

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
Adviser

NOV 21ST, 85
Date

Sally M. Hotchkiss
Dean of the Graduate School

November 26, 1985
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

	PAGE
ABSTRACT	ii
ACKNOWLEDGEMENTS	iv
TABLE OF CONTENTS	v
LIST OF SYMBOLS	vii
LIST OF FIGURES	xiv
LIST OF TABLES	xvi
INTRODUCTION	xvii
 CHAPTER	
I. Concept of Finite Element Method	1
II. Basic Numerical Solution Process for Nonlinear Problems	16
III. The Mathematical Theory of Plasticity	20
IV. Matrix Formulation and Basic Expressions for Two Dimensional Problems	27
V. Finite Element Expressions and Program Structure	31
VI. Test Problems with Input and Output	40
VII. Solution of Some Typical Problems and Results	58
VIII. Conclusion	73
APPENDIX A. Instructions for Preparing Input Data for Linear Strain Hardening Program Master	76
APPENDIX B. Instructions for Preparing Input Data for Nonlinear Strain Hardening Program Mster2	83
APPENDIX C. Finite Element Program for Linear Strain Hardening Materials	86

TABLE OF CONTENTS

	PAGE
APPENDIX D. Finite Element Program for Nonlinear Strain Hardening Materials	117
BIBLIOGRAPHY	135
REFERENCES	136

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
$\bar{\epsilon}$	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 and θ 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
$\xi_i, \eta_j, \xi_k, \xi_p, \eta_q$	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 and θ 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
1.1 General Three Dimensional Body	2
1.2 Library of Elements	9
I.3 Element Loading	14
V.1 Program Outline for Linear and Nonlinear Two Dimensional Elasto-Plastic Problems	32
V.2 Process of Reducing the Incremental Stress to Yield Surface	38
VI.1 Plane Stress Elasto-Plastic Problem of a Circular Hole in a Uniformly Stressed Infinite Plate	41
VI.2 Spread of Plastic Zones at Different Load Levels for Plane Stress Elasto-Plastic Problem of a Circular Hole in a Uniform- ly Stressed Infinite Plate	41
VI.3 Stress Concentration Factor k_{σ_e} at Different Levels of Applied Loading σ_e for Elasto- Plastic Problem of a Circular Hole in a Uniformly Stressed Infinite Plate	45
VI.4 Thick Walled Cylinder Subjected to Internal Pressure	46
VI.5 Comparison of Linear and Nonlinear Program Outputs for Stresses for Thick Walled Cylinder under Internal Pressure	55
VI.6 Comparison of Linear and Nonlinear Program Outputs for Effective Plastic Strains for Thick Walled Cylinder under Internal Pressure	56
VII.1 Geometry of Compact Tension (CT) Specimen ..	60
VII.2 Load Vs Load Line Displacement Graph for CT Specimen	60
VII.3 Spread of Plastic Zones for CT Specimen	61
VII.4 Fully Plastic Cylinder - Comparison of Stre- sses for Perfectly Plastic Material and Linear Strain Hardening Material	63

FIGURE	PAGE
VII.5 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with a Single Circular Hole	66
VII.6 Enlarged Scale Drawing of Spread of Plastic Zone for an Infinite Plate with a Single Circular Hole	66
VII.7 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with Two Circular Holes	69
VII.8 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with Three Circular Holes	69
VII.9 Displacement Values at Different Levels of Loading for a Perforated Tension Strip ...	72

LIST OF TABLES

TABLE		PAGE
I.1	List of Variables and Elasticity Matrices ...	7
1.2	Nodal Displacements, Strain Matrices and Elemental Volumes or Areas	12
IV.1	Derivatives of Stress Invariants	29
IV.2	Constants c for Various Yield Conditions.. ...	29
IV.3	Elasticity Matrices	30
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	43
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	44
VI.3	Comparison of Linear Program Output with Closed Form Solution for Elastic Thick Walled Cylinder Under Internal Pressure ...	49
VI.4	Comparison of Linear Program Output with Closed Form Solution for Partly Plastic Thick Walled Cylinder of Perfectly Plastic Material under Internal Pressure	50
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	52
VI.6	Comparison of Linear Program Output with Closed Form Solution for Radial Displacement for Thick Walled Cylinder under Internal Pressure	53
VII.1	Maximum Values of Tangential Stresses and Stress Concentration Factors for a Perforated Tension Strip	70

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:

$$\begin{array}{l} 1. \text{ Surface tractions} \\ 2. \text{ Body forces} \end{array} \quad \begin{array}{l} \mathbf{f}^S = \begin{bmatrix} f_x^S \\ f_y^S \\ f_z^S \end{bmatrix} \\ \mathbf{f}^b = \begin{bmatrix} f_x^b \\ f_y^b \\ f_z^b \end{bmatrix} \end{array}$$

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

I.1

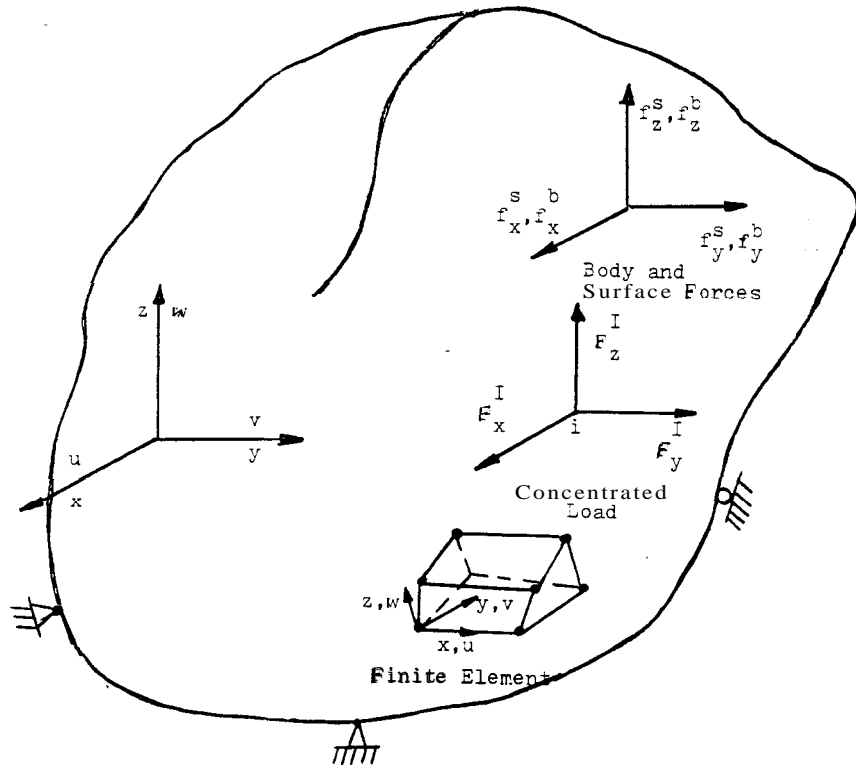


Fig. I.1 General Three Dimensional Body.

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

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

The strain components corresponding to U are,

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

The corresponding stresses are,

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

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,

$$\int_V \bar{\epsilon}^T \sigma \, dV = \int_V \bar{U}^T f^b \, dV + \int_S \bar{U}^S f^S \, dS + \sum_I \bar{U}^I f^I \quad I.5$$

where ,

$$\begin{aligned} \bar{\epsilon}^T &= \text{Virtual strain} \\ &= \{\bar{\epsilon}_{xx} \quad \bar{\epsilon}_{yy} \quad \bar{\epsilon}_{zz} \quad \bar{\gamma}_{xx} \quad \bar{\gamma}_{yy} \quad \bar{\gamma}_{zz}\} \end{aligned} \quad I.6$$

$$\begin{aligned} \bar{U} &= \text{Virtual displacement} \\ &= \{\bar{u} \quad \bar{v} \quad \bar{w}\} \end{aligned} \quad I.7$$

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 within an element is,

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

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,

$$\{\hat{U}\}^T = \{\hat{U}_1 \hat{U}_2 \hat{U}_3 \dots \hat{U}_M\} \quad \text{I.9}$$

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

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

where B^e is strain matrix. Finally,

$$\sigma^e = D^e \epsilon^e \quad , \quad \text{I.11}$$

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

The equilibrium 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:

$$\begin{aligned}
\sum_e \int_V \bar{\epsilon}^{eT} \sigma^e dV &= \sum_e \int_V \bar{U}^{eT} f^{b(e)} dV \\
&+ \sum_e \int_S \bar{U}^{s(e)T} f^{s(e)} dS \\
&+ \sum_e \bar{U}^{IT} f^I
\end{aligned} \quad , \quad \text{I.12}$$

where \sum_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),

$$\begin{aligned}
\hat{U}^T \left[\sum_e \int_V B^{eT} D^e B^e dV \right] \hat{U} &= \hat{U}^T \left\{ \sum_e \int_V N^{eT} f^{b(e)} dV \right\} \\
&+ \left\{ \sum_e \int_S N^{(s)eT} f^{s(e)} dS \right\} \\
&+ F
\end{aligned} \quad \text{I.13}$$

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.

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

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

the equilibrium equations reduce to

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

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

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

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

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

In the expression (I.18),

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

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

and $R_C = F = \text{concentrated loads} . \quad \text{I.21}$

The formulation of equilibrium 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 equilibrium 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 { ϵ } ^T	Stress vector { σ } ^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 = Poisson'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,

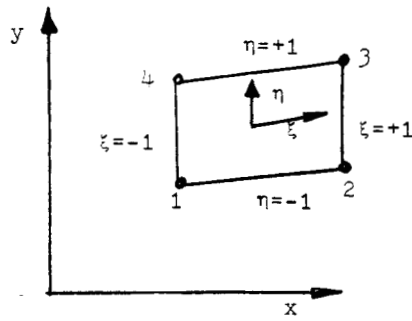
$$U^e = \sum_{i=1}^n N_i^e \hat{U}_i^e \quad , \quad 1.22$$

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

$$\begin{bmatrix} x^e \\ y^e \end{bmatrix} = \sum_{i=1}^n \begin{bmatrix} N_i^e & 0 \\ 0 & N_i^e \end{bmatrix} \begin{bmatrix} x_i^e \\ y_i^e \end{bmatrix} \quad . \quad 1.23$$

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,

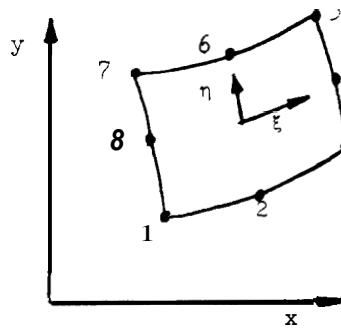
$$J^e = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n \frac{\partial N_i^e}{\partial \xi} x_i^e & \sum_{i=1}^n \frac{\partial N_i^e}{\partial \xi} y_i^e \\ \sum_{i=1}^n \frac{\partial N_i^e}{\partial \eta} x_i^e & \sum_{i=1}^n \frac{\partial N_i^e}{\partial \eta} y_i^e \end{bmatrix} \quad . \quad 1.24$$



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

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

(a) The Linear Element



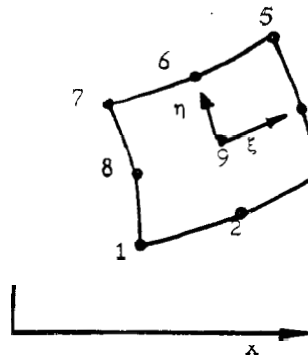
For corner nodes, 1, 3, 5, 7

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

For midside nodes, 2, 4, 6, 8

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

(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,

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

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,

$$\epsilon^e = \sum_{i=1}^n B_i^e U_i^e \quad , \quad I.25$$

where B^e is the strain matrix.

The elemental volume is given as,

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

The expressions for 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

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

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.

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

The integration is to be performed numerically. If the

integrand in (1.28) is denoted as,

$$B^e D^e B^e t^e \det J^e = T^e, \quad I.29$$

then,

$$\{K\} = \sum_e \int_{-1}^{+1} \int_{-1}^{+1} T^e d\xi d\eta. \quad I.30$$

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,

$$\{K\} = \sum_e \left\{ \sum_i \sum_j w_i w_j T(\xi_i, \eta_j) \right\}, \quad I.31$$

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,

$$\{K\} = \sum_e \left[\sum_i \sum_j \sum_k w_i w_j w_k T(\xi_i, \eta_j, \zeta_k) \right]. \quad I.32$$

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).

$$\begin{aligned} R_B &= \sum_e \int_V N^e T f^{b(e)} dV \\ &= \sum_e \int_{-1}^{+1} \int_{-1}^{+1} N^e T f^{b(e)} t^e \det J^e d\xi d\eta \\ &= \text{body forces} \end{aligned} \quad I.33$$

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

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

$$R_B = \sum_e \left\{ \int_{-1}^{+1} \int_{-1}^{+1} g^e d\xi d\eta \right\} \tag{I.34}$$

$$= \sum_e \left\{ \sum_i \sum_j w_i w_j g(\xi_i, \eta_j) \right\} . \tag{I.35}$$

TABLE I.2

NODAL DISPLACEMENTS, STRAIN MATRICES
AND ELEMENTAL VOLUMES OR AREAS

Type of problem	Nodal displacement	Strain Matrix	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:

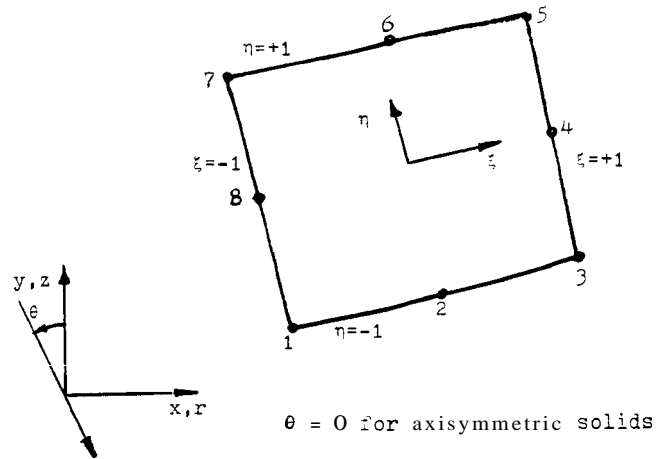
$$\begin{bmatrix} P_{xi} \\ P_{yi} \end{bmatrix} = \int_{V^e} N_i^e \rho g \begin{bmatrix} \sin(\theta) \\ -\cos(\theta) \end{bmatrix} dV, \quad \text{I.36}$$

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

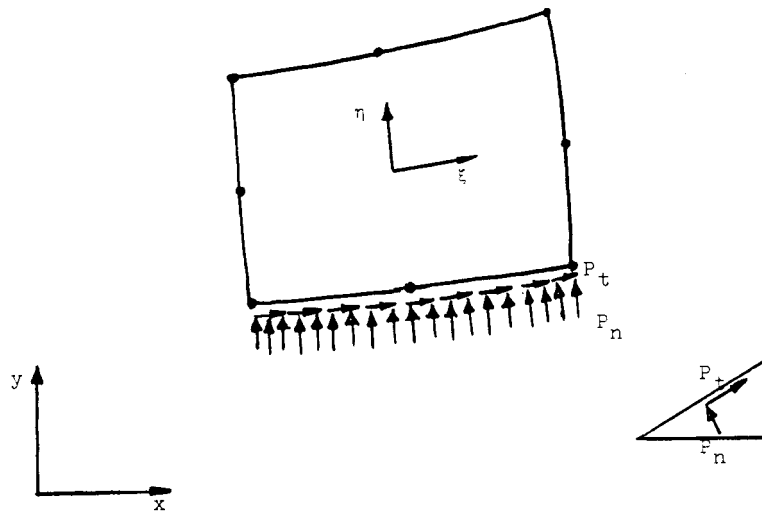
technique, resulting in:

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

For axisymmetric problems t is replaced by $2nr_P$ 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 accommodated. 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:

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

where integration is done along the element loaded edge, arbitrarily 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 equilibrium 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,

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

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 equilibrium equations amount to finding the solution of equations,

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

where $\kappa(U)$ can be interpreted as a measure of departure from the equilibrium. 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,

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

where ,

M = total number of variables in the system,

and

i = the particular node number.

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

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

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

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

This allows the correction vector of unknown nodal displace-

ments U to be obtained from the deviation from the equilibrium 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:

$$\Delta U^r = [K(U^r)]^{-1} \kappa(U)^r. \quad \text{II.5}$$

A trial value $U^{\hat{u}}$ is assumed; the tangential stiffness $K(U^0)$ corresponding to this displacement is then obtained; the departure from equilibrium value $\kappa(U)^0$ 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:

$$\Delta U^r = [K(U^0)]^{-1} \kappa(U)^r. \quad \text{II.6}$$

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:

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

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 multi-axial stresses will cause yielding
3. Relations between stresses and strains when plastic flow is occurring

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

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

which can also be expressed as,

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

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

Yield criteria can be defined by the relation,

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

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,

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

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,

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

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

$$\begin{aligned} J'_1 &= S_1 + S_2 + S_3 = 0 \\ J'_2 &= \frac{1}{2} (S_1^2 + S_2^2 + S_3^2) = -(S_1 S_2 + S_2 S_3 + S_3 S_1) \\ J'_3 &= \frac{1}{3} (S_1^3 + S_2^3 + S_3^3) = S_1 S_2 S_3 \quad . \end{aligned} \quad \text{III.6}$$

The yield criteria can now be written as,

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

The Tresca criterion can be reduced to the simple form,

$$(\sigma_1 - \sigma_3) = 2k \quad , \quad \text{III.8}$$

where the maximum and minimum principal stresses are known a priori.

The Von Mises criterion is associated with equation,

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

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,

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

If we define effective stress as

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

then yield criteria may be written as

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

σ_e can also be explicitly written as

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

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

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

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:

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

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,

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

As

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

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

$$\{d\epsilon\}_T = \{D\}^{-1} \{d\sigma\} + d\lambda \frac{\partial F}{\partial \sigma_{ij}} . \quad \text{III.18}$$

From (III.3), we have

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

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

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

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

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

using strain hardening hypothesis.

Differentiating equation (III.19),

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

When plastic yielding is occurring, df is zero. Equation (III.22) can be written as

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

where

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

and

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

Let

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

Premultiplying both sides of equation (III.18) with

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

we get

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

From (III.23),

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

Therefore,

$$\{a\}^T \{D\} \{d\epsilon_{ij}\}_T = d\lambda \left[A \{a\} + \{a\}^T \{D\} \frac{\partial F}{\partial \sigma_{ij}} \right] \quad ,$$

and

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

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

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

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

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

where

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

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,

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

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

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

If we define effective plastic strain increment as

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

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

then together with the relation

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

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 $\sqrt{3}k$,

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

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

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

and

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

CHAPTER IV

MATRIX FORMULATION AND BASIC EXPRESSIONS FOR TWO DIMENSIONAL PROBLEMS

It has been shown¹ 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:

$$\begin{aligned}\sigma_m &= \frac{J_1}{3} = \frac{1}{3} (\sigma_{xx} + \sigma_{yy} + \sigma_{zz}) \\ \sigma_e &= J_2^{1/2} = \left[\frac{1}{2} (S_x^2 + S_y^2 + S_z^2) + \tau_{xy}^2 + \tau_{yz}^2 + \tau_{zx}^2 \right]^{1/2} \\ \phi &= \frac{1}{3} \sin^{-1} \left[\frac{-3\sqrt{3} J_3}{2 \sigma_e^3} \right], \quad -\frac{1}{6}\pi \leq \phi \leq \frac{1}{6}\pi, \quad \text{IV.1}\end{aligned}$$

where,

$$J_3 = \left[S_x S_y S_z + (2\tau_{xy} \tau_{yz} \tau_{zx}) - S_x \tau_{yz}^2 - S_y \tau_{zx}^2 - S_z \tau_{xy}^2 \right]$$

and,

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

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

1. Tresca

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

¹G.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

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

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,

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

From (IV.1),

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

We can now write,

$$\{a\} = c_1 \{a_1\} + c_2 \{a_2\} + c_3 \{a_3\} , \quad \text{IV.6}$$

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,

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

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
$\{D\}$	$\frac{E}{(1-\nu^2)} \begin{bmatrix} 1 & \nu & 0 & 0 \\ \nu & 1 & 0 & 0 \\ 0 & 0 & \frac{(1-\nu)}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & \frac{\nu}{(1-\nu)} & 0 & \frac{\nu}{(1-\nu)} \\ \frac{\nu}{(1-\nu)} & 1 & 0 & \frac{\nu}{(1-\nu)} \\ 0 & 0 & \frac{(1-2\nu)}{2(1-\nu)} & 0 \\ \frac{\nu}{(1-\nu)} & \frac{\nu}{(1-\nu)} & 0 & 1 \end{bmatrix}$
$\{d_D\}$	$\begin{bmatrix} \frac{E}{(1+\nu)} a_1 + E\nu \frac{(a_1+a_2)}{(1-\nu^2)} \\ \frac{E}{(1+\nu)} a_2 + E\nu \frac{(a_1+a_2)}{(1-\nu^2)} \\ Ga_3 \\ \frac{E}{(1+\nu)} a_4 + E\nu \frac{(a_1+a_2)}{(1-\nu^2)} \end{bmatrix}$	$\begin{bmatrix} \frac{E}{(1+\nu)} a_1 + E\nu \frac{(a_1+a_2+a_4)}{(1+\nu)(1-2\nu)} \\ \frac{E}{(1+\nu)} a_2 + E\nu \frac{(a_1+a_2+a_4)}{(1+\nu)(1-2\nu)} \\ Ga_3 \\ \frac{E}{(1+\nu)} a_4 + E\nu \frac{(a_1+a_2+a_4)}{(1+\nu)(1-2\nu)} \end{bmatrix}$

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

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

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 nonlinear strain hardening application. The nonlinear strain hardening program utilizes a total of twenty eight subroutines. The controlling segment calling the subroutines is called **MASIER** for linear strain hardening program, and **MSTER2** for nonlinear strain hardening program. A brief description of each subroutine is presented as follows:

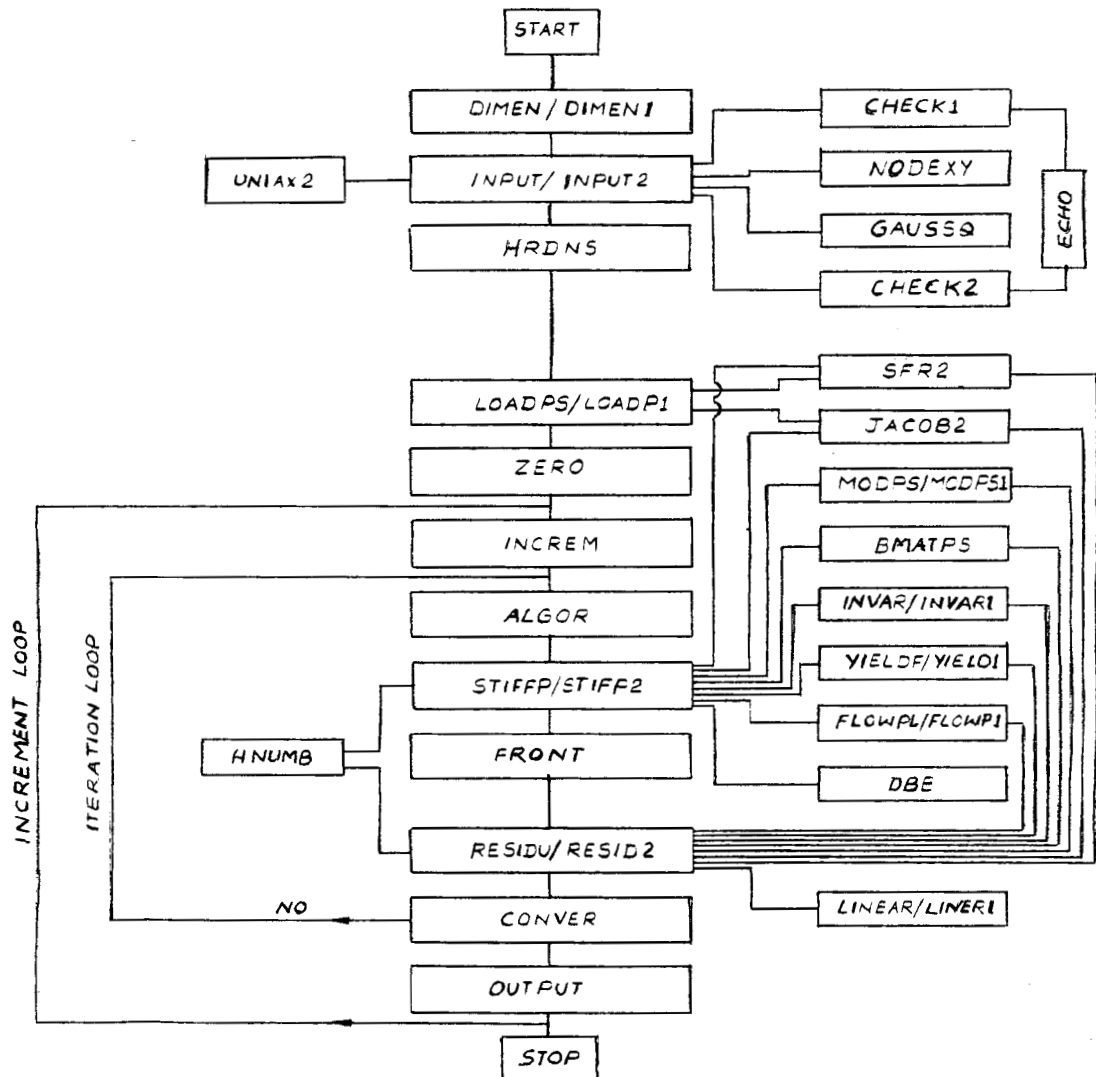


Fig. V.1 Program Outline for Linear and Nonlinear Two Dimensional Elasto-plastic Problems.

