NATIONAL CENTER FOR EARTHQUAKE ENGINEERING RESEARCH
State University of New York at Buffalo

A FINITE ELEMENT FORMULATION FOR NONLINEAR VISCOPLASTIC MATERIAL USING A Q-MODEL

by

Osei K. Gyebi and Gautam Dasgupta
Department of Civil Engineering and
Engineering Mechanics
Columbia University
New York, NY 10027-6699

Technical Report NCEER-87-0005

November 2, 1987

This research was conducted at Columbia University and was partially supported by the National Science Foundation under Grant No. ECE 86-07591.

NOTICE

A FINITE ELEMENT FORMULATION FOR NONLINEAR VISCOPLASTIC MATERIAL USING A Q-MODEL

by

Osei K. Gyebi[1] and Gautam Dasgupta[2]

November 2, 1987

Technical Report NCEER-87-0005

NCEER Contract Number 86-2033

NSF Master Contract Number ECE-86-07591

and

NSF Grant Number ECE-85-15249

1 Doctoral Student, Dept. of Civil Engineering and Engineering Mechanics, Columbia University

2 Associate Professor, Dept. of Civil Engineering and Engineering Mechanics, Columbia University

NATIONAL CENTER FOR EARTHQUAKE ENGINEERING RESEARCH
State University of New York at Buffalo
Red Jacket Quadrangle, Buffalo, NY 14261

ABSTRACT

A direct comparison of the rheological behavior of a mass supported by a complex spring and a mass supported by a spring-dashpot arrangement provides a means of establishing a relationship between material damping in the frequency domain and the frequency dependent $Q(\omega)$. For the special case where $Q(\omega)$ is constant over a given frequency range, the expansion of $Q^{-1}(\omega)$ into a Laurent Series yields a set of damping coefficients whose values are determined by minimizing the mean square error of the series over the prescribed frequency range. The resulting damping expression is used in conjunction with an elastoplastic constitutive matrix in finite element discretization to produce a viscoplastic model suitable for a direct step by step time integration. The proposed model is very convenient for use in finite element discretization for the analyses of earthquake, blast, shock, and other soil-structure interaction problems involving cyclic loading.

TABLE OF CONTENTS

Preceding page blank

LIST OF FIGURES

SECTION 1
INTRODUCTION

The complex behavior of geologic materials has in recent years generated extensive investigation by researchers to develop material models to predict the path-dependent behavior of such materials within the framework of the theory of classical plasticity. To better predict the hysteresis displayed by soils when loaded and unloaded hydrostatically, the cap model (27) was developed. This model, which is based on a modified classical plasticity theory, was further modified to include the effect of kinematic hardening (11). The bounding surface plasticity theory which utilizes two yield surfaces was proposed to simulate the behavior of sand under cyclic loading (19). While the above mentioned models predict the path-dependent behavior of material fairly well, they fail to address the dependence of material behavior on time. In many geomechanical problems, however, material behavior is also governed by rheological behavior. To account for energy dissipation which is associated with foundation media in soil-structure interaction problems, the material has been idealized to be viscoelastic solid (23,24). The effect of Q on wave attenuation and velocity dispersion in geologic materials has been investigated by researchers in recent years within the framework of viscoelasticity (2,3,4). In Ref. 28, the convolution integral relating stress to strain history in geologic media (assumed viscoelastic) has been transformed into a convergent sequence of constant coefficient differential operators of increasing order, through the use of Pade approximants, with the value of Q assumed over a prescribed frequency range. In this format, the convolution integral is reduced into a form suitable for time stepping in finite difference schemes for wave propagation problems. The above mentioned viscoelastic models, however, fail to account for the path-dependent behavior of materials. A more realistic model would be a viscoplastic model which accounts for the path dependence as well as the energy dissipation characteristics of the material. Several attempts have been made by researchers to predict the time as well as path-dependence of material in recent years (5,6,18,20).

The contribution of this research is to provide explicit representation of the viscous and energy dissipation characteristics of material suitable for time stepping in finite element discretization through the Q-model, $Q^{-1}(\omega)$ being a measure of energy dissipation per cycle in a medium during cyclic loading. A relation between damping in the frequency domain and the frequency dependent $Q^{-1}(\omega)$ is established through a direct comparison of rheological representation of a mass supported by a complex spring and a mass supported by a spring-dashpot arrangement, in the frequency domain. For the special case where $Q(\omega)$ is constant over a prescribed frequency range, the expansion of $Q^{-1}(\omega)$ into a Laurent Series yields a set of damping coefficients whose values are determined by minimizing the mean square error of the series over the prescribed frequency range. The resulting damping expression is used in conjunction with an elastoplastic constitutive matrix in finite element discretization to produce an elasto-viscoplastic model suitable for step by step time integration. It should be mentioned that the proposed model follows quite a different approach from those proposed in Refs. 5 and 6. Also, earlier finite element models for elastoviscoplastic materials have focused on the quasi-static behavior of the materials (25,26). The merits of the proposed model lie not only in the ease of its application in dynamic problems but also in the readiness in which the parameters involved could be determined from dynamic tests in the laboratory. The proposed model will find very useful application in seismic problems such as design of dams, bridges, buildings, subjected to earthquake excitation. The model could be adapted to suit viscoelastic problems simply by replacing the elastoplastic constitutive matrix with an elastic constitutive matrix. This makes the model suitable for a much wider variety of problems.

SECTION 2
SURVEY OF SOIL MODELS

2.1 Simple Plasticity Models

The elastic ideally-plastic soil model with a fixed yield surface defined by

$$F(\sigma_{ij}) = 0 \tag{2-1}$$

characterizes the earlier plasticity soil models. Here σ_{ij} defines a stress point in the stress space. In terms of stress invariants, the yield surface is given by

$$F(J_1, J_2, J_3) = 0 \tag{2-2}$$

where J_1, J_2, J_3 are the first, second, and third stress invariants, respectively.

The material is elastic when the stress point lies inside the yield surface, in which case changes in stresses result in recoverable deformations. The increment in elastic strain is given by

$$d\varepsilon_{ij}^{e} = \frac{1}{2G} ds_{ij} + \frac{1}{9K} \delta_{ij} d\sigma_{kk} \tag{2-3}$$

where:

ds_{ij} = Increment in deviatoric stress
$d\sigma_{kk}$ = Increment in volumetric stress
δ_{ij} = Kronecker delta
K = Bulk modulus
G = Shear modulus

On the yield surface, total strain increment is given by

$$d\varepsilon_{ij} = d\varepsilon_{ij}^{e} + d\varepsilon_{ij}^{p} \tag{2-4}$$

where $d\varepsilon_{ij}^{e}$ is the elastic strain increment defined by equation (2-3) and

$$d\varepsilon_{ij}^{p} = \lambda \frac{\partial q}{\partial \sigma_{ij}} \tag{2-5}$$

where:

q = plastic potential
λ = nonnegative scalar function
$d\varepsilon_{ij}^{P}$ = plastic strain rate

If F = q then equation (2-5) becomes

$$d\varepsilon_{ij}^{p} = \begin{cases} \lambda \dfrac{\partial F}{\partial \sigma_{ij}} & F=0 \\ 0 & F<0 \end{cases} \tag{2-6}$$

and the flow rule is said to be associated, and hence the plastic strain rate is normal to the yield surface at the current stress point. Furthermore, for a convex yield surface, uniqueness is assured. In general, however, when the yield surface does not coincide with the plastic potential, i.e., $F \neq q$, the corresponding flow rule is said to be non-associated. In this case, uniqueness cannot, in general, be proved.

Stress outside the yield surface, i.e., $F > 0$ is not permitted.

Uniqueness and Stability

Uniqueness, stability, and continuity are basic requirements for continuum models. From Drucker's stability postulate (Ref. 7) nonnegative work must be done by an external agent in any excursion from equilibrium. In particular for any stress cycle, where σ_{ij}^{*} is the stress at equilibrium state

$$\int \left(\sigma_{ij} - \sigma^*_{ij}\right) d\varepsilon_{ij} \geq 0 \tag{2-7}$$

The equal sign applies only for elastic or reversible paths. Satisfying Drucker's postulate is sufficient (but not necessary) to insure uniqueness and continuity.

By eliminating the elastic or reversible strains and by choosing σ^*_{ij} (equation (2-7) on the yield surface, the condition for stability in the "small" for elastic-plastic models is obtained, i.e.,

$$d\sigma_{ij}\, d\varepsilon^p_{ij} \geq 0 \tag{2-8}$$

This means that the yield condition can only move outward (or not move) at a stress point, i.e., work softening or strain softening is not permitted.

The condition for stability in the "large" for elastoplastic materials is given by

$$\left(\sigma_{ij} - \sigma^*_{ij}\right) d\varepsilon^p_{ij} \geq 0 \tag{2-9}$$

It can be noted that equation (2-9) requires the normality of the plastic strain rate vector, and the convexity of the yield surface. (See Figure 2-1.)

2.2 von Mises

The von Mises yield condition is given by

$$\sqrt{J'_2} = k \tag{2-10}$$

where J'_2 is the second invariant of the deviatoric stress and is given by

$$J'_2 = \frac{1}{2}\, s_{ij}\, s_{ij} \tag{2-11}$$

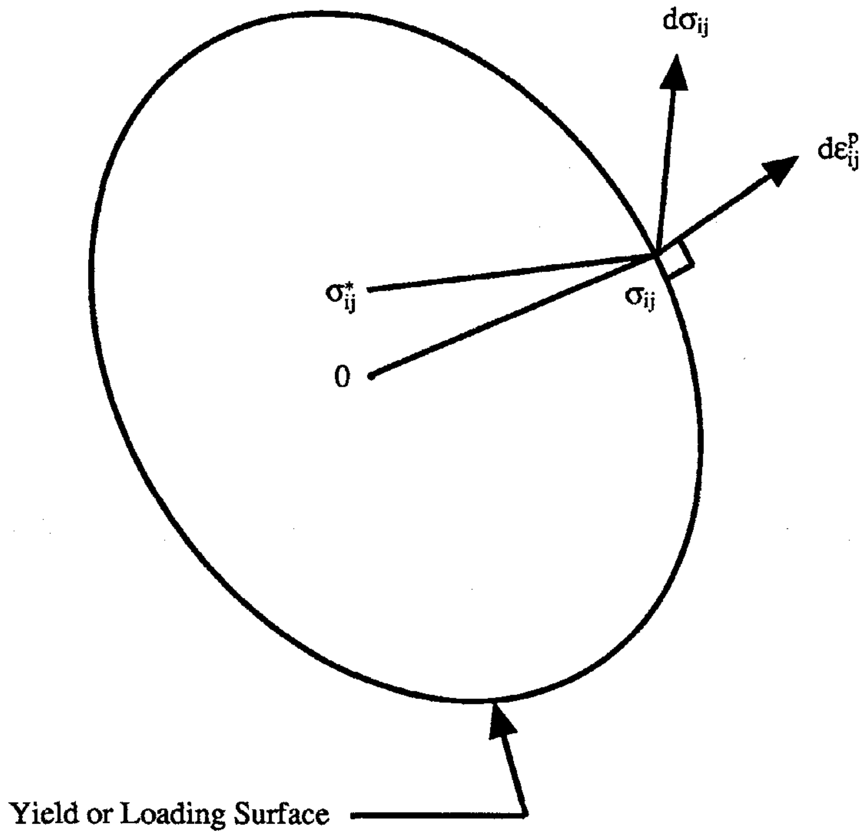


FIGURE 2-1 Geometric Interpretation of Drucker's Stability Postulate

and k is a constant.

The von Mises yield surface is a cylinder in the principal stress space (see Figure 2-2), and is a good representation of the failure surface of many saturated clays.

2.3 Drucker and Prager

Drucker and Prager, Ref. (10), proposed an ideally plastic yield condition for granular materials given by

$$F = \sqrt{J_2'} + \alpha J_1 - k = 0 \tag{2-12}$$

This is a modified Mohr-Coulomb failure criterion which reduces to the Mohr-Coulomb equations in a triaxial test when

$$\alpha = \frac{2 \sin \phi}{\sqrt{3}\ (3 - \sin \phi)} \tag{2-13}$$

and

$$k = \frac{6c \cos \phi}{\sqrt{3}\ (3 - \sin \phi)} \tag{2-14}$$

In equation (2-12), J_1 and J_2' are, respectively, the first and second invariants of stress and stress deviator. α and k are defined by equations (2-13) and (2-14) respectively, ϕ is the friction angle and c is cohesion of soil. (See Figures 2-3 and 2-4.) This model satisfies Drucker's postulates for stability and uniqueness when used with the associated flow rule as defined earlier. However, the model has the following shortcomings:

1. Due to the normality principle and the concept of the associated flow rule, considerable dilatancy effects are predicted which are much greater than observed experimentally;

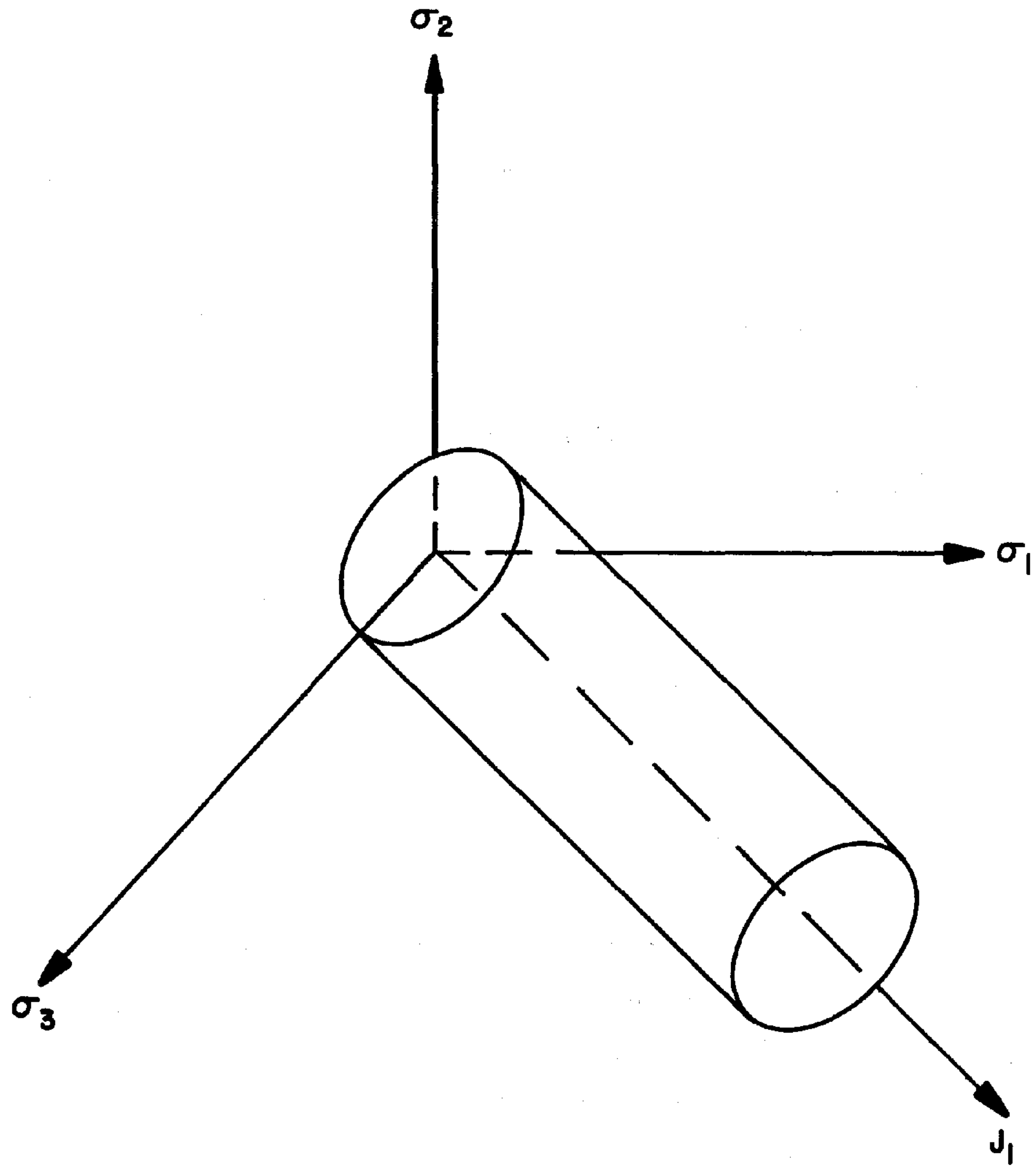


FIGURE 2-2 Von Mises Yield Surface Cylinder

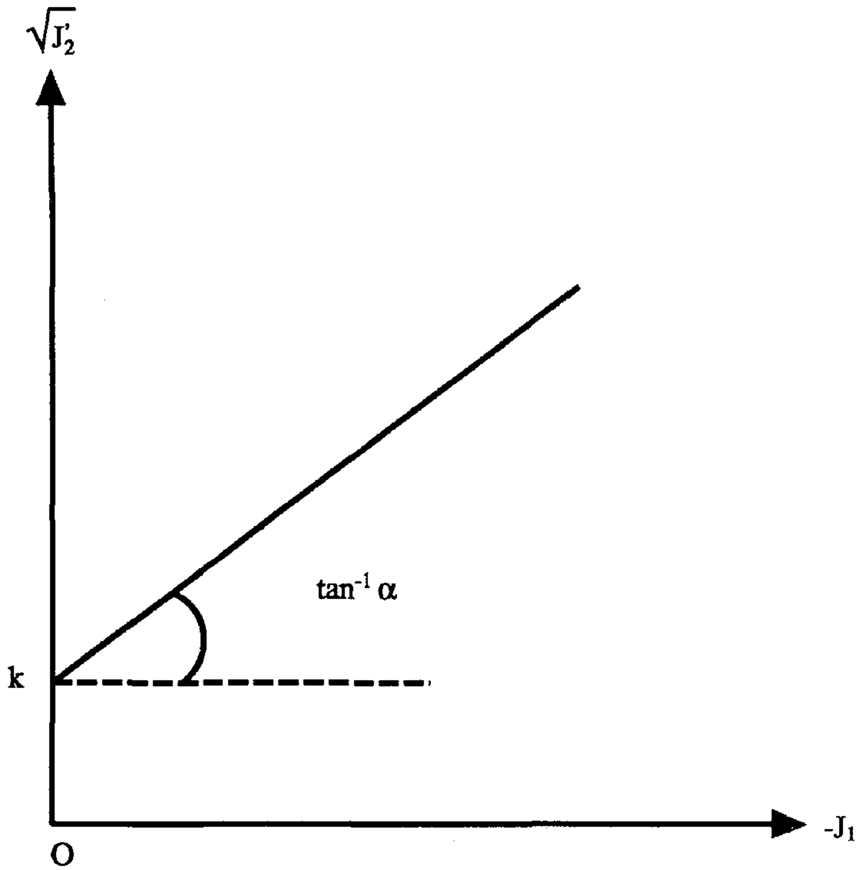


FIGURE 2-3 Drucker - Prager Yield Surface

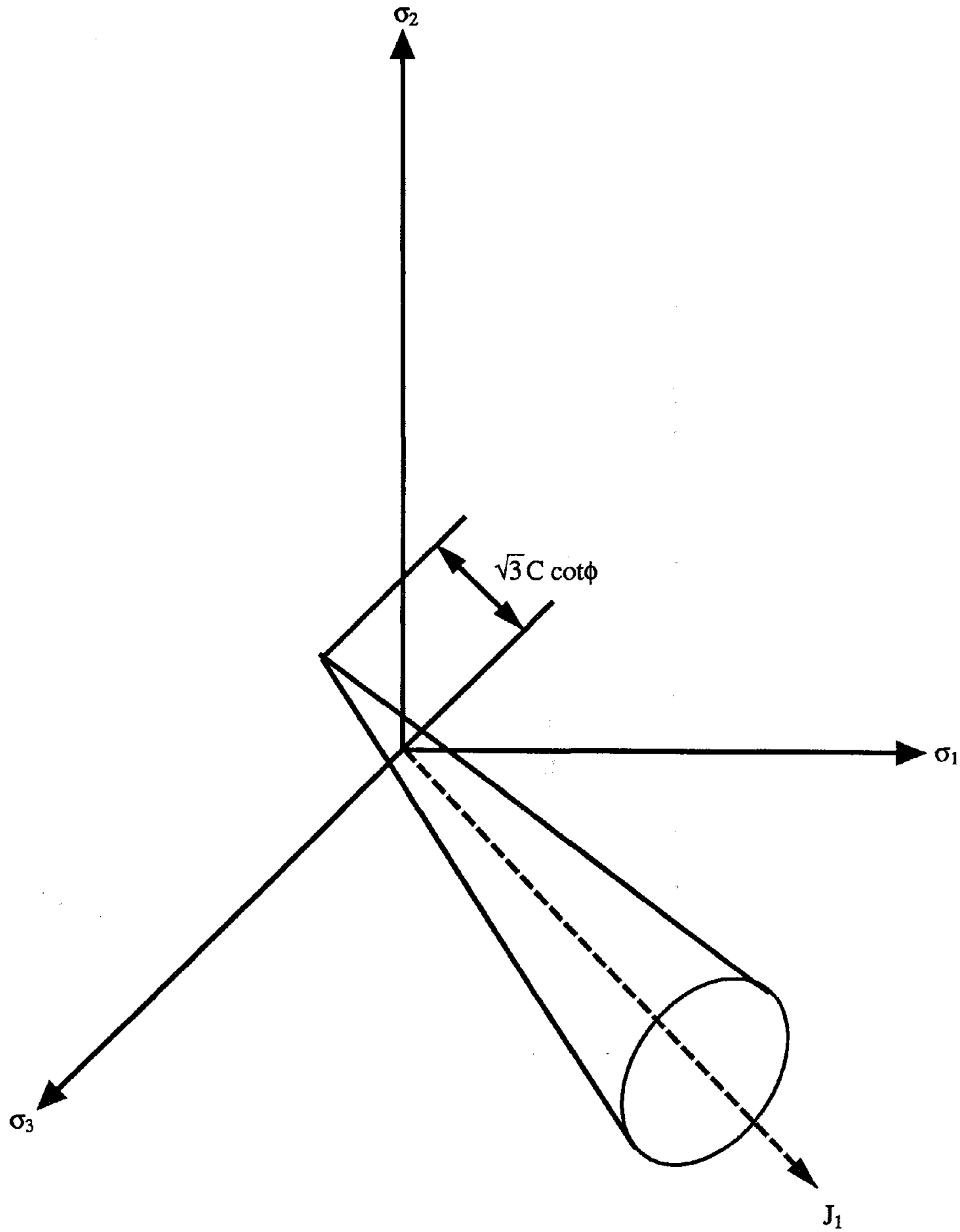


FIGURE 2-4 Drucker - Prager Yield Surface (Cone)

2. Experimental observations have shown that considerable hysteresis in a hydrostatic load-unloading path occurs which cannot be predicted using the same elastic bulk modulus of loading and unloading and a yield surface which does not cross the hydrostatic (J_1) axis; and

3. Soils pass through the fluid state at high pressures where shear strength does not vary with hydrostatic pressure. Therefore, the yield condition should essentially be independent of J_1 for large J_1.

2.4 The Development of the Cap Model

To control the plastic volumetric change or dilatation of soils, Drucker, Gibson and Henkel (Ref. 8) added a movable cap to the Drucker-Prager model, which crosses the hydrostatic loading axis. This modification reproduces better the hysteresis which soils display when loaded and unloaded hydrostatically.

Several strain hardening plasticity models based on critical-state concepts have since then been developed by a group at Cambridge University.

A group of researchers from MIT also worked on a similar model where the yield curves are ellipses of constant eccentricity. However, Drucker-Prager's postulate of stability in the small is not fully satisfied at all points on the yield surface and, therefore, the irreversibility condition

$$\sigma_{ij}\, d\varepsilon_{ij}^{p} \geq 0 \tag{2-15}$$

is not satisfied.

DiMaggio and Sandler (Ref. 27) have proposed a new cap model for granular soil which satisfies continuity, stability, and uniqueness conditions.

The new cap model has an ideally plastic modified Drucker-Prager yield condition denoted by

$$f_1\left(J_1\,,\sqrt{J_2'}\right) = 0 \tag{2-16}$$

and a strain-hardening cap, which expands or contracts as the plastic volumetric strain decreases or increases, respectively, denoted by

$$f_2\left(J_1, \sqrt{J_2'}, \varepsilon_v^p\right) = 0 \tag{2-17}$$

A full discussion of the above model and its application to McCormick Ranch sand is covered in a different section.

Modifications to the cap model to include kinematic hardening has been made for materials whose hysteresis is independent of strain rate (Ref. 11). This modification is achieved by replacing the stress tensor σ_{ij} by the quantity (σ_{ij} - α_{ij}), where α_{ij} is the tensor whose components are memory parameters defining the translation of the yield surface in stress space. The kinematic hardening is assumed to occur in shear only. Hence

$$\alpha_{ii} = 0 \tag{2-18}$$

where the summation convention of the subscripts apply. The memory parameter α_{ij} is governed by a kinematic hardening rule given by

$$\alpha_{ij} = C_\alpha\, \dot{e}_{ij}^p \tag{2-19}$$

where $\dot{e}_{ij}^p$ are the deviatoric components of plastic strain, and C_α is a constant. Extensive coverage of this subject has been made in Ref. 11.

2.5 Bounding Surface Plasticity Theory

The bounding surface (Ref. 19) is represented in the 2-D octahedral, shear-vs-normal stress space by a half-ellipse with a variable axis length (Figure 2-5). The ellipse's horizontal axis coincides with the normal stress axis and has one of its end points fixed to the origin of coordinate system,

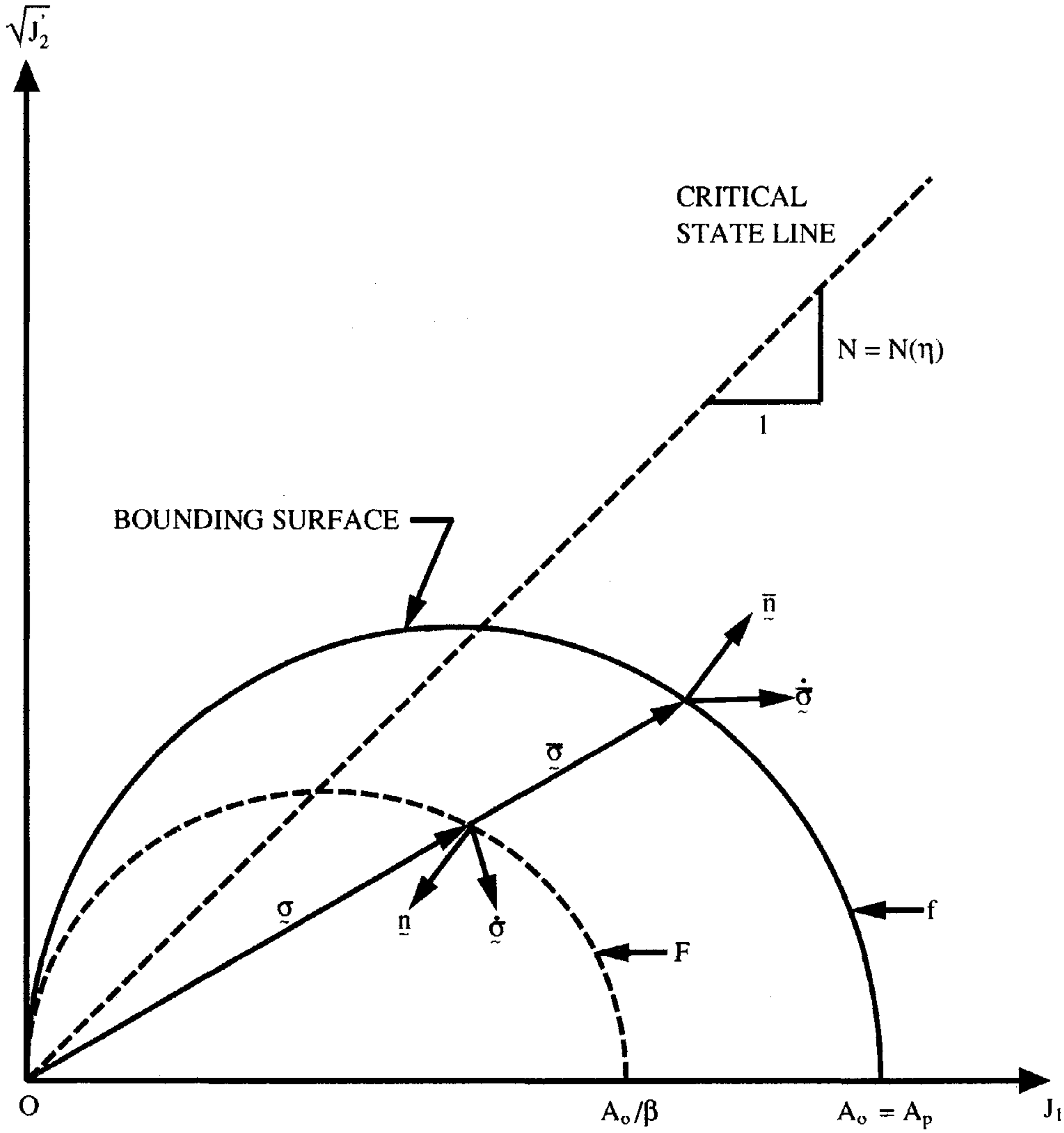


FIGURE 2-5 Bounding Surface

and the other end point, as well as the ratio of the axis lengths, is determined by the hardening law. The hardening law is obtained along the standard triaxial stress path. An ellipse is formulated such that:

1. It contains the stress point representing the state of stress in the triaxial test sample at all times; and

2. The "vector" normal at this stress point is proportional to the plastic strain rate "vector" measured in the triaxial test. The major axis and the ratio between the axes are then expressed as functions of the stress level and accumulated plastic strain. Under this new hardening law, the bounding surface, for any chosen loading path, always expands.

Elasto-Plastic Constitutive Equations

$$\dot{\underset{\sim}{\varepsilon}}^{p} = \frac{\underset{\sim}{n}:\dot{\underset{\sim}{\sigma}}}{k_p}\,\underset{\sim}{n} \qquad (2\text{-}20)$$

where:

$< >$ = Operation $< L > = LH(L)$
H = Heaviside's step function
$\dot{\underset{\sim}{\varepsilon}}^{p}$ = Plastic strain rate tensor
k_p = Generalized plastic modulus
$\underset{\sim}{n}$ = Second order tensor such that $\underset{\sim}{n}:\underset{\sim}{n} = 1$

$$\dot{\underset{\sim}{\varepsilon}} = \underset{\sim}{D}^{-1} : \dot{\underset{\sim}{\sigma}} - < \frac{\underset{\sim}{n}:\dot{\underset{\sim}{\sigma}}}{k_p} > \underset{\sim}{n} \qquad (2\text{-}21)$$

or inverted

$$\dot{\underset{\sim}{\sigma}} = \underset{\sim}{D}:\dot{\underset{\sim}{\varepsilon}} - <\frac{\underset{\sim}{n}:\underset{\sim}{D}:\dot{\underset{\sim}{\varepsilon}}}{k_p + \underset{\sim}{n}:\underset{\sim}{D}:\underset{\sim}{n}}> \underset{\sim}{D}:\underset{\sim}{n} \qquad (2\text{-}22)$$

where $\dot{\underset{\sim}{\varepsilon}}$ is the total strain rate tensor.

Radial Mapping

The radial projection of a stress point representing $\underset{\sim}{\sigma}$ produces $\bar{\underset{\sim}{\sigma}}$ on the bounding surface, where

$$\bar{\underset{\sim}{\sigma}} = \beta\underset{\sim}{\sigma} \qquad (2\text{-}23)$$

with β, the radial mapping scalar, being unity when $\underset{\sim}{\sigma}$ lies on the bounding surface itself.

Loading Condition

Loading path: $\underset{\sim}{n} = \bar{\underset{\sim}{n}}$; $k_p = \bar{k}_p$ with $\beta = 1$ (2-24)

unloading path: $n = -\bar{n}$; $k_p = H_u/(\beta-1)$ (2-25)

with $\dot{\underset{\sim}{\sigma}}$ pointing inwards from F

reloading path: $\underset{\sim}{n} = \bar{\underset{\sim}{n}}$: $k_p = H_R(\beta-1)$ (2-26)

with $\dot{\underset{\sim}{\sigma}}$ pointing outwards from F where H_u and H_R are material parameters

$$\bar{\underset{\sim}{n}} = \frac{\underset{\sim}{\nabla} f}{||\underset{\sim}{\nabla} f||} \qquad (2\text{-}27)$$

where the operator $\underset{\sim}{\nabla}$ is defined in the stress space and $\underset{\sim}{\nabla}$f is evaluated at the stress state $\bar{\underset{\sim}{\sigma}}$.

$$\bar{k}_p = -\frac{1}{||\nabla f||}\frac{\partial f}{\partial \eta}\sqrt{\frac{1}{2} - \frac{1}{6}\,[\mathrm{tr}\,(\underset{\sim}{\eta})]^2} \qquad (2\text{-}28)$$

where η is the equivalent shear strain defined by the hardening law

$$\eta = \int \dot{\eta} = \int \frac{1}{2} = \dot{\underset{\sim}{e}}^p \dot{\underset{\sim}{e}}^p \qquad (2\text{-}29)$$

and $\dot{\underset{\sim}{e}}^p$ is the deviator of $\dot{\underset{\sim}{\varepsilon}}^p$. The link between the generalized plastic moduli k_p and $\bar{k}_p$ is given by

$$\frac{\underset{\sim}{n}:\dot{\underset{\sim}{\sigma}}}{k_p} = \frac{\bar{\underset{\sim}{n}}:\dot{\bar{\underset{\sim}{\sigma}}}}{\bar{k}_p} \geq 0 \qquad (2\text{-}30)$$

Bounding Surface

The bounding surface f is defined as

$$f = \frac{\bar{J}_2'}{\bar{J}_1 N^2} + \bar{J}_1 - A_o = 0 \qquad (2\text{-}31)$$

where:

$$\bar{J}_2' = \beta^2 J_2' \qquad (2\text{-}32)$$

and

$$\bar{J}_1 = \beta J_1 \qquad (2\text{-}33)$$

The strain dependency of the hardening surface f is defined by the slope $N = N(\eta)$, of the critical state line, and the axis, $A_o = pA$ of the half-ellipse; where $A = A(\eta)$ and p is the atmospheric pressure.

The bounding surface parameters A and N are related to the shear dilatation angle α by

$$\tan\alpha = \frac{\bar{J}_1 \sqrt{\bar{J}_2'}}{3\left(N^2 \bar{J}_1^2 - \bar{J}_2'\right)} \tag{2-34}$$

$$= \frac{\sqrt{\frac{1}{2}\, \bar{\underset{\sim}{\eta}}^d : \bar{\underset{\sim}{\eta}}^d}}{\mathrm{tr}(\bar{\underset{\sim}{\eta}})}$$

where $\bar{\underset{\sim}{\eta}}^d = \bar{\underset{\sim}{\eta}} - \frac{1}{3}\,\mathrm{tr}(\bar{\underset{\sim}{\eta}})\underset{\sim}{\delta}$. The main feature of this 8-parameter model is its hardening law which is defined along the standard triaxial test path to emphasize the shear dilatation of sands. The model's ability to simulate important characteristics of sand under cyclic loading compares well with laboratory tests.

2.6 The Endochronic Theory

The Endochronic Theory is based on the hypothesis that the current state of stress is a linear function of the entire history of plastic strain, with the history defined with respect to a time scale (intrinsic time) which is itself a property of the material at hand.

The equations governing the Endochronic Theory are as follows:

$$\begin{aligned} \sigma_{ij} &= H_D \int_0^{Z_D} \rho(Z_D - Z')\, \frac{\partial\theta_{ij}}{\partial Z'}\, dZ' \\ &\quad + \delta_{ij} H_H \int_0^{Z_H} \phi(Z_H - Z')\, \frac{\partial\theta}{\partial Z'} \end{aligned} \tag{2-35}$$

$$\begin{aligned} d\sigma_{ij} &= [K_o\,(d\varepsilon - d\theta) - 2/3\; G_o\, d\varepsilon]\;\delta_{ij} \\ &\quad + 2G_o\,(d\varepsilon_{ij} - d\theta_{ij}) \end{aligned} \tag{2-36}$$

where the intrinsic time scales, Z_D and Z_H, are positive, monotonically increasing quantities defined by the ff expressions:

$$dZ_D^2 = k_{oo}\, d\zeta_D^2 + k_{ol}\, d\zeta_H^2 \tag{2-37}$$

$$dZ_H^2 = k_{lo}\, d\zeta_D^2 + k_{ll}\, d\zeta_H^2 \tag{2-38}$$

where:

H_D = Hardening function

H_H = Softening function

$d\theta_{ij}$ and $d\theta$ are defined such that

$$d\varepsilon_{ij}^p = d\theta_{ij} + \frac{1}{3}\, d\theta\delta_{ij} \tag{2-39}$$

or

$$d\theta = d\varepsilon_{kk}^p \tag{2-40}$$

Also K_o G_o denote the bulk and elastic shear moduli, respectively. $\rho(z)$ and $\phi(z)$ are material functions. The intrinsic time measures $d\zeta_D$ and $d\zeta_H$ are defined as follows.

$$d\zeta_D^2 = d\theta_{ij}\, d\theta_{ij} \tag{2-41}$$

$$d\zeta_H^2 = |\, d\theta\,|^2 \tag{2-42}$$

and the k_{rs} are elements of the material-dependent coupling matrix [k]. It is evident that the materials described by the above equations are plastic strain history dependent but strain rate independent (i.e., are independent of the natural time scale given by a clock). The following kernel functions may be used to represent soils using the Endochronic Theory

$$\rho(z) = \frac{e^{-kz}}{\sqrt{z}} \tag{2-43}$$

$$\phi(z) = \frac{\phi_o}{\sqrt{z}} \tag{2-44}$$

$$H_D(z) = H_o + (H_\infty - H_o)(1 - e^{-\gamma z}) \tag{2-45}$$

$$H_H(\theta) = (\theta_m - \theta)^{-\beta} \tag{2-46}$$

where k, H_0, H_{∞}, γ, θ_m and β are constants. For numerical purposes, the incremental form of equation (2-35) may be used as follows

$$\begin{aligned} d\sigma_{ij} &= \left[H_D \int_0^{Z_D} \rho(Z_D - Z') \frac{\partial^2 \theta_{ij}}{\partial z'^2} dZ' \right] dZ_D \\ &\quad + \delta_{ij} \left[H_H \int_0^{Z_H} \phi(Z_H - Z') \frac{\partial^2 \theta}{\partial Z'^2} dZ' \right] dZ_H \end{aligned} \tag{2-47}$$

2.7 The Hyperbolic Stress-Strain Law

Considered as a hybrid of plasticity theory and endochronic theory, the hyperbolic stress-strain law was proposed by Kondner and his co-workers to model nonlinear stress-strain relations in soil (Ref. 22). The nonlinear stress-strain curve is represented by a hyperbola of the form

$$(\sigma_1 - \sigma_3) = \frac{\varepsilon_a}{a + b\varepsilon_a} \tag{2-48}$$

in which $(\sigma_1 - \sigma_3)$ is the principal stress difference, ε_a is the axial strain, and a and b are parameters whose values are determined experimentally. As shown in Figure 2-6, these parameters are the reciprocals of the initial slope (initial tangent modulus) and the asymptote to the σ - ε curve. To determine the parameters a and b, equation (2-48) may be transformed into the form

$$\frac{\varepsilon_a}{(\sigma_1 - \sigma_3)} = a + b\varepsilon_a \tag{2-49}$$

As shown in Figure 2-7, the parameters a and b are, respectively, the intercept and the slope of the straight line. The ratio

$$\frac{(\sigma_1 - \sigma_3) \text{ failure}}{(\sigma_1 - \sigma_3) \text{ ultimate}} = R_f \tag{2-50}$$

where $R_f < 1$, and is a correlation factor called "failure ratio." Values of R_f for a variety of different soils have been found to range from 0.5 to 1.0 and to be essentially independent of confining pressure.

2.8 Earlier Viscoplastic Models

Viscoplasticity is a term which has been used by researchers to describe material behavior whereby all plastic strains in the material are developed with <u>time</u>; that is, there is a delayed plasticity, in contrast with elasto-plastic strains, which are produced instantaneously, i.e. are independent of time. This section outlines some of the earlier viscoplastic models used for time-dependent materials.

Rheology

Consider the rheological model of an elasto-viscoplastic material shown in Figure 2-8. It consists of a spring which is in series with a slider and dashpot system in parallel. The spring gives the elastic response while the dashpot and slider allow viscous deformation only for stresses greater than a certain limit.

