, KOUT
READ (20) ISMSQ, JSMSQ, SMSQ, IAXIAL
WRITE(14) ISMSQ, JSMSQ, SMSQ, IAXIAL
WRITE(2, 101) ISMSQ, JSMSQ, SMSQ, IAXIAL

101 FORMAT(1H1, 10X, 6HISMSQ=, I3, 10X, 6HJSMSQ=, I3, 10X, 7HIAXIAL=, I3)

CALL PBLQCKCMSQ, IFC, ILC, IFR, ILR, NDSPL

100 CONTINUE
RETURN
END

SUBROUTINE FORMAT(T, AD)

AD 001
AD 002
FORMSATION OF MATRIX A

This routine is to form the matrix which when post-multiplied by the coefficients A, produce the radial and axial deflections at the head corners of the triangle.

DIMENSION T(2,3), R(6), Z(6), AD(12,12), DAH(6,6)

R(1) = T(1,1)
R(2) = (T(1,1) + T(1,2)) / 2.
R(3) = T(1,2)
P(4) = (T(1,2) + T(1,3)) / 2.
R(5) = T(1,3)
Z(1) = T(2,1)
Z(2) = (T(2,1) + T(2,2)) / 2.
Z(3) = T(2,2)
Z(4) = (T(2,2) + T(2,3)) / 2.
Z(5) = T(2,3)
Z(6) = (T(2,1) + T(2,3)) / 2.

DO 50 I = 1, 6
   DAH(I,1) = R(I) * R(I)
   DAH(I,2) = R(I)
   DAH(I,3) = 1.
   DAH(I,4) = Z(I) * Z(I)
   DAH(I,5) = Z(I)
   DAH(I,6) = R(I) * Z(I)
50 CONTINUE

INVERSION OF MATRIX DAH

NUMSIX = 6
CALL INVPV(NUMSIX, DAH)

DO 48 I = 1, 12
   DO 47 J = 1, 12
      AD(I, J) = 0.0
47 CONTINUE
48 CONTINUE
   DO 52 I = 1, 6
      DO 51 J = 1, 6
         AD(I, J) = DAH(I, J)
         AD(I+6, J+6) = DAH(I, J)
51 CONTINUE
52 CONTINUE

RETURN
END

FORTRAN SOURCE

Subroutine INVPV(N, ASUBST)

DIMENSION ASUBST(6,6), LOC(6), LOCC(6), INVLOC(6), A(6,6)

INITIALIZING LOC

DO 700 I = 1, N
DO 750 J=1,N
A(I,J)=ASUBST(I,J)
CONTINUE

750 CONTINUE
NM1=N-1
DO 201 I=1,N
LOC(I)= I
201 CONTINUE
DO 208 I=1,N
C TO FIND THE VALUE OF J TO GIVE THE LARGEST VALUE OF A(I,J)
C
BIG = 0.0
L = 0
DO 203 J=1,N
K = LOC(J)
IF(BIG-ABS(A(K,I))>202,203,203
CONTINUE
BIG = ABS(A(K,I))
L = K
203 CONTINUE
IF(L)500,501,502
500 WRITE(2,503)
503 FORMAT(1H1/10X,16HL IS NEGATIVE =,15//)
GO TO 905
501 WRITE(2,504)
504 FORMAT(1H1/10X,9HL IS ZERO//)
GO TO 905
202 CONTINUE
IF(L-LOC(I))211,210,209
209 CONTINUE
K = LOC(I)
LOC(I) = L
LOC(L) = K
GO TO 210
211 CONTINUE
K = LOC(I)
LOC(I) = L
LOC(I) = L
N=INVLOC(L)
LOC(N) = K
210 CONTINUE
DO 102 K=1,N
M=LOC(K)
INVLOC(M)=K
102 CONTINUE
C THE PIVOT P
---------
P = A(L,I)
A(L,I) = 1.0
C REDUCING THE PIVOTAL ROW
-------------
DO 204 K=1,N
A(L,K) = A(L,K) / P
204 CONTINUE
DO 207 J=1,N
IF(J-L)205,207,205
205 CONTINUE
R = A(J,I)
A (J, I) = 0, C
DO 206 K = 1, N

SUBTRACTING R * PIVOTAL ROW FROM OTHER ROWS

A (J, K) = A (J, K) - A (L, K) * R

CONTINUE
CONTINUE
CONTINUE
CONTINUE

DO 801 K = 1, N
LOCC (K) = LOC (K)
CONTINUE
CONTINUE
CONTINUE
CONTINUE

DO 804 J = 1, NM1
DO 802 K = J, N
L = LOC (K)
INVLOC (L) = K
CONTINUE
CONTINUE
CONTINUE
CONTINUE

K = INVLOC (J)
CONTINUE
CONTINUE
CONTINUE
CONTINUE

A (I, J) = A (I, K)
A (I, K) = B

CONTINUE
CONTINUE
CONTINUE
CONTINUE

LOCC (K) = LOC (J)
LOCC (J) = J
CONTINUE
CONTINUE
CONTINUE
CONTINUE

RE-ARRANGEMENT OF ROWS

DO 901 K = 1, N
L = LOC (K)
INVLOC (L) = K
CONTINUE
CONTINUE
CONTINUE
CONTINUE

DO 904 J = 1, NM1
FORMAT (/NO, 7THROW I =, I2/)
DO 902 K = 1, N
L = INVLOC (K)
LOCC (L) = K
CONTINUE
CONTINUE
CONTINUE
CONTINUE

K = LOC (I)
CONTINUE
CONTINUE
CONTINUE
CONTINUE

R = A (I, J)
A (I, J) = A (K, J)
A (K, J) = B

CONTINUE
CONTINUE
CONTINUE
CONTINUE

INVLOC (K) = INVLOC (I)
INVLOC (I) = I
CONTINUE
CONTINUE
CONTINUE
CONTINUE

CONTINUE
CONTINUE
CONTINUE
CONTINUE

DO 752 I = 1, N
DO 754 J = 1, N
ASUBST (I, J) = A (I, J)
CONTINUE
CONTINUE
CONTINUE
CONTINUE

RETURN
END

*FORTRAN SOURCE

DIMENSION T (2, 3), S (15), SA (12, 12)
P1 = 4. * ATAN (1.)
A = L - ENU
\[ p = \left(1 - 2 \times \text{FNU}\right) / 2. \]

\[ c = 1 + \text{FNU} \]

\[ d = \sqrt{\left(1 + \text{FNU}\right) \times \left(1 - 2 \times \text{FNU}\right) \times 2 \times \pi} \]

\[
\text{DO } 2 \text{ I} = 1, 12 \\
\text{DO } 1 \text{ J} = 1, 12 \\
\text{SA} \text{I}, \text{J} = 0, 0 \\
1 \text{ CONTINUE}
\]

2 CONTINUE

\[
\text{CALL INTI15(T, S)} \\
\text{WRITE(2, 28)} (S(M), M = 1, 15).
\]

29 \[
\text{FORMAT(10X}, 5 \text{HS(}}, 1), 19 \text{X}, 5 \text{HS(}}, 2), 19 \text{X}, 5 \text{HS(}}, 3), 19 \text{X}, 5 \text{HS(}}, 4), 19 \text{X}, 5 \text{HS(}}, 5) \]

1 \[
/1X, 1 \text{PS24.14} / \\
2 \[
/1X, 1 \text{PS24.14} / \\
3 \[
/1X, 1 \text{PS24.14} / \\
4 \[
/1X, 1 \text{PS24.14} / \\
5 \[
/1X, 1 \text{PS24.14} / \\
\]

SA(1, 1) = (5 - \text{FNU}) \times S(4) \quad *D \\
SA(1, 2) = 3 \times S(3) \quad *D \\
SA(1, 3) = C \times S(1) \quad *D \\
SA(1, 4) = C \times S(5) \quad *D \\
SA(1, 5) = C \times S(2) \quad *D \\
SA(1, 6) = 3 \times S(6) \quad *D \\
SA(1, 10) = 6 \times \text{FNU} \times S(6) \quad *D \\
SA(1, 11) = 3 \times \text{FNU} \times S(3) \quad *D \\
SA(1, 12) = 3 \times \text{FNU} \times S(4) \quad *D \\
SA(2, 2) = 2 \times S(1) \quad *D \\
SA(2, 3) = S(7) \quad *D \\
SA(2, 4) = S(9) \quad *D \\
SA(2, 5) = S(8) \quad *D \\
SA(2, 6) = 2 \times S(2) \quad *D \\
SA(2, 10) = 4 \times \text{FNU} \times S(2) \quad *D \\
SA(2, 11) = 2 \times \text{FNU} \times S(1) \quad *D \\
SA(2, 12) = 2 \times \text{FNU} \times S(3) \quad *D \\
SA(3, 3) = A \times S(11) \quad *D \\
SA(3, 4) = A \times S(13) \quad *D \\
SA(3, 5) = A \times S(12) \quad *D \\
SA(3, 6) = S(8) \quad *D \\
SA(3, 10) = 2 \times \text{FNU} \times S(8) \quad *D \\
SA(3, 11) = \text{FNU} \times S(7) \quad *D \\
SA(3, 12) = \text{FNU} \times S(1) \quad *D \\
SA(4, 4) = (A \times S(15) + 4 \times B \times S(5)) \times D \\
SA(4, 5) = (A \times S(14) + 2 \times B \times S(21)) \times D \\
SA(4, 6) = (S(10) + 2 \times B \times S(6)) \times D \\
SA(4, 7) = 4 \times B \times S(6) \quad *D \\
SA(4, 8) = 2 \times B \times S(2) \quad *D \\
SA(4, 10) = 2 \times \text{FNU} \times S(10) \quad *D \\
SA(4, 11) = \text{FNU} \times S(9) \quad *D \\
SA(4, 12) = A \times S(5) \quad *D \\
SA(5, 5) = (A \times S(13) + B \times S(11)) \times D \\
SA(5, 6) = (S(9) + A \times S(3)) \times D \\
SA(5, 7) = 2 \times B \times S(3) \quad *D \\
SA(5, 8) = 9 \times S(1) \quad *D \\
SA(5, 10) = 2 \times \text{FNU} \times S(9) \quad *D \\
SA(6, 11) = \text{FNU} \times S(8) \quad *D \\
SA(6, 12) = S(2) / 2. \quad *D \\
SA(6, 6) = (2 \times S(5) + B \times S(4)) \times D \\
SA(6, 7) = 2 \times B \times S(4) \quad *D \\
SA(6, 8) = B \times S(3) \quad *D \\
SA(6, 10) = 4 \times \text{FNU} \times S(5) \quad *D \\
SA(6, 11) = 2 \times \text{FNU} \times S(2) \quad *D \\
SA(6, 12) = (2 \times \text{FNU} + B) \times S(6) \quad *D \\
SA(7, 7) = 4 \times B \times S(4) \quad *D \\
SA(7, 8) = 2 \times B \times S(3) \quad *D


\[ S(10) = \left\{ \frac{(x+y) - 3*(7z + z*x + x*x) + z*x}{B-C} \right\} \]

\[ + \left( \frac{y*y - 4 - z*x*(z*z + z*x + x*x)}{C-A} \right) \]

\[ \frac{X * Y * (X * X + X * Y + Y * Y)}{20} \]

\[ + \frac{(Z - X)}{5} \]

\[ - B*B*(Z*Z+Z*X+X*X+2.*Y*(Z+X)+3.*Y*Y)))/60 \]

RETURN

END

*FORTRAN SOURCE

SUBROUTINE LAST 5(R, Z, S)

DIMENSION R(3), Z(3), A(3), B(3), S(15)

TOL = 1.0E-04

DO 7 I = 1, 3

J = I + 1

IF (J-I) 28, 30, 30

J = 1

CONTINUE

28 CONTINUE

J = I

CONTINUE

30 CONTINUE

R J M I = R(J)-R(I)

IF (ABS(R J M I) - TOL) 4, 4, 3

A(IJ) = 0.0

B(IJ) = 0.0

\[ A(IJ) = R(J)*Z(I) - R(I)*Z(J) / R(J)^2 \]

GO TO 7

A(IJ) = (R(J)*Z(I) - R(I)*Z(J)) / R(J)^2

B(IJ) = (Z(J)-Z(I)) / R(J)^2

R J 2 M I = R(J)^2

R J 3 M I = R(J)^3

R J 4 M I = R(J)^4

R J 5 M I = R(J)^5

S(I1) = S(I1) + A(KI)*RLOG

S(I2) = S(I2) + (A(KI)+A(IJ))*A(KI)*RLOG

S(I3) = S(I3) + (A(KI)+A(IJ))*A(IJ)*RLOG

S(I4) = S(I4) + (A(KI)+A(IJ))*A(IJ)*A(IJ)*RLOG

S(I5) = S(I5) + (A(KI)+A(IJ))*A(IJ)*A(IJ)*A(IJ)*RLOG

RETURN

END

*FORTRAN SOURCE

SUBROUTINE SLOG(TOL, A, B, R, S)

DIMENSION A(3), B(3), R(3), S(15)

DO 1 I = 1, 3

IF (ABS(R(I)) - TOL) 1, 1, 2

CONTINUE

1 CONTINUE

I = I

K I = I + 2

CONTINUE

2 CONTINUE

K I = K I - 3

CONTINUE

3 CONTINUE

RIL O G = ALOG(R(I))

A K M J R I = A(KI) - A(IJ) * RIL O G

S(I1) = S(I1) + A K M J R I

S(I2) = S(I2) + A(KI)+A(IJ) * AKMJRI / 2.


X = A(KI)

Y = A(IJ)

RETURN

END
\[ \text{AK5MJS} = (x-y) \times (x^2 + x^2 y + x^2 y^2 + x^2 y^3 + y^4) \]

\[ \text{S(15)} = \text{S(15)} + \text{AK5MJS} \times \text{RILOG/5}. \]

1 CONTINUE
RETURN
END

*FORTRAN SOURCE
SUBROUTINE FORMOD(SA, AD, QD)
DIMENSION SA(12,12), AD(12,12), SD(12,12), QD(12,12)

C POST-MULTIPLYING THE MATRIX(SA) BY THE MATRIX(AD)

DO 310 I=1,12
DO 309 J=1,12
SD(I,J)=0.0
DO 308 K=1,12
SD(I,J)=SD(I,J)+SA(I,K)*AD(K,J)
308 CONTINUE
309 CONTINUE
310 CONTINUE
WRITE(2,16)
16 FORMAT(1H1//56X, 9HMATRIX SD/)
CALL PRIDOZ(SD)

C ZEROING MATRIX SA (HAVING FINISHED WITH IT IN ITS INITIAL FORM)

DO 602 I=1,12
DO 601 J=1,12
SA(I,J)=0.0
601 CONTINUE
602 CONTINUE

C TRANSPOSING THE MATRIX AD (STORING THIS IN SA)

DO 834 I=1,12
DO 833 J=1,12
SA(J,I)=AD(I,J)
833 CONTINUE
834 CONTINUE
WRITE(2,15)
15 FORMAT(1H1//56X, 9HMATRIX AD TRANSPOSED/)
CALL PRIDOZ(SA)

C USING TAD TRANSPOSED TO PRE-MULTIPLY PROD(=ST*TAD)

DO 410 I=1,12
DO 409 J=1,12
QD(I,J)=0.0
DO 408 K=1,12
QD(I,J)=QD(I,J)+SA(I,K)*SD(K,J)
408 CONTINUE
COMPLETING THE SIX NODES OF THE TRIANGLE

HERE THE EVEN SUFFIXES OF NODSIX ARE FORMED AND ENTERED

ALSO THEIR DIMENSIONS ARE FORMED AND ENTERED

DO 227 I=2,6,2
  J=I-1
  K=I+1
  IF(K-6)228,230,231
  K=K-6
  GO TO 228
230 WRITE(2,233)
  233 FORMAT(1X,3HK=6/)
228 CONTINUE
  NA=NODSIX(J)
  NB=NODSIX(K)
  NODSIX(I) = (NA + NB)/2
  NC = NODSIX(I)
  R(NC)=(P(NA) + R(NB))/2.
  Z(NC) = (Z(NA) + Z(NB))/2.
227 CONTINUE
DO 528 I=1,6
  NODE = NODSIX(I)
  RAD(I) = R(NODE)
  AXI(I) = Z(NODE)
528 CONTINUE
WRITE(2,529)
  529 FORMAT(1X,6(14,2F8.3))
529 CONTINUE
RETURN
SUBROUTINE ADDELE(NTWELV,NDSPRL,IAXIAL,ASEMBY,NASSEM,REARGN)

DIMENSION NTWELV(12),ASEMBY(NASSEM,NASSEM),REARGN(12,12)

IDISMR = 1 DIRECTION DISPLACEMENT NUMBER MINUS

TWICE THE NUMBER OF DISPLACEMENT NUMBERS PER RADIAL LINE OF

PHYSICAL SQUARES TIMES ONE LESS THAN THE AXIAL ROW NUMBER

ADDITION ELEMENT TO ASSEMBLY

---------

ICONST=2*NDSPRL*(IAXIAL-1)

DO 703 I=1,12

IDISPN = NTWFLV(I)

DO 704 J=1,12

JDISPN = NTWFLV(J)

ASEMBY(IDISMR,JDISMR) = ASEMBY(IDISMR,JDISMR) + REARGN(I,J)

704 CONTINUE

WRITE(2,16)

16 FORMAT(1H1/49X,23H PART OF ASSEMBLY MATRIX/) A

CALL PNTASP(ASEMBY,NTWELV,ICONST,NASSEM)

RETURN

END

SUBROUTINE PNSMSQ(LARSQR,NDSPRL,ISMSQ,JSMSQ,SMSQ)

DIMENSION SMSQ(NDSPRL,NDSPRL)

TO PRINT OUT SMSQ AND ITS INDICES

WRITE(2,4)LARSQR,ISMSQ,JSMSQ

4 FORMAT(1H1/16X,20HLARGE SQUARE INDEX =,I2,7X,

1 24HS SMALL SQUARE ROW INDEX =,I2,7X,

2 27HSMALL SQUARE COLUMN INDEX =,I2//)

WRITE(2,206)(J,J=1,NDSPRL)

206 FORMAT(4X,14(5X,I3)//)

DO 301 I=1,NDSPRL

WRITE(2,208)(I,(SMSQ(I,J),J=1,NDSPRL))

301 CONTINUE

RETURN

SUBROUTINE PRMAHD(N,A)

DIMENSION A(6,6)

PRMAH O = PRINT MATRIX HALF DOZEN

---------

DO 701 I=1,N

WRITE(2,702)(A(I,J),J=1,N)

702 FORMAT(6F20.12)

RETURN
SUBROUTINE PNTASPI (A, NTWELV, ICONST, NASSEM)
DIMENSION A(NASSEM, NASSEM, NTWELV(12))
WRITE(2, 206) (NTWELV(J), J=1, 12)
206 FORMAT(4X, 13, 7X, 13, 7X, 13, 7X, 13, 7X, 13, 7X, 13, 7X, 13, 7X, 13, 7X, 13, 7X,
J = NTWELV(1) - ICONST
J = NTWELV(2) - ICONST
J = NTWELV(3) - ICONST
J = NTWELV(4) - ICONST
J = NTWELV(5) - ICONST
J = NTWELV(6) - ICONST
J = NTWELV(7) - ICONST
J = NTWELV(8) - ICONST
J = NTWELV(9) - ICONST
J = NTWELV(10) - ICONST
J = NTWELV(11) - ICONST
J = NTWELV(12) - ICONST
DO 301 I=1, 12
K = NTWELV(I) - ICONST
WRITE(2, 207) K
207 FORMAT(1X, 13)
WRITE(7, 208) (A(I, J), J=1, 12)
208 FORMAT(1X, 1P12E10.3/)
301 CONTINUE
RETURN
END

SUBROUTINE PPIDOZ (A)
DIMENSION A(12, 12)
WRITE(2, 206) (J, J=1, 12)
206 FORMAT(9X, 14, 18)
DO 207 I=1, 12
WRITE(2, 207) I
207 FORMAT(1X, 13)
WRITE(2, 208) (A(I, J), J=1, 12)
208 FORMAT(3X, 1X, 13, 3X, 1P14E8.1)
301 CONTINUE
RETURN
END

