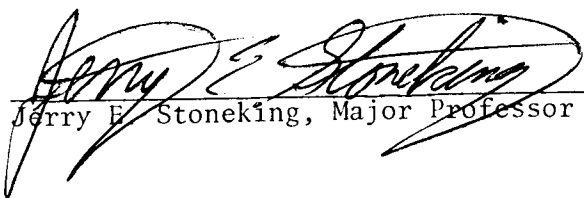
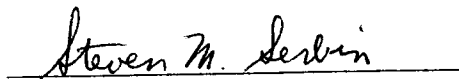


To the Graduate Council:

I am submitting herewith a thesis written by William J. Dorris entitled "Nonlinear Analysis of Reinforced Concrete Using the Finite Element Method." I recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Engineering Science.


Jerry E. Stoneking, Major Professor

We have read this thesis and
recommend its acceptance:


Steven M. Serbin


Bruce M. Dewey


Vice Chancellor
Graduate Studies and Research

Thesis
80
.D677
cop. 2

NONLINEAR ANALYSIS OF REINFORCED CONCRETE
USING THE FINITE ELEMENT METHOD

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

William J. Dorris

June 1980

3041748

ACKNOWLEDGMENTS

This study represents a belief in the value of education. The author would like to thank his parents, James and Charlotte Dorris, for teaching him the importance of making the effort to learn.

The numerical calculations performed in this study were done on an IBM 360/65 operated by The University of Tennessee Computing Center.

ABSTRACT

The major drawback of many finite element structural codes is the assumption of linear material behavior. An algorithm is presented which approximates nonlinear material properties with a piecewise linear solution. A computer code is developed to utilize the algorithm in the analysis of reinforced concrete structures. Due to difficulty in coding the algorithm, several suggestions are made for the completion and improvement of the computer program.

TABLE OF CONTENTS

SECTION	PAGE
1. INTRODUCTION	1
2. FINITE ELEMENT MODEL	4
2.1. Introduction	4
2.2. Stiffness Derivation	4
2.2.1. Element stiffness	4
2.2.2. Global stiffness	16
2.2.3. Boundary conditions	18
2.3. Material Behavior	18
2.3.1. Concrete	18
2.3.1.1. Biaxial tension	18
2.3.1.2. Biaxial compression	24
2.3.1.3. Tension-compression	33
2.3.1.4. Complete transition surface for concrete	33
2.3.2. Steel properties	36
2.3.2.1. Tension	36
2.3.2.2. Compression	36
2.3.3. Composite material	37
3. METHOD OF SOLUTION	39
3.1. Introduction	39
3.2. Survey of Methods	40
3.2.1. Secant stiffness method	40
3.2.2. Initial strain	43
3.2.3. Initial stress	45
3.2.4. Tangent stiffness	46
3.2.5. The proposed method of solution	49
3.3. Calculation of the Initial Stress	51
3.3.1. Primary cracking	51
3.3.2. Secondary cracking	55
3.3.3. Elastic 1 to elastic 2	55
3.3.4. Elastic 2 to plastic	56
3.3.5. Plastic	59
4. EXAMPLES	60
4.1. Introduction	60
4.2. Example I	60
4.3. Example II	63

SECTION	PAGE
5. RECOMMENDATIONS	69
5.1. Introduction	69
5.2. Recommendations for Completing and Improving the Code	69
5.3. Recommendations for Further Study	72
LIST OF REFERENCES	74
APPENDIXES	77
A. USER'S GUIDE TO "CRACK"	78
B. LISTING OF FORTRAN STATEMENTS IN "CRACK"	87
VITA	130

LIST OF FIGURES

FIGURE	PAGE
1. Local Coordinate System and Degrees of Freedom for Tocher Plate Element	6
2. Layering System	17
3. Transition Criterion for Concrete in Biaxial Tension	20
4. Cracked Element Coordinate System	22
5. Transition Criterion for Concrete in Biaxial Compression	26
6. Assumed Material Behavior for Concrete in Uniaxial Compression	27
7. Failure Criterion for Concrete in Biaxial Compression . . .	32
8. Transition Criterion for Concrete in Tension-Compression . .	34
9. Complete Transition Surface for Concrete	35
10. Assumed Material Behavior of Steel Reinforcement	38
11. Secant Stiffness Method	41
12. Initial Strain and Initial Stress Methods	44
13. Tangent Stiffness Iteration	48
14. Calculation of the Correct Stress State for a Bilinearly Elastic Material	52
15. Calculation of the Transition Stress State for Concrete Exceeding Biaxial Tension Transition Criterion	54
16. Calculation of Initial Stress for Concrete Undergoing Elastic to Plastic Transition	57
17. Example I	61
18. Load Deflection Curve for Example I	62
19. Example II	65
20. Load Deflection Curves for Example II	66

1. INTRODUCTION

In recent years, there have been many advances in the field of structural analysis. The appearance of high speed computers has made possible the development of numerical methods useful in the analysis of structures. Many problems which would be difficult (if not impossible) to solve using classical analytical techniques can now be accurately solved using numerical methods.

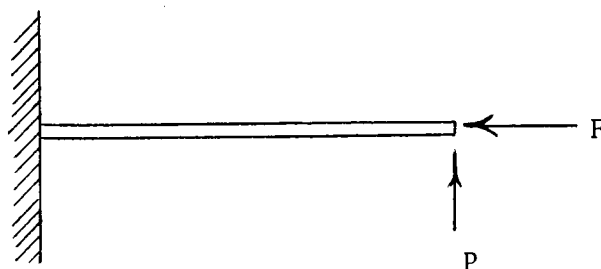
One of the most widely used and highly developed numerical methods is the finite element method. Originally introduced as a structural analysis technique by Turner et al. in 1956, the method has advanced rapidly and is now a useful tool of both the design engineer and the structural analyst. Numerous general purpose computer codes have been developed to perform fast and efficient analyses of many different types of structures and loading conditions.

The use of the finite element method as a tool for the analysis of structures has one major drawback. The method can easily be used in the analysis of linear structures, but unfortunately, an engineer often encounters situations where nonlinear systems must be analyzed. On those occasions, many of the general programs become useless since they are applicable to only linear systems. The engineer must then turn to some special purpose programs which employ special algorithms that attempt to compensate for nonlinear behavior.

The problem of nonlinearity in structures can arise from two sources; namely, geometric and material nonlinearity.

a. Geometric Nonlinearity. In the derivation of strain-displacement relations, it is generally assumed that displacements are small enough that higher order terms can be neglected. This assumption is necessary to insure a linear relation between the strains and displacements. A violation of this assumption requires the analyst to resort to a more complex analysis based on finite displacement strain expressions. For example, the deformation of a slender beam subjected to large transverse loads may be large enough to require the use of nonlinear strain displacement relations.

Another type of geometric nonlinearity is found in structural configurations such as the one shown below.



The axial stiffness of this structure is dependent on the transverse displacement of the end. This change in stiffness is not allowed in the usual linear analysis, therefore, a more complicated solution technique must be used to analyze this structure.

b. Material Nonlinearity. An assumption in linear structural analysis is that a linear constitutive relation exists between stress and strain. Many materials possess material properties of this nature at low stress levels, but as the stress increases, this linear relation

becomes less accurate. Since many structures must function at high stress levels an assumption of linear constitutive relations is not always valid. Reinforced concrete is a common structural material which is often loaded into the nonlinear range. Nonlinear material properties also exist in some materials at low stress levels, making linear analysis impossible for any loading range. Many biological materials fit into this category and must be analyzed in order to design much of the biomedical equipment used today.

The purpose of this research is to develop an algorithm for the nonlinear analysis of reinforced concrete slabs and shells. The algorithm is designed to analyze nonlinear behavior due to material properties and any geometric nonlinearity is neglected. The solution technique developed is piecewise linear and allows for changes in material properties of both the steel reinforcement and concrete.

2. FINITE ELEMENT MODEL

2.1. INTRODUCTION

In this study, reinforced concrete structures are analyzed using a finite element displacement model. Each finite element is triangular in shape with five degrees of freedom for each of its three corner nodes. Through the thickness, every element consists of a variable number of constant thickness layers. Each layer is allowed to possess an independent set of material properties which determines the stiffness contribution of that layer to the stiffness of the element. The total stiffness of the structure is constructed by summing the contributions made by each element. The following sections describe the derivation of the expression for the stiffness of an individual element and the assumed material behavior for the layers.

2.2. STIFFNESS DERIVATION

2.2.1. ELEMENT STIFFNESS

The basis of the analysis performed in this study is a three noded triangular element. Each of the three corner nodes has 3 translational and 2 rotational degrees of freedom:

$$(1a) \quad \{\Delta(x,y)\} = \begin{Bmatrix} u(x,y) \\ v(x,y) \\ w(x,y) \\ \theta_x(x,y) \\ \theta_y(x,y) \end{Bmatrix}$$

where,

$$(1b) \quad \theta_x = \frac{\partial w(x,y)}{\partial y}$$

$$(1c) \quad \theta_y = \frac{\partial w(x,y)}{\partial x}$$

The nodal degrees of freedom and the local coordinate system for an element are shown in Figure 1. The stiffness matrix for this element is derived using the principle of virtual work in a manner outlined by Zienkiewicz. (Ref. 16)

The first step in the derivation is to choose suitable interpolation functions which represent the variation of the generalized displacements within an element. For simplicity, the Tocher (Ref. 15) element, with the following displacement functions, is used in the present research.

$$(2a) \quad u(x,y) = \alpha_1 + \alpha_2 x + \alpha_3 y$$

$$(2b) \quad v(x,y) = \alpha_4 + \alpha_5 x + \alpha_6 y$$

$$(2c) \quad w(x,y) = \alpha_7 + \alpha_8 x + \alpha_9 y + \alpha_{10} x^2 + \alpha_{11} xy + \alpha_{12} y^2 \\ + \alpha_{13} x^3 + \alpha_{14} (x^2 y + xy^2) + \alpha_{15} y^3$$

$$(2d) \quad \theta_x(x,y) = \frac{\partial w(x,y)}{\partial y} = \alpha_9 + \alpha_{11} x + 2\alpha_{12} y + \alpha_{14} (x^2 + 2xy) \\ + 3\alpha_{15} y^2$$

$$(2e) \quad \theta_y(x,y) = -\frac{\partial w(x,y)}{\partial x} = -\{\alpha_8 + 2\alpha_{10} x + \alpha_{11} y + 3\alpha_{13} x^2 \\ + \alpha_{14} (2xy + y^2)\}$$

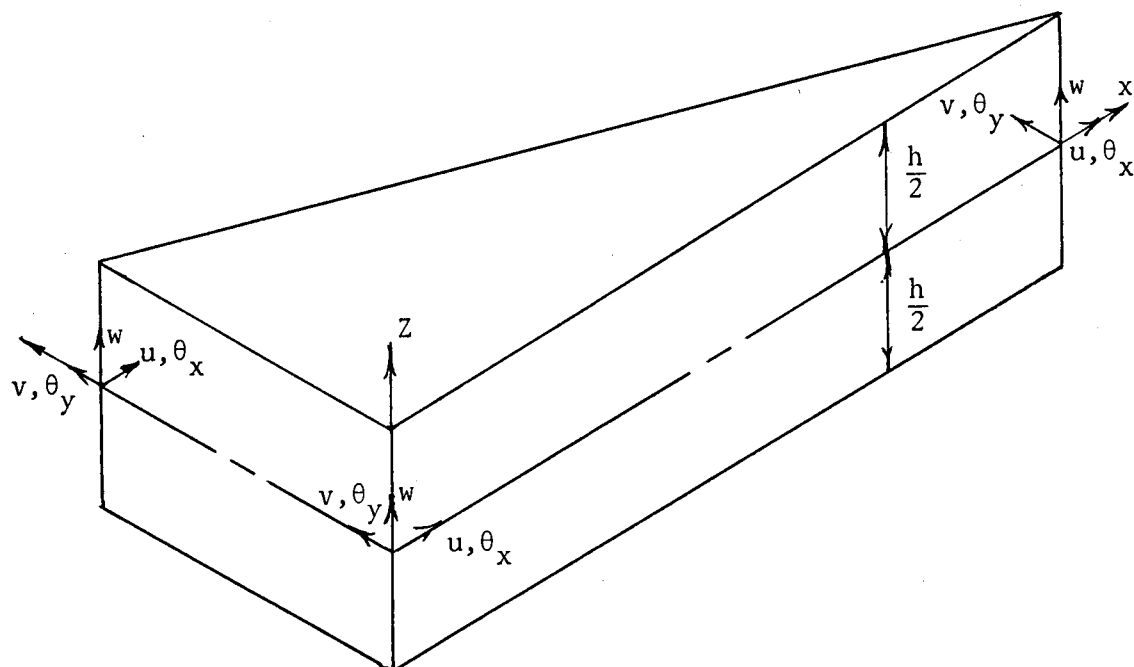


Figure 1. Local coordinate system and degrees of freedom for Tocher plate element

where,

α_i are constants to be determined.

x, y are local spatial coordinates.

The above system of equations can be written in matrix forms as

$$(2f) \quad \{\Delta(x, y)\} = [A(x, y)]\{\alpha\}$$

Equations (2a)-(2f) are valid for any point in the element and in particular at the 3 nodes which define the element. For example, at node 3:

$$(3a) \quad u(x_3, y_3) = \alpha_1 + \alpha_2 x_3 + \alpha_3 y_3$$

$$(3b) \quad v(x_3, y_3) = \alpha_4 + \alpha_5 x_3 + \alpha_6 y_3$$

$$(3c) \quad w(x_3, y_3) = \alpha_7 + \alpha_8 x_3 + \alpha_9 y_3 + \alpha_{10} x_3^2 + \alpha_{11} x_3 y_3 + \alpha_{12} y_3^2 \\ + \alpha_{13} x_3^3 + \alpha_{14} (x_3^2 y_3 + x_3 y_3^2) + \alpha_{15} y_3^3$$

$$(3d) \quad \theta_x(x_3, y_3) = \alpha_9 + \alpha_{11} x_3 + 2\alpha_{12} y_3 + \alpha_{14} (x_3^2 + 2x_3 y_3) + 3\alpha_{15} y_3^2$$

$$(3e) \quad \theta_y(x_3, y_3) = -\{\alpha_8 + 2\alpha_{10} x_3 + \alpha_{11} y_3 + 3\alpha_{13} x_3^2 + \alpha_{14} (2x_3 y_3 + y_3^2)\}$$

or in matrix form

$$(3f) \quad \{\Delta(x_3, y_3)\} = [A(x_3, y_3)] \{\alpha\}$$

Combining similar systems for the other two nodes with the system above, one obtains

$$(4) \quad \{\bar{\Delta}\} = \begin{Bmatrix} \{\Delta(x_1, y_1)\} \\ \{\Delta(x_2, y_2)\} \\ \{\Delta(x_3, y_3)\} \end{Bmatrix} = \begin{bmatrix} [A(x, y)] \\ [A(x, y)] \\ [A(x, y)] \end{bmatrix} \{\alpha\} = [\bar{A}] \{\alpha\}$$

Solving for $\{\alpha\}$ gives

$$(5) \quad \{\alpha\} = [\bar{A}]^{-1} \{\bar{\Delta}\}$$

Assuming one knows the nodal displacements, $\{\bar{\Delta}\}$, the following functional representation for the displacement field of the entire element can be obtained by combining equations (2f) and (5)

$$(6) \quad \{\Delta(x, y)\} = [A(x, y)] [\bar{A}]^{-1} \{\bar{\Delta}\}$$

In this study, emphasis is placed on the analysis of plates and shells. Therefore, it is appropriate to introduce the following basic assumptions of classical plate theory:

- a. Deflections are small enough that linear strain-displacements relations are valid.
- b. Normals to the undeformed surface remain normal to the deformed surface. This assumption implies that deformation due to transverse shear is negligible.
- c. All problems possess geometric linearity.

Using the first assumption above, one can express the components of the strain field at the element mid-surface as

$$(7a) \quad \epsilon_x(x, y, 0) = \frac{\partial u(x, y)}{\partial x}$$

$$(7b) \quad \epsilon_y(x, y, 0) = \frac{\partial v(x, y)}{\partial y}$$

$$(7c) \quad \gamma_{xy}(x,y,0) = \frac{\partial u(x,y)}{\partial y} + \frac{\partial v(x,y)}{\partial x}$$

$$(7d) \quad \chi_x(x,y,0) = \frac{\partial^2 w(x,y)}{\partial x^2} = - \frac{\partial \theta_y(x,y)}{\partial x}$$

$$(7e) \quad \chi_y(x,y,0) = \frac{\partial^2 w(x,y)}{\partial y^2} = \frac{\partial \theta_x(x,y)}{\partial y}$$

$$(7f) \quad \chi_{xy}(x,y,0) = 2 \frac{\partial^2 w(x,y)}{\partial x \partial y}$$

or rewriting in matrix notation:

$$(7g) \quad \begin{Bmatrix} \varepsilon(x,y,0) \\ \chi(x,y,0) \end{Bmatrix} = [D] \{\Delta(x,y)\}$$

where

$$[D] = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 & 0 & 0 \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{\partial}{\partial x} \\ 0 & 0 & 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & 2\frac{\partial^2}{\partial x \partial y} & 0 & 0 \end{bmatrix}$$

Substituting (7g) into (6) gives the following representation for the mid-surface strain in an element

$$(8a) \quad \left\{ \begin{array}{c} \varepsilon(x,y,0) \\ \hline \chi(x,y,0) \end{array} \right\} = [D] [A(x,y)] [\bar{A}]^{-1} \{\bar{\Delta}\}$$

or

$$(8b) \quad \left\{ \begin{array}{c} \varepsilon(x,y,0) \\ \hline \chi(x,y,0) \end{array} \right\} = [B(x,y)] [\bar{A}]^{-1} \{\bar{\Delta}\}$$

where

$$[B(x,y)] = [D] [A(x,y)]$$

The above expressions are valid only at the mid-surface ($Z = 0$ in local coordinates) of an element. The variation of strain through the thickness can be found by considering assumption (b) above. This assumption allows one to use the Kirchhoff-Love assumption which expresses the strain at the distance Z from the mid-surface as:

$$(9a) \quad \varepsilon_x(x,y,Z) = \varepsilon_x(x,y,0) - Z \frac{\partial^2 w(x,y)}{\partial x^2} = \varepsilon_x(x,y,0) - Z \chi_x(x,y,0)$$

$$(9b) \quad \varepsilon_y(x,y,Z) = \varepsilon_y(x,y,0) - Z \frac{\partial^2 w(x,y)}{\partial y^2} = \varepsilon_y(x,y,0) - Z \chi_y(x,y,0)$$

$$(9c) \quad \gamma_{xy}(x,y,Z) = \gamma_{xy}(x,y,0) - 2Z \frac{\partial^2 w(x,y)}{\partial x \partial y} = \gamma_{xy}(x,y,0) - Z \chi_{xy}(x,y,0)$$

The Kirchhoff-Love assumption can be written in matrix form as

$$(10) \quad \{\epsilon(x,y,Z)\} = [Z] \begin{Bmatrix} \epsilon_x(x,y,0) \\ \chi(x,y,0) \end{Bmatrix}$$

where,

$$[Z] = \begin{bmatrix} 1 & 0 & 0 & -Z & 0 & 0 \\ 0 & 1 & 0 & 0 & -Z & 0 \\ 0 & 0 & 1 & 0 & 0 & -Z \end{bmatrix}$$

If one substitutes (8a) into (10), the following expression for the strain field in the entire element is obtained:

$$(11) \quad \{\epsilon(x,y,Z)\} = [Z] [B(x,y)] [\bar{A}]^{-1} \{\bar{\Delta}\}$$

Now that the element strain field is known in terms of the assumed interpolation functions, one can find a similar expression for the element stress field. In general, the stress field in a body subjected to plane stress is given by

$$(12) \quad \{\sigma(x,y)\} = \begin{Bmatrix} \sigma_x(x,y) \\ \sigma_y(x,y) \\ \tau_{xy}(x,y) \end{Bmatrix} = [C] [\{\epsilon(x,y)\} - \{\epsilon_0(x,y)\}] - \{\sigma_0(x,y)\}$$

where

$[C]$ represents a matrix of suitable material properties

$\{\epsilon_0(x,y)\}$ represents any initial strain in the body

$\{\sigma_0(x,y)\}$ represents any initial stress in the body.

To get an expression for the stress field in an element in terms of the interpolation functions and nodal displacements, substitute (11) into (12). This substitution yields:

$$(13) \quad \{\sigma(x,y,z)\} = [C] \left[[Z][B(x,y)] [\bar{A}]^{-1} \{\bar{\Delta}\} - \{\epsilon_o(x,y)\} \right] + \{\sigma_o(x,y)\}$$

Now that the relationship between deformation and the internal resisting force is defined, one can apply the principle of virtual work to the element. In general, this principle states that

$$(14) \quad \delta W_E = \delta U_i$$

The above equation means that for any compatible virtual displacement $\{\delta\bar{\Delta}\}$ from an equilibrium state, the external virtual work, δW_E , is equal to the virtual strain energy, δU_i .

The work done by any concentrated nodal forces $\{\bar{F}\}$, due to a generalized virtual displacement of the nodes is given by

$$(15a) \quad \delta W_{CF} = \{\bar{F}\}^T \{\delta\bar{\Delta}\}$$

and the work done by any distributed forces, $\{p(x,y)\}$, applied to the element is given by

$$(15b) \quad \delta W_{DF} = \iint_A \{p(x,y)\}^T \{\Delta(x,y)\} dx dy$$

Substituting (6) into (15b) gives

$$(15c) \quad \delta W_{DF} = \iint_A \{p(x,y)\}^T [A(x,y)] dx dy [\bar{A}]^{-1} \{\delta\bar{\Delta}\}$$

Combining (15a) and (15c) one obtains as expression for the total external work done due to a virtual displacement:

$$(16) \quad \begin{aligned} \delta W_E &= \delta W_{CF} + \delta W_{DF} \\ &= \{\bar{F}\}^T \{\delta\bar{\Delta}\} + \iint_A \{p(x,y)\}^T [A(x,y)] dx dy [\bar{A}]^{-1} \{\delta\bar{\Delta}\} \end{aligned}$$

The strain energy present in an element due to a virtual nodal displacement is

$$(17) \quad \delta U_i = \iiint_{Vol} \{\sigma\}^T \{\delta\epsilon\} dx dy dz$$

Using (10), one can express the strain, $\{\delta\epsilon\}$, due to the virtual displacement as

$$(18) \quad \{\delta\epsilon\} = [Z] [B(x,y)] [\bar{A}]^{-1} \{\delta\bar{\Delta}\}$$

Substituting (16) and (17) into (14) yields

$$(19) \quad \begin{aligned} \{\bar{F}\} + \iint_A \{p(x,y)\}^T [A(x,y)] dx dy [\bar{A}]^{-1} \{\delta\bar{\Delta}\} \\ = \iiint_{Vol} \{\sigma\}^T [Z] [B(x,y)] [\bar{A}]^{-1} dx dy dz \{\delta\bar{\Delta}\} \end{aligned}$$

Since the above must be true for any compatible virtual displacement, then

$$\begin{aligned}
 (20) \quad \{\bar{F}\} + \iint_A \{p(x,y)\}^T [A(x,y)] dx dy [\bar{A}]^{-1} \\
 = \iiint_{Vol} \{\sigma\}^T [Z] [B(x,y)] dx dy dz [\bar{A}]^{-1}
 \end{aligned}$$

The substitution of (13) into the above expression results in

$$\begin{aligned}
 (21) \quad \{\bar{F}\} + \iint_A \{p(x,y)\}^T [A(x,y)] dx dy [\bar{A}]^{-1} \\
 = \iiint_{Vol} ([A]^{-1})^T [B(x,y)]^T [Z]^T [C] [Z] [B(x,y)] \\
 [\bar{A}]^{-1} dx dy dz \{\bar{\Delta}\} \\
 + \iiint_{Vol} [\sigma_o(x,y) - [C]\{\epsilon_o(x,y)\}] [Z] [B(x,y)] [\bar{A}]^{-1} dx dy dz
 \end{aligned}$$

or

$$(22a) \quad \{\bar{F}\}_{CF} + \{\bar{F}\}_{DF} + \{\bar{F}\}_{\sigma} + \{\bar{F}\}_{\epsilon} = [K] \{\bar{\Delta}\}$$

where,

$$(22b) \quad \{\bar{F}\}_{CF} = \text{concentrated nodal forces}$$

$$(22c) \quad \{\bar{F}\}_{DF} = \text{nodal forces due to distributed loads}$$

$$= \iint_A \{p(x,y)\}^T [A(x,y)] dx dy [\bar{A}]^{-1}$$

$$\begin{aligned}
 (22d) \quad \{F\}_\sigma &= \text{nodal forces due to initial stress} \\
 &= - \iiint_{\text{Vol}} \{\sigma_0(x,y)\}^T [Z] [B(x,y)] dx dy dz [\bar{A}]^{-1}
 \end{aligned}$$

$$\begin{aligned}
 (22e) \quad \{\bar{F}\}_\epsilon &= \text{nodal forces due to initial strain} \\
 &= \iiint_{\text{Vol}} [C] \{\epsilon_0(x,y)\} [Z] [B(x,y)] dx dy dz [\bar{A}]^{-1}
 \end{aligned}$$

and

$$\begin{aligned}
 (22f) \quad [K] &= \text{element stiffness matrix} \\
 &= \iiint_{\text{Vol}} ([\bar{A}]^{-1})^T [B(x,y)]^T [Z]^T [C] [Z] [B(x,y)] \\
 &\quad [\bar{A}]^{-1} dx dy dz
 \end{aligned}$$

Since $[\bar{A}]^{-1}$ is independent of x, y, z and $[B]$ is dependent on only x and y one can rewrite the above as

$$\begin{aligned}
 (23) \quad [K] &= ([\bar{A}]^{-1})^T \iint_A [B(x,y)]^T \int_{-t/2}^{t/2} [Z]^T [C] [Z] dz \\
 &\quad [B(x,y)] dx dy [\bar{A}]^{-1}
 \end{aligned}$$

From equation (23) one can easily introduce the concept of a layered element. The stiffness of a layered element is the same as the expression above except that the integration with respect to z is carried out layer by layer in the form.

$$\begin{aligned}
\int_{-t/2}^{t/2} [Z]^T [C] [Z] dz = & \int_{t_1}^{t_2} [Z]^T [C] [Z] dz + \int_{t_2}^{t_3} [Z]^T [C] [Z] dz \\
& + \dots + \int_{t_j}^{t_{j+1}} [Z]^T [C] [Z] dz + \dots + \int_{t_{NL}}^{t_{NL+1}} [Z]^T [C] [Z] dz
\end{aligned}$$

where

t_j , t_{j+1} refer to the distance from the mid-surface to the top and bottom of layer j , respectively (See Figure 2). NL = total number of layers.

The effect of the layering scheme is to allow an out of plane variation in the stress and strain values, thus introducing a pseudo third dimension to the element. This enables each element to follow a nonlinear constitutive law more closely, since it will now be able to change properties (or fail) in a more gradual manner.

2.2.2. GLOBAL STIFFNESS

The relationship between the element degrees of freedom and the external forces applied to that element is given by (22a). The corresponding system of equations for the entire structure is found in the following manner:

- a. The structure is divided into a number of triangular regions (elements) with each node of an element having a number and position with respect to a global numbering and coordinate system.

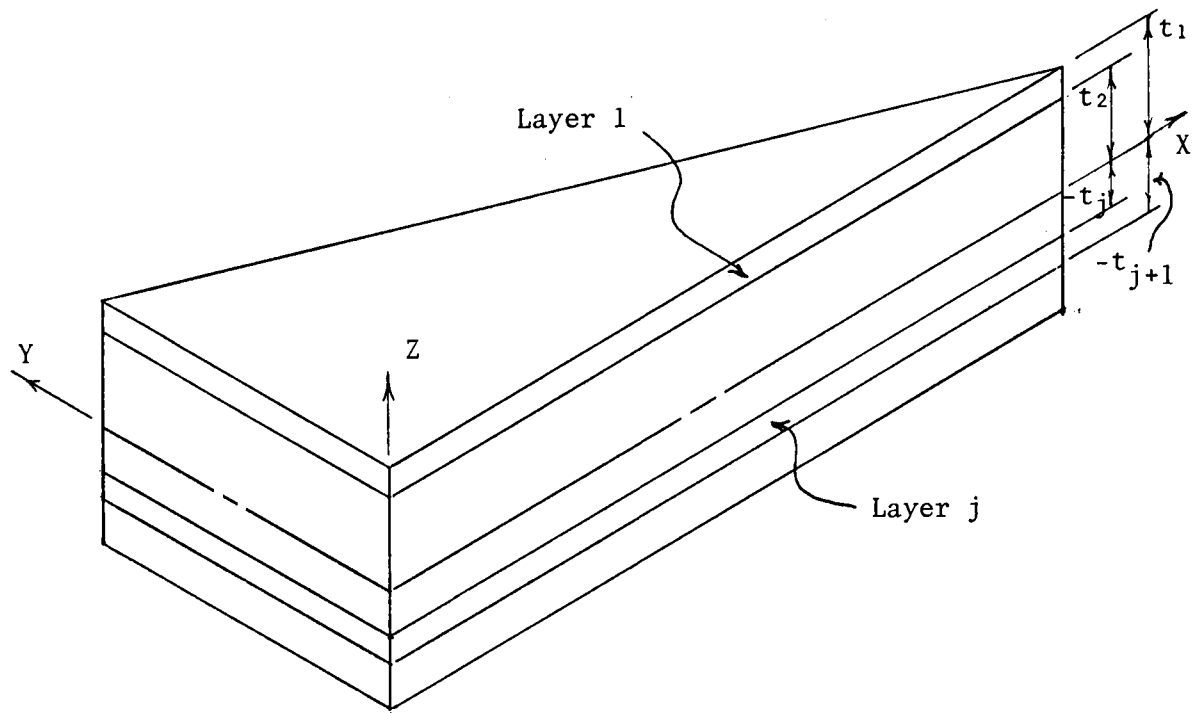


Figure 2. Layering system

b. The system of force-displacement (stiffness) equations for each element is derived using (22a). Note that this system is based on local coordinate and numbering systems.

c. The system of force-displacement equations for each element is transformed to a system based on the global coordinate and numbering systems.

d. The system of force-displacement equations for the entire structure is found by summing all of the transformed element systems.

2.2.3. BOUNDARY CONDITIONS

The global system of equations found in the previous section does not possess a unique solution, that is, the global stiffness matrix is singular. Physically, this means that rigid body motion of the structure has not been constrained. This difficulty may be corrected by specifying a suitable number of displacement boundary conditions.

2.3. MATERIAL BEHAVIOR

2.3.1. CONCRETE

2.3.1.1. BIAXIAL TENSION. When first loaded, the concrete is assumed to be a linearly elastic, homogenous isotropic material. Therefore, the stress can be related to the strain by

$$(24) \quad \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix} = \frac{1}{(1-\nu^2)} \begin{bmatrix} E & \nu & 0 \\ \nu & E & 0 \\ 0 & 0 & \frac{E(1-\nu)}{2} \end{bmatrix} \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{Bmatrix}$$

Following the work of Kupfer, Hilsdorf and Rush (Ref. 8), F. R. Hand (Ref. 6) suggested that concrete under biaxial tension will crack when the following relationship is satisfied

$$(25) \quad \tau_{\text{oct}} + \frac{1-\alpha}{1+\alpha} p - \left(\frac{2\sqrt{2}}{3}\right) \left(\frac{\alpha}{1+\alpha}\right) f_d = 0$$

where,

τ_{oct} = octahedral shear stress

$$= \frac{1}{3} \{2\sigma_1^2 + 2\sigma_2^2 - 2\sigma_1\sigma_2\}^{1/2} \quad (\text{for plane stress})$$

p = Hydrostatic stress

$$= \frac{\sigma_1 + \sigma_2}{3} \quad (\text{for plane stress})$$

$$\alpha = f_t / f_d$$

f_t = Concrete tensile strength

f_d = Plastic stress level for concrete

The above relation is shown in Figure 3.