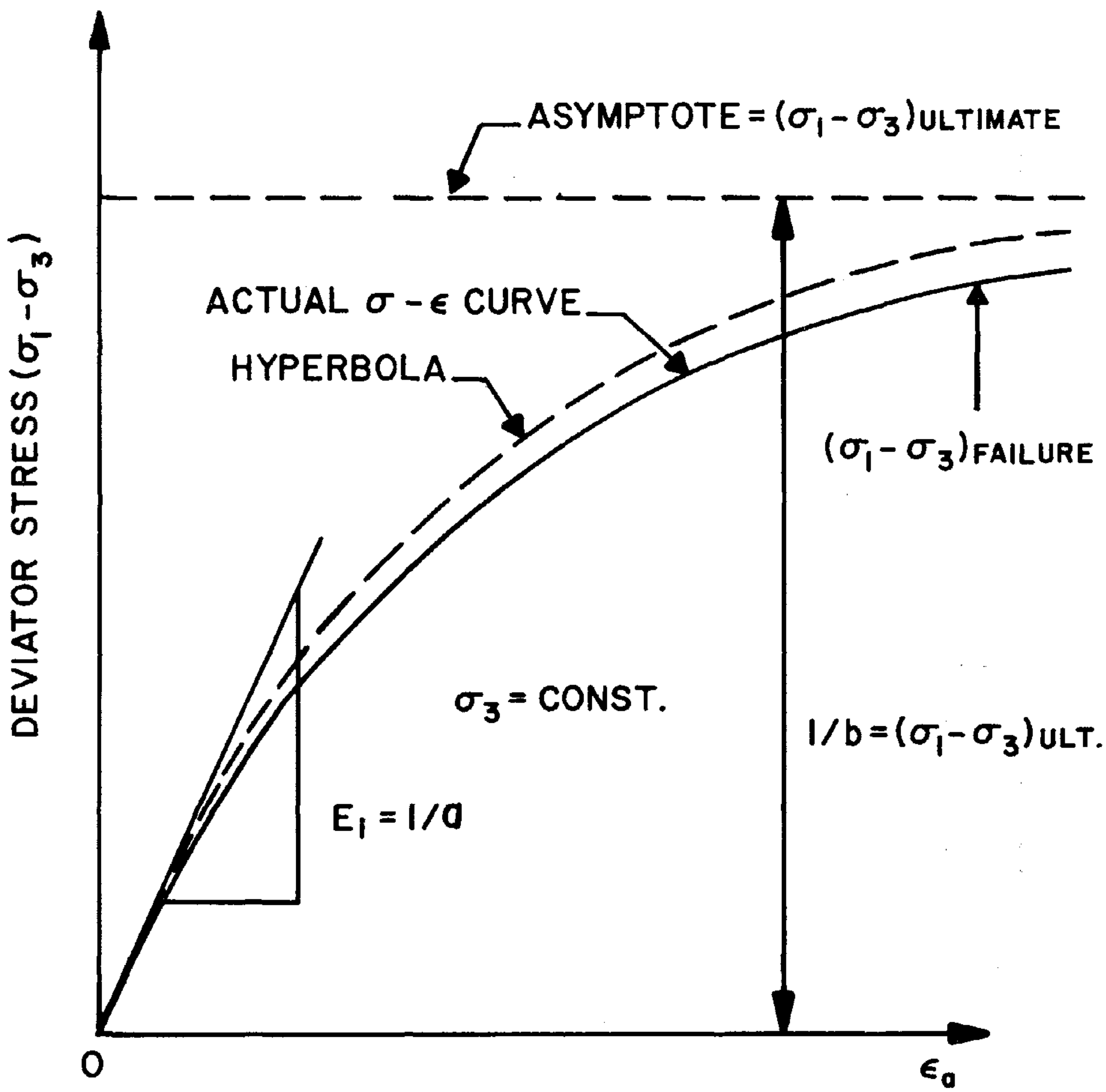


FIGURE 2-6 Hyperbolic Representation Of Stress-Strain Curve

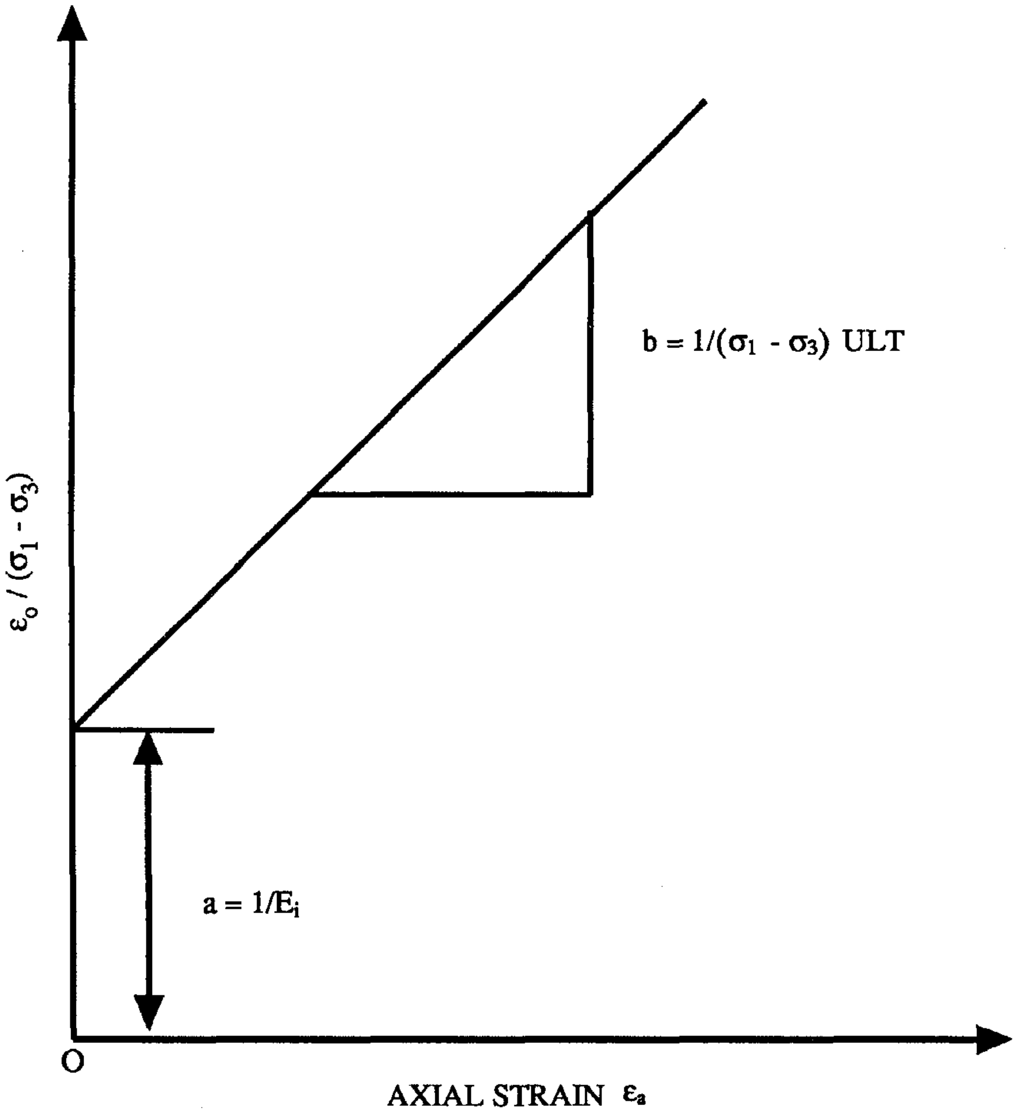


FIGURE 2-7 Transformed Hyperbolic Representation of Stress-Strain Curve

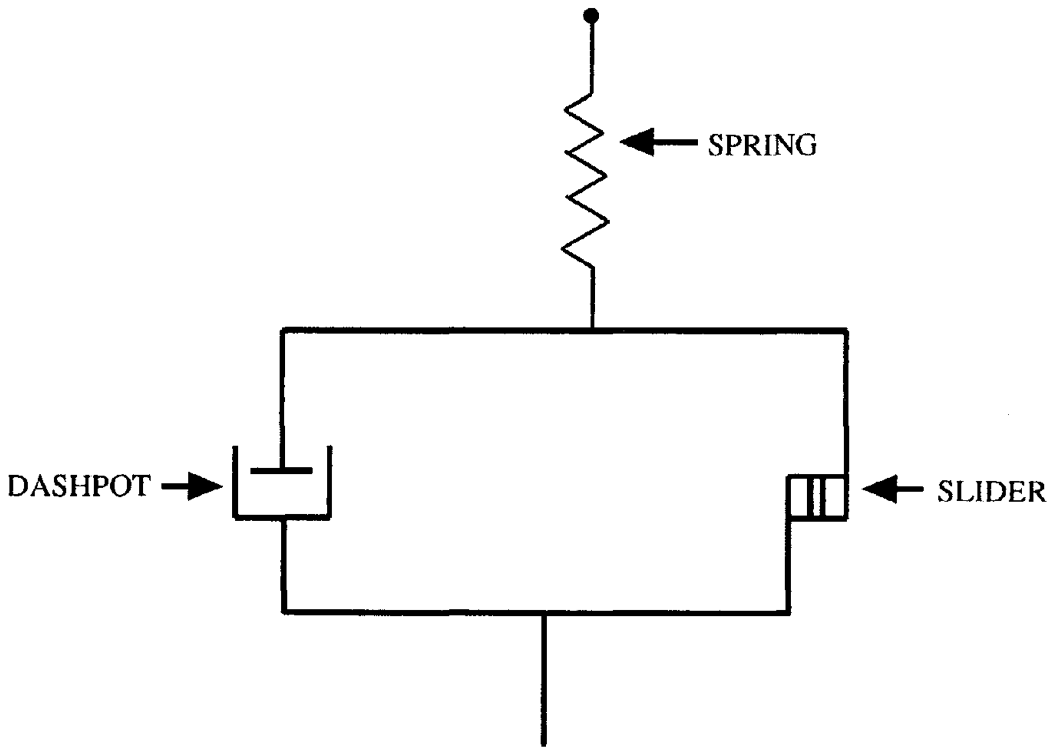


FIGURE 2-8 Rheological Model Of Elasto-Viscoplasticity

Figure 2-9 shows a stress-strain curve in 1-D to illustrate the behavior of an elasto-viscoplastic model. If $\sigma \leq \sigma_y$ where σ_y is the yield stress of the material, no viscoplastic strains are developed. However, if $\sigma > \sigma_y$ viscoplastic strains are developed at a finite rate which depends on the excess of $\sigma - \sigma_y$. A steady state is reached at time t = T when the stress is on the yield surface and no further increase in viscoplastic strains occur. A full discussion on the development of viscoplastic strains is covered in later sections in this report.

2.8.1. The Rigid-Viscoplastic Models

In rigid-viscoplastic models, materials show rigid behavior (no deformation) when applied stress is below a certain limit and show viscous response when the limit is exceeded. These are referred to as Bingham materials.

For such materials, the constitutive equation can be expressed in the form

$$s_{ij} = \frac{\dot{e}_{ij}}{2\lambda} + 2\,\eta\dot{e}_{ij} \tag{2-51}$$

where s_{ij} is the deviatoric stress given by

$$s_{ij} = \sigma_{ij} - 1/3\sigma_{kk}\,\delta_{ij} \tag{2-52}$$

$\dot{e}_{ij}$ is the deviatoric strain rate given by

$$\dot{e}_{ij} = \dot{\varepsilon}_{ij} - 1/3\dot{\varepsilon}_{kk}\,\delta_{ij} \tag{2-53}$$

where:

η = Viscosity coefficient
λ = Scaler multiplier

Consider the von Mises yield condition

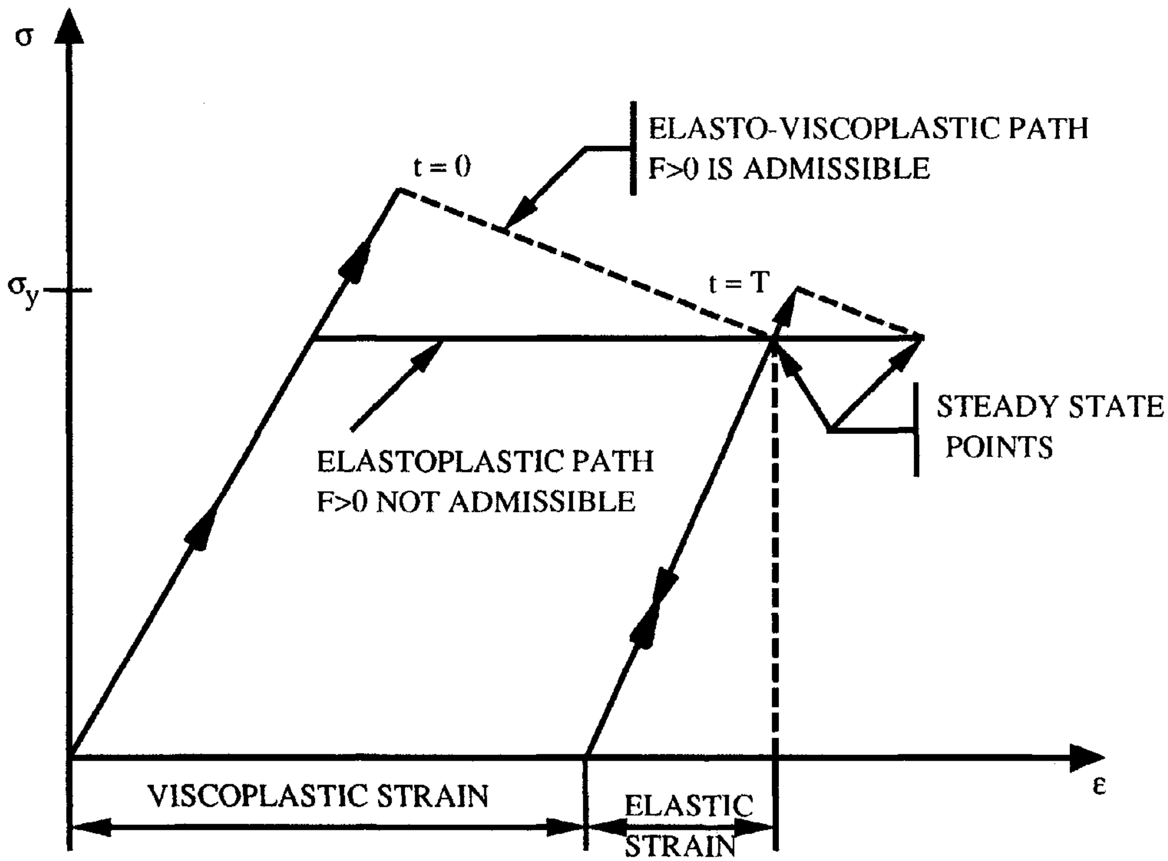


FIGURE 2-9 Typical Stress-Strain Curve For Elasto-Viscoplastic Model

$$J_2' = k^2 \tag{2-54}$$

where J_2' is the second invariant of deviatoric stress given by

$$J_2' = \frac{1}{2} s_{ij} s_{ij} \tag{2-55}$$

and k is the yield stress in pure shear, and rewrite equation (2-51) in the form

$$s_{ij} = \left(\frac{1}{2\lambda} + 2\eta \right) \dot{e}_{ij} = 2\eta' \dot{e}_{ij} \tag{2-56}$$

where η' is a variable viscosity coefficient.

From equation (2-54) we can obtain

$$2\eta' = 2\eta \left(1 + \frac{k}{2\eta\sqrt{I_e'}} \right) = \frac{\sqrt{J_2'}}{\sqrt{I_e'}} = \frac{2\eta}{1\text{-}k/\sqrt{J_2'}} \tag{2-57}$$

where:

$$I_e' = \frac{1}{2} \dot{e}_{ij} \dot{e}_{ij}$$

Generally $\eta' > \eta$ and $\eta' = \eta$ only when $\sqrt{J_2'} \to \infty$. From equation (2-57) we can rewrite equation (2-51) in the form

$$s_{ij} = 2\eta \left(1 - \frac{k}{\sqrt{2'}} \right)^{-1} \dot{e}_{ij} \text{ if } J_2' > k^2 \tag{2-58}$$

For the rate of work to be positive during plastic deformation

$$s_{ij}\dot{e}_{ij} = 2k\sqrt{I_e'} + 4\eta I_e' > 0 \tag{2-59}$$

This, however, becomes zero as $J_2' \to k^2$.

The volume remains incompressible during deformation, i.e., $\dot{\varepsilon}_{ii} = 0$. For $J_2' \leq k^2$ the body is rigid.

2.8.2 Elasto-Viscoplastic Models

In elasto-viscoplastic models, the material behavior is elastic when applied stress is below a certain limit, and shows viscous response under the action of higher stress.

The constitutive equation for such a material behavior is composed of elastic and viscoplastic components.

We have

$$\dot{e}_{ij} = \dot{e}_{ij}^{E} + \dot{e}_{ij}^{VP} \tag{2-60}$$

or

$$\dot{e}_{ij} = \frac{\dot{s}_{ij}}{2G} + \frac{2\lambda}{1+4\lambda\eta}\, s_{ij} \tag{2-61}$$

or from equation (2-58)

$$\dot{e}_{ij} = \frac{\dot{s}_{ij}}{2G} + \frac{1}{2\eta}\left(1 - \frac{k}{\sqrt{J_2'}}\right) s_{ij} \text{ if } J_2' > k^2 \tag{2-62}$$

and $\sigma_{kk} = 3K\varepsilon_{kk}$ (elastic volumetric behavior). In the above equations, $\dot{e}_{ij}, \dot{e}_{ij}^{E}, \dot{e}_{ij}^{VP}$ are respectively the total, elastic, and viscoplastic strain rates, and G is the shear modulus.

Equation (2-62) can be written more generally as

$$\dot{e}_{ij} = \frac{\dot{s}_{ij}}{2G} + \frac{1}{2\eta} < \left(1 - \frac{k}{\sqrt{J'_2}}\right) > s_{ij} \tag{2-63}$$

where the notation < > is defined such that

$$<F> = 0 \text{ if } F \leq 0$$
$$<F> = F \text{ if } F > 0 \tag{2-64}$$

where F is an arbitary function. The above model can be viewed as a modification of the Prandtl-Reuss model to include the viscous effect of materials.

Loading Condition

The loading condition may be defined locally using the rate of working. Now

$$\dot{W} = \dot{W}^{E} + \dot{W}^{VP} \tag{2-65}$$

where $\dot{W}, \dot{W}^{E}, \dot{W}^{VP}$ denote, respectively, the total rate of working, the rate of work due to elastic and viscoplastic deformations. Hence

$$s_{ij}\dot{e}_{ij} = s_{ij}\dot{e}_{ij}^{E} + s_{ij}\dot{e}^{VP} \tag{2-66}$$

or

$$s_{ij}\dot{e}_{ij} = \frac{1}{2G}\left(\dot{J}'_2 + 2\frac{G}{\eta}\left(1 - \frac{k}{\sqrt{J'_2}}\right)J'_2\right) \tag{2-67}$$

Notice that $\dot{W}^{VP} > 0$ if $J_2' > k^2$. If

$$\dot{J}_2' > k^2 \text{ and } s_{ij}\dot{e}_{ij} > 0 \qquad (2\text{-}68)$$

then we have loading. There are three types of loading:

1. If in addition to equation (2-68)

$$\dot{J}_2' > 0, \text{ and therefore } \dot{W}^{VP} > 0, \dot{W}^{E} > 0 \qquad (2\text{-}69)$$

the process is called total loading.

2. If in addition to equation (2-68)

$$\dot{J}_2' = 0, \text{ and therefore } \dot{W}^{VP} > 0, \dot{W}^{E} = 0 \qquad (2\text{-}70)$$

then we have neutral loading.

3. If in addition to equation (2-68)

$$\dot{J}_2' < 0 \text{ but } \dot{W} > 0 \qquad (2\text{-}71)$$

we have partial loading. If

$$\begin{aligned} &J_2' > k^2 \text{ and } s_{ij}\dot{e}_{ij} = 0 \\ &\text{i.e. when } -\dot{W}^{E} = \dot{W}^{VP} > 0 \end{aligned} \qquad (2\text{-}72)$$

then we have a state of pure relaxation. If

$$J_2' > k^2 \text{ and } \dot{s}_{ij}\dot{e}_{ij} < 0 \tag{2-73}$$

the process is called <u>quasi-unloading</u> because the decrease of the stress is more rapid than in a relaxation process, and therefore corresponds to stress decrease at the boundary of the body, while still $\dot{W}^{VP} > 0$. A state of pure unloading (instantaneous) corresponds to $\dot{J}_2' = \infty$.

During stress relaxation at constant strain, equation (2-67) becomes

$$\dot{J}_2' + \frac{2G}{\eta}\left(1 - \frac{k}{\sqrt{J_2'}}\right)J_2' = 0 \tag{2-74}$$

which has a solution of the form

$$\sqrt{J_2'}\ (t) = k + \left(\sqrt{J_2'^{o}} - k\right) \exp\left[-\frac{G}{\eta}\ (t-t_o)\right] \tag{2-75}$$

where:

$$\sqrt{J_2'^{o}} = \sqrt{J_2'}\ (t_o)$$

and t_o is the reference time at which relaxation process begins. Now when $t \rightarrow \infty \sqrt{J_2'} \rightarrow k^+$ (from above)

$$\lim_{t \rightarrow \infty} \sqrt{J_2'}\ (t) = k^+ \tag{2-76}$$

In other words, as the time approaches infinity, the stress deviator relaxes towards a point on the yield surface $J_2' = k^2$.

Perzyna (Ref. 6) has introduced more parameters in the visco-plastic model including hardening properties. He postulated a constitutive equation of the form

$$\dot{e}_{ij} = \frac{\dot{s}_{ij}}{2G} + \gamma < \phi(F) > \frac{\partial f}{\partial \sigma_{ij}} \tag{2-77}$$

$$\dot{\varepsilon}_{ii} = \frac{\dot{\sigma}_{ii}}{3K} \tag{2-78}$$

where:

$$F = F\left(\sigma_{ij}, \varepsilon_{ij}^{p}\right) = f\left(\frac{\sigma_{ij}, \varepsilon_{ij}^{p}}{\kappa}\right) - 1 \tag{2-79}$$

is the statical yield function. G and K are, respectively, the elastic shear and bulk moduli, and γ is a material constant. The symbol $< \phi(F) >$ is defined as follows

$$<\phi(F)> = \begin{cases} 0 & \text{for} \quad F \leq 0 \\ \phi(F) & \text{for} \quad F > 0 \end{cases} \tag{2-80}$$

The function $f(\sigma_{ij}, \varepsilon_{ij}^{P})$ depends on the state of stress σ_{ij} and on the state of anelastic strain ε_{ij}^{P} and

$$\kappa = \kappa(W_p) = \kappa\left(\int_{0}^{\varepsilon_{ij}^{p}} \sigma_{ij} \, d\varepsilon_{ij}^{p}\right) \tag{2-81}$$

is the work-hardening parameter. Consider the anelastic part of the constitutive equation (2-77)

$$\dot{\varepsilon}_{ij}^{p} = \gamma\phi(F) \frac{\partial f}{\partial \sigma_{ij}} \tag{2-82}$$

squaring both sides of equation (2-82) we obtain

$$\dot{\varepsilon}_{ij}^{p} \, \dot{\varepsilon}_{ij}^{p} = \gamma^2 \left\{\phi(F) \frac{\partial f}{\partial \sigma_{ij}}\right\}^2 \tag{2-83}$$

Now let

$$I_2^p = \frac{1}{2}\,\dot{\varepsilon}_{ij}^p\,\dot{\varepsilon}_{ij}^p \qquad (2\text{-}84)$$

where I_2^P is the second invariant of the plastic strain rate. Then equation (2-83) becomes

$$2I_2^p = \gamma^2\left\{\phi(F)\,\frac{\partial f}{\partial \sigma_{ij}}\right\}^2 \qquad (2\text{-}85)$$

or

$$F - \phi^{-1}\left(\frac{I_2^p}{\frac{1}{2}\gamma^2\,\frac{\partial f}{\partial \sigma_{ij}}\,\frac{\partial f}{\partial \sigma_{ij}}}\right)^{\frac{1}{2}} = 0 \qquad (2\text{-}86)$$

where ϕ^{-1} denotes the functional inverse of ϕ or

$$f\left(\sigma_{ij}, \varepsilon_{ij}^p\right) = \kappa\,(W_p)\left\{1 + \phi^{-1}\left[\frac{I_2^p}{\frac{1}{2}\gamma^2\,\frac{\partial f}{\partial \sigma_{kl}}\,\frac{\partial f}{\partial \sigma_{kl}}}\right]^{\frac{1}{2}}\right\} \qquad (2\text{-}87)$$

This expression implicitly represents dynamical yield condition for elasto-viscoplastic work hardening materials and also describes the dependence of the yield criterion on the strain rate. From equation (2-82) it is evident that the plastic strain rate vector $\dot{e}_{ij}^p$ in the 9-D stress hyper-space is always along the normal to the subsequent loading surface. (See Figure 2-10.) The yield theory described above reduces to classical rate independent (inviscid) theory for vanishingly small strain rates.

Different forms of the function ϕ (F) in equation (2-77) have been proposed by Perzyna in Ref. 6 as

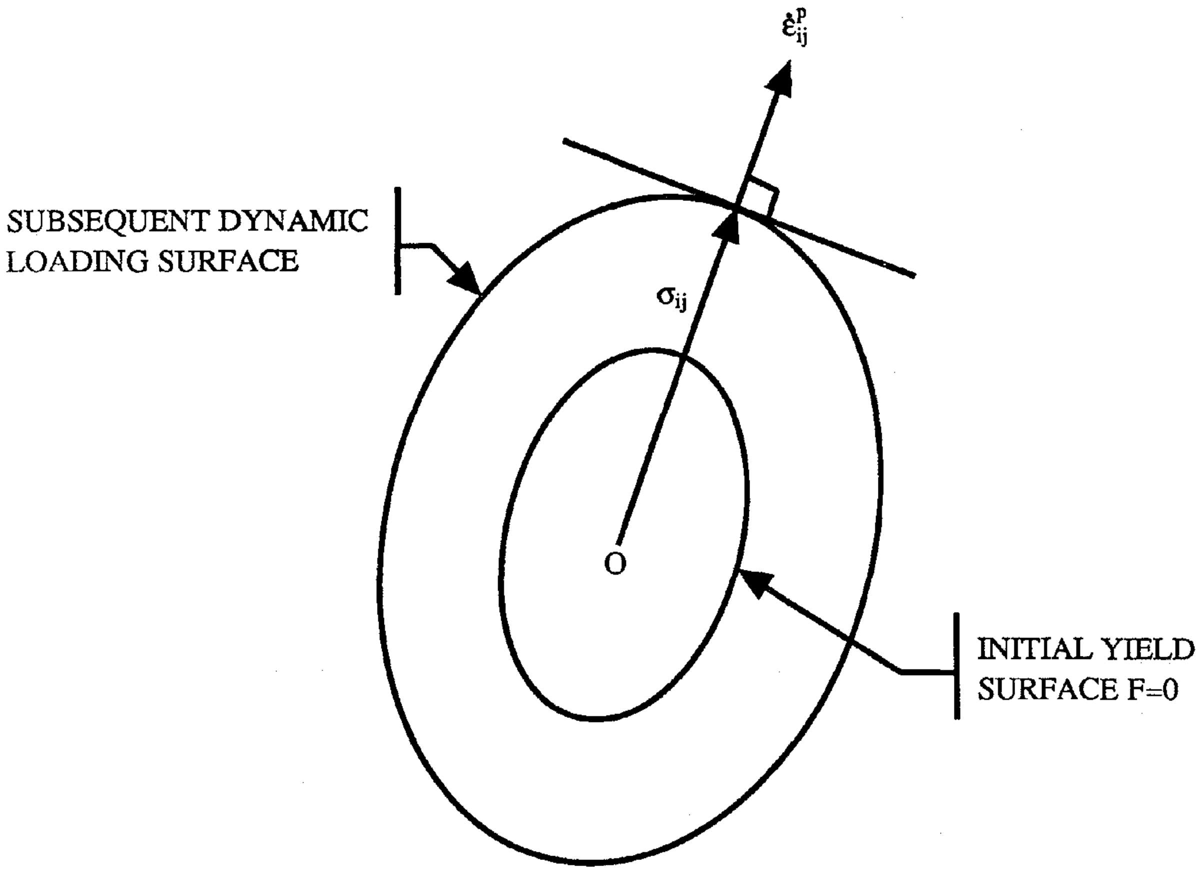


FIGURE 2-10 Typical Yield Surface For An Elasto-Viscoplastic Material

$$\phi(F) = F^{\delta}$$

$$\phi(F) = F$$

$$\phi(F) = \exp F - 1 \tag{2-88}$$

$$\phi(F) = \sum_{\alpha=1}^{N} A\alpha \left[\exp F^{\alpha} - 1\right]$$

$$\phi(F) = \sum_{\alpha=1}^{N} B_{\alpha} F^{\alpha}$$

The above functions, however, describe perfectly plastic rate sensitive material under dynamic loading.

Hohenemser and Prager have proposed an elasto-viscoplastic constitutive equation of the form

$$\dot{e}_{ij} = \gamma\,[F_M]\,\frac{s_{ij}}{\sqrt{J_2'}} \tag{2-89}$$

and

$$\sqrt{J_2'} = k_o \left(1 + \frac{\dot{I}_2'}{\gamma}\right) \tag{2-90}$$

where k_o is the yield stress in simple shear and

$$\dot{I}_2' = \frac{1}{2}\,\dot{e}_{ij}\,\dot{e}_{ij} \tag{2-91}$$

γ is a material constant and

$$F_M = \frac{\sqrt{J_2'}}{k_o} - 1 \qquad (2\text{-}92)$$

To generalize equation (2-89) to include the effect of work hardening, Rosenblatt (Ref. 12) proposed a constitutive equation of the form

$$\dot{e}_{ij} = \gamma\,[\phi\,(F)]\,\frac{S_{ij}}{\sqrt{J_2'}} \qquad (2\text{-}93)$$

This relation, however, violates the normality requirement sufficient for uniqueness, in the limit of vanishing inelastic strain rates except for the special case where the expression is governed by equation (2-92).

2.8.3 Elastic-Viscoplastic Models

In these models, the material behavior is elastic if stresses are below a certain limit. If stresses exceed this limit, material exhibits instantaneous plastic deformation in addition to a delayed (viscous) deformation. The total strain rate is given by

$$\dot{\varepsilon}_{ij} = \dot{\varepsilon}_{ij}^{E} + \dot{\varepsilon}_{ij}^{P} + \dot{\varepsilon}_{ij}^{VP} \qquad (2\text{-}94)$$

where $\dot{\varepsilon}_{ij}^{E}$, $\dot{\varepsilon}_{ij}^{P}$, $\dot{\varepsilon}_{ij}^{VP}$ denote, respectively, the elastic, plastic, viscoplastic strain rates.

Yannis F. Dafalias (Ref. 20) has used the above postulate to model the behavior of cohesive soils. Dafalias used constitutive equations of the form

$$\begin{aligned} \dot{\varepsilon}_{ij} &= C_{ijkl}\,\dot{\sigma}_{kl} \\ &+ \left\langle \frac{1}{\kappa_p}\frac{\partial f}{\partial \sigma_{ij}}\,\dot{\sigma}_{ij} \right\rangle \frac{\partial f}{\partial \sigma_{ij}} + \left\langle \phi\left(\Delta\hat{\sigma}\right)\right\rangle R_{ij}^{v} \end{aligned} \qquad (2\text{-}95)$$

where:

C_{ijkl} = Elastic compliance

$f(\sigma_{ij}, q_n^p) = 0$ = Yield surface

$$\dot{q}_n^p = < \frac{1}{\kappa_p} \frac{\partial f}{\partial \sigma_{ij}} \dot{\sigma}_{ij} > r_n^p \tag{2-96}$$

(the plastic modulus)

$$\kappa_p = - \frac{\partial f}{\partial q_n^p} r_n^p \tag{2-97}$$

and r_n^p is a function of the state only, $\kappa_p > 0$ denotes stable response and $\kappa_p \leq 0$ denotes unstable response, $\Delta\hat{\sigma}$ is overstress. The notation < > is defined such that

$<F> = 0$ if $F \leq 0$ and

$<F> = F$ if $F > 0$

From normality principle

$$R_{ij}^{v} = \frac{\partial F}{\partial \bar{\sigma}_{ij}} \tag{2-98}$$

where F defines the bounding surface given by

$$F\left(\bar{\sigma}_{ij}, q_n^p\right) = 0 \tag{2-99}$$

The functional form of F = 0 can be similar to the form of f = 0. Bars over stress quantities indicate points on F = 0, and the actual stress lies in or on F = 0.

2.9 Viscoelastic-Plastic Models

In the viscoelastic-plastic model (Ref. 5), the total strain is the sum of a viscoelastic component and a plastic component, i.e.

$$e_{ij} = e_{ij}^{VE} + e_{ij}^{p} \tag{2-100}$$

where e_{ij}, $e_{ij,}^{VE}$, e_{ij}^{p} denote, respectively, the total, viscoelastic, and plastic strains. The viscoelastic component of strain follows a creep integral law of classical linear viscoelastic theory of the form

$$e_{ij}^{VE}(t) = s_{ij}^{o}(x)J_1(t) + \int_{o}^{t} J_1(t-\tau)\,\frac{\partial s_{ij}(x,\tau)}{\partial \tau}\,d\tau \tag{2-101}$$

$$\varepsilon_{kk}^{VE}(t) = \sigma_{kk}^{o}(x)J_2(t) + \int_{o}^{t} J_2\,(t-\tau)\,\frac{\partial \sigma_{kk}(x,\tau)}{\partial \tau}\,d\tau \tag{2-102}$$

where J_1 and J_2 denote, respectively, the creep functions in shear and isotropic compression (or dilatation) which may have finite jump discontinuities at $t = 0$, $s_{ij}^{o}(x)$ stands for $s_{ij}(x,0^{+})$, and so on. The initial response of the viscoelastic solid of the type represented by equations (2-101) and (2-102) is assumed to be elastic. That is

$$e_{ij}^{E} = e_{ij}^{VE}(0) = \frac{s_{ij}}{2G} \tag{2-103}$$

Similarly

$$\varepsilon_{kk}^{E} = \varepsilon_{kk}^{VE}(0) = \frac{\sigma_{kk}}{3K} \tag{2-104}$$

Consider the arbitrary 9-dimensional yield surface in stress space

$$f = f\left(\sigma_{ij}, \varepsilon_{ij}^{p}, \chi_{ij}, \kappa_{ij}\right) \tag{2-105}$$

$i = 1.2,3$

$j = 1,2,3$

where:

σ_{ij} = State of stress at a generic point in the stress space

$\varepsilon^{P}{}_{ij}$ = Plastic strain

χ_{ij} = Work-hardening effect due to time history

κ_{ij} = Effect of work-hardening due to path history alone

Define the functional

$$\chi_{ij} = \chi_{ij}\left(\varepsilon_{kl}^{V} - \varepsilon_{kl}^{E}\right) \tag{2-106}$$

and

$$\varepsilon_{ij}^{p} = \varepsilon_{ij}^{p}\left(\sigma_{kl}, \chi_{mn}, \kappa_{pq}\right) \tag{2-107}$$

where ε_{kl}^{V} denotes time dependent strain tensor and ε_{kl}^{E} denotes elastic strain tensor. Now consider the time rate of f in equation (2-105)

$$\dot{f} = \frac{\partial f}{\partial \sigma_{ij}} \dot{\sigma}_{ij} + \frac{\partial f}{\partial \varepsilon_{ij}^{p}} \dot{\varepsilon}_{ij}^{p} + \frac{\partial f}{\partial \chi_{ij}} \dot{\chi}_{ij} + \frac{\partial f}{\partial \kappa_{ij}} \dot{\kappa}_{ij} \tag{2-108}$$

The loading criterion is as follows: If

$$\frac{\sigma f}{\partial \sigma_{ij}} \dot{\sigma}_{ij} + \frac{\partial f}{\partial \chi_{ij}} \dot{\chi}_{ij} < 0 \text{ and } f < 0; \text{(unloading)} \tag{2-109}$$

If

$$\frac{\partial f}{\partial \sigma_{ij}}\,\dot{\sigma}_{ij} + \frac{\partial f}{\partial \chi_{ij}}\,\dot{\chi}_{ij} = 0 \text{ and } f = 0; \text{ (neutral loading)} \tag{2-110}$$

If

$$\frac{\partial f}{\partial \sigma_{ij}}\,\dot{\sigma}_{ij} + \frac{\partial f}{\partial \chi_{ij}}\,\dot{\chi}_{ij} > 0 \text{ and } f = 0; \text{ (loading)} \tag{2-111}$$

Constitutive Equations

Consider the yield surface given by

$$f = f(\sigma_{ij}\ \kappa_{ij}, \chi) \tag{2-112}$$

where f is the instantaneous yield surface similar to equation (2-105). However, here $\kappa_{ij} = \varepsilon^p_{ij}$ and χ_{ij} is replaced by χ.

$$\dot{f} = \frac{\partial f}{\partial \sigma_{ij}}\,\dot{\sigma}_{ij} + \frac{\partial f}{\partial \varepsilon^p_{ij}}\,\dot{\varepsilon}^p_{ij} + \frac{\partial f}{\partial \chi}\,\dot{\chi} \tag{2-113}$$

Since $\dot{\varepsilon}^p_{ij}$ is directed to the normal of the instantaneous loading surface f

$$\dot{\varepsilon}^p_{ij} = \begin{cases} \Lambda \dfrac{\partial f}{\partial \sigma_{ij}} & \text{if } f = 0 \\ 0 & \text{if } f < 0 \end{cases} \tag{2-114}$$

Substitute equation (2-114) into equation (2-113)

$$\frac{\partial f}{\partial \sigma_{ij}}\,\dot{\sigma}_{ij} + \frac{\partial f}{\partial \varepsilon^p_{ij}}\,\Lambda\frac{\partial f}{\partial \sigma_{ij}} + \frac{\partial f}{\partial \chi}\,\dot{\chi} = 0 \tag{2-115}$$

Therefore

$$\Lambda = - \frac{\dfrac{\partial f}{\partial \sigma_{kl}} \dot{\sigma}_{kl} + \dfrac{\partial f}{\partial \chi} \dot{\chi}}{\dfrac{\partial f}{\partial \varepsilon^p_{kl}} \dfrac{\partial f}{\partial \sigma_{kl}}} \tag{2-116}$$

Substitute equation (2-116) into equation (2-114)

$$\begin{aligned} \dot{\varepsilon}^p_{ij} &= \frac{\dfrac{\partial f}{\partial \sigma_{kl}} \dot{\sigma}_{ij} + \dfrac{\partial f}{\partial \chi} \dot{\chi}}{\dfrac{\partial f}{\partial \varepsilon^p_{kl}} \dfrac{\partial f}{\partial \sigma_{kl}}} \frac{\partial f}{\partial \sigma_{ij}} \text{ if } f = 0 \\ \dot{\varepsilon}^p_{ij} &= 0 \text{ if } f < 0 \end{aligned} \tag{2-117}$$

SECTION 3
THE CAP MODEL AND DEVELOPMENT OF AN EXPLICIT FORM OF ELASTO-PLASTIC CONSTITUTIVE MATRIX OF MATERIAL

3.1 The Cap Model

The cap model (Ref. 1) is a classical incremental plasticity model defined by a yield surface and a plastic strain rate vector (Figure 3-1). The model exhibits three different modes of behavior: elastic, failure, and cap.

Elastic Mode: The elastic mode of behavior occurs when the stress point is within the failure envelope, and stress changes result in recoverable deformations.

Failure Mode: During the failure mode of behavior, the stress point lies on the failure envelope with a stress-strain relation given by equation (3-1). As shown in Figure 3-1, the associated flow rule requires that the plastic strain rate vector be directed upward and to the left. Therefore, the plastic strain during failure is composed of a deviatoric or shear component together with a volumetric, or dilatant component.

Cap Mode: The cap mode of behavior occurs when the stress point lies on the movable cap and pushes it outward. The stress-strain relation is given by equation (3-2). As shown in Figure 3-1. the associated flow rule requires that during cap action the plastic strain rate vector be directed upward and to the right. This implies that the plastic strain rate produces an irreversible decrease in volume in conjunction with the irreversible shearstrain. This reduction in volume is referred to as compaction.

This compaction leads to an increase in the cap parameter ε_V^P which, in turn, through equation (3-5), leads to an increase in $X(\kappa)$ and hence the cap moves to the right. Either J_1 or $\sqrt{J_2'}$ or both must increase to maintain the cap mode of behavior.

In soils, the dilatancy associated with failure leads to a decrease in $\bar{\varepsilon}_V^P$, resulting in a leftward movement of the cap. This cap movement is limited if and when the cap reaches the stress point

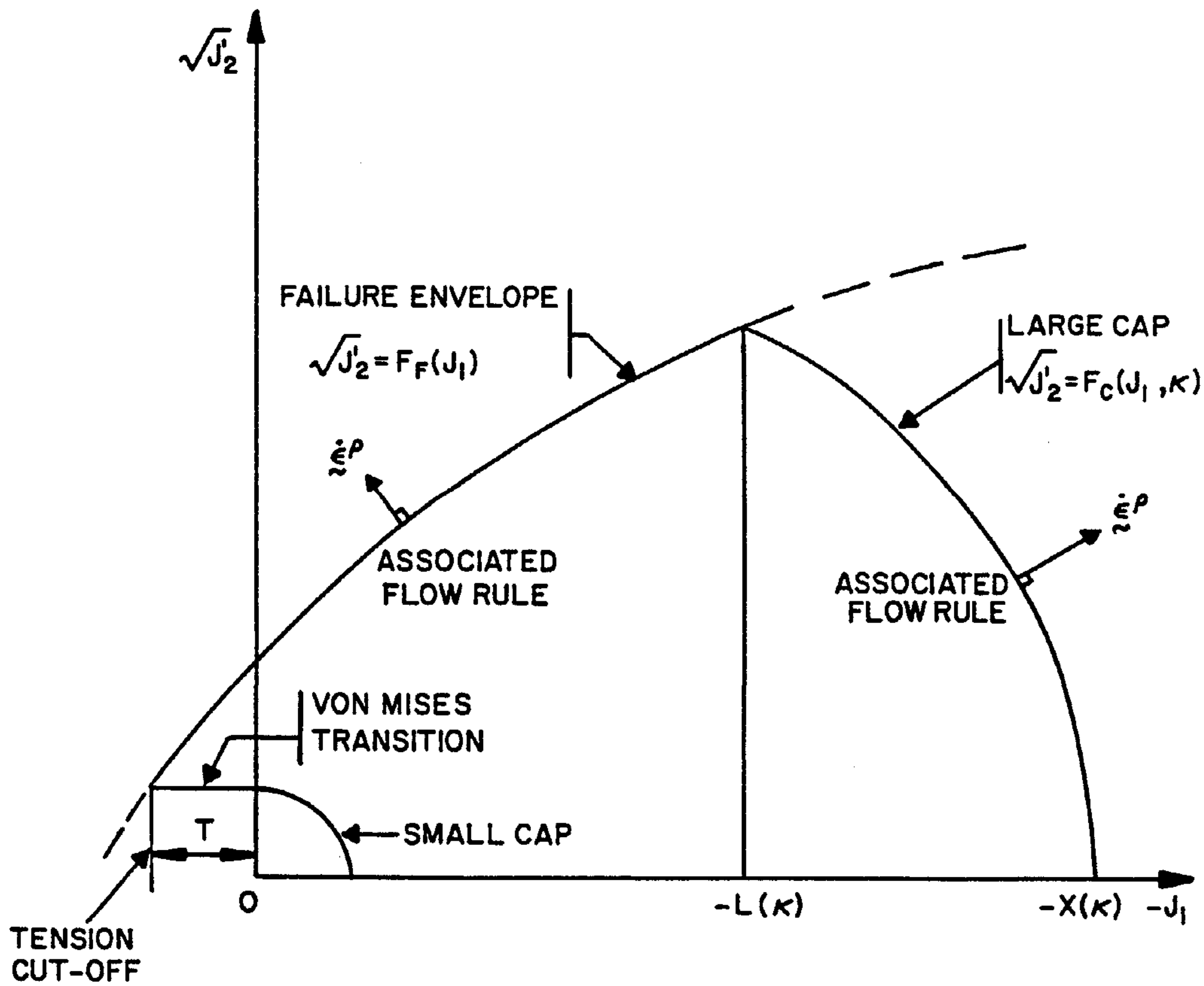


FIGURE 3-1 Typical Yield Surface In Cap Model

(so that the stress point is at a corner of the yield surface). When this occurs, the associated flow rule requires that the plastic strain rate vector lie between the outward drawn normals at the corner. At this point, the subsequent plastic straining is purely in shear.

Constitutive Equations (see Figure 3-1)

The following constitutive relations were developed for the McCormick Ranch Sand (Ref. 1).

Failure Mode:

$$F_F(J_1) = A - C \exp(BJ_1) \quad J_1 < L \tag{3-1}$$

Cap Mode:

$$F_c(J_1, \kappa) = \frac{1}{R} \sqrt{\left\{[X(\kappa) - L(\kappa)]^2 - [J_1 - L(\kappa)]^2\right\}} \quad L < J_1 < X \tag{3-2}$$

where:

$$L(\kappa) = \begin{cases} \kappa \text{ if } \kappa < 0 \\ 0 \text{ if } \kappa \geq 0 \end{cases} \tag{3-3}$$

$$X(\kappa) = \kappa - RF_F(\kappa) \tag{3-4}$$

and A,B,C, and R are material parameters and κ is a hardening parameter.

$$\bar{\varepsilon}_V^p = W\{\exp[DX(\kappa)] - 1\} \tag{3-5}$$

$$\dot{\varepsilon}_V^p = \begin{cases} \dot{\bar{\varepsilon}}_V^p \text{ if } \dot{\bar{\varepsilon}}_V^p \leq 0 \text{ or } \kappa < J_1 \text{ and } \kappa < 0 \\ 0 \quad \text{otherwise} \end{cases} \tag{3-6}$$

Tension cutoff is assumed to be represented by

$$J_1 = T \tag{3-7}$$

where T denotes the maximum allowable hydrostatic tension.

