STUDY OF PLANE ELASTO-PLASTIC PROBLEMS UTILIZING NONLINEAR FINITE ELEMENT ANALYSIS

by

Kanapathy Ramalingam

Submitted in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in the

Mechanical Engineering

Program

Javed Alam — Nov 21st, 85

Adviser — Date

Sally M. Hotchkiss — November 26, 1985

Dean of the Graduate School — Date

YOUNGSTOWN STATE UNIVERSITY

December, 1985

ABSTRACT

STUDY OF PLANE ELASTO-PLASTIC PROBLEMS UTILIZING NONLINEAR FINITE ELEMENT ANALYSIS

Kanapathy Ramalingam
Master of Science
Youngstown State University, 1985

The objective in this thesis is to present and demonstrate the use of Finite Element based methods for the solution of problems involving plasticity. Incremental theory of plasticity is used and only the elasto-plastic problems of the type having elastic and plastic strains in the same order of magnitude are considered.

Study of the method herein includes the preparation of two computer programs: one for linear strain hardening material, and the other for nonlinear strain hardening material to solve plane stress or plane strain or axisymmetric problems. The programs are tested by comparing the results obtained for two elasto-plastic problems with the corresponding solution / experimental data found in the literature. The problems studied are: (1) a thick walled cylindrical pressure vessel subjected to internal pressure, and (2) a circular hole in a uniformly stressed infinite flat plate.

Also included are the results for a compact tensile test fracture specimen, fully plastic cylinder with linear

strain hardening material, and a tension member with circular holes.

In general very good results are obtained for the two elasto-plastic problems when compared to established solutions. From this investigation there is evidence that the programs can be used for successful analysis of other plane elasto-plastic problems, with nonlinear strain hardening relationship, for which close formed solutions may not exist.

ACKNOWLEDGEMENTS

I would like to dedicate this thesis to my wife Sundari, and our sons Ashok and Vijay, for their patience, understanding and encouragement.

I wish to acknowledge my debt of gratitude to professors Dr.J.Alam and Dr.Frank A.D'Isa for their time and effort in the development and review of my thesis. I would like to acknowledge my appreciation to Mr.Carl H. Minton of ADS Machinery Corp. for his continuous encouragement and interest.

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF SYMBOLS

SYMBOL	DEFINITION
a	Inner radius
$\{a\}$	Derivative of loading function with respect to stresses
$\{a_1\}$	Derivative of σ_m with respect to stresses
$\{a_2\}$	Derivative of σ_e with respect to stresses
$\{a_3\}$	Derivative of J_3 with respect to stresses
A	Constant (Chapter III)
b	Outer radius
B	Strain displacement matrix
B^e	Strain displacement matrix of an element
B_i^e	Strain displacement matrix of an element at node i
c	Yield radius
c_1, c_2, c_3	Constants relating $\{a\}$ to $\{a_1\}$, $\{a_2\}$, and $\{a_3\}$ defined in Chapter IV
det	Determinant
$\{d_D\}$	Matrix product of $\{D\}$ and $\{a]$
D	Elasticity matrix
D^e	Elasticity matrix of an element
$\{D_{ep}\}$	Elasto-plastic matrix
E	Modulus of elasticity
f^b	Vector of body forces
$f^{b(e)}$	Vector of element body forces
f^s	Vector of surface tractions
$f^{s(e)}$	Vector of element surface forces
f^I, F	Vector of externally applied concentrated forces

SYMBOL	DEFINITION
f_x^b, f_y^b, f_z^b	Body force components
f_x^s, f_y^s, f_z^s	Surface force components
F_x^I, F_y^I, F_z^I	Externally applied concentrated force components
g	Acceleration due to gravity
G	Shear modulus
H'	Hardening parameter
$\{I\}$	Identity matrix
$\{J\}$	Jacobian matrix of transformation
$\{J^e\}$	Jacobian matrix of transformation of an element
J_1, J_2, J_3	Invariants of stress tensor
J_1', J_2', J_3'	Invariants of deviatoric stress tensor
k	Strain hardening or yield function
$k_{\sigma_\theta}, k_{\sigma_e}, k_{\epsilon_e}$	Stress / strain concentration factors defined in Chapter VI
$\{K\}$	Structure stiffness matrix
$\{K(U)\}$	Structure stiffness matrix corresponding to displacement vector U
$\{K_T\}$	Tangent stiffness matrix
n	Number of nodes of an element
M	Total number of degrees of freedom in the system
NGAUS	Number of Gauss points
$\{N\}$	Matrix of shape functions

SYMBOL	DEFINITION
N^e	Matrix of shape functions of an element
N_i^e	Shape functions for local node i of an element
$\{N^{(s)e}\}$	Matrix of surface shape functions of an element
r, θ, z	Global cylindrical polar coordinates
P	Pressure
P_c	Critical pressure
P_n	Normal pressure on an element
P_t	Tangential pressure on an element
P_{xi}	Equivalent nodal force along x axis for node i
P_{yi}	Equivalent nodal force along y axis for node i
r	Radius at a point
r_p	Radial distance of Gauss point
$\{R\}$	Equivalent nodal forces of the structure
R_B	Body forces
R_C	Concentrated forces
R_S	Surface forces
S	Surface area
S^e	Surface area of an element
S_{ij}	Deviator stress tensor
S_1, S_2, S_3	Principal deviatoric stresses
S_x, S_y, S_z	Deviatoric stress components
	Thickness

SYMBOL	DEFINITION
t^e	Element thickness
T	Tolerance
u_d	Radial displacement
U	Displacement vector
U^e	Displacement vector of an element
$\hat{U}$	Vector of global displacement components at nodal points
$\hat{U}_i$	Vector of global displacement components for node i
$\bar{U}$	Virtual displacement vector
$\hat{U}_i^e$	Vector of global displacement components for node i for an element
U^s	Surface displacement
U^I	Displacement at the application of concentrated forces
$\bar{\hat{U}}$	Global virtual displacement components at element nodes
U^0	Vector of initial displacement
U^r	Displacement vector at r^{th} iteration
u,v,w	Displacement components
u_i^e, v_i^e, w_i^e	Displacement components of an element at node i
V	Volume
V^e	Element volume
w_i, w_j, w_k, w_p, w_q	Weight functions at sampling points i,j,k,p and q
w^p	Plastic work

SYMBOL	DEFINITION
x, y, z	Global cartesian coordinates of the system
x^e, y^e	Global cartesian coordinates of an element
x_i^e, y_i^e	Global cartesian coordinates of an element at node i
α	Ratio of inner to outer radius
β	Ratio of r to outer radius
δ	Kronecker delta
μ	Ratio of yield to outer radius
ϵ	Strain vector
ϵ	Virtual strain vector
ϵ^e	Element strain vector
ϵ_{ij}	Strain tensor
ϵ'_{ij}	Deviatoric strain tensor
$\epsilon'_{ij(p)}$	Deviatoric plastic strain tensor
$\epsilon_{xx}, \epsilon_{yy}, \epsilon_{zz}$	Normal strains along x,y and z axes
$\epsilon_{rr}, \epsilon_{\theta\theta}$	Normal strains along r and6 directions
ϵ_p	Vector of total plastic strain
$\{d\epsilon_{ij}\}_T$	Incremental total strain tensor
$\{d\epsilon_{ij}\}_e$	Incremental elastic strain tensor
$\{d\epsilon_{ij}\}_p$	Incremental plastic strain tensor
$\{d\epsilon_p\}_e$	Effective plastic strain increment
θ	Angle, circumferential coordinate
ϕ	A stress invariant parameter (Chapter IV)
	Poission's ratio

SYMBOL	DEFINITION
ξ, η	Natural coordinate system of isoparametric element
$\xi_i, \eta_j, \zeta_k, \xi_p, \eta_q$	Sampling points
ρ	Mass density
$\gamma_{xy}, \gamma_{yz}, \gamma_{zx}$	Shear strains in xy,yz and zx planes
γ_{rz}	Shear strain in rz plane
σ	Stress vector
$\bar{\sigma}$	Virtual stress vector
σ^e	Element stress vector
σ_{ij}	Stress tensor
σ'_{ij}	Deviatoric stress tensor
$\sigma_1, \sigma_2, \sigma_3$	Principal stresses
σ_e	Effective stress
σ_y	Current yield stress
σ_y^0	Uniaxial yield stress
σ_m	Mean stress
$d\sigma_e$	Incremental stress assuming elastic behaviour
$\sigma_{xx}, \sigma_{yy}, \sigma_{zz}$	Normal stresses along x,y and z axes
$\sigma_{rr}, \sigma_{\theta\theta}$	Normal stresses along r and0 directions
$\tau_{xy}, \tau_{yz}, \tau_{zx}$	Shear stresses in xy,yz and zx planes
τ_{rz}	Shear stress in rz plane
$\kappa(U)$	Residual forces
$\kappa(U)^r$	Residual forces at r^{th} iteration
$\kappa_i(U)^r$	Residual forces at r^{th} iteration at node i
ΔU	Correction factor for displacement
$d\lambda$	Proportionality constant (Chapter III)

SYMBOL	DEFINITION
$\{\ \}$	A rectangular or square matrix
$\{\ \}^{-1}$	Matrix inverse
$\{\ \}^{T}$	Matrix transpose
$\{\ \}^{-T}$	Matrix inverse transpose
Σ	Summation of the mathematical terms that follow
Σ_e	Summation over all the finite elements in the system
CPU	Central Processing Unit

LIST OF FIGURES

FIGURE PAGE

LIST OF TABLES

INTRODUCTION

With the current rapid technological progress the Theory of Plasticity has been brought forcibly into the forefront of engineering application and design. Although the Theory of Plasticity has advanced considerably, rigorous solutions to a large class of problems are still not available. The application of Finite Element Techniques seems to be an ideal choice for solving such problems, as the technique is an accepted versatile tool for linear analysis and ideally suited for employing the powerful method of successive elastic solutions with incremental Theory of Plasticity. The purpose of this thesis is to introduce two Finite Element Programs, one for linear strain hardening materials, and the other for nonlinear type strain hardening materials and to prove the accuracy of the proposed solution techniques for two dimensional materially nonlinear problems.

Chapter I forms an introduction to two dimensional continuum problems with the basic theory for isoparametric elements. Chapter II discusses the general nonlinear problem and the solution technique adopted for numerical solution. Chapter III considers two dimensional elasto-plastic problems and the basic theoretical expressions for a general continuum. In Chapter IV the expressions for plane stress / plane strain and axisymmetric situations are manipulated into forms suitable for numerical analysis.

Chapter V describes the computer procedures for Finite Element Programs presented in Appendices C and D. Instructions for the use of these programs are given in Appendices A and B. In Chapter VI several test examples are considered with a discussion on the accuracy of the program results. The accuracy of the programs are established by comparing the results with established solutions. In Chapter VII some additional elasto-plastic solutions are given with increasing order of stress concentration and complexity. Concluding remarks are presented in Chapter VIII.

CHAPTER I

CONCEPT OF FINITE ELEMENT METHOD

The Finite Element Method is a numerical procedure for solving a continuum mechanics problem with an accuracy acceptable to engineers. In this method an approximate solution is attempted by dividing the continuum into a discrete number of finite elements. These elements are connected at a discrete number of points along their periphery known as nodes. In structural mechanics applications displacement based finite element formulation is generally used, wherein displacement of the continuum is described in terms of the displacement of the element nodes. The programs in the Appendices are written using this method of formulation.

If the equilibrium of a general three dimensional body such as in Fig. 1.1 is considered, the external forces acting on the body are:

1. Surface tractions $f^S = \begin{bmatrix} f_x^s \\ f_y^s \\ f_z^s \end{bmatrix}$

2. Body forces $f^b = \begin{bmatrix} f_x^b \\ f_y^b \\ f_z^b \end{bmatrix}$

3. Concentrated forces $f^I = \begin{bmatrix} F_x^I \\ F_y^I \\ F_z^I \end{bmatrix}$ I.1

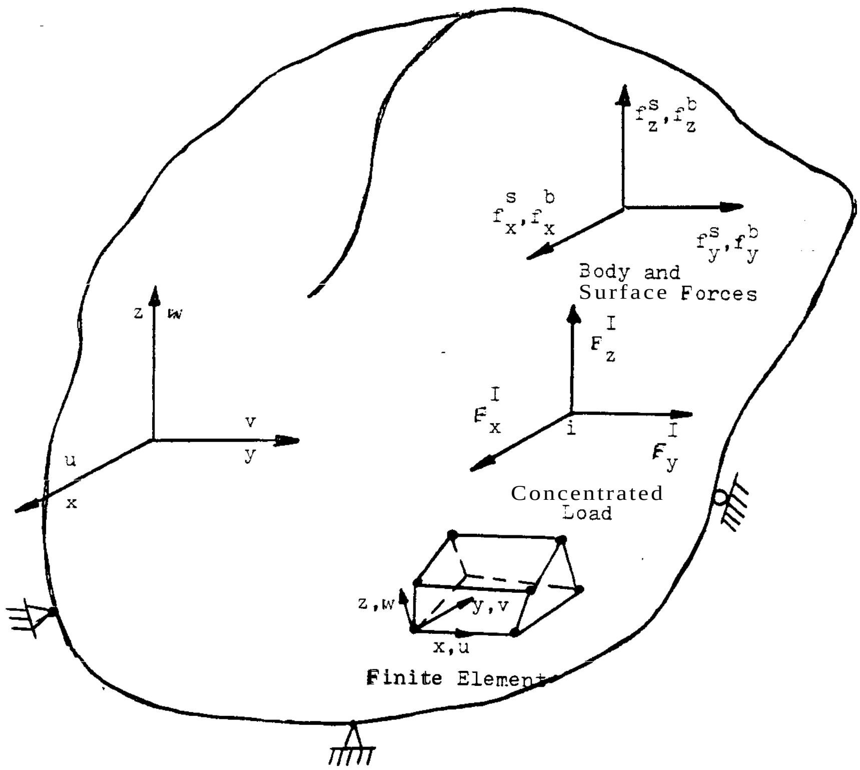


Fig. I.1 General Three Dimensional Body.

The displacement of the body from the unloaded configuration is denoted by:

$U^T = \{ u \ v \ w \}$

The strain components corresponding to U are,

$$\epsilon^T = \{\epsilon_{xx} \quad \epsilon_{yy} \quad \epsilon_{zz} \quad \gamma_{xx} \quad \gamma_{yy} \quad \gamma_{zz}\} \qquad \text{I.3}$$

The corresponding stresses are,

$$\sigma^T = \{\sigma_{xx} \quad \sigma_{yy} \quad \sigma_{zz} \quad \tau_{xy} \quad \tau_{yz} \quad \tau_{zx}\} \qquad \text{I.4}$$

In order to calculate the response of the body to the externally applied forces, the governing equilibrium equations are to be obtained. These equilibrium equations are developed first at element level. The element equilibrium equations can be obtained by the use of the principle of virtual displacements. This principle states that for the equilibrium of a body under any compatible, small virtual displacements, the total internal virtual work must be equal to the external virtual work. Therefore we have,

$$\int_V \bar{\epsilon}^T \sigma \, dV = \int_V \bar{U}^T f^b \, dV + \int_S \bar{U}^{S^T} f^s \, dS + \Sigma_1 \bar{U}^{I^T} f^I \qquad \text{I.5}$$

where,

$$\bar{\epsilon}^T = \text{Virtual strain}$$

$$= \{\bar{\epsilon}_{xx} \quad \bar{\epsilon}_{yy} \quad \bar{\epsilon}_{zz} \quad \bar{\gamma}_{xx} \quad \bar{\gamma}_{yy} \quad \bar{\gamma}_{zz}\} \qquad \text{I.6}$$

$$\bar{U}^T = \text{Virtual displacement}$$

$$= \{\bar{u} \quad \bar{v} \quad \bar{w}\} \qquad \text{I.7}$$

The subscript S denotes that surface displacements are considered and the subscript I denotes the displacements at the point of application of concentrated forces.

The equation in (I.5) is an expression of equilibrium and also contains the compatibility and constitutive requirements in the finite element formulation. Although the virtual work equation is written in the global coordinate system x,y,z of the body it is equally valid in any other system of coordinates.

In the finite element displacement method, displacement with in an element is,

$$U^e = N^e \hat{U} \quad , \qquad I.8$$

where {N} is the set of interpolation functions termed the shape functions. The subscript e denotes an element, and $\hat{U}$ is a vector of three global displacement components u_i, v_i, w_i at all nodal points. If there are M finite element nodal points, U is a vector dimension 3M. More generally,

$$\{ \hat{U} \}^T = \{ \hat{U}_1 \ \hat{U}_2 \ \hat{U}_3 \ldots \hat{U}_M \} \qquad I.9$$

Although all nodal point displacements are listed in (I.9) for a given element only the displacements at the nodes of the element affect the displacement and strain distribution within the element. The element strains are evaluated using

$$\epsilon^e = B^e \hat{U} \quad , \qquad I.10$$

where B^e is strain matrix. Finally,

$$\sigma^e = D^e \epsilon^e \quad , \qquad I.11$$

where D^e is elasticity matrix of element e and the material law can vary from element to element.

The equlibrium equations corresponding to the nodal point displacements of the assembly of finite elements can now be written as a sum of integrations over the volume and areas of all finite elements:

$$\Sigma_e \int_V \bar{\epsilon}^{e^T} \sigma^e \, dV = \Sigma_e \int_V \bar{U}^{e^T} f^{b(e)} \, dV + \Sigma_e \int_S \bar{U}^{s(e)^T} f^{s(e)} \, dS + \Sigma_e \bar{U}^{I^T} f^I \quad , \qquad \text{I.12}$$

where Σ_e denotes summation over all the elements. The integrations in (I.12) are performed over the element volumes and surfaces, and for convenience we may use different element coordinate systems in the calculations. Substituting for element displacements, strains, and stresses from (1.8) to (I.11) into (I.12),

$$\hat{U}^T \left[\Sigma_e \int_V B^{e^T} D^e B^e dV \right] \hat{U} = \hat{U}^T \left[\left\{ \Sigma_e \int_V N^{e^T} f^{b(e)} dV \right\} + \left\{ \Sigma_e \int_S N^{(s)e^T} f^{s(e)} dS \right\} + F \right] \qquad \text{I.13}$$

The surface shape function matrix $\{N^{(s)e}\}$ is obtained from shape function matrix $\{N^e\}$ by substitution of element surface coordinates. $\{F\}$ is a vector of the externally applied forces to the nodes of the element in the structure. The i^{th} component in $\{F\}$ is the concentrated nodal force corresponding to the i^{th} displacement component in $\hat{U}$.

The unknown nodal point displacements are obtained from (I.13) invoking virtual displacement theorem and imposing unit virtual displacements in turn at all displacement components.

$$\therefore \{ \bar{\hat{U}}^T \} = \{ I \} \quad . \qquad \text{I.14}$$

Letting $\{\hat{U}\} = \{U\}$ I.15

the equlibrium equations reduce to

$$\{K\}\{U\} = \{R\}, \qquad \text{I.16}$$

where $\{K\}$ = stiffness matrix of the structure

$$= \Sigma_e \int_V B^{e^T} D^e B^e \, dV \qquad , \qquad \text{I.17}$$

and $\{R\}$ = equivalent nodal forces of the structure

$$= \{R_B\} + \{R_S\} + \{R_C\} \qquad . \qquad \text{I.18}$$

In the expression (I.18),

$$R_B = \Sigma_e \int_V N^{e^T} f^{b(e)} dV = \text{body forces} , \qquad \text{I.19}$$

$$R_S = \Sigma_e \int_S N^{(S)e^T} f^{s(e)} dS = \text{surface forces}, \qquad \text{I.20}$$

and R_C = F = concentrated loads. 1.21

The formulation of equlibrium equations (1.16) include the assembly process to obtain the structure matrices from the element matrices, referred to as the direct stiffness method. The resulting linear simultaneous equations are then solved for the unknown nodal variables using the frontal solution technique. Stresses can then be determined.

Although equlibrium equations (1.16) are written for general three dimensional analysis, the scope of this thesis is restricted to two dimensional plane stress / plane strain and axisymmetric problems. The appropriate displacement, stress, and strain variables as well as elasticity matrices for isotropic material are listed in Table 1.1.

TABLE I.1

LIST OF VARIABLES AND ELASTICITY MATRICES

Type of problem	Displacement	Strain vector $\{\epsilon\}^T$	Stress vector $\{\sigma\}^T$	Elasticity matrix $\{D\}$
Plane stress	u,v	$\{\epsilon_{xx}\ \epsilon_{yy}\ \gamma_{xy}\}$	$\{\sigma_{xx}\ \sigma_{yy}\ \tau_{xy}\}$	$\frac{E}{(1-\nu^2)}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{(1-\nu)}{2} \end{bmatrix}$
Plane strain	u,v	$\{\epsilon_{xx}\ \epsilon_{yy}\ \gamma_{xy}\}$	$\{\sigma_{xx}\ \sigma_{yy}\ \tau_{xy}\}$	$\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} (1-\nu) & \nu & 0 \\ \nu & (1-\nu) & 0 \\ 0 & 0 & \frac{(1-2\nu)}{2} \end{bmatrix}$
Axisymmetric	u,w	$\{\epsilon_{rr}\ \epsilon_{r\theta}\ \epsilon_{zz}\ \gamma_{rz}\}$	$\{\sigma_{rr}\ \sigma_{r\theta}\ \sigma_{zz}\}$	$\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} (1-\nu) & \nu & 0 & 0 \\ \nu & (1-\nu) & \nu & 0 \\ 0 & \nu & (1-\nu) & 0 \\ 0 & 0 & 0 & \frac{(1-2\nu)}{2} \end{bmatrix}$

Notation:

$\epsilon_{xx} = \frac{\partial u}{\partial x}$; $\epsilon_{yy} = \frac{\partial v}{\partial y}$; $\epsilon_{zz} = \frac{\partial w}{\partial z}$; $\epsilon_{rr} = \frac{\partial u}{\partial r}$; $\epsilon_{\theta\theta} = \frac{u}{r}$; $\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}$; $\gamma_{rz} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial r}$

E = Young's modulus ; ν = Poission's ratio

The element stiffness matrix calculations in the programs in Appendices C & D are done using a single type of element known as the isoparametric element, as these elements are versatile, well tried, and tested. The library of elements in the program is shown in Fig. 1.2 with their special coordinate system (ξ, η),

In isoparametric element formulation the shape functions, which are defined in element natural coordinate system (ξ, η) define the element coordinates and element displacement in terms of their nodal values. The displacements can be expressed as,

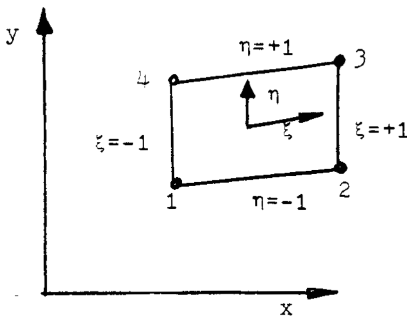
$$U^e = \sum^n N^e \hat{U}^e_i \quad , \qquad 1.22$$

where N^e is matrix of shape functions and $\hat{U}^e$ is vector of nodal displacements. Similarly,

$$\begin{bmatrix} x^e \\ y^e \end{bmatrix} = \sum_{i=1}^{n} \begin{bmatrix} N^e_i & 0 \\ 0 & N^e_i \end{bmatrix} \begin{bmatrix} x^e_i \\ y^e_i \end{bmatrix} \quad . \qquad I.23$$

For axisymmetric problems, x & y are replaced by r & z in equation (1.23). The Jacobian matrix which is used in coordinate transformation is evaluated as,

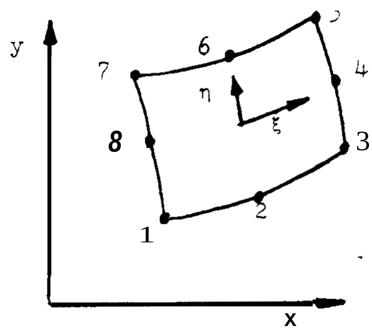
$$J^e = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} \frac{\partial N^e_i}{\partial \xi} x^e_i & \sum_{i=1}^{n} \frac{\partial N^e_i}{\partial \xi} y^e_i \\ \sum_{i=1}^{n} \frac{\partial N^e_i}{\partial \eta} x^e_i & \sum_{i=1}^{n} \frac{\partial N^e_i}{\partial \eta} y^e_i \end{bmatrix} \quad . \qquad I.24$$



$$N_i^e(\xi,\eta) = \frac{(1+\xi\xi_i)(1+\eta\eta_i)}{4} .$$

Local Node	ξ_i	η_i
1	-1	-1
2	+1	-1
3	+1	+1
4	-1	+1

(a) The Linear Element



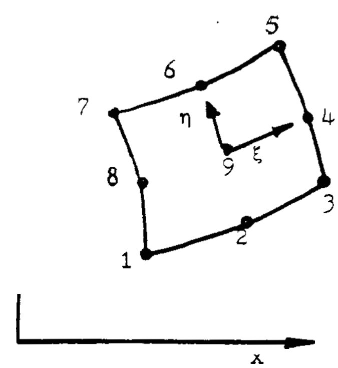
For corner nodes, 1,3,5,7

$$N_i^e = \frac{(1+\xi\xi_i)(1+\eta\eta_i)(\xi\xi_i+\eta\eta_i-1)}{4} .$$

For midside nodes, 2,4,6,8

$$N_i^e = \frac{\xi_i^2}{2}(1+\xi\xi_i)(1-\eta^2)+\frac{\eta_i^2}{2}(1+\eta\eta_i)(1-\xi^2) .$$

(b) Eight Noded Parabolic Element



For corner nodes, 1,3,5,7

$$N_i^e = \frac{(\xi^2+\xi\xi_i)(\eta^2+\eta\eta_i)}{4} .$$

For midside nodes,

$$N_i^e = \frac{\eta_i^2}{2}(\eta^2-\eta\eta_i)(1-\xi^2)+\frac{\xi_i^2}{2}(\xi^2-\xi\xi_i)(1-\eta^2) .$$

For central node,

$$N_i^e = (1-\xi^2)(1-\eta^2) .$$

(c) Nine Noded Parabolic Element

For (b)&(c) the following applies:

Local Node	ξ_i	η_i
1	-1	-1
2	0	-1
3	+1	-1
4	+1	0
5	+1	+1
6	0	+1
7	-1	+1
8	-1	0
9	0	0

Fig. I.2 Library of Elements.

The strain displacement relationship is expressed as,

$$\epsilon^e = \sum_{i=1}^{n} B_i^e \hat{U}_i^e \quad , \qquad I.25$$

where B^e is the strain matrix.

The elemental volume is given as,

$$dV = t^e \det J^e d\xi \, d\eta \, , \qquad I.26$$

The expressions for $\hat{U}_i^e$, B_i^e and dV are given in Table I.2.

The Cartesian shape function derivatives used in the strain matrix B_i^e in Table I.2 may be obtained by the chain rule of differentiation:

$$\frac{\partial N_i^e}{\partial x} = \frac{\partial N_i^e}{\partial \xi} \frac{\partial \xi}{\partial x} + \frac{\partial N_i^e}{\partial \eta} \frac{\partial \eta}{\partial x}$$

and

$$\frac{\partial N_i^e}{\partial y} = \frac{\partial N_i^e}{\partial \xi} \frac{\partial \xi}{\partial y} + \frac{\partial N_i^e}{\partial \eta} \frac{\partial \eta}{\partial y} \quad . \qquad I.27$$

The terms $\frac{\partial \xi}{\partial x}$, $\frac{\partial \eta}{\partial x}$, $\frac{\partial \xi}{\partial y}$, and $\frac{\partial \eta}{\partial y}$ may be determined from the inverse of the Jacobian matrix (I.24).

Let us now consider the evaluation of stiffness matrix {K} of the structure defined in (I.17). The integration is performed in the element natural coordinate system for each element and summed up for all elements of the structure.

$$\{K\} = \Sigma_e \int_{-1}^{+1} \int_{-1}^{+1} B^{e^T} D^e B^e t^e \det J^e \, d\xi \, d\eta \, . \qquad 1.28$$

The integration is to be performed numerically. If the integrand in (1.28) is denoted as,

$$B^{e^T} D^e B^e t^e \det J^e = T^e \,, \qquad \text{I.29}$$

then,

$$\{K\} = \Sigma_e \int_{-1}^{+1}\int_{-1}^{+1} T^e \, d\xi \, d\eta \,. \qquad \text{I.30}$$

The numerical integration is carried out utilizing Gauss quadrature. In two dimensions we find the quadrature formula to yield the expression for stiffness matrix as,

$$\{K\} = \Sigma_e \left\{ \Sigma_i \; \Sigma_j \; w_i \; w_j \; T\,(\xi_i, \eta_j) \right\}, \qquad \text{I.31}$$

where w_i, and w_j are weights for Gauss quadrature at sampling points ξ_i and η_j. Similarly in three dimensions we have,

$$\{K\} = \Sigma_e \left[\Sigma_i \; \Sigma_j \; \Sigma_k \;\; w_i \; w_j \; w_k \; T(\xi_i, \eta_j, \zeta_k) \right]. \qquad \text{I.32}$$

While it is not necessary to use the same number of Gauss points in each direction, it is most common to do so. Similarly, numerical integration is to be done for each element for determining the body forces R_B (I.19) and surface forces R_S (1.20).

$$R_B = \Sigma_e \int_V N^{e^T} f^{b(e)} dV$$

$$= \Sigma_e \int_{-1}^{+1} \int_{-1}^{+1} N^{e^T} f^{b(e)} t^e \det J^e \, d\xi \, dq.$$

$$= \text{body forces} \qquad \text{I.33}$$

If the integrand in (I.33) is denoted as,

$$g^e = N^{e^T} f^{b(e)} t^e \det J^e ,$$

$$R_B = \Sigma_e \left\{ \int_{-1}^{+1} \int_{-1}^{+1} g^e \, d\xi \, d\eta \right\} \quad \text{I.34}$$

$$= \Sigma_e \left\{ \Sigma_i \Sigma_j w_i w_j g(\xi_i, \eta_j) \right\} . \quad \text{I.35}$$

TABLE I.2

NODAL DISPLACEMENTS, STRAIN MATRICES AND ELEMENTAL VOLUMES OR AREAS

Type of problem	Nodal displacement $\hat{U}_i^e$	Strain Matrix B_i^e	dV / dS
Plane stress	$\begin{bmatrix} u_i^e \\ v_i^e \end{bmatrix}$	$\begin{bmatrix} \frac{\partial N_i^e}{\partial x} & 0 \\ 0 & \frac{\partial N_i^e}{\partial y} \\ \frac{\partial N_i^e}{\partial y} & \frac{\partial N_i^e}{\partial x} \end{bmatrix}$	$t^e \det J^e \, d\xi \, d\eta$
Plane strain	$\begin{bmatrix} u_i^e \\ v_i^e \end{bmatrix}$	$\begin{bmatrix} \frac{\partial N_i^e}{\partial x} & 0 \\ 0 & \frac{\partial N_i^e}{\partial y} \\ \frac{\partial N_i^e}{\partial y} & \frac{\partial N_i^e}{\partial x} \end{bmatrix}$	$\det J^e \, d\xi \, d\eta$
Axisymmetric	$\begin{bmatrix} u_i^e \\ w_i^e \end{bmatrix}$	$\begin{bmatrix} \frac{\partial N_i^e}{\partial r} & 0 \\ \frac{N_i^e}{r} & 0 \\ 0 & \frac{\partial N_i^e}{\partial z} \\ \frac{\partial N_i^e}{\partial z} & \frac{\partial N_i^e}{\partial r} \end{bmatrix}$	$2\pi r \det J^e \, d\xi \, d\eta$

In the finite element analysis of structures by the displacement method, the only permissible form of loading, other than initial stressing, is by the prescription of concentrated loads at the nodal points. Consequently, forms of loading such as gravity action (body forces), pressures assigned to element surfaces (surface forces) must be reduced to equivalent nodal loads.

Point loads:

For each particular node the applied concentrated loads are associated with one of the elements attached to it and with the appropriate degrees of freedom of that element.

Gravity loading:

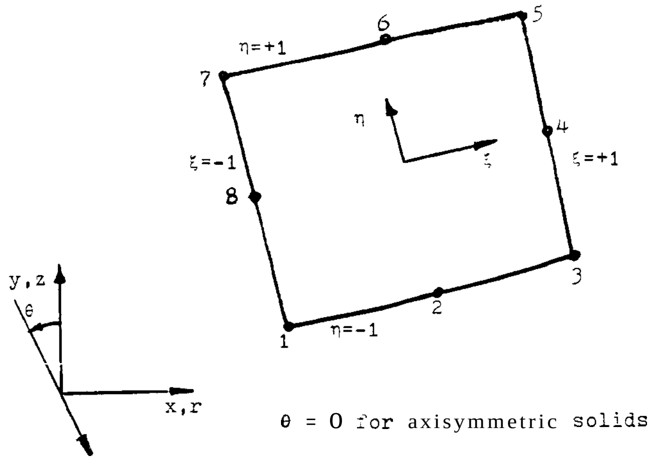
Gravity forces are equivalent to a body force / unit volume acting within the solid in the direction of the gravity axis. For plane stress or plane strain problems, the direction in which the gravity acts need not coincide with either of the coordinate axes. Fig. I.3 (a) illustrates an isoparametric element subjected to gravity forces at an angle θ. Invoking the principle of virtual work, we obtain the expression:

$$\begin{bmatrix} P_{xi} \\ P_{yi} \end{bmatrix} = \int_{V^e} N_i^e \, \rho \, g \begin{bmatrix} \sin(\theta) \\ -\cos(\theta) \end{bmatrix} dV, \qquad \text{I.36}$$

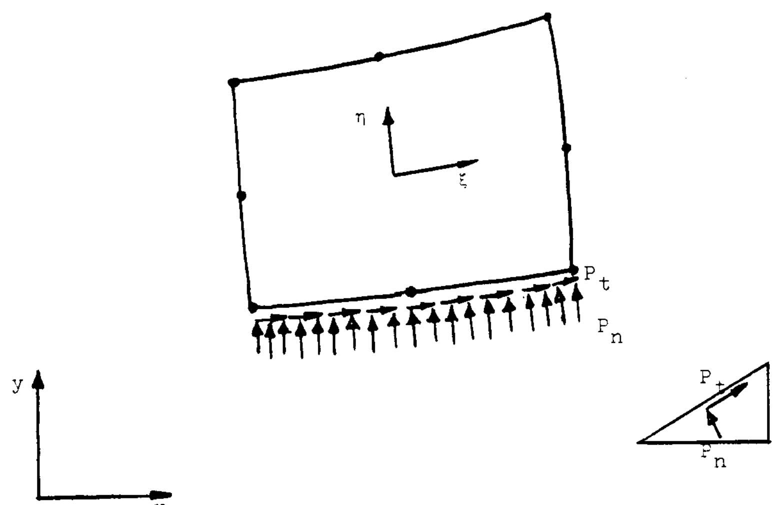
which yields the equivalent nodal forces due to a gravitational acceleration g acting on a material of mass density ρ. The subscript "i" ranges over the node numbers. It is again essential to resort to a Gaussian numerical integration technique, resulting in:

$$\begin{bmatrix} P_{xi} \\ P_{yi} \end{bmatrix} = \sum_{p=1}^{NGAUS} \sum_{q=1}^{NGAUS} \rho g t \begin{bmatrix} \sin\theta \\ -\cos\theta \end{bmatrix} N_i^e (\xi_p, \eta_q)\, w_p\, w_q \det J^e . \qquad \text{I.37}$$

For axisymmetric problems t is replaced by $2\pi r_p$ where r_p is the radial distance to the Gauss point under consideration.



(a) Specification of the Gravity Axis for Two Dimensional Problems



(b) Normal and Tangential Loads per Unit Length Applied to a Parabolic Isoparametric Element

Fig. I.3 Element Loading.

Distributed edge loading:

Any element edge will be allowed to have a distributed loading per unit length in a normal and tangential direction prescribed to it as shown in Fig. I.3 (b). These distributed forces need not be constant, but can vary independently along the element edge. Since parabolic isoparametric elements are being employed, then at best a parabolic loading distribution can be accomodated. The variation will be defined by prescribing the normal and tangential values at three nodal points forming the element edge. The three nodes forming the loaded edge must be in an anticlockwise sequence. A pressure normal to the face is assumed to be positive if it acts into the element. A tangential load is positive if it acts in an anticlockwise direction. Invoking the principle of virtual work, the consistent nodal forces for node i can be written as:

$$P_{xi} = \int_S N_i^e \left(P_t \frac{\partial x}{\partial \xi} - P_n \frac{\partial y}{\partial \xi}\right) d\xi$$

$$P_{yi} = \int_S N_i^e \left(P_n \frac{\partial x}{\partial \xi} + P_t \frac{\partial y}{\partial \xi}\right) d\xi \quad , \qquad \text{I.38}$$

where integration is done along the element loaded edge, arbitarily considered as $\eta = -1$ in Fig. I.3(b). If a distributed load acts on the interface between two elements, then the load can be considered to be acting on either one or the other element.

