OBJECTIVE FUNCTIONS FOR NONLINEAR STRUCTURAL ANALYSIS

by

David Brownfield Horne

Thesis submitted to the Graduate Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE
in
Civil Engineering

APPROVED:

S. M. Holzer, Chairman

A. E. Somers, Jr.

R. H. Plaut

July, 1978

Blacksburg, Virginia

ACKNOWLEDGMENTS

The author offers his sincerest thanks to Dr. S. M. Holzer for his guidance, for his encouragement, and for editing this report. The suggestions given by Dr. R. H. Plaut and Dr. A. E. Somers and their review of the manuscript are greatly appreciated.

The encouragement, friendship, and assistance by Jay Fitzgerald, Rick Sink, and Steve and Tish Glosh are also acknowledged.

The financial support provided by the Department of Civil Engineering and the computing center is greatly acknowledged.

Special thanks are due Ann Winesett for her expert typing of the manuscript.

The author also wishes to express his gratitude for the prayers and the moral and financial support of his family, his father, his mother, his brother, Dan, his sister, Amy, and her dog, Nemo.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF TABLES

I. INTRODUCTION

Solution procedure which have been applied to the analysis of structural systems may be divided into two categories: equilibrium methods and minimization methods. For static analysis with a minimization technique, one most often uses the principle of minimum potential energy to formulate the problem. One shortcoming of minimizing the potential energy is that one can not solve for unstable equilibrium states since the energy for these states is not a minimum. Dynamic analysis by minimization methods is plagued by the problem of selecting an objective function since, at the present time, no objective has been developed which is guaranteed to assume a relative minimum when dynamic equilibrium is satisfied.

An objective function that is suitable for both stable and unstable equilibrium states and that is guaranteed to assume a relative minimum at a dynamic equilibrium state is presented. Also presented is a brief review of the state of the art of nonlinear structural analysis. The finite element model used to simulate the response of plane frame and truss systems and the solution algorithm are presented. Although provisions for material nonlinearities and dynamic analysis are included in the formulation of the mathematic model, no demonstration problems are run to test the ability of the objective function to analyze inelastic structures or structures under dynamic loading. The results of a computer program using this objective function and the results of a computer

program written by Stinnett [47] are compared to demonstrate the applicability and the limitations of this objective function. Presented in the appendices are a listing of the computer program which uses the objective function, a user's guide, and flow charts of the computer program.

II. STATE OF THE ART

The methods of formulating mathematical models for nonlinear structural analysis are presented in this chapter. The solution of nonlinear dynamics problems and the selection of an appropriate time step for this solution are also addressed. Finally, a few comments about the coordinate systems used in nonlinear analysis are made.

Nonlinearities in structural mechanics are divided into two classes: geometric and material. Geometric nonlinearities may be introduced into the mathematical model by using nonlinear strain-displacement relations or by satisfying equilibrium for the deformed state. The lack of a unique solution and the need to use an iterative solution process are major complications in nonlinear structural analysis [32]. To avoid physically unimportant solutions which may arise due to a lack of solution uniqueness, one must apply the loads in sufficiently small increments [57].

At the present time, two methods are used to formulate nonlinear structural problems: the equilibrium approach and the minimization approach. In the equilibrium approach, the equations of equilibrium are formulated directly and then solved. In the minimization approach, an objective function which assumes a relative minimum at an equilibrium state is formulated. Minimization of the objective function yields the values of the generalized coordinates at the equilibrium state. Tillerson, Stricklin, and Haisler [52, p. 68] reveal the magnitude of the problem of selecting a solution technique for nonlinear structural analysis when they say:

"The problem dependent nature of nonlinear structural analysis most certainly precludes designation of a particular method of solution as the technique for use in all nonlinear analyses."

Equilibrium Approach

In the equilibrium method, the equations of equilibrium are formulated directly and then solved. This procedure has been used by a number of authors [2, 3, 4, 21, 22, 30, 36, 52] and may be characterized by the need to form a stiffness matrix. This matrix is called the tangent stiffness matrix. In forming a tangent stiffness matrix, one assumes linear behavior over each load (time) step and computes the stiffness coefficients based upon the stresses and strains at the beginning of the load (time) step. The size of the load (time) step must be kept small to minimize the effect of nonlinearities. Kalev and Gluck [22] use a constant stiffness matrix; they derive the stiffness matrix for the initial load (time) step and use this matrix for all load (time) steps. Other researchers [2, 3, 4, 21, 30, 36, 52] update the stiffness matrix after each load (time) step. Still others use a combination of the variable stiffness method and constant stiffness method during a load step. For each method, one satisfies the equations of equilibrium by an iterative process until the force imbalance is within a specified tolerance. Zienkiewicz [57] notes the similarities of these methods and Newton's method for solving nonlinear equations. In fact, for a single degree of freedom system, the variable stiffness method is equivalent to Newton's method

for the solution of nonlinear equations. Mondkar and Powell [34] note the equivalence of the variable stiffness method for the solution of multi-degree of freedom systems and the Newton-Raphson method for the solution of nonlinear equations.