When equation (25) is satisfied, a crack is assumed to form in a direction perpendicular to the maximum principal stress. The stress

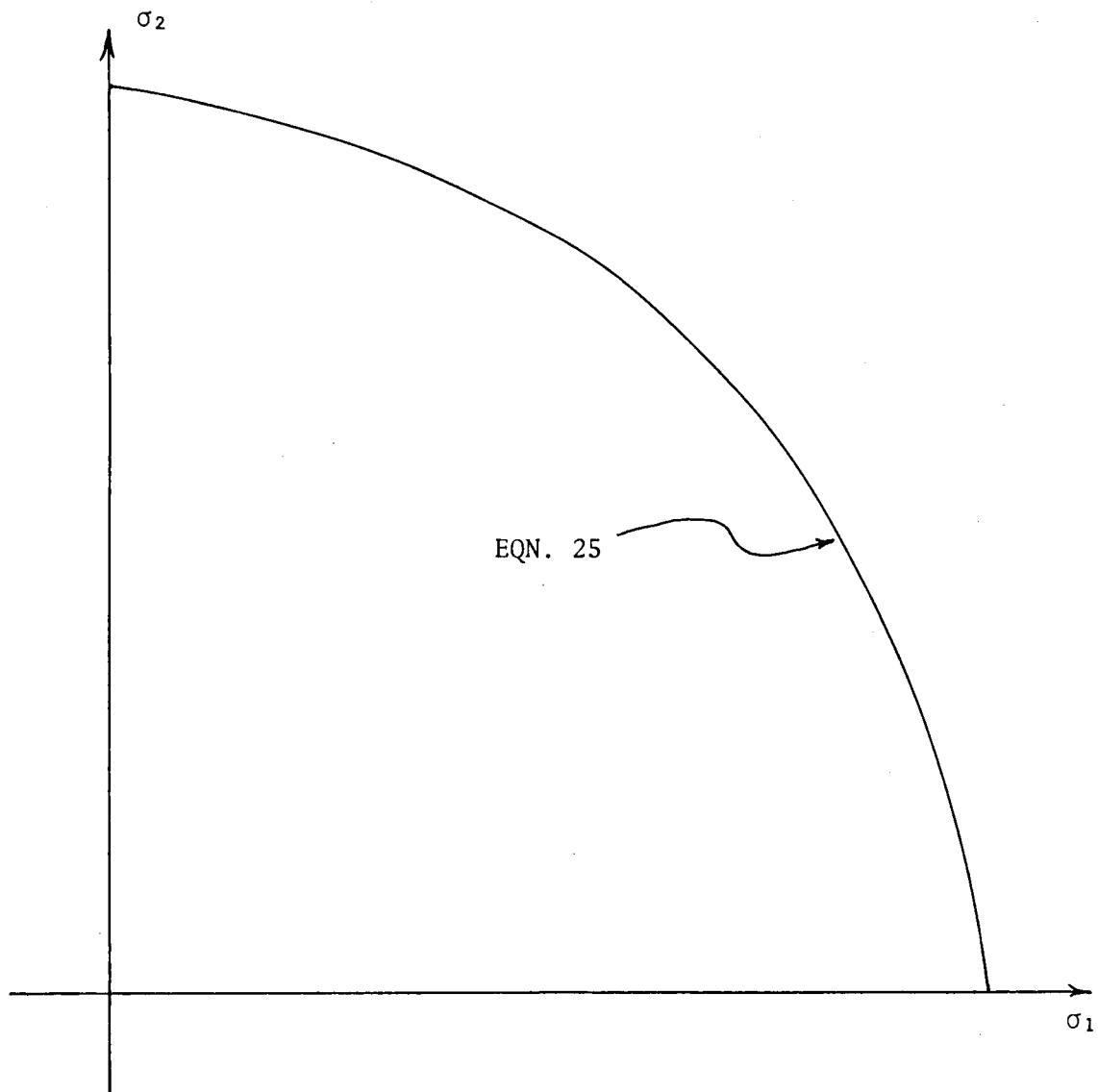


Figure 3. Transition criterion for concrete in biaxial tension

normal to the crack is then set equal to zero and a small amount of shear is introduced along the crack. This can be shown by considering the $x'y'$ coordinate system shown in Figure 4. The stress in this system is related to the strain by

$$(26a) \quad \begin{Bmatrix} \sigma_{x'} \\ \sigma_{y'} \\ \tau_{x'y'} \end{Bmatrix} = \begin{bmatrix} E & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{kE}{2(1+\nu)} \end{bmatrix} \begin{Bmatrix} \epsilon_{x'} \\ \epsilon_{y'} \\ \gamma_{x'y'} \end{Bmatrix}$$

or

$$(26b) \quad \{\sigma'\} = [C'_{CR}] \{\epsilon'\}$$

Stress and strain, being second order tensors, can be referenced to the global $x'y'$ system with the following coordinate transformation

$$(27a) \quad \sigma_{km} = \sigma'_{ji} \beta_{mi}$$

where

β_{ij} = Cosine of the angle between the i th axis of the $x'y'$ system and the j th axis of the $x'y'$ system. For a rotational transformation, $[\beta]$ is given by

$$(27b) \quad [\beta] = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Substituting equation (27b) into equation (27a) yields

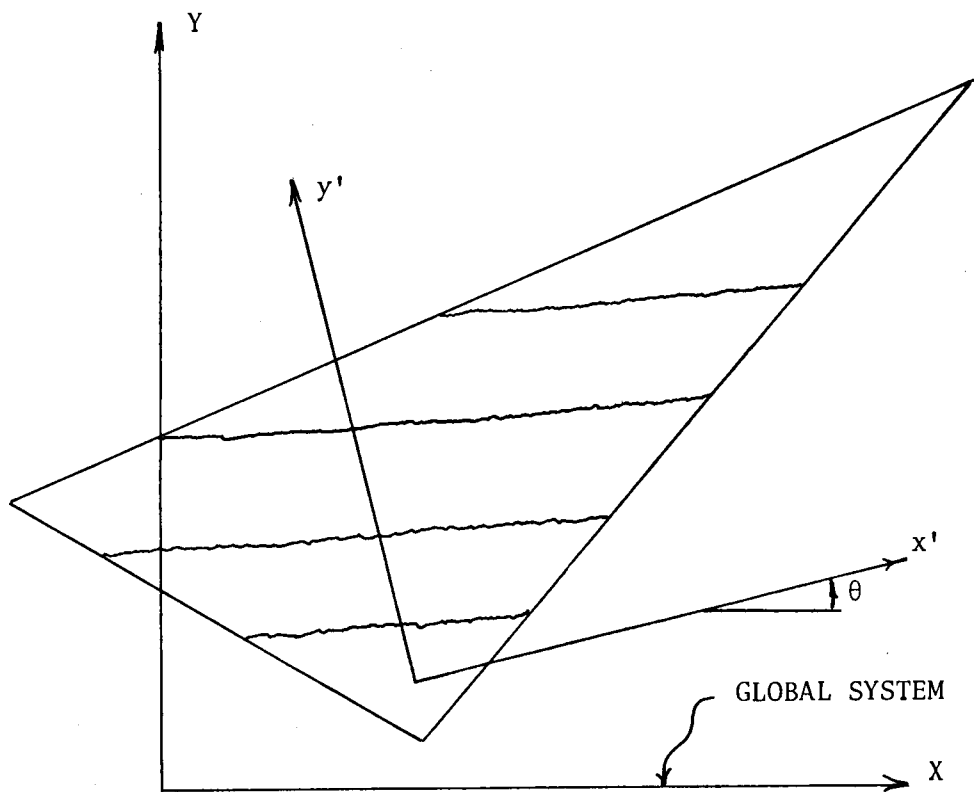


Figure 4. Cracked element coordinate system

$$(28a) \quad \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix} = \begin{bmatrix} \cos^2\theta & \sin^2\theta & -2\sin\theta \cos\theta \\ \sin^2\theta & \cos^2\theta & 2\sin\theta \cos\theta \\ \sin\theta \cos\theta & -\sin\theta \cos\theta & \cos^2\theta - \sin^2\theta \end{bmatrix} \begin{Bmatrix} \sigma_{x'} \\ \sigma_{y'} \\ \tau_{x'y'} \end{Bmatrix}$$

or

$$(28b) \quad \{\sigma\} = [T] \{\sigma'\}$$

Likewise, the strain in the xy system is related to the strain in the x'y' system by

$$(29a) \quad \{\epsilon\} = [T] \{\epsilon'\}$$

or

$$(29b) \quad \{\epsilon'\} = [T]^{-1} \{\epsilon\}$$

However, since rotational transformations are orthogonal,

$$(30) \quad [T]^{-1} = [T]^T$$

Combining (26b), (28b), (29b) and (30) yields

$$(31a) \quad \{\sigma\} = [C_{CR}] \{\epsilon\}$$

where,

$$(31b) \quad [C_{CR}] = [T] [C'_{CR}] [T]^T$$

The factor k (shear retention factor) is introduced in equation (26a) to give the cracked concrete a slightly more realistic representation. k can vary from 0 to 1. If k equals 0, one implies that the cracks are parallel, frictionless surfaces incapable of sustaining shear. On the other hand, a value of k equal to 1 implies that the cracking fails to reduce the capability to sustain shear along the crack. A reasonable representation of the behavior of cracked concrete is obtained by choosing $0 < k < 1$. Hand showed that the actual value chosen does not greatly affect the final results. The value used in this study is $k = 0.4$.

When the stresses first combine to satisfy equation (25), a crack is introduced and the stresses are adjusted as discussed above. The stress is then allowed to increase until equation (25) is again satisfied. At this point, it is assumed that the element has failed and obeys the following constitutive law:

$$(32) \quad \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{Bmatrix}$$

2.3.1.2. BIAXIAL COMPRESSION. We initially assume that the concrete behaves as a linearly elastic, isotropic, homogeneous material with the stress-strain relationship expressed by (1). The stress is allowed to increase in this manner until

$$(33) \quad \tau_{\text{oct}} + \sqrt{2} \frac{\beta-1}{2\beta-1} p - \left(\frac{\sqrt{2}}{3}\right) \frac{\beta}{2\beta-1} f_c = 0$$

where

β = Experimental constant (Ref. 8)

= 1.16

f_c = Transition stress between elastic 1 and elastic 2 for the uniaxial case.

Equation (33) is illustrated in Figure 5.

Hand (Ref. 6) suggests that the above relationship can be used as the transition criterion for a change in the linearly elastic properties. Following this suggestion the stress-strain relation is changed to

$$(34) \quad \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix} = \frac{1}{(1-\nu^2)} \begin{bmatrix} E_d & \nu & 0 \\ \nu & E_d & 0 \\ 0 & 0 & \frac{E_d(1-\nu)}{2} \end{bmatrix} \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{Bmatrix}$$

This represents a bilinear stress-strain relationship shown for the uniaxial case by section OAB of Figure 6.

One the concrete is in the second elastic range, the stress increases until

$$(35) \quad \tau_{\text{oct}} + \sqrt{2} \left(\frac{\beta-1}{2\beta-1}\right) p - \left(\frac{\sqrt{2}}{3}\right) \left(\frac{\beta}{2\beta-1}\right) f_d = 0$$

where,

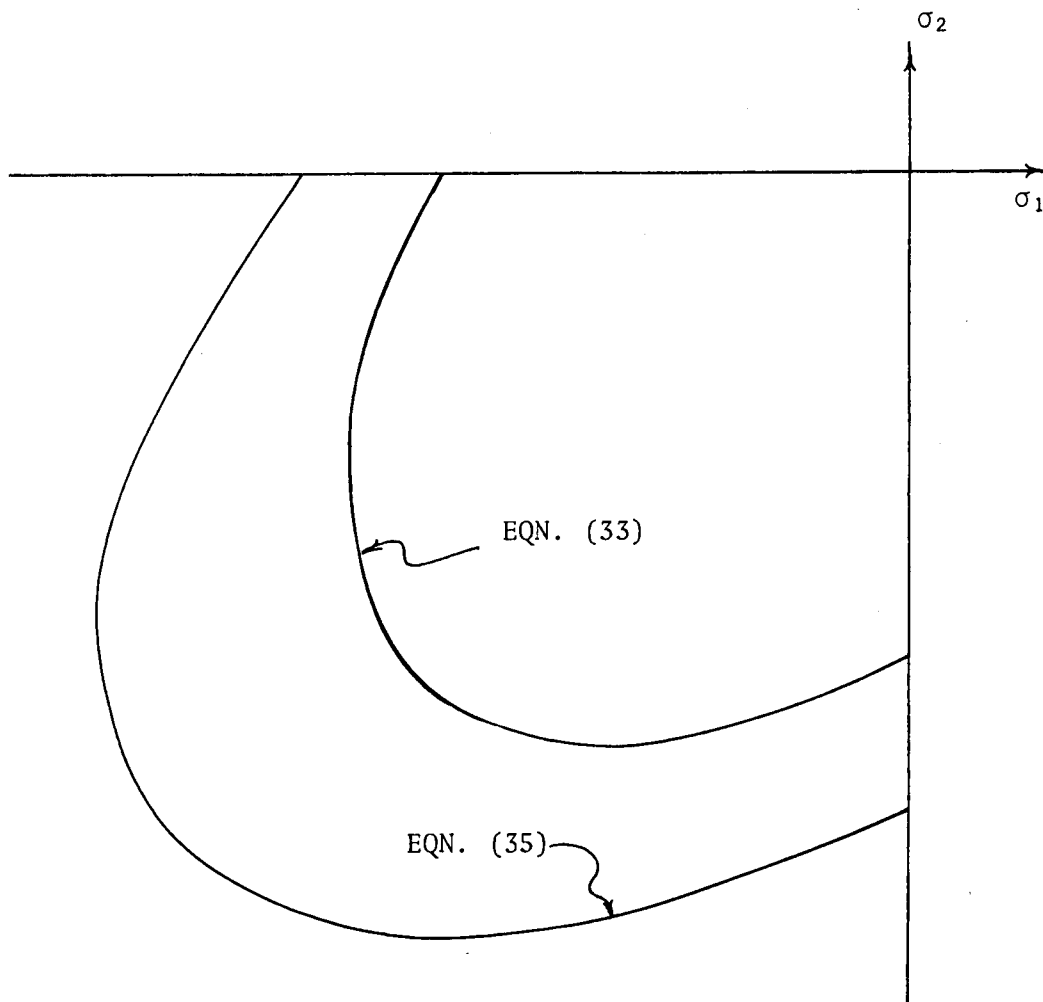


Figure 5. Transition criterion for concrete in biaxial compression

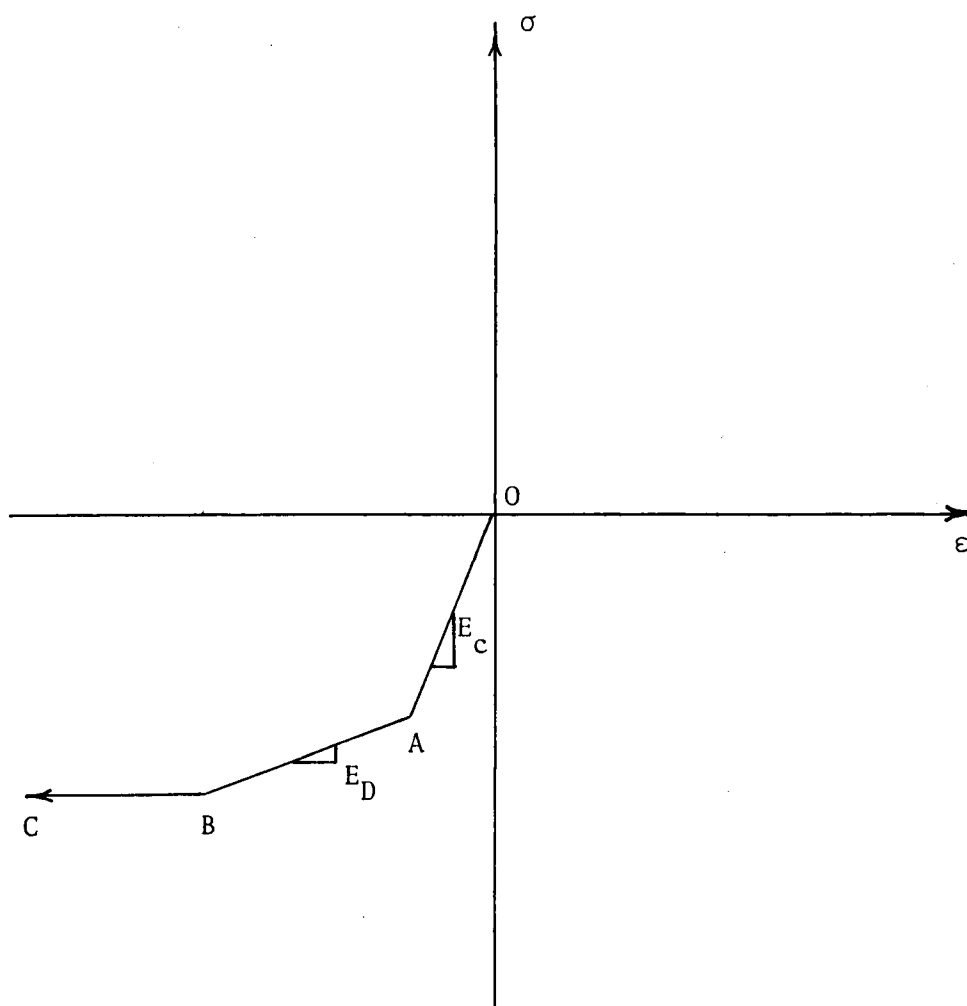


Figure 6. Assumed material behavior for concrete in uniaxial compression

f_d = Plastic transition stress for the uniaxial case.

At this point, the concrete becomes fully plastic. Section BC of Figure 6 illustrates fully plastic behavior for the uniaxial case. Equation (35) is shown in Figure 5.

When considering material properties in a plastic material, one is forced to speak in terms of incremental stress-incremental strain relationships because total stress and total strain can no longer be related. The incremental strain consists of an elastic component and a plastic component.

$$(36) \quad \{\delta\epsilon\} = \{\delta\epsilon\}_E + \{\delta\epsilon\}_P$$

Where δ indicates incremental quantities and the subscripts E and P represent the elastic and plastic components, respectively.

The elastic strain increment can be found by using the elastic constitutive law and the stress increment.

$$(37) \quad \{\delta\epsilon\}_E = [C]^{-1} \{\delta\sigma\}$$

Where $[C]$ is the matrix of material constants expressed in (34).

To find the plastic strain increment, a constitutive relation suggested by von Mises is used:

$$(38) \quad \{\delta\epsilon\}_P = \lambda \frac{\partial Y(\{\sigma\}, \kappa)}{\partial \{\sigma\}}$$

where

λ = some undetermined constant

Y = equation of the yield surface (eqn. 35)

κ = strain hardening parameter (0 for fully plastic behavior)

Equations (37) and (38), when substituted in eqn. (36), yield an expression for the total strain increment:

$$(39) \quad \{\delta\epsilon\} = [C]^{-1} \{\delta\sigma\} + \lambda \frac{\partial Y}{\partial \{\sigma\}} (\{\sigma\}, \kappa)$$

When plastic yielding occurs in a material, the stresses must satisfy the yield criterion:

$$(40) \quad Y (\{\sigma\}, \kappa) = 0$$

That is, the stress values must remain on the yield surface. Mathematically, this requires (for the two-dimensional case) that

$$(41a) \quad \frac{\partial Y}{\partial \sigma_1} \delta\sigma_1 + \frac{\partial Y}{\partial \sigma_2} \delta\sigma_2 + \frac{\partial Y}{\partial \kappa} \delta\kappa = 0$$

or

$$(41b) \quad \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\}^T \{\delta\sigma\} + A\lambda = 0$$

where

$$A = \frac{1}{\lambda} \frac{\partial Y}{\partial \kappa} \delta\kappa$$

Equations (17) and (19b) represent a system of simultaneous equations with 2 unknowns, $\{\delta\sigma\}$ and λ . This system can be written in matrix form as

$$(42a) \quad \begin{Bmatrix} \{\delta\epsilon\} \\ 0 \end{Bmatrix} = \begin{bmatrix} [C]^{-1} & \frac{\partial Y}{\partial \{\sigma\}} \\ \frac{\partial Y}{\partial \{\sigma\}}^T & A \end{bmatrix} \begin{Bmatrix} \{\delta\sigma\} \\ \lambda \end{Bmatrix}$$

Solving this system yields

$$(42b) \quad \lambda = \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\}^T [C] \left[\left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\}^T [C] \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\} - A \right]^{-1} \{\delta\epsilon\}$$

$$(42c) \quad \{\delta\sigma\} = [C] - [C] \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\} \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\}^T [C] \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\} \left[\left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\}^T [C] \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\} - A \right]^{-1} \{\delta\epsilon\}$$

Equation (42c) gives the desired elasto-plastic stress-strain relationship and allows one to relate the total incremental strain to the total incremental stress:

$$(43a) \quad \{\delta\sigma\} = [C]_{EP} \{\delta\epsilon\}$$

where

$$(43b) \quad [C]_{EP} = [C] - [C] \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\} \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\}^T [C] \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\} \left[\left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\}^T [C] \left\{ \frac{\partial Y}{\partial \{\sigma\}} \right\} - A \right]^{-1}$$

and

$$(43c) \quad \frac{\partial Y}{\partial \sigma_x} = \frac{2\sigma_x - \sigma_y}{3\{2\sigma_x^2 + 2\sigma_y^2 - 2\sigma_1\sigma_2\}^{1/2}} + \left(\frac{\sqrt{2}}{3} \right) \left(\frac{\beta-1}{2\beta-1} \right)$$

$$(43d) \quad \frac{\partial Y}{\partial \sigma_x} = \frac{2\sigma_x - \sigma_y}{3\{2\sigma_x^2 + 2\sigma_y^2 - 2\sigma_1\sigma_2\}^{1/2}} + \left(\frac{\sqrt{2}}{3}\right)\left(\frac{\beta-1}{2\beta-1}\right)$$

$$(43e) \quad \frac{\partial Y}{\partial \tau_{xy}} = \frac{1}{\{2\sigma_x^2 + 2\sigma_y^2 - 2\sigma_x\sigma_y + 6\tau_{xy}^2\}^{1/2}}$$

In this study, it is assumed that the concrete becomes fully plastic which means $A = 0$. The matrix $[C]_{EP}$ is formed numerically using equations (43b)-(43e) and the $[C]$ matrix from (34).

Once the concrete becomes plastic it is allowed to deform according to (22a) until

$$(44) \quad 0 = \epsilon_{oct} + \sqrt{2} \left(\frac{\beta-1}{2\beta-1} \right) p_\epsilon - \left(\frac{\sqrt{2}}{3} \right) \frac{\beta}{2\beta-1} \epsilon_u$$

Where,

ϵ_{oct} = Octahedral shear strain

$$= 1/3\{2\epsilon_1^2 + 2\epsilon_2^2 - 2\epsilon_1\epsilon_2\}^{1/2}$$

p_ϵ = Hydrostatic strain

$$= \frac{\epsilon_1 + \epsilon_2}{3} \quad (\text{for plane strain})$$

ϵ_u = Crushing strain for concrete in the uniaxial case.

Equation (44) is illustrated in Figure 7.

When equation (44) is satisfied, the element fails by crumbling and loses all stiffness. This is shown by

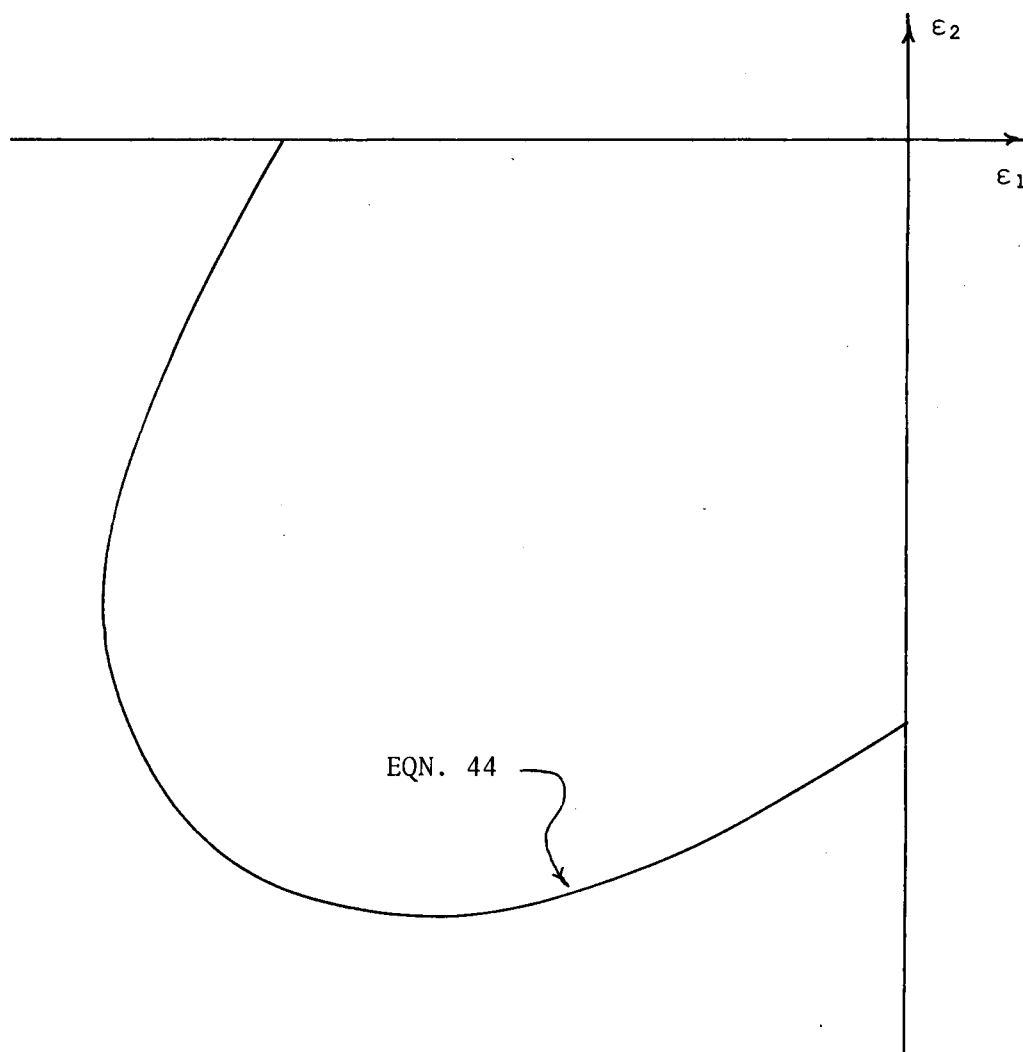


Figure 7. Failure criterion for concrete in biaxial compression

$$(45) \quad \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{Bmatrix}$$

2.3.1.3. TENSION-COMPRESSION. Linear elasticity is once again the initial assumption and the material properties are expressed by eqn. (26a). The failure mode for elements in this stress region is cracking. The failure criterion was chosen so that it would join the points $(f_t, 0)$ and $(0, -f_d)$ in the 2-dimensional stress space. This requirement is suitably satisfied by

$$(46) \quad \left(\frac{\sigma_1}{f_t} \right)^2 - \left(\frac{\sigma_2}{f_d} \right)^2 - 1 = 0$$

This relationship is shown in Figure 8.

Once the stresses in an element combine to satisfy eqn. (46), a crack is assumed to have formed in a manner similar to cracks formed in biaxial tension. The material properties for a single crack in tension-compression are expressed in eqn. (31a).

After the formation of a second crack, if it appears, an element in tension-compression is assumed to have no stiffness and material properties expressed by eqn. (32).

2.3.1.4. COMPLETE TRANSITION SURFACE FOR CONCRETE. The combination of the transition surfaces described in the previous sections will give the complete yield surface assumed for concrete. This surface is illustrated in Figure 9. Examination of this figure shows that it has

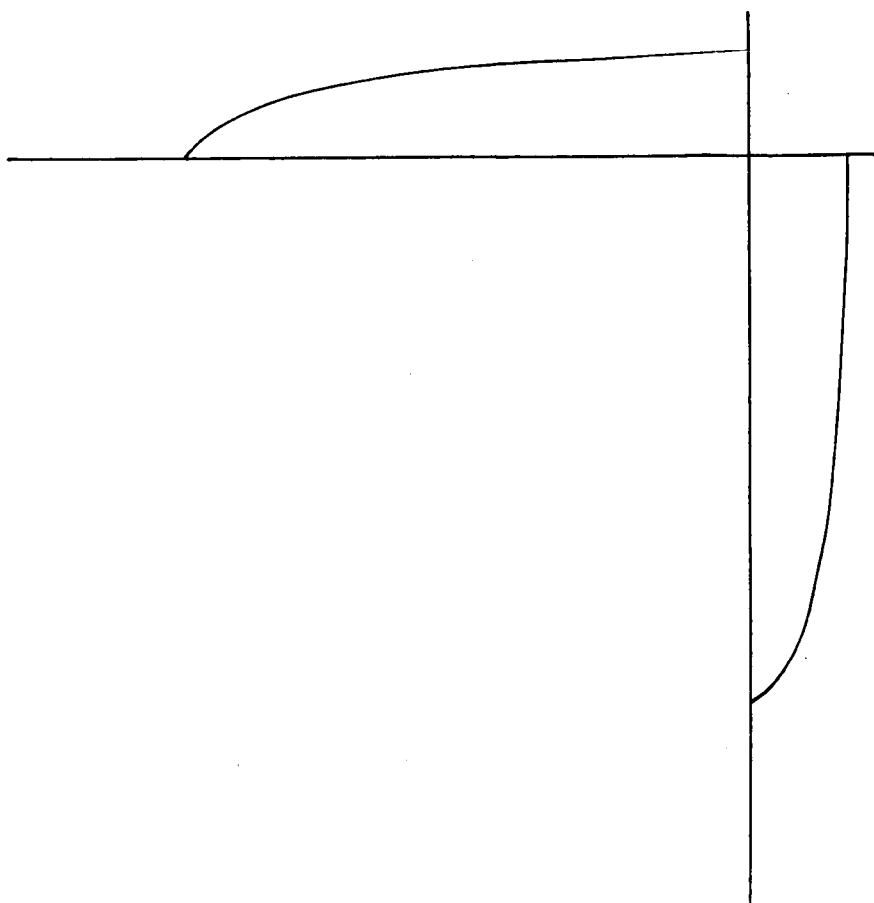


Figure 8. Transition criterion for concrete
in tension-compression

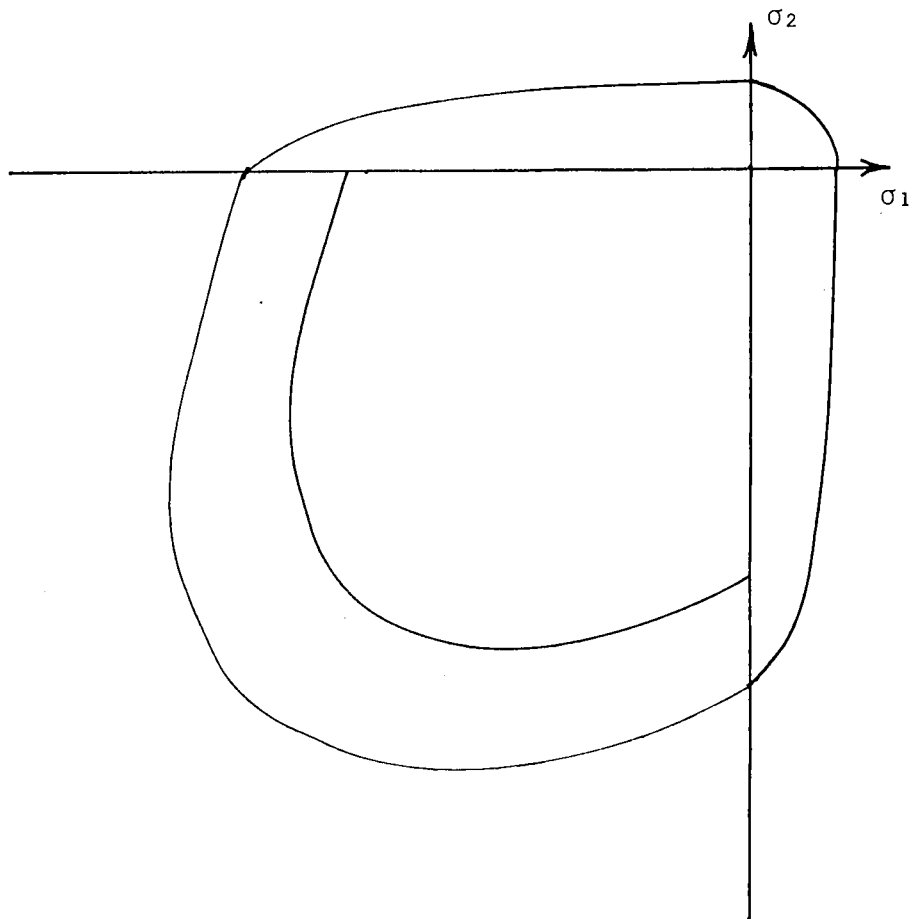


Figure 9. Complete transition surface
for concrete

been assumed that concrete possesses much more strength in compression than it does in tension. Because of this property, concrete is usually strengthened in the tension zone of a structure by adding steel reinforcement rods.

2.3.2. STEEL PROPERTIES

2.3.2.1. TENSION. The steel reinforcing rods are considered to be in a state of uniaxial stress, linearly elastic until a specific stress value, f_y , is obtained. The material properties while in the linear elastic range are

$$(47) \quad \begin{Bmatrix} \sigma_{x'} \\ \sigma_{y'} \\ \tau_{x'y'} \end{Bmatrix} = \begin{bmatrix} E_S & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} \epsilon_{x'} \\ \epsilon_{y'} \\ \gamma_{x'y'} \end{Bmatrix}$$

where the x' axis is along the axis of the reinforcing rod.

After the stress value, f_y , is obtained, the steel is treated as a perfectly plastic material with the incremental stress-strain relationship.

$$(48) \quad \begin{Bmatrix} d\sigma_{x'} \\ d\sigma_{y'} \\ d\tau_{x'y'} \end{Bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} d\epsilon_{x'} \\ d\epsilon_{y'} \\ d\gamma_{x'y'} \end{Bmatrix}$$

2.3.2.2. COMPRESSION. Steel, unlike concrete, behaves in the same manner both in tension and compression. Therefore, it is assumed that

the material properties expressed in (47) are also valid for compression. The complete stress-strain relationship is shown in Figure 10.