DIMENSIONS A(12, 12)
WRITE(2, 206) (J, J=1, 12)
206 FORMAT(5X, T2, 8X, T2, 8X, I2, 8X, I2, 8X, I2, 8X, I2, 8X, I2, 8X, I2, 8X, I2, 8X, I2, 8X, 1)
207 FORMAT(IX, 12)
WRITE(2, 208) (A(I, J), J=1, 12)
208 FORMAT(1X, 1P12E10.3/)
RETURN
END
"ENTER"

DATA
GO12DATA
.1050
561
30.0 E 06 0.3
1000

*FORTRAN SOURCE:
INPUT PARAMETERS

NPHSQR = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG RADIAL LENGTH
NPHSQA = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG AXIAL LENGTH

DERIVED PARAMETERS

NUMSQS = NUMBER OF SQUARES (TOTAL)
NUMTRI = NUMBER OF TRIANGLES
NTVRNO = NUMBER OF VERTEX NODES TO SPECIFY TRIANGLES
NRADWI = NUMBER OF DISTINCT RADII (INCLUDING I/R AND O/R) IN WIDTH
(N.E.SIDES OF PHYSICAL SQUARES)
NAXILN = NUMBER OF DISTINCT AXIAL LENGHTS INCLUDING BOTH ENDS
(N.E.SIDES OF PHYSICAL SQUARES)
NTVRNO = NUMBER OF DISTINCT VERTEX NODES
NODPRL = TOTAL NODES PER RADIAL WIDTH
NODPALN = TOTAL NODES PER AXIAL LENGTH
NUMNOD = TOTAL NUMBER OF NODES
LASMAT = NUMBER OF ROWS OF LARGE ASSEMBLY MATRIX
NDSRL = NUMBER OF DISPLACEMENTS PER RADIAL LENGTH
ASEMBY = ASSEMBLY (PARTIAL) OF ELEMENTS IN ONE ROW (PHYSICAL)
NASMRY = SIZE OF ASSEMBLY
NTRAT = WIDTH OF TREATMENT MATRIX

10 FORMAT (212)
READ (1,10) NPHSQR,NPHSQA
WRITE (2,20) NPHSQR,NPHSQA
20 FORMAT (1H1/2X,52H NUMBER OF ROWS OF PHYSICAL SQUARES PER RADIAL WIDTH,D34)
1TH=12,2X,52H NUMBER OF ROWS OF PHYSICAL SQUARES PER AXIAL LENGTH=D34
212/1)
C
NUMSQS=NPHSQR*NPHSQA
WRITE (2,21) NUMSQS
21 FORMAT (1H0,2AH TOTAL NUMBER OF RECTANGLES = ','I4)
C
NUMTRI=2*NUMSQS
WRITE (2,22) NUMTRI
22 FORMAT (1H1,20H NUMBER OF TRIANGLES = ','I4)
C
NTVRNO=3*NUMTRI
WRITE (2,23) NTVRNO
23 FORMAT (1H1,58H NUMBER OF VERTEX NODES REQUIRED TO SPECIFY ALL TRIANGLES)
1GLES = ','I5)
C

NRADWI = NPHSQR + 1
WRITE(2,24)NRADWI
24 FORMAT(1H,35HNUMBER OF DISTINCT RADII IN WIDTH =,12)
MAXILN = NPHSQA + 1
WRITE(2,25)MAXILN
25 FORMAT(1H,33HNUMBER OF DISTINCT AXIAL LENGTHS =,13)
NDVRNO = MAXILN * NRADWI
WRITE(2,26)NDVRNO
26 FORMAT(1H,33HNUMBER OF DISTINCT VERTEX NODES =,14)
NDPRWI = 2 * NPHSQR + 1
WRITE(2,27)NDPRWI
27 FORMAT(1H,40HTOTAL NUMBER OF NODES PER RADIAL WIDTH =,12)
NDPALN = 2 * NPHSQA + 1
WRITE(2,28)NDPALN
28 FORMAT(1H,40HTOTAL NUMBER OF NODES PER AXIAL LENGTH =,14)
NUMNOD = NDPRWI * NDPALN
WRITE(2,29)NUMNOD
29 FORMAT(1H,23HWHOLE NUMBER OF NODES =,15)
LASMAT = 2 * NUMNOD
WRITE(2,30)LASMAT
30 FORMAT(1H,41HTOTAL NUMBER OF ROWS OF LARGE ASSEMBLY MATRIX =,15)
NDSPRL = 2 * NDPRWI
WRITE(2,31)NDSPRL
31 FORMAT(1H,43HNUMBER OF DISPLACEMENTS PER RADIAL LENGTH =,12)
NASSEM = 3 * NDSPRL
WRITE(2,32)NASSEM
32 FORMAT(1H,33HSIZE OF THE ROW ASSEMBLY MATRIX =,13)
NTREAT = 5 * NDSPRL
WRITE(2,33)NTREAT
33 FORMAT(1H,27HTWIDTH OF TREATMENT MATRIX =,13)
NTWOSQ = NUMBER OF EXTRA DISPL. FOR EACH EXTRA ROW OF RADIAL RECTANG
NTWOSQ = 2 * NDSPRL
WRITE(2,34)NTWOSQ
34 FORMAT(1H,77HNUMBER OF EXTRA DISPLACEMENTS FOR EACH ADDITIONAL ROW
1 OF RADIAL) OF RECTANGLES =,13)
CALL DECOMP(NDSPRL, NASSEM, NTREAT, NTWOSQ, NPHSQR, NPHSQA)
STOP
END
*FORTRAN SOURCE
SUBROUTINE DECOMP(NDSPRL, NASSEM, NTREAT, NTWOSQ, NPHSQR, NPHSQA)
C
18TH MARCH 1970
C
DIMENSION TREATM(NASSEM, NTREAT), SMSQ(NDSPRL, NDSPRL), LOC(NTWOSQ)
C
DIMENSION LESTRA(100)
C
CALL TIME(A)
THIS PROGRAM IS TO DECOMPOSE THE ASSEMBLY STIFFNESS MATRIX INTO
LOWER AND UPPER MATRICES USING PARTIAL PIVOTING

LIST OF SUBROUTINES FOR DECOMPOSITION

- TAPDIS
  - PNSMSQ
  - ZFSMSQ
- FIR3SQ
  - WRTBSQ
  - ZETRET
  - (PNSMSQ)
  - (ADTRET)
- NEXDOZ
  - (PNSMSQ)
  - (ADTRET)
  - (WRTBSQ)
- LUDEPY
- WTOTAP
  - PPNVEC
  - WTRETJ
  - (PNSMSQ)
  - (PNSMSQ)
  - (PNSMSQ)
- WTAPEN
  - (PPNVEC)
  - (PNSMSQ)
- TRANSF

IASSEM=14
IUPTAP=15
LOWTAP=16
NDISC=20
CALL TAPDIS(SMSG,NPHSQR,NPHSGA,NDSPRL,NDISC,IASSEM)

NDT1M2=2*NDSPRL
NDT2P1=NDT1M2+1
NDT1M3=3*NDSPRL
NDT1M5=5*NDSPRL
NI=NDT1M3
NJS=NDT1M2
LMLES=1
CALL FIR3SQ(NDSPRL,TREATM,NCOLN,LARSQR,ISMSQ,NDISC,NASSEM,NTREAT)

LARSQR=1
NCLMD=5
NJ=NDT1M5
CALL RFSTA(NRESTA,LESTRA)

GO TO 2

1 CONTINUE

NCLNO=3
NJ=NDTIM3

2 CONTINUE

CALL NEXDOZ(NDSPRL,TREATM,NCOLN,LARSQR,1MSQ,NPHSQA,NDISC,NASSEM,INTREAT)

INSTAN=2

CALL PNTRET(NDSPRL,TREATM,LARSQR,NASSEM,NTREAT,INSTAN)

CALL RFSTRA(LARSQR,NDSPRL,NRESTA,LESTRA,TREATM,NJ,NASSEM,INTREAT,LASMAT,LMLES)

INSTAN=3

CALL PNTRET(NDSPRL,TREATM,LARSQR,NASSEM,NTREAT,INSTAN)

CALL LUDEPV(TREATM,LCC,NI,NJ,NJS,NASSEM,NTREAT,NTWOSQ)

CALL WTOTAP(LOWTAP,1UPTAP,LARSQR,TREATM,LOC,NDSPRL,NJS,NCLNO,NASSEM,NTREAT,NTWOSQ)

CALL ZEORES(LARSQR,NDSPRL,NRESTA,LESTRA,TREATM,NJ,NASSEM,INTREAT,LASMAT,LMLES)

CALL TRANSF(TREATM,NDTIM2,NDTIM3,NDTIM5,MASSEM,MTREAT,NDSPRL)

LARSQR=LARSQR+1

IF(LARSQR-NPHSQA)2,1,3

3 CONTINUE

NI=NDSPRL
NCLNO=1
NJ=NDSPRL
NJS=NDSPRL-1

CALL LUDEPV(TREATM,LOC,NI,NJ,NJS,NASSEM,NTREAT,NTWOSQ)

CALL WTAPEN(LOWTAP,1UPTAP,LARSQR,TREATM,LOC,NDSPRL,NJS,NCLNO,NASSEM,NTREAT,NTWOSQ)

WRITE(2,999)

999 FORMAT(1H1/5X,1RHEND OF CALCULATION/)

CALL TIME(B)

TIM=B-A

WRITE(2,3600)TIM

3600 FORMAT(1H1/4X,12HTIME TAKEN =,I4,8HSECONDS,/)

STOP

END

SUBROUTINE TAPDIS(SMSQ,NPHSQR,NPHSQA,NDSPRL,NDISC,NASSEM)

DIMENSION SMSQ(NDSPRL,NDSPRL)
16 NOVEMBER 69

PROGRAM TO TRANSFER DATA FROM TAPE TO DISC

---

NONENCLATURE
---

NSPRL = NUMBER OF DISPLACEMENTS PER RADIAL LINE OF PHYSICAL SQUARE
NPQS = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG THE AXIAL LENGTH
NPQS2 = NPQS-2, USED FOR TAILING OFF THE BANDED MATRIX
NPQS3 = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG THE RADIAL LENGTH
ISMSQ = ROW INDEX OF SMALL SQUARE
JSMSQ = BEGINNING COLUMN INDEX OF SMALL SQUARE
IROW = ROW INDEX = 1, 2, OR 3
IROWE = END ROW INDEX
J0 = COLUMN INDEX WITHIN SMALL SQUARE BEGINING (SPACIAL)
JE = COLUMN INDEX WITHIN SMALL SQUARE ENDING (SPACIAL)
IB = ROW INDEX WITHIN SMALL SQUARE BEGINING (SPACIAL)
IE = ROW INDEX WITHIN SMALL SQUARE ENDING (SPACIAL)
IP = INDEXING ROW
NCOLN = NUMBER OF COLUMNS OF SMALL SQUARES
NSPRL = NUMBER OF DISPLACEMENTS PER RADIAL LINE OF PHYSICAL SQUARE
JSMSQ = COLUMN INDEX OF SMALL SQUARE (GLOBAL)
JCOLN = COLUMN INDEX 1 TO NCOLN
SMQ = SMALL SQUARE = PARTICNED MATRIX
NROWSQ = NUMBER OF ROWS IN MATRIX
ISMSQM = MAXIMUM NUMBER OF ROWS OF SMALL SQUARES IN WHOLE MATRIX

REWIND IASSEM
READ(IASSEM)KOUNT
NCOUNT=0

SETTING INITIAL VALUES
---

THROW=1.0
NROWSQ=2*NPQS+1
NSPRL=2*NROWSQ
ISMSQM=2*NROWSQ+1
NPQS2=ISMSQM-2
ISMSQ=1
JSMSQ=1
IROW=1
IROWE=3
J0=1

BEGINING OF MAIN LOOP SETTING UP DATA ON DISK (EXCEPT FOR LAST PART)
---

1 CONTINUE
IF(ISMSQ*NSPRL
ID=IE-NSPRL+1
IR=IB

2 CONTINUE
IF(IROWE-IROW)3, 4, 5

3 CONTINUE
CALL TRACE
STOP
4 CONTINUE
NCOLN=7
GO TO 6
5 CONTINUE
NCOLN=5
GO TO 6

THIS SECTION (TO 16) IS TO READ FROM TAPE FOR THE FIRST 3 OR 5 SMALL SQUARES AND PUT ON TO DISC WITH TWO ZERO SMALL SQUARES.

6 CONTINUE
JF=JB+NCOLN*NDSPRL-1
JSMSQ=JSMSQ+1
WRITE(INDISC)ISMSQ,NCOLN,JB,JE
DO 16 JCOLN=1,NCOLN
IF(JCOLN-NCOLN+2)7,8,9
7 CONTINUE
JCOLN LESS THAN NCOLN
8 CONTINUE
JCOLN=NCOLN
NCOUNT=NCOUNT+1
READ(IASSEM)ISNSQT,JSNSQT,JSNSQ,IAXIAL
CALL PNSMSQ(IAXIAL,NDSPRL,ISNSQT,JSNSQ,JSNSQ)
IF(ISMSQT-ISNSQ)10,11,12
9 CONTINUE
CALL ZESMSQ(NDSPRL,JSNSQ)
GO TO 14
10 CONTINUE
WRITE(2,41)ISMSQ,JSNSQ
CALL TRACE
STOP
11 CONTINUE
IF(JSMSQT-JSNSQ)13,14,15
12 CONTINUE
WRITE(2,42)JSMSQ,JSNSQ
CALL TRACE
STOP
13 CONTINUE
CALL TRACE
WRITE(2,43)JSMSQ,JSNSQ
STOP
14 CONTINUE
WRITE(INDISC)JSNSQ,JSNSQ,JSNSQ
JSNSQ=JSNSQ+1
GO TO 16
15 CONTINUE
WRITE(2,44)JSNSQ,JSNSQ
CALL TRACE
STOP
16 CONTINUE
IROW=IROW+1
TEST TO SEE IF END ROW
IF(IROW-IROWE)17,18,19
17 CONTINUE.
BEFORE END OF ROW

10 CONTINUE

END OF ROW

ISMSQ=ISMSQ+1
IF(ISMSQ-NPRSM2)21,22,23
10 CONTINUE
JSMSQB=JSMSQB+2
ISMSQ=ISMSQ+1
IROWF=2
IROW=1
JB=JB+2*NDSPRL
GO TO 1
21 CONTINUE
GO TO 1
22 CONTINUE

ISMSQ=NPRSM2
23 CONTINUE

ISMSQ.GT.NPRSM2

FOR LAST THREE ROWS OF SQUARES

IROW=1
IROWE=3
25 CONTINUE
IE=ISMSQ*NDSPRL
IB=IE-NDSPRL+1
IR=IB

26 CONTINUE
IF(IROW=2)27,28,29
27 CONTINUE
NCOLN=5
GO TO 30
28 CONTINUE
NCOLN=3
JSMSQB=JSMSQB+2
JB=JB+NCOLN*NDSPRL
GO TO 30
29 CONTINUE
NCOLN=3
GO TO 30

INDICES FOR BEGINING OF ROW

JE=JB+NCOLN*NDSPRL-1
JSMSQ=JSMSQB
WRITE(NDTSC)ISMSQ,NCOLN,JB,JE
GO 37 JCOLN=1,NCOLN
NCOUNT=NCOUNT+1
READ(IASSEM)ISMSQT,JSMSQT,SMSC,IXIAL
CALL PNSMSOIIAXIAL,NDSPRL,ISMSQT,JSMSQT,SMSC)
IF(ISMSQT-ISMSQ)31,32,33
CONTINUE
WRITE(2,41)ISMSQT,JSMSQ
CALL TRACE
STOP
CONTINUE
IF(JSMSQT-JMSQ)34,35,36
CONTINUE
WRITE(2,42)ISMSQT,JSMSQ
CALL TRACE
STOP
CONTINUE
WRITE(2,43)JSMSQT,JSMSQ
CALL TRACE
STOP
CONTINUE
WRITE(NDISC)ISMSQ,JSMSQ,SMSQ
J$MSQ=JSMSQ+1
GO TO 37
CONTINUE
WRITE(2,44)JSMSQT,JSMSQ
CALL TRACE
STOP
CONTINUE
IR0W=IB0W+1
UPDATE ROW INDEX
IF(IROW-IROWF)38,39,40
CONTINUE
TSMSQ=TSMSQ+1
GO TO 25
CONTINUE
GO TO 38
CONTINUE
END FILE NDISC
WRITE(2,45) !<0
UNT,NCOUNT
RETURN
FORMAT(1H1,38X,17HISMSQT FROM TAPE),I*, 20HLESS THAN 42
FORMAT(1H1/10X,20HNUMBER OF WRITES ON TAPE =,13/
END
*FORTRAN SOURCE
SUBROUTINE PNSMSQ(LARSQR, NOSPRL, ISMSQ, JSMSQ, SMSQ)
DIMENSION SMSQ(NDSPRL, NDSRRL)
TO PRINT OUT SMSQ AND ITS INDICES
WRITE(2,4)LARSQR, ISMSQ, JSMSQ
4 FORMAT(1H1/16X,20HLARGE SQUARE INDEX =,12,7X,
1 24HSMALL SQUARE ROW INDEX =,12,7X,
2 27HSMALL SQUARE COLUMN INDEX =,12,7//)
WRITE(2,206)(J,J=1, NDSPRL)
206 FORMAT(4X,14(5X,13)///)
DO 301 I=1, NDSPRL
WRITE(2,208)(J, ISMSQ(I,J), J=1, NDSPRL)
208 FORMAT(3X,12,2X,1P14E8.1/)
301 CONTINUE
RETURN
END
*FORTRAN SOURCE
SUBROUTINE ZESMSQ(NDSPRL, SMSQ)
FORMAT(1H1,38X,17HJSMSQT FROM TAPE),I4,20HLESS THAN IN ROUTINE, I4)TAD1211
FORMAT(1H1,38X,17HJSMSQT FROM TAPE),I4,20HMORE THAN IN ROUTINE, I4)TAD1212
FORMAT(1H1,38X,17HJSMSQT FROM TAPE),I4,20HMORE THAN IN ROUTINE, I4)TAD1213
FORMAT(1H1,38X,17HJSMSQT FROM TAPE),I4,20HMORE THAN IN ROUTINE, I4)TAD1214
FORMAT(1H1/10X,20HNUMBER OF WRITES ON TAPE =,13/
END
*FORTRAN SOURCE
SUBROUTINE ZESMSQ(NDSPRL, SMSQ)
SUBROUTINE FIR3SQ(NDSPRL, TREATM, NCOLN, LARSQR, ISMSQ, NDISC, NASSEM, INTREAT)

DIMENSION TREATM(NASSEM, NTREAT), SMSQ(NDSPRL, NDSPRL)

REWIND 20
READ(NDISC) ISMSQ, NCOLN, JB, JE

CALL WRTBSQ(ISMSQ, NCOLN, JB, JE)
LARSQR=1
CALL ZETRET(NDSPRL, TREATM, NASSEM, NTREAT)
DO 5 JCOL=1, NCOLN
READ(NDISC) ISMSQ, JMSQ, SMSQ
CALL PNSMSQ(LARSQR, NDSPRL, ISMSQ, JMSQ, SMSQ)
CALL ADTPET(LARSQR, NDSPRL, ISMSQ, JMSQ, SMSQ, TREATM, NASSEM, NTREAT)
5 CONTINUE
LARSQR=0
RETURN

SUBROUTINE WRTBSQ(ISMSQ, NCOLN, JB, JE)

WRITE(2,2) ISMSQ, NCOLN, JB, JE
2 FORMAT(1X, 17H NUMBER OF ROW INDEX=, 12, 2X, 29H NUMBER OF COLUMNS OF SQUARE =, 14, 5X, 14H COLUMN END)
RETURN

SUBROUTINE ZETRET(NDSPRL, TREATM, NASSEM, NTREAT)

DIMENSION TREATM(NASSEM, NTREAT)