Subroutines DIMEN / DIMEN1

These subroutines preset variables to upgrade the magnitude of the maximum 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 situations. 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 / INVARI

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

Subroutines YELDF / YIELD1

These subroutines calculate $\{a\}$ vector. YELDF is for linear program and YIELD1 is for nonlinear program.

Subroutines FLOWL / FLOWP1

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

Subroutines STIFFP / STIFFP2

These subroutines calculate element stiffness matrix $\{K\}$. STIFFP is used in linear strain hardening program and STIFFP2 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,

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

$$\begin{aligned} \sigma^r &= \sigma^{(r-1)} + d\sigma^r , \\ &= \sigma^{(r-1)} + d\sigma_e^r - d\lambda \{d_D\} . \end{aligned} \quad V.3$$

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,

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

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

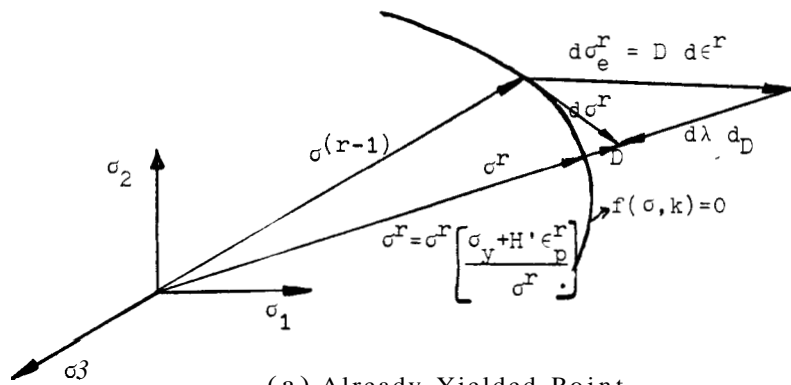
$$\text{number of parts} \leq \left[\frac{\sigma_e^r - \sigma_y}{\sigma_y} \right] 8 + 1 . \quad V.5$$

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

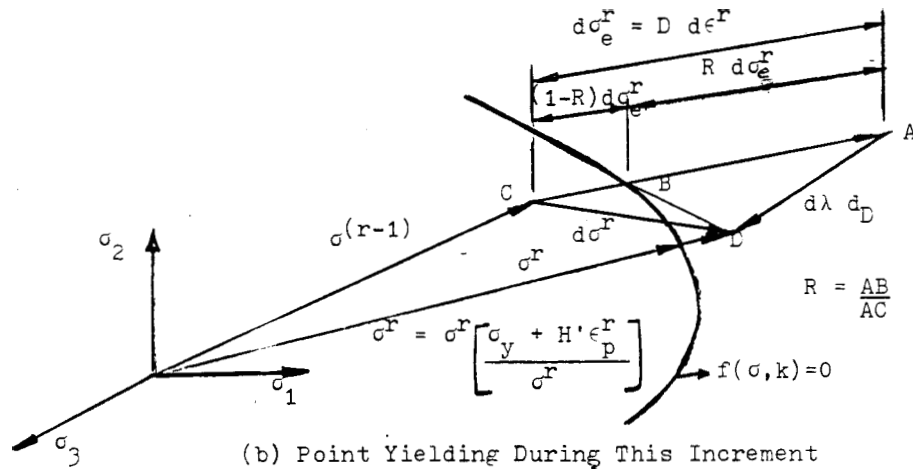
RESIDU is used in linear program and RESID2 in nonlinear program.

Subroutine CONVER

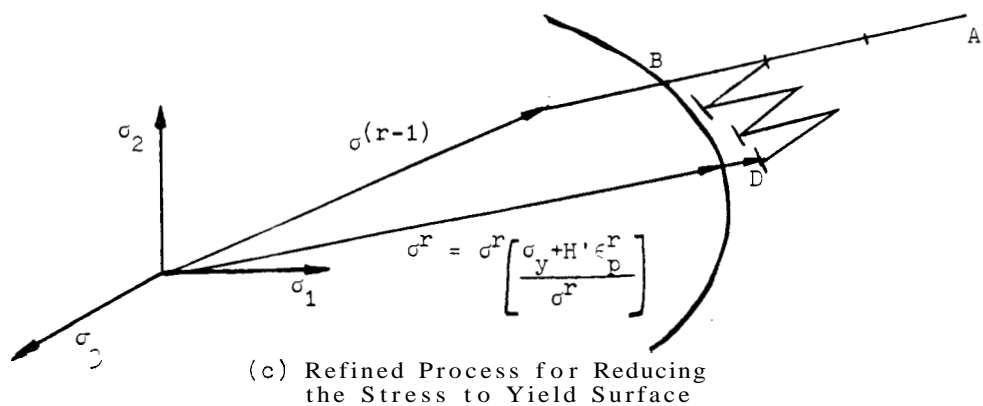
This subroutine is used in both linear and nonlinear programs. This subroutine checks the convergence



(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:

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

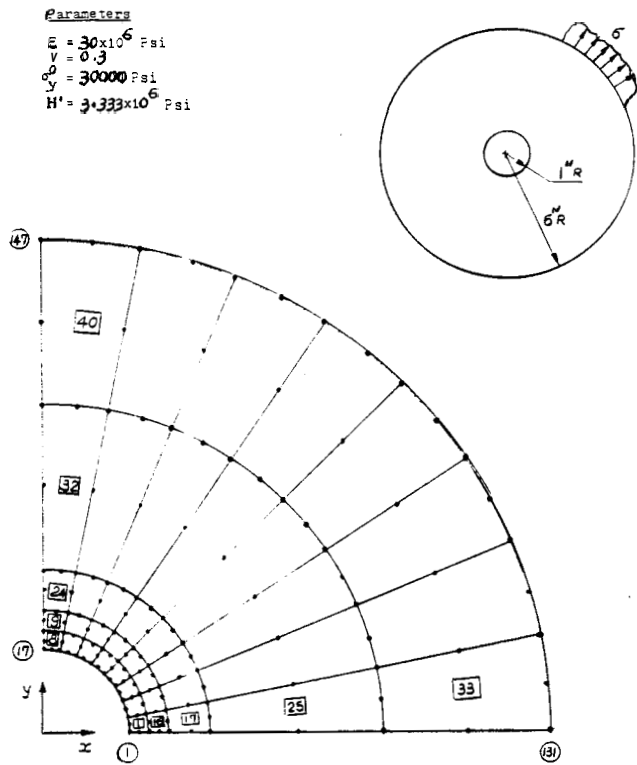


Fig. VI.1 Plane Stress Elasto-plastic Problem of A Circular Hole in A Uniformly Stressed Infinite Plate.

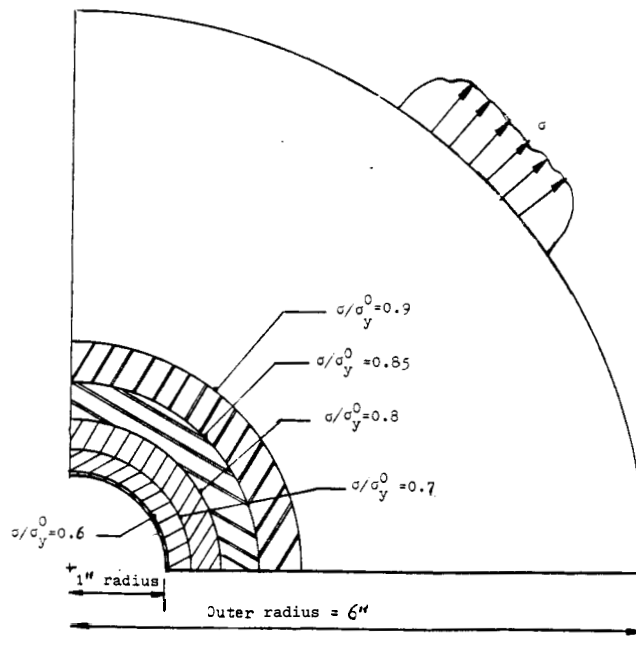


Fig. VI.2 Spread of Plastic Zones at Different Load Levels for Plane Stress Elasto-plastic Problem of A Circular Hole in A Uniformly Stressed Infinite Plate.

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

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

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 reference² in Table VI.2. The stress / strain concentration factors from the program compare very well with the values from reference². 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.

²I.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		
	σ_e @ Hole (ksi)	σ_θ @ Hole (ksi)	Effective Plastic Strain (E.P.S.)	Elastic Strain ($\frac{\sigma_e}{E}$)	Total Strain (E.P.S.+Elastic)	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

^aI.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.

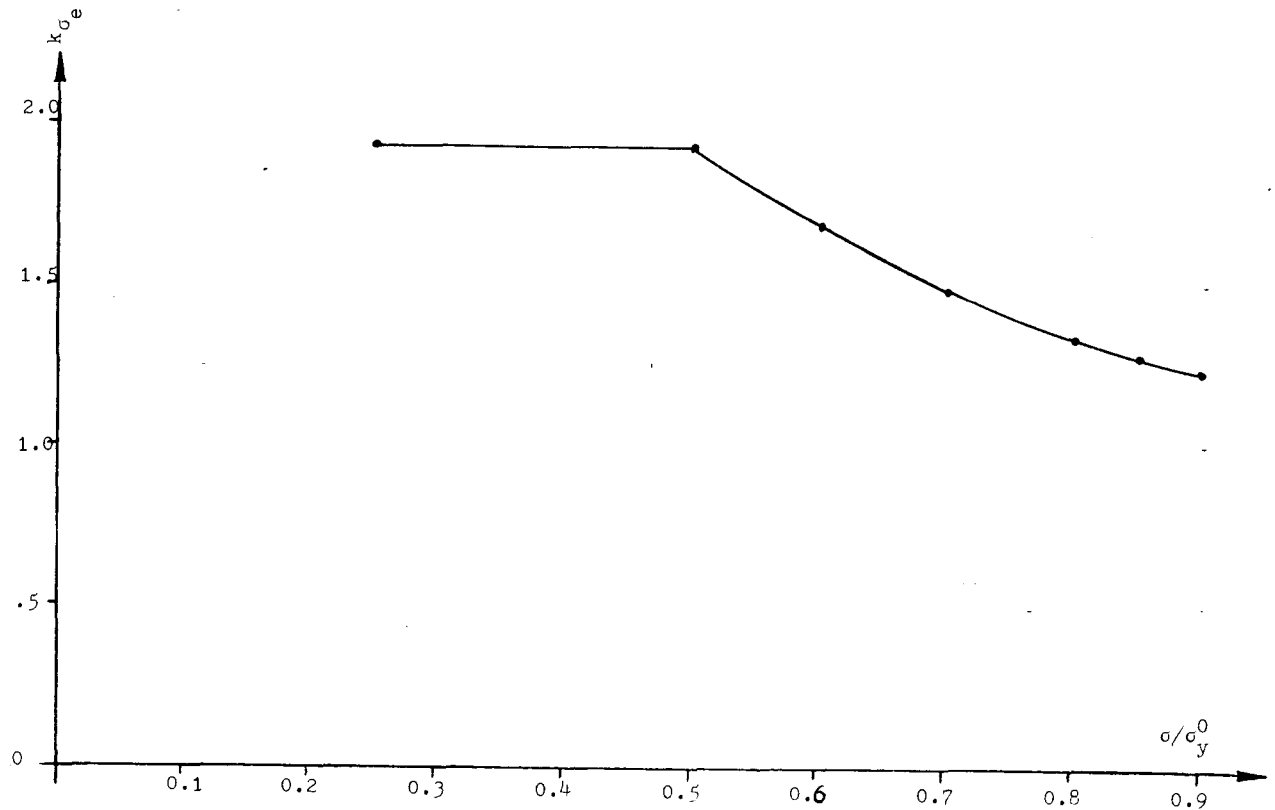


Fig. VI.3 Stress Concentration Factor k_{σ_e} at Different Levels of Applied Loading for Elasto-plastic Problem of a Circular Hole in a Uniformly Stressed Infinite Plate.

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:

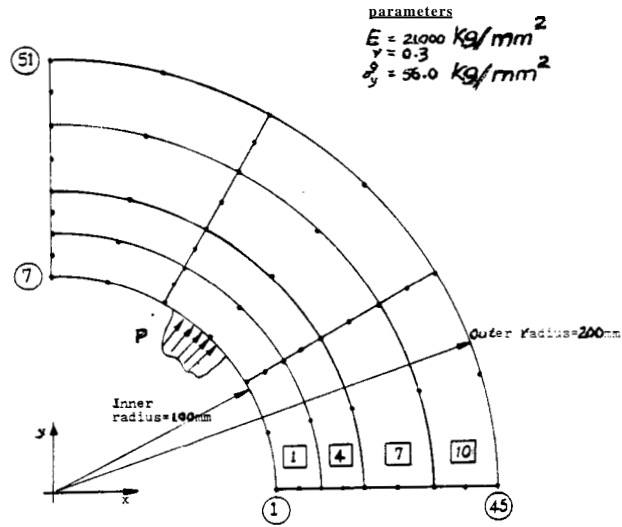


Fig. VI.4 Thick Walled Cylinder Subjected to Internal Pressure.

1. Perfectly Elastic Cylinder.^{3,4}

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

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

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

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

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

⁴Aris 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}

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

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

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

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

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

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

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

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

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

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

⁵Nadai, Plasticity, pp.196-97.

⁶Phillips, Plasticity, pp.184-85.

⁷Phillips, plasticity, pp.186-90.

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

$a \leq r \leq c$
(Plastic Region)

VI.12

$$u_d = \frac{\sqrt{3}}{2} \frac{\sigma_y c^2}{Er} \quad .$$

VI.13

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

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}		$\sigma_{\theta\theta}$	
			(Compressive)		Calculated	Program
			Calculated Values (Kg/mm ²)	Program Output (Kg/mm ²)	Calculated Values (Kg/mm ²)	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}		$\sigma_{\theta\theta}$	
			(Compressive)			
			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/MM² 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		Partly Plastic & Perfect Plasticity		Partly Plastic & $H'=366.279 \text{ Kg/mm}^2$	
		Calculated Values	Program Output	Calculated Values	Program Output	Calculated Values	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

^aEvaluated for a Pressure of 23.35 Kg/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:

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

where

σ_y^0 = Uniaxial yield stress,

H' = Hardness value depending on the total plastic strain at $(r-1)^{\text{th}}$ 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 nonlinear program so as to have,

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

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 nonlinear programs at element 1 , Gauss point-1 are

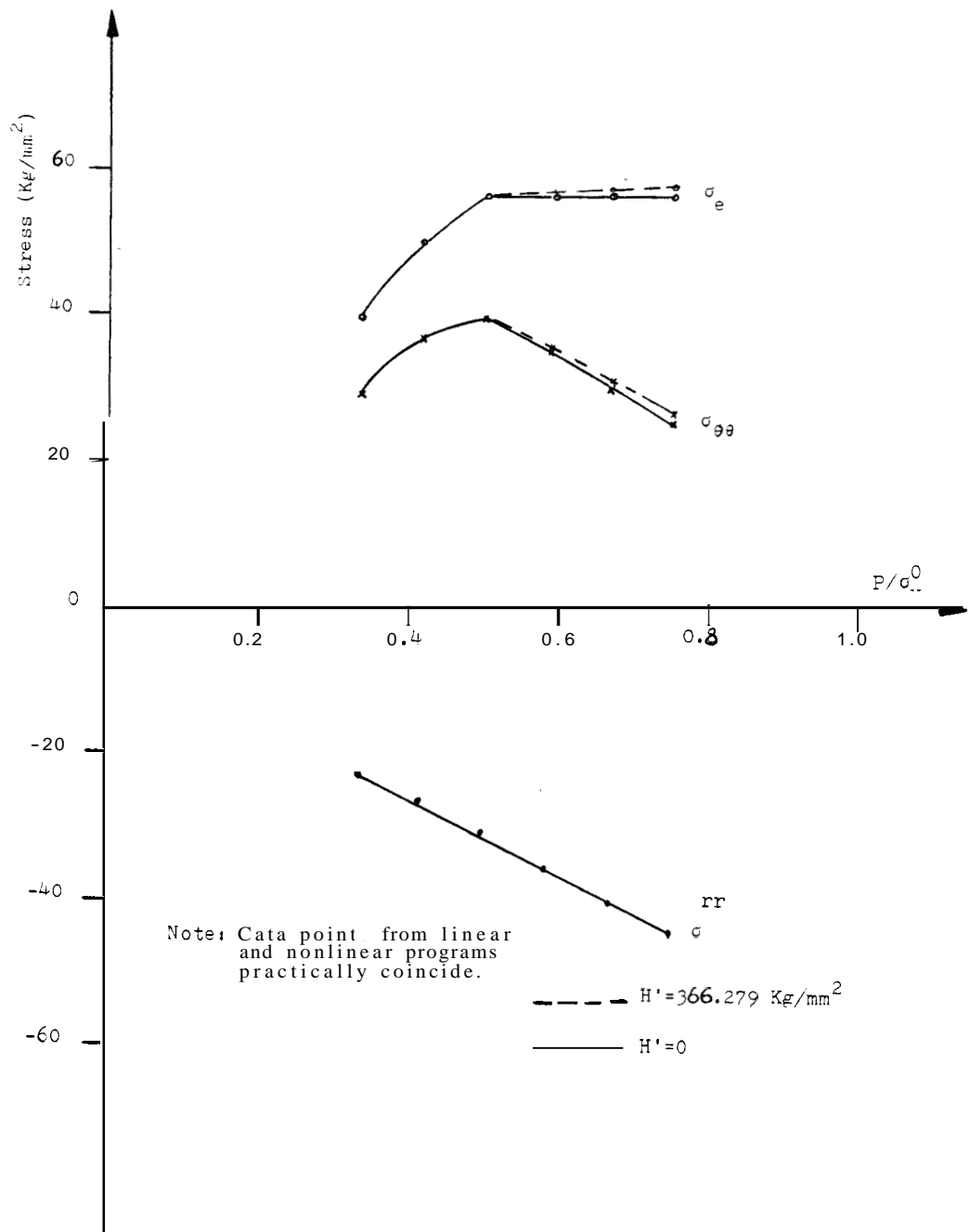


Fig. VI.5 Comparison of Linear and Nonlinear Program Outputs for Stresses for Thick Walled Cylinder Under Internal Pressure.

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

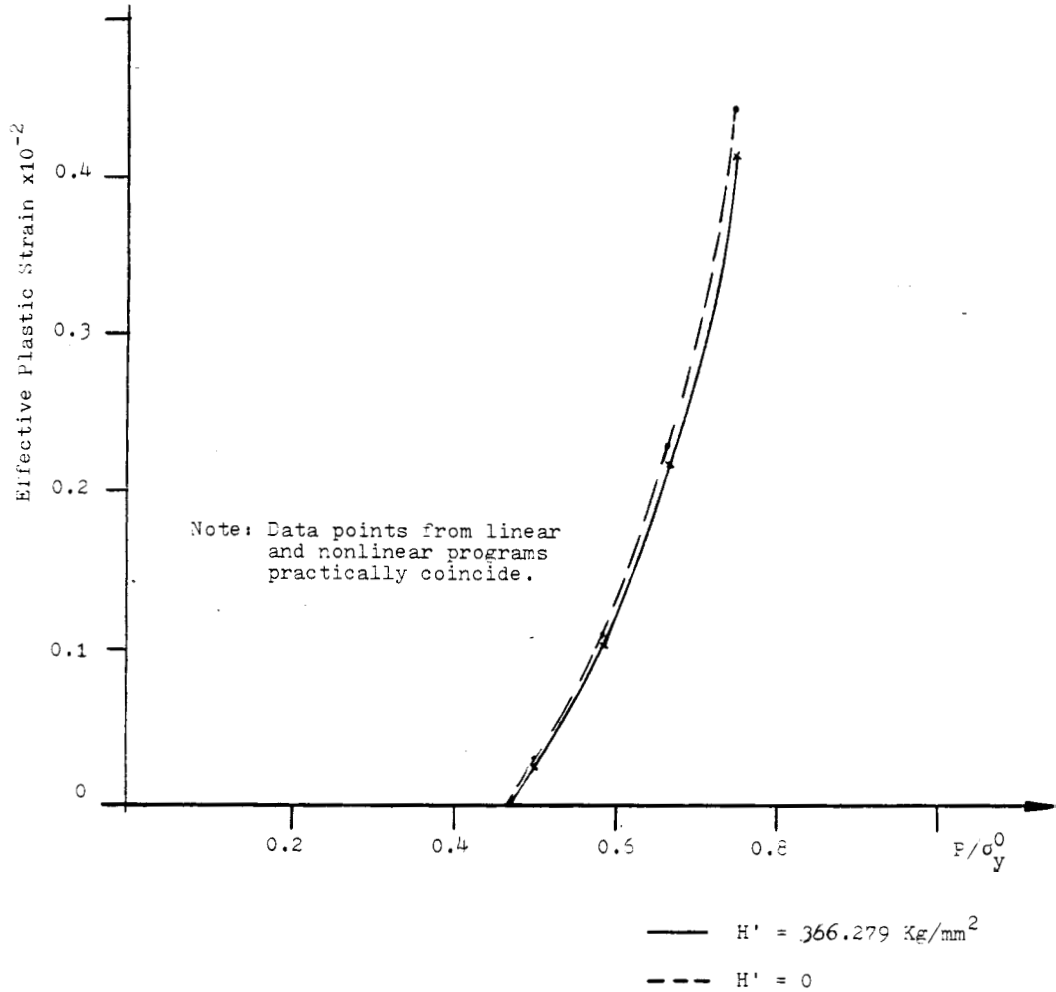


Fig. VI.6 Comparison of Linear and Non-linear Program Outputs for Effective Plastic Strains for Thick Walled Cylinder Under Internal Pressure.

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

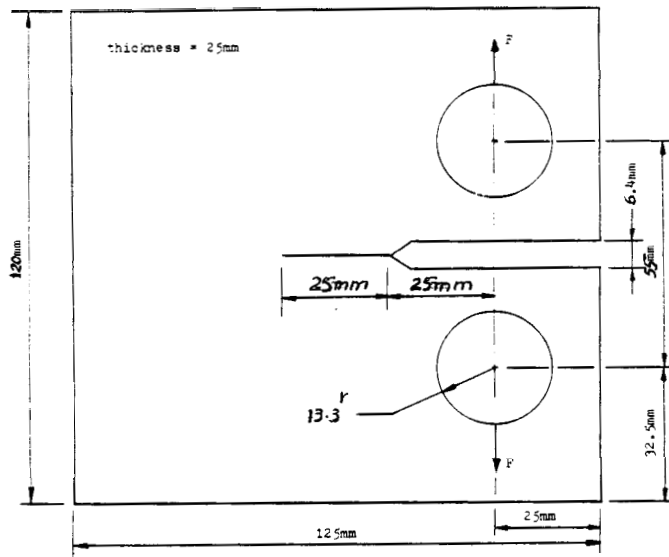


Fig. VII.1 Geometry of Compact Tension (CT) Specimen.

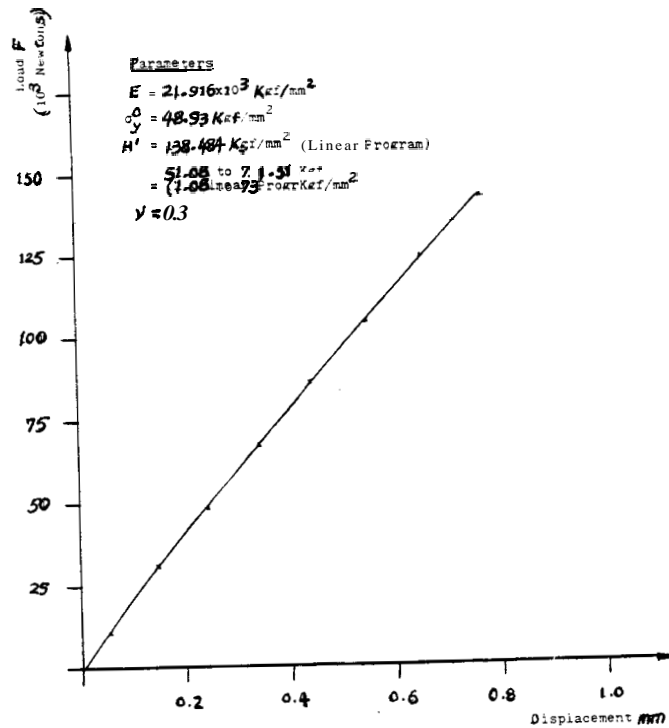
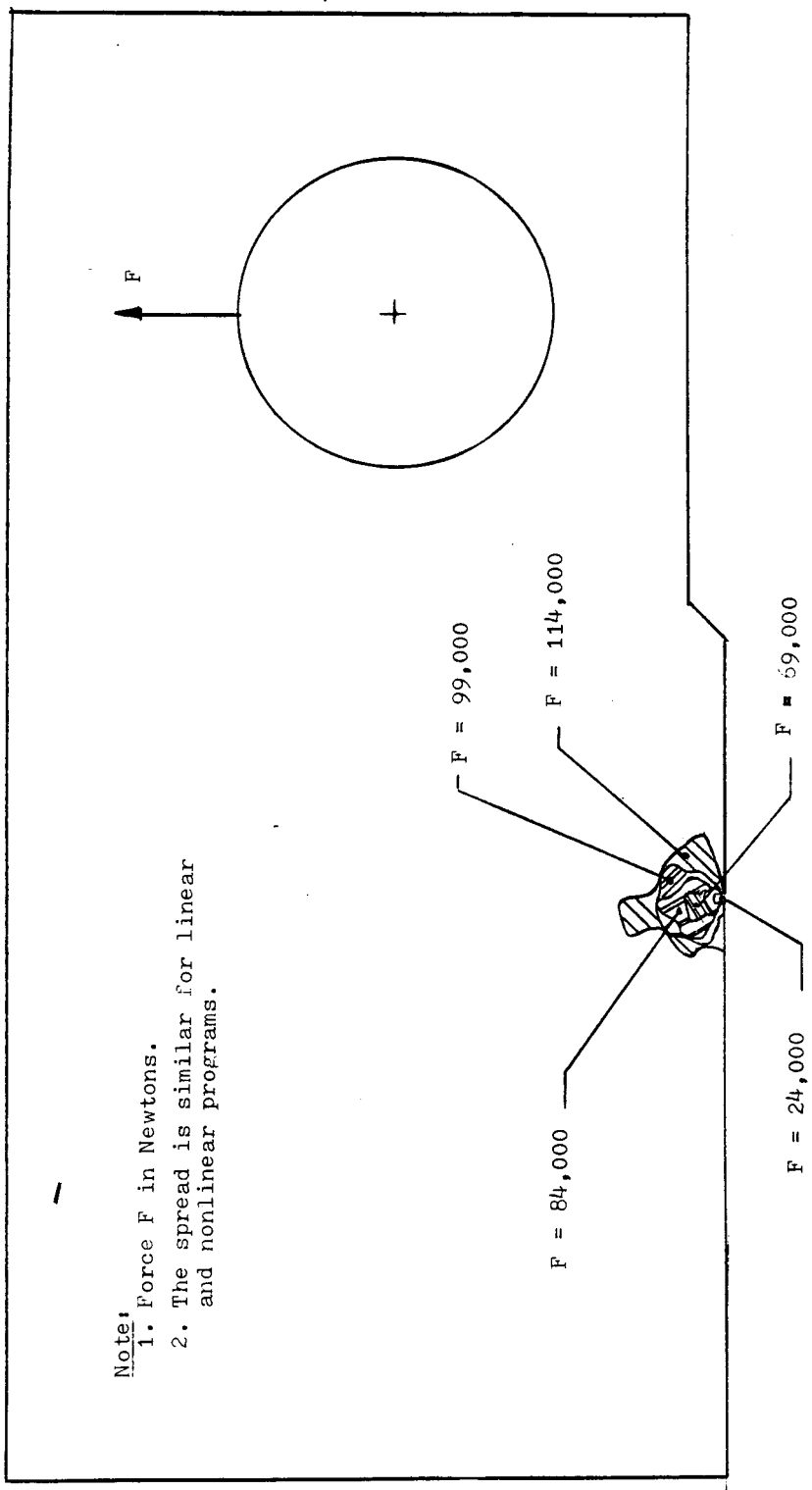


Fig. VII.2 Load vs Load Line Displacement Graph for CT Specimen.