Another method by which the equations of equilibrium are formulated and solved is the psuedo-force method [48, 50, 56]. In the pseudo-force method, all nonlinear terms are transposed to the right hand side of the equilibrium equation. Thus, for a static process, the equations of equilibrium become

$$K\,q = Q + Q^* \tag{2.1}$$

where K is the stiffness matrix, q is a vector of the displacements, Q is a vector of the external forces, and Q* is a vector which represents the effect of initial strains and geometric nonlinearities. Since the pseudo-forces, Q_i^*, are functions of the unknown displacements, an iterative solution must be used, and one uses the previous solution for the displacements to establish the values of the pseudo-forces. Stricklin and Haisler [48] note that the pseudo-force method is restricted to small strains and that the method is not always dependable when material nonlinearities are considered.

Minimization Approach

An objective function which assumes a relative minimum at an equilibrium state is formulated and then minimized via mathematical programming techniques [16] in the minimization approach [12, 14,

17, 18, 28, 32, 44, 46, 47] to nonlinear structural analysis. The objective function must be convex and must also have a well-defined minimum. Frequently, energy functions are used as objective functions in the minimization approach. Hence, this method is also known as the energy minimization approach.

For elastic static analysis one may use as an objective function the total potential energy, which is the sum of the strain (internal) energy and the potential energy of the external forces. For inelastic static analysis, the objective function may consist of the sum of the internal energy and a potential function of the external forces. This objective function, a quasi-potential energy function, is similar to the potential energy function of elastic systems. Although it is path dependent for inelastic systems over a set of load increments, the potential energy function is not path dependent for a single load step since it is assumed that the strain at a point must either monotonically increase or monotonically decrease during a load step [17]. For dynamic analysis, one adds an appropriate term to the objective function to represent the inertial effects. However, Park [42] notes that the feasibility of dynamic analysis by a minimization approach has not yet been fully established.

Three advantages of using a minimization approach to formulate the nonlinear structural problem are that computer storage space is saved, that nonlinearities are easily included, and that a stiffness matrix is not formed [52]. The disadvantage of using a minimization approach to formulate nonlinear structural problems is that one

encounters difficulties with the convergence to a solution within a finite number of iterations and in a limited amount of computer time [32, 52]. Anand, Weisgerber, and Shaw [1] note in their study that a minimization formulation required more computer time for the solution of a structural dynamic problem than the time needed by an equilibrium formulation for the same solution.

Time Integration Schemes

For linear dynamic analysis with an equilibrium formulation, modal analysis [31] is recommended by some authors [15, 35, 38] to integrate the equations of motion. Direct integration schemes may be considered an alternative for linear dynamic analysis, particularly if one desires the response only for a short period of time [5, 51]. However, for linear dynamic analysis with a minimization formulation or for nonlinear dynamic analysis, direct numerical integration of the equations of motion is a necessity [51]. Direct integration operators employ difference approximations to develop recurrence relationships to predict the response of a structural system over a given time period. This may be summarized as follows (from Ref. [47].

1. The state of the system is known at the beginning of a time interval.
2. The motion is approximated over the time interval as a function of the time step, initial conditions, and possibly conditions at the end of the time step.

3. The state of the system at the end of the time step is obtained by imposing conditions of dynamic equilibrium.
4. The computed conditions at the end of the time interval become the initial conditions for the beginning of the next time step and the process is repeated.

Direct integration schemes are divided into two groups, explicit and implicit time integration schemes. An explicit scheme will only use data from the beginning of a time step or from previous steps to predict the state of the system at the end of a time step. An implicit time integration scheme will use conditions at both the beginning and the end of a time step to predict the state of the system.

Each integration operator must meet certain requirements in order to gain acceptance. It must be computationally efficient and numerically precise. Dunham, Nickell, and Stickler [11] list three requirements in which a direct integration operator must be numerically precise:

1. There must be no artificial attenuation for any exciting frequency.
2. There must be no artificial growth in vibration periods or frequencies.
3. The operator must be unconditionally stable.