ITREAT = 3*NDSPRL
JTREAT = 5*NDSPRL
DO 10 I=1, ITREAT
DO 20 J=1, JTREAT
TREATM(I, J)=0.0
10 CONTINUE
20 CONTINUE
RETURN

SUBROUTINE ADTPET(LARSQR, NDSPRL, ISMSQ, JMSQ, SMSQ, TREATM, NASSEM, INTREAT)

ROUTINE TO ADD SMSQ TO APPROPRIATE PART OF TREATM

DIMENSION SMSQ(NDSPRL, NDSPRL), TREATM(NASSEM, NTREAT)

CHILTON 42*42 126,210
Subroutine NXDOZ (NDSPRL, TREATM, NCOLN, LARSQR, ISMSQ, NPHSQA, NDISC, NASSEM, NTREAT)

Dimension SMSQ (NDSPRL, NDSPRL), TREATM (NASSEM, NTREAT)

ISMSQM = 2 * NPHSQA + 1
NPRSM2 = ISMSQM - 2
IF (ISMSQ - NPRSM2 < 0, 10, 21)
10 CONTINUE
DO 20 JCOL = 1, 2
READ (NDISC) ISMSQ, JSMSQ, SMSQ
CALL PNSMSQ (LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ)
CALL ADTRET (LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ, TREATM, NASSEM, NTREAT)
20 CONTINUE
21 CONTINUE
INDEX = 0
30 CONTINUE
INDEX = INDEX + 1
NDSPRL = INDEX * NDSPRL
READ (NDISC) ISMSQ, NCOLN, JB, JE
CALL WRTBSQ (ISMSQ, NCOLN, JB, JE)

IF (5 - NCOLN < 0, 50, 60, 70)
50 CONTINUE
60 CONTINUE
NSTOP = 5
GO TO 90
70 CONTINUE
NSTOP = 3
GO TO 80
80 CONTINUE
DO 90 JCOL = 1, NSTOP
READ (NDISC) ISMSQ, JSMSQ, SMSQ
CALL PNSMSQ (LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ)
CALL ADTRET (LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ, TREATM, NASSEM, NTREAT)
90 CONTINUE
IF(INDEX-1)100,110,120
100 CONTINUE
WRITE(2,99)INDEX
99 FORMAT(1HI,10X,17HINDEX IN NEXDOZ = ,I2/
STOP
110 CONTINUE
GO TO 30
120 CONTINUE
RETURN
END

*FORTAN SOURCE
SUBROUTINE LUPD(T,LOC,NI,NJ,NJS,NASSEM,NTREAT,NTWOSQ)
DIMENSION T(NASSEM,NTREAT),LOC(NTWOSQ)
DIMENSION P(NTWOSQ),FACTOR(NASSEM)
WRITE(2,300)NI,NJ,NJS,NASSEM,NTREAT,NTWOSQ
300 FORMAT(IHO,4HNI =,I3,5X,4HNJ =,I3,5X,5HNJS =,î
37 3HMASSEM =,I3,
38 1?X,PHNTPT-:AT =,I3,5X,9HNTWOSQ =,I3/)

LOWER-UPPER DECOMPOSITION WITH PARTIAL PIVOTING
---------------------------------------------
INTEP-CHANGING ROWS FOR LARGEST ELEMENT IN PIVOTAL COLUMN TO
BECOME PIVOT, AND KEEPING RECORD OF ROW INTERCHANGE NUMBERS IN LOC

NOMENCLATURE
--------------
T(I, J) = TREATMENT MATRIX
LOC(J) = LOCATION ARRAY
NI = NUMBER OF ROWS IN TREATMENT MATRIX
NJ = NUMBER OF COLUMNS IN TREATMENT MATRIX
NJS = NUMBER OF COLUMNS SEARCHED =NI-1 OR LESS

10 DO I=1,NJS
LOC(I)=I
10 CONTINUE

J=PIVOT ROW COUNT
---------------------------------------------
DO 100 J=1,NJS
100 FINDING LARGEST ELEMENT IN COLUMN J

BIG=0.0
WRITE(2,208)(J,(T(K,J),K=J,NI))
208 FORMAT(2X,I2,1X,(1X,1P14F8.1)/)
DO 30 K=J,NI
 IF(BIG-ABS(T(K,J))15,25,25
15 CONTINUE
BIG=ABS(T(K,J))
L=K
30 CONTINUE
IF(L-J)40,70,50

40 STOP

50 DO 60 K=J,NJ
   STORED=T(J,K)
   T(J,K)=T(L,K)
   T(L,K)=STORED

60 CONTINUE

LOC(J)=L

70 CONTINUE
P(J)=T(J,J)

THIS IS END OF SECTION WHERE INTER-CHANGING ROWS (WHERE REQUIRED) IS CARRIED OUT, AND THE ROW CHANGED WITH ROW(J) IS ROW(L) AND THIS IS RECORDED IN LOC(J), L IS GREATER OR EQUAL TO J

REMAINING ROW COUNT N

JP1=J+1

DO 90 M=JP1,NI
   MULTIPLIERS
   T(M,J)=T(N,J)/T(J,J)
   (THIS SHOULD BE LESS THAN 1)
   FACTOR(M)=1.0-ABS(T(M,J))
   COLUMN COUNT K

   DO 80 K=JP1,NJ
   REMAINING ROW MINUS MULTIPLIER TIME PIVOT ROW

   T(M,K)=T(M,K)-T(M,J)*T(J,K)

80 CONTINUE

END OF A REMAINING ROW, FOR ONE MULTIPLIER

90 CONTINUE
   WRITE(2,30R)(J,(FACTOR(M),M=JP1,NI))
30R FORMAT(1H,15HCOLUMN NUMBER =,I2/(1X,1P15F8.1))

END OF A COLUMN OF MULTIPLIERS FOR ONE PIVOT

100 CONTINUE

END OF PIVOTS (NJS OF THEM)

WRITE(2,400)((J,P(J)),J=1,NJS)
400 FORMAT(1H,1HPIVOT VALUE,/(1X,6(13,1PE17.10)))
SUBROUTINE WTOTAP(LOWTAP, IUPTAP, LARSQR, TREATM, LOC, NDSRPL, NJS, NCOLWTAP, NASSEM, NTREAT, NTWOSQ)

* FORTRAN SOURCE *

DIMENSION LOC(NTWOSQ), TREATM(NASSEM, NTREAT)

THIS SEGMENT IS TO WRITE LOC (RECORD OF ROW INTER-CHANGES), AND THE TREATM TAPE.

AND AN UPPER PORTION TO IUPTAP TAPE

* * * * * * * * *
I I AND * * * * *

WRITING TO LOWTAP

-------------------
WRITE(LOWTAP) LARSQR, LOC

WRITE(2, 6) LARSQR

6 FORMAT(1H1, 52X, 15HLARSQR NUMBER =, I3)

CALL PRNVECILOC, NJS, NTWOSQ)

JSMSQB = 2*LARSQR - 1
ISMSQB = 2*LARSQR - 1
JSMSQO = JSMSQB - 1
ISMSQO = ISMSQB - 1

DO 10 J = 1, 2
JSMSQO = JSMSQO + 1
ISMSQO = ISMSQO + 1
ISMSQO = ISMSQO - 1
NRONU = 4 - J

CALL WTRETJ(NDSRPL, TREATM, J, ISMSQO, JSMSQO, NRONU, LARSQR, LOWTAP, NASSEM, NTREAT)

10 CONTINUE

WRITING TO IUPTAP

-------------------
ISMSQO = ISMSQO - 1
JSMSQO = JSMSQO - 1

ISMSQO = ISMSQO + 1

DO 20 I = 1, 2
ISMSQO = ISMSQO + 1
SUBROUTINE INFPRNVEC(L,N,NTWOSQ)
C  THIS SUBROUTINE IS TO PRINT OUT A VECTOR OF INTEGERS
C
DIMENSION L(NTWOSQ)

WRITE(2,100)(L(I),I=1,N)
100 FORMAT(1X,3014/)
RETURN
END

SUBROUTINE WTRETJ(NDSPRL,TREATM,J,ISMSQO,JSMSQO, NCLNO,LARSQR, WREJ
C  THIS SEGMENT IS TO WRITE THE MULTIPLIERS ON TO TAPE IN COLUMN FORM
C
WRITE(LOWTAP)LARSQR,JSMSQO,NRCNU
WRITE(2,6)LARSQR,JSMSQO,NRCNU
6 FORMAT(/9X,15MLARSQR NUMBER =,I2,10X,15HCOLUMN NUMBER =I3,10X,
24HNUMBER OF ROWS OF SMSQ =,I3/)

MR0PJL=NRCNU+J-1
DO 2 I=J,MR0PJL
2 CONTINUE
DO 9 JSM=1,NDSPRL
JTM=JSM*(J+1-1)*NDSPRL
DO 7 ISM=1,NDSPRL
ITM=ISM*(I-1)*NDSPRL
SMSQ(ISM,JSM)=TREATM(ITM, JTREAT)
7 CONTINUE
8 CONTINUE
WRITE(LOWTAP)LARSQR,JSMSQO,JSMSQO,SMSQ
CALL PNSMSQ0(LARSQR,NDSPRL,ISMSQO,JSMSQO,SMSQ)
C
RETURN
END

SUBROUTINE WTRET(F,NDSPRL,TREATM,I,ISMSQO,JSMSQO,NCLNO,LARSQR, WREI
C
C TRANSFERING A PART OF TREATM TO SMSQ
C
DO 9 JSM=1,NDSPRL
JTM=JSM*(J+1-1)*NDSPRL
DO 7 ISM=1,NDSPRL
ITM=ISM*(I-1)*NDSPRL
SMSQ(ISM,JSM)=TREATM(ITM, JTREAT)
7 CONTINUE
8 CONTINUE
WRITE(LOWTAP)LARSQR,JSMSQO,JSMSQO,SMSQ
CALL PNSMSQ0(LARSQR,NDSPRL,ISMSQO,JSMSQO,SMSQ)
C
C
RETURN
END
DIMENSION TREATM(NASSEM,NTREAT),SMSQ(NDSPRL,NDSPRL)

THIS SEGMENT IS TO WRITE UPPER PART ON TO TAPE IN ROWS OF SMSQ

WRITE(IUPTAP)LARSQR,ISMSQO,NCLNO
WRITE(2,81)LARSQR,ISMSQO,NCLNO
81 FORMAT(/9X,15HLAPSQR NUMBER =,12,10X,12HROW NUMBER =,I3,10X,
1 27HNUMBER OF COLUMNS OF SMSQ =,I3/)
NCLPI=NCLNO+I-1
DO 9 J=1,NCLPI
JSMSQO=JSMSQO+1

TRANSFERRING A PART OF TREATM TO SMSQ

DO 8 ISM=1,NDSPRL
ITREAT=ISM+(I-1)*NDSPRL
DO 7 JSM=1,NDSPRL
SNSQ(ISM,JSM)=TREATM(ITREAT,JTREAT)
7 CONTINUE
8 CONTINUE

WRITE(IUPTAP)LARSQR,ISMSQO,NCLNO
WRITE(2,81)LARSQR,ISMSQO,NCLNO
CALL PNSMSQ(LARSQR,NDSPRL,ISMSQO,JSMSQO,SMSQ)
9 CONTINUE

RETURN
END

*FORTRAN SOURCE
SUBROUTINE TRANSF(TREATM,NDTIM2,NDT2P1,NDTIM3,NDTIM5,NASSEM,NTREAT)
DIMENSION TREATM(NASSEM,NTREAT)

THIS SEGMENT IS TO BE USED AFTER DATA HAS BEEN TRANSFERRED TO TAPE.

IT IS TO TRANSFER THE LAST THREE SMSQ OF TREATM TO THE 1ST THREE.

NOMENCLATURE

NDTIM2 = NDSPRL + NDSPRL
NDT2P1 = NDTIM2 + 1
NDTIM3 = NDTIM2 + NDSPRL
NDTIM5 = NDTIM2 + NDTIM3

PUTTING LAST 3 TREATM INTO FIRST 3 TREATM
DO 20 I=NDT2P1,NDT1M3
   ISA=I-NDT1M2
   DO 10 J=NDT2P1,NDT1M5
      JSA=J-NDT1M2
      TREATM(ISA,JSA)=TREATM(I,J)
 10    CONTINUE
20 CONTINUE

RETURN
END

*FORTRAN SOURCE

SUBROUTINE WTPEN(LOWTAP,UPUTAP,LARSQR,TREATM,LOC,NDSPRL,NJS,NCOLN)
   1,NASSEM,NTREAT,NTWOSQ)
   DIMENSION LOC(NTWOSQ),TREATM(NASSEM,NTREAT),SMSQ(NDSPRL,NDSPRL)
   THIS IS THE LAST WRITE TO BOTH TAPES
   IT WRITES LOC (RECORD OF ROW INTER-CHANGES), AND THE FIRST SMSQ OF
   TREATM CONTAINING THE MULTIPLIERS) TO LOWTAP TAPE
   AND THE FIRST SMSQ OF TREATM CONTAINING UPPER PORTION TO UPUTAP
   TAPE
   WRITE(LOWTAP)LARSQR,LOC
   WRITE(2,6)LARSQR
   6 FORMAT(1X,52X,15HLARSQR NUMBER =,I3)
   CALL PRNVEC(LOC,NJS,NTWOSQ)
   JSMSQR=2*LARSQR-1
   ISMSQR=2*LARSQR-1
   NRONU=NCOLN
   TRANSFERRING PART OF TREATM TO SMSQ

DO 8 JSM=1,NDSPRL
   DO 7 ISM=1,NDSPRL
      SMSQ(ISM,JSM)=TREATM(ISM,JSM)
 7 CONTINUE
8 CONTINUE

WRITE(LOWTAP)LARSQR,JSMSQR,NRGNU
WRITE(2,62)LARSQR,JSMSQR,NRGNU
62 FORMAT(1X,15HNUMBER OF ROWS OFSMSQ =,I3)

WRITE(LOWTAP)LARSQR,JSMSQB,JSMSQB,SMSQ

WRITE(LOWTAP)LARSQR,JSMSQB,NRGNU
WRITE(2,62)LARSQR,JSMSQB,NRGNU
62 FORMAT(1X,15HNUMBER OF ROWS OFSMSQ =,I3)
CALL PNSMSQ(LARSQR,NDSPRL,ISMSQB,JSMSQR,SMSQ)
WRITE(IUPTAP)LARSQR,ISMSQB,NCCLN
WRITE(2,63)LARSQR,ISMSQB,NCGLA
63 FORMAT(9X,15MINLARSQR NUMBER =,I2,10X,12HRW NUMBER =,I3,10X,
1 2THNUMBER OF COLUMNS OF SMSQ =,I3/)
WRITE(IUPTAP)LARSQR,ISMSQB,JSMSQB,SMSQ
RETURN
END

*FORTRAN SOURCE
SUBROUTINE ZEROWI(A,NJ,IO,NASSEM,NTREAT)
DIMENSION A(NASSEM,NTREAT)

THIS SUBROUTINE IS TO ZERO ROW IO, EXCEPT PUTTING 1.0E 20 IN
A(IO,IO), IN ORDER TO APPLY A RESTRAT

ZERO-ING ROW IO

DO 10 J=1,NJ
A(IO,J)=0.0
10 CONTINUE

PUTTING 1.0E 20 IN PIVOT A(IO,IO)

A(IO,IO)=1.0E 20
WRITE(2,20)IO
20 FORMAT(1H1///27X,15HNUMBER OF ROW =,I2/)
RETURN
END

*FORTRAN SOURCE
SUBROUTINE REDSTA(NRESTA,LESTRA)

THIS PROGRAM IS TO READ IN THE NUMBER OF DISPLACEMENT POSITIONS
TO BE RESTRAINED, AND THEIR INDIVIDUAL NUMBERS

DIMENSION LESTRA(100)
510 FORMAT(12)
READ (1,510)NRESTA
WRITE(2,520)NRESTA
520 FORMAT(/9X,50HNUMBER OF DISPLACEMENT POSITIONS TO BE RESTRAINED =,I2)
521 FORMAT(16I5)
READ (1,521)(LESTRA(M),M=1,NRESTA)
WRITE(2,522)(LESTRA(M),M=1,NRESTA)
522 FORMAT(1H1,25I4)
RETURN
END

*FORTRAN SOURCE
SUBROUTINE PNTRET(NDSPRL,TREATM,LARSQR,NASSEM,NTREAT,INSTAN)

THIS SUBROUTINE IS TO PRINT OUT THE MATRIX TREATM, AND TO INDICATE PNTR 12
WHEN IT WAS PRINTED OUT. IT IS TO BE PRINTED OUT IN BLOCKS OF PNTR 13
SMT, IN FIVE COLUMNS OF THREE PARTITIONED MATRICES PNTR 14
DIMENSION TREATM(NASSEM, NTREAT), SMSQ(NDSPRL, NDSPRL) PNTR 15
WRITE(2,5)LARSQR, INSTAN PNTR 16
5 FORMAT(1H1///35X,15HLARSQR NUMBER =,12,17X,15HINSTAN NUMBER =,12) PNTR 17
DO 10 J=1,5 PNTR 18
  DO 9 I=1,3 PNTR 19
    DO 8 JSM=1,NDSPRL PNTR 20
      JTREAT=JSM+(J-1)*NDSPRL PNTR 21
     DO 7 ISM=1,NDSPRL PNTR 22
       ITREAT=ISM+(I-1)*NDSPRL PNTR 23
       SMSQ(ISM, JSM)=TREATM(ITREAT, JTREAT) PNTR 24
    CONTINUE PNTR 25
  CONTINUE PNTR 26
CALL PNSMSQ(LARSQR, NDSPRL, INSTAN, SMSQ) PNTR 27
10 CONTINUE RETURN PNTR 28
END PNTR 29

**FORTRAN SOURCE**

SUBROUTINE RESTRA(LARSQR, NDSPRL, NRESTA, LESTRA, TREATM, NJ, NASSEM, RESF 1
  NTREAT, LASMAT, LNMLES) RESF 2
  DIMENSION LESTRA(LASMAT), TREATM(NASSEM, NTREAT) RESF 3
  THIS SUBROUTINE IS TO APPLY RESTRAINTS WHERE REQUIRED. RESF 4
  NOMENCLATURE RESF 5
    NUPL5Q=NUMBER OF ROWS UP TO THE END OF A PARTICULAR LARGE SQUARE RESF 6
    NDOSBL=NUMBER OF DISPLACEMENT NUMBERS BEFORE THIS LARGE SQUARE RESF 7
    NUPL5Q=(2*LARSQR+1)*NDSPRL RESF 8
    WRITE(2,10)NUPL5Q, LARSQR RESF 9
10 FORMAT(1H1///35X,15HROWS UP TO THE END OF LARGE SQUARE ,13) RESF 10
    LNMLES=LAST NUMBER OF M IN THE VECTOR LESTRA WHEN THIS SUBROUTINE RESF 11
    WAS USED BEFORE PLUS ONE RESF 12
    NDOSBL=(LARSQR-1)*NDSPRL*2 RESF 13
    LNMP3N=LNMLES PLUS 3*NDSPRL RESF 14
    IF(LESTRA(LNMLES))450,500,40 RESF 15
   CONTINUE RESF 16
    LNMP3N=LNMLES+3*NDSPRL-1 RESF 17
   DO 400 M=LNMLES, LNMP3N RESF 18
   CONTINUE RESF 19
   RETURN RESF 20
END RESF 21
IF(LESTRA(M)-NUPLSQ)220,210,100

100 CONTINUE
WRITE(2,101)LESTRA(M),NUPLSQ
101 FORMAT(1H,10HLESTRA(M)=,I5,3X,12HGREATERTHAN,I5)
GO TO 500

210 CONTINUE
WRITE(2,211)LESTRA(M),NUPLSQ
211 FORMAT(1H,10HLESTRA(M)=,I5,3X,23HEQUALTOWHNPLSWHICH=,I5)
GO TO 230

C

220 CONTINUE
WRITE(2,221)LESTRA(M),NUPLSQ
221 FORMAT(1H,10HLESTRA(M)=,I5,3X,24HLESSTHANWPNPLSQWHICH=,I5)
IF(LESTRA(M))450,500,230