2.3.3. COMPOSITE MATERIAL

The material properties of the individual components of reinforced concrete are now defined. The material properties of the composite material are given by

$$(49) \quad [C_{com}] = P_s [C_s] + (1.0 - P_s) [C_c]$$

where

P_s = fraction of layer volume which the steel occupies

$[C_s]$ = steel material properties matrix

$[C_c]$ = concrete material properties matrix.

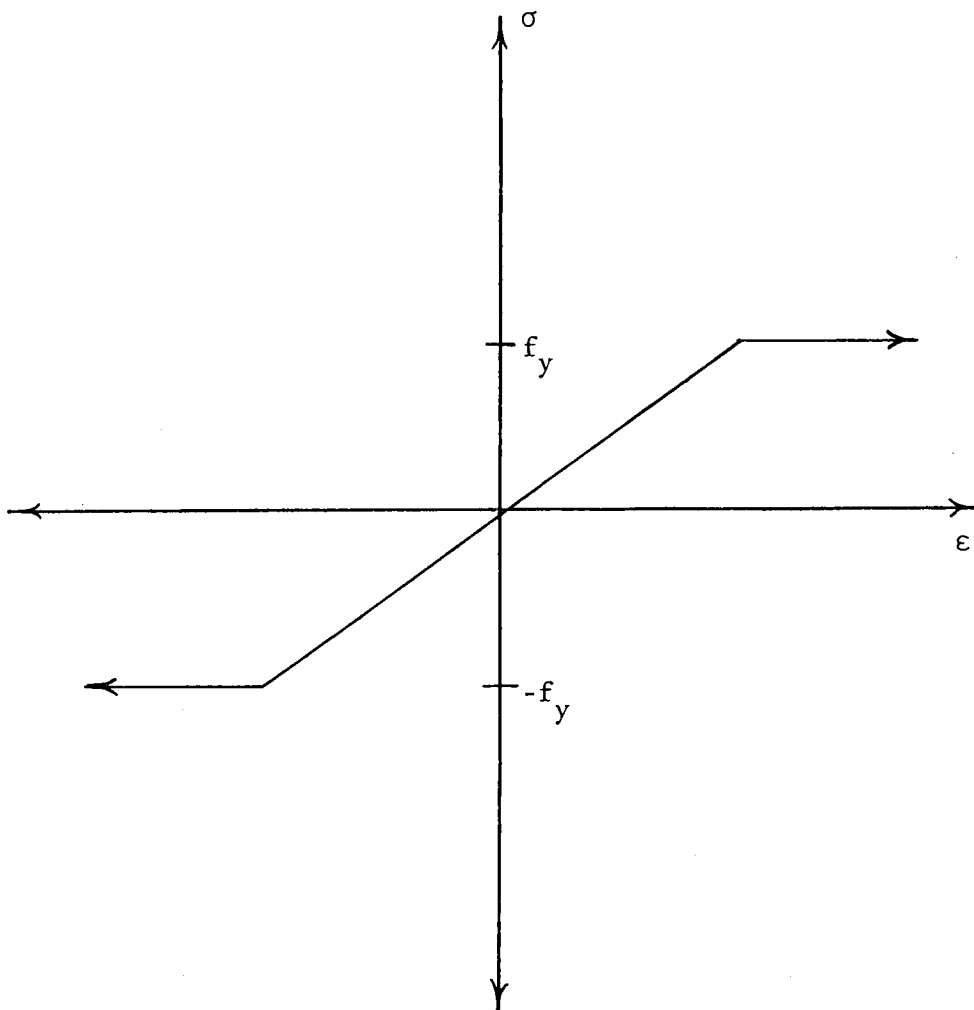


Figure 10. Assumed material behavior of steel reinforcement

3. METHOD OF SOLUTION

3.1. INTRODUCTION

The solution to any solid mechanics problem must satisfy the following conditions:

1. Strain Compatibility: The strain field of any solution must be integrable. That is, the strain components must integrate to give a unique, single valued displacement field, excluding any arbitrary rigid body motion.

2. Equilibrium: The sum of all forces acting on any section of the solution domain must be identically equal to zero.

3. Constitutive Law: The stress and strain must satisfy

$$(50) \quad \{\sigma\} - \{\sigma_u\} - [C] \{ \{\epsilon\} - \{\epsilon_u\} \} = \{0\}$$

where $[C]$ can in general be a function of time, temperature, or stress state.

In the formulation of the displacement method, the first two conditions are inherently satisfied. To satisfy the compatibility requirement, the expression for the strain field was derived from a displacement field which was known to be single valued and continuous throughout the solution domain. The principle of virtual work insures equilibrium and was used to form the final stiffness equations. Therefore, if we can base our nonlinear analysis on the linear formulation, we can be assured that equilibrium and compatibility requirements will be satisfied by any solution of

$$(51) \quad \{\bar{F}\}_{TOT} = \{\bar{F}\}_{CF} + \{\bar{F}\}_{DF} + \{\bar{F}\}_{\sigma} + \{\bar{F}\}_{\epsilon} = [K] \{\bar{\Delta}\}$$

where the terms in the above equation were derived in Section 2.

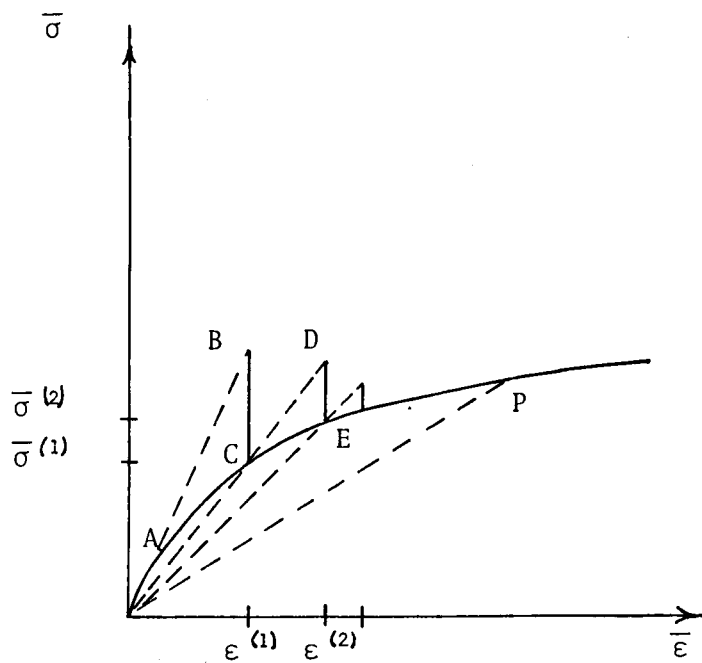
As mentioned before, the nonlinearity to be considered in this project is introduced by a nonlinear constitutive law. This constitutive law has the form expressed by eqn. (50), except $[C]$ is considered to be a function of the stress state only.

It is now clear that the solution to a nonlinear problem must simultaneously satisfy eqns. (50) and (51). This is accomplished by making appropriate adjustments in $[C]$, $\{\sigma_o\}$ or $\{\epsilon_o\}$. Various methods have been formulated based on this idea and several are presented in the next section of this chapter. The algorithm used in this research was based on the adjustment of $\{\sigma_o\}$. A detailed explanation of this method is given in Section 3.2.5.

3.2. SURVEY OF METHODS

3.2.1. SECANT STIFFNESS METHOD

This method obtains a solution to the nonlinear problem by adjusting $[C]$, the matrix of material properties. In general, this method is designed to find a set of material constants $[C_s]$ which yields a linearly elastic equilibrium configuration which also satisfies the constitutive law, eqn. (50). The adjusted material properties are found using an effective stress ($\bar{\sigma}$) vs. effective strain ($\bar{\epsilon}$) curve such as Figure 11.



$$\bar{\sigma} = \frac{1}{3} \{ (\sigma_1 - \sigma_2)^2 + (\sigma_1 - \sigma_3)^2 + (\sigma_2 - \sigma_3)^2 \}^{1/2}$$

$$\bar{\epsilon} = \frac{1}{3} \{ (\epsilon_1 - \epsilon_2)^2 + (\epsilon_1 - \epsilon_3)^2 + (\epsilon_3 - \epsilon_2)^2 \}^{1/2}$$

Figure 11. Secant stiffness method

To use this method, one first applies the entire load using an elastic set of material constants $[C]$. Suppose that some element in the structure loads along line OA of Figure 11. The correct effective stress for the effective strain experienced by this element is $\bar{\sigma}^{(1)}$. One can now calculate a new set of material constants based on a "secant modulus" defined by

$$(52a) \quad E_S = \frac{\bar{\sigma}^{(1)}}{\bar{\epsilon}^{(1)}}$$

The new set of material properties is given by

$$(52b) \quad [C_S^{(1)}] = \left(\frac{E_S}{E} \right) [C]$$

where E is the modulus used in the original $[C]$ matrix.

A new stiffness matrix $[K]$ is now computed with adjustments made for all elements which fail to satisfy the assumed constitutive law. A new linearly elastic solution is then found and a new secant modulus, E_S , is computed for all elements not satisfying the assumed stress-strain relation of eqn. (50). The stiffness matrix is again reformulated based on the new material properties:

$$[C_S^{(2)}] = \left(\frac{E_S}{E} \right) [C]$$

The process is repeated until all elements have stress and strain values which approximately satisfy the assumed constitutive law.

Although this method is easily utilized and generally gives quick convergence, it has several limitations. The main limitation is due to

the fact that instead of loading along the true stress-strain curve (OACEP), a solution is obtained by loading along a linear stress-strain relationship (OP). This makes it impossible to account for any work hardening which may occur in the material. Also, since only the final equilibrium state is obtained, it is implied that the solution is dependent on only the final load applied and is independent of how the stress and strain develop during loading.

3.2.2. INITIAL STRAIN

This method involves an incremental application of the total load. A small portion of the load is applied to the structure and adjustments are made in the solution via the initial strain term of equation (50). When the iterations on a particular load increment become sufficiently close to satisfying the constitutive law, a new load increment is applied to the structure. This is repeated until the entire load has been applied to the structure.

The initial strain is adjusted using the following logic:

Assume that some element in the structure moves from pt. A to pt. B (Figure 12) due to a load increment.

One assumes that the stress state obtained by the solution of eqn. (51) (using the material relations indicated at pt. A) is correct and then adjusts the value of $\{\epsilon\}$ to agree with the assumed constitutive relation. This results in a change of strain from $\{\epsilon^{(1)}\}$ to $\{\epsilon^{(2)}\}$. The difference in these two strains is then treated as an initial strain,

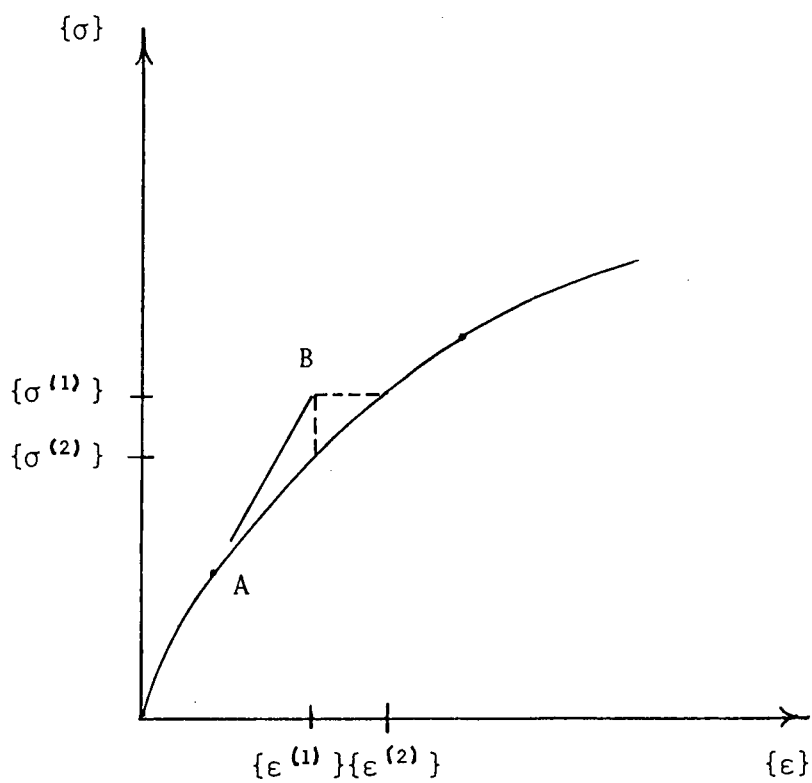


Figure 12. Initial strain and initial stress methods

$$(53) \quad \{\epsilon_o\} = \{\epsilon^{(2)}\} - \{\epsilon^{(1)}\}$$

and is introduced as part of the external load vector in eqn. (51).

The expression for the nodal loads due to an initial strain was derived in Section 2 and is given by

$$(54) \quad \{\bar{F}_\epsilon\} = \iiint [C]\{\epsilon_o\}[Z][B] dx dy [\bar{A}]^{-1} dz$$

Each element having stress and strain values not agreeing with the assumed constitutive law has an initial strain force associated with it. After all initial strain forces have been added to the global force vector, the system (51) is solved again. This process is repeated until all elements satisfy (approximately) the assumed material laws.

The initial strain method has been applied to many types of material behavior and generally gives good results with a few iterations. This method can be used for a plastic material with work hardening, but fails when applied to a perfectly plastic material. This is due to the fact that a perfectly plastic material does not have a unique strain state for values of stress in the plastic range.

3.2.3. INITIAL STRESS

This method is similar to the initial strain method. The total load is applied in increments with the solution to each increment improved until both eqns. (50) and (51) are sufficiently close to being satisfied. The difference between this method and the initial strain method is that we will now compute an excess force based on an initial stress.

To illustrate the method, again consider the element loaded from pt. A to pt. B in Figure 12. Hold the strain $\{\epsilon^{(1)}\}$ constant and find the corresponding stress state $\{\sigma^{(2)}\}$. The difference between the original state of stress and the second state is then treated as an initial stress

$$(55a) \quad \{\sigma_o\} = \{\sigma^{(1)}\} - \{\sigma^{(2)}\}$$

and converted to a force by eqn. (20d) of Section 2

$$(55b) \quad \{\bar{F}_\sigma\} = - \iiint_{vol} \{\sigma_o\}^T [Z] [B] dx dy [\bar{A}]^{-1} dz$$

An initial stress force is calculated in the above manner for each element not satisfying the assumed constitutive law. After all initial stress forces have been calculated, a new set of displacements are calculated. New stress and strain values are computed and the above process is repeated. When the displacements came sufficiently close to satisfying eqns. (50) and (51), a new load increment is applied to the structure.

This method was introduced by Zienkiewicz et al. (Ref. 17) and they report good results with rapid convergence. The main advantage of this method is that it may be applied to the case of a perfectly plastic material since one can now find a unique value for the initial stress term for any stress state.

3.2.4. TANGENT STIFFNESS

This method, like the two previous methods, is incrementally linear. One assumes that a satisfactory solution to the nonlinear

problem can be found by using a series of linear approximations. As before, one iteratively improves the answers for each load increment until a satisfactory solution is found.

In general, this method can be viewed as an improved initial stress method. The similarity between the two methods can be seen in the following description of the procedure:

Assume an element is loaded from pt. A to pt. B in Figure 13. An initial stress force is computed for the element as described above and it is added to the global force vector $\{\bar{F}\}$. In addition to this residual force, one now adjusts the stiffness of this element by using the material properties given by the assumed constitutive law for the pt. B'. This means that when one evaluates the displacements due to the residual force, the stress increases tangent to the stress-strain curve.

An initial stress force and a new set of material constants are calculated for each element not satisfying eqn. (50). When the global residual force vector is complete, one calculates additional displacements and repeats the above process until some convergence criterion is satisfied.

This method gives good results with very few iterations and is applicable to material nonlinearities of all types. In particular, it is used in this project to analyze perfectly plastic reinforced concrete.

A detailed description of this method is given in the next section.

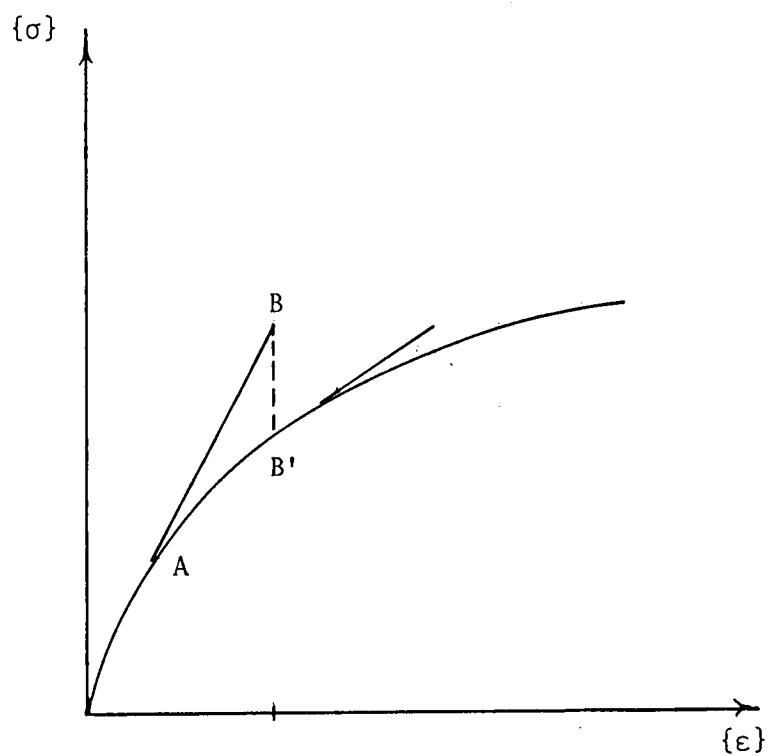


Figure 13. Tangent stiffness iteration

3.2.5. THE PROPOSED METHOD OF SOLUTION

A computer code was developed to perform the nonlinear analysis using the tangent stiffness method. The program was written in FORTRAN and given the name "CRACK." (See Appendix B for a listing.) The program consists of the following sequence of steps for an arbitrary load increment "i":

1. A set of incremental displacements are calculated from

$$(56) \quad \{\bar{F}\}_i = [K]_i \{\bar{\Delta}\}_i$$

where, the subscript i indicates quantities valid for the ith increment only. $[K]$ was assembled using quantities indicated by the final stress-state of the previous increment.

The incremental displacements $\{\bar{\Delta}\}_i$ are then added to the total displacements $\{\bar{\Delta}\}$.

2. A set of incremental stress and strain values are calculated using

$$(57a) \quad \{\epsilon(x_c, y_c, Z)\}_i = [Z] [B_{(x_c, y_c, Z)}] [\bar{A}]^{-1} \{\bar{\Delta}\}_i$$

and

$$(57b) \quad \{\sigma(x_c, y_c, Z)\}_i = [C] \{\epsilon(x_c, y_c, Z)\}_i$$

where, the subscript C indicates the centroidal coordinates. The incremental values $\{\sigma\}_i$, $\{\epsilon\}_i$ are then added to the total values $\{\sigma\}$, $\{\epsilon\}$ for the element.

3. Transition criteria are checked for each layer using the total stress states calculated in step 2. The transition criteria are outlined in Section 2.

4. If there are no elements with layers which have exceeded transition criteria add 1 to i and return to step 1.

5. Calculate the initial stress term for each layer exceeding a transition criterion:

$$\{\sigma_o\} = \{\sigma^{(2)}\} - \{\sigma^{(1)}\}$$

This initial stress is converted to a force using

$$\{\bar{F}_\sigma\} = \iiint \{\sigma_o\}^T [Z] [B] dx dy dz [\bar{A}]^{-1}$$

Note: In the above expression it must be recognized that $\{\sigma_o\}$ is considered to be constant through the thickness of the layer it is calculated for and zero elsewhere.

After an initial stress force is calculated for all the layers in an element which have exceeded a transition criterion, an element force vector is assembled and added to the global force vector.

6. Each layer exceeding a transition criterion changes material properties as explained in Section 2 and the element stiffness is adjusted. Instead of reformulating the entire global stiffness matrix, an element stiffness matrix based on the difference in properties is formed and added to the global matrix.

7. Steps 5 and 6 are repeated until all elements with layers changing properties have been adjusted.

8. Check for convergence. If the ℓ_2 norm (the square root of the sum of the squares of all components) of the initial stress force is less than 10% of the ℓ_2 norm of the i th incremental force, the solution is satisfactory and a new increment is applied. If the solution is not satisfactory, repeat steps 1 through 7 using the initial stress force as the load.

3.3. CALCULATION OF THE INITIAL STRESS

The two previous sections have outlined the method of analysis. The two major steps in this method are:

1. Changing the element stiffness due to a change of material properties.

2. Calculation of an initial stress (residual) force.

The element stiffness, its dependence on material properties and the material properties themselves were defined in Section 2. The purpose of this section is to explain in detail how the initial stress is calculated for each material transition.

3.3.1. PRIMARY CRACKING

Suppose a layer of some element loads from pt. A to pt. B in Figure 14. As explained in Section 3.2, one holds the strain state constant at $\{\epsilon^{(1)}\}$ and finds the corresponding stress state $\{\sigma^{(2)}\}$, which satisfies the constitutive law. This stress may be expressed as

$$(58a) \quad \{\sigma^{(2)}\} = [C_1]\{\epsilon_t\} + [C_2]\{\{\epsilon^{(1)}\} - \{\epsilon_t\}\}$$

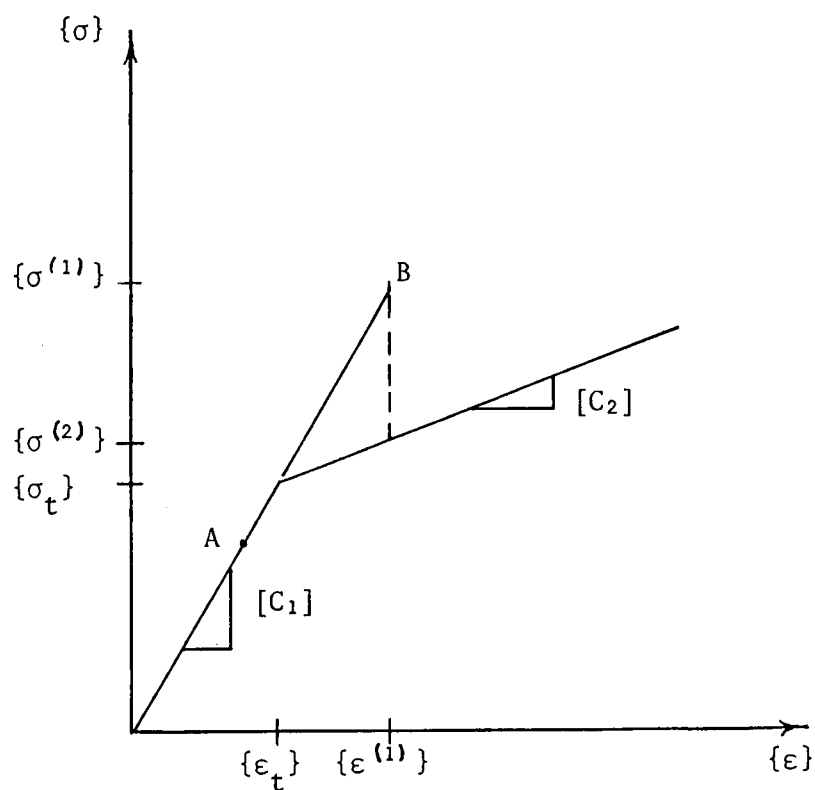


Figure 14. Calculation of the correct stress state for a bilinearly elastic material

But, one can also say

$$(58b) \quad \{\epsilon_t\} = [C_1]^{-1} \{\sigma_t\}$$

Substituting (58b) into (58a) gives

$$(58c) \quad \{\sigma^{(2)}\} = [C_1][C_1]^{-1} \{\sigma_t\} + [C_2]\{\epsilon^{(1)}\} - [C_2][C_1]^{-1} \{\sigma_t\}$$

or

$$(58d) \quad \{\sigma^{(2)}\} = [C_2]\{\epsilon^{(1)}\} + [[I] - [C_1]^{-1}[C_2]] \{\sigma_t\}$$

The initial stress is then given by

$$(59) \quad \{\sigma_0\} = \{\sigma^{(2)}\} - \{\sigma^{(1)}\}$$

In order to evaluate (9d) one must know $\{\sigma_t\}$. This may be found by solving for the intersection of the radial line OAB and the yield surface (see Figure 15). For biaxial tension, the yield surface is given by eqn. (2) of Section 2 and the point of intersection, B', is given by

$$(60a) \quad \sigma_1 = \frac{2\sqrt{2}\alpha}{3(1+\alpha)} f_d \left\{ \frac{1}{\frac{\sqrt{2}}{3} \left(\frac{1-\alpha}{1+\alpha}\right)^{(m+1)} + \frac{\sqrt{2}}{3} \{m^2-m+1\}^{1/2}} \right\}$$

$$(60b) \quad \sigma_2 = m\sigma_1$$

where

$$M = \frac{\sigma_2^{(1)}}{\sigma_1^{(1)}}$$

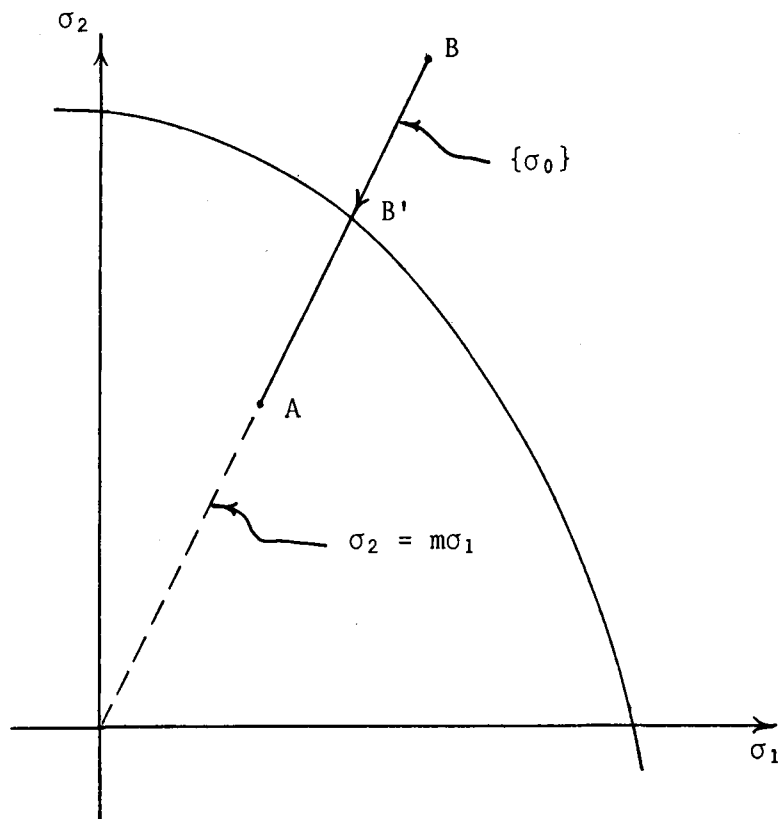


Figure 15. Calculation of the transition stress state for concrete exceeding biaxial tension transition criterion

For primary cracking in tension-compression, the yield surface is given by eqn. (15) of Section 2 and the transition stress $\{\sigma_t\}$ is given by

$$(61a) \quad \sigma_1 = \left\{ \left(\frac{1}{f_t} \right)^2 + \left(\frac{m}{f_d} \right)^2 \right\}^{-1/2}$$

$$(61b) \quad \sigma_2 = m\sigma_1$$

3.3.2. SECONDARY CRACKING

As explained in Section 2, it is assumed that failure occurs when secondary cracks appear in an element. Therefore, one can express the correct stress state for any element with secondary cracking as

$$(62a) \quad \{\sigma^{(2)}\} = \{0\}$$

and the initial stress as:

$$(62b) \quad \{\sigma_0\} = \{\sigma^{(2)}\} - \{\sigma^{(1)}\} = -\{\sigma^{(1)}\}$$

3.3.3. ELASTIC 1 TO ELASTIC 2

This situation is similar to the case of primary cracking. We first find the correct stress using eqn. (58d) with $[C_1]$ and $[C_2]$ given by (1) and (12) of Section 2. The transition stress is found in the same manner and the coordinates for this point are

$$(63a) \quad \sigma_1 = \frac{\sqrt{2} \beta f_c}{3(2\beta-1)} \left\{ \frac{1}{\frac{\sqrt{2}}{3} \left(\frac{\beta-1}{2\beta-1} \right) (m+1) - \frac{\sqrt{2}}{3} (m^2-m+1)^{1/2}} \right\}$$

$$(63b) \quad \sigma_2 = m\sigma_1$$

This initial stress is given by

$$(64) \quad \{\sigma_o\} = \{\sigma^{(2)}\} - \{\sigma^{(1)}\}$$

3.3.4. ELASTIC 2 TO PLASTIC

Suppose a layer passing from the elastic region to the plastic region moves from pt. A to pt. B in Figure 16. In this load step, the element has undergone an elastic and a plastic strain. The first step in finding the initial stress is to correct the current stress values to allow for the plastic strain. This can be done by finding the amount of stress, $\{\Delta\sigma_E\}$, which accumulated after the plastic transition and, using the elastic 2 properties, convert it to plastic strain, $\{\epsilon_p\}$. By using the elasto-plastic material properties (eqn. 43 of Section 2) evaluated at B', the plastic strain can be converted to plastic stress, $\{\Delta\sigma_p\}$. The net effect is that the plastic stress increases in a direction tangent to the curve at B'.

The point of transition, B', is found in a manner similar to the method used in primary cracking. The coordinates of this point are:

$$(65a) \quad \sigma_1'' = \frac{\sqrt{2} \beta f_d}{3(2\beta-1)} \left\{ \frac{1}{\frac{\sqrt{2}}{3} \left(\frac{\beta-1}{2\beta-1}\right)^{(m+1)} - \frac{\sqrt{2}}{3} (m-m+1)^{1/2}} \right\}$$

$$(65b) \quad \sigma_2'' = m\sigma_1''$$

So, the quantity, $\{\Delta\sigma\}$, mentioned above is given by,

$$(66a) \quad \Delta\sigma_{E1} = \sigma_1'' - \sigma_1'$$

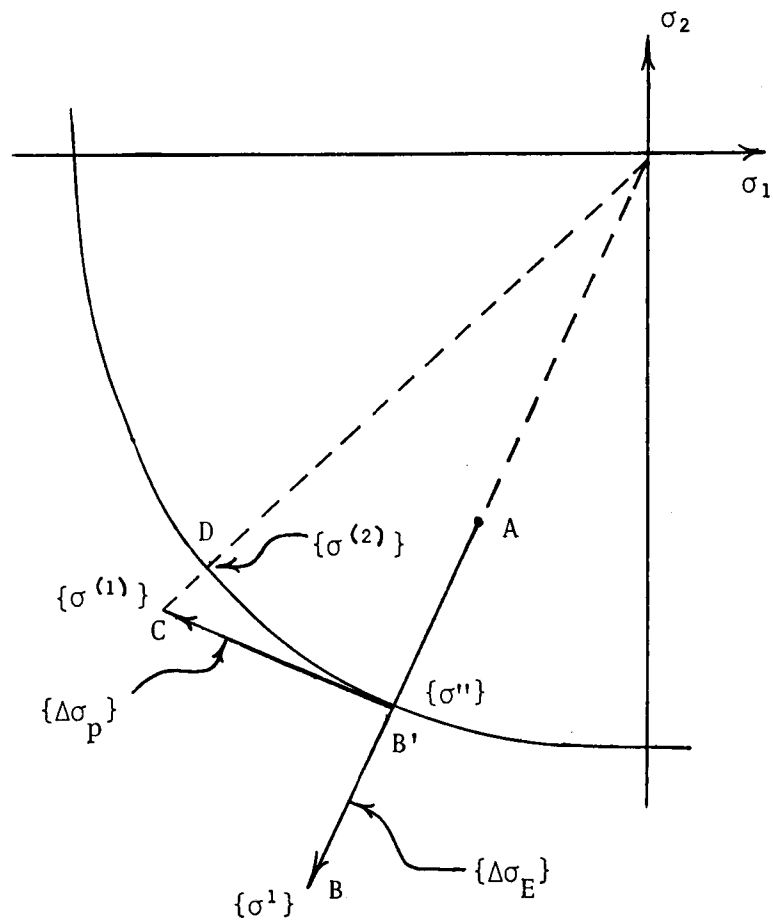


Figure 16. Calculation of initial stress for concrete undergoing elastic to plastic transition

$$(66b) \quad \Delta\sigma_{E2} = \sigma_2'' - \sigma_2'$$

The plastic strain can be found using the linear elastic 2 stress-strain relationship

$$(67) \quad \{\epsilon_p\} = [C_2]^{-1} \{\Delta\sigma\}$$

where $[C_2]$ is given by (12) of Section 2.

The approximate plastic stress is then given by

$$(68) \quad \{\sigma_p\} = [C_{EP}]\{\epsilon_p\}$$

where $[C_{EP}]$ is derived in Section 2 and given by Eqn. (43b).