CHAPTER II

BASIC NUMERICAL SOLUTION PROCESS FOR NONLINEAR PROBLEMS

In the Finite Element formulation of Chapter I equlibrium equations (1.16) were derived assuming that the displacements are infinitesimally small and the material is linearly elastic. It was also assumed that the boundary conditions remain unchanged during the application of loads. Under these assumptions the equations,

$$\{K\} \{U\} = \{R\} \qquad I.16$$

correspond to a linear analysis of the structure, since the displacement vector $\{U\}$ is linearly proportional to load vector $\{R\}$. The assumption of small displacements has entered in the evaluation of stiffness matrix $\{K\}$, and load vector $\{R\}$ as all integrations were done over the original volume of finite elements. The strain matrix $\{B\}$ of each element was considered to be constant and independent of element displacements. Elasticity matrix $\{D\}$ was considered to be constant. Boundary conditions remained unchanged. These basic assumptions explain in a way what is generally meant by a nonlinear analysis. The scope of the present thesis is restricted to materially nonlinear only formulation. The nonlinear effect lies in the nonlinear stress-strain relationship. The displacements and strains are small; therefore the usual engineering stress and strain measures are employed. Since in nonlinear problems coefficients of stiffness matrix {K} depend on the unknown nodal displacement vector {U} direct solution of equation (I.16) is generally impossible and some sort of iterative procedure must be adopted.

The most frequently used iteration schemes for the solution of nonlinear finite element equations are some form of Newton - Raphson iteration. The finite element equlibrium equations amount to finding the solution of equations,

$$\varkappa(U) = \{K\}\{U\} - \{R\} = 0 \ , \qquad \text{II.1}$$

where $\varkappa(U)$ can be interpreted as a measure of departure from the equlibrium. If a true solution to the problem exists at $U^r + \Delta U^r$ after r^{th} iteration, then the Newton - Raphson iteration schemes yield,

$$\varkappa_i(U)^r = \sum_{j=1}^{M} \left(\frac{\partial \varkappa_i}{\partial U_j}\right)^r \Delta U_j^r \ , \qquad \text{II.2}$$

where,

M = total number of variables in the system,

and

i = the particular node number.

Substituting (II.1) in (II.2),

$$\varkappa(U)^r = \left[K(U) + K'(U)\right] \Delta U^r \ . \qquad \text{II.3}$$

The Newton-Raphson process can finally be written in the form as:

$$\Delta U^r = \left[K(U^r) + K'(U^r)\right]^{-1} \varkappa(U)^r \ . \qquad \text{II.4}$$

This allows the correction vector of unknown nodal displacements U to be obtained from the deviation from the equlibrium condition. An iterative scheme must be followed.

The tangential stiffness method:

In structural applications {K} is equal to the local gradient of the force / displacement relationship of the structure and is called the tangential stiffness. Since an incremental analysis is performed in such cases, the problem is linearized and written as:

$$\Delta U^{r} = \left[K(U^{r})\right]^{-1} \varkappa(U)^{r}. \qquad \text{II.5}$$

A trial value U^{0} is assumed; the tangential stiffness $K(U^{0})$ corresponding to this displacement is then obtained; the departure from equlibrium value $\varkappa(U)^{0}$ is determined; then the correction ΔU^{0} is determined from (II.5); and the improved approximation to the unknown now is:

$$U^{1} = U^{0} + \Delta U^{0}$$

The iteration process is then continued until the solution converges.

The initial stiffness method:

In the tangential stiffness method the computational cost per iteration, particularly for large order systems, for the calculation and factorization of the tangent stiffness matrix can be expensive. In the initial stiffness method, complete equation solution need only be performed for the first iteration and subsequent approximations to the solution performed thru:

$$\Delta U^{r} = \left[K(U^{0})\right]^{-1} \varkappa(U)^{r} . \qquad \text{II.6}$$

This initial stress method corresponds to a linearization of the response about the initial configuration of the finite element system and may result in a very slowly convergent or even divergent solution. An approach somewhat between the tangential stiffness and the initial stiffness methods is sometimes employed, wherein the stiffness matrix is updated at selected iterative intervals, when a prior knowledge of the system behaviour is known.

Solution convergence criteria:

The convergence criterion adopted for the termination of the iteration is by measuring the out of balance load vector at the end of each iteration and setting the norm of the out-of-balance load vector to be within a preset tolerance of the norm of the total applied forces as:

$$\frac{\sqrt{\sum_{j=1}^{M} \left(\kappa_j(U)^r \right)^2}}{\sqrt{\sum_{j=1}^{M} (R_j)^2}} \times 100 \leq T , \qquad \text{II.7}$$

where

r = iteration number,

and

M = total number of degrees of freedom in the structure.

CHAPTER III

THE MATHEMATICAL THEORY OF PLASTICITY

Whereas the stresses and strains are linearly related by Hooke's law in the elastic range, the relationship is nonlinear in the plastic range as is evident from the uniaxial stress / strain curve. The mathematical theory of plasticity provides a theoretical relationship between stresses and strains for a material loaded in the plastic range. The three ingredients necessary to formulate the plasticity theory are:

1. Relationship between stresses and strains in the elastic range
2. Yield criteria for deciding which combination of multiaxial stresses will cause yielding
3. Relations between stresses and strains when plastic flow is occuring

Using the tensor subscript notation, the generalization of Hooke's law leads to,

$$\epsilon_{ij} = \frac{\sigma_{ij}}{2G} - \frac{\delta_{ij}}{E}\nu(\sigma_{xx}+\sigma_{yy}+\sigma_{zz}) , \qquad \text{III.1}$$

which can also be expressed as,

$$\epsilon'_{ij} = \frac{S_{ij}}{} . \qquad \text{III.2}$$

where ϵ'_{ij} and S_{ij} are strain and stress deviator tensors, respectively.

Yield criteria can be defined by the relation,

$$F(\sigma_{ij}) = k \ , \qquad \text{III.3}$$

where the function F can be looked upon as a loading function and k is a yield function or a strain hardening function, and would depend on the complete previous stress and strain history of the material and its strain hardening properties. If isotropy is assumed then rotation of axes must not affect yielding. Therefore, the relation (III.3) can be written as,

$$F(\sigma_1 \ \sigma_2 \ \sigma_3) = k \ . \qquad \text{III.4}$$

Since it is always assumed that hydrostatic tension or compression does not influence yielding, we can assume that only stress deviators enter into the yield function,

$$F(S_1 \ S_2 \ S_3) = k \ . \qquad \text{III.5}$$

Alternatively, S_1, S_2, and S_3 can be written in terms of the invariants of deviatoric stresses as,

$$J'_1 = S_1+S_2+S_3 = 0$$

$$J'_2 = \frac{1}{2}\,(S_1^2+S_2^2+S_3^2) = -(S_1S_2+S_2S_3+S_3S_1)$$

$$J'_3 = \frac{1}{3}\,(S_1^3+S_2^3+S_3^3) = S_1S_2S_3 \ . \qquad \text{III.6}$$

The yield criteria can now be written as,

$$F(J'_2 \ , J'_3) = k \ . \qquad \text{III.7}$$

The Tresca criterion can be reduced to the simple form,

$$(\sigma_1-\sigma_3) = 2k \ , \qquad \text{111.8}$$

where the maximum and minimum principal stresses are known apriori.

The Von Mises criterion is associated with equation,

$$J_2' = k^2 , \qquad \text{III.9}$$

in which k is a material parameter to be determined.

For most materials these two laws are used. The second deviatoric stress invariant can be written as,

$$J_2' = \frac{1}{2} \sigma'_{ij} \sigma'_{ij} . \qquad \text{III.10}$$

If we define effective stress as

$$\sigma_e = \sqrt{\frac{3}{2}} \, (\sigma'_{ij} \sigma'_{ij})^{\frac{1}{2}} , \qquad \text{III.11}$$

then yield criteria may be written as

$$\sigma_e = \sqrt{3} \, (J_2')^{\frac{1}{2}} = \sqrt{3} \, k . \qquad \text{III.12}$$

σ_e can also be explicitly written as

$$\sigma_e = \frac{1}{\sqrt{2}} \left[(\sigma_{xx} - \sigma_{yy})^2 + (\sigma_{yy} - \sigma_{zz})^2 + (\sigma_{zz} - \sigma_{xx})^2 + 6(\tau_{xy}^2 + \tau_{yz}^2 + \tau_{zx}^2) \right]^{\frac{1}{2}} . \qquad \text{III.13}$$

For uniaxial tensile test σ_e becomes the σ_{xx} and the physical meaning of constant k can be obtained under uniaxial tensile test when

$$\sigma_e = \sigma_y = \sqrt{3} \, k . \qquad \text{III.14}$$

Since the plastic strains are dependent on the loading path, in general it becomes necessary to compute differentials or increments of plastic strain throughout the loading history and then obtain the total strains by summation. Starting with the definition of work hardening and postulating the existence of a loading function and linearity between increments of stress and increments of strain, the general flow rule for a strain hardening material is expressed as:

$$\{d\epsilon_{ij}\}_p = d\lambda \frac{\partial F}{\partial \sigma_{ij}} . \qquad \text{III.15}$$

This plastic strain increment vector must always be normal to the yield surface. $d\lambda$ is a proportionality constant as yet undetermined , and depends on the stress, strain and their history in general. During any infinitesimal increment of stress, changes of strain are assumed to be divisible into elastic and plastic parts. Hence,

$$\{d\epsilon_{ij}\}_T = \{d\epsilon_{ij}\}_e + \{d\epsilon_{ij}\}_p . \qquad \text{III.16}$$

As

$$\{d\epsilon_{ij}\}_e = \{D\}^{-1}\{d\sigma\} , \qquad \text{III.17}$$

where $\{D\}$ is elasticity matrix, we can write (III.16) as

$$\{d\epsilon\}_T = \{D\}^{-1}\{d\sigma\} + d\lambda \frac{\partial F}{\partial \sigma_{ij}} . \qquad \text{111.18}$$

From (111.3), we have

$$f(\sigma_{ij}, k) = F(\sigma_{ij}) - k = 0, \qquad \text{III.19}$$

where k keeps changing as material work hardens and k can be written as,

$$k = f(w^p) = f\left(\int \sigma_{ij} (d\epsilon_{ij})_p\right), \qquad \text{III.20}$$

using the work hardening hypothesis. k can also be written as

$$k = f(\epsilon_p) = f(\int d\epsilon_p), \qquad \text{III.21}$$

using strain hardening hypothesis.

Differentiating equation (III.19),

$$df = \frac{dF}{\partial \sigma_{ij}} d\sigma_{ij} + \frac{df}{d(k)} dk \quad . \qquad \text{III.22}$$

When plastic yielding is occuring, df is zero. Equation (111.22) can be written as

$$\{a\}^T d\sigma - A\, d\lambda = 0 \, , \qquad \text{III.23}$$

where

$$\{a\}^T = \frac{\partial F}{\partial \sigma_{ij}} \, , \qquad \text{III.24}$$

and

$$A = -\frac{1}{d\lambda} \frac{\partial F}{\partial k} dk \qquad \text{III.25}$$

Let

$$\{d_D\} = \{D\} \{a\} \quad . \qquad \text{III.26}$$

Premultiplying both sides of equation (III.18) with

$$\{d_D\}^T = \{a\}^T \{D\} \qquad \text{III.27}$$

we get

$$\{a\}^T \{D\} \{d\epsilon_{ij}\}_T = \{a\}^T d\sigma + \{a\}^T \{D\} d\lambda \frac{\partial F}{\partial \sigma_{ij}} \quad . \qquad \text{III.28}$$

From (III.23),

$$\{a\}^T d\sigma = A\, d\lambda$$

Therefore,

$$\{a\}^T \{D\} \{d\epsilon_{ij}\}_T = d\lambda \;\; A \{a\} + \{a\}^T \{D\} \frac{\partial F}{\partial \sigma_{ij}} \, ,$$

and

$$d\lambda = \frac{\{a\}^T \{D\} \{d\epsilon_{ij}\}_T}{(A + a^T D\, a)} \; . \qquad \text{III.29}$$

On substituting (111.29) in (III.18),

$$\{d\epsilon_{ij}\}_T = \{D\}^{-1} \{d\sigma\} + \frac{\{a\}^T\{D\} \{d\epsilon_{ij}\}_T\{a\}}{(A+a^TDa)}$$

$$\{d\sigma\} = \{d\epsilon_{ij}\}_T \left[\{D\} - \frac{d_D^T d_D}{(A+a^TDa)} \right]$$

$$= \{d\epsilon_{ij}\}_T \left[\{D\} - \frac{d_D^T d_D}{(A+d_D^T a)} \right] \qquad \text{III.30}$$

$$= \{D_{ep}\} \{d\epsilon_{ij}\}_T , \qquad \text{III.31}$$

where

$$\{D_{ep}\} = \left[D - \frac{d_D^T d_D}{(A+d_D^T a)} \right] . \qquad \text{III.32}$$

The elasto-plastic matrix $\{D_{ep}\}$ takes the place of elasticity matrix $\{D\}$ in incremental analysis. It is symmetric, positive definite, and the expression (111.32) is valid whether 'A' is zero or not.

When hardening is considered, parameter k must be considered, on which the shifts of the yield surface depend. With a work hardening hypothesis,

$$dk = \{\sigma\}^T \{d\epsilon_{ij}\}_p . \qquad \text{III.33}$$

Substituting the flow rule (III.15), in (III.33),

$$dk = d\lambda \{\sigma\}^T \{a\} = dh [a\}^T \{\sigma\} . \qquad \text{III.34}$$

If we define effective plastic strain increment as

$$\{d\epsilon_p\}_e = \frac{\sqrt{2}}{3} \left\{\epsilon'_{ij(p)}\epsilon'_{ij(p)}\right\}^{\frac{1}{2}} \qquad \text{III.35}$$

$$= \frac{\sqrt{2}}{3}\left[\left((d\epsilon_{xx})_p-(d\epsilon_{yy})_p\right)^2+\left((d\epsilon_{yy})_p-(d\epsilon_{zz})_p\right)^2+\left((d\epsilon_{zz})_p-(d\epsilon_{xx})_p\right)^2+6\left((d\gamma_{xy})_p^2+(d\gamma_{yz})_p^2+(d\gamma_{zx})_p^2\right)\right]^{\frac{1}{2}}, \qquad \text{III.36}$$

then together with the relation

$$\frac{d\sigma_e}{d\epsilon_p} = H' \quad , \qquad \text{III.37}$$

we can experimentally determine the hardening parameter from a simple uniaxial yield test, as the slope of effective stress / plastic strain curve. Since uniaxial tensile yield, σ_Y is equal to $\sqrt{3}k$,

$$A = -\frac{1}{d\lambda}\frac{\partial F}{\partial k}dk = \frac{1}{d\lambda}\frac{d\sigma_y}{dk}dk \quad , \qquad \text{III.38}$$

from which, using (III.33) and (III.34), and after some manipulation can be shown to give:

$$d\lambda = (d\epsilon_p)_e, \qquad \text{III.39}$$

and

$$A = H' \quad . \qquad \text{III.40}$$

CHAPTER IV

MATRIX FORMULATION AND BASIC EXPRESSIONS FOR TWO DIMENSIONAL PROBLEMS

It has been shown[1] that for isotropic materials, it is convenient to express the yield criteria as functions of three stress invariants (σ_m, σ_e, ϕ) given by:

$$\sigma_m = \frac{J_1}{3} = \frac{1}{3}\left(\sigma_{xx}+\sigma_{yy}+\sigma_{zz}\right)$$

$$\sigma_e = J_2^{\frac{1}{2}} = \left[\frac{1}{2}\left(S_x^2+S_y^2+S_z^2\right)+\tau_{xy}^2+\tau_{yz}^2+\tau_{zx}^2\right]^{\frac{1}{2}}$$

$$\phi = \frac{1}{3}\sin^{-1}\left[\frac{-3\sqrt{3}\,J_3}{2\,\sigma_e^3}\right], -\frac{1}{6}\pi \le \phi \le \frac{1}{6}\pi \,, \qquad \text{IV.1}$$

where,

$$J_3 = \left[S_x S_y S_z + \left(2\tau_{xy}\tau_{yz}\tau_{zx}\right) - S_x\tau_{yz}^2 - S_y \;\; {}_{zx} \quad {}_{zxy}\right.$$

and,

$$S_x = \sigma_{xx}-\sigma_m \;;\; S_y = \sigma_{yy}-\sigma_m \;;\; S_z = \sigma_{zz}-\sigma_m .$$

The yield surfaces for several classical yield conditions can now be written as:

1. Tresca

$$F = 2\,\sigma_e \cos\phi - \sigma_y = 0 \,, \qquad \text{IV.2}$$

---

[1] G.C.Nayak and O.C.Zienkiewicz, "Convenient Form of Stress Invariants for Plasticity." Proceedings of American Society of Civil Engineers, 98, ST4, 949-954 (1972)

2. Von Mises

$$F = \sqrt{3}\,\sigma_e - \sigma_Y = 0 \,. \qquad \text{IV.3}$$

In (IV.2) and (IV.3), σ_Y, the current yield stress depends on "k". These forms yield to a very convenient definition of {a} used in the determination of $\{D_{ep}\}$, and we can write,

$$a^T = \frac{\partial F}{\partial \sigma_{ij}} = \left[\frac{\partial F}{\partial \sigma_m} \frac{\partial \sigma_m}{\partial \sigma_{ij}} + \frac{\partial F}{\partial \sigma_e} \frac{\partial \sigma_e}{\partial \sigma_{ij}} + \frac{\partial F}{\partial \phi} \frac{\partial \phi}{\partial \sigma_{ij}} \right] . \qquad \text{IV.4}$$

From (IV.1),

$$\frac{\partial \phi}{\partial \sigma_{ij}} = - \frac{\sqrt{3}}{2} \frac{1}{\cos 3\phi} \left[\frac{1}{\sigma_e^3} \frac{\partial J_3}{\partial \sigma_{ij}} - 3 \frac{J_3}{\sigma_e^4} \frac{\partial \sigma_e}{\partial \sigma_{ij}} \right] . \qquad \text{IV.5}$$

We can now write,

$$\{a\} = c1 \{a_1\} + c_2 \{a_2\} + c_3 \{a_3\} \,, \qquad \text{IV.6}$$

where the vectors $\{a_1\}$, $\{a_2\}$, $\{a_3\}$ and constants c_1, c_2, c_3 are given in Tables IV.1 and IV.2, respectively.

For the two dimensional plane stress / plane strain and axisymmetric cases we shall employ 4x4 {D} matrix and the vector {a} will be modified to,

$$\{a\}^T = \left[\frac{\partial F}{\partial \sigma_{xx}} \; \frac{\partial F}{\partial \sigma_{YY}} \; \overline{\partial \tau_{xy}} \; \frac{\partial F}{\partial \sigma_{zz}} \right] , \qquad \text{IV.7}$$

with the $\{d_D\}$ matrix used in $\{D_{ep}\}$ as given in Table IV.3.

Whenever {a] vector is not uniquely defined for certain stress combinations, singularities arise. In the program a check is made to find out if ϕ is equal to 30°, and then constants c_1, c_2, c_3 are rewritten explicitly, when ϕ approaches 30°. That is, the corners are rounded off.

TABLE IV.1

DERIVATIVES OF STRESS INVARIANTS

Vector	Definition	Values
$\{a_1\}^T$	$\left(\frac{\partial \sigma_m}{\partial \sigma_{ij}}\right)$	$\frac{1}{3}\{1,1,1,0,0,0\}$
$\{a_2\}^T$	$\left(\frac{\partial \sigma_e}{\partial \sigma_{ij}}\right)$	$\frac{1}{2\sigma_e}\{S_x, S_y, S_z, 2\tau_{yz}, 2\tau_{zx}, 2\tau_{xy}\}$
$\{a_3\}$	$\left(\frac{\partial J_3}{\partial \sigma_{ij}}\right)$	$\begin{bmatrix} S_y S_z - \tau_{yz}^2 + \frac{1}{3}\sigma_e^2 \\ S_x S_z - \tau_{zx}^2 + \frac{1}{3}\sigma_e^2 \\ S_x S_y - \tau_{xy}^2 + \frac{1}{3}\sigma_e^2 \\ 2(\tau_{zx}\tau_{xy} - S_x\tau_{yz}) \\ 2(\tau_{xy}\tau_{yz} - S_y\tau_{zx}) \\ 2(\tau_{yz}\tau_{xy} - S_z\tau_{xy}) \end{bmatrix}$

TABLE IV.2

CONSTANTS c FOR VARIOUS YIELD CONDITIONS

Yield condition	c_1	c_2	c_3
Tresca	0	$(2\cos\phi\,(1+\tan\phi\tan 3\phi))$	$\left(\frac{\sqrt{3}\sin\phi}{\sigma_e^2\cos 3\phi}\right)$
Von Mises	0	$\sqrt{3}$	0

TABLE IV.3

ELASTICITY MATRICES

Vector	Plane Stress	Plane Strain / Axisymmetry
$\{D\}$	$\frac{E}{(1-\nu^2)}\begin{bmatrix} 1 & \nu & 0 & 0 \\ \nu & 1 & 0 & 0 \\ 0 & 0 & \frac{(1-\nu)}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\frac{E(1-\nu)}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1 & \frac{\nu}{(1-\nu)} & 0 & \frac{\nu}{(1-\nu)} \\ \frac{\nu}{(1-\nu)} & 1 & 0 & \frac{\nu}{(1-\nu)} \\ 0 & 0 & \frac{(1-2\nu)}{2(1-\nu)} & 0 \\ \frac{\nu}{(1-\nu)} & \frac{\nu}{(1-\nu)} & 0 & 1 \end{bmatrix}$
$\{d_D\}$	$\begin{bmatrix} \frac{E}{(1+\nu)} a_1 + E\nu\frac{(a_1+a_2)}{(1-\nu^2)} \\ \frac{E}{(1+\nu)} a_2 + E\nu\frac{(a_1+a_2)}{(1-\nu^2)} \\ Ga_3 \\ \frac{E}{(1+\nu)} a_4 \quad E\nu\frac{(a_1+a_2)}{(1-\nu^2)} \end{bmatrix}$	$\begin{bmatrix} \frac{E}{(1+\nu)} a_1 + E\nu\frac{(a_1+a_2+a_4)}{(1+\nu)(1-2\nu)} \\ \frac{E}{(1+\nu)} a_2 + E\nu\frac{(a_1+a_2+a_4)}{(1+\nu)(1-2\nu)} \\ Ga_3 \\ \frac{E}{(1+\nu)} a_4 + E\nu\frac{(a_1+a_2+a_4)}{(1+\nu)(1-2\nu)} \end{bmatrix}$

CHAPTER V

FINITE ELEMENT EXPRESSIONS AND PROGRAM STRUCTURE

The expressions derived in the previous Chapters describe fully the stress / strain relation in the elasto-plastic state. Since the elasto-plastic matrix $\{D_{ep}\}$ depends on the state of total stress, the piecewise linearization process is adopted, wherein small increments of load are prescribed and in each increment the material is treated as quasi elastic, with a constant elasto-plastic matrix. The material tangent stiffness matrix $\{K_T\}$, during the increment, is evaluated using

$$\{K_T\} = \int \{B\}^T \{D_{ep}\} \{B\} dV . \qquad V.1$$

The solution process for the nonlinear problem described in Chapter II is utilized. The program is outlined in Fig. V.1.

There are twenty five subroutines for linear strain hardening application; fifteen of these are common to non-linear strain hardening application. The nonlinear strain hardening program utilizes a total of twenty eight subroutines. The controlling segment calling the subroutines is called MASTER for linear strain hardening program , and MSTER2 for nonlinear strain hardening program. A brief description of each subroutine is presented as follows:

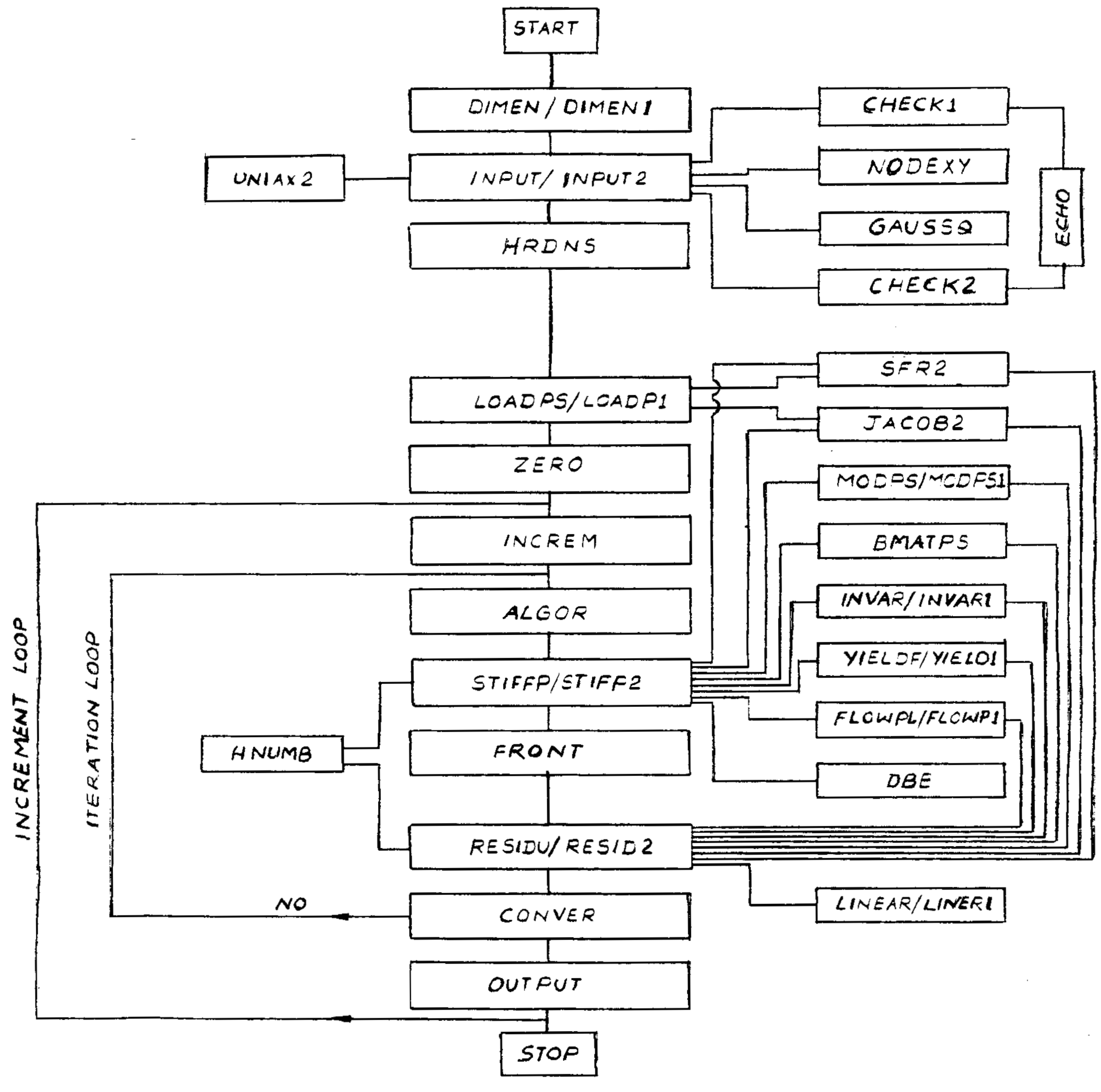


Fig. V.1 Program Outline for Linear and Nonlinear Two Dimensional Elasto-plastic Problems.

Subroutines DIMEN / DIMEN1

These subroutines preset variables to upgrade the magnitude of the maximun size problems. DIMEN is for linear strain hardening program, and DIMEN1 is for nonlinear strain hardening program.

Subroutines INPUT / INPUT2

These subroutines accept most of the input data. INPUT is for linear strain hardening program, and INPUT2 is for nonlinear program.

Subroutine CHECK1

This subroutine checks main control data and is common for both programs.

Subroutine CHECK2

This subroutine checks remainder of input data and is common for both programs.

Subroutine NODEXY

This subroutine generates the midside nodes of straight sides of elements and central node of 9-noded elements. This subroutine is common for linear and nonlinear programs.

Subroutine ECHO

This subroutine writes the remaining data cards after an error has been detected in input data cards, and is common for both programs.

Subroutine GAUSSQ

This subroutine is common for both programs and sets up Gaussian quadrature data for numerical integration.

Subroutine UNIAX2

This subroutine reads uniaxial test data for stress and total strains for nonlinear program.

Subroutine HRDNS

This subroutine prepares the hardness / plastic strain array from the uniaxial test data for nonlinear program.

Subroutines LOADPS / LOADP1

These subroutines evaluate the consistent nodal forces for each element for plane and axisymmetric situat- ions. LOADPS is used in linear strain hardening program and LOADP1 in nonlinear program.

Subroutine SFR2

This subroutine evaluates the shape functions and their derivatives at the Gaussian points and is common for both programs.

Subroutine JACOB2

This subroutine evaluates the Jacobian matrix and the cartesian shape function derivatives for both programs.

Subroutines MODPS / MODPS1

These subroutines evaluate the {D} matrix for plane and axisymmetric situations. MODPS is used in linear program and MODPS1 in nonlinear program.

Subroutines INVAR / INVARl

These subroutines calculate the stress invariants and the current value of yield function. INVAR is for linear program and INVARl is used in nonlinear program.

Subroutines YIELDF / YIELD1

These subroutines calculate {a} vector. YIELDF is for linear program and YIELD1 is for nonlinear program.

Subroutines FLOWPL / FLOWP1

These subroutines calculate $\{d_D\}$ vector. FLOWPL is used in linear program and FLOWP1 is for nonlinear program.

Subroutines STIFFP / STIFP2

These subroutines calculate element stiffness matrix {K}. STIFFP is used in linear strain hardening program and STIFP2 in nonlinear strain hardening program.

Subroutine ZERO

This subroutine is used in both linear and nonlinear programs. This subroutine initializes several arrays which are employed for accumulation of data.

Subroutine DBE

This subroutine is used in both linear and nonlinear programs. This subroutine is used for matrix multiplication of {D} matrix by {B} matrix.

Subroutine INCREM

This subroutine is used in both linear and nonlinear programs. This subroutine increases the applied loading.

Subroutine ALGOR

This subroutine sets an equation resolution index, which determines initial stiffness approach or tangential stiffness approach or a combination of both for reformulation of element stiffness matrix. This is a common subroutine for both linear and nonlinear programs.

Subroutine FRONT

This subroutine is used in both linear and nonlinear programs. This subroutine is the equation solver by the frontal method.

Subroutines LINEAR / LINER1

These subroutines evaluate stresses and strains assuming linear elastic behaviour. LINEAR is used in linear strain hardening program and LINER1 in nonlinear program.

Subroutines RESIDU / RESID2

These subroutines reduce the stresses to the yield surface and evaluate the equivalent nodal forces. Comparison of these nodal forces with the applied loads give the residual forces. The procedure consists of the following steps:

1. Applied loads for the r^{th} iteration are residual forces
2. For the residual forces compute,

   $$d\sigma_e^r = \{D\}\, d\epsilon^r .$$

3. Compute,

   $$\sigma^r = \sigma^{(r-1)} + d\sigma_e^r .$$

   Check if $f(\sigma^r) < 0$, with $\{K\}$ referring to the initial value at the start of increment. If it is satisfied only elastic strain changes occur and the process is stopped.
4. If $f(\sigma^r) \geq 0$ and also $f(\sigma^{(r-1)}) = 0$, the element was in

yield at the start of increment and continues to yield. Evaluate,

$$d\sigma^r = \{D\}\, d\epsilon^r - d\lambda\, \{d_D\} , \qquad \text{V.2}$$

$$\sigma^r = \sigma^{(r-1)} + d\sigma^r ,$$

$$= \sigma^{(r-1)} + d\sigma_e^r - d\lambda\, \{d_D\} . \qquad \text{V.3}$$

5. If $f(\sigma^r) > 0$, but $f(\sigma^{(r-1)}) < 0$ find the intermediate stress value at which yield begins and compute σ^r from that point,

6. When a finite sized increment is taken the resulting σ^r may deviate from the yield surface and a scaling factor may be used to bring the point to the yield surface. Therefore,

$$\sigma^r = \sigma^r \left[\frac{\sigma_y + H' \epsilon_p^r}{\sigma^r} \right] . \qquad \text{V.4}$$

Or the excess stress from the yield surface at each increment may be divided into a

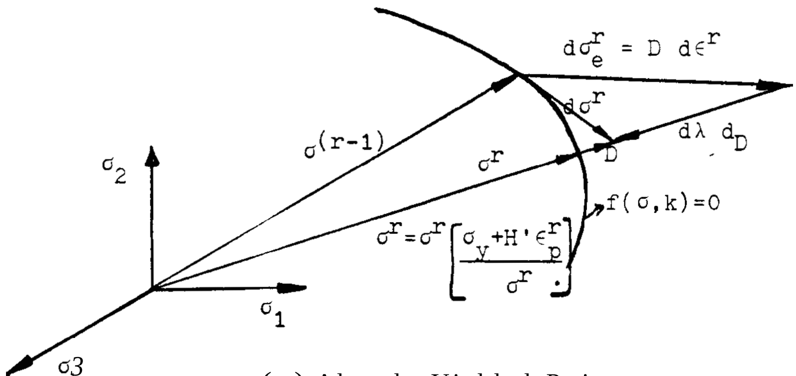
$$\text{number of parts} \leq \left[\frac{\sigma_e^r - \sigma_y}{\sigma_y} \right] 8 + 1 \quad . \qquad \text{V.5}$$

The above process is shown in Fig. V.2 .

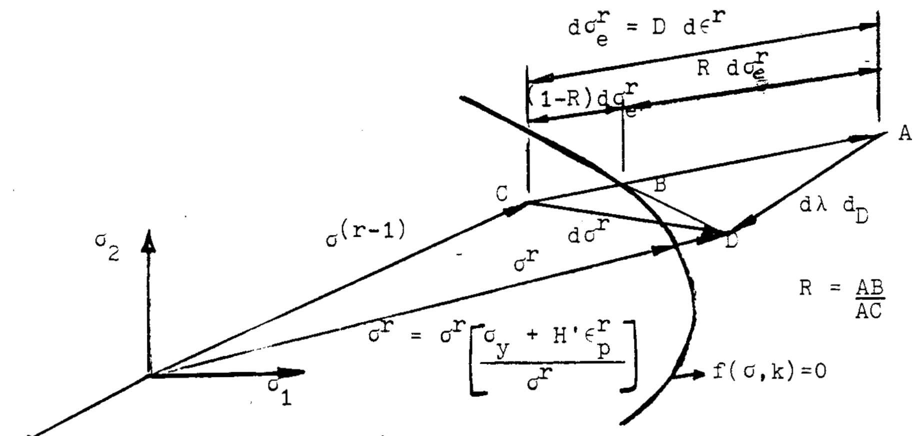
RESIDU is used in linear program and RESID2 in nonlinear program.

Subroutine CONVER

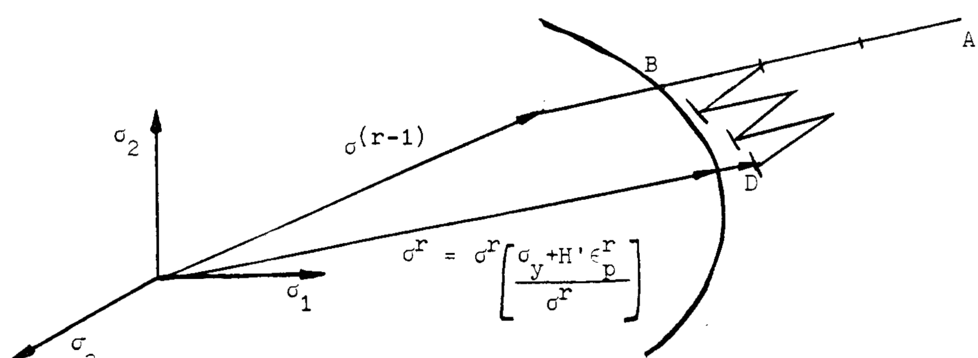
This subroutine is used in both linear and nonlinear programs. This subroutine checks the convergence



(a) Already Yielded Point



(b) Point Yielding During This Increment



(c) Refined Process for Reducing the Stress to Yield Surface

Fig. V.2 Process of Reducing The Incremental Stress to Yield Surface.

of iteration process.

Subroutine HNUMB

This subroutine is used in nonlinear strain hardening program. This subroutine returns the hardness value depending on the effective plastic strain.

Subroutine OUTPUT

This subroutine is used in both linear program and in nonlinear program. This subroutine outputs displacements, reactions and stresses.

CHAPTER VI

TEST PROBLEMS WITH INPUT AND OUTPUT

In this Chapter an attempt is made to establish the accuracy of the Finite Element Programs presented in previous Chapters for the linear and nonlinear strain hardening applications. Typical elasto-plastic test problems are chosen for which either closed form solutions or practical iterative solutions exist for comparison with program output.