C

230 CONTINUE
I=I+1
ID=LESTRA(M)-NODSBL

C

CALL ZEROWI(TREATM,NJ,IC,NASSEM,NTREAT)

C

WRITE(2,231)LARSQR,NUPLSQ,M,LESTRA(M),NODSBL,IO,I
231 FORMAT(1H,10HLARSQR=N,IO,I)

C

300 CONTINUE

C

400 CONTINUE
GO TO 500

C

450 CONTINUE
WRITE(2,451)
451 FORMAT(1H,32HLESTRA(M)ISNEGATIVEANDEQUALS,/)GO TO 500

C

500 RETURN

C

END

*FORTRAN SOURCE

SUBROUTINE ZEODS(LARSQR,NDSPRL,NRESTA,LESTRA,TREATM,NJ,NASSEM, ZERS 1
INTREAT,LASMAT,LNLMEs)
DIMENSION LESTRA(LASMAT),TREATM(NASSEM,NTREAT)
ZERS 2

C

THIS SUBROUTINE IS TO ZERO RESTRAINTS WHERE PREVIOUSLY APPLIED BY ZERS 5

C

SUBROUTINE RESTRA ONLY IF THEY OCCUR IN THE FIRST TWO ROWS OF SMSQZERS 7

C

NUPLSQ=(2*LARSQR+1)*NDSPRL
ZERS 8
C

NODSBL=(LARSQR-1)*NDSPRL*2
ZERS 9
C

IF(LESTRA(LNLMES))450,500,40
ZERS 10
C

LNMP2N=LNLMES+2*NDSPRL-1
ZERS 11
DO 400 M = LNMLES, LNMP2N
C IF(LFSTRA(M) - NULPSQ) 220, 210, 100
C 100 CONTINUE
C GO TO 500
C 210 CONTINUE
C GO TO 230
C 220 CONTINUE
C IF(LFSTRA(M)) 450, 500, 230
C 230 CONTINUE
LNMLES = LNMLES + 1
I = I + 1
IO = LESTRA(M) - NOSBL
C TREATM(IO, IO) = 0.0
C WRITE(2, 231) LARSQR, NULPSQ, M, LESTRA(M), NOSBL, IO, I, TREATM(IO, IO)
C 231 FORMAT(1H, 10HLARSQR N = , I5/1X,
1 RHNULPSQ = , I5/1X,
2 3HM = , I5/1X,
3 IOHLFSTRA(M) = , I5/1X,
4 IHNOSBL = , I5/1X,
5 4HTIO = , I5/1X,
6 3HI = , I5/1X,
7 14HTREATM(IO, IO) = , 1PE8.1/
C 300 CONTINUE
C 400 CONTINUE
C GO TO 500
C 450 CONTINUE
C WRITE(2, 451)
C 451 FORMAT(1H, 32HLESTRA(M) IS NEGATIVE AND EQUALS, /
C 500 CONTINUE
C RETURN
END
*ENTER
C DATA
GO14DATA
1050
NOMENCLATURE

INPUT PARAMETERS

NPHSQR = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG RADIAL LENGTH
NPHSQA = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG AXIAL LENGTH

DERIVED PARAMETERS

NUMSQS = NUMBER OF SQUARES (TOTAL)
NUMTRI = NUMBER OF TRIANGLES
NTVRNO = NUMBER OF VERTEX NODES TO SPECIFY TRIANGLES
NRADWI = NUMBER OF DISTINCT RADII (INCLUDING I/R AND O/R) IN WIDTH
(N.I.SIDES OF PHYSICAL SQUARES)
MAXILN = NUMBER OF DISTINCT AXIAL LENGTHS INCLUDING BOTH ENDS
(N.I.SIDES OF PHYSICAL SQUARES)
NTVRNO = NUMBER OF DISTINCT VERTEX NODES
NPRW = NUMBER OF NODES PER RADIAL WIDTH
NPAUX = NUMBER OF NODES PER AXIAL LENGTH
NUMNOD = TOTAL NUMBER OF NODES
LASMAT = NUMBER OF ROWS OF LARGE ASSEMBLY MATRIX
NDSPRL = NUMBER OF DISPLACEMENTS PER RADIAL LENGTH
ASSEM = SIZE OF ASSEMBLY
NTREAT = WIDTH OF TREATMENT MATRIX

1001, 15, 0, 1
10 FORMAT(2I2)
PREAD(1,10)NPHSQR,NPHSQA
WRITE(2,20)NPHSQR,NPHSQA
20 FORMAT(1H1/2x,52HNUMBER OF ROWS OF PHYSICAL SQUARES PER RADIAL WIDTH=,DIAS)
212/
1TH=,12,2x,52HNUMBER OF ROWS OF PHYSICAL SQUARES PER AXIAL LENGTH=,DIAS
30 FORMAT(2I2)
NUMSQS=NPHSQR*NPHSQA
WRITE(2,21)NUMSQS
21 FORMAT(1H0,28HTOTAL NUMBER OF RECTANGLES =,I4)
NUMTRI=2*NUMSQS
WRITE(2,22)NUMTRI
22 FORMAT(1H0,20HNUMBER OF TRIANGLES =,I4)
```Fortran

      NTVRNO=3*NUMTRI
      WRITE(2,23)NTVRNO
  23 FORMAT(1H,53HNUMBER OF VERTEX NODES REQUIRED TO SPECIFY ALL TRIANGLES =,I5)

      NPAWSI=NPHSQR+1
      WRITE(2,24)NPAWSI
  24 FORMAT(1H,35HNUMBER OF DISTINCT RADII IN WIDTH =,I2)

      NAXILN=NPHSQA+1
      WRITE(2,25)NAXILN
  25 FORMAT(1H,33HNUMBER OF DISTINCT AXIAL LENGTHS =,I3)

      NOVRNO=NAXILN*NPAWSI
      WRITE(2,26)NOVRNO
  26 FORMAT(1H,32HNUMBER OF DISTINCT VERTEX NODES =,I4)

      NOPRIW=2*NPHSQR+1
      WRITE(2,27)NOPRIW
  27 FORMAT(1H,39HTOTAL NUMBER OF NODES PER RADIAL WIDTH =,I2)

      NOPALN=2*NPHSQA+1
      WRITE(2,28)NOPALN
  28 FORMAT(1H,39HTOTAL NUMBER OF NODES PER AXIAL LENGTH =,I4)

      NUMNON=NOVRNO*NOPRIW
      WRITE(2,29)NUMNON
  29 FORMAT(1H,37HNUMBER OF NODES =,I4)

      LASMAT=2*NUMNON
      WRITE(2,30)LASMAT
  30 FORMAT(1H,38HNUMBER OF ROWS OF LARGE ASSEMBLY MATRIX =,I5)

      NDSPRL=2*NOPRIW
      WRITE(2,31)NDSPRL
  31 FORMAT(1H,41HTOTAL NUMBER OF DISPLACEMENTS PER RADIAL LENGTH =,I2)

      NASSEM=3*NDSPRL
      WRITE(2,32)NASSEM
  32 FORMAT(1H,41HTOTAL NUMBER OF DISPLACEMENTS FOR EACH EXTRA ROW OF RADIAL RECTANGLES =,I3)

      NTREAT=5*NDSPRL
      WRITE(2,33)NTREAT
  33 FORMAT(1H,27HTOTAL SIZE OF THE ROW ASSEMBLY MATRIX =,I3)

      NTWOSQ=2*NDSPRL
      WRITE(2,34)NTWOSQ
  34 FORMAT(1H,77HTOTAL NUMBER OF EXTRA DISPLACEMENTS FOR EACH ADDITIONAL ROW (RADIAL) OF RECTANGLES =,I3)

      CALL LOWUPT(NDSPRL,NASSEM,NTREAT,LASMAT,NPHSQR,NPHSQA,NTWOSQ)

STOP
END

*FORTRAN

SUBROUTINE LOWUPT(NDSPRL,NASSEM,NTREAT,LASMAT,NPHSQR,NPHSQA,NTWOSQ)

THIS PROGRAM IS TO OBTAIN DISPLACEMENTS X FROM LOADS B.
```

BY INTERCHANGES, OPERATING WITH LOWER FACTORS AND UPPER PORTION OF
DECOMPOSED PARTITION BANDED MATRIX ACTING ON B

LIST OF SUBROUTINES FOR LOWER UPPER TREATMENT

ZEROVE  READVE  PRNVEC  PRNLOC  PNSMSQ  ADTRET  PUBTOY  LOWTRI  PUYTOB  REIUTA  UPTRI

DIMENSION B(LASMAT), Y(NTREAT), TREATM(NASSEM, NTREAT), LOC(NTWOSQ)

CHILTON B(4242), Y(210), LOC(84), TREATM(126, 210)

THIS PROGRAM IS TO OBTAIN X FROM B BY OPERATING WITH UPPER AND
LOWER TRIANGLES (OF BANDED MATRIX) ON B

CALL TIME(A)

IUPTAP=15
LOWTAP=16
TDISPL=17
NDISC=20

NOSPRL=2*(2*NPHSQR+1)
NAXIAL=2*NPHSQA+1
NTOT = NAXIAL*NOSPRL

WRITE(2,10) NAXIAL, NTOT
10 FORMAT(9X, 33HNUMBER OF ROWS OF SMALL SQUARES =, 13/
19X, 26HTOTAL LENGTH OF VECTOR B =, 14/)

WRITE(2, 20)
20 FORMAT(9X, 22HZERO VECTOR B = ZEROVE/)

CALL ZEROVE(B, NTOT, LASMAT)

WRITE(2, 30)
30 FORMAT(9X, 25HREAD VECTOR B = READVE(B)/)

CALL READVE(B, LASMAT)
55
X
I C
C (C
C
C
IR = 1
IE=NTOT
CALL  PRNVEC!LASMAT
INITIALISING CONSTANTS
NJS=2*NDSPRL
KJ=2
IYO=0
NJB=2
NI= 3*NDSPRL
GO TO 200
100 CONTINUE
NJS=NDSPRL-1
KJ=1
NJB=0
NI=NDSPRL
WRITE(2,120)
120 FORMAT(/9X,30HLAST LOWER TRIANGLE TREATMENT./)
200 CONTINUE
WRITE(2,240)
240 FORMAT(/9X,21H MULTIPLIES FROM TAPE/) CALL RELOTA(LHMTAP,LARSOR,LOC,NJS,KJ,TREAT,NDSPRL,NTWOSQ,NASSEM, INTREAT)
WRITE(2,260)
260 FORMAT(/9X,19H PUT B TO Y = PUBTOY/) CALL PUBTOY(B,Y,NDSPRL,ISTART,LARSOR,IYO,IB,IE,1END,NJB,LASMAT, INTREAT)
CALL PRNVEC(B,IB,IE,LASMAT)
WRITE(2,270)
270 FORMAT(/9X,45HLower triangle acting on y with inter-changes/) CALL LOWTRI(TREAT,Y,NI,NJS,LCC,NASSEM,NTREAT,NTWOSQ)
WRITE(2,280)
280 FORMAT(/9X,19H PUT Y TO B = PUYTOB/) CALL PUYTOB(Y,B,IB,IE,IYO,LASMAT,NTREAT)
CALL PRNVEC(B,IB,IE,LASMAT)
IF(LARSOR-NPHSQA)200,100,300
300 CONTINUE
IYO=0
LRSQIN=NPHSQA+1
ISM=1
NJL=0
N=NDSPL
K=1
WRITE(2,390)
390 FORMAT(1H1//35X,52H FROM 1UPTAP TAPE, FOR BACKWARD SUBSTITUTION BY U1PPTRI/)
400 CONTINUE
C CALL REFUTA(1UPTAP,LARSQR,TREATM,LRSQIN,ISM,NASSEM,NTREAT,NDSPL)
C WRITE(2,260)
C CALL PUBTOY(B,Y,NDSPL,ISTART,LARSQR,1YO,IB,IE,IEND,NJB,LASMAT,
INTREAT)
C CALL PRNVEC(R,IB,IE,LASMAT)
WRITE(2,420)
420 FORMAT(/9X,26H UPPER TR INGLE ACTING ON Y/)
C CALL UPPTRI(TREATM,Y,N,K,NASSEM,NTREAT)
C WRITE(2,290)
C CALL PUYTOR(Y,B,IB,IE,1YO,LASMAT,NTREAT)
C CALL PRNVEC(B,IB,IE,LASMAT)
ISM = 2
LRSQIN=LRSQIN-I
C IF(LARSQR-1)900,800,500
C 500 CONTINUE
C IF(LRSQIN-NPHSQA)700,600,900
C 600 CONTINUE
C NJS = 2
N=3*NDSPL
K=NDSPL+1
GO TO 400
C 700 CONTINUE
C NJS=4
N=5*NDSPL
K=3*NDSPL+1
GO TO 400
C 800 CONTINUE
C IE=NTOT
10=1
WRITE(2,910)
910 FORMAT(1H1//10X,10H X-VECTOR =/)
900 CONTINUE
C
999 FORMAT(1H1/52X,18HEND OF CALCULATION/)
CALL TIME(AA)
STIME=AA-A
WRITE(2,3600)STIME
3600 FORMAT(1H1/48X,12HTIME TAKEN =,I4,8HSECONDS./)
STOP
END

*FORTRAN
SUBROUTINE ZERVE(B,NTOT,LASMAT)
DIMENSION B(LASMAT)
 THIS SEGMENT IS TO INITIALLY ZERO THE VECTOR B
DO 1 I=1,NTOT
B(I)=0.0
1 CONTINUE
RETURN
END

*FORTRAN
SUBROUTINE READVE(B,LASMAT)
DIMENSION B(LASMAT)
KOUNT=0
90 CONTINUE
100 FORMAT(15,5X,F10.4)
READ (1,100)I,A
WRITE(2,110)I,A
110 FORMAT(10X,18HP0SITION OF LOAD =,I4,10X,6HLOAD =,F20.8)
IF(I)120,140,150
120 STOP
C
140 RETURN
150 CONTINUE
IF(B(I))160,170,160
160 CONTINUE
WRITE(2,165)B(I)
165 FORMAT(1H1/10X,32HPREVIOUS VALUE AT SAME POSITION =,F10.4/)
GO TO 120
170 CONTINUE
B(I)=A
KOUNT=KOUNT+1
WRITE(2,175)KOUNT
175 FORMAT(9X,16HNUMBER READ IN =,I4)
GO TO 90
END

*FORTRAN
SUBROUTINE PRNVEC(B,IB,IE,LASMAT)
DIMENSION B(LASMAT)
WRITE(2,208)(I,B(I),I=IB,IE)
208 FORMAT(1H1/11X,10(I3,1PE9.2)/))
RETURN
END

*FORTRAN
SUBROUTINE RELOTA(LOHTAP,LARSRG,LOC,NJS,KJ,TREATN,DSPRL,NTWOSQ, RLOT 1
DIMENSION LOG(NTWOSQ), SMSQ(NDSPRL, NDSPL), TREATM(NASSEM, NTREAT)

THIS SEGMENT IS TO READ LOC, THE LOCATIONS OF THE ROW INTER-CHANGES, RLOT, AND THE SMSQ CONTAINING THE MULTIPLIERS FROM LOWTAP TAPE AND TRANSFER THEM TO TREATM.

NOMENCLATURE

NJS = NUMBER OF ROW INTER-CHANGES (USUALLY = 2 * NDSPRL, BUT LAST = NDSPRL - 1)

KJ = NUMBER OF COLUMNS OF SMSQ (USUALLY = 2, EXCEPT LAST = 1)

READ(LOWTAP) LARSQR, LOC

CALL PRNLOC(LARSQR, LOC, NJS, NTWOSQ)

DO 20 J = 1, KJ

READ(LOWTAP) LARSQR, JSYSQ, NROM

WRITE(2, 140) LARSQR, JSYSQ, NROM

DO 10 I = 1, NROM

READ(LOWTAP) LARSQR, ISYSQ, JSYSQ, SMSQ

CALL PNSMSQ(LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ)

CALL ANTRET(LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ, TREATM, NASSEM, NTREAT)

10 CONTINUE

20 CONTINUE

RETURN

140 FORMAT(/, /, //, 29X, 15H LARSQR NUMBER =, I2, 10X, 14H COLUMN INDEX =, I3, 10X, RLOT)

1 16H NUMBER OF ROWS =, I2)

END

*FORTRAN

SUBROUTINE PRNLOC(LARSQR, LOC, KJS, NTWOSQ)

DIMENSION LOC(NTWOSQ)

THIS SEGMENT IS TO WRITE OUT THE LARSQR NUMBER AND NJS OF THE LOC NUMBERS

WRITE(2, 100) LARSQR

100 FORMAT(1H1, /, //, 52X, 15H LARSQR NUMBER =, I2, /)

WRITE(2, 120)(LOC(I), I = 1, NJS)

120 FORMAT(1X, 205, 10X)

RETURN

END

*FORTRAN

SUBROUTINE PUBTOY(Y, NDSPRL, ISTART, LARSQR, IY, IB, IE, IEND, NJB, PBTY)

PLTC 1

PLTC 2

PLTC 3

PLTC 4

PLTC 5

PLTC 6

PLTC 7

PLTC 8

PLTC 9

PLTC 10

PLTC 11

PLTC 12

PLTC 13

PLTC 14

PLTC 15

PNTY 1

PNTY 2

PNTY 3

PNTY 4

PNTY 5

PNTY 6

PNTY 7

PNTY 8

PNTY 9

PNTY 10

PNTY 11

PNTY 12

PNTY 13

PNTY 14

PNTY 15
DIMENSION B(LASMAT), Y(NTREAT)

THIS SEGMENT IS TO TRANSFER A PORTION OF B TO Y, USUALLY 3*NDSPRL.