When this plastic stress is added to the transition stress, the total stress state will only approximately satisfy the assumed constitutive law, as shown by pt. C in Figure 16. The true stress state, $\{\sigma^{(2)}\}$, is found by locating the point where the radial line OC intersects the yield surface. The analytical expression for this point was given by eqns. (65a) and (65b) where M is now given by

$$M = \frac{\sigma_1^{(1)}}{\sigma_2^{(1)}}$$

Once this point is known, the initial stress is found by

$$\{\sigma_o\} = \{\sigma^{(2)}\} - \{\sigma^{(1)}\}$$

3.3.5. PLASTIC

After a layer has become plastic, one approximates its behavior by loading it tangent to the yield surface. This is accomplished by using the elasto-plastic material properties derived in Section 2. The difference between the stress state obtained by the tangent loading (pt. C of Figure 16) and the true stress-strain (pt. D of Figure 16) is the initial stress. The only difference between plastic material and material the elastic-plastic transition is that for the plastic case we know that since we have assumed ideal plasticity, the full strain increment is plastic. Therefore, the strain does not have to be separated into plastic and elastic components as it was for material making the elastic-plastic transition.

4. EXAMPLES

4.1. INTRODUCTION

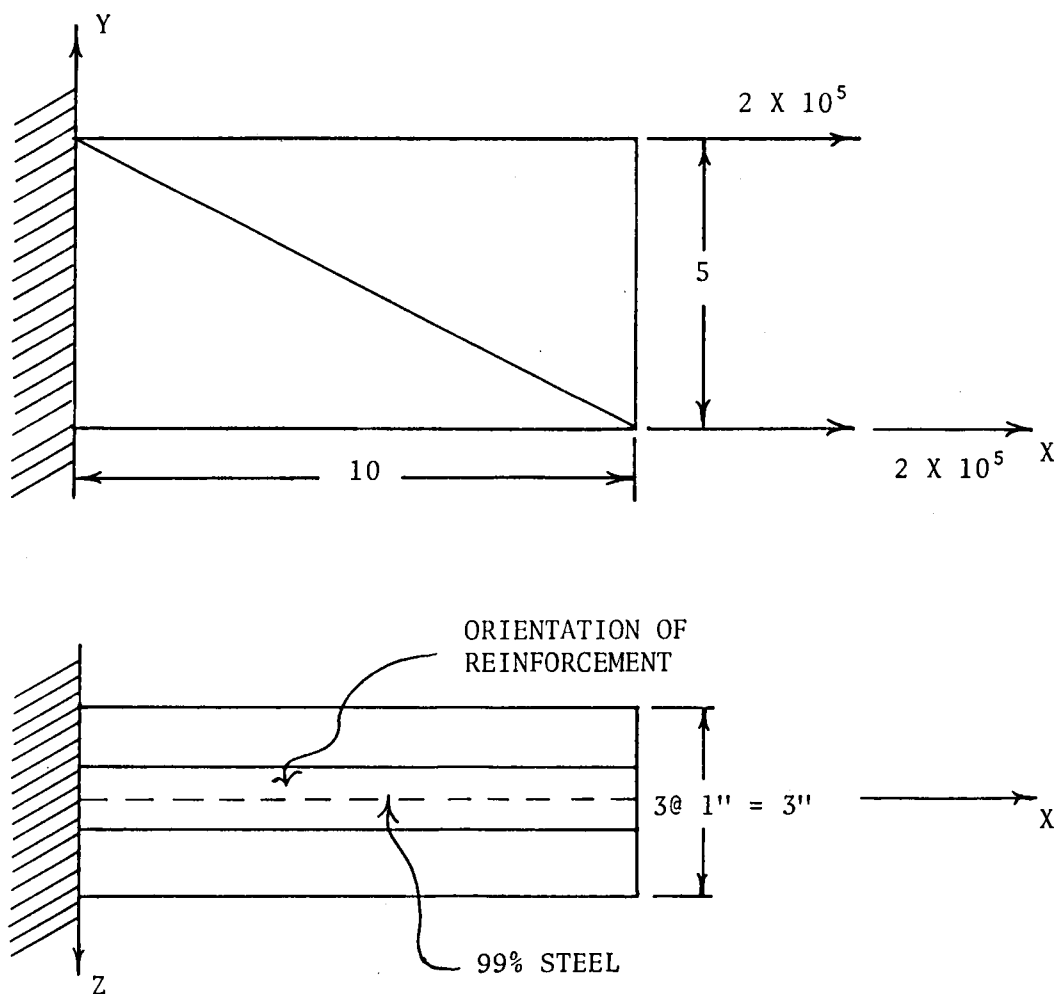
The purpose of this chapter is to show that the algorithm developed can model the nonlinear material behavior of reinforced concrete. For the purpose of demonstration and verification, several example problems are discussed. The example problems selected were chosen to give clear, simple demonstrations of the algorithm.

4.2. EXAMPLE I

The first problem to be considered is shown in Figure 17 and demonstrates the steel reinforcement and concrete cracking capabilities of the program. For comparison purposes, the theoretical solution is also presented.

The load deflection curves for the two methods of solution are shown in Figure 18. The load deflection curve can be separated into two distinct regions. The first represents the linear response of the structure up to the ultimate tensile strength of the concrete. After the tensile limit of the concrete is reached, a sudden increase in deflection occurs because the steel reinforcement is forced to carry the full load since the concrete has now lost all strength in the direction of the load.

The predictions of the computer code agree very well with the theoretical solution up to the concrete tensile limit. As the load is increased after this point, the deflection predicted by the computer



MATERIAL PROPERTIES

$$E_c = 3.5 \times 10^6 \text{ psi}$$

$$f_c = 4800 \text{ psi}$$

$$E_D = 0.88 \times 10^6 \text{ psi}$$

$$\epsilon_u = 3 \times 10^{-3} \text{ psi}$$

$$\nu = 0.0$$

$$E_s = 30 \times 10^6 \text{ psi}$$

$$f_t = 546 \text{ psi}$$

$$\theta = 90.0^\circ$$

$$f_d = 3600 \text{ psi}$$

$$f_y = 45,000 \text{ psi}$$

Figure 17. Example I

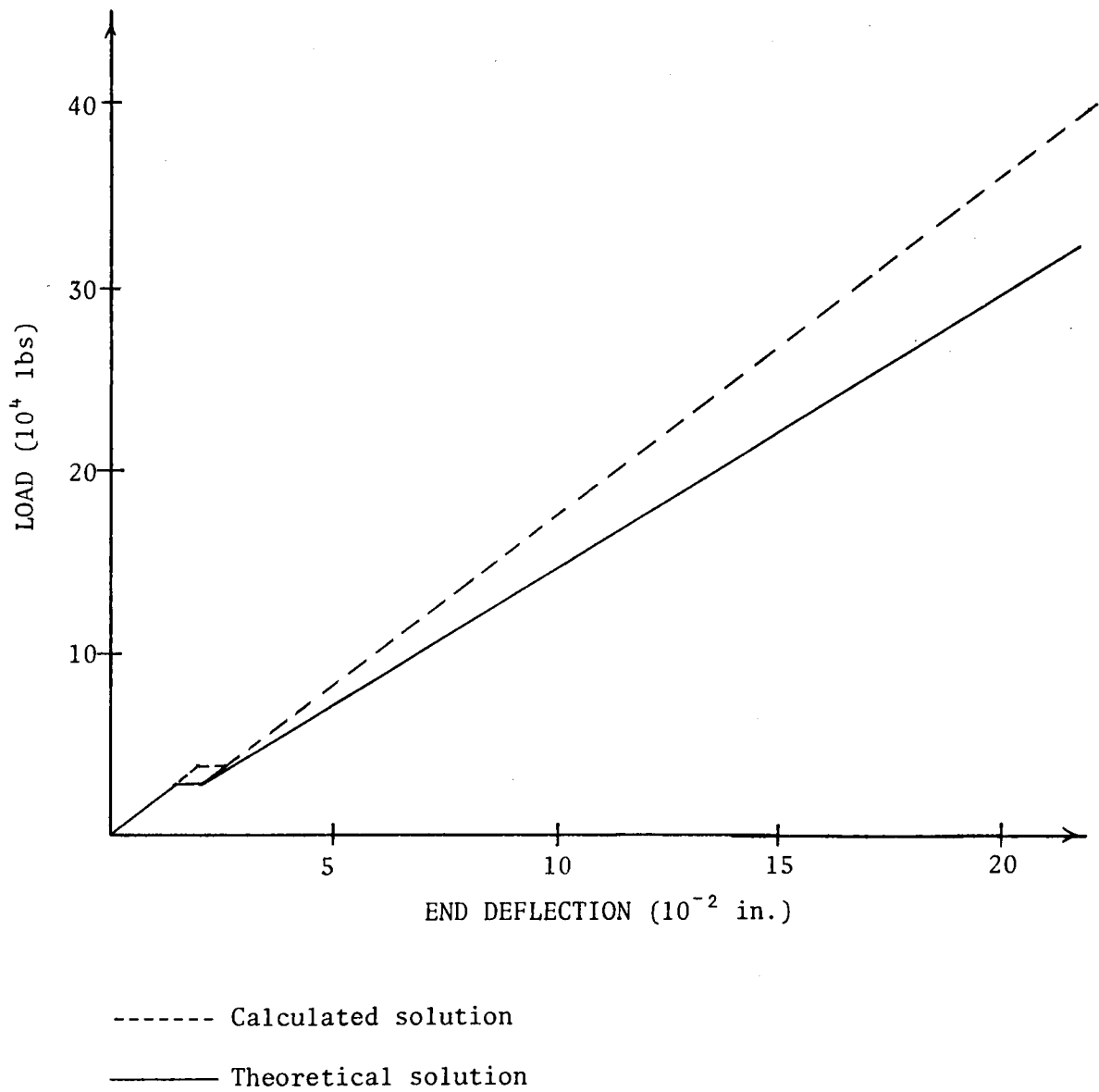


Figure 18. Load deflection curve
for Example I

program becomes more and more inaccurate. This is due to the fact that when cracking occurs, the algorithm outlined in this study states that the concrete stiffness must be adjusted. The expression for the stiffness of cracked concrete is given by eqn. (31b). Inspection of the terms in this equation shows that the adjusted concrete stiffness is heavily dependent on the principal angle θ , and therefore, also dependent on the stress values obtained from the code. It is well known that stresses obtained from the displacement method are generally less accurate than the deflections. Therefore, it must be expected that any inaccuracy will be greatly magnified by the multiplication in eqn. (31b). Despite this difficulty, the code seems to give fairly good results for loads not much greater than the tensile limit of the concrete.

The results of Example I are summarized in Table I.

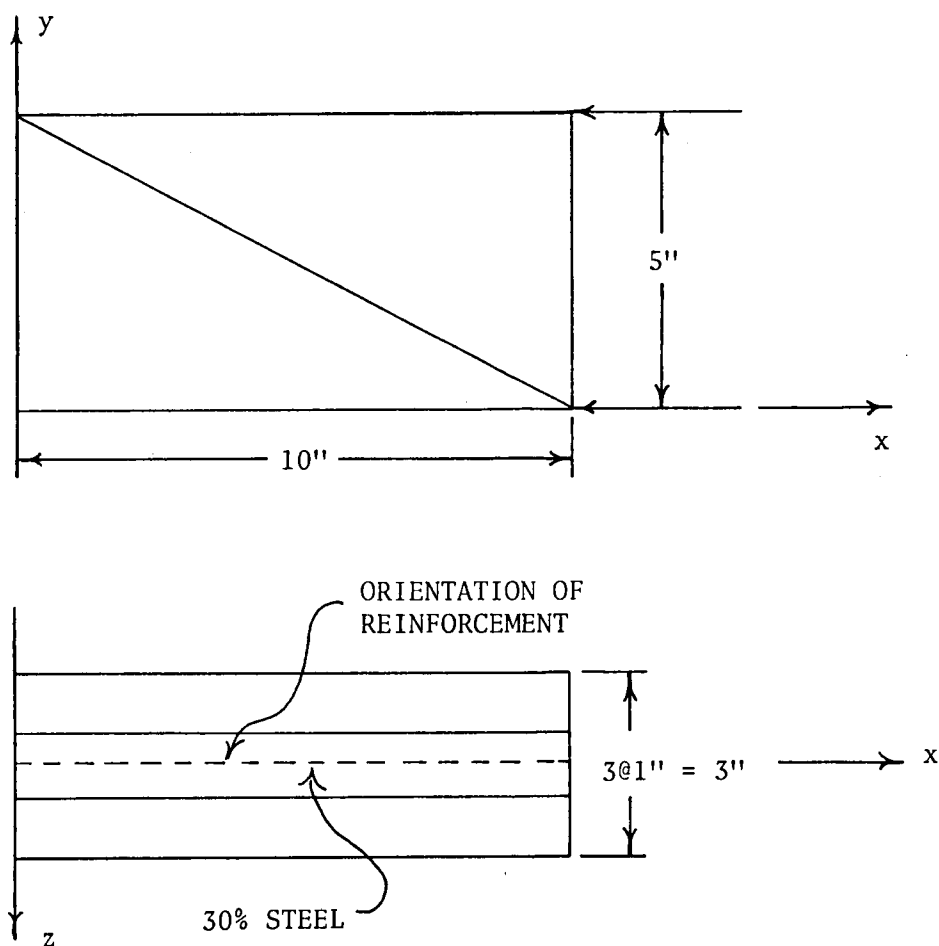
4.3. EXAMPLE II

To demonstrate the computations performed in the biaxial compression zone the problem shown in Figure 19 was chosen as the second example. This problem forces the program to perform the calculations described for the bilinear elastic and plastic behavior of the concrete.

The load deflection curve predicted by the computer code is compared to the theoretical solution in Figure 20. This figure indicates that the code does not effectively model the structure when the loads cause the structure to leave the first linear region. In fact, examination of the results given in Table II show that the computer code predicts a completely linear response. This is obviously an

TABLE I. SUMMARY OF RESULTS: EXAMPLE I

APPLIED LOAD (lb)	DEFLECTION (in)		CONCRETE STRESS (PSI)		STEEL STRESS (PSI)	
	CALC.	S.O.M.	CALC.	S.O.M.	CALC.	S.O.M.
2×10^4	1.09×10^{-3}	1.09×10^{-3}	381	381	3,270	3,266
4×10^4	2.59×10^{-3}	2.70×10^{-3}	0.0	0.0	7,780	8,081
10×10^4	5.86×10^{-3}	6.73×10^{-3}	0.0	0.0	17,600	20,202
16×10^4	9.13×10^{-3}	1.08×10^{-2}	0.0	0.0	27,400	32,323
22×10^4	1.24×10^{-2}	1.48×10^{-2}	0.0	0.0	37,200	44,444
28×10^4	1.57×10^{-2}	1.89×10^{-2}	0.0	0.0	YIELDED	68,686
34×10^4	FAILED	FAILED	-	-	-	-
40×10^4	-	-	-	-	-	-



MATERIAL PROPERTIES

$$E_c = 3.5 \times 10^6 \text{ psi}$$

$$E_d = 0.88 \times 10^6 \text{ psi}$$

$$\nu = 0.0$$

$$f_t = 546 \text{ psi}$$

$$f_d = 3,000 \text{ psi}$$

$$f_c = 3,800 \text{ psi}$$

$$\epsilon_u = 3 \times 10^{-3}$$

$$E_s = 30 \times 10^6 \text{ psi}$$

$$f_y = 65,000 \text{ psi}$$

Figure 19. Example II

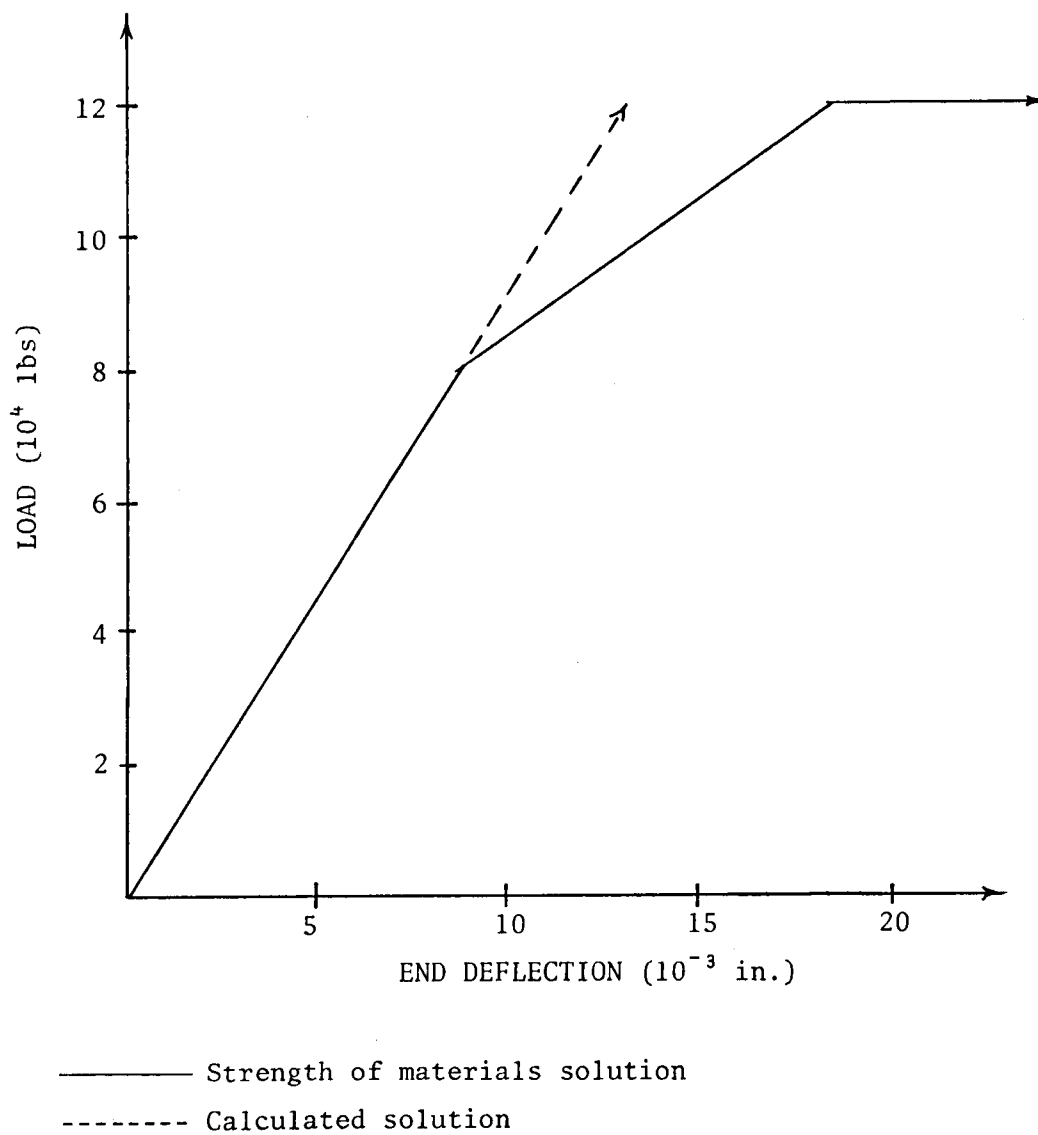


Figure 20. Load deflection curves
for Example II

TABLE II. SUMMARY OF RESULTS: EXAMPLE II

APPLIED LOAD (lbs)	DEFLECTION (10^{-3} in)		CONCRETE STRESS (PSI)		STEEL STRESS (PSI)	
	CALC.	S.O.M	CALC.	S.O.M	CALC.	S.O.M.
4 4 4 X 10^4	4.34	4.336	1,520	1,518	13,000	13,010
8 X 10^4	8.67	8.793	3,010	3,020	26,000	26,380
10 X 10^4	10.8	13.57	3,200	3,440	32,500	40,710
12 X 10^4	13.0	18.1	3,390	FAILED	39,100	54,270
14 X 10^4	15.2	-	3,580	-	45,600	FAILED
	15.7	-	3,630	-	47,200	-
	16.3	-	3,680	-	48,900	-
	16.8	-	3,730	-	50,500	-
	17.4	-	3,770	-	52,100	-

error in the code itself since the algorithm developed earlier would definitely cause the code to predict a nonlinear response.

In conclusion, it must be stated that the results of the sample problems indicate that the computer code is not completely functional. The results of Example I indicate that the code gives reasonable results for simple uniaxial tension, but Example II indicates that the code does not function for uniaxial compression. All attempts to obtain results for more complicated examples failed to produce reasonable answers. Therefore, it must be concluded that much more effort is needed to develop and verify a computer code which utilizes the nonlinear algorithm outlined in this study.

5. RECOMMENDATIONS

5.1. INTRODUCTION

Although the code developed in this study is not functional at the present time, a great deal has been accomplished and learned. The theoretical basis has been fully explored and an algorithm has been proposed. In formulating the algorithm, a major portion of the ground-work has been completed. The writing, debugging and verification of code is by no means an easy task, but it is a much different task than forming the theoretical framework of any numerical method. Even though much has been accomplished, there are additional tasks which should be done to fulfill the original scope of this project. Also, during the course of this study, many questions arose which could be the subject of further research. The purpose of this chapter is to propose and discuss recommendations for further work and research.

5.2. RECOMMENDATIONS FOR COMPLETING AND IMPROVING THE CODE

The development of a code to carry out the algorithm presented is a project in itself. The organization and programming techniques necessary require a deep knowledge of computer programming. In this respect, this study has been a very educational experience for the author. During the course of this project, much insight has been gained concerning the development of a large computer code. Based on this experience, the following recommendations concerning the completion of the code are made:

1. The program should be subdivided into a larger number of subroutines. At the present time, there are only 11 subroutines in the code, some of which are quite long and complex. This makes debugging a very difficult procedure since each subroutine must perform many functions. A more suitable arrangement would be one in which each subroutine performed only a small number of very well defined tasks. This would make it much easier to find the source of errors.

2. A simpler element, such as the constant strain triangle should be used to develop the code. This would aid the development process by removing the bending action and moments present in the element currently being used. Once the subroutines and calling sequences have been determined and verified, the addition of a suitable plate element and the layering scheme would be a rather simple matter.

3. The present yield surface is too complex for development purposes. A simpler yield surface, such as the Tresca yield surface, could be used to make the code easier to debug. The performance of the algorithm is not dependent on the actual yield surface used in the model, so it seems reasonable that the first step should be to use a simple yield surface. After the program has been completed for a simple yield surface, the yield surface currently in the model could be exchanged for the Tresca yield surface.

The main difficulty in this project seemed to be that the effort necessary to write such a computer code was underestimated. As a result, an attempt was made to accomplish too much in one single step.

Incorporating the recommendations above, a more fruitful and reasonable sequence of steps might be as follows:

Step I: Basic Code Development. The basic framework of the code would be written in this step. To make this step more manageable the following restrictions and assumptions should be made:

1. The constant strain triangle should be used as the basic element.
2. The Tresca yield surface should be used.
3. An isotropic, homogeneous material law (no layering) should be assumed for each element.

Step II: Addition of Bending Action. The goal of this step would be to allow the code to analyze problems without use of plane loading. This step should consist of only the addition of a plate element and a layering scheme to the code developed in Step I.

Step III: Addition of Nonuniform Material Properties. This step would accomplish the addition of steel reinforcement to the model completed in Step II. Many of the problems encountered in this study were due to the presence of the steel reinforcing rods. These problems would be minimized if the programmer could concentrate on this aspect alone.

Step IV: Addition of a More Realistic Yield Surface. At this point, the code would be ready for the addition of the final yield surface. The preceding steps developed a code using a yield surface which does not have the experimental basis that the yield surface proposed by Hand does have. The goal of this step would be to include

a more realistic yield surface and to complete the final verification of the code by comparing computed results with experimental data.

5.3. RECOMMENDATIONS FOR FURTHER STUDY

The series of steps outlined above would complete the original goals of this project. However, there would still remain many profitable areas of study. Some possible topics are:

1. Optimum Updating of Global Stiffness. The major computational expense involved in finite element analysis is invested in the solution of the global force-deflection equations. This expense is magnified in nonlinear analysis since the stiffness matrix is not constant and therefore must be factored many times. One way to reduce the number of factorizations required is to update the stiffness matrix, thus saving the expense of a complete solution to the new system of equations. However, an updated system of equations would probably converge to the correct solution with fewer iterations. Therefore, some investigation is required to determine the optimum time to update the stiffness matrix.

2. Iterative vs Direct Equation Solving. In this study, a Choleske decomposition technique is used to solve the force-deflection system of equations. This technique gives a direct solution to a system of equations, but requires a great deal of computation. An alternative would be to use an iterative type of solution such as Jacobi or Successive Over Relaxation. The advantage in an iterative solution is that each iteration can be performed with much less computation than a

direct solution. The results of a comparison of the two solution techniques for different classes of problems would be very useful in choosing the more efficient technique.

LIST OF REFERENCES

LIST OF REFERENCES

1. Bathe, K. J. and Wilson, E. L., Numerical Methods in Finite Element Analysis, Prentice-Hall, Englewood Cliffs, 1976.
2. Bell, Kolbein, "Triangular Plate Bending Elements," Finite Element Methods in Stress Analysis, I. Holand and K. Bell, ed., Tapir Press, Trondheim, Norway, 1969.
3. Burden, R. L., Faires, J. D. and Reynolds, A. C., Numerical Analysis, Prindle, Weber and Schmidt, Boston, 1978.
4. Cook, Robert D., Concepts and Applications of Finite Element Analysis, John Wiley and Sons, New York, 1974.
5. Fung, Y. C., A First Course in Continuum Mechanics, 2nd ed., Prentice-Hall, Englewood Cliffs, 1977.
6. Hand, F. R., Schnobrich, W. C. and Pecknold, D. A. W., "A Layered Finite Element Nonlinear Analysis of Reinforced Concrete Plates and Shells," Research Supported by the National Science Foundation (GK-11190).
7. Huebner, Kenneth H., The Finite Element Method for Engineers, John Wiley and Sons, New York, 1975.
8. Kupfer, H., Hilsdorf, H. K. and Rüsch, H., "Behavior of Concrete under Biaxial Stresses," Journal, American Concrete Institute, Vol. 66, No. 8, August 1969, pp. 656-666.
9. Lin, C. S., "Nonlinear Analysis of Reinforced Concrete Slabs and Shells," Ph.D. Thesis, University of California, Berkeley, California, September, 1972.
10. Noble, B. and Daniel, J. W., Applied Linear Algebra, 2nd ed., Prentice-Hall, Englewood Cliffs, 1977.
11. Popov, E. P., Introduction to Mechanics of Solids, Prentice-Hall, Englewood Cliffs, 1968.
12. Przemienicki, J. S., Theory of Matrix Structural Analysis, McGraw-Hill, New York, 1968.
13. Timoshenko, S. P. and Woinowsky-Krieger, Theory of Plates and Shells, McGraw-Hill, New York, 1959.
14. Timoshenko, S. P. and Goodier, J. N., Theory of Elasticity, 3rd ed., McGraw-Hill, New York, 1934.

15. Tocher, J. L., "Analysis of Plate Bending Using Triangular Elements," Ph.D. Dissertation, Department of Civil Engineering, University of California, Berkeley, California, 1962.
16. Zienkiewicz, O. C., The Finite Element Method in Engineering Science, McGraw-Hill, New York, 1971.
17. Zienkiewicz, O. C., Valliappan, S. and King, I. P., "Elasto-Plastic Solutions of Engineering Problems 'Initial Stress,' Finite Element Approach," International Journal for Numerical Methods in Engineering, Vol. I, 75-100, 1969.

APPENDIXES

APPENDIX A

USER'S GUIDE TO "CRACK"

The following deck of cards must be supplied to run "CRACK." Some of the input Sections listed below are required and some are optional. In general, each input Section is preceded by a card giving the number of data cards to be read for that Section. If the user wishes to omit an optional Section, this is indicated by inserting a blank card for that Section.

SECTION I - GENERAL PROBLEM INFORMATION (Required Section)

Card 1 (3I5)

<u>Col.</u>	<u>Variable</u>	<u>Description</u>
1- 5	NUMEL	No. of elements in mesh
6-10	NUMPTS	No. of nodes in mesh
11-15	NUMAT	No. of different material types

SECTION II - MATERIAL PROPERTIES

NUMAT Cards with the following information must be supplied. Material properties must be input in order, i.e., the first group of Cards are for Material No. 1.

Card 1 (5E10.3)

<u>Col.</u>	<u>Variable</u>	<u>Description</u>
1-10	FC	Transition stress between elastic regions in biaxial compression
11-20	EC	Elastic modulus in first elastic region.

<u>Col.</u>	<u>Variable</u>	<u>Description</u>
21-30	FD	Transition stress between elastic and plastic regions in biaxial compression.
31-40	ED	Elastic modulus in second elastic region.
41-50	FT	Maximum tensile stress in biaxial tension.

Card 2 (5E10.3)

1-10	V	Poisson's ratio for concrete
11-20	EU	Crushing strain of concrete
21-30	DEN	Concrete density
31-40	FY	Yield stress for reinforcement
41-50	ES	Elastic modulus of reinforcement.

NOTE: A more complete description of the above variables can be found in Section 2.

SECTION III - NODAL POINT INPUT (Required Section)

1- 5 NCDS Total no. of nodal pt. input cards

Nodal coordinate cards (3I5, 6E10.3)

1- 5	N1	Node number of first node in a generation sequence.
6-10	N2	Node number of last node in a generation sequence. If no generation is being requested
11-15	N3	Generation increment. Node numbers will be generated as follows: The first node generated will have a node number, N, given by

<u>Col.</u>	<u>Variable</u>	<u>Description</u>
		$N = N1 + N3$ and so on until $N = N2$.
		Nodal coordinates will be evenly spaced between $N1$ and $N2$.
16-25	R(1)	X - coordinate of $N1$
26-35	R(2)	Y - coordinate of $N1$
36-45	R(3)	Z - coordinate of $N1$
46-55	R(4)	X - coordinate of $N2$
56-65	R(5)	Y - coordinate of $N2$
66-75	R(6)	Z - coordinate of $N2$
NOTE: If nodal generation is not requested ($N2 = N1$) columns 46-75 may be left blank.		

SECTION IV - DISTRIBUTED LOAD SETS (Optional Section)

Card 1 (I5)

1- 5 NDLS Number of distributed load sets to be input.

If this problem does not require any distributed loading, a blank card is required.

Distributed load card(s) (I5, 5E10.4)
(If NDLS = 0, skip this Section)

1- 5	NDL	Distributed load set number
6-15	DFOR(NDL,1)	X - component of distributed force
16-25	DFOR(NDL,2)	Y - component of distributed force
26-35	DFOR(NDL,3)	Z - component of distributed force
36-45	DFOR(NDL,4)	X - component of distributed moment
46-55	DFOR(NDL,5)	Y - component of distributed moment

NOTE: The distributed loads are input with respect to the element's local coordinate system. See Section 2 for a description of the local coordinate system.

SECTION V - ELEMENT CONNECTIVITY (Required Section)

Card 1 (I5)

<u>Col.</u>	<u>Variable</u>	<u>Description</u>
1- 5	NCDS	Total number of data cards used to describe the element connectivity.

Element Generation Cards

First Card (I5, 5X, I5, E10.3, 5I5)

1- 5	M	Element no. of first element in generation scheme
11-15	II	Generation Increment. If $II = 0$, no generation is performed. Element numbers are generated by $M2 = M + II$. Where $M2$ is the 2nd element in the generation scheme
16-25	TT	Element thickness
26-30	IR(1)	Node 1 of element M
31-35	IR(2)	Node 2 of element M
36-40	IR(3)	Node 3 of element M
41-45	NMAT(M)	Material type for element M
46-50	NDIS(M)	Distributed load type for element M

Second Card (5X, I5, I5X, 3I5)

(Skip this card if $II = 0$)

6-10	L	Element no. for last element in generation scheme
------	---	---

<u>Col.</u>	<u>Variable</u>	<u>Description</u>
26-30	IRR(1)	Node 1 of element L
31-35	IRR(2)	Node 2 of element L
36-40	IRR(3)	Node 3 of element L

NOTE: In all generation schemes, the material and distributed load types and element thickness for all elements in the scheme are specified on the first card.

SECTION VI - LAYERING SCHEME (Required Section)

Card 1 (I5)

1- 5	NCDS	No. of data cards in this Section Layer thickness cards (4I5, 6E10.3, 4E10.3)
1- 5	N 1	First element in layering scheme
6-10	N 2	Last element in layering scheme
11-15	N 3	Increment between successive elements in this layering scheme
16-20	NLRS	Number of layers in this scheme
21-30	TLR(1)	Thickness of layer 1
31-40	TLR(2)	Thickness of layer 2
	TLR(NLRS)	Thickness of layer NLRS

If NLRS 6, a second card must be used to input the last layers.

SECTION VII - REINFORCEMENT SCHEMES (Optional Section)

Card 1 (2I5)

1- 5	NTYPE	Reinforcement type number
6-10	NSLRS	Number of reinforcement layers in this scheme

Card(s) 2 (I5, 4E10.3)

There should NSLRS of these cards

<u>Col.</u>	<u>Variable</u>	<u>Description</u>
1- 5	LN	Element layer number which contains this layer of reinforcement
6-15	SPERCI	% of layer cross-section for the first row of reinforcement
16-25	ANG 1	Orientation of this row of reinforcement with respect to the local X-axis
26-35	SPERC2	% of layer cross-section for the 2nd row of reinforcement
36-45	ANG 2	Orientation of the 2nd row with respect to the local X-axis (degrees)

NOTE: A layer of reinforcement does not have to contain 2 rows (orientations) of reinforcement. If a layer of reinforcement is to contain only 1 row of reinforcement, SPERC2 = ANG 2 = 0.