Note:

1. Force F in Newtons.
2. The spread is similar for linear and nonlinear programs.

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 equations⁸,

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

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

and

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

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 / mm² for P_c, for a perfectly plastic material.

The problem was analysed at a pressure of 46.7 Kg/mm² 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/mm² and 377.316 Kg/mm². 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.

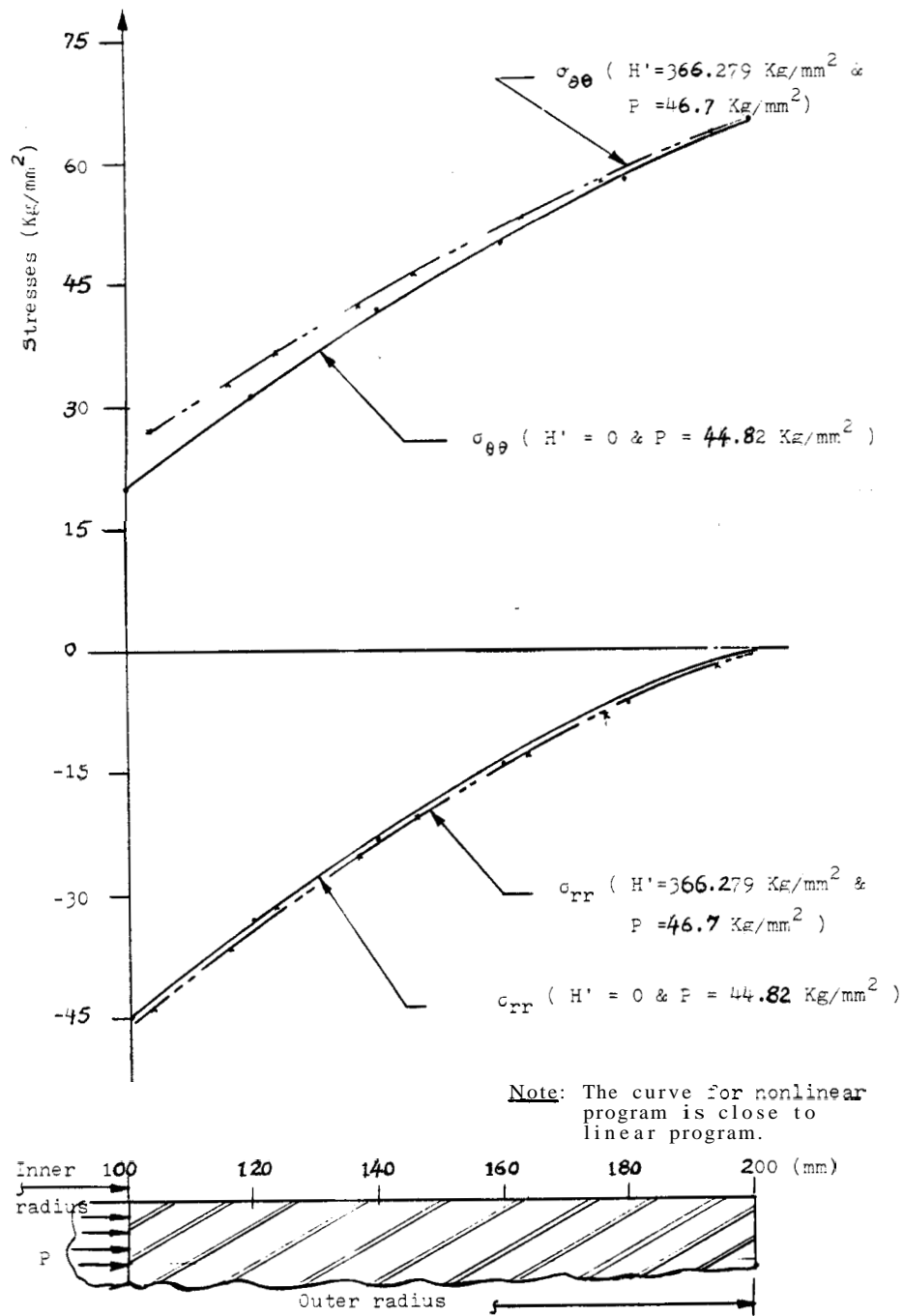


Fig. VII.4 Fully Plastic Cylinder- Comparison of Stresses for Perfectly Plastic Material and Linear Strain Hardening Material.

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 worthwhile for such fully plastic cylinder analysis to limit the analysis to relatively easy perfectly plastic type of material, 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° 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,

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

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.

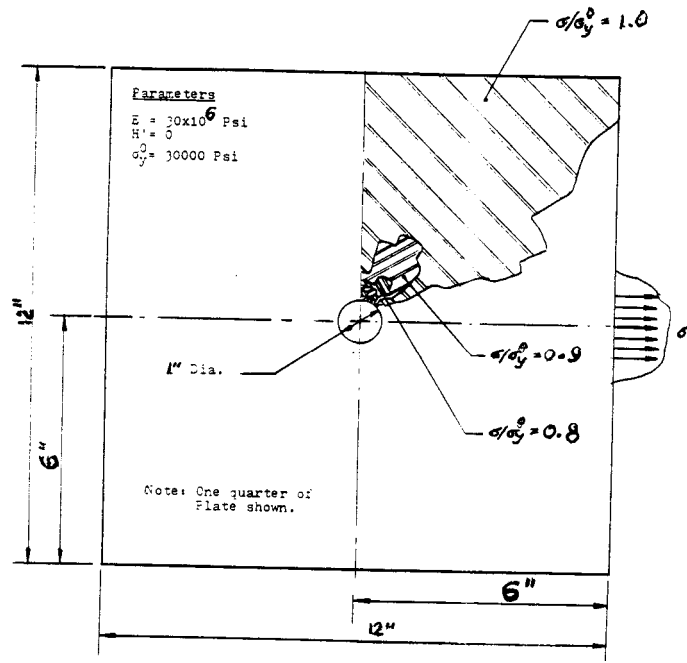


Fig. VII.5 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with a Single Circular Hole .

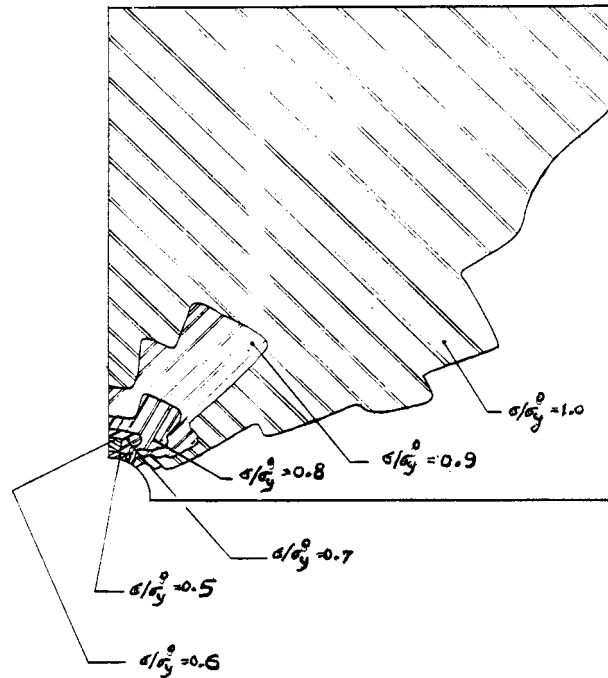


Fig. VII.6 Enlarged Scale Drawing of Spread of Plastic Zone for an Infinite Plate with a Single Circular Hole.

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 $0.95 \sigma_Y^0$, 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 (11.7) 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 $0.85 \sigma_Y^0$, where σ_Y^0 is the uniaxial yield stress.

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

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

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 $k_{\sigma_{\theta}}$ can be evaluated from the expressions (VII.3) and (VII.4) . At σ/σ_Y^0 value of 0.25, when the plasticity has not set in, the calculated $\sigma_{\theta\theta}$ and $k_{\sigma_{\theta}}$ 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 $k_{\sigma_{\theta}}$ for a Gauss point with $r = 0.55253$ and $\theta = 80.53$.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

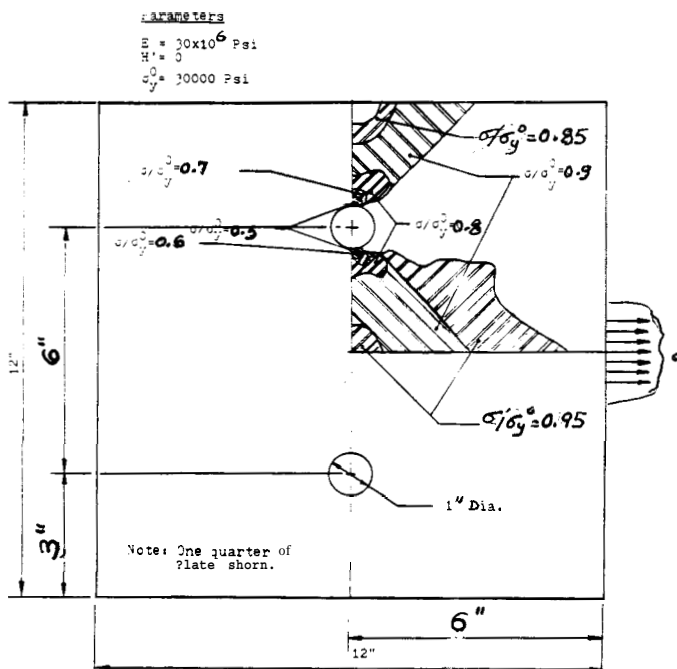


Fig. VII.7 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with Two Circular Holes.

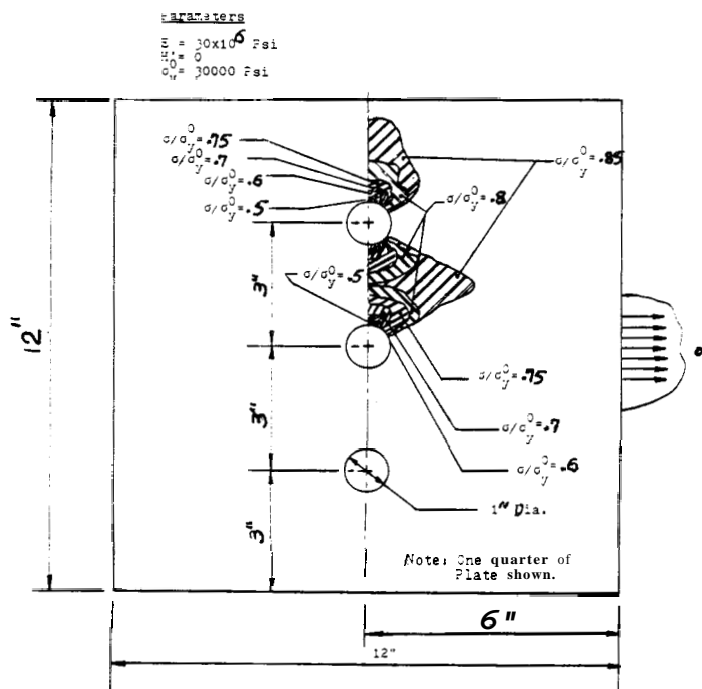


Fig. VII.8 Spread of Plastic Zones at Different Levels of Loading for an Infinite Plate with Three Circular Holes.

TABLE VII.1
 MAXIMUM VALUES OF TANGENTIAL STRESSES AND STRESS
 CONCENTRATION FACTORS FOR A PERFORATED TENSION STRIP.

$\frac{\sigma}{\sigma_y^0}$	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)	$k_{\sigma_{\theta}}$	$\sigma_{\theta\theta}$ (Ksi)	$k_{\sigma_{\theta}}$	$\sigma_{\theta\theta}$	$k_{\sigma_{\theta}}$
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 Fig. 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.

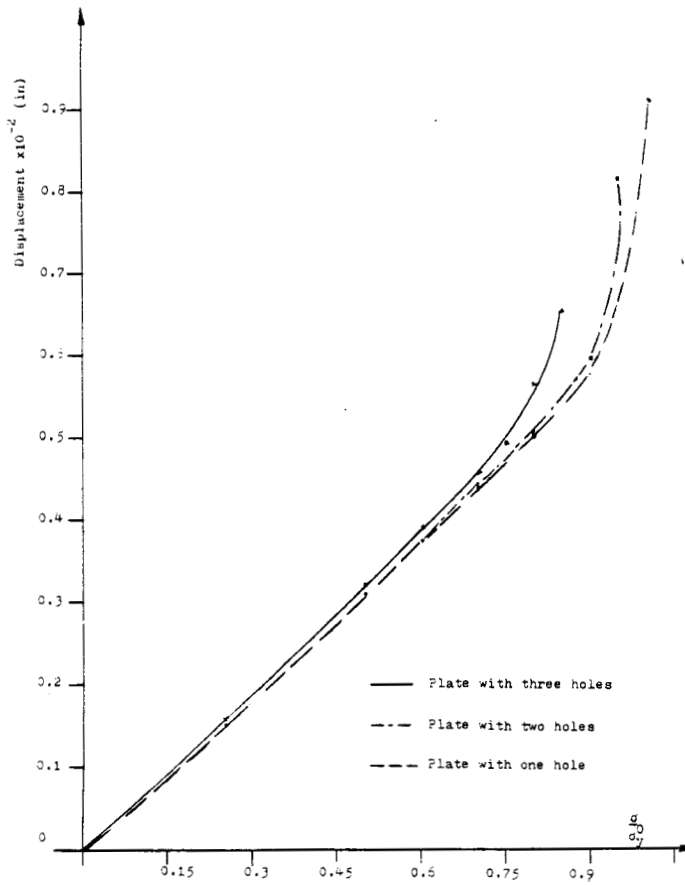


Fig. VII.9 Displacement Values at Different Levels of Loading for a Perforated Tension Strip.

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 accommodate, 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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
1-72	--	Title of the problem.

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

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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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.
21-25	LNODS(NUMEL,3)	3rd nodal connection number.
26-30	LNODS(NUMEL,4)	4th nodal connection number.
31-35	LNODS(NUMEL,5)	5th nodal connection number.
36-40	LNODS(NUMEL,6)	6th nodal connection number.
41-45	LNODS(NUMEL,7)	7th nodal connection number.
46-50	LNODS(NUMEL,8)	8th 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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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 restricted to y(or z) direction only.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
		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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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)	Poisson'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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
1-72	--	Title of the load case.

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

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
1-5	IPLOD	Applied concentrated load control parameter:

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
6-10	IGRAV	0-No concentrated loads. 1-Concentrated loads applied. Gravity loading control parameter:
11-15	IEDGE	0-No gravity loading. 1-Gravity loads to be input. Distributed edge loading control parameter 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 load nodal point.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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.

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

ELEMENT FACE TOPOLOGY CARD(4I5)

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

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

DISTRIBUTED LOAD CARDS(6F10.3)

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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 component 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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
26-30	NOUIP(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	NOUIP(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 FACT0 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.

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

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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^0 (or cohesion factor for Y 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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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.

<u>Columns</u>	<u>Variables</u>	<u>Entry</u>
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 NMB 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


```

DO 50 IITER = 1,MITER
C
C*** CALL ROUTINE WHICH SELECTS SOLUTION ALGORITHM VARIABLE KRESL
CALL ALGOR(FIXED,IINCS,IITER,KRESL,MTOTV,NALGO,
          NTOTV)
C
C*** CHECK WHETHER A NEW EVALUATION OF THE STIFFNESS MATRIX IS REQUIRED
IF (KRESL.EQ.1) CALL STIFF(COORD,EPSTN,IINCS,LNOOS,MATNO,
          MEVAB,MMAIS,MPOIN,MTOTV,NELEM,NEVAR,NGAUS,NNODE,
          NSTRE,NSTRI,POSGP,PROPS,WEIGP,MELEM,MTOTG,
          STRSG,NTYPE,NCRIT)
C
C*** SOLVE EQUATIONS
CALL FRONT(ASDIS,ELoad,EQRHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS,IITER,
          GLOAD,GSTIF,LOCFL,LNOOS,KRESL,MBOFA,MELEM,MEVAB,MFRON,
          MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDCST,NDDEN,NELEM,NEVAR,
          NNODE,NOFIX,NPIVO,NPOIN,NTOTV,TBESP,TLBAB,TREAC,
          VECRV)
C
C*** CALCULATE RESIDUAL FORCES
CALL RESIDU(ASDIS,COORD,EFFST,FLOAD,FACTO,IITER,LNDD,
          LPROP,MATNO,MELEM,MMAIS,MPOIN,MTOTG,MTOT,NOFN,
          LPROP,MATNO,MELEM,MMAIS,NBXY,NTIFF,MTOTV,NDOFN,
          NELEM,NEVAR,NGRUS,NNOGP,TDISP,EPSTN,POSGP,PROPS,
          NSTRE,NCRIT,STRSG,WEIGP,TDISP,FP)
C
C*** CHECK FOR CONVERGENCE
CALL CONVER(ELoad,IITER,LNOOS,MELEM,MEVAB,MTOTV,NCHEK,NDOFN,
          NFIX,NEVAR,NNODE,NTOTV,PVALU,STFOR,TLoad,TOFOR,TOLER)
C
C*** OUTPUT RESULTS IF REQUIRED
IF (IITER.EQ.1.AND.NOUTP(1).GT.0)
  CALL OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
            MPOIN,MVFIX,STRSG,TDISP,TREAC,EPSTN,NTYPE,NCHEK,
            EPSTN)
C
C*** IF SOLUTION HAS CONVERGED STOP ITERATING AND OUTPUT RESULTS
IF (NCHEK.EQ.0) GO TO 75
50 CONTINUE
C
C***
IF (NALGO.EQ.2) GO TO 75
75 STOP
CALL OUTPUT(IITER,MTOTG,MTOTV,MVFIX,NELEM,NGAUS,NOFIX,NOUTP,
          MPOIN,MVFIX,STRSG,TDISP,TREAC,EPSTN,NTYPE,NCHEK,
          NPBSN)
100 CONTINUE
STOP
END

```

```

MAS00560
MAS00570
MAS00580
MAS00590
MAS00600
MAS00610
MAS00620
MAS00630
MAS00640
MAS00650
MAS00660
MAS00670
MAS00680
MAS00690
MAS00700
MAS00710
MAS00720
MAS00730
MAS00740
MAS00750
MAS00760
MAS00770
MAS00780
MAS00790
MAS00800
MAS00810
MAS00820
MAS00830
MAS00840
MAS00860
MAS00870
MAS00880
MAS00890
MAS00900
MAS00910
MAS00920
MAS00930
MAS00940
MAS00950
MAS00960
MAS00970
MAS00980
MAS00990
MAS01000
MAS01010
MAS01020
MAS01030
MAS01040
MAS01050
MAS01060
MAS01070
MAS01080
MAS01090
MAS01100
MAS01110
MAS01120

```

```

SUBROUTINE DIMEN(MBUFA,MELEM,MEVAB,MFRON,MMATS,MPDIN,MSTIF,MTOTG, DIM00010
HTOTV,MVFIX,NDOFN,NPROP,NSTRE) DIM00020
C***** DIM00030
C*** THIS SUBROUTINE PRESETS VARIABLES ASSOCIATED WITH DYNAMIC DIM00040
DIMENSIONING DIM00050
C***** DIM00060
C***** DIM00070
IMPLICIT REAL*8(A-H,O-Z) DIM00080
MBUFA = 10 DIM00090
MELEM = 50 DIM00100
MFRON = 80 DIM00110
MMATS = 5 DIM00120
HPIIN = 180 DIM00130
MSTIF = (MFRON*MFRON-MFRON)/2.0*MFRON DIM00140
MTOTG = MELEM*9 DIM00150
NDOFN = 2 DIM00160
HTOTV = MPDIN*NDOFN DIM00170
MVFIX = 30 DIM00180
NPROP = 7 DIM00190
MEVAB = NDOFN*9 DIM00200
RETURN DIM00210
END DIM00220
DIM00230

SUBROUTINE INPUT(COORD,IFFIX,LNODS,MATNO,MELEM,MEVAB,MFRON,MMATS, INP00010
MPDIN,HTOTV,MVFIX,NALGO, INP00020
NCRIT,NDFRO,NDOFN,NELEM, INP00030
NEVAB,NGAUS,NGAU2, INP00040
NINCS,NMATS,NNODE,NDFIX,NPOIN,NPROP,NSTRE,NSTR1, INP00050
NTOTG,NTOTV,NTYPE,NVFIX,POSGP,PRESC,PROPS,WEIGP) INP00060
C***** INP00070
C*** THIS SUBROUTINE ACCEPTS MOST OF THE INPUT DATA INP00080
C***** INP00090
IMPLICIT REAL*8(A-H,O-Z) INP00100
DIMENSION COORD(MPDIN,2),IFFIX(HTOTV),LNODS(MELEM,9), INP00110
MATNO(MELEM),NDFRO(MELEM), INP00120
NDFIX(MVFIX),POSGP(4),PRESC(MVFIX,NDOFN), INP00130
NDFIX(MVFIX),POSGP(4),PRESC(MVFIX,NDC), INP00140
PROPS(MMATS,NPROP),TITLE(18),WEIGP(4) INP00150
REWIND 1 INP00160
REWIND 2 INP00170
REWIND 3 INP00180
REWIND 4 INP00190
REWIND 8 INP00200
READ(5,920) TITLE INP00210
WRITE(7,920) TITLE INP00220
920 FORMAT(18A4) INP00230
C*** READ THE FIRST DATA CARD AND ECHO IT IMMEDIATELY INP00240
READ(5,900) NPOIN,NELEM,NVFIX,NTYPE,NNODE,NMATS,NGAUS, INP00250
NALGO,NCRIT,NINCS,NSTRE INP00260
900 FORMAT(11I5) INP00270
NEVAB = NDOFN*NNODE INP00280
NSTR1 = NSTRE+1 INP00290
IF (NTYPE.EQ.3) NSTR1 = NSTRE INP00300
NTOTV = NPOIN*NDOFN INP00310
NGAU2 = NGAUS+NGAUS INP00320
NTOTG = NELEM*NGAU2 INP00330
WRITE(6,901) NPOIN,NELEM,NVFIX,NTYPE,NNODE,NMATS,NGAUS,MEVAB, INP00340
NALGO,NCRIT,NINCS,NSTRE INP00350
901 FORMAT(//8H NPOIN =,I4,4X,8H NELEM =,I4,4X,8H NVFIX =,I4,4X, INP00360
8H NTYPE =,I4,4X,8H NNODE =,I4,4X, INP00370
8H NMATS =,I4,4X,8H NGAUS =,I4,4X, INP00380
4X,8H NEVAB =,I4,4X,8H NALGO =,I4,4X, INP00390
8H NCRIT =,I4,4X,8H NINCS =,I4,4X,8H NSTRE =,I4) INP00400
CALL CHECK1(NDOFN,NELEM,NGAUS,NMATS,NNODE,NPOIN, INP00410
NSTRE,NTYPE,NVFIX,NCRIT,NALGO,NINCS) INP00420
C*** READ THE ELEMENT NODAL CONNECTIONS,AND THE PROPERTY NUMBERS INP00430
C***** INP00440
WRITE(6,902) INP00450
902 FORMAT(//8H ELEMENT,3X,8HPROPERTY,6X,12HNODE NUMBERS) INP00460
DO 2 IELEM = 1,NELEM INP00470
READ(5,900) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE) INP00480
2 WRITE(6,903) NUMEL,MATNO(NUMEL),(LNODS(NUMEL,INODE),INODE=1,NNODE) INP00490
903 FORMAT(1X,15,19,6X,815) INP00500
C***** INP00510
C***** INP00520
C***** INP00530
C***** INP00540
C***** INP00550