The first problem considered is that of elasto-plastic stress and strain concentration factors at a circular hole in a uniformly stressed (radially, that is) infinite plate. For this plane stress problem, the problem parameters and the finite element model are illustrated in Fig. VI.1. The problem is analysed using the linear strain hardening program. The applied stress σ is increased from $0.5\sigma_Y^0$ to $0.9\sigma_Y^0$ in seven (7) steps. The spread of plastic zone at different load levels is shown in Fig. VI.2. The various stress and strain concentration factors are defined as follows:

$$k_{\sigma_\theta} = \frac{\sigma_\theta \text{ @ Hole}}{\sigma} ,$$

$$k_{\sigma_e} = \frac{\sigma_e \text{ @ Hole}}{\sigma} ,$$

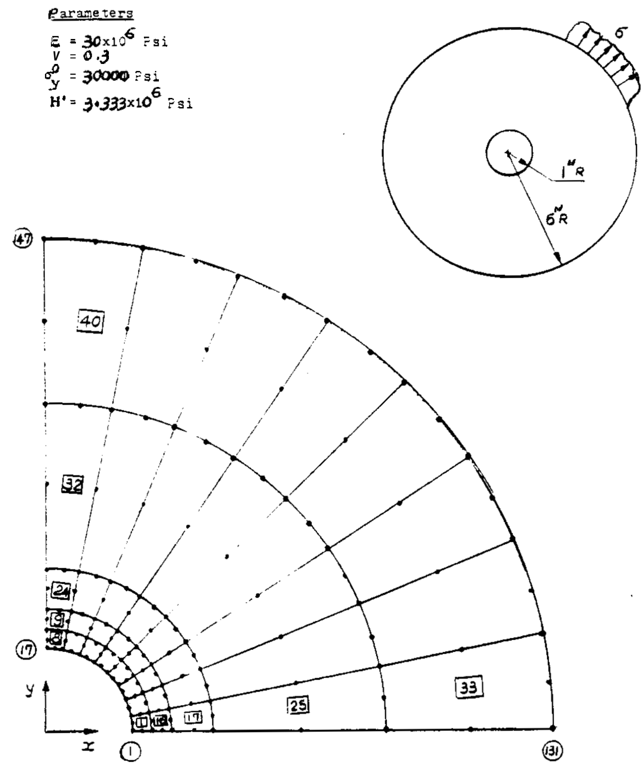


Fig. VI.1 Plane Stress Elasto-plastic Problem of A Circular Hole in A Uniformly Stressed Infinite Plate.

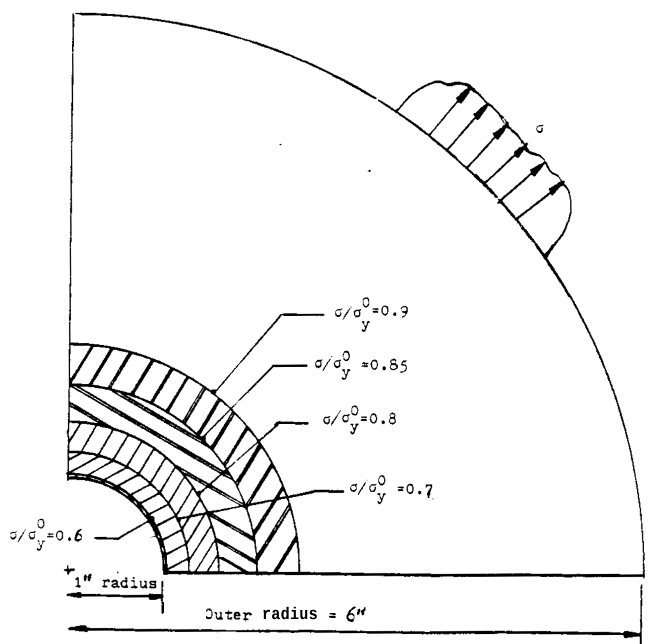


Fig. VI.2 Spread of Plastic Zones at Different Load Levels for Plane Stress Elasto-plastic Problem of A Circular Hole in A Uniformly Stressed Infinite Plate.

$$k_{\epsilon_e} = \frac{\epsilon_{e\ @\ Hole}}{\epsilon_{e\ @\ \infty}} \text{, and}$$

$$\epsilon_{e\ @\ \infty} = \frac{\sigma}{E}\frac{2}{3}(1+\nu) \ . \qquad \text{VI.1}$$

The output from the linear strain hardening program is summarized in Table VI.1 for the points around the circular hole. The stress / strain concentration factors are compared with the values from reference[2] in Table VI.2. The stress / strain concentration factors from the program compare very well with the values from reference[2]. Before the onset of plasticity, the program yields a stress concentration factor k of 1.96, which compares very well with the theoretical value. of 2.0. The stress concentration factor k_{σ_e} is plotted at different levels of loading in Fig. VI.3. These results show that the strain concentration increases as the material in the most highly stressed region becomes plastic. This happens as the stress concentration decreases.

The second selected problem to show the agreement of the finite element solutions with theoretical predictions is the classic plane strain problem of a thick walled cylinder subjected to internal pressure. The problem is studied under three different levels of final loading:

1. The entire cylinder is perfectly plastic.

---

[2] I.S.Tuba, "Elastic-Plastic Stress and Strain Concentration Factors at a Circular Hole in a Uniformly Stressed Infinite Plate," Journal of Applied Mechanics, Trans. ASME,710/September 1965.

TABLE VI.1

PROGRAM OUTPUT OF STRESSES, STRAINS AND STRESS / STRAIN CONCENTRATION FACTORS FOR THE PLANE STRESS ELASTO-PLASTIC PROBLEM OF A CIRCULAR HOLE IN A UNIFORMLY STRESSED INFINITE PLATE.

Load	Stresses		Strains			Concentration Factors		
$\frac{\sigma}{\sigma_y^0}$	σ_e @ Hole (ksi)	σ_θ @ Hole (ksi)	Effective Plastic Strain (E.P.S.)	Elastic Strain $\left(\frac{\sigma_e}{E}\right)$	Total Strain (E.P.S.+Elastic) ϵ_e	k_{σ_e}	k_{σ_θ}	k_{ϵ_e}
0.25	14.317	14.678	0.0	0.47723×10^{-3}	0.47723×10^{-3}	1.909	1.957	2.203
0.50	28.634	29.356	0.0	0.95447×10^{-3}	0.95447×10^{-3}	1.909	1.957	2.203
0.60	30.383	31.965	0.114932×10^{-3}	1.01277×10^{-3}	1.12770×10^{-3}	1.688	1.776	2.169
0.70	31.049	33.012	0.314915×10^{-3}	1.03497×10^{-3}	1.34988×10^{-3}	1.479	1.572	2.225
0.80	31.893	34.658	0.567985×10^{-3}	1.0631×10^{-3}	1.63109×10^{-3}	1.329	1.444	2.353
0.90	33.003	36.612	0.90108×10^{-3}	1.1001×10^{-3}	2.00118×10^{-3}	1.222	1.356	2.566

TABLE VI.2

COMPARISON OF STRESS / STRAIN CONCENTRATION FACTORS FOR THE PLANE STRESS ELASTO-PLASTIC PROBLEM OF A CIRCULAR HOLE IN A UNIFORMLY STRESSED INFINITE PLATE.

$\frac{\sigma}{\sigma_y^0}$	k_{σ_e}		k_{σ_θ}		k_{ϵ_e}	
	Theory[a]	Program	Theory[a]	Program	Theory[a]	Program
0.25	2.0	1.91	2.0	1.96	2.0	2.2
0.5	2.0	1.91	2.0	1.96	2.0	2.2
0.6	1.85	1.69	1.85	1.78	2.1	2.17
0.7	1.55	1.48	1.55	1.57	2.23	2.23
0.8	1.36	1.33	1.36	1.44	2.42	2.35
0.9	1.23	1.22	1.23	1.36	2.7	2.57

[a] I.S.Tuba, "Elastic-Plastic Stress and Strain Concentration Factors at a Circular Hole in a Uniformly Stressed Infinite Plate," Journal of Applied Mechanics, Trans. ASME, 710/September 1965.

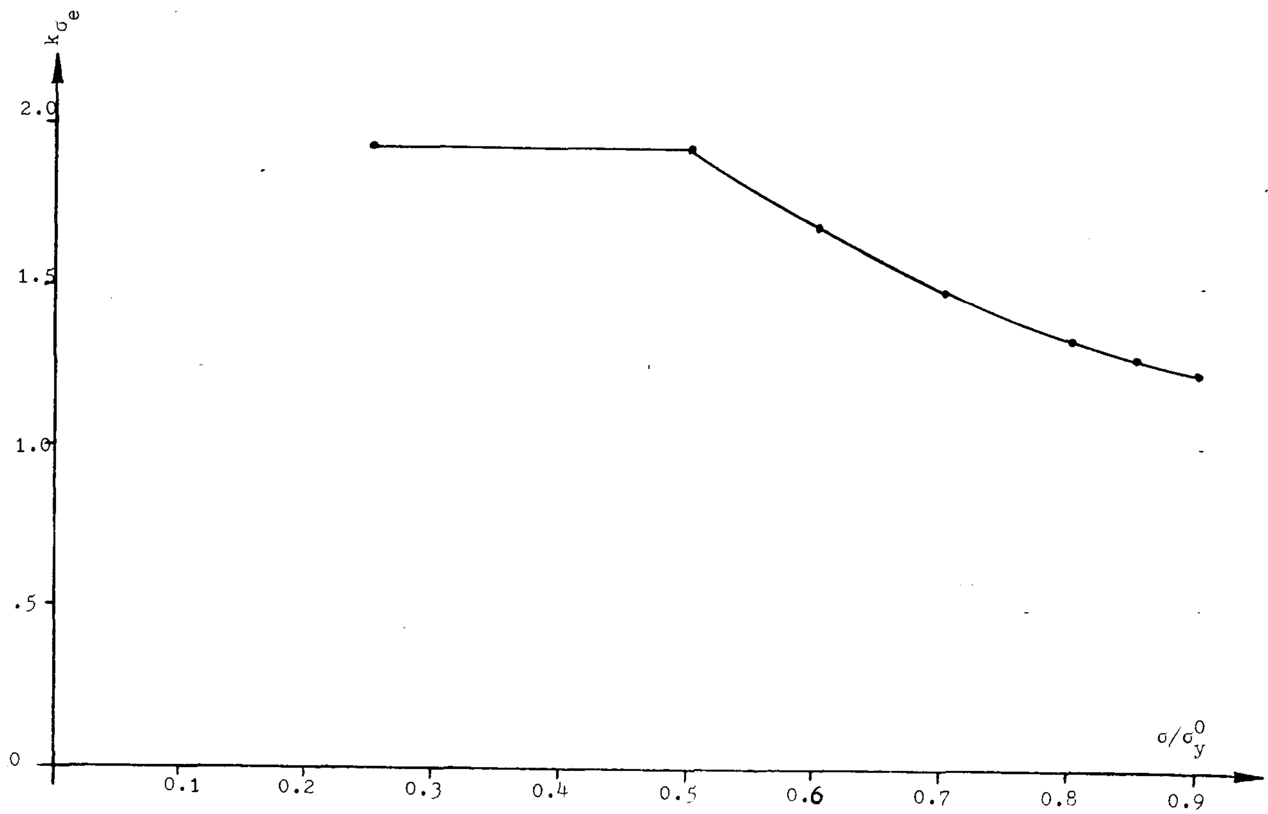


Fig. VI.3 Stress Concentration Factor k_{σ_e} at Different Levels of Applied Loading for Elasto-plastic Problem of a Circular Hole in a Uniformly Stressed Infinite Plate.

2. The cylinder is partly plastic, and made up of a perfectly plastic material.
3. The cylinder is partly plastic, and made up of a linear strain hardening material.

The parameters of the problem and the finite element modeling of the problem are illustrated in Fig. VI.4.

The closed form solutions from theory for the three different levels of final loading are as below:

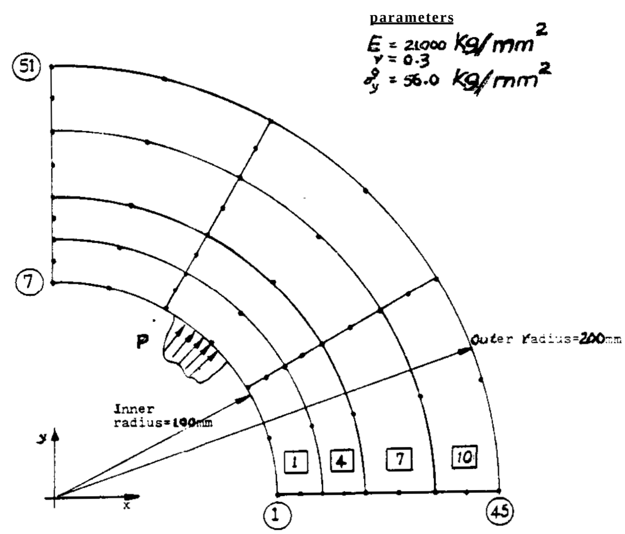


Fig. VI.4 Thick Walled Cylinder Subjected to Internal Pressure.

1. Perfectly Elastic Cylinder.[3,4]

$$P < P_c = \left(1-\frac{a^2}{b^2}\right)\frac{\sigma_y^0}{\sqrt{3}} \quad . \qquad \text{VI.2}$$

$$\sigma_{rr} = P\,\frac{a^2}{(b^2-a^2)}\left(1-\frac{b^2}{r^2}\right) . \qquad \text{VI.3}$$

$$\sigma_{\theta\theta} = P\,\frac{a^2}{(b^2-a^2)}\left(1+\frac{b^2}{r^2}\right) . \qquad \text{VI.4}$$

$$u_d = \frac{3}{2}\frac{P}{E}\frac{b^2\,a^2}{(b^2-a^2)}\quad\frac{1}{r} \quad . \qquad \text{VI.5}$$

---

[3]A. Nadai, Plasticity (New York and London: McGraw-Hill Book Company, Inc., 1931), p.194.

[4]Aris Phillips, Introduction to Plasticity (New York: The Ronald Press Company, 1956), p.183.

2. Partly Plastic Cylinder and made up of a Perfectly Plastic Material.[5,6]

$$\beta = \frac{r}{b} \quad ; \quad \alpha = \frac{a}{b} \quad ; \quad \mu = \frac{c}{b} \quad . \qquad \text{VI.6}$$

$$\sigma_{rr} = \frac{-1}{\sqrt{3}} \sigma_y^0 \mu^2 \left(\frac{1}{\beta^2} - 1 \right) \quad . \quad c < r < b \text{ (Elastic Region)}$$

$$\sigma_{\theta\theta} = \frac{1}{\sqrt{3}} \sigma_y^0 \mu^2 \left(\frac{1}{\beta^2} + 1 \right) \quad . \qquad \text{VI.7}$$

$$\sigma_{rr} = -P + \frac{2\sigma_y^0}{\sqrt{3}} \ln\left(\frac{\beta}{\alpha}\right) \quad . \quad a < r < c \text{ (Plastic Region)}$$

$$\sigma_{\theta\theta} = -P + \frac{2\sigma_y^0}{\sqrt{3}} \ln\left[\frac{\beta}{\alpha} + 1\right] \qquad \text{VI.8}$$

$$P = \frac{\sigma_y^0}{\sqrt{3}} \left(2 \ln\left(\frac{\mu}{\alpha}\right) + 1 - \mu^2 \right) . \qquad \text{VI.9}$$

$$u_d = \frac{\sqrt{3}}{2} \frac{\sigma_y^0 c^2}{Er} \quad . \qquad \text{VI.10}$$

3. Partly Plastic Cylinder and made up of a Linear Strain Hardening Material.[7]

$$\sigma_{rr} = \frac{\sigma_y}{\sqrt{3}} \left(\frac{1}{b^2} - \frac{1}{r^2} \right) \quad .$$

$$\sigma_{\theta\theta} = \frac{\sigma_y}{\sqrt{3}} \left(\frac{1}{b^2} + \frac{1}{r^2} \right) \quad . \quad c \leq r \leq b \text{ (Elastic Region)} \qquad \text{VI.11}$$

---

[5]Nadai, Plasticity, pp.196-97.

[6]Phillips, Plasticity, pp.184-85.

[7]Phillips, plasticity, pp.186-90.

$$\sigma_{rr} = \frac{-2\sigma_y}{\sqrt{3}}\left[\frac{-c^2}{2b^2} + \frac{1}{2} + \frac{1}{(9H'+4E)}\left\{4E\ \ln\left(\frac{c}{r}\right) + \frac{9H'}{2}\left(\frac{c^2}{r^2} + \frac{c^2}{a^2} - 1\right)\right\}\right].$$

$$\sigma_{\theta\theta} = \frac{-2\sigma_y}{\sqrt{3}}\left[\frac{-c^2}{2b^2} + \frac{1}{2} + \frac{1}{(9H'+4E)}\left\{4E\left(\ln\left(\frac{c}{r}\right) - 1\right) - 9H'\left(\frac{c^2}{r^2} - \frac{c^2}{a^2} + 1\right)\right\}\right].$$

$$a \leq r \leq c \qquad \text{VI.12}$$

(Plastic Region)

$$u_d = \frac{\sqrt{3}}{2}\frac{\sigma_y c^2}{Er} . \qquad \text{VI.13}$$

$$P = \frac{2}{\sqrt{3}}\sigma_y\left[\frac{1}{2} - \frac{c^2}{2b^2} + \frac{1}{(9H'+4E)}\left\{4E\ \ln\left(\frac{c}{a}\right) + \frac{9H'}{2}\left(\frac{c^2}{a^2} - 1\right)\right\}\right]. \qquad \text{VI.14}$$

In the equations (VI.2) thru (VI.14) we have,

a = Inner radius, b = Outer radius,

c = Yield radius, H'= Hardening parameter,

r = Radius, u_d= Radial displacement,

E = Young's modulus, σ_y= Yield stress,

σ_{rr}=Radial stress, $\sigma_{\theta\theta}$=Tangential stress,

σ_Y^0 =Uniaxial yield stress, P= Pressure,

and P_c = Critical pressure.

The output from the linear strain hardening program is compared in Table VI.3 for the elastic cylinder for stresses. Stresses show an excellent comparison with the results obtained from expressions (VI.3) and (VI.4).

For the case of a partly plastic cylinder, made up of a perfectly plastic material, the stress output from linear strain hardening program is compared in Table VI.4 . Once the cylinder is partly plastic, the boundary of the

TABLE VI.3

COMPARISON OF LINEAR PROGRAM OUTPUT WITH CLOSED FORM SOLUTION FOR ELASTIC THICK WALLED CYLINDER UNDER INTERNAL PRESSURE.

Element	Gauss Point	Estimated Radius (mm)	σ_{rr} (Compressive) Calculated Values (Kg/mm^2)	σ_{rr} (Compressive) Program Output (Kg/mm^2)	$\sigma_{\theta\theta}$ Calculated Values (Kg/mm^2)	$\sigma_{\theta\theta}$ Program Output (Kg/mm^2)
1	1	104.227	20.877	20.885	36.443	36.452
	3	115.774	15.444	15.437	31.011	31.004
4	1	124.227	12.391	12.394	27.958	27.961
	3	135.774	9.105	9.102	24.672	24.669
7	1	146.34	6.755	6.759	22.321	22.326
	3	163.66	3.84	3.836	19.407	19.403
10	1	176.34	2.229	2.231	17.795	17.797
	3	193.66	0.518	0.516	16.085	16.083

TABLE VI.4

COMPARISON OF LINEAR PROGRAM OUTPUT WITH CLOSED FORM SOLUTION FOR PARTLY PLASTIC THICK WALLED CYLINDER OF PERFECTLY PLASTIC MATERIAL UNDER INTERNAL PRESSURE.

Element	Gauss Point	Estimated Radius (mm)	σ_{rr} (Compressive)		$\sigma_{\theta\theta}$	
			Calculated Values (Kg/mm^2)	Program Output (Kg/mm^2)	Calculated Values (Kg/mm^2)	Program Output (Kg/mm^2)
1	1	104.227	39.353	39.354	25.310	25.237
	3	115.774	32.559	32.568	32.104	32.093
4	1	124.227	28.002	28.005	36.66	36.645
	3	135.774	22.255	22.268	42.408	42.269
7	1	146.34	17.409	17.372	47.254	46.976
	3	163.66	10.209	10.343	51.594	51.802
10	1	176.34	5.925	5.965	47.31	47.588
	3	193.66	1.377	1.380	42.761	43.002

plastic region, r = c, at pressure P is obtained from expression (VI.9) by trial and error method. Once again very close agreement is seen between the program output, and the calculated stress values using expressions (VI.7) and (VI.8) for a pressure P of 42.03 Kg/mm^2.

The stress output from the linear strain hardening program is compared in Table VI.5 for the case of a partly plastic cylinder of a linear strain hardening material. The boundary of the plastic region, r = c , is obtained from the expression (VI.14) by trial and error method, for pressure P acting inside the cylinder. The stresses show good agreement with the results obtained from expressions (VI.11) and (VI.12) for a pressure P of 42.03 Kg/mm^2.

In Table VI.6 a comparison is made of the radial displacement for the three different cases between the program output and the results obtained from expressions (VI.5),(VI.10) and (VI.13). Once again close agreement is obtained.

The assumption of linear strain hardening may not be adequate for certain situations, The modified program considers the nonlinear type of strain hardening material. If uniaxial test data are available at a discrete number of points, namely the stress and the corresponding total strains, the modified program represents the uniaxial stress / plastic strain relationship in a piecewise linear fashion. The program calculates the hardness array for the regions

TABLE VI.5

COMPARISON OF LINEAR PROGRAM OUTPUT WITH CLOSED FORM SOLUTION FOR PARTLY PLASTIC THICK WALLED CYLINDER OF HARDNESS(H') 366.279 KG/MM^2 UNDER INTERNAL PRESSURE .

Element	Gauss Point	Estimated Radius (mm)	σ_{rr} (Compressive)		$\sigma_{\theta\theta}$	
			Calculated Values (Kg/mm^2)	Program Output (Kg/mm^2)	Calculated Values (Kg/mm^2)	Program Output (Kg/mm^2)
1	1	104.227	42.185	39.276	25.49	27.073
	3	115.774	35.131	32.338	31.51	33.472
4	1	124.227	30.455	27.708	35.605	37.753
	3	135.774	24.612	21.914	40.822	43.076
7	1	146.34	19.723	17.014	45.262	47.533
	3	163.66	9.674	10.039	48.889	50.355
10	1	176.34	5.615	5.797	44.83	46.249
	3	193.66	1.305	1.341	40.52	41.792

TABLE VI.6

COMPARISON OF LINEAR PROGRAM OUTPUT WITH CLOSED FORM SOLUTION FOR RADIAL DISPLACEMENT FOR THICK WALLED CYLINDER UNDER INTERNAL PRESSURE.

Nodal Point	Radius (mm)	Radial Displacement[a] (mm): Elastic Case — Calculated Values	Elastic Case — Program Output	Partly Plastic & Perfect Plasticity — Calculated Values	Partly Plastic & Perfect Plasticity — Program Output	Partly Plastic & $H'=366.279$ Kg/mm^2 — Calculated Values	Partly Plastic & $H'=366.279$ Kg/mm^2 — Program Output
1	100	0.222	0.212	0.591	0.616	0.560	0.596
9	110	0.202	0.196	0.537	0.556	0.509	0.53
14	120	0.185	0.184	0.493	0.51	0.467	0.494
21	130	0.171	0.173	0.455	0.473	0.431	0.459
29	140	0.159	0.165	0.422	0.445	0.400	0.432
30	155	0.143	0.154	0.381	0.413	0.361	0.401
37	170	0.131	0.146	0.348	0.391	0.33	0.38
43	185	0.12	0.14	0.32	0.374	0.303	0.363
51	200	0.111	0.135	0.296	0.361	0.280	0.351

[a]Evaluated for a Pressure of 23.35Kg/mm^2 for the elastic case and 42.03 Kg/mm^2 for Partly Plastic cases.

between uniaxial test points as the slope of uniaxial stress / plastic strain. Instantaneous yield stress σ_Y is then calculated as:

$$\sigma_y = \sigma_y^0 + H' \, d\epsilon_p \,, \qquad \text{VI.15}$$

where

σ_Y^0 = Uniaxial yield stress,

H' = Hardness value depending on the total plastic strain at $(r-1)^{th}$ iteration,

and $d\epsilon_p$ = Increase in plastic strain at r^{th} iteration.

The accuracy of the nonlinear strain hardening program is established by comparing the results between linear and nonlinear programs, for the classical plane strain problem of a thick walled cylinder under internal pressure, for which linear program gives excellent results for stresses and displacements. For comparison purposes, two cases of uniaxial test data are prepared for the non-linear program so as to have,

1. Perfect Plasticity ($H' = 0$),
2. A Fixed Hardness Value ($H' = 366.279$ Kg / mm^2).

Both the above cases are also run on the linear program. Stress output from both programs at Gauss point-1 of element 1 are compared in Fig. VI.5 for different levels of loading for both cases. Results are in very good agreement.

Effective plastic strain output from linear and and nonlinear programs at element 1 , Gauss point-1 are

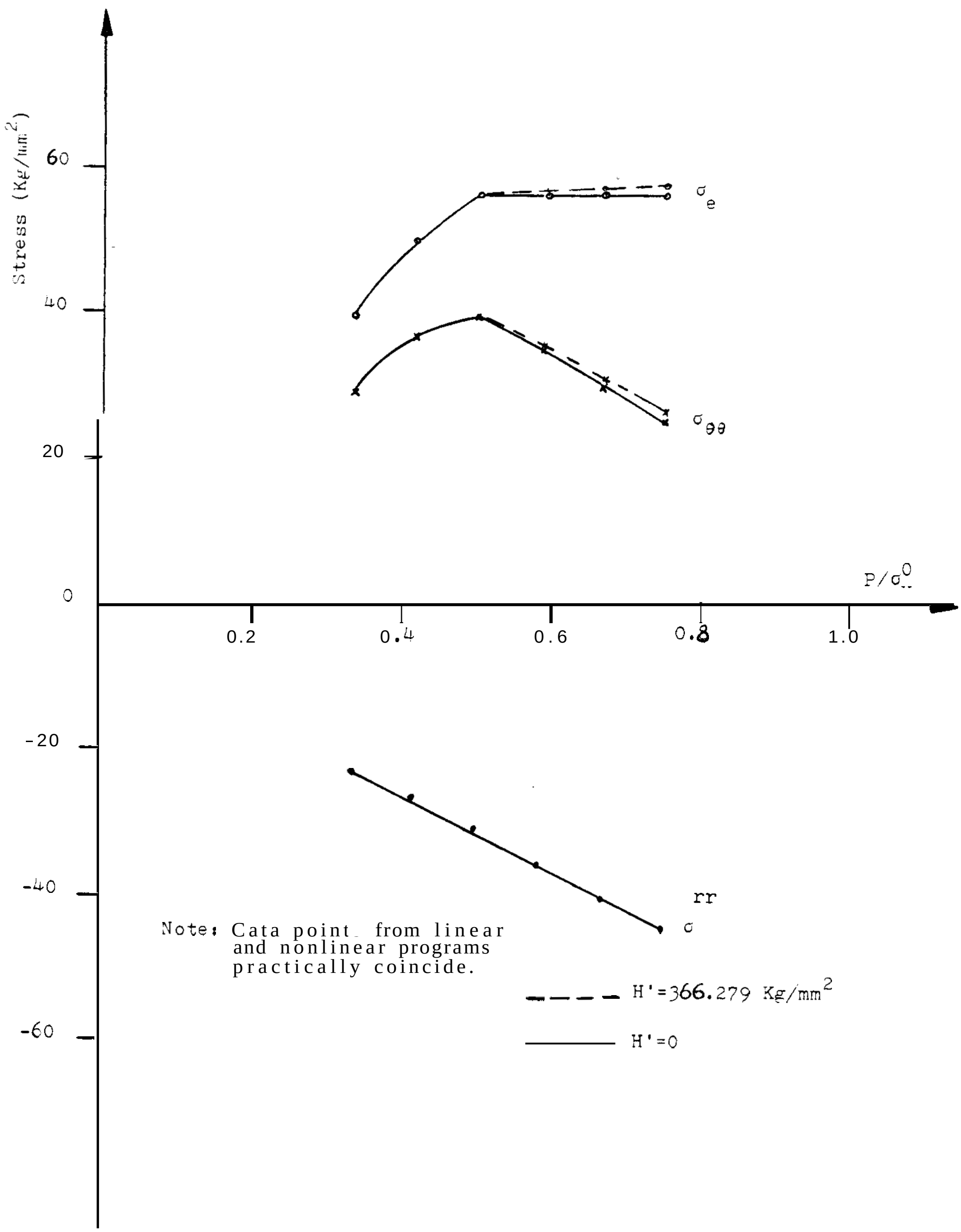


Fig. VI.5 Comparison of Linear and Nonlinear Program Outputs for Stresses for Thick Walled Cylinder Under Internal Pressure.

compared in Fig. V1.6 for defferent levels of loading for both cases. Once again results are in excellent agreement.

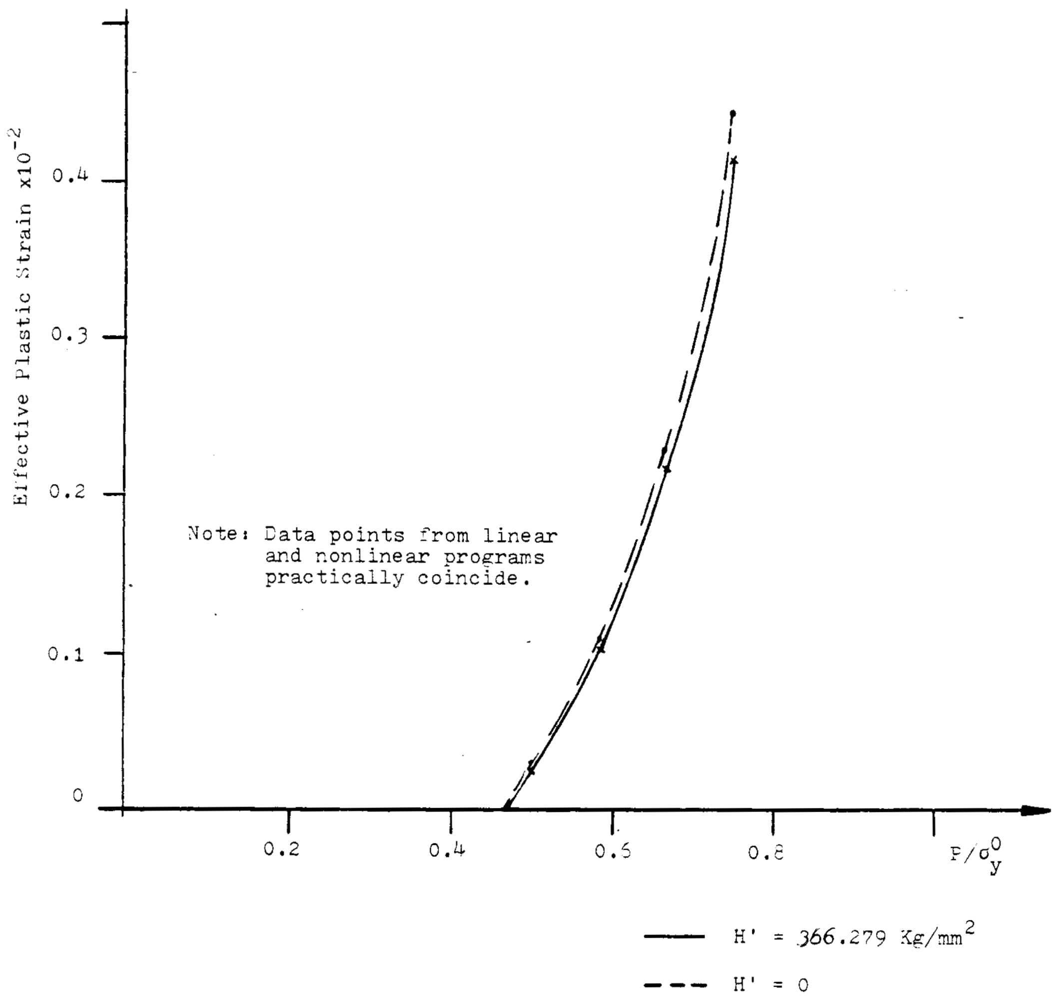


Fig. VI.6 Comparison of Linear and Non-linear Program Outputs for Effective Plastic Strains for Thick Walled Cylinder Under Internal Pressure.

In each of the above studies, a tolerance (II.7) of 1% was considered for convergence testing. From the results obtained it appears that both the linear and nonlinear programs are reliable in predicting elasto-plastic stresses, displacements, and effective plastic strain for plane stress / plane strain problems.

CHAPTER VII

SOLUTION OF SOME TYPICAL PROBLEMS AND RESULTS

In this chapter some typical elasto-plastic problems of practical interest are solved utilizing the linear and nonlinear programs.

Compact Tension (CT) Fracture Specimen

The first problem of a CT specimen is of great interest in Fracture Mechanics analysis. The, specimen is used to determine the fracture toughness of the material, which requires an accurate determination of the stress and strain field near the crack tip. The geometry of the CT fracture specimen is given in Fig. VII.1. This problem is analysed using one half of the specimen with forty five quadratic elements and one hundred and sixty five nodes. While modeling the specimen, the bolt hole across which the load is applied is ignored and the applied load is considered as a uniform pressure across the bolt hub. To accurately model the crack tip behaviour more elements are used near the crack tip. The tangent stiffness method with two point integration was employed. In view of considerable computing time involved in such an analysis, after some trial and error, a tolerance (II.7) of 6% was chosen. The problem was analysed as a plane strain problem. The load vs load-line displacement graph is plotted in Fig. VII.2 . The specimen was analysed using both linear and nonlinear programs. A hardness value of 138.484 Kgf/mm^2 was used in the linear program. In the nonlinear program the range of hardness was between 51.08 and 731.51 Kgf/mm^2. The load vs load-line displacement curve shows no appreciable difference between the linear and nonlinear cases. Total CPU time used in the analysis was about 56 seconds, about the same for both linear and nonlinear programs. The load was applied in eight steps. The spread of plastic zone is shown in Fig. VII.3

At present it is difficult to judge the accuracy of the numerical solutions obtained for the elasto-plastic crack problems because of the absence of exact solutions. Indeed the elasto-plastic analysis of cracked bodies by the finite element method must be performed with sufficient care to maintain a balance between solution accuracy and computational time. The programs can be used with confidence to determine stresses, displacements, and effective plastic strains.

Fully Plastic Thick Walled Cylinder

The second problem is an extension of the classical plane strain problem of thick walled cylinder subjected to high enough internal pressure so that the entire cylinder is plastic. The engineering applications of such thick

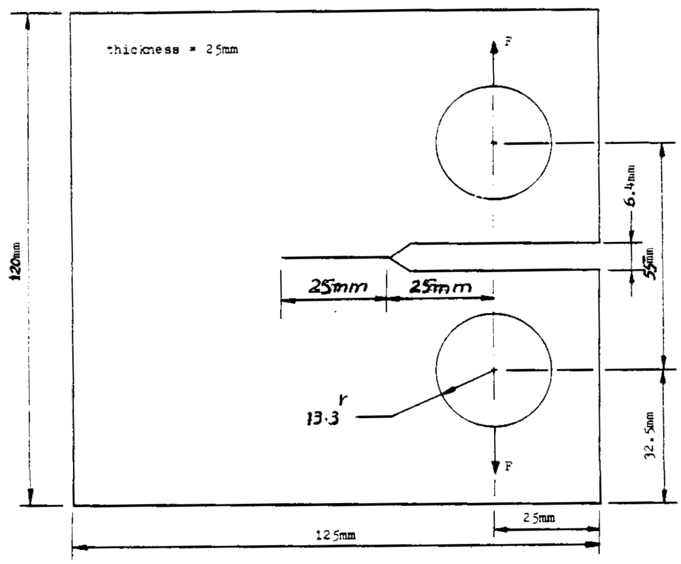


Fig. VII.1 Geometry of Compact Tension (CT) Specimen.

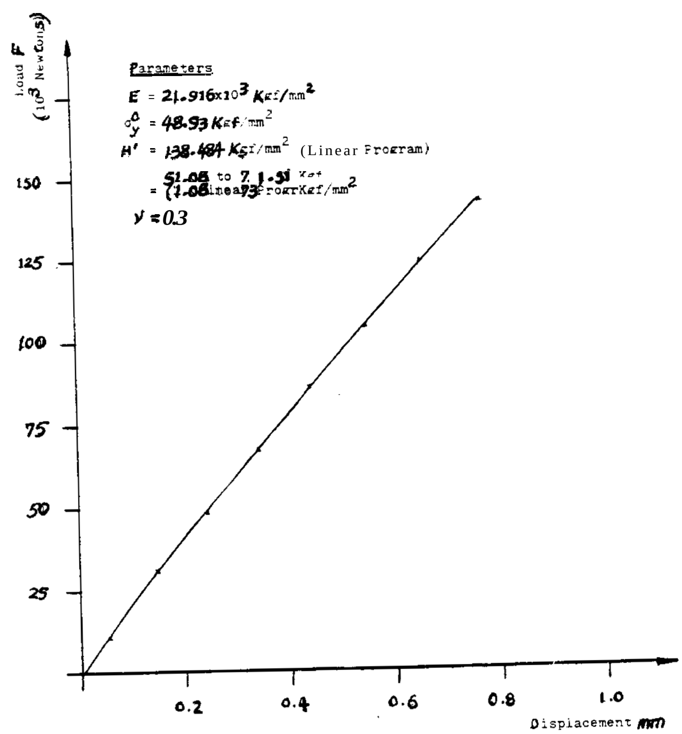


Fig. VII.2 Load Vs Load
Line Displacement Graph for CT Specimen.