3.1.1 Derivatives of Cap Functions

Consider the yield function in terms of stress invariants and a hardening parameter given by

$$F\left(J_1, \sqrt{J_2'}, \kappa\right) = 0 \tag{3-8}$$

where:

$$J_1 = \sigma_{ij}\,\delta_{ij} = \sigma_{kk} \tag{3-9}$$

in which δ_{ij} is the Kronecker delta and

$$J_2' = \frac{1}{2}\, s_{ij}\, s_{ij} = \frac{1}{2}\left(\sigma_{ij} - \frac{1}{3}\,\sigma_{kk}\,\delta_{ij}\right)\left(\sigma_{ij} - \frac{1}{3}\,\sigma_{kk}\,\delta_{ij}\right) \tag{3-10}$$

and κ is the hardening parameter. Now differentiating equation (3-8) we get

$$\partial F = \frac{\partial F}{\partial J_1}\,\partial J_1 + \frac{1}{2\sqrt{J_2'}}\,\frac{\partial F}{\partial\sqrt{J_2'}}\,\partial J_2' + \frac{\partial F}{\partial \kappa}\,\partial\kappa \tag{3-11}$$

Divide equation (3-11) by σ_{ij} and get

$$\frac{\partial F}{\partial \sigma_{ij}} = \frac{\partial F}{\partial J_1}\,\frac{\partial J_1}{\partial \sigma_{ij}} + \frac{1}{2\sqrt{J_2'}}\,\frac{\partial F}{\partial\sqrt{J_2'}}\,\frac{\partial J_2'}{\partial \sigma_{ij}} \tag{3-12}$$

or

$$\frac{\partial F}{\partial \sigma_{ij}} = \frac{\partial F}{\partial J_1}\,\delta_{ij} + \frac{s_{ij}}{2\sqrt{J_2'}}\,\frac{\partial F}{\partial \sqrt{J_2'}} \tag{3-13}$$

Failure Mode:

From equation (3-1)

$$F_F = \sqrt{J_2'} - A + C\exp\left(BJ_1\right) \tag{3-14}$$

Hence

$$\frac{\partial F_F}{\partial J_1} = CB\exp\left(BJ_1\right) \tag{3-15}$$

$$\frac{\partial F_F}{\partial \sqrt{J_2'}} = 1 \tag{3-16}$$

Hence from equation (3-13)

$$\frac{\partial F_F}{\partial \sigma_{ij}} = CB\exp\left(BJ_1\right)\,\delta_{ij} + \frac{s_{ij}}{2\sqrt{J_2'}} \tag{3-17}$$

Cap Mode:

From equation (3-2)

$$F_c = \sqrt{J_2'} - \frac{1}{R}\sqrt{\left\{[X(\kappa) - L(\kappa)]^2 - [J_1 - L(\kappa)]^2\right\}} \tag{3-18}$$

Differentiating equation (3-18) we get

$$-R^2 \sqrt{J_2'}\, \partial F_c + 2R^2 \partial J_2' = (X-L)\partial X + (J_1 - X)\ \partial L + (L - J_1)\ \partial J_1 \tag{3-19}$$

Divide equation (3-19) by $\partial\sigma_{ij}$ and rearrange terms

$$\frac{\partial F_c}{\partial \sigma_{ij}} = \frac{2s_{ij}}{\sqrt{J_2'}} - \frac{(L - J_1)}{R^2 \sqrt{J_2'}}\, \delta_{ij} \tag{3-20}$$

$$\frac{\partial F_c}{\partial X} = \frac{(L - X)}{R^2 \sqrt{J_2'}} \tag{3-21}$$

$$\frac{\partial F}{\partial L} = \frac{(X - J_1)}{R^2 \sqrt{J_2'}} \tag{3-22}$$

From equation (3-4)

$$\frac{\partial X}{\partial \kappa} = 1 + RCB \exp(B\kappa) \tag{3-23}$$

From equation (3-3)

$$\frac{\partial L}{\partial \kappa} = \begin{cases} 1 & \text{if } \kappa < 0 \\ 0 & \text{if } \kappa \geq 0 \end{cases} \tag{3-24}$$

Now

$$\frac{\partial F_c}{\partial \kappa} = \frac{\partial F_c}{\partial X} \bullet \frac{\partial X}{\partial \kappa} + \frac{\partial F_c}{\partial L} \bullet \frac{\partial L}{\partial \kappa} \tag{3-25}$$

$$= \frac{(L - X)}{R^2 \sqrt{J_2'}} [1 + RCB \exp(B\kappa)] + \frac{(X - J_1)}{R^2 \sqrt{J_2'}} \quad (3\text{-}26)$$

$$\text{if } \kappa < 0$$

$$\frac{\partial F_c}{\partial \kappa} = \frac{(L - x)}{R^2 \sqrt{J_2'}} [1 + RCB \exp(B\kappa)] \text{ if } \kappa \geq 0 \quad (3\text{-}27)$$

Elasto-Plastic Constitutive Matrix

Consider the yield surface defined by

$$F(\underset{\sim}{\sigma}, \kappa) = 0 \quad (3\text{-}28)$$

where:

κ = Hardening parameter

$\underset{\sim}{\sigma}$ = General state of stress

From incremental theory of plasticity

$$d\underset{\sim}{\varepsilon} = d\underset{\sim}{\varepsilon}^e + d\underset{\sim}{\varepsilon}^p \quad (3\text{-}29)$$

where:

$d\underset{\sim}{\varepsilon}$ = Increment of total strain

$d\underset{\sim}{\varepsilon}^e$ = Increment of elastic strain

$d\underset{\sim}{\varepsilon}^p$ = Increment of plastic strain

Now

$$d\underset{\sim}{\varepsilon}^e = [D]^{-1} d\underset{\sim}{\sigma} \quad (3\text{-}30)$$

$$d\underset{\sim}{\varepsilon}^{p} = \lambda \frac{\partial F}{\partial \underset{\sim}{\sigma}} \quad \text{if } F = 0$$
$$= 0 \qquad \text{if } F < 0 \tag{3-31}$$

This is known as the normality principle and since F is assumed to coincide with the plastic potential, this phenomenon is also referred to as associated flow rule. Equation (3-29) becomes

$$d\underset{\sim}{\varepsilon} = [D]^{-1}\, d\underset{\sim}{\sigma} + \lambda \frac{\partial F}{\partial \underset{\sim}{\sigma}} \tag{3-32}$$

Now differentiating equation (3-28) we get

$$\partial F = \frac{\partial F}{\partial \underset{\sim}{\sigma}}\, d\underset{\sim}{\sigma} + \frac{\partial F}{\partial \kappa}\, d\kappa = 0 \tag{3-33}$$

or

$$\left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^{T} d\underset{\sim}{\sigma} - A\lambda = 0 \tag{3-34}$$

where:

$$A = -\frac{1}{\lambda}\frac{\partial F}{\partial \kappa}\, d\kappa \tag{3-35}$$

and

$$d\underset{\sim}{\sigma} = [d\sigma_1\ d\sigma_2\ d\sigma_3\ d\sigma_4\ d\sigma_5\ d\sigma_6]^{T} \tag{3-36}$$

is the vector form of the increment of the stress tensor. Hence

$$d\sigma_1 = d\sigma_{11}$$

$$d\sigma_2 = d\sigma_{22}$$

$$d\sigma_3 = d\sigma_{33}$$

$$d\sigma_4 = d\sigma_{23} = d\sigma_{32}$$

$$d\sigma_5 = d\sigma_{13} = d\sigma_{31}$$

$$d\sigma_6 = d\sigma_{12} = d\sigma_{21} \tag{3-37}$$

Multiply equation (3-32) by $\left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T$ [D]. We get

$$\left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D]\, d\underset{\sim}{\varepsilon} = \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T d\underset{\sim}{\sigma} + \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D] \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\} \lambda \tag{3-38}$$

From equation (3-34)

$$\left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T d\underset{\sim}{\sigma} = A\lambda \tag{3-39}$$

Substituting equation (3-39) into equation (3-38) we get

$$\left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D]\, d\underset{\sim}{\varepsilon} = \left[A + \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D] \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\} \right] \lambda \tag{3-40}$$

$$\lambda = d\underset{\sim}{\varepsilon} \bullet \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D] \bullet \left[A + \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D] \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\} \right]^{-1} \tag{3-41}$$

From equation (3-32)

$$d\underset{\sim}{\sigma} = [D]\, d\underset{\sim}{\varepsilon} - \lambda[D] \frac{\partial F}{\partial \underset{\sim}{\sigma}} \tag{3-42}$$

Substituting value of λ into equation (3-42) we get

$$d\underset{\sim}{\sigma} = \left([D] - [D] \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\} \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D] \bullet \left(A + \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D] \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\} \right)^{-1} \right) d\underset{\sim}{\varepsilon} \tag{3-43}$$

The elasto-plastic stress strain relation can be expressed in incremental form as

$$d\underset{\sim}{\sigma} = [D^{ep}]\ d\underset{\sim}{\varepsilon} \tag{3-44}$$

where $[D^{ep}]$, the elasto-plastic constitutive matrix, is given by

$$[D^{ep}] = [D] - [D] \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\} \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D] \bullet \left(A + \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\}^T [D] \left\{\frac{\partial F}{\partial \underset{\sim}{\sigma}}\right\} \right)^{-1} \tag{3-45}$$

and [D] is the elastic constitutive matrix. The work hardening parameter κ is taken as the amount of plastic work done during plastic deformation. Thus

$$d\kappa = \sigma_1\, d\varepsilon_1^p + \sigma_2\, d\varepsilon_2^p + \bullet\bullet\bullet + \sigma_6\, d\varepsilon_6^p = \underset{\sim}{\sigma}^T d\underset{\sim}{\varepsilon}^p \tag{3-46}$$

Substituting equation (3-46) into equation (3-31) we get

$$d\kappa = \lambda \underset{\sim}{\sigma}^T \frac{\partial F}{\partial \underset{\sim}{\sigma}} \tag{3-47}$$

Eliminate λ by substituting equation (3-47) into equation (3-35) and get

$$A = - \frac{\partial F}{\partial \kappa} \underset{\sim}{\sigma}^T \frac{\partial F}{\partial \underset{\sim}{\sigma}} \tag{3-48}$$

SECTION 4
THE PROPOSED Q-MODEL

Wave energy dissipation in a medium results not only from dispersion of wave energy from the source but also from damping by energy losses within the medium. In this report, attention is focused on the latter.

Material damping has been modeled in diverse ways in recent years. The Rayleigh damping of the form

$$C = \alpha M + \beta K \tag{4-1}$$

has been used with some success in time stepping schemes. Here C is the damping matrix, α, β are constants, M and K are mass and stiffness matrices, respectively.

The proposed Q-model emphasizes the viscous and energy dissipation characteristics of the material. This model is of great advantage because all the parameters involved are readily determined from tests, and the resulting damping expression is very easy to apply in time stepping schemes, and thus lends itself readily for use in computer codes. The Q-model is used in conjunction with elastic or elastoplastic constitutive matrix to produce a viscoelastic or viscoplastic material model, respectively.

4.1 Theoretical Development

The specific dissipation factor, sometimes referred to as the coefficient of internal friction is defined as

$$Q^{-1} = \frac{\Delta W}{2\pi W} \tag{4-2}$$

where W is the elastic strain energy stored per unit cycle per unit volume, and ΔW is the energy dissipated per unit cycle per unit volume.

In terms of phase angle

$$Q^{-1} = \tan \phi \tag{4-3}$$

where ϕ is the phase angle between stress and strain.

In terms of logarithmic decrement

$$Q^{-1} = \frac{\delta}{\pi} \tag{4-4}$$

where δ is the logarithmic decrement.

Now consider the complex spring-mass system as shown in Figure 4-1. The equation of motion of the system can be expressed as

$$F(t) - k^* x = m\ddot{x} \tag{4-5}$$

where k^* is the complex spring constant given by

$$k^* = k(1 + i\tan \phi) \tag{4-6}$$

where ϕ is the coefficient of internal friction, and m is the mass of the system. From equation (4-5)

$$m\ddot{x} + k(1 + i\tan \phi)\ x = F(t) \tag{4-7}$$

By Fourier Transformation of equation (4-7) we get

$$-\omega^2 mx + k(1 + i\tan \phi)\ x = \overline{F}(\omega) \tag{4-8}$$

Substitute equation (4-3) into equation (4-8) and get

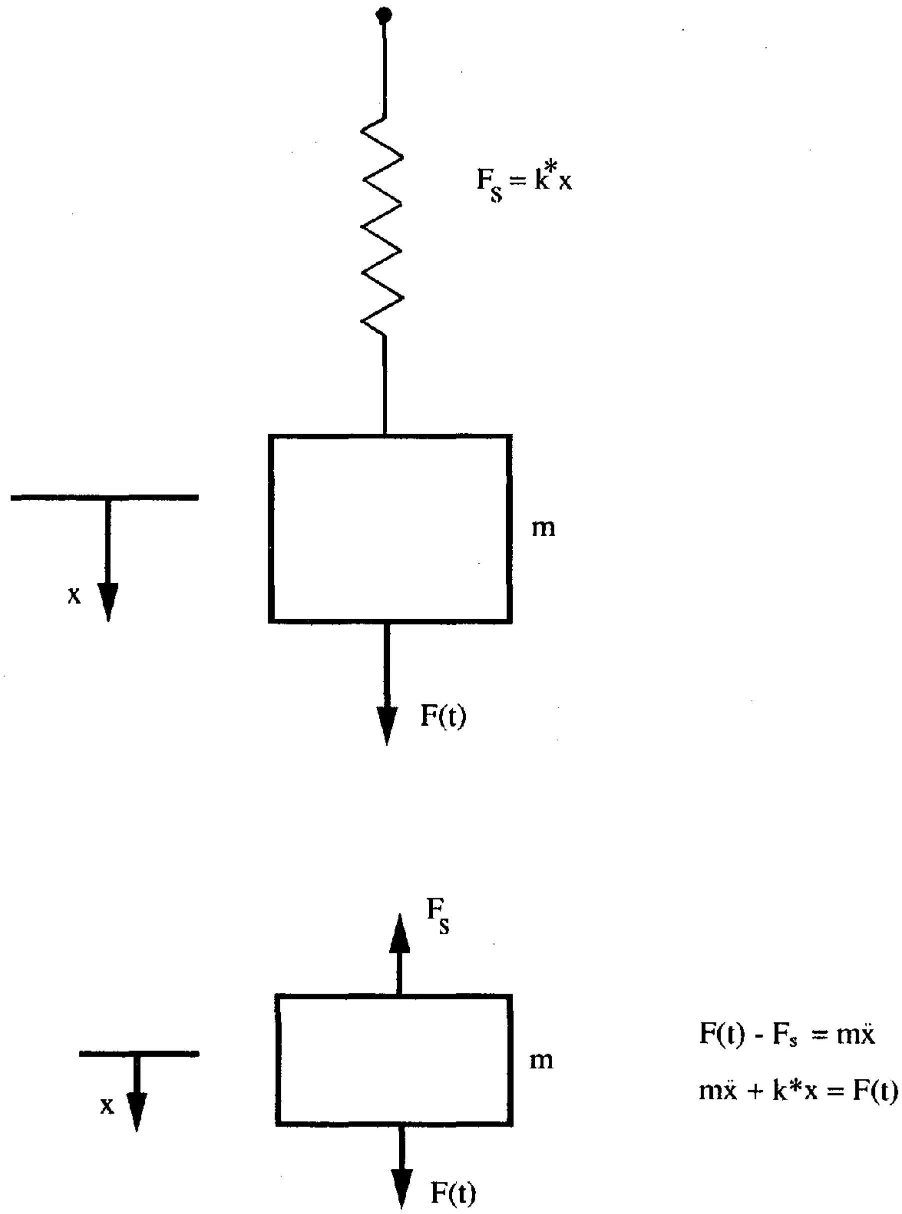


FIGURE 4-1 Mechanical Model For Single Degree Of Freedom Anelastic Spring Mass System

$$-\omega^2 mx + k(1 + iQ^{-1})\ x = \overline{F}\ (\omega) \tag{4-9}$$

Alternatively, for a single degree of freedom system, the equation of motion of the system can be expressed as

$$m\ddot{x} + c(t)\dot{x} + kx = F(t) \tag{4-10}$$

where:

m = mass constant

c = damping constant

k = spring constant

Taking Fourier Transform of equation (4-10) we get

$$-\omega^2 mx + (k + i\omega\overline{c}\ (\omega))\ x = \overline{F}\ (\omega) \tag{4-11}$$

or

$$-\omega^2 mx + k\left(1 + \frac{i\omega\overline{c}\ (\omega)}{k}\right)x = \overline{F}\ (\omega) \tag{4-12}$$

Comparing the complex terms of equations (4-9) and (4-12) we get

$$Q^{-1} = \frac{\omega\overline{c}\ (\omega)}{k} \tag{4-13}$$

or

$$\overline{c}\ (\omega) = \frac{kQ^{-1}}{\omega} \tag{4-14}$$

In general, $\overline{c}(\omega)$ could be expanded in the form of a Laurent Series with "even powers of ω." For simplicity we can write

$$\bar{c}\ (\omega) = a_o + \frac{a_1}{\omega^2} + \frac{a_2}{\omega^4} + \cdots \tag{4-15}$$

in order to avoid the involvement of higher derivatives of x(t) associated with terms like b's in the general form:

$$\bar{c}\ (\omega) = (b_1\,\omega^2 + b_2\,\omega^4 + \cdots) + a_o + \frac{a_1}{\omega^2} + \frac{a_2}{\omega^4} + \cdots \tag{4-16}$$

Consider a constant Q-model (i.e., frequency independent over a prescribed frequency range). Let

$$Q(\omega) \approx Q_o \text{ between } \Omega_1 < \omega < \Omega_2,\ \omega > 0 \tag{4-17}$$

then from equation (4-14)

$$\bar{c}\ (\omega) = \frac{kQ_o^{-1}}{\omega} \approx a_o + \frac{a_1}{\omega^2} + \frac{a_2}{\omega^4} \tag{4-18}$$

or

$$kQ_o^{-1} \approx a_o\,\omega + \frac{a_1}{\omega} + \frac{a_2}{\omega^3} \quad \omega \geq 0 \tag{4-19}$$

The mean square error $e(\Omega_1,\Omega_2)$ between Ω_1 and Ω_2 is given by

$$e = \int_{\Omega_1}^{\Omega_2} \left(kQ_o^{-1} - a_o\,\omega - \frac{a_1}{\omega} - \frac{a_2}{\omega^3} \right)^2 d\omega \tag{4-20}$$

Minimization of $e(\Omega_1,\Omega_2)$ to obtain the constant a's gives

$$\frac{\partial e}{\partial a_o} = -2\int_{\Omega_1}^{\Omega_2}\left(kQ_o^{-1} - a_o\omega - \frac{a_1}{\omega} - \frac{a_2}{\omega^3}\right)\omega d\omega = 0 \tag{4-21}$$

or

$$\frac{kQ_o^{-1}}{2}\left(\Omega_2^2 - \Omega_1^2\right) - \frac{a_o}{3}\left(\Omega_2^3 - \Omega_1^3\right) - a_1\left(\Omega_2 - \Omega_1\right) + a_2\left(\frac{1}{\Omega_2} - \frac{1}{\Omega_1}\right) = 0 \tag{4-22}$$

$$\frac{\partial e}{\partial a_1} = -2\int_{\Omega_1}^{\Omega_2}\left(kQ_o^{-1} - a_o\omega - \frac{a_1}{\omega} - \frac{a_2}{\omega^3}\right)\frac{1}{\omega}\, d\omega = 0 \tag{4-23}$$

or

$$\begin{aligned} &- kQ_o^{-1}\ln\left(\frac{\Omega_2}{\Omega_1}\right) - a_o\left(\Omega_2 - \Omega_1\right) \\ &+ a_1\left(\frac{1}{\Omega_2} - \frac{1}{\Omega_1}\right) + \frac{a_2}{3}\left(\frac{1}{\Omega_2^3} - \frac{1}{\Omega_1^3}\right) = 0 \end{aligned} \tag{4-24}$$

$$\frac{\partial e}{\partial a_2} = -2\int_{\Omega_1}^{\Omega_2}\left(kQ_o^{-1} - a_o\omega - \frac{a_1}{\omega} - \frac{a_2}{\omega^3}\right)\frac{1}{\omega^3}\, d\omega = 0 \tag{4-25}$$

or

$$\begin{aligned} &- \frac{kQ_o^{-1}}{2}\left(\frac{1}{\Omega_2^2} - \frac{1}{\Omega_1^2}\right) + a_o\left(\frac{1}{\Omega_2} - \frac{1}{\Omega_1}\right) \\ &+ \frac{a_1}{3}\left(\frac{1}{\Omega_2^3} - \frac{1}{\Omega_1^3}\right) + \frac{a_2}{5}\left(\frac{1}{\Omega_2^5} - \frac{1}{\Omega_1^5}\right) = 0 \end{aligned} \tag{4-26}$$

Equations (4-22), (4-24) and (4-26) can be expressed in matrix form as

$$\begin{bmatrix} \frac{1}{3}\left(\Omega_2^3 - \Omega_1^3\right) & \left(\Omega_2 - \Omega_1\right) & -\left(\frac{1}{\Omega_2} - \frac{1}{\Omega_1}\right) \\ \left(\Omega_2 - \Omega_1\right) & -\left(\frac{1}{\Omega_2} - \frac{1}{\Omega_1}\right) & -\frac{1}{3}\left(\frac{1}{\Omega_2^3} - \frac{1}{\Omega_1^3}\right) \\ -\left(\frac{1}{\Omega_2} - \frac{1}{\Omega_1}\right) & -\frac{1}{3}\left(\frac{1}{\Omega_2^3} - \frac{1}{\Omega_1^3}\right) & -\frac{1}{5}\left(\frac{1}{\Omega_2^5} - \frac{1}{\Omega_1^5}\right) \end{bmatrix} \begin{bmatrix} a_o \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \frac{kQ_o^{-1}}{2}\left(\Omega_2^2 - \Omega_1^2\right) \\ -kQ_o^{-1}\ln\left(\frac{\Omega_2}{\Omega_1}\right) \\ -\frac{kQ_o^{-1}}{2}\left(\frac{1}{\Omega_2^2} - \frac{1}{\Omega_1^2}\right) \end{bmatrix}$$

The above system of equations can be solved for the values of a_o, a_1 and a_2. Multiply both sides of equation (4-15) by $i\omega\bar{x}$, and get

$$\bar{c}(\omega)\, i\omega\bar{x} = a_o i\omega\bar{x} + \frac{a_1}{\omega^2} i\omega\bar{x} + \frac{a_2 i\omega\bar{x}}{\omega^4} \tag{4-27}$$

or

$$\bar{c}(\omega) i\omega\bar{x} = a_o i\omega\bar{x} - \frac{a_1\bar{x}}{i\omega} + \frac{(i)^4 a_2\bar{x}}{(i\omega)^3} \tag{4-28}$$

Define the operator

$$D = \frac{d}{dt} = i\omega \tag{4-29}$$

then the inverse operator

$$D^{-1} = \int_o^t d\tau = \frac{1}{i\omega} \tag{4-30}$$

From the relation in equations (4-29) and (4-30), equation (4-28) can be rewritten in the time domain as

$$c(t)\,Dx(t) = a_o Dx(t) - a_1 D^{-1} x(t) + a_2 D^{-3} x(t) \tag{4-31}$$

or

$$c(t)\dot{x} = a_o \dot{x} - a_1 \int_0^t x(\tau)d\tau + a_2 \int_0^t\int_0^\tau\int_0^\tau x(\tau)d\tau d\tau d\tau \tag{4-32}$$

Substituting equation (4-32) into equation (4-10), the equation of motion of the system becomes

$$\begin{aligned} F(t) &= m\ddot{x} + a_o \dot{x} + kx - a_1 \int_0^t x(\tau)d\tau \\ &+ a_2 \int_0^t\int_0^\tau\int_0^\tau x(\tau)d\tau d\tau d\tau \end{aligned} \tag{4-33}$$

4.2 Numerical Procedure

The equation of motion of the system as presented in equation (4-33) can be solved numerically. Two integration schemes -- the explicit time integration scheme using the Central Difference Method and the implicit time integration scheme using the Newmark Method -- are considered here.

Let

$$z_1 = \int_0^t\int_0^\tau\int_0^\tau x(\tau)d\tau d\tau d\tau \tag{4-34}$$

$$z_3 = \int_0^t x(\tau)d\tau \tag{4-35}$$

$$z_4 = x \tag{4-36}$$

$$z_5 = \dot{x} \tag{4-37}$$

$$z_6 = \ddot{x} \tag{4-38}$$

Then

$$\frac{d}{dt}\begin{Bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{Bmatrix} = \begin{Bmatrix} z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \end{Bmatrix} \tag{4-39}$$

and equation (4-33) becomes

$$F(t) = mz_6 + a_o z_5 + kz_4 - a_1 z_3 + a_2 z_1 \tag{4-40}$$

from which

$$z_6 = \frac{1}{m}\left[F(t) - a_o z_5 - kz_4 + a_1 z_3 - a_2 z_1\right] \tag{4-41}$$

4.2.1 Explicit Time Integration Scheme Using Central Difference Method

From Central Difference Method the following expressions are obtained

$$^{t}\dot{z}_1 = \frac{1}{2\Delta t}\left(^{t+\Delta t}z_1 - {}^{t-\Delta t}z_1\right) = {}^{t}z_2 \tag{4-42}$$

$$^{t}\dot{z}_2 = \frac{1}{2\Delta t}\left(^{t+\Delta t}z_2 - {}^{t-\Delta t}z_2\right) = {}^{t}z_3 \tag{4-43}$$

$$^{t}\dot{z}_3 = \frac{1}{2\Delta t}\left(^{t+\Delta t}z_3 - {}^{t-\Delta t}z_3\right) = {}^{t}z_4 \tag{4-44}$$

$$^{t}\dot{z}_4 = \frac{1}{2\Delta t}\left(^{t+\Delta t}z_4 - {}^{t-\Delta t}z_4\right) = {}^{t}z_5 \tag{4-45}$$

$$^{t}\dot{z}_5 = \frac{1}{2\Delta t}\left({}^{t+\Delta t}z_5 - {}^{t-\Delta t}z_5\right) = {}^{t}z_6 \tag{4-46}$$

From Taylor's expansion of z_4 we get

$$^{t+\Delta t}z_4 = {}^{t}z_4 + \Delta t\, {}^{t}\dot{z}_4 + \frac{\Delta t^2}{2}\, {}^{t}\ddot{z}_4 + \cdots \tag{4-47}$$

$$^{t}\ddot{z}_4 = \frac{2}{\Delta t^2}\left({}^{t+\Delta t}z_4 - {}^{t}z_4 - \Delta t\, {}^{t}\dot{z}_4\right) \tag{4-48}$$

Substituting value of ${}^{t}\dot{z}_4$ from equation (4-45) into equation (4-48) and rearranging terms, we get

$$^{t}\ddot{z}_4 = \frac{1}{\Delta t^2}\left({}^{t+\Delta t}z_4 - 2\,{}^{t}z_4 + {}^{t-\Delta t}z_4\right) = {}^{t}z_6 \tag{4-49}$$

Similar derivations result in the following expressions

$$^{t}\ddot{z}_1 = \frac{1}{\Delta t^2}\left({}^{t+\Delta t}z_1 - 2\,{}^{t}z_1 + {}^{t-\Delta t}z_1\right) = {}^{t}z_3 \tag{4-50}$$

$$^{t}\ddot{z}_2 = \frac{1}{\Delta t^2}\left({}^{t+\Delta t}z_2 - 2\,{}^{t}z_2 + {}^{t-\Delta t}z_2\right) = {}^{t}z_4 \tag{4-51}$$

$$^{t}\ddot{z}_3 = \frac{1}{\Delta t^2}\left({}^{t+\Delta t}z_3 - 2\,{}^{t}z_3 + {}^{t-\Delta t}z_3\right) = {}^{t}z_5 \tag{4-52}$$

The equilibrium equation for a multi-degree of freedom system is given at time t by

$$[M]\ddot{\underset{\sim}{X}}_t + [C]\dot{\underset{\sim}{X}}_t + [K]\underset{\sim}{X}_t = \underset{\sim}{F}(t) \tag{4-53}$$

Substituting values of $\ddot{\underset{\sim}{X}}_t$, $\dot{\underset{\sim}{X}}_t$, $\underset{\sim}{X}_t$ and [C] into equation (4-53), we get

$$[M]^t\underset{\sim}{z}_6 + [A_o]^t\underset{\sim}{z}_5 - [A_1]^t\underset{\sim}{z}_3 + [A_2]^t\underset{\sim}{z}_1 + [K]^t\underset{\sim}{z}_4 = \underset{\sim}{F}(t) \tag{4-54}$$

where the sign (.) under a letter denotes 'vector.'

Further substitution and rearrangement of terms give the explicit integration scheme

$$\left(\frac{1}{\Delta t^2}\,[M] + \frac{1}{2\Delta t}\,[A_o]\right)\,{}^{t+\Delta t}\underset{\sim}{z}_4 = \underset{\sim}{F}(t) - \left([K] - \frac{2}{\Delta t^2}\,[M]\right)^{t}\underset{\sim}{z}_4$$
$$- \left(\frac{1}{\Delta t^2}\,[M] - \frac{1}{2\Delta t}\,[A_o]\right)\,{}^{t-\Delta t}\underset{\sim}{z}_4 + [A_1]\,{}^{t}\underset{\sim}{z}_3 - [A_2]\,{}^{t}\underset{\sim}{z}_1 \tag{4-55}$$

where:

$$[A_o] = \alpha[K]$$

$$[A_1] = \beta[K] \tag{4-56}$$

$$[A_2] = \gamma[K]$$

Taylor's expansion of $\underset{\sim}{z}_3$, $\underset{\sim}{z}_2$, $\underset{\sim}{z}_1$ results in the following expressions

$${}^{t+\Delta t}\underset{\sim}{z}_3 = {}^{t}\underset{\sim}{z}_3 + \Delta t^{t}\underset{\sim}{z}_4 + \frac{\Delta t^2}{2}\,{}^{t}\underset{\sim}{z}_5 + \cdots \tag{4-57}$$

$${}^{t+\Delta t}\underset{\sim}{z}_2 = {}^{t}\underset{\sim}{z}_2 + \Delta t^{t}\underset{\sim}{z}_3 + \frac{\Delta t^2}{2}\,{}^{t}\underset{\sim}{z}_4 + \cdots \tag{4-58}$$

$${}^{t+\Delta t}\underset{\sim}{z}_1 = {}^{t}\underset{\sim}{z}_1 + \Delta t^{t}\underset{\sim}{z}_2 + \frac{\Delta t^2}{2}\,{}^{t}\underset{\sim}{z}_3 + \cdots \tag{4-59}$$

4.3. Implicit Time Integration Scheme Using Newmark Method

From Newmark's Method we assume

$$ {}^{t+\Delta t}\underset{\sim}{z}_5 = {}^{t}\underset{\sim}{z}_5 + \Delta t\left[(1-\delta)\,{}^{t}\underset{\sim}{z}_6 + \delta\,{}^{t+\Delta t}\underset{\sim}{z}_6\right] \tag{4-60} $$

$$ {}^{t+\Delta t}\underset{\sim}{z}_4 = {}^{t}\underset{\sim}{z}_4 + \Delta t\,{}^{t}\underset{\sim}{z}_5 + \Delta t^2\left[\left(\frac{1}{2}-\lambda\right){}^{t}\underset{\sim}{z}_6 + \lambda\,{}^{t+\Delta t}\underset{\sim}{z}_6\right] \tag{4-61} $$

where δ and λ are integration parameters, and $\underset{\sim}{z}_4$, $\underset{\sim}{z}_5$ and $\underset{\sim}{z}_6$ are the displacement, velocity, and acceleration vectors, respectively. The dynamic equation of motion at time $t + \Delta t$ is given by

$$ [M]\ddot{\underset{\sim}{X}}_{t+\Delta t} + [C]\dot{\underset{\sim}{X}}_{t+\Delta t} + [K]\underset{\sim}{X}_{t+\Delta t} = \underset{\sim}{F}(t+\Delta t) \tag{4-62} $$

or

$$ \begin{aligned} [M]\,{}^{t+\Delta t}\underset{\sim}{z}_6 + [A_0]^{t+\Delta t}\underset{\sim}{z}_5 - [A_1]^{t+\Delta t}\underset{\sim}{z}_3 + [A_2]^{t+\Delta t}\underset{\sim}{z}_1 \\ + [K]^{t+\Delta t}\underset{\sim}{z}_4 = \underset{\sim}{F}(t+\Delta t) \end{aligned} \tag{4-63} $$

From equation (4-61)

$$ {}^{t+\Delta t}\underset{\sim}{Z}_6 = \frac{1}{\lambda\Delta t^2}\left({}^{t+\Delta t}\underset{\sim}{z}_4 - {}^{t}\underset{\sim}{z}_4 - \Delta t\,{}^{t}\underset{\sim}{z}_5\right) - \left(\frac{1}{2\lambda} - 1\right)^{t}\underset{\sim}{z}_6 \tag{4-64} $$

substitute values ${}^{t+\Delta t}\underset{\sim}{z}_6$ from equation (4-64) into equation (4-60) and get

$$ {}^{t+\Delta t}\underset{\sim}{z}_5 = {}^{t}\underset{\sim}{z}_5 + \Delta t(1-\delta)\,{}^{t}\underset{\sim}{z}_6 + \delta\Delta t\,{}^{t+\Delta t}\underset{\sim}{z}_6 \tag{4-65} $$

Substitute values of ${}^{t+\Delta t}\underset{\sim}{z}_5$ and ${}^{t+\Delta t}\underset{\sim}{z}_6$ from equations (4-65) and (4-64) into equation (4-63) and rearrange terms and get

$$
\begin{aligned}
&\left(\frac{1}{\lambda\Delta t^2}[M] + \frac{\delta}{\lambda\Delta t}[A_o] + [K]\right)^{t+\Delta t} \underset{\sim}{z}_4 \\
&\quad = \underset{\sim}{F}(t+\Delta t) + [M]\left(\frac{1}{\lambda\Delta t^2}\,{}^t\underset{\sim}{z}_4 + \frac{1}{\lambda\Delta t}\,{}^t\underset{\sim}{z}_5 + \left(\frac{1}{2\lambda} - 1\right){}^t\underset{\sim}{z}_6\right) \\
&\quad + [A_o]\left(\frac{\delta}{\lambda\Delta t}\,{}^t\underset{\sim}{z}_4 + \left(\frac{\delta}{\lambda} - 1\right){}^t\underset{\sim}{z}_5 + \frac{\Delta t}{2}\left(\frac{\delta}{\lambda} - 2\right){}^t\underset{\sim}{z}_6\right) \\
&\quad + [A_1]^{t+\Delta t}\underset{\sim}{z}_3 - [A_2]^{t+\Delta t}\underset{\sim}{z}_1
\end{aligned}
\tag{4-66}
$$

where again

$$[A_o] = \alpha[K]$$

$$[A_1] = \beta[K]$$

$$[A_2] = \gamma[K]$$

α,β,γ are constants from Q-model.

4.4 The Numerical Algorithm

1. Determine stress-strain relation of material using any appropriate elastoplastic material model.

2. Obtain elasto-plastic constitutive matrix $[D^{ep}]$.

3. Obtain elasto-plastic stiffness matrix using the relation

$$[K] = \int_v [B]^T [D^{ep}] [B]\, dv$$

where [B] is the strain displacement matrix.

4. Solve the dynamic equation of motion

$$[M]\ddot{\underset{\sim}{X}} + [C]\dot{\underset{\sim}{X}} + [K]\underset{\sim}{X} = \underset{\sim}{F}(t)$$

where:

$$[C]\,\dot{\underset{\sim}{X}} = [A_o]\,\dot{\underset{\sim}{X}}(\tau) - [A_1] \int_o^t \underset{\sim}{X}(\tau)\, d\tau + [A_2] \int_o^t\int_o^\tau\int_o^\tau \underset{\sim}{X}(\tau) d\tau d\tau d\tau$$

$$[A_o] = \alpha[K]$$

$$[A_1] = \beta[K]$$

$$[A_2] = \gamma[K]$$

α, β, γ are constants from Q-model.

4.4.1 Explicit Time Integration Scheme

For explicit Time Integration Scheme using the Central Difference Method, the dynamic equation of motion at time t is solved as follows:

1. Select time step Δt, $\Delta t < \Delta t_{cr}$

2. Compute integration constants

$$\tau_o = \frac{1}{\Delta t^2}$$

$$\tau_1 = \frac{1}{2\Delta t}$$

$$\tau_2 = 2\tau_o$$

$$\tau_3 = \frac{1}{\tau_2}$$

3. Initialize $\underset{\sim}{z}_1$, $\underset{\sim}{z}_2$, $\underset{\sim}{z}_3$, $\underset{\sim}{z}_4$, $\underset{\sim}{z}_5$, $\underset{\sim}{z}_6$ and calculate

$$^{-\Delta t}\underset{\sim}{z}_4 = {}^{0}\underset{\sim}{z}_4 - \Delta t \; {}^{0}\dot{\underset{\sim}{z}}_4 + \tau_3 \; {}^{0}\ddot{\underset{\sim}{z}}_4$$

4. Obtain effective mass matrix

$$[\hat{M}] = \tau_o \; [M] + \tau_1 [A_o]$$

5. Triangularlize $[\hat{M}]$: $\hat{M} = [L]\,[D]\,[L]^T$

6. For each time step:

a. Calculate effective force at time t:

$$\hat{\underset{\sim}{F}}(t) = \underset{\sim}{F}(t) - ([K] - \tau_2 [M]) \; {}^{t}\underset{\sim}{z}_4 - (\tau_o [M] - \tau_1 [A_o]) \; {}^{t-\Delta t}\underset{\sim}{z}_4$$
$$+ [A_1] \; {}^{t}\underset{\sim}{z}_3 - [A_2] \; {}^{t}\underset{\sim}{z}_3$$

b. Solve for displacements at time $t + \Delta t$

$$[L]\,[D]\,[L]^T \; {}^{t+\Delta t}\underset{\sim}{z}_4 = \hat{\underset{\sim}{F}}(t)$$

c. Solve for $\underset{\sim}{z}_1$, $\underset{\sim}{z}_2$, $\underset{\sim}{z}_3$ at time $t + \Delta t$ as follows

$$ {}^{t+\Delta t}\underset{\sim}{z}_1 = {}^{t}\underset{\sim}{z}_1 + \Delta t\, {}^{t}\underset{\sim}{z}_2 + \frac{1}{\tau_2}\, {}^{t}\underset{\sim}{z}_3 $$

$$ {}^{t+\Delta t}\underset{\sim}{z}_2 = {}^{t}\underset{\sim}{z}_2 + \Delta t\, {}^{t}\underset{\sim}{z}_3 + \frac{1}{\tau_2}\, {}^{t}\underset{\sim}{z}_4 $$

$$ {}^{t+\Delta t}\underset{\sim}{z}_3 = {}^{t}\underset{\sim}{z}_3 + \Delta t\, {}^{t}\underset{\sim}{z}_4 + \frac{1}{\tau_2}\, {}^{t}\underset{\sim}{z}_5 $$

also

$$ {}^{t}\underset{\sim}{z}_5 = \tau_1 \left(- {}^{t-\Delta t}\underset{\sim}{z}_4 + {}^{t+\Delta t}\underset{\sim}{z}_4 \right) $$

$$ {}^{t}\underset{\sim}{z}_6 = \tau_o \left({}^{t-\Delta t}\underset{\sim}{z}_4 - 2\, {}^{t}\underset{\sim}{z}_4 + {}^{t+\Delta t}\underset{\sim}{z}_4 \right) $$

4.4.2 Implicit Time Integration Scheme

For Implicit Time Integration scheme using Newmark's method, the dynamic equation of motion at time $t + \Delta t$ is solved as follows:

1. Select time step Δt
2. Compute integration parameters $\delta \geq 0.5$ $\lambda \geq 0.25\,(0.5 + \delta)^2$

$$ \tau_o = \frac{1}{\lambda \Delta t^2} $$

$$ \tau_1 = \frac{\delta}{\lambda \Delta t} $$

$$\tau_2 = \frac{1}{\lambda \Delta t}$$

$$\tau_3 = \frac{1}{2\lambda} - 1$$

$$\tau_4 = \frac{\delta}{\lambda} - 1$$

$$\tau_5 = \frac{\Delta t}{2}\left(\frac{\delta}{\lambda} - 2\right)$$

$$\tau_6 = \Delta t(1 - \delta)$$

$$\tau_7 = \delta \Delta t$$

3. Initialize $\underset{\sim}{z}_1, \underset{\sim}{z}_2, \underset{\sim}{z}_3, \underset{\sim}{z}_4, \underset{\sim}{z}_5, \underset{\sim}{z}_6$

4. Form effective stiffness matrix

$$[\hat{K}] = [K] + \tau_o [M] + \tau_1 [A_o]$$

5. Triangularize $[\hat{K}]$: $[\hat{K}] = [L][D][L]^T$

6. For each time step

   a. Compute $\underset{\sim}{z}_1(t + \Delta t)$, $\underset{\sim}{z}_2(t + \Delta t)$, $\underset{\sim}{z}_3(t + \Delta t)$ from Taylor's expansion as follows

$$^{t+\Delta t}\underset{\sim}{z}_1 = {}^{t}\underset{\sim}{z}_1 + \Delta t\, {}^{t}\underset{\sim}{z}_2 + \frac{\Delta t^2}{2}\, {}^{t}\underset{\sim}{z}_3$$

$$^{t+\Delta t}\underset{\sim}{z}_2 = {}^{t}\underset{\sim}{z}_2 + \Delta t\, {}^{t}\underset{\sim}{z}_3 + \frac{\Delta t^2}{2}\, {}^{t}\underset{\sim}{z}_4$$

$$^{t+\Delta t}\underset{\sim}{z}_3 = {}^{t}\underset{\sim}{z}_3 + \Delta t\, {}^{t}\underset{\sim}{z}_4 + \frac{\Delta t^2}{2}\, {}^{t}\underset{\sim}{z}_5$$

b. Compute effective loads at time $t + \Delta t$

$$\hat{\underset{\sim}{F}}(t + \Delta t) = \underset{\sim}{F}(t + \Delta t) + [M] \left(\tau_o{}^t\underset{\sim}{z}_4 + \tau_2{}^t\underset{\sim}{z}_5 + \tau_3{}^t z_6\right)$$
$$+ [A_o] \left(\tau_1{}^t\underset{\sim}{z}_4 + \tau_4{}^t\underset{\sim}{z}_5 + \tau_5{}^t\underset{\sim}{z}_6\right) + [A_1]^{t+\Delta t}\underset{\sim}{z}_3 - [A_2]^{t+\Delta t}\underset{\sim}{z}_1$$

c. Solve for displacements at time $t + \Delta t$:

$$[L]\,[D]\,[L]^T\ {}^{t+\Delta t}\underset{\sim}{z}_4 = \hat{\underset{\sim}{F}}(t + \Delta t)$$

d. Calculate accelerations and velocities at time $t + \Delta t$:

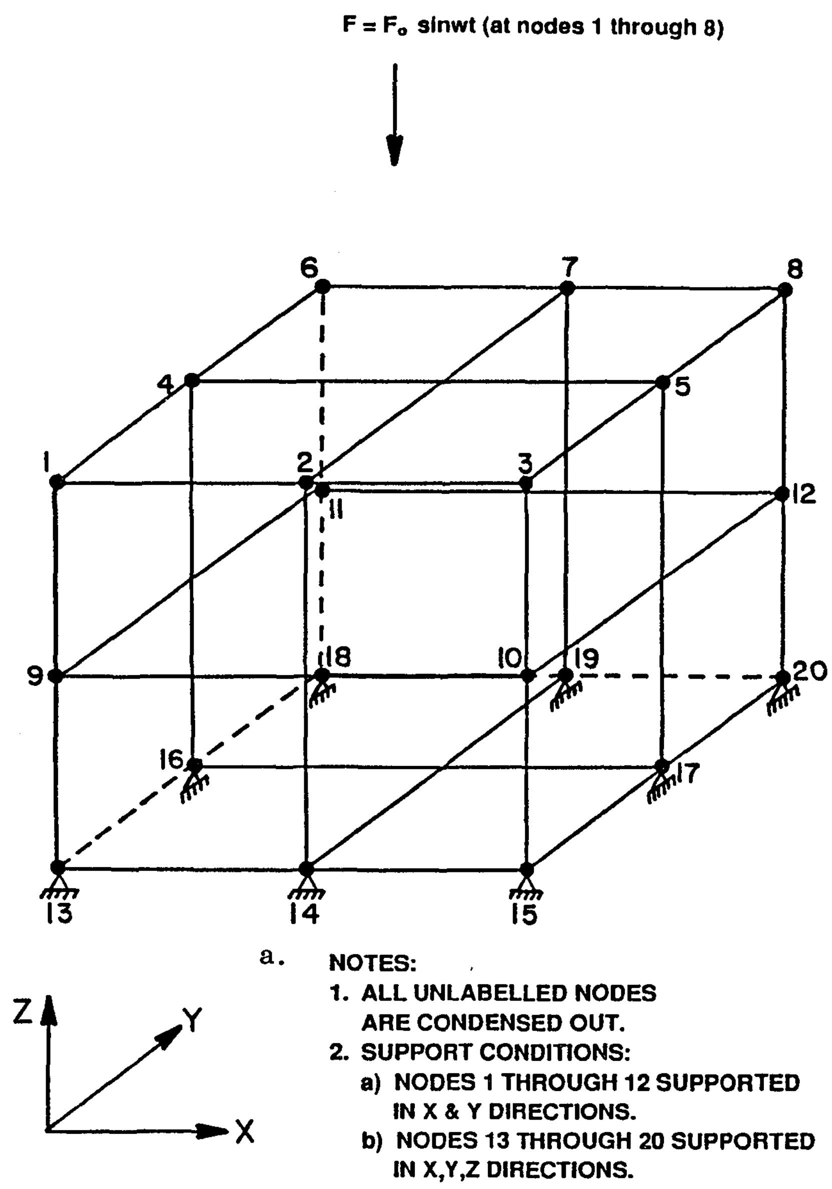
$${}^{t+\Delta t}\underset{\sim}{z}_6 = \tau_o \left({}^{t+\Delta t}\underset{\sim}{z}_4 - {}^t\underset{\sim}{z}_4\right) - \tau_2{}^t\underset{\sim}{z}_5 - \tau_3{}^t\underset{\sim}{z}_6$$
$${}^{t+\Delta t}\underset{\sim}{z}_5 = {}^t\underset{\sim}{z}_5 + \tau_6{}^t\underset{\sim}{z}_6 + \tau_7{}^{t+\Delta t}\underset{\sim}{z}_6$$

4.5 Numerical Example

The three-dimensional problem shown in Fig. 4-2A is used to test the new proposed model. The system consists of a medium discretized into three-dimensional Lagrangian elements whose interior nodes have been condensed out in the stiffness matrix formulation by the substructure technique (16). The surface nodes were subjected to a sinusoidal excitation as shown in Fig. 4-2A The support conditions are also indicated in the figure. For clarity, only the displacement history in the direction of excitation (Z direction) for four nodes (nodes 1,2,3,4) on the surface were plotted. The displacement history at these nodes are sufficient to display the essential features of the proposed model. The cap model (1) was utilized to generate stress-strain relations from which an elastoplastic constitutive matrix was derived (see Section 3).