```

```

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
      904 FORMAT(775H NODE,10X,1HX,10X,1HY)
      6 READ(5,905) IPOIN,(COORD(IPOIN, IDIME), IDIME=1,2)
      905 FORMAT(15,6F10.5)
      IF (IPOIN.NE.NPOIN) GO TO 6
C
C*** INTERPOLATE COORDINATES OF MID-SIDE NODES
C
      CALL NDOFX(COORD,LNODS,MELEM,MPOIN,NELEM,NNUDE)
      DO 10 IPOIN = 1,NPOIN
      10 WRITE(6,906) IPOIN,(COORD(IPOIN, IDIME), IDIME=1,2)
      906 FORMAT(1X,15,3F10.3)
C
C*** READ THE FIXED VALUES
C
      WRITE(6,907)
      907 FORMAT(775H NODE,6X,4HCODE,6X,12HFIXED VALUES)
      DO 8 IVPFX = 1,NVPFX
      READ(5,908) NDFIX(IVPFX),IFPRE,(PRESC(IVFIX, IDOFN), IDOFN=1,NDOFN)
      WRITE(6,908) NDFIX(IVFIX),IFPRE,(PRESC(IVFIX, IDOFN), IDOFN=1,NDOFN)
      NLUCA = (NDFIX(IVFIX)-1)*NDOFN
      IDOFF = 10** (NDOFN-1)
      DO 8 IDOFN = 1,NDOFN
      NGASH = NLUCA+IDOFN
      IF(IFPRE.LT.IDOFF) GO TO 8
      IFFIX(NGASH)=1
      IFFIX(NGASH)=1
      IFPRE=IFPRE-IDOFF
      IDOFF = IDOFF/10
      8
      908 FORMAT(1X,14,5X,15,5X,5F10.6)
C
C*** READ THE AVAILABLE SELECTION OF ELEMENT PROPERTIES
C
      16 WRITE(6,910)
      910 FORMAT(777H 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,14,3X,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,
      * NVPFX, NDFRO, NDOFN, NELEM, NMATS, NNUDE, NDFIX, NPOIN,
      * NVPFX)
      RETURN
      ENO

```

```

INP00560
INP00570
INP00580
INP00590
INP00600
INP00610
INP00620
INP00630
INP00640
INP00650
INP00660
INP00670
INP00680
INP00690
INP00700
INP00710
INP00720
INP00730
INP00740
INP00750
INP00760
INP00770
INP00780
INP00790
INP00800
INP00810
INP00820
INP00830
INP00840
INP00850
INP00860
INP00870
INP00880
INP00890
INP00900
INP00920
INP00930
INP00940
INP00950
INP00960
INP00970
INP00980
INP00990
INP01000
INP01010
INP01020
INP01030
INP01040
INP01050
INP01060
INP01070
INP01080
INP01090
INP01100
INP01110
INP01120

```

```

SUBROUTINE CHECK1(NDOFN,NELEM,NGAUS,NMATS,NNODE,NPOIN,
NSTR,NLROH,NVFIX,NCRIT,NALGO,NINCS)
C*****
C*** THIS SUBROUTINE CHECKS THE MAIN CONTROL DATA
C*****
      IMPLICIT REAL*4(A-H,O-Z)
      DIMENSION NLROH(24)
      DO 10 IERROR = 1,12
10  NERROR(IERROR)=0
C*** CREATE THE DIAGNOSTIC MESSAGES
C
      IF (NPOIN.LT.0) NERROR(1) = 1
      IF (NELEM.NNODE.LT.NPOIN) NERROR(2) = 1
      IF (NVFIX.LT.2.OR.NVFIX.GT.NPOIN) NERROR(3) = 1
      IF (NINCS.LT.1) NERROR(4) = 1
      IF (NTYPE.LT.1.OR.NTYPE.GT.3) NERROR(5)=1
      IF (NNODE.LT.4.OR.NNODE.GT.9) NERROR(6) = 1
      IF (NDOFN.LT.2.OR.NDOFN.GT.5) NERROR(7) = 1
      IF (NMATS.LT.1.OR.NMATS.GT.NELM) NERROR(8) = 1
      IF (NGAUS.LT.2.OR.NGAUS.GT.3) NERROR(10) = 1
      IF (NALGO.LT.1.OR.NALGO.GT.4) NERROR(11) = 1
      IF (NSTR.LT.3.OR.NSTR.GT.5) NERROR(12) = 1
C*** EITHER RETURN, OR ELSE PRINT THE ERRORS DIAGNOSED
C
      KERROR = 0
      DO 20 IERROR = 1,12
      IF (NERROR(IERROR).EQ.0) GO TO 20
      KERROR = 1
      WRITE(6,900) IERROR
900  FORMAT('31H *** DIAGNOSIS BY CHECK1, ERROR,13)
      CONTINUE
20  IF (KERROR.EQ.0) RETURN
C*** OTHERWISE ECHO ALL THE REMAINING DATA WITHOUT FURTHER COMMENT
C
      CALL ECHO
      END

```

```

SUBROUTINE GAUSSQ(NGAUS,POSGP,WEIGP)
C*****
C*** THIS SUBROUTINE SETS UP THE GAUSS-LEGENDRE INTEGRATION CONSTANTS
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 5
4  POSGP(1)=-0.774596669241483
   POSGP(2) = 0.0
   WEIGP(1) = 0.5555555555555556
   WEIGP(2) = 0.8888888888888889
5  KGAUS = NGAUS/2
   DO # 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*** THIS SUBROUTINE INTERPOLATES THE MID SIDE NODES OF STRAIGHT
C    SIDES OF ELEMENTS AND THE CENTRAL NODE OF 9 NODED ELEMENTS
C*****
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION COORD(MPOIN,2),LNODS(MELEM,9)
      IF (NNODE.EQ.4) RETURN
C*** LOOP OVER EACH ELEMENT
      DO 30 IELEM = 1,NELEM
C*** LOOP OVER EACH ELEMENT EDGE
      NNOD1 = 9
      IF (NNODE.EQ.8) NNOD1 = 7
      DO 20 INODE = 1,NNOD1,2
      IF (INODE.EQ.9) GO TO 50
C*** COMPUTE THE NODE NUMBER OF THE FIRST NODE
      NODST = LNODS(IELEM,INODE)
      IGASH = INODE+2
      IF (IGASH.GT.8) IGASH = 1
C*** COMPUTE THE NODE NUMBER OF THE LAST NODE
      NODFN = LNODS(IELEM,IGASH)
      MIDPT = INODE+1
C*** COMPUTE THE NODE NUMBER OF THE INTERMEDIATE NODE
      NODMD = LNODS(IELEM,MIDPT)
      TOTAL = DABS(COORD(NODMD,1))+DABS(COORD(NODMD,2))
C*** IF THE COORDINATES OF THE INTERMEDIATE NODE ARE BOTH ZERO
C    INTERPOLATE BY A STRAIGHT LINE
      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)

40  KOUNT = 1
      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
N0000010
N0000020
N0000030
N0000040
N0000050
N0000060
N0000070
N0000080
N0000090
N0000100
N0000110
N0000120
N0000130
N0000140
N0000150
N0000160
N0000170
N0000180
N0000190
N0000200
N0000210
N0000220
N0000230
N0000240
N0000250
N0000260
N0000270
N0000280
N0000290
N0000300
N0000310
N0000320
N0000330
N0000340
N0000350
N0000360
N0000370
N0000380
N0000390
N0000400
N0000410
N0000420
N0000430
N0000440
N0000450
N0000460
N0000470
N0000480
N0000490
N0000500
N0000510
N0000520
N0000530
N0000540
N0000550
N0000560
N0000570
N0000580
N0000590
N3000600
N0000610
N0000620
N0000630
N0000640

```

```

SUBROUTINE CHECK2(COORD, IFFIX, LNODS, MATNO, MELEM, NFRON, MPOIN, MTOTV, CHE00010
: MVMIX, IFFIX, LNODS, MATNO, MELEM, NFRON, MPOIN, MTOTV, CHE00020
: NVMIX, NDFRO, NDOFN, NELEM, NMATS, NNODE, NOFIX, NPOIN, CHE00030
: ***** CHE00040
C*** THIS SUBROUTINE CHECKS THE REMAINDER OF THE INPUT DATA CHE00050
: ***** CHE00070
C IMPLICIT REAL*8(A-H,O-Z) CHE00080
: ***** CHE00090
C DIMENSION COORD(MPOIN,2), IFFIX(MTOTV), LNODS(MELEM,9), CHE00100
: MATNO(MELEM), NDFRO(MELEM), NEROR(24), NOFIX(MVMIX) CHE00110
C*** CHECK AGAINST TWO IDENTICAL NONZERO NODAL COORDINATES CHE00120
: ***** CHE00130
C DO 5 IEROR = 13,24 CHE00140
5 NEROR(IEROR) = 0 CHE00150
DO 10 IFLEM = 1,NELEM CHE00160
10 NDFRO(IFLEM) = 0 CHE00170
DO 40 IPOIN = 2,NPOIN CHE00180
KPUIN = IPOIN-1 CHE00190
DO 40 JPOIN = 1,KPOIN CHE00200
DO 20 IDIME = 1,2 CHE00210
IF (COORD(IPOIN, IDIME) .NE. COORD(JPOIN, IDIME)) GO TO 30 CHE00220
CONTINUE CHE00230
NEROR(13) = NEROR(13)+1 CHE00240
30 CONTINUE CHE00250
40 CONTINUE CHE00260
C*** CHECK THE LIST OF ELEMENT PROPERTY NUMHEHS CHE00270
: ***** CHE00280
C DO 50 IFLEM = 1,NELEM CHE00290
50 IF (MATNO(IFLEM) .LE. 0 .OR. MATNO(IFLEM) .GT. NMATS) NEROR(14) = CHE00300
: NEROR(14)+1 CHE00310
: ***** CHE00320
C*** CHECK FOR IMPOSSIBLE NOOE NUMBERS CHE00330
: ***** CHE00340
C DO 70 IFLEM = 1,NELEM CHE00350
DO 60 INODE = 1,NNODE CHE00360
IF (LNODS(IFLEM, INODE) .EQ. 0) NEROR(15) = NEROR(15)+1 CHE00370
60 IF (LNODS(IFLEM, INODE) .LT. 0 .OR. LNODS(IFLEM, INODE) .GT. NPOIN) CHE00380
: NEROR(16) = NEROR(16)+1 CHE00390
70 CONTINUE CHE00400
C*** CHECK FOR ANY REPETITION OF A NODE NUMBER WITHIN AN ELEMENT CHE00410
: ***** CHE00420
C DO 140 IPOIN = 1,NPOIN CHE00430
KSTAR = 0 CHE00440
DO 100 IFLEM = 1,NELEM CHE00450
KZERO = 0 CHE00460
DO 70 INODE = 1,NNODE CHE00470
IF (LNODS(IFLEM, INODE) .NE. IPOIN) GO TO 90 CHE00480
KZERO = KZERO+1 CHE00490
IF (KZERO .GT. 1) NEROR(17) = NEROR(17)+1 CHE00500
90 CONTINUE CHE00510
C*** SEEK FIRST, LAST, AND INTERMEDIATE APPEARANCES OF NODE CHE00520
: ***** CHE00530
: ***** CHE00540
: ***** CHE00550

```

```

C      IF(KSTAR.NE.0) GO TO 80
C      KSTAR = IELEM
C      CHE00560
C      CHE00570
C      CHE00580
C*** CALCULATE INCREASE OR DECREASE IN FRONTWIDTH AT EACH ELEMENT STAGE
C      CHE00590
C      CHE00600
00      NDFRO(IELEM) = NDFRO(IELEM)+NDOFN
C      CHE00610
C      CHE00620
C      CHE00630
C      CHE00640
C*** AND CHANGE THE SIGN OF THE LAST APPEARANCE OF EACH NODE
C      CHE00650
C      CHE00660
C      CHE00670
C      CHE00680
C      CHE00690
C      CHE00700
C      CHE00710
C      CHE00720
C      CHE00730
C      CHE00740
C      CHE00750
C      CHE00760
C      CHE00770
C      CHE00780
C      CHE00790
C      CHE00800
C      CHE00810
C      CHE00820
C      CHE00830
C      CHE00840
C      CHE00850
C      CHE00860
C      CHE00870
C      CHE00880
C      CHE00890
C      CHE00900
C      CHE00910
C      CHE00920
C      CHE00930
C      CHE00940
C      CHE00950
C      CHE00960
C      CHE00970
C      CHE00980
C      CHE00990
C      CHE01000
C      CHE01010
C      CHE01020
C      CHE01030
C      CHE01040
C      CHE01050
C      CHE01060
C      CHE01070
C      CHE01080
C      CHE01090
C      CHE01100
C      CHE01110
C      CHE01120
C      CHE01130
C      CHE01140
C      CHE01150
C      CHE01160
C      CHE01170
C      CHE01180
C      CHE01190
C      CHE01200
C      CHE01210
C      CHE01220
C      CHE01230
C      CHE01240
C      CHE01250
C      CHE01260
C      CHE01270
C      CHE01280
C      CHE01290
C      CHE01300
C      CHE01310
C      CHE01320
C      CHE01330
C      CHE01340
C      CHE01350
C      IF(KSTAR.NE.0) GO TO 80
C      KSTAR = IELEM
C*** CALCULATE INCREASE OR DECREASE IN FRONTWIDTH AT EACH ELEMENT STAGE
00      NDFRO(IELEM) = NDFRO(IELEM)+NDOFN
C*** AND CHANGE THE SIGN OF THE LAST APPEARANCE OF EACH NODE
C      KLAST = IELEM
C      NLAST = INODE
C      CHE00660
C      CHE00670
C      CHE00680
C      CHE00690
C      CHE00700
C      CHE00710
C      CHE00720
C      CHE00730
C      CHE00740
C      CHE00750
C      CHE00760
C      CHE00770
C      CHE00780
C      CHE00790
C      CHE00800
C      CHE00810
C      CHE00820
C      CHE00830
C      CHE00840
C      CHE00850
C      CHE00860
C      CHE00870
C      CHE00880
C      CHE00890
C      CHE00900
C      CHE00910
C      CHE00920
C      CHE00930
C      CHE00940
C      CHE00950
C      CHE00960
C      CHE00970
C      CHE00980
C      CHE00990
C      CHE01000
C      CHE01010
C      CHE01020
C      CHE01030
C      CHE01040
C      CHE01050
C      CHE01060
C      CHE01070
C      CHE01080
C      CHE01090
C      CHE01100
C      CHE01110
C      CHE01120
C      CHE01130
C      CHE01140
C      CHE01150
C      CHE01160
C      CHE01170
C      CHE01180
C      CHE01190
C      CHE01200
C      CHE01210
C      CHE01220
C      CHE01230
C      CHE01240
C      CHE01250
C      CHE01260
C      CHE01270
C      CHE01280
C      CHE01290
C      CHE01300
C      CHE01310
C      CHE01320
C      CHE01330
C      CHE01340
C      CHE01350
C*** CHECK THAT COORDINATES FOR AN UNUSED NODE HAVE NOT BEEN SPECIFIED
110 WRITE(6,900) IPDIN
900 FORMAT(//15H CHECK WHY NODE,I4,14H NEVER APPEARS)
NEROR(18) = NEROR(18)+1
SIGMA = 0.0
DO 120 IDIME = 1,2
120 SIGMA = SIGMA+DABS(CDPRO(IPDIN,IDIME))
IF (SIGMA.NE.0.0) NEROR(19) = NEROR(19)+1
C*** CHECK THAT AN UNUSED NODE NUMBER IS NOT A RESTRAINED NODE
DO 130 IVFIX = 1,NVFIX
130 IF(NDFIX(IVFIX).EQ.IPDIN) NEROR(20) = NEROR(20) + 1
140 CONTINUE
C*** CALCULATE THE LARGEST FRONT WIDTH
NFRON = 0
KFRON = 0
DO 150 IELEM = 1,NELEM
150 NFRON = NFRON+NDFRO(IELEM)
IF (NFRON.GT.KFRON) KFRON = NFRON
WRITE(6,905) KFRON
905 FORMAT(//13H MAXIMUM FRONTWIDTH ENCOUNTERED =,I5)
IF (KFRON.GT.MFRON) MFRON(21) = 1
C*** CONTINUE CHECKING THE DATA FOR THE FIXED VALUES
DO 170 IVFIX = 1,NVFIX
IF(NDFIX(IVFIX).LE.0.OR.NDFIX(IVFIX).GT.NPDIN) NEROR(22) =
NEROR(22)+1
KOUNT = 0
NLOCA = (NDFIX(IVFIX)-1)*NDOFN
DO 160 IDOEN = 1,NDOFN
160 NLOCA = NLOCA+1
IF(IVFIX(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.NDFIX(IVFIX).EQ.NDFIX(JVFIX)) NEROR(24)
= NEROR(24)+1
KEROR = 0
DO 180 IERUR = 13,24
IF(NEROR(IERUR).EQ.0) GO TO 180
KERUR = 1
WRITE(6,910) IERUR,NEROR(IERUR)
910 FORMAT(//31H *** DIAGNOSIS BY CHECK2,ERROR,I3,6X,
18H ASSOCIATED NUMBER ,I5)
180 CONTINUE
IF(KEROR.NE.0) GO TO 200
C*** RETURN ALL NUOAL CONNECTION NUMBERS TO POSITIVE VALUES
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** IF DATA ERRORS HAVE BEEN DETECTED BY SUBROUTINES CHECK1 OR
CHECK2, THIS SUBROUTINE READS AND WRITES THE REHAIRING DATA CARDS
C*****
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION NTITL(80)
      WRITE (6,900)
900  FORMAT(//50H NOW FULLOYS A LISTING OF POST-DISASTER OATA CARDS/)
10   READ (5,905)NTITL
905  FORMAT(80A1)
      WRITE (6,910) NTITL
910  FORMAT(20X,80A1)
      GO TO 10
      END
ECHO0010
ECHO0020
ECHO0030
ECHO0040
ECHO0050
ECHO0060
ECHO0070
ECHO0080
ECHO0090
ECHO0100
ECHO0110
ECHO0120
ECHO0130
ECHO0140
ECHO0150
ECHO0160
ECHO0170

SUBROUTINE LOADPS(COORD,LNODS,MATNO,MELEM,MMATS,MPOIN,NELEM,
NEVAB,NGAUS,NNODE,NPQIN,NSTRE,NTYPE,POSGP,
PROPS,RLoad,WEIG,NDOFN)
C*****
C** THIS SUBROUTINE EVALUATES THE CONSISTENT NODAL FORCES FOR EACH
ELEMENT
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),
      NUPRS(4),PGASH(2),POINT(2),PUSGP(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 IFVAB = 1,NEVAB
10   RLOAD(IELEM,IEVAB) = 0.0
      READ(5,901) TITLE
901  FORMAT(18A4)
      WRITE(6,903) TITLE
903  FORMAT(1H0,18A4)
C** READ DATA CONTROLLING LOADING-TYPES TO BE INPUTED
      READ(5,919) IPLDD,IGRAV,IEDGE
      WRITE(6,919) IPLDD,IGRAV,IEDGE
919  FORMAT(3I5)
C** READ NODAL POINT LOADS
      IF (IPLDD.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(15,2F10.3)
C** ASSOCIATE THE NODAL POINT LOADS WITH AN ELEMENT
      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.NPQIN) GO TO 20
500  CONTINUE
      IF (IGRAV.EQ.0) GO TO 600
C** GRAVITY LOADING SECTION
LOAD00010
LOAD00020
LOAD00030
LOAD00040
LOAD00050
LOAD00060
LOAD00070
LOAD00080
LOAD00090
LOAD00100
LOAD00110
LOAD00120
LOAD00130
LOAD00140
LOAD00150
LOAD00160
LOAD00170
LOAD00180
LOAD00190
LOAD00200
LOAD00210
LOAD00220
LOAD00230
LOAD00240
LOAD00250
LOAD00260
LOAD00270
LOAD00280
LOAD00290
LOAD00300
LOAD00310
LOAD00320
LOAD00330
LOAD00340
LOAD00350
LOAD00360
LOAD00370
LOAD00380
LOAD00390
LOAD00400
LOAD00410
LOAD00420
LOAD00430
LOAD00440
LOAD00450
LOAD00460
LOAD00470
LOAD00480
LOAD00490
LOAD00500
LOAD00510
LOAD00520
LOAD00530
LOAD00540
LOAD00550

```



```

90 CONTINUE
LOO CONTINUE
IF (IEDGE.EQ.0) GO TO 700
C*** DISTRIBUTED EDGE LOADS SECTION
C
932 READ (5,932) NEDGE
FORMAT(15)
WRITE (6,912) NEDGE
912 FORMAT (1H0,5X,21HND. OF LOAOEO EDGES =,15)
WRITE (6,915)
915 FORMAT (1H0,5X,38HLIST OF LOAOEO EDGES AND APPLIED LOADS)
NDEEG = 3
NCOOE = NNODE
IF (NNODE.EQ.4) NDEEG = 2
IF (NNODE.EQ.9) NCOOE = 8
C*** LOOP OVER EACH LOAOEO EGE
C
DO 160 IEDEG = 1,NEDGE
C*** READ DATA LOCATING THE LOAOEO EGE AND APPLIED LOAD
C
902 READ (5,902) NEASS,(NOPRS(IDEDEG),IDEDEG=1,NDEEG)
FORMAT(415)
WRITE (6,913) NEASS,(NOPRS(IDEDEG),IDEDEG=1,NDEEG)
913 FORMAT (110,5X,315)
READ (5,914) ((PRESS(IDEDEG,IODFN),IODFN=1,2),IDEDEG=1,NDEEG)
WRITE (6,914) ((PRESS(IDEDEG,IODFN),IODFN=1,2),IDEDEG=1,NDEEG)
914 FORMAT(6F10,3)
ETASP = -1.0
C*** CALCULATE THE COORDINATES OF THE NODES OF THE ELEMENT EGE
C
DO 100 IDEDEG = 1,NDEEG
LNODE = NOPRS(IDEDEG)
DO 100 IDIME = 1,2
100 ELCOD(IDIME,IDEDEG) = COORD(LNODE,IDIME)
C*** ENTER LOOPS FOR LINEAR NUMERICAL INTEGRATION
C
DO 150 IGAUS = 1,NGAUS
EXISP = PLUSP(IGAUS)
C*** EVALUATE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS
C
CALL SF92 (DERIV,ETASP,EXISP,NNODE,SHAPE)
C*** CALCULATE COMPONENTS OF THE EQUIVALENT NODAL LOADS
C
DO 110 IODFN = 1,2
PGASH (IODFN)=0.0
DGASH (IODFN)=0.0
DO 110 IDEDEG = 1,NDEEG
PGASH(IODFN) = PGASH(IODFN)+PRESS(IDEDEG,IODFN)*SHAPE(IDEDEG)
110 OGASH (IODFN) = DGASH(IODFN)+ELCOD(IODFN,IDEDEG)*DERIV(1,IDEDEG)
OVOLU = YEIGP(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
RADIUS = 0.0
DO 125 IDEDEG = 1,NDEEG
125 RAOUS = RADIUS*SHAPE(IDEDEG)+ELCOD(1,IDEDEG)
OVOLU = OVOLU+TWOPI*RADIUS
115 CONTINUE
C*** ASSOCIATE THE EQUIVALENT NODAL EGE LOADS WITH AN ELEMENT
C
DO 120 INODE = 1,NNODE
NLOCA = IABS(LNODES(NEASS,INODE))
IF (NLOCA.EQ.NOPRS(1)) GO TO 130
120 CONTINUE
130 JNODE = INODE+NDEEG-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
140 RLOAD(NEASS,NGASH) = RLOAD(NEASS,NGASH)+SHAPE(KOUNT)*PXCOM+OVOLU
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 (6,908) IELEM, (RLOAD(IELEM,IEVAB),IEVAB=1,NEVAB)
905 FORMAT(1X,14,5X,8E12.4/(10X,8E12.4))
RETURN
END

```

```

LQA01110
LQA01120
LQA01130
LQA01140
LQA01150
LQA01160
LQA01170
LQA01180
LQA01190
LQA01200
LQA01210
LQA01220
LQA01230
LQA01240
LQA01250
LQA01260
LQA01270
LQA01280
LQA01290
LQA01300
LQA01310
LQA01320
LQA01330
LQA01340
LQA01350
LQA01370
LQA01380
LQA01390
LQA01400
LQA01410
LQA01420
LQA01430
LQA01440
LQA01450
LQA01460
LQA01470
LQA01480
LQA01490
LQA01500
LQA01510
LQA01520
LQA01530
LQA01540
LQA01550
LQA01560
LQA01570
LQA01580
LQA01590
LQA01600
LQA01610
LQA01620
LQA01630
LQA01640
LQA01650
LQA01660
LQA01670
LQA01680
LQA01690
LQA01700
LQA01710
LQA01720
LQA01730
LQA01740
LQA01750
LQA01760
LQA01770
LQA01780
LQA01790
LQA01800
LQA01810
LQA01820
LQA01830
LQA01840
LQA01850
LQA01860
LQA01870
LQA01880
LQA01890
LQA01900
LQA01910
LQA01920
LQA01930
LQA01940
LQA01950
LQA01960
LQA01970
LQA01980
LQA01990
LQA02000
LQA02010
LQA02020