Direct integration operators which have been applied to structural dynamics and wave propagation problems include the Newmark-β method [9, 37], the Houbolt method [19], the Wilson-θ method [55], the

Gurtin averaging method [10], and the central difference method [24]. Nickell [40] verifies that the Newmark-β method ($\beta = \frac{1}{4}$, $\gamma = \frac{1}{2}$), the Wilson-θ method ($\theta > 1.4$), the Gurtin averaging method, and the Houbolt method are all unconditionally stable for linear problems. The central difference method is conditionally stable [54]. Hughes [20] demonstrates that Newmark's method ($\beta = \frac{1}{4}$, $\gamma = \frac{1}{2}$) is equivalent to the trapezoidal rule and that the trapezoidal rule is convergent, stable, and globally accurate to the second order with respect to the time step. Belytschko and Shoeberle [8] have developed an energy convergence criterion based upon the law of kinetic energy for use with Newmark's method.

A topic which is still open for debate is which of the direct integration operators is the best one for use in structural dynamics problems. Krieg and Key [26] indicate that the central difference method with a diagonal mass matrix is accurate and computationally efficient. Stricklin, et al. [49] compare Houbolt's method with Newmark's method and find Houbolt's method better. In fact, they find that Newmark's method becomes numerically unstable for their problem. Dunham, Nickell, and Stickler [11], Goodreau and Taylor [15], and Weeks [54] find the Newmark-β Method ($\beta = \frac{1}{4}$, $\gamma = \frac{1}{2}$) to be superior to the Houbolt method and the Wilson-θ method. Weeks [54] notes that the Newmark-β method ($\beta = \frac{1}{4}$, $\gamma = \frac{1}{2}$) experiences only period elongation but that the Houbolt method and the Wilson-θ method experience both period elongation and amplitude attenuation. In what may be the most complete comparative study of direct integration

schemes, Mikkola, Tuomala and Sinisalo [33] find that if accurate predictions are needed the central difference method is best. If there are less stringent accuracy demands, if there are time derivatives in the forcing function, or if there are material nonlinearities, Mikkola, Tuomala and Sinisalo [33] prefer Newmark's method.

Time Steps

The selection of a time step to integrate the equations of motion accurately is of great importance. The time step selected must also allow the integration scheme to remain numerically stable. In order to choose a time step for use in a structural dynamics problem, Nickell [39] recommends dividing structural dynamics problems into two categories: structural vibration and wave propagation. For structural vibration problems the lower modes dominate the solution, and the time step will only integrate the lower modes accurately [5]. This indicates that stability, not accuracy, is of prime concern in the selection of a time step [20]. Newmark [37] lists the maximum value of the time step to be used with his integration scheme as a function of the period for a linear structure. Leech, Hsu and Mack [27] demonstrate that the largest time step for a linear structure for the central difference method is

$$\Delta t \leq \frac{2}{\omega_n} \tag{2.2}$$

where Δt is the time step and ω_n is the largest natural frequency of the system. Krieg [25] shows that the central difference method

has the largest time step for stability of all explicit finite difference schemes.

One must remember that these time steps were developed for the free vibration of linear systems. A nonlinear system may require a time step of different length for stability. When selecting a time step for the solution of a nonlinear dynamic problem, one must consider the expected response of the system, and the method of formulation of the problem. At the present time, there are no set rules for selecting a time step a priori for the solution of nonlinear dynamics problems.

For wave propagation problems, the high frequency response is of primary interest and the time step must be selected using different criteria. For wave propagation problems, the time step is based upon the time it takes a wave to pass through the smallest element. For a one-dimensional element, the maximum time step is

$$\Delta t = \alpha\left(\frac{\Delta \ell}{C_L}\right) \tag{2.3}$$

where Δt is the time step, α is a constant which ranges from 0.2 to 0.9, $\Delta \ell$ is the minimum element dimension, and C_L is the pressure wave velocity [6].

Coordinate Systems

Two types of coordinate systems are used for nonlinear structural analysis. For one type of coordinate system, the Lagrangian, the reference state of the variables is a previous state of the system. The previous configuration of the system is the reference state for

the updated Lagrangian formulation while in the total Lagrangian formulation, all variables are referred to the initial unstressed state [3]. The other type of coordinate system, the Eulerian, is a formulation in which the variables are referenced to the desired state of the system [4,7]. Bathe, Ramm, and Wilson [4] compare the Lagrangian formulation with the Eulerian formulation and find the Lagrangian formulation is more general and computationally more efficient.

III. MATHEMATICAL MODELING

Basic Approach

A minimization approach is used to formulate and to solve nonlinear structural problems. Nonlinearities included in the formulation of the objective function are:

1. Equilibrium is computed for the deformed state.
2. The flexural and axial deformations are coupled by a nonlinear strain-displacement relation.
3. Material nonlinearities are modeled by a bilinear stress-strain law.