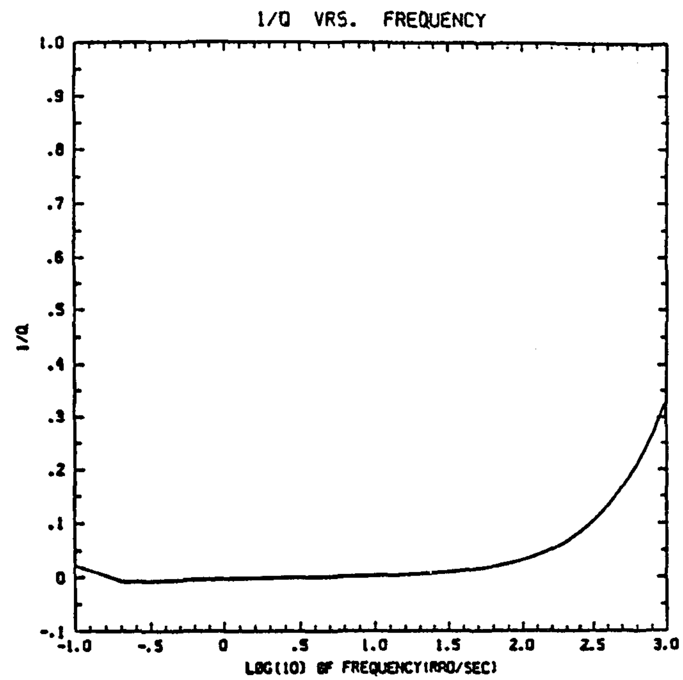
Four different tests were performed:

Test 1. This test was performed to investigate the effect of Q when the proposed viscoplastic model is adapted to predict responses in viscoelastic media. The elastoplastic constitutive matrix



a.

NOTES:
1. ALL UNLABELLED NODES ARE CONDENSED OUT.
2. SUPPORT CONDITIONS:
   a) NODES 1 THROUGH 12 SUPPORTED IN X & Y DIRECTIONS.
   b) NODES 13 THROUGH 20 SUPPORTED IN X,Y,Z DIRECTIONS.



b. Q = 30

FIGURE 4-2 Test Problem

was replaced by an elastic constitutive matrix in the stiffness matrix formulation. For a constant time step $\Delta t/T$ of 0.2 the value of the constant Q was varied from 10, 20, 30, 40, 50, 100, 500, 1000, using 200 time steps in the implicit time integration algorithm. The displacement history for the various values of Q are shown in Figure 4-3.

Test 2. For a constant time step $\Delta t/T = 0.1$ the effect of Q on displacement was observed for 200 time steps using the outlined implicit time integration algorithm. Here T denotes the fundamental period of the system under investigation. The value of Q was varied from 10, 20, 30, 40, 50, 100, 500, to 1000. The displacement history due to the applied sinusoidal excitation is shown in Figure 4-4.

Test 3. For a constant Q value of 30 the model was tested for convergence by varying the time step $\Delta t/T$ as follows: $\Delta t/T = 0.1, 0.08, 0.05, 0.025$ for 200 time steps using the proposed implicit time integration algorithm. The displacement history due to the applied sinusoidal excitation is shown in Figure 4-5.

Test 4. Test 4 was performed to investigate the stability of the model using the proposed explicit time integration algorithm. Here for a constant Q value of 30, the time step Δt was varied from $\Delta t/T = 0.01$ to 0.005 and the corresponding displacement history observed (Figure 4-6).

4.6 Discussions

From Figure 4-5, it is observed that for the implicit time integration scheme using Newmark's method, the model is stable for $\Delta t/T \leq 0.1$, where T is the fundamental period of the system. It is also noted that no further accuracy of the system response is achieved by using smaller time steps. On the other hand, for the explicit time integration scheme using the Central Difference scheme, a much smaller time step of $\Delta t/T \leq 0.01$ is required for stability (Figure 4-6). Clearly, the implicit time integration scheme is by far superior to the explicit time integration scheme in terms of savings in computer time.

The effect of the value of Q on the displacements history is clearly shown in Figs. 4-4. It is observed that larger values of Q result in less damping, hence larger displacements, and vice versa. The choice of an appropriate value of Q for a particular material or medium is based on dynamic tests which can readily be performed in the laboratory. A Q value of 35 has been observed on laboratory tests performed on Ottawa sand in Ref. 29.

Fig. 4-2B shows a plot of 1/Q versus logarithm to base 10 of frequency for a constant Q value of 30.

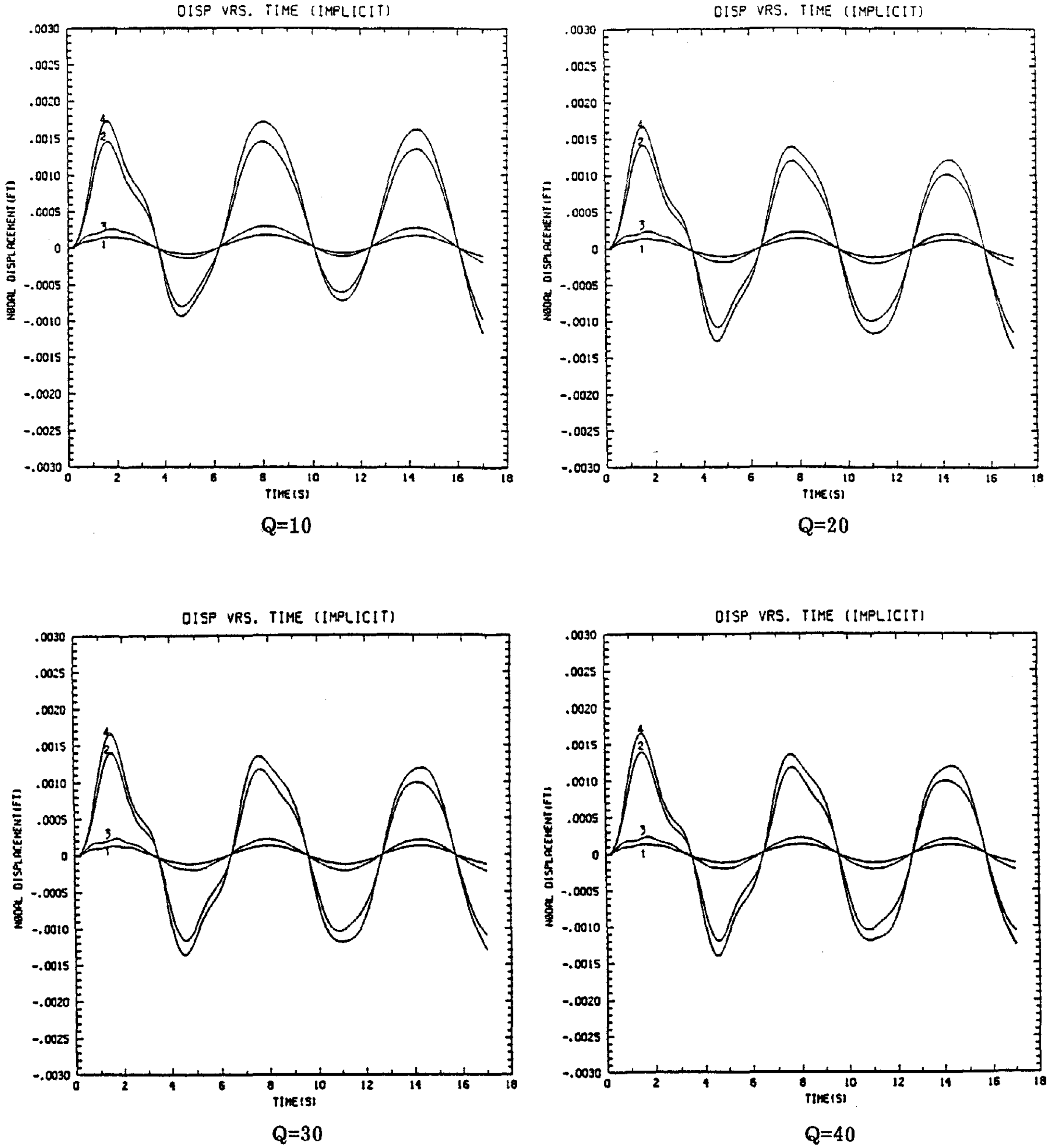
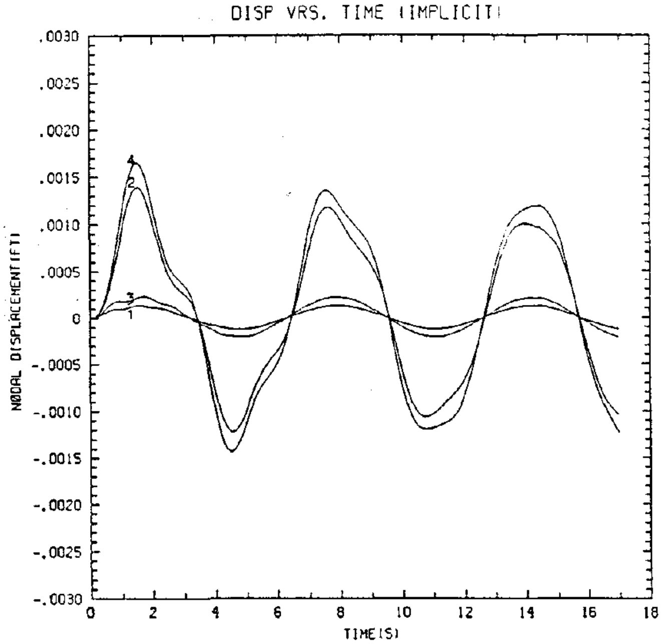
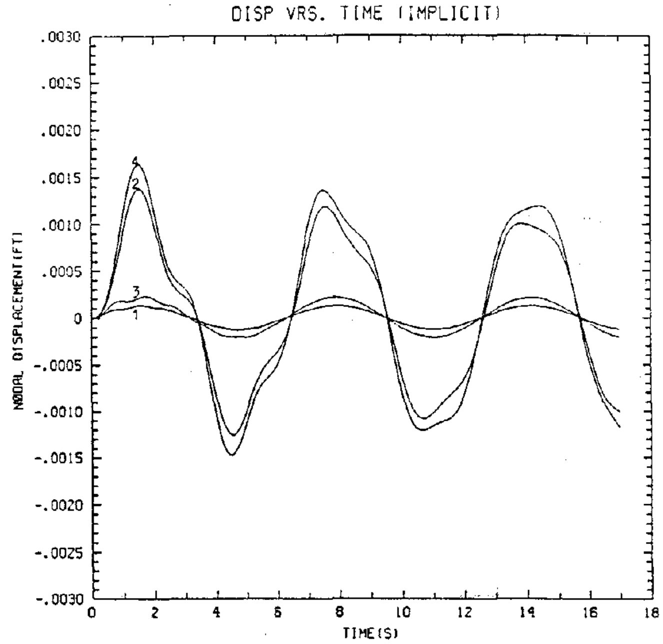


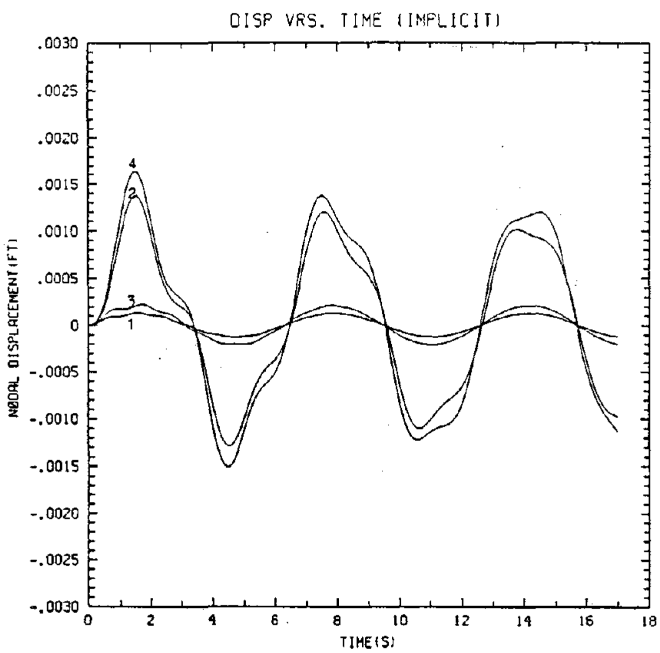
FIGURE 4-3 Test 1: Displacement History for Various Values of Q and a constant time step $\Delta t/T = 0.2$.



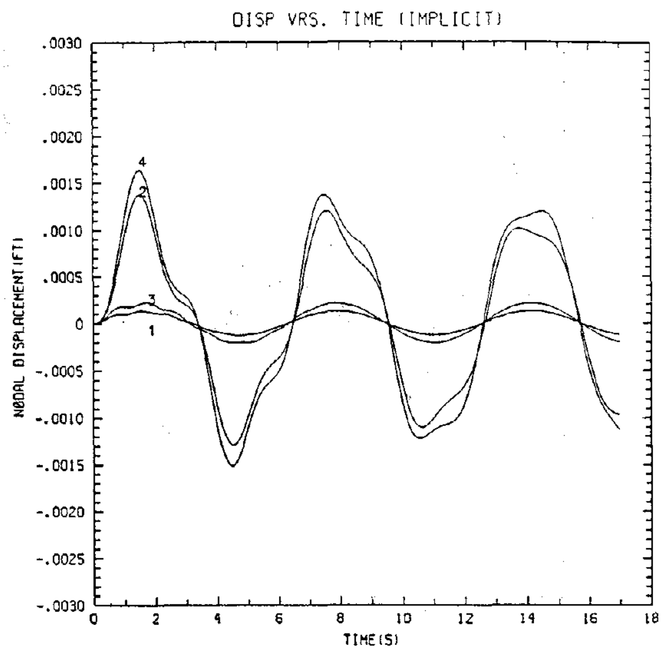
Q=50



Q=100

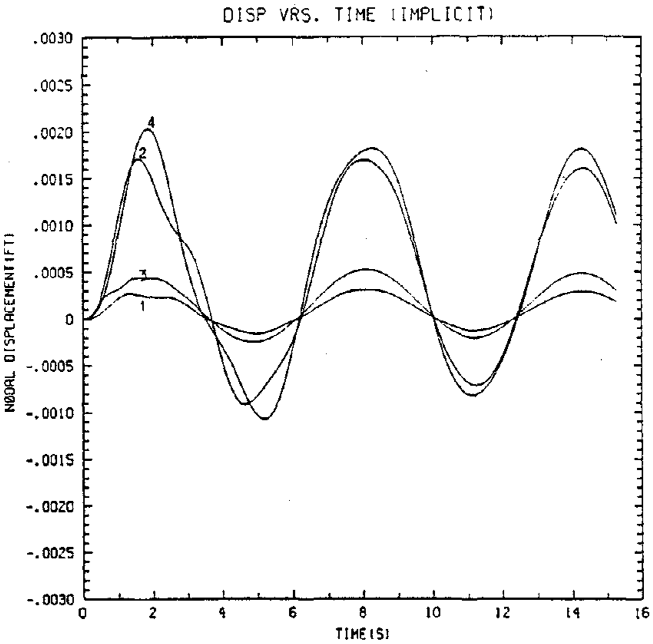


Q=500

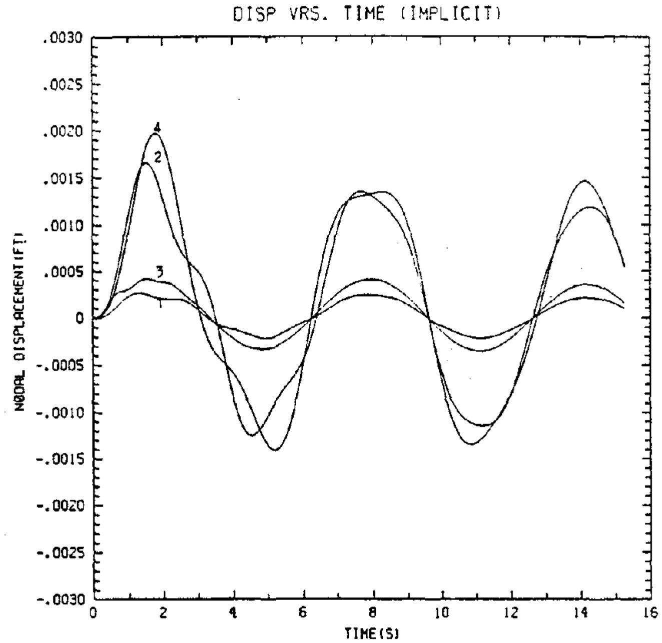


Q=1000

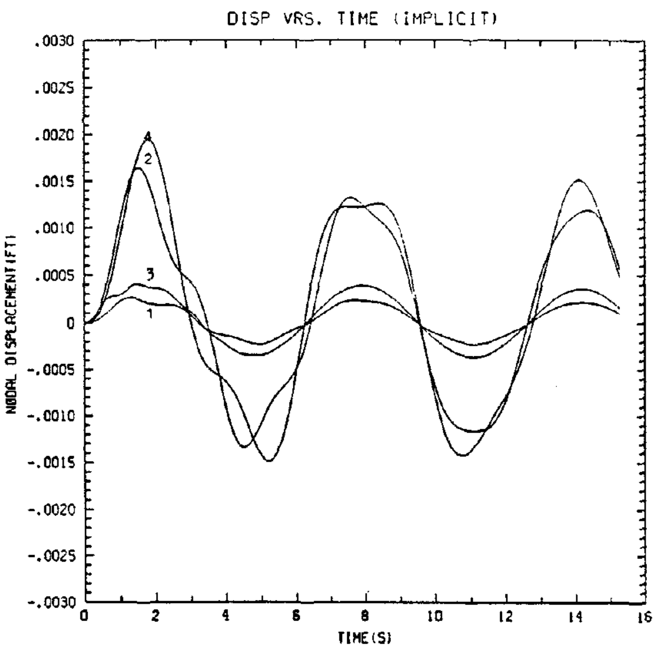
FIGURE 4-3 Test 1: Displacement History for Various Values of Q and a constant time step $\Delta t/T = 0.2$. (Cont'd)



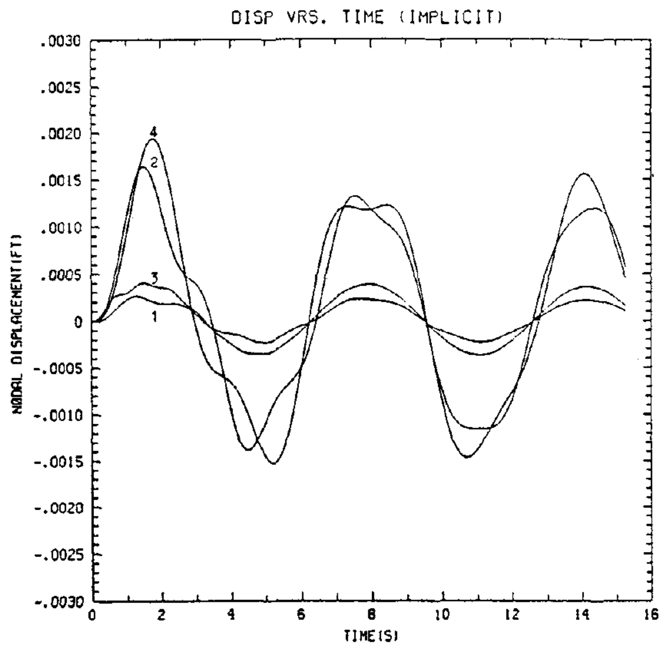
Q=10



Q=20

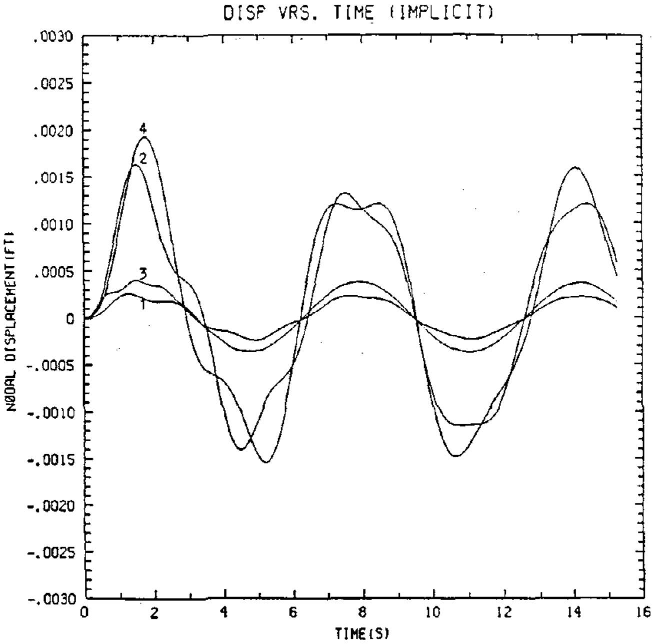


Q=30

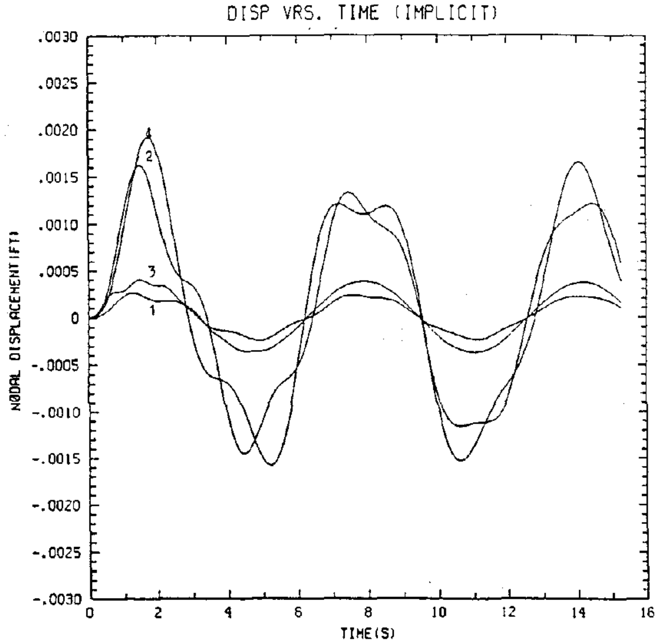


Q=40

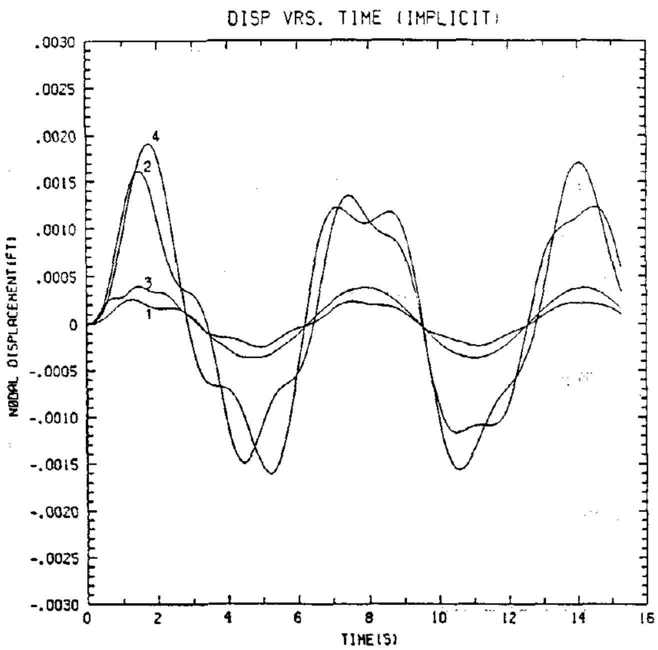
FIGURE 4-4 Test 2: Displacement History Due to Applied Sinusoidal Excitation for a constant time step $\Delta t/T = 0.1$



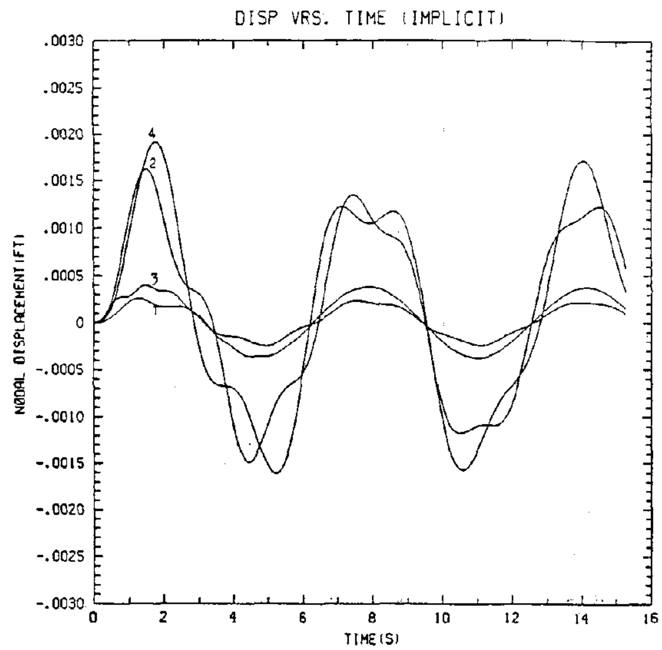
Q=50



Q=100

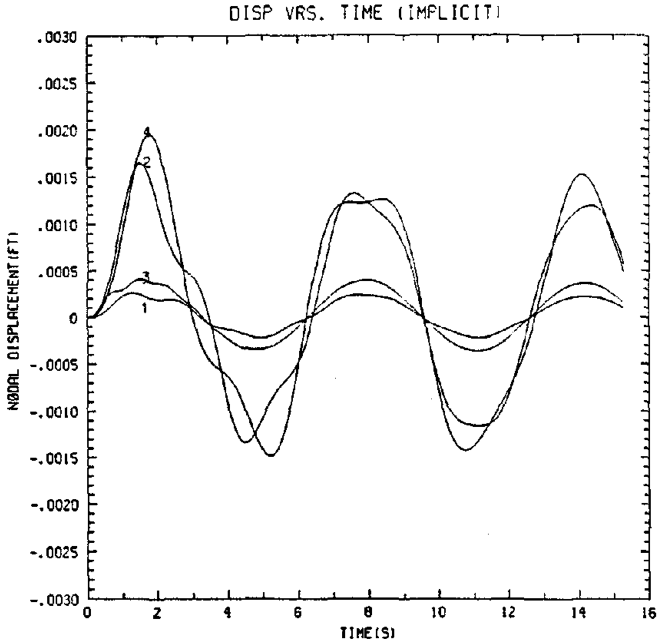


Q=500

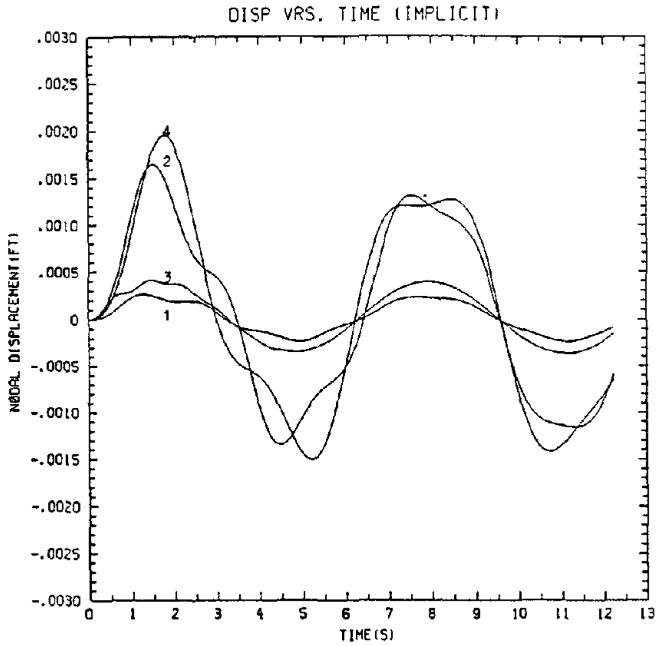


Q=1000

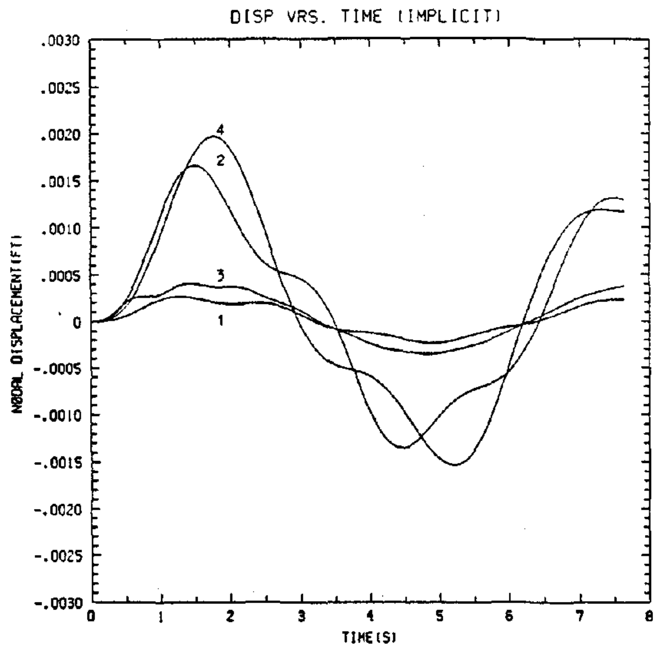
FIGURE 4-4 Test 2: Displacement History Due to Applied Sinusoidal Excitation for a constant time step $\Delta t/T = 0.1$. (Cont'd)



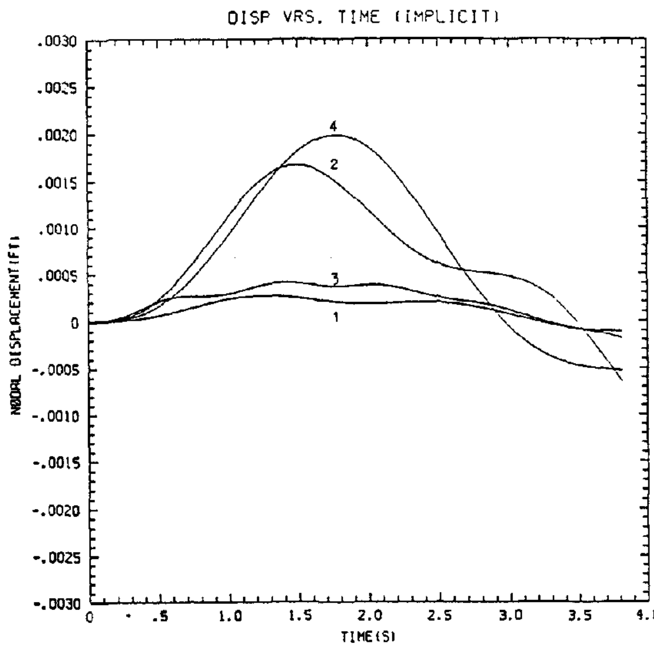
$\Delta t/T=0.1$



$\Delta t/T=0.08$

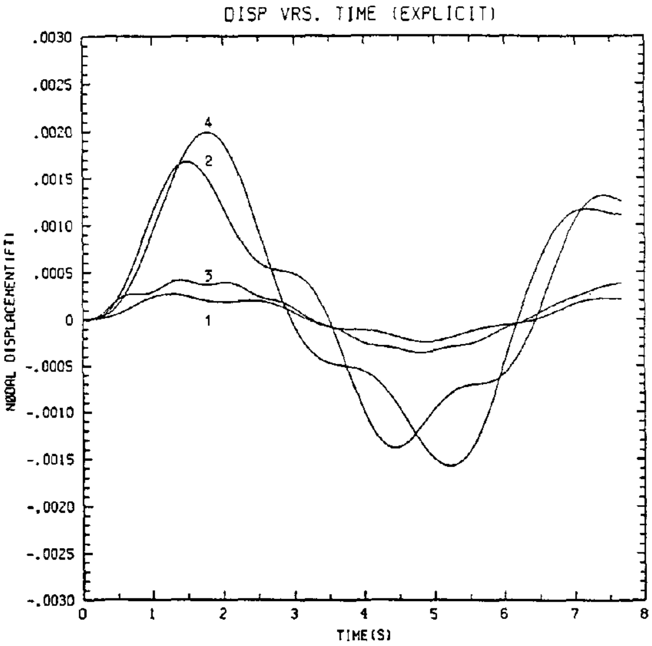


$\Delta t/T=0.05$

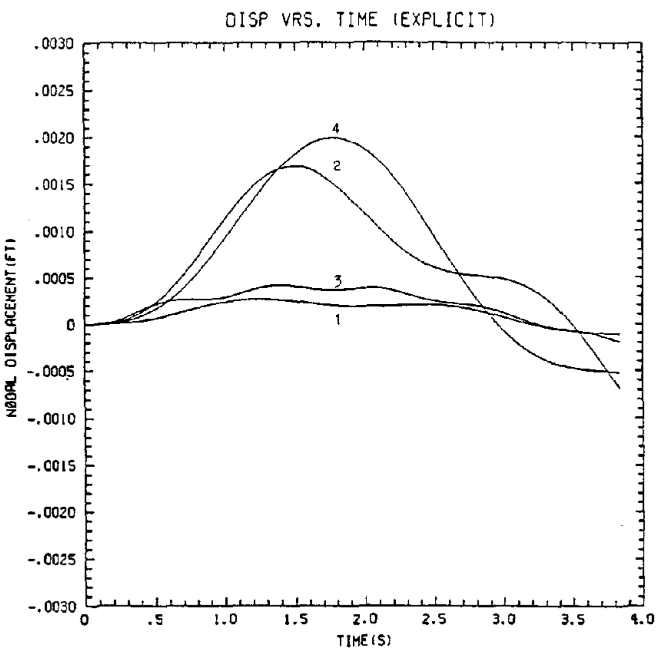


$\Delta t/T=0.025$

FIGURE 4-5 Test 3: Displacement History Due to Applied Sinusoidal Excitation for a constant Q value of 30.



Δ t/T=0.01



Δt/T=0.005

FIGURE 4-6 Test 4: Displacement History for a constant Q model of 30 with the time step Δt varied from $\Delta t/T = 0.01$ to 0.005.

SECTION 5
CONCLUSION

The proposed Q-model provides a means to conveniently establish a viscoplastic model which greatly enhances the analyses of earthquake, shock, blast and other vibration excitations often encountered in soil-structure interaction problems.

The frequency dependent $Q^{-1}(\omega)$ is expanded into a Laurent Series which generates a set of damping coefficients. For the special case when Q is constant over a prescribed frequency range, such damping coefficients are readily determined through minimization of the mean square error over the prescribed frequency range.

Numerical tests performed with Q constant over a frequency range of 0.1 to 10 radians/second show the model to be stable and accurate, provided the time step Δt is sufficiently small.

SECTION 6
REFERENCES

1. Sandler, I.S. and Rubin D., "An Algorithm and a Modular Subroutine for the Cap Model." International Journal for Numerical and Analytical Methods in Geomechanics, Vol 3, 1979, pp. 173-186.

2. Liu, H.-P., Anderson, D.L., and Kanamori, H., "Velocity Dispersion due to Anelasticity; Implications for Seismology and Mantle Composition." Geophys. J. Roy. astr. Soc (1976), Vol 47, pp. 41-58.

3. Kjartansson, E., "Constant Q-Wave Propagation and Attenuation." Journal of Geophysical Research, Vol 84, No. B9, Aug. 10, 1979, pp. 4737-4748.

4. Strick, E., "The Determination of Q, Dynamic Viscosity and Transient Creep Curves from Wave Propagation Measurements." Geophys. J. Roy. astr. Soc (1967), Vol. 13, pp. 197-218.

5. Naghdi, P.M. and Murch, S.A. "On the Mechanical Behavior of Viscoelastic/Plastic Solids. Journal Of Applied Mechanics, ASME, pp. 321-328, Sept. 1963.

6. Perzyna, R., "The Constitutive Equations for Work-Hardening and Rate Sensitive Plastic Materials." Proc. of Vibration Problems, Vols. 3,4, pp. 281-290, 1963 (in English).

7. Drucker, D.C., "On Uniqueness in the Theory of Plasticity." Quarterly of Applied Math, Vol 14, 1956, pp. 35-42.

8. Drucker, D.C., Gibson, R.E., and Henkel, D.J., "Soil Mechanics and Work-Hardening Theories of Plasticity." Transactions, ASCE, Vol. 122, 1957, pp. 338-346.

9. Drucker, D.C., and Palgen, L., "On Stress-Strain Relations Suitable for Cyclic and Other Loading." Journal of Applied Mechanics, Vol. 48, pp. 479-485 (1981).

10. Drucker, D.C. and Prager, W., "Soil Mechanics and Plasticity Analysis or Limit Design." Quarterly of Applied Mathematics, Vol. 10, 1952, pp. 157-175.

11. Sandler, I.S., DiMaggio, F.L., and Baron, M.L., "An Extension of the Cap Model-Inclusion of Pore Pressure Effects and Kinematic Hardening to Represent an Anisotropic Wet Clay." Mechanics of Engineering Materials, eds., Desai, C.S. and Gallagher, R.H., Chap. 28, John Wiley & Sons, Ltd. 1984.

12. Rosenblatt M., "A Set of Constitutive Equations for Rocks and Soils." Communication at D.A.S.A. Materials Working Group Meeting, 1970.

13. Reiner, M., "Plastic Yielding in Anelasticity." Journal of the Mechanics and Physics of Solids, Vol. 8, 1960 pp. 255-261.

14. Schofield, A.N., and Wroth, P., Critical State Soil Mechanics. McGraw-Hill, Ltd. 1968.

15. Norwich, A.S., and Berry, B.S., Anelastic Relaxation in Crystalline Solids. Academic Press, N.Y., 1972.

16. Zienkiewicz, O.C., The Finite Element Method, 3rd ed., McGraw - Hill, N.Y., 1977.

17. Bathe, K.J. and Wilson, E.L. Numerical Methods in Finite Element Analysis. Prentice-Hall, Englewood Cliffs, N.J. 1976.

18. Cristescu, N. and Suliciu, I., Viscoplasticity, Martinus Nijhoff Publishers, Boston 1982.

19. Aboim, C.A., and Roth, W.H., "Bounding-Surface-Plasticity Theory Applied to Cyclic Loading of Sand." International Symposium on Numerical Models in Geomechanics, Zurich, Sept. 1982, eds., Dungar, R., Pande, G.N., Studer, J.A.

20. Dafalias, Y.F., "An Elastoplastic -Viscoplastic Constitutive Modelling of Cohesive Soils." International Symposium on Numerical Models in Geomechanics, Zurich, 9/1982, eds. Dungar, R., Pande, G.N., Studer, J.A.

21. Valanis, K.C. and Read, A., "A New Endochronic Plasticity Theory for Soils." Systems, Science and Software Report SSS-R-80-4294 (1979).

22. Kondner, R.L., "Hyperbolic Stress-Strain Response of Soils" Proceedings of Soil Mechanics and Foundation Engineering, Division of ASCE, Vol. 89, 1963.

23. Dasgupta, G., "A Numerical Solution for Viscoelastic Half Planes." Journal of the Engineering Mechanics Division, ASCE, Vol. 102, No. EM4, Aug. 1976, pp. 601-612.

24. Dasgupta, G. and Chopra, A.K., "Dynamic Stiffness Matrix for Viscoelastic Half Planes," Journal of the Engineering Mechanics Division, ASCE, Vol. 105, No. EM5, Oct. 1979, pp. 729-745.

25. Zienkiewicz, O.C. and Cormeau, I.C., "Visco-plasticity Plasticity and Creep in Elastic Solids - A Unified Numerical Solution Approach." International Journal for Numerical Methods in Engineering, Vol. 8, pp. 821-84b (1974).

26. Cormeau, I., "Numerical Stability in Quasi-static Elasto-Viscoplasticity." International Journal for Numerical Method in Engineering, Vol. 9, pp. 109-127 (1975).

27. DiMaggio, F.L. and Sandler, I.S., "Material Model for Granular Soils." Journal of the Engineering Mechanics Division, ASCE, Vol. 97, No. EM3, Proc. Paper 8212, June 1971, pp. 935-950.

28. Day, S.M. and Minster, B.J., "Numerical Simulation of Attenuated Wavefields using Pade Approximant Method." Geophys. J. Roy. Astr. Soc. (1984), Vol. 78, pp. 105-118.

29. Richart, F.E., Woods, R.D. and Hall, J.R., Jr., "Vibrations of Soils and Foundations." Prentice-Hall, Inc., Englewood Cliffs, N.J., 1970, pp. 162-167.

30. Abramowitz, M. and Stegun, I.A., Handbook of Mathematical Functions. Dover Publications, Inc., N.Y., 1970.

C O M P U T E R

P R O G R A M

The Computer Program follows the following steps:

1. Evaluation of Stress-Strain Relation of Material Using an Appropriate Material Model. In the test problem the cap model(1) was used. The routine STRESS generates the stress-strain relation.

2. Evaluation of the 3-D Material Constitutive Matrix. The derivation of the elastoplastic constitutive matrix is shown in section 3 of the text. The subroutine DMATRX evaluates either elastic or elastoplastic constitutive matrix by setting the counter MMODE to 1 or 3 respectively.

3. Evaluation of 3-D Stiffness Matrix. The subroutine STIFF3D evaluates the 3-D stiffness matrix from a user supplied nodal coordinates of a discretized medium. In the test problem the medium was discretized into 3-D Lagrangian elements with three nodes in X,Y,Z directions - a total of 27 nodes (fig.13a). The interior nodes were then condensed out by the sub structure techique. Integration for stiffness matrix was performed over three Gauss points User has the option to supply the stiffness matrix, in which case steps 1, 2, and 3 can be omitted.

4. Evaluation of Mass Matrix. The subroutine GMASS evaluates the global mass matrix of the discretized medium from user supplied nodal coordinates. User has the option to supply the mass matrix.

5. Evaluation of Damping Coefficients. Damping coefficients are evaluated from the Q-Model by the subroutine QMOD. This routine is called by EXPLCT and IMPLCT for the evaluation of displacement history.

6. Evaluation of Response History. Two integration schemes have been presented. The subroutines EXPLCT and IMPLCT evaluate the response history of the system by the explicit and implicit direct step by step time integration schemes respectively.

U S E R   S U P P L I E D   I N P U T

1. STIFFNESS AND MASS MATRIX EVALUATION.

```
STIFF----------Stiffness Matrix of discretized medium
               (Elastic or Elastoplastic).
XMASS----------Mass Matrix of discretized medium
DM-------------Material Constitutive Matrix. This is
```

supplied only when STIFF is not supplied by user.

XXC,YYC,ZZC----Nodal Coordinates of spatially discretized medium. Omit when STIFF and XMASS are supplied by user.

NCNSTR,JBNDRY--Number of Constraints in the spatially discretized medium. Omit when STIFF and XMASS are supplied by user.

NGPS-----------Number of Gauss points required for the formulation of the stiffness and mass matrices. Omit when STIFF and XMASS are user supplied.

NODE-----------Total number of nodes in the spatially discretized medium.

NODB3----------3*NODE

NDOF-----------Number of degrees of freedom in the discretized medium.

MX,MY,MZ-------Number of nodes in the X,Y,Z directions respectively for the Lagrangian element.

2. EVALUATION OF DAMPING COEFFICIENTS FROM THE Q-MODEL.

Q--------------Specific Dissipation Factor of material

OMEG1,OMEG2----Lower and Upper Frequency Limits for which Q is constant.

3. EVALUATION OF DYNAMIC RESPONSE.

CONST1,CONST2--Ratio of time step and fundamental period of the system for the explicit and implicit time integration schemes respectively. These constants help to select appropriate time steps for stability and accuracy.

ICOUNT---------A Counter which indicates whether explicit or implicit time integration scheme is used.
ICOUNT=0 EXPLICIT
ICOUNT=1 IMPLICIT

MTIME1,MTIME2--Number of time steps required for the system response for the explicit and implicit time integration schemes respectively.