```

```

SUBROUTINE ZERO (ELOAD,MELEM,MEVAB,MPOIN,MTOTG,MTOTV,NDOFN,NELEM, ZER00010
* NEVAB,NGAUS,NSTR1,NTOTG,EPSTN,EFFST, ZER00020
* TLOAD,NVFIX,STRSG,TOISP,TFACT, ZER00030
* TLOAD,TREAC,MVFIX) ZER00040
C***** ZER00050
C*** THIS SUBROUTINE INITIALISES VARIOUS ARRAYS TO ZERO ZER00070
C***** ZER00080
C***** ZER00090
IMPLICIT REAL*8(A-H,O-Z) ZER00100
DIMENSION LLOAD (MELEM,MEVAB),STRSG(+,MTOTG),TOISP(MTOTV), ZER00110
* TLOAD(MELEM,MEVAB),TREAC(MVFIX,2),CPSTN(MTOTG), ZER00120
* EFFST(MTOTG) ZER00130
TFACT = 0.0 ZER00140
DO 30 IELEM = 1,NELEM ZER00150
DO 30 IEVAB = 1,NEVAB ZER00160
ELOAD(IELEM,IEVAB) = 0.0 ZER00170
30 TLOAD(IELEM,IEVAB) = 0.0 ZER00180
DO 40 ITOTV = 1,NTOTV ZER00190
40 TOISP(ITOTV) = 0.0 ZER00200
DO 50 IVFIX = 1,NVFIX ZER00210
DO 50 IDOJFN = 1,NDOFN ZER00220
50 TREAC(IVFIX,IDOJFN) = 0.0 ZER00230
DO 60 ITOTG = 1,NTOTG ZER00240
EPSTN(ITOTG) = 0.0 ZER00250
EFFST(ITOTG) = 0.0 ZER00260
DO 60 ISTR1 = 1,NSTR1 ZER00270
60 STRSG(ISTR1,ITOTG) = 0.0 ZER00280
RETURN ZER00290
END ZER00300

```

```

SUBROUTINE INCREM(ELOAD,FIXED,IINCS,MELEM,MEVAB,MITER, INC00010
* MTOTV,MVFIX,NDOFN,NELEM,NEVAB,NOUTP, INC00020
* NOFIX,NTOTV,NVFIX,PRESG,RLOAD,TFACT, INC00030
* TLOAD,TOLER) INC00040
C***** INC00050
C*** THIS SUBROUTINE INCREMENTS THE APPLIED LOADING INC00060
C***** INC00070
C***** INC00080
C***** INC00090
IMPLICIT REAL*8(A-H,O-Z) INC00100
DIMENSION LLOAD(MELEM,MEVAB),FIXED(MTOTV),IEFIX(MTOTV), INC00110
* NOUTP(2),NOFIX(MVFIX),PRESG(MVFIX,NDOFN), INC00120
* RLOAD(MELEM,MEVAB),TLOAD(MELEM,MEVAB) INC00130
WRITE(6,100) IINCS, IINCRMENT NUMBER, I5) INC00140
900 FORMAT(900) 5KACTB INCREMNT WITH NDOJFN(2), NOUTP(2) INC00150
READ(5 (2F10.5,3I5) INC00160
950 FORMAT INC00170
TFACT = TFACT+FACTO INC00180
WRITE(6,960) TFACT, TOLER, MITER, NOUTP(1), NOUTP(2) INC00190
960 FORMAT(10,5X,13HLOAD FACTOR =,F10.5,5X, INC00200
* 24H CONVERGENCE TOLERANCE =,F10.5,5X,24HMAX. NO. OF ITE RATIONS =, INC00210
* I5, //27H INITIAL OUTPUT PARAMETER =,I5,5X, INC00220
* 24H FINAL OUTPUT PARAMETER =,I5) INC00230
DO 80 IELEM = 1,NELEM INC00240
DO 80 IEVAB = 1,NEVAB INC00250
80 ELOAD(IELEM,IEVAB) = ELOAD(IELEM,IEVAB) + RLOAD(IELEM,IEVAB) * FACTO INC00260
TLOAD(IELEM,IEVAB) = TLOAD(IELEM,IEVAB) + RLOAD(IELEM,IEVAB) * FACTO INC00270
C*** INTERPRET FIXITY DATA IN VECTOR FORM INC00290
C***** INC00300
DO 100 ITOTV = 1,NTOTV INC00310
100 FIXED(ITOTV) = 0.0 INC00320
DO 110 IVFIX = 1,NVFIX INC00330
NLOCA = (NOFIX(IVFIX)-1)*NDOFN INC00340
DO 110 IDOJFN = 1,NDOFN INC00350
NGASH = NLOCA+IDOJFN INC00360
FIXED(NGASH) = PRESG(IVFIX,IDOJFN)*FACTO INC00370
110 CONTINUC INC00380
RETURN INC00390
END INC00400

```

```

SUBROUTINE ALGOR(FIXED,IINCS,IITER,KRESL,
                MTOTV,NALGO,NTOTV)
C*****
C*** THIS SUBROUTINE SETS EQUATION RESOLUTION INDEX, KRESL
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 IOTV = 1,NTOTV
FIXED(IOTV) = 0.0
100 CONTINUE
RETURN
END

```

ALG00010
ALG00020
ALG00030
ALG00040
ALG00050
ALG00060
ALG00070
ALG00080
ALG00090
ALG00100
ALG00110
ALG00120
ALG00130
ALG00140
ALG00150
ALG00160
ALG00170
ALG00180
ALG00190
ALG00200
ALG00210

```

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

```

ST100010
ST100020
ST100050
ST100040
ST100050
ST100060
ST100070
ST100080
ST100090
ST100100
ST100110
ST100120
ST100140
ST100150
ST100180
ST100170
ST100180
ST100190
ST100200
ST100210
ST100220
ST100230
ST100240
ST100250
ST100260
ST100270
ST100280
ST100300
ST100310
ST100320
ST100330
ST100340
ST100350
ST100360
ST100370
ST100380
ST100400
ST100410
ST100420
ST100430
ST100440
ST100450
ST100460
ST100470
ST100480
ST100490
ST100500
ST100510
ST100520
ST100540
ST100550

```

C*** EVALUATE THE SHAPE FUNCTIONS, ELEMENTAL VOLUME ETC.
      CALL SFR2(DERIV,ETASP,CXISP,NNODE,SHAPE)
      CALL JACOB2(CARID,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
      NNODE,SHAPE)
      DVOLU = DJACB*WEIGP(I GAUS)*WEIGP(J GAUS)
      IF (NTYPE.CO.3) DVOLU = DVOLU*TWOP1*GPCOD(1,KGASP)
      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
C*** EVALUATE THE B AND DB MATRICES
      CALL DMATPS(DMATX,CARTD,NNODE,SHAPE,GPCOD,NTYPE,KGASP)
      IF (ITNCS.CO.1) GO TO 80
      IF (EPSIN(KGAUS).CO.0.0) GO TO 80
      DO 90 ISTR1 = 1,NSTR1
90    STRES(ISTR1) = STRSOT(ISTR1,KGAUS)
      CALL INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,
      THETA,VARJ2,YIELD)
      CALL YIELD(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
      PROPS,SINT3,STEFF,THETA,VARJ2)
      CALL FLOWPL(AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS)
      DO 100 ISTR2 = 1,NSTR2
      DO 100 JSTR2 = 1,NSTR2
100    DMATX(ISTR2,JSTR2) = DMATX(ISTR2,JSTR2) - ABETA * DVECT(ISTR2) *
      * DVECT(JSTR2)
80    CONTINUE
      CALL DBI(DMATX,DBMAT,DMATX,MEVAB,NEVAB,NSTR2,NSTR1)
C*** CALCULATE THE ELEMENT STIFFNESSES
      DO 30 IEVAB = 1,NEVAB
      DO 30 JEVAB = IEVAB,NEVAB
      DO 10 ISTR1 = 1,NSTR1
30    ESTIF(IEVAB,JEVAB) = ESTIF(IEVAB,JEVAB) + DMATX(ISTR1,IEVAB) *
      * DBMAT(ISTR1,JEVAB) + DVOLU
50    CONTINUE
C*** CONSTRUCT THE LOWER TRIANGLE OF THE STIFFNESS MATRIX
      DO 60 IEVAB = 1,NEVAB
      DO 60 JEVAB = 1,NEVAB
60    ESTIF(JEVAB,IEVAB) = ESTIF(IEVAB,JEVAB)
C*** STORE THE STIFFNESS MATRIX, STRESS MATRIX, AND SAMPLING POINT
      COORDINATES FOR EACH ELEMENT ON OISC FILE
70    WRITE(1) ESTIF
      CONTINUE
      RETURN
      LND

```

```

STI00560
STI00570
STI00580
STI00590
STI00600
STI00610
STI00620
STI00630
STI00640
STI00650
STI00660
STI00670
STI00680
STI00690
STI00700
STI00710
STI00720
STI00730
STI00740
STI00750
STI00760
STI00770
STI00780
STI00790
STI00800
STI00810
STI00820
STI00830
STI00840
STI00850
STI00860
STI00870
STI00880
STI00890
STI00900
STI00910
STI00920
STI00930
STI00940
STI00950
STI00960
STI00970
STI00980
STI00990
STI01000
STI01010
STI01020
STI01030
STI01040
STI01050
STI01060

```

```

SUBROUTINE FRONT(AOIS,ELoad,EORHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS, FR000010
, IITER,GLoad,GSTIF,LOCCEL,LNJDS,KRESL,MBUFA,MELEM, FR000020
, MEVAB,MFRON,MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST, FR000030
, NDOFN,NELEM,NEVAB,NNODE,NDFIX,NPIVO,NPOIN, FR000040
, NTOTV,TDISP,TLOAD,TREAC,VECRV) FR000050
C***** FR000060
C FR000070
C*** THIS SUBROUTINE UNDERTAKES EQUATION SOLUTION BY THE FRONTAL FR000080
METHOD FR000090
C FR000100
C***** FR000110
IMPLICIT REAL*8(A-H,O-Z) FR000120
DIMENSION ASDIS(MTOTV),ELoad(MELEM,MEVAB),EORHS(MBUFA), FR000130
, EQUAT(MFRON,MBUFA),ESTIF(MEVAB,MEVAB),FIXED(MTOTV), FR000140
, IFFIX(MTOTV),NPIVO(MBUFA),VECRV(MFRON),GLoad(MFRON), FR000150
, GSTIF(MSTIF),LNJDS(MELEM,9),LOCCEL(MEVAB),NACVA(MFRON), FR000160
, NAMEV(MBUFA),NDEST(MEVAB),NDFIX(MVFIX),NOUTP(2), FR000170
, TDISP(MTOTV),TLOAD(MELEM,MEVAB),TREAC(MVFIX,NDOFN) FR000180
NFUNC(I,J) = (J-J-1)/2+1 FR000190
C FR000200
C*** CHANGE THE SIGN OF THE LAST APPEARANCE OF EACH NODE FR000210
C FR000220
IF (IINCS.GT.1.OR.IITER.GT.1) GO TO 455 FR000230
DO 140 IPOIN = 1,NPOIN FR000240
KLAST = 0 FR000250
DO 130 IELEM = 1,NELEM FR000260
DO 120 INODE = 1,NNODE FR000270
IF (LNJDS(IELEM,INODE).NE.IPOIN) GO TO 120 FR000280
KLAST = IELEM FR000290
NLAST = INODE FR000300
120 CONTINUE FR000310
130 CONTINUE FR000320
IF (KLAST.NE.0) LNJDS(KLAST,NLAST) = -IPOIN FR000330
140 CONTINUE FR000340
455 CONTINUE FR000350
C FR000360
C*** START BY INITIALIZING EVERYTHING THAT MATTERS TO ZERO FR000370
C FR000380
DO 450 IBUFA = 1,MBUFA FR000390
EORHS(IBUFA) = 0.0 FR000400
DO 150 ISTIF = 1,MSTIF FR000410
GSTIF(ISTIF) = 0.0 FR000420
DO 160 IFRON = 1,MFRON FR000430
GLoad(IFRON) = 0.0 FR000440
VECRV(IFRON) = 0.0 FR000450
NACVA(IFRON) = 0 FR000460
DO 160 IBUFA = 1,MBUFA FR000470
EQUAT(IFRON,IBUFA) = 0.0 FR000480
C FR000490
C*** AND PREPARE FOR DISC READING AND WRITING OPERATIONS FR000500
C FR000510
NBUFA = 0 FR000520
IF (KRESL.GT.1) NBUFA = MBUFA FR000530
REWIND 1 FR000540
REWIND 2 FR000550

```

```

RCUIND 4
REWIND 4
REWIND 2
C
C*** ENTER MAIN ELEMENT ASSEMBLY -REDUCTION LOOP
NFRON = 0
KELVA = 0
DO 320 IELEM = 1,NELEM
IF(KRESL.GT.1) GO TO 400
KEVAR = 0
READ(1) ESTIF
DO 170 INODE = 1,NNODE
DO 170 IODFN = 1,NDOFN
NPOST = (INODE-1)*NDOFN+IODFN
LOCNO = LNDS(IELEM,INODE)
IF (LOCNO.GT.0) LOCEL(NPOST) = (LOCNO-1)*NDOFN+IODFN
IF (LOCNO.LT.0) LOCEL(NPOST) = (LOCNO+1)*NDOFN-IODFN
170 CONTINUE
C
C*** START BY LOOKING FOR EXISTING DESTINATIONS
DO 210 IEVAR = 1,NEVAR
NIKNO = IABS(LOCEL(IEVAR))
KEXIS = 0
DO 180 IFRON = 1,NFRON
IF (NIKNO.LE.NACVA(IFRON)) GO TO 180
KEVAR = KEVAR+1
KEXIS = 1
NDEST(KEVAR) = IFRON
180 CONTINUE
IF (KEXIS.NE.0) GO TO 210
C
C*** WE NOW SEEK NEW EMPTY PLACES FOR DESTINATION VECTOR
DO 170 IFRON = 1,NFRON
IF (NACVA(IFRON).NE.0) GO TO 190
NACVA(IFRON) = NIKNO
KEVAR = KEVAR+1
NDEST(KEVAR) = IFRON
GO TO 200
190 CONTINUE
C
C*** THE NEW PLACES MAY DEMAND AN INCREASE IN CURRENT FRONTWIDTH
200 IF (NDEST(KEVAR).GT.NFRON) NFRON = NDEST(KEVAR)
210 CONTINUE
WRITE(8) LOCEL,NDEST,NACVA,NFRON
400 IF(KRESL.GT.1) READ(8) LOCEL,NDEST,NACVA,NFRON
C
C*** ASSEMBLE ELEMENT LUAS
DO 220 IEVAR = 1,NEVAR
IDEST = NDEST(IEVAR)
GLDAD(IDEST) = GLDAD(IDEST)+ELDAD(IELEM,IEVAR)

```

```

FR000560
FR000570
FR000580
FR000590
FR000600
FR000610
FR000620
FR000630
FR000640
FR000650
FR000660
FR000670
FR000680
FR000690
FR000700
FR000710
FR000720
FR000730
FR000740
FR000750
FR000760
FR000770
FR000780
FR000790
FR000800
FR000810
FR000820
FR000830
FR000840
FR000850
FR000860
FR000870
FR000880
FR000890
FR000900
FR000910
FR000920
FR000930
FR000940
FR000950
FR000960
FR000970
FR000980
FR000990
FR001000
FR001010
FR001020
FR001030
FR001040
FR001050
FR001060
FR001070
FR001080
FR001090
FR001100

```



```

C
C*** ASSEMBLE THE ELEMENT STIFFNESSES-BUT NOT IN RESOLUTION
C
  IF(KRESL.GT.1) GO TO 402
  DO 222 JEVAB = 1,IEVAB
  JDEST = NDEFCT(JEVAB)
  NGASH = NFUNC(JDEST,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
  ON 310 JEVAB = 1,NEVAB
  NIKND=-LUCCL(IEVAB)
  IF(NIKND.LE.0) GO TO 310
C
C*** FIND POSITIONS OF VARIABLES READY FOR ELIMINATION
C
  DO 300 IFRON = 1,MFRON
  IF(MACVA(IFRON).NE.NIKND) 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(3) EQUAT,EORHS,NPIVD,NAMEV
  GO TO 406
408 WRITE(4) EORHS
  READ(2) EQUAT,EORHS,NPIVD,NAMEV
40h CONTINUE
C
C*** EXTRACT THE COEFFICIENTS OF THE KEY 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
  EORHS(NBUFA) = GLOAD(IFRON)
  GLOAD(IFRON) = 0.0
  KELVA = K/LVA+1
  NAMEV(NBUFA) = NIKND
  NPIVD(NBUFA) = IFRON
C
--
FR001110
FR001120
FR001130
FR001140
FR001150
FR001160
FR001170
FR001180
FR001190
FR001200
FR001210
FR001230
FR001240
FR001250
FR001260
FR001270
FR001280
FR001290
FR001300
FR001310
FR001320
FR001330
FR001340
FR001350
FR001360
FR001370
FR001380
FR001390
FR001400
FR001410
FR001420
FR001430
FR001440
FR001450
FR001460
FR001470
FR001480
FR001490
FR001500
FR001510
FR001520
FR001530
FR001540
FR001550
FR001560
FR001570
FR001580
FR001590
FR001600
FR001610
FR001620
FR001630
FR001640
FR001650

```

```

C*** DEAL WITH PIVOT
C
    PIVOT = EQUAT(IFRON,NBUFA)
    IF(PIVOT.GT.0.0) GO TO 235
    WRITE(4,900) NIKND,PIVOT
900  FORMAT(IH0,3X,52HNEGATIVE OR ZERO PIVOT ENCOUNTERED FOR VARIABLE N
    *D,14,10H IF VALUE ,E17.6)
    STOP
235  CONTINUE
    EQUAT(IFRON,NBUFA) = 0.0
C
C*** ENQUIRE WHETHER PRESENT VARIABLE IS FREE OR PRESCRIBED
C
    IF (IFFIX(NIKND).EQ.0) GO TO 250
C
C*** DEAL WITH A PRESCRIBED DEFLECTION
C
    DO 240 JFRON = 1,NFRON
240  GLOAD(JFRON) = GLOAD(JFRON)-FIXED(NIKND)*EQUAT(JFRON,NBUFA)
    GO TO 280
C
C*** ELIMINATE A FREE VARIABLE-DEAL WITH THE RIGHT HAND SIDE FIRST
C
250  ON 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
260  NGASH = LFRON+NLOCA
    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.(0.1) WRITE(2) EQUAT,EQRHS,NPIV),NAMEV
    BACKSPACE 2

```

```

FR001660
FR001670
FR001680
FR001690
FR001700
FR001710
FR001720
FR001730
FR001740
FR001750
FR001760
FR001770
FR001780
FR001790
FR001800
FR001810
FR001820
FR001830
FR001840
FR001850
FR001860
FR001870
FR001880
FR001890
FR001900
FR001910
FR001920
FR001930
FR001940
FR001950
FR001960
FR001970
FR001980
FR001990
FR002000
FR002010
FR002020
FR002030
FR002040
FR002050
FR002060
FR002080
FR002090
FR002100
FR002110
FR002120
FR002130
FR002140
FR002150
FR002160
FR002170
FR002180
FR002190
FR002200

```

```

C** ENTER BACK-SUBSTITUTION PHASE. LOOP BACKWARDS THROUGH VARIABLES
C
C
C** READ A NEW BLOCK OF EQUATIONS - IF NEEDED
C
IF (NBUFA.NE.0) GO TO 412
BACKSPACE 2
READ(2) EQUAT,EQRHS,NPIVD,NAMEV
BACKSPACE 3
NBUFA = NBUFA
IF (KRESL.(0.1) GO TO 412
BACKSPACE 4
READ(4) EQRHS
BACKSPACE 4
412 CONTINUE
C** PREPARE TO BACK-SUBSTITUTE FROM THE CURRENT EQUATION
C
IFRIN = NPIVD(NBUFA)
NIKND = NAMEV(NBUFA)
PIVOT = EQUAT(IFRIN,NBUFA)
IF (IFFIX(NIKND).NE.0) VECRV(IFRIN) = FIXED(NIKND)
IF (IFFIX(NIKND).EQ.0) EQUAT(IFRIN,NBUFA) = 0.0
C** BACK-SUBSTITUTE IN THE CURRENT EQUATION
C
DO 330 JFRON = 1,IFRIN
330 EQRHS(NBUFA) = EQRHS(NBUFA)-VECRV(JFRON)*EQUAT(JFRON,NBUFA)
C** PUT THE FINAL VALUES UNDER THEY BELONG
C
IF (IFFIX(NIKND).EQ.0) VECRV(IFRIN) = EQRHS(NBUFA)/PIVOT
IF (IFFIX(NIKND).NE.0) FIXED(NIKND) = -EQRHS(NBUFA)
NBUFA = NBUFA-1
ASDIS(NIKND) = VECRV(IFRIN)
340 CONTINUE
C** ADD DISPLACEMENTS TO PREVIOUS TOTAL VALUES
C
DO 345 ITOTV = 1,NTUTV
345 TDISP(ITOTV) = TDISP(ITOTV)+ASDIS(ITOTV)
C** STORE REACTIONS FOR PRINTING LATER
C
KBOUN = 1
DO 370 IPUIN = 1,NPOIN
NLUCA = (IPUIN-1)*NDOFN
DO 350 IDOFN = 1,NDOFN
NGASH = NLUCA+IDOFN
IF (IFFIX(NGASH).GT.0) GO TO 360
350 CONTINUE
GO TO 370
360 DO 510 IDOFN = 1,NDOFN
510 NGASH = NLUCA+IDOFN
TREAC(KBOUN,IDOFN) = TREAC(KBOUN,IDOFN)+FIXED(NGASH)
KBOUN = KBOUN+1
370 CONTINUE
C** ADD REACTIONS INTO THE TOTAL LOAD ARRAY
C
DO 700 IPUIN = 1,NPOIN
DO 710 IELEM = 1,NELEM
DO 710 INODE = 1,NNODE
NLUCA = IABS(LNODES(IELEM,INODE))
IF (IPUIN.EQ.NLUCA) GO TO 720
710 CONTINUE
DO 730 IDOFN = 1,NDOFN
NGASH = (INODE-1)*NDOFN+IDOFN
MGASH = (IPUIN-1)*NDOFN+IDOFN
730 TLOAD(IELEM,NGASH) = TLOAD(IELEM,NGASH)+FIXED(MGASH)
700 CONTINUE
RETURN
END
FR002210
FR002220
FR002230
FR002240
FR002250
FR002260
FR002270
FR002280
FR002290
FR002300
FR002310
FR002320
FR002330
FR002340
FR002350
FR002360
FR002370
FR002380
FR002390
FR002400
FR002410
FR002420
FR002430
FR002440
FR002450
FR002460
FR002470
FR002480
FR002490
FR002500
FR002510
FR002520
FR002530
FR002540
FR002550
FR002560
FR002570
FR002580
FR002590
FR002600
FR002610
FR002620
FR002630
FR002640
FR002650
FR002660
FR002670
FR002680
FR002690
FR002700
FR002710
FR002720
FR002730
FR002740
FR002750
FR002760
FR002770
FR002780
FR002790
FR002800
FR002810
FR002820
FR002830
FR002840
FR002850
FR002860
FR002870
FR002880
FR002890
FR002900
FR002910
FR002920
FR002930
FR002940
FR002950

```

```

SUBROUTINE RESIDU(ASDIS,COORD,EFFST,ELoad,FACT,IITER,LNODES,
* LPROP,MATNO,MELEM,MMATS,MPOIN,MTOTG,MTOTV,NDOFN,
* NELEM,NEVAB,NGAUS,NNODE,NSTR1,NTYPE,POSGP,PROPS,
* NSTRE,NCRIT,STRSG,WEIGP,TDISP,EPSTN)
C*****
C**** THIS SUBROUTINE REDUCES THE STRESSES TO THE YIELD SURFACE AND
C***** EVALUATES THE EQUIVALENT NODAL FORCES
C*****
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION ASDIS(MTOTV),AVECT(4),CARTD(2,9),COORD(MPOIN,2),
* DVIA(4),DVECT(4),EFFST(MTOTG),ELCDD(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),DLCDD(2,9),DESIG(4),SIGMA(4),SGTOT(4),
* DMATX(4,4),DERIV(2,9),SHAPE(9),GPCDD(2,9),
* EPSTN(MTOTG),TDISP(MTOTV),MATNO(MELEM),BMATX(4,18)
      ROOT3 = 1.73205080757
      TWUPI = 6.283185308
      DO 10 IELEM = 1,NELEM
      DO 10 I'VAB = 1,NEVAB
10  ELoad(IELEM,I'VAB) = 0.0
      KGAUS = 0
      DO 20 I'LEM = 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**** COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE
C***** ELEMENT NODAL POINTS
      DO 30 INODE =1,NNODE
      LNODE = IABS(LNODES(IELEM,INODE))
      NPOSN = (LNODE-1)*NDOFN
      DO 30 IDOFN = 1,NDOFN
      NPOSN = NPIJSN+1
      ELCDD(IDOFN,INODE)=COORD(LNODE,IDOFN)
30  ELDIS(IDOFN,INODE)=ASDIS(NPOSN)
      CALL MDOPS(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 = KGAUS*1
      CALL SFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
      CALL JACOB2 (CARTD,DERIV,DJACB,ELCDD,GPCDD,IELEM,KGASP,
RES00010
RES00020
RES00030
RES00040
RES00050
RES00060
RES00070
RES00080
RES00090
RES00100
RES00110
RES00120
RES00130
RES00140
RES00150
RES00160
RES00170
RES00180
RES00190
RES00200
RES00210
RES00220
RES00230
RES00240
RES00250
RES00260
RES00270
RES00280
RES00290
RES00300
RES00310
RES00320
RES00330
RES00340
RES00350
RES00360
RES00370
RES00380
RES00390
RES00400
RES00410
RES00420
RES00430
RES00440
RES00450
RES00460
RES00470
RES00480
RES00490
RES00500
RES00510
RES00520
RES00530
RES00540
RES00550