EXCEPT LAST PORTION = NDSPRL

ISTART=2*LARSQR-1
IEND=ISTART+NJB
ID=(ISTART-1)*NDSPRL+1
IE=IEND*NDSPRL

IY=IYO

DO 10 I=ISTART,IE
[Y=IY+1]
Y(IY)=B(I)
10 CONTINUE
RETURN
END

*FORTRAN
SUBROUTINE LOWTRI(A,B,NJ,NLCC,NASSEM,NTREAT,NTWOSQ)
DIMENSION A(NASSEM,NTREAT), B(NASSEM), LOC(NTWOSQ)

FINDING Y SUCH THAT L*Y=B,

WHERE
L=LOWER TRIANGULAR MATRIX
B=RIGHT-HAND-SIDE VECTOR
Y=UNKNOWN INTERMEDIATE VECTOR
J=COLUMN INDEX

DO 20 J=1,NJ

INTERCHANGE TWO VALUES OF B

S=B(J)
K=LOC(J)
B(J)=B(K)
B(K)=S

JPI=J+1

DO 10 I=JPI,NJ
B(I)=B(I)-A(I,J)*S
10 CONTINUE

COMPLETED COLUMN J

20 CONTINUE

COMPLETED NJ COLUMNS

Y IS NOW SITUATED IN COLUMN B AS FAR AS NJS
SUBROUTINE PUYTOB(Y,B,IB,IE,IY,O,LASMAT,NTREAT)

DIMENSION B(LASMAT),Y(NTREAT)

THE PURPOSE OF THIS ROUTINE IS TO TRANSFER Y BACK ON TO B

IY=IY

DO 10 I=IB,IE
    IY=IY+1
    B(I)=Y(IY)
10 CONTINUE

RETURN

END

SUBROUTINE RIUTA(IUPTAP,LARSQR,TREATH,LRSQIN,ISM,NASSEM,NTREAT,NDSPRL)

DIMENSION TREATH(NASSEM,NTREAT),LRSQIN(NDSPRL,NDSPRL)

THIS SEGMENT IS TO READ THE UPPER-HALF OF THE BANDED MATRIX FROM

THE TAPE IUPTAP, READY FOR BACKWARD SUBSTITUTION. THIS IS DONE BY

READING OFF THE LAST ROW AND THERE-AFTER THE TWO PREVIOUS ROWS

OF SMALL SQUARES UNTIL THE FIRST TWO ROWS

CONTINUE

DO 60 I=1,ISM
60 CONTINUE

READ(IUPTAP)LARSQR,ISMSQ,JCLNO

TEST IF TAPE IS AT CORRECT LARSQR, IF BEFORE READ ON, IF AFTER REWIN

IF(LARSQR-LRSQIN)20,40,70

CONTINUE

BEFORE CORRECT LARSQR

CONTINUE

DO 30 J=1,JCLNO
30 CONTINUE

RETURN TO 10

GO TO 10

CONTINUE

AT CORRECT LARSQR
SUBROUTINE UPPT (A, B, N, K, NASSEM, NTREAT)

DIMENSION A(NASSEM, NTREAT), B(NTREAT)

FINISHING THAT UX=Y, BY BACKWARD SUBSTITUTION

WHERE
U=UPPER TRIANGULAR MATRIX
Y=RIGHT-HAND-SIDE VECTOR
X=UNKNOWN VECTOR
I=ROW INDEX

DO 40 L=K, N
    I=N+1-L
    IP1=I+1
    IF(IP1-N)10,10,30

    DO 20 J=IP1, N
        B(I)=B(I)-A(I,J)*B(J)
    CONTINUE

    B(I)=B(I)/A(I,I)

    X(I) NOW OCCUPIES THE PLACE OF Y(I)

CONTINUE

UPTR 1
UPTR 2
UPTR 3
UPTR 4
UPTR 5
UPTR 6
UPTR 7
UPTR 8
UPTR 9
UPTR 10
UPTR 11
UPTR 12
UPTR 13
UPTR 14
UPTR 15
UPTR 16
UPTR 17
UPTR 18
UPTR 19
UPTR 20
UPTR 21
UPTR 22
UPTR 23
UPTR 24
UPTR 25
UPTR 26
UPTR 27
UPTR 28
UPTR 29
UPTR 30
UPTR 31
UPTR 32
UPTR 33
**FORTRAN**

SUBROUTINE PNSMSQ(LARSQR,NDSPRL,ISMSQ,JSMSQ,SMSQ)
  DIMENSION SMSQ(NDSPRL,NDSPRL)
  ROUTINE TO PRINT OUT SMSQ AND ITS INDICES
  WRITE(2,4)LARSQR,ISMSQ,JSMSQ,NDSPRL
  4 FORMAT(1H1,16X,?0HLARGE SQUARE INDEX =,I2,7X,23HSMALL SQUARE ROW INDEX =,I2,16X,7X,27HSMALL SQUARE COLUMN INDEX =,I2,23X,36X,47HNUMBER OF DISTINCT SMALL SQUARES IN EACH ROW =,I2)
  RETURN
  END

SUBROUTINE ADTRET(LARSQR,NDSPRL,ISMSQ,JSMSQ,TREATM,NASSEM,NRTREAT)
  DIMENSION SMSQ(NDSPRL,NDSPRL),TREATM(NASSEM,NRTREAT)
  CHILTON 42*42 126,210
  NOMENCLATURE
  IP = ROW INDEX OF PARTITION OF TREATMENT
  JP = COLUMN INDEX OF PARTITION OF TREATMENT
  LARSQR = INDEX OF LARGEST SQUARE (SAME AS AXIAL INDEX)
  TREATM = TREATMENT MATRIX
  IP = ISMSQ-2*(LARSQR-1)
  JP = JSMSQ-2*(LARSQR-1)
  IB = (IP-1)*NDSPRL+1
  IE = IP*NDSPRL
  JB = (JP-1)*NDSPRL+1
  JE = JP*NDSPRL
  I = 0
  DO 10 IA=IB,IE
    I = I+1
    J = 0
    DO 20 JA=JB,JE
      J = J+1
      TREATM(IA,JA)=SMSQ(I,J)
    20 CONTINUE
  10 CONTINUE
  RETURN
  END

**FORTRAN**

SUBROUTINE PTREAT(LARSQR,ISMSQ,JSMSQ,TREATM,NASSEM,NRTREAT,IB,IE,JPTREAT)
  DIMENSION TREATM(NASSEM,NRTREAT)
  IB = I ROW BEGINNING
  IE = I ROW ENDING

END
C  JB = J COLUMN BEGINING
C  JE = J COLUMN ENDING
C
205  FORMAT(1H1//9H LARSQR =,12,8H I SMSG =,12,8H JSMSG =,12,
1 17H I ROW BEGINING =,13,15H I ROW ENDING =,13,
2 20H J COLUMN BEGINING =,13,18H J COLUMN ENDING =,13//)
WRITE(2,205) LARSQR, I SMSG, JSMSG, IB, IF, JB, JE

206  FORMAT(6X,14I8)
WRITE(2,206) (J, J=JB, JE)

208  FORMAT(2X,12,2X,1P14F8.1/)
WRITE(2,208) I, (TREATM (I, J), J=JB, JE)

301  CONTINUE
RETURN
FND

*ENTER

DATA
GO15DATA
1050

1     422500.0
43    845000.0
85    845000.0
127   845000.0
169   845000.0
211   845000.0
253   845000.0
295   845000.0
337   845000.0
379   845000.0
421   845000.0
463   845000.0
505   845000.0
547   845000.0
589   845000.0
631   845000.0
673   845000.0
715   845000.0
757   845000.0
799   845000.0
841   845000.0
883   845000.0
925   845000.0
967   845000.0
1009  845000.0
1051  845000.0
1093  845000.0
1135  845000.0
1177  845000.0
1219  845000.0
1261  845000.0
1303  845000.0
1345  845000.0
1387  845000.0
1429  845000.0
1471  845000.0
1513  845000.0
1555  845000.0
1597  845000.0
1639  845000.0
1681  845000.0
1723  845000.0
1765  845000.0
1807  845000.0
1849  845000.0
1891  845000.0
1933  845000.0
1975  845000.0
2017  845000.0
2059  845000.0
2101  845000.0
2143  845000.0
2185  845000.0
2227  845000.0
2269  845000.0
2311  845000.0
2353  845000.0
2395  845000.0
2437  845000.0
2479  845000.0
2521  845000.0
2563  845000.0
2605  845000.0
2647  845000.0
2689  845000.0
2731  845000.0
2773  845000.0
2815  845000.0
2857  845000.0
2899  845000.0
3001  845000.0

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1807</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>1840</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>1891</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>1933</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>1975</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2017</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2059</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2101</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2143</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2185</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2227</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2269</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2311</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2353</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2395</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2437</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2470</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2521</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2563</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2605</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2647</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2689</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2731</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2773</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2815</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2857</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2899</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2941</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>2983</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3025</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3067</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3109</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3151</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3193</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3235</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3277</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3319</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3361</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3403</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3445</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3487</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3529</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3571</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3613</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3655</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3697</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3739</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3781</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3823</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3865</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3907</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3949</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>3991</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>4033</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>4075</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>4117</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>4159</td>
<td>845000.0</td>
<td></td>
</tr>
<tr>
<td>4201</td>
<td>422500.0</td>
<td></td>
</tr>
<tr>
<td>4243</td>
<td>845000.0</td>
<td></td>
</tr>
</tbody>
</table>
DISPLACEMENT CHECK

INPUT PARAMETERS

NPHSQR=NUMBER OF ROWS OF PHYSICAL SQUARES ALONG RADIAL LENGTH
NPHSQA=NUMBER OF ROWS OF PHYSICAL SQUARES ALONG AXIAL LENGTH

DERIVED PARAMETERS

NUMSQS=NUMBER OF SQUARES (TOTAL)
NUMTRI=NUMBER OF TRIANGLES
NTVRND=NUMBER OF VERTEX NODES TO SPECIFY TRIANGLES
NRADWI=NUMBER OF DISTINCT RADII (INCLUDING I/R AND O/R) IN WIDTH
(I.E. SIDES OF PHYSICAL SQUARES)
MAXILN=NUMBER OF DISTINCT AXIAL LENGTHS INCLUDING BOTH ENDS
(I.E. SIDES OF PHYSICAL SQUARES)
NTVRND=NUMBER OF DISTINCT VERTEX NODES
NODRHI=TOTAL NODES PER RADIAL WIDTH
NODALN=TOTAL NODES PER AXIAL LENGTH
NUMNOD=TOTAL NUMBER OF NODES
LASMAT=NUMBER OF ROWS OF LARGE ASSEMBLY MATRIX
NDSPL=NUMBER OR DISPLACEMENTS PER RADIAL LENGTH
ASEM=ASSEMBLY (PARTIAL) OF ELEMENTS IN ONE ROW (PHYSICAL)
NASSEM=SIZE OF ASSEMBLY
NTREAT=WIDTH OF TREATMENT MATRIX

10 FORMAT(2I2)
READ (1,10)NPHSQR,NPHSQA
WRITE(2,20)NPHSQR,NPHSQA
20 FORMAT(1H1/2X,5ZI),NUMSQS=NPHSQR*NPHSQA
WRITE(2,21)NUMSQS
21 FORMAT(1H0,2H),TOTAL NUMBER OF RECTANGLES =,14)
NUMTRI=2*NUMSQS
WRITE(2,22)NUMTRI
22 FORMAT(1H0,2H),NUMBER OF TRIANGLES =,14)
NTVRND=2*NUMTRI
WRITE(2,23)NTVRND
23 FORMAT(1H0,5H),NUMBER OF VERTEX NODES REQUIRED TO SPECIFY ALL TRIANGLES
1GLES =,15)
NRAODI=NPHSQR+1

NOMENCLATURE
WRITE(2,24)NPRADWI
C 24 FORMAT(1H,35HNUMBER OF DISTINCT RADI IN WIDTH =,12)
C MAXILN=NPHSQA+1
C WRITE(2,25)MAXILN
C 25 FORMAT(1H,33HNUMBER OF DISTINCT AXIAL LENGTHS =,13)
C NDRVNO=MAXILN*NPRADWI
C WRITE(2,26)NDRVNO
C 26 FORMAT(1H,33HNUMBER OF DISTINCT VERTEX NODES =,14)
C NOPRWI=2*NPHSQR+1
C WRITE(2,27)NOPRWI
C 27 FORMAT(1H,40HTOTAL NUMBER OF NODES PER RADIAL WIDTH =,12)
C NOPALN=2*NPHSQA+1
C WRITE(2,28)NOPALN
C 28 FORMAT(1H,40HTOTAL NUMBER OF NODES PER AXIAL LENGTH =,14)
C NUMMOD=NTPRWI*NOPALN
C WRITE(2,29)NUMMOD
C 29 FORMAT(1H,23HWHOLE NUMBER OF NODES =,15)
C LASMAT=2*NUMMOD
C WRITE(2,30)LASMAT
C 30 FORMAT(1H,41HNUMBER OF ROWS OF LARGE ASSEMBLY MATRIX =,15)
C NDSPRL=2*NPRW
C WRITE(2,31)NDSPRL
C 31 FORMAT(1H,43HNUMBER OF DISPLACEMENTS PER RADIAL LENGTH =,12)
C NASSEM=3*NDSPRL
C WRITE(2,32)NASSEM
C 32 FORMAT(1H,33HSIZE OF THE ROW ASSEMBLY MATRIX =,13)
C NTREAT=5*NDSPRL
C WRITE(2,33)NTREAT
C 33 FORMAT(1H,27WIDTH OF TREATMENT MATRIX =,13)
C CNTWOSQ=NUMBER OF EXTRA DISPL. FOR EACH EXTRA ROW OF RADIAL RECTANG
C CNTWOSQ=2*NDSPRL
C WRITE(2,34)CNTWOSQ
C 34 FORMAT(1H,77HNUMBER OF EXTRA DISPLACEMENTS FOR EACH ADDITIONAL ROW
C (RADIAL) OF RECTANGLES =,13)
C C CALL CEQDIS(NDSPRL,NASSEM,NTREAT,LASMAT,NPHSQR,NPHSQA,CNTWOSQ)
C CEDS 96
C STOP
C FNO
C "FORTRAN"
C C SUBROUTINE CEQDIS(NDSPRL,NASSEM,NTREAT,LASMAT,NPHSQR,NPHSQA,CNTWOSQ)
C I)
C 10TH APRIL 1970
C DIMENSION TREATM(NMSEM,NTREAT),B(LASMAT),X(NTREAT),
C 1SMSQ(NDSPRL,NDSPRL),Y(NASSEM),C(LASMAT)
C C THIS PROGRAM IS TO MULTIPLY THE STIFFNESS MATRIX TIMES THE
C C DISPLACEMENTS TO OBTAIN THE NCDIAL LOADS
C C
CALL TIM{INSTALL)
IASSE=14
NOISC=20
IDISPL=17

READ(1DISP1)NPSORL,NPSQAL,NPRLDS,NAXIAL,NTOT,8
WRTF(12,100)NPSORL,NPSQAL,NPRLDS,NAXIAL,NTOT

LIST OF SUBROUTINES FOR DISPLACEMENT CHECK

TAPDIS
  PNSMSQ
  ZEAMSQ

FIN3SQ
  WRBSQ
  ZETRET
  (PNSMSQ)
  ADTRET

MEXDOZ
  (PNSMSQ)
  (ADTRET)

PUBTOY

RINVEC

MULTAX

PUYTOB

TRANSF

100 FORMAT(1H1/10X,
1 5HNUMBER OF ROWS OF PHYSICAL SQUARES PER RADIAL WIDTH=',12/10X,
2 5HNUMBER OF ROWS OF PHYSICAL SQUARES PER AXIAL LENGTH=',12/10X,
3 4HNUMBER OF DISPLACEMENTS PER RADIAL LINE=',13/10X,
4 48NUMBER OF RADIAL LINES OF NODES ALONG THE AXIS=',12/10X,
5 23HTOTAL NUMBER OF NODES=',14/)

CALL TAPDIS(SIMSG,NPHSQR,NPHSQA,NDSPPL,NOISC,IASSEM)

NDT1M2=2*NDSPRL
NDT2P1=NDT1M2+1
NDT1M3=3*NDSPRL
NDT3M5=5*NDSPRL
IS=1
NI=NDT1M3

CALL FTA3SQ(MDSRL,TREATM,NCOLN,LARSQR,ISMSQ,NOISC,NASSEM,NTREAT)

LARSQR=1
NJB=4
IY0=0
NJ=NDT3M5
GO TO 2

1 CONTINUE

MJ=NDT3M3
SUBROUTINE TRANSFER DATA FROM TAPE TO DISC

NOMENCLATURE

NDSRFL = NUMBER OF DISPLACEMENTS PER RADIAL LINE OF PHYSICAL SQUARE

NPHSQA = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG THE AXIAL LENGTH

NPRSQ2 = NPHPSQA - 2, USED FOR TAILING OFF THE BANDED MATRIX

NPHSQR = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG THE RADIAL LENGTH

ISMSQ = ROW INDEX OF SMALL SQUARE

JMSQ = BEGINNING COLUMN INDEX OF SMALL SQUARE

IROW = ROW INDEX = 1, 2, OR 3

IROWE = END ROW INDEX


```c
C
JB = COLUMN INDEX WITHIN SMALL SQUARE BEGINING (SPACIAL)     TADI 21
JE = COLUMN INDEX WITHIN SMALL SQUARE ENDING (SPACIAL)       TADI 22
IB = ROW INDEX WITHIN SMALL SQUARE BEGINING (SPACIAL)        TADI 23
IE = ROW INDEX WITHIN SMALL SQUARE ENDING (SPACIAL)          TADI 24
IR = INDEXING ROW
NCOLN = NUMBER OF COLUMNS OF SMALL SQUARES
NDSPRL = NUMBER OF DISPLACEMENTS PER RADIAL LINE OF PHYSICAL SQUARE
JSMSQ = COLUMN INDEX OF SMALL SQUARE (GLOBAL)
JCOLN = COLUMN INDEX 1 TO NCOLN
SMSQ = SMALL SQUARE = PARTICNED MATRIX
NROWSQ = NUMBER OF ROWS IN MATRIX
ISMSQ= MAXIMUM NUMBER OF ROWS OF SMALL SQUARES IN WHOLE MATRIX

REWIND IASSEM
READ(IASSEM)KOUNT
NCOUNT=0

SETTING INITIAL VALUES
----------------------------------------
THROW=1.0
NROWSQ=2*NPHSQR+1
NDSPRL=2*NROWSQ
ISMSQ=2*NPHSQA+1
NPRSM2=ISMSQM-2
ISMSQ=1
JSMSQB=1
IROW=1
IROWE=3
JB=1

BEGINING OF MAIN LOOP SETTING UP DATA ON DISK(EXCEPT FOR LAST PART)
----------------------------------------
1 CONTINUE
IE=ISMSQ*NDSPRL
IB=IE-NDSPRL+1
IR=IB

2 CONTINUE
IF(IROWE-IROW)3,4,5
3 CONTINUE
CALL TRACE
STOP
4 CONTINUE
NCOLN=7
GO TO 6
5 CONTINUE
NCOLN=5
GO TO 6

16 CONTINUE
JE=JB+NCOLN*NDSPRL-1
JSMSQ=JSMSQB
WRITE(NDISC)ISMSQ,NCOLN,JB,JE
DO 16 JCOLN=1,NCOLN
```

This section (to 16) is to read from tape for the first 3 or 5 small squares and put on to disc with two zero small squares.