SAMPLE INPUT

```
.250 .65 .18 .67 2.5 .066 66.67 40.0 1 .3 .4903 .25      001
0.  0.  0.                                               002
1.  0.  0.                                               003
2.  0.  0.                                               004
```

```
0.   1.   0.                                                    005
2.   1.   0.                                                    006
0.   2.   0.                                                    007
1.   2.   0.                                                    008
2.   2.   0.                                                    009
0.   0.   1.                                                    010
2.   0.   1.                                                    011
0.   2.   1.                                                    012
2.   2.   1.                                                    013
0.   0.   2.                                                    014
1.   0.   2.                                                    015
2.   0.   2.                                                    016
0.   1.   2.                                                    017
2.   1.   2.                                                    018
0.   2.   2.                                                    019
1.   2.   2.                                                    020
2.   2.   2.                                                    021
1.   1.   0.                                                    022
1.   0.   1.                                                    023
2.   1.   1.                                                    024
1.   2.   1.                                                    025
0.   1.   1.                                                    026
1.   1.   2.                                                    027
1.   1.   1.                                                    028
6                                                               029
69 1 .005 1000 .2 200 3                                         030
30. .1 10.                                                      031
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18                    032
19 20 21 22 23 24 25 26 28 29 31 32 34 35 37                    033
38 40 41 43 44 46 47 49 50 52 53 55 56 58 59                    034
61 62 63 64 65 66 67 68 69 70 71 72 73 74 75                    035
76 77 78 79 80 81                                               036
```

LINE BY LINE DESCRIPTION OF INPUT:

Line 1: Soil properties and cap constants for the Cap Model used to generate the stress-strain relations. In order these are:
A, B, C, D, R, W, BULK(Bulk Modulus), SHEAR(Shear Modulus),MAT, TCUT, FCUT, YNU(Poisson's Ratio).
The above are cap constants except as noted.

Line 2-28: X, Y, Z coordinates of discretized medium.

Line 29: MAXITR- the number of iterations required to generate the stress-strain relation using the Cap Model.