```

```

      NNODE,SHAPE)
      DVOLU = DJACB*WEIGP(KGAUS)*WEIGP(JGAUS)
      IF (NTYPE.EQ.3) DVOLU = DVOLU*(WOP1-GPCDD(1,KGASP))
      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
      CALL DMATPS(DMATX,CARTD,NNODE,SHAPE,GPCDD,NTYPE,KGASP)
      CALL LINEAR(CARTD,DMATX,ELDIS,LPROP,MMATS,NDOFN,NNODE,NSTRE,
      NTYPE,PROPS,STRAN,STRES,KGASP,GPCDD,SHAPE)
      PREYS = UNIAX+EPSTN(KGAUS)*HARDS
      DO 150 ISTR1 = 1,NSTR1
      DESIG(ISTR1) = STRES(ISTR1)
      SIGMA(ISTR1) = STRSG(ISTR1,KGAUS)*STRES(ISTR1)
150  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 YIELD(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
      PROPS,SINT3,STEFF,THETA,VARJ2)
      CALL FLIWPL(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
110  BGASH = BGASH+AVECT(ISTR1)+SGTOT(ISTR1)
      SGTOT(ISTR1) = SGTOT(ISTR1)+STRES(ISTR1)-DLAMD*DVECT(ISTR1)
      FPSTN(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
RES00560
RES00570
RES00580
RES00590
RES00600
RES00610
RES00620
RES00630
RES00640
RES00650
RES00660
RES00670
RES00680
RES00690
RES00700
RES00710
RES00720
RES00730
RES00740
RES00750
RES00760
RES00770
RES00780
RES00790
RES00800
RES00810
RES00820
RES00830
RES00840
RES00850
RES00860
RES00870
RES00880
RES00890
RES00900
RES00910
RES00920
RES00930
RES00940
RES00950
RES00960
RES00970
RES00980
RES00990
RES01000
RES01010
RES01020
RES01030
RES01040
RES01050
RES01060
RES01070
RES01080
RES01090
RES01100

C 90 CONTINUE
C***
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
      ELEMENT NODES
290 MGASH = 0
      DO 140 INODE = 1,NNODE
      DO 140 IDOFN = 1,NDOFN
      MGASH = MGASH+1
      DO 140 ISTR1 = 1,NSTRE
140* ELOAD(I,ELEM,MGASH) = ELGAD(I,ELEM,MGASH)+BMATX(ISTR1,MGASH)
      * STRSG(ISTR1,KGAUS)*DVOLU
40 CONTINUE
20 CONTINUE
      RETURN
      END
RES01110
RES01120
RES01130
RES01140
RES01150
RES01160
RES01170
RES01180
RES01190
RES01200
RES01210
RES01220
RES01230
RES01240
RES01250
RES01260
RES01270
RES01280
RES01290
RES01300

```

```

SUBROUTINE CONVER(ELoad,ITER,LNOODS,MELEM,MFVAB,MTOTV,NCHEK,
                 NDOFN,NELEN,NEVAB,NNODE,NTOTV,PVALU,STFOR,
                 TLOAD,TOFOR,TOLFR)
C*****
C** THIS SUBROUTINE CHECKS FOR CONVERGENCE OF THE ITERATION PROCESS
C*****
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION ELoad(MELEM,MFVAB),LNOODS(MELEM,NNODE),STFOR(MTOTV),
           TLOAD(MELEM,MFVAB)
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(LNOODS(IELEM,INODE))
DO 40 IDOFN = 1,NDOFN
KCVAB = KEVAB+1
NPOSI = (LOCNO-1)*NDOFN+IDOFN
40 STFOR(NPOSI) = STFOR(NPOSI)+ELoad(IELEM,KEVAB)
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 IFVAB = 1,NEVAB
10 ELoad(IELEM,IFVAB) = TLOAD(IELEM,IFVAB)-ELoad(IELEM,IFVAB)
RESID = DSORT(RESID)
RETOT = DSORT(RETOT)
RATIO = 100.0*RESID/RETOT
IF (RATIO.GT.TOLER) NCHEK = 1
IF (ITER.EQ.1) GO TO 20
IF (RATIO.GT.PVALU) NCHEK = 999
20 PVALU = RATIO
WRITE(0,30) ITER,NCHEK,RATIO,REMAX
30 FORMAT(1H0,3X,14HITERATION NO =,I3,
          3X,18HCONVERGENCE CODE =,I4,3X,28HNRDM OF RESIDUAL SUM
          *RATIO =,E14.6,3X,18HMAXIMUM RESIDUAL =,E14.6)
RETURN
END

```

```

CON00010
CON00020
CON00030
CON00040
CON00050
CON00060
CON00070
CON00080
CON00090
CON00100
CON00110
CON00120
CON00130
CON00140
CON00150
CON00160
CON00170
CON00180
CON00190
CON00200
CON00210
CON00220
CON00230
CON00240
CON00250
CON00260
CON00270
CON00280
CON00290
CON00300
CON00310
CON00320
CON00330
CON00340
CON00350
CON00360
CON00370
CON00380
CON00390
CON00400
CON00410
CON00420
CON00430
CON00440
CON00450
CON00460
CON00470
CON00480
CON00490
CON00500

```

```

SUBROUTINE OUTPUT(IITER,MTDTG,MTDTV,MVFIX,NELEM,NGAUS,NDFIX,
NOUTP,NPOIN,NVFIX,STRSG,TDISP,TREAC,EPSTN,
NTYPE,NCHK,FFST)
C*****
C*** THIS SUBROUTINE OUTPUTS DISPLACEMENTS, REACTIONS AND STRESSES
C*****
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION NDFIX(MVFIX),NOUTP(2),STRSG(4,MTDTG),STRSP(3),
          TDISP(MDTV),TREAC(MVFIX,2),EPSTN(MDTG),EFFST(MDTG)
KOUTP = NOUTP(1)
IF (IITER.GT.1) KOUTP = NOUTP(2)
IF (IITER.LO.1.AND.NCHK.EQ.0) KOUTP = NOUTP(2)
WRITE (6,15) IITER
15 FORMAT (1H0,3X,14HITERATION NO =,I3)
C*** OUTPUT DISPLACEMENTS
C
      F (KOUTP,LT.1) GO TO 10
      WRITE (6,900)
900  FORMAT(1H0,5X,13HDISPLACEMENTS)
      IF (NTYPE.NF.3) WRITE (6,950)
950  FORMAT (1H0,6X,4HNODE,6X,7HX-DISP.,7X,7HY-DISP.)
      IF (NTYPE.FO.3) WRITE (6,955)
955  FORMAT (1H0,6X,4HNODE,6X,7HX-DISP.,7X,7HZ-DISP.)
      ON 20 IPDIN = 1,NPOIN
      NGASH = IPDIN*2
      NGISH = NGASH-2+1
20  WRITE (6,910) IPDIN, (TDISP(IGASH),IGASH=NGISH,NGASH)
910  FORMAT(110,3E14.6)
10  CONTINUE
C*** OUTPUT REACTIONS
C
      IF (KOUTP.LT.2) GO TO 30
      WRITE (6,920)
920  FORMAT(1H0,5X,9HREACTIONS)
      IF (NTYPE.NF.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,7HX-REAC., 7X,7HZ-REAC.)
      DO 40 IVFIX = 1,NVFIX
40  WRITE (6,910) NDFIX(IVFIX),(TREAC(IVFIX,IDOIN),IDOIN=1,2)
30  CONTINUE
C*** OUTPUT STRESSES
C
      IF (KOUTP.LT.3) GO TO 50
      IF (NTYPE.NF.3) WRITE (6,970)
970  FORMAT(1H0,1X,4HG.P.,6X,9HXX-STRESS,5X,3HYY-STRESS,5X,
      * 3HXY-STRESS,5X,9HZZ-STRESS,6X,8HMAX P.S.,6X,8HMIN P.S.,3X,
      * 5HANGLE,4X,10HEFF.STRESS,3X,6HE.P.S.)
      IF (NTYPE.FO.3) WRITE (6,975)
975  FORMAT(1H0,1X,4HG.P.,6X,9HXX-STRESS,5X,3HZZ-STRESS,5X,
      * 9HRY-STRESS,5X,9HRT-STRESS,6X,8HMAX P.S.,6X,8HMIN P.S.,3X,
      * 5HANGLE,4X,10HEFF.STRESS,3X,6HE.P.S.)
      KGAUS = 0
      DO 60 ILEM = 1,NELEM
      KELGS = 0
      WRITE (6,930) ILEM
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 = DSORT(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(ISTR1),ISTR1=1,3),EFFST(KGAUS),EPSTN(KGAUS)
940  FORMAT(15,2X,6E14.6,F8.3,2E14.6)
50  CONTINUE
      RETURN
      END

```

```

OUT00010
OUT00020
OUT00030
OUT00040
OUT00050
OUT00060
OUT00070
OUT00080
OUT00090
OUT00100
OUT00110
OUT00120
OUT00130
OUT00140
OUT00150
OUT00160
OUT00170
OUT00180
OUT00190
OUT00200
OUT00210
OUT00220
OUT00230
OUT00240
OUT00250
OUT00260
OUT00270
OUT00280
OUT00290
OUT00300
OUT00310
OUT00320
OUT00330
OUT00340
OUT00350
OUT00360
OUT00370
OUT00380
OUT00390
OUT00400
OUT00410
OUT00420
OUT00430
OUT00440
OUT00450
OUT00460
OUT00470
OUT00480
OUT00490
OUT00500
OUT00510
OUT00520
OUT00530
OUT00540
OUT00550
OUT00560
OUT00570
OUT00580
OUT00590
OUT00600
OUT00610
OUT00620
OUT00630
OUT00640
OUT00650
OUT00660
OUT00670
OUT00680
OUT00690
OUT00700
OUT00710
OUT00720
OUT00730
OUT00740
OUT00750
OUT00760
OUT00770
OUT00780
OUT00790
OUT00800

```

```

SUBROUTINE CFR2(DERIV,ETASP,EXISP,NNODE,SHAPE)
C*****
C*** THIS SUBROUTINE EVALUATES THE SHAPE FUNCTIONS AND THEIR
C DERIVATIVES FOR LINEAR, QUADRATIC LAGRANGIAN AND SERENDIPITY
C ISOPARAMETRIC 2-D ELEMENTS
C*****
      IMPLICIT REAL*4(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*** SHAPE FUNCTIONS FOR 4 NODDED ELEMENT
      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*** SHAPE FUNCTION DERIVATIVES
      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
      SS = S*S
      TT = T*T
      STT = S*T
      SST = S*S*T
      STT = S*T*T
      STT = S*T*T
      STT = S*T*T
      STT = S*T*T
      STT = S*T*T
C*** SHAPE FUNCTIONS FOR 8 NODDED ELEMENT
      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*** SHAPE FUNCTION DERIVATIVES

```

SFR00010
SFR00020
SFR00030
SFR00040
SFR00050
SFR00060
SFR00070
SFR00080
SFR00090
SFR00100
SFR00110
SFR00120
SFR00130
SFR00140
SFR00150
SFR00160
SFR00170
SFR00180
SFR00190
SFR00200
SFR00210
SFR00220
SFR00230
SFR00240
SFR00250
SFR00260
SFR00270
SFR00280
SFR00290
SFR00300
SFR00310
SFR00320
SFR00330
SFR00340
SFR00350
SFR00360
SFR00370
SFR00380
SFR00390
SFR00400
SFR00410
SFR00420
SFR00430
SFR00440
SFR00450
SFR00460
SFR00470
SFR00480
SFR00490
SFR00500
SFR00510
SFR00520
SFR00530
SFR00540
SFR00550


```

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*.5
ST = S*T
TT = T*.1
S1 = S+1.0
T1 = T+1.0
S2 = S*.5
T2 = T*.3
SS = 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
SFR00580
SFR00581
SFR00580
SFR00590
SFR00600
SFR00610
SFR00620
SFR00630
SFR00640
SFR00650
SFR00660
SFR00680
SFR00690
SFR00700
SFR00710
SFR00720
SFR00730
SFR00740
SFR00750
SFR00760
SFR00770
SFR00780
SFR00790
EFROOAOO
SFR00810
SFR00820
SFR00830
SFR00840
SFR00850
SFR00860
SFR00870
SFR00880
SFR00890
SFR00900
SFR00910
SFR00920
SFR00930
SFR00940
SFR00950
SFR00960
SFR00970
SFR00980
SFR00990
SFR01000
SFR01020
SFR01030
SFR01040
SFR01050
SFR01060
SFR01070
SFR01080
SFR01090
SFR01100
SFR01110
SFR01120
SFR01130
SFR01140
SFR01150
SFR01160
SFR01170
SFR01180

```

```

      SURROUTINE JACOB2(CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
      NNODE,SHAPE)
C****
C*** THIS SUBROUTINE EVALUATES THE JACOBIAN MATRIX AND THE CARTESIAN
      SHAPE FUNCTION DERIVATIVES
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),XJACB(2,2)
C*** CALCULATE THE COORDINATES OF THE SAMPLING POINT
      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*** CREATE JACOBIAN MATRIX XJACB
      DO 4 IDIME = 1,2
      DO 4 JDIME = 1,2
      XJACB(IDIME,JDIME) = 0.0
      DO 4 INODE = 1,NNODE
      XJACB(IDIME,JDIME) = XJACB(IDIME,JDIME) + DERIV(IDIME,INODE)
      * ELCOD(JDIME,INODE)
      4 CONTINUE
C*** CALCULATE DETERMINANT AND INVERSE OF JACOBIAN MATRIX
      DJACB = XJACB(1,1)*XJACB(2,2) - XJACB(1,2)*XJACB(2,1)
      IF(DJACB) 6,6,8
      6 WRITE(6,600) IELEM
      STOP
      8 CONTINUE
      XJACI(1,1) = XJACB(2,2)/DJACB
      XJACI(2,2) = XJACB(1,1)/DJACB
      XJACI(1,2) = -XJACB(1,2)/DJACB
      XJACI(2,1) = -XJACB(2,1)/DJACB
C*** CALCULATE CARTESIAN DERIVATIVES
      DO 10 IDIME = 1,2
      DO 10 INODE = 1,NNODE
      CARTD(IDIME,INODE) = 0.0
      DO 10 JOIME = 1,2
      CARTD(IDIME,INODE) = CARTD(IDIME,INODE) + XJACI(IDIME,JOIME)
      * DERIV(JOIME,INODE)
      10 CONTINUE
      600 FORMAT(//,36H PROGRAM HALTED IN SUBROUTINE JACOB2,/,11X,
      *22H ZERO OR NEGATIVE AREA,/,10X,16H ELEMENT NUMBER,15)
      RETURN
      END

```

JAC00560


```

SUBROUTINE INVAR(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STEMP, INV00010
                    THETA,VARJ2,YIELD) INV00020
C..... INV00030
C..... INV00040
C*** THIS SUBROUTINE EVALUATES THE STRESS INVARIANTS AND THE CURRENT INV00050
C..... VALUE OF THE YIELD FUNCTION INV00060
C..... INV00070
C..... INV00080
IMPLICIT REAL*8(A-H,O-Z) INV00090
DIMENSION DEVIA(4),PROPS(MMATS,7),STEMP(4) INV00100
ROOT3 = 1.73205080757 INV00110
SMEAN = (STEMP(1)+STEMP(2)+STEMP(4))/3.0 INV00120
DEVIA(1) = STEMP(1)-SMEAN INV00130
DEVIA(2) = STEMP(2)-SMEAN INV00140
DEVIA(3) = STEMP(3) INV00150
DEVIA(4) = STEMP(4)-SMEAN INV00160
VARJ2 = DEVIA(3)*DEVIA(3)+0.5*(DEVIA(1)*DEVIA(1)+DEVIA(2)* INV00170
        DEVIA(2)+DEVIA(4)+DEVIA(4)) INV00180
VARJ3 = DEVIA(4)*DEVIA(4)+DEVIA(4)-VARJ2 INV00190
STEFF = DSQRT(VARJ2) INV00200
IF (STEFF.EQ.0.0) GO TO 10 INV00210
SINT3 = -3.0+ROOT3*VARJ3/(2.0+VARJ2*STEFF) INV00220
IF (SINT3.GT.1.0) SINT3 = 1.0 INV00230
GO TO 20 INV00240
10 SINT3 = 0.0 INV00250
20 CONTINUE INV00260
IF (SINT3.LT.-1.0) SINT3 = -1.0 INV00270
IF (SINT3.GT.1.0) SINT3 = 1.0 INV00280
THETA = DARSIN(SINT3)/3.0 INV00290
GO TO (1,2,3,4),NCRIT INV00300
C*** TRESCA INV00310
1 YIELD = 2.0*DCOS(THETA)*STEFF INV00320
RETURN INV00330
C*** VON MISES INV00340
2 YIELD = ROOT3*STEFF INV00350
RETURN INV00360
C*** MOHR-COULOMB INV00370
3 PHIRA = PROPS(LPROP,7)*0.017453292 INV00380
SNPHI = DSIN(PHIRA) INV00390
YIELD = SMEAN+SNPHI+STEFF*(DCOS(THETA)-DSIN(THETA)+SNPHI/ROOT3) INV00400
RETURN INV00410
C*** DRUCKER-PRAGER INV00420
4 PHIRA = PROPS(LPROP,7)*0.017453292 INV00430
SNPHI = DSIN(PHIRA) INV00440
YIELD = 6.0*SMEAN+SNPHI/(ROOT3*(3.0-SNPHI))+STEFF INV00450
RETURN INV00460
END INV00470

```

```

      SUBROUTINE YIELD (AVECT,DEVIA,LEPROP,MMATS,NCRIT,NSTR1,
      PROPS,SINT3,STEFF,THETA,VARJ2)
C*****
C*** THIS SUBROUTINE EVALUATES THE FLOY VECTOR
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
      FRIC1 = PROPS(LEPROP,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*** CALCULATE VECTOR A1
      VECA1(1) = 1.0
      VECA1(2) = 1.0
      VECA1(3) = 0.0
      VECA1(4) = 1.0
C*** CALCULATE VECTOR A2
10  DO 10 ISTR1 = 1,NSTR1
      VECA2(ISTR1) = DEVIA(ISTR1)/(2.0*STEFF)
      VECA2(3) = DEVIA(3)/STEFF
C*** CALCULATE VECTOR A3
      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*** TRFSCA
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
2  CONS1 = 0.0
      CONS2 = ROOT3
      CONS3 = 0.0
      GO TO 40
C*** MOHR-COULUMB
3  CONS1 = DABS(FRICT+0.017453292)/3.0
      ABTHE = DABS(THETA*57.29577951308)
      IF (ABTHE.LT.29.0) GO TO 30
      CONS3 = 0.0
      PLNTH = 1.0
      IF (THETA.GT.0.0) PLUMI = -1.0
      CONS2 = 0.5*(ROOT3+PLUMI*CONS1+ROOT3)
      GO TO 41
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*** DRUCKE R-PRAGER
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

```

```

YIE00010
YIE00020
YIE00030
YIE00040
YIE00050
YIE00060
YIE00070
YIE00080
YIE00090
YIE00100
YIE00110
YIE00120
YIE00130
YIE00140
YIE00150
YIE00160
YIE00170
YIE00180
YIE00190
YIE00200
YIE00210
YIE00220
YIE00230
YIE00240
YIE00250
YIE00260
YIE00270
YIE00280
YIE00290
YIE00300
YIE00310
YIE00320
YIE00330
YIE00340
YIE00350
YIE00360
YIE00370
YIE00380
YIE00390
YIE00400
YIE00410
YIE00420
YIE00430
YIE00440
YIE00450
YIE00460
YIE00470
YIE00480
YIE00490
YIE00500
YIE00510
YIE00520
YIE00530
YIE00540
YIE00550
YIE00560
YIE00570
YIE00580
YIE00590
YIE00600
YIE00610
YIE00620
YIE00630
YIE00640
YIE00650
YIE00660
YIE00670
YIE00680
YIE00690
YIE00700
YIE00710
YIE00720
YIE00730
YIE00740
YIE00750
YIE00760
YIE00770
YIE00780
YIE00790
YIE00800
YIE00810
YIE00820
YIE00830
YIE00840
YIE00850
YIE00860
YIE00870
YIE00880

```

```

      SUBROUTINE FLOWPL(AVECT,ABETA,DVECT,NTYPE,PROPS,
                     LPROP,NSTR1,MMATS)
      C*****
      C*** THIS SUBROUTINE EVALUATES THE PLASTIC D VECTOR
      C*****
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AVECT(4),DVECT(4),PROPS(MMATS,7)
      YOUNG = PROPS(LPROP,1)
      POISS = PROPS(LPROP,2)
      HARUS = 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 = HARUS
      DO 80 ISTR1 = 1,NSTR1
80 DENOM = DENOM+AVECT(ISTR1)*DVECT(ISTR1)
      ABETA = 1.0/DENOM
      RETURN
      END

```

```

      SUBROUTINE DHE(BMATX,DBMAT,DMATX,MEVAB,NEVAB,NSTRE,NSTR1)
      C*****
      C*** THIS SUBROUTINE MULTIPLIES THE D-MATRIX BY THE B-MATRIX
      C*****
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION BMATX(NSTR1,MEVAB),DBMAT(NSTR1,NEVAB),
      DMATX(NSTR1,NSTR1)
      DO 2 ISTR1 = 1,NSTR1
      DO 2 IEVAB = 1,NEVAB
      DBMAT(ISTR1,IEVAB) = 0.0
      DO 2 JSTRE = 1,NSTRE
      DMATX(ISTR1,JSTRE) = DBMAT(ISTR1,IEVAB)+
      *DMATX(ISTR1,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*** THIS SUBROUTINE EVALUATES STRESSES AND STRAINS ASSUMING LINEAR
      ELASTIC BEHAVIOR
      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 IODFN = 1,NDOFN
      DO 20 JDOFN = 1,NDOFN
      BGASH = 0.0
      DO 10 INODE = 1,NNODE
10 BGASH = BGASH+CARTD(JDOFN,INODE)*ELDIS(IODFN,INODE)
20 AGASH(IODFN,JDOFN) = BGASH
      C*** CALCULATE THE STRAINS
      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) = 2*STRAN(4)+ELDIS(1,INODE)*SHAPE(INODE)/GPCOD(1,KGASP)
      C*** AND THE CORRESPONDING STRESSES
      DO 40 ISTR1 = 1,NSTRE
      STRES(ISTR1) = 0.0
      DO 40 JSTRE = 1,NSTRE
40 STRES(ISTR1) = STRES(ISTR1)+DMATX(ISTR1,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

	DO 100 IINCS =1,NINCS	MST00560
C	*** READ DATA FOR CURRENT INCREMENT	MST00570
C	CALL INCREM(ELJAD,FIXED,IINCS,MELEM,MEVAB,NITER,MTOTV, MVFIX,NDOFN,NELEM,NEVAB,NOUTP,NOFIX,NTOTV, NVFIX,PRESC,RLJAD,TFACT,TLOAD,TOLER)	MST00580
		MST00590
C	*** LOOP OVER EACH ITERATION	MST00600
C	DO 50 IITER = 1,NITER	MST00610
C	*** CALL ROUTINE WHICH SELECTS SOLUTION ALGORITHM VIAIAULE KRESL	MST00620
C	CALL ALGOR(FIXED,IINCS,IITER,KRESL,MTOTV,NALGO, NTOTV)	MST00630
		MST00640
C	*** CHECK WHETHER A NEW EVALUATION OF THE STIFFNESS MATRIX IS REQUIRED	MST00650
C	IF (KRESL.EQ.1) CALL STIFP2(COORD,EPSTN,IINCS,LNODS,MATNO, MEVAB,MMATS,MPOIN,MTOTV,NELEM,NEVAB,NGAUS,NNODE, NSTRE,NSTR1,POSGP,PROPS,WEIGP,MELEM,MTOTG, STRSG,NTYPE,NCRIT,HVALU,PSTRN,NUMB, HARDS,PLSTN,NMATS)	MST00660
		MST00670
C	*** SOLVE EQUATIONS	MST00680
C	CALL FRONT(ASDIS,ELJAD,EQRHS,EQUAT,ESTIF,FIXED,IFFIX,IINCS,IITER, GLJAD,GSTIF,LOCEL,LNODS,KRESL,MBUFA,MELEM,MEVAB,MFRON, MSTIF,MTOTV,MVFIX,NACVA,NAMEV,NDEST,NDOFN,NELEM,NEVAB, NNODE,NOFIX,NPJOB,NPOIN,NTOTV,TDISP,TLOAD,TREAC, VECRV)	MST00690
		MST00700
C	*** CALCULATE RESIDUAL FORCES	MST00710
C	CALL RESID2(ASDIS,COORD,EFFST,ELJAD,FACT0,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)	MST00720
		MST00730
C	*** CHECK FOR CONVERGENCE	MST00740
C	CALL CONVER(ELJAD,IITER,LNODS,MELEM,MEVAB,MTOTV,NCHEK,NDOFN, NELEM,NEVAB,NNODE,NTOTV,PVALU,SFOR,TLOAD,TDFOR,TOLER)	MST00750
		MST00760
C	*** OUTPUT RESULTS IF REQUIRED	MST00770
C	IF (NITER.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)	MST00780
		MST00790
C	*** IF SOLUTION HAS CONVERGED STOP ITERATING AND OUTPUT RESULTS	MST00800
C	IF (NCHEK.EQ.0) GO TO 75	MST00810
		MST00820
		MST00830
		MST00840
		MST00850
		MST00860
		MST00870
		MST00880
		MST00890
		MST00900
		MST00910
		MST00920
		MST00930
		MST00940
		MST00950
		MST00960
		MST00970
		MST00980
		MST00990
		MST01000
		MST01010
		MST01020
		MST01030
		MST01040
		MST01050
		MST01060
		MST01070
		MST01080
		MST01090
		MST01100
		MST01110
		MST01120
		MST01130
		MST01140
		MST01150
		MST01160
		MST01170
		MST01180
		MST01190
		MST01200
		MST01210
		MST01220