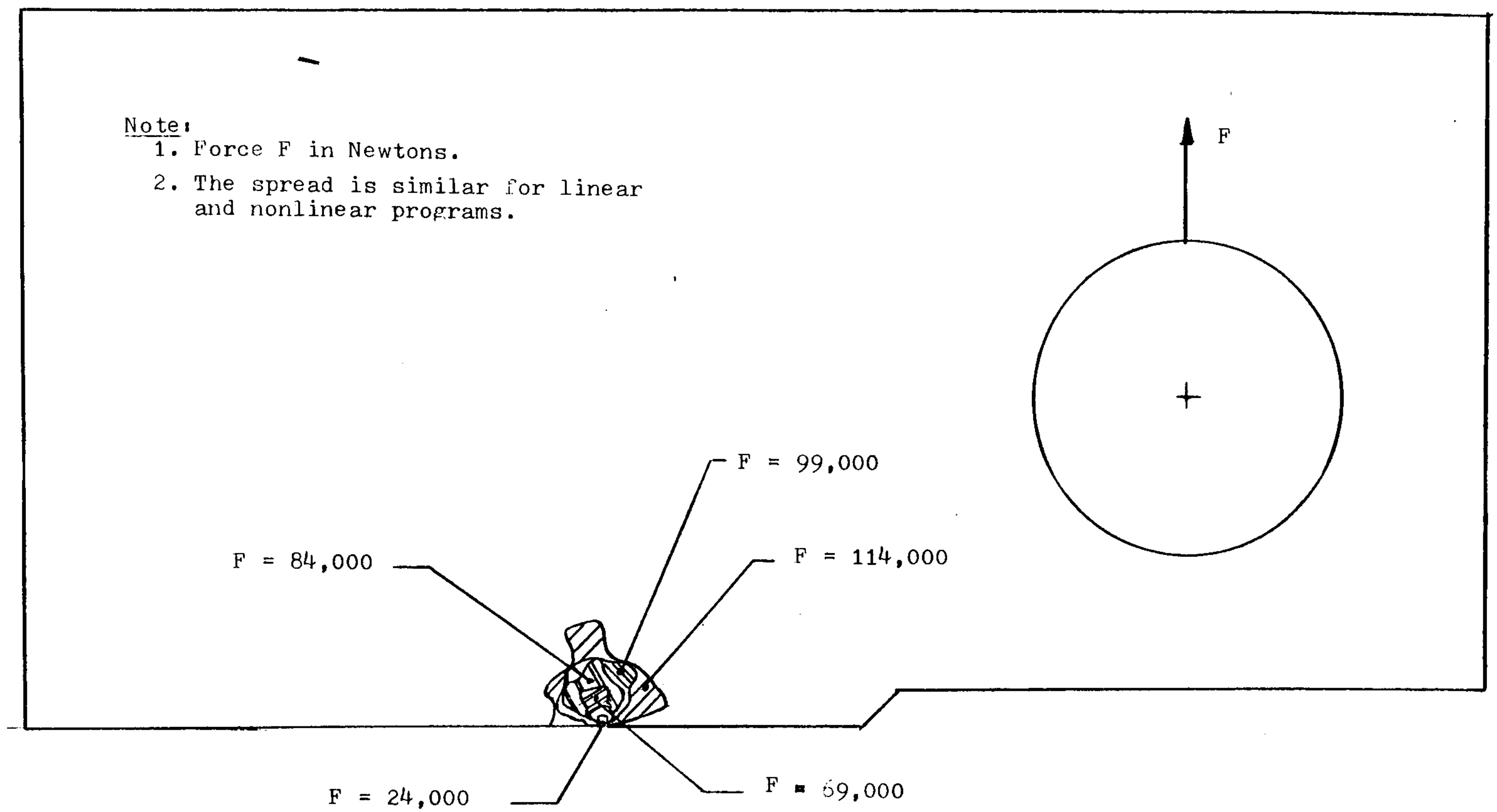


Fig. VII.3 Spread of Plastic Zones for CT Specimen.

walled pressure vessels are too many to list here. The stresses and the critical pressure to maintain the yielding are obtained from the equations[8]:

$$\sigma_{rr} = -2 \frac{\sigma_y^0}{\sqrt{3}} \ln\left(\frac{b}{r}\right) ,$$

$$\sigma_{\theta\theta} = 2 \frac{\sigma_y^0}{\sqrt{3}} \left[1-\ln\left(\frac{b}{r}\right)\right], \qquad \text{VII.1}$$

and

$$P_c = 2 \frac{\sigma_y^0}{\sqrt{3}} \ln\left(\frac{b}{a}\right) , \qquad \text{VII.2}$$

where,

a = inner radius, b = outer radius,

r = radius at the point , and σ_Y^0 = yield stress.

For the thick walled cylinder considered in Chapter VI, the expression (VII.2) yields a value of 44.82 Kg / mm^2 for P_c, for a perfectly plastic material.

The problem was analysed at a pressure of 46.7 Kg/mm^2 for a linear as well as a nonlinear type of strain hardening material. The hardness value of 366.279 Kg/mm2 was input in the linear program. In the nonlinear program hardness values ranged between 188.891 Kg/mm^2 and 377.316 Kg/mm^2. The stress output from the programs are plotted in Fig. VII.4 and compared with the corresponding values calculated from expression (VII.1).

8Nadai, Plasticity, p. 188.

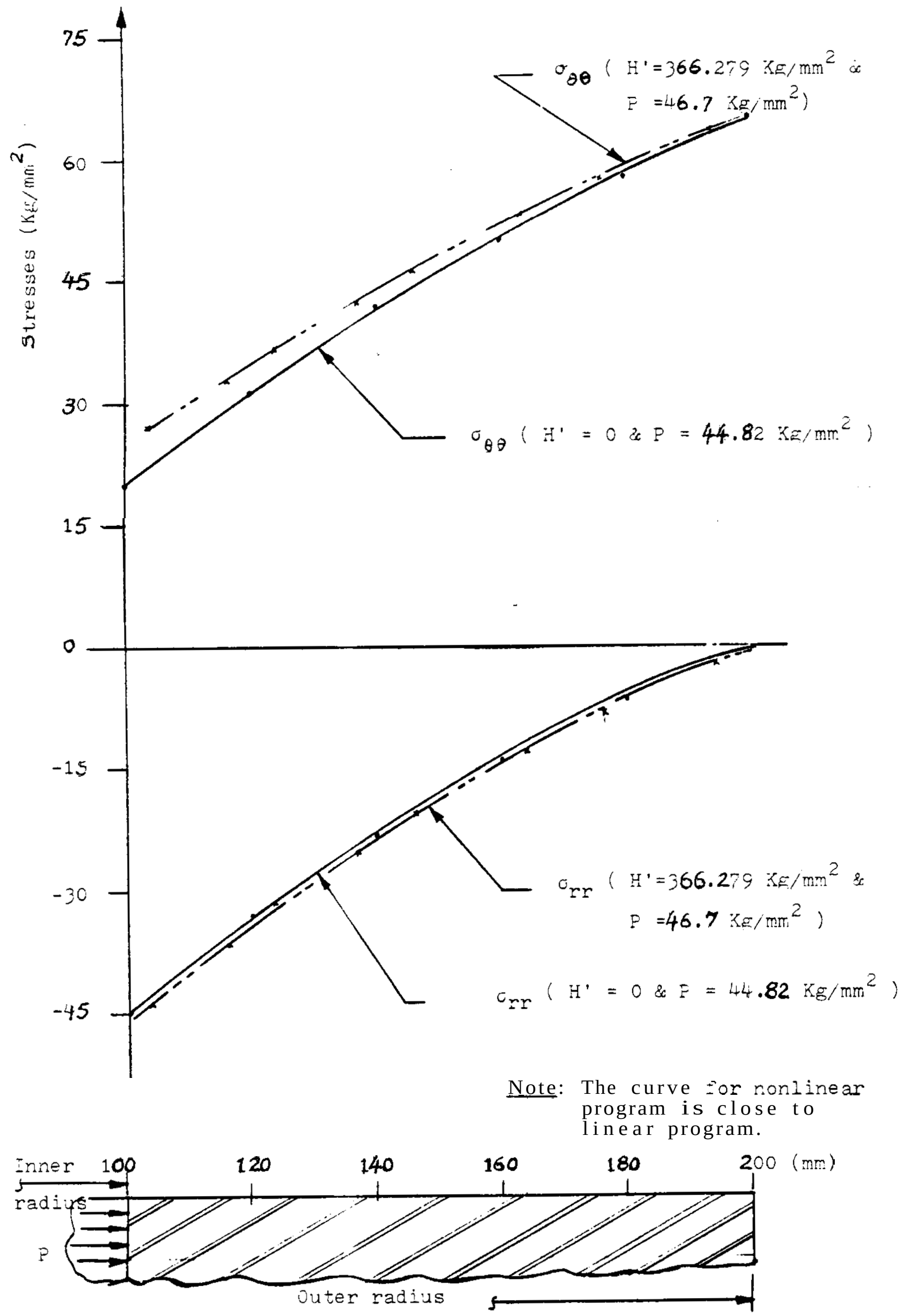


Fig. VII.4 Fully Plastic Cylinder– Comparison of Stresses for Perfectly Plastic Material and Linear Strain Hardening Material.

The resulting σ_{rr} , $\sigma_{\theta\theta}$ values are higher for a strain hardening type of material as was expected; however the differences in σ_{rr} ,and $\sigma_{\theta\theta}$ values between the program output and the closed form solutions of (VII.1) are small for a perfectly plastic material. It may indeed be worth-while for such fully plastic cylinder analysis to limit the analysis to relatively easy perfectly plastic type of mate-rial, for a large number of noncritical applications , to reduce computational time and cost.

Perforated Tension Strip

The third problem is the plane stress problem of a perforated tension strip. A perfectly plastic type of material is chosen for the strip and the strip itself is a 12" x 12" square plate. The perforations are small circular holes one inch in diameter. The problem is analysed to study the effect of stress concentrations and the collapse load as the number of perforations (holes) are increased from one to two and to three as explained below.

Infinite Plate With A Single Circular Hole

The problem under this case is an infinite plate with a single circular hole as perforation in the middle and subjected to uniform tension. This problem is analysed with forty quadratic elements and one hundred and forty seven nodes for one quarter of the plate with the help of the linear program. The spread of plastic zone at different levels of loading is shown in Fig. VII.5 . A 3.0% convergence tolerance (II.7) was set and it took a total CPU time of approximately 56 seconds to perform the analysis. It is seen from the enlarged scale drawing of spread of plastic zone, Fig. VII.6 , that as the strip tension σ increases, yielding spreads and very soon tends to progress along two comparatively narrow strips symmetrically situated with respect to the axis of tension and at an angle of about 45^{o} with the direction of tension. This has, of course, been confirmed by experiments. The stress distribution in a tension or compression member in the form of a wide plate containing a small cylindrical hole is well known in the case of an elastic material and the stresses are given by,

$$\sigma_{rr} = \frac{\sigma}{2}\left[1 - \frac{a^2}{r^2} + \left(1 - 4\frac{a^2}{r^2} + 3\frac{a^4}{r^4}\right)\cos(2\theta)\right],$$

$$\sigma_{\theta\theta} = \frac{\sigma}{2}\left[1 + \frac{a^2}{r^2} - \left(1 + 3\frac{a^4}{r^4}\right)\cos(2\theta)\right], \quad \text{VII.3}$$

where

σ = applied tension , a = radius of the hole,

and

r, θ = polar coordinates of the point. According to this expression the first yielding is expected to start at the two points situated on the boundary of the hole on a diameter perpendicular to the direction of tension, and at a value of σ equal to one third of yield stress.

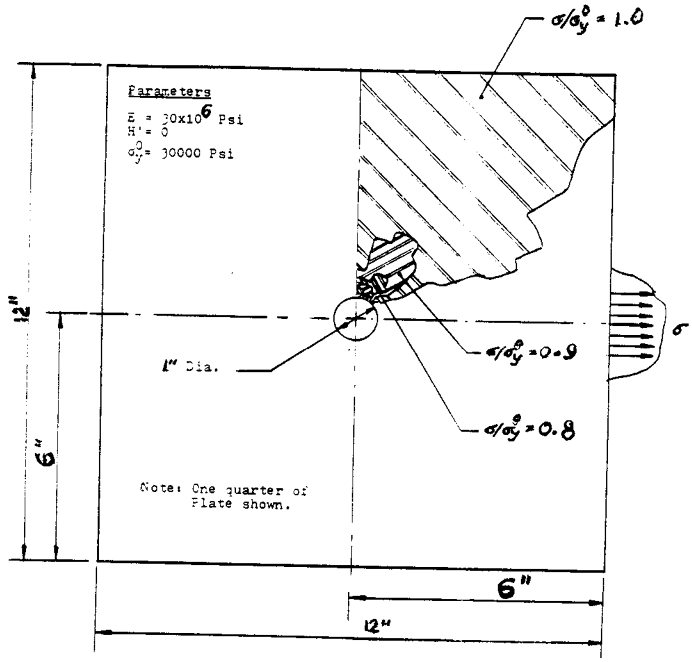
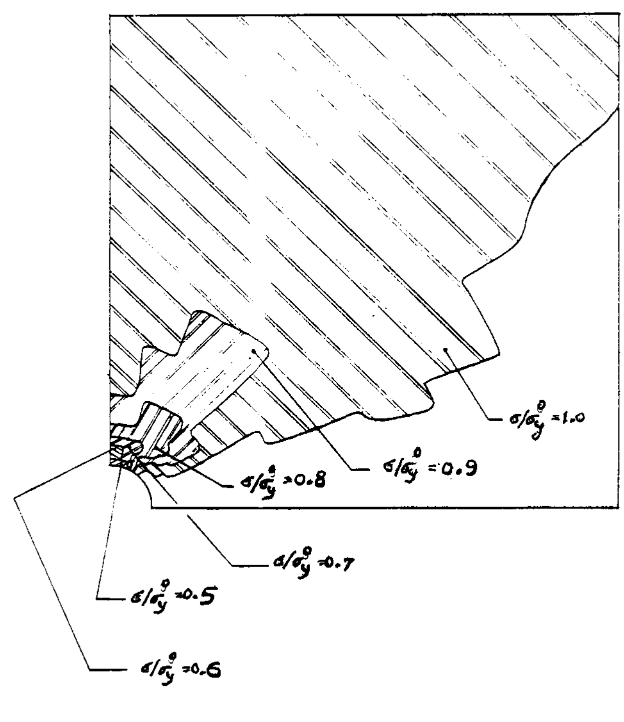


Fig. VII.5 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with a Single Circular Hole .



Pig, VII.6 Enlarged Scale Drawing of Spread of Plastic Zone for an Infinite Plate with a Single Circular Hole.

In numerical procedures the collapse condition is deemed to have occurred if the iterative analysis failed to converge for an incremental load. As expected the collapse load for this case is found to be the uniaxial yield stress.

Infinite Plate With Two(2) Circular Holes

As this is an extension of the previous case, all problem parameters are kept the same as for the previous case except the location of holes. One quarter of the plate was modeled using twenty six quadratic elements and one hundred and one nodes in this analysis. A 3.0 % convergence tolerance (11.7) was set and it took a total CPU time of about 47 seconds to complete the analysis.

The spread of plastic zones at different σ/σ_Y^0 values of loading is shown in Fig. VII.7. It would appear from the program results and the trial runs that the collapse load σ for this case would be 0.95 σ_Y^0 , where σ_Y^0 is the uniaxial yield stress,

Infinite Plate With Three(3) Circular Holes

As this is also an extension of the previous two cases, all problem parameters are kept the same as before except the location of holes. One quarter of the plate was modeled using forty-one quadratic elements and one hundred and fifty-six nodes. As in previous cases 3.0 % convergence tolerance (II.7) was set and it took a total CPU time of approximately 81 seconds to complete the analysis.

The spread of plastic zones at different levels of loading is shown in Fig. VII.8. The collapse load of this case is determined to be 0.85 $\sigma^0_{Y'}$ where σ^0_Y is the uniaxial yield stress.

The stress concentration factor k for all the cases is defined as:

$$k_{\sigma_\theta} = \frac{\text{Maximum value of } \sigma_{\theta\theta}}{\sigma} \qquad \text{VII.4}$$

Maximum values of tangential stresses and the stress concentration factors for these three cases are tabulated in Table VII.1 . The values of $\sigma_{\theta\theta}$ listed in Table VII.1 are the maximum values chosen from the program output at that level of loading.

In the elastic range of loading for the plate with a single circular hole $\sigma_{\theta\theta}$, and k_{σ_θ} can be evaluated from the expressions (VII.3) and (VII.4) . At σ/σ^0_Y value of 0.25, when the plasticity has not set in, the calculated $\sigma_{\theta 0}$ and k_{σ_θ} values of 17.5225 Ksi and 2.336 compare very well with the program output of 18.5934 Ksi for $\sigma_{\theta\theta}$ and 2.47912 for k_{σ_θ} for a Gauss point with r = 0.55253 and $\theta = 80.53^o$ .It should however be noted that the accuracy of the finite element solution can further be improved by placing more elements at the critical locations. In all the cases as the applied tension increases there is more plasticity and the stress concentration factor decreases. It would appear logical from elementary considerations to

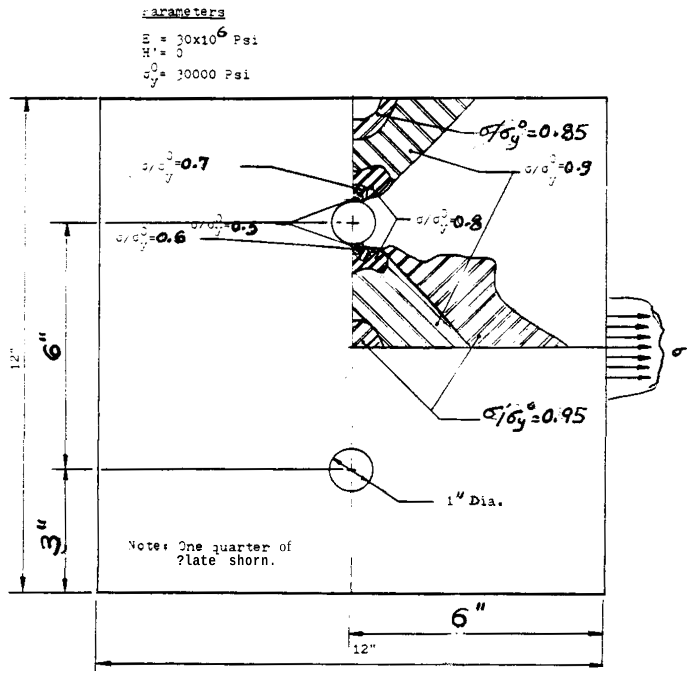


Fig. VII.7 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with Two Circular Holes.

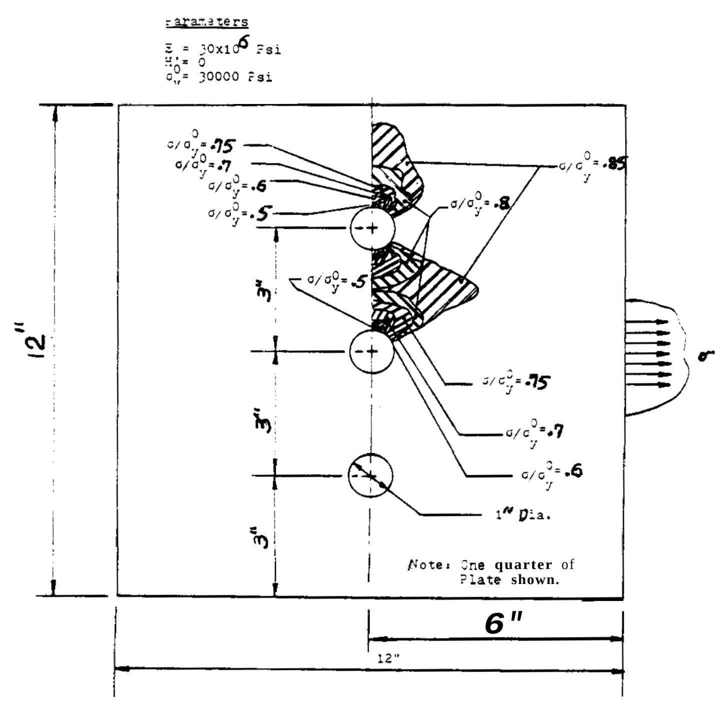


Fig. VII.8 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with Three Circular Holes.

TABLE VII.1

MAXIMUM VALUES OF TANGENTIAL STRESSES AND STRESS

CONCENTRATION FACTORS FOR A PERFORATED TENSION STRIP.

$\frac{\sigma_0}{\sigma_y}$	Infinite Plate With A Single Circular Hole		Infinite Plate With Two (2) Circular Holes		Infinite Plate With Three (3) Circular Holes	
	$\sigma_{\theta\theta}$ (Ksi)	k_{σ_θ}	$\sigma_{\theta\theta}$ (Ksi)	k_{σ_θ}	$\sigma_{\theta\theta}$	k_{σ_θ}
0.25	18.5934	2.479	18.384	2.451	19.3	2.573
0.50	32.445	2.163	32.327	2.155	34.42	2.295
0.60	34.905	1.939	35.058	1.948	35.91	1.995
0.70	36.526	1.739	36.296	1.728	38.53	1.835
0.75	-	-	-	-	39.68	1.764
0.80	37.51	1.563	37.529	1.564	41.39	1.725
0.85	-	-	-	-	42.40	1.663
0.90	40.038	1.483	39.274	1.455		
1.0	46.926	1.564	45.035	1.58		

expect a drop in the collapse load of tension strip proportional to the loss of strip surface area as the number of holes increase. The collapse load came down about 5% and 15% respectively for the tension strip with two and three holes, while the loss of strip surface area due to increased number of holes was 8.3% and 16.67% respectively. While the elastic analysis predicts an increased stress concentration factor, and therefore a conservative design, the elasto-plastic analysis would enormously help in furthering the understanding of stress concentration factors, stresses and collapse loads after the onset of plasticity and thus promote the better use of metals with aptly chosen factors of safety in machine design. For all the above cases of study, the displacement values at center line of strip along the loading direction are plotted in Big. VII.9 for different levels of loading. Such a displacement graph would greatly facilitate in accurately predicting the collapse load, as well as help in confining the design area to the left of the kink.

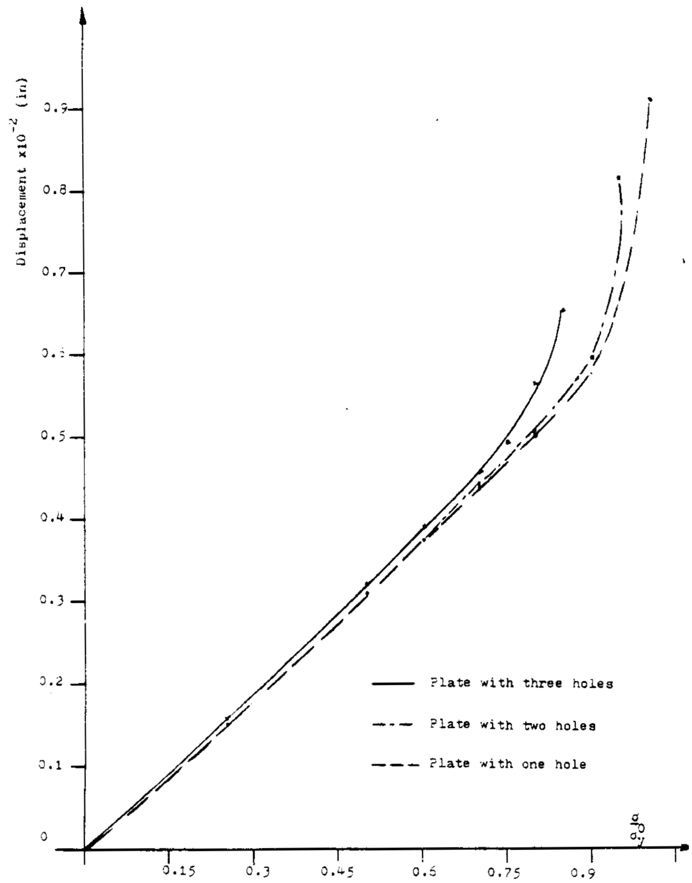


Fig. VII.9 Displacement Values at Different Levels of Loading for a Perforated Tension Strip.

CHAPTER VIII

CONCLUSION

Nonlinear Finite Element analysis is a powerful tool for solving engineering problems in which plasticity plays a dominant part. Appendix C contains a Finite Element Program for linear strain hardening material and Appendix D includes the modifications required for a complete Finite Element Program for nonlinear type of strain hardening materials. These programs are capable of solving two dimensional Plane Stress / Plane Strain / Axisymmetric Problems with a material only nonlinearity with a measure of confidence.

Finite Element formulation procedures and the essential Mathematical Theory of Plasticity are presented in Chapters I thru V.

The accuracy of the programs is established in Chapter VI for metal plasticity, by comparing the program output for two typical Elasto-Plastic Test Problems. The first problem is the plane stress problem of Elasto-Plastic Stress and Strain Concentration Factors at a Circular Hole in a Uniformly Stressed Infinite Plate. The second problem is the classical plane strain problem of a Thick Walled Cylinder Subjected to Internal Pressure, when the cylinder is elastic or partly plastic. The program outputs are in close agreement with theoretical predictions.

A small selection of elasto-plastic problems of practical interest have been solved in Chapter VII. The first problem solved is a CT Fracture Test Specimen. The second problem is an extension of the plane strain problem of Fully Plastic Thick Walled Cylinder Under Internal Pressure for a linear and nonlinear type of strain hardening material. The third problem solved is the plane stress problem of a Perforated Tension Strip, when the perforations are gradually increased from one to two to three.

The programs are comprehensive extending plasticity concepts to two types of metal plastic yield criteria, namely Tresca and Von Mises as well as potential surfaces which are used in rocks, concretes and soils. These programs have not been tested on nonmetals. Various iterative procedures ranging from the simple Initial Stress Method to the Tangential Stiffness Method as well as a combination of both methods have been incorporated in the same program. The nonlinear program can be used with ease if uniaxial test data are available for stress / strains at discrete number of test points,

Although the programs are effective the user must understand the problem well enough to make a Finite Element Model and to choose the load steps and convergence tolerance for an incremental analysis. It would be wise to do an Elastic Analysis first and then proceed to Elasto-Plastic Analysis with a large enough convergence tolerance of the order of 5-6% . Even then several analyses may be required for the same problem to get a satisfactory result. Since nonlinear analysis requires more computer time, a balance is to be struck between the desired accuracy and the computing costs. It would be desirable to start with small number of load increments and less number of iterations to know the order of total CPU time required. In the sample analyses conducted in Chapters VI and VII, generally total CPU time has been restricted to a maximum of approximately sixty(60) seconds for one complete nonlinear analysis. A nonlinear analysis may fail to converge, in spite of a good program, because of a bug in the data, numerical error, greater nonlinearity than what the program can accomodate, or a prescribed load greater than the structure's collapse load.

APPENDIX A

Instructions for Preparing Input Data for Linear Strain Hardening Program Master

INSTRUCTIONS FOR PREPARING INPUT DATA

FOR LINEAR STRAIN HARDENING PROGRAM MASTER

I. CARD SET 1 TITLE CARD(18A4) - One card.

Columns	Variables	Entry
1-72	--	Title of the problem.

II. CARD SET 2 CONTROL CARD(11I5) - One card.

Columns	Variables	Entry
1-5	NPOIN	Total number of nodal points.
6-10	NELEM	Total number of elements.
11-15	NVFIX	Total number of boundary points where one or more degrees of freedom are restrained.
16-20	NTYPE	Parameter defining the problem: 1-Plane stress, 2-Plane strain, 3-Axisymmetry.
21-25	NNODE	Number of nodes per element: 4-Linear quadrilateral element, 8-Quadratic serendipity element, 9-Quadratic Lagrangian element.
26-30	NMATS	Total number of different materials.
31-35	NGAUS	Order of numerical integration: 2-Two point Gauss quadrature rule 3-Three point Gauss quadrature rule.
36-40	NALGO	Nonlinear solution parameter: 1-Initial stiffness method. 2-Tangential stiffness method. 3-Combined algorithm, wherein element stiffnesses are recalculated for the first iteration of each load increment only. 4-Combined algorithm, wherein element stiffnesses are recalculated for the second iteration of each load increment only.
41-45	NCRIT	Yield criterion parameter: 1-Tresca. 2-Von Mises. 3-Mohr-Coulomb. 4-Drucker Prager.
46-50	NINCS	Number of increments in which the total loading is to be applied.
51-55	NSTRE	Number of independent stress components at a point: 3-Plane stress / Plane strain, 4-Axisymmetry.

III. CARD SET 3 ELEMENT CARDS(11I5) - One card for each element. Total NELEM number of cards.

Columns	Variables	Entry
1-5	NUMEL	Element number.
6-10	MATNO(NUMEL)	Material property number.
11-15	LNODS(NUMEL,1)	1st nodal connection number.
16-20	LNODS(NUMEL,2)	2nd nodal connection number.
		.
51-55	LNODS(NUMEL,9)	9th nodal connection number.

Notes:

1. Columns 31-55 remain blank for linear 4-noded elements.
2. Columns 51-55 remain blank for 8-noded elements.
3. The nodal connection number must be listed in an anticlockwise sequence starting from any corner node.

IV. CARD SET 4 NODE CARDS(I5,2F10.5) - One card for each node whose coordinates must be input.

Columns	Variables	Entry
1-5	IPOIN	Nodal point number.
6-15	COORD(IPOIN,1)	x(or r)coordinate of the node.
16-25	COORD(IPOIN,2)	y(or z)coordinate of the node.

Notes:

1. The total number of cards in this set may differ from NPOIN input in card set 2, since for quadratic elements whose sides are linear, intermediate nodal coordinates may be interpolated from corner nodes automatically.
2. For Lagrangial elements the coordinates of the 9th (central) node are never input.
3. The coordinates of the highest numbered node must be input regardless of whether it is a midside node or not.

V. CARD SET 5 RESTRAINED NODE CARDS(1X,I4,5X,I5,5X,2F10.5)- One card for each restrained node. Total of NVFIX cards. NVFIX is input in card set 2.

Columns	Variables	Entry
2-5	NOFIX(IVFIX)	Restrained node number.
11-15	IFPRE	Restraint code: 01-Nodal displacement restricted to x(or r) direction only. 10-Nodal displacement restiicted to y(or z) direction only.

Columns	Variables	Entry
		11-Nodal displacement restrained in both coordinate directions.
21-30	PRESC(IVFIX,1)	Prescribed value of x(or r) component of nodal displacement.
31-40	PRESC(IVFIX,2)	Prescribed value of y(or z) component of nodal displacement.

VI. CARD SET 6 MATERIAL CARDS
CONTROL CARD(I5) - One card.

Columns	Variables	Entry
1-5	NUMAT	Material identification number.

PROPERTIES CARDS(7F10.5) - One card for each different material.

1-10	PROPS(NUMAT,1)	Young's modulus, E.
11-20	PROPS(NUMAT,2)	Poission's ratio, ν.
21-30	PROPS(NUMAT,3)	Material thickness, t. (leave blank for plane strain and axisymmetric problems; note also that for plane stress problem loading is for unit thickness.)
31-40	PROPS(NUMAT,4)	Mass density, ρ.
41-50	PROPS(NUMAT,5)	Uniaxial yield stress, σ^0. (or cohesion factor for Y Mohr-Coulomb or Drucker-Prager materials)
51-60	PROPS(NUMAT,6)	Strain hardening parameter, H'.
61-70	PROPS(NUMAT,7)	Friction angle in degrees for Mohr-Coulomb and Drucker-Prager materials only.

Note: This card set to be repeated for each different material. Total of NMATS card sets. NMATS input in card set 2.

VII. CARD SET 7 LOAD CASE TITLE CARD(18A4) - One card.

Columns	Variables	Entry
1-72	--	Title of the load case.

VIII. CARD SET 8 LOAD CONTROL CARD(3I5) - One card.

Columns	Variables	Entry
1-5	IPLOD	Applied concentrated load control parameter:

Columns	Variables	Entry
		0–No concentrated loads. 1–Concentrated loads applied.
6-10	IGRAV	Gravity loading control parameter: 0–No gravity loading. 1– Gravity loads to be input.
11-15	IEDGE	Distributed edge loading control parameterr 0–No distributed edge loads. 1– Distributed edge loads are to be input.

IX. CARD SET 9 APPLIED LOAD CARDS(I5,2F10.3) - One card for each loade nodal point.

Columns	Variables	Entry
1-5	LODPT	Node number.
6-15	POINT(1)	Load component in x(or r) direction.
16-25	POINT(2)	Load component in y(or z) direction.

Notes:

1. The last card should be that for the highest numbered node whether it is loaded or not.
2. For axisymmetry problems the loads input should be the total loading on the circumferential ring passing through the nodal point concerned.
3. If IPLOD =0 in card set 8 omit this set.

X. CARD SET 10 GRAVITY LOADING CARD(2F10.3) - One card.

Columns	Variables	Entry
1-10	THETA	Angle of gravity axis measured from the positive y-axis. See Fig. 1.3.
11-20	GRAVY	Gravity constant, specified as a multiple of the gravitational acceleration g.

Note: If IGRAV =0 in card set 8 omit this set.

XI. CARD SET 11 DISTRIBUTED EDGE LOAD CARDS CONTROL CARD(I5) - One card.

Columns	Variables	Entry
1-5	NEDGE	Number of element edges on which distributed loads are to be applied.

ELEMENT FACE TOPOLOGY CARD(4I5)

Columns	Variables	Entry
1-5	NEASS	The element number with which the edge is associated.
6-10	NOPRS(1)	List of nodal points in an
11-15	NOPRS(2)	anticlockwise sequence of the
16-20	NOPRS(3)	nodes of the element on which the distributed loading acts.

Note: For linear 4-noded element the columns remain blank.

DISTRIBUTED LOAD CARDS(6F10.3)

Columns	Variables	Entry
1-10	PRESS(1,1)	Value of normal component of distributed load at node NOPRS(1).
11-20	PRESS(1,2)	Value of tangential component of distributed load at node NOPRS(1).
21-30	PRESS(2,1)	Value of normal component of distributed load at node NOPRS(2).
31-40	PRESS(2,2)	Value of tangential conponent of distributed load at node NOPRS(2).
41-50	PRESS(3,1)	Value of normal component of distributed load at node NOPRS(3).
51-60	PRESS(3,2)	Value of tangential component of distributed load at node NOPRS(3).

Notes:

1. For 4-noded elements columns 41-60 remain blank.
2. Element face topology card and distributed load cards must be repeated for every element edge on which a distributed load acts. The element edges can be considered in any order.
3. If IEDGE =0 in card set 8 omit this card set.

XII. CARD SET 12 LOAD INCREMENT CONTROL CARDS(2F10.5,3I5)- One card for each load increment. Total of NINCS cards. NINCS input in card set 2.

Columns	Variables	Entry
1-10	FACTO	Applied load factor for this increment specified as a factor of the loading input in card sets 8 to 11.
11-20	TOLER	Convergence tolerance factor.
21-25	MITER	Maximum number of iterations allowed for the load increment.

Columns	Variables	Entry
26-30	NOUTP(1)	Output control parameter after 1st iteration; 0-No output. 1-Output displacements, 2-Output displacements and reactions. 3-Output displacements, reactions and stresses.
31-35	NOUTP(2)	Output control parameter for converged results: 0-No output. 1-Output displacements. 2-Output displacements and reactions. 3-Output displacements, reactions and stresses.

Note: The applied loading factors are cumulative. If FACTO is specified as 0.2,0.3,0.4 for the first three load increments, then the total load during the third increment is 0.9 times the load input in card sets 8 to 11.

APPENDIX B

Instructions for Preparing Input Data for Nonlinear Strain Hardening Program Mster2

INSTRUCTIONS FOR PREPARING INPUT DATA FOR NONLINEAR STRAIN HARDENING PROGRAM MSTER2

I. CARD SETS 1 THRU 5 - Same as the card sets 1 thru 5 presented in Appendix A.

II. CARD SET 6 MATERIAL CARDS
CONTROL CARD(I5) - One card.

Columns	Variables	Entry
1-5	NUMAT	Material identification number.

PROPERTIES CARDS(6F10.5) - One card for each different material.

1-10	PROPS(NUMAT,1)	Young's modulus, E.
11-20	PROPS(NUMAT,2)	Poission's ratio, ν.
21-30	PROPS(NUMAT,3)	Material thickness, t. (leave blank for plane strain and axisymmetric problems; note also that for plane stress problem loading is for unit thickness.)
31-40	PROPS(NUMAT,4)	Mass density, ρ.
41-50	PROPS(NUMAT,5)	Uniaxial yield stress, σ_Y^0 (or cohesion factor for Mohr-Coulomb or Drucker-Prager materials).
51-60	PROPS(NUMAT,6)	Friction angle in degrees for Mohr-Coulomb and Drucker-Prager materials only.

Note: This card set to be repeated for each different material. Total of NMATS card sets. NMATS input in card set 2.

III. CARD SET 7 UNIAXIAL TEST DATA
CONTROL CARD(5X,I5)-One card.

Columns	Variables	Entry
6-10	NUMAT	Material identification number.

NUMBER OF TEST DATA POINTS(5X,I5)-One card.

6-10	NUMB	Number of uniaxial test data points.

UNIAXIAL TEST STRESS AND TOTAL STRAIN VALUES (F10.4,6X,E14.8)-One card for each test point.

Columns	Variables	Entry
1-10	USTRES(NUMAT,INUMB)	Uniaxial test stress.
17-30	USTRN(NUMAT,INUMB)	Uniaxial test total strain corresponding to the test stress.