Line 30: NCNSTR, ICOUNT, CONST1, MTIME1, CONST2, MTIME2, NGPS

Line 31: Q, OMEG1, OMEG2

Line 32-36: JBNDRY

```
C***************************************************************
C                                                              *
C                                                              *
C                                                              *
C        THIS PROGRAM COMPUTES VISCOPLASTIC CONSTITUTIVE       *
C        RELATIONS.                                            *
C        THE PROGRAM COMBINES A CAP MODEL FOR THE              *
C        ELASTO-PLASTIC PART, AND A  Q-MODEL FOR THE           *
C        VISCOUS ( DAMPING ) PART TO GENERATE A                *
C        VISCOPLASTIC MODEL                                    *
C                                                              *
C        THE PROGRAM UTILIZES A 3-D FINITE ELEMENT             *
C        SPATIAL DISCRETIZATION FORMULATION.                   *
C                                                              *
C                                                              *
C                                                              *
C***************************************************************
C
C
C
C
C
C
C         D E F I N I T I O N   O F   V A R I A B L E S
C        ________________________________________________
C
C
C   CONST1,CONST2--THE RATIO OF TIME STEP AND FUNDAMENTAL PERIOD OF
C                  THE SYSTEM IN THE DYNAMIC SAMPLE PROBLEM FOR
C                  EXPLICIT AND IMPLICIT TIME INTEGRATION RESP.
C   DFDSIG---------FIRST DERIVATIVE OF THE YIELD FUNCTION WITH RESPECT
C                  TO THE STRESS VECTOR
C   DFDKAPA--------FIRST DERIVATIVE OF THE YIELD FUNCTION WITH RESPECT
C                  TO THE HARDENING PARAMETER
C   DM-------------MATERIAL CONSTITUTIVE MATRIX
C   ICOUNT---------A COUNTER WHICH INDICATES WHETHER EXPLICIT OR
C                  IMPLICIT TIME INTEGRATION SCHEME IS IN USE
C                  ICOUNT=0  EXPLICIT
C                  ICOUNT=1  IMPLICIT
C   NDOF-----------NUMBER OF DEGREES OF FREEDOM IN THE SAMPLE PROBLEM
C   ITER1----------A COUNTER FOR ITERATIONS TO GENERATE STRESS-STRAIN
C                  RELATIONS OF THE MATERIAL
C   JBNDRY---------PARAMETER DENOTING NODES WITH CONSTRAINTS
C   JSIZE----------SIZE OF REDUCED STIFFNESS AND MASS MATRICES AFTER
C                  ELIMINATION DUE TO PRESCRIBED BOUNDARY CONDITIONS
C   MAXITR---------MAXIMUM NUMBER OF ITERATIONS TO GENERATE THE
C                  STRESS-STRAIN RELATION OF THE MATERIAL IN THE CAP MODEL
C   MMAT-----------TYPE OF MATERIAL
C                             MMAT=1  SOIL
C                             MMAT=2  ROCK
C   MMODE----------MODE OF MATERIAL BEHAVIOR
C                             MMODE=0  TENSION MODE
C                             MMODE=1  ELASTIC MODE
C                             MMODE=2  FAILURE MODE
C                             MMODE=3  CAP MODE
C   MTIME1,MTIME2--MAXIMUM NUMBER OF TIME STEPS USED IN THE DYNAMIC
C                  SAMPLE PROBLEM FOR EXPLICIT AND IMPLICIT TIME
C                  INTEGRATION SCHEMES RESP.
C   NCNSTR---------NUMBER OF CONSTRAINTS IN THE SAMPLE PROBLEM
C   NODE-----------NUMBER OF NODES IN THE SAMPLE PROBLEM
```

```
C    NODB3-----------3*NODE
C    PROP------------A VECTOR CONTAINING THE MATERIAL CONSTANTS IN
C                    THE CAP MODEL
C    SIGIJ-----------TOTAL STRESS VECTOR
C    SIGMA-----------TOTAL STRESS VECTOR
C    STIFF-----------REDUCED STIFFNESS MATRIX IN ACCORDANCE WITH
C                    PRESCRIBED BOUNDARY CONDITIONS
C    STIFFL----------GLOBAL STIFFNESS MATRIX
C    TTWOG-----------TWO TIMES THE MATERIAL SHEAR MODULUS
C    VKAPA-----------MATERIAL HARDENING PARAMETER
C    XXC,YYC,ZZC-----COORDINATES OF ELEMENTS IN SAMPLE PROBLEM
C    XMASS-----------REDUCED MASS MATRIX IN ACCORDANCE WITH
C                    PRESCRIBED BOUNDARY CONDITIONS
C    ZMASS-----------GLOBAL MASS MATRIX
C
C
C
C
C
C    ROUTINES CALLED  1) STRESS
C                     2) DMATRX
C                     3) STIF3D
C                     4) REMOVE
C                     5) GMASS
C                     6) EXPLCT
C                     7) IMPLCT
C
C
C
C
C       U S E R   S U P P L I E D   I N P U T S
C       ---------------------------------------
C
C
C 1. STIFFNESS AND MASS MATRIX EVALUATION
C
C    STIFF-------------STIFFNESS MATRIX OF SYSTEM (ELASTIC OR ELASTOPLASTIC)
C    XMASS-------------MASS MATRIX OF SYSTEM.
C    DM----------------MATERIAL CONSTITUTIVE MATRIX. THIS IS SUPPLIED ONLY
C                      WHEN [STIFF] IS NOT SUPPLIED BY USER.
C    XXC,YYC,ZZC-------NODAL COORDINATES OF SPATIALLY DISCRETIZED MEDIUM.
C                      OMIT WHEN [STIFF] AND [XMASS] ARE SUPPLIED BY USER.
C    NCNSTR,JBNDRY-----NUMBER OF CONSTRAINTS IN THE SPATIALLY DISCRETIZED
C                      MEDIUM. OMIT WHEN [STIFF] AND [XMASS] ARE SUPPLIED
C                      BY USER.
C    NGPS--------------NUMBER OF GAUSS POINTS REQUIRED FOR THE FORMULATION
C                      OF STIFFNESS AND MASS MATRICES. OMIT WHEN [STIFF]
C                      AND [XMASS] ARE SUPPLIED BY USER.
C    NODE--------------TOTAL NUMBER OF NODES IN THE SYSTEM.
C    NODB3-------------3*NODE
C    NDOF--------------NUMBER OF DEGREES OF FREEDOM IN THE SYSTEM.
C    MX,MY,MZ----------NUMBER OF NODES (PER PLANE) IN THE X,Y,Z, DIRECTIONS
C                      RESPECTIVELY FOR THE LAGRANGIAN ELEMENT.
C
C    IN THE TEST PROBLEM THE CAP MODEL WAS USED TO GENERATE STRESS-STRAIN
C    RELATIONS FROM WHICH THE ELASTOPLASTIC CONSTITUTIVE MATIX WAS
C    EVALUATED. THIS WAS THEN USED TO EVALUATE THE STIFFNESS MATRIX.
C
C
C
```

```
C
C 2. EVALUATION OF DAMPING CONSTANTS FROM THE Q-MODEL
C
C    Q--------------SPECIFIC DISSIPATION FACTOR OF MATERIAL
C    OMEG1, OMEG2---LOWER AND UPPER FREQUENCY LIMITS FOR THE
C                   CONSTANT Q - VALUE.
C
C
C
C 3. EVALUATION OF DYNAMIC RESPONSE
C
C   CONST1,CONST2---RATIO OF TIME STEP AND FUNDAMENTAL PERIOD OF THE SYSTEM
C                   FOR THE EXPLICIT AND IMPLICIT TIME INTEGRATION RESP.
C                   THESE CONSTANTS HELP TO SELECT APPROPIATE TIME STEPS
C                   FOR STABILITY AND ACCURACY. THE FUNDAMENTAL PERIOD OF
C                   THE SYSTEM IS AUTOMATICALLY EVALUATED BY THE PROGRAM.
C   ICOUNT----------A COUNTER WHICH INDICATES WHETHER EXPLICIT OR IMPLICIT
C                   TIME INTEGRATION SCHEME IS USED.
C                   ICOUNT=0  EXPLICIT
C                   ICOUNT=1  IMPLICIT
C   MTIME1,MTIME2---NUMBER OF TIME STEPS REQUIRED FOR THE SYSTEM RESPONSE
C                   FOR THE EXPLICIT AND IMPLICIT TIME INTEGRATION SCHEMES
C                   RESP.
C
C***********************************************************************
C
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      PARAMETER (NODE=27,NODB3=81,NDOF=12,
     ,   MX=3,MY=3,MZ=3)
      COMMON /QCONST/ Q,OMEG1,OMEG2
      DIMENSION PROP(12),SIGIJ(6,1),SIGMA(6,1),DFDSIG(6,1),
     ,   JBNDRY(200), YMINV(NDOF,NDOF),WWRK1(NDOF,NDOF),
     ,   WWRK2(NDOF),WWRK3(NDOF),WWRK4(NDOF),WWRK5(NDOF),
     ,   ZZ1(NDOF),ZZ2(NDOF),ZZ3(NDOF),ZZ4(NDOF),ZZ4I(NDOF),
     ,   ZZ5(NDOF),ZZ6(NDOF),ZZ66(NDOF),AA0(NDOF,NDOF),
     ,   AA1(NDOF,NDOF),AA2(NDOF,NDOF),WWRK6(NDOF),
     ,   WWRK7(NDOF),WWRK8(NDOF,1),WWRK9(NDOF),DM(6,6),
     ,   DDRVTS(NODE,3,10),FFNC(NODE,3,10),DDNRM(NODE,3),
     ,   BBSAVE(NODE,3),WWRK(NODB3,NODB3),NNLST(NODE),
     ,   XXX(NODE),YYY(NODE),ZZZ(NODE),XXC(NODE),YYC(NODE),
     ,   ZZC(NODE),STIFF(NDOF,NDOF),STIFFL(NODB3,NODB3),
     ,   ZMASS(NODB3,NODB3),XMASS(NDOF,NDOF),SHAPE(NODE),
     ,   BIGN(3,NODB3),BIGNT(NODB3,3),FN(NODB3,NODB3),
     ,   FFORCE(NDOF,1),EFK(NDOF,NDOF),WWRK10(NDOF,NDOF),
     ,   WWRK11(NDOF,1),WWRK12(NDOF,1),WWRK13(NDOF,NDOF),
     ,   EFMAS(NDOF,NDOF)
C
C
C
C
C************************************************************
C
C
C          I N P U T   V A R I A B L E S
C      ---------------------------------------
C
C
C        INPUT SOIL PROPERTIES AND CONSTANTS FOR CAP MODEL
```

```
C
      READ(5,*)(PROP(I),I=1,12)
C
C
C         INPUT NODAL COORDINATES
C
      DO 1 I=1,NODE
    1 READ(5,*)XXC(I),YYC(I),ZZC(I)
C
C
      READ(5,*)MAXITR
C
      READ(5,*)NCNSTR,ICOUNT,CONST1,MTIME1,CONST2,MTIME2,NGP
C
      READ(5,*)Q,OMEG1,OMEG2
C
C        INPUT BOUNDARY CONSTRAINTS
C
      READ(5,*)(JBNDRY(I),I=1,NCNSTR)
      PRINT*,'JBNDRY(I)=',(JBNDRY(I),I=1,NCNSTR)
C
C
      MMAT=1
      VKAPA=0.13304
C
C
C
C*****************************************************************
C
C
C
C          COMPUTE STRESS-STRAIN RELATIONS
C          -------------------------------
C
C
C
C        DECLARE INITIAL STRESSES
C
      DO 3 I=1,6
    3 SIGIJ(I,1)=0.
C
C
C
      DO 4 ITER1=1,MAXITR
C
C
      CALL STRESS(SIGIJ,MMAT,VKAPA,PROP,SIGMA,DFDSIG,ITER1,
     ,       MMODE,TTWOG,DFDKAP)
C
C
      DO 5 II=1,6
    5 SIGIJ(II,1)=SIGMA(II,1)
C
C
      SIGMX=SIGMA(1,1)
      SIGMY=SIGMA(2,1)
      SIGMZ=SIGMA(3,1)
      SIGMXY=SIGMA(6,1)
      SIGMXZ=SIGMA(5,1)
      SIGMYZ=SIGMA(4,1)
```

```
C
C
C
      WRITE(7,1000)
      WRITE(7,1001)ITER1,MMODE
      WRITE(7,1002)
      WRITE(7,1003)SIGMX,SIGMY,SIGMZ,SIGMXY,SIGMXZ,SIGMYZ,
                   VKAPA
C
C
    4 CONTINUE
C
C
C  END OF COMPUTATION OF STRESS-STRAIN RELATIONS
C
C
C************************************************************
C
C
C
C             COMPUTE THE CONSTITUTIVE MATRIX
C             -------------------------------
C
C
C
      CALL DMATRX(SIGMA,PROP,MMODE,DFDSIG,DFDKAP,
     ,             TTWOG,DM)
C
      WRITE(7,1004)
      WRITE(7,1005)((DM(II,JJ),JJ=1,6),II=1,6)
C
C*******************************************************
C
C
C             COMPUTE STIFFNESS MATRIX
C             ------------------------
C
C
      CALL STIF3D(DM,MX,MY,MZ,NGP,XXC,YYC,ZZC,NODE,
     ,    NODB3,DDRVTS,FFNC,DDNRM,BBSAVE,WWRK,NNLST,
     ,    XXX,YYY,ZZZ,AJAC,STIFFL)
C
C
C             REDUCE STIFFNESS  MATRIX IN ACCORDANCE WITH
C             PRESCRIBED BOUNDARY CONDITIONS
C
      CALL REMOVE(STIFFL,NODB3,STIFF,NDOF,JBNDRY,NCNSTR,
     ,             JSIZE)
C
C
C
C************************************************************
C
C
C             COMPUTE GLOBAL MASS MATRIX
C             --------------------------
C
      CALL GMASS(FFNC,AJAC,XXX,YYY,ZZZ,NODE,NODB3,NGP,
     ,            SHAPE,BIGN,BIGNT,FN,ZMASS)
C
```

```
C           REDUCE MASS MATRIX IN ACCORDANCE WITH
C           PRESCRIBED BOUNDARY CONDITIONS
C
      CALL REMOVE(ZMASS,NODB3,XMASS,NDOF,JBNDRY,NCNSTR,
     ,             JSIZE)
C
C
C
C
C*************************************************************
C
C
C
      WRITE(7,1006)
      WRITE(7,1007)((STIFF(II,JJ),JJ=1,NDOF),II=1,NDOF)
C
      WRITE(7,1008)
      WRITE(7,1009)((XMASS(II,JJ),JJ=1,NDOF),II=1,NDOF)
C
C
C
C*************************************************************
C
C
C
C          COMPUTE THE DYNAMIC RESPONSE OF MATERIAL
C          ---------------------------------------
C
C
C
C           TO PERFORM EITHER EXPLICIT OR IMPLICIT TIME
C           INTEGRATION. ICOUNT IS SET TO 0 OR 1 RESP.
C
      IF(ICOUNT.EQ.0) THEN
C
      CALL EXPLCT(XMASS,STIFF,NDOF,CONST1,MTIME1,WWRK1,
     ,  WWRK2,WWRK3,WWRK4,WWRK5,WWRK8,WWRK10,WWRK11,
     ,  WWRK12,WWRK13,ZZ1,ZZ2,ZZ3,ZZ4,ZZ4I,
     ,  ZZ5,ZZ6,ZZ66,AA0,AA1,AA2,FFORCE,EFMAS)
C
C
C
C
      ELSE
C
      CALL IMPLCT(XMASS,STIFF,NDOF,CONST2,MTIME2,WWRK1,
     ,  WWRK2,WWRK3,WWRK4,WWRK5,WWRK6,WWRK7,WWRK8,WWRK9,
     ,  WWRK10,WWRK11,WWRK12,ZZ1,ZZ2,ZZ3,ZZ4,ZZ5,ZZ6,
     ,  ZZ66,AA0,AA1,AA2,FFORCE,EFK)
C
C
C
      ENDIF
C
C
C
C*****************************************************************
C
C
C
```

```
 1000 FORMAT(///////,4X,'ITER1',4X,'MMODE')
 1001 FORMAT(3X,I3,4X,I2)
 1002 FORMAT(///,10X,'SIGMX',9X,'SIGMY',9X,'SIGMZ',
     , 9X,'SIGMXY',9X,'SIGMXZ',9X,'SIGMYZ',9X,'VKAPA')
 1003 FORMAT(2X,8F10.6)
 1004 FORMAT(///,2X,'[   D   ]')
 1005 FORMAT(2X,6F12.6)
 1006 FORMAT(///,2X,'[   STIFFNESS   ]')
 1007 FORMAT(2X,8F9.4)
 1008 FORMAT(///,2X,'[   XMASS   ]')
 1009 FORMAT(2X,8F9.4)
C
C
C
C
      STOP
      END




      SUBROUTINE STRESS(STR0,MAT,HKAPA,YPROP,SIGM,DFDS,ITER,
     +            MODE,TWOG,DFDK)
C
C*************************************************************************
C
C
C
C           THIS SUBROUTINE DETERMINES THE STRESS-STRAIN BEHAVIOR
C             OF THE MATERIAL USING THE CAP MODEL
C
C
C************************************************************************
C
C
C        D E F I N I T I O N   O F    T E R M S
C        ____________________________________________
C
C   BULK----------BULK MODULUS OF MATERIAL
C   CAPL----------FUNCTION DEFINING POSITION OF CAP
C   DEPS----------CHANGE OF STRAIN VECTOR
C   CONV----------CONVERGENCE FACTOR FOR CAP ITERATION
C   DEV-----------CHANGE OF VOLUMETRIC STRAIN
C   DEVP----------CHANGE IN PLASTIC VOLUMETRIC STRAIN
C   DFDJ1---------FIRST DERIVATIVE OF YIELD SURFACE WITH RESPECT TO
C                 THE FIRST INVARIANT OF STRESSES
C   DFDS----------CHANGE IN YIELD SURFACE WITH RESPECT TO STRESS VECTOR
C   EVP-----------PLASTIC VOLUMETRIC STRAIN
C   F-------------FUNCTION DEFINING FAILURE SURFACE
C   FCAP----------FUNCTION DEFINING CAP SURFACE
C   FCUT----------THE VALUE OF J1 FOR WHICH F(J1)=0.
C   GEOP----------INITIAL HYDROSTATIC PRESSURE
C   HKAPA---------HARDENING PARAMETER
```

```
C    ITERA---------COUNTER FOR CAP ITERATION
C    MAT-----------TYPE OF MATERIAL
C                   MAT=1  SOIL
C                   MAT=2  ROCK
C    MODE----------MODE OF MATERIAL BEHAVIOR
C                   MODE=0  TENSION MODE
C                   MODE=1  ELASTIC MODE
C                   MODE=2  FAILURE MODE
C                   MODE=3  CAP MODE
C    NIT-----------MAXIMUM NUMBER OF ITERATIONS FOR CAP CONVERGENCE
C    SHEAR---------SHEAR MODULUS OF MATERIAL
C    SIGKK0--------INITIAL VOLUMETRIC STRESS
C    SIGM----------TOTAL STRESS VECTOR
C    STR-----------DEVIATORIC STRESS VECTOR
C    TCUT----------TENSION CUT-OFF (MAXIMUM TENSION PERMITTED)
C    VARJ1---------FIRST INVARIANT OF STRESSES
C    VARJ2---------SECOND INVARIANT OF DEVIATORIC STRESSES
C    YA,YB,YC,YD,YR,YW,----MATERIAL CONSTANTS IN CAP MODEL
C                           (OBTAINED EXPERIMENTALLY)
C    YNU-----------POISSON'S RATIO
C    YPROP--------VECTOR CONTAINIG THE MATERIAL CONSTANTS IN CAP MODEL
C    X------------FUNCTION DEFINING POSITON OF ELLIPTIC CAP
C
C
C
C***********************************************************************
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION STR(6,1),DEPS(6,1),STR0(6,1),DFDS(6,1),SIGM(6,1),
     +          YPROP(6)
C
C************************ FUNCTION STATEMENTS *************************
C
C          THE FOLLOWING ARE STATEMENT FUNCTIONS DEFINING
C                   CAP MODEL EQUATIONS
C
      EXPS(Z)=DMAX1(DBLE(-500.),DEXP(Z))
C
C     FAILURE ENVELOPE FUNCTION FOR VARJ2  "SQRT J2PRIME"
C
      F(VARJ1)=YA-YC*EXPS(YB*VARJ1)
C
C     CAP STATEMENT FUNCTIONS (CAPL=BIGL(HKAPA),XX=X(HKAPA))
C
      BIGL(HKAPA)=DMIN1(DBLE(0.0),HKAPA)
      R(CAPL)=YR
      X(HKAPA)=HKAPA-R(BIGL(HKAPA))*F(HKAPA)
      EVP(XX)=YW*(EXPS(YD*XX)-1.0)
      FCAP(VARJ1,XX,CAPL)=DSQRT(DABS((XX-CAPL)**2-(VARJ1-CAPL)**2))/
     /R(CAPL)
C
C     ELASTIC MODULI FUNCTIONS (EV IS CURRENT VALUE OF EVP(XX)).
C
      BMOD(VARJ1,EV)=BULK
      SMOD(VARJ2,EV)=SHEAR
C
C********************** END OF FUNCTION STATEMENTS ********************
C
C
```

```
C
C
      YA=YPROP(1)
      YB=YPROP(2)
      YC=YPROP(3)
      YD=YPROP(4)
      YR=YPROP(5)
      YW=YPROP(6)
      BULK=YPROP(7)
      SHEAR=YPROP(8)
      MAT=YPROP(9)
      TCUT=YPROP(10)
      FCUT=YPROP(11)
      YNU=YPROP(12)
C
      CONV=.001
      NIT=60
      GEOP=0.
      ITERA=0
C
C
C
C
C
      DO 1 I=1,6
      DEPS(I,1)=0.0
      DFDS(I,1)=0.0
      STR(I,1)=0.0
    1 CONTINUE
C
      IF(ITER.LE.10)DEPS(3,1)=-0.005
      IF(ITER.GT.10.AND.ITER.LE.15)DEPS(3,1)=0.002
      IF(ITER.GT.15)DEPS(3,1)=0.001
C
C         THE FOLLOWING ARE EQNS 15,16,17 RESP.
C
      CAPL=BIGL(HKAPA)
      XX=X(HKAPA)
      EVPI=EVP(XX)
C
C CALCULATE DEVIATORIC STRAIN INCREMENTS
C
      DEV=DEPS(1,1)+DEPS(2,1)+DEPS(3,1)
      DEVB3=DEV/3.0
      DEPX=DEPS(1,1)-DEVB3
      DEPY=DEPS(2,1)-DEVB3
      DEPZ=DEPS(3,1)-DEVB3
C
C CALCULATE INITIAL DEVIATORIC STRESSES
C
      SIGKK0=(STR0(1,1)+STR0(2,1)+STR0(3,1))/3.0
      STR0(1,1)=STR0(1,1)-SIGKK0
      STR0(2,1)=STR0(2,1)-SIGKK0
      STR0(3,1)=STR0(3,1)-SIGKK0
C
C  INITIAL STRESS INVARIANTS
C
      VARJ1I=3.*(SIGKK0+GEOP)
      VARJ2I=DSQRT((STR0(1,1)**2+STR0(2,1)**2+STR0(3,1)**2+
     +2.*STR0(4,1)**2+2.*STR0(5,1)**2+2.*STR0(6,1)**2)/2.)
```

```
C
C     ELASTIC MATERIAL PROPERTIES
C
      THREEK=3.*BMOD(VARJ1I,EVPI)
      TWOG=2.*SMOD(VARJ2I,EVPI)
C
C****************************************
C                                       *
C          ELASTIC TRIAL                *
C                                       *
C****************************************
C
      STR(1,1)=STR0(1,1)+TWOG*DEPX
      STR(2,1)=STR0(2,1)+TWOG*DEPY
      STR(3,1)=STR0(3,1)+TWOG*DEPZ
      STR(4,1)=STR0(4,1)+TWOG*DEPS(4,1)
      STR(5,1)=STR0(5,1)+TWOG*DEPS(5,1)
      STR(6,1)=STR0(6,1)+TWOG*DEPS(6,1)
C
      RATIO=1.0
      MODE=1
C
C   STRESS INVARIANTS
C
      VARJ1=THREEK*DEV+VARJ1I
      VARJ2=DSQRT((STR(1,1)**2.+STR(2,1)**2.+STR(3,1)**2.+
     +2.*STR(4,1)**2.+2.*STR(5,1)**2.+2.*STR(6,1)**2.)/2.)
C
C****************************************
C                                       *
C           TENSILE CODING              *
C                                       *
C****************************************
C
      TENCUT=DMIN1(FCUT,TCUT+3.*GEOP)
C
C            THE FOLLOWING CONDITION APPLIES
C            WHEN TENSION LIMIT IS NOT EXCEEDED
C
      IF(VARJ1.LT.TENCUT)GO TO 10
C
C            THE FOLLOWING CONDITION APPLIES
C            WHEN TENSION LIMIT IS EXCEEDED
C            THAT IS SET J1=T
C
      VARJ1=TENCUT
      RATIO=0.0
      MODE=0
C
C          . CONDITION FOR EITHER ROCK MODEL OR
C            KAPPA'(DOT) IS.GE.ZERO
C
      IF(MAT.EQ.2.OR.HKAPA.GE.0.0)GO TO 200
C
C           TENSION DILATANCY CODING
C
      HKAPA1=DMAX1(DBLE(0.0),HKAPA+CONV*F(HKAPA))
C
C           EQUATION 16 FOLLOWS
C
```

```
      XXL=X(HKAPA1)
      DENOM=EVP(XXL)-EVPI
      IF(DENOM.GT.0.0)GO TO 5
      HKAPA=0.0
      GO TO 200
C
C           EQN. 22 FOLLOWS
C
5     DEVP=DEV-(VARJ1-VARJ1I)/THREEK
C
      HKAPA=HKAPA+DEVP*(HKAPA1-HKAPA)/DENOM
C
C           EQNS (18), & (20) FOLLOW
C
      HKAPA=DMIN1(DBLE(0.0),HKAPA)
C
      GO TO 200
C
C****************************************
C                                       *
C         CHECK FAILURE ENVELOPE        *
C                                       *
C****************************************
C
10    CONTINUE
C
C            THE FOLLOWING CONDITION IMPLIES FAILURE
C                    MODE DOES NOT APPLY
C
      IF(VARJ1.LT.CAPL)GO TO 30
C
C             VON MISES TRANSITION
C
      VONMIS=FCAP(CAPL,XX,CAPL)
      FJ1=F(VARJ1)
      FF=VARJ2-DMIN1(FJ1,VONMIS)
      IF(FF.LE.0.0)GO TO 200
C
C     FAILURE SURFACE CALLCULATION
C
      MODE=2
      DFDJ1=0.0
C
C     CALLCULATION OF DFE/DJ1 @ J1E
C
      IF(FJ1.LT.VONMIS)DFDJ1=(FJ1-F(VARJ1+CONV*VARJ2))/(CONV*VARJ2)
C
C     EQN (33) FOLLOWS
C
      DEVP=3.*DFDJ1*FF/(3.*THREEK*DFDJ1**2+0.5*TWOG)
C
      VARJ1=VARJ1-THREEK*DEVP
C
C     DILATANCY AND CORNER CODING
C
      IF(MAT.EQ.1.AND.HKAPA.LT.0.0.AND.VARJ1.GT.CAPL)GO TO 60
      VARJ1=DMAX1(VARJ1,CAPL)
      GO TO 70
60    CONTINUE
C
```

```
C      EQN (37) FOLLOWS
C
       HKAPA1=BIGL(VARJ1)
       XXL=X(HKAPA1)
       IF(DEVP.LE.0.0)GO TO 70
       DEVPT=DMAX1(DEVP,EVP(XXL)-EVPI)
       HKAPA=HKAPA+(HKAPA1-HKAPA)*DEVP/DEVPT
C
70     CONTINUE
       FJ1=F(VARJ1)
       RATIO=DMIN1(FJ1,VONMIS)/VARJ2
C
C*************************************
C                                    *
C            CAP CALCULATION         *
C                                    *
C*************************************
C
30     CONTINUE
C
C      CONDITION FOR CAP MODE COMPUTATION
C
       IF(VARJ1.LT.XX)GO TO 40
       IF(VARJ2.LE.FCAP(VARJ1,XX,CAPL))GO TO 200
40     CONTINUE
C
       VARJ1E=VARJ1
       VARJ2E=VARJ2
C
C                   AN INITIAL VALUE FOR THE HARDENING
C                   PARAMETER KAPPA IS ASSUMED AND THEN
C                   REFINED USING THE REGULA FALSI
C                   ITERATION PROCEDURE
C
       HKAPA1=HKAPA
       HKAPA2=VARJ1E
       IF(VARJ1E.LE.XX)FL=(HKAPA-VARJ1E)/(HKAPA-XX)
       IF(VARJ1E.GT.XX)FL=2.*VARJ2E/(VARJ2E+FCAP(VARJ1E,XX,CAPL))-1.0
C
       XR=X(HKAPA2)
       VARJ1R=VARJ1E-THREEK*(EVP(XR)-EVPI)
       FR=(XR-VARJ1R)/(HKAPA2-XR)
       COMP=CONV*F((FL*XR-FR*XX)/(FL-FR))
       IF(DABS(VARJ1)+VARJ2.LT.COMP)GO TO 200
C
       MODE=3
       FOLD=0.0
C
C
C***********************************************************
C
C            ITERATION FOR CAP CONVERGENCE BEGINS
C
C***********************************************************
C
C
       DO 190 ITERA=1,NIT
       HKAPA=(FL*HKAPA2-FR*HKAPA1)/(FL-FR)
       XX=X(HKAPA)
       DEVP=EVP(XX)-EVPI
```

```
      VARJ1=VARJ1E-THREEK*DEVP
      CAPL=BIGL(HKAPA)
      IF(VARJ1.LE.XX)FC=(HKAPA-VARJ1)/(HKAPA-XX)
      IF(VARJ1.GE.CAPL)FC=(XX-VARJ1)/(CAPL-XX)
      IF(VARJ1.LE.XX.OR.VARJ1.GE.CAPL)GO TO 300
C
      VARJ2=FCAP(VARJ1,XX,CAPL)
      DELJ1=CONV*(XX-VARJ1)
      DESP=0.0
      IF(VARJ1+DELJ1.NE.VARJ1)DESP=(DEVP/6.)*(DELJ1/(VARJ2-
     -FCAP(VARJ1+DELJ1,XX,CAPL)))
      VARJ2T=VARJ2+TWOG*DESP
      FC=(VARJ2E-VARJ2T)/(VARJ2E+VARJ2T)
C
C
C*****************************************************************
C
C             OUTLET OF CAP ITERATION LOOP FOLLOWS
C
      IF(DABS(VARJ2E-VARJ2T).LE.COMP)GO TO 195
      IF(FC.GT.0.0.AND.VARJ1-CAPL.GE.DELJ1)GO TO 195
C
C******************************************************************
C
  300 IF(FC.GT.0.)GO TO 320
C
      HKAPA2=HKAPA
      FR=FC
      IF(FOLD.LT.0.)FL=0.5*FL
      GO TO 190
  320 CONTINUE
C
      HKAPA1=HKAPA
      FL=FC
      IF(FOLD.GT.0.)FR=0.5*FR
  190 FOLD=FC
C
C
C
      VARJ1=DMAX1(VARJ1,XX)
      IF(VARJ1.GT.BIGL(HKAPA2))VARJ1=CAPL
      VARJ2=DMIN1(VARJ2E,FCAP(VARJ1,XX,CAPL))
  195 RATIO=0.0
      IF(VARJ2E.NE.0.)RATIO=VARJ2/VARJ2E
  200 CONTINUE
C
C
C           COMPUTE FINAL DEVIATORIC STRESSES
C
      STR(1,1)=STR(1,1)*RATIO
      STR(2,1)=STR(2,1)*RATIO
      STR(3,1)=STR(3,1)*RATIO
      STR(6,1)=STR(6,1)*RATIO
      STR(5,1)=STR(5,1)*RATIO
      STR(4,1)=STR(4,1)*RATIO
C
C
C           COMPUTE FINAL TOTAL STRESSES
C
      SIGKB3=VARJ1/3.-GEOP
```

```
C
      SIGM(1,1)=STR(1,1)+SIGKB3
      SIGM(2,1)=STR(2,1)+SIGKB3
      SIGM(3,1)=STR(3,1)+SIGKB3
      SIGM(4,1)=STR(4,1)
      SIGM(5,1)=STR(5,1)
      SIGM(6,1)=STR(6,1)
C
C
C
      XX=X(HKAPA)
      CAPL=BIGL(HKAPA)
      IF(MODE.LE.1) GO TO 99
      IF(MODE.EQ.2) THEN        !FAILURE MODE
      DFDJ1=YB*YC*EXPS(YB*VARJ1)
      DFDK=0.0
      ENDIF
C
      IF(MODE.EQ.3) THEN        !CAP MODE
      DFDJ1=2.*(VARJ1-CAPL)/(YR*YR*VARJ2)
      DFDX=(CAPL-XX)/(YR*YR*VARJ2)
      DXDK=1.+(YR*YC*YB*(EXPS(YB*HKAPA)))
      DFDL=(XX-VARJ1)/(YR*YR*VARJ2)
      IF(HKAPA.GE.0.) THEN
      DLDK=0.
      ELSE
      DLDK=1.
      ENDIF
      DFDK=DFDX*DXDK+DFDL*DLDK
      ENDIF
C
      IF(VARJ2.LT.1.E-25) GO TO 99
      FCAP2=2.*VARJ2
      DFDS(1,1)=STR(1,1)/FCAP2+DFDJ1
      DFDS(2,1)=STR(2,1)/FCAP2+DFDJ1
      DFDS(3,1)=STR(3,1)/FCAP2+DFDJ1
      DFDS(4,1)=STR(4,1)/FCAP2
      DFDS(5,1)=STR(5,1)/FCAP2
      DFDS(6,1)=STR(6,1)/FCAP2
C
C
C
   99 CONTINUE
C
C
      RETURN
      END


      SUBROUTINE DMATRX(SIGM,XPROP,MODE,DF,DFDK,TWOG,D)
C
C*****************************************************************.
C                                                                *
C        THIS SUBROUTINE COMPUTES THE ELASTO-PLASTIC             *
C        MATRIX FROM THE CAP MODEL                               *
C                                                                *
C    ARGUMENTS :                                                 *
C                                                                *
C       E , XNU -------  YOUNG'S MODULUS AND POISSON'S           *
```

```
C                              RATIO RESPECTIVELY                        *
C                                                                        *
C        D-----------------CONSTITUTIVE MATRIX                           *
C                                                                        *
C         ALL OTHER ARGUMENTS ARE DEFINED  IN                            *
C         THE SUBROUTINE                                                 *
C                                                                        *
C                                                                        *
C      THIS SUBROUTINE IS CALLED BY :                                    *
C                                    (1) MAIN PROGRAM                    *
C                                                                        *
C      THIS SUBROUTINE CALLS   :                                         *
C                                    (1) TRANSP                          *
C                                    (2) MXMULT                          *
C                                                                        *
C*************************************************************************
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION SIGM(6,1),SIGMAT(1,6),DF(6,1),DFT(1,6),
     ,    BB(6,1),BM(1,1),AM(1,1),PD(6,6),QD(1,6),RD(6,6),
     ,    SD(6,6),D(6,6),XPROP(12)
C
C
C
      XNU=XPROP(12)
      A=1.-XNU
      B=1.-2.*XNU
C
C           NOTE: TWOG=E/(1+XNU)
C
C
C
C
C
C
      D(1,1)=TWOG*A/B
      D(2,1)=TWOG*XNU/B
      D(2,2)=TWOG*A/B
      D(3,1)=TWOG*XNU/B
      D(3,2)=TWOG*XNU/B
      D(3,3)=TWOG*A/B
      D(4,1)=0.
      D(4,2)=0.
      D(4,3)=0.
      D(4,4)=0.5*TWOG
      D(5,1)=0.
      D(5,2)=0.
      D(5,3)=0.
      D(5,4)=0.
      D(5,5)=0.5*TWOG
      D(6,1)=0.
      D(6,2)=0.
      D(6,3)=0.
      D(6,4)=0.
      D(6,5)=0.
      D(6,6)=0.5*TWOG
      D(1,2)=D(2,1)
      D(1,3)=D(3,1)
```

```
      D(1,4)=D(4,1)
      D(1,5)=D(5,1)
      D(1,6)=D(6,1)
      D(2,3)=D(3,2)
      D(2,4)=D(4,2)
      D(2,5)=D(5,2)
      D(2,6)=D(6,2)
      D(3,4)=D(4,3)
      D(3,5)=D(5,3)
      D(3,6)=D(6,3)
      D(4,5)=D(5,4)
      D(4,6)=D(6,4)
      D(5,6)=D(6,5)
C
C
C
C
C
C        IN THE ELASTIC MODE MATERIAL BEHAVIOR IS
C        ELASTIC .  THEREFORE THE CONSTITUTIVE MATRIX
C        IS ELASTIC
C
C
C             ELASTIC MODE
      IF(MODE.EQ.0.OR.MODE.EQ.1)GO TO 20
C
C
C        IN THE FAILURE AND CAP MODES THE
C        CONSTITUTIVE MATRIX IS COMPUTED AS
C        ELASTO-PLASTIC
C
C
C*****************************************************
C                                                    *
C       COMPUTE ELASTO-PLASTIC CONSTITUTIVE MATRIX   *
C                                                    *
C*****************************************************
C
C          COMPUTE [DF] TRANSPOSE
C
      CALL TRANSP(DF,DFT,6,1)
C
C          COMPUTE [SIGMA] TRANSPOSE
C
      CALL TRANSP(SIGM,SIGMAT,6,1)
C
C
      CALL MXMULT(D,DF,BB,6,6,6,1)
C
      CALL MXMULT(DFT,BB,BM,1,6,6,1)
C
      CALL MXMULT(SIGMAT,DF,AM,1,6,6,1)
C
C         HARD  REPRESENTS THE HARDENING FUNCTION
      HARD=-DFDK*AM(1,1)
C
      HM=1./(HARD+BM(1,1))
C
      DO 12 II=1,6
      DO 12 JJ=1,6
```

```
   12 PD(II,JJ)=D(II,JJ)*HM
C
      CALL MXMULT(DFT,PD,QD,1,6,6,6)
      CALL MXMULT(DF,QD,RD,6,1,1,6)
      CALL MXMULT(D,RD,SD,6,6,6,6)
C
      DO 14 II=1,6
      DO 14 JJ=1,6
   14 D(II,JJ)=D(II,JJ)-SD(II,JJ)
C
C
C
C
   20 CONTINUE
C
C
      RETURN
      END


        SUBROUTINE MULTVC(A,B,C,NR)
C
C*****************************************************************
C                                                                *
C                                                                *
C                    THIS  SUBROUTINE PERFORMS                   *
C                   MULTIPLICATION OF MATRICES                   *
C                    IN SYMMETRIC STORAGE FORM.                  *
C                                                                *
C                                                                *
C*****************************************************************
C                                                                *
C  ARGUMENTS:    A   ---- OUTPUT VECTOR OF LENGTH N(N+1)/2 CON-  *
C                         TAINING THE RESULT OF [B].{C} IN SYM-  *
C                         METRIC STORAGE FORM.                   *
C                B   ---- INPUT/OUTPUT VECTOR OF LENGTH N(N+1)/2 *
C                         WHICH IS THE SYMMETRIC STORAGE FORM.   *
C                C   ---- INPUT/OUTPUT VECTOR OF LENGTH N.       *
C                NR  ---- ROW DIMENSION OF THE MATRIX [B].       *
C                                                                *
C                                                                *
C                                                                *
C                                                                *
C*****************************************************************
C
C
C
        IMPLICIT REAL*8(A-H,O-Z)
        DIMENSION A(990),B(990),C(44)
        INTEGER IR,NR,KL,KS,KSS,KSL,I
        DO 101 I=1,NR
101     A(I)=0.0
        DO 103  IR=1,NR
           KL=IR*(IR-1)/2
              SUM=0.0
              DO 102  KS=1,NR
                 KSS=KS*(KS-1)/2
                 IF(KS.LT.IR) KSL=KL+KS
                 IF(KS.GE.IR) KSL=KSS+IR
```

```
              SUM=SUM+B(KSL)*C(KS)
102        CONTINUE
          A(IR)=SUM
103     CONTINUE
        RETURN
        END


      SUBROUTINE REMOVE(XM,NODB3,YM,KDOF,KBNDRY,KONSTR,
                        ISIZE)
C
C
C*************************************************************
C                                                            *
C                                                            *
C      THIS SUBROUTINE ELIMINATES APPROPRIATE ROWS AND       *
C        COLUMNS FROM THE GLOBAL MATRIX TO SATISFY           *
C                 SUPPORT CONDITIONS                         *
C                                                            *
C                                                            *
C                                                            *
C                                                            *
C  THIS SUBROUTINE IS CALLED BY :                            *
C                              (1) MAIN PROGRAM              *
C                                                            *
C  THIS SUBROUTINE CALLS :    NONE                           *
C                                                            *
C*************************************************************
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION XM(NODB3,NODB3),YM(KDOF,KDOF),KBNDRY(200)
C
C
C_____________________________________________________*
C_____________________________________________________*
C        ARGUMENTS ARE:                               *
C             XM----BIG MATRIX                        *
C             YM----SMALL MATRIX                      *
C          ISIZE----SIZE OF REDUCED MATRIX            *
C             ( ISIZE X ISIZE )                       *
C   IROW, JCOL------VECTORS CONTAINING THE            *
C                   ORIGINAL DIRECTION (3i-2),(3i-1)  *
C                   (3i) FOR EACH NODE AS REPRESENTED*
C                   IN THE REDUCED MTRIX              *
C                                                     *
C_____________________________________________________*
C_____________________________________________________
C
      M=1
      DO 1 I=1,NODB3
C
C
C          THE FOLLOWING DESCRIBE THE BOUNDARY CONDITIONS
C          BY SETTING DISPLACEMENTS AT SUPPORTS TO ZERO
C
      DO 10 L=1,KONSTR
```

```
      IF(I.EQ.KBNDRY(L))GO TO 1
   10 CONTINUE
C
C
C
      N=1
      DO 2 J=1,NODB3
C
      DO 11 L=1,KONSTR
      IF(J.EQ.KBNDRY(L))GO TO 2
   11 CONTINUE
C
C
C
      YM(M,N)=XM(I,J)
C
      N=N+1
C
    2 CONTINUE
C
      M=M+1
C
    1 CONTINUE
C
      M=M-1
      N=N-1
      ISIZE=M
C
      RETURN
      END


      SUBROUTINE TRANSP(AA,BB,IROW,ICOL)
C
C*********************************************
C                                            *
C              THIS SUBROUTINE TRANSPOSES    *
C                   MATRICES                 *
C                                            *
C               [BB]=[AA] TRANSPOSE          *
C                                            *
C   ARGUMENTS ARE:                           *
C       AA------(INPUT) MATRIX TO BE TRANSPOSED *
C       BB------(OUTPUT) TRANSPOSED MATRIX   *
C  IROW,ICOL---NUMBER OF ROWS AND COLUMNS RESP.*
C              OF THE INPUT MATRIX AA        *
C                                            *
C                                            *
C THIS SUBROUTINE IS CALLED BY:              *
C                          (1) MAIN PROGRAM  *
C                          (2) DMATRX        *
C THIS SUBROUTINE CALLS :  NONE              *
C*********************************************
C
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
```

```
      DIMENSION AA(IROW,ICOL),BB(ICOL,IROW)
C
C
      DO 1 M=1,IROW
      DO 2 J=1,ICOL
      BB(J,M)=AA(M,J)
    2 CONTINUE
    1 CONTINUE
C
      RETURN
      END


      SUBROUTINE MXMULT(AA,BB,CC,IROWA,ICOLA,IROWB,ICOLB)
C
C*************************************************
C                                                *
C     THIS SUBROUTINE MULTIPLIES TWO MATRICES    *
C                                                *
C   ARGUMENTS ARE:                               *
C                                                *
C           [CC] = [AA][BB]                      *
C                                                *
C          IROWA & IROWB ARE NUMBER OF ROWS IN   *
C          THE MATRICES [AA] , [BB] RESP.        *
C                                                *
C          ICOLA & ICOLB ARE NUMBER OF COLUMNS   *
C          IN MATRICES [AA] , [BB] RESP.         *
C                                                *
C                                                *
C                                                *
C   THIS SUBROUTINE IS CALLED BY :               *
C                              (1) MAIN PROGRAM*
C                              (2) DMATRX        *
C                                                *
C   THIS SUBROUTINE CALLS :    NONE              *
C                                                *
C*************************************************
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AA(IROWA,ICOLA),BB(IROWB,ICOLB),
     *            CC(IROWA,ICOLB)
C
      DO 1 M=1,IROWA
      DO 2 J=1,ICOLB
C
      SUM=0.
C
      DO 3 JJ=1,ICOLA
      CC(M,J)=AA(M,JJ)*BB(JJ,J)
      SUM=SUM+CC(M,J)
    3 CONTINUE
C
      CC(M,J)=SUM
C
    2 CONTINUE
    1 CONTINUE
```

```
C
      RETURN
      END


      SUBROUTINE PLOT2D(YDISP,XTIME,LDOF,LTIME)
C
C
C THIS ROUTINE PLOTS RESULTS FROM COMPUTER OUTPUT
      IMPLICIT REAL(A-H,O-Z)
      DIMENSION XTIME(1000),YDISP(1000,100)
C
C             SET THE Y (DISPLACEMENT) MINIMUM AND MAXIMUM AS
C             -.003 AND +.003 RESPECTIVELY
C
      CALL AGSETF('Y/MINIMUM.', -.003)
      CALL AGSETF('Y/MAXIMUM.',  .003)
C
C             SET UP THE LEFT LABEL
C      CALL AGSETF('LABEL/NAME.','L')
C      CALL AGSETI('LINE/NUMBER.',100)
C      CALL AGSETP('LINE/TEXT.','NODAL DISPLACEMENT$',1)
C
C             SET UP BOTTOM LABEL
C      CALL AGSETF('LABEL/NAME.','B')
C      CALL AGSETI('LINE/NUMBER.',-100)
C      CALL AGSETP('LINE/TEXT.','TIME$',1)
C
C
C
C           SET UP LABELS
C
      CALL ANOTAT('TIME(S)$','NODAL DISPLACEMENT(FT)$',
                  0,0,0,0)
C
C
C
C          DRAW BOUNDARY ARUOND THE EDGE OF THE PLOTTER FRAME
C
      CALL BNDARY
C
C         DRAW THE GRAPH, USING EZMXY
C
      CALL EZMXY(XTIME,YDISP,LTIME,LDOF,LTIME,
               'DISP VRS. TIME (EXPLICIT)$')
C
      RETURN
      END


      SUBROUTINE EXPLCT(XMAS,STIF,NDOF,CONST,NTIME,WRK1,
     ,      WRK2,WRK3,WRK4,WRK5,WRK8,WRK10,WRK11,WRK12,
     ,      WRK13,Z1,Z2,Z3,Z4,Z4I,Z5,Z6,Z66,
     ,      A0,A1,A2,FORCE,EFMAS)
C
C*****************************************************************
```

```
C                                                                     *
C                                                                     *
C                                                                     *
C     D E F I N I T I O N     O F     V A R I A B L E S               *
C    ____________________________________________________             *
C                                                                     *
C                                                                     *
C    STIF---------------- STIFFNESS MATRIX OF SYSTEM                  *
C    TAU0,TAU1,TAU2,TAU3-INTEGRATION PARAMETERS                       *
C    XMAS-----------------MASS MATRIX OF SYSTEM                       *
C    WRK1 THRU WRK6------WORKING MATRIX AND VECTORS USED AS           *
C                        INTERMEDIARY STAGES IN VARIOUS STAGES        *
C    Z1------------------ TRIPLE INTEGRATION OF DISPLACEMENT          *
C                        VECTOR WITH RESPECT TO TIME                  *
C    Z2-------------------DOUBLE INTEGRATION OF DISPLACEMENT          *
C                        VECTOR WITH RESPECT TO TIME                  *
C    Z3-------------------INTEGRATION OF DISPLACEMENT VECTOR          *
C                        WITH RESPECT TO TIME                         *
C    Z4-------------------VECTOR CONTAINING NODAL DISPLACEMENT        *
C                        AT TIME T                                    *
C    Z4I------------------VECTOR CONTAINING NODAL DISPLACEMENT        *
C                        AT TIME T-DELT                               *
C    Z5-------------------VELOCITY VECTOR                             *
C    Z6-------------------ACCELERATION VECTOR                         *
C                                                                     *
C                                                                     *
C                                                                     *
C      ROUTINES CALLED  1) QMODEL                                     *
C                       2) VMULT                                      *
C                       3) PLOT2D                                     *
C**********************************************************************
C
C  THIS ROUTINE COMPUTES EXPLICIT TIME INTEGRATION
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION XMAS(NDOF,NDOF),STIF(NDOF,NDOF),Z1(NDOF),
     ,   Z2(NDOF),Z3(NDOF),Z4(NDOF),Z4I(NDOF),Z5(NDOF),
     ,   Z6(NDOF),WRK1(NDOF,NDOF),WRK2(NDOF),WRK3(NDOF),
     ,   WRK4(NDOF),WRK5(NDOF),FORCE(NDOF,1),A0(NDOF,NDOF),
     ,   A1(NDOF,NDOF),A2(NDOF,NDOF),Z66(NDOF),WRK6(5050),
     ,   YMAS(5050),EFMAS(NDOF,NDOF),UD(100),XB(3,1),
     ,   WRK8(NDOF,1),WRK10(NDOF,NDOF),WRK11(NDOF,1),
     ,   WRK12(NDOF,1),WRK13(NDOF,NDOF),EIGM(100,100),
     ,   EIGV(100)
C
      REAL ZTIME(1000),ZDISP(1000,100)
C
C
C****************************************************************************************
C
      T0=0.
      TIME=T0
      PI=3.1415927
C
C
C          EVALUATE THE NATURAL FREQUENCY OF THE SYSTEM.
C         THIS IS ESSENTIAL FOR SELECTION OF TIME STEP FOR STABILITY
C
      CALL EIGN(STIF,XMAS,NDOF,EIGM,EIGV)
```

```
C
      OMSQ=0.
      DO 13 I=1,NDOF
   13 OMSQ=DMAX1(EIGV(I),OMSQ)
      OMEG1=DSQRT(OMSQ)
C
      PRIOD=2.*PI/OMEG1
      DELT=PRIOD*CONST
C
C
C***********************************************************************
C
C                COMPUTE INTEGRATION PARAMETERS
C
      TAU0=1./(DELT**2.)
      TAU1=1./(2.*DELT)
      TAU2=2.*TAU0
      TAU3=1./TAU2
C
C
C
C************************************************************
C
C        COMPUTE ALPHA, BETA, GAMMA CONSTANTS FROM Q - MODEL
C
C
      CALL QMODEL(XB)
C
      ALPHA=XB(1,1)
      BETA=XB(2,1)
      GAMMA=XB(3,1)
C
C
C         COMPUTE DAMPING MATRICES [A0], [A1], [A2]
C
      DO 6 II=1,NDOF
      DO 6 JJ=1,NDOF
      A0(II,JJ)=ALPHA*STIF(II,JJ)
      A1(II,JJ)=BETA*STIF(II,JJ)
      A2(II,JJ)=GAMMA*STIF(II,JJ)
    6 CONTINUE
C
C
C***********************************************************
C
C
C               COMPUTE INITIAL VALUES OF VARIABLES
C
      DO 1 I=1,NDOF
      Z1(I)=0.
      Z2(I)=0.
      Z3(I)=0.
      Z4(I)=0.
      Z5(I)=0.
    1 CONTINUE
C
C
C              TRIANGULARIZE MASS MATRIX
C
      CALL DECOMP(XMAS,WRK10,NDOF,NDOF,1)
```

```
C
C
C
C         COMPUTE INITIAL ACCELERATION
C
C           FIRST COMPUTE [K]*{Z4}
      CALL VMULT(STIF,Z4,Z6,NDOF,NDOF,NDOF)
C
C              NEXT COMPUTE [A0]*{Z5}
      CALL VMULT(A0,Z5,Z66,NDOF,NDOF,NDOF)
C
C
      DO 2 I=1,NDOF
      WRK8(I,1)=FCN(TIME,I)-Z6(I)-Z66(I)
    2 CONTINUE
C
      CALL SOLVEQ(WRK10,WRK8,WRK11,NDOF,NDOF,1,1,1,1,1)
C
      DO 14 I=1,NDOF
   14 Z6(I)=WRK11(I,1)
C
C
C             COMPUTE Z4(-DELT)
      DO 3 M=1,NDOF
    3 Z4I(M)=Z4(M)-DELT*Z5(M)+TAU3*Z6(M)
C
C
C************************************************************************
C
C               TIME ITERATION BEGINS
C
      DO 4 JTIME=1,NTIME
C
C
C            COMPUTE EFFECTIVE FORCE VECTOR
C
      DO 5 I=1,NDOF
      DO 5 J=1,NDOF
    5 WRK1(I,J)=STIF(I,J)-TAU2*XMAS(I,J)
      CALL VMULT(WRK1,Z4,WRK2,NDOF,NDOF,NDOF)
C
      DO 12 I=1,NDOF
      DO 12 J=1,NDOF
   12 WRK1(I,J)=TAU0*XMAS(I,J)-TAU1*A0(I,J)
      CALL VMULT(WRK1,Z4I,WRK3,NDOF,NDOF,NDOF)
      CALL VMULT(A1,Z3,WRK4,NDOF,NDOF,NDOF)
      CALL VMULT(A2,Z1,WRK5,NDOF,NDOF,NDOF)
C
C
C
      DO 7 I=1,NDOF
    7 FORCE(I,1)=FCN(TIME,I)-WRK2(I)-WRK3(I)-WRK4(I)-WRK5(I)
C
C
C
C
C             COMPUTE EFFECTIVE MASS MATRIX
C
      DO 8 I=1,NDOF
      DO 8 J=1,NDOF
```

```
    8 EFMAS(I,J)=TAU0*XMAS(I,J)+TAU1*A0(I,J)
C
C
C         TRIANGULARIZE EFFECTIVE MASS MATRIX
C
      CALL DECOMP(EFMAS,WRK13,NDOF,NDOF,1)
C
C          CALCULATE DISPLACEMENT VECTOR AT TIME T+DELT
C
      CALL SOLVEQ(WRK13,FORCE,WRK12,NDOF,NDOF,1,1,1,1)
C
      DO 15 I=1,NDOF
   15 UD(I)=WRK12(I,1)
C
C
      WRITE(7,2000)
      WRITE(7,2001)(UD(I),I=1,NDOF)
C
C
C           REARRANGE DATA FOR PLOTTING
C
      ZTIME(JTIME)=TIME
      DO 11 IJ=1,4
      JJ=IJ+6
   11 ZDISP(JTIME,IJ)=Z4(JJ)
C
C
C         INCREMENT TIME AND UPDATE VARIABLES
C
      TIME=TIME+DELT
C
C
      DO 10 I=1,NDOF
      Z1(I)=Z3(I)/TAU2+Z1(I)+DELT*Z2(I)
      Z2(I)=Z4(I)/TAU2+Z2(I)+DELT*Z3(I)
      Z3(I)=Z5(I)/TAU2+Z3(I)+DELT*Z4(I)
      Z4I(I)=Z4(I)
      Z4(I)=UD(I)
   10 CONTINUE
C
C
C
C
    4 CONTINUE
C
C
C***********************************************************
C
C
C           PLOT THE RESULTS
C
      CALL PLOT2D(ZDISP,ZTIME,4,NTIME)
C
C
C
C
C**************************************************************
C
C
 2000 FORMAT(//,6X,'DISPLACEMENT')