```

SUBROUTINE DIMEN1 ( MBUFA,MELEM,MEVAR,MFRON,MMATS,MPOIN,MSTIF, MTOTG,
MTOTV,MVFIX,NDOFN,NPROP,NSTRE) DIM00010
C***** DIM00020
C***** THIS SUBROUTINE PRESETS VARIABLES ASSOCIATED WITH DYNAMIC DIMENSIONING DIM00030
C***** DIM00040
C***** DIM00050
C***** DIM00060
C***** DIM00070
C***** DIM00080
C***** DIM00090
C***** DIM00100
C***** DIM00110
C***** DIM00120
C***** DIM00130
C***** DIM00140
C***** DIM00150
C***** DIM00160
C***** DIM00170
C***** DIM00180
C***** DIM00190
C***** DIM00200
C***** DIM00210
C***** DIM00220
C***** DIM00230
C***** DIM00240
C***** DIM00250
C***** DIM00260
C***** DIM00270
C***** DIM00280
C***** DIM00290
C***** DIM00300
C***** DIM00310
C***** DIM00320
C***** DIM00330
C***** DIM00340
C***** DIM00350
C***** DIM00360
C***** DIM00370
C***** DIM00380
C***** DIM00390
C***** DIM00400
C***** DIM00410
C***** DIM00420
C***** DIM00430
C***** DIM00440
C***** DIM00450
C***** DIM00460
C***** DIM00470
C***** DIM00480
C***** DIM00490
C***** DIM00500
C***** DIM00510
C***** DIM00520
C***** DIM00530
C***** DIM00540
C***** DIM00550

```

```

SUBROUTINE INPUT2 ( COORD,IFFIX,LNJDS,MATNO,MELEM,MEVAR,MFRON, MNP00010
, MMATS,MPOIN,MTOTV,MVFIX,NALGO, INP00020
, NCRIT,NDFRO,NDOFN,MELEM, INP00030
, NEVAR,NGAUS,NGAU2, INP00040
, NINCS,MMATS,NNODE,NDFIX,NPOIN,NPROP,NSTRE,NSTR1, INP00050
, NTOTG,NTOTV,NTYPE,MVFIX,POSGP,PRESC,PROPS,WEIGP, INP00060
, USTRES,USTRN,NUMB) INP00070
C***** INP00080
C***** THIS SUBROUTINE ACCEPTS MOST OF THE INPUT DATA INP00090
C***** INP00100
C***** ** INP00110
C***** INP00120
C***** INP00130
C***** DIMENSION COORD(MPOIN,2), IFFIX(MTOTV),LNJDS(MELEM,9), INP00140
, MATNO(MELEM),NDFRO(MELEM), INP00150
, NDFIX(MVFIX),POSGP(4),PRESC(MVFIX,NDOFN), INP00160
, PROPS(MMATS,NPROP),TITLE(18),WEIGP(4), INP00170
, USTRES(MMATS,25),USTRN(MMATS,25) INP00180
REWIND 1 INP00190
REWIND 2 INP00200
REWIND 3 INP00210
REWIND 4 INP00220
REWIND 8 INP00230
READ (5,920) TITLE INP00240
WRITE (6,920) TITLE INP00250
920 FORMAT(18A4) INP00260
C***** INP00270
C***** READ THE FIRST DATA CARD AND ECHO IT IMMEDIATELY INP00280
C***** INP00290
READ(5,900) NPOIN,NELEM,MVFIX,NTYPE,NNODE,MMATS,NGAUS, INP00300
, NALGO,NCRIT,NINCS,NSTRE INP00310
900 FORMAT(11I5) INP00320
NEVAR = NDOFN*NNODE INP00330
NSTR1 = NSTRE+1 INP00340
IF (NTYPE.EQ.3) NSTR1 = NSTRE INP00350
NTOTV = NPOIN*NDOFN INP00360
NGAU2 = NGAUS*NGAUS INP00370
NTOTG = NELEM*NGAU2 INP00380
WRITE(6,901) NPOIN,NELEM,MVFIX,NTYPE,NNODE,MMATS,NGAUS,NEVAR, INP00390
, NALGO,NCRIT,NINCS,NSTRE INP00400
901 FORMAT (//8H NPOIN =,I4,4X,8H NELEM =,I4,4X,8H MVFIX =,I4,4X, INP00410
,8H NTYPE =,I4,4X,8H NNODE =,I4,4X, INP00420
,8H MMATS =,I4,4X,8H NGAUS =,I4, INP00430
,4X,8H NEVAR =,I4,4X,8H NALGO =,I4,4X, INP00440
,8H NCRIT =,I4,4X,8H NINCS =,I4,4X,8H NSTRE =,I4) INP00450
CALL CHECK1(NDOFN,NELEM,NGAUS,MMATS,NNODE,NPOIN, INP00460
, NSTRE,NTYPE,MVFIX,NCRIT,NALGO,NINCS) INP00470
C***** INP00480
C***** READ THE ELEMENT NODAL CONNECTIONS,AND THE PROPERTY NUMBERS INP00490
C***** INP00500
WRITE(6,902) INP00510
902 FORMAT (//8H ELEMENT,3X,8HPROPERTY,6X,12HNODE NUMBERS) INP00520
DO 3 IELM = 1,NELEM INP00530
READ(5,900) NUMEL,MATNO(NUMEL),(LNJDS(NUMEL,INODE),INODE=1,NNODE) INP00540
2 WRITE(6,903) NUMEL,MATNO(NUMEL),(LNJDS(NUMEL,INODE),INODE=1,NNODE) INP00550

```

```

903 FORMAT(1X,I5,I9,6X,B15)
C
C*** ZERO ALL THE NODAL COORDINATES,PRIOR TO READING SOME OF THEM*
DO 4 IPOINT = 1,NPOINT
  DO 4 IDIME = 1,2
    COORD(IPOINT,IDIME) = 0.0
C
C*** READ SOME NODAL COORDINATES,FINISHING WITH THE LAST NODE OF ALL.
WRITE(6,904)
904 FORMAT(//5H NODE,10X,1HX,10X,1HY)
6 READ(5,905) IPOINT,(COORD(IPOINT,IDIME),IDIME=1,2)
905 FORMAT(15,6F10.5)
IF (IPOINT.NE.NPOINT) GO TO 6
C
C*** INTERPOLATE COORDINATES OF MID-SIDE NODES
CALL NODEXY(COORD,LNODES,MELEM,MPOINT,NELEM,NNODE)
DO 10 IPOINT = 1,NPOINT
  10 WRITE(6,906) IPOINT,(COORD(IPOINT,IDIME),IDIME=1,2)
906 FORMAT(1X,I5,3F10.3)
C
C*** READ THE FIXED VALUES
WRITE(6,907)
307 FORMAT(//5H NODE,6X,4HCODE,6X,12HFIXED VALUES)
DO 8 IVPFIX = 1,NVPFIX
  READ(5,908) NDFIX(IVPFIX),IFPRE,(PRESC(IVPFIX,IODFN),IODFN=1,NDOFN)
  WRITE(6,908) NDFIX(IVPFIX),IFPRE,(PRESC(IVPFIX,IODFN),IODFN=1,NDOFN)
  NLOCA = (NDFIX(IVPFIX)-1)*NDOFN
  IFDOF = 10**((NDOFN-1)
  DO 8 IODFN = 1,NDOFN
    NGASH = NLOCA+IODFN
    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
WRITE(6,910)
910 FORMAT(//7H NUMBER,6X,18HELEMENT PROPERTIES)
DO 18 IMATS = 1,NMATS
  READ(5,910) NUMAT
  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
CALL UNIAx2(USTRES,USTRN,NUMB,MMATS,NMATS)
C
C*** SET UP GAUSSIAN TNTCGRATION CONSTANTS
CALL GAUSSO(NGAUS,POSOP,WEIGP)
CALL CHECK2(COORD,IFFIX,LNODES,MATNO,MELEM,MFRON,MPOINT,MIDTV,
  MVPFIX,NDFIX,NDOFN,NELEM,NMATS,NNODE,NDFIX,NPOINT,
  NVPFIX)
RETURN
END

```

```

INP00560
INP00570
INP00580
INP00590
INP00600
INP00610
INP00620
INP00630
INP00640
INP00650
INP00660
INP00670
INP00680
INP00690
INP00700
INP00710
INP00720
INP00730
INP00740
INP00750
INP00760
INP00770
INP00780
INP00790
INP00800
INP00810
INP00820
INP00830
INP00840
INP00850
INP00860
INP00870
INP00880
INP00890
INP00900
INP00910
INP00920
INP00930
INP00940
INP00950
INP00960
INP00970
INP00980
INP00990
INP01000
INP01010
INP01020
INP01030
INP01040
INP01050
INP01060
INP01070
INP01080
INP01090
INP01100
INP01120
INP01130
INP01140
INP01150
INP01160
INP01170
INP01180

```

```

SUBROUTINE UNIAX2 (USTRES, USTRN, NUMB, NMATS, NMATS)
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)
C DIMENSION USTRES(NMATS,25), USTRN(NMATS,25)
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** READ TEST DATA AND PRINT
C
  WRITE (6,200)
200 FORMAT (I10,28H THE UNIAXIAL TEST VALUES ARE//
  ,5X,6HNUMBER,4X,1H1,5X,9HSTRESS(I),5X,9HSTRAIN(I)//)
  DO 10 IMATS = 1, NMATS
    READ (5,100) NUMAT
    READ (5,110) PNUMH
  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 IHATS = 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
UNI00010
UNI00020
UNI00030
UNI00040
UNI00050
UNI00060
UNI00070
UNI00080
UNI00090
UNI00100
UNI00110
UNI00120
UNI00130
UNI00140
UNI00150
UNI00160
UNI00170
UNI00180
UNI00190
UNI00200
UNI00210
UNI00220
UNI00230
UNI00240
UNI00250
UNI00260
UNI00270
UNI00280
UNI00290
UNI00300
UNI00310
UNI00320
UNI00330
UNI00340
UNI00350
UNI00360
UNI00370
UNI00380
UNI00390
UNI00400
UNI00410
UNI00420
UNI00430

```

```

SUBROUTINE HRDNC (USTRES, USTRN, NUMB, PROPS, MMATS, NMATS,
                 HVALU, YSTRES, PSTRN)
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)
C DIMENSION USTRES(MMATS,25), USTRN(MMATS,25), PROPS(MMATS,6),
C           YSTRES(MMATS,25), PSTRN(MMATS,25), HVALU(MMATS,25)
C*** READ UNIAXIAL PROPERTIES
C
C DO 10 IMATS = 1, NMATS
C   YOUNG = PROPS (IMATS,1)
C   YIELD = PROPS (IMATS,5)
C*** INITIALIZE THE HVALU, PSTRN, YSTRES ARRAYS
C
C   MCOL = 25
C   DO 50 ICOL = 1, MCOL
C     HVALU (IMATS, ICOL) = 0.0
C     PSTRN (IMATS, ICOL) = 0.0
C     YSTRES (IMATS, ICOL) = 0.0
C   50 CONTINUE
C*** CHANGE THE TEST DATA TO STRESS-PLASTIC STRAIN VALUES
C IF NOT PERFECTLY PLASTIC
C
C   KOUNT = 1
C   DO 20 I = 1, NUMB
C     IF (USTRES (IMATS, I) .LT. YIELD) GO TO LO
C     YSTRES (IMATS, KOUNT) = USTRES (IMATS, I)
C     PSTRN (IMATS, KOUNT) = (USTRN (IMATS, I) - (USTRES (IMATS, I) / YOUNG))
C     KOUNT = KOUNT + 1
C   20 CONTINUE
C   KOUNT = KOUNT - 1
C*** CALCULATE THE HVALU, THE HARDNESS VALUES FROM THE SLOPE OF
C STRESS VS EFFECTIVE PLASTIC STRAIN CURVE
C
C   HVALU (IMATS, 1) = DABS ((YSTRES (IMATS, 1) - YIELD) / (PSTRN (IMATS, 1)))
C   K = KOUNT
C   DO 30 J = 2, KOUNT
C     HVALU (IMATS, J) = DABS ((YSTRES (IMATS, J) - YSTRES (IMATS, (J-1))) /
C                             (PSTRN (IMATS, J) - PSTRN (IMATS, (J-1))))
C   30 CONTINUE
C   10 CONTINUE
C   FOR TESTING
C   WRITE (C, 100)
C   100 FORMAT (1H0, 2X, 8HMATERIAL, 4X, 6HNUMBER, 1X, 5HHARRAY, 5X,
C             20MPLASTIC STRAIN ARRAY)
C   DO 35 IMATS = 1, NMATS
C
C     DO 45 INUMB = 1, NUMB
C       WRITE (C, 110) IMATS, INUMB, HVALU (IMATS, INUMB), PSTRN (IMATS, INUMB)
C     115 FORMAT (1H0, 5X, 15, 5X, 15, 5X, E14.8, 5X, F14.8)
C     45 CONTINUE
C   35 CONTINUE
C TESTING OVER
C RETURN
C END

```

```

HRD00010
HRD00020
HRD00030
HRD00040
HRD00050
HRD00060
HRD00070
HRD00080
HRD00090
HRD00100
HRD00110
HRD00120
HRD00130
HRD00140
HRD00150
HRD00160
HRD00170
HRD00180
HRD00190
HRD00200
HRD00210
HRD00220
HRD00230
HRD00240
HRD00250
HRD00260
HRD00270
HRD00280
HRD00290
HRD00300
HRD00310
HRD00320
HRD00330
HRD00340
HRD00350
HRD00360
HRD00370
HRD00380
HRD00390
HRD00400
HRD00410
HRD00420
HRD00430
HRD00440
HRD00450
HRD00460
HRD00470
HRD00480
HRD00490
HRD00500
HRD00510
HRD00520
HRD00530
HRD00540
HRD00550
HRD00560
HRD00570
HRD00580
HRD00590
HRD00600
HRD00610
HRD00620
HRD00630

```

```

SUBROUTINE LOADP1(COORD,LNODES,MATNO,MELEM,MMATS,MPOIN,NELEM,
NEVAR,NGAUS,NNODE,NPOIN,NSTRE,NTYPE,POSGP,
PROPS,RLOAD,WEIG,NDOFN)
C*****
C*** THIS SUBROUTINE EVALUATES THE CONSISTENT NODAL FORCES FOR EACH
ELEMENT
C*****
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION CARTD(2,9),COORD(MPOIN,2),DERIV(?,9),DGASH(2),
MATX(4,4),ELCDD(2,9),LNODS(MELEM,9),MATNO(MELEM),
NUPRS(4),PGASH(2),POINT(2),POSGP(4),PRESS(4,2),
PROPS(MMATS,6),RLOAD(MELEM,18),SHAPE(9),STRAN(4),
STRES(4),TITLE(18),
WEIG(4),GPCDD(2,9)
TWOPT = 6.283185308
DO 10 IFLEM = 1,NELEM
DO 10 IFVAR = 1,NEVAR
10 RLOAD(IFLEM,IEVAR) = 0.0
901 READ(5,901) TITLE
901 FORMAT(18A4)
903 WRITE(6,903) TITLE
903 FORMAT(1H0,18A4)
C*** READ DATA CONTROLLING LOADING TYPES TO BE INPUTED
READ(5,919) IPLUD,IGRAV,IEDGE
WRITE(6,919) IPLUD,IGRAV,IEDGE
919 FORMAT(3I5)
C*** READ NODAL POINT LOADS
IF (IPLUD.FO.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(15,2F10.3)
C*** ASSOCIATE THE NODAL POINT LOADS WITH AN ELEMENT
DO 30 IFLEM = 1,NELEM
DO 30 INDOE = 1,NNODE
NLCCA = IABS(LNODES(IFLEM,INDOE))
IF (LODPT.ND.NLCCA) GO TO 40
30 CONTINUE
40 DO 50 IDOFN = 1,2
NGASH = (INDOE-1)*2+IDOFN
50 RLOAD(IFLEM,NGASH) = POINT(IDOFN)
IF (LODPT.LT.NPOIN) GO TO 20
500 CONTINUE
IF (IGRAV.FO.0) GO TO 600
C*** GRAVITY LOADING SECTION

```

```

LDA00010
LDA00020
LDA00030
LDA00040
LDA00050
LDA00060
LDA00070
LDA00080
LDA00090
LDA00100
LDA00110
LDA00120
LDA00130
LDA00140
LDA00150
LDA00160
LDA00170
LDA00180
LDA00190
LDA00200
LDA00210
LDA00220
LDA00230
LDA00240
LDA00250
LDA00260
LDA00270
LDA00280
LDA00290
LDA00300
LDA00310
LDA00320
LDA00330
LDA00340
LDA00350
LDA00360
LDA00370
LDA00380
LDA00390
LDA00400
LDA00410
LDA00420
LDA00430
LDA00440
LDA00450
LDA00460
LDA00470
LDA00480
LDA00490
LDA00500
LDA00510
LDA00520
LDA00530
LDA00540
LDA00550

```

```

C*** HLAD GRAVITY ANGLE AND GRAYITATIONAL CUNSTANT
C
C   READ (5,905) THETA,GRAVY
906 FORMAT(2F10.3)
C   WRITE (6,911) THETA,GRAVY
911 FORMAT(1H0,16H GRAVITY ANGLE =,F10.3,19H GRAVITY CUNSTANT =,F10.3)
C   THETA = THETA/57.295779514
C*** LOOP OVER EACH ELEMENT
C
C   DO 90 ILLM = 1,NELEM
C*** SET UP PRELIMINARY CONSTANTS
C
C   LPROP = MATND(ILEM)
C   THICK = PROPS(LPROP,3)
C   DENSE = PROPS(LPROP,4)
C   IF (DENSE.EQ.0.0) GO TO 90
C   GXCOM = DENSE*GRAVY*DSIN(THETA)
C   GYCOM = DENSE*GRAVY*DCOS(THETA)
C*** COMPUTE COORDINATES OF THE ELEMENT NODAL POINTS
C
C   DO 60 INODE = 1,NNODE
C     LNODE = TABS(LNODES(IELEM,INODE))
C     DO 60 IDIME = 1,2
C       ELCOD(IDIME,INODE) = COORD(LNODE,IDIME)
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
C
C   KGASP = 0
C   DO 80 IGAUS = 1,NGAUS
C     DO 80 JGAUS = 1,NGAUS
C     EXTSP = PQSCP(IGAUS)
C     ETASP = PQSCP(JGAUS)
C*** COMPUTE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS AND ELEMENTAL
C   VOLUME
C
C   CALL SFR2 (DERIV,ETASP,EXISP,NNODE,SHAPE)
C   KGASP = KGASP+1
C   CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCOD,IELEM,KGASP,
C     NNODE,SHAPE)
C   OVULU = DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
C   IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
C   IF (NTYPE.EQ.3) DVOLU = DVOLU*TWOP1*GPCJQ(1,KGASP)
C*** CALCULATE LOADS AND ASSOCIATE WITH ELEMENT NODAL POINTS
C
C   DO 70 INODE = 1,NNODE
C     NGASH = (INODE-1)*2+1
C     MGASH = (INODE-1)*2+2
C     RLOAD(ILEM,NGASH) = RLOAD(IELEM,NGASH)+GXCOM*SHAPE(INODE)*DVOLU
C     RLOAD(ILEM,MGASH) = RLOAD(IELEM,MGASH)+GYCOM*SHAPE(INODE)*DVOLU
70 CONTINUE
80 CONTINUE

```

```

LQA00560
LQA00570
LQA00580
LQA00590
LQA00600
LQA00610
LQA00620
LQA00630
LQA00640
LQA00650
LQA00660
LQA00670
LQA00680
LQA00690
LQA00700
LQA00710
LQA00720
LQA00730
LQA00740
LQA00750
LQA00760
LQA00770
LQA00780
LQA00790
LQA00800
LQA00810
LQA00820
LQA00830
LQA00840
LQA00850
LQA00860
LQA00870
LQA00880
LQA00890
LQA00900
LQA00910
LQA00920
LQA00930
LQA00940
LQA00950
LQA00960
LQA00970
LQA00980
LQA00990
LQA01000
LQA01010
LQA01020
LQA01030
LQA01040
LQA01050
LQA01060
LQA01070
LQA01080
LQA01090
LQA01100

```

```

90 CONTINUE
600 CONTINUE
  IF (IEDGE.EQ.0) GO TO 700
C*** DISTRIBUTED EDGE LOADS SECTION
  READ (5,932) NEDGE
932 FORMAT(15)
  WRITE (6,912) NEDGE
912 FORMAT (1H0,5X,21HNO. OF LOADED EDGES =,15)
  WRITE (6,915)
915 FORMAT (1H0,5X,38HLIST OF LOADED EDGES AND APPLIED LOADS)
  NNODE = 3
  NCODE = NNODE
  IF (NNODE.EQ.4) NNODE = 2
  IF (NNODE.EQ.9) NCODE = 8
C*** LOOP OVER EACH LOADED EDGE
  DO 140 IEDGE = 1,NEDGE
C*** READ DATA LOCATING THE LOADED EDGE AND APPLIED LOAD
  READ (5,902) NEASS,(NPRS(IDEDE),IDEDE=1,NODEG)
902 FORMAT(415)
  WRITE (6,913) NEASS,(NPRS(IDEDE),IDEDE=1,NODEG)
913 FORMAT (1H0,5X,315)
  READ (5,914) ((PRESS(IDEDE,IODFN),IODFN=1,2),IDEDE=1,NODEG)
914 FORMAT(6F10.3)
  ETASP = -1.0
C*** CALCULATE THE COORDINATES OF THE NODES OF THE ELEMENT EDGE
  DO 100 IODEG = 1,NODEG
  LNODE = NPRS(IDEDE)
  DO 100 IOIME = 1,2
  100 ELCOO(IOIME,IODEG) = COORD(LNODE,IOIME)
C*** ENTER LOOPS FOR LINEAR NUMERICAL INTEGRATION
  DO 150 IGAUS = 1,NGAUS
  EXISP = POSGP(IGAUS)
C*** EVALUATE THE SHAPE FUNCTIONS AT THE SAMPLING POINTS
  CALL SFR2 (DERIV,ETASP,EXISP,NNODE,SHAPE)
C*** CALCULATE COMPONENTS OF THE EQUIVALENT NODAL LOADS
  DO 110 IODFN = 1,2
  PGASH (IODFN)=0.0
  DGASH (IODFN) =0.0
  DO 110 IODEG = 1,NODEG
  PGASH(IODFN) = PGASH(IODFN)+PRESS(IDEDE,IODFN)+SHAPE(IDEDE)
  110 DGASH (IODFN) = DGASH(IODFN)+ELCOO(IODFN,IODEG)*DERIV(1,IODEG)
  DVOLU = WEIGP(IGAUS)
  PYCOM = 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(IDEDE)*ELCOO(1,IODEG)
  DVOLU = DVOLU+TWOPI*RADUS
  115 CONTINUE
C*** ASSOCIATE THE EQUIVALENT NODAL EDGE LOADS WITH AN ELEMENT
  DO 120 INODE = 1,NNODE
  NLOCA = IABS(LNODES(NEASS,INODE))
  IF (NLOCA.EQ.NPRS(1)) GO TO 130
  120 CONTINUE
  130 JNODE = INODE+NODEG-1
  KOUNT = 0
  DO 140 KNODE = INODE,JNODE
  KDUNT = KOUNT+1
  NGASH = (KNODE-1)*NODEG+1
  HGASH = (KNODE-1)*NODEG+2
  IF (KNODE.GT.NCODE) NGASH = 1
  IF (KNODE.GT.NCODE) HGASH = 2
  RLOAD(NEASS,NGASH) = RLOAD(NEASS,NGASH)+SHAPE(KOUNT)*PYCOM*DVOLU
  140 RLOAD(NEASS,HGASH)=RLOAD(NEASS,HGASH)+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,14,5X,8E12.4/(10X,8E12.4))
  RETURN
  END
LJAO1110
LJAO1120
LJAO1130
LJAO1140
LJAO1150
LJAO1160
LJAO1170
LJAO1180
LJAO1190
LJAO1200
LJAO1210
LJAO1220
LJAO1230
LJAO1240
LJAO1250
LJAO1260
LJAO1270
LJAO1280
LJAO1290
LJAO1300
LJAO1310
LJAO1320
LJAO1330
LJAO1340
LJAO1350
LJAO1360
LJAO1370
LJAO1380
LJAO1390
LJAO1400
LJAO1410
LJAO1420
LJAO1430
LJAO1440
LJAO1450
LJAO1460
LJAO1470
LJAO1480
LJAO1490
LJAO1500
LJAO1510
LJAO1520
LJAO1530
LJAO1540
LJAO1550
LJAO1560
LJAO1570
LJAO1580
LJAO1590
LJAO1600
LJAO1610
LJAO1620
LJAO1630
LJAO1640
LJAO1650
LJAO1660
LJAO1670
LJAO1680
LJAO1690
LJAO1700
LJAO1710
LJAO1720
LJAO1730
LJAO1740
LJAO1750
LJAO1760
LJAO1770
LJAO1780
LJAO1790
LJAO1800
LJAO1810
LJAO1820
LJAO1830
LJAO1840
LJAO1850
LJAO1860
LJAO1870
LJAO1880
LJAO1890
LJAO1900
LJAO1910
LJAO1920
LJAO1930
LJAO1940
LJAO1988
LJAO1970
LJAO1980
LJAO1990
LJAO2000
LJAO2010
LJAO2020

```