A step by step procedure must be used in order to determine the response of the structure at a discrete number of points in time or for a discrete number of load increments [57]. At these points in time or for a loading condition, the generalized coordinates which satisfy the equations of equilibrium are computed. By using the method of virtual work and D'Alembert's principle, one obtains the equations of dynamic equilibrium

$$G_i = m_i \ddot{q}_i + \frac{\partial \pi}{\partial q_i} - Q_i = 0 \qquad i = 1, 2, \ldots n \qquad (3.1)$$

where

q_i, $\ddot{q}_i$ = displacement, acceleration of the ith generalized coordinate

Q_i = ith generalized external force

m_i = mass contribution of the ith generalized coordinate

π = internal energy

n = number of generalized coordinates

Objective Function

Since a minimization technique is being used to formulate the structural analysis problem, the selection of an objective function to be minimized is critical. The objective function must be convex, must have a well-defined minimum, and must assume a relative minimum at an equilibrium state. The objective function, E, used in this report is of the form suggested by Oden [41, p. 262]

$$E = G^T G \tag{3.2}$$

where each G_i represents an equation of equilibrium (Eq. 3.1). The objective function, E, is positive definite, and therefore, convex. The objective function assumes a relative minimum at an equilibrium state.

Discretization of Motion

Since a minimization approach is being used, a direct integration scheme must be employed to integrate the equations of motion. The time domain is divided into a finite number of time steps of length Δt and the motion is approximated over each time step using Newmark's method ($\beta = \frac{1}{4}$, $\gamma = \frac{1}{2}$) [37]. For this method, the equations of motion over each time step are

$$\ddot{q}_i = \frac{\ddot{q}_{ai} + \ddot{q}_{bi}}{2} \tag{3.3}$$

$$\dot{q}_i = \dot{q}_{ai} + \frac{\ddot{q}_{ai} + \ddot{q}_{bi}}{2} t \tag{3.4}$$

$$q_i = q_{ai} + \dot{q}_{ai} t + \frac{\ddot{q}_{ai} + \ddot{q}_{bi}}{4} t^2 \tag{3.4}$$

The equations for the displacements and the accelerations at the end of the time step become

$$q_{bi} = d_i + \ddot{q}_{bi} \frac{(\Delta t)^2}{4} \tag{3.6}$$

$$\ddot{q}_{bi} = \frac{4}{(\Delta t)^2}(q_{bi} - d_i) \tag{3.7}$$

where

$$d_i = q_{ai} + \dot{q}_{ai} \Delta t + \ddot{q}_{ai} \frac{(\Delta t)^2}{4} \tag{3.8}$$

Finite Element Model [From Ref. 47]

A plane frame system is idealized as an assemblage of one dimensional finite elements rigidly connected at the external nodes. The finite element model used in this paper is presented by Holzer and Somers in Ref. 18 and by Stinnett in Ref. 47. The model is based on the assumptions of the classical beam-column:

1. The plane of bending coincides with the longitudinal plane of symmetry.
2. Plane sections remain plane and normal to the deformed reference axis.
3. Normal strains and rotations are infinitesimally small.
4. The constitutive law at any point of the element is defined by the constitutive law of the material.
5. The effect of shear deformations is negligible.

Large displacements are allowed as long as the small deformation assumption is not violated. Since these deformations may be kept arbitrarily small through the number of elements used, there are no limitations on the nodal displacements.

Both material and geometric nonlinearities are included in the model formulation. Geometric nonlinearities are admitted by the coupling of axial and flexural deformations in the strain-displacement relationship and by satisfying equilibrium for the deformed state. Material nonlinearities are modeled by a bilinear stress-strain law. Structural damping is included by the dissipation of the internal energy during inelastic unloading.

The state of the element is described by the configuration of its reference axis. Any point on the reference axis may be defined by the deflection components u and v where

u = axial deflection of the reference axis

v = transverse deflection of the reference axis

The transverse deflection is approximated as a cubic polynomial, and the axial displacement is approximated as a quadratic polynomial. This quadratic variation is accomplished by introducing an internal node at the center of the element.

Including the internal node serves two purposes. First, it allows for a varying neutral axis. This permits modeling complex strain fields which result from material and geometric nonlinearities. The second purpose is that it allows for describing the strain field

of an element from any reference axis parallel to the longitudinal fibers [18].

The displacement field for each element is expressed in terms of four local deformation components: $\bar{u}_1$, $\bar{u}_2$, $\bar{u}_3$, and $\bar{u}_4$. These are defined as follows (See Fig. 3.1).