6 CONTINUE
```
IF(JCOLN-NCOLN+2)7,9,9
7 CONTINUE
C
JCOLN LESS THAN NCOLN
8 CONTINUE
C
JCOLN=NCOLN
C
NCOUNT=NCOUNT+1
READ(IASSET)ISMSQT,JSMSQT,SMSC,IAXIAL
CALL PNSMSQ(IAXIAL,NDSPRL,ISMSQT,JSMSQT,SMSC)
C
IF(ISMSQT-ISMSQ)10,11,12
9 CONTINUE
CALL ZESMSQ(NDSPRL,ISMSQ)
GO TO 14
10 CONTINUE
WRITE(2,41)ISMSQT,ISMSQ
CALL TRACE
STOP
11 CONTINUE
IF(JSMSQT-JMSQ)13,14,15
12 CONTINUE
WRITE(2,42)ISMSQT,JSMSQ
CALL TRACE
STOP
13 CONTINUE
CALL TRACE
WRITE(2,43)JSMSQT,JSMSQ
STOP
14 CONTINUE
WRITE(NDISC)ISMSQ,JSMSQ,SMSC
JSMSQ=JSMSQ+1
GO TO 16
15 CONTINUE
WRITE(2,44)JSMSQT,JSMSQ
CALL TRACE
STOP
16 CONTINUE
IROW=IROW+1
C TEST TO SEE IF END ROW
C
IF(IROW-IROWE)17,10,19
17 CONTINUE
C BEFORE END OF ROW
C
18 CONTINUE
C END OF ROW
C
ISMSQ=ISMSQ+1
IF(ISMSQ-NPRSM2)21,22,23
19 CONTINUE
JSMSQB=JSMSQB+2
ISMSQ=ISMSQ+1
IROWE=2
IROW=1
JB=JB+2*NDSPRL
GO TO 1
21 CONTINUE
GO TO 1
22 CONTINUE
C C ISMSQ=NPRSM2
C 23 CONTINUE
C C ISMSQ.GT.NPRSM2
C 24 CONTINUE
C C FOR LAST THREE ROWS OF SQUARES
C IROW=1
IROWE=3
25 CONTINUE
IE=ISMSQ*NDSPRL
IB=IE-NDSPRL+1
IR=IB
C 26 CONTINUE
IF(IROW-2)27,28,29
27 CONTINUE
NCOLN=5
GO TO 30
28 CONTINUE
NCOLN=3
JSMSQ=JSMSQ+2
JB=JB+NCOLN*NDSPRL
GO TO 30
29 NCOLN=3
GO TO 30
C C INDICES FOR BEGINING OF ROW
C 30 CONTINUE
JE=JB+NCOLN*NDSPRL-1
ISMSQ=JSMSQ
WRITE(NDISC)ISMSQ,NCOLN,JB,JE
DO 37 JCOLN=1,NCOLN
NCOUNT=NCOUNT+1
READ(LASSEM)JSMSQT,JSMSQT,SMSC,IAXIAL
CALL PNSMSQ(IAXIAL,NDSPRL,ISMSQT,JSMSQT,JSMSQ)
IF(ISMSQT-ISMSQ)31,32,33
31 CONTINUE
WRITE(2,41)ISMSQT,ISMSQ
CALL TRACE
STOP
32 CONTINUE
IF(JSMSQT-ISMSQ)34,35,36
33 CONTINUE
WRITE(2,42)ISMSQT,ISMSQ
CALL TRACE
STOP
34 CONTINUE
WRITE(2,43)JSMSQT,JSMSQ
CALL TRACE
STOP
35 CONTINUE
WRITE(NDISC)ISMSQ,JSMSQ,SMSC
JSMSQ = JSMSQ + 1
GO TO 37
36 CONTINUE
WRITE(2,44)JSMSQT,JSMSQ
CALL TRACE
STOP
37 CONTINUE
IROW = IROW + 1
UP-DATING ROW INDEX
38 CONTINUE
ISMSQ = ISMSQ + 1
GO TO 25
39 CONTINUE
GO TO 38
40 CONTINUE
END FILE NDISC
WRITE(2,45)K0,UN,UNT,UNT
RETURN
61 FOR 'AT IH l,38X,37H SMSQ FROM TAPE), I4,20H LESS THAN
38 FORMAT I1H,15X,17H SMSQ FROM TAPE), I4,20H MORE THAN
42 FORMAT I1H,15X,17H SMSQ FROM TAPE), I4,20H MORE THAN
43 FORMAT I1H,15X,26H NUMBER OF WRITES ON TAPE =,13/
44 FORMAT I1H,I1OX,26H NUMBER OF READS OFF TAPE =,13/
END

*FORTRAN
SUBROUTINE PNSMSQ(LARSQR,NDSPRL,ISMSQ,JSMSQ,SMSQ)
DIMENSION SMSQ(NDSPRL,NDSPRL)
FORMAT (1H,15X,20H LARGESQ INDEX =,12,7X,24H SMALL SQ ROW INDEX =,12,7X,27H SMALL SQ COLUMN INDEX =,12)
IIF(LARSQR)205,302,303
205 CONTINUE
WRITE(2,206)I, J=1,NDSPRL)
206 FORMAT (1X,14(5X,13//))
DO 301 I=1,NDSPRL
WRITE(2,207)I
207 FORMAT (1X,13)
WRITE(2,208)SMSQ(I,J), J=1,NDSPRL
208 FORMAT (5X,-6P14F8.0/)}
301 CONTINUE
302 CONTINUE
303 CONTINUE
RETURN
END

*FORTRAN
SUBROUTINE ZFSMSQ(NDSPRL,SMSQ)
DIMENSION SMSQ(NDSPRL,NDSPRL)
DO 1 I=1,NDSPRL
DO 1 J=1,NDSPRL
SMSQ(I,J)=0.0
1 CONTINUE
CONTINUE
RETURN
END

SUBROUTINE FIRE3SQ(NDSPRL, TREATM, NCOLN, LARSQR, ISMSQ, NDISC, NASEM, NINTREAT)
DIMENSION TREATM(NASEM, NINTREAT), SMSQ(NDSPRL, NDSPRL)

REWIND 20
READ(NDISC) ISMSQ, NCOLN, JB, JE

CALL WRTBSQ(LARSQR, ISMSQ, NCOLN, JB, JE)
LARSQR=1
CALL ZETRET(NDSPRL, TREATM, NASEM, NINTREAT)
DO 5 JCOL=1,3
READ(NDISC) ISMSQ, JSMSQ, SMSQ
CALL PNSMSQ(LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ)
CALL ADTRET(LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ, TREATM, NASEM, NINTREAT)
CONTINUE
LARSQR=0
RETURN
END

SUBROUTINE WRTBSQ(LARSQR, ISMSQ, NCOLN, JB, JE)
C
THIS SUBROUTINE IS TO WRITE OUT THE LARGE SQUARE NUMBER
C (CORRESPONDS TO AXIAL NUMBER), SMALL SQUARE ROW NUMBER, THE NUMBER
C OF COLUMNS OF SMALL SQUARES AND THE BEGINING AND END COLUMN NUMBER.
C S OF THE COMPLETE ASSEMBLY MATRIX
WRITE(2,2)LARSQR, ISMSQ, NCOLN, JB, JE
2 FORMAT(1H, 12HLARGE SQUARE, 13, 25H, SMALL SQUARE ROW INDEX =, I3,
1 32H, NUMBER OF SMALL SQUARES IN ROW =, I2,
2 29H, MATRIX ELEMENT COLUMNS BEGIN, I4, 4H, END, I4)
RETURN
END

SUBROUTINE ZETRET(NDSPRL, TREATM, NASEM, NINTREAT)
DIMENSION TREATM(NASEM, NINTREAT)

ITREAT = 3*NDSPRL
JTREAT = 5*NDSPRL
DO 10 I=1, ITREAT
DO 20 J=1, JTREAT
TREATM(I, J) = 0.0
CONTINUE
20 CONTINUE
RETURN
END

SUBROUTINE ADTRET(LARSQR, NDSPRL, ISMSQ, JSMSQ, SMSQ, TREATM, NASEM, NINTREAT)
ROUTINE TO ADD SMSQ TO APPROPRIATE PART OF TREATM

DIMENSION SMSQ(NDSPRL,NDSPRL),TREATM(NASSEM,NTREAT)

CHILTON 42*42 126,210

NOMENCLATURE

IP = ROW INDEX OF PARTITION OF TREATM
JP = COLUMN INDEX OF PARTITION OF TREATM
LARSQR = INDEX OF LARGE SQUARE (SAME AS AXIAL INDEX)
TREATM = TREATMENT MATRIX
IP = ISMSQ-2*(LARSQR-1)
JP = JSMSQ-2*(LARSQR-1)
IB = (IP-1)*NDSPRL+1
IE = IP *NDSPRL
JB = (JP-1)*NDSPRL+1
JE = JP *NDSPRL
I = 0
DO 10 IA=IB,IE
I = I+1
J = 0
DO 20 JA=JB,JE
J = J+1
TREATM(IA,JA)=SMSQ(I, J)
20 CONTINUE
10 CONTINUE
RETURN
END

*FORTRAN

SUBROUTINE NXDZ(NDSPRL,TREATM,NCOLN,LARSQR,JSMSQ,NPHSQA,NDISC,NASSEM,NTREAT)

DIMENSION TREATM(NASSEM,NTREAT),SMSQ(NDSPRL,NDSPRL)

ISMSQM=2*NPHSQA+1
NPRSM2=ISMSQM-2
IF(ISMSQ-NPRSM2)10,21,21
10 CONTINUE
DO 20 JCOL=1,2
READ(NDISC)ISMSQ,JSMSQ,SMSQ
CALL PNSMSQ(LARSQR,NDSPRL,ISMSQ,JSMSQ,SMSQ)
CALL ADTREI(LARSQR,NDSPRL,ISMSQ,JSMSQ,SMSQ,TREATM,NASSEM,NTREAT)
20 CONTINUE
21 CONTINUE
INDEX = 0
30 CONTINUE
INDEX=INDEX+1
INOSPL=INDEX*NDSPRL
READ(NDISC)ISMSQ,NCOLN,JB,JE
CALL WRBSQ(LARSQR,ISMSQ,NCOLN,JB,JE)

IF(5-NCOLN)50,60,70
50 CONTINUE
60 CONTINUE
NSTOP=5
GO TO 80
70 CONTINUE
NSTOP=3
GO TO 90
80 CONTINUE
   DO 90 JCNL=1,NSTDP
      READ(INDISC) ISMS0,JSMSO,SMSQ
      CALL PNSMSQ(ILARSQR,NDSPRL,ISMS0,JSMSO,SMSQ)
      CALL ADTRET(LARSQR,NDSPRL,ISMS0,JSMSO,SMSQ,TREATM,NASSEM,NTREAT)
90 CONTINUE
   IF(INDEX-1)110,110,120
100 CONTINUE
   WRITE(2,99)INDEX
   99 FORMAT(1H1,10X,17HTNDEX INDEX NXTDOZ =,I2/)
   STOP
110 CONTINUE
   GO TO 30
120 CONTINUE
   RETURN
END

*FORTRAN
SUBROUTINE TRANS(TREATM,NDTIM2,NDT2P1,NDTIM3,NDTIM5,NASSEM,NTREAT)
   DIMENSION TREATM(NASSEM,NTREAT)
   THIS SEGMENT IS TO BE USED AFTER DATA HAS BEEN TRANSFERRED TO TAPE.
   IT IS TO TRANSFER THE LAST THREE SMSQ OF TREATM TO THE 1ST THREE.

   NOMENCLATURE

   NDTIM2 = NDSPRL + NDSPRL
   NDT2P1 = NDTIM2 + 1
   NDTIM3 = NDTIM2 + NDSPRL
   NDTIM5 = NDTIM2 + NDTIM3

   PUTTING LAST 3 TREATM INTO FIRST 3 TREATM

   DO 20 I=NDTIM2,NDT2P1,NDTIM3
      ISA=I-NDTIM2
      DO 10 J=NDT2P1,NDTIM5
         JSA=J-NDTIM2
         TREATM(ISA,JSA)=TREATM(I,J)
10      CONTINUE
20      CONTINUE
   RETURN
END

*FORTRAN
SUBROUTINE PRNVEC(B,IB,IE,LASRAT)
   DIMENSION B(LASMAT)
   WRITE(2,209)(I,B(I),I=IB,IE)
SUBROUTINE PUBTOY(B,Y,NDSPRL,ISTART,LARSQR,IYO,IB,IE,IEND,NJB,
ILLASMAT,NASSFM)
DIMENSION B(ILASMAT),Y(NASSFM)

ISTART=2*LARSQR-1
IEND=ISTART+NJB
IB=(ISTART-1)*NDSPRL+1
IE=IEND*NDSPRL

DO 10 I=IB,IE
   TY=IY+1
   Y(IY)=B(I)
10 CONTINUE

RETURN

SUBROUTINE YTOB(Y,B,IB,IE,IYO,IS,NASSFM,LASMAT)
DIMENSION B(LASMAT),Y(NASSFM)

TY=IY0+JS-1
DO 10 I=IB,IE
   TY=TY+1
   B(I)=Y(TY)
10 CONTINUE

RETURN

SUBROUTINE MULTAX(A,X,IS,NI,NJ,Y,NASSFM,NTREAT)
DIMENSION A(NASSFM,NTREAT),X(NTREAT),Y(NASSFM)

WRITE(2,201)IS,NI,NJ,NASSFM,NTREAT
201 FORMAT(H1,5X,9HSTART =13,9X,11HNUMBER OF I,13,5X,3HNJ=13,10X,
18HNASSEM =13,5X,8HLASMAT =13)

DO 20 I=IS,NI

RETURN
Y(1) = 0.0
DO 10 J = 1, NJ
  Y(I) = Y(I) + A(I,J) * X(J)
10 CONTINUE
20 CONTINUE
30 FORMAT(1H, 1OHVECTOR Y = [5X, 1P14E8.1])
C  WRITE(2,30)(Y(I), I=1S, NI)
RETURN
END

ENTER

DATA
G016DATA
1050
COMPUTING 30 MINUTES
EXECUTION 40 MINUTES
STORE 90 BLOCKS
TAPE 10 N0121 VLA47 NODE
TAPE 17 N060 TREATMENT
INPUT 1 GO3RDATA
OUTPUT 0 LINEPRINTER 3000 LINES
OUTPUT 2 LINEPRINTER 8000 LINES
COMPILER FORTRAN

*RUN SOURCE

FORTRAN

MASTER STRESS STRAIN

NOMENCLATURE

INPUT PARAMETERS

NPHSQR = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG RADIAL LENGTH
NPHSQA = NUMBER OF ROWS OF PHYSICAL SQUARES ALONG AXIAL LENGTH

DERIVED PARAMETERS

NUMSQS = NUMBER OF SQUARES (TOTAL)
NUMTRI = NUMBER OF TRIANGLES
NTVRN0 = NUMBER OF VERTEX NODES TO SPECIFY TRIANGLES
NRADWI = NUMBER OF DISTINCT RADII (INCLUDING R/R AND D/R) IN WIDTH
NAXILN = NUMBER OF DISTINCT AXIAL LENGTHS, INCLUDING BOTH ENDS
NADWRI = NUMBER OF DISTINCT VERTEX NODES
NORWI = TOTAL NODES PER RADIAL WIDTH
NORALN = TOTAL NODES PER AXIAL LENGTH
NUMNO = TOTAL NUMBER OF NODES
LASMAT = NUMBER OF ROWS OF LARGE ASSEMBLY MATRIX
NDSPRL = NUMBER OF DISPLACEMENTS PER RADIAL LENGTH
ASEMBY = ASSEMBLY (PARTIAL) OF ELEMENTS IN ONE ROW (PHYSICAL)
NASHE = SIZE OF ASSEMBLY

10 FORMAT(2,12)
READ (1,10) NPHSQR , NPHSQA
WRITE (2,20) NPHSQR , NPHSQA

20 FORMAT(1I1/2X,52HNUMBER OF ROWS OF PHYSICAL SQUARES PER RADIAL WIDTH
1TH=,12,2X,52HNUMBER OF ROWS OF PHYSICAL SQUARES PER AXIAL LENGTH=,212/)

21 FORMAT(1H0,21H TOTAL NUMBER OF RECTANGLES =,14/

22 FORMAT(1H ,20HNUMBER OF TRIANGLES =,14)

23 FORMAT(1H ,58HNUMBER OF VERTEX NODES REQUIRED TO SPECIFY ALL TRIANGLES=

1GLES =,15)

NPADWI = NPHSQR + 1
WRITE (2,24) NPADWI
24  FORMAT(IH,35HNUMBER OF DISTINCT RADII IN WIDTH =,I2)
   C
   C NAXILN=NPHSQA+1
   C WRITE(2,25)NAXILN
   C
25  FORMAT(IH,33HNUMBER OF DISTINCT AXIAL LENGTHS =,I3)
   C
   C NDVRNO=NAXILN*MRADWI
   C WRITE(2,26)NDVRNO
   C
26  FORMAT(IH,33HNUMBER OF DISTINCT VERTEX NODES =,I4)
   C
   C NOPRLN=2*NPHSQA+1
   C WRITE(2,27)NOPRLN
   C
27  FORMAT(IH,40HTOTAL NUMBER OF NODES PER RADIAL WIDTH =,I2)
   C
   C NOPALN=2*NPHSQA+1
   C WRITE(2,28)NOPALN
   C
28  FORMAT(IH,40HTOTAL NUMBER OF NODES PER AXIAL LENGTH =,I2)
   C
   C NUMNOD=NOPRLN*NOPALN
   C WRITE(2,29)NUMNOD
   C
29  FORMAT(IH,23HWHOLE NUMBER OF NODES =,I5)
   C
   C LASMAT=2*NUMNOD
   C WRITE(2,30)LASMAT
   C
30  FORMAT(IH,41HNUMBER OF ROWS OF LARGE ASSEMBLY MATRIX =,I5)
   C
   C NOSPRL=2*NOPRLN
   C WRITE(2,31)NOSPRL
   C
31  FORMAT(IH,43HNUMBER OF DISPLACEMENTS PER RADIAL LENGTH =,I2)
   C
   C NASSEM=3*NOSPRL
   C WRITE(2,32)NASSEM
   C
32  FORMAT(IH,33HSIZE OF THE NEW ASSEMBLY MATRIX =,I3)
   C
   C NTREAT=5*NOSPRL
   C WRITE(2,33)NTREAT
   C
33  FORMAT(IH,27HWIDTH OF TREATMENT MATRIX =,I3)
   C
   C NTWOSQ=NUMBER OF EXTRA DISPL. FOR EACH EXTRA ROW OF RADIAL RECTANG.
   C
   C NTWOSQ=2*NOSPRL
   C WRITE(2,34)NTWOSQ
   C
34  FORMAT(IH,77HNUMBER OF EXTRA DISPLACEMENTS FOR EACH ADDITIONAL ROW =,I3)
   C
   C 1W(RADIAL) OF RECTANGLES =,I3)
   C
   C CALL STFSTA(NPHSQR,NPHSQA,NDVRNO,NOPRLN,NOPALN,NUMNOD,LASMAT)
   C 1,NOSPRL)
   C
   C STOP
   C END
   C SUBROUTINE STFSTA(NPHSQR,NPHSQA,NDVRNO,NOPRLN,NOPALN,NUMNOD,LASMAT)
   C 1,NOSPRL)
   C
   C THIS IS TO FIND STRESSES AT THE PRINCIPAL NODES
   C
   C LIST OF SUBROUTINES
   C
   C 1 = FORMSE
   C
   C 2 = READRZ
   C
3 = REDISP
4 = SPLITDS
5 = FORMNO
6 = COLECT
7 = AINESS
8 = FORMEA
9 = FMCICT
10 = ORTOPR
11 = FORMCE
12 = BACSTA
13 = PRINTC
14 = TRANSC
15 = FRMCTC
16 = PNTCTC
17 = EQUIVA
18 = RECIPR
19 = MULMAT
20 = CTCICT
21 = PNCICT
22 = COEFFA
23 = OBTANA
24 = PRINTA
25 = OBTNUC
26 = DIFFER
27 = PUUCER
28 = XBARS
29 = CHNGEA
30 = PUNCH
30TH JANUARY 71

DIMENSION XG(NUMNO), YG(NUMNO), B(LASMAT), UG(NUMNO), WG(NUMNO),
       SF(4,4)

DIMENSION XG(63), YG(63), B(126), UG(63), WG(63), SF(4,4)

CALL ITIME(BEGIN)

READING IN YOUNGS MODULUS OF ELASTICITY AND POISSON'S RATIO

NODTAP = 10

10 FORMAT(E10.3, F10.3)
READ (1, 10) E, ENU
WRITE(2, 20) E, ENU

20 FORMAT(1H11/10X, 3OH YOUNGS MODULUS OF ELASTICITY =, 2PE9.1/10X,
       1 18H POISSON'S RATIO =, 0PF10.3/)

CALL FORMSE(SE, E, ENU)

CALL PRNTSE(SE)

CALL READRZ(NPHSQ, NDVRNO, NOPRL, NOPALM, NUMNO, NDSRL, NODTAP)

CALL REDISP(IDISPL, B, LASMAT)

CALL SPLTDS(LASMAT, NUMNO, B, UG, WG)

500 FORMAT(5X, 5I5)
READ(1, 500) LOBEAX, LOENAX, LOBERA, LOENRA
WRITE(2, 502) LOBEAX, LOENAX, LOBERA, LOENRA

502 FORMAT(1H11//10X,
       1 3OHLOCATION AT BEGINING OF AXIS =, 15/10X,
       2 3OHLOCATION AT END OF AXIS =, 15/10X,
       3 3OHLOCATION AT BEGINING OF RADIUS, 15/10X,
       4 3OHLOCATION AT END OF RADIUS, 15/)

NE= NUMBER OF COEFFICIENTS IN THE EQUATION OF THE SURFACE
NP= NUMBER OF POINTS, THIS MUST BE GREATER THAN NE

NE= 15
NP= 19
CALL FORNOD(LOBEAX, LOENAX, LOBERA, LOENRA, NPHSQ, XG, YG, UG, WG,

       SF, NUMNO, NE, NP, NPHSQ)

CALL ITIME(END)

21 FORMAT(1HO, 21H LENGTH OF TIME TAKEN =, 110, 8HSECONDS.)
SUBROUTINE CONSTI(SE)
DIMENSION SE(4,4)
THIS IS TO FORM THE MATRIX WHICH STATES THE CONSTITUTIVE RELATION
S BETWEEN STRESS S AND STRAIN E, THAT IS S = SE * E
READING IN MATERIAL PROPERTIES

CALL FORMSE(SE,E,ENU)
CALL PRNTSE(SE)
RETURN
END
SUBROUTINE FORMSE(SE,E,ENU)
DIMENSION SE(4,4)
THE PURPOSE OF THIS SUBROUTINE IS TO FORM THE MATRIX WHICH WHEN
POST MULTIPLIED BY THE AXI-SYMMETRIC STRAINS WILL GIVE THE
AXI-SYMMETRIC STRESS

NOMENCLATURE

E = YOUNG'S MODULUS OF ELASTICITY
ENU = POISSON'S RATIO
EAMBD=LA4BDA ONE OF LAME'S CONSTANTS
G = MODULUS OF RIGIDITY, THE SECOND LAME CONSTANT
CAY = K, THE BULK MODULUS

G=E/(2.0*(1.0+ENU))
CAY = F/(3.0*(1.0-2.0*ENU)),
FAMBSDA = CAY - 2.0*G/3.0
SE(1,1)=FAMBSDA+2.0*G
SE(2,2)=SE(1,1)
SE(3,3)=SE(1,1)
SE(4,4)=G
SE(1,2)=FAMBSDA
SUBROUTINE PRNTSE(SE)
DIMENSION SE(4,4)

THIS IS TO PRINT OUT SE

WRITE(2,30)
30 FORMAT(1H1/43X,36H- - - - - - - - - -  - - - - - - - - - - - - - - //)

WRITE(2,40)((SE(I,J),J=1,4),I=1,4)
40 FORMAT((22X ,4 C 2PE10*2,1IX)//)//)

RETURN
END

SUBROUTINE READRZ(NPSOA,NDVRKO,NDPRWI,NOPALN,NUMNOD,NDSPRL,NODTAPI)

DIMENSION R(NUMNOD),Z(NUMNOD)
DIMENSION R(63),Z(63)
INTEGER STANOD,FTNNOD

NDVRNO = NUMBER OF DISTINCT VERTEX NODES

READING FROM TAPE AND WRITING NODES AND THEIR R Z ORDINATES

REWRIND NODTAP

WRITE(2,98)
98 FORMAT(1H1/32X,4HNODE,19X,7HZ(NODE),19X,7HR(NODE)//)
DO 100 INODE=1,NDVRNO
  READ(NODTAP)NODE,(Z(NODE),R(NODE))
  WRITE(2,193)NODE,(Z(NODE),R(NODE))
193 FORMAT(33X,13,16X,F10.4,16X,F10.4)
100 CONTINUE

THIS SECTION IS TO FORM THE R Z ORDINATES ON THE RADIAL LINES BETWEEN

THE NODES

WRITE(2,203)NOPALN
203 FORMAT(1H0,44HTOTAL NO OF NODES/AXIAL LENGTH(NOPALN) =,I2//)

WRITE(2,98)
DO 300 INODAX=1,NOPALN,2
  WRITE(2,204)INODAX
  204 FORMAT(1H,10X,8HINODAX =,I2)
  STANOD=STANOD+(INODAX-1)*NOPRWI
  FINNOD=FINNOD+INODAX*NOPRWI-1
  WRITE(2,205)STANOD,FINNOD
  205 FORMAT(1H,1X,8HSTANOD =,I2,10X,8HF1NNOD =,I2)
  DO 200 NODE=STANOD,FINNOD,2
    J=NODE-1
    K=NODE+1
    WRITE(2,206)J,K
    206 FORMAT(1H0,20X,2HJ =,I2,3X,2HK =,I2)
    R(NODE)=(R(J)+R(K))/2.0
    Z(NODE)=(Z(J)+Z(K))/2.0
    WRITE(2,193)NODE,(Z(NODE),R(NODE))
  200 CONTINUE
300 CONTINUE

THIS SECTION FORMS THE NODES (AND THEIR R Z ORDINATES) BETWEEN
RADIi

WRITE(2,301)NPHSQA,NOPRWI,NDSPRL
301 FORMAT(1H0,10X,150HNUMBER OF ROWS OF PHYSICAL SQUARES PER AXIAL LENGTH(NPHSQA),I2/10X,
2/10X,
3 48HTOTAL NUMBER OF NODES PER RADIAL WIDTH(NOPRWI) =,I2/10X,
4 50HNUMBER OF DISPLACEMENTS PER RADIAL LENGTH(NDSPRL)=,I2)

DO 500 IAXIAL=1,NPHSQA
  WRITE(2,98)IAXIAL
  98 FORMAT(1H0,10X,8HIAXIAL =,I2)
  STANOD=IAXIAL+(IAXIAL-1)*NDSPRL
  FINNOD=IAXIAL+NDSPRL
  WRITE(2,205)STANOD,FINNOD
  DO 400 NODE=STANOD,FINNOD
    J=NODE-NOPRWI
    K=NODE+NOPRWI
    WRITE(2,206)J,K
    206 FORMAT(1H0,20X,2HJ =,I2,3X,2HK =,I2)
    R(NODE)=(R(J)+R(K))/2.0
    Z(NODE)=(Z(J)+Z(K))/2.0
    WRITE(2,193)NODE,(Z(NODE),R(NODE))
  400 CONTINUE
500 CONTINUE

RETURN
END

SUBROUTINE PEDISP(IDISPL,B,LASMAI)
DIMENSION B(LASMAI)
DIMENSION BLASMAI)
DIMENSION B( 126)

THIS S/R IS TO READ THE DISPLACEMENTS FROM MAGNETIC TAPE (IDISPL)

READ(IDISPL)NPSQRL,NPSQAL,NPLDS,NAXIAL,NTOT,B
WRITE(2,100)NPSQRL,NPSQAL,NPLDS,NAXIAL,NTOT
100 FORMAT(1H0,10X,
1 52HNUMBER OF ROWS OF PHYSICAL SQUARES PER RADIAL WIDTH =,I2/10X,
SUBROUTINE SPLTDSI(LASMAT, NUMNOD, B, U, W)
S/R TO SPLIT DISPLACEMENTS INTO RADIAL(U) AND AXIAL(W)

DIMENSION B(LASMAT), U(NUMNOD), W(NUMNOD)
DIMENSION B(126), U(63), W(63)
ODD NUMBERED B ARE RADIAL
EVEN NUMBERED B ARE AXIAL

DO 10 I = 1, NUMNOD
M = 2*I - 1
N = 2*I
U(I) = B(M)
W(I) = B(N)
WRITE(2,8)I, U(I), W(I)
8 FORMAT(1H, 10X, 14H NODE NUMBER = I2, 10X, 8H U(NODE) = 1PE10.3, 110X, 8H W(NODE) = 1PE10.3)
10 CONTINUE
RETURN
END

SUBROUTINE FORNODILOBEAX, LOENAX, LOERA, LOENRA, NPHSQR, XG, YG, UG, WG, ISE, NUMNOD, NE, NP, NPHSQA)
DIMENSION XG(NUMNOD), YG(NUMNOD), UG(NUMNOD), WG(NUMNOD),
ISTRESA(NPHSQR, NPHSQA, 4), STRANA(NPHSQR, NPHSQA, 4),
2RADIA(NPHSQR, NPHSQA), AXIALA(NPHSQR, NPHSQA)
DIMENSION XG(63), YG(63), UG(63), WG(63)
ISTRESA(3, 4, 4), STRANA(3, 4, 4),
2RADIA(3, 4), AXIALA(3, 4)
DIMENSION ST(4, 4), X(19), Y(19), U(19), W(19), NODNUM(19), STRESS(4), ISTRAIN(4)

THIS IS TO FORM THE NODE NUMBERS AROUND A PRINCIPAL NODE

NOMENCLATURE

NPHSQR=NUMBER OF PHYSICAL SQUARES PER RADIAL LENGTH
NUMPRL=NODES PER RADIAL LENGTH
LOBEAX=LOCATION BEGINING ALONG AXIS
LOENAX=LOCATION ENDING ALONG AXIS
LOBERA=LOCATION BEGINING ALONG RADIUS
LOENRA=LOCATION ENDING ALONG RADIUS
IXISTA=AXIAL START INDEX
IXIEND=AXIAL END INDEX
IXADEND=RADIAL END NUMBER
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.

NODPRL=2*NPHSQR+1
IXISTA=LOBEAX+1
IXIEND=LOENAX-1
NADSTA=LOBERA+1
NADEND=LOENRA-1
DO 100 IAXIAL=IXISTA,IXIEND
TAXIM1=2*(IAXIAL-1)*NODPRL
DO 101 IRADIUS=NADSTA,NADEND
CALL ITIME(IBEGIN)
MODEP=PRINCIPAL NODE
NODNUM=ARRAY OF NODE NUMBERS
THE INTENTION IS TO FORM A SET OF NODE NUMBERS IN ORDER TO CALL
ORDINATES AND DISPLACEMENTS OF NODES AROUND A PRINCIPAL NODE
THEN USE METHOD OF LEAST SQUARES TO FIT SURFACES TO THE DISPLACEMENTS OF THAT REGION IN ORDER TO FIND THE STRAINS AND STRESSES AT THAT NODE.
SUBROUTINE COLECT(NODNUM, XG, YG, UG, WG, NODEPR, SF, NE, NP, NUMNOD, R, Z, STRESS, STRAIN)

DIMENSION XG(NUMNOD), YG(NUMNOD), UG(NUMNOD), WG(NUMNOD)

DIMENSION SE(4,4), X(19), Y(19), U(19), W(19), NODENUM(19), STRESS(4), STRAIN(4)

THIS IS TO GET THE X, Y, U, AND W ARRAYS AROUND THE PRINCIPAL NODES

DO 10 I=1, NP

NODE=NODNUM(I)
X(I)=XG(NODE)
Y(I)=YG(NODE)
U(I)=UG(NODE)
W(I)=WG(NODE)

10 CONTINUE

R=XG(NODEPR)
Z=YG(NODEPR)

CALL AINESS(NE, NP, X, Y, U, W, R, Z, SE, STRESS, STRAIN)

RETURN

END

SUBROUTINE AINESS(NE, NP, X, Y, U, W, P, Z, SF, STRESS, STRAIN)

DIMENSION X(19), Y(19), U(19), W(19), A(30), STRAIN(4), SE(4,4),
STRESS(4)

TO CALCULATE THE FOUR STRAINS AT A POINT (R, Z) AND USING THE
CONSTITUTIVE MATRIX (SE) CALCULATE THE STRESSES AT THAT POINT
GIVEN THE ARRAYS OF RADIAL DISPLACEMENTS U AND AXIAL DISPLACEMENTS W FOR THE POINTS WHOSE ARRAY OF ORDINATES ARE X AND Y
CALL FORMEA(NE,NP,X,Y,U,W,A)
CALL RZSTRA(R,Z,NE,A,STRAIN)
CALL STRESSES(STRESS,SE,STRAIN)
WRITE(2,610)(STRESS(M),M=1,4)
610 FORMAT(12X,15H$$(\Theta,\Theta)''=,F10.4,6X,7HS(R,R)=,F10.4,6X,7HS(Z,Z)=,F10.4/)
RETURN
END
SUBROUTINE FORMEA(NE,NP,X,Y,U,W,A)
DIMENSION X(19),Y(19),C(19,15),CICT(15,19),U(19),AU(15),W(19),AW(15),A(30)
CALL FMCICT(NE,NP,X,Y,C,CICT,XMIN,YMIN,XPRINC,YPRINC)
CALL COEFFA(NE,NP,X,Y,C,CICT,XMIN,YMIN,XPRINC,YPRINC,U,AU)
CALL COEFFA(NE,NP,X,Y,C,CICT,XMIN,YMIN,XPRINC,YPRINC,W,AW)
CALL F0RMWA(AU,AW,A,NE)
RETURN
END
SUBROUTINE FMCICT(NE,NP,X,Y,C,CICT,XMIN,YMIN,XPRINC,YPRINC)
DIMENSION X(19),Y(19),C(19,15),CICT(15,19),CT(15,19),CTC(15,15),CICT(15,15),CTC(15,15)
TO FORM CICT
CALL ORTOPR(X,Y,NP,XMIN,YMIN,XPRINC,YPRINC)
CALL FORMCE(X,Y,NP,NE,C)
CALL BACSTA(X,Y,NP,XMIN,YMIN,XPRINC,YPRINC)
CALL PRINTC(C,NE,NP)
CALL TRANSC(C,NP,NE,CT)
CALL PRNTC(CT,NE,NP)
CALL FRMCTC(CT,C,CTC,NE,NP)
120 FORMAT(1H1/10X,10HMATRIX CTC/)
WRITE(2,120)
CALL PNTCTC(CTC,NE)
CALL EQUIVA(CTC,NE,D)
CALL RECIPP(CTC,NE)
121 FORMAT(1H1/10X,10HMATRIX (CTC)**(-1)/)
WRITE(2,121)
CALL PNTCTC(CTC,NE)
SUBROUTINE ORTOPR(X, Y, MP, XMIN, YMIN, XPRINC, YPRINC)
DIMENSION X(19), Y(19)
TO TRANSFER ORIGIN TO PRINCIPAL NODE AND SCALE
XMIN=5.6E-70
YMIN=5.6E-70
NPRTNC=(NP+1)/2
XPRINC=X(NPRTNC)
YPRINC=Y(NPRTNC)
DO 10 I=1, NP
X(I)=X(I)-XPRINC
Y(I)=Y(I)-YPRINC
10 CONTINUE
DO 20 I=1, NP
IF(ABS(X(I))-1.0E-7)16,16,15
15 CONTINUE
XMIN=AMIN1(XMIN, ABS(X(I)))
16 CONTINUE
IF(ABS(Y(I))-1.0E-7)18,18,17
17 CONTINUE
YMIN=AMIN1(YMIN, ABS(Y(I)))
18 CONTINUE
DO 20 I=1, NP
X(I)=X(I)/XMIN
Y(I)=Y(I)/YMIN
20 CONTINUE
RETURN
END
SUBROUTINE FORMCE(X, Y, NP, NE, C)
DIMENSION C(19,15), X(19), Y(19)
THIS PROGRAM IS TO GENERATE THE MATRIX C WHICH RELATES DISPLACEMENTS U VIA THEIR ORDINATES TO THE COEFFICIENTS A
SUCH THAT U=C*A + E
DO 10 I=1, NP
C(I,1)=1.0
C(I,2)=X(I)
C(I,3)=Y(I)
10 CONTINUE
RETURN
END
SUBROUTINE BACSTA(X,Y,NP,XMIN,YMIN,XPRINC,YPRINC)
DIMENSION X(19),Y(19)
C TO TRANSFER THE ORIGIN BACK TO ITS INITIAL ORIGIN
DO 10 I=1,NP
X(I)=X(I)*XMIN+XPRINC
Y(I)=Y(I)*YMIN+YPRINC
10 CONTINUE
RETURN
END

SUBROUTINE PRINTC(C,NE,NP)
DIMENSION C(19,15)
WRITE(2,120)
DO 100 I=1,NE
write (2,140) (C(I,J),J=1,NE)
140 FORMAT(1X,1P10E12.4)
120 FORMAT(1H1,10X,9HMATRIX C/)
100 CONTINUE
C THIS PROGRAM WRITES OUT MATRIX C
RETURN
END

SUBROUTINE TRAMSC(C,NE,NP)
DIMENSION C(19,15),CT(15,19)
C TO TRANSPOSE MATRIX C INTO CT
NJ=21
DO 20 I=1,NP
DO 10 J=1,NF
CT(J,I)=C(I,J)
10 CONTINUE
20 CONTINUE
RETURN
END

SUBROUTINE PRNTCT(CT,NE,NP)
DIMENSION CT(15,19)
120 FORMAT(1H1,10X,9HMATRIX CT/)
WRITE(2,120)
DO 100 I=1,NE
WRITE(2,160)CT(I,1:19)
100 CONTINUE
RETURN
END
SUBROUTINE FPMCTC(CT, C, CTC, NE, NP)
DIMENSION CT(15,19), C(19,15), CTC(15,15)

TO MULTIPLY CT BY C TO FORM CTC
L = 15
M = 15
M = N

DO 30 I = 1, NE
DO 20 J = 1, NE
CTC(I, J) = 0.0
DO 10 K = 1, NP
CTC(I, J) = CTC(I, J) + CT(I, K) * C(K, J)
10 CONTINUE
20 CONTINUE
30 CONTINUE
RETURN

SUBROUTINE PNTCTC(CTC, NE)
DIMENSION CTC(15,15)

THIS IS TO PRINT MATRIX CTC

DO 100 I = 1, NE
FORMAT(///10X,2H1=,13/)
WRITE(2,160) I
160 FORMAT(1X,1P10F12.4)/)
WRITE(2,140) (CTC(I, J), J = 1, NE)
100 CONTINUE

RETURN

SUBROUTINE FOUIVA(CTC, NE, P)
DIMENSION CTC(15,15), O(15,15)

THIS SUBROUTINE IS TO MAKE MATRIX D EQUAL TO MATRIX CTC

DO 2 0 J = 1, ME
O(1, J) = CTC(I, J)
CONTINUE
RETURN

SUBROUTINE RFCIPR(ASUBST, N)

PARTIAL PIVOTING

FORMAT(1X,1P10F12.4)/)
WRITE(2,140) (CTC(I, J), J = 1, NE)
100 CONTINUE

RETURN

SUBROUTINE INVPOO(ASUBST, N)

PARTIAL PIVOTING
DO DOUBLE PRECISION PRODUC
DOUBLE PRECISION A(15,15),BIG,P,R,B
DIMENSION LOC(15),INVLOC(15),LOC(15),ASUBST(15,15)

DO 2000 I=1,N
   DO 1000 J=1,N
      A(I,J)=ASUBST(I,J)
   CONTINUE
2000 CONTINUE

CONTINUE

INITIALIZING LOC AND INVLOC
------------------------------------

PRODUC=1.0
NM1=N-1
DO 201 I=1,N
   LOC(I)=I
   INVLOC(I)=I
201 CONTINUE

TO FIND THE VALUE OF J TO GIVE THE LARGEST VALUE OF A(I,J)
-------------------

BIG = 0.0
L = 0

IN THE FOLLOWING, ONCE A ROW HAS BEEN USED IT CANT BE USED AGAIN
FOR PIVOTING, IT FOLLOWS THAT ALL ROWS MUST BE USED ONCE.

DO 203 J=I,N
   K = LOC(J)
   IF(BIG-ABS ( A(K,I)))202,603,603
   CONTINUE
   BIG = ABS ( A(K,I))
   L = K
202 CONTINUE

IF(L-LOC(I))211,210,209
209 CONTINUE

THE PIVOT P
-------------

WRITE(2,G)I,A(L,I),L
FORMAT(1X,2HI=,I2,10X,7HA(L,I)=,1PD10.3,10X,2HL=,12)
PRODUC=PRODUC*A(L,I)
355 FORMAT(1H,48X,1SHPRODUCT OF A(L,I)=,1PD10.3)
WRITE(2,355)PRODUC
P=1.0/ A(L,I)
A(L,I) = 1.0

C
REDUCING THE PIVOTAL ROW