```

      SUBROUTINE STIFP2(COORD,EPSTN,IINCS,LNODS,MATNO,NEVAB,MMATS,
      * MPOIN,MTOTV,NELEM,NEVAH,NGAUS,NNODE,NSTRE,
      * NSTRI,POS GP,PROPS,WEIGP,MELEM,MTOTG,
      * STRSG,NTYPE,NCRIT,HVALU,
      * PSTRN,NUMB,HARDS,PLSTN,MMATS)
      *****
C*** THIS SUBROUTINE EVALUATES THE STIFFNESS MATRIX FOR EACH ELEMENT
C IN TURN
C*****
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION MMATX(4,18),CARTO(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),POS GP(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
      REUINO = 1
      KGAUS = 0
C*** LOOP OVER EACH ELEMENT
      DO 70 IELEM = 1,NELEM
      LPRPP = MATNO(IELEM)
C*** EVALUATE THE COORDINATES OF THE ELEMENT NODAL POINTS
      DO 10 INODE = 1,NNODE
      LNODE = IAB(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*** INITIALIZE THE ELEMENT STIFFNESS MATRIX
      DO 20 IEVAR = 1,NEVAB
      DO 20 JEVAB = 1,NEVAB
20  ESTIF(IEVAR,JEVAB) = 0.0
      KGASP = 0
C*** ENTER LOOPS FOR AREA NUMERICAL INTEGRATION
      DO 50 IGAUS = 1,NGAUS
      EXISP = POS GP(IGAUS)
      DO 50 JGAUS = 1,NGAUS
      ETASP = POS GP(JGAUS)
      KGASP = KGASP + 1
      KGAUS = KGAUS + 1
C

```

```

STI00010
STI00020
STI00030
STI00040
STI00050
STI00060
STI00070
STI00080
STI00090
STI00100
STI00110
STI00120
STI00130
STI00140
STI00150
STI00160
STI00170
STI00180
STI00190
STI00200
STI00210
STI00220
STI00230
STI00240
STI00250
STI00260
STI00270
STI00280
STI00290
STI00300
STI00310
STI00320
STI00330
STI00340
STI00350
STI00360
STI00370
STI00380
STI00390
STI00400
STI00410
STI00420
STI00430
STI00440
STI00450
STI00460
STI00470
STI00480
STI00490
STI00500
STI00510
STI00520
STI00530
STI00540
STI00550

```

```

C*** EVALUATE THE D-MATRIX
C      CALL MDPS1(DMATX,LPROP,MMATS,NTYPE,PROPS)
C*** EVALUATE THE SHAPC FUNCTIONS,ELEMENTAL VOLUME ETC.
C      CALL SH2P(DERIV,ETASP,EXISP,NNODE,SHAPE)
C      CALL JACOB2(CARTD,DERIV,DJACB,ELCDD,GPCDD,IELEM,KGASP,
C              NNODE,SHAPE)
C      DVOLU = DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
C      IF (NTYPE.EQ.3) DVOLU = DVOLU*TWOP1-GPCDD(1,KGASP)
C      IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
C*** EVALUATE THE B AND OIL MATRICES
C      CALL HMATPS(BMATX,CARTD,NNODE,SHAPE,GPCDD,NTYPE,KGASP)
C      IF (TIMES.EQ.1) GO TO 80
C      IF (PSTRN(KGAUS).EQ.0.0) GO TO 80
C      DO 90 ISTR1 = 1,NSTR1
90     STRS(ISTR1) = STRSG(ISTR1,KGAUS)
C      CALL INVAR1(DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,STRES,
C              THETA,VARJ2,YIELD)
C      CALL YIELD1(AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
C              PROPS,SINT3,STEFF,THETA,VARJ2)
C      PLSTN = PSTRN(KGAUS)
C      CALL HNUMB (PSTRN,HVALU,NUMB,MMATS,LPROP,
C              PLSTN,HARDS,PROPS,MMATS)
C      CALL FLDWP1(AVECT,ABETA,DVECT,NTYPE,PROPS,LPROP,NSTR1,MMATS,
C              HARDS)
C      DO 106 ISTR2 = 1,NSTR2
C      DO 100 JSTRC = 1,NSTRC
100     DMATX(ISTRC,JSTRC) = DMATX(ISTR2,JSTR2) - ABETA * DVECT(ISTR2) *
C      * DVECT(JSTR2)
C      DO 80 CONTINUE
C      CALL DDF(DMATX,DBMAT,DBMAT,MEVAB,NEVAB,NSTR2,NSTR1)
C*** CALCULATE THE ELEMENT STIFFNESSES
C      DO 30 IEVAB = 1,NEVAB
C      DO 30 JEVAB = IEVAB,NEVAB
C      DO 30 ISTR2 = 1,NSTR2
30     ESTIF(IEVAB,JEVAB) = ESTIF(IEVAB,JEVAB) + BMATX(ISTR2,IEVAB) *
C      * DBMAT(ISTR2,JEVAB) * DVOLU
C      DO 50 CONTINUE
C*** CONSTRUCT THE LOWER TRIANGLE OF THE STIFFNESS MATRIX
C      DO 60 IEVAB = 1,NEVAB
C      DO 60 JEVAB = 1,NEVAB
60     ESTIF(JEVAB,IEVAB) = ESTIF(IEVAB,JEVAB)
C*** STORE THE STIFFNESS MATRIX,STRESS MATRIX,AND SAMPLING POINT
C      COORDINATES FOR EACH ELEMENT ON DISC FILE
C      WRITE(1) ESTIF
C      CONTINUE
C      RETURN
C      END

```

ST100560
ST100570
ST100580
ST100590
ST100600
ST100610
ST100620
ST100630
ST100640
ST100650
ST100660
ST100670
ST100680
ST100690
ST100700
ST100710
ST100720
ST100730
ST100740
ST100750
ST100760
ST100770
ST100780
ST100790
ST100800
ST100810
ST100820
ST100830
ST100840
ST100850
ST100860
ST100870
ST100880
ST100900

ST100910
ST100920
ST100930
ST100940
ST100950
ST100960
ST100970
ST100980
ST100990
ST101000
ST101010
ST101020

ST101040
ST101050
ST101060
ST101070
ST101080
ST101090
ST101100

ST101110
ST101120
ST101130

```

SUBROUTINE FLDWP1(AVECT,ABETA,DVECT,NTYPE,PROPS,
LPROP,NSTR1,MMATS,HARDS)
C*****
C*** THIS SUBROUTINE EVALUATES THE PLASTIC D VECTOR
C*****
IMPLICIT REAL*8(A-H,D-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
80 DENOM = DENOM+AVECT(ISTR1)*DVECT(ISTR1)
ABETA = 1.0/DENOM
RETURN
END

```

FLD00010
FLD00020
FLD00030
FLD00040
FLD00050
FLD00060
FLD00070
FLD00080
FLD00090
FLD00100
FLD00110
FLD00120
FLD00130
FLD00140
FLD00150
FLD00160
FLD00170
FLD00180
FLD00190
FLD00200
FLD00210
FLD00220
FLD00230
FLD00240
FLD00250
FLD00260
FLD00270
FLD00280
FLD00290
FLD00300
FLD00310

```

SUBROUTINE RESID2(ASDIS,COORD,EFFST,ELOAD,FACT0,ITER,LNODS, RES00010
, LPROP,MATNO,MELEM,MMATS,MPDIN,MTOTG,MTOTV,NDQFN, RES00020
, NELEM,NEVAB,NGAUS,NNODE,NSTR1,NTYPE,POSGP,PROPS, RES00030
, NSTRE,NCRIT,STRSG,WEIGP,TDISP,EPSTN, RES00040
, HVALU,PSTRN,NUMB, RES00050
, HAROS,PLSTN,MMATS) RES00060
C***** RES00070
C*** THIS SUBROUTINE REDUCES THE STRESSES TO THE YIELD SURFACE AND RES00080
EVALUATES THE EQUIVALENT NODAL FORCES RES00090
C***** RES00100
IMPLICIT REAL*8(A-H,O-Z) RES00110
DIMENSION ASDIS(MTOTV),AVECT(4),CARTO(2,9),COORD(MPDIN,2), RES00120
, DLVIA(4),DVECT(4),EFFST(MTOTG),ELCOD(2,9),ELDIS(2,9), RES00130
, ELOAD(MELEM,18),LNODS(MELEM,9),POSGP(4),PROPS(MMATS,6), RES00140
, STRAN(4),STRES(4),STRSG(4,MTOTG), RES00150
, WEIGP(4),DLCOD(2,9),DESIG(4),SIGMA(4),SGTOT(4), RES00160
, DMATX(4,4),DCRIV(2,9),SHAPE(9),GPCOD(2,9), RES00170
, EPSTN(MTOTG),TDISP(MTOTV),MATNO(MELEM),HMATX(4,18), RES00180
, HVALU ( MMATS,25 ), RES00190
, PSTRN (MMATS,25) RES00200
ROOT3 = 1.73205080757 RES00210
TWOPI = 6.283185308 RES00220
DO 10 IELEM = 1,NELEM RES00230
DC 10 IEVAB = 1,NEVAB RES00240
10 ELOAD(IELEM,IEVAB) = 0.0 RES00250
KGAUS = 0 RES00260
DO 20 IELEM = 1,NELEM RES00270
LPROP = MATNO(IELEM) RES00280
UNIAX = PROPS(LPROP,5) RES00290
FRICT = PROPS(LPROP,6) RES00300
IF (NCRIT.EQ.3) UNIAX =PROPS(LPROP,5)*DCOS(FRICT*0.017453292) RES00310
IF (NCRIT.EQ.4) UNIAX =6.0*PROPS(LPROP,5)*DCOS(FRICT*0.017453292)/ RES00320
* (ROOT3*(3.0-DCOS(FRICT*0.017453292))) RES00330
C*** COMPUTE COORDINATE AND INCREMENTAL DISPLACEMENTS OF THE RES00340
ELEMENT NODAL POINTS RES00350
C***** RES00360
DO 30 INODE = 1,NNODE RES00370
LNODE = IABS(LNODS(IELEM,INODE)) RES00380
NPQSN = (LNODE-1)*NDQFN RES00390
DO 30 IDQFN = 1,NDQFN RES00400
NPQSN = NPQSN+1 RES00410
ELCOD(IDQFN,INODE)=COORD(LNODE,IDQFN) RES00420
30 ELDIS(IDQFN,INODE)=ASDIS(NPQSN) RES00430
CALL MODPS1(DMATX,LPROP,MMATS,NTYPE,PROPS) RES00440
THICK = PROPS(LPROP,3) RES00450
KGAUS = 0 RES00460
DO 40 JGAUS = 1,NGAUS RES00470
DC 40 JGAUS = 1,NGAUS RES00480
EXITSP = POSGP(JGAUS) RES00490
ETASP = POSGP(JGAUS) RES00500
KGAUS = KGAUS+1 RES00510
KGASP = KGASP+1 RES00520
RES00530
RES00540
RES00550

```

```

ESTKS = EFFST (KGAUS)
PSTND = EPSTN (KGAUS)
PLSTN = PSTND
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 SFK2 (DRIV,ETASP,EXISP,NNODE,SHAPE)
CALL JACOB2 (CARTD,DERIV,DJACB,ELCOD,GPCUD,TELEM,KGASP,
NNODE,SHAPE)
DVOLU = DJACH*WEICP(KGAUS)+WEICP(JGAUS)
IF (NTYPE.EQ.3) DVJLU = DVOLU*TWOP1*GPCUD(1,KGASP)
IF (THICK.NE.0.0) DVOLU = DVOLU*THICK
CALL BMATPS (BMATX,CARTD,NNODE,SHAPE,GPCUD,NTYPE,KGASP)
CALL LINER1 (CARTD,DMATX,ELDIS,LPROP,MMATS,NDUFN,NNODE,NSTRE,
NTYPE,PROPS,STRAN,STRES,KGASP,GPCUD,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 INVARI ( 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) GJ TJ 60
REACT = ESCUR/(YIELD-EFFST(KGAUS))
GO TO 70
50 ESCUR = YIELD-EFFST(KGAUS)
IF (ESCUR.LE.0.0) GO TO 50
REACT = 1.0
70 MSTEP = ESCUR*8.0/PREYS+1.0
ASTEP = MSTEP
REDUC = 1.0-REACT
DO 80 ISTR1 = 1,NSTR1
SGTGT(ISTR1) = STRSG(ISTR1,KGAUS)+REDUC*STRES(ISTR1)
80 STRES(ISTR1) = REACT*STRES(ISTR1)/ASTEP
DO 90 ISTR1 = 1,NSTR1
CALL INVARI ( 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
110 BGASH = BGASH+AVECT(ISTR1)+SGTOT(ISTR1)
SGTOT(ISTR1) = SGTGT(ISTR1)+STRES(ISTR1)-DLAMD*DVECT(ISTR1)
EPSTN(KGAUS) = EPSTN(KGAUS)+ DLAMD*BGASH/YIELD
GO CONTINUE
PSTNN = EPSTN (KGAUS)
CALL INVARI ( DEVIA,LPROP,MMATS,NCRIT,PROPS,SINT3,STEFF,SGTOT,
THETA,VARJ2,YIELD)
DEPSTN = PSTNN - PSTND
PLSTN = PSTNN
IF (PLSTN.EQ.0.0) HAROS = 0.0
IF (PLSTN.GT.0.0) CALL
HNUMB (PSTRN,HVALU,NUMB,MMATS,LPROP,PLSTN,HARDS,PROPS,NMATS)
CURYS = PREYS
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
ELEMENT NODES
190 MGASH = 0
DO 140 INODE = 1,NNODE
DO 140 IDUFN = 1,NDUFN
MGASH = MGASH+1
DO 140 ISTR1 = 1,NSTRE
140 ELDAD(IELEM,MGASH) = ELDAD(IELEM,MGASH)+BMATX(ISTR1,MGASH)
* STRSG(ISTR1,KGAUS)*DVOLU
to CONTINUE
20 CONTINUE
RETURN
END
RES00560
RES00570
RES00580
RES00590
RES00600
RES00610
RES00620
RES00630
RES00640
RES00650
RES00660
RES00670
RES00680
RES00690
RES00700
RES00710
RES00720
RES00730
RES00740
RES00750
RES00760
RES00770
RES00780
RES00790
RES00800
RES00810
RES00820
RES00830
RES00840
RES00850
RES00860
RES00870
RES00880
RES00890
RES00900
RES00910
RES00920
RES00930
RES00940
RES00950
RES00960
RES00970
RES00980
RES00990
RES01000
RES01010
RES01020
RES01030
RES01040
RES01050
RES01060
RES01070
RES01080
RES01090
RES01100
RES01110
RES01120
RES01130
RES01140
RES01150
RES01160
RES01170
RES01180
RES01190
RES01200
RES01210
RES01220
RES01230
RES01240
RES01250
RES01260
RES01270
RES01280
RES01290
RES01300
RES01310
RES01320
RES01330
RES01340
RES01350
RES01360
RES01370
RES01380
RES01390
RES01400
RES01410
RES01420
RES01430
RES01440
RES01450
RES01460

```

```

SUBROUTINE LINER1(CARTD,DMATX,ELDIS,LPRJP,MMATS,NDOFN,NNODE,NSTRE, LIN00010
NTYPE,PROPS,STRAN,STRES,KGASP,GPCDD,SHAPE) LIN00020
C***** LIN00030
C***** LIN00040
C*** THIS SUBROUTINE EVALUATES STRESSES AND STRAINS ASSUMING LINEAR ELASTIC BEHAVIOUR LIN00050
C***** LIN00060
C***** LIN00070
C***** LIN00080
C***** LIN00090
IMPLICIT REAL*8(A-H,O-Z) LIN00100
DIMENSION AGASH(2,2),CARTD(2,9),DMATX(4,4),ELDIS(2,9), LIN00110
PROPS(MMATS,6),STRAN(4),STRES(4), LIN00120
GPCDD(2,9),SHAPE(9) LIN00130
POISS = PROPS(LPROP,2) LIN00140
DO 20 IJDFN = 1,NDOFN LIN00150
OJ = IJDFN
DO 20 JDFN = 1,NDOFN LIN00160
IJ = JDFN
BGASH = 0.0 LIN00170
ON 10 INDFN = 1,NNODE LIN00180
10 BGASH = BGASH+CARTD(IJDFN,IJDFN)*ELDIS(INDFN,INUDE) LIN00190
20 AGASH(INDFN,IJDFN) = MGASH LIN00200
C***** LIN00210
C***** LIN00220
C***** LIN00230
C*** CALCULATE THE STRAINS LIN00240
C***** LIN00250
C***** LIN00260
C***** LIN00270
C***** LIN00280
C***** LIN00290
C***** LIN00300
C***** LIN00310
C***** LIN00320
C***** LIN00330
C***** LIN00340
C***** LIN00350
C***** LIN00360
C***** LIN00370
C***** LIN00380
C***** LIN00390
STRAN(1) = AGASH(1,1)
STRAN(2) = AGASH(2,2)
STRAN(3) = AGASH(1,2)+AGASH(2,1)
STRAN(4) = 0.0
DO 30 INUDE = 1,NNODE
30 STRAN(4) = STRAN(4)+ELDIS(1,INUDE)*SHAPE(INUDE)/GPCDD(1,KGASP)
C***** AND THE CORRESPONDING STRESSES
C*****
DO 40 IJSTRE = 1,NSTRE
STRES(IJSTRE) = 0.0
DO 40 JSTRE = 1,NSTRE
40 STRES(IJSTRE) = STRES(IJSTRE)+DMATX(IJSTRE,IJSTRE)*STRAN(IJSTRE)
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, HNU00010
PROPS,MMATS) HNU00020
C***** HNU00030
C***** HNU00040
C***** HNU00050
C***** HNU00060
C***** HNU00070
C***** HNU00080
C***** HNU00090
C***** HNU00100
C***** HNU00110
C***** HNU00120
C***** HNU00130
C***** HNU00140
C***** HNU00150
C***** HNU00160
C***** HNU00170
C***** HNU00180
C***** HNU00190
C***** HNU00200
C***** HNU00210
C***** HNU00220
C***** HNU00230
C***** HNU00240
C***** HNU00250
C***** HNU00260
C***** HNU00270
C***** HNU00280
C***** HNU00290
C***** HNU00300
C***** HNU00310
C***** HNU00320
C***** HNU00330
C*** THIS SUBROUTINE CHOOSES THE HARDNESS VALUE ,HARDS,DEPENDING ON THE TOTAL EFFECTIVE PLASTIC STRAIN
C*****
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION HVALU(MMATS,25),PSTRN(MMATS,25),
PROPS(MMATS,6)
C*****
C*** FOR THE EFFECTIVE PLASTIC STRAIN,PLSTN,CHOOSE THE HARDNESS VALUE
C*****
IF (PLSTN.LE.0.0) GO TO 5
IF (PLSTN.LE.PSTRN(LPROP,1)) GO TO 15
DO 13 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 HARDC = 0.0
RETURN
15 HARDC = HVALU(LPROP,1)
RETURN
25 HARDC = HVALU(LPROP,INDEX)
RETURN
35 HARDC = HVALU(LPROP,NUMB)
RETURN
END

```

```

SUBROUTINE INVAR1( DEVI, LPROP, MMATS, NCRIT, PROPS, SINT3, STEFF, INVO0010
  STEMP, THETA, VARJ2, YIELD) INVO0020
C..... INVO0030
C*** THIS SUBROUTINE EVALUATES THE STRESS INVARIANTS AND THE CURRENT INVO0040
  VALUE OF THE YIELD FUNCTION INVO0050
C..... INVO0060
  IMPLICIT REAL*8(A-H,I,J-Z) INVO0070
  DIMENSION DEVI(4), PROPS(MMATS,6), STEMP(4) INVO0080
  ROOT3 = 1.73205080757 INVO0090
  SMEAN = (STEMP(1)+STEMP(2)+STEMP(4))/3.0 INVO0100
  DEVI(1) = STEMP(1)-SMEAN INVO0110
  DEVI(2) = STEMP(2)-SMEAN INVO0120
  DEVI(3) = STEMP(3) INVO0130
  DEVI(4) = STEMP(4)-SMEAN INVO0140
  VARJ2 = DEVI(3)*DEVI(3)+0.5*(DEVI(1)*DEVI(1)+DEVI(2)* INVO0150
    DEVI(2)+DEVI(4)*DEVI(4)) INVO0160
  VARJ3 = DEVI(4)*DEVI(4)*DEVI(4)-VARJ2 INVO0170
  STEFF = DSQRT(VARJ2) INVO0180
  IF (STEFF.EQ.0.0) GO TO 10 INVO0190
  SINT3 = -3.0*ROOT3+VARJ3/(2.0+VARJ2*STEFF) INVO0200
  IF (SINT3.GT.1.0) SINT3 = 1.0 INVO0210
  GO TO 20 INVO0220
10 SINT3 = 0.0 INVO0230
20 CONTINUE INVO0240
  IF (SINT3.LT.-1.0) SINT3 = -1.0 INVO0250
  IF (SINT3.GT.1.0) SINT3 = 1.0 INVO0260
  THETA = DARSIN(SINT3)/3.0 INVO0270
  GO TO (1,2,3,4),NCRIT INVO0280
C*** TRESCA INVO0290
1 YIELD = 2.0*DCOS(THETA)*STEFF INVO0300
  RETURN INVO0310
C*** VON MISES INVO0320
2 YIELD = ROOT3*STEFF INVO0330
  RETURN INVO0340
C*** MOHR-COULOMB INVO0350
3 PHIPA = PROPS(LPROP,7)*0.017453292 INVO0360
  SNPHI = DSIN(PHIPA) INVO0370
  YIELD = SMEAN*SNPHI+STEFF*(DCOS(THETA)-DSIN(THETA)+SNPHI/ROOT3) INVO0380
  RETURN INVO0390
C*** CRICKER-PHAGER INVO0400
4 PHIPA = PROPS(LPROP,7)*0.017453292 INVO0410
  SNPHI = DSIN(PHIPA) INVO0420
  YIELD = 6.0*SMEAN*SNPHI/(ROOT3*(3.0-SNPHI))+STEFF INVO0430
  RETURN INVO0440
  END INVO0450
      INVO0470

```

```

      IEL01
      SUBROUTINE Y (AVECT,DEVIA,LPROP,MMATS,NCRIT,NSTR1,
      PRJPS,SINT3,STEFF,THETA,VARJ2)
      C*****
      C*** THIS SUBROUTINE EVALUATES THE FLOY VECTOR
      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)
      COSH = DCOS(THETA)
      COST3 = DCOS(3.0*THETA)
25  ROOT3 = 1.73205080757
      C*** CALCULATE VECTOR A1
      VECA1(1) = 1.0
      VECA1(2) = 1.0
      VECA1(3) = 0.0
      VECA1(4) = 1.0
      C*** CALCULATE VECTOR A2
      DO 10 ISTR1 = 1,NSTR1
10  VECA2(ISTR1) = DEVIA(ISTR1)/(2.0*STEFF)
      VECA2(3) = DEVIA(3)/STEFF
      C*** CALCULATE VECTOR A3
      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*** TRESCA
      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*** VON MISES
      2  CONS1 = 0.0
      CONS2 = ROOT3
      CONS3 = 0.0
      GO TO 40
      C*** MOHR-COULOMB
      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+(ANTH*TANT3)*CONS1+(TANT3-TANTH)*ROOT3)
      CONS3 = (ROOT3+SINTH+3.0*CONS1*COSTH)/(2.0*VARJ2+COST3)
      GO TO 40
      C*** DRUCKER-PRAGER
      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
      YIE00010
      YIE00020
      YIE00030
      YIE00040
      YIE00050
      YIE00060
      YIE00070
      YIE00080
      YIE00090
      YIE00100
      YIE00110
      YIE00120
      YIE00130
      YIE00140
      YIE00150
      YIE00160
      YIE00170
      YIE00180
      YIE00190
      YIE00200
      YIE00210
      YIE00220
      YIE00230
      YIE00240
      YIE00250
      YIE00260
      YIE00270
      YIE00280
      YIE00290
      YIE00300
      YIE00310
      YIE00320
      YIE00330
      YIE00340
      YIE00350
      YIE00360
      YIE00370
      YIE00380
      YIE00390
      YIE00400
      YIE00410
      YIE00420
      YIE00430
      YIE00440
      YIE00450
      YIE00460
      YIE00470
      YIE00480
      YIE00490
      YIE00500
      YIE00510
      YIE00520
      YIE00530
      YIE00540
      YIE00550
      YIE00560
      YIE00570
      YIE00580
      YIE00590
      YIE00600
      YIE00610
      YIE00620
      YIE00630
      YIE00640
      YIE00650
      YIE00660
      YIE00670
      YIE00680
      YIE00690
      YIE00700
      YIE00710
      YIE00720
      YIE00730
      YIE00740
      YIE00750
      YIE00760
      YIE00770
      YIE00780
      YIE00790
      YIE00800
      YIE00810
      YIE00820
      YIE00830
      YIE00840
      YIE00850
      YIE00860
      YIE00870
      YIE00880

```

```

SUBROUTINE MOOPSI(DMATX,LPROP,MMATS,NTYPE,PROPS)
C *****
C ***** THIS SUBROUTINE EVALUATES THE D-MATRIX *****
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 ***** D MATRIX FOR PLANE STRESS CASE
      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 ***** D MATRIX FOR PLANE STRAIN CASE
      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 ***** D MATRIX FOR AXISYMMETRIC CASE
      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

```

```

MJD00010
MJD00020
MJD00030
MJD00040
MJD00050
MJD00060
MJD00070
MJD00080
MJD00090
MJD00100
MJD00110
MJD00120
MJD00130
MJD00140
MJD00150
MJD00160
MJD00170
MJD00180
MJD00190
MJD00200
MJD00210
MJD00220
MJD00230
MJD00240
MJD00250
MJD00260
MJD00270
MJD00280
MJD00290
MJD00300
MJD00310
MJD00320
MJD00330
MJD00340
MJD00350
MJD00360
MJD00370
MJD00380
MJD00390
MJD00400
MJD00410
MJD00420
MJD00430
MJD00440
MJD00450
MJD00460
MJD00470
MJD00480
MJD00490
MJD00500
MJD00510
MJD00520
MJD00530
MJD00540

```


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),