SECTION VIII - ELEMENT REINFORCEMENT TYPE (Required Section)

Card 1 (I5)

1- 5	NCDS	No. of data cards in this Section
		Reinforcement Type (4I5)
1- 5	N 1	First element in reinforcement type generation scheme
6-10	N 2	Last element in reinforcement type generation scheme
11-15	N 3	Generation increment
16-20	NTYPE	Reinforcement type being generated

SECTION IX - BOUNDARY CONDITIONS

<u>Card 1</u> (I5)		
<u>Col.</u>	<u>Variable</u>	<u>Description</u>
1- 5	NCDS	No. of boundary condition cards
		Boundary condition card(s) (3I5, 2X, 3I1)
1- 5	N 1	First node in a boundary condition generation scheme
6-10	N 2	Last node in a boundary condition generation scheme
11-15	N 3	Node number increment to be used in generation scheme
18	KOND(1)	.EQ. 1 $u = 0$
		.EQ. 2 $\theta_x = 0$
19	KOND(2)	.EQ. 1 $v = 0$
		.EQ. 2 $\theta_y = 0$
20	KOND(3)	.EQ. 1 $w = 0$
		.EQ. 2 $\theta_z = 0$

NOTE: The user must be careful to restrain any rotation which acts to the plane of an element.

SECTION X - CONCENTRATED NODAL LOADS

<u>Card 1</u> (I5)		
1- 5	NCDS	No. of NODAL LOAD DATA CARDS
		NODAL LOAD CARD(s) (3I5, 6E10.3)
1- 5	N 1	Number of FIRST node in a load generation sequence

<u>Col.</u>	<u>Variable</u>	<u>Description</u>
6-10	N 2	Number of last node in the load generation sequence
11-15	N 3	Generation increment to be used in nodal load generation
16-25	R(1)	x-component of concentrated load
26-35	R(2)	y-component of concentrated load
36-45	R(3)	z-component of concentrated load
46-55	R(4)	x-component of concentrated moment
56-65	R(5)	y-component of concentrated moment
66-75	R(6)	z-component of concentrated moment

SECTION XI - LOAD INCREMENTS

Card 1 (2I5)

1- 5	NLI	No. of load increments
6-10	NCDS	No. of load increment data cards
Load Increment data (3I5, E10.3)		
1- 5	N 1	No. of load increment at the beginning of a generation scheme
6-10	N 2	No. of load increment at the end of a generation scheme. If no generation is requested $N2 = 0$ or $N2 = N1$.
11-15	N 3	Load number generation increment
16-25	PERC	Load increment percentage of the total load

APPENDIX B

LISTING OF FORTRAN STATEMENTS IN "CRACK"

```

C *****MAIN0010
C
C THIS PROGRAM ANALYSES REINFORCED CONCRETE PLATES AND SHELLS USING MAIN0020
C THE CRACKED LAYERED FINITE ELEMENT METHOD OF ANALYSIS. THE CAP- MAIN0030
C ACITY OF THIS PROGRAM IS 150 ELEMENTS AND 150 NODES. IT ALLOWS MAIN0040
C FOR EACH ELEMENT TO BE COMPOSED OF UP TO TEN (10) LAYERS OF CON- MAIN0050
C CRETE AND CONTAIN THREE LAYERS OF REINFORCEMENT. THE BOUNDARY MAIN0060
C CONDITIONS ARE EITHER FIXED OR FREE. A MAXIMUM OF THIRTY LOAD MAIN0070
C INCREMENTS MAY BE APPLIED AND EACH LOAD INCREMENT IS IN TERMS OF MAIN0080
C PERCENT OF TOTAL LOAD. APPROXIMATELY 310K BYTES OF CORE IS REQ- MAIN0090
C UIRED FOR EXECUTION OF THIS PROGRAM. GLOBAL Z DIRECTION IS MAIN0100
C CONSIDERED GRAVITY DIRECTION. MAIN0110
C THIS PROGRAM WAS WRITTEN AT CLARKSON COLLEGE OF TECHNOLOGY MAIN0120
C BY THOMAS BALLARD AND ANDREW KRAL. MAIN0130
C THE PROGRAM WAS COMPLETED AND REVISED AT THE UNIVERSITY MAIN0140
C OF TENNESSEE BY WILLIAM DORRIS.
C
C *****MAIN0150
C *****MAIN0160
C DOUBLE PRECISION AK,AB,BX,A,AL,ST,DL,BLOAD,CLOAD
C DOUBLE PRECISION TR,D,E,F,AR,DL,R
C DOUBLE PRECISION TS,PS,SR,CELCON
C COMMON/MAT/FCS(4),ECS(4),FDS(4),EDS(4),FTS(4),VS(4),EUS(4),FY,
C ES,DENS(4),BETA,NMAT(300)
C COMMON/ELCON/A(15,15),TR(3,3),AR(4),TS(3,3,2,3),PS(2,3),SR(6)
C COMMON/ELVAR/EPTOT(3,10),CC(3,3,10),CS(3,3,6),STT(3,10),SSTR(2,3)
C COMMON/STRINC/EPP(3,10),STP(3,10),EP(3,10)
C COMMON/CST/TJ(11,300),C(3,3)
C COMMON/STIF/ST(18,18),D(3,3),E(3,3),F(3,3)
C COMMON/EXFR/ITRAN(10,300),IST(6,300),EXF(18),EF(18),IBLANK(10,300)
C COMMON/STRES/B(6,15),STR(5,10),EST(5,10),FS(3,6)
C COMMON/DISP/ DL(1200),DLL(18),U(1200)
C COMMON/IMPRUV/RESID(200) , UPRIME(200)
C COMMON /DA/ DAT(2,300)
C COMMON /NUNIT/NSTIFF,ICOUNT
C ICOUNT = 1
C DIMENSION XYZ(3,200),R(6),T(300),KO(3,300),RR(3),TLR(10),NL(300),
C *S_LR(19,2),NTS(300),WT(300),IVAL(6,300),KEND(3),ALOAD(1200),
C *BLOAD(1200),XM(15),NEN(15),AL(1200),SI(3),ALI(30),AK(9000),
C *AB(9000) ,IEL(300),IPRINT(300),GD(6),ITR(10,300),RD(2),
C *CD(3,3,10),IR(6),IRR(3),CLOAD(1200)
C DIMENSION BX(90),DFOR(5,5),NDIS(300)
C DIMENSION CELCON(542) , CELVAR(210)
C EQUIVALENCE (CELCON(1),A(1,1)) , (CELVAR(1),EPTOT(1,1))
C INTEGER SR
C REAL YES/4H YES/,NO/4H NO/
C DATA NIT/10/
C DATA NEN/1,2,3,4,5,7,8,9,10,11,13,14,15,16,17/
C DATA NEL/300/,NPT/200/,NRTNC/228/
C DATA P1/3.1415926/
C BETA = 1.16
C DATA NTP1/1/,NTP2/2/,NTP4/4/,NTP5/8/,NTP6A/9/,NTP6B/10/
C DATA NTP3A/3/ , NTP3B/11/
C DEFINE FILE 2(300,700,U,12)
C DEFINE FILE 8(150,12,U,NPT1)
C DEFINE FILE 4(150,2000,U,IBLK)
C 1 FORMAT (315)
C 2 FORMAT (1H1,28H GENERAL PROBLEM INFORMATION ,/1H ,28(' '),//
C * 5X,21H NUMBER OF ELEMENTS ,15/,
C * 5X,21H NUMBER OF POINTS ,15/,'NUMBER OF MATERIALS',15)
C 3 FORMAT (/31H **** DIMENSION EXCEEDED ****)

```

```

4  FORMAT (/36H NUMBER OF ELEMENTS IS GREATER THAN ,I3)          MAINC470
5  FORMAT (/34H NUMBER OF POINTS IS GREATER THAN ,I3)            MAINC480
6  FORMAT (5E10.3)                                                MAINC490
7  FORMAT (//,' TABLE OF MATERIAL PROPERTIES',/1H ,28('='),//   MAIN0500
  *5X,' CONCRETE PROPERTIES',/,
  *10X,' TRANSITION STRESS BETWEEN FIRST AND SECOND ELASTIC REGIONS = ' MAINC520
  *',E10.3/,
  *12X,' ELASTIC MODULUS BETWEEN FIRST AND SECOND ELASTIC REGIONS = ' MAINC530
  *',E10.3/,
  *15X,' TRANSITION STRESS BETWEEN ELASTIC AND PLASTIC REGIONS = ', MAINC560
  *E10.3/,17X,' ELASTIC MODULUS BETWEEN ELASTIC AND PLASTIC REGIONS = ' MAINC570
  *',E10.3/,40X,' TENSILE STRENGTH OF CONCRETE = ',E10.3/,
  *41X,' POISSONS RATIO FOR CONCRETE = ',E10.3/,
  *41X,' CRUSHING STRAIN OF CONCRETE = ',E10.3/,
  *42X,' WEIGHT DENSITY OF CONCRETE = ',E10.3,//
  *5X,' REINFORCEMENT PROPERTIES',/,
  *11X,' YIELD STRENGTH OF REINFORCEMENT = ',E10.3/,
  *11X,' ELASTIC MODULUS OF REINFORCEMENT = ',E10.3//)
9  FORMAT (//,27H TABLE OF NODAL COORDUNATES ,/1H ,27('='),//) MAINC650
10 FORMAT (I5)                                                    MAINC660
11 FORMAT (3I5,6E10.3)                                           MAINC670
12 FORMAT (3X,I5,2X,I5,1X,I5,4X,E10.3,5X,E10.3,5X,E10.3,10X,E10.3,5X,E10.3,5X,E10.3) MAINC680
13 FORMAT (4X,99H FROM THRU INCR STARTING POINT COORDINAT
  CES END POINT COORDINATES ,/,
  *113H PT. PT. X Y Z
  *C X Y Z ,/)
14 FORMAT (//,29H TABLE OF ELEMENT PROPERTIES ,/1H ,29('='),//) MAINC730
15 FORMAT (//,5X,50H FROM THRU INCR THICKNESS NCDE NUMBERS
  C/,5X,50H FLMT ELMT 1 2 3 ,//)
16 FORMAT (I5,5X,I5,E10.3,6I5)
17 FORMAT (7X,I3,9X,I3,12X,E10.3,3X,I3,2X,I3,1X,I3,3I5)
18 FORMAT (5X,I5,15X,3I5)
19 FORMAT (5X,I5,21X,I3,2(2X,I3))
20 FORMAT (/35H **** ELEMENT NUMBERING ERROR **** )
21 FORMAT (//,24H ELEMENT LAYERING SCHEME ,/1H ,24('='),//)
22 FORMAT (5X,19H NUMBER OF SCHEMES=,I5//)
23 FORMAT (5X,51H FROM THRU INCR NUMBER THICKNESS CF LAYER NO. /,
  * 5X,121H FLMT ELMT OF LAYERS 1 2 3
  * 4 5 6 7 8 9
  *10//)
24 FORMAT (4I5,6E10.3/,4E10.3)
25 FORMAT (5X,4I5)
26 FORMAT (/36H **** ERROR IN LAYERING SCHEME **** /,
  * 59H THE SUM OF THE THICKNESSES OF THE LAYERS OF ELEMENT
  *NUMBER,I3,35HIS GREATER THAN THE TOTAL THICKNESS,E10.3)
27 FORMAT (3X,3I5,2X,I5,4X,10E10.3)
28 FORMAT (//,/' REINFORCEMENT PROPERTIES',/1H ,24('='),//)
29 FORMAT (5X,34H NUMBER OF REINFORCEMENT SCHEMES =,I5)
300 FORMAT (/15X,' LAYER NUMBER',10X,' STEEL PERC1',10X,' ANG1',10X,
  *' STEEL PERC2',10X,' ANG2')
34 FORMAT (20X,I2,14X,E10.3,9X,E10.3,8X,E10.3,8X,E10.3)
33 FORMAT (/5X,7H SCHEME,I5,/5X,' NUMBER OF STEEL LAYERS IN THIS SCHEME
  CE =,I5//)
31 FORMAT (/5X,76H * - ANGLE OF REINFORCEMENT IS MEASURED COUNTERCLOCK
  *WISE FROM LOCAL X-AXES /,5X,
  * ' **--PERC STEEL IS PERC PER LAYER CF THE ELEMENT')
32 FORMAT (2I5)
33 FORMAT (I5,4E10.3)
35 FORMAT (/ 22H ELEMENT REINFORCEMENT/,1H ,22('='))

```



```

35 FORMAT(5X,21H FROM THRU INCR TYPE /,5X,                                MAIN1070
*      11H ELMT ELMT )                                                    MAIN1080
37 FORMAT(4I5)                                                            MAIN1090
39 FORMAT(//24H BOUNDARY CONDITION TABLE,/1H ,25('='),/)                MAIN1100
40 FORMAT(/5X,27H FROM THRU INCR  CONDITION*/,5X,                          MAIN1110
*      25H PT PT      XYZ)                                              MAIN1120
41 FORMAT (3I5,2X,3I1)                                                    MAIN1130
42 FORMAT(5X,3I5,6X,3I1)                                                  MAIN1140
43 FORMAT(/5X,38H *CONDITION = 0 FOR NO SPECIFICATION /,5X,              MAIN1150
*      44H CONDITION = 1 FOR SPECIFIES DISPLACEMENT /,5X,              MAIN1160
*      40H CONDITION = 2 FOR SPECIFIES ROTATION )                        MAIN1170
44 FORMAT (//50H TABLE OF DISTRIBUTED LOADS (ELEMENT COORDINATES) /)    MAIN1180
45 FORMAT(///21H NO DISTRIBUTED LOADS,/1H ,21('='),/)                    MAIN1190
46 FORMAT (/38H LOAD  DISTRIBUTED LOADS IN DIRECTION /,
*      53H SET      X      Y      Z      XX      YY /)
47 FORMAT (15,5F10.4)
48 FORMAT (3I5,2X,5E10.3)                                                  MAIN1240
49 FORMAT(//42H TABLE OF NODAL LOADS (GLOBAL COORDINATES),/1H ,42('=' MAIN1250
*      31,/)
50 FORMAT (/15H NO NODAL LOADS )                                          MAIN1260
51 FORMAT(/5X,43H FROM THRU INCR  APPLIED FORCES OR MOMENTS /,5X,       MAIN1270
*      76H PT PT      X      Y      Z      MX MAIN1280
*      MY      MZ/)                                                    MAIN1290
52 FORMAT (3I5,6E10.3)                                                    MAIN1300
53 FORMAT(5X,3I5,2X,6E10.3)                                              MAIN1310
54 FORMAT(//25H TABLE OF LOAD INCREMENTS,/1H ,25('='),/)              MAIN1320
55 FORMAT (2I5)                                                            MAIN1330
56 FORMAT(/5X,28H FROM THRU INCR  PERCENT OF /,5X,                      MAIN1340
*      28H LOAD LOAD      TOTAL LOAD /,5X,                            MAIN1350
*      10H INCR INCR/)                                                MAIN1360
57 FORMAT(/5X,29H NUMBER OF LOAD INCREMENTS = 15/)                      MAIN1370
58 FORMAT (3I5,E10.3)                                                    MAIN1380
59 FORMAT(5X,3I5,2X,E10.3)                                              MAIN1390
60 FORMAT(///1H ,130('*'),///16H LOAD INCREMENT ,15)                   MAIN1400
61 FORMAT(//10X,30H TABLE OF NODAL DISPLACEMENTS /,                   MAIN1410
*      20X,61H DISPLACEMENT IN DIRECTIONS  ROTATION ABOUT AXES (R MAIN1420
*      *ADIANS) /,10X,                                                  MAIN1430
*      65H NODE      X      Y      Z      X      Y                      MAIN1440
*      Z /)                                                            MAIN1450
62 FORMAT(10X,15,3X,6E10.3)                                              MAIN1460
63 FORMAT(/10X,46H TABLE OF LAYER STRESSES(ELEMENT COORDINATES)//,    MAIN1470
*      220X,' ELEMENT      LAYER      STRESSES IN THE CONCRETE(LBS/SQIN.) MAIN1480
*      UNIAXIAL STRESS IN THE REINCEMENT',//,                          MAIN1490
*      220X,9H NUMBER,4X,8H NUMBER,3X,8H SIGMA X,5X,8H SIGMA Y,6X,7H TAU X MAIN1500
*      Y,26X,5H ROW1,12X,5H ROW2 )                                     MAIN1510
64 FORMAT(20X,15)                                                        MAIN1520
65 FORMAT(33X,15,1X,E10.3,5X,E10.3,5X,E10.3)                            MAIN1530
66 FORMAT(95X,E10.3,7X,E10.3)                                            MAIN1540
67 FORMAT(//57H TABLE OF LAYERS THAT HAVE EXCEEDED TRANSITION CRITER MAIN1550
*      *IA /,                                                            MAIN1560
*      47H ELMT LAYER  PRESENT MATERIAL  REINFORCEMENT/,              MAIN1570
*      47H  NUM  NUM      STATE**      STATE*** /,                    MAIN1580
*      35X,12H 1      2 /)                                              MAIN1590
68 FORMAT (1X,15)                                                        MAIN1600
69 FORMAT (7X,15,7X,15)                                                  MAIN1610
70 FORMAT (1H*,35X,A4,5X,A4)                                             MAIN1620
71 FORMAT (/19H ** MATERIAL STATE /,                                     MAIN1630
*      33H 2 = CRACKED IN ONE DIRECTION /,                            MAIN1640
*      33H 5 = SECOND ELASTIC REGION /,                                MAIN1650
*      23H 7 = PLASTIC REGION /,                                       MAIN1660

```

```

*      52H      9 = CRUSHED REGION OR CRACKED IN TWO DIRECTIONS /, MAIN1680
*      26H *** REINFORCEMENT STATE /, MAIN1690
*      33H      YES = REINFORCEMENT YIELDING /, MAIN1700
*      33H      NO = ELASTIC REINFORCEMENT /) MAIN1710
72 FORMAT (//29H **** NO REINFORCEMENT **** /) MAIN1720
73 FORMAT (//36H **** ERROR IN NODE NUMBERING **** /, MAIN1730
*      59H THE LOCAL Y-COORDINATE OF NODE 2 PLUS THE SUM OF THE LOCAL/, MAIN1740
*      59H X-COORDINATES OF NODES 2 AND 3 IS VERY SMALL. THIS MAY /, MAIN1750
*      29H RESULT IN NUMERICAL ERRORS. /, MAIN1760
*      35H ** RENUMBER THE NODES FOR ELEMENT ,I5, 3H ** ) MAIN1770
74 FORMAT (7X,I5,23X,A4,5X,A4) MAIN1780
75 FORMAT (//,17H **** FLAG **** /, MAIN1790
*      59H THE ITERATIONS ABOUT A LOAD INCREMENT HAS EXCEEDED NIT. /, MAIN1800
*      39H A FAILURE MECHANISM HAS BEEN SET UP ) MAIN1810
*****
C      THIS SECTION OF THE PROGRAM INPUTS THE DATA , SETS INITIAL
C      VALJES AND PERFORMS PRELIMINARY CALCULATIONS
C
C      *****
C      INPUT GENERAL PROBLEM INFORMATION
C      READ 1,NUMEL,NUMPTS,NUMAT MAIN1830
C      PRINT 2, NUMEL,NUMPTS,NUMAT MAIN1840
C      CHECK FOR ERRORS MAIN1850
C      IF(NJMEL .LE. NEL) GO TO 101 MAIN1860
C      PRINT 3 MAIN1870
C      PRINT 4, NEL MAIN1880
C      GO TO 1000 MAIN1890
101 IF (NUMPTS .LE. NPT) GO TO 102 MAIN1900
C      PRINT 3 MAIN1910
C      PRINT 5, NPT MAIN1920
C      GO TO 1000 MAIN1930
102 NQ = NUMPTS * 6 MAIN1940
C
C      INPUT ELASTIC CONSTANTS MAIN1950
C
C      DO 2000 NM=1,NUMAT
C      READ 6, FCS(NM),ECS(NM),FDS(NM),EDS(NM),FTS(NM),VS(NM),
C      EUS(NM),DENS(NM),FY,ES
C      PRINT 7, FCS(NM),ECS(NM),FDS(NM),EDS(NM),FTS(NM),VS(NM),
C      EUS(NM),DENS(NM),FY,ES
2000 CONTINUE
C
C      INPUT NODAL COORDINATES MAIN1980
C
C      PRINT 8 MAIN1990
C      READ 9, NCDS MAIN2000
C      PRINT 13 MAIN2010
C      DO 106 N = 1,NCDS MAIN2020
C      READ 11, N1,N2,N3,(R(I),I=1,6) MAIN2030
C      PRINT 12, N1,N2,N3,(R(I),I=1,6) MAIN2040
C      IF(N2 .LE. 0) N2 = N1 MAIN2050
C      IF(N3 .LE. 0) N3 = 1 MAIN2060
C      DO 103 I = 1,3 MAIN2070
103 XYZ(I,N1) = R(I) MAIN2080
C      IF (N1 .EQ. N2) GO TO 106 MAIN2090
C      DO 104 I = 1,3 MAIN2100
104 XYZ(I,N2) = R(I+3) MAIN2110
C      DIF = N2 - N1 MAIN2120

```

EN3 = N3	MAIN2130
FAC = EN3/DIF	MAIN2140
N1 = N1 + N3	MAIN2150
N2 = N2 - N3	MAIN2160
DO 105 I = 1,3	MAIN2170
R(I) = FAC*(R(I+3) - R(I))	MAIN2180
DO 105 J = N1,N2,N3	MAIN2190
105 XYZ(I,J) = XYZ(I,J-N3) + R(I)	MAIN2200
106 CONTINUE	MAIN2210
C	
C INPUT DISTRIBUTED LOADS	
C	
READ 9,NDLS	
IF (NDLS .EQ. 0) PRINT 45	
IF (NDLS .EQ. 0) GO TO 910	
PRINT 44	
PRINT 46	
DO 900 LS=1,NDLS	
READ 47,NDL,(DFOR(NDL,K),K=1,5)	
PRINT 47 ,NDL,(DFOR(NDL,K),K=1,5)	
900 CONTINUE	
910 CONTINUE	
C	
C INPUT ELEMENT PROPERTIES	
C	
PRINT 14	MAIN2230
READ 9, NCDS	MAIN2240
PRINT 15	MAIN2250
ND = 0	MAIN2260
DO 111 N = 1,NCDS	MAIN2270
IF (ND .EQ. NCDS) GO TO 112	MAIN2280
ND = ND + 1	MAIN2290
READ 16,M,II,TT,(IR(I),I=1,3),NMAT(M),NDIS(M),NTS(M)	
PRINT 17,M,II,TT,(IR(I),I=1,3),NMAT(M),NDIS(M),NTS(M)	
T(M) = TT	MAIN2320
DO 107 K = 1,3	MAIN2330
107 KQ(K,M) = IR(K)	MAIN2340
IF (II .EQ. 0) GO TO 111	MAIN2350
ND = ND + 1	MAIN2360
READ 18, L,(IRR(II),I=1,3)	MAIN2370
PRINT 19, L,(IRR(II),I=1,3)	MAIN2380
NI = (L-M)/II	MAIN2390
DO 108 K = 1,3	MAIN2400
108 IR(K) = (IRR(K) - IR(K))/NI	MAIN2410
MD = M + II	MAIN2420
DO 109 J = MD, L, II	MAIN2430
T(J) = TT	MAIN2440
NDIS(J) = NDIS(M)	
NTS(J) = NTS(M)	
NMAT(J) = NMAT(M)	
DO 109 K = 1,3	MAIN2450
109 KQ(K,J) = KQ(K,J-II) + IR(K)	MAIN2460
CHECK FOR ERRORS	MAIN2470
DO 110 K = 1,3	MAIN2480
IF (KQ(K,L) .EQ. IRR(K)) GO TO 110	MAIN2490
PRINT 20	MAIN2500
GO TO 1000	MAIN2510
110 CONTINUE	MAIN2520
111 CONTINUE	MAIN2530
112 CONTINUE	MAIN2540

C		
C	INPUT LAYERING SCHEME	MAIN2550
C		
	PRINT 21	MAIN2560
	READ 9, NCDS	MAIN2570
	PRINT 22, NCDS	MAIN2580
	PRINT 23	MAIN2590
	DO 115 I = 1, NCDS	MAIN2600
	READ 24, N1, N2, N3, NLRS, (TLR(N), N=1, NLRS)	MAIN2610
	PRINT 27, N1, N2, N3, NLRS, (TLR(N), N=1, NLRS)	MAIN2620
	IF (N2 .EQ. 0) N2 = N1	MAIN2630
	IF (N3 .EQ. 0) N3 = 1	MAIN2640
	DO 114 N = N1, N2, N3	MAIN2650
	TT = T(N)/2.	MAIN2660
	TJ(1, N) = TT	MAIN2670
	DO 113 J = 1, NLRS	MAIN2680
	113 TJ(J+1, N) = TJ(J, N) - TLR(J)	MAIN2690
	NL(N) = NLRS	MAIN2700
C	CHECK FOR ERRORS	MAIN2710
	IF (ABS(TJ(NLRS+1, N) + TT) .LE. .001) GO TO 114	MAIN2720
	PRINT 26, N, T(N)	MAIN2730
	GO TO 1000	MAIN2740
	114 CONTINUE	MAIN2750
	115 CONTINUE	MAIN2760
C		
C	INPUT LOCATION OF REINFORCEMENT	MAIN2770
C		
	PRINT 28	MAIN2780
	READ 9, NTP	MAIN2790
	PRINT 29, NTP	MAIN2800
	IF (NTP .EQ. 0) GO TO 500	MAIN2810
	CALL RZERO(SLR, 19, 2, 1, 1)	
	DO 117 I = 1, NTP	MAIN2850
	READ 32, NTYPE, NSLRS	MAIN2860
	PRINT 30, NTYPE, NSLRS	MAIN2870
	PRINT 300	MAIN2880
	SLR(1, NTYPE) = NSLRS	MAIN2890
	L = 1	MAIN2900
	DO 116 J = 1, NSLRS	MAIN2910
	READ 33, LN, SPERC1, ANG1, SPERC2, ANG2	MAIN2920
	SLR(L+1, NTYPE) = LN	MAIN2930
	SLR(L+2, NTYPE) = SPERC1/100.	MAIN2940
	SLR(L+3, NTYPE) = ANG1*PI/180.	MAIN2950
	SLR(L+4, NTYPE) = SPERC2/100.	MAIN2960
	SLR(L+5, NTYPE) = ANG2*PI/180.	MAIN2970
	NSPL = 2	
	IF (SLR(L+4, NTYPE) .EQ. 0.0) NSPL = 1	
	SLR(L+6, NTYPE) = NSPL	
	L = L+6	
	PRINT 34, LN, SPERC1, ANG1, SPERC2, ANG2	MAIN2990
	SR(J) = NSPL	
	116 CONTINUE	MAIN3000
	117 CONTINUE	MAIN3010
	500 CONTINUE	MAIN3020
	IF (NTP .EQ. 0) PRINT 72	MAIN3030
	IF (NTP .NE. 0) PRINT 31	MAIN3040
C		
C	INPJT REINFORCEMENT TYPE FOR EACH ELEMENT	MAIN3050
C		
	CALL INTVAL(NTS, NUMEL, 1, 0)	

```

READ 9, NCDS
PRINT 35
PRINT 36
DO 119 N = 1, NCDS
  READ 37, N1, N2, N3, NTYPE
  PRINT 25, N1, N2, N3, NTYPE
  IF (N2 .EQ. 0) N2 = N1
  IF (N3 .EQ. 0) N3 = 1
  DO 118 L = 1, N1, N2, N3
118 NTS(L) = NTYPE
119 CONTINUE
C. CALCULATE HALF BAND WIDTH
  NBAND = 0
  DO 121 M = 1, NUMEL
    MIN = KQ(1, M)
    MAX = KQ(1, M)
    DO 120 I = 2, 3
      IF (KQ(I, M) .LT. MIN) MIN = KQ(I, M)
      IF (KQ(I, M) .GT. MAX) MAX = KQ(I, M)
120 CONTINUE
    MBAND = (MAX - MIN + 1)*6
    IF (MBAND .GT. NBAND) NBAND = MBAND
121 CONTINUE
  NC = NBAND
  NR = NBAND
  IF (NR .GT. NQ) NR = NQ
C FOR EACH ELEMENT CALCULATE THE TRANSFORMATION MATRIX(TR)
C AND THE ELEMENT DIMENSION MATRIX(AR)
C 1) LOCAL Y-AXIS DEFINED BY DIRECTION FROM NODE 1 TO NODE 2
C 2) LOCAL Z-AXIS DEFINED BY CROSS-PRODUCT OF SIDE 13 WITH
C SIDE 12 (R13 X R12)
C 3) LOCAL X-AXIS DEFINED BY CROSS-PRODUCT OF LOCAL Y-AXIS
C AND LOCAL Z-AXIS (J X K)
CALL RZERO(ALOAD, NQ, 1, 1, 1)
DO 128 L = 1, NUMEL
  N1 = KQ(1, L)
  N2 = KQ(2, L)
  N3 = KQ(3, L)
  X12 = XYZ(1, N2) - XYZ(1, N1)
  Y12 = XYZ(2, N2) - XYZ(2, N1)
  Z12 = XYZ(3, N2) - XYZ(3, N1)
  YL = SQRT(X12*X12 + Y12*Y12 + Z12*Z12)
  TR(2, 1) = X12/YL
  TR(2, 2) = Y12/YL
  TR(2, 3) = Z12/YL
  X13 = XYZ(1, N3) - XYZ(1, N1)
  Y13 = XYZ(2, N3) - XYZ(2, N1)
  Z13 = XYZ(3, N3) - XYZ(3, N1)
  TR(3, 1) = Y13*Z12 - Y12*Z13
  TR(3, 2) = X12*Z13 - X13*Z12
  TR(3, 3) = X13*Y12 - X12*Y13
  ZL = SQRT(TR(3, 1)*TR(3, 1) + TR(3, 2)*TR(3, 2) + TR(3, 3)*TR(3, 3))
  TR(3, 1) = TR(3, 1)/ZL
  TR(3, 2) = TR(3, 2)/ZL
  TR(3, 3) = TR(3, 3)/ZL
  TR(1, 1) = TR(2, 2)*TR(3, 3) - TR(3, 2)*TR(2, 3)
  TR(1, 2) = TR(3, 1)*TR(2, 3) - TR(2, 1)*TR(3, 3)
  TR(1, 3) = TR(2, 1)*TR(3, 2) - TR(3, 1)*TR(2, 2)
  AR(1) = X12*TR(2, 1) + Y12*TR(2, 2) + Z12*TR(2, 3)
  AR(2) = X13*TR(1, 1) + Y13*TR(1, 2) + Z13*TR(1, 3)

```

PAIN3080
 PAIN3090
 PAIN3100
 PAIN3110
 PAIN3120
 PAIN3130
 PAIN3140
 PAIN3150
 PAIN3160
 PAIN3170
 PAIN3180
 PAIN3190
 PAIN3200
 PAIN3210
 PAIN3220
 PAIN3230
 PAIN3240
 PAIN3250
 PAIN3260
 PAIN3270
 PAIN3280
 PAIN3290
 PAIN3300

PAIN3330
 PAIN3340
 PAIN3350

PAIN3360
 PAIN3370
 PAIN3380
 PAIN3390

```

      AR(3)=X13*TR(2,1)+Y13*TR(2,2)+Z13*TR(2,3)
      AR(4)=ZL/2.
C      WRITE(6,355) L
C      WRITE(6,354)AR(1),AR(2),AR(3),AR(4)
354  FORMAT(4E12.3)
355  FORMAT(17H COORD CHECK FOR:,I5,E12.4)
      CHECK=ABS(AR(1)-2.*AR(3)-AR(2))
      WRITE(6,355) L,CHECK
      IF (CHECK .GT. 0.01) GO TO 122
      PRINT 73, L
      GO TO 1000
C      CALCULATE REINFORCEMENT TRANSFORMATION MATRIX AND PERCENT REINFOR-
C      CEMENT PER LAYER
122  NTYPE = NTS(L)
      IF (NTYPE .EQ. 0) GO TO 127
      NSLRS = SLR(1,NTYPE)
      CALL RZERO(TS,3,3,2,3)
      CALL INTVAL(SR,6,1,0)
      DO 124 I = 1,NSLRS
      II = 5*I
      NSPL = 2
      IF (SLR(II,NTYPE) .EQ. 0.) NSPL = 1
      SR(I) = NSPL
      DO 123 J = 1,NSPL
      JJ = 6*I
      IF (J .EQ. 1) JJ = JJ - 2
      TS(1,1,J,I) = (COS(SLR(JJ,NTYPE)))**2
      TS(1,2,J,I) = (SIN(SLR(JJ,NTYPE)))**2
      TS(2,1,J,I) = TS(1,2,J,I)
      TS(2,2,J,I) = TS(1,1,J,I)
      TS(1,3,J,I) = -2.*COS(SLR(JJ,NTYPE))*SIN(SLR(JJ,NTYPE))
      TS(2,3,J,I) = -TS(1,3,J,I)
      TS(3,1,J,I) = TS(2,3,J,I)/2.
      TS(3,2,J,I) = TS(1,3,J,I)/2.
      TS(3,3,J,I) = TS(1,1,J,I) - TS(1,2,J,I)
123  CONTINUE
124  CONTINUE
      DO 126 I = 1,NSLRS
      II = (I-1)*6 + 2
      LN = SLR(II,NTYPE)
      SR(I+3) = LN
      PS(1,I)=SLR(II+1,NTYPE)
      PS(2,I)=SLR(II+3,NTYPE)
126  CONTINUE
127  CONTINUE
      IF (NTYPE .EQ. 0) NSLRS = 0
      CALL CONST(L,NSLRS,NL,NUMEL)
      CALL STIFF
      I2=L
      DAT(1,L) = I1
      DAT(2,L) = I2
      WRITE(NTP1) CELCON
      CALL RZERO(SSR,2,3,1,1)
      WRITE(NTP6A) CELVAR
      WRITE (NTP2*I2) ST
      NLS=NDIS(L)
      IF (NLS .EQ. 0) GO TO 145
      R(1)=DFOR(NLS,1)
      R(2)=DFOR(NLS,2)
      R(3)=DFOR(NLS,3)