$\bar{u}_1$ = one half of the change in element length

$\bar{u}_2, \bar{u}_3$ = relative end rotations of the element

$\bar{u}_4$ = internal node deflection

A local reference frame which rotates and translates with the element is used to relate the generalized coordinates to the local deformation components. The x-axis of the local reference axis is collinear with the chord joining the two ends of the element, and the origin is located at the midpoint of the chord. When deforming, the element is assumed first to undergo a rigid body motion and then to deform such that elements endpoints are symmetric about the local origin, and that the slopes match the boundary conditions at the nodes [18].

Transforming the global displacements into the local deformation components is accomplished as follows (Fig. 3.1)

$$\bar{u}_1 = \Delta/2 \tag{3.9}$$

$$\bar{u}_2 = U_{i3} - \delta \tag{3.10}$$

$$\bar{u}_3 = U_{j3} - \delta \tag{3.11}$$

where

$$\delta = \beta - \alpha \tag{3.12}$$

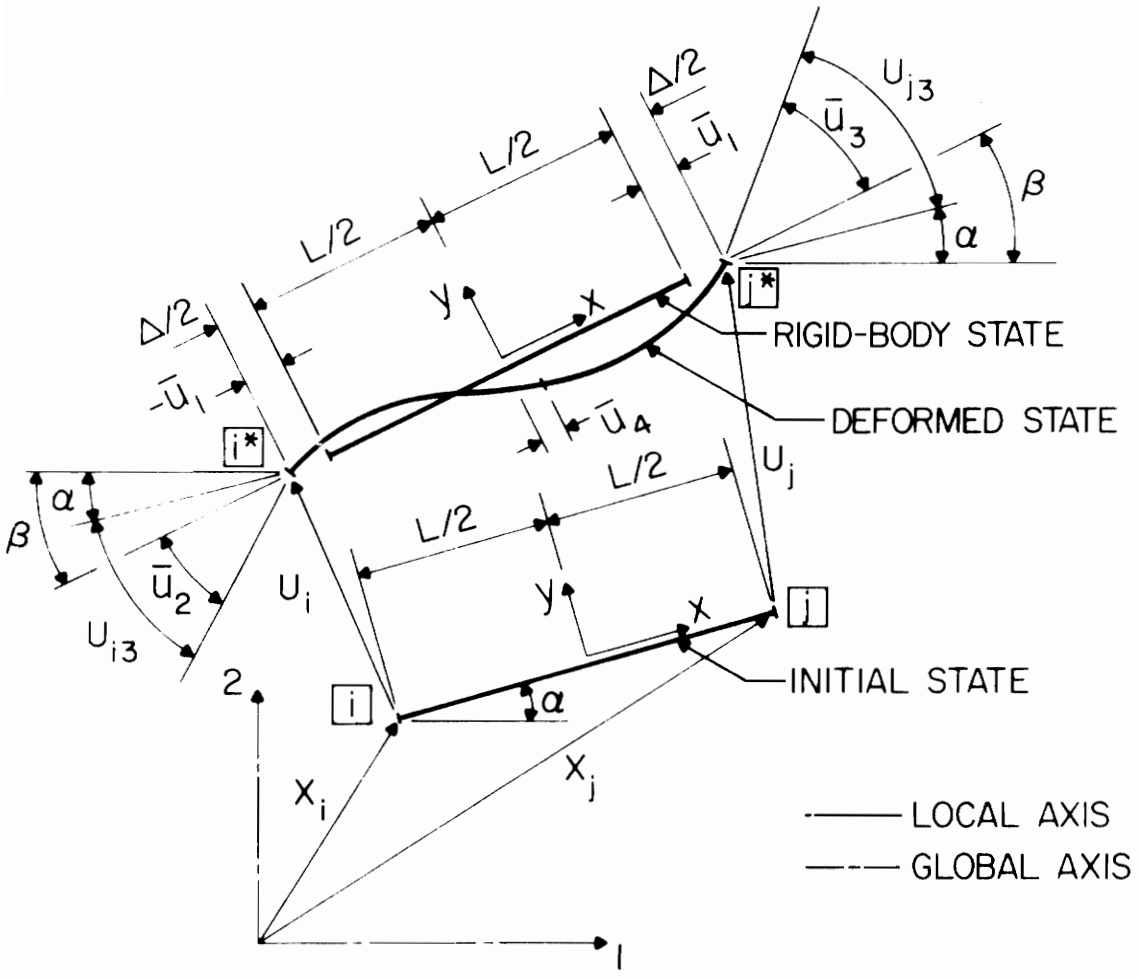


Fig. 3.1. Finite Element

$$\Delta = |\Delta X^*| - |\Delta X| \tag{3.13}$$

$$|\Delta X^*| = [(\Delta X_1^*)^2 + (\Delta X_2)^2]^{1/2} \tag{3.14}$$

$$|\Delta X| = [(\Delta X_1^*)^2 + (\Delta X_2)^2]^{1/2} \tag{3.15}$$

$$\Delta X = X_j - X_i \tag{3.16}$$

$$\Delta X^* = \Delta X + \Delta U \tag{3.17}$$

$$\Delta U = U_j - U_i \tag{3.18}$$

$$\delta = \mathrm{Tan}^{-1}\ (aa/bb) \tag{3.19}$$

$$aa = \Delta X_1 \Delta X_2^* - \Delta X_1^* \Delta X_2 \tag{3.20}$$

$$bb = \Delta X_1^* \Delta X_1 + \Delta X_2^* \Delta X_2 \tag{3.21}$$

In Eqs. 3.9 - 3.21, Δ = the change in element length; U_{i3}, U_{j3} = rotations of the nodes i, j; U_i, U_j = deflection vectors of nodes i,j; X_i, X_j = position vectors of nodes i, j; δ = chord rotation; $|\Delta x^*|$ = chord length; and L = initial element length.

The local deformation component, $\bar{u}_4$, is treated as a generalized coordinate and is computed directly in the solution process.

With the local deformations defined, it is now possible to compute the strain-displacement relation. The strain-displacement relation is defined in terms of the normalized coordinates