Notes:

1. USTRES(NUMAT,INUMB) must be input in increasing order.
2. USTRES(NUMAT,INUMB) and USTRN(NUMAT,INUMB) must form the pairs of test stress and total strain.
3. Total NUMB cards input.

IV. CARD SET 8 THRU 13 - Same as the card sets 7 thru 12 presented in Appendix A.

APPENDIX C

Finite Element Program For Linear Strain Hardening Materials

FINITE ELEMENT PROGRAM FOR LINEAR STRAIN HARDENING MATERIALS

```
C       MASTER PLAST
C*****************************************************************************
C       PROGRAM FOR THE ELASTO-PLASTIC ANALYSIS OF PLANE STRESS*
C       PLANE STRAIN AND AXISYMMETRIC SOLIDS
C*****************************************************************************
        IMPLICIT REAL*8(A-H,O-Z)
        DIMENSION ASDIS(400),COORD(180,2),ELOAD(50,18),ESTIF(18,18),
     .          EQRHS(10), EQUAT (100,10),FIXED(360),GLOAD(80),GSTIF(3240),
     .              IFFIX(360),LNODS(50,9),LOCEL(18),MATNO(50),
     .              NACVA(80),NAMEV(10),NDEST(18),NDFRO(50),NOFIX(50),
     .              NOUTP(2), NPIVO(10),
     *              POSGP(4),PRESC(30,2),PROPS(5,7),RLOAD(50,18),
     .              STFOR(360),TREAC(50,2),VECRV(80),WEIGP(4),
     .              STRSG(4,360),TDISP(360),TLOAD(50,18),
     4              TOFOR(360),EPSTN(360),EFFST(360)
C
C***  PRESET VARIABLES ASSOCIATED WITH DYNAMIC DIMENSIONING
C
        CALL DIMEN(MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,MSTIF,MTOTG,MTOTV,
                MVFIX,NDOFN,NPROP,NSTRE)
C
C***  CALL THE SUBROUTINE YHICH READS HOST OF THE PROBLEM DATA
C
        CALL INPUT(COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,MFRON,MMATS,
     *          MPOIN,MTOTV,MVFIX,NALGO,
     *          NCRIT,NDFRO,NDOFN,NELEM,NEVAB,NGAUS,NGAU2,
     *          NINCS, NMATS,NNODE,NOFIX,NPOIN,NPROP,NSTRE,
     *          NSTR1,NTOTG,NTOTV,
     .          NTYPE,NVFIX,POSGP,PRESC,PROPS,WEIGP)
C
C***  CALL THE SURROUTINE YHICH COMPUTES THE CONSISTENT LOAD VECTORS
C     FOR EACH ELEMENT AFTER READING THE RELEVANT INPUT DATA
C
        CALL LOADPS(COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,NELEM,
     .          NEVAB,NGAUS,NNODE,NPOIN,NSTRE,NTYPE,POSGP,
                PROPS,RLOAD,WEIGP,NDOFN)
C
C***  INITIALIZE CERTAIN ARRAYS
C
        CALL ZERO(ELOAD,MELEM,MEVAB,MPOIN,MTOTG,MTOTV,NDOFN,NELEM,
     *          NEVAB,NGAUS,NSTR1,NTOTG,EPSTN,EFFST,
                NTOTV,NVFIX,STRSG,TDISP,TFACT,
     *          TLOAD,TREAC,MVFIX)
C
C***  LOOP OVER EACH INCREMENT
C
        DO 100 IINCS =1,NINCS
C
C***  READ DATA FOR CURRENT INCREMENT
C
        CALL INCREM(ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER,MTOTV,
     .          MVFIX,NDOFN,NELEM,NEVAB,NOUTP,NOFIX,NTOTV,
     .          NVFIX,PRESC,RLOAD,TFACT,TLOAD,TOLER)
C
C***  LOOP OVER EACH ITERATION
```

```
      DO 50 IITER = 1,MITER
C
C*** CALL ROUTINE WHICH SELECTS SOLUTION ALGJRITHH VARIABLE KRESL
C
      CALL ALGOR(FIXED,IINCS,IITER,KRESL,MTOTV,NALGO,
     .           NTOTV)
C
C*** CHCCK WHETHFR A NEW EYALUATXON OF THE STIFFNESS MATRIX IS REQUIRED
C
      IF (KRESL.EQ.1) CALL STIFFP(COORD,EPSTN,IINCS,LNODS,MATNO,
     .                MEVAB,MMATS,MPOIN,MTOTV,NELEM,NEVAB,NGAUS,NNODE,
     .                NSTRE,NSTRI,POSGP,PROPS,WEIGP,MELEM,MTOTG,
     .                STRSG,NTYPE,NCRIT)
C
C*** SOLVE EQUATIONS
C
      CALL FRONT(ASDIS,ELOAD,EQRHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS,IITER,
     .           GLOAD,GSTIF,LOCEL,LNODS,KRESL,MBUFA,MELEM,MEVAB,MFRON,
     .           MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,NDOFN,NELEM,NEVAB,
     .           NNODE,NOFIX,NPIVO,NPOIN,NTOTV,TDISP,TLOAD,TREAC,
     .           VECRV)
C
C*** CALCULATE RESIDUAL FORCES
C
      CALL RESIDU(ASDIS,COORD,EFFST,ELOAD,FACTO,IITER,LNODS,
     .            LPROP,MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,NDOFN,
     .            NELEM,NEVAB,NGAUS,NNODE,NSTRE,TDISP,EPSTN,POSGP,PROPS,
     .            NSTRE,NCRIT,STRSG,WEIGP,TDISP,EP)
C
C*** CHECK FOR CONVERGENCE
C
      CALL CONVER(ELOAD,IITER,LNODS,MELEM,MEVAB,MTOTV,NCHEK,NDOFN,
     .            NELEM,NEVAB,NNODE,NTOTV,PVALU,STFOR,TLOAD,TOFOR,TOLER)
C
C**** OUTPUT RESULTS IF REQUIRED
C
      IF (IITER.EQ.1.AND.NOUTP(1).GT.0)
     . CALL OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
     .             NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,NTYPE,NCHEK,
     .             EFFST)
C
C*** IF SOLUTION HAS CONVERGED STOP ITERATING AND OUTPUT RESULTS
C
      IF (NCHEK.EQ.0) GO TO 75
   50 CONTINUE
C***
C
      IF (NALGO.EQ.2) GO TO 75
      STOP
   75 CALL OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
     .             NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,NTYPE,NCHEK,
     .             EFFST)
  100 CONTINUE
      STOP
      END
```

```
      SUBROUTINE DIMEN(MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,MSTIF,MTOTG,
     *                 MTOTV,MVFIX,NDOFN,NPROP,NSTRE)
C**********************************************************************
C
C**** THIS SUBROUTINE PRESETS VARIABLES ASSOCIATED WITH DYNAMIC
C     DIMENSIONING
C
C**********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      MBUFA =10
      MELEM =50
      MFRON =80
      MMATS =5
      MPOIN =180
      MSTIF =(MFRON*MFRON-MFRON)/2.0+MFRON
      MTOTG =MELEM*9
      NDOFN =2
      MTOTV =MPOIN*NDOFN
      MVFIX =30
      NPROP =7
      MEVAB =NDOFN*9
      RETURN
      END
```

```
      SUBROUTINE INPUT(COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,MFRON,MMATS,
     *                 MPOIN,MTOTV,MVFIX,NALGO,
     *                 NCRIT,NDFRO,NDOFN,NELEM,
     *                 NEVAB,NGAUS,NGAU2,
     *                 NINCS,NMATS,NNODE,NOFIX,NPOIN,NPROP,NSTRE,NSTR1,
     *                 NTOTG,NTOTV,NTYPE,NVFIX,POSGP,PRESC,PROPS,WEIGP)
C**********************************************************************
C
C*** THIS SUBROUTINE ACCEPTS MOST OF THE INPUT DATA
C
C**********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION COORD(MPOIN,2),IFFIX(MTOTV),LNODS(MELEM,9),
     *          MATNO(MELEM),NDFRO(MELEM),
     *          NOFIX(MVFIX),POSGP(4),PRESC(MVF X,NDOFN),
     *          NOFIX(MVFIX),POSGP(4),PRESC(MVFIX,NDC
     *          PROPS(MMATS,NPROP),TITLE(18),WEIGP(4)
      REWIND 1
      REWIND 2
      REWIND 3
      REWIND 4
      REWIND 8
      READ (5,920) TITLE
      WRITE (6,920) TITLE
  920 FORMAT(18A4)
C
C*** READ THE FIRST DATA CARD AND ECHO IT IMMEDIATELY
C
      READ(5,900) NPOIN,NELEM,NVFIX,NTYPE,NNODE,NMATS,NGAUS,
     *            NALGO,NCRIT,NINCS,NSTRE
  900 FORMAT(11I5)
      NEVAB = NDOFN*NNODE
      NSTR1 = NSTRE+1
      IF (NTYPE.EQ.3) NSTR1 = NSTRE
      NTOTV = NPOIN*NDOFN
      NGAU2 = NGAUS*NGAUS
      NTOTG = NELEM*NGAU2
      WRITE(6,901) NPOIN,NELEM,NVFIX,NTYPE,NNODE,NMATS,NGAUS,NEVAB,
     *             NALGO,NCRIT,NINCS,NSTRE
  901 FORMAT (//8H NPOIN =,I4,4X,8H NELEM =,I4,4X,8H NVFIX =,I4,4X,
     *8H NTYPE =,I4,4X,8H NNODE =,I4,//
     *8H NMATS =,I4,4X,8H NGAUS =,I4,
     *4X,8H NEVAB =,I4,4X,8H NALGO =,I4//
     *8H NCRIT =,I4,4X,8H NINCS =,I4,4X,8H NSTRE =,I4)
      CALL CHECK1(NDOFN,NELEM,NGAUS,NMATS,NNODE,NPOIN,
     *            NSTRE,NTYPE,NVFIX,NCRIT,NALGO,NINCS)
C
C*** READ THE ELEMENT NODAL CONNECTIONS,AND THE PROPERTY NUMBERS
C
      WRITE(6,902)
  902 FORMAT (//8H ELEMENT,3X,8HPROPERTY,6X,12HNODE NUMBERS)
      DO 2 IELEM = 1,NELEM
      READ(5,900) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE)
    2 WRITE(6,903) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE)
  903 FORMAT(1X,I5,I9,6X,8I5)
C
```

```
C*** ZERO ALL THE NODAL COORDINATES,PRIOR TO READING SOME OF THEM.
C
      DO 4 IPOIN = 1,NPOIN
      DO 4 IDIME = 1,2
    4 COORD(IPOIN,IDIME) = 0.0
C
C*** READ SOME NODAL COORDINATES,FINISHING WITH THE LAST NODE OF ALL.
C
      WRITE(6,904)
  904 FORMAT(//5H NODE,10X,1HX,10X,1HY)
    6 READ(5,905) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,2)
  905 FORMAT(I5,6F10.5)
      IF (IPOIN.NE.NPOIN) GO TO 6
C
C*** INTERPOLATE COORDINATES OF MID-SIDE NODES
C
      CALL NODEXY(COORD,LNODS,MELEM,MPOIN,NELEM,NNODE)
      DO 10 IPOIN =1,NPOIN
   10 WRITE(6,906) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,2)
  906 FORMAT(1X,I5,3F10.3)
C
C*** READ THE FIXED VALUES
C
      WRITE(6,907)
  907 FORMAT(//5H NODE,6X,4HCODE,6X,12HFIXED VALUES)
      DO 8 IVFIX = 1,NVFIX
      READ(5,908) NOFIX(IVFIX),IFPRE,(PRESC(IVFIX,IDOFN),IDOFN=1,NDOFN)
      WRITE(6,908) NOFIX(IVFIX),IFPRE,(PRESC(IVFIX,IDOFN),IDOFN=1,NDOFN)
      NLOCA = (NOFIX(IVFIX)-1)*NDOFN
      IFDOF = 10**(NDOFN-1)
      DO 8 IDOFN = 1,NDOFN
      NGASH = NLOCA+IDOFN
      IF(IFPRE.LT.IFDOF) GO TO 8
      IFFIX(NGASH)=1
      IFPRE=IFPRE-IFDOF
    8 IFDOF = IFDOF/10
  908 FORMAT(1X,I4,5X,I5,5X,5F10.6)
C
C*** READ THE AVAILABLE SELECTION OF ELEMENT PROPERTIES
C
   16 WRITE(6,910)
  910 FORMAT(//7H NUMBER,6X,18HELEMENT PROPERTIES)
      DO 18 IMATS =1,NMATS
      READ(5,900) NUMAT
      READ(5,930) (PROPS(NUMAT,IPROP),IPROP=1,NPROP)
  930 FORMAT(7F10.5)
   18 WRITE(6,911) NUMAT,(PROPS(NUMAT,IPROP),IPROP=1,NPROP)
  911 FORMAT(1X,I4,3X,8E14.6)
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL GAUSSQ(NGAUS,POSGP,WEIGP)
      CALL CHECK2(COORD,IFFIX,LNODS,MATNO,MELEM,MFRON,MPOIN,MTOTV,
     *            MVFIX,NDFRO,NDOFN,NELEM,NMATS,NNODE,NOFIX,NPOIN,
     *            NVFIX)
      RETURN
      END
```

```
      SUBROUTINE CHECK1(NDOFN,NELEM,NGAUS,NMATS,NNODE,NPOIN,
     .                  NSTRE,NTYPE,NVFIX,NCRIT,NALGO,NINCS)
C*********************************************************************
C
C*** THIS SUBROUTINE CHECKS THE MAIN CONTROL DATA
C
C*********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION NEROR(24)
      DO 10 IEROR = 1,12
   10 NEROR(IEROR)=0
C
C*** CREATE THE DIAGNOSTIC MESSAGES
C
      IF (NPOIN.LE.0) NEROR(1) = 1
      IF (NELEM*NNODE.LT.NPOIN) NEROR(2) = 1
      IF (NVFIX.LT.2.OR.NVFIX.GT.NPOIN) NEROR(3) = 1
      IF (NINCS.LT.1) NEROR(4) = 1
      IF (NTYPE.LT.1.OR.NTYPE.GT.3) NEROR(5)=1
      IF (NNODE.LT.4.OR.NNODE.GT.9) NEROR(6) = 1
      IF (NDOFN.LT.2.OR.NDOFN.GT.5) NEROR(7) = 1
      IF (NMATS.LT.1.OR.NMATS.GT.NELEM) NEROR(8) = 1
      IF (NGAUS.LT.2.OR.NGAUS.GT.3) NEROR(10) =1
      IF (NALGO.LT.1.OR.NALGO.GT.4) NEROR(11) = 1
      IF (NSTRE.LT.3.OR.NSTRE.GT.5) NEROR(12) = 1
C
C*** EITHER RETURN,OR ELSE PRINT THE ERRORS DIAGNOSED
C
      KEROR = 0
      DO 20 IEROR = 1,12
      IF (NEROR(IEROR).EQ.0) GO TO 20
      KEROR = 1
      WRITE(6,900) IEROR
  900 FORMAT(//31H *** DIAGNOSIS BY CHECK1, ERROR,I3)
   20 CONTINUE
      IF(KEROR.EQ.0) RETURN
C
C*** OTHERWISE ECHO ALL THE REMAINING DATA WITHOUT FURTHER COMMENT
C
      CALL ECHO
      END
```

```
      SUBROUTINE GAUSSQ(NGAUS,POSGP,WEIGP)
C*********************************************************************
C
C*** THIS SUBROUTINE SETS UP THE GAUSS-LEGENDRE INTEGRATION CONSTANTS
C
C*********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION POSGP(4),WEIGP(4)
      IF (NGAUS.GT.2) GO TO 4
    2 POSGP(1) = -0.577350269189626
      WEIGP(1) = 1.0
      GO TO 6
    4 POSGP(1)=-0.774596669241483
      POSGP(2) = 0.0
      WEIGP(1) = 0.555555555555556
      WEIGP(2) = 0.888888888888889
    6 KGAUS = NGAUS/2
      DO 8 IGASH = 1,KGAUS
      JGASH = NGAUS+1-IGASH
      POSGP(JGASH) = -POSGP(IGASH)
      WEIGP(JGASH) = WEIGP(IGASH)
    8 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE NODEXY(COORD,LNODS,MELEM,MPOIN,NELEM,NNODE)
C*****************************************************************************
C
C**** THIS SUBROUTINE INTERPOLATES THE MID SIDE NODES OF STRAIGHT
C     SIDES OF ELEMENTS AND THE CENTRAL NODE OF 9 NODED ELEMENTS
C
C*****************************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION COORD(MPOIN,2),LNODS(MELEM,9)
      IF (NNODE.EQ.4) RETURN
C
C*** LOOP OVER EACH ELEMENT
C
      DO 30 IELEM = 1,NELEM
C
C*** LOOP OVER EACH ELEMENT EDGE
C
      NNOD1 = 9
      IF (NNODE.EQ.8) NNOD1 = 7
      DO 20 INODE = 1,NNOD1,2
      IF (INODE.EQ.9) GO TO 50
C
C*** COMPUTE THE NODE NUMBER OF THE FIRST NODE
C
      NODST = LNODS(IELEM,INODE)
      IGASH = INODE+2
      IF (IGASH.GT.8) IGASH = 1
C
C*** COMPUTE THE NODE NUMBER OF THE LAST NODE
C
      NODFN = LNODS(IELEM,IGASH)
      MIDPT = INODE+1
C
C*** COMPUTE THE NODE NUMBER OF THE INTERMEDIATE NODE
C
      NODMD = LNODS(IELEM,MIDPT)
      TOTAL =DABS(COORD(NODMD,1))+DABS(COORD(NODMD,2))
C
C*** IF THE COORDINATES OF THE INTERMEDIATE NODE ARE BOTH ZERO
C    INTERPOLATE BY A STRAIGHT LINE
C
      IF (TOTAL.GT.0.0) GO TO 20
      KOUNT = 1
   10 COORD(NODMD,KOUNT) = (COORD(NODST,KOUNT)+COORD(NODFN,KOUNT))/2.0
      KOUNT = KOUNT+1
      IF (KOUNT.EQ.2) GO TO 10
   20 CONTINUE
      GO TO 30
   50 LNODE=LNODS(IELEM,INODE)
      TOTAL =DABS(COORD(LNODE,1))+DABS(COORD(LNODE,2))
      IF (TOTAL.GT.0.0) GO TO 30
      LNOD1=LNODS(IELEM,1)
      LNOD3 = LNODS(IELEM,3)
      LNOD5 = LNODS(IELEM,5)
      LNOD7 = LNODS(IELEM,7)

      KOUNT = 1
   40 COORD(LNODE,KOUNT) = (COORD(LNOD1,KOUNT)+COORD(LNOD3,KOUNT)
     *+COORD(LNOD5,KOUNT)+COORD(LNOD7,KOUNT))/4.0
      KOUNT = KOUNT+1
      IF (KOUNT.EQ.2) GO TO 40
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CHECK2(COORD,IFFIX,LNODS,MATNO,MELEM,MFRON,MPOIN,MTOTV,
     .                  MVFIX,NDFRO,NOOFN,NELEM,NMATS,NNODE,NOFIX,NPOIN
C*********************************************************************
C
C**** THIS SUBROUTINE CHECKS THE REMAINDER OF THE INPUT DATA
C
C*********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION COORD(MPOIN,2),IFFIX(MTOTV),LNODS(MELEM,9),
     .          MATNO(MELEM),NDFRO(MELEM),NEROR(24),NOFIX(MVFIX)
C
C*** CHECK AGAINST TWO IDENTICAL NONZERO NODAL COORDINATES
C
      DO 5 IEROR = 13,24
    5 NEROR(IEROR) = 0
      DO 10 IELEM = 1,NELEM
   10 NDFRO(IELEM) = 0
      DO 40 IPOIN = 2,NPOIN
      KPOIN = IPOIN-1
      DO 30 JPOIN = 1,KPOIN
      DO 20 IDIME = 1,2
      IF(COORD(IPOIN,IDIME).NE.COORD(JPOIN,IDIME)) GO TO 30
   20 CONTINUE
      NEROR(13) = NEROR(13)+1
   30 CONTINUE
   40 CONTINUE
C
C**** CHECK THE LIST OF ELEMENT PROPERTY NUMBERS
C
      DO 50 IELEM = 1,NELEM
   50 IF(MATNO(IELEM).LE.0.OR. MATNO(IELEM).GT.NMATS) NEROR(14)=
     . NEROR(14)+1
C
C**** CHECK FOR IMPOSSIBLE NODE NUMBERS
C
      DO 70 IELEM = 1,NELEM
      DO 60 INODE = 1,NNODE
      IF (LNODS(IELEM,INODE).EQ.0) NEROR(15) = NEROR(15)+1
   60 IF (LNODS(IELEM,INODE).LT.0.OR.LNODS(IELEM,INODE).GT.NPOIN)
     . NEROR(16)=NEROR(16)+1
   70 CONTINUE
C
C**** CHECK FOR ANY REPETITION OF A NODE NUMBER WITHIN AN ELEMENT
C
      DO 140 IPOIN = 1,NPOIN
      KSTAR = 0
      DO 100 IELEM = 1,NELEM
      KZERO = 0
      DO 90 INODE = 1,NNODE
      IF (LNODS(IELEM,INODE).NE.IPOIN) GO TO 90
      KZERO = KZERO+1
      IF(KZERO.GT.1) NEROR(17)=NEROR(17)+1
C
C*** SEEK FIRST,LAST,AND INTERMEDIATE APPEARANCES OF NODE
```

```
C
      IF(KSTAR.NE.0) GO TO 80
      KSTAR = IELEM
C
C***  CALCULATE INCREASE OR DECREASE IN FRONTWIDTH AT EACH ELEMENT STAGE
C
      NDFRO(IELEM) = NDFRO(IELEM)+NDOFN
 00   CONTINUE
C
C***  AND CHANGE THE SIGN OF THE LAST APPEARANCE OF EACH NODE
C
      KLAST = IELEM
      NLAST = INODE
 90   CONTINUE
100   CONTINUE
      IF(KSTAR.EQ.0) GO TO 110
      IF(KLAST.LT.NELEM) NDFRO(KLAST+1) = NDFRO(KLAST+1)-NDOFN
      LNODS(KLAST,NLAST) = -IPOIN
      GO TO 140
C
C***  CHECK THAT COORDINATES FOR AN UNUSED NODE HAVE NOT BEEN SPECIFIED
C
110   WRITE(6,900) IPOIN
900   FORMAT(/15H CHECK WHY NODE,I4,14H NEVER APPEARS)
      NERORR(18)=NERORR(18)+1
      SIGMA = 0.0
      DO 120 IDIME = 1,2
120   SIGMA=SIGMA+DABS(COORD(IPOIN,IDIME))
      IF (SIGMA.NE.0.0) NEROR(19) = NEROR(19)+1
C
C***  CHECK THAT AN UNUSED NODE NUMBER IS NOT A RESTRAINED NODE
C
      DO 130 IVFIX = 1,NVFIX
130   IF(NOFIX(IVFIX).EQ.IPOIN) NEROR(20) = NEROR(20) +1
140   CONTINUE
C
C***  CALCULATE THE LARGEST FRONT WIDTH
C
      NFRON = 0
      KFRON = 0
      DO 150 IELEM = 1,NELEM
      NFRON = NFRON+NDFRO(IELEM)
150   IF (NFRON.GT.KFRON) KFRON = NFRON
      WRITE(6,905) KFRON
905   FORMAT(//33H MAXIMUM FRONTWIDTH ENCOUNTERED =,I5)
      IF(KFRON.GT.MFRON) NEROR(21) =1
C***  CONTINUE CHECKING THE DATA FOR THE FIXED VALUES
C
C
      DO 170 IVFIX= 1,NVFIX
      IF(NOFIX(IVFIX).LE.0.OR.NOFIX(IVFIX).GT.NPOIN) NEROR(22)=
     . NEROR(22)+1
      KOUNT =0
      NLOCA= (NOFIX(IVFIX)-1)*NDOFN
      DO 160 IDOFN = 1,NDOFN
      NLOCA = NLOCA+1
160   IF(IFFIX(NLOCA).GT.0) KOUNT = 1
      IF (KOUNT.EQ.0) NEROR(23) = NEROR(23)+1
      KVFIX = IVFIX-1
      DO 170 JVFIX = 1,KVFIX
170   IF(IVFIX.NE.1.AND.NOFIX(IVFIX).EQ.NOFIX(JVFIX)) NEROR(24)
     . = NEROR(24)+1
      KEROR = 0
      DO 180 IEROR = 13,24
      IF(NEROR(IEROR).EQ.0) GO TO 180
      KEROR =1
      WRITE(6,910) IEROR,NEROR(IEROR)
910   FORMAT(//31H *** DIAGNOSIS BY CHECK2,ERROR,I3,6X,
     .        18H ASSOCIATED NUMBER ,I5)
180   CONTINUE
      IF(KEROR.NE.0) GO TO 200
C
C***  RETURN ALL NODAL CONNECTION NUMBERS TO POSITIVE VALUES
C
      DO 190 IELEM=1,NELEM
      DO 190 INODE=1,NNODE
190   LNODS(IELEM,INODE) = IABS(LNODS(IELEM,INODE))
      RETURN
200   CALL ECHO
      END
```

```
      SUBROUTINE ECHO
C***********************************************************************
C
C*** IF DATA ERRORS HAVE BEEN DETECTED BY SUBROUTINES CHECK1 OR
C    CHECK2, THIS SUBROUTINE READS AND WRITES THE REMAINING DATA CARDS
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION NTITL(80)
      WRITE (6,900)
  900 FORMAT(//50H NOW FOLLOWS A LISTING OF POST-DISASTER DATA CARDS/)
   10 READ (5,905)NTITL
  905 FORMAT(80A1)
      WRITE (6,910) NTITL
  910 FORMAT(20X,80A1)
      GO TO 10
      END
```

```
      SUBROUTINE LOADPS(COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,NELEM,
     .                  NEVAB,NGAUS,NNODE,NPOIN,NSTRE,NTYPE,POSGP,
     .                  PROPS,RLOAD,WEIGP,NDOFN)
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE CONSISTENT NODAL FORCES FOR EACH
C     ELEMENT
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION CARTD(2,9),COORD(MPOIN,2),DERIV(2,9),DGASH(2),
     .          DMATX(4,4),ELCOD(2,9),LNODS(MELEM,9),MATNO(MELEM),
     .          NOPRS(4),PGASH(2),POINT(2),POSGP(4),PRESS(4,2),
     .          PROPS(MMATS,7),RLOAD(MELEM,18),SHAPE(9),STRAN(4),
     .          STRES(4),TITLE(18),
     .          WEIGP(4),GPCOD(2,9)
      TWOPI = 6.283185308
      DO 10 IELEM =1,NELEM
      DO 10 IEVAB = 1,NEVAB
   10 RLOAD(IELEM,IEVAB) =0.0
      READ(5,901) TITLE
  901 FORMAT(18A4)
      WRITE(6,903) TITLE
  903 FORMAT(1H0,18A4)
C
C*** READ DATA CONTROLLING LOADING-TYPES TO BE INPUTED
C
      READ(5,919) IPLOD,IGRAV,IEDGE
      WRITE(6,919) IPLOD,IGRAV,IEDGE
  919 FORMAT(3I5)
C
C*** READ NODAL POINT LOADS
C
      IF (IPLOD.EQ.0) GO TO 500
   20 READ (5,931) LODPT,(POINT(IDOFN),IDOFN=1,2)
      WRITE (6,931) LODPT,(POINT(IDOFN),IDOFN=1,2)
  931 FORMAT(I5,2F10.3)
C
C*** ASSOCIATE THE NODAL POINT LOADS WITH AN ELEMENT
C
       DO 30 IELEM = 1,NELEM
       DO 30 INODE = 1,NNODE
       NLOCA = IABS(LNODS(IELEM,INODE))
      IF (LODPT.EQ.NLOCA) GO TO 40
   30 CONTINUE
   40 DO 50 IDOFN =1,2
      NGASH = (INODE-1)*2+IDOFN
   50 RLOAD(IELEM,NGASH) = POINT(IDOFN)
      IF (LODPT.LT.NPOIN) GO TO 20
  500 CONTINUE
      IF (IGRAV.EQ.0) GO TO 600
C
C*** GRAVITY LOADING SECTION
C
C
```

```
C*** READ GRAVITY ANGLE AND GRAVITATIONAL CONSTANT
C
      READ (5,906) THETA,GRAVY
  906 FORMAT(2F10.3)
      WRITE (6,911) THETA,GRAVY
  911 FORMAT(1H0,16H GRAVITY ANGLE =,F10.3,19H GRAVITY CONSTANT =,F10.3)
      THETA = THETA/57.295779514
C
C*** LOOP OVER EACH ELEMENT
C
      DO 90 IELEM = 1,NELEM
C
C*** SET UP PRELIMINARY CONSTANTS
C
      LPROP = MATNO(IELEM)
      THICK = PROPS(LPROP,3)
      DENSE = PROPS(LPROP,4)
      IF (DENSE.EQ.0.0) GO TO 90
      GXCOM = DENSE*GRAVY*DSIN(THETA)
      GYCOM = DENSE*GRAVY*DCOS(THETA)
C
C*** COMPUTE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 60 INODE = 1,NNODE
      LNODE = IABS(LNODS(IELEM,INODE))
      DO 60 IDIME = 1,2
   60 ELCOD(IDIME,INODE) = COORD(LNODE,IDIME)
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      KGASP = 0
      DO 80 IGAUS = 1,NGAUS
      DO 80 JGAUS = 1,NGAUS
      EXISP = POSGP(IGAUS)
      ETASP = POSGP(JGAUS)
C
C*** COMPUTE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS AND ELEMENTAL
C    VOLUME
C
      CALL SFR2 (DERIV,ETASP,EXISP,NNODE,SHAPE)