```

MAIN3680

MAIN3690

MAIN3700

MAIN3710

MAIN3720

MAIN3730

MAIN3740

MAIN3820

MAIN3830

MAIN3840

MAIN3850

MAIN3860

MAIN3870

MAIN3890

MAIN3900

MAIN3910

MAIN3920

MAIN3930

MAIN3940

MAIN3950

MAIN3960

MAIN3970

MAIN3980

MAIN3990

MAIN4000

MAIN4010

MAIN4030

MAIN4040

MAIN4050

MAIN4060

MAIN4070

MAIN4080

MAIN4090

MAIN4120

MAIN4140

MAIN4170

MAIN4180

```

R(4)=DFOR(NLS,4)
R(5)=DFOR(NLS,5)
WRITE (6,350) L
C 350 FFORMAT(30H STIFFNESS MATRIX FOR ELEMENT:,I2)
351 FFORMAT(6E12.4)
CALL RZERO(XM,15,1,1,1)
SLP1=AR(3)/AR(2)
SLP2=(AR(3)-AR(1))/AR(2)
AA=AR(2)*AR(1)*0.5
AX=AR(1)*AR(2)*AR(2)/6.
AY=AR(2)*AR(1)*(AR(1)+AR(3))/6.
AXX=AR(2)*AR(2)*AR(2)*AR(1)/12.
AYY=AR(2)*AR(1)*(AR(1)+AR(1)+AR(3)+AR(3)*AR(3))/12.
AXY=AR(2)*AR(2)*AR(1)*(AR(1)+2.*AR(3))/24.
AXXX=AR(2)*AR(2)*AR(2)*AR(2)*AR(1)/20.
AYYY=(AR(2)*AR(1)/60.)*(33.*AR(1)*AR(1)*AR(1)
1 -27.*AR(1)*AR(1)*AR(3)+3.*AR(1)*AR(3)*AR(3)
2 +3.*AR(3)*AR(3)*AR(3))
AXXY=0.5*((SLP2*SLP2*AR(2)**5/5.)+(SLP2*AR(1)*AR(2)**4/2.)
1 +(AR(1)*AR(1)*AR(2)**3/3.)-(SLP1*SLP1*AR(2)**5/5.))
AYYX=(1./3.)*((SLP2**3*AR(2)**5/5.)+(3.*SLP2*SLP2*AR(2)**4/4.)
1 +(SLP2*AR(1)*AR(1)*AR(2)**3)-(SLP1**3*AR(2)**5/5.)
2 +(AR(2)*AR(2)*AR(1)**3/2.))
C WRITE(6,351)AXXX,AXXY,AYYX,AYYY
X4(1) = AA*R(1)
X4(2) = AX*R(1)
X4(3) = AY*R(1)
X4(4) = AA*R(2)
X4(5) = AX*R(2)
X4(6) = AY*R(2)
X4(7) = AA*R(3)
X4(8)=AX*R(3)-AA*R(5)
X4(9) = AY*R(3) + AA*R(4)
X4(10) = AXX*R(3) - 2.*AX*R(5)
X4(11) = AXY*R(3) + AX*R(4) - AY*R(5)
X4(12) = AYY*R(3) + 2.*AY*R(4)
X4(13)=AXXX*R(3) - 3.*AXX*R(5)
X4(14) = (AXXY+AYYX)*R(3) + (AXX+2.*AXY)*R(4) -(2.*AXY+AYY)*R(5)
X4(15) = AYYY*R(3) + 3.*AYY*R(4)
CALL DRZERO(AL,18,1)
DO 138 I = 1,15
II = NEN(I)
DO 138 J = 1,15
138 AL(II) = AL(II) + A(J,I)*XM(J)
C TRANSFORM ELEMENT LOAD VECTOR
DO 142 I = 1,6
II = (I - 1)*3
DO 140 J = 1,3
S1(J) = 0.0
DO 140 JJ = 1,3
140 S1(J) = S1(J) + TR(JJ,J)*AL(II+JJ)
DO 141 J = 1,3
141 AL(II+J) = S1(J)
142 CONTINUE
C ADD ELEMENT LOAD VECTOR TO NODAL LOAD VECTOR
DO 144 I = 1,3
II = (KQ(I,L) - 1)*6
DO 143 J = 1,6
JJ = (I - 1)*6

```

MAIN4660
 MAIN4670
 MAIN4680
 MAIN4690
 MAIN4700
 MAIN4710
 MAIN4720

MAIN4740
 MAIN4750
 MAIN4760
 MAIN4770

MAIN4840
 MAIN4850
 MAIN4860
 MAIN4870
 MAIN4880
 MAIN4890
 MAIN4900
 MAIN4910
 MAIN4920
 MAIN4930
 MAIN4940
 MAIN4950
 MAIN4960
 MAIN4970
 MAIN4980
 MAIN4990
 MAIN5000
 MAIN5010
 MAIN5020

143	ALOAD(II+J) = ALOAD(II+J) + AL(JJ+J)	MAIN5030
144	CONTINUE	MAIN5040
145	CONTINUE	
	NM=NMAT(L)	
	DEN=DENS(NM)	
	WT(L)=DEN*AA*T(L)/3.0	
C	ADD ELEMENT WEIGHTS TO NODAL LOAD VECTOR	MAIN5050
	DO 146 I = 1,3	MAIN5070
	II = (KQ(I,L) - 1)*6 + 3	MAIN5080
145	ALOAD(II) = ALOAD(II) - WT(L)	MAIN5090
147	CONTINUE	MAIN5100
128	CONTINUE	MAIN4210
	REWIND NTP1	
	REWIND NTP6A	
C		
C	INPUT BOUNDARY CONDITIONS	MAIN4220
C		
	CALL INTVAL(IVAL,6,NUMPTS,0)	
	PRINT 39	MAIN4260
	READ 9, NCDS	MAIN4270
	PRINT 40	MAIN4280
	DO 130 N = 1, NCDS	MAIN4290
	READ 41, N1,N2,N3,(KOND(I),I=1,3)	MAIN4300
	PRINT 42, N1,N2,N3,(KOND(I),I=1,3)	MAIN4310
	IF (N2 .EQ. 0) N2 = N1	MAIN4320
	IF (N3 .EQ. 0) N3 = 1	MAIN4330
	DO 130 J = N1,N2,N3	MAIN4340
	DO 130 K = 1,3	MAIN4350
	KK = K	MAIN4360
	IF (KOND(K) .EQ. 0) GO TO 130	MAIN4370
	IF (KOND(K) .EQ. 2) KK=K+3	MAIN4380
	IVAL(KK,J) = 1	MAIN4390
130	CONTINUE	MAIN4400
	PRINT 43	MAIN4410
C		
C	INPUT NODAL LOADS	MAIN5110
C		
	PRINT 49	MAIN5120
	READ 9, NCDS	MAIN5130
	IF (NCDS .NE. 0) GO TO 148	MAIN5140
	PRINT 50	MAIN5150
	GO TO 157	MAIN5160
148	PRINT 51	MAIN5170
	DO 150 N = 1, NCDS	MAIN5180
	READ 52, N1,N2,N3,(R(I),I=1,6)	MAIN5190
	PRINT 53, N1,N2,N3,(R(I),I=1,6)	MAIN5200
	IF (N2 .EQ. 0) N2 = N1	MAIN5210
	IF (N3 .EQ. 0) N3 = 1	MAIN5220
	DO 150 J = N1,N2,N3	MAIN5230
	JJ = (J - 1)*6	MAIN5240
	DO 149 K = 1,6	MAIN5250
	JJ = JJ + 1	MAIN5260
149	ALOAD(JJ) = ALOAD(JJ) + R(K)	MAIN5270
150	CONTINUE	MAIN5280
157	CONTINUE	MAIN5290
C		
C	INPUT LOAD INCREMENTS	MAIN5300
C		
	PRINT 54	MAIN5310
	READ 55, NLI,NCDS	MAIN5320


```

NTYPE=NTS(1)
NSLRS=SLR(1,NTYPE)
CALL CCNST5(1,NL,CD)
CALL STIFF2
I2 = DAT(2,I)
WRITE(NTP2,I2) ST
181 CONTINUE
IF (IREAD .GT. 0) WRITE(NTP6B) CELVAR
IF (IREAD .LT. 0) WRITE(NTP6A) CELVAR
IF (IFEXEL .EQ. 0) GO TO 170
DO 165 J = 1,3
KK = (KQ(J,I) - 1)*6
KJ = (J-1)*6 + 1
KJJ = KJ + 5
DO 165 M = KJ,KJJ
KK = KK + 1
CLOAD(KK)=CLOAD(KK)+EF(M)
SUM = SUM + CLOAD(KK)*CLOAD(KK)
165 CONTINUE
723 FORMAT(6E12.3)
170 CONTINUE
WRITE(6,723) (CLOAD(NF),NF=1,NQ)
REWIND NTP1
REWIND NTP6A
REWIND NTP6B
IREAD=IREAD*NREAD
EFNORM = SQRT(SUM)
CHKFAC = 0.1*FNORM
CHKSOL = 0.01*FNORM
WRITE(6,6600)
6600 FORMAT(13H FORCE NORMS:)
WRITE(6,351) FNORM,EFNORM
IF (EFNORM .EQ. 0.0) GO TO 187
NI = 2
CALL ASSEM(NC,NBAND,NR,NQ,AK,NUMEL,NTP2,NI,CLOAD,KC,IVAL,IEL,
*NUMPTS,61000)
ISOLVE=1
IFACT=1
IF (EFNORM .LT. CHKSOL) ISOLVE=0
IF (EFNORM .LT. CHKFAC) IFACT=0
IF (ISOLVE .EQ. 0) GO TO 187
MIT = MIT + 1
NI=2
WRITE(6,5000) MIT
5000 FORMAT(1H , ' EQNS. SOLVED AGAIN.MIT=',12)
IF (IFACT .EQ. 1) CALL FACTOR(NBAND,NC,NR,NC,AK,AB)
CALL SOLVE(NBAND,NQ,NR,NC,CLOAD,AK,AB)
DO 185 J=1,NQ
U(J)=U(J)+DL(J)
185 CONTINUE
CALL DRZERO(CLOAD,NQ,1)
IF (MIT .LE. NIT) GO TO 161
PRINT 75
GO TO 1000
C IF THE NUMBER OF ITERATIONS ON ONE LOAD INCREMENT IS GREATER THAN
C NIT, FLAG PROGRAM. IF THE DISPLACEMENTS HAVE CONVERGED TO WITHIN
C 5 PERCENT OF THE PREVIOUS DISPLACEMENTS, GO TO NEXT ITERATION
C PRINT DISPLACEMENTS
187 PRINT 60,IT
PRINT 61

```

MAIN5880
MAIN5890

MAIN5930
MAIN5940
MAIN5950
MAIN5960
MAIN5970
MAIN5980
MAIN5990

MAIN6010

MAIN6050

MAIN6180
MAIN6190
MAIN6200
MAIN6210
MAIN6220
MAIN6230
MAIN6250
MAIN6260
MAIN6270

DO 190 J = 1, NUMPTS	MAIN6280
JJ = (J-1)*6 + 1	MAIN6290
JK = JJ + 5	MAIN6300
M = 0	MAIN6310
DO 189 K = JJ, JK	MAIN6320
M = M + 1	MAIN6330
193 DD(M) = U(K)	MAIN6340
PRINT 62, J, (DD(M), M=1, 6)	MAIN6350
190 CONTINUE	MAIN6360
C PRINT STRESSES	MAIN6370
PRINT 63	MAIN6380
DO 194 I=1, NUMEL	
READ(NTP1) CELCON	
IF (IREAD .GT. 0) READ(NTP6A) CELVAR	
IF (IREAD .LT. 0) READ(NTP6B) CELVAR	
721 NLRS = NL(I)	MAIN6400
NTYPE = NTS(I)	MAIN6440
PRINT 64, I	MAIN6450
DO 194 J = 1, NLRS	MAIN6460
PRINT 65, J, (STT(I, J), I=1, 3)	MAIN6470
IF (NTYPE .EQ. 0) GO TO 194	MAIN6480
NSLRS = SLR(1, NTYPE)	MAIN6490
DO 191 K = 1, NSLRS	
IND=(K-1)*6+2	
LYR=SLR(IND, NTYPE)	
IF (LYR .EQ. J) GO TO 192	
191 CONTINUE	
GO TO 194	
192 INDR=6*K+1	
NSPL=SLR(INDR, NTYPE)	
PRINT 66, (SSTR(KK, K), KK=1, NSPL)	MAIN6540
194 CONTINUE	MAIN6550
REWIND NTP1	
REWIND NTP6A	
REWIND NTP6B	
C PRINT OUT ELEMENT LAYERS THAT HAVE EXCEEDED A TRANSITION CRITERIA	MAIN6560
195 CONTINUE	MAIN6570
PRINT 67	MAIN6580
DO 200 I = 1, NUMEL	MAIN6590
II = DAT(1, I)	MAIN6600
READ(NTP1) CELCON	
NLRS = NL(I)	MAIN6620
IF (IPRINT (I) .EQ. 0) GO TO 200	MAIN6630
PRINT 68, I	MAIN6640
DO 200 J = 1, NLRS	MAIN6650
IJ = ITRAN(J, I)	MAIN6660
ICON = 0	MAIN6670
IF (ITR(J, I) .EQ. ITRAN(J, I)) ICON = ICON + 1	MAIN6680
IF (ICON .EQ. 0) PRINT 69, J, IJ	MAIN6690
NTYPE = NTS(I)	MAIN6700
IF (NTYPE .EQ. 0) GO TO 200	MAIN6710
NSLRS = SLR(1, NTYPE)	MAIN6720
DO 196 K = 1, NSLRS	MAIN6730
KN = 6*(K-1) + 2	
IF (J .EQ. SLR(KN, NTYPE)) GO TO 197	
196 CONTINUE	MAIN6750
GO TO 200	MAIN6760
197 IN = 6*K + 1	
NSPL = SLR(IN, NTYPE)	
DO 199 KK = 1, NSPL	MAIN6780

```

      KJ = (K-1)*2 + KK
      IF (IST(KJ,1) .NE. 2) GO TO 198
      RD(KK) = YES
      GO TO 199
198  RD(KK) = NO
199  CONTINUE
      IF (ICON .NE. 0) PRINT 74, J, (RD(KKK),KKK=1,NSPL)
      IF (ICON .EQ. 0) PRINT 74,J, (RD(KKK),KKK=1,NSPL)
200  CONTINUE
201  CONTINUE
      REWIND NTP1
      DO 210 I = 1,10
      DO 210 J = 1,NUMEL
      ITR(I,J) = ITRAN(I,J)
210  CONTINUE
1000 STOP
      END
*****
C
C      S U B R O U T I N E   E X F C R
C      THIS SUBROUTINE PERFORMS THE FOLLOWING CALCULATIONS:
C      1) CHECK EACH LAYER TO SEE IF IT HAS CHANGED MATERIAL PROP-
C      ERTIES.
C      2) CALCULATE THE CHANGE IN MATERIAL PROPERTIES
C      3) CALCULATE THE AMOUNT OF EXCESS (INITIAL) STRESS
C      4) CALCULATE THE EXCESS STRESS RESULTANT
C
C      *****
C      SUBROUTINE EXFOR(L,NL,NE,NTS,SLR,CD,IFEXEL,IFEX)
C      DOUBLE PRECISION AA,STML,TRR,AR,D,E,F,EB
C      DOUBLE PRECISION TS,PS,SR,CELCON
C      COMMON/MAT/FCS(4),ECS(4),FDS(4),EDS(4),FTS(4),VS(4),EUS(4),FY,
C      ES,DENS(4),BETA,NMAT(300)
C      COMMON/ELCON/AA(15,15),TRR(3,3),AR(4),TS(3,3,2,3),PS(2,3),SR(6)
C      COMMON/ELVAR/EPTOT(3,10),CC(3,3,10),CS(3,3,6),ST(3,10),SSTR(2,3)
C      COMMON/STRINC/EPP(3,10),STP(3,10),EP(3,10)
C      COMMON/CST/TJ(11,300),CON(3,3)
C      COMMON/STIF/STML(18,18),D(3,3),E(3,3),F(3,3)
C      COMMON/EXFR/ITRAN(10,300),IST(6,300),EXF(18),EF(18),IT(10,300)
C      COMMON/STRES/BB(6,15),STR(5,10),EST(5,10),FS(3,6)
C      INTEGER SR
C      REAL*4 M
C      DIMENSION NL(300),NTS(300),SLR(19,2),IS(6),CD(3,3,10),EXS(
*3),CN(3,3),STN(3),BA(15,6),ESS(3),NEN(15),T(3,3),DEP(3),DSP(3),
*STL(3),ESTL(3),EXR(6)
C      DIMENSION CINVC(3,3),CINV(3,3)
C      DATA PI/3.1415926/
C      DATA SQ2/1.417424/,NEN/1,2,3,4,5,7,8,9,10,11,13,14,15,16,17/
C      NM=NMAT(L)
C      FC=FCS(NM)
C      EC=ECS(NM)
C      FD=FDS(NM)
C      ED=EDS(NM)
C      FT=FTS(NM)
C      V=VS(NM)
C      EU=EUS(NM)
C      DEN=DENS(NM)
C      *****
C
C
C

```

MAIN6790
 MAIN6800
 MAIN6810
 MAIN6820
 MAIN6830
 MAIN6840
 MAIN6850
 MAIN6860
 MAIN6870
 MAIN6880

 MAIN6890
 MAIN6900
 MAIN6910
 MAIN6920

 MAIN6940

EXFR0090

 EXFRC120

 EXFRC140

```

C
C
C
CHECK TRANSITION CRITERION FOR CONCRETE
EXFRC150

*****
IFEXEL = 0
ALPHA = FT/FD
NLR = NL(L)
EXFRC170
EXFRC180

C
C
CHECK EACH LAYER TO SEE WHICH TRANSITION CRITERION IS APPLICABLE

DO 30 J = 1,NLR
SIG2 = ABS(0.10*STR(5,J))
IF (STR(4,J) .LT. SIG2 .AND. STR(4,J) .GT. 0.0) STR(4,J)=0.0
I = ITRAN(J,L)
GO TO (1,1,30,4,5,30,7,8,9,30) , I
1 IF (STR(4,J) .LE. 0.0) GO TO 7
IF ( STR(5,J) .LT. 0.0 ) GO TO 4
*****
EXFRC210
EXFRC220

C
C
C
CHECK TRANSITION CRITERIA BETWEEN ELASTIC AND CRACKED REGIONS
EXFRC270

*****
TR= STR(1,J) + SQ2*(1.-ALPHA)*STR(2,J)/(1.+ALPHA) - SQ2*2.*ALPHA
**FD/(3.*(1.+ALPHA))
IF (TR .LT. 0.) GO TO 30
IT(J,L) = 1
ITRAN(J,L) = 2
IF (I .EQ. 2) ITRAN(J,L) = 3
IFEX = IFEX + 1
IFEXEL = 1
GO TO 30
*****
EXFRC280
EXFRC290

C
C
C
CHECK FOR CRACKS IN TENSION-COMPRESSION

*****
4 ITRAN(J,L)=4
5 TR = (STR(4,J)/FT)**2 + (STR(5,J)/FD)**2 -1.0
IF ( TR .LT. 0.0 ) GO TO 30
ITRAN(J,L)=5
IF ( I .EQ. 5 ) ITRAN(J,L)=6
IFEX = IFEX+1
IFEXEL = 1
IT(J,L) = 1
GO TO 30
7 ITRAN(J,L) = 7
*****
EXFRC790

C
C
C
CHECK TRANSITION CRITERIA BETWEEN FIRST AND SECOND ELASTIC REGIONSEXFRC400

*****
TR = +STR(1,J) + SQ2*(BETA-1.)*STR(2,J)/(2.*BETA-1.)-SQ2*BETA*FC/
*(3.*(2.*BETA-1.))
IF (TR .LT. 0.0) GO TO 30
ITRAN(J,L) = 8
IT(J,L) = 1
IFEX = IFEX + 1
IFEXEL = 1
GO TO 30
*****
EXFRC410
EXFRC420
EXFRC470
EXFRC480
EXFRC490
EXFRC500

```

```

C
C CHECK TRANSITION CRITERIA BETWEEN ELASTIC AND PLASTIC REGIONS EXFRC510
C
C *****
8 TR = STR(1,J) + SQ2*(BETA-1.)*STR(2,J)/(2.*BETA-1.) - SQ2*BETA* EXFRC820
  *FD/(3.*(2.*BETA-1.)) EXFR0530
  IF (TR .LT. 0.0) GO TO 30
  ITRAN(J,L) = 9 EXFR0580
  IT(J,L) = 1 EXFR0590
  IF (I .EQ. 2) ITRAN(J,L) = 9 EXFR0600
  IFEX = IFEX + 1 EXFR0610
  IFEXEL = 1
6 GO TO 30 EXFR0620
C *****
C CHECK TRANSITION CRITERIA BETWEEN PLASTIC AND CRUSHED REGIONS EXFRC630
C
C *****
9 TR = EST(1,J) + SQ2*(BETA-1.)*EST(2,J)/(2.*BETA-1.)-SQ2*BETA*EU EXFRC640
  * / (3.*(2.*BETA-1.)) EXFR0650
  IFEX = IFEX+1
  IFEXEL = 1
  IF (TR .LT. 0.0) GO TO 30
  ITRAN(J,L) = 10 EXFR0700
  IT(J,L) = 1 EXFR0710
  IFEX = IFEX + 1 EXFR0720
  IFEXEL = 1
30 CONTINUE EXFR0830
C *****
C CHECK REINFORCEMENT FOR YIELDING EXFRC840
C
C *****
NTYPE = NTS(L) EXFRC850
IF (NTYPE .EQ. 0) GO TO 41 EXFRC860
NSLS = SLR(1,NTYPE) EXFRC870
DO 40 J=1,NSLS EXFRC880
  NSPL=SLR(6+J+1,NTYPE)
  DO 40 K = 1,NSPL EXFRC900
    STRMAG=ABS(SSTR(K,J))
    IF (STRMAG .LT. FY) GO TO 40
    KJ = (J-1)*2 + K EXFRC920
    IF (IST(KJ,L) .EQ. 2) GO TO 40 EXFRC930
    IST(KJ,L) = 2 EXFRC940
    IS(KJ) = 1 EXFPC950
    IFEX = IFEX + 1 EXFRC960
    IFEXEL = 1
40 CONTINUE EXFRC970
41 CONTINUE
IF (IFEXEL .EQ. 0) RETURN EXFR0980
C *****
C CALCULATE DIFFERENCE BETWEEN OLD AND NEW MATERIAL PROPERTIES IN EXFRC990
C CONCRETE AND REINFORCEMENT. CALCULATE EXCESS STRESSES IN ELEMENT EXFR1000
C
C *****

```

```

CALL RZERO(CD,3,3,NLR,1)
CALL RZERO(EXS,3,1,1,1)
CALL RZERO(EXF,18,1,1,1)
CALL RZERO(EXR,6,1,1,1)
JZ=1
DO 105 J = 1,NLR
  I = ITRAN(J,L)
  PRINT 500, I
500 FORMAT(3H I=,I2)
  IF(I .EQ. 2 .AND. IT(J,L) .EQ. 1)GO TO 44
  IF(I .EQ. 3 .AND. IT(J,L) .EQ. 1)GO TO 44
  IF(I .EQ. 5 .AND. IT(J,L) .EQ. 1) GO TO 555
  IF(I .EQ. 6 .AND. IT(J,L) .EQ. 1) GO TO 555
C   IF (I .EQ. 9 .AND. IT(J,L) .EQ. 1) GO TO 46
  IF(I .EQ. 8 .AND. IT(J,L) .EQ. 1)GO TO 47
  IF (I .EQ. 9 .AND. IT(J,L) .EQ. 1) GO TO 50
  IF ( I .EQ. 9 ) GO TO 82
  GO TO 95
C *****
C
C   ADJUST MATERIAL PROPERTIES AND CALCULATE EXCESS STRESS FOR
C   CRACKS IN BIAXIAL TENSION
C *****
44 CONTINUE
  CALL CCNST2(L,CN,I,J,STL)
  DO 45 K = 1,3
  DO 45 KK = 1,3
  PRINT 810, K, KK, J, CC(K, KK, J)
810 FORMAT(4H CC(,I2,I2,I2,2H)=,E15.7)
  PRINT 809, K, KK, CN(K, KK)
809 FORMAT(4H CN(,I2,I2,2H)=,E15.7)
  CD(K, KK, J)=CN(K, KK)-CC(K, KK, J)
  45 CC(K, KK, J)=CN(K, KK)
  IF (STR(4, J) .LT. 0.1) GO TO 8000
  M=STR(5, J)/STR(4, J)

```

EXFR1090
EXFR1100

EXFR1110

EXFR1150
EXFR1160
EXFR1170

```

C1=SQ2*(1.0-ALPHA)/(1.0+ALPHA)
C2=2.0*SQ2*ALPHA/(3.0*(1.0+ALPHA))
YP1=C2*FD/((C1/3.)*(M+1.)+SQRT(M*M-M+1.)*(SQ2/3.))
YP2=M*YP1
GO TO 8100
8000 YP1=FT
      YP2=0.0
8100 CONTINUE
      WRITE(6,1111)
      WRITE(6,1100) YP1,YP2
      THETA = -STR(3,J)
      STL(1)=(YP1+YP2)/2.+(YP1-YP2)*COS(2.*THETA)/2.
      STL(2)=(YP1+YP2)/2.-(YP1-YP2)*COS(2.*THETA)/2.
      STL(3)=(YP1-YP2)*SIN(2.*THETA)/2.
      PS1=STR(4,J)
      PS2=0.0
      CALL STRES4(PS1,PS2,J)
      DO 3451 K=1,3
        STN(K) = ST(K,J)
3451 CONTINUE
      PS1=0.0
      PS2=YP2
      CALL STRES4(PS1,PS2,J)
      GO TO 95
*****
C
C
C   ADJUST MATERIAL PROPERTIES AND CALCULATE EXCESS STRESS FOR
C   CRACKS IN TENSION - COMPRESSION
C
C *****
555 CALL CNST2(L,CN,I,J,STL)
      DO 556 K=1,3
        DO 556 KK=1,3
          CD(K,KK,J)=CN(K,KK)-CC(K,KK,J)
          CC(K,KK,J)=CN(K,KK)
556 CONTINUE
      IF (STR(4,J) .LT. 0.1) GO TO 8200
      M=STR(5,J)/STR(4,J)
      IF (M .GT. 500.) GO TO 8400
      YP1=1.0/(SQRT((1.0/FT)*(1.0/FT)+(M/FD)*(M/FD)))
      YP2=M*YP1
      GO TO 8300
8200 YP1=FT
      YP2=0.0
      GO TO 8300
8400 YP1=0.0
      YP2=-FD
8300 CONTINUE
      WRITE(6,1111)
      WRITE(6,1100) YP1,YP2
      THETA=-STR(3,J)
      STL(1)=(YP1+YP2)/2.+(YP1-YP2)*COS(2.*THETA)/2.
      STL(2)=(YP1+YP2)/2.-(YP1-YP2)*COS(2.*THETA)/2.
      STL(3)=(YP1-YP2)*SIN(2.*THETA)/2.
      PS1=STR(4,J)
      PS2=0.0
      CALL STRES4(PS1,PS2,J)
      DO 3456 K=1,3
        STN(K)= ST(K,J)
3456 CONTINUE

```

EXFR1200


```

PS1=0.0
PS2=YP2
CALL STRES4(PS1,PS2,J)
GO TO 95
C *****
C   ADJUST MATERIAL PROPERTIES AND CALCULATE EXCESS STRESS FOR
C   THE TRANSITION FROM ELASTIC 1 TO ELASTIC 2
C *****
47 IF (STR(4,J) .EQ. 0.0) GO TO 600
M = STR(5,J)/STR(4,J)
WRITE(6,1100) M
IF (M .GT. 100.0) GO TO 600
C1=SQ2*((BETA-1.0)/(2.0*BETA-1.0))
C2=(SQ2/3.0)*(BETA/(2.0*BETA-1.0))
YP1=(C2*FC)/((C1/3.0)*(M+1.0)-SQRT(M*M-M+1.0)*(SQ2/3.0))
YP2 = M*YP1
WRITE(6,1111)
WRITE(6,1100) YP1,YP2
GO TO 601
600 YP1 = 0.0
    YP2 = -FC
601 THETA = -STR(3,J)
C PRINT 706, THETA
STL(1) = (YP1+YP2)/2. + (YP1-YP2)*CCS(2.*THETA)/2.
STL(2) = (YP1+YP2)/2. - (YP1-YP2)*CCS(2.*THETA)/2.
STL(3) = (YP1-YP2)*SIN(2.*THETA)/2.
WRITE(6,1111)
WRITE(6,1100) YP1,YP2
1111 FORMAT(24H POINT ON YIELD SURFACE:)
PRINT 700, STL(1)
PRINT 701, STL(2)
PRINT 702, STL(3)
CALL CCNST2(L,CN,I,J,STL)
DO 48 K = 1,3
DO 48 KK = 1,3
PRINT 810, K, KK, J, CC(K, KK, J)
PRINT 809, K, KK, CN(K, KK)
CD(K, KK, J)=CN(K, KK)-CC(K, KK, J)
48 CC(K, KK, J)=CN(K, KK)
CINVC(1,1)=1.0-(ED/EC)
CINVC(2,2)=CINVC(1,1)
CINVC(3,3)=CINVC(1,1)
CINVC(1,2)=0.0
CINVC(1,3)=0.0
CINVC(2,1)=CINVC(1,2)
CINVC(2,3)=0.0
CINVC(3,1)=CINVC(1,3)
CINVC(3,2)=CINVC(2,3)
CALL STRES3(J,CN,STN,STL,CINVC,L)
GO TO 95
C *****
C   ADJUST MATERIAL PROPERTIES AND CALCULATE EXCESS STRESS FOR
C   ELEMENTS PASSING FROM ELASTIC 2 TO PLASTIC
C *****
50 IF (STR(4,J) .EQ. 0.0) GO TO 610
M = STR(5,J)/STR(4,J)
IF (M .GT. 100.0) GO TO 610
WRITE(6,1100) M
C1=SQ2*((BETA-1.0)/(2.0*BETA-1.0))
C2=(SQ2/3.0)*(BETA/(2.0*BETA-1.0))

```

EXFR1280

EXFR1290

EXFR1300

EXFR1210

```

      YP1=(C2*FD)/((C1/3.)*(M+1.)-SQRT(M*M-M+1.)*(SQ2/3.))
      YP2 = M*YP1
      GO TO 611
510  YP1 = 0.0
      YP2 = -FD
611  THETA = -STR(3,J)
      PRINT 706, THETA
706  FJRMAT(7H THETA=,E10.3)
      STL(1) = (YP1+YP2)/2. + (YP1-YP2)*CCS(2.*THETA)/2.
      STL(2) = (YP1+YP2)/2. - (YP1-YP2)*CCS(2.*THETA)/2.
      STL(3) = (YP1-YP2)*SIN(2.*THETA)/2.
      WRITE(6,1111)
      WRITE(6,1100) YP1,YP2
      PRINT 700, STL(1)
      PRINT 701, STL(2)
      PRINT 702, STL(3)
700  FJRMAT(8H STL(1)=,E10.3)
701  FJRMAT(8H STL(2)=,E10.3)
702  FJRMAT(8H STL(3)=,E10.3)
      CALL CCNST3(CN,J,STL)
      DO 1200 ID=1,3
      DSP(ID) = ST(ID,J) - STL(ID)
1200  CONTINUE
      DEP(1) = (DSP(1)-V*DSP(2))/ED
      DEP(2) = (DSP(2)-V*DSP(1))/ED
      DEP(3) = 2.*(1.0+V)*DSP(3)/ED
      PRINT 703, DEP(1)
      PRINT 704, DEP(2)
      PRINT 705, DEP(3)
703  FJRMAT(8H DEP(1)=,E10.3)
704  FJRMAT(8H DEP(2)=,E10.3)
705  FJRMAT(8H DEP(3)=,E10.3)
55  DO 56 II = 1,3
      DSP(II) = STL(II)
      DO 56 JJ = 1,3
      DSP(II) = DSP(II)+CN(II,JJ)*DEP(JJ)
56  PRINT 707, II,DSP(II)
707  FJRMAT(5H DSP(,12,2H)=,E10.3)
57  PS1 = (DSP(1)+DSP(2))/2. + SQRT(((DSP(1)-DSP(2))/2.)**2+DSP(3)**2) EXFR1400
      PS2 = (DSP(1)+DSP(2))/2. - SQRT(((DSP(1)-DSP(2))/2.)**2+DSP(3)**2) EXFR1410
      THETA = ATAN(DSP(3)*2./(DSP(1)-DSP(2)))/2. EXFR1420
      IF(ABS(THETA) .LT. .001 .AND. DSP(1) .LT. DSP(2))THETA=PI/2.
      PRINT 706, THETA
C      *****
C      ADJUST MATERIAL PROPERTIES AND CALCULATE EXCESS STRESS FOR
C      ELEMENTS IN THE PLASTIC RANGE
C      *****
82  IF ( IT(J,L) .NE. 1 ) PS1=STR(4,J)
      IF ( IT(J,L) .NE. 1 ) PS2=STR(5,J)
C      WRITE(6,1112)
C      WRITE(6,1100) (EP(KL,J),KL=1,3)
1112  FJRMAT(20H INCREMENTAL STRAIN:)
      M = PS2/PS1
      WRITE(6,1100) M
      C1=SQ2*((BETA-1.0)/(2.0*BETA-1.0))
      C2=(SQ2/3.0)*(BETA/(2.*BETA-1.0))
      YP1=(C2*FD)/((C1/3.)*(M+1.)-SQRT(M*M-M+1.)*(SQ2/3.))
1100  FJRMAT(3E13.5)
      YP2 = M*YP1
      WRITE(6,1111)