$$\xi = \frac{2}{L} x \qquad -1 \leq \xi \leq 1 \tag{3.22}$$

and

$$\eta = \frac{y}{L} \tag{3.23}$$

The axial displacement may be expressed as

$$u = \phi_1 \bar{u}_1 + \phi_4 \bar{u}_2 \tag{3.24}$$

where

$$\phi_1 = \xi \tag{3.25}$$

$$\phi_4 = 1 - \xi^2 \tag{3.26}$$

The transverse displacement may be expressed as

$$v = \phi_2 q_2 + \phi_3 q_3 \tag{3.27}$$

where

$$q_2 = \bar{u}_2 L/2 \tag{3.28}$$

$$q_3 = \bar{u}_3 L/2 \tag{3.29}$$

$$\phi_2 = (\xi + 1)(\xi - 1)^2/4 \tag{3.30}$$

$$\phi_3 = (\xi - 1)(\xi + 1)^2/4 \tag{3.31}$$

The strain-displacement relation of a beam-column is

$$\varepsilon(x,y) = \frac{du}{dx} + \frac{1}{2}\left(\frac{dv}{dx}\right)^2 - y\,\frac{d^2v}{dx^2} \tag{3.32}$$

or, in normalized coordinates

$$\varepsilon(\xi,\eta) = \frac{2}{L}\frac{du}{d\xi} + \frac{2}{L^2}\left(\frac{dv}{d\xi}\right)^2 - \frac{4\eta}{L}\left(\frac{d^2v}{d\xi^2}\right) \tag{3.33}$$

When Eq. 3.24 and Eq. 3.27 are substituted into Eq. 3.33, the axial and flexural terms become first order polynomials in ξ and the coupling term becomes a fourth order polynomial in ξ. This is detrimental to the convergence characteristics of the element and the coupling term is reduced to first order. This transformation is shown in Ref. 18. Thus, the strain-displacement relation becomes

$$\varepsilon(\xi,\eta) = \left(\frac{2}{L}\bar{u}_1 + \mu\right) + \left(\nu - \frac{4}{L}\bar{u}_4\right) - \eta[(3\xi - 1)\bar{u}_2 + (3\xi + 1)\bar{u}_3] \tag{3.34}$$

where

$$\mu = \frac{1}{15}\left(\bar{u}_2^2 - \frac{1}{2}\bar{u}_2\bar{u}_3 + \bar{u}_3^2\right) \tag{3.35}$$

$$\nu = \frac{1}{16}\left(\bar{u}_3^2 - \bar{u}_2^2\right) \tag{3.36}$$

Internal Energy [From Ref. 47]

The total internal energy of the system is the sum of the internal energies of all the individual members. The internal energy of the jth member is defined as

$$\pi_j = \frac{V_j}{4}\int_{-1}^{1}\int_{-1}^{1} \pi_j^*(\xi,\eta)\, d\xi\, d\eta \tag{3.37}$$

where

π_j^* = the internal-energy density at any point ξ,η in the longitudinal plane of the element