      KGASP = KGASP+1
      CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .             NNODE,SHAPE )
      DVOLU = DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
      IF (NTYPE.EQ.3) DVOLU = DVOLU*TWOPI*GPCOD(1,KGASP)
C
C*** CALCULATE LOADS AND ASSOCIATE WITH ELEMENT NODAL POINTS
C
      DO 70 INODE = 1,NNODE
      NGASH = (INODE-1)*2+1
      MGASH = (INODE-1)*2+2
      RLOAD(IELEM,NGASH) = RLOAD(IELEM,NGASH)+GXCOM*SHAPE(INODE)*DVOLU
   70 RLOAD(IELEM,MGASH) = RLOAD(IELEM,MGASH)+GYCOM*SHAPE(INODE)*DVOLU
   80 CONTINUE
```

```
   90 CONTINUE
  100 CONTINUE
      IF (IEDGE.EQ.0) GO TO 700
C
C*** DISTRIBUTED EDGE LOADS SECTION
C
      READ (5,932) NEDGE
  932 FORMAT(I5)
      WRITE (6,912) NEDGE
  912 FORMAT (1H0,5X,21HNO. OF LOADED EDGES =,I5)
      WRITE (6,915)
  915 FORMAT (1H0,5X,38HLIST OF LOADED EDGES AND APPLIED LOADS)
      NODEG = 3
      NCODE = NNODE
      IF (NNODE.EQ.4) NODEG = 2
      IF (NNODE.EQ.9) NCODE = 8
C
C*** LOOP OVER EACH LOADED EDGE
C
      DO 160 IEDGE = 1,NEDGE
C
C*** READ DATA LOCATING THE LOADED EDGE AND APPLIED LOAD
C
      READ (5,902) NEASS,(NOPRS(IODEG),IODEG=1,NODEG)
  902 FORMAT(4I5)
      WRITE (6,913) NEASS,(NOPRS(IODEG),IODEG=1,NODEG)
  913 FORMAT (I10,5X,3I5)
      READ (5,914) ((PRESS(IODEG,IDOFN),IDOFN=1,2),IODEG=1,NODEG)
      WRITE (6,914) ((PRESS(IODEG,IDOFN),IDOFN =1,2),IODEG=1,NODEG)
  914 FORMAT(6F10.3)
      ETASP = -1.0
C
C*** CALCULATE THE COORDINATES OF THE NODES OF THE ELEMENT EDGE
C
      DO 100 IODEG = 1,NODEG
      LNODE = NOPRS(IODEG)
      DO 100 IDIME = 1,2
  100 ELCOD(IDIME,IODEG) = COORD(LNODE,IDIME)
C
C*** ENTER LOOPS FOR LINEAR NUMERICAL INTEGRATION
C
      DO 150 IGAUS = 1,NGAUS
      EXISP = POSGP(IGAUS)
C
C*** EVALUATE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS
C
      CALL SF92 (DERIV,ETASP,EXISP,NNODE,SHAPE)
C
C*** CALCULATE COMPONENTS OF THE EQUIVALENT NODAL LOADS
C
      DO 110 IDOFN = 1,2
      PGASH (IDOFN)=0.0
      DGASH (IDOFN) =0.0
      DO 110 IODEG = 1,NODEG
      PGASH(IDOFN) = PGASH(IDOFN)+PRESS(IODEG,IDOFN)*SHAPE(IODEG)

  110 DGASH (IDOFN) = DGASH(IDOFN)+ELCOD(IDOFN,IODEG)*DERIV(1,IODEG)
      DVOLU = WEIGP(IGAUS)
      PXCOM = DGASH(1)*PGASH(2)-DGASH(2)*PGASH(1)
      PYCOM = DGASH(1)*PGASH(1)+DGASH(2)*PGASH(2)
      IF(NTYPE.NE.3) GO TO 115
      RADUS =0.0
      DO 125 IODEG =1,NODEG
  125 RADUS = RADUS +SHAPE(IODEG)*ELCOD(1,IODEG)
      DVOLU = DVOLU*TWOPI*RADUS
  115 CONTINUE
C
C*** ASSOCIATE THE EQUIVALENT NODAL EDGE LOADS WITH AN ELEMENT
C
      DO 120 INODE = 1,NNODE
      NLOCA = IABS(LNODS(NEASS,INODE))
      IF (NLOCA.EQ.NOPRS(1)) GO TO 130
  120 CONTINUE
  130 JNODE = INODE+NODEG-1
      KOUNT = 0
      DO 140 KNODE = INODE,JNODE
      KOUNT = KOUNT+1
      NGASH = (KNODE-1)*NODFN+1
      MGASH = (KNODE-1)*NODFN+2
      IF (KNODE.GT.NCODE) NGASH = 1
      IF (KNODE.GT.NCODE) MGASH = 2
      RLOAD(NEASS,NGASH) = RLOAD(NEASS,NGASH)+SHAPE(KOUNT)*PXCOM*DVOLU
  140 RLOAD(NEASS,MGASH)=RLOAD(NEASS,MGASH)+SHAPE(KOUNT)*PYCOM*DVOLU
  150 CONTINUE
  160 CONTINUE
  700 CONTINUE
      WRITE(6,907)
  907 FORMAT(1H0,5X,36H TOTAL NODAL FORCES FOR EACH ELEMENT)
      DO 290 IELEM = 1,NELEM
  290 WRITE (6,905) IELEM, (RLOAD(IELEM,IEVAB),IEVAB=1,NEVAB)
  905 FORMAT(1X,I4,5X,8E12.4/(10X,8E12.4))
      RETURN
      END
```

```
      SUBROUTINE ZERO (ELOAD,MELEM,MEVAB,MPOIN,MTOTG,MTOTV,NDOFN,NELEM,
     .                 NEVAB,NGAUS,NSTR1,NTOTG,EPSTN,EFFST,
     .                 NTOTV,NVFIX,STRSG,TDISP,TFACT,
     .                 TLOAD,TREAC,MVFIX)
C***********************************************************************
C
C**** THIS SUBROUTINE INITIALISES VARIOUS ARRAYS TO ZERO
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ELOAD (MELEM,MEVAB),STRSG(4,MTOTG),TDISP(MTOTV),
     .          TLOAD(MELEM,MEVAB),TREAC(MVFIX,2),EPSTN(MTOTG),
     .          EFFST(MTOTG)
      TFACT =0.0
      DO 30 IELEM =1,NELEM
      DO 30 IEVAB =1,NEVAB
      ELOAD(IELEM,IEVAB)= 0.0
   30 TLOAD(IELEM,IEVAB)= 0.0
      DO 40 ITOTV =1,NTOTV
   40 TDISP(ITOTV)=0.0
      DO 50 IVFIX= 1,NVFIX
      DO 50 IDOFN =1,NDOFN
   50 TREAC(IVFIX,IDOFN)= 0.0
      DO 60 ITOTG =1,NTOTG
      EPSTN(ITOTG)=0.0
      EFFST(ITOTG)= 0.0
      DO 60 ISTR1 =1,NSTR1
   60 STRSG(ISTR1,ITOTG)= 0.0
      RETURN
      END
```

```
      SUBROUTINE INCREM(ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER,
     .                  MTOTV,MVFIX,NDOFN,NELEM,NEVAB,NOUTP,
     .                  NOFIX,NTOTV,NVFIX,PRESC,RLOAD,TFACT,
     .                  TLOAD,TOLER)
C***********************************************************************
C
C*** THIS SUHROUTINE INCREMENTS THE APPLIED LOADING
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ELOAD(MELEM,MEVAB),FIXED(MTOTV),IFFIX(MTOTV),
     .          NOUTP(2),NOFIX(MVFIX),PRESC(MVFIX,NDOFN),
     .          RLOAD(MELEM,MEVAB),TLOAD(MELEM,MEVAB)
      WRITE(6,900) IINCS
  900 FORMAT(1H0,5X,17HINCREMENT NUMBER ,I5)
      READ(5,950) FACTO,TOLER,MITER,NOUTP(1),NOUTP(2)
  950 FORMAT(2F10.5,3I5)
      TFACT = TFACT+FACTO
      WRITE(6,960) TFACT,TOLER,MITER,NOUTP(1),NOUTP(2)
  960 FORMAT(1H0,5X,13HLOAD FACTOR =,F10.5,5X,
     .24H CONVERGENCE TOLERANCE =,F10.5,5X,24HMAX. NO. OF ITERATIONS =,
     . I5,//27H INITIAL OUTPUT PARAMETER =,I5,5X,
     .24HFINAL OUTPUT PARAMETER=,I5)
      DO 80 IELEM = 1,NELEM
      DO 80 IEVAB = 1,NEVAB
      ELOAD(IELEM,IEVAB)=ELOAD(IELEM,IEVAB)+RLOAD(IELEM,IEVAB)*FACTO
   80 TLOAD(IELEM,IEVAB)=TLOAD(IELEM,IEVAB)+RLOAD(IELEM,IEVAB)*FACTO
C
C*** INTERPRET FIXITY DATA IN VECTOR FORM
C
      DO 100 ITOTV = 1,NTOTV
  100 FIXED(ITOTV) = 0.0
      DO 110 IVFIX = 1,NVFIX
      NLOCA = (NOFIX(IVFIX)-1)*NDOFN
      DO 110 IDOFN = 1,NDOFN
      NGASH = NLOCA+IDOFN
      FIXED(NGASH) = PRESC(IVFIX,IDOFN)*FACTO
  110 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE ALGOR(FIXED,IINCS,IITER,KRESL,
     .                  MTOTV,NALGO,NTOTV)
C***********************************************************************
C
C**** THIS SUBROUTINE SETS EQUATION RESOLUTION INDEX, KRESL
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION FIXED(MTOTV)
      KRESL = 2
      IF (NALGO.EQ.1.AND.IINCS.EQ.1.AND.IITER.EQ.1) KRESL =1
      IF (NALGO.EQ.2) KRESL = 1
      IF (NALGO.EQ.3.AND.IITER.EQ.1) KRESL = 1
      IF (NALGO.EQ.4.AND.IINCS.EQ.1.AND.IITER.EQ.1) KRESL = 1
      IF (NALGO.EQ.4.AND.IITER.EQ.2) KRESL=1
      IF (IITER.EQ.1) RETURN
      DO 100 ITOTV = 1,NTOTV
      FIXED(ITOTV) = 0.0
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE STIFFP(COORD,EPSTN,IINCS,LNODS,MATNO,MEVAB,MMATS,
     .                  MPOIN,MTOTV,NELEM,NEVAB,NGAUS,NNODE,NSTRE,
     .                  NSTR1,POSGP,PROPS,WEIGP,MELEM,MTOTG,
     .                  STRSG,NTYPE,NCRIT)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE STIFFNESS MATRIX FOR EACH ELEMENT
C    IN TURN
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION BMATX(4,18),CARTD(2,9),COORD(MPOIN,2),DBMAT(4,18),
     .          DERIV(2,9),DEVIA(4),DMATX(4,4),
     .          ELCOD(2,9),EPSTN(MTOTG),ESTIF(18,18),LNODS(MELEM,9),
     .          MATNO(MELEM),POSGP(4),PROPS(MMATS,7),SHAPE(9),
     .          WEIGP(4),STRES(4),STRSG(4,MTOTG),
     .          DVECT(4),AVECT(4),GPCOD(2,9)
      TWOPI = 6.283185308
      REWIND 1
      KGAUS = 0
C
C*** LOOP OVER EACH ELEMENT
C
      DO 70 IELEM = 1,NELEM
      LPROP = MATNO(IELEM)
C
C*** EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 10 INODE = 1,NNODE
      LNODE = IABS(LNODS(IELEM,INODE))
      IPOSN =(LNODE-1)*2
      DO 10 IDIME = 1,2
      IPOSN = IPOSN + 1
   10 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
      THICK = PROPS(LPROP,3)
C
C*** INITIALIZE THE ELEMENT STIFFNESS MATRIX
C
      DO 20 IEVAB = 1,NEVAB
      DO 20 JEVAB = 1,NEVAB
   20 ESTIF(IEVAB,JEVAB) = 0.0
      KGASP = 0
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      DO 50 IGAUS = 1,NGAUS
      EXISP = POSGP(IGAUS)
      DO 50 JGAUS = 1,NGAUS
      ETASP = POSGP(JGAUS)
      KGASP = KGASP+1
      KGAUS = KGAUS+1
C
C*** EVALUATE THE D-MATRIX
C
      CALL MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
```

```
C
C*** EVALUATE THE SHAPE FUNCTIONS,ELEMENTAL VOLUME ETC.
C
      CALL SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .            NNODE,SHAPE)
      DVOLU = DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF (NTYPE.EQ.3) DVOLU = DVOLU*TWOPI*GPCOD(1,KGASP)
      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      IF (IINCS.EQ.1) GO TO 80
      IF (EPSTN(KGAUS).EQ.0.0) GO TO 80
      DO 90 ISTR1 = 1,NSTR1
   90 STRES(ISTR1) = STRSG(ISTR1,KGAUS)
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,
     .           THETA,VARJ2,YIELD)
      CALL YIELDF(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .            PROPS,SINT3,STEFF,THETA,VARJ2)
      CALL FLOWPL(AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS)
      DO 100 ISTRE = 1,NSTRE
      DO 100 JSTRE = 1,NSTRE
  100 DMATX(ISTRE,JSTRE)=DMATX(ISTRE,JSTRE)-ABETA*DVECT(ISTRE)*
     *DVECT(JSTRE)
   80 CONTINUE
      CALL DBE(BMATX,DBMAT,DMATX,MEVAB,NEVAB,NSTRE,NSTR1)
C
C*** CALCULATE THE ELEMENT STIFFNESSES
C
      DO 50 IEVAB = 1,NEVAB
      DO 50 JEVAB = IEVAB,NEVAB
      DO 30 ISTRE = 1,NSTRE
   30 ESTIF(IEVAB,JEVAB)=ESTIF(IEVAB,JEVAB)+BMATX(ISTRE,IEVAB)*
     *DBMAT(ISTRE,JEVAB)*DVOLU
   50 CONTINUE
C
C*** CONSTRUCT THE LOWER TRIANGLE OF THE STIFFNESS MATRIX
C
      DO 60 IEVAB = 1,NEVAB
      DO 60 JEVAB = 1,NEVAB
   60 ESTIF(JEVAB,IEVAB) = ESTIF(IEVAB,JEVAB)
C
C*** STORE THE STIFFNESS MATRIX,STRESS MATRIX,AND SAMPLING POINT
C    COORDINATES FOR EACH ELEMENT ON DISC FILE
C
      WRITE(1) ESTIF
   70 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FRONT(ASOIS,ELOAD,EQRHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS,
     .                 IITER,GLOAD,GSTIF,LOCEL,LNODS,KRESL,MBUFA,MELEM,
     .                 MEVAB,MFRON,MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,
     .                 NDOFN,NELEM,NEVAB,NNODE,NOFIX,NPIVO,NPOIN,
     *                 NTOTV,TDISP,TLOAD,TREAC,VECRV)
C**********************************************************************
C
C**** THIS SUBROUTINE UNDERTAKES EQUATION SOLUTION BY THE FRONTAL
C     METHOD
C
C**********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ASDIS(MTOTV),ELOAD(MELEM,MEVAB),EQRHS(MBUFA),
     *          EQUAT(MFRON,MBUFA),ESTIF(MEVAB,MEVAB),FIXED(MTOTV),
     *          IFFIX(MTOTV),NPIVO(MBUFA),VECRV(MFRON),GLOAD(MFRON),
     *          GSTIF(MSTIF),LNODS(MELEM,9),LOCEL(MEVAB),NACVA(MFRON),
     *          NAMEV(MBUFA),NDEST(MEVAB),NOFIX(MVFIX),NOUTP(2),
     +          TDISP(MTOTV),TLOAD(MELEM,MEVAB),TREAC(MVFIX,NDOFN)
      NFUNC(I,J) = (J*J-J)/2+I
C
C*** CHANGE THE SIGN OF THE LAST APPEARANCE OF EACH NODE
C
      IF (IINCS.GT.1.OR.IITER.GT.1) GO TO 455
      DO 140 IPOIN = 1,NPOIN
      KLAST = 0
      DO 130 IELEM = 1,NELEM
      DO 120 INODE = 1,NNODE
      IF (LNODS(IELEM,INODE).NE.IPOIN) GO TO 120
      KLAST = IELEM
      NLAST = INODE
  120 CONTINUE
  130 CONTINUE
      IF (KLAST.NE.0) LNODS(KLAST,NLAST) = -IPOIN
  140 CONTINUE
  455 CONTINUE
C
C*** START BY INITIALIZING EVERYTHING THAT MATTERS TO ZERO
C
      DO 450 IBUFA = 1,MBUFA
  450 EQRHS(IBUFA) = 0.0
      DO 150 ISTIF = 1,MSTIF
  150 GSTIF(ISTIF) = 0.0
      DO 160 IFRON = 1,MFRON
      GLOAD(IFRON) = 0.0
      VECRV(IFRON) = 0.0
      NACVA(IFRON) = 0
      DO 160 IBUFA = 1,MBUFA
  160 EQUAT(IFRON,IBUFA) = 0.0
C
C*** AND PREPARE FOR DISC READING AND WRITING OPERATIONS
C
      NBUFA = 0
      IF (KRESL.GT.1) NBUFA = MBUFA
      REWIND 1
      REWIND 2
```

```
      REWIND 2
      REWIND 4
      REWIND 8
C
C*** ENTER MAIN ELEMENT ASSEMBLY-REDUCTION LOOP
C
      NFRON = 0
      KELVA = 0
      DO 320 IELEM = 1,NELEM
      IF(KRESL.GT.1) GO TO 400
      KEVAB = 0
      READ(1) ESTIF
      DO 170 INODE = 1,NNODE
      DO 170 IDOFN = 1,NDOFN
      NPOSI = (INODE-1)*NDOFN+IDOFN
      LOCNO = LNODS(IELEM,INODE)
      IF (LOCNO.GT.0) LOCEL(NPOSI) = (LOCNO-1)*NDOFN+IDOFN
      IF (LOCNO.LT.0) LOCEL(NPOSI) = (LOCNO+1)*NDOFN-IDOFN
  170 CONTINUE
C
C*** START BY LOOKING FOR EXISTING DESTINATIONS
C
      DO 210 IEVAB = 1,NEVAB
      NIKNO = IABS(LOCEL(IEVAB))
      KEXIS = 0
      DO 180 IFRON = 1,NFRON
      IF (NIKNO.NE.NACVA(IFRON)) GO TO 180
      KEVAB = KEVAB+1
      KEXIS = 1
      NDEST(KEVAB) = IFRON
  180 CONTINUE
      IF (KEXIS.NE.0) GO TO 210
C
C*** WE NOW SEEK NEW EMPTY PLACES FOR DESTINATION VECTOR
C
      DO 190 IFRON = 1,MFRON
      IF (NACVA(IFRON).NE.0) GO TO 190
      NACVA(IFRON) = NIKNO
      KEVAB = KEVAB+1
      NDEST(KEVAB) = IFRON
      GO TO 200
  190 CONTINUE
C
C*** THE NEW PLACES MAY DEMAND AN INCREASE IN CURRENT FRONTWIDTH
C
  200 IF (NDEST(KEVAB).GT.NFRON) NFRON = NDEST(KEVAB)
  210 CONTINUE
      WRITE(8) LOCEL,NDEST,NACVA,NFRON
  400 IF(KRESL.GT.1) READ(8) LOCEL,NDEST,NACVA,NFRON
C
C*** ASSEMBLE ELEMENT LOADS
C
      DO 220 IEVAB = 1,NEVAB
      IDEST = NDEST(IEVAB)
      GLOAD(IDEST) = GLOAD(IDEST)+ELOAD(IELEM,IEVAB)
```

```
C
C*** ASSEMBLE THE ELEMENT STIFFNESSES-BUT NOT IN RESOLUTION
C
      IF(KRESL.GT.1) GO TO 402
      DO 222 JEVAB = 1,IEVAB
      JDEST =NDEST(JEVAB)
      NGASH = NFUNC(IDEST,JDEST)
      NGISH = NFUNC(JDEST,IDEST)
      IF(JDEST.GE.IDEST) GSTIF(NGASH)=GSTIF(NGASH)+ESTIF(IEVAB,JEVAB)
      IF(JDEST.LT.IDEST) GSTIF(NGISH)=GSTIF(NGISH)+ESTIF(IEVAB,JEVAB)
  222 CONTINUE
  402 CONTINUE
  220 CONTINUE
C
C*** RE-EXAMINE EACH ELEMENT NODE,TO ENQUIRE WHICH CAN BE ELIMINATED
C
      DO 310 IEVAB = 1,NEVAB
      NIKNO=-LOCEL(IEVAB)
      IF(NIKNO.LE.0) GO TO 310
C
C*** FIND POSITIONS OF VARIABLES READY FOR ELIMINATION
C
      DO 300 IFRON = 1,NFRON
      IF(NACVA(IFRON).NE.NIKNO) GO TO 300
      NBUFA = NBUFA+1
C
C*** WRITE EQUATIONS TO DISC OR TO TAPE
C
      IF (NBUFA.LE.MBUFA) GO TO 406
      NBUFA = 1
      IF(KRESL.GT.1) GO TO 408
      WRITE(2) EQUAT,EQRHS,NPIVO,NAMEV
      GO TO 406
  408 WRITE(4) EQRHS
      READ(2) EQUAT,EQRHS,NPIVO,NAMEV
  406 CONTINUE
C
C*** EXTRACT THE COEFFICIENTS OF THE NEW EQUATION FOR ELIMINATION
C
      IF (KRESL.GT.1) GO TO 404
      DO 230 JFRON = 1,MFRON
      IF (IFRON.LT.JFRON) NLOCA = NFUNC(IFRON,JFRON)
      IF (IFRON.GE.JFRON) NLOCA = NFUNC(JFRON,IFRON)
      EQUAT(JFRON,NBUFA) = GSTIF(NLOCA)
  230 GSTIF(NLOCA) = 0.0
  404 CONTINUE
C
C*** AND EXTRACT THE CORRESPONDING RIGHT HAND SIDES
C
      EQRHS(NBUFA) = GLOAD(IFRON)
      GLOAD(IFRON) = 0.0
      KELVA = KELVA+1
      NAMEV(NBUFA) = NIKNO
      NPIVO(NBUFA) = IFRON
C
```

```
C*** DEAL WITH PIVOT
C
      PIVOT = EQUAT(IFRON,NBUFA)
      IF(PIVOT.GT.0.0) GO TO 235
      WRITE(6,900) NIKNO,PIVOT
  900 FORMAT(1H0,3X,52HNEGATIVE OR ZERO PIVOT ENCOUNTERED FOR VARIABLE N
     *O. ,I4,10H OF VALUE ,E17.6)
      STOP
  235 CONTINUE
      EQUAT(IFRON,NBUFA) = 0.0
C
C*** ENQUIRE WHETHER PRESENT VARIABLE IS FREE OR PRESCRIBED
C
      IF (IFFIX(NIKNO).EQ.0) GO TO 250
C
C*** DEAL WITH A PRESCRIBED DEFLECTION
C
      DO 240 JFRON = 1,NFRON
  240 GLOAD(JFRON) = GLOAD(JFRON)-FIXED(NIKNO)*EQUAT(JFRON,NBUFA)
      GO TO 280
C
C*** ELIMINATE A FREE VARIABLE-DEAL WITH THE RIGHT HAND SIDE FIRST
C
  250 DO 270 JFRON = 1,NFRON
      GLOAD(JFRON) = GLOAD(JFRON)-EQUAT(JFRON,NBUFA)*EQRHS(NBUFA)/PIVOT
C
C*** NOW DEAL WITH THE COEFFICIENTS IN CORE
C
      IF(KRESL.GT.1) GO TO 418
      IF(EQUAT(JFRON,NBUFA).EQ.0.0) GO TO 270
      NLOCA = NFUNC(0,JFRON)
      CUREQ = EQUAT(JFRON,NBUFA)
      DO 260 LFRON = 1,JFRON
      NGASH = LFRON+NLOCA
  260 GSTIF(NGASH) = GSTIF(NGASH)-CUREQ*EQUAT(LFRON,NBUFA)
     * /PIVOT
  418 CONTINUE
  270 CONTINUE
  280 EQUAT(IFRON,NBUFA) = PIVOT
C
C*** RECORD THE NEW VACANT SPACE,AND REDUCE FRONTWIDTH IF POSSIBLE
C
      NACVA(IFRON) = 0
      GO TO 290
C
C*** COMPLETE THE ELEMENT LOOP IN THE FORWARD ELIMINATION
C
  300 CONTINUE
  290 IF (NACVA(NFRON).NE.0) GO TO 310
      NFRON = NFRON-1
      IF(NFRON.GT.0) GO TO 290
  310 CONTINUE
  320 CONTINUE
      IF (KRESL.EQ.1) WRITE(2) EQUAT,EQRHS,NPIVO,NAMEV
      BACKSPACE 2
```

```
C
C*** ENTER BACK-SUBSTITUTION PHASE.LOOP BACKWARDS THROUGH VARIABLES
C
      DO 340 IELVA = 1,KELVA
C
C*** READ A NEW BLOCK OF EQUATIONS - IF NEEDED
C
      IF (NBUFA.NE.0) GO TO 412
      BACKSPACE 2
      READ(2) EQUAT,EQRHS,NPIVO,NAMEV
      BACKSPACE 2
      NBUFA = MBUFA
      IF (KRESL.EQ.1) GO TO 412
      BACKSPACE 4
      READ(4) EQRHS
      BACKSPACE 4
  412 CONTINUE
C
C*** PREPARE TO BACK-SUBSTITUTE FROM THE CURRRENT EQUATION
C
      IFRON = NPIVO(NBUFA)
      NIKNO = NAMEV(NBUFA)
      PIVOT = EQUAT(IFRON,NBUFA)
      IF(IFFIX(NIKNO).NE.0) VECRV(IFRON) = FIXED(NIKNO)
      IF(IFFIX(NIKNO).EQ.0) EQUAT(IFRON,NBUFA) = 0.0
C
C*** BACK-SUBSTITUTE IN THE CURRENT EQUATION
C
      DO 330 JFRON = 1,NFRON
  330 EQRHS(NBUFA) = EQRHS(NBUFA)-VECRV(JFRON)*EQUAT(JFRON,NBUFA)
C
C*** PUT THE FINAL VALUES WHERE THEY BELONG
C
      IF (IFFIX(NIKNO).EQ.0) VECRV(IFRON)=EQRHS(NBUFA)/PIVOT
      IF (IFFIX(NIKNO).NE.0) FIXED(NIKNO)=-EQRHS(NBUFA)
      NBUFA = NBUFA-1
      ASDIS(NIKNO) = VECRV(IFRON)
  340 CONTINUE
C
C*** ADD DISPLACEMENTS TO PREVIOUS TOTAL VALUES
C
      DO 345 ITOTV = 1,NTOTV
  345 TDISP(ITOTV) = TDISP(ITOTV)+ASDIS(ITOTV)
C
C*** STORE REACTIONS FOR PRINTING LATER
C
      KBOUN = 1
      DO 370 IPOIN = 1,NPOIN
      NLOCA = (IPOIN-1)*NDOFN
      DO 350 IDOFN = 1,NDOFN
      NGUSH = NLOCA+IDOFN
      IF (IFFIX(NGUSH).GT.0) GO TO 360
  350 CONTINUE
      GO TO 370
  360 DO 510 IDOFN = 1,NDOFN
      NGASH = NLOCA+IDOFN
  510 TREAC(KBOUN,IDOFN) = TREAC(KBOUN,IDOFN)+FIXED(NGASH)
      KBOUN = KBOUN+1
  370 CONTINUE
C
C*** ADD REACTIONS INTO THE TOTAL LOAD ARRAY
C
      DO 700 IPOIN = 1,NPOIN
      DO 710 IELEM = 1,NELEM
      DO 710 INODE = 1,NNODE
      NLOCA = IABS(LNODS(IELEM,INODE))
      IF(IPOIN.EQ.NLOCA) GO TO 720
  710 CONTINUE
  720 DO 730 IDOFN = 1,NDOFN
      NGASH = (INODE-1)*NDOFN+IDOFN
      MGASH = (IPOIN-1)*NDOFN+IDOFN
  730 TLOAD(IELEM,NGASH) = TLOAD(IELEM,NGASH)+FIXED(MGASH)
  700 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE RESIDU(ASDIS,COORD,EFFST,ELOAD,FACTO,IITER,LNODS,
     *                  LPROP,MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,NDOFN,
     *                  NELEM,NEVAB,NGAUS,NNODE,NSTR1,NTYPE,POSGP,PROPS,
     *                  NSTRE,NCRIT,STRSG,WEIGP,TDISP,EPSTN)
C***********************************************************************
C
C**** THIS SUBROUTINE REDUCES THE STRESSES TO THE YIELD SURFACE AND
C     EVALUATES THE EQUIVALENT NODAL FORCES
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ASDIS(MTOTV),AVECT(4),CARTD(2,9),COORD(MPOIN,2),
     *          DEVIA(4),DVECT(4),EFFST(MTOTG),ELCOD(2,9),ELDIS(2,9),
     *          ELOAD(MELEM,18),LNODS(MELEM,9),POSGP(4),PROPS(MMATS,7),
     *          STRAN(4),STRES(4),STRSG(4,MTOTG),
     *          WEIGP(4),DLCOD(2,9),DESIG(4),SIGMA(4),SGTOT(4),
     *          DMATX(4,4),DERIV(2,9),SHAPE(9),GPCOD(2,9),
     *          EPSTN(MTOTG),TDISP(MTOTV),MATNO(MELEM),BMATX(4,18)
      ROOT3 =1.73205080757
      TWOPI = 6.283185308
      DO 10 IELEM = 1,NELEM
      DO 10 IEVAB = 1,NEVAB
   10 ELOAD(IELEM,IEVAB) = 0.0
      KGAUS = 0
      DO 20 IELEM = 1,NELEM
      LPROP = MATNO(IELEM)
      UNIAX = PROPS(LPROP,5)
      HARDS = PROPS(LPROP,6)
      FRICT = PROPS(LPROP,7)
      IF (NCRIT.EQ.3) UNIAX =PROPS(LPROP,5)*DCOS(FRICT*0.017453292)
      IF (NCRIT.EQ.4) UNIAX =6.0*PROPS(LPROP,5)*DCOS(FRICT*0.017453292)/
     * (ROOT3*(3.0-DSIN(FRICT*0.017453292)))
C
C*** COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      DO 30 INODE =1,NNODE
      LNODE = IABS(LNODS(IELEM,INODE))
      NPOSN = (LNODE-1)*NDOFN
      DO 30 IDOFN = 1,NDOFN
      NPOSN = NPOSN+1
      ELCOD(IDOFN,INODE)=COORD(LNODE,IDOFN)
   30 ELDIS(IDOFN,INODE)=ASDIS(NPOSN)
      CALL MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
      THICK = PROPS(LPROP,3)
      KGASP = 0
      DO 40 IGAUS = 1,NGAUS
      DO 40 JGAUS = 1,NGAUS
      EXISP = POSGP(IGAUS)
      ETASP = POSGP(JGAUS)
      KGAUS = KGAUS +1

      KGASP = KGASP+1
      CALL SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
```

```
                NNODE,SHAPE)
      DVOLU = DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF (NTYPE.EQ.3) DVOLU = DVOLU*TWOPI*GPCOD(1,KGASP)
      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
      CALL BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      CALL LINEAR (CARTD,DMATX,ELDIS,LPROP,MMATS,NDOFN,NNODE,NSTRE,