```

```
 2001 FORMAT(2X,8F12.6)
C
C
C
C
      RETURN
      END
C
C
C
C         EXTERNAL FUNCTION FOR THE FORCE VECTOR AT EACH TIME STEP
      FUNCTION FCN(TIME,I)
      IMPLICIT REAL*8(A-H,O-Z)
      OMEGA=1.
      FCN=0.
      IF(I.GE.5)FCN=.1*DSIN(OMEGA*TIME)
      RETURN
      END


      SUBROUTINE VMULT(AA,BB,CC,IROWA,ICOLA,IROWB)
C
C
C*****************************************************************
C
C
C         THIS ROUTINE MULTIPLIES THE MATRIX [AA] BY
C         THE VECTOR {BB} TO GIVE THE VETOR {CC}
C              {CC} = [AA]{BB}
C
C******************************************************************
C
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AA(IROWA,ICOLA),BB(IROWB),CC(IROWA)
C
      DO 1 M=1,IROWA
C
      SUM=0.
C
      DO 2 JJ=1,ICOLA
      CC(M)=AA(M,JJ)*BB(JJ)
      SUM=SUM+CC(M)
    2 CONTINUE
C
      CC(M)=SUM
C
    1 CONTINUE
C
      RETURN
      END


      SUBROUTINE BNDARY
C
C
```

```
      CALL PLOTIT(    0,     0,0)
      CALL PLOTIT(32767,     0,1)
      CALL PLOTIT(32767,32767,1)
      CALL PLOTIT(    0,32767,1)
      CALL PLOTIT(    0,     0,1)
C
C
      RETURN
      END



      SUBROUTINE IMPLCT(XMAS,STIF,NDOF,CONST,NTIME,
     ,   WRK1,WRK2,WRK3,WRK4,WRK5,WRK6,WRK7,WRK8,WRK9,
     ,   WRK10,WRK11,WRK12,Z1,Z2,Z3,Z4,Z5,Z6,Z66,
     ,   A0,A1,A2,FORCE,EFFK)
C
C
C*******************************************************************
C                                                                  *
C       THIS ROUTINE COMPUTES IMPLICIT TIME INTEGRATION            *
C                                                                  *
C*******************************************************************
C                                                                  *
C                                                                  *
C                                                                  *
C     D E F I N I T I O N   O F   V A R I A B L E S                *
C     -----------------------------------------------              *
C                I/O                                               *
C     DEL-------------------NEWMARK'S INTEGRATION PARAMETER        *
C     DELT--------I--------TIME INTERVALS FOR NUMERICAL INTEGRATION  *
C     EIGM--------O--------MATRIX STORING EIGENVETORS OF SYSTEM    *
C                          COLUMNWISE                              *
C     EIGV--------O--------VECTOR STORING EIGENVALUES              *
C     STIF--------I--------STIFFNESS MATRIX OF SYSTEM              *
C     TAU0 THRU TAU7-----INTEGRATION PARAMETERS                    *
C     UD----------O--------NODAL DISPLACEMENT AT TIME T (SAME AS Z4) *
C     XMAS--------I--------MASS MATRIX OF SYSTEM                   *
C     YMAS------------------MASS MATRIX OF SYSTEM IN SYMMETRIC STORAGE*
C                          FORM                                    *
C     WRK1 THRU WRK9-----WORKING VECTORS USED AS INTERMEDIARY      *
C                          STEPS IN VARIOUS STAGES OF COMPUTATION  *
C     Z1--------------------TRIPLE INTEGRATION OF DISPLACEMENT     *
C                          VECTOR WITH RESPECT TO TIME             *
C     Z2--------------------DOUBLE INTEGRATION OF DISPLACEMENT     *
C                          VECTOR WITH RESPECT TO TIME             *
C     Z3--------------------INTEGRATION OF DISPLACEMENT VECTOR     *
C                          WITH RESPECT TO TIME                    *
C     Z4--------------------VECTOR CONTAINING NODAL DISPLACEMENT   *
C                          AT TIME T                               *
C     Z5--------------------VELOCITY VECTOR                        *
C     Z6--------------------ACCELERATION VECTOR                    *
C                                                                  *
C                                                                  *
C                                                                  *
C     ROUTINES CALLED   1) QMODEL                                  *
C                       2) VMULT                                   *
C                       3) EIGN                                    *
C                                                                  *
```

```
C*******************************************************************
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION XMAS(NDOF,NDOF),STIF(NDOF,NDOF),Z1(NDOF),
     ,   Z2(NDOF),Z3(NDOF),Z4(NDOF),Z5(NDOF),Z6(NDOF),
     ,   Z66(NDOF),WRK1(NDOF,NDOF),WRK2(NDOF),WRK3(NDOF),
     ,   WRK4(NDOF),WRK5(NDOF),WRK6(NDOF),WRK7(NDOF),
     ,   FORCE(NDOF,1),YSTIF(5050),A0(NDOF,NDOF),
     ,   A1(NDOF,NDOF),A2(NDOF,NDOF),EFFK(NDOF,NDOF),
     ,   UD(100),XB(3,1),WRK8(NDOF,1),YMAS(5050),
     ,   WRK9(NDOF),WRK10(NDOF,NDOF),WRK11(NDOF,1),
     ,   WRK12(NDOF,1),EIGM(100,100),EIGV(100)
C
      REAL ZTIME(200),ZDISP(200,100)
C
C
C
C*********************************************************************
C
C
C
C
C
      T0=0.
      TIME=T0
      PI=3.1415927
C
C
C
C     EVALUATE THE NATURAL FREQUENCIES OF THE SYSTEM
C     THIS IS ESSENTIAL FOR ACCURACY IN CHOICE OF TIME STEP DELT
C
      CALL EIGN(STIF,XMAS,NDOF,EIGM,EIGV)
C
C    EVALUATE THE FUNDAMENTAL FREQUENCY AND PERIOD OF SYSTEM
C
      OMSQ=0.
      DO 16 I=1,NDOF
   16 OMSQ=DMAX1(EIGV(I),OMSQ)
      OMEG1=DSQRT(OMSQ)
      PRIOD=2.*PI/OMEG1
C
C
      DELT=PRIOD*CONST
C
C
C******************************************************************
C
C
C
C        COMPUTE INTEGRATION PARAMETERS
C
      DEL=1.
      XLAMD=0.25*(0.5+DEL)**2.
C
      TAU0=1./(XLAMD*DELT**2.)
      TAU1=DEL/(XLAMD*DELT)
      TAU2=1./(XLAMD*DELT)
```

```
      TAU3=(1./2.*XLAMD)-1.
      TAU4=DEL/XLAMD-1.
      TAU5=DELT*(DEL/XLAMD-2.)/2.
      TAU6=DELT*(1.-DEL)
      TAU7=DEL*DELT
C
C*****************************************************************
C
C         COMPUTE ALPHA, BETA, GAMMA CONSTANTS FROM Q - MODEL
C
      CALL QMODEL(XB)
C
      ALPHA=XB(1,1)
      BETA=XB(2,1)
      GAMMA=XB(3,1)
C
C
C         COMPUTE DAMPING MATRICES [A0],[A1],[A2]
C
      DO 3 II=1,NDOF
      DO 3 JJ=1,NDOF
      A0(II,JJ)=ALPHA*STIF(II,JJ)
      A1(II,JJ)=BETA*STIF(II,JJ)
      A2(II,JJ)=GAMMA*STIF(II,JJ)
    3 CONTINUE
C
C
C*******************************************************************
C              COMPUTE INITIAL VALUES OF VARIABLES
C
      DO 1 I=1,NDOF
      Z1(I)=0.
      Z2(I)=0.
      Z3(I)=0.
      Z4(I)=0.
      Z5(I)=0.
    1 CONTINUE
C
C
C        TRIANGULARIZE THE MASS MATRIX
      CALL DECOMP(XMAS,WRK10,NDOF,NDOF,1)
C
C        COMPUTE INITIAL ACCELERATION AS FOLLOWS
C                  -1
C      {Z6(0)}=[M]  {F(0)-[K]{Z4(0)}-[A0]{Z5(0)}} (SINCE
C      [A1]{Z3(0)},[A2]{Z1(0)} ARE IDENTICALLY EQUAL TO ZERO
C
C        FIRST COMPUTE [K]{Z4(0)}
      CALL VMULT(STIF,Z4,WRK9,NDOF,NDOF,NDOF)
C
C
C           NEXT COMPUTE [A0]{Z5(0)}
      CALL VMULT(A0,Z5,Z66,NDOF,NDOF,NDOF)
C
C
      DO 2 I=1,NDOF
      WRK8(I,1)=FFCN(TIME,I)-WRK9(I)-Z66(I)
    2 CONTINUE
C
C              COMPUTE INITIAL VALUE OF Z6
```

```
      CALL SOLVEQ(WRK10,WRK8,WRK11,NDOF,NDOF,1,1,1,1,1)
      DO 102 I=1,NDOF
  102 Z6(I)=WRK11(I,1)
C*******************************************************************
C
C
C
C          COMPUTE EFFECTIVE STIFFNESS MATRIX
C
      DO 5 I=1,NDOF
      DO 5 J=1,NDOF
    5 EFFK(I,J)=STIF(I,J)+TAU0*XMAS(I,J)+TAU1*A0(I,J)
C
C
C
C                TRIANGULARIZE THE EFFECTIVE STIFFNESS MATRIX
C                AND STORE IN WRK1
C
      CALL DECOMP(EFFK,WRK1,NDOF,NDOF,1)
C
C
C************************************************************************
C
C              TIME ITERATION BEGINS
C
      DO 4 JTIME=1,NTIME
C
C
C
C           CALCULATE EFFECTIVE LOADS AT TIME T+DELT
C
      DO 7 II=1,NDOF
    7 WRK2(II)=TAU0*Z4(II)+TAU2*Z5(II)+TAU3*Z6(II)
C
C
C              COMPUTE [XMAS][WRK2]
      DO 104 MM=1,NDOF
      SUM1=0.
      DO 105 JJ=1,NDOF
      WRK3(MM)=XMAS(MM,JJ)*WRK2(JJ)
      SUM1=SUM1+WRK3(MM)
  105 CONTINUE
      WRK3(MM)=SUM1
  104 CONTINUE
C
      DO 8 II=1,NDOF
    8 WRK4(II)=TAU1*Z4(II)+TAU4*Z5(II)+TAU5*Z6(II)
C
      CALL VMULT(A0,WRK4,WRK5,NDOF,NDOF,NDOF)
C
C
C
C          COMPUTE Z1(T+DELT), Z2(T+DELT), Z3(T+DELT) USING
C          TAYLOR'S EXPANSION
C
      DO 9 II=1,NDOF
      Z1(II)=Z1(II)+Z2(II)*DELT+Z3(II)*0.5*DELT**2.
      Z2(II)=Z2(II)+Z3(II)*DELT+Z4(II)*0.5*DELT**2.
      Z3(II)=Z3(II)+Z4(II)*DELT+Z5(II)*0.5*DELT**2.
    9 CONTINUE
```

```
C
      CALL VMULT(A1,Z3,WRK6,NDOF,NDOF,NDOF)
      CALL VMULT(A2,Z1,WRK7,NDOF,NDOF,NDOF)
C
C
C         COMPUTE EFFECTIVE FORCE AT TIME T+DELT
C
      TTIME=TIME+DELT
      DO 10 I=1,NDOF
   10 FORCE(I,1)=FFCN(TTIME,I)+WRK3(I)+WRK5(I)-
     -WRK6(I)-WRK7(I)
C
C
C        CALCULATE DISPLACEMENT VECTOR AT TIME T+DELT
      CALL SOLVEQ(WRK1,FORCE,WRK12,NDOF,NDOF,1,1,1,1)
C
      DO 14 I=1,NDOF
      UD(I)=WRK12(I,1)
   14 CONTINUE
      WRITE(7,2000)
      WRITE(7,2001)(UD(I),I=1,12)
C
C
C
C----------------------------------------
C            REARRANGE DATA FOR PLOTTING
C
      ZTIME(JTIME)=TIME
      DO 11 IJ=1,4
      JJ=IJ+6
   11 ZDISP(JTIME,IJ)=Z4(JJ)
C----------------------------------------
C
C
C
C            COMPUTE ACCELERATION AND VELOCITY VECTORS
C                    AT TIME T+DELT
C
      DO 12 I=1,NDOF
      SAV6=Z6(I)
      Z6(I)=TAU0*(UD(I)-Z4(I))-TAU2*Z5(I)-TAU3*SAV6
      Z5(I)=Z5(I)+TAU6*SAV6+TAU7*Z6(I)
      Z4(I)=UD(I)
   12 CONTINUE
C
C
C
C
C                    INCREMENT TIME
C
      TIME=TIME+DELT
C
C
C
C
    4 CONTINUE
C
C
C
C
```

```
C*************************************************************
C
C
C                  PLOT RESULTS
C
C         SET THE DISPLACEMENT AXIS TO A MINIMUM AND
C         MAXIMUM OF -.003 1NCH AND +.003 INCH RESPECTIVELY
C
C
      CALL AGSETF('Y/MINIMUM.', -.003)
      CALL AGSETF('Y/MAXIMUM.',  .003)
C
C
C                 SET UP LABELS
C
      CALL ANOTAT('TIME(S)$','NODAL DISPLACEMENT(FT)$',
     ;            0,0,0,0)
C
C
C          DRAW THE BOUNDARY AROUND THE PLOTTER FRAME
C
      CALL BNDARY
C
C            DRAW THE GRAPH, USING EZMXY
C
      CALL EZMXY(ZTIME,ZDISP,NTIME,4,NTIME,
     ;    'DISP VRS. TIME (IMPLICIT)$')
C
 2000 FORMAT(//,6X,'DISPLACEMENT')
 2001 FORMAT(2X,12F8.3)
C
C
C**********************************************************
C
C
C
      RETURN
      END
C
C
C          EXTERNAL FUNCTION FOR THE FORCE VECTOR AT EACH TIME STEP
      FUNCTION FFCN(TTIME,I)
      IMPLICIT REAL*8(A-H,O-Z)
      OMEGA=1.
      FFCN=0.
      IF(I.GE.5)FFCN=.1*DSIN(OMEGA*TTIME)
      RETURN
      END


      SUBROUTINE QMODEL(B)
C
C
C***************************************************************
C
C   THIS ROUTINE COMPUTES THE ALPHA, BETA, GAMMA CONSTANTS
C   FROM MEAN SQUARE ERROR IN THE Q - MODEL
C
C***************************************************************
```

```
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON /QCONST/ Q,OMEG1,OMEG2
C
      DIMENSION A(3,3),B(3,1),C(3,1),WRK(3,3)
C
C
C*******************************************************************
C
C
C          D E F I N I T I O N   O F   T E R M S
C         ---------------------------------------
C
C     A ------------------ MATRIX WHOSE ELEMENTS ARE THE COEFFICIENT
C                          OF THE VARIABLES IN THE SYSTEM OF EQUATIONS
C                          IN THE MEAN SQUARE ERROR FORMULATION
C                          MATRIX IS STORED AS FULL MATRIX
C     B ------------------ A VECTOR WHOSE ELEMENTS FORM THE VARIABLES
C                          IN THE SYSTEM OF EQUATIONS FROM THE MEAN
C                          SQUARE ERROR FORMULATION
C     C------------------- A VECTOR WHOSE ELEMENTS FORM THE CONSTANTS
C                          IN THE SYSTEM OF EQUATIONS FROM THE MEAN
C                          SQUARE ERROR FORMULATION
C     OMEG1,OMEG2--------- DENOTE THE LOWER AND UPPER LIMITS RESP.
C                          FOR WHICH THE VARIABLE, OMEGA, IS CONSTANT
C
C************************************************************************
C
C
C
      A(1,1)=(OMEG2**3.-OMEG1**3.)/3.
      A(2,1)=(OMEG2-OMEG1)
      A(2,2)=-(1./OMEG2-1./OMEG1)
      A(3,1)=-(1./OMEG2-1./OMEG1)
      A(3,2)=-(1./(OMEG2**3.)-1./(OMEG1**3.))/3.
      A(3,3)=-(1./(OMEG2**5.)-1./(OMEG1**5.))/5.
      A(1,2)=A(2,1)
      A(1,3)=A(3,1)
      A(2,3)=A(3,2)
C
C
      C(1,1)=(OMEG2**2.-OMEG1**2.)/(2.*Q)
      C(2,1)=-(DLOG(OMEG2/OMEG1))/Q
      C(3,1)=-(1./(OMEG2**2.)-1./(OMEG1**2.))/(2.*Q)
C
C
C           TRIANGULARIZE [ A ]
      CALL DECOMP(A,WRK,3,3,1)
C
C           SOLVE THE SYSTEM OF EQUATIONS FOR { B }
      CALL SOLVEQ(WRK,C,B,3,3,1,1,1,1)
C
C
C
      RETURN
      END
```

```
      SUBROUTINE STIF3D(DMTRX,NX,NY,NZ,NGS,X,Y,Z,
     ,    NDIM,NDIM3,DRVTS,FNC,DNRM,BSAVE,WRK,NLST,
     ,    XX,YY,ZZ,AJAC,STIF1)
C
C                  ****************************************
C                  *                                      *
C                  *     P R O G R A M     S T I F 3 D    *
C                  *                                      *
C                  ****************************************
C
C-----------------------------------------------------------------------
C
C  A IN-CORE DOUBLE PRECISION PROGRAM TO CALCULATE THE ELEMENT STIFFNESS
C  MATRIX OF A 3-D COMPLEX ELEMENT OF ARBITRARY BRICK SHAPE WITH
C  BOUNDARY NODES ONLY (i.e. SERENDIPITY ELEMENTS); THE NUMBER OF NODES
C  ALONG EACH DIRECTION CAN BE GIVEN AS AN INPUT.
C
C  THE CONCEPT OF OBTAINING THE STIFFNESS MATRIX IS :
C        1. GENERATE INTERNAL NODAL POINTS FROM THE GIVEN BOUNDARY NODES
C           SUCH THAT IT FORMS A LAGRANGIAN ELEMENT
C        2. CALCULATE ALL OF THE SHAPE FUNCTIONS CORRESPONDING L. ELEM.
C        3. FORM STIFFNESS MATRIX OF THE L. ELEMENT BY USING GAUSS-
C           LEGENDRE QUADRATRUE INTEGRATION SCHEME.
C        4. CONDENCE OUT ALL OF THE INTERNAL NODES; THE STIFFNESS MATRIX
C           IS OBTAINED.
C
C
C-----------------------------------------------------------------------
C
C VARIABLE NAME LISTING :
C ***********************
C
C    STIF1 ...... LOCAL STIFFNESS MATRIX WITH DIMENSION 3*NDIM x 3*NDIM
C    DRVTS ...... MATRIX STORAGE WHICH CONTAINS THE DERIVATIVES OF THE
C                 NONLINEAR INTERPOLATION FUNCTION AT EACH GAUSS POINT.
C                 (i.e. IT HAS DIMENSION NDIM x 3 x 10;
C                       3 --- INDICATES DERIVATIVES AT EACH DIRECTION;
C                       10--- INDICATES THE NUMBER OF GAUSS POINTS.
C    DMTRX ...... A 6 x 6 MATRIX CONTAINS THE MATERIAL CONSTANT MATRIX.
C      FNC ...... MATRIX STORAGE WHICH CONTAINS THE VALUES OF THE
C                 NONLINEAR INTERPOLATION FUNCTIONS OBTAINED AT EACH
C                 GAUSS POINT. IT HAS SAME DIMENSION AS DRVTS.
C     DNRM ...... MATRIX STORAGE WITH DIMENSION NDIM x 3, WHICH CONTAINS
C                 THE VALUES OF THE NORMALIZATION FACTORS.
C    X,Y,Z ...... ARRAYS OF LENGTH NDIM, WHICH CONTAIN THE X, Y, AND Z
C                 COORDINATES OF NODES IN GLOBAL COORDINATES SYSTEM;
C                 WHERE NDIM(TOTAL NUMBER OF NODES)=NX*NY*NZ.
C XX,YY,ZZ ...... ARRAYS OF LENGTH NDIM, WHICH CONATINS A SET OF LOCAL
C                 COORDINATES.
C    BSAVE ...... STORAGE OF CALCULATED DERIVATIVES OF SHAPE FUNCTIONS
C                 WHICH ARE USED TO FORM THE [B]-MATRIX.
C      WRK ...... A 3*NDIM x 3*NDIM WORKING ARRAY FOR WORKING PACE.
C     WGHT ...... VECTOR OF LENGTH 10 CONTAINS THE WEIGHTS FOR EACH
C                 GAUSS POINT.
C      NGS ...... ACTUAL REQUIRED GAUSS POINTS.
C     NDIM ...... TOTAL NUMBER OF NODES
C    NDIM3 ...... INITIAL DIMENSION OF STIFFNESS MATRIX.(3*NDIM)
```

```
C     NLST ...... A LOCAL NODAL NUMBER LIST FOR REDUCING THE LAGRANGIAN
C                 ELEMENT STIFFNESS MATRIX TO SEREDIPITY ONE PROPERLY.
C     IRED ...... INDICATOR IF THE CONDENCING PROCEDURE IS REQUIRED;
C                 IRED = 0 ... NO CONDENCE; IRED = 1 ... CONDENCE REQ.
C NX,NY,NZ ...... NUMBER OF NODES ALONG X, Y, AND Z DIRECTION, RESPECT.
C    ITAPE ...... LOGICAL UNIT NUMBER ON WHICH THE STIFFNESS MATRIX OF
C                 LAGRANGIAN ELEMENT IS SAVE IF IT IS NECESSERARY. IF
C                 ITAPE = 0, NO SAVE IS DONE.
C-----------------------------------------------------------------------
C
C
       IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION STIF1(NDIM3,NDIM3),DRVTS(NDIM,3,10),DMTRX(6,6),
     +           FNC(NDIM,3,10),DNRM(NDIM,3),BSAVE(NDIM,3),X(NDIM),
     +           Y(NDIM),Z(NDIM),WRK(NDIM3,NDIM3),NLST(NDIM),WGHT(10),
     +           XX(NDIM),YY(NDIM),ZZ(NDIM)
C
C
C OBTAIN A LOCAL NODAL NUMBER LISTING --- BOUNDARY NODES IN THE FRONT
C FOLLOWED BY INTER NODES
C
      CALL LCNODE(NLST,NX,NY,NZ,NDIM)
C
C OBTAIN THE SHAPE FUNCTIONS
C
      CALL SHAPE1(XX,YY,ZZ,DRVTS,FNC,DNRM,NDIM,NX,NY,NZ,NGS,WRK,NDIM3,0)
C
C OBTAIN THE LOCAL STIFFNESS MATRIX
C
      CALL LCSTIF(STIF1,DRVTS,DMTRX,FNC,DNRM,BSAVE,X,Y,Z,WRK,NGS,NDIM,
     +            NDIM3,NLST,0,NX,NY,NZ,0,AJAC)
C
C
      RETURN
      END


C***********************************************************************
       SUBROUTINE BMTRX1(X,Y,Z,XX,YY,ZZ,BMTRX,NDIM,NX,NY,NZ,NGS,WRK,
     +                   VWRK,AJAC,IGEN)
C***********************************************************************
C
C  CALCULATE THE B-MATRIX BY USING GAUSS-LEGENDRE QUADRATURE FORMULA.
C
C      X ...... ARRAY OF GLOBAL NODAL COORDINATES IN X.
C      Y ...... ARRAY OF GLOBAL NODAL COORDINATES IN Y.
C      Z ...... ARRAY OF GLOBAL NODAL COORDINATES IN Z.
C     XX ...... ARRAY OF NORMAL (natural) NODAL COORDINATES IN XX.
C     YY ...... ARRAY OF NORMAL (natural) NODAL COORDINATES IN YY.
C     ZZ ...... ARRAY OF NORMAL (natural) NODAL COORDINATES IN ZZ.
C BMTRX ...... ARRAY OF DIMENSION (3*NPE x 3 x NGS), WHICH CONTAINS THE
C              EVALUATED B-MATRIX AT EACH GAUSS POINTS; Where
C              (NPE --- number of nodes per element)
C  NDIM ...... FIRST ROW DIMENSION OF X,Y,Z,XX,YY,ZZ AND BMTRX IN
C              CALLING PROGRAM.
C    NX ...... NUMBER OF NODES ALONG X.
C    NY ...... NUMBER OF NODES ALONG Y.
C    NZ ...... NUMBER OF NODES ALOGN Z.
C   NGS ...... NUMBER OF GAUSS POINTS REQUIRED IN NUMERICAL INTEGRATION.
```

```
C   WRK ...... WORKING SPACE OF LENGTH AT LEAST 3*NDIM.
C  VWRK ...... WORKING VECTOR OF LENGTH AT LEAST AS SAME AS X, Y, AND Z.
C  AJAC ...... ARRAY OF LENGTH NGS WITH JACOBIAN VALUES AT EACH GAUSS
C              POINTS. (i.e. determinent of JACOBIAN matrix)
C  IGEN ...... NODAL POINT COORDINATE GENERATIONS INDICATOR
C                  IGEN = 0    NORMAL COORDINATES ARE GENERATED;
C                  IGEN = 1    NORMAL COORDINATES ARE FROM INPUT.
C.................................................................
C
C  SUBROUTINES CALLED :
C
C                  DERVTS ---- CALCULATE DERIVATIVES.
C                  JACOBN ---- CALCULATE INVERSE OF JACOBIAN MATRIX, AND
C                              VALUE OF JACOBIAN.
C                  NCOORD ---- GENERATE A SET NORMAL NODAL COORDINATES.
C                  NUMINT ---- PERFORM THE NUMERICAL INTEGRATIONS.
C
C-----------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X(NDIM),Y(NDIM),Z(NDIM),XX(NDIM),YY(NDIM),ZZ(NDIM),
     +     BMTRX(NDIM,3,10),VWRK(NDIM),WRK(1),AJAC(10),
     +     DVX(3,10),DVY(3,10),DVZ(3,10),DFUNC(3),AJACB(3,3),
     +     XINT1(10),XINT2(10),XINT3(10),DVXX(10),DVYY(10),DVZZ(10)

C
      NPE=8+4*(NX+NY+NZ-6)
      IWRK1=0
      IWRK2=IWRK1+NDIM
      IWRK3=IWRK2+NDIM
C
C INITIALIZE XX, YY, ZZ, VWRK, WRK AND BMTRX.
C
      DO 2 I=1,NDIM
      DO 1 J=1,3
      DO 1 K=1,NGS
      BMTRX(I,J,K)=0.0
    1 CONTINUE
      IK1=IWRK1+I
      IK2=IWRK2+I
      IK3=IWRK3+I
      IF(IGEN.EQ.0) THEN
      XX(I)=0.0
      YY(I)=0.0
      ZZ(I)=0.0
      ENDIF
      WRK(IK1)=X(I)
      WRK(IK2)=Y(I)
      WRK(IK3)=Z(I)
      VWRK(I)=0.0
    2 CONTINUE
      DO 3 I=1,3
      DO 3 J=1,NGS
      DVX(I,J)=0.0
      DVY(I,J)=0.0
      DVZ(I,J)=0.0
    3 CONTINUE
C
C GENERATE NODAL COORDINATES IN NORMAL (NATURAL) COORDINATES
C IF IT IS REQUIRED.
```

```
C
      IF(IGEN.EQ.0) CALL NCOORD(XX,YY,ZZ,NX,NY,NZ,NDIM)
C
C CALCULATE THE NORMALIZED INTERPOLATION FUNCTIONS (I.F.) AND THE
C CORRESPONDING DERIVATIVES
C   dF/dXX=Sum(dF(i)/dXX), i=1,NPE;  etc....
C   where
C         F(i)=Fi(XX)*Fi(YY)*Fi(ZZ),              i.e.
C         dF(i)/dXX=Fi(YY)*Fi(ZZ)*(dFi(XX)/dXX)  etc.....
C
      DO 4 IPE=1,NPE                     !Evaluations node by node
        TRAS1=1.0
        TRAS2=1.0
        TRAS3=1.0
        IPTX=0
        IPTY=0
        IPTZ=0
        DO IXYZ=1,NPE
        X(IXYZ)=0.
        Y(IXYZ)=0.
        Z(IXYZ)=0.
        END DO
        DO 5 INOD=1,NPE
        IF(INOD.EQ.IPE) GO TO 5          !Only choose i .ne. j the term
C
C  EVALUATE NORMALIZATION FACTORS AND CHOOSE APPROPRIATE NODAL POINTS.
C
        IF(YY(INOD).EQ.YY(IPE).AND.ZZ(INOD).EQ.ZZ(IPE)) THEN
        IPTX=IPTX+1                      !Interpolation function along XX
        TERM1=XX(IPE)-XX(INOD)           !Normalization factor in XX
        TRAS1=TRAS1*TERM1
        X(IPTX)=XX(INOD)                 !XX-points of interpolation func
        ENDIF
        IF(XX(INOD).EQ.XX(IPE).AND.ZZ(INOD).EQ.ZZ(IPE)) THEN
        IPTY=IPTY+1                      !Interpolation function along YY
        TERM2=YY(IPE)-YY(INOD)           !Normalization factor in YY
        TRAS2=TRAS2*TERM2
        Y(IPTY)=YY(INOD)                 !YY-points of interpolation func
        ENDIF
        IF(XX(INOD).EQ.XX(IPE).AND.YY(INOD).EQ.YY(IPE)) THEN
        IPTZ=IPTZ+1                      !Interpolation function along ZZ
        TERM3=ZZ(IPE)-ZZ(INOD)           !Normalization factor in ZZ
        TRAS3=TRAS3*TERM3
        Z(IPTZ)=ZZ(INOD)                 !ZZ-points of interpolation func
        ENDIF
   5    CONTINUE
C
C PERFORM NUMERICAL INTEGRATIONS WITHOUT WEIGHTS (IWG=0)
C
      CALL NUMINT(X,NDIM,IPTX,NGS,XINT1,0)
      CALL NUMINT(Y,NDIM,IPTY,NGS,XINT2,0)
      CALL NUMINT(Z,NDIM,IPTZ,NGS,XINT3,0)
      DNRM=TRAS1*TRAS2*TRAS3
C
C CALCULATE DERIVATIVES ABOUT X, Y, AND Z.
C
      CALL DERVTS(DVXX,X,NDIM,IPTX,VWRK,NGS)
      CALL DERVTS(DVYY,Y,NDIM,IPTY,VWRK,NGS)
      CALL DERVTS(DVZZ,Z,NDIM,IPTZ,VWRK,NGS)
C
```

```
C  Saving the evaluated values
C
      IK1=IWRK1+IPE
      IK2=IWRK2+IPE
      IK3=IWRK3+IPE
      DO 10 IGS=1,NGS
      XINT23=XINT2(IGS)*XINT3(IGS)
      XINT13=XINT1(IGS)*XINT3(IGS)
      XINT12=XINT1(IGS)*XINT2(IGS)
      XXDV=DVXX(IGS)*XINT23/DNRM
      YYDV=DVYY(IGS)*XINT13/DNRM
      ZZDV=DVZZ(IGS)*XINT12/DNRM
      BMTRX(IPE,1,IGS)=XXDV
      BMTRX(IPE,2,IGS)=YYDV
      BMTRX(IPE,3,IGS)=ZZDV
C Calculate dX(XX,YY,ZZ)/dXX=Sum(X(i)*dF(i)/dXX), i=1,NPE     etc.....
C for each GAUSS point
C
      DVX(1,IGS)=DVX(1,IGS)+WRK(IK1)*XXDV
      DVX(2,IGS)=DVX(2,IGS)+WRK(IK1)*YYDV
      DVX(3,IGS)=DVX(3,IGS)+WRK(IK1)*ZZDV
      DVY(1,IGS)=DVY(1,IGS)+WRK(IK2)*XXDV
      DVY(2,IGS)=DVY(2,IGS)+WRK(IK2)*YYDV
      DVY(3,IGS)=DVY(3,IGS)+WRK(IK2)*ZZDV
      DVZ(1,IGS)=DVZ(1,IGS)+WRK(IK3)*XXDV
      DVZ(2,IGS)=DVZ(2,IGS)+WRK(IK3)*YYDV
      DVZ(3,IGS)=DVZ(3,IGS)+WRK(IK3)*ZZDV
   10 CONTINUE
    4 CONTINUE
C
C  Evaluate the inverse of JACOBIAN-Matrix and determinant of Jacobian
C  at each GAUSS points.
C
      NXYZ=6
      DO 11 IGS=1,NGS
        DO 12 IJB=1,3
        DVXX(IJB)=DVX(IJB,IGS)
        DVYY(IJB)=DVY(IJB,IGS)
        DVZZ(IJB)=DVZ(IJB,IGS)
   12 CONTINUE
      CALL JACOBN(DVXX,DVYY,DVZZ,AJAC1,AJACB,NXYZ)
      AJAC(IGS)=AJAC1
C
C  Form B-matrix at each GAUSS point in terms of X, Y, and Z values,
C  i.e. {dF(i)/dX(i)}=[AJACB]*{dF(i)/dXX(i)}, where
C        X(i) ..... Global coordinates;
C       XX(i) ..... Natural coordinates;
C         i   ..... the nodal numbers
C
      DO 6 IPE=1,NPE                          !Evaluate it node by node.
      DO 7 IDR=1,3
      DFUNC(IDR)=0.0
      DO 8 IDC=1,3
      DFUNC(IDR)=DFUNC(IDR)+AJACB(IDR,IDC)*BMTRX(IPE,IDC,IGS)
    8 CONTINUE
    7 CONTINUE
      BMTRX(IPE,1,IGS)=DFUNC(1)
      BMTRX(IPE,2,IGS)=DFUNC(2)
      BMTRX(IPE,3,IGS)=DFUNC(3)
    6 CONTINUE
```

```
   11  CONTINUE
C
C
       RETURN
       END

C                                                                       DEC00001
C***********************************************************************DEC00002
       SUBROUTINE DECOMP(SAVE,A,NROW,NDIM,ISYM)                         DEC00003
C***********************************************************************DEC00004
C                                                                       DEC00005
C  A DOUBLE PRECISION CODE WHICH PERFORMS THE LU-DECOMPOSITION OF THE   DEC00006
C  SQUARE MATRIX [A]; [A]=[L]*[U]. IF [A] IS SYMTRIC MATRIX (i.e.ISYM=1)DEC00007
C  THEN THE SYMTRIC MATRIX DECOMPOSITION IS PERFORMED; [A]=Trans[L]*[L].DEC00008
C                                                                       DEC00009
C-----------------------------------------------------------------------DEC00010
C     A ..... DECOMPOSED MATRIX     A=LU.                               DEC00011
C  SAVE.......INPUT MATRIX TO BE DECOMPOSED. THIS MATRIX REMAINS INTACT DEC0011A
C             UPON RETURN                                               DEC0011B
C  NROW ..... ROW DIMENSION OF MATRIX [A].                              DEC00012
C  NDIM ..... INITIAL DIMENSION OF MATRIX [A].                          DEC00013
C  ISYM ..... SYMETRIC INDICATOR; ISYM=0 ... NONSYM. BUT POSITIVE.      DEC00014
C                                 ISYM=1 ... SYMMET. AND POSITIVE.      DEC00015
C                                 ISYM=3 ... NONSYM. AND NONPOSIT.      DEC00016
C-----------------------------------------------------------------------DEC00017
C                                                                       DEC00018
       IMPLICIT REAL*8 (A-H,O-Z)                                        DEC00019
       DIMENSION A(NDIM,NDIM),SAVE(NDIM,NDIM)                           DEC00020
C                                                                       DEC00021
C SAVE ORIGINAL MATRIX BEFORE DECOMPOSITION                             DEC0021A
C                                                                       DEC0021B
       DO 6 I=1,NROW                                                    DEC0021C
       DO 6 J=1,NROW                                                    DEC0021D
    6  A(I,J)=SAVE(I,J)                                                 DEC0021E
C                                                                       DEC0021F
C START DECOMPOSITION.                                                  DEC00022
C                                                                       DEC00023
       DO 1 IELE=1,NROW                                                 DEC00024
C                                                                       DEC00025
C DETERMINE ELEMENT OF [L].                                             DEC00026
C                                                                       DEC00027
       DO 2 IROW=IELE,NROW                                              DEC00028
       CSUM=0.                                                          DEC00029
       IF(IELE.EQ.1) GO TO 997                                          DEC00030
       DO 3 ISUM=1,IELE-1                                               DEC00031
       CSUM=CSUM+A(IROW,ISUM)*A(ISUM,IELE)                              DEC00032
    3  CONTINUE                                                         DEC00033
  997  A(IROW,IELE)=A(IROW,IELE)-CSUM                                   DEC00034
       AA=A(IELE,IELE)                                                  DEC00035
       IF(ISYM.LT.3.AND.AA.LT.1.D-10) GO TO 998   !CHECK POSIT. AND SYM.DEC00036
       IF(DABS(AA).LT.1.D-10) GO TO 998           !CHECK IF PIVOT = 0.  DEC00037
       IF(ISYM.EQ.1) THEN                                               DEC00038
       IF(IROW.EQ.IELE) THEN                                            DEC00039
       A(IELE,IELE)=DSQRT(AA)                                           DEC00040
       ELSE                                                             DEC00041
       A(IROW,IELE)=A(IROW,IELE)/A(IELE,IELE)                           DEC00042
       ENDIF                                                            DEC00043
       ENDIF                                                            DEC00044
    2  CONTINUE                                                         DEC00045
```

```
C
C DETERMINE ELEMENTS OF [U]
C
      IF(IELE.EQ.NROW) GO TO 1
      DO 4 JCOL=IELE+1,NROW
      RSUM=0.0
      IF(ISYM.EQ.1) GO TO 995
      IF(IELE.EQ.1) GO TO 996
      DO 5 ISUM=1,IELE-1
      RSUM=RSUM+A(IELE,ISUM)*A(ISUM,JCOL)
    5 CONTINUE
 996  A(IELE,JCOL)=(A(IELE,JCOL)-RSUM)/A(IELE,IELE)
      GO TO 4
 995  A(IELE,JCOL)=A(JCOL,IELE)
    4 CONTINUE
    1 CONTINUE
C
      RETURN
  998 CONTINUE
      IF(AA.LT.0.) THEN
      PRINT 1000, IROW, AA
      ELSE
      PRINT 1001, IROW
      ENDIF
 1000 FORMAT(/,' **** ERROR : MATRIX IS NON-POSITIVE DEFINITE ...',/,
     +         '              EQUATION NUMBER ', I10,/,
     +         '              PIVOTING VALUE =', D10.4,/)
 1001 FORMAT(/,' **** ERROR : MATRIX IS NEARLY SINGULAR ...',/,
     +         '              EQUATION NUMBER ',I10,/)
      STOP
      END
C
C***********************************************************************
      SUBROUTINE DERVTS(DVXI,XX,NDIM,NPT,WRK,NGS,NWRK,ICOL)
C***********************************************************************
C
C  CALCULATE THE DERIVATIVE (Numerically) OF INTERPOLATION FUNCTIONS X
C      BY USING GAUSS-LEGENDRE QUADRATURE FORMULA.
C
C   DVXI ...... ARRAY OF LENGTH NGS, THE DERIVATIVES OF THE
C               INTERPOLATION FUNCTION AT EACH GAUSS POINTS.
C     XX ...... ARRAY OF INITIAL LENGTH NDIM, CONTAINS THE NODAL
C               COORDINATES IN THE INTERPOLATION FUNCTION (i.e. In
C               normal coordinate system).
C   NDIM ...... INITIAL DIMENSION OF ARRAY X.
C    NPT ...... ACTUAL LENGTH OF ARRAY X.
C    WRK ...... WORKING SPACE WITH DIMENSION NWRK X NWRK.
C    NGS ...... NUMBER OF GAUSS POINTS QUIRED.
C   NWRK ...... INITIAL DIMENSION OF MATRIX WRK.
C   ICOL ...... I-TH COLUMN, WHICH WILL BE USED AS A WORKING SPACE.
C
C  SUBROUTINES CALLED :
C                      NUMINT ----- A NUMERICAL INTERATION ROUTINE.
C
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION XX(NDIM),WRK(NWRK,NWRK),DVXI(10),DVXJ(10)
C
C  INITINALIZE WRK
```

```
C
      DO 1 I=1,NPT
      WRK(I,ICOL)=XX(I)
    1 CONTINUE
      DO 2 I=1,10
      DVXI(I)=0.0                      !Initialize dFi(XX)/dXX.
    2 CONTINUE
C
C    Fi(xx)=(XX-XX(1))*(XX-XX(2))*......*(XX-XX(i-1))*(XX-XX(i+1))*....
C  hence :
C    dFi(XX)/dXX=Sum[dFij(XX)/dXX],jj=1,NPT;
C  there
C    dFij(XX)/dXX=(XX-XX(1))*(XX-XX(2))*......*(XX-XX(j-1))*(XX-XX(j+1)*
C                 ......*(XX-XX(i-1))*(XX-XX(i+1)*......
C
        DO 3 ITERM=1,NPT               !Calculate dFij(XX)/dXX.
        IPT=0
          DO 4 IX=1,NPT                !Choose proper XX-points.
          IF(IX.EQ.ITERM) GO TO 4      !Skip XX(i) and XX(j).
          IPT=IPT+1
          XX(IPT)=WRK(IX,ICOL)
    4     CONTINUE
        CALL NUMINT(XX,NDIM,IPT,NGS,DVXJ,0)
        DO 5 IG=1,NGS
        DVXI(IG)=DVXI(IG)+DVXJ(IG)      !Sum of dFij(XX)/dXX in j.
    5   CONTINUE
    3 CONTINUE
C
C
      RETURN
      END
C
C***********************************************************************
      SUBROUTINE FORMVK(VWRK,VWK)
C***********************************************************************
C
C  FORM AN APPROPERIATE B-MTRIX FOR INTERPOLATION FUNCTION # INOD.
C
C    VWRK ...... A SUB-MATRIX OF B-MATRIX FOR NODAL NUMBER (INOD) WITH
C                DIMENSION 6 x 3.
C     VWK ...... AN ARRAY OF LENGTH 3, WHICH CONTAINS THE DERIVATIVES OF
C                THE INTERPOLATION FUNCTIONS WITH RESPECT TO X, Y, AND Z
C                IN GLOBAL COORDINATE SYSTEM.
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION VWRK(6,3),VWK(3)
C
C  INITIALIZE MATRIX VWRK.
C
      DO 1 IVK=1,6
      DO 1 JVK=1,3
    1 VWRK(IVK,JVK)=0.0
C
C  FORM VWRK SUB-MATRIX.
C
      VWRK(1,1)=VWK(1)
      VWRK(2,2)=VWK(2)
      VWRK(3,3)=VWK(3)
      VWRK(4,1)=VWRK(2,2)
```

```
      VWRK(4,2)=VWRK(1,1)
      VWRK(5,1)=VWRK(3,3)
      VWRK(5,3)=VWRK(1,1)
      VWRK(6,2)=VWRK(3,3)
      VWRK(6,3)=VWRK(2,2)
C
C  FINISH
C
      RETURN
      END
C
C***********************************************************************
      SUBROUTINE GCOORD(X,Y,Z,NX,NY,NZ,NDIM,NLST)
C***********************************************************************
C
C  GENERATE A SET OF NODAL COORDINATES, WHICH ARE NOT ON THE ELEMENT
C  EDGES, IN GLOBAL COORDINATES SYSTEM IN ORDER TO FORM A LAGRANGIAN
C  ELEMENTS FROM THE GIVEN SERENDIPITY ELEMENTS.
C
C-----------------------------------------------------------------------
C    X,Y,Z ...... NODAL COORDINATES IN X, Y, AND Z-DIRECTIONS. AS INPUT
C                 THEY MUST CONTAIN THE COORDINATES OF THE NODES WHICH
C                 ARE ON THE EDGES IN PROPER SEQUENCE.
C NX,NY,NZ ...... NUMBER OF NODES IN X, Y, AND Z-DIRECTIONS.
C     NDIM ...... INITIAL DIMENSION OF X, Y, AND Z VECTORS.
C     NLST ...... LOCAL NODAL NUMBER LISTING.
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X(NDIM),Y(NDIM),Z(NDIM),NLST(NDIM)
C
      NPE1=8+4*(NX+NY+NZ-6)                 !NUMBER OF ELEMENTS ON EDGES
      IPT1=NPE1+1
      NNX=NX
      NNY=NY
      NNZ=NZ
      IF(NNX.LE.0) NNX=1
      IF(NNY.LE.0) NNY=1
      IF(NNZ.LE.0) NNZ=1
C
C GENERATE NODES
C
      NZ1=2*(NX+NY-2)
      NZ2=NZ1+4*(NZ-2)
      RATIO=1./DFLOAT(NZ-1)
      DO 1 IZ=1,NZ
      IZ1=NZ1+4*(IZ-2)                      !STARTING POINT
      IF(IZ.EQ.1) IZ1=NX
      IF(IZ.EQ.NZ) IZ1=IZ1+NX
      DX0=0.
      DY0=0.
      DZ0=0.
      DX1=0.
      DY1=0.
      DZ1=0.
      IF(IZ.GT.1.AND.IZ.LT.NZ) THEN
      DX0=(X(IZ1+3)-X(IZ1+1))*RATIO
      DX1=(X(IZ1+4)-X(IZ1+2))*RATIO
      DY0=(Y(IZ1+3)-Y(IZ1+1))*RATIO
      DY1=(Y(IZ1+4)-Y(IZ1+2))*RATIO
```

```
      DZ0=(Z(IZ1+3)-Z(IZ1+1))*RATIO
      DZ1=(Z(IZ1+4)-Z(IZ1+2))*RATIO
      ENDIF
      ITY1=IZ1+1
      IEY1=ITY1+1
      DO 2 IY=1,NNY
      IF((IZ.EQ.1.OR.IZ.EQ.NZ).AND.IY.EQ.NNY) GO TO 2
      IF((IZ.EQ.1.OR.IZ.EQ.NZ).AND.IY.EQ.1) GO TO 2
      IF(IY.EQ.1.OR.IY.EQ.NY) THEN
      IF(IY.EQ.1) ITY=IZ1+1
      IF(IY.EQ.NY) ITY=IZ1+3
      IEY=ITY+1
      X0=X(ITY)
      Y0=Y(ITY)
      Z0=Z(ITY)
      X1=X(IEY)
      Y1=Y(IEY)
      Z1=Z(IEY)
      ELSE
      X0=X(ITY1)+DFLOAT(IY-1)*DX0
      Y0=Y(ITY1)+DFLOAT(IY-1)*DY0
      Z0=Z(ITY1)+DFLOAT(IY-1)*DZ0
      X1=X(IEY1)+DFLOAT(IY-1)*DX1
      Y1=Y(IEY1)+DFLOAT(IY-1)*DY1
      Z1=Z(IEY1)+DFLOAT(IY-1)*DZ1
      IF(IZ.EQ.1.OR.IZ.EQ.NZ) THEN
      ITY1=IEY1+1
      IEY1=ITY1+1
      ENDIF
      ENDIF
      X2=X0
      X3=X1
      DX=(X1-X0)/DFLOAT(NNX-1)
      DY=(Y1-Y0)/DFLOAT(NNY-1)
      DZ=(Z1-Z0)/DFLOAT(NNZ-1)
      IF(IZ.EQ.1.OR.IZ.EQ.NZ) GO TO 89
      IF(IY.GT.1.AND.IY.LT.NY) THEN
      X2=X2-DX
      X3=X3+DX
      ENDIF
   89 CONTINUE
      DO 3 IX=1,NNX
      DDX1=X0-X2
      DDX2=X0-X3
      IF(DABS(DDX1).LE.1.D-8.OR.DABS(DDX2).LE.1.D-8) GO TO 9
      X(IPT1)=X0
      Y(IPT1)=Y0
      Z(IPT1)=Z0
      IPT1=IPT1+1
    9 X0=X0+DX
      Y0=Y0+DY
      Z0=Z0+DZ
    3 CONTINUE
    2 CONTINUE
    1 CONTINUE
C
      RETURN
      END
C
C***********************************************************************
```

```
      SUBROUTINE JACOBN(DX,DY,DZ,AJAC,AJACB,NXYZ)
C***********************************************************************
C
C  PROGRAM TO EVALUATE THE JACOBIAN - MATRIX AND ITS DETERMINANT.
C
C    DX ...... DERIVATIVES WITH RESPECT TO X, DIMENSION 3.
C    DY ...... DERIVATIVES WITH RESPECT TO Y, DIMENSION 3.
C    DZ ...... DERIVATIVES WITH RESPECT TO Z, DIMENSION 3.
C  AJAC ...... DETERMINANT OF JACOBIAN MATRIX
C AJACB ...... THE INVERSE JACOBIAN MATRIX.
C  NXYZ ...... THE INITIAL DIMENSION OF ARRAYS DX, DY, AND DZ.
C
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DX(NXYZ),DY(NXYZ),DZ(NXYZ),AJACB(3,3)
C
C  Evaluate the determinant of JACOBIAN - matrix.
C
      AJAC=DX(3)*(DY(1)*DZ(2)-DY(2)*DZ(1))-(DY(3)*(DX(1)*DZ(2)-
     +     DX(2)*DZ(1)))+DZ(3)*(DX(1)*DY(2)-DX(2)*DY(1))
C
C  Evaluate the inverse of JACOBIAN - matrix.
C
      AJACB(1,1)=(DY(2)*DZ(3)-DY(3)*DZ(2))/AJAC
      AJACB(1,2)=(DY(3)*DZ(1)-DY(1)*DZ(3))/AJAC
      AJACB(1,3)=(DY(1)*DZ(2)-DY(2)*DZ(1))/AJAC
      AJACB(2,1)=(DX(3)*DZ(2)-DX(2)*DZ(3))/AJAC
      AJACB(2,2)=(DX(1)*DZ(3)-DX(3)*DZ(1))/AJAC
      AJACB(2,3)=(DX(2)*DZ(1)-DX(1)*DZ(2))/AJAC
      AJACB(3,1)=(DX(2)*DY(3)-DX(3)*DY(2))/AJAC
      AJACB(3,2)=(DX(3)*DY(1)-DX(1)*DY(3))/AJAC
      AJACB(3,3)=(DX(1)*DY(2)-DX(2)*DY(1))/AJAC
C
      RETURN
      END
C
C***********************************************************************
      SUBROUTINE LCNODE(NLST,NX,NY,NZ,NDIM)
C***********************************************************************
C
C  THIS SUBROUTINE GENERATES A NODAL NUMBER LIST OF A LAGURANGIAN
C  ELEMENT SUCH THAT THE FIRST "NSE" NUMBERS ARE IN THE SEQUENCE OF A
C  SERENDIPITY ELEMENT AND FOLLOWED BY THE "OFF EDGE" NODAL NUMBERS.
C
C-----------------------------------------------------------------------
C      NLST ...... ARRAY; AFTER RETURN IT CONTAINS THE RESULT.
C  NX,NY,NZ ...... NUMBER OF NODES ALONG X, Y, AND Z DIRECTION.
C      NDIM ...... INITIAL DIMENSION OF ARRAY NLST.
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION NLST(NDIM)
C
      NSE=8+4*(NX+NY+NZ-6)
      NLE=NX*NY*NZ
C
C INITIALIZE NLST
C
      DO 1 I=1,NDIM
```

```
 1      NLST(I)=0
C
C NUMBERING START
C
C 1. FACE AT IZ=1, AND IZ=NZ
C
        IZ1=0
        INOD1=1
        IZN=NSE-2*(NX+NY-2)
        INODZ=NLE-NX*NY+1
        ISE1=NSE
        ISEZ=INODZ+2*(NX+NY-2)-1
        DO 2 IY=1,NY
        DO 3 IX=1,NX
        IF((IY.GT.1.AND.IY.LT.NY).AND.(IX.GT.1.AND.IX.LT.NX)) GO TO 900
        IZ1=IZ1+1
        IZN=IZN+1
        NLST(IZ1)=INOD1
        NLST(IZN)=INODZ
        GO TO 901
  900   ISE1=ISE1+1
        ISEZ=ISEZ+1
        NLST(ISE1)=INOD1
        NLST(ISEZ)=INODZ
  901   INOD1=INOD1+1
        INODZ=INODZ+1
    3   CONTINUE
    2   CONTINUE
        IF(NZ.LE.2) RETURN
        DO 4 IZ=2,NZ-1
        DO 5 IY=1,NY
        DO 6 IX=1,NX
        IF(IY.GT.1.AND.IY.LT.NY) GO TO 902
        IF(IX.GT.1.AND.IX.LT.NX) GO TO 902
        IZ1=IZ1+1
        NLST(IZ1)=INOD1
        GO TO 903
  902   ISE1=ISE1+1
        NLST(ISE1)=INOD1
  903   INOD1=INOD1+1
    6   CONTINUE
    5   CONTINUE
    4   CONTINUE
C
        RETURN
        END
C
C***********************************************************************
        SUBROUTINE LCSTIF(STIF1,DRVTS,DMTRX,FNC,DNRM,BSAVE,X,Y,Z,WRK,NGS,
     +                   NDIM,NDIM3,NLST,IRED,NX,NY,NZ,ITAPE,AJAC)
C***********************************************************************
C
C  A DOUBLE PRECISION IN-CORE PROGRAM TO FORM THE LOCAL STIFFNESS MATRIX
C  OF COMPLEX CUBIC ELEMENTS. AT THE FIRST, THE STIFFNESS MATRIX IS
C  FORMED FOR LAGRANGE ELEMENT, AND THEN A REDUCTION IS PERFROMED TO
C  REDUCE THE L.G. STIFFNESS MATRIX TO A SERENDIPITY ELEMENT STIFFNESS
C  MATRIX, (i.e. NODAL POINTS ON THE EDGES OF THE CUBIC ONLY.).
C
C-----------------------------------------------------------------------
C    STIFF ...... LOCAL STIFFNESS MATRIX WITH DIMENSION 3*NOD x 3*NOD
```

```
C    DRVTS ...... MATRIX STORAGE WHICH CONTAINS THE DERIVATIVES OF THE
C                 NONLINEAR INTERPOLATION FUNCTION AT EACH GAUSS POINT.
C                 (i.e. IT HAS DIMENSION NOD x 3 x 6;
C                       3 --- INDICATES DERIVATIVES AT EACH DIRECTION;
C                       6 --- INDICATES THE NUMBER OF GAUSS POINTS.
C    DMTRX ...... A 6 x 6 MATRIX CONTAINS THE MATERIAL CONSTANT MATRIX.
C      FNC ...... MATRIX STORAGE WHICH CONTAINS THE VALUES OF THE
C                 NONLINEAR INTERPOLATION FUNCTIONS OBTAINED AT EACH
C                 GAUSS POINT. IT HAS SAME DIMENSION AS DRVTS.
C     DNRM ...... MATRIX STORAGE WITH DIMENSION NOD x 3, WHICH CONTAINS
C                 THE VALUES OF THE NORMALIZATION FACTORS.
C    X,Y,Z ...... ARRAYS OF LENGTH NOD, WHICH CONTAIN THE X, Y, AND Z
C                 COORDINATES OF NODES IN GLOBAL COORDINATES SYSTEM;
C                 WHERE NOD=NX*NY*NZ.
C    BSAVE ...... STORAGE OF CALCULATED DERIVATIVES OF SHAPE FUNCTIONS
C                 WHICH ARE USED TO FORM THE [B]-MATRIX.
C      WRK ...... A 3*NOD x 3*NOD WORKING ARRAY FOR WORKING PACE.
C     WGHT ...... VECTOR WITH LENGTH 6 CONTAINS THE WEIGHTS FOR EACH
C                 GAUSS POINT.
C      NGS ...... ACTUAL REQUIRED GAUSS POINTS.
C     NDIM ...... INITIAL DIMENSION OF X, Y, AND Z.
C    NDIM3 ...... INITIAL DIMENSION OF STIFFNESS MATRIX.
C     NLST ...... A LOCAL NODAL NUMBER LIST FOR REDUCING THE LAGRANGIAN
C                 ELEMENT STIFFNESS MATRIX TO SEREDIPITY ONE PROPERLY.
C     IRED ...... INDICATOR IF THE CONDENCING PROCEDURE IS REQUIRED;
C                 IRED = 0 ... NO CONDENCE; IRED = 1 ... CONDENCE REQ.
C NX,NY,NZ ...... NUMBER OF NODES ALONG X, Y, AND Z DIRECTION, RESPECT.
C    ITAPE ...... LOGICAL UNIT NUMBER ON WHICH THE STIFFNESS MATRIX OF
C                 LAGRANGIAN ELEMENT IS SAVE IF IT IS NECESSERARY. IF
C                 ITAPE = 0, NO SAVE IS DONE.
C-----------------------------------------------------------------------
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION STIF1(NDIM3,NDIM3),DRVTS(NDIM,3,10),DMTRX(6,6),
     +          FNC(NDIM,3,10),DNRM(NDIM,3),BSAVE(NDIM,3),X(NDIM),
     +          Y(NDIM),Z(NDIM),WRK(NDIM3,NDIM3),NLST(NDIM),WGHT(10)
C
      NOD=NX*NY*NZ                    !FIND THE TOTAL NUMBER OF NODES.
      NOD3=3*NOD
      NSE=8+4*(NX+NY+NZ-6)            !FIND # NODES OF SEREN. ELMEMENT.
      NSE3=3*NSE
C
C  OBTAIN THE WEIGHTS CORRESPONDING TO THE REQUIRED NUMBER OF GAUSS
C  POINTS.
C
      CALL NUMINT(X,NDIM,NX,NGS,WGHT,1)
C
C  OBTAIN THE GLOBAL COORDINATES OF A LAGRANGIAN ELEMENT FROM THE GIVEN
C  COORDINATES OF THE BOUNDARY NODES.
C
      CALL GCOORD(X,Y,Z,NX,NY,NZ,NDIM,NLST)
C
C  RENUMBING THE GLOBAL COORDINATES CALCULATED FROM ABOVE.
C
      CALL RENUMB(X,Y,Z,NX,NY,NZ,NDIM,0,NLST)
C
C  EVALUATE THE LOCAL STIFFNESS MATRIX IN LAGRANGIAN ELEMENT FORM.
C
      CALL STIFF1(WRK,DRVTS,DMTRX,FNC,DNRM,STIF1,WGHT,NGS,NOD,NDIM3,
```

```
     +             NDIM,X,Y,Z,BSAVE,AJAC)
C
      IF(ITAPE.NE.0) WRITE(ITAPE) WRK
      IF(IRED.EQ.0) THEN
      DO 1 I=1,NOD3
      DO 1 J=1,NOD3
      STIF1(I,J)=WRK(I,J)
      STIF1(I,I)=DABS(WRK(I,I))
      IF(I.NE.J) STIF1(J,I)=STIF1(I,J)
    1 CONTINUE
C
      ELSE
C
C   CONDENCE OUT THE INTERIOR NODES FROM [STIF1] SUCH THAT AT RETURN
C   THE STIFFNESS MATRIX REPRESENTS THE BOUNDARY NODES ONLY.
C
C          T [K11], [K12] T                                  -1
C     [K] = |              | ;       [K'] = [K11] - [K12]*[K22]   *[K21]
C          | [K21], [K22] |
C          -              -
C
C   REFORM THE STIFFNESS MATRIX SO THAT THE CONDENSING PROCEDURE CAN BE