```
C C
DO 204 K=1,N
   A(L,K) = A(L,K) * P
C
204 CONTINUE
DO 207 J=1,N
   IF(J-L)205,607,205
C
205 CONTINUE
   R = A(J,I)
   A(J,I) = 0.0
DO 206 K=1,N
   A(J,K) = A(J,K) - A(L,K) * R
```

SUBTRACTING R * PIVOTAL ROW FROM OTHER ROWS

```
C C
206 CONTINUE
C
607 CONTINUE
207 CONTINUE
DO 801 K=1,N
   LOC(K)=LOC(K)
C
801 CONTINUE
C C
RE-ARRANGEMENT OF COLUMNS
C
C
DO 804 J=1, NM1
   K=INVLOC(J)
   DO 903 I=1,N
      B=A(I,J)
      A(I,J)=A(I,K)
      A(I,K)=B
C
903 CONTINUE
   L=LOC(J)
   LOC(K)=L
   LOC(J)=J
   INVLOC(J)=J
   INVLOC(L)=K
C
804 CONTINUE
```

C C
RE-ARRANGEMENT OF ROWS
C
C
DO 901 K=1,N
   L=LOC(K)
   LOC(K)=L
   INVLOC(L)=K
C
901 CONTINUE
```

DO 604 I=1, NM1
   FORMAT(//X,7HP,8W I = ,12/)
      K=LOC(I)
   DO 903 J=1,N
      R=A(I,J)
      A(I,J)=A(K,J)
      A(K,J)=R
C
903 CONTINUE
   L=INVLOC(I)
   INVLOC(K)=L
   INVLOC(I)=I
   LOC(I)=I
   LOC(K)=K
CONTINUE
CONTINUE
DO 4000 I=1,N
    DO 3000 J=1,N
        ASUBST(I,J)=A(I,J)
3000 CONTINUE
CONTINUE
RETURN
END

SUBROUTINE PRINTA(I,L,LOC,INVL0C,MARKER,N,A)
DIMENSION LOC(15),INVL0C(15),A(15,15)

THIS SUBROUTINE IS TO PRINT OUT THE MATRIX DURING THE STAGES OF INVERSION BY PARTIAL PIVOTING. I GIVES THE STAGE. L GIVES THE ROW NUMBER OF THE LARGEST ELEMENT FOR COLUMN I. ARRAY LOC GIVES THE ROW NUMBERS OF LARGEST ELEMENTS WHICH HAVE BEEN FOUND. INVL0C IS THE INVERSE OF LOC.