     *             NTYPE,PROPS,STRAN,STRES,KGASP,GPCOD,SHAPE)
      PREYS = UNIAX+EPSTN(KGAUS)*HARDS
      DO 150 ISTR1 = 1,NSTR1
      DESIG(ISTR1) = STRES(ISTR1)
  150 SIGMA(ISTR1) = STRSG(ISTR1,KGAUS)+STRES(ISTR1)
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SIGMA,
     .           THETA,VARJ2,YIELD)
      ESPRE = EFFST(KGAUS)-PREYS
      IF (ESPRE.GE.0.0) GO TO 50
      ESCUR = YIELD-PREYS
      IF (ESCUR.LE.0.0) GO TO 60
      RFACT = ESCUR/(YIELD-EFFST(KGAUS))
      GO TO 70
   50 ESCUR = YIELD-EFFST(KGAUS)
      IF (ESCUR.LE.0.0) GO TO 60
      RFACT =1.0
   70 MSTEP = ESCUR*8.0/UNIAX+1.0
      ASTEP = MSTEP
      REDUC = 1.0-RFACT
      DO 80 ISTR1 = 1,NSTR1
      SGTOT(ISTR1) = STRSG(ISTR1,KGAUS)+REDUC*STRES(ISTR1)
   80 STRES(ISTR1) = RFACT*STRES(ISTR1)/ASTEP
      DO 90 ISTEP = 1,MSTEP
      CALL INVAR (DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SGTOT,
     *            THETA,VARJ2,YIELD)
      CALL YIELDF (AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .            PROPS,SINT3,STEFF,THETA,VARJ2)
      CALL FLOWPL (AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS)
      AGASH = 0.0
      DO 100 ISTR1 = 1,NSTR1
  100 AGASH = AGASH+AVECT(ISTR1)*STRES(ISTR1)
      DLAMD = AGASH*ABETA
      IF (DLAMD.LT.0.0) DLAMD = 0.0
      BGASH = 0.0
      DO 110 ISTR1 = 1,NSTR1
      BGASH = BGASH+AVECT(ISTR1)*SGTOT(ISTR1)
  110 SGTOT(ISTR1) = SGTOT(ISTR1)+STRES(ISTR1)-DLAMD*DVECT(ISTR1)
      EPSTN(KGAUS) = EPSTN(KGAUS)+DLAMD*BGASH/YIELD
   90 CONTINUE
      CALL INVAR (DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SGTOT,
     .            THETA,VARJ2,YIELD)
      CURYS = UNIAX+EPSTN(KGAUS)*HARDS
      BRING = 1.0
      IF (YIELD.GT.CURYS) BRING = CURYS/YIELD
      DO 130 ISTR1 = 1,NSTR1
  130 STRSG(ISTR1,KGAUS) = BRING*SGTOT(ISTR1)
      EFFST(KGAUS) = BRING*YIELD
C**** ALTERNATIVE LOCATION OF STRESS REDUCTION LOOP TERMINATION CARD
C  90 CONTINUE
C***
      GO TO 190
   60 DO 180 ISTR1 = 1,NSTR1
  180 STRSG(ISTR1,KGAUS) = STRSG(ISTR1,KGAUS)+DESIG(ISTR1)
      EFFST(KGAUS) = YIELD
C
C*** CALCULATE THE EQUIVALENT NODAL FORCES AND ASSOCIATE WITH THE
C    ELEMENT NODES
  190 MGASH = 0
      DO 140 INODE = 1,NNODE
      DO 140 IDOFN = 1,NDOFN
      MGASH = MGASH+1
      DO 140 ISTRE = 1,NSTRE
  140 ELOAD(IELEM,MGASH) = ELOAD(IELEM,MGASH)+BMATX(ISTRE,MGASH)
     *    *STRSG(ISTRE,KGAUS)*DVOLU
   40 CONTINUE
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CONVER(ELOAD,IITER,LNODS,MELEM,MEVAB,MTOTV,NCHEK,
     .                  NDOFN,NELEM,NEVAB,NNODE,NTOTV,PVALU,STFOR,
     .                  TLOAD,TOFOR,TOLER)
C*********************************************************************
C
C*** THIS SUBROUTINE CHECKS FOR CONVERGENCE OF THE ITERATION PROCESS
C
C*********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ELOAD(MELEM,MEVAB),LNODS(MELEM,9),STFOR(MTOTV),
     .          TOFOR(MTOTV),TLOAD(MELEM,MEVAB)
      NCHEK = 0
      RESID = 0.0
      RETOT = 0.0
      REMAX = 0.0
      DO 5 ITOTV = 1,NTOTV
      STFOR(ITOTV) = 0.0
      TOFOR(ITOTV) = 0.0
    5 CONTINUE
      DO 40 IELEM = 1,NELEM
      KEVAB = 0
      DO 40 INODE = 1,NNODE
      LOCNO = IABS(LNODS(IELEM,INODE))
      DO 40 IDOFN = 1,NDOFN
      KEVAB = KEVAB+1
      NPOSI = (LOCNO-1)*NDOFN+IDOFN
      STFOR(NPOSI) = STFOR(NPOSI)+ELOAD(IELEM,KEVAB)
   40 TOFOR(NPOSI) = TOFOR(NPOSI)+TLOAD(IELEM,KEVAB)
      DO 50 ITOTV = 1,NTOTV
      REFOR = TOFOR(ITOTV)-STFOR(ITOTV)
      RESID = RESID+REFOR*REFOR
      RETOT =RETOT+TOFOR(ITOTV)*TOFOR(ITOTV)
      AGASH =DABS(REFOR)
   50 IF (AGASH.GT.REMAX) REMAX = AGASH
      DO 10 IELEM = 1,NELEM
      DO 10 IEVAB = 1,NEVAB
   10 ELOAD(IELEM,IEVAB) = TLOAD(IELEM,IEVAB)-ELOAD(IELEM,IEVAB)
      RESID =DSQRT(RESID)
      RETOT =DSQRT(RETOT)
      RATIO = 100.0*RESID/RETOT
      IF (RATIO.GT.TOLER) NCHEK = 1
      IF (IITER.EQ.1) GO TO 20
      IF (RATIO.GT.PVALU) NCHEK = 999
   20 PVALU = RATIO
      WRITE(6,30) IITER, NCHEK,RATIO,REMAX
   30 FORMAT(1H0,3X,14HITERATION NO =,I3,
     .       3X,18HCONVERGENCE CODE =,I4,3X,28HNORM OF RESIDUAL SUM
     .RATIO =,E14.6,3X,18HMAXIMUM RESIDUAL =,E14.6)
      RETURN
      END
```

```
      SUBROUTINE OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,
     .                  NOUTP,NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,
     .                  NTYPE,NCHEK,EFFST)
C*********************************************************************
C
C**** THIS SUBROUTINE OUTPUTS DISPLACEMENTS,REACTIONS AND STRESSES
C
C*********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION NOFIX(MVFIX),NOUTP(2),STRSG(4,MTOTG),STRSP(3),
     .          TDISP(MTOTV),TREAC(MVFIX,2),EPSTN(MTOTG),EFFST(MTOTG)
      KOUTP = NOUTP(1)
      IF (IITER.GT.1) KOUTP = NOUTP(2)
      IF (IITER.EQ.1.AND.NCHEK.EQ.0) KOUTP = NOUTP(2)
      WRITE (6,15) IITER
   15 FORMAT (1H0,3X,14HITERATION NO =,I3)
C
C*** OUTPUT DISPLACEMENTS
C
      IF (KOUTP.LT.1) GO TO 10
      WRITE (6,900)
  900 FORMAT(1H0,5X,13HDISPLACEMENTS)
      IF (NTYPE.NE.3) WRITE (6,950)
  950 FORMAT (1H0,6X,4HNODE,6X,7HX-DISP.,7X,7HY-DISP.)
      IF (NTYPE.EQ.3) WRITE (6,955)
  955 FORMAT (1H0,6X,4HNODE,6X,7HR-DISP.,7X,7HZ-DISP.)
      DO 20 IPOIN = 1,NPOIN
      NGASH = IPOIN*2
      NGISH = NGASH-2+1
   20 WRITE(6,910) IPOIN,(TDISP(IGASH),IGASH=NGISH,NGASH)
  910 FORMAT(I10,3E14.6)
   10 CONTINUE
C
C**** OUTPUT REACTIONS
C
      IF (KOUTP.LT.2) GO TO 30
      WRITE(6,920)
  920 FORMAT(1H0,5X,9HREACTIONS)
      IF (NTYPE.NE.3) WRITE (6,960)
  960 FORMAT (1H0,6X,4HNODE,6X,7HX-REAC.,7X,7HY-REAC.)
      IF (NTYPE.EQ.3) WRITE (6,965)
  965 FORMAT (1H0,6X,4HNODE,6X,7HR-REAC., 7X,7HZ-REAC.)
      DO 40 IVFIX = 1,NVFIX
   40 WRITE (6,910) NOFIX(IVFIX),(TREAC(IVFIX,IDOFN),IDOFN=1,2)
   30 CONTINUE
C
C*** OUTPUT STRESSES
C
      IF (KOUTP.LT.3) GO TO 50
      IF(NTYPE.NE.3) WRITE(6,970)
  970 FORMAT(1H0,1X,4HG.P.,6X,9HXX-STRESS,5X,9HYY-STRESS,5X,
     * 9HXY-STRESS,5X,9HZZ-STRESS,6X,8HMAX P.S.,6X,8HMIN P.S.,3X,
     * 5HANGLE,4X,10HEFF.STRESS,3X,6HE.P.S.)
      IF (NTYPE.EQ.3) WRITE(6,975)
  975 FORMAT(1H0,1X,4HG.P.,6X,9HRR-STRESS,5X,9HZZ-STRESS,5X,

     *9HRZ-STRESS,5X,9HTT-STRESS,6X,8HMAX P.S.,6X,8HMIN P.S.,3X,
     * 5HANGLE,4X,10HEFF.STRESS,3X,6HE.P.S.)
      KGAUS = 0
      DO 60 IELEM = 1,NELEM
      KELGS = 0
      WRITE(6,930) IELEM
  930 FORMAT(1H0,5X,13HELEMENT NO. =,I5)
      DO 60 IGAUS = 1,NGAUS
      DO 60 JGAUS = 1,NGAUS
      KGAUS = KGAUS+1
      KELGS = KELGS+1
      XGASH = (STRSG(1,KGAUS)+STRSG(2,KGAUS))*0.5
      XGISH = (STRSG(1,KGAUS)-STRSG(2,KGAUS))*0.5
      XGESH = STRSG(3,KGAUS)
      XGOSH =DSQRT(XGISH*XGISH+XGESH*XGESH)
      STRSP(1)=XGASH+XGOSH
      STRSP(2)=XGASH-XGOSH
      IF (XGISH.EQ.0.0) XGISH =0.1E-20
      STRSP(3) =DATAN(XGESH/XGISH)*28.647889757
   60 WRITE (6,940) KELGS,(STRSG(ISTR1,KGAUS),ISTR1 =1,4)
     *,(STRSP(ISTRE),ISTRE=1,3),EFFST(KGAUS),EPSTN(KGAUS)
  940 FORMAT(I5,2X,6E14.6,F8.3,2E14.6)
   50 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
C**********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE SHAPE FUNCTIONS AND THEIR
C     DERIVATIVES FOR LINEAR,QUADRATIC LAGRANGIAN AND SERENDIPITY
C     ISOPARAMETRIC 2-D ELEMENTS
C
C**********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DERIV(2,9),SHAPE(9)
      S = EXISP
      T = ETASP
      IF (NNODE.GT.4) GO TO 10
      ST = S*T
C
C*** SHAPE FUNCTIONS FOR 4 NODED ELEMENT
C
      SHAPE(1) = (1-T-S+ST)*0.25
      SHAPE(2) = (1-T+S-ST)*0.25
      SHAPE(3) = (1+T+S+ST)*0.25
      SHAPE(4) = (1+T-S-ST)*0.25
C
C*** SHAPE FUNCTION DERIVATIVES
C
      DERIV(1,1)=(-1+T)*0.25
      DERIV(1,2)=(+1-T)*0.25
      DERIV(1,3)=(+1+T)*0.25
      DERIV(1,4)=(-1-T)*0.25
      DERIV(2,1)=(-1+S)*0.25
      DERIV(2,2)=(-1-S)*0.25
      DERIV(2,3)=(+1+S)*0.25
      DERIV(2,4)=(+1-S)*0.25
      RETURN
   10 IF (NNODE.GT.8) GO TO 30
      S2 = S*2.0
      T2 = T*2.0
      SS = S*S
      TT = T*T
      ST = S*T
      SST= S*S*T
      STT = S*T*T
      ST2 = S*T*2.0
C
C*** SHAPE FUNCTIONS FOR 8 NODED ELEMENT
C
      SHAPE(1)=(-1.0+ST+SS+TT-SST-STT)/4.0
      SHAPE(2)=(1.0-T-SS+SST)/2.0
      SHAPE(3)=(-1.0-ST+SS+TT-SST+STT)/4.0
      SHAPE(4)=(1.0+S-TT-STT)/2.0
      SHAPE(5)=(-1.0+ST+SS+TT+SST+STT)/4.0
      SHAPE(6)=(1.0+T-SS-SST)/2.0
      SHAPE(7)=(-1.0-ST+SS+TT+SST-STT)/4.0
      SHAPE(8)=(1.0-S-TT+STT)/2.0
C
C**** SHAPE FUNCTION DERIVATIVES
```

```
      DERIV(1,1)=(T+S2-ST2-TT)/4.0
      DERIV(1,2)=-S+ST
      DERIV(1,3)=(-T+S2-ST2+TT)/4.0
      DERIV(1,4)=(1.0-TT)/2.0
      DERIV(1,5)=(T+S2+ST2+TT)/4.0
      DERIV(1,6)=(-S-ST)
      DERIV(1,7) = (-T+S2+ST2-TT)/4.0
      DERIV(1,8) = (-1.0+TT)/2.0
      DERIV(2,1) = (S+T2-SS-ST2)/4.0
      DERIV(2,2) = (-1.0+SS)/2.0
      DERIV(2,3) = (-S+T2-SS+ST2)/4.0
      DERIV(2,4) = (-T-ST)
      DERIV(2,5) = (S+T2+SS+ST2)/4.0
      DERIV(2,6) = (1.0-SS)/2.0
      DERIV(2,7) = (-S+T2+SS-ST2)/4.0
      DERIV(2,8) = (-T+ST)
      RETURN
   30 CONTINUE
      SS = S*S
      ST = S*T
      TT = T*T
      S1 = S+1.0
      T1 = T+1.0
      S2 = S*2.0
      T2 = T*2.0
      S9 = S-1.0
      T9 = T-1.0
C
C*** SHAPE FUNCTIONS FOR 9 NODED ELEMENT
C
      SHAPE(1) = 0.25*S9*ST*T9
      SHAPE(2) = 0.5*(1.0-SS)*T*T9
      SHAPE(3) = 0.25*S1*ST*T9
      SHAPE(4) =0.5*S*S1*(1.0-TT)
      SHAPE(5) = 0.25*S1*ST*T1
      SHAPE(6) = 0.5*(1.0-SS)*T*T1
      SHAPE(7) = 0.25*S9*ST*T1
      SHAPE(8) = 0.5*S*S9*(1.0-TT)
      SHAPE(9) = (1.0-SS)*(1.0-TT)
C
C*** SHAPE FUNCTION DERIVATIVES
C
      DERIV(1,1) = 0.25*T*T9*(-1.0+S2)
      DERIV(1,2) = -ST*T9
      DERIV(1,3) = 0.25*(1.0+S2)*T*T9
      DERIV(1,4) =0.5*(1.0+S2)*(1.0-TT)
      DERIV(1,5) = 0.25*(1.0+S2)*T*T1
      DERIV(1,6) =-ST*T1
      DERIV(1,7) =0.25*(-1.0+S2)*T*T1
      DERIV(1,8) =0.5*(-1.0+S2)*(1.0-TT)
      DERIV(1,9) =-S2*(1.0-TT)
      DERIV(2,1) =0.25*(-1.0+T2)*S*S9
      DERIV(2,2) =0.5*(1.0-SS)*(-1.0+T2)
      DERIV(2,3) =0.25*S*S1*(-1.0+T2)

      DERIV(2,4) =-ST*S1
      DERIV(2,5) =0.25*S*S1*(1.0+T2)
      DERIV(2,6) =0.5*(1.0-SS)*(1.0+T2)
      DERIV(2,7) =0.25*S*S9*(1.0+T2)
      DERIV(2,8) =-ST*S9
      DERIV(2,9) =-T2*(1.0-SS)
      RETURN
      END
```

```
      SUBROUTINE JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     *                  NNODE,SHAPE)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE JACOBIAN MATRIX AND THE CARTESIAN
C    SHAPE FUNCTION DERIVATIVES
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION CARTD(2,9),DERIV(2,9),ELCOD(2,9),GPCOD(2,9),SHAPE(9),
     *          XJACI(2,2),XJACM(2,2)
C
C*** CALCULATE THE COORDINATES OF THE SAPMPLING POINT
C
      DO 2 IDIME = 1,2
      GPCOD(IDIME,KGASP) = 0.0
      DO 2 INODE = 1,NNODE
      GPCOD(IDIME,KGASP)=GPCOD(IDIME,KGASP)+ELCOD(IDIME,INODE)
     **SHAPE(INODE)
    2 CONTINUE
C
C*** CREATE JACOBIAN MATRIX XJACM
C
      DO 4 IDIME =1,2
      DO 4 JDIME =1,2
      XJACM(IDIME,JDIME)=0.0
      DO 4 INODE = 1,NNODE
      XJACM(IDIME,JDIME)=XJACM(IDIME,JDIME)+DERIV(IDIME,INODE)*
     *ELCOD(JDIME,INODE)
    4 CONTINUE
C
C*** CALCULATE DETERMINANT AND INVERSE OF JACOBIAN MATRIX
C
      DJACB = XJACM(1,1)*XJACM(2,2)-XJACM(1,2)*XJACM(2,1)
      IF(DJACB) 6,6,8
    6 WRITE(6,600) IELEM
      STOP
    8 CONTINUE
      XJACI(1,1)=XJACM(2,2)/DJACB
      XJACI(2,2)=XJACM(1,1)/DJACB
      XJACI(1,2)=-XJACM(1,2)/DJACB
      XJACI(2,1)=-XJACM(2,1)/DJACB
C
C*** CALCULATE CARTESIAN DERIVATIVES
C
      DO 10 IDIME =1,2
      DO 10 INODE =1,NNODE
      CARTD(IDIME,INODE)=0.0
      DO 10 JDIME =1,2
      CARTD(IDIME,INODE)=CARTD(IDIME,INODE)+XJACI(IDIME,JDIME)*
     *DERIV(JDIME,INODE)
   10 CONTINUE
  600 FORMAT(//,36H PROGRAM HALTED IN SUBROUTINE JACOB2,/,11X,
     *22H ZERO OR NEGATIVE AREA,/,10X,16H ELEMENT NUMBER,I5)
      RETURN
      END
```

```
      SUBROUTINE MODPS(DMATX,LPROP,MMATS,NTYPE,PROPS)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE D-MATRIX
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DMATX(4,4),PROPS(MMATS,7)
      YOUNG = PROPS(LPROP,1)
      POISS = PROPS(LPROP,2)
      DO 10 ISTR1 = 1,4
      DO 10 JSTR1 = 1,4
   10 DMATX(ISTR1,JSTR1) = 0.0
      IF (NTYPE.NE.1) GO TO 4
C
C*** D MATRIX FOR PLANE STRESS CASE
C
      CONST = YOUNG / (1.0-POISS*POISS)
      DMATX(1,1) = CONST
      DMATX(2,2) = CONST
      DMATX(1,2) = CONST*POISS
      DMATX(2,1) = CONST*POISS
      DMATX(3,3) = (1.0-POISS)*CONST/2.0
      RETURN
    4 IF (NTYPE.NE.2) GO TO 6
C
C*** D MATRIX FOR PLANE STRAIN CASE
C
      CONST = YOUNG*(1.0-POISS)/((1.0+POISS)*(1.0-2.0*POISS))
      DMATX(1,1) = CONST
      DMATX(2,2) = CONST
      DMATX(1,2) = CONST*POISS/(1.0-POISS)
      DMATX(2,1) = CONST*POISS/(1.0-POISS)
      DMATX(3,3) = (1.0-2.0*POISS)*CONST/(2.0*(1.0-POISS))
      RETURN
    6 IF (NTYPE.NE.3) GO TO 8
C
C*** D MATRIX FOR AXISYMMETRIC CASE
C
      CONST = YOUNG * (1.0-POISS)/((1.0+POISS)*(1.0-2.0*POISS))
      CONSS = POISS/(1.0-POISS)
      DMATX(1,1) = CONST
      DMATX(2,2) = CONST
      DMATX(3,3) = CONST*(1.0-2.0*POISS)/(2.0*(1.0-POISS))
      DMATX(1,2) = CONST*CONSS
      DMATX(1,4) = CONST*CONSS
      DMATX(2,1) = CONST*CONSS
      DMATX(2,4) = CONST*CONSS
      DMATX(4,1) = CONST*CONSS
      DMATX(4,2) = CONST*CONSS
      DMATX(4,4) = CONST
    8 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE STRAIN-DISPLACEMENT MATRIX
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION BMATX(4,18),CARTD(2,9),SHAPE(9),GPCOD(2,9)
      NGASH = 0
      DO 10 INODE = 1,NNODE
      MGASH = NGASH+1
      NGASH = MGASH+1
      BMATX(1,MGASH)= CARTD(1,INODE)
      BMATX(1,NGASH) = 0.0
      BMATX(2,MGASH) = 0.0
      BMATX(2,NGASH) = CARTD(2,INODE)
      BMATX(3,MGASH) = CARTD(2,INODE)
      BMATX(3,NGASH) = CARTD(1,INODE)
      IF (NTYPE.NE.3) GO TO 10
      BMATX(4,MGASH) = SHAPE(INODE)/GPCOD(1,KGASP)
      BMATX(4,NGASH) = 0.0
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STEMP,
     .                 THETA,VARJ2,YIELD)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE STRESS INVARIANTS AND THE CURRENT
C    VALUE OF THE YIELD FUNCTION
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DEVIA(4),PROPS(MMATS,7),STEMP(4)
      ROOT3 = 1.73205080757
      SMEAN = (STEMP(1)+STEMP(2)+STEMP(4))/3.0
      DEVIA(1) = STEMP(1)-SMEAN
      DEVIA(2) = STEMP(2)-SMEAN
      DEVIA(3) = STEMP(3)
      DEVIA(4) = STEMP(4)-SMEAN
      VARJ2 = DEVIA(3)*DEVIA(3)+0.5*(DEVIA(1)*DEVIA(1)+DEVIA(2)*
     .        DEVIA(2)+DEVIA(4)*DEVIA(4))
      VARJ3 = DEVIA(4)*DEVIA(4)*DEVIA(4)-VARJ2
      STEFF =DSQRT(VARJ2)
      IF (STEFF.EQ.0.0) GO TO 10
      SINT3 = -3.0*ROOT3*VARJ3/(2.0*VARJ2*STEFF)
      IF (SINT3.GT.1.0) SINT3 = 1.0
      GO TO 20
   10 SINT3 = 0.0
   20 CONTINUE
      IF (SINT3.LT.-1.0) SINT3 = -1.0
      IF (SINT3.GT.1.0) SINT3 = 1.0
      THETA = DARSIN(SINT3)/3.0
      GO TO (1,2,3,4),NCRIT
C*** TRESCA
    1 YIELD = 2.0*DCOS(THETA)*STEFF
      RETURN
C*** VON MISES
    2 YIELD = ROOT3*STEFF
      RETURN
C*** MOHR-COULOMB
    3 PHIRA = PROPS(LPROP,7)*0.017453292
      SNPHI =DSIN(PHIRA)
      YIELD = SMEAN*SNPHI+STEFF*(DCOS(THETA)-DSIN(THETA)*SNPHI/ROOT3)
      RETURN
C*** DRUCKER-PRAGER
    4 PHIRA = PROPS(LPROP,7)*0.017453292
      SNPHI =DSIN(PHIRA)
      YIELD = 6.0*SMEAN*SNPHI/(ROOT3*(3.0-SNPHI))+STEFF
      RETURN
      END
```

```
      SUBROUTINE YIELDF(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .                  PROPS,SINT3,STEFF,THETA,VARJ2)
C*********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE FLOW VECTOR
C
C*********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AVECT(4),DEVIA(4),PROPS(MMATS,7),
     *          VECA1(4),VECA2(4),VECA3(4)
      IF (STEFF.EQ.0.0) RETURN
      FRICT = PROPS(LPROP,7)
      ABTHE = DABS(THETA * 57.29577951308)
      IF (ABTHE.LT.29.0) GO TO 15
      IF (ABTHE.GE.29.0) GO TO 25
   15 TANTH =DTAN(THETA)
      TANT3 =DTAN(3.0*THETA)
      SINTH =DSIN(THETA)
      COSTH =DCOS(THETA)
      COST3 =DCOS(3.0*THETA)
   25 ROOT3 = 1.73205080757
C
C*** CALCULATE VECTOR A1
C
      VECA1(1) = 1.0
      VECA1(2) = 1.0
      VECA1(3) = 0.0
      VECA1(4) = 1.0
C
C*** CALCULATE VECTOR A2
C
      DO 10 ISTR1 = 1,NSTR1
   10 VECA2(ISTR1) = DEVIA(ISTR1)/(2.0*STEFF)
      VECA2(3) = DEVIA(3)/STEFF
C
C*** CALCULATE VECTOR A3
C
      VECA3(1) = DEVIA(2)*DEVIA(4)+VARJ2/3.0
      VECA3(2) = DEVIA(1)*DEVIA(4)+VARJ2/3.0
      VECA3(3) = -2.0*DEVIA(3)*DEVIA(4)
      VECA3(4) = DEVIA(1)*DEVIA(2)-DEVIA(3)*DEVIA(3)+VARJ2/3.0
      GO TO (1,2,3,4),NCRIT
C
C**** TRESCA
C
    1 CONS1 = 0.0
      ABTHE =DABS(THETA*57.29577951308)
      IF (ABTHE.LT.29.0) GO TO 20
      CONS2 = ROOT3
      CONS3 = 0.0
      GO TO 40
   20 CONS2 = 2.0*(COSTH+SINTH*TANT3)
      CONS3 = ROOT3*SINTH/(VARJ2*COST3)
      GO TO 40
C


C*** VON MISES
C
    2 CONS1 =0.0
      CONS2 = ROOT3
      CONS3 = 0.0
      GO TO 40
C
C*** MOHR-COULOMB
C
    3 CONS1 =DSIN(FRICT*0.017453292)/3.0
      ABTHE =DABS(THETA*57.29577951308)
      IF (ABTHE.LT.29.0) GO TO 30
      CONS3 = 0.0
      PLUMI = 1.0
      IF (THETA.GT.0.0) PLUMI = -1.0
      CONS2 = 0.5*(ROOT3+PLUMI*CONS1*ROOT3)
         GO TO 40
   30 CONS2 = COSTH*((1.0+TANTH*TANT3)+CONS1*(TANT3-TANTH)*ROOT3)
      CONS3 = (ROOT3*SINTH+3.0*CONS1*COSTH)/(2.0*VARJ2*COST3)
      GO TO 40
C
C*** DRUCKER-PRAGER
C
    4 SNPHI =DSIN(FRICT*0.017453292)
      CONS1 = 2.0*SNPHI/(ROOT3*(3.0-SNPHI))
      CONS2 = 1.0
      CONS3 = 0.0
   40 CONTINUE
      DO 50 ISTR1 = 1,NSTR1
   50 AVECT(ISTR1) = CONS1*VECA1(ISTR1)+CONS2*VECA2(ISTR1)+CONS3*
     *   VECA3(ISTR1)
      RETURN
      END
```

```
      SUBROUTINE FLOWPL(AVECT,ABETA,DVECT,NTYPE,PROPS,
     .                  LPROP,NSTR1,MMATS)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE PLASTIC D VECTOR
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AVECT(4),DVECT(4),PROPS(MMATS,7)
      YOUNG =PROPS(LPROP,1)
      POISS =PROPS(LPROP,2)
      HARDS =PROPS(LPROP,6)
      FMUL1 = YOUNG/(1.0+POISS)
      IF (NTYPE.EQ.1) GO TO 60
      FMUL2 = YOUNG*POISS*(AVECT(1)+AVECT(2)+AVECT(4))/((1.0+POISS)*
     *(1.0-2.0*POISS))
      DVECT(1) = FMUL1*AVECT(1)+FMUL2
      DVECT(2) = FMUL1*AVECT(2)+FMUL2
      DVECT(3) =0.5*AVECT(3)*YOUNG/(1.0+POISS)
      DVECT(4) =FMUL1*AVECT(4)+FMUL2
      GO TO 70
   60 FMUL3=YOUNG*POISS*(AVECT(1)+AVECT(2))/(1.0-POISS*POISS)
      DVECT(1) = FMUL1*AVECT(1)+FMUL3
      DVECT(2) = FMUL1*AVECT(2)+FMUL3
      DVECT(3) = 0.5*AVECT(3)*YOUNG/(1.0+POISS)
      DVECT(4) = FMUL1*AVECT(4)+FMUL3
   70 DENOM = HARDS
      DO 80 ISTR1 = 1,NSTR1
   80 DENOM = DENOM+AVECT(ISTR1)*DVECT(ISTR1)
      ABETA = 1.0/DENOM
      RETURN
      END


      SUBROUTINE DBE(BMATX,DBMAT,DMATX,MEVAB,NEVAB,NSTRE,NSTR1)
C***********************************************************************
C
C*** THIS SUBROUTINE MULTIPLIES THE D-MATRIX BY THE B-MATRIX
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION BMATX(NSTR1,MEVAB),DBMAT(NSTR1,MEVAB),
     .          DMATX(NSTR1,NSTR1)
      DO 2 ISTRE = 1,NSTRE
      DO 2 IEVAB = 1,NEVAB
      DBMAT(ISTRE,IEVAB) = 0.0
      DO 2 JSTRE = 1,NSTRE
      DBMAT(ISTRE,IEVAB)=DBMAT(ISTRE,IEVAB)+
     *DMATX(ISTRE,JSTRE)*BMATX(JSTRE,IEVAB)
    2 CONTINUE
      RETURN
      END


      SUBROUTINE LINEAR(CARTD,DMATX,ELDIS,LPROP,MMATS,NDOFN,NNODE,NSTRE,
     .                  NTYPE,PROPS,STRAN,STRES,KGASP,GPCOD,SHAPE)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES STRESSES AND STRAINS ASSUMING LINEAR
C    ELASTIC BEHAVIOUR
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AGASH(2,2),CARTD(2,9),DMATX(4,4),ELDIS(2,9),
     .          PROPS(MMATS,7),STRAN(4),STRES(4),
     .          GPCOD(2,9),SHAPE(9)
      POISS = PROPS(LPROP,2)
      DO 20 IDOFN = 1,NDOFN
      DO 20 JDOFN = 1,NDOFN
      BGASH = 0.0
      DO 10 INODE = 1,NNODE
   10 BGASH = BGASH+CARTD(JDOFN,INODE)*ELDIS(IDOFN,INODE)
   20 AGASH(IDOFN,JDOFN) = BGASH
C
C*** CALCULATE THE STRAINS
C
      STRAN(1) = AGASH(1,1)
      STRAN(2) = AGASH(2,2)
      STRAN(3) = AGASH(1,2)+AGASH(2,1)
      STRAN(4) = 0.0
      DO 30 INODE = 1,NNODE
   30 STRAN(4) = STRAN(4)+ELDIS(1,INODE)*SHAPE(INODE)/GPCOD(1,KGASP)
C
C*** AND THE CORRESPONDING STRESSES
C
      DO 40 ISTRE = 1,NSTRE
      STRES(ISTRE) = 0.0
      DO 40 JSTRE = 1,NSTRE
   40 STRES(ISTRE) = STRES(ISTRE)+DMATX(ISTRE,JSTRE)*STRAN(JSTRE)
      IF(NTYPE.EQ.1) STRES(4)=0.0
      IF (NTYPE.EQ.2) STRES(4)=POISS*(STRES(1)+STRES(2))
      RETURN
      END
```