```

EXFR1260

EXFR1320

EXFR1360
EXFR1370
EXFR1380
EX

EXFR1400
EXFR1410
EXFR1420

EXFR1430

EXFR1480

```

WRITE(6,1100) YP1,YP2
STL(1) = (YP1+YP2)/2.+(YP1-YP2)*COS(2.*THETA)/2.
STL(2) = (YP1+YP2)/2.-(YP1-YP2)*COS(2.*THETA)/2.
STL(3) = (YP1-YP2)*SIN(2.*THETA)/2.
PRINT 700, STL(1)
PRINT 701, STL(2)
PRINT 702, STL(3)
CALL CCNST4(CN,J,STL)
DO 58 II = 1,3
STN(II) = ST(II,J)-STL(II)
58 CONTINUE
DO 60 KK = 1,3
DO 60 K = 1,3
CD(KK,K,J) = CN(KK,K)-CC(KK,K,J)
50 CC(KK,K,J) = CN(KK,K)
DO 61 II = 1,3
ST(II,J)=STL(II)
EPP(II,J) = EP(II,J)
61 STP(II,J) = ST(II,J)
GO TO 95
65 DO 66 II = 1,3
DEP(II) = EP(II,J)-EPP(II,J)
STL(II) = STP(II,J)
DO 66 K = 1,3
65 CN(II,K) = CC(II,K,J)
GO TO 55
CALL RZERO(CN,3,3,1,1)
DO 76 II = 1,3
DO 76 JJ = 1,3
CD(II,JJ,J) = CN(II,JJ) - CC(II,JJ,J)
75 CC(II,JJ,J) = CN(II,JJ)
DO 77 II = 1,3
77 STN(II) = ST(II,J) - STP(II,J)
GO TO 95
45 STN(1)=ST(1,J)
STN(2)=ST(2,J)
STN(3)=ST(3,J)
CALL CCNST2(L,CN,I,J,STL)
DO 49 K=1,3
DO 49 KK = 1,3
CD(K,KK,J)=CN(K,KK)-CC(K,KK,J)
49 CC(K,KK,J)=CN(K,KK)
C
C *****
C
C CALCULATE EXCESS STRESS RESULTANT
C
C *****
95 CONTINUE
WRITE(6,904) J , (STN(JK),JK=1,3)
904 FORMAT(25H EXCESS STRESS FOR LAYER:,I3,6E12.4)
80 NTYPE=NTS(L)
IF (NTYPE .EQ. 0) GO TO 9300
IF(SLR(2,NTYPE) .NE. J) GO TO 9000
NSPL=SLR(7,NTYPE)
NLAYER=1
GO TO 9200
9000 IF(SLR(8,NTYPE) .NE. J) GO TO 9100
NSPL=SLR(13,NTYPE)
NLAYER=2

```

EXFR1490
EXFR1500
EXFR1510

EXFR1530
EXFR1550
EXFR1560
EXFR1570
EXFR1640
EXFR1650
EXFR1660
EXFR1670
EXFR1680

EXFR1690
EXFR1700
EXFR1710
EXFR1720
EXFR1730
EXFR1740
EXFR1750
EXFR1760
EXFR1770

EXFR1810
EXFR1820
EXFR1830
EXFR1840
EXFR1850
EXFR1860
EXFR1870

```

GO TO 9200
9100 TPS=0.0
   IF(SLR(14,NTYPE) .NE. J) GO TO 9300
   NSPL=SLR(19,NTYPE)
   NLAYER=3
9200 IF(NSPL .EQ. 0) GO TO 9300
   TPS=0.0
   DO 120 K=1,NSPL
   TPS=TPS+PS(K,NLAYER)
   JZ=NLAYER
   KJ=(JZ-1)*2+K
   IF(IS(KJ) .NE. 1) GO TO 120
   DO 108 MM=1,3
   DO 108 N=1,3
   CD(MM,N,J)=CD(MM,N,J)-CS(MM,N,KJ)*PS(K,NLAYER)
108 CS(MM,N,KJ)=0.0
120 CONTINUE
   JZ=JZ+1
9300 CONTINUE
   DO 100 K=1,3
   EXR(K) = EXR(K) + (1.0-TPS)*STN(K)*(TJ(J,L)-TJ(J+1,L))
   EXR(K+3)=EXR(K+3)-(1.-TPS)*STN(K)*(TJ(J,L)**2-TJ(J+1,L)**2)*.5
100 CONTINUE
   WRITE(6,904) J , (EXR(JK),JK=1,6)
105 CONTINUE
*****
C
C
C   COMPUTE EXCESS NODAL FORCE VECTOR
C
C   *****
C   CALL RZERO(EF,18,1,1,1)
C   DO 122 I = 1,15
C   DO 122 J = 1,6
C   BA(I,J) = 0.0
C   DO 122 K = 1,15
C   DO 122 I = 1,15
C   DO 122 J = 1,6
C   IJ = NEN(I)
125 EF(IJ)=EF(IJ)+BA(I,J)*EXR(J)
C
C   TRANSFORM EXCESS NODAL FORCE VECTOR
C
C   DO 140 I=1,6
C   II=(I-1)*3
C   DO 130 J=1,3
C   ESS(J)=0.0
C   DO 130 JJ=1,3
130 ESS(J)=ESS(J)+TRR(JJ,J)*EF(II+JJ)
C   DO 135 J=1,3
C   EF(II+J)=ESS(J)
C   IJI=II+J
C   WRITE(6,904) IJI,EF(IJI)
135 CONTINUE
140 CONTINUE
   RETURN
   END
*****
C
C
C   S U B R O U T I N E   S T R E S S

```

EXFR2150

EXFR2180

EXFR2190

EXFR2200

EXFR2210

EXFR2220

EXFR2230

EXFR2240

EXFR2250

EXFR2270

EXFR2420

EXFR2430

```

C      THIS PURPOSE OF THIS SUBROUTINE IS TO CALCULATE THE
C      STRESS IN EACH LAYER OF AN ELEMENT
C
C      *****
C      SUBROUTINE STRESS(L,KQ,NE,SLR,NL,NTS,ITNO,MIT)
C      DOUBLE PRECISION A,S,DL,TR,AR,D,E,F,S1,BA,B,CLL,EO
C      DOUBLE PRECISION TS,PS,SR,CELCON
C      COMMON/MAT/FCS(4),ECS(4),FDS(4),EDS(4),FTS(4),VS(4),EUS(4),FY,
C      ES,DENS(4),BETA,NMAT(300)
C      COMMON/ELCON/A(15,15),TR(3,3),AR(4),TS(3,3,2,3),PS(2,3),SR(6)
C      COMMON/ELVAR/EPTOT(3,10),CC(3,3,10),CS(3,3,6),ST(3,10),SSTR(2,3)
C      COMMON/STRINC/EPP(3,10),STP(3,10),EP(3,10)
C      COMMON/CST/TJ(11,300),C(3,3)
C      COMMON/STIF/S(18,18),D(3,3),E(3,3),F(3,3)
C      COMMON/EXFR/ITRAN(10,300),IST(6,300),EXF(18),EF(18),IT(10,300)
C      COMMON/STRES/B(6,15),STR(5,10),EST(5,10),FS(3,6)
C      COMMON/DISP/ DL(1200),DLL(18),U(1200)
C      INTEGER SR
C      DIMENSION KQ(3,200), S1(3),BA(6,15),NTS(200),
C      SLR(19,2),CN(3,3),STN(3),TT(3,3),NL(300),CD(3,3)
C      DIMENSION SN(3),STL(3),EO(6)
C      DIMENSION CINVC(3,3)
C      DATA PI/3.1415926/
C      NLS = NL(L)
C      NM=NMAT(L)
C      FC=FCS(NM)
C      EC=ECS(NM)
C      FD=FDS(NM)
C      ED=EDS(NM)
C      FT=FTS(NM)
C      V=VS(NM)
C      EU=EUS(NM)
C      DEN=DENS(NM)
C      IF(ITNO.EQ.1.AND.MIT.EQ.1)CALL RZERO(EPTOT,3,10,1,1)
C      CALL DRZERO(B,6,15,1,1)
C      XCEN=AR(2)/3.0
C      YCEN=(AR(1)+AR(3))/3.0
C      WRITE(6,900)XCEN,YCEN
C      FORMAT(7H STRESS,3E12.3)
C      AA=0.5*AR(1)*AR(2)
C      AX=AR(1)*AR(2)*AR(2)/6.0
C      AY=AR(1)*AR(2)*(AR(1)+AR(3))/6.0
C      B(1,2) = AA
C      B(2,6) = AA
C      B(3,3) = AA
C      B(3,5) = AA
C      B(4,10) = +2.0*AA
C      B(4,13) = +6.0*AX
C      B(4,14) = +2.0*AY
C      B(5,12) = +2.0*AA
C      B(5,14) = +2.0*AX
C      B(5,15) = +6.0*AY
C      B(6,11) = +2.0*AA
C      B(6,14) = +4.0*(AX+AY)
C      K = 0
C      DO 2 I = 1,3
C      N = (KQ(I,L) - 1)*6 + 1
C      NN = N + 5
C      DO 2 J = N,NN
C      K = K + 1

```

STRS0100

STRS0110

STRS0180
STRSC190
STRS0200
STRSC210

STRS0240

STRS0260
STRS0270

STRS0290
STRS0300
STRSC310
STRS0320
STRSC330
STRSC340
STRS0350

```

      DLL(K)=DL(J)
2  CONTINUE
C  TRANSFORM ELEMENT DISPLACEMENTS
      DO 7 I = 1,6
      DO 801 IX=1,3
801  S1(IX)=0.
      K = (I - 1)*3 + 1
      KK = K + 2
      DO 4 J = 1,3
      JJ = K + J - 1
      4  S1(J) = DLL(JJ)
      DO 5 J = K, KK
      5  DLL(J) = 0.0
      M = 0
      DO 6 J = K, KK
      M = M + 1
      DO 6 N = 1,3
      6  DLL(J) = DLL(J) + TR(M,N)*S1(N)
      7  CONTINUE
C  *****
C  CALCULATE MIDDLE SURFACE STRAIN VECTOR
C  *****
      CALL DRZERO(E0,6,1)
      CALL DRZERO(BA,6,15)
      DO 20 I = 1,6
      DO 20 J = 1,15
      DO 20 K = 1,15
20  BA(I,J) = BA(I,J) + B(I,K)*A(K,J)/AA
      DO 22 I = 1,3
      DO 22 J = 1,5
      JJ = (I-1)*5 + J
      KK = (I-1)*6 + J
22  DLL(JJ) = DLL(KK)
      DO 25 I = 1,6
      DO 25 J = 1,15
      E0(I) = E0(I) + BA(I,J)*DLL(J)
25  CONTINUE
      GO TO 6789
      E0(3)=0.0
      E0(6)=0.0
      STRMAX=0.0
      CRVMAX=0.0
      DO 3000 IN=1,3
      STRN=ABS(E0(IN))
      CRV=ABS(E0(IN+3))
      IF(STRN.GT. STRMAX) STRMAX=STRN
      IF(CRV.GT. CRVMAX) CRVMAX=CRV
3000 CONTINUE
      DO 3100 IN=1,3
      STRN=ABS(E0(IN))
      CHKSTR=0.75*STRMAX
      CRV=ABS(E0(IN+3))
      CHKCRV=0.75*CRVMAX
      IF(STRN.LT. CHKSTR) E0(IN)=0.0
      IF(CRV.LT. CHKCRV) E0(IN+3)=0.0
3100 CONTINUE
6789 CONTINUE
      WRITE(6,400) L

```

```

      STRS0360
      STRSC370
      STRSC380
      STRSC390
      STRSC400
      STRSC410
      STRSC420
      STRSC430
      STRSC440
      STRSC450
      STRS0460
      STRSC470
      STRS0480
      STRSC490
      STRSC500
      STRSC510
      STRS0520
      STRS0530
      STRS0540

```

```

      STRSC550

```

```

      STRSC590
      STRSC600
      STRSC610
      STRS0620
      STRSC630
      STRSC640
      STRSC650
      STRSC660
      STRS0670
      STRS0700
      STRS0710

```

```

WRITE(6,401) (EO(I),I=1,6)
403 FORMAT(32H MID-SURFACE STRAIN FOR ELEMENT: ,I2)
401 FORMAT(6E14.6)

C
C   CALCULATE LAYER STRAINS
C
DO 30 I = 1,3
DO 30 K = 1,NLS
EP(I,K) = EO(I) - (TJ(K,L) + TJ(K+1,L))/2.*EC(I+3)
EPTOT(I,K) = EPTOT(I,K) + EP(I,K)
30 CONTINUE
*****
C
C   CALCULATE LAYER STRESSES
C
C   *****
IF(IITNO.EQ. 1 .AND. MIT.EQ. 1)CALL RZERC(ST,3,NLS,1,1)
DO 33 K = 1,NLS
DO 33 I = 1,3
DO 34 II = 1,3
ST(I,K) = ST(I,K) + CC(I,II,K)*EP(II,K)
34 CONTINUE
33 CONTINUE
*****
C
C   CALCULATE STRESSES IN REINFORCEMENT
C
C   *****
NTYPE=NTS(L)
IF(NTYPE.EQ. 0) GO TO 56
NSL=SLR(1,NTYPE)
DO 55 I=1,NSL
LN=6*(I-1)+2
LYR = SLR(LN,NTYPE)
IN=6*I+1
NSPL=SLR(IN,NTYPE)
DO 45 J=1,NSPL
JN=6*I-(2-J)*2
ANG=SLR(JN,NTYPE)
C1=cos(ANG)*cos(ANG)*EP(1,LYR)
C2=sin(ANG)*sin(ANG)*EP(2,LYR)
C3=sin(ANG)*cos(ANG)*EP(3,LYR)
STLMOD = ES
NYLD = 2*(I-1) + J
IF (IST(NYLD,L) .EQ. 2) STLMOD = 0.0
SSTR(J,I) = SSTR(J,I) + STLMOD*(C1+C2+C3)
45 CONTINUE
55 CONTINUE
56 CONTINUE
*****
C
C   CALCULATE PRINCIPAL LAYER STRESSES AND STRAINS
C   THE FOLLOWING QUANTITIES WILL BE STORED IN THE ARRAY STR:
C   STR(1,J):OCTAHEDRAL SHEAR STRESS FOR LAYER J
C   STR(2,J):HYDROSTATIC STRESS FOR LAYER J
C   STR(3,J):THE ANGLE BETWEEN THE LOCAL X-AXIS AND THE MAXIMUM
C   PRINCIPAL STRESS
C   STR(4,J):MAXIMUM PRINCIPAL STRESS FOR LAYER J
C   STR(5,J):MINIMUM PRINCIPAL STRESS FOR LAYER J
C   THE CORRESPONDING STRAIN VALUES ARE STORED IN THE ARRAY EST

```

STRS0730

STRS0740

STRSC750

STRS0760

STRSC770

STRSC790

STRS0810

STRSC830

STRS1030

```

C
C *****
PRINT 700 ,L
700 FORMAT(24H STR MATRIX FOR ELEMENT:,I3)
DO 60 I = 1,NLS
V1 = (ST(1,I)+ST(2,I))/2.
V2 = (ST(1,I)-ST(2,I))/2.
PS1 = V1 + SQRT(V2**2+ST(3,I)**2)
PS2 = V1 - SQRT(V2**2+ST(3,I)**2)
THETA = ATAN((2.*ST(3,I))/(ST(1,I)-ST(2,I)))/2.
SHEAR = ABS(ST(3,I))
IF(SHEAR .LT. 0.01 .AND. ST(1,I) .LT. ST(2,I))THETA=PI/2.
TOCT = SQRT(2.*PS1**2 - 2.*PS1*PS2 + 2.*PS2**2)/3.
P = (PS1 + PS2)/3.
STR(1,I) = TOCT
STR(2,I) = P
STR(3,I) = THETA
STR(4,I) = PS1
STR(5,I) = PS2
701 FORMAT(I5,6E12.4)
V1 = (EPTOT(1,I)+EPTOT(2,I))/2.
V2 = (EPTOT(1,I)-EPTOT(2,I))/2.
PS1 = V1 + SQRT(V2**2 + (EPTOT(3,I)/2.)**2)
PS2 = V1 - SQRT(V2**2 + (EPTOT(3,I)/2.)**2)
THETA = ATAN((EPTOT(3,I))/(EPTOT(1,I)-EPTOT(2,I)))/2.
IF(EPTOT(3,I) .EQ. 0. .AND. EPTOT(1,I) .LT. EPTOT(2,I))THETA=PI/2.
TOCT = SQRT(2.*PS1**2 - 2.*PS1*PS2 + 2.*PS2**2)/3.
P = (PS1 + PS2)/3.
EST(1,I) = TOCT
EST(2,I) = P
EST(3,I) = THETA
EST(4,I) = PS1
EST(5,I) = PS2
C PRINT 705, I,EST(1,I)
C PRINT 706, I,EST(2,I)
C PRINT 707, I,EST(3,I)
C PRINT 708, I,EST(4,I)
C PRINT 709, I,EST(5,I)
705 FORMAT(6H EST(1,I2,2H)=,E10.3)
706 FORMAT(6H EST(2,I2,2H)=,E10.3)
707 FORMAT(6H EST(3,I2,2H)=,E10.3)
708 FORMAT(6H EST(4,I2,2H)=,E10.3)
709 FORMAT(6H EST(5,I2,2H)=,E10.3)
60 CONTINUE
925 CONTINUE
RETURN
ENTRY STRES2(I,CD,STN)
DO 804 IX=1,3
804 STN(IX)=0.
DO 65 J = 1,3
DO 65 K = 1,3
65 STN(J) = STN(J) - CD(J,K)*EP(K,I)
RETURN
ENTRY STRES3(I,CN,STN,STL,CINVC,L)
DO 806 K=1,3
806 SN(K)=0.0
DO 67 K=1,3
DO 67 J=1,3
SN(K)=SN(K)+CN(K,J)*EPTOT(J,I)+CINVC(K,J)*STL(J)
67 CONTINUE

```

STRS1040
STRS1050
STRS1060

STRS1090

STRS1100
STRS1110
STRS1120
STRS1130
STRS1140
STRS1150
STRS1160

STRS1170
STRS1180

STRS1200
STRS1210

STRS1220
STRS1230
STRS1240
STRS1250
STRS1260
STRS1270
STRS1280

STRS1290

STRS1300

STRS1320
STRS1330
STRS1340
STRS1350

STRS1370


```

DO 68 J=1,3
STN(J)=ST(J,I)-SN(J)
ST(J,I)=SN(J)
68 CONTINUE
RETURN
ENTRY STRES4(PS1,PS2,J)
THETA=-STR(3,LN)
ST(1,J)=(PS1+PS2)/2. + (PS1-PS2)*CCS(2.*THETA)/2.
ST(2,J)=(PS1+PS2)/2. - (PS1-PS2)*COS(2.*THETA)/2.
ST(3,J)=(PS1-PS2)*SIN(2.*THETA)/2.
RETURN
END
*****
C      S U B R O U T I N E   A S S E M
C      THIS SUBROUTINE ASSEMBLES THE GLOBAL STIFFNESS MATRIX
C      AND APPLIES BOUNDARY CONDITIONS TO THE SYSTEM
C      *****
C      SUBROUTINE ASSEM(NC,NBAND,NR,NQ,AK,NUMEL,NTP2,NI,BLOAD,KQ,
C      *IVAL,IEL,NUMPTS,*)
C      DOUBLE PRECISION A,ST,AK,BLOAD,TRM,AR,D,E,F
C      DOUBLE PRECISION TS,PS,SR,CELCON
COMMON/ELCON/A(15,15),TRM(3,3),AR(4),TS(3,3,2,3),PS(2,3),SR(6)
COMMON/STIF/ST(18,18),D(3,3),E(3,3),F(3,3)
COMMON /DA/ DAT(2,300)
COMMON /NUNIT/NSTIFF,ICOUNT
DIMENSION AK(NR,NC),KQ(3,300),IL(18),IVAL(6,200),IEL(300),
*BLOAD(1200)
REAL NEG/4HNEG./,ZERO/4HZERO/
DATA NTP3A/3/, NTP3B/11/
REWIND NTP3A
REWIND NTP3B
NSTIFF = (-1)**ICOUNT
WRITE(6,600) NSTIFF
WRITE(6,660) NR,NC
660 FORMAT(4H NR=,I3,3HNC=,I3)
600 FORMAT(1H ,*ASSEM:IREAD=,I2)
610 FORMAT(6E12.4)
NSET = (NQ - 1)/NR + 1
DO 17 N = 1,NSET
CALL DRZERO(AK,NR,NC)
IF (NI .EQ. 1) GO TO 2
NRBLK = NR
IF (N .EQ. NSET) NRBLK=NQ-(N-1)*NR
IF (NSTIFF .GT. 0) READ(NTP3A) AK
IF (NSTIFF .LT. 0) READ(NTP3B) AK
2 I2 = N*NR
I1 = I2 - NR + 1
IF (I2 .GT. NQ) I2 = NQ
IF (NI .EQ. 3) GO TO 20
C      ASSEMBLE STIFFNESS INTO BLOCKS
C
DO 10 J = 1,NUMEL
IF (IEL(J) .NE. 1) GO TO 10
I12 = DAT(2,J)
DO 5 I = 1,3
IJ2 = KQ(I,J)*6
IJ1 = IJ2 - 5
DO 4 IJ = IJ1,IJ2

```

STRS1380

ASSMC020

ASN40 4

ASSM001

ASSMC070

ASSMC090

ASSMC100

ASSMC140

ASSMC160

ASSMC170

ASSMC180

ASSMC190

ASSM0200

ASSM0210

ASSMC220

ASSM0230

ASSM0240

ASSMC250

ASSMC260

ASSMC270

IF (IJ - I1) 4,6,3	ASSM0280
3 IF(IJ - I2) 6,6,4	ASSMC290
4 CONTINUE	ASSMC300
5 CONTINUE	ASSM0310
GO TO 10	ASSM0320
6 READ (NTP2,I12) ST	
C WRITE(6,203)	
C WRITE(6,201) ((ST(LL,MM),MM=1,18),LL=1,18)	
K = 0	ASSM0340
DO 7 L = 1,3	ASSMC350
ILJ = 6*(KQ(L,J) - 1)	ASSMC360
DO 7 M = 1,6	ASSMC370
K = K + 1	ASSM0380
IL(K) = ILJ + M	ASSMC390
7 CONTINUE	ASSMC400
DO 9 L = 1,18	ASSM0410
IF (IL(L) .LT. I1 .OR. IL(L) .GT. I2) GO TO 9	ASSMC420
IN = IL(L) - I1 + 1	ASSM0430
DO 8 M = 1,18	ASSMC440
IF (ST(L,M) .EQ. 0.) GO TO 8	ASSMC450
JN = IL(M) - IL(L) + 1	ASSMC460
IF (JN .LT. 1) GO TO 8	ASSMC470
AK(IN,JN) = AK(IN,JN) + ST(L,M)	ASSMC480
8 CONTINUE	ASSM0490
9 CONTINUE	ASSMC500
10 CONTINUE	ASSM0510
C WRITE(6,202)	
C WRITE(6,201) ((AK(LL,MM),MM=1,NC),LL=1,NR)	
C	
C MODIFY STIFFNESS FOR BOUNDARY CONDITIONS	ASSMC520
C	
20 CONTINUE	
DO 13 J=1,NUMPTS	ASSMC540
DO 13 I1 = 1,6	ASSM0550
NEQ = (J-1)*6 + I1	ASSMC560
IF (NEQ .LT. I1) GO TO 13	
I = NEQ - I1 + 1	ASSMC570
IF (IVAL(I1,J) .EQ. 0) GO TO 12	ASSM0590
IF (NEQ .GT. I2) GO TO 30	
IF (NI .EQ. 3) GO TO 21	ASSMC600
AK(I,1) = 1.0	ASSM0610
30 CONTINUE	
NRMIN = NEQ - NBAND + 1	
IF (NRMIN .GT. I2) GO TO 13	
DO 40 K = 2,NBAND	ASSMC620
MX=I-K+1	ASSMC630
IF (MX .LT. 0) GO TO 11	
IF (MX .GT. NR) GO TO 11	
AK(MX,K)=0.	ASSMC650
11 IF (NEQ .GT. I2) GO TO 40	
AK(I,K) = 0.0	
40 CONTINUE	
IF (NEQ .GT. I2) GO TO 13	
21 BLOAD(NEQ)=0.0	ASSMC670
12 IF (NEQ .GT. I2) GO TO 13	
13 CONTINUE	ASSM0700
C WRITE(6,200)	
C WRITE(6,201) AK	
200 FORMAT(27H MODIFIED STIFFNESS MATRIX:)	
201 FORMAT(6E15.7)	

```

202 FORMAT(28H ASSEMBLED STIFFNESS MATRIX:)
203 FORMAT(26H STIFFNESS READ FROM NTP2:)
C   CHECK FOR ERRORS
    II = I2 - I1 + 1
    DO 16 I = 1,II
      NEQ = (N - 1)*NR + I
C   WRITE(6,661) I , AK(I,1)
651 FORMAT(3H I=,I3,8HAK(I,1)=,E12.5)
    IF (AK(I,1)) 14,15,16
14 ERROR = NEG
    GO TO 18
15 ERROR = ZERO
    GO TO 18
16 CONTINUE
    IF (NSTIFF .LT. 0) WRITE(NTP3A) AK
    IF (NSTIFF .GT. 0) WRITE(NTP3B) AK
17 CONTINUE
    REWIND NTP3A
    REWIND NTP3B
C   WRITE(6,611)
611 FORMAT(12H LOAD VECTOR)
C   WRITE(6,610) (BLOAD(ILM),ILM=1,NQ)
    REWIND NTP2
    RETURN
18 PRINT 19, ERROR,NEQ
    RETURN 1
19 FORMAT(// 44H      **** ERROR IN THE STIFFNESS MATRIX **** /
*           10H      **** ,A4, 25H DIAGONAL          **** /
*           17H      **** ROW NO. , 15,18H          **** /
*           40H      **** ***** SUBROUTINE ASSEM **** ***** )
    END
C   *****
C   S U B R O U T I N E   F A C T O R
C   THIS SUBROUTINE PERFORMS A CHOLESKI DECOMPOSITION OF
C   THE FORM:
C   F = L * D * (L)T * U
C   *****
C   SUBROUTINE FACTOR(18W,NDFREE,NR,NC,S1,S2)
C   DOUBLE PRECISION S1,S2
C   COMMON /NUNIT/NSTIFF,ICOUNT
C   DIMENSION S1(NR,NC),S2(NR,NC)
C   DATA NTP3A/3/,NTP3B/11/,NBCKSB/4/
    N=0
    DO 1000 NEQ=1,NDFREE
      NBLK=(NEQ-1)/NR+1
      NREAD=NBLK-N
      IF(NREAD .EQ. 0) GO TO 500
      N=NBLK
      IF (N .EQ. 1) GO TO 400
      DO 300 I=1,NR
        DO 300 J=1,NC
          S2(I,J)=S1(I,J)
300 CONTINUE
      IBLK = NBLK - 1
      WRITE(NBCKSB,IBLK) S1
400 CONTINUE
      WRITE(6,66) NSTIFF
85 FORMAT(1H , 'FACTOR:NSTIFF=',I3)
      IF (NSTIFF .GT. 0) READ(NTP3B)S1

```

ASSMC710
ASSMC720
ASSMC730
ASSMC740

ASSMC750
ASSMC760
ASSMC770
ASSMC780
ASSMC790
ASSMC800

ASSMC820

ASSMC840
ASSMC850
ASSMC860
ASSMC870
ASSMC880
ASSMC890
ASSMC900
ASSMC910

```

      IF (NSTIFF .LT. 0) READ(NTP3A)S1
500  IF (NEC .NE. 1) GO TO 800
      TEMP=1.0/S1(1,1)
      DO 600 J=2,IBW
        S1(1,J)=S1(1,J)*TEMP
600  CONTINUE
      GO TO 1000
      IBLK = NBLK
      WRITE(NBCKSB,IBLK) S1
800  CONTINUE
      LL=NEQ-1
      NN=NDFREE-LL
      IF (NN .GT. IBW) NN=IBW
      DO 60 J=1,NN
        L=IBW-J
        SUM=0.0
        IF (L .EQ. 0) GO TO 50
        IF (LL .LT. L) L=LL
        DO 40 K=1,L
          K1=NEQ-((N-1)*NR+K)
          K2=1+K
          K3=J+K
          IF (K1 .GT. 0) GO TO 45
          K1=K1+NR
          SUM=SUM+S2(K1,K2)*S2(K1,K3)*S2(K1,1)
          GO TO 40
45     CONTINUE
          SUM=SUM+S1(K1,K2)*S1(K1,K3)*S1(K1,1)
40     CONTINUE
50     I=NEQ-(N-1)*NR
          S1(I,J)=S1(I,J)-SUM
          IF (J .EQ. 1) GO TO 60
          S1(I,J)=S1(I,J)/S1(I,1)
60     CONTINUE
1000  CONTINUE
      IBLK = NBLK
      WRITE(NBCKSB,IBLK) S1
      ICOUNT=ICOUNT+1
      REWIND NBCKSB
      REWIND NTP3A
      RETURN
      END
C *****
C      S U B R O U T I N E      S O L V E
C      THIS SUBROUTINE PERFORMS THE BACK-SUBSTITUTION TO SOLVE
C      THE SYTEM OF SIMULTANFOUS EQUATIONS
C *****
C      SUBROUTINE SOLVE(IBW,NDFREE,NR,NC,P,S1,S2)
C      DOUBLE PRECISION S1,S2,U,P,DLL
C      COMMON/DISP/ U(1200),DLL(18),UTOT(1200)
C      DIMENSION S1(NR,NC),S2(NR,NC),P(NDFREE)
C      DATA NBCKSB/4/
C      NSET=(NDFREE-1)/NR + 1
C      IBLK = 1
C      READ(NBCKSB,IBLK) S1
C      U(1)=P(1)/S1(1,1)
C      N=1
C      DO 20 I=2,NDFREE

```

```

      NBLK=(I-1)/NR+1
      NREAD=NBLK-N
      IF (NREAD .EQ. 0) GO TO 5
      DO 110 IJ=1,NR
      DO 110 JI=1,NC
      S2(IJ,JI)=S1(IJ,JI)
110  CONTINUE
      IBLK = NBLK
      READ(NBCKSB,IBLK) S1
      N=N+1
      5  II=I+1
      J=II-IBW
      IF (II .LE. IBW) J=1
      IK=I-1
      SUM=0.0
      DO 10 K=J,IK
      KK=II-K
      KR=K-((N-1)*NR)
      IF (KR .GT. 0) GO TO 15
      KR=KR+NR
      SUM=SUM+S2(KR,KK)*S2(KR,1)*U(K)
      GO TO 10
15  CONTINUE
      SUM=SUM+S1(KR,KK)*S1(KR,1)*U(K)
10  CONTINUE
      I2=I-(N-1)*NR
      U(I)=(P(I)-SUM)/S1(I2,1)
20  CONTINUE
      WRITE(6,510)
510  FORMAT(23H END OF FIRST BACKSOLVE)
C      *****
C      SECOND BACK SOLVE
C      *****
      N=NSET
      BACKSPACE NBCKSB
      NL = NCFREE - (NSET-1)*NR
      U(NDFREE) = U(NDFREE)/S1(NL,1)
      IBLK = NSET
      READ(NBCKSB,IBLK) S1
      DO 40 NN=2,NDFREE
      I=NDFREE+1-NN
      NBLK=(I-1)/NR+1
      NREAD=NBLK-N
      IF (NREAD .EQ. 0) GO TO 100
      N = N-1
C      BACKSPACE NBCKSB
      IBLK = NBLK
      READ(NBCKSB,IBLK) S1
C      BACKSPACE NBCKSB
100  CONTINUE
      IR=I-(N-1)*NR
      LL=I-1
      J=LL+IBW
      IF (J .GT. NDFREE) J=NDFREE
      L=I+1
      SUM=0.0
      DO 30 K=L,J
      KK=K-LL
      SUM=SUM+S1(IR,KK)*U(K)
30  CONTINUE