V_j = the volume of the element

ξ,η = normalized coordinates for longitudinal, transverse coordinates, respectively

and the internal-energy density is defined as

$$\pi_j^* = \int_0^{\varepsilon_j} \sigma_j\, d\varepsilon \tag{3.38}$$

where

ε_j, σ_j = strain, stress at any point ξ,η

For systems governed by a linear elastic stress-strain law, the internal energy may be computed in closed form. For such systems, Eq. 3.38 becomes

$$\pi_j^* = \frac{E\,\varepsilon_j^2}{2} \tag{3.39}$$

where E is the modulus of elasticity. Combining Eq. 3.34, Eq. 3.37 and Eq. 3.39, one obtains

$$\begin{aligned}\pi_j = {} & \frac{EAL}{2}\left(\frac{2}{L}\bar{u}_1 + \mu\right)^2 + \frac{EAL}{6}\left(\nu - \frac{4}{L}\bar{u}_4\right)^2 \\ & + \frac{2EA}{3L}\left(\frac{h^2}{4} + 3a^2\right)\left(\bar{u}_2^2 + \bar{u}_2\bar{u}_3 + \bar{u}_3^2\right) \\ & + EAa\left(\frac{2}{L}\bar{u}_1 + \mu\right)\left(\bar{u}_3 - \bar{u}_2\right) \\ & + EAa\left(\nu - \frac{4}{L}\bar{u}_4\right)\left(\bar{u}_2 + \bar{u}_3\right)\end{aligned} \tag{3.40}$$

where

A = area of element cross section

L = length of element

a = distance from reference axis to centroidal axis

h = height of the element

However, for systems governed by an inelastic stress-strain law, Eq. 3.37 must be integrated numerically. Kamat [23] shows that Gaussian quadrature is the most efficient method to integrate Eq. 3.37 to obtain the strain energy. Using Gaussian quadrature, Eq. 3.37 may be expressed as

$$\pi_j = \frac{V_j}{4} \sum_{i=1}^{n} \sum_{k=1}^{m} H_i \, H_k \, \pi_j^* \, (\xi_i, \eta_k) \tag{3.41}$$

where

H_i, H_k = Gaussian weights

ξ_i, η_k = Gaussian interpolation points.

The internal energy of an inelastic system is determined solely by the state of the Gauss points. The computation of the internal-energy density at a Gauss point is accomplished through an algorithm centered on the strain (ε_ℓ) corresponding to the last equilibrium position. In this algorithm there are three locations on the stress-strain diagram where ε_ℓ may be located (Fig. 3.2):

1. ε_ℓ is on the initial elastic path, the unloading-reloading path from inelastic compression (1).
2. ε_ℓ is on the path of inelastic tension (2).
3. ε_ℓ is on the path of inelastic compression (3).

Trial solutions are provided by the minimization process. Computing the internal-energy density of a point for a trial solution is accomplished in the following manner (subscripts denoting element and Gauss points are omitted):

$$\pi^* = \pi_\ell^* + \Delta\pi^* \tag{3.42}$$

$$\pi_\ell^* = \int_0^{\varepsilon_\ell} \sigma \, d\varepsilon \tag{3.43}$$

$$\Delta\pi^* = \int_{\varepsilon_\ell}^{\varepsilon} \sigma d\varepsilon \qquad \begin{cases} >0 \text{ if } \varepsilon > \varepsilon_\ell \\ <0 \text{ if } \varepsilon < \varepsilon_\ell \end{cases} \tag{3.44}$$

where

π^*_ℓ = internal-energy density at the last equilibrium position

π^* = internal-energy density of the trial solution

$\Delta\pi^*$ = the change of the internal-energy density

After the internal energy and the local deformation vector of an element have been found, one may then determine the variation of the internal energy with respect to the generalized coordinates as follows:

$$\frac{\partial\pi}{\partial q_i} = \sum_{m=1}^{nm} \frac{\partial\pi_m}{\partial q_i} \tag{3.45}$$

where

nm = the number of members

q_i = the ith generalized coordinate

Using the chain rule of differentiation, Eq. 3.45 becomes for a particular member

$$\frac{\partial\pi_j}{\partial q_i} = \sum_{k=1}^{4} \frac{\partial\bar{u}_k}{\partial q_i} \frac{\partial\pi_j}{\partial\bar{u}_k} \tag{3.46}$$