APPENDIX D

Finite Element Program For Nonlinear Strain Hardening Materials

FINITE ELEMENT PROGRAM FOR NONLINEAR STRAIN HARDENING MATERIALS

```
C      MSTER2 PLAST
C**********************************************************************
C      PROGRAM FOR THE ELASTO-PLASTIC ANALYSIS OF PLANE STRESS,
C      PLANE STRAIN AND AXISYMMETRIC SOLIDS
C**********************************************************************
       IMPLICIT REAL*8(A-H,O-Z)
       DIMENSION ASDIS(400),COORD(180,2),ELOAD(50,18),ESTIF(18,18),
      .          EQRHS(10),EQUAT(100,10),FIXED(360),GLOAD(80),GSTIF(3240),
      .           IFFIX(360),LNODS(50,9),LOCEL(18),MATNO(50),
      .           NACVA(80),NAMEV(10),NDEST(18),NOFRO(50),NOFIX(50),
      .           NOUTP(2), NPIVO(10),
      .           POSGP(4),PRESC(30,2),PROPS(5,6),RLOAD(50,18),
      .           STFOR(360),TREAC(50,2),VECRV(80),WEIGP(4),
      .           STRSG(4,360),TDISP(360),TLOAD(50,18),
      .           TOFOR(360),EPSTN(360),EFFST(360),
      .           USTRES(5,25),USTRN(5,25),YSTRES(5,25),
      .           PSTRN(5,25),HVALU(5,25)
C
C*** PRESET VARIABLES ASSOCIATED WITH DYNAMIC DIMENSIONING
C
       CALL DIMEN ( MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,MSTIF,MTOTG,
      .      MTOTV,MVFIX,NDOFN,NPROP,NSTRE)
C
C*** CALL THE SUBROUTINE WHICH READS MOST OF THE PROBLEM DATA
C
       CALL INPUT2 (   COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,MFRON,
      .     MMATS,MPOIN,MTOTV,MVFIX,NALGO,
      .            NCRIT,NOFRO,NDOFN,NE
      .            NINCS, NMATS,NNODE,NLEM,NEVAB,NGAUS,NGAU2,
      .            NSTR1,NTOTG,NTOTV,   OFIX,NPOIN,NPROP,NSTRE,
      .            NTYPE,NVFIX,POSGP,PRESC,PROPS,WEIGP,
      .            USTRES,USTRN,NUMB)
C
C*** CALL THE SUBROUTINE WHICH CALCULATES THE HARDNESS VALUE
C    ARRAYS FROM UNIAXIAL TEST DATA
C
       CALL HRDNS (USTRES,USTRN, NUMB,PROPS,MMATS,NMATS,
      .            HVALU,YSTRES,PSTRN)
C
C*** CALL THE SUBROUTINE WHICH COMPUTES THE CONSISTENT LOAD VECTORS
C    FOR EACH ELEMENT AFTER READING THE RELEVANT INPUT DATA
C
       CALL LOADP1(COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,NELEM,
      .            NEVAB,NGAUS,NNODE,NPOIN,NSTRE,NTYPE,POSGP,
      .            PROPS,RLOAD,WEIGP,NDOFN)
C
C*** INITIALIZE CERTAIN ARRAYS
C
       CALL ZERO(ELOAD,MELEM,MEVAB,MPOIN,MTOTG,MTOTV,NDOFN,NELEM,
      .           NEVAB,NGAUS,NSTR1,NTOTG,EPSTN,EFFST,
      .           NTOTV,NVFIX,STRSG,TDISP,TFACT,
      .           TLOAD,TREAC,MVFIX)
C
C*** LOOP OVER EACH INCREMENT
C
```

```
      DO 100 IINCS =1,NINCS
C
C***  READ DATA FOR CURRENT INCREMENT
C
      CALL INCREM(ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER,MTOTV,
     .            MVFIX,NDOFN,NELEM,NEVAB,NOUTP,NOFIX,NTOTV,
     .            NVFIX,PRESC,RLOAD,TFACT,TLOAD,TOLER)
C
C***  LOOP OVER EACH ITERATION
C
      DO 50 IITER = 1,MITER
C
C***  CALL ROUTINE WHICH SELECTS SOLUTION ALGORITHM VARIABLE KRESL
C
      CALL ALGOR(FIXED,IINCS,IITER,KRESL,MTOTV,NALGO,
     .           NTOTV)
C
C***  CHECK WHETHER A NEW EVALUATION OF THE STIFFNESS MATRIX IS REQUIRED
C
      IF (KRESL.EQ.1) CALL STIFP2(COORD,EPSTN,IINCS,LNODS,MATNO,
     .                MEVAB,MMATS,MPOIN,NTOTV,NELEM,NEVAB,NGAUS,NNODE,
     .                NSTRE,NSTR1,POSGP,PROPS,WEIGP,MELEM,MTOTG,
     .                STRSG,NTYPE,NCRIT,HVALU,PSTRN,NUMB,
     .                                     HARDS,PLSTN,NMATS)
C
C***  SOLVE EQUATIONS
C
      CALL FRONT(ASDIS,ELOAD,EQRHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS,IITER,
     .           GLOAD,GSTIF,LOCEL,LNODS,KRESL,MBUFA,MELEM,MEVAB,MFRON,
     .           MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,NDOFN,NELEM,NEVAB,
     .           NNODE,NOFIX,NPIVO,NPOIN,NTOTV,TDISP,TLOAD,TREAC,
     .           VECRV)
C
C***  CALCULATE RESIDUAL FORCES
C
       CALL RESID2(ASDIS,COORD,EFFST,ELOAD,FACTO,IITER,LNODS,
     .             LPROP,MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,NDOFN,
     .             NELEM,NEVAB,NGAUS,NNODE,NSTR1,NTYPE,POSGP,PROPS,
     .             NSTRE,NCRIT,STRSG,WEIGP,TDISP,EPSTN,
     .             HVALU,PSTRN,NUMB, HARDS, PLSTN, NMATS )
C
C***  CHECK FOR CONVERGENCE
C
      CALL CONVER(ELOAD,IITER,LNODS,MELEM,MEVAB,MTOTV,NCHEK,NDOFN,
     .            NELEM,NEVAB,NNODE,NTOTV,PVALU,STFOR,TLOAD,TOFOR,TOLER)
C
C***  OUTPUT RESULTS IF REQUIRED
C
      IF (IITER.EQ.1.AND.NOUTP(1).GT.0)
     . CALL OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
     .             NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,NTYPE,NCHEK)
C
C***  IF SOLUTION HAS CONVERGED STOP ITERATING AND OUTPUT RESULTS
C
      IF (NCHEK.EQ.0) GO TO 75


   50 CONTINUE
C
C***
C
      IF (NALGO.EQ.2) GO TO 75
      STOP
   75 CALL OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
     .            NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,NTYPE,NCHEK,
     .            EFFST)
  100 CONTINUE
      STOP
      END
```

```
      SUBROUTINE   DIMEN1 (   MBUFA,MELEM,MEVAB,MFRON,MMATS,MPOIN,MSTIF,
     .             MTOTG, MTOTV,MVFIX,NDOFN,NPROP,NSTRE)
C**********************************************************************
C
C**** THIS SUBROUTINE PRESETS VARIABLES ASSOCIATED WITH DYNAMIC
C     DIMENSIONING
C
C**********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      MBUFA =10
      MELEM =50
      MFRON =80
      MMATS =5
      MPOIN =180
      MSTIF =(MFRON*MFRON-MFRON)/2.0+MFRON
      MTOTG =MELEM*9
      NDOFN =2
      MTOTV =MPOIN*NDOFN
      MVFIX =30
      NPROP =6
      MEVAB =NDOFN*9
      RETURN
      END
```

```
      SUBROUTINE INPUT2 (    COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,MFRON,
     .              MMATS,MPOIN,MTOTV,MVFIX,NALGO,
     .                    NCRIT,NDFRO,NDOFN,NELEM,
     .                    NEVAB,NGAUS,NGAU2,
     .                    NINCS,NMATS,NNODE,NOFIX,NPOIN,NPROP,NSTRE,NSTR1,
     .                    NTOTG,NTOTV,NTYPE,NVFIX,POSGP,PRESC,PROPS,WEIGP,
     .                    USTRES,USTRN,NUMB)
C**********************************************************************
C
C*** THIS SUBROUTINE ACCEPTS HOST OF THE INPUT DATA
C
C**********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION COORD(MPOIN,2),IFFIX(MTOTV),LNODS(MELEM,9),
     .          MATNO(MELEM),NDFRO(MELEM),
     .          NOFIX(MVFIX),POSGP(4),PRESC(MVFIX,NDOFN),
     .          PROPS(MMATS,NPROP),TITLE(18),WEIGP(4),
     .          USTRES(MMATS,25),USTRN(MMATS,25)
      REWIND 1
      REWIND 2
      REWIND 3
      REWIND 4
      REWIND 8
      READ (5,920) TITLE
      WRITE (6,920) TITLE
  920 FORMAT(18A4)
C
C*** READ THE FIRST DATA CARD AND ECHO IT IMMEDIATELY
C
      READ(5,900) NPOIN,NELEM,NVFIX,NTYPE,NNODE,NMATS,NGAUS,
     .            NALGO,NCRIT,NINCS,NSTRE
  900 FORMAT(11I5)
      NEVAB = NDOFN*NNODE
      NSTR1 = NSTRE+1
      IF (NTYPE.EQ.3) NSTR1 = NSTRE
      NTOTV = NPOIN*NDOFN
      NGAU2 = NGAUS*NGAUS
      NTOTG = NELEM*NGAU2
      WRITE(6,901) NPOIN,NELEM,NVFIX,NTYPE,NNODE,NMATS,NGAUS,NEVAB,
     .            NALGO,NCRIT,NINCS,NSTRE
  901 FORMAT (//8H NPOIN =,I4,4X,8H NELEM =,I4,4X,8H NVFIX =,I4,4X,
     .8H NTYPE =,I4,4X,8H NNODE =,I4,//
     .8H NMATS =,I4,4X,8H NGAUS =,I4,
     .4X,8H NEVAB =,I4,4X,8H NALGO =,I4//
     .8H NCRIT =,I4,4X,8H NINCS =,I4,4X,8H NSTRE =,I4)
      CALL CHECK1(NDOFN,NELEM,NGAUS,NMATS,NNODE,NPOIN,
     .            NSTRE,NTYPE,NVFIX,NCRIT,NALGO,NINCS)
C
C*** READ THE ELEMENT NODAL CONNECTIONS,AND THE PROPERTY NUMBERS
C
      WRITE(6,902)
  902 FORMAT (//8H ELEMENT,3X,8HPROPERTY,6X,12HNODE NUMBERS)
      DO 3 IELEM = 1,NELEM
      READ(5,900) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE)
    2 WRITE(6,903) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE)
```

```
  903 FORMAT(1X,I5,I9,6X,8I5)
C
C*** ZERO ALL THE NODAL COORDINATES,PRIOR TO READING SOME OF THEM*
C
      DO 4 IPOIN = 1,NPOIN
      DO 4 IDIME = 1,2
    4 COORD(IPOIN,IDIME) = 0.0
C
C*** READ SOME NODAL COORDINATES,FINISHING WITH THE LAST NODE OF ALL.
C
      WRITE(6,904)
  904 FORMAT(//5H NODE,10X,1HX,10X,1HY)
    6 READ(5,905) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,2)
  905 FORMAT( I5,6F10.5)
      IF (IPOIN.NE.NPOIN) GO TO 6
C
C*** INTERPOLATE COORDINATES OF MID-SIDE NODES
C
      CALL NODEXY(COORD,LNODS,MELEM,MPOIN,NELEM,NNODE)
      DO 10 IPOIN =1,NPOIN
   10 WRITE(6,906) IPOIN,(COORD(IPOIN,IDIME),IDIME=1,2)
  906 FORMAT(1X,I5,3F10.3)
C
C*** READ THE FIXED VALUES
C
      WRITE(6,907)
  907 FORMAT(//5H NODE,6X,4HCODE,6X,12HFIXED VALUES)
      DO 8 IVFIX = 1,NVFIX
      READ(5,908) NOFIX(IVFIX),IFPRE,(PRESC(IVFIX,IDOFN),IDOFN=1,NDOFN)
      WRITE(6,908) NOFIX(IVFIX),IFPRE,(PRESC(IVFIX,IDOFN),IDOFN=1,NDOFN)
      NLOCA = (NOFIX(IVFIX)-1)*NDOFN
      IFDOF = 10**(NDOFN-1)
      DO 8 IDOFN = 1,NDOFN
      NGASH = NLOCA+IDOFN
      IF(IFPRE.LT.IFDOF) GO TO 8
      IFFIX(NGASH)=1
      IFPRE=IFPRE-IFDOF
    8 IFDOF = IFDOF/10
  908 FORMAT(1X,I4,5X,I5,5X,5F10.6)
C
C*** READ THE AVAILABLE SELECTION OF ELEMENT PROPERTIES
C
   16 WRITE(6,910)
  910 FORMAT(//7H NUMBER,6X,18HELEMENT PROPERTIES)
      DO 18 IMATS =1,NMATS
      READ(5,900) NUMAT
      READ(5,930) (PROPS(NUMAT,IPROP),IPROP=1,NPROP)
  930 FORMAT(6F10.5)
   18 WRITE(6,911) NUMAT,(PROPS(NUMAT,IPROP),IPROP=1,NPROP)
  911 FORMAT(1X,I4,3X,6E14.6)
C
C*** READ AND WRITE UNIAXIAL TEST DATA
C
      CALL UNIAX2 (USTRES,USTRN,NUMB,MMATS,NMATS)
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL GAUSSQ(NGAUS,POSGP,WEIGP)
      CALL CHECK2(COORD,IFFIX,LNODS,MATNO,MELEM,MFRON,MPOIN,MTOTV,
     .            MVFIX,NDFRO,NDOFN,NELEM,NMATS,NNODE,NOFIX,NPOIN,
     .            NVFIX)
      RETURN
      END
```

```
      SUBROUTINE UNIAX2 (USTRES,USTRN,NUMB,MMATS,NMATS)
C***********************************************************************
C
C     THIS SUBROUTINE READS UNIAXIAL TEST DATA FOR STRESSES AND
C     STRAINS AND PRINTS THEM
C
C***********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION USTRES(MMATS,25),USTRN(MMATS,25)
C
C*** INITIALIZE THE USTRES,USTRN ARRAYS
C
      DO 25 IMATS = 1,NMATS
      MCOL = 25
      DO 35 ICOL = 1,MCOL
      USTRES (IMATS,ICOL) = 0.0
      USTRN (IMATS,ICOL) = 0.0
   35 CONTINUE
   25 CONTINUE
C
C*** READ TEST DATA AND PRINT
C
      WRITE (6,200)
  200 FORMAT (1H0,28HTHE UNIAXIAL TEST VALUES ARE//
     *         5X,6HNUMBER,4X,1HI,5X,9HSTRESS(I),5X,9HSTRAIN(I)//)
      DO 10 IMATS = 1,NMATS
      READ (5,100) NUMAT
      READ (5,110) NUMB
  100 FORMAT (5X,I5)
  110 FORMAT (5X,I5)
      DO 15 INUMB = 1,NUMB
      READ (5,115) USTRES(NUMAT,INUMB),USTRN(NUMAT,INUMB)
  115 FORMAT (F10.4,6X,E14.8)
   15 CONTINUE
   10 CONTINUE
      DO 20 IMATS = 1,NMATS
      DO 30 I = 1,NUMB
      WRITE (6,130) IMATS,I,USTRES(IMATS,I),USTRN(IMATS,I)
  130 FORMAT (5X,I5,5X,I5,4X,F10.4,4X,E14.8)
   30 CONTINUE
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE HRDNS (USTRES,USTRN,NUMB,PROPS,MMATS,NMATS,
     '                  HVALU,YSTRES,PSTRN)
C***********************************************************************
C
C     THIS SUBROUTINE EVALUATES THE HARDNESS VALUES INTERPOLATED AS
C     THE SLOPE OF STRESS VS PLASTIC STRAIN CURVE FROM THE
C     UNIAXIAL TEST DATA
C
C***********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION USTRES(MMATS,25),USTRN(MMATS,25),PROPS(MMATS,6),
     .          YSTRES(MMATS,25),PSTRN(MMATS,25),HVALU(MMATS,25)
C
C***  READ UNIAXIAL PROPERTIES
C
      DO 10 IMATS = 1,NMATS
      YOUNG = PROPS (IMATS,1)
      YIELD = PROPS (IMATS,5)
C
C*** INITIALIZE THE HVALU,PSTRN,YSTRES ARRAYS
C
      MCOL = 25
      DO 50 ICOL = 1,MCOL
      HVALU (IMATS,ICOL) = 0.0
      PSTRN (IMATS,ICOL) = 0.0
      YSTRES(IMATS,ICOL) = 0.0
   50 CONTINUE
C
C***  CHANGE THE TEST DATA TO STRESS-PLASTIC STRAIN VALUES
C     IF NOT PERFECTLY PLASTIC
C
      KOUNT = 1
      DO 20 I = 1,NUMB
      IF (USTRES(IMATS,I).LT.YIELD) GO TO LO
      YSTRES(IMATS,KOUNT) = USTRES(IMATS,I)
      PSTRN(IMATS,KOUNT) = (USTRN(IMATS,I)-(USTRES(IMATS,I)/YOUNG))
      KOUNT = KOUNT +1
   20 CONTINUE
      KOUNT = KOUNT-1
C
C***  CALCULATE THE HVALU,THE HARDNESS VALUES FROM THE SLOPE OF
C     STRESS VS EFFECTIVE PLASTIC STRAIN CURVE
C
      HVALU (IMATS,1) = DABS((YSTRES(IMATS,1)-YIELD)/(PSTRN(IMATS,1)))
      K= KOUNT
      DO 30 J =2,KOUNT
      HVALU(IMATS,J) = DABS((YSTRES(IMATS,J)-YSTRES(IMATS,(J-1)))/
     .                 (PSTRN(IMATS,J)-PSTRN(IMATS,(J-1))))
   30 CONTINUE
   10 CONTINUE
C     FOR TESTING
      WRITE (6,100)
  100 FORMAT (1H0,2X,8HMATERIAL,4X,6HNUMBER,13X,5HARRAY,5X,
     .       20HPLASTIC STRAIN ARRAY)
      DO 35 IMATS = 1,NMATS

      DO 45 INUMB = 1,NUMB
      WRITE (6,115) IMATS,INUMB,HVALU(IMATS,INUMB),PSTRN(IMATS,INUMB)
  115 FORMAT (1H0,5X,I5 ,5X,I5,5X,E14.8,5X,E14.8)
   45 CONTINUE
   35 CONTINUE
C    TESTING OVER
      RETURN
      END
```

```
      SUBROUTINE LOADP1(COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,NELEM,
     .                  NEVAB,NGAUS,NNODE,NPOIN,NSTRE,NTYPE,POSGP,
     .                  PROPS,RLOAD,WEIGP,NDOFN)
C**********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE CONSISTENT NODAL FORCES FOR EACH
C    ELEMENT
C
C**********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION CARTD(2,9),COORD(MPOIN,2),DERIV(2,9),DGASH(2),
     .          DMATX(4,4),ELCOD(2,9),LNODS(MELEM,9),MATNO(MELEM),
     .          NOPRS(4),PGASH(2),POINT(2),POSGP(4),PRESS(4,2),
     .          PROPS(MMATS,6),RLOAD(MELEM,18),SHAPE(9),STRAN(4),
     .          STRES(4),TITLE(18),
     .          WEIGP(4),GPCOD(2,9)
      TWOPI = 6.283185308
      DO 10 IELEM =1,NELEM
      DO 10 IEVAB = 1,NEVAB
   10 RLOAD(IELEM,IEVAB) =0.0
      READ(5,901) TITLE
  901 FORMAT(18A4)
      WRITE(6,903) TITLE
  903 FORMAT(1H0,18A4)
C
C*** READ DATA CONTROLLING LOADING TYPES TO BE INPUTED
C
      READ(5,919) IPLOD,IGRAV,IEDGE
      WRITE(6,919) IPLOD,IGRAV,IEDGE
  919 FORMAT(3I5)
C
C*** READ NODAL POINT LOADS
C
      IF (IPLOD.EQ.0) GO TO 500
   20 READ (5,931) LODPT,(POINT(IDOFN),IDOFN=1,2)
      WRITE (6,931) LODPT,(POINT(IDOFN),IDOFN=1,2)
  931 FORMAT(I5,2F10.3)
C
C*** ASSOCIATE THE NODAL POINT LOADS WITH AN ELEMENT
C
      DO 30 IELEM = 1,NELEM
      DO 30 INODE = 1,NNODE
      NLOCA = IABS(LNODS(IELEM,INODE))
      IF (LODPT.EQ.NLOCA) GO TO 40
   30 CONTINUE
   40 DO 50 IDOFN =1,2
      NGASH = (INODE-1)*2+IDOFN
   50 RLOAD(IELEM,NGASH) = POINT(IDOFN)
      IF (LODPT.LT.NPOIN) GO TO 20
  500 CONTINUE
      IF (IGRAV.EQ.0) GO TO 600
C
C*** GRAVITY LOADING SECTION
C
C
```

```
C*** READ GRAVITY ANGLE AND GRAVITATIONAL CONSTANT
C
      READ (5,906) THETA,GRAVY
  906 FORMAT(2F10.3)
      WRITE (6,911) THETA,GRAVY
  911 FORMAT(1H0,16H GRAVITY ANGLE =,F10.3,19H GRAVITY CONSTANT =,F10.3)
      THETA = THETA/57.295779514
C
C*** LOOP OVER EACH ELEMENT
C
      DO 90 IELEM = 1,NELEM
C
C*** SET UP PRELIMINARY CONSTANTS
C
      LPROP = MATNO(IELEM)
      THICK = PROPS(LPROP,3)
      DENSE = PROPS(LPROP,4)
      IF (DENSE.EQ.0.0) GO TO 90
      GXCOM = DENSE*GRAVY*DSIN(THETA)
      GYCOM = DENSE*GRAVY*DCOS(THETA)
C
C*** COMPUTE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 60 INODE = 1,NNODE
      LNODE = IABS(LNODS(IELEM,INODE))
      DO 60 IDIME = 1,2
   60 ELCOD(IDIME,INODE) = COORD(LNODE,IDIME)
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      KGASP = 0
      DO 80 IGAUS = 1,NGAUS
      DO 80 JGAUS = 1,NGAUS
      EXISP = POSGP(IGAUS)
      ETASP = POSGP(JGAUS)
C
C*** COMPUTE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS AND ELEMENTAL
C    VOLUME
C
      CALL SFR2 (DERIV,ETASP,EXISP,NNODE,SHAPE)
      KGASP = KGASP+1
      CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .            NNODE,SHAPE )
      DVOLU = DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
      IF (NTYPE.EQ.3) DVOLU = DVOLU*TWOPI*GPCOD(1,KGASP)
C
C*** CALCULATE LOADS AND ASSOCIATE WITH ELEMENT NODAL POINTS
C
      DO 70 INODE = 1,NNODE
      NGASH = (INODE-1)*2+1
      MGASH = (INODE-1)*2+2
      RLOAD(IELEM,NGASH) = RLOAD(IELEM,NGASH)+GXCOM*SHAPE(INODE)*DVOLU
   70 RLOAD(IELEM,MGASH) = RLOAD(IELEM,MGASH)+GYCOM*SHAPE(INODE)*DVOLU
   80 CONTINUE
```

```
   90 CONTINUE
  600 CONTINUE
      IF (IEDGE.EQ.0) GO TO 700
C
C***  DISTRIBUTED EDGE LOADS SECTION
C
      READ (5,932) NEDGE
  932 FORMAT(I5)
      WRITE (6,912) NEDGE
  912 FORMAT (1H0,5X,21HNO. OF LOADED EDGES =,I5)
      WRITE (6,915)
  915 FORMAT (1H0,5X,38HLIST OF LOADED EDGES AND APPLIED LOADS)
      NODEG = 3
      NCODE = NNODE
      IF (NNODE.EQ.4) NODEG = 2
      IF (NNODE.EQ.9) NCODE = 8
C
C***  LOOP OVER EACH LOADED EDGE
C
      DO 160 IEDGE = 1,NEDGE
C
C***  READ DATA LOCATING THE LOADED EDGE AND APPLIED LOAD
C
      READ (5,902) NEASS,(NOPRS(IODEG),IODEG=1,NODEG)
  902 FORMAT(4I5)
      WRITE (6,913) NEASS,(NOPRS(IODEG),IODEG=1,NODEG)
  913 FORMAT (I10,5X,3I5)
      READ (5,914) ((PRESS(IODEG,IODFN),IODFN=1,2),IODEG=1,NODEG)
      WRITE (6,914) ((PRESS(IODEG,IODFN),IODFN =1,2),IODEG=1,NODEG)
  914 FORMAT(6F10.3)
      ETASP = -1.0
C
C***  CALCULATE THE COORDINATES OF THE NODES OF THE ELEMENT EDGE
C
      DO 100 IODEG = 1,NODEG
      LNODE = NOPRS(IODEG)
      DO 100 IDIME = 1,2
  100 ELCOD(IDIME,IODEG) = COORD(LNODE,IDIME)
C
C***  ENTER LOOPS FOR LINEAR NUMERICAL INTEGRATION
C
      DO 150 IGAUS = 1,NGAUS
      EXISP = POSGP(IGAUS)
C
C***  EVALUATE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS
C
      CALL SFR2 (DERIV,ETASP,EXISP,NNODE,SHAPE)
C
C***  CALCULATE COMPONENTS OF THE EQUIVALENT NODAL LOADS
C
      DO 110 IDOFN = 1,2
      PGASH (IDOFN)=0.0
      DGASH (IDOFN) =0.0
      DO 110 IODEG = 1,NODEG
      PGASH(IDOFN) = PGASH(IDOFN)+PRESS(IODEG,IDOFN)*SHAPE(IODEG)

  110 DGASH (IDOFN) = DGASH(IDOFN)+ELCOD(IDOFN,IODEG)*DERIV(1,IODEG)
      DVOLU = WEIGP(IGAUS)
      PXCOM = DGASH(1)*PGASH(2)-DGASH(2)*PGASH(1)
      PYCOM = DGASH(1)*PGASH(1)+DGASH(2)*PGASH(2)
      IF(NTYPE.NE.3) GO TO 115
      RADUS =0.0
      DO 125 IODEG =1,NODEG
  125 RADUS = RADUS +SHAPE(IODEG)*ELCOD(1,IODEG)
      DVOLU = DVOLU*TWOPI*RADUS
  115 CONTINUE
C
C***  ASSOCIATE THE EQUIVALENT NODAL EDGE LOADS WITH AN ELEMENT
C
      DO 120 INODE = 1,NNODE
      NLOCA = IABS(LNODS(NEASS,INODE))
      IF (NLOCA.EQ.NOPRS(1)) GO TO 130
  120 CONTINUE
  130 JNODE = INODE+NODEG-1
      KOUNT = 0
      DO 140 KNODE = INODE,JNODE
      KOUNT = KOUNT+1
      NGASH = (KNODE-1)*NDOFN+1
      MGASH = (KNODE-1)*NDOFN+2
      IF (KNODE.GT.NCODE) NGASH = 1
      IF (KNODE.GT.NCODE) MGASH = 2
      RLOAD(NEASS,NGASH) = RLOAD(NEASS,NGASH)+SHAPE(KOUNT)*PXCOM*DVOLU
  140 RLOAD(NEASS,MGASH)=RLOAD(NEASS,MGASH)+SHAPE(KOUNT)*PYCOM*DVOLU
  150 CONTINUE
  160 CONTINUE
  700 CONTINUE
      WRITE(6,907)
  907 FORMAT(1H0,5X,36H TOTAL NODAL FORCES FOR EACH ELEMENT)
      DO 290 IELEM = 1,NELEM
  290 WRITE (6,905) IELEM, (RLOAD(IELEM,IEVAB),IEVAB=1,NEVAB)
  905 FORMAT(1X,I4,5X,8E12.4/(10X,8E12.4))
      RETURN
      END
```

```
      SUBROUTINE STIFP2(COORD,EPSTN,IINCS,LNODS,MATNO,MEVAB,MMATS,
     *                  MPOIN,MTOTV,NELEM,NEVAB,NGAUS,NNODE,NSTRE,
     *                  NSTRI,POSGP,PROPS,WEIGP,MELEM,MTOTG,
     *                  STRSG,NTYPE,NCRIT, HVALU,
     *                  PSTRN,NUMB, HARDS,PLSTN,NMATS)
C*********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE STIFFNESS MATRIX FOR EACH ELEMENT
C    IN TURN
C
C*********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION BMATX(4,18),CARTD(2,9),COORD(MPOIN,2),DBMAT(4,18),
     *          DERIV(2,9),DEVIA(4),DMATX(4,4),
     *          ELCOD(2,9),EPSTN(MTOTG),ESTIF(18,18),LNODS(MELEM,9),
     *          MATNO(MELEM),POSGP(4),PROPS(MMATS,6),SHAPE(9),
     *          WEIGP(4),STRES(4),STRSG(4,MTOTG),
     *          DVECT(4),AVECT(4),GPCOD(2,9),
     *          HVALU(MMATS,25),
     *          PSTRN(MMATS,25)
      TWOPI = 6.283185308
      REWIND 1
      KGAUS = 0
C
C*** LOOP OVER EACH ELEMENT
C
      DO 70 IELEM = 1,NELEM
      LPROP =  MATNO(IELEM)
C
C*** EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
C
      DO 10 INODE = 1,NNODE
      LNODE = IABS(LNODS(IELEM,INODE))
      IPOSN =(LNODE-1)*2
      DO 10 IDIME = 1,2
      IPOSN = IPOSN + 1
   10 ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
      THICK = PROPS(LPROP,3)
C
C*** INITIALIZE THE ELEMENT STIFFNESS MATRIX
C
      DO 20 IEVAB = 1,NEVAB
      DO 20 JEVAB = 1,NEVAB
   20 ESTIF(IEVAB,JEVAB) = 0.0
      KGASP = 0
C
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
      DO 50 IGAUS = 1,NGAUS
      EXISP = POSGP(IGAUS)
      DO 50 JGAUS = 1,NGAUS
      ETASP = POSGP(JGAUS)
      KGASP = KGASP+1
      KGAUS = KGAUS+1
C
```

```
C*** EVALUATE THE D-MATRIX
C
      CALL MODPS1(DMATX,LPROP,MMATS,NTYPE,PROPS)
C
C*** EVALUATE THE SHAPE FUNCTIONS,ELEMENTAL VOLUME ETC.
C
      CALL SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .            NNODE,SHAPE)
      DVOLU = DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF (NTYPE.EQ.3) DVOLU = DVOLU*TWOPI*GPCOD(1,KGASP)
      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
C
C*** EVALUATE THE B AND DB MATRICES
C
      CALL BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      IF (IINCS.EQ.1) GO TO 80
      IF (EPSTN(KGAUS).EQ.0.0) GO TO 80
      DO 90 ISTR1 = 1,NSTR1
   90 STRES(ISTR1) = STRSG(ISTR1,KGAUS)
      CALL INVAR1(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,
     .            THETA,VARJ2,YIELD)
      CALL YIELD1(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .            PROPS,SINT3,STEFF,THETA,VARJ2)
      PLSTN = EPSTN (KGAUS)
      CALL HNUMB  ( PSTRN,HVALU, NUMB,MMATS,LPROP,
     .              PLSTN,HARDS,PROPS,NMATS)
      CALL FLOWP1(AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS,
     .            HARDS)
      DO 100 ISTRE = 1,NSTRE
      DO 100 JSTRE = 1,NSTRE
  100 DMATX(ISTRE,JSTRE)=DMATX(ISTRE,JSTRE)-ABETA*DVECT(ISTRE)*
     .DVECT(JSTRE)
   80 CONTINUE
      CALL DBE(BMATX,DBMAT,DMATX,MEVAB,NEVAB,NSTRE,NSTR1)
C
C***  CALCULATE THE ELEMENT STIFFNESSES
C
      DO 50 IEVAB = 1,NEVAB
      DO 50 JEVAB = IEVAB,NEVAB
      DO 30 ISTRE = 1,NSTRE
   30 ESTIF(IEVAB,JEVAB)=ESTIF(IEVAB,JEVAB)+BMATX(ISTRE,IEVAB)*
     .DBMAT(ISTRE,JEVAB)*DVOLU
   50 CONTINUE
C
C*** CONSTRUCT THE LOWER TRIANGLE OF THE STIFFNESS MATRIX    .
C
      DO 60 IEVAB = 1,NEVAB
      DO 60 JEVAB = 1,NEVAB
   60 ESTIF(JEVAB,IEVAB) = ESTIF(IEVAB,JEVAB)
C
C*** STORE THE STIFFNESS MATRIX,STRESS MATRIX,AND SAMPLING POINT
C    COORDINATES FOR EACH ELEMENT ON DISC FILE
C
      WRITE(1) ESTIF

   70 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FLOWP1(AVECT,ABETA,DVECT,NTYPE,PROPS,
     .                  LPROP,NSTR1,MMATS,HARDS)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES THE PLASTIC D VECTOR
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AVECT(4),DVECT(4),PROPS(MMATS,6)
      YOUNG = PROPS(LPROP,1)
      POISS =PROPS(LPROP,2)
      FMUL1 = YOUNG/(1.0+POISS)
      IF (NTYPE.EQ.1) GO TO 60
      FMUL2 = YOUNG*POISS*(AVECT(1)+AVECT(2)+AVECT(4))/((1.0+POISS)*
     .(1.0-2.0*POISS))
      DVECT(1) = FMUL1*AVECT(1)+FMUL2
      DVECT(2) = FMUL1*AVECT(2)+FMUL2
      DVECT(3) =0.5*AVECT(3)*YOUNG/(1.0+POISS)
      DVECT(4) =FMUL1*AVECT(4)+FMUL2
      GO TO 70
   60 FMUL3=YOUNG*POISS*(AVECT(1)+AVECT(2))/(1.0-POISS*POISS)
      DVECT(1) = FMUL1*AVECT(1)+FMUL3
      DVECT(2) = FMUL1*AVECT(2)+FMUL3
      DVECT(3) = 0.5*AVECT(3)*YOUNG/(1.0+POISS)
      DVECT(4) = FMUL1*AVECT(4)+FMUL3
   70 DENOM = HARDS
      DO 80 ISTR1 = 1,NSTR1
   80 DENOM = DENOM+AVECT(ISTR1)*DVECT(ISTR1)
      ABETA = 1.0/DENOM
      RETURN
      END
```

```
      SUBROUTINE RESID2(ASDIS,COORD,EFFST,ELOAD,FACTO,IITER,LNODS,
     .                  LPROP,MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,NDOFN,
     .                  NELEM,NEVAB,NGAUS,NNODE,NSTR1,NTYPE,POSGP,PROPS,
     .                  NSTRE,NCRIT,STRSG,WEIGP,TDISP,EPSTN,
     .                         HVALU,PSTRN,NUMB,
     .                  HAROS,PLSTN,NMATS)
C************************************************************************
C
C**** THIS SUBROUTINE REDUCES THE STRESSES TO THE YIELD SURFACE AND
C     EVALUATES THE EQUIVALENT NODAL FORCES
C
C************************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ASDIS(MTOTV),AVECT(4),CARTD(2,9),COORD(MPOIN,2),
     .          DEVIA(4),DVECT(4),EFFST(MTOTG),ELCOD(2,9),ELDIS(2,9),
     .          ELOAD(MELEM,18),LNODS(MELEM,9),POSGP(4),PROPS(MMATS,6),
     .          STRAN(4),STRES(4),STRSG(4,MTOTG),
     .          WEIGP(4),DLCOD(2,9),DESIG(4),SIGMA(4),SGTOT(4),
     .          DMATX(4,4),DERIV(2,9),SHAPE(9),GPCOD(2,9),
     .          EPSTN(MTOTG),TDISP(MTOTV),MATNO(MELEM),BMATX(4,18),
     .          HVALU ( MMATS,25 ),
     .          PSTRN (MMATS,25)
      ROOT3 =1.73205080757
      TWOPI = 6.283185308
      DO 10 IELEM = 1,NELEM
      DO 10 IEVAB = 1,NEVAB
   10 ELOAD(IELEM,IEVAB) = 0.0
      KGAUS = 0
      DO 20 IELEM = 1,NELEM
      LPROP = MATNO(IELEM)
      UNIAX = PROPS(LPROP,5)
      FRICT = PROPS(LPROP,6)
      IF (NCRIT.EQ.3) UNIAX =PROPS(LPROP,5)*DCOS(FRICT*0.017453292)
      IF (NCRIT.EQ.4) UNIAX =6.0*PROPS(LPROP,5)*DCOS(FRICT*0.017453292)/
     . (ROOT3*(3.0-DSIN(FRICT*0.017453292)))
C
C*** COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      DO 30 INODE =1,NNODE
      LNODE = IABS(LNODS(IELEM,INODE))
      NPOSN = (LNODE-1)*NDOFN
      DO 30 IDOFN = 1,NDOFN
      NPOSN = NPOSN+1
      ELCOD(IDOFN,INODE)=COORD(LNODE,IDOFN)
   30 ELDIS(IDOFN,INODE)=ASDIS(NPOSN)
      CALL MODPS1( DMATX,LPROP,MMATS,NTYPE,PROPS)
      THICK = PROPS(LPROP,3)
      KGASP = 0
      DO 40 IGAUS = 1,NGAUS
      DO 40 JGAUS = 1,NGAUS
      EXISP = POSGP(IGAUS)
      ETASP = POSGP(JGAUS)
      KGAUS = KGAUS +1
      KGASP = KGASP+1
```

```
      ESTRS = EFFST (KGAUS)
      PSTNO = EPSTN (KGAUS)
      PLSTN = PSTNO
      IF (PLSTN.EQ.0.0) HARDS = 0.0
      IF (PLSTN.GT.0.0) CALL
     .    HNUMB  (PSTRN,HVALU, NUMB,MMATS,LPROP,
     .            PLSTN,HARDS,PROPS,NMATS)
      CALL  SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL  JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
     .            NNODE,SHAPE)
      DVOLU = DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
      IF (NTYPE.EQ.3) DVOLU = DVOLU*TWOPI* GPCOD(1,KGASP)
      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
      CALL  BMATPS(BMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      CALL  LINER1 (CARTD,DMATX,ELDIS,LPROP,MMATS,NDOFN,NNODE,NSTRE,
     .            NTYPE,PROPS,STRAN,STRES,KGASP,GPCOD,SHAPE)
      IF (EPSTN (KGAUS).EQ.0.0) PREYS = UNIAX
      IF (EPSTN (KGAUS).GT.0.0) PREYS = EFFST(KGAUS)
      DO 150 ISTR1 = 1,NSTR1
      DESIG(ISTR1) = STRES(ISTR1)
  150 SIGMA(ISTR1) = STRSG(ISTR1,KGAUS)+STRES(ISTR1)
      CALL  INVAR1 (     DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,
     .       SIGMA,THETA,VARJ2,YIELD)
      IF   (EPSTN (KGAUS).GT.0.0) GO TO 50
      ESCUR = YIELD-PREYS
      IF (ESCUR.LE.0.0) GO TO 60
      RFACT =ESCUR/(YIELD-EFFST(KGAUS))
      GO TO 70
   50 ESCUR = YIELD-EFFST(KGAUS)
      IF (ESCUR.LE.0.0) GO TO 60
      RFACT =1.0
   70 MSTEP = ESCUR*8.0/PREYS+1.0
      ASTEP = MSTEP
      REDUC = 1.0-RFACT
      DO 80 ISTR1 = 1,NSTR1
      SGTOT(ISTR1) = STRSG(ISTR1,KGAUS)+REDUC*STRES(ISTR1)
   80 STRES(ISTR1) = RFACT*STRES(ISTR1)/ASTEP
      DO 90 ISTEP = 1,MSTEP
      CALL  INVAR1(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SGTOT,
     .            THETA,VARJ2,YIELD)
      CALL  YIELD1 (AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .            PROPS,SINT3,STEFF,THETA,VARJ2)
      CALL  FLOWP1 (AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS,
     .            HARDS)
      AGASH = 0.0
      DO 100 ISTR1 = 1,NSTR1
  100 AGASH = AGASH+AVECT(ISTR1)*STRES(ISTR1)
      DLAMD = AGASH*ABETA
      IF (DLAMD.LT.0.0) DLAMD = 0.0
      BGASH = 0.0
      DO 110 ISTR1 = 1,NSTR1
      BGASH = BGASH+AVECT(ISTR1)*SGTOT(ISTR1)
  110 SGTOT(ISTR1) = SGTOT(ISTR1)+STRES(ISTR1)-DLAMD*DVECT(ISTR1)
      EPSTN(KGAUS) = EPSTN(KGAUS)+ DLAMD*BGASH/YIELD
   90 CONTINUE
      PSTNN = EPSTN (KGAUS)
      CALL  INVAR1(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SGTOT,
     .            THETA,VARJ2,YIELD)
      DEPSTN = PSTNN - PSTNO
      PLSTN = PSTNN
      IF (PLSTN.EQ.0.0) HARDS = 0.0
      IF (PLSTN.GT.0.0) CALL
     .    HNUMB (PSTRN,HVALU,NUMB,MMATS,LPROP,PLSTN,HARDS,PROPS,NMATS)
      CURYS = PREYS                * DEPSTN * HARDS
      BRING = 1.0
      IF (YIELD.GT.CURYS) BRING = CURYS/YIELD
      DO 130 ISTR1 = 1,NSTR1
  130 STRSG(ISTR1,KGAUS) = BRING*SGTOT(ISTR1)
      EFFST(KGAUS) = BRING*YIELD
C**** ALTERNATIVE LOCATION OF STRESS REDUCTION LOOP TERMINATION CARD
C  90 CONTINUE
C****
      GO TO 190
   60 DO 180 ISTR1 = 1,NSTR1
  180 STRSG(ISTR1,KGAUS) = STRSG(ISTR1,KGAUS)+DESIG(ISTR1)
      EFFST(KGAUS) = YIELD
C
C*** CALCULATE THE EQUIVALENT NODAL FORCES AND ASSOCIATE WITH THE
C     ELEMENT NODES
  190 MGASH = 0
      DO 140 INODE = 1,NNODE
      DO 140 IDOFN = 1,NDOFN
      MGASH = MGASH+1
      DO 140 ISTRE = 1,NSTRE
  140 ELOAD(IELEM,MGASH) = ELOAD(IELEM,MGASH)+BMATX(ISTRE,MGASH)
     .  * STRSG(ISTRE,KGAUS)*DVOLU
   40 CONTINUE
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE LINER1(CARTD,DMATX,ELDIS,LPROP,MMATS,NDOFN,NNODE,NSTRE,
     .                  NTYPE,PROPS,STRAN,STRES,KGASP,GPCOD,SHAPE)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATES STRESSES AND STRAINS ASSUMING LINEAR
C    ELASTIC BEHAVIOUR
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AGASH(2,2),CARTD(2,9),DMATX(4,4),ELDIS(2,9),
     .          PROPS(MMATS,6),STRAN(4),STRES(4),
     .          GPCOD(2,9),SHAPE(9)
      POISS = PROPS(LPROP,2)
      DO 20 IDOFN = 1,NDOFN
      DO 20 JDOFN = 1,NDOFN
      BGASH = 0.0
      DO 10 INODE = 1,NNODE
   10 BGASH = BGASH+CARTD(JDOFN,INODE)*ELDIS(IDOFN,INODE)
   20 AGASH(IDOFN,JDOFN) = BGASH
C
C*** CALCULATE THE STRAINS
C
      STRAN(1) = AGASH(1,1)
      STRAN(2) = AGASH(2,2)
      STRAN(3) = AGASH(1,2)+AGASH(2,1)
      STRAN(4) = 0.0
      DO 30 INODE = 1,NNODE
   30 STRAN(4) = STRAN(4)+ELDIS(1,INODE)*SHAPE(INODE)/GPCOD(1,KGASP)
C
C*** AND THE CORRESPONDING STRESSES
C
      DO 40 ISTRE = 1,NSTRE
      STRES(ISTRE) = 0.0
      DO 40 JSTRE = 1,NSTRE
   40 STRES(ISTRE) = STRES(ISTRE)+DMATX(ISTRE,JSTRE)*STRAN(JSTRE)
      IF(NTYPE.EQ.1) STRES(4)=0.0
      IF (NTYPE.EQ.2) STRES(4)=POISS*(STRES(1)+STRES(2))
      RETURN
      END
```

```
      SUBROUTINE HNUMB(PSTRN,HVALU,NUMB,MMATS,LPROP,PLSTN,HARDS,
     .                 PROPS,NMATS)
C***********************************************************************
C
C     THIS SUBROUTINE CHOOSES THE HARDNESS VALUE ,HARDS,DEPENDING
C     ON THE TOTAL EFFECTIVE PLASTIC STRAIN
C
C***********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION HVALU(MMATS,25),PSTRN(MMATS,25),
     .          PROPS(MMATS,6)
C
C
C***  FOR THE EFFECTIVE PLASTIC STRAIN,PLSTN,CHOOSE THE HARDNESS
C     VALUE
C
      IF (PLSTN.EQ.0.0) GO TO 5
      IF (PLSTN.LE.PSTRN(LPROP,1)) GO TO 15
      DO 10 INUMB = 2,NUMB
      INDEX = INUMB
      IF (PSTRN(LPROP,(INUMB-1)) .LT.PLSTN.AND.PLSTN.LE.
     .    PSTRN(LPROP, INUMB))   GO TO 25
   10 CONTINUE
      IF (PLSTN.GT.PSTRN(LPROP,NUMB)) GO TO 35
    5 HARDS = 0.0
      RETURN
   15 HARDS = HVALU(LPROP,1)
      RETURN
   25 HARDS = HVALU(LPROP,INDEX)
      RETURN
   35 HARDS = HVALU(LPROP,NUMB)
      RETURN
      END
```

```
      SUBROUTINE   INVAR1(  DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,
     .                 STEMP,THETA,VARJ2,YIELD)
C***********************************************************************
C
C*** THIS SUBROUTINE EVALUATSS THE STRESS INVARIANTS AND THE CURRENT
C    VALUE OF THE YIELD FUNCTION
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DEVIA(4),PROPS(MMATS,6),STEMP(4)
      ROOT3 = 1.73205080757
      SMEAN = (STEMP(1)+STEMP(2)+STEMP(4))/3.0
      DEVIA(1) = STEMP(1)-SMEAN
      DEVIA(2) = STEMP(2)-SMEAN
      DEVIA(3) = STEMP(3)
      DEVIA(4) = STEMP(4)-SMEAN
      VARJ2 = DEVIA(3)*DEVIA(3)+0.5*(DEVIA(1)*DEVIA(1)+DEVIA(2)*
     .        DEVIA(2)+DEVIA(4)*DEVIA(4))
      VARJ3 = DEVIA(4)*DEVIA(4)*DEVIA(4)-VARJ2
      STEFF =DSQRT(VARJ2)
      IF (STEFF.EQ.0.0) GO TO 10
      SINT3 = -3.0*ROOT3*VARJ3/(2.0*VARJ2*STEFF)
      IF (SINT3.GT.1.0) SINT3 = 1.0
      GO TO 20
   10 SINT3 = 0.0
   20 CONTINUE
      IF (SINT3.LT.-1.0) SINT3 = -1.0
      IF (SINT3.GT.1.0) SINT3 = 1.0
      THETA = DARSIN(SINT3)/3.0
      GO TO (1,2,3,4),NCRIT
C*** TRESCA
    1 YIELD = 2.0*DCOS(THETA)*STEFF
      RETURN
C*** VON MISES
    2 YIELD = ROOT3*STEFF
      RETURN
C*** MOHR-COULCMB
    3 PHIRA = PROPS(LPROP,7)*0.017453292
      SNPHI =DSIN(PHIRA)
      YIELD = SMEAN*SNPHI+STEFF*(DCOS(THETA)-DSIN(THETA)*SNPHI/ROOT3)
      RETURN
C*** DRUCKER-PRAGER
    4 PHIRA = PROPS(LPROP,7)*0.017453292
      SNPHI =DSIN(PHIRA)
      YIELD = 6.0*SMEAN*SNPHI/(ROOT3*(3.0-SNPHI))+STEFF
      RETURN
      END
```

--IEL01--

```
      SUBROUTINE Y        (AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
     .                     PROPS,SINT3,STEFF,THETA,VARJ2)
C***********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE FLOY VECTOR
C
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AVECT(4),DEVIA(4),PROPS(MMATS,6),
     .          VECA1(4),VECA2(4),VECA3(4)
      IF (STEFF.EQ.0.0) RETURN
      FRICT = PROPS(LPROP,6)
      ABTHE = DABS (THETA * 57.29577951308)
      IF (ABTHE .LT. 29.0 ) GO TO 15
      IF (ABTHE .GE. 29.0) GO TO 25
   15 TANTH =DTAN(THETA)
      TANT3 =DTAN(3.0*THETA)
      SINTH =DSIN(THETA)
      COSTH =DCOS(THETA)
      COST3 =DCOS(3.0*THETA)
   25 ROOT3 = 1.73205080757
C
C*** CALCULATE VECTOR A1
C
      VECA1(1) = 1.0
      VECA1(2) = 1.0
      VECA1(3) = 0.0
      VECA1(4) = 1.0
C
C*** CALCULATE VECTOR A2
C
      DO 10 ISTR1 = 1,NSTR1
10    VECA2(ISTR1) = DEVIA(ISTR1)/(2.0*STEFF)
      VECA2(3) = DEVIA(3)/STEFF
C
C*** CALCULATE VECTOR A3
C
      VECA3(1) = DEVIA(2)*DEVIA(4)+VARJ2/3.0
      VECA3(2) = DEVIA(1)*DEVIA(4)+VARJ2/3.0
      VECA3(3) = -2.0*DEVIA(3)*DEVIA(4)
      VECA3(4) = DEVIA(1)*DEVIA(2)-DEVIA(3)*DEVIA(3)+VARJ2/3.0
      GO TO (1,2,3,4),NCRIT
C
C*** TRESCA
C
    1  CONS1 = 0.0
       ABTHE =DABS(THETA*57.29577951308)
       IF (ABTHE.LT.29.0) GO TO 20
       CONS2 = ROOT3
       CONS3 = 0.0
       GO TO 40
   20  CONS2 = 2.0*(COSTH+SINTH*TANT3)
       CONS3 = ROOT3*SINTH/(VARJ2*COST3)
       GO TO 40
C

C*** VON MISES
C
    2  CONS1 =0.0
       CONS2 = ROOT3
       CONS3 = 0.0
       GO TO 40
C
C*** MOHR-COULOMB
C
    3  CONS1 =DSIN(FRICT*0.017453292)/3.0
       ABTHE =DABS(THETA*57.29577951308)
       IF (ABTHE.LT.29.0) GO TO 30
       CONS3 = 0.0
       PLUMI = 1.0
       IF (THETA.GT.0.0) PLUMI = -1.0
       CON52 = 0.5*(ROOT3+PLUMI*CONS1*ROOT3)
         GO TO 40
   30  CONS2 = COSTH*((1.0+TANTH*TANT3)+CONS1*(TANT3-TANTH)*ROOT3)
       CONS3 = (ROOT3*SINTH+3.0*CONS1*COSTH)/(2.0*VARJ2*COST3)
       GO TO 40
C
C*** DRUCKER-PRAGER
C
    4  SNPHI =DSIN(FRICT*0.017453292)
       CONS1 = 2.0*SNPHI/(ROOT3*(3.0-SNPHI))
       CONS2 = 1.0
       CONS3 = 0.0
   40  CONTINUE
       DO 50 ISTR1 = 1,NSTR1
   50  AVECT(ISTR1) = CONS1*VECA1(ISTR1)+CONS2*VECA2(ISTR1)+CONS3*
     *   VECA3(ISTR1)
       RETURN
       END
```

```
      SUBROUTINE MODPS1(DMATX,LPROP,MMATS,NTYPE,PROPS)
C**********************************************************************
C
C**** THIS SUBROUTINE EVALUATES THE D-MATRIX
C
C**********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DMATX(4,4),PROPS(MMATS,6)
      YOUNG = PROPS(LPROP,1)
      POISS = PROPS(LPROP,2)
      DO 10 ISTR1 = 1,4
      DO 10 JSTR1 = 1,4
   10 DMATX(ISTR1,JSTR1) = 0.0
      IF (NTYPE.NE.1) GO TO 4
C
C*** D MATRIX FOR PLANE STRESS CASE
C
      CONST = YOUNG / (1.0-POISS*POISS)
      DMATX(1,1) = CONST
      DMATX(2,2) = CONST
      DMATX(1,2) = CONST*POISS
      DMATX(2,1) = CONST*POISS
      DMATX(3,3) = (1.0-POISS)*CONST/2.0
      RETURN
    4 IF (NTYPE.NE.2) GO TO 6
C
C*** D MATRIX FOR PLANE STRAIN CASE
C
      CONST = YOUNG*(1.0-POISS)/((1.0+POISS)*(1.0-2.0*POISS))
      DMATX(1,1) = CONST
      DMATX(2,2) = CONST
      DMATX(1,2) = CONST*POISS/(1.0-POISS)
      DMATX(2,1) = CONST*POISS/(1.0-POISS)
      DMATX(3,3) = (1.0-2.0*POISS)*CONST/(2.0*(1.0-POISS))
      RETURN
    6 IF (NTYPE.NE.3) GO TO 8
C
C*** D MATRIX FOR AXISYMMETRIC CASE
C
      CONST = YOUNG * (1.0-POISS)/((1.0+POISS)*(1.0-2.0*POISS))
      CONSS = POISS/(1.0-POISS)
      DMATX(1,1) = CONST
      DMATX(2,2) = CONST
      DMATX(3,3) = CONST*(1.0-2.0*POISS)/(2.0*(1.0-POISS))
      DMATX(1,2) = CONST*CONSS
      DMATX(1,4) = CONST*CONSS
      DMATX(2,1) = CONST*CONSS
      DMATX(2,4) = CONST*CONSS
      DMATX(4,1) = CONST*CONSS
      DMATX(4,2) = CONST*CONSS
      DMATX(4,4) = CONST
    8 CONTINUE
      RETURN
      END
```

BIBLIOGRAPHY

Nadai, A. Plasticity. New York and London: McGraw-Hill Book Company, Inc., 1931.

Nayak, G.C. and Zienkiewicz, O.C. "Convenient Form of Stress Invariants for Plasticity." Proceedings of American Society of Civil Engineers, 98, ST4, 949-954 (1972).

Phillips, Aris. Introduction to Plasticity. New York: The Ronald Press Company, 1956.

Tuba, I.S. "Elastic-Plastic Stress and Strain Concentration Factors at a Circular Hole in a Uniformly Stressed Infinite Plate." Journal of Applied Mechanics, Trans. ASME, 710/September 1965.

REFERENCES

Bathe, Klaus-Jurgen. Finite Element Procedures in Engineering Analysis, Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1982.

D'Isa, Frank A. Mechanics of Metals. Reading, Massachusetts: Academic Press, 1968.

Hammel, J.W. and Bodisco, U.V. and Mattheck, C."An Elastic-Plastic Finite Element Analysis of a CT Fracture Specimen." Computers & Structures, Vol.13, pp.757-770, 1981.

Hinton, E. and Owen, D.R.J. Finite Elements in Plasticity: Theory and Practice. Swansea, U.K.: Pineridge Press Limited, 1980.

Mendelson, Alexander. Plasticity: Theory and Application. New York: The Macmillan Company, 1968.

Owen, D.R.J. and Fawkes, A.J. Engineering Fracture Mechanics: Numerical Methods and Applications. Swansea, U.K.: Pineridge Press Ltd., 1983.

Zienkiewicz, O.C. and 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),