INVL0C(J) STORES THE COLUMN NUMBER K FOR WHICH ROW J HAS THE LARGEST ELEMENT IN THAT COLUMN K, AT STAGE I.

WHEREAS
LOC(K) STORES THE ROW NUMBER J, WHICH IS THE ROW OF THE LARGEST IN COLUMN K AT STAGE I.

MARKER GIVES THE PATH AFTER IF STATEMENT, AS FOLLOWS:
211 IF LOC(I) IS LESS THAN L
210 IF LOC(I) IS EQUAL TO L
209 IF LOC(I) IS GREATER THAN L

WRITE(2,10) I
10 FORMAT(1H1/41X,36HCOLUMN WHICH HAS JUST BEEN PIVOTED =,I2/)
WRITE(2,20)L
20 FORMAT(1H0,45HROW IN WHICH LARGEST ELEMENT HAS BEEN FOUND =,I2/)
WRITE(2,30)MARKER
30 FORMAT(1H0,9HFLOW PAST,I4, 9H CONTINUE/)
WRITE(2,40)N
40 FORMAT(1H0,23HSIZE OF SQUARE MATRIX =,I2/)
50 FORMAT(1H0,(3X,I2,3X))
WRITE(2,50)(LOC(J),J=1,N)

DO 100 IA=1,N
60 FORMAT(1H,1X,I2,2X,I2)
WRITE(2,60)IA,INVL0C(IA)
70 FORMAT(1H,1P15E.1/)
WRITE(2,70)A(IA,JC),JC=1,N
100 CONTINUE
RETURN

SUBROUTINE MULMAT(CTC,NF,D,P)
DIMENSION CTC(15,15),D(15,15),P(15,15)

THIS ROUTINE IS TO MULTIPLY MATRIX CTC BY D
DO 30 I=1,NE
DO 20 J=1,NE

P(I,J)=0.0
DO 10 K=1,NE
P(I,J)=P(I,J)+CTC(I,K)*D(K,J)
10 CONTINUE
20 CONTINUE
30 CONTINUE
RETURN
END

SUBROUTINE CTCTCT(CTC,CT,NF,NP,CICT)
DIMENSION CTC(15,15),CT(15,19),CICT(15,19)

THIS IS TO PRE-MULTIPLY CT BY CTC (WHICH HAS BEEN INVERTED)
DO 30 I=1,NE
DO 20 J=1,NP
CICT(I,J)=0.0
DO 10 K=1,NE
CICT(I,J)=CICT(I,J)+CTC(I,K)*CT(K,J)
10 CONTINUE
20 CONTINUE
30 CONTINUE
RETURN
END

SUBROUTINE PMCICT(CICT,NE,NP)
DIMENSION CICT(15,19)

THIS IS TO PRINT MATRIX CICT
120 FORMAT(1H1/10X,14HMATRIX CICT.)
WRITE(2,120)
DO 100 I=1,NE
160 FORMAT(///10X,2HI=,13/) WRITE(2,160)I
140 FORMAT((1X,1P10E12.4)) WRITE(2,140)(CICT(I,J),J=1,NP)
100 CONTINUE
RETURN
END

SUBROUTINE COEFFA(NE,NP,X,Y,C,CICT,XMIN,YMIN,XPRINC,YPRINC,U,A)
DIMENSION X(19),Y(19),C(19,15),CICT(15,19),U(19),A(15),UC(19),IER(25)

TO FIND THE COEFFICIENTS A, AND VERIFY THEM
CALL OBTANA(CICT,U,NE,NP,A)
CALL PRINTA(A,NE)
CALL OBTNUC(C,A,UC,NE,NP)
CALL DIFFER(U, UC, ER, NP)
CALL PUCER(U, UC, ER, X, Y, NP)
CALL XBARSD(MP, NP, EBAR, ESD)
WRITE(2, 29)EBAR, FS, NP
10 FORMAT(1H0, 17HMEAN DIFFERENCE =,1PE10.3, 10X,
20HSTANDARD DEVIATION =,1PE10.3, 10X,
20HNUMBER OF VALUES =,1I3//)
CALL CHNGEA(A, YMIN, YMIN, XPRNC, YPRNC, NE)
CALL PRINTA(A, NE)
RETURN
END

SUBROUTINE PRINTA(A, NE)
DIMENSION A(15)
10 FORMAT(1H1)
DO 100 I=1, NE
WRITE(2, 120)I, A(I)
120 FORMAT(1H0, 10X, 2HA!, 1H0,2H)=,F20.12)
100 CONTINUE
RETURN
END

SUBROUTINE OBTANA(CICT, U, NE, NP, A)
DIMENSION A(15), CICT(15, 19), U(19)
TO PRE-MULTIPLY U BY CICT TO OBTAIN A

L = 15
M = 19
DO 30 I=1, NE
A(I) = 0.0
DO 10 K=1, NP
A(I) = A(I) + CICT(I, K)*U(K)
10 CONTINUE
30 CONTINUE
RETURN
END

SUBROUTINE DIFFER(U, UC, ER, NP)
TO FIND THE DIFFERENCE BETWEEN U AND THE CALCULATED UC

L = 19
DO 10 I=1, NP
ER(I) = U(I) - UC(I)
10 CONTINUE
RETURN
END

SUBROUTINE PUCER(U, UC, ER, X, Y, NP)
DIMENSION U(19), UC(19), ER(19), X(19), Y(19)
TO PRINT U (THE ORIGINAL VALUES), UC (THE LEAST SQUARE FIT) AND THE ER
DIFFERENCES ER

WRITE(2,10)
10 FORMAT(1X,16X,9X,16X,9X,13X,13X,8HUC(LEAST SQ),8X,10HDIFFERENCE//)

L=19

DO 20 I=1,NP

WRITE(2,30)I,X(I),Y(I),U(I),UC(I),ER(I)

20 CONTINUE

RETURN

END

SUBROUTINE XBARSD( X , N , XBAR , SD )

THIS IS TO CALCULATE THE MEAN AND STANDARD DEVIATION OF AN ARRAY

DIMENSION X(19)

S=0.0
S2=0.0

DO 10 I=1,N

S=S+X(I)

10 CONTINUE

XBAR=S/FLOAT ( N )

DO 20 I=1,N

D=X(I)-XBAR

S2=S2+D*D

20 CONTINUE

NM1=N-1

SD=SQRT ( S2/FLOAT (NM1) )

RETURN

END

SUBROUTINE CHNGEA(A,XMIN,YMIN,XPRINC,YPRINC,NE)

TO CHANGE BACK A, TO ITS ORIGINAL ORIGIN AND SCALE

DIMENSION A(15), B(15)

DO 10 I=1,NE

B(I)=A(I)

10 CONTINUE

B(1)=A(1)

P=1.0/XMIN
Q=1.0/YMIN

B(2)=A(2)*P
B(3)=A(3)*Q

P2=P*P
PQ=P*Q
Q2=Q*Q

B(4)=A(4)*P2
B(5)=A(5)*PQ
B(6)=A(6)*Q2

P3 =P2*P
\[ P_{20} = P \times Q \]
\[ P_{20} = P \times Q \]
\[ Q_{3} = Q \times Q \]
\[ B(7) = A(7) \times P_{3} \]
\[ B(8) = A(8) \times P_{20} \]
\[ B(9) = A(9) \times P_{20} \]
\[ B(10) = A(10) \times Q_{3} \]
\[ P_{4} = P_{3} \times P \]
\[ P_{30} = P \times Q \]
\[ P_{20} = P \times Q_{2} \]
\[ Q_{3} = P \times Q_{3} \]
\[ B(11) = A(11) \times P_{4} \]
\[ B(12) = A(12) \times P_{30} \]
\[ B(13) = A(13) \times P_{20} \]
\[ B(14) = A(14) \times P_{20} \]
\[ B(15) = A(15) \times Q_{3} \]
\[ P_{4} = P_{3} \times P \]
\[ P_{3} = P_{2} \times Q \]
\[ P_{2} = P_{2} \times Q_{2} \]
\[ Q_{4} = Q \times Q_{3} \]
\[ B(11) = A(11) \times P_{4} \]
\[ B(12) = A(12) \times P_{30} \]
\[ B(13) = A(13) \times P_{20} \]
\[ B(14) = A(14) \times P_{20} \]
\[ B(15) = A(15) \times Q_{3} \]

\[ R = X \times P \]
\[ Z = Y \times P \]
\[ R_{2} = R \times R \]
\[ R_{2} = R \times R \]
\[ Z_{2} = Z \times Z \]
\[ R_{3} = R \times R \]
\[ R_{3} = R \times R \]
\[ Z_{3} = Z \times Z \]
\[ R_{4} = R \times R \]
\[ R_{4} = R \times R \]
\[ Z_{4} = Z \times Z \]
\[ R_{5} = R \times R \]
\[ R_{5} = R \times R \]
\[ Z_{5} = Z \times Z \]

\[ A(1) = B(1) - R \times B(2) - Z \times B(3) + R \times B(4) + R \times Z \times B(5) + 7 \times B(6) - R \times B(7) - R \times Z \times B(8) - R \times Z \times B(9) - Z \times B(10) + R \times B(11) + R \times Z \times B(12) + R \times Z \times B(13) + R \times Z \times B(14) + Z \times B(15) \]

\[ A(2) = B(2) - 2 \times R \times B(4) - 3 \times 0 \times R \times Z \times B(9) + Z \times B(10) - 4 \times 0 \times R \times Z \times B(11) + 2 \times 0 \times R \times Z \times B(12) - 4 \times 0 \times R \times Z \times B(13) - 2 \times 0 \times R \times Z \times B(14) - 2 \times 0 \times R \times Z \times B(15) \]

\[ A(3) = B(3) - R \times B(5) - 2 \times 0 \times Z \times B(6) + 2 \times 0 \times Z \times B(9) + Z \times Z \times B(10) - 3 \times 0 \times R \times Z \times B(12) - 3 \times 0 \times R \times Z \times B(13) - 3 \times 0 \times R \times Z \times B(14) - 3 \times 0 \times R \times Z \times B(15) \]

\[ A(4) = B(4) - 3 \times 0 \times R \times B(7) - Z \times B(8) + 6 \times 0 \times R \times B(11) + 3 \times 0 \times R \times B(12) + Z \times B(13) + 4 \times 0 \times R \times B(14) + 4 \times 0 \times R \times B(15) \]

\[ A(5) = B(5) - 2 \times 0 \times R \times B(8) - 2 \times 0 \times Z \times B(5) + 3 \times 0 \times R \times B(12) + 4 \times 0 \times R \times B(13) + 3 \times 0 \times Z \times B(14) + 6 \times 0 \times Z \times B(15) \]

\[ A(6) = B(6) - R \times B(9) - Z \times B(10) + R \times B(11) + 3 \times 0 \times R \times B(14) + 6 \times 0 \times Z \times B(15) \]

\[ A(7) = B(7) - 4 \times 0 \times R \times B(11) - Z \times B(12) \]
A(8)=R(8)-3.0*R*B(12)-2.0*Z*B(13)
A(9)=R(9)-2.0*R*B(13)-3.0*Z*B(14)
A(10)=R*B(14)-4.0*Z*B(15)
A(11)=B(11)
A(12)=B(12)
A(13)=B(13)
A(14)=B(14)
A(15)=B(15)
RETURN
END
SUBROUTINE PUNCHA(NE,NP,A,X,Y,U)
TO PUNCH OUT A ,X,Y AND U
DIMENSION A(15),X(19),Y(10),U(19)
WRITE(3,10)
WRITE(3,20)
DO 40 I=1,NP
WRITE(3,30)X(I),Y(I),U(I)
CONTINUE
RETURN
SUBROUTINE OBTNUC(C,A,UC,NE,NP)
DIMENSION C(19,15),A(15),UC(19)
TO MULTIPLY C*A TO GIVE UC, THE CALCULATED VALUES OF U
L=19
M=15
DO 30 I=1,NP
UC(I)=0.0
DO 10 K=1,NE
UC(I)=UC(I)+C(I,K)*A(K)
CONTINUE
RETURN
SUBROUTINE FORMWA(AU,AW,A,NE)
DIMENSION AU(15),AW(15),A(30)
TO FORM THE WHOLE OF A FROM AU AND AW
DO 10 I=1,NE
A(I)=AU(I)
A(I+NE)=AW(I)
10 CONTINUE
NW=2*NE
WRITE(2,20)(A(I),I=1,NW)
20 FORMAT(1H1,10HWHOLE OF A/1X,1P15E8,1/
RETURN
END
SUBROUTINE PDISTP(SA)
DIMENSION SA(4,30)
WRITE(2,10)
10 FORMAT(1H1,44HPOSITI0NAL STRAIN MATRIX TOBE * COEFFICIENTS/)
DO 40 I=1,4
WRITE(2,20)I
40 FORMAT(1H0,2HI=,12)
WRITE(2,30)(SA(I,J),J=1,30)
30 FORMAT(6X,10F11.4/)
40 CONTINUE
RETURN
END
SUBROUTINE RZSTRA(R,Z,ME,A,STRAIN)
DIMENSION S A (4,30),SIRA IN(4)
THIS IS TO CALCULATE THE STRAINS AT POINT WHOSE ORDINATES(R,Z)
CALL DISTRA(R,Z,SA,NE).
CALL PDISTR(SA)
CALL STRANA(NE,A,SA,STRAIN)
WRITE(2,510)(STRAIN(M),M=1,4)
510 FORMAT(12X,15HE(THETA,THETA)=,F10.6,6X,7HE(R,R)=,F10.6,6X,7HE(Z,Z)=,F10.6/
RETURN
END
SUBROUTINE DISTRA(R,Z,0,NE)
DIMENSION B(3,15),C(2,15),0(4,30)
TO FIND D SO THAT STRAINS = D*A
WHERE(A(J),J=1,NE)ARE THE COEFFICIENTS OF THE LEAST SQUARES BEST
FIT TO A POLYNOMINAL SURFACE, WHERE
STRAIN(1)=HOOP STRAIN E(THETA,THETA)= U/R
STRAIN(2)=RADIAL STRAIN E(R,R) = DU/DR
STRAIN(3)=AXIAL STRAIN E(Z,Z) = DW/DZ
STRAIN(4)=RADIAL-AXIAL SHEAR STRAIN E(R,Z) = DU/DZ + DW/DR
B IS RELATED TO U , C IS RELATED TO W
NW=2*N  
D0 10 I=1,4  
D0 5 J=1,NW  
B(I,J)=0.0  
5 CONTINUE  
10 CONTINUE  
D0 20 J=1,NE  
D0 20 I=1,2  
B(I,J)=0.0  
C(I,J)=0.0  
20 CONTINUE  
R(3,J)=0.0  
30 CONTINUE  
C  
CALCULATING POWERS OF R AND Z  
C  
RMI=1.0/R  
R2=R*R  
R3=R2*R  
R4=R3*R  
Z2=Z*Z  
Z3=Z2*Z  
Z4=Z3*Z  
RZ=R*Z  
R2Z=R2*Z  
R2Z2=R2*Z2  
R2Z3=R2*Z3  
R3Z=R3*Z  
R2Z2=R2*Z2  
RZ3=R*Z3  
E(THETA,THETA), (HOOP STRAIN) = U/R  
B(1, 1)=RM1  
B(1, 2)=1.0  
B(1, 3)=Z*RM1  
B(1, 4)=R  
B(1, 5)=Z  
B(1, 6)=Z2*RM1  
B(1, 7)=R2  
B(1, 8)=RZ  
B(1, 9)=Z2  
B(1,10)=Z3*RM1  
B(1,11)=R3  
B(1,12)=R2Z  
B(1,13)=RZ2  
B(1,14)=Z3  
B(1,15)=Z4*RM1  
B(1,16)=R4  
C  
99
B(1, 17) = R * Z
B(1, 18) = R * Z * Z
B(1, 19) = R * Z * Z * Z
B(1, 20) = Z
B(1, 21) = Z * Z

RADIAL STRAIN = DU/DR = E(R, R)
B(2, 2) = 1.0
B(2, 4) = 2.0 * R
B(2, 5) = Z
B(2, 7) = 3.0 * R * Z
B(2, 8) = 2.0 * R * Z
B(2, 9) = Z * Z
B(2, 11) = 4.0 * R * Z
B(2, 12) = 3.0 * R * Z * Z
B(2, 13) = 2.0 * R * Z * Z
B(2, 14) = Z * Z
B(2, 16) = 5.0 * R * Z
B(2, 17) = 4.0 * R * Z * Z
B(2, 18) = 3.0 * R * Z * Z
B(2, 19) = 2.0 * R * Z * Z
B(2, 20) = Z * Z

AXIAL STRAIN E(Z, Z) = DW/DR
C(1, 3) = 1.0
C(1, 5) = R
C(1, 6) = 2.0 * Z
C(1, 8) = R * Z
C(1, 9) = 2.0 * R * Z
C(1, 10) = 3.0 * R * Z
C(1, 12) = R * Z
C(1, 13) = 2.0 * R * Z
C(1, 14) = 3.0 * R * Z
C(1, 15) = 4.0 * Z
C(1, 17) = R * 4
C(1, 18) = 2.0 * R * 3
C(1, 19) = 3.0 * R * 3
C(1, 20) = 4.0 * R * 3
C(1, 21) = 5.0 * Z

B(3, J) = DU/DR AND C(2, J) = DW/DR
DO 40 J = 1, NE
C(2, J) = B(2, J)
B(3, J) = C(1, J)
40 CONTINUE

BUILDING UP D FROM B AND C
DO 50 J = 1, NE
DIMENSION S (4), STRAIN (4), SA (4,30), A(30)

THIS SEGMENT IS TO OBTAIN THE FOUR COMPONENTS OF STRAIN

\[ E(\Theta,\Theta) = \frac{U}{R}, \quad E(R,R) = \frac{DU}{DR}, \quad E(Z,Z) = \frac{DW}{DZ}, \]
\[ E(R,Z) = \frac{DU}{DZ} + \frac{DW}{DR}. \]

DO 20 I=1,4
   STRAIN(I)=0.0
   DO 10 J=1,NW
      STRAIN(I)=STRAIN(I)+SA(I,J)*A(J)
   CONTINUE

RETURN
END

SUBROUTINE STRES(STRESS, SF, STRAIN)
DIMENSION STRESS(4), SE(4,4), STRAIN(4)

THIS SUBROUTINE IS TO OBTAIN STRESS FROM STRAIN THE STRESS

COMPONENTS

\[ S(\Theta,\Theta), \quad S(R,R), \quad S(Z,Z), \quad S(R,Z) \]

ARE OBTAINED FROM THE STRAINS

\[ E(\Theta,\Theta), \quad E(R,R), \quad E(Z,Z), \quad E(R,Z) \]

BY PRE-MULTIPLYING THE STRAINS BY THE MATRIX SF, WHICH REPRESENTS
THE ELASTIC CONSTITUTIVE RELATIONSHIPS (STRESS STRAIN RELATION).

THE MATRIX SE IS FORMED BY THE SUBROUTINE FORMSE
C
C  SF*A
C
DO 20 I=1,4
    STRESS(I)=0.0
    DO 10 J=1,4
       STRESS(I)=STRESS(I)+SE(I,J)*STRAIN(J)
    CONTINUE
10
20 CONTINUE
C
RETURN
C
END

*ENTER
DATA
GO19DATA
1050
30.0 E 06 0.3
$2121 1 51 1 11