```

```

40 U(I)=U(I)-SUM
RETURN
END
*****
C      S U B R O U T I N E   C O N S T
C      THIS SUBROUTINE SETS THE VALUES IN THE MATERIAL
C      PROPERTIES MATRICES
C
C      *****
C      SUBROUTINE CONST(L,NSLRS,NL,NE)
C      DOUBLE PRECISION AM,STML,TRM,AR,D,E,F
C      DOUBLE PRECISION TS,PS,SR,CELCON
C      COMMON/ELCON/AM(15,15),TRM(3,3),AR(4),TS(3,3,2,3),PS(2,3),SR(6)
C      COMMON/ELVAR/EPTOT(3,10),CC(3,3,10),CS(3,3,6),ST(3,10),SSTR(2,3)
C      COMMON/STRING/EPP(3,10),STP(3,10),EP(3,10)
C      COMMON/CST/TJ(11,300),C(3,3)
C      COMMON/STIF/STML(18,18),D(3,3),E(3,3),F(3,3)
C      COMMON/STRES/B(6,15),STR(5,10),FST(5,10),FS(3,6)
C      COMMON/MAT/FCS(4),ECS(4),FDS(4),EDS(4),FTS(4),VS(4),EUS(4),FY,
C      ES,DENS(4),BETA,NMAT(300)
C      DATA SQ2/1.417424/,PI/3.141593/
C      REAL NU
C      INTEGER SR
C      DIMENSION FF(3,3),DD(3,3),DF(3,3),DFD(3,3),DQ(3,3)
C      DIMENSION CD(3,3,10),STL(3)
C      DIMENSION NL(300),T(3,3),CN(3,3),CT(3,3),IT(10),A(6,6),DP(6)
C      NM=NMAT(L)
C      FC=FCS(NM)
C      EC=ECS(NM)
C      FD=FDS(NM)
C      ED=EDS(NM)
C      FT=FTS(NM)
C      V=VS(NM)
C      EU=EUS(NM)
C      DEN=DENS(NM)
C
C      FORM INITIAL CONSTITUTIVE MATRICES FOR CONCRETE AND REINFORCEMENT
C
C      NLRS = NL(L)
C      CALL RZERO(CC,3,3,NLRS,1)
C      DO 1 I = 1,NLRS
C      CC(1,1,I) = EC/(1.-V**2)
C      CC(1,2,I) = V*CC(1,1,I)
C      CC(2,1,I) = CC(1,2,I)
C      CC(2,2,I) = CC(1,1,I)
C      CC(3,3,I) = CC(1,1,I)*(1.-V)/2.
C      IF (NSLRS.EQ. 0) GO TO 60
C      CALL RZERO(CS,3,3,6,1)
C      DO 5 I = 1,NSLRS
C      NSPL = SR(I)
C      LN = SR(I+3)
C      DO 5 J = 1,NSPL
C      KI = (I-1)*2 + J
C      DO 4 KK = 1,3
C      DO 4 JJ = 1,3
C      CS(KK,JJ,KI) = TS(KK,1,J,I)*TS(JJ, 1,J,I)*ES
C      CONTINUE
C      GO TO 60
C      ENTRY CONST2(L,CN,ITR,NLR,STL)
C      PRINT 500, ITR

```

```

500 FORMAT(5H ITR=,I2)
    IF(ITR.EQ. 3)GO TO 27
    IF(ITR.EQ. 6) GO TO 27
    IF(ITR.EQ. 8) GO TO 20
    THETA = STR(3,NLR) + PI/2.
    PRINT 501, NLR,STR(3,NLR)
501 FORMAT(6H STR(3,I2,2H)=E10.3)
    PRINT 502, THETA
502 FFORMAT(7H THETA=,E10.3)
    T(1,1) = COS(THETA)**2
    T(1,2) = SIN(THETA)**2
    T(1,3) = -2.*COS(THETA)*SIN(THETA)
    T(2,1) = T(1,2)
    T(2,2) = T(1,1)
    T(2,3) = -T(1,3)
    T(3,1) = T(2,3)/2.
    T(3,2) = T(1,3)/2.
    T(3,3) = T(1,1) - T(1,2)
    CALL RZERO(CN,3,3,1,1)
    CED=EC
    IF(ITR.EQ. 11)CED=ED
    CN(1,1) =CED
    CN(3,3)=0.4*CFD/(2.*(1.+V))
C   CN(3,3) HAS A SHEAR RETENTION FACTOR EQUAL TO .4
    CALL RZERO(CT,3,3,1,1)
    DO 10 I = 1,3
    DO 10 J = 1,3
    DO 10 K = 1,3
10  CT(I,J) = CT(I,J) + T(I,K)*CN(K,J)
    CALL RZERO(CN,3,3,1,1)
    DO 15 I = 1,3
    DO 15 J = 1,3
    DO 15 K = 1,3
15  CN(I,J) = CN(I,J) + CT(I,K)*T(J,K)
    RETURN
20  CONTINUE
    CALL RZERO(CN,3,3,1,1)
    CN(1,1) = ED/(1.-V**2)
    CN(1,2) = V*CN(1,1)
    CN(2,1) = CN(1,2)
    CN(2,2) = CN(1,1)
    CN(3,3) = CN(1,1)*(1.-V)/2.
    RETURN
27  CALL RZERO(CN,3,3,1,1)
    RETURN
    ENTRY CONST3(CN,J,STL)
26  CONTINUE
    ENTRY CONST4(CN,J,STL)
28  FX = (-STL(2)+2.*STL(1))/(3.*SQRT(2.*STL(1)**2-2.*STL(1)*STL(2)
    *+2.*STL(2)**2+6.*STL(3)))+ SQ2*(BETA-1.)/(3.*(2.*BETA-1.))
    FY = (-STL(1)+2.*STL(2))/(3.*SQRT(2.*STL(1)**2-2.*STL(1)*STL(2)
    *+2.*STL(2)**2+6.*STL(3)))+ SQ2*(BETA-1.)/(3.*(2.*BETA-1.))
    FXY = 1./SQRT(2.*STL(1)**2-2.*STL(1)*STL(2)+2.*STL(2)**2+6.*STL(
    *3))
    PRINT 700, FX,FY,FXY
700 FORMAT(4H FX=,E10.3,3HFY=,E10.3,4HFXFY=,E10.3)
    FF(1,1)=FX*FX
    FF(1,2)=FX*FY
    FF(1,3)=FX*FXY
    FF(2,2)=FY*FY

```

CONSC410

CONSC420

CONSC430

CONSC440

CONSC450

CONSC460

CONSC470

CONSC480

CONSC490

CONSC500

CONSC560

CONSC570

CONSC580

CONSC600

CONSC610

CONSC620

CONSC630

CONSC650

CONSC660

CONSC670

CONSC720

CONSC730

CONSC740

CONSC750

CONSC760

CONSC770

CONSC950

CONSC960

CONSC970

CONSC980

CONSC101-

```

FF(2,3)=FY*FX
FF(3,3)=FXY*FXY
FF(2,1)=FF(1,2)
FF(3,2)=FF(2,3)
FF(3,1)=FF(1,3)
DD(1,1)=ED/(1.-V*V)
DD(1,2)=ED*V/(1.-V*V)
DD(2,1)=DD(1,2)
DD(2,2)=DD(1,1)
DD(3,3)=ED*(1.-V)/(2.*(1.-V*V))
DD(1,3)=0.0
DD(2,3)=0.0
DD(3,1)=0.0
DD(3,2)=0.0
DO 35 II=1,3
DO 35 JJ=1,3
DF(II,JJ)=0.0
DO 35 KK=1,3
35 DF(II,JJ)=DF(II,JJ)+DD(II,KK)*FF(KK,JJ)
DO 25 II=1,3
DO 25 JJ=1,3
DFD(II,JJ)=0.0
DO 25 KK=1,3
25 DFD(II,JJ)=DFD(II,JJ)+DF(II,KK)*DD(KK,JJ)
Q=FX*(DD(1,1)*FX+DD(1,2)*FY)+FY*(DD(1,2)*FX+DD(2,2)*FY)
*+FXY*FXY*DD(3,3)
DO 23 II=1,3
DO 23 JJ=1,3
23 DQ(II,JJ)=DFD(II,JJ)/Q
DO 21 II=1,3
DO 21 JJ=1,3
21 CN(II,JJ)=DD(II,JJ)-DQ(II,JJ)
RETURN
60 DO 805 IX=1,3
DO 805 JX=1,3
E(IX,JX)=0.
F(IX,JX)=0.
805 D(IX,JX)=0.
DO 82 II=1,NLRS

```

```

CONS1240
CONS1250
CONS1260
CONS1270
CONS1280
CONS1290
CONS1300

```



```

      ZTJ1 = TJ(II,L)
      ZTJ = TJ(II+1,L)
      IF (NSLRS .EQ. 0) GO TO 73
      DO 65 I = 1,3
      IF (SR(I+3) .EQ. II) GO TO 68
65  CONTINUE
      GO TO 73
68  NSPL = SR(II)
      TPS=0.0
      DO 70 K = 1,NSPL
      KI = (I-1)*2 + K
      TPS=TPS+PS(K,I)
      DO 70 KK = 1,3
      DO 70 JJ = 1,3
      D(KK,JJ) = D(KK,JJ) + (ZTJ1-ZTJ)*CS(KK,JJ,KI)*PS(K,I)
      E(KK,JJ) = E(KK,JJ) - (ZTJ1**2-ZTJ**2)*CS(KK,JJ,KI)*.5*PS(K,I)
70  F(KK,JJ) = F(KK,JJ) + (ZTJ1**3-ZTJ**3)*CS(KK,JJ,KI)*PS(K,I)/3.
      DO 81 I=1,3
      DO 81 J=1,3
      D(I,J) = D(I,J) + (ZTJ1-ZTJ)*CC(I,J,II)*(1.0-TPS)
      E(I,J) = E(I,J) - (ZTJ1**2-ZTJ**2)*CC(I,J,II)*(1.0-TPS)/2.
81  F(I,J) = F(I,J) + (ZTJ1**3-ZTJ**3)*CC(I,J,II)*(1.0-TPS)/3.
      GO TO 82
73  DO 80 I = 1,3
      DO 80 J = 1,3
      D(I,J) = D(I,J) + (ZTJ1-ZTJ)*CC(I,J,II)
      E(I,J) = E(I,J) - (ZTJ1**2-ZTJ**2)*CC(I,J,II)/2.
      F(I,J) = F(I,J) + (ZTJ1**3-ZTJ**3)*CC(I,J,II)/3.
80  CONTINUE
82  CONTINUE
      GO TO 90
      ENTRY CONST5(L,NL,CD)
      DO 86 I=1,3
      DO 86 J=1,3
      D(I,J)=0.
      E(I,J)=0.
86  F(I,J)=0.
      NLRS = NL(L)
      PRINT 600, NLRS
600 FORMAT(6H NLRS=,I2)

      DO 85 II=1,NLRS
      ZTJ1 = TJ(II,L)
      ZTJ = TJ(II+1,L)
      IF (NSLRS .EQ. 0) GO TO 730
      DO 650 I=1,3
      IF (SR(I+3) .EQ. II) GO TO 680
650 CONTINUE
      GO TO 730
680 NSPL=SR(II)
      TPS=0.0
      DO 710 K=1,NSPL
      KI=(I-1)*2+K
      TPS=TPS+PS(K,I)
710 CONTINUE
730 CONTINUE
C  PRINT 601, ZTJ1,ZTJ
601 FORMAT(6H ZTJ1=,E10.3,5X,4HZTJ=,E10.3)
      DO 85 I = 1,3
      DO 85 J = 1,3

```

CONS1310
 CONS1320
 CONS1330
 CONS1340
 CONS1350
 CONS1360
 CONS1370
 CONS1380
 CONS1390
 CONS1400
 CONS1410
 CONS1420
 CONS1430
 CONS1440
 CONS1450
 CONS1460
 CONS1470
 CONS1480
 CONS1490
 CONS1500
 CONS1510
 CONS1520
 CONS1530
 CONS1540
 CONS1550
 CONS1560
 CONS1570
 CONS1580
 CONS1590
 CONS1600
 CONS1610

CONS1630

CONS1650
 CONS1660

CONS1670
 CONS1680

```

652 FORMAT(4H CD(I2,I2,I2,2H)=,E10.3)
C PRINT 602, I,J,II,CD(I,J,II)
D(I,J)=D(I,J)+(ZTJ1-ZTJ)*CD(I,J,II)*(1.0-TPS)
E(I,J)=E(I,J)+(-(ZTJ1**2-ZTJ**2)*CD(I,J,II)*(1.-TPS)/2.)
85 F(I,J)=F(I,J)+(ZTJ1**3-ZTJ**3)*CD(I,J,II)*(1.-TPS)/3.
90 RETURN
END
C *****
C SUBROUTINE STIFF
C THIS SUBROUTINE FORMS ELEMENTAL STIFFNESS MATRIX FOR EACH
C ELEMENT
C *****
C SUBROUTINE STIFF
C DOUBLE PRECISION AA,ST,BA,BB,D,E,F,TR,AR,X,Y,S1,S2
C DOUBLE PRECISION A,AX,AY,AXY,AXX,AYY
C DOUBLE PRECISION TS,PS,SR,CELCON
COMMON/ELCON/AA(15,15),TR(3,3),AR(4),TS(3,3,2,3),PS(2,3),SR(6)
COMMON/STIF/ST(18,18),D(3,3),E(3,3),F(2,3)
DIMENSION BA(15,15),BB(15,15),NST(15),S1(3,3),S2(3,3)
DIMENSION X(3),Y(3)
DATA NST/1,2,3,4,5,7,8,9,10,11,13,14,15,16,17/
CALL DRZERO(AA,15,15)
X(1)=0.0
Y(1)=0.0
X(2)=0.0
Y(2)=AR(1)
X(3)=AR(2)
Y(3)=AR(3)
DO 100 I=1,3
J=(I-1)*5+1
AA(J,1)=1.0
AA(J,2)=X(I)
AA(J,3)=Y(I)
AA(J+1,4)=1.0
AA(J+1,5)=X(I)
AA(J+1,6)=Y(I)
AA(J+2,7)=1.0
AA(J+2,8)=X(I)
AA(J+2,9)=Y(I)
AA(J+2,10)=X(I)*X(I)
AA(J+2,11)=X(I)*Y(I)
AA(J+2,12)=Y(I)*Y(I)
AA(J+2,13)=X(I)*X(I)*X(I)
AA(J+2,14)=X(I)*X(I)*Y(I)+X(I)*Y(I)*Y(I)
AA(J+2,15)=Y(I)*Y(I)*Y(I)
AA(J+3,9)=+1.0
AA(J+3,11)=+X(I)
AA(J+3,12)=+2*Y(I)
AA(J+3,14)=+(X(I)*X(I)+2.*X(I)*Y(I))
AA(J+3,15)=+3.*Y(I)*Y(I)
AA(J+4,8)=-1.0
AA(J+4,10)=-2.*X(I)
AA(J+4,11)=-Y(I)
AA(J+4,13)=-3.*X(I)*X(I)
AA(J+4,14)=-2.*X(I)*Y(I)-Y(I)*Y(I)
100 CONTINUE
CALL INVERT (AA,15)
ENTRY STIFF2
A=AR(2)*AR(1)*.5

```

CONS1720
CONS1730

STIF0010

STIFC040

STIFC050

STIF0760

```

AX=AR(1)*AR(2)*AR(2)/6.
AY=AR(2)*AR(1)*{AR(1)+AR(3)}/6.
AXY=AR(2)*AR(2)*AR(1)*{AR(1)+2.*AR(3)}/24.
AXX=AR(2)*AR(2)*AR(2)*AR(1)/12.
Ayy=AR(2)*AR(1)*{AR(1)*AR(1)+AR(1)*AR(3)+AR(3)*AR(3)}/12.
C  WRITE(6,352)A,AX,AY,AXY,AXX,Ayy
352 FORMAT(6E12.3)
CALL DRZERO(BA,15,15)
BA(2,2) = A*D(1,1)
BA(2,3) = A*D(1,3)
BA(2,5) = BA(2,3)
BA(2,6) = A*D(1,2)
BA(3,3) = A*D(3,3)
BA(3,5) = BA(3,3)
BA(3,6) = A*D(2,3)
BA(5,5) = BA(3,3)
BA(5,6) = BA(3,6)
BA(6,6) = A*D(2,2)
BA(2,10) =+2.*A*E(1,1)
BA(2,11) =+2.*A*E(1,3)
BA(2,12) =+2.*A*E(1,2)
BA(2,13) =+6.*AX*E(1,1)
BA(2,14) =+2.*{AY*E(1,1) + AX*E(1,2) + 2.*E(1,3)*{AX+AY}}
BA(2,15) =+6.*AY*E(1,2)
BA(3,10) =+ BA(2,11)
BA(3,11) =+2.*A*E(3,3)
BA(3,12) =+2.*A*E(2,3)
BA(3,13) =+6.*AX*E(1,3)
BA(3,14) =+2.*{AY*E(1,3) + AX*E(2,3) + 2.*E(3,3)*{AX+AY}}
BA(3,15) =+6.*AY*E(2,3)
BA(5,10) = BA(3,10)
BA(5,11) = BA(3,11)
BA(5,12) = BA(3,12)
BA(5,13) = BA(3,13)
BA(5,14) = BA(3,14)
BA(5,15) = BA(3,15)
BA(6,10) =+BA(2,12)
BA(6,11) = BA(3,12)
BA(6,12) =+2.*A*E(2,2)
BA(6,13) =+6.*AX*E(1,2)
BA(6,14) =+2.*{AY*E(1,2) + AX*E(2,2) + 2.*E(2,3)*{AX+AY}}
BA(6,15) =+6.*AY*E(2,2)
BA(10,10) = 4.*A*F(1,1)
BA(10,11) =+4.*A*F(1,3)
BA(10,12) =+4.*A*F(1,2)
BA(10,13) = 12.*AX*F(1,1)
BA(10,14) = 4.*{AY*F(1,1) + AX*F(1,2) + 2.*F(1,3)*{AX+AY}}
BA(10,15) =+12.*AY*F(1,2)
BA(11,11) = 4.*A*F(3,3)
BA(11,12) = 4.*A*F(2,3)
BA(11,13) =+12.*AX*F(1,3)
BA(11,14) =+4.*{AY*F(1,3) + AX*F(2,3) + 2.*F(3,3)*{AX+AY}}
BA(11,15) = 12.*AY*F(2,3)
BA(12,12) = 4.*A*F(2,2)
BA(12,13) =+12.*AX*F(1,2)
BA(12,14) =+4.*{AY*F(1,2) + AX*F(2,2) + 2.*F(2,3)*{AX+AY}}
BA(12,15) = 12.*AY*F(2,2)
BA(13,13) = 36.*AXX*F(1,1)
BA(13,14) = 12.*{AXY*F(1,1) + AXX*F(1,2) + 2.*F(1,3)*{AXX+AXY}}
BA(13,15) =+36.*AXY*F(1,2)
STIF0860
STIFC870
STIFC880
STIFC890
STIFC900
STIFC910
STIFC920
STIFC930
STIFC940
STIF0950
STIFC960
STIFC970
STIFC980
STIFC990
STIF1000
STIF1010
STIF1020
STIF1030
STIF1040
STIF1050
STIF1060
STIF1070
STIF1080
STIF1090
STIF1100
STIF1110
STIF1120
STIF1130
STIF1140
STIF1150
STIF1160
STIF1170
STIF1180
STIF1190
STIF1200
STIF1210
STIF1220
STIF1230
STIF1240
STIF1250
STIF1260
STIF1270
STIF1280
STIF1290
STIF1300
STIF1310
STIF1320
STIF1330
STIF1340
STIF1350
STIF1360
STIF1370

```

```

      BA(14,14) =+F(1,1)*(4.*Ayy)+F(2,2)*(4.*Axx)+F(2,3)*(16.*Axx+32.*
*   AXY+16.*Ayy)+F(1,2)*(8.*AXY)+F(1,3)*(16.*Ayy+16.*AXY)+F(2,3)*
*   (16.*Axx+16.*AXY)
      BA(14,15) =+12.*(Ayy*F(1,2) + AXY*F(2,2) + 2.*F(2,3)*(AXY+Ayy))
      BA(15,15) = 36.*Ayy*F(2,2)
      DO 6 I = 1,15
      II = I + 1
      DO 6 J = II,15
6     BA(J,I) = BA(I,J)
      CALL DRZERO(ST,18,18)
      CALL DRZERO(BB,15,15)
      DO 9 K = 1,15
      DO 9 I = 1,15
      DO 9 J = 1,15
9     BB(K,I) = BB(K,I) + AA(J,K)*BA(J,I)
      DO 10 K = 1,15
      DO 10 I = 1,15
      BA(K,I) = 0.0
      DO 10 J = 1,15
10    BA(K,I) = BA(K,I) + BB(K,J)*AA(J,I)
      DO 11 I = 1,15
      J = NST(I)
      DO 11 K = 1,15
      LL = NST(K)
11    ST(J,LL) = BA(I,K)
C     TRANSFORM STIFFNESS MATRIX
C     WRITE(6,204)
C     WRITE(6,202) ((TR(IR,JC),JC=1,3),IR=1,3)
202  FORMAT(3E12.3)
204  FORMAT(22H TRANSFORMATION MATRIX)
      DO 18 MM = 1,6
      LL = (MM-1)*3
      DO 18 NN = 1,6
      KK = (NN-1)*3
      DO 12 I = 1,3
      KKI = KK + I
      DO 12 J = 1,3
      LLJ = LL + J
12    S1(I,J) = ST(KKI,LLJ)
      DO 14 I = 1,3
      DO 14 J = 1,3
      S2(I,J) = 0.0
      DO 14 JJ = 1,3
14    S2(I,J) = S2(I,J) + TR(JJ,I)*S1(JJ,J)
      DO 16 I = 1,3
      DO 16 J = 1,3
      S1(I,J) = 0.0
      DO 16 JJ = 1,3
16    S1(I,J) = S1(I,J) + S2(I,JJ)*TR(JJ,J)
      DO 17 I = 1,3
      KKK = KK + I
      DO 17 J = 1,3
      LLL = LL + J
17    ST(KKK,LLL) = S1(I,J)
18    CONTINUE
      RETURN
      END
      SUBROUTINE IMPROV(IBW,NDFREE,NR,NC,BLCAD,AK1,AK2)
      COMMON/DISP/ U(1200),DLL(18),UTOT(1200)
      COMMON/IMPRUV/RESID(200),UPRIME(200)

```

STIF1380
 STIF1390
 STIF1400
 STIF1410
 STIF1420
 STIF1430
 STIF1440
 STIF1450
 STIF1460

 STIF1500
 STIF1510
 STIF1530
 STIF1540
 STIF1550
 STIF1560
 STIF1570
 STIF1580
 STIF1590
 STIF1600
 STIF1610
 STIF1620
 STIF1630
 STIF1640
 STIF1650

 STIF1660
 STIF1670
 STIF1680
 STIF1690
 STIF1700
 STIF1710
 STIF1720
 STIF1730
 STIF1740
 STIF1750
 STIF1760
 STIF1770
 STIF1780
 STIF1790
 STIF1800
 STIF1810
 STIF1820
 STIF1830
 STIF1840
 STIF1850
 STIF1860
 STIF1870
 STIF1880
 STIF1890
 STIF1900
 STIF1910
 STIF1920

```

C      DOUBLE PRECISION RESID,SUM,BLOAD,AK1,AK2,U,DLL
      DIMENSION AK1(NR,NC),AK2(NR,NC)
      DIMENSION BLOAD(NDFREE)
      NITER = 0
      NTP3B=11
      NTP3A=3
      IREAD=-1
      MAXIT=3
      SUMSQ=0.0
      DO 1 IJ=1,NDFREE
        UPRIME(IJ)=U(IJ)
        SUMSQ=SUMSQ + U(IJ)*U(IJ)
1      CONTINUE
        DNORM=SQRT(SUMSQ)
        CHECK=0.01*DNORM
5      N=0
        NITER=NITER+1
        WRITE(6,2000) NITER
2000  FORMAT(15H ITERATION NO.: ,I2)
        DO 100 IR=1,NDFREE
          NBLK=(IR-1)/NR + 1
          NREAD=NBLK-N
          IF(NREAD.EQ. 0) GO TO 10
          IF (NBLK.EQ. 1) GO TO 3
          DO 2 IJ=1,NR
            DO 2 JI=1,NC
              AK2(IJ,JI)=AK1(IJ,JI)
2          CONTINUE
3          N=N+1
          IF(IREAD.GT. 0) READ(NTP3B) AK1
33         FORMAT(E12.5,4E13.5)
          IF(IREAD.LT. 0) READ(NTP3A) AK1
10         CONTINUE
          IR2=(IR+1)-IBW
          IF(IR2.LT. 1) IR2=1
          IC=IR
          IF(IR2.GT. 1) IC=IBW
          NUMCOL=IC-1
          IC=IC+1
          IR2=IR2-1
          SUM=0.0
          IRB=IR2-(NBLK-1)*NR
          IF(IR.EQ. 1) GO TO 21
          DO 20 KC=1,NUMCOL
            IC=IC-1
            IR2=IR2+1
            IRB=IRB+1
            IF(IRB.LE. 0) GO TO 15
            SUM=SUM + AK1(IRB,IC)*UPRIME(IR2)
            GO TO 20
15         IRB2=IRB+NR
            SUM=SUM + AK2(IRB2,IC)*UPRIME(IR2)
20         CONTINUE
21         IRB=IRB+1
            IC=(NDFREE+1)-IR
            IF (IC.GT. IBW) IC=IBW
            IF (IRB.GT. NR) IRB=1
            DO 30 KC=1,IC
              IR2=IR2+1
              SUM=SUM + AK1(IRB,KC)*UPRIME(IR2)

```

```

30 CONTINUE
   RESID(IR)=BLOAD(IR)-SUM
100 CONTINUE
   WRITE(6,1012)
   WRITE(6,1011) (RESID(KL),KL=1,108)
1012 FORMAT(10H RESIDUAL:)
   REWIND NTP3A
   REWIND NTP3B
   CALL SOLVE(18W,NDFREE,NR,NC,RESID,AK1,AK2)
   SUMSQ=0.0
   DO 200 I=1,NDFREE
     UPRIME(I)=UPRIME(I)+U(I)
     SUMSQ=SUMSQ+U(I)*U(I)
200 CONTINUE
   ERNORM=SQRT(SUMSQ)
   WRITE(6,1000) DNORM,ERNORM
1000 FORMAT(7H DNORM=,E13.6,10H ERNORM=,E13.6)
   WRITE(6,1010)
   WRITE(6,1011) (U(K),K=1,108)
1010 FORMAT(26H DISPLACEMENT CORRECTIONS:)
1011 FORMAT(6E13.5)
C   IF (ERNORM .LT. CHECK) GO TO 300
   IF (NITER .LT. MAXIT) GO TO 5
   GO TO 300
   WRITE(6,600)
600 FORMAT(49H SOLUTION DID NOT CONVERGE WITHIN SPECIFIED ERROR)
   RETURN
300 DO 400 IJ=1,NDFREE
     U(IJ)=UPRIME(IJ)
400 CONTINUE
   RETURN
END
SUBROUTINE INVERT (A,NUM)
C   IMPLICIT REAL*8 (A-H,O-Z)
   DIMENSION A(15,15),IP(15),IN(15,2),PI(15)
101  FORMAT(20H MATRIX IS SINGULAR )
   DO 1 J=1,NUM
1    IP(J)=0
     DO 14 I=1,NUM
       AM=0.
       DO 6 J=1,NUM
         IF (IP(J)-1)2,6,2
2        DO 5 K=1,NUM
          IF (IP(K)-1)3,5,19
3          IF (ABS(AM)-ABS(A(J,K)))4,4,5
4          IR=J
           IC=K
           AM=A(J,K)
5          CONTINUE
6          CONTINUE
           IP(IC)=IP(IC)+1
           IF (IR-IC)7,9,7
7          DO 8 L=1,NUM
            SW=A(IR,L)
            A(IR,L)=A(IC,L)
8            A(IC,L)=SW
9            IN(I,1)=IR
             IN(I,2)=IC
             PI(I)=A(IC,IC)
             IF (PI(I))10,18,10

```

```

10  A(IC,IC)=1.
    DO 11 L=1,NUM
11  A(IC,L)=A(IC,L)/PI(I)
    DO 14 L1=1,NUM
    IF (L1-IC) 12,14,12
12  T=A(L1,IC)
    A(L1,IC)=0.
    DO 13 L=1,NUM
13  A(L1,L)=A(L1,L)-A(IC,L)*T
14  CONTINUE
    DO 17 I=1,NUM
    L=NUM+1-I
    IF (IN(L,1)-IN(L,2)) 15,17,15
15  JR=IN(L,1)
    JC=IN(L,2)
    DO 16 K=1,NUM
    SW=A(K,JR)
    A(K,JR)=A(K,JC)
    A(K,JC)=SW
16  CONTINUE
17  CONTINUE
    RETURN
18  WRITE(3,101)
19  RETURN
    END
    SUBROUTINE INTVAL(IARRAY,NROW,NCOL,INTEG)
    DIMENSION IARRAY(NROW,NCOL)
    DO 120 J=1,NCOL
    DO 110 I=1,NROW
    IARRAY(I,J)=INTEG
110 CONTINUE
120 CONTINUE
    RETURN
    END
    SUBROUTINE RZERO(ARRAY,NDIM1,NDIM2,NDIM3,NDIM4)
    DIMENSION ARRAY(NDIM1,NDIM2,NDIM3,NDIM4)
    DO 140 L=1,NDIM4
    DO 130 K=1,NDIM3
    DO 120 J=1,NDIM2
    DO 110 I=1,NDIM1
    ARRAY(I,J,K,L)=0.0
110 CONTINUE
120 CONTINUE
130 CONTINUE
140 CONTINUE
    RETURN
    END
    SUBROUTINE DRZERO(DARRAY,NDIM1,NDIM2)
C  DOUBLE PRECISION DARRAY
    DIMENSION DARRAY(NDIM1,NDIM2)
    DO 120 J=1,NDIM2
    DO 110 I=1,NDIM1
    DARRAY(I,J)=0.0
110 CONTINUE
120 CONTINUE
    RETURN
    END
//GD.FT01F001 DD DSN=&ENG1,UNIT=SYSDA,DISP=(NEW,PASS,DELETE),
// SPACE=(TRK,(20,10),RLSE),DCB=(RECFM=VRS,BLKSIZE=5000)
//GD.FT02F001 DD DSN=&ENG2,UNIT=SYSDA,DISP=(NEW,PASS,DELETE),

```

```

// SPACE=(TRK,(10,5),RLSE),DCB=(RECFM=VBS,BLKSIZE=4000)
//GJ.FT03F001 DD DSN=&ENG3,UNIT=SYSDA,DISP=(NEW,PASS,DELETE),
// SPACE=(TRK,(20,10),RLSE),DCB=(RECFM=VBS,BLKSIZE=4000)
//GJ.FT04F001 DD DSN=&ENG4,UNIT=SYSDA,DISP=(NEW,PASS,DELETE),
// SPACE=(TRK,(20,10),RLSE),DCB=BLKSIZE=3600
//GJ.FT08F001 DD DSN=&ENG8,UNIT=SYSDA,DISP=(NEW,PASS,DELETE),
// SPACE=(TRK,(20,10),RLSE),DCB=BLKSIZE=3600
//GJ.FT09F001 DD DSN=&ENG9,UNIT=SYSDA,DISP=(NEW,PASS,DELETE),
// SPACE=(TRK,(20,10),RLSE),DCB=(RECFM=VBS,BLKSIZE=2012)
//GJ.FT10F001 DD DSN=&ENG10,UNIT=SYSDA,DISP=(NEW,PASS,DELETE),
// SPACE=(TRK,(20,10),RLSE),DCB=(RECFM=VBS,BLKSIZE=2012)
//GJ.FT11F001 DD DSN=&ENG11,UNIT=SYSDA,DISP=(NEW,PASS,DELETE),
// SPACE=(TRK,(20,10),RLSE),DCB=(RECFM=VBS,BLKSIZE=3600)
//GJ.SYSIN DD *
2 4 1
3.0E03 3.5E06 3.80E03 .88E06 5.46E02
.0DE00 3.E-03 0.E00 6.50E04 3.0E07
4
1 1 0 0.0 0.0 0.0
2 2 0 1.0E01 0.0 0.0
3 3 0 0.0 5.0E00 0.0
4 4 0 1.0E01 5.0 0.0
0
2
1 0 3.0E00 1 2 3 1
2 0 3.0E00 4 3 2 1
1
1 2 1 3 1.0E00 1.0E00 1.0E00
1
1 1
2 3.000E01 9.000E01
1
1 2 1 1
5
1 1 1 111
2 2 1 001
1 4 1 222
3 3 1 101
4 4 1 001
1
2 4 2 -0.8E05
9 3
1 2 1 2.50E01
3 5 1 1.25E01
5 9 1 3.125E00

```


VITA

William J. Dorris was born in Chattanooga, Tennessee on May 12, 1954. He graduated from Notre Dame High School in 1972 and received a Bachelor of Science degree from The University of Tennessee, Knoxville, in June of 1977. After completing the requirements for a Master of Science degree, Mr. Dorris plans to work as a staff member at the Los Alamos Scientific Laboratory in Los Alamos, New Mexico.