C   DONE CORRECTLY.
C
      CALL REFORM(WRK,STIF1,NDIM3,NLST,NDIM,NOD)
      CALL SWITCH(WRK,STIF1,NDIM3,NOD,NSE,0)
C                                                   -1
C   FORM A R-H-S MATRIX IN ORDER TO CALCULATE [K22]   *(K21].
C
      NK22=NOD3-NSE3
      DO 3 I=1,NK22
      DO 3 J=1,NSE3
      JK21=NK22+J
      STIF1(I,J)=WRK(I,JK21)
    3 CONTINUE
C                    -1
C   CALCULATE [K22]   *[K21].
C
      CALL SOLVEQ(WRK,STIF1,NK22,NDIM3,NSE3,NDIM3,1,0)
C
C                                          -1
C   CALCULATE [K'] = [K11] - [K12]*[K22]   *[K21].
C
      DO 4 I=1,NSE3
      IK12=NK22+I
      IK11=IK12
      DO 5 J=1,NSE3
      SUM=0.0
      JK11=NK22+J
      DO 6 K=1,NK22
      SUM=SUM+WRK(IK12,K)*STIF1(K,J)
    6 CONTINUE
      WRK(IK11,JK11)=WRK(IK11,JK11)-SUM
      IF(DABS(WRK(IK11,JK11)).LT.1.D-10) WRK(IK11,JK11)=0.0
    5 CONTINUE
    4 CONTINUE
C
C   PUT [STIF1] BACK IN CORRECT ORDER.
C
      DO 7 I=1,NSE3
```

```
        IK11=NK22+I
        DO 8 J=I,NSE3
        JK11=NK22+J
        STIF1(I,J)=WRK(IK11,JK11)
        STIF1(J,I)=STIF1(I,J)
        STIF1(I,I)=DABS(STIF1(I,I))
    8   CONTINUE
    7   CONTINUE
C
        ENDIF
C   PROGRAM ENDS
C
        RETURN
        END
C
C
      SUBROUTINE MPRNT(A,NR,NC,NCOL,NRD,NCD,NOUT)
C---- PRINT A REAL MATRIX OF SIZE A(NR,NC)
      REAL*8  A(NRD,NCD)
      DO 100 J=1,NC,NCOL
      IF(NCOL.EQ.8) JH=J+7
      IF(NCOL.EQ.4) JH=J+3
      IF(JH-NC) 75,75,50
50    JH=NC
75    IF(NCOL.EQ.4) WRITE(NOUT,4000) (N,N=J,JH)
      IF(NCOL.EQ.8) WRITE(NOUT,3000) (N,N=J,JH)
      DO 100 I=1,NR
      IF(NCOL.EQ.4) WRITE(NOUT,4001) I, (A(I,K),K=J,JH)
      IF(NCOL.EQ.8) WRITE(NOUT,3001) i, (A(i,K),K=J,JH)
100   continue
      RETURN
3000  FORMAT(/8X,8I15)
3001  FORMAT(1X,I4,3x,8d15.6)
4000  FORMAT(/8X,4I15)
4001  FORMAT(1X,I4,3X,4D15.6)
      END
C
C*************************************************************************
      SUBROUTINE NCOORD(X1,Y1,Z1,NX,NY,NZ,NDIM)
C*************************************************************************
C
C   PROGRAM TO GENERATE A SET OF NODAL COORDINATES IN NATURAL (NORMAL)
C   COORDINATES IN 3-D WITH ALL NODAL POINTS ON THE EDGES ONLX.
C
C   X1,Y1,Z1 ........ ARRAXS OF ELEMENT NODAL COORDINATES (NATURE).
C   NX,NY,NZ ........ NUMBERS OF NODES ALONG Z,X, AND Y DIRECTIONS.
C       NDIM ........ INITIAL DIMESION OF ARRAXS Z1,X1,AND Y1.
C-------------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X1(NDIM),Y1(NDIM),Z1(NDIM)

C
C   DETERMINE THE INCREMENTS OF X1,Y1,AND Z1.
C   IN NORMAL COORDINATES   -1<= X1,Y1,Z1 <= 1.
C   LENGTH OF EACH EDGES = 2.
C
      IF(NX.GT.1) DX1=2./DFLOAT(NX-1)
      IF(NY.GT.1) DY1=2./DFLOAT(NY-1)
      IF(NZ.GT.1) DZ1=2./DFLOAT(NZ-1)
```

```
C
C  CALCULATE Z,X,AND Y VALUES IN THE NATURAL (NORMAL) COORDINATES
C
      IPT=0
      NNZ=NZ
      NNY=NY
      NNX=NX
      IF(NZ.EQ.0) NNZ=1
      IF(NY.EQ.0) NNY=1
      IF(NX.EQ.0) NNX=1
C  START NODAL POINT NUMBERING.
      IF(NZ.GE.1) Z0=-1.-DZ1
      DO 1 IZ=1,NNZ
      Z0=Z0+DZ1
      IF(NY.GE.1) Y0=-1.-DY1
      DO 2 IY=1,NNY
      Y0=Y0+DY1
      IF(NX.GE.1) X0=-1.-DX1
      DO 3 IX=1,NNX
      IPT=IPT+1
      X0=X0+DX1
      X1(IPT)=X0
      Y1(IPT)=Y0
      Z1(IPT)=Z0
    3 CONTINUE
    2 CONTINUE
    1 CONTINUE
C
      RETURN
      END
C
C*********************************************************************
      SUBROUTINE NUMINT(X,NDIM,NX,NGS,XR,IWG)
C*********************************************************************
C
C  NUMERICAL INTEGRATION BY QAUSS-LEGENDRE QUADRATURE FORMULA FOR
C  INTERPOLATION FUNCTIONS IN NATURAL (NORMAL) COORDINATES ONLY.
C
C     X ........ ARRAY OF LENGTH NX, WHICH CONTAINS THE NODAL
C  NDIM ........ INITIAL DIMENSION OF ARRAY X.
C    NX ........ ACTUAL LENGTH OF ARRAY X.(TOT # OF NODES)
C   NGS ........ NUMBER OF GAUSS POINTS REQUIRED FOR THE INTEGRATIONS.
C                IT HAS TO BE ONE OF THE FOLLOWING :
C                NGS = 2, 3, 4, 5, 6,7,8,9,10.
C    XR ........ ARRAY OF LENGTH NGS, WHICH CONTAIN THE INTEGRATED
C                VALUES AT EACH GAUSS POINTS.
C   IWG ........ EVALUATION INDICATOR
C                IWG = 0    ----- EVALUATE FUNCTION AT GAUSS POINT ONLY
C                                 (i.e. without multiplying the WEIGHTS.
C                IWG = 1    ----- TRANSFER THE VALUES OF WEIGHTS INTO
C                                 XR.
C                IWG = 2    ----- PERFORM THE NUMERICAL INTEGRATION
C                                 COMPLETELY - i.e. G=Sum(wt(i)F(gs(i))]
C
C---------------------------------------------------------------------
C
       IMPLICIT REAL*8 (A-H,O-Z)
       DIMENSION GS2(2),WT2(2),GS3(3),WT3(3),GS4(4),WT4(4),
     +           GS5(5),WT5(5),GS6(6),WT6(6),GS7(7),WT7(7),
     +           GS8(8),WT8(8),GS9(9),WT9(9),GS10(10),
```

```
     +          WT10(10),X(NDIM),XR(10)
C
C   DATA BLOCKS OF GAUSS POINTS AND CORRESPONDING WEIGHTS.
C
C   1. DATA BLOCK FOR TWO-POINT GAUSS QUADRATURE
C
        DATA GS2/-0.5773502692, 0.5773502692/
        DATA WT2/ 1.0000000000, 1.0000000000/
C
C   2. DATA BLOCK FOR THREE-POINT GAUSS QUADRATURE
C
        DATA GS3/-0.7745966692, 0.0000000000, 0.7745966692/
        DATA WT3/ 0.5555555555, 0.8888888889, 0.5555555555/
C
C   3. DATA BLOCK FOR FOUR-POINT GAUSS QUADRATURE
C
        DATA GS4/-0.8611363116,-0.3399810435, 0.3399810435,
     +          0.8611363116/
        DATA WT4/ 0.3478548451, 0.6521451548, 0.6521451548,
     +          0.3478548451/
C
C   4. DATA BLOCK FOR FIVE-POINT GAUSS QUADRATURE
C
        DATA GS5/-0.9061798459,-0.5384693101, 0.0000000000,
     +          0.5384693101, 0.9061798459/
        DATA WT5/ 0.2369268850, 0.4786286705, 0.5688888889,
     +          0.4786286705, 0.2369268850/
C
C   5. DATA BLOCK FOR SIX-POINT GAUSS QUADRATURE
C
        DATA GS6/-0.9324695142,-0.6612093865,-0.2386191861,
     +          0.2386191861, 0.6612093865, 0.9324695142/
        DATA WT6/ 0.1713244924, 0.3607615730, 0.4679139346,
     +          0.4679139346, 0.3607615730, 0.1713244924/
C
C   6. DATA BLOCK FOR SEVEN-POINT GAUSS QUADRATURE
C
        DATA GS7/-0.9491079123,-0.7415311856,-0.4058451514,
     +          0.0000000000, 0.4058451514, 0.7415311856,
     +          0.9491079123/
        DATA WT7/ 0.1294849662, 0.2797053915, 0.3818300505,
     +          0.4179591837, 0.3818300505, 0.2797053915,
     +          0.1294849662/
C
C   7. DATA BLOCK FOR EIGHT-POINT GAUSS QUADRATURE
C
        DATA GS8/-0.9602898565,-0.7966664774,-0.5255324099,
     +         -0.1834346425, 0.1834346425, 0.5255324099,
     +          0.7966664774, 0.9602898565/
        DATA WT8/ 0.1012285363, 0.2223810345, 0.3137066459,
     +          0.3626837834, 0.3626837834, 0.3137066459,
     +          0.2223810345, 0.1012285363/
C
C   8. DATA BLOCK FOR NINE-POINT GAUSS QUADRATURE
C
        DATA GS9/-0.9681602395,-0.8360311073,-0.6133714327,
     +         -0.3242534234, 0.0000000000, 0.3242534234,
     +          0.6133714327, 0.8360311073, 0.9681602395/
        DATA WT9/ 0.0812743884, 0.1806481607, 0.2606106964,
     +          0.3123470770, 0.3302393550, 0.3123470770,
```

```
     +            0.2606106964, 0.1806481607, 0.0812743884/
C
C  9. DATA BLOCK FOR TEN-POINT GAUSS QUADRATURE
C
      DATA GS10/-0.9739065285,-0.8650633667,-0.6794095683,
     +          -0.4333953941,-0.1488743390, 0.1488743390,
     +           0.4333953941, 0.6794095683, 0.8650633667,
     +           0.9739065285/
      DATA WT10/ 0.0666713443, 0.1494513492, 0.2190863625,
     +           0.2692667193, 0.2955242247, 0.2955242247,
     +           0.2692667193, 0.2190863625, 0.1494513492,
     +           0.0666713443/
C
C ----------------------- END OF DATA BLOCKS -------------------------
C
      IF(NX.LT.0) GO TO 99
      IDN=IWG-1
C
C  START EVALUATION PROCEDURE (POINT BY POINT) :
C
      DO 1 IG=1,NGS
        XX=1.0
        IF(NGS.EQ.2) THEN
          WT=WT2(IG)
          GS=GS2(IG)
        ENDIF
        IF(NGS.EQ.3) THEN
          WT=WT3(IG)
          GS=GS3(IG)
        ENDIF
        IF(NGS.EQ.4) THEN
          WT=WT4(IG)
          GS=GS4(IG)
        ENDIF
        IF(NGS.EQ.5) THEN
          WT=WT5(IG)
          GS=GS5(IG)
        ENDIF
        IF(NGS.EQ.6) THEN
          WT=WT6(IG)
          GS=GS6(IG)
        ENDIF
        IF(NGS.EQ.7) THEN
          WT=WT7(IG)
          GS=GS7(IG)
        ENDIF
        IF(NGS.EQ.8) THEN
          WT=WT8(IG)
          GS=GS8(IG)
        ENDIF
        IF(NGS.EQ.9) THEN
          WT=WT9(IG)
          GS=GS9(IG)
        ENDIF
        IF(NGS.EQ.10) THEN
          WT=WT10(IG)
          GS=GS10(IG)
        ENDIF
C  EVALUATE EACH TERM OF THE INTERPOLATION FUNCTIONS AT THE GAUSS POINT.
C
```

```
      IF(IDN) 96,97,96
 96   IF(NX.NE.0) THEN
      DO 2 IX=1,NX
      XX=XX*(GS-X(IX))
  2   CONTINUE
      ELSE
      XX=1.
      ENDIF
      XR(IG)=XX
      IF(IWG.EQ.2) XR(IG)=WT*XX
      GO TO 1
 97   XR(IG)=WT
  1 CONTINUE
C
C  END OF GAUSS-LEGENDRE QUADRATURE EVALUATIONS
C
      RETURN
 99   WRITE(6,1001) NX
1001  FORMAT(/,
     +' ***** ERROR IN NUMERICAL INTEGRATION, NO FUNCTIONS EXIST *****',
     +/,' ***** ORDER OF INTERPOLATION FUNCTION NITP =',I8)
      RETURN
      END
```

```
C
C***********************************************************************
      SUBROUTINE REFORM(STIF1,STIF2,NDIM1,NLST,NDIM2,NPE)
C***********************************************************************
C
C  THIS SUBROUTINE RE-ORDERS THE STIFFNESS MATRIX SUCH THAT IT HAS THE
C  "EDGE-NODES" IN THE FRONT AND ALL OF THE "OFF-EDGE" NODES ON THE
C  BACK.
C
C-----------------------------------------------------------------------
C  STIF1 ...... ARRAY OF DIMENSION 3*NPE X 3*NPE. IT CONTAINS THE INPUT
C               STIFFNESS MATRIX OF A LAGURANGIAN ELEMENT.
C  STIF2 ...... ARRAY OF DIMENSION 3*NPE X 3*NPE. AFTER RETURN IT HAS
C               THE RE-ORDERED STIFFNESS MATRIX.
C  NDIM1 ...... INITIAL DIMENSION OF MATRICES "STIF1" AND "STIF2".
C   NLST ...... LOCAL NODAL NUMBER LISTING.
C  NDIM2 ...... INITIAL DIMENSION OF VECTOR "NLST".
C    NPE ...... TOTAL NUMBER OF NODES OF THE GIVEN LAGURANGIAN ELEMENT.
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION STIF1(NDIM1,NDIM1),STIF2(NDIM1,NDIM1),NLST(NDIM2)
C
C INITIALIZE STIF2 ...
C
      DO 1 I=1,NDIM3
      DO 1 J=1,NDIM3
  1   STIF2(I,J)=0.0
C
C PERFORMING THE RE-ORDERING ......
C
      DO 2 I=1,NPE
      IROW1=3*(I-1)
      JROW1=3*(NLST(I)-1)
      DO 3 J=1,NPE
```

```
      ICOL1=3*J-3
      JCOL1=3*(NLST(J)-1)
      DO 4 KI=1,3
      IROW=IROW1+KI
      JROW=JROW1+KI
      DO 5 KJ=1,3
      ICOL=ICOL1+KJ
      JCOL=JCOL1+KJ
      STIF2(IROW,ICOL)=STIF1(JROW,JCOL)
    5 CONTINUE
    4 CONTINUE
    3 CONTINUE
    2 CONTINUE
C
      RETURN
      END
C
C************************************************************************
      SUBROUTINE RENUMB(X,Y,Z,NX,NY,NZ,NDIM,NDIR,NLST)
C************************************************************************
C
C  THIS PROGRAM RENUMBERS THE NODAL NUMBER SEQUENCE SUCH THAT IT FORMS
C  A LAGUANGIAN ELEMENT FROM SERENDIPITY ELEMENT, OR V.V.S..
C
C------------------------------------------------------------------------
C    X,Y,Z ...... X, Y, AND Z COORDINATES WHICH ARE IN THE SEQUENCE THAT
C                 FIRST "NSE" VALUES ARE NODES ON EDGES AND REST OF THEM
C                 (i.e. NX*NY*NZ-NSE) ARE THE NODES OFF THE EDGES.
C           NODE: ALL OF THE VALUES MUST BE IN THE CORRECT SEQUENCE.
C NX,NY,NZ ...... NUMBER OF NODES ALONG X, Y, AND Z DIRECTION, RESPECT.
C     NDIM ...... INITIAL DIMENSION OF THE CALLING PROGRAM.
C     NDIR ...... OPERATING INDICATOR :
C                 NDIR = 0 ... RENUMBER TO L. ELEMENT;
C                 NDIR = 1 ... RENUMBER TO S. ELEMENT.
C     NLST ...... LOCAL NODAL NUMBER LIST.
C------------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION X(NDIM),Y(NDIM),Z(NDIM),NLST(NDIM)
C
C  FIND THE TOTAL NUMBERS OF NODES IN SERENDIPITY ELEMENT AND
C  CORRESPONDING LAGURANGIAN ELEMENT.
C
      NSE=8+4*(NX+NY+NZ-6)            !# OF NODES OF S. ELEMENT
      NLE=NX*NY*NZ                    !# OF NODES OF L. ELEMENT
      NDF=NLE-NSE                     !DIFFERENCE BTW. NSE & NLE
C
      IF(NDIR.EQ.1) GO TO 9999
C
      DO 1 I=1,NDF
      IST=NSE+I
      ILC=NLST(IST)
      NCH=IST-ILC
      XSAVE=X(ILC)
      YSAVE=Y(ILC)
      ZSAVE=Z(ILC)
      X(ILC)=X(IST)
      Y(ILC)=Y(IST)
      Z(ILC)=Z(IST)
      IF(NCH.LT.1) GO TO 1
```

```
      DO 2 J=1,NCH                                                      REN00045
      JLC=ILC+J                                                         REN00046
      XSAV1=X(JLC)                                                      REN00047
      YSAV1=Y(JLC)                                                      REN00048
      ZSAV1=Z(JLC)                                                      REN00049
      X(JLC)=XSAVE                                                      REN00050
      Y(JLC)=YSAVE                                                      REN00051
      Z(JLC)=ZSAVE                                                      REN00052
      XSAVE=XSAV1                                                       REN00053
      YSAVE=YSAV1                                                       REN00054
      ZSAVE=ZSAV1                                                       REN00055
 2    CONTINUE                                                          REN00056
 1    CONTINUE                                                          REN00057
C                                                                       REN00058
      RETURN                                                            REN00059
 9999 CONTINUE                                                          REN00060
      RETURN                                                            REN00061
      END                                                               REN00062
C                                                                       SHA00001
C***********************************************************************SHA00002
      SUBROUTINE SHAPE1(XX,YY,ZZ,DRVTS,FNC,DNRM,NDIM,NX,NY,NZ,NGS,WRK,  SHA00003
     +                  NWRK,IGEN)                                      SHA00004
C***********************************************************************SHA00005
C                                                                       SHA00006
C  CALCULATE THE INTERPOLATION FUNCTIONS ABOUT EVERY NODAL POINT BY     SHA00007
C  USING GAUSS-LEGENDRE QUADRATURE FORMULA.                             SHA00008
C                                                                       SHA00009
C    XX ...... ARRAY OF NORMAL (natural) NODAL COORDINATES IN XX.       SHA00010
C    YY ...... ARRAY OF NORMAL (natural) NODAL COORDINATES IN YY.       SHA00011
C    ZZ ...... ARRAY OF NORMAL (natural) NODAL COORDINATES IN ZZ.       SHA00012
C DRVTS ...... ARRAY OF DIMENSION (NPE x 3 x NGS), WHICH CONTAINS THE   SHA00013
C              EVALUATED DERIVATIVES ABOUT NORMAL COORDINATES SYSTEM    SHA00014
C              AT EACH GAUSS POINTS; Where                              SHA00015
C              (NPE --- number of nodes per element)                    SHA00016
C   FNC ...... ARRAY OF NPE x 3 x 10, WHICH CONTAINS THE VALUES OF THE  SHA00017
C              INTERPOLATION FUNCTION AT EACH GAUSS POINT.              SHA00018
C  DNRM ...... ARRAY OF NPE x 3, WHICH CONTAINS THE VALUES OF THE       SHA00019
C              NORMALIZATION FACTORS IN X, Y, AND Z AT EACH NODE.       SHA00020
C  NDIM ...... FIRST ROW DIMENSION OF X,Y,Z,XX,YY,ZZ AND DRVTS IN       SHA00021
C              CALLING PROGRAM.(EQ # OF NODES)                          SHA00022
C    NX ...... NUMBER OF NODES ALONG X.                                 SHA00023
C    NY ...... NUMBER OF NODES ALONG Y.                                 SHA00024
C    NZ ...... NUMBER OF NODES ALONG Z.                                 SHA00025
C   NGS ...... NUMBER OF GAUSS POINTS REQUIRED IN NUMERICAL INTEGRATION.SHA00026
C   WRK ...... WORKING SPACE OF DIMENSION NWRK X NWRK.                  SHA00027
C  NWRK ...... INITIAL DIMENSION OF MATRIX WRK.                         SHA00028
C  IGEN ...... NORMAL COORDINATES GENERATION INDICATOR :                SHA00029
C                     IGEN = 0    NORMAL COORDINATES ARE GENERATED;     SHA00030
C                     IGEN = 1    NORMAL COORDINATES ARE FROM INPUT.    SHA00031
C.......................................................................SHA00032
C                                                                       SHA00033
C  SUBROUTINES CALLED :                                                 SHA00034
C                                                                       SHA00035
C                     DERVTS ---- CALCULATE DERIVATIVES.                SHA00036
C                     NCOORD ---- GENERATE A SET NORMAL NODAL COORDINATES. SHA00037
C                     NUMINT ---- PERFORM THE NUMERICAL INTEGRATIONS.   SHA00038
C                                                                       SHA00039
C-----------------------------------------------------------------------SHA00040
C                                                                       SHA00041
      IMPLICIT REAL*8 (A-H,O-Z)                                         SHA00042
```

```
      DIMENSION XX(NDIM),YY(NDIM),ZZ(NDIM),DRVTS(NDIM,3,10),
     +     WRK(NWRK,NWRK),DNRM(NDIM,3),XINT1(10),XINT2(10),XINT3(10),
     +     DVXX(10),DVYY(10),DVZZ(10),FNC(NDIM,3,10)
C
      NPE=NX*NY*NZ
      IWRK1=0
      IWRK2=IWRK1+NDIM
      IWRK3=IWRK2+NDIM
C
C INITIALIZE XX, YY, ZZ, VWRK, WRK, DRVTS AND FNC.
C
      DO 2 I=1,NDIM
      DO 1 J=1,3
      DO 1 K=1,NGS
      DRVTS(I,J,K)=0.0
      FNC(I,J,K)=0.0
    1 CONTINUE
      IK1=IWRK1+I
      IK2=IWRK2+I
      IK3=IWRK3+I
      IF(IGEN.EQ.0) THEN
      XX(I)=0.0
      YY(I)=0.0
      ZZ(I)=0.0
      ELSE
      WRK(IK1,1)=XX(I)
      WRK(IK2,1)=YY(I)
      WRK(IK3,1)=ZZ(I)
      ENDIF
      WRK(I,2)=0.0
    2 CONTINUE
C
C GENERATE NODAL COORDINATES IN NORMAL (NATURAL) COORDINATES
C IF IT IS REQUIRED.
C
      IF(IGEN.EQ.0) THEN
      CALL NCOORD(XX,YY,ZZ,NX,NY,NZ,NDIM)
      DO IKK=1,NPE
      IK1=IWRK1+IKK
      IK2=IWRK2+IKK
      IK3=IWRK3+IKK
      WRK(IK1,1)=XX(IKK)
      WRK(IK2,1)=YY(IKK)
      WRK(IK3,1)=ZZ(IKK)
      END DO
      END IF
C
C CALCULATE THE NORMALIZED INTERPOLATION FUNCTIONS (I.F.) AND THE
C CORRESPONDING DERIVATIVES IN X, Y, AND Z, RESPECTIVELY.
C   dF/dXX=Sum(dF(i)/dXX), i=1,NPE;  etc....
C   where
C        F(i)=Fi(XX)*Fi(YY)*Fi(ZZ),             i.e.
C        dF(i)/dXX=Fi(YY)*Fi(ZZ)*(dFi(XX)/dXX)  etc.....
C
      DO 4 IPE=1,NPE                    !Evaluations node by node
        TRAS1=1.0
        TRAS2=1.0
        TRAS3=1.0
        IPTX=0
        IPTY=0
```

```
         IPTZ=0
         IK1=IWRK1+IPE
         IK2=IWRK2+IPE
         IK3=IWRK3+IPE
         DO IXYZ=1,NPE
         XX(IXYZ)=0.
         YY(IXYZ)=0.
         ZZ(IXYZ)=0.
         END DO
         DO 5 INOD=1,NPE
         IF(INOD.EQ.IPE) GO TO 5        !Only choose i .ne. j the term
         IKN1=IWRK1+INOD
         IKN2=IWRK2+INOD
         IKN3=IWRK3+INOD
C
C  EVALUATE NORMALIZATION FACTORS AND CHOOSE APPROPRIATE NODAL POINTS.
C
         IF(WRK(IKN2,1).EQ.WRK(IK2,1).AND.WRK(IKN3,1).EQ.WRK(IK3,1))
     +   THEN
         IPTX=IPTX+1                    !Interpolation function along XX
         TERM1=WRK(IK1,1)-WRK(IKN1,1)  !Normalization factor in XX
         TRAS1=TRAS1*TERM1
         XX(IPTX)=WRK(IKN1,1)           !XX-points of interpolation func
         ENDIF
         IF(WRK(IKN1,1).EQ.WRK(IK1,1).AND.WRK(IKN3,1).EQ.WRK(IK3,1))
     +   THEN
         IPTY=IPTY+1                    !Interpolation function along YY
         TERM2=WRK(IK2,1)-WRK(IKN2,1)  !Normalization factor in YY
         TRAS2=TRAS2*TERM2
         YY(IPTY)=WRK(IKN2,1)           !YY-points of interpolation func
         ENDIF
         IF(WRK(IKN1,1).EQ.WRK(IK1,1).AND.WRK(IKN2,1).EQ.WRK(IK2,1))
     +   THEN
         IPTZ=IPTZ+1                    !Interpolation function along ZZ
         TERM3=WRK(IK3,1)-WRK(IKN3,1)  !Normalization factor in ZZ
         TRAS3=TRAS3*TERM3
         ZZ(IPTZ)=WRK(IKN3,1)           !ZZ-points of interpolation func
         ENDIF
    5    CONTINUE
       DNRM(IPE,1)=TRAS1
       DNRM(IPE,2)=TRAS2
       DNRM(IPE,3)=TRAS3
C
C PERFORM NUMERICAL INTEGRATIONS WITHOUT WEIGHTS (IWG=0)
C
       CALL NUMINT(XX,NDIM,IPTX,NGS,XINT1,0)
       CALL NUMINT(YY,NDIM,IPTY,NGS,XINT2,0)
       CALL NUMINT(ZZ,NDIM,IPTZ,NGS,XINT3,0)
C
C CALCULATE DERIVATIVES ABOUT X, Y, AND Z.
C
       CALL DERVTS(DVXX,XX,NDIM,IPTX,WRK,NGS,NWRK,2)
       CALL DERVTS(DVYY,YY,NDIM,IPTY,WRK,NGS,NWRK,2)
       CALL DERVTS(DVZZ,ZZ,NDIM,IPTZ,WRK,NGS,NWRK,2)
C
C  Saving the evaluated values
C
       DO 10 IGS=1,NGS
       FNC(IPE,1,IGS)=XINT1(IGS)
       FNC(IPE,2,IGS)=XINT2(IGS)
```

```
      FNC(IPE,3,IGS)=XINT3(IGS)
      DRVTS(IPE,1,IGS)=DVXX(IGS)
      DRVTS(IPE,2,IGS)=DVYY(IGS)
      DRVTS(IPE,3,IGS)=DVZZ(IGS)
   10 CONTINUE
    4 CONTINUE
C
C  RETURN BACK XX, YY, AND ZZ VALUE.
C
      DO 11 I=1,NPE
      IK1=IWRK1+I
      IK2=IWRK2+I
      IK3=IWRK3+I
      XX(I)=WRK(IK1,1)
      YY(I)=WRK(IK2,1)
      ZZ(I)=WRK(IK3,1)
   11 CONTINUE
C
C  FINISH
C
      RETURN
      END
C
C***********************************************************************
      SUBROUTINE SOLVEQ(A,SAV1,B,NEQ,NRD,NB,NBD,ISYM,JUMP)
C***********************************************************************
C
C  A IN CORE LINEAR EQUATION SOLVER SUBROUTINE IN DOUBLE PRECISION.
C  IT CALLS A LU-DECOMPOSITION SUBROUTINE "DECOMP"
C
C-----------------------------------------------------------------------
C     A ...... DECOMPOSED COEFFECIENT MATRIX IN LU FORM
C     B ...... SOLUTION AFTER SOLVING SYSTEM OF EQUATIONS
C  SAV1....... INPUT R.H.S VECTORS
C   NEQ ...... NUMBER OF EQUATIONS
C    NB ...... NUMBER OF RIGHT HAND SIDE VECTORS
C   NBD ...... INITIAL COLUMN DIMENSION OF MATRIX [B]
C  ISYM ...... SYMMETRIC INDICATOR; ISYM=0 ... NONSYM.;ISYM=1 ... SYM.
C  JUMP ...... OPERATING INDICATOR; JUMP=1 ... LU-DECOMP. IS SKIPPED
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(NRD,NRD),B(NRD,NBD),SAV1(NRD,NBD)
C
C  SAVE R.H.S VECTORS BEFORE SYSTEM OF EQUATIONS ARE SOLVED
C
      DO 7 I=1,NEQ
      DO 7 J=1,NBD
    7 B(I,J)=SAV1(I,J)
C
C  FORM LU DECOMPOSITION FORM
C
C       IF(JUMP.NE.1) THEN
C       DO 8 I=1,NEQ
C       DO 8 J=1,NEQ
C     8 SAV2(I,J)=A(I,J)
C       CALL DECOMP(SAV2,A,NEQ,NRD,ISYM)
C       ENDIF
C
C  CALCULATE MATRIX [Y] FROM EQUATION [L][Y]=[B]
```

```
C
      DO 1 ICOL=1,NB
      B(1,ICOL)=B(1,ICOL)/A(1,1)
      DO 2 IROW=2,NEQ
      SUM=0.
      DO 3 ISUM=1,IROW-1
      SUM=SUM+A(IROW,ISUM)*B(ISUM,ICOL)
    3 CONTINUE
      B(IROW,ICOL)=(B(IROW,ICOL)-SUM)/A(IROW,IROW)
    2 CONTINUE
    1 CONTINUE
C
C  OBTAIN THE FINAL SOLUTION FROM [U][X]=[Y]
C
      DO 4 ICOL=1,NB
      IBRW=NEQ
      IF(ISYM.EQ.1) B(IBRW,ICOL)=B(IBRW,ICOL)/A(NEQ,NEQ)
      DO 5 IROW=2,NEQ
      ISTR=IBRW
      IBRW=IBRW-1
      SUM=0.
      DO 6 ISUM=ISTR,NEQ
      SUM=SUM+A(IBRW,ISUM)*B(ISUM,ICOL)
    6 CONTINUE
      B(IBRW,ICOL)=B(IBRW,ICOL)-SUM
      IF(ISYM.EQ.1) B(IBRW,ICOL)=B(IBRW,ICOL)/A(IBRW,IBRW)
    5 CONTINUE
    4 CONTINUE
C
C  EQUATION HAS BEEN SOLVED
C
      RETURN
      END
C
C*********************************************************************
      SUBROUTINE STIFF1(STIFF,DRVTS,DMTRX,FNC,DNRM,WRK,WGHT,NGS,NOD,
     +                  NDIM3,NDIM,X,Y,Z,BSAVE,AJAC)
C*********************************************************************
C
C   THIS SUBROUTINE CALCULATES THE LOCAL STIFFNESS MATRIX BY USING GAUSS
C   QUADRATURE FORMULA FOR ELEMENTS WITH NONLINEAR INTERPOLATION
C   FUNCTIONS.
C
C   STIFF ...... LOCAL STIFFNESS MATRIX WITH DIMENSION 3*NOD x 3*NOD
C   DRVTS ...... MATRIX STORAGE WHICH CONTAINS THE DERIVATIVES OF THE
C                NONLINEAR INTERPOLATION FUNCTION AT EACH GAUSS POINT.
C                (i.e. IT HAS DIMENSION NOD x 3 x 10;
C                      3 --- INDICATES DERIVATIVES AT EACH DIRECTION;
C                      10 --- INDICATES THE NUMBER OF GAUSS POINTS.
C   DMTRX ...... A 6 x 6 MATRIX CONTAINS THE MATERIAL CONSTANT MATRIX.
C     FNC ...... MATRIX STORAGE WHICH CONTAINS THE VALUES OF THE
C                NONLINEAR INTERPOLATION FUNCTIONS OBTAINED AT EACH
C                GAUSS POINT. IT HAS SAME DIMENSION AS DRVTS.
C    DNRM ...... MATRIX STORAGE WITH DIMENSION NOD x 3, WHICH CONTAINS
C                THE VALUES OF THE NORMALIZATION FACTORS.
C     WRK ...... A NDIM3 X NDIM3 WORKING SPACE FOR WORKING PACE.
C    WGHT ...... VECTOR OF LENGTH 10 CONTAINS THE WEIGHTS FOR EACH
C                GAUSS POINT.
C     NGS ...... ACTUAL REQUIRED GAUSS POINTS.
C     NOD ...... NUMBER OF NODES FOR THE ELEMENT.
```

```
C   NDIM3 ...... INITIAL DIMENSION OF STIFFNESS MATRIX.
C    NDIM ...... INITIAL DIMENSION OF X, Y, AND Z.
C   X,Y,Z ...... ARRAYS OF LENGTH NOD, WHICH CONTAIN THE X, Y, AND Z
C                COORDINATES OF NODES IN GLOBAL COORDINATES SYSTEM.
C
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION STIFF(NDIM3,NDIM3),DRVTS(NDIM,3,10),DMTRX(6,6),
     +          BSAVE(NDIM,3),FNC(NDIM,3,10),DNRM(NDIM,3),
     +          WRK(NDIM3,NDIM3),WGHT(10),VWRK(6,3),DVX(3),DVY(3),
     +          DVZ(3),AJACM(3,3),VWK(3),X(NDIM),Y(NDIM),Z(NDIM)
      NGSX=NGS
      NGSY=NGS
      NGSZ=NGS
C
C  INITIALIZE STIFF
C
      NDF=3*NOD
      DO 1 IST=1,NDIM3
      DO 1 JST=1,NDIM3
   1  STIFF(IST,JST)=0.0
C
C  START EVALUATING GAUSS POINT BY GAUSS POINT
C    [STIFF] = Sum{WGHT(IGS].[STIFF(IGS)]}; IGS=1,NGS.
C    where the sumation has to be Tripled.
C    i.e. [STIFF] = Sum{wght(i)*Sum{wght(j)*Sum{wght(k)*STIFF[I,J,K]}}}
C
      DO 2 IGSX=1,NGSX              !Sum in XX-direction
      DO 12 IGSY=1,NGSY             !Sum in YY-direction
      DO 22 IGSZ=1,NGSZ             !Sum in ZZ-direction
        DO IZR=1,3
          DVX(IZR)=0.0
          DVY(IZR)=0.0
          DVZ(IZR)=0.0
        END DO
C
C  FORM THE SHAPE FUNCTIONS AND ITS DERIVATIVES
C       N(XXi,YYj,ZZk) = f(XXi)*g(YYj)*h(ZZk)
C   dN(XXi,YYj,ZZk)/dXX = (df(XXi)/dXX)*g(YYj)*h(ZZk)
C   dN(XXi,YYj,ZZk)/dYY = f(XXi)*(dg(YYj)/dYY)*h(ZZk)
C   dN(XXi,YYj,ZZk)/dZZ = f(XXi)*g(YYj)*(dh(ZZk)/dZZ)
C
      DO 14 INZ=1,NOD
      AINT12=FNC(INZ,1,IGSX)*FNC(INZ,2,IGSY)      !f(XXi)*g(YYj)
      AINT13=FNC(INZ,1,IGSX)*FNC(INZ,3,IGSZ)      !f(XXi)*h(ZZk)
      AINT23=FNC(INZ,2,IGSY)*FNC(INZ,3,IGSZ)      !g(YYj)*h(ZZk)
      DNMT=DNRM(INZ,1)*DNRM(INZ,2)*DNRM(INZ,3)
      DX=DRVTS(INZ,1,IGSX)                        !df(XXi)/dXX
      DY=DRVTS(INZ,2,IGSY)                        !dg(YYj)/dYY
      DZ=DRVTS(INZ,3,IGSZ)                        !dh(ZZk)/dZZ
      WRK(INZ,7)=DX*AINT23/DNMT           !dN(XXi,YYj,ZZk)d/XX
      WRK(INZ,8)=DY*AINT13/DNMT           !dN(XXi,YYj,ZZk)d/YY
      WRK(INZ,9)=DZ*AINT12/DNMT           !dN(XXi,YYj,ZZk)d/ZZ
C
C  FORM JACOBIAN MATRIX
C
C     dX(XXi,YYj,ZZk)/dXX= Sum[X(ip)*dN(XXi,YYj,ZZk)/dXX]
C     dX(XXi,YYj,ZZk)/dYY= Sum[X(ip)*dN(XXi,YYj,ZZk)/dYY]
C     dX(XXi,YYj,ZZk)/dZZ= Sum[X(ip)*dN(XXi,YYj,ZZk)/dZZ]
```

```
C     dY(XXi,YYj,ZZk)/dXX= Sum[Y(ip)*dN(XXi,YYj,ZZk)/dXX]
C     dY(XXi,YYj,ZZk)/dYY= Sum[Y(ip)*dN(XXi,YYj,ZZk)/dYY]
C     dY(XXi,YYj,ZZk)/dZZ= Sum[Y(ip)*dN(XXi,YYj,ZZk)/dZZ]
C     dZ(XXi,YYj,ZZk)/dXX= Sum[Z(ip)*dN(XXi,YYj,ZZk)/dXX]
C     dZ(XXi,YYj,ZZk)/dYY= Sum[Z(ip)*dN(XXi,YYj,ZZk)/dYY]
C     dZ(XXi,YYj,ZZk)/dZZ= Sum[Z(ip)*dN(XXi,YYj,ZZk)/dZZ]
C
C  where ip=1 - Nod
C
C     [J] = [{ dX/dXX, dY/dXX, dZ/dXX },
C            { dX/dYY, dY/dYY, dZ/dYY },
C            { dX/dZZ, dY/dZZ, dZ/dZZ }]
C
      DVX(1)=DVX(1)+X(INZ)*WRK(INZ,7)
      DVX(2)=DVX(2)+X(INZ)*WRK(INZ,8)
      DVX(3)=DVX(3)+X(INZ)*WRK(INZ,9)
      DVY(1)=DVY(1)+Y(INZ)*WRK(INZ,7)
      DVY(2)=DVY(2)+Y(INZ)*WRK(INZ,8)
      DVY(3)=DVY(3)+Y(INZ)*WRK(INZ,9)
      DVZ(1)=DVZ(1)+Z(INZ)*WRK(INZ,7)
      DVZ(2)=DVZ(2)+Z(INZ)*WRK(INZ,8)
      DVZ(3)=DVZ(3)+Z(INZ)*WRK(INZ,9)
   14 CONTINUE
C
C  EVALUATE THE INVERSE OF THE JACOBIAN MATRIX AND ITS DETERMINENT
C
      CALL JACOBN(DVX,DVY,DVZ,AJAC,AJACM,3)
C
C  FORM dFi/dX, ...... X --- in GLOBAL COORDINATE SYSTEM
C                                 T                                    T
C   {dN/dX, dN/dY, dN/dZ}  = Inverse[J] . {dN/dXX, dN/dYY, dN/dZZ}
C
      DO 3 INOD=1,NOD
      VWK(1)=0.0
      VWK(2)=0.0
      VWK(3)=0.0
      DO 15 IVV=1,3
      IV=6+IVV
      VWK(1)=VWK(1)+AJACM(1,IVV)*WRK(INOD,IV)
      VWK(2)=VWK(2)+AJACM(2,IVV)*WRK(INOD,IV)
      VWK(3)=VWK(3)+AJACM(3,IVV)*WRK(INOD,IV)
   15 CONTINUE
      IF(DABS(VWK(1)).LT.1.D-10) VWK(1)=0.0
      IF(DABS(VWK(2)).LT.1.D-10) VWK(2)=0.0
      IF(DABS(VWK(3)).LT.1.D-10) VWK(3)=0.0
      BSAVE(INOD,1)=VWK(1)
      BSAVE(INOD,2)=VWK(2)
      BSAVE(INOD,3)=VWK(3)
C
C  B-Matrix has the form :
C
C          T dNi/dX      0        0       ......  T
C          |    0     dNi/dY      0       ......  |
C          |    0        0     dNi/dZ     ......  |
C     [B] = | dNi/dY   dNi/dX      0       ......  |
C          | dNi/dZ      0     dNi/dX     ......  |
C          |    0     dNi/dZ  dNi/dY     ......  |
C          -                                      -
C
      CALL FORMVK(VWRK,VWK)
```

```
C
C  STEP 1 : CALCULATE [WRK(IGS)] = Trans[B(IGS)].[D(IGS)]
C
        IB1=3*(INOD-1)+1
        IB2=IB1+1
        IB3=IB2+1
        DO 4 IWK=1,6
        WRK(IB1,IWK)=0.0
        WRK(IB2,IWK)=0.0
        WRK(IB3,IWK)=0.0
        DO 5 ISUM=1,6
        WRK(IB1,IWK)=WRK(IB1,IWK)+VWRK(ISUM,1)*DMTRX(ISUM,IWK)
        WRK(IB2,IWK)=WRK(IB2,IWK)+VWRK(ISUM,2)*DMTRX(ISUM,IWK)
        WRK(IB3,IWK)=WRK(IB3,IWK)+VWRK(ISUM,3)*DMTRX(ISUM,IWK)
    5   CONTINUE
        IF(DABS(WRK(IB1,IWK)).LT.1.D-10) WRK(IB1,IWK)=0.0
        IF(DABS(WRK(IB2,IWK)).LT.1.D-10) WRK(IB2,IWK)=0.0
        IF(DABS(WRK(IB3,IWK)).LT.1.D-10) WRK(IB3,IWK)=0.0
    4   CONTINUE
    3   CONTINUE
C
C  STEP 2 : EVALUATE LOCAL STIFFNESS MATRIX AT GAUSS POINT "IGS".
C           [STIFF(IGS)] = [WRK(IGS)].[B(IGS)] ; AT G.P. "IGS".
C
        WGHTS=WGHT(IGSX)*WGHT(IGSY)*WGHT(IGSZ)
        DO 6 INOD=1,NOD
        IB1=3*(INOD-1)+1
        IB2=IB1+1
        IB3=IB2+1
        VWK(1)=BSAVE(INOD,1)
        VWK(2)=BSAVE(INOD,2)
        VWK(3)=BSAVE(INOD,3)
        CALL FORMVK(VWRK,VWK)
        DO 7 IST=1,NDF
        SUM1=0.0
        SUM2=0.0
        SUM3=0.0
        IF(IST.LT.IB1) GO TO 97     !AVOID REPEAT CALCULATIONS (DUE SYM.)
        DO 8 ISUM=1,6
        SUM1=SUM1+WRK(IST,ISUM)*VWRK(ISUM,1)
        IF(IST.LT.IB2) GO TO 8      !AVOID REPEAT CALCULATIONS (DUE SYM.)
        SUM2=SUM2+WRK(IST,ISUM)*VWRK(ISUM,2)
        IF(IST.LT.IB3) GO TO 8      !AVOID REPEAT CALCULATIONS (DUE SYM.)
        SUM3=SUM3+WRK(IST,ISUM)*VWRK(ISUM,3)
    8   CONTINUE
        STIFF(IST,IB1)=STIFF(IST,IB1)+AJAC*WGHTS*SUM1
        STIFF(IST,IB2)=STIFF(IST,IB2)+AJAC*WGHTS*SUM2
        STIFF(IST,IB3)=STIFF(IST,IB3)+AJAC*WGHTS*SUM3
        IF(DABS(STIFF(IST,IB1)).LT.1.D-10) STIFF(IST,IB1)=0.0
        IF(DABS(STIFF(IST,IB2)).LT.1.D-10) STIFF(IST,IB2)=0.0
        IF(DABS(STIFF(IST,IB3)).LT.1.D-10) STIFF(IST,IB3)=0.0
   97   STIFF(IB1,IST)=STIFF(IST,IB1)
        STIFF(IB2,IST)=STIFF(IST,IB2)
        STIFF(IB3,IST)=STIFF(IST,IB3)
    7   CONTINUE
    6   CONTINUE
C
   22   CONTINUE
   12   CONTINUE
    2   CONTINUE
```

```
C
C  FINISH
C
      RETURN
      END
C
C***********************************************************************
      SUBROUTINE SWITCH(STIF1,WRK,NDIM,NOD,NSE,IDR)
C***********************************************************************
C
C  A IN-CORE SUBROUTINE WHICH SWITCHES BLOCK SUBMATRICES OF STIFFNESS
C  MATRIX FROM THE GIVEN FORM.
C
C                      T [K22]     [K21] T           T [K11]    K[12] T
C  MAKE [STIF1] =      |                 |  FROM [K] = |                |  ;
C                      | [K12]     [K11] |           | [K21]    K[22] |
C                      -                 -           -                -
C  OR VICE VER.
C
C-----------------------------------------------------------------------
C   STIF1 ...... OUT PUT SWITCHED MATRIX.
C     WRK ...... INPUT MATRIX TO BE SWITCHED.
C    NDIM ...... INITIAL DIMENSION OF MATRICES [STIF1] AND [WRK].
C     NOD ...... TOTAL NUMBER OF NODES OF A LAGRANGIAN ELEMENT.
C     NSE ...... NUMBER OF NODES ON THE BOUNDARY.
C     IDR ...... SWITCHING DIRECTION INDICATOR.
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION STIF1(NDIM,NDIM),WRK(NDIM,NDIM)
C
C  DECIDE SWITCHING DIRECTION.
C
      NSE3=3*NSE
      NOD3=3*NOD
      NK22=NOD3-NSE3
      IF(IDR.EQ.1) THEN
      NSAV=NSE3
      NSE3=NK22
      NK22=NSAV
      ENDIF
C
      NSTP=NK22
      IF(NSTP.LT.NSE3) NSTP=NSE3
      DO 1 I=1,NSTP
      IK22=NSE3+I
      IS22=NK22+I
      DO 2 J=1,NSTP
      JK22=NSE3+J
      JS22=NK22+J
C
C--- SWITCH [K21] ......
      IF(I.LE.NK22.AND.J.LE.NSE3) STIF1(I,JS22)=WRK(IK22,J)
C--- SWITCH [K21] ......
      IF(I.LE.NSE3.AND.J.LE.NK22) STIF1(IS22,J)=WRK(I,JK22)
C--- SWITCH [K22] ......
      IF(I.LE.NK22.AND.J.LE.NK22) STIF1(I,J)=WRK(IK22,JK22)
C--- SWITCH [K11] ......
      IF(I.LE.NSE3.AND.J.LE.NSE3) STIF1(IS22,JS22)=WRK(I,J)
C
```

```
    2  CONTINUE                                                    SWI00056
    1  CONTINUE                                                    SWI00057
C                                                                  SWI00058
       RETURN                                                      SWI00059
       END                                                         SWI00060


      SUBROUTINE GMASS(FNC,AJAC,XX,YY,ZZ,NODE,NODB3,NGS,SHAPE,
     ,                 BIGN,BIGNT,FN,XMAS)
C
C***********************************************************************
C
C      THIS SUBROUTINE COMPUTES GLOBAL MASS MATRIX BY THE
C      CONSISTENT MASS APPROACH
C
C***********************************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION SHAPE(NODE),FNC(NODE,3,10),BIGN(3,NODB3),
     ,          BIGNT(NODB3,3),FN(NODB3,NODB3),WGT1(10),
     ,          WGT2(10),WGT3(10),XMAS(NODB3,NODB3)
C
C
C***********************************************************************
C                   DEFINITION OF VARIABLES
C   VARIABLE    I/O       DEFINITION
C   --------    ----      ----------
C
C   FNC---------I---------MATRIX STORAGE CONTAINING THE VALUES
C                         NONLINEAR INTERPOLATION FUNCTIONS EVALUATED
C                         AT EACH GAUSS POINT. ITS DIMENSION IS
C                         (NODE,3,10); WHERE
C                         3----INDICATES EVALUATION IN X,Y,Z DIRECTIONS
C                         10---INDICATES NUMBER OF GAUSS POINTS
C   SHAPE-----------------ARRAY OF SHAPE FUNCTIONS AT NODE
C   BIGN------------------MATRIX PRESENTATION OF SHAPE FUNCTIONS IN
C                         ACCORDANCE WITH 3-D ELASTICITY.
C                         i.e [BIGN]= N1 0 0 N2 0 0 N3 0 0 --- N(NODE) 0 0
C                                     0 N1 0 0 N2 0 0 N3 0 ----0 N(NODE) 0
C                                     0 0 N1 0 0 N2 0 0 N3 ----0 0 N(NODE)
C   BIGNT-----------------TRANSPOSE OF [BIGN]
C   FN--------------------THE FUNCTION [BIGN]T*[BIGN]
C   WGT1,WGT2,WGT3--------VECTOR CONTAINING THE WEIGHTS OF THE GAUSS
C                         LEGENDRE QUADRATURE FOR INTEGRATION IN THE
C                         X,Y,Z DIRECTIONS RESPECTIVELY
C   XMAS------------------GLOBAL MASS MATRIX
C
C
C***************************************************************************
C
C
      RHO=120.
C
C      OBTAIN WEIGHTS FOR THE GAUSS-LEGENDRE QUADRATURE
C
      CALL NUMINT(XX,NODE,NODE,NGS,WGT1,1)
      CALL NUMINT(YY,NODE,NODE,NGS,WGT2,1)
      CALL NUMINT(ZZ,NODE,NODE,NGS,WGT3,1)
C
```

```
C       INITIALIZE MASS MATRIX
      DO 1 I=1,NODB3
      DO 1 J=1,NODB3
    1 XMAS(I,J)=0.
C
      DO 2 IGSX=1,NGS
      DO 2 IGSY=1,NGS
      DO 2 IGSZ=1,NGS
C
C       OBTAIN SHAPE FUNCTIONS FOR EACH NODE (EVALUATED AT EACH GAUSS POINT
C       N(XXi,YYi,ZZi)=f(XXi).g(YYi).h(ZZi)
c
      DO 3 IPE=1,NODE
    3 SHAPE(IPE)=FNC(IPE,1,IGSX)*FNC(IPE,2,IGSY)*FNC(IPE,3,IGSZ)
C
C           PLACE SHAPE FUNCTIONS IN MATRIX FORM FOR 3-D ELASTICITY
C
      DO 4 I=1,3
      DO 4 J=1,NODB3
    4 BIGN(I,J)=0.
C
      DO 5 IPE=1,NODE
      J=3*IPE-2
      JJ=J+1
      JJJ=J+2
      BIGN(1,J)=SHAPE(IPE)
      BIGN(2,JJ)=SHAPE(IPE)
      BIGN(3,JJJ)=SHAPE(IPE)
    5 CONTINUE
C
C                     T
C          COMPUTE [N] [N]
C
      CALL TRANSP(BIGN,BIGNT,3,NODB3)
      CALL MXMULT(BIGNT,BIGN,FN,NODB3,3,3,NODB3)
C
C       MULTIPLY THE FUNCTION FN BY THE WEIGHTS FOR THE GAUSS-LEGENDRE
C       QUADRATURE
C
      WGHTS=WGT1(IGSX)*WGT2(IGSY)*WGT3(IGSZ)
      DO 6 I=1,NODB3
      DO 6 J=1,NODB3
    6 XMAS(I,J)=XMAS(I,J)+WGHTS*FN(I,J)
C
C
    2 CONTINUE
C
C
C       COMPUTE THE CONSISTENT GLOBAL MASS MATRIX
C
      DO 7 I=1,NODB3
      DO 7 J=1,NODB3
    7 XMAS(I,J)=RHO*AJAC*XMAS(I,J)/32.2
C
C
      PRINT*,'AJAC=',AJAC,'WGHTS=',WGHTS
      PRINT*,'WGT1=',(WGT1(I),I=1,3)
      PRINT*,'WGT2=',(WGT2(I),I=1,3)
      PRINT*,'WGT3=',(WGT3(I),I=1,3)
C
```

```
C
C
      RETURN
      END


      SUBROUTINE EIGN(A,B,NDOF,X,EIGV)
C
C***************************************************************************
C
C           THIS SUBPROGRAM SOLVES THE GENERALIZED EIGENPROBLEM
C                 USING THE GENERALIZED JACOBI ITERATION
C
C
C
C                    DESCRIPTION OF VARIABLES
C                    ------------------------
C
C         VARIABLE     I/O                  DESCRIPTION
C         --------    ------                -----------
C            A----------I/O---------STIFFNESS MATRIX (POSITIVE DEFINITE)
C                                   ON OUTPUT [A] CONTAINS DIAGONALIZED
C                                   STIFFNESS MATRIX
C            B----------I/O---------MASS MATRIX (POSITIVE DEFINITE). ON OUTPUT
C                                   [B] CONTAINS  DIAGONALIZED MASS MATRIX
C            X----------I/O---------MATRIX STORING EIGENVECTORS ON SOLUTION EXIT
C                                   ON OUTPUT [X] CONTAINS EIGENVECTORS STORED
C                                   COLUMNWISE
C            EIGV-------I/O---------VECTOR STORING EIGENVALUES ON SOLUTION EXIT
C            D----------I-----------WORKING VECTOR
C            NDOF-------I-----------ORDER OF MATRICES A AND B
C            RTOL-------I-----------CONVERGENCE TORELANCE(USUALLY SET AT 10.**-12)
C            NSMAX------I-----------MAX NUMBER OF SWEEPS ALLOWED (USUALLY SET TO 15
C            IFPR-------I-----------FLAG FOR PRINTING DURING ITERATION
C                                   EQ=0 NO PRINTING
C                                   EQ=1 INTERMEDIATE RESULTS ARE PRINTED
C
C
C
C***************************************************************************
C
C
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION A(NDOF,NDOF),B(NDOF,NDOF),X(100,100),EIGV(100),
     ,          D(100)
C
C
      RTOL=10.**-12
      IFPR=0
      NSMAX=15
C
C
C
C      INITIALIZE EIGENVALUE AND EIGENVECTOR MATRICES
C
      DO 10 I=1,NDOF
      IF(A(I,I).GT.0..AND.B(I,I).GT.0.)GO TO 4
```

```
      WRITE(7,2020)
      RETURN
C
    4 D(I)=A(I,I)/B(I,I)
   10 EIGV(I)=D(I)
      DO 30 I=1,NDOF
      DO 20 J=1,NDOF
   20 X(I,J)=0.
   30 X(I,I)=1.
      IF(NDOF.EQ.1)RETURN
C
C         INITIALIZE SWEEP COUNTER AND BEGIN ITERATION
C
      NSWEEP=0.
      NR=NDOF-1
   40 NSWEEP=NSWEEP+1
      IF(IFPR.EQ.1)WRITE(7,2000)NSWEEP
C
C CHECK IF PRESENT OFF-DIAGONAL ELEMENT IS LARGE ENOUGH TO REQUIRE ZEROING
C
      EPS=(.01**NSWEEP)**2
      DO 210 J=1,NR
      JJ=J+1
      DO 210 K=JJ,NDOF
      EPTOLA=(A(J,K)*A(J,K))/(A(J,J)*A(K,K))
      EPTOLB=(B(J,K)*B(J,K))/(B(J,J)*B(K,K))
      IF(EPTOLA.LT.EPS.AND.EPTOLB.LT.EPS)GO TO 210
C
C IF ZEROING IS REQUIRED, CALCULATE THE ROTATION MATRIX ELEMENT CA AND CG
C
      AKK=A(K,K)*B(J,K)-B(K,K)*A(J,K)
      AJJ=A(J,J)*B(J,K)-B(J,J)*A(J,K)
      AB=A(J,J)*B(K,K)-A(K,K)*B(J,J)
      CHECK=(AB*AB+4.*AKK*AJJ)/4.
      IF(CHECK)50,60,60
   50 WRITE(7,2020)
      RETURN
C
   60 SQCH=DSQRT(CHECK)
      D1=AB/2.+SQCH
      D2=AB/2.-SQCH
      DEN=D1
      IF(DABS(D2).GT.DABS(D1))DEN=D2
      IF(DEN)80,70,80
   70 CA=0.
      CG=-A(J,K)/A(K,K)
      GO TO 90
   80 CA=AKK/DEN
      CG=-AJJ/DEN
C
C PERFORM THE GENERALIZED ROTATION TO ZERO THE PRESENT OFF-DIAGONAL ELEMENT
C
   90 IF(NDOF-2)100,190,100
  100 JP1=J+1
      JM1=J-1
      KP1=K+1
      KM1=K-1
      IF(JM1-1)130,110,110
  110 DO 120 I=1,JM1
      AJ=A(I,J)
```

```
      BJ=B(I,J)
      AK=A(I,K)
      BK=B(I,K)
      A(I,J)=AJ+CG*AK
      B(I,J)=BJ+CG*BK
      A(I,K)=AK+CA*AJ
  120 B(I,K)=BK+CA*BJ
  130 IF(KP1-NDOF)140,140,160
  140 DO 150 I=KP1,NDOF
      AJ=A(J,I)
      BJ=B(J,I)
      AK=A(K,I)
      BK=B(K,I)
      A(J,I)=AJ+CG*AK
      B(J,I)=BJ+CG*BK
      A(K,I)=AK+CA*AJ
  150 B(K,I)=BK+CA*BJ
  160 IF(JP1-KM1)170,170,190
  170 DO 180 I=JP1,KM1
      AJ=A(J,I)
      BJ=B(J,I)
      AK=A(I,K)
      BK=B(I,K)
      A(J,I)=AJ+CG*AK
      B(J,I)=BJ+CG*BK
      A(I,K)=AK+CA*AJ
  180 B(I,K)=BK+CA*BJ
  190 AK=A(K,K)
      BK=B(K,K)
      A(K,K)=AK+2.*CA*A(J,K)+CA*CA*A(J,J)
      B(K,K)=BK+2.*CA*B(J,K)+CA*CA*B(J,J)
      A(J,J)=A(J,J)+2.*CG*A(J,K)+CG*CG*AK
      B(J,J)=B(J,J)+2.*CG*B(J,K)+CG*CG*BK
      A(J,K)=0.
      B(J,K)=0.
C
C  UPDATE EIGENVECTOR MATRIX AFTER EACH ROTATION
C
      DO 200 I=1,NDOF
      XJ=X(I,J)
      XK=X(I,K)
      X(I,J)=XJ+CG*XK
  200 X(I,K)=XK+CA*XJ
  210 CONTINUE
C
C        UPDATE THE EIGENVALUES AFTER EACH SWEEP
C
      DO 220 I=1,NDOF
      IF(A(I,I).GT.0..AND.B(I,I).GT.0.)GO TO 220
      WRITE(7,2020)
      RETURN
C
  220 EIGV(I)=A(I,I)/B(I,I)
      IF(IFPR.EQ.0)GO TO 230
      WRITE(7,2030)
      WRITE(7,2010)(EIGV(I),I=1,NDOF)
C
C             CHECK FOR CONVERGENCE
C
  230 DO 240 I=1,NDOF
```

```
      TOL=RTOL*D(I)
      DIF=DABS(EIGV(I)-D(I))
      IF(DIF.GT.TOL)GO TO 280
  240 CONTINUE
C
C  CHECK ALL OFF-DIAGONAL ELEMENTS TO SEE IF ANOTHER SWEEP IS REQUIRED
C
      EPS=RTOL**2
      DO 250 J=1,NR
      JJ=J+1
      DO 250 K=JJ,NDOF
      EPSA=(A(J,K)*A(J,K))/(A(J,J)*A(K,K))
      EPSB=(B(J,K)*B(J,K))/(B(J,J)*B(K,K))
      IF((EPSA.LT.EPS).AND.(EPSB.LT.EPS))GO TO 250
      GO TO 280
  250 CONTINUE
C
C FILL OUT BOTTOM TRIANGLE OF RESULTANT MATRICES AND SCALE EIGENVECTORS
C
  255 DO 260 I=1,NDOF
      DO 260 J=1,NDOF
      A(J,I)=A(I,J)
  260 B(J,I)=B(I,J)
      DO 270 J=1,NDOF
      BB=DSQRT(B(J,J))
      DO 270 K=1,NDOF
  270 X(K,J)=X(K,J)/BB
      RETURN
C
C     UPDATE [D] MATRIX AND START NEW SWEEP, IF ALLOWED
C
  280 DO 290 I=1,NDOF
  290 D(I)=EIGV(I)
      IF(NSWEEP.LT.NSMAX)GO TO 40
      GO TO 255
C
C
C
 2000 FORMAT(27H0SWEEP NUMBER IN * EIGN * =,I4)
 2010 FORMAT(1H0,6E20.12)
 2020 FORMAT(25H0*** ERROR SOLUTION STOP /
     ,         30H MATRICES NOT POSITIVE DEFINITE)
 2030 FORMAT(36H0CURRENT EIGENVALUES IN * EIGN * ARE,/)
      END
```