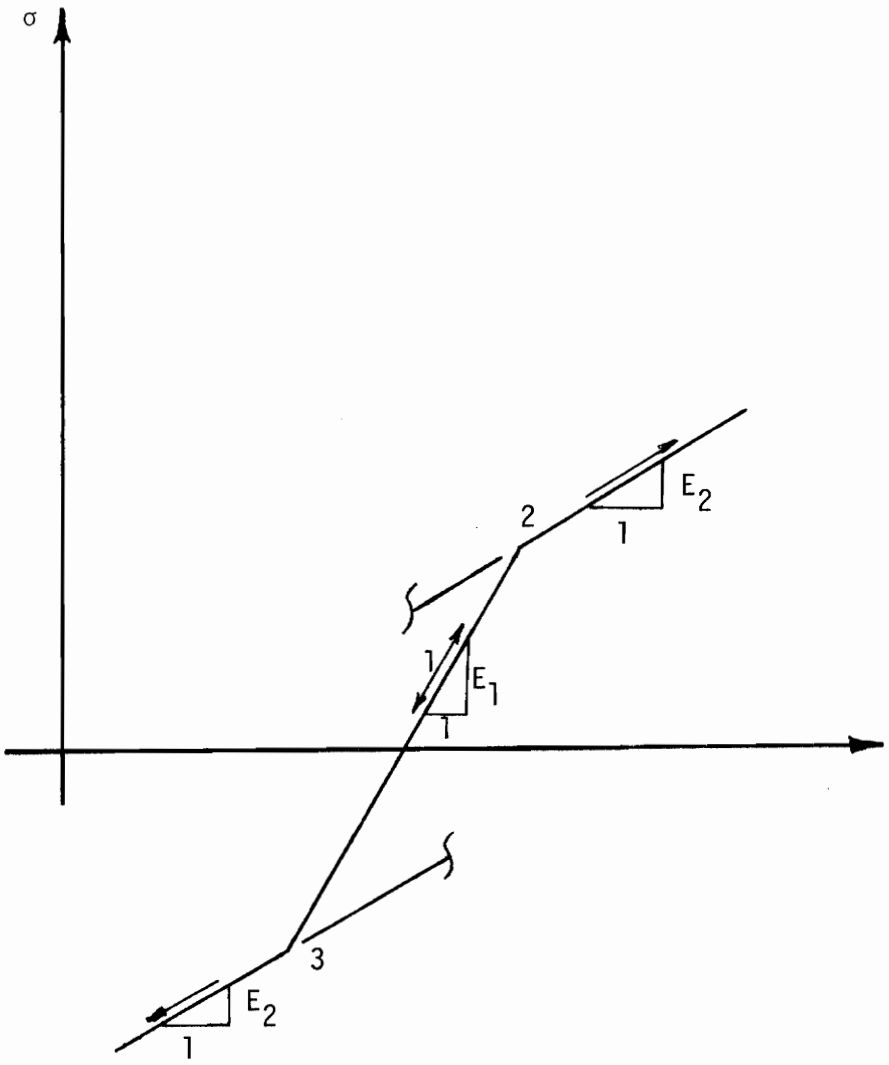


Fig. 3.2. Inelastic Loading and Unloading Paths

where, for a linear elastic system,

$$\frac{\partial \pi_j}{\partial \bar{u}_1} = 2EA \left(\frac{2}{L} \bar{u}_1 + \mu\right) + \frac{2}{L} EAa \left(\bar{u}_3 - \bar{u}_2\right) \tag{3.47}$$

$$\begin{aligned}\frac{\partial \pi_j}{\partial \bar{u}_2} = {} & \frac{2EA}{3L} \left(\frac{h^2}{4} + 3a^2\right) \left(2 \bar{u}_2 + \bar{u}_3\right) \\ & - EAa \left[\frac{2}{L} \bar{u}_1 + \mu - \frac{1}{15} \left(\bar{u}_3 - \bar{u}_2\right) \left(2\bar{u}_2 - \frac{1}{2} \bar{u}_3\right)\right] \\ & + EAa \left[\nu - \frac{4}{L} \bar{u}_4 - \frac{1}{8} \bar{u}_2 \left(\bar{u}_2 + \bar{u}_3\right)\right] \\ & + \frac{EAL}{15} \left(\frac{2}{L} \bar{u}_1 + \mu\right) \left(2 \bar{u}_2 - \frac{1}{2} \bar{u}_3\right) \\ & - \frac{EAL}{24} \left(\nu - \frac{4}{L} \bar{u}_4\right) \bar{u}_2\end{aligned} \tag{3.48}$$

$$\begin{aligned}\frac{\partial \pi_j}{\partial \bar{u}_3} = {} & \frac{2EA}{3L} \left(\frac{h^2}{4} + 3a^2\right) \left(\bar{u}_2 + 2 \bar{u}_3\right) \\ & + EAa \left[\frac{2}{L} \bar{u}_1 + \mu + \frac{1}{15} \left(2 \bar{u}_3 - \frac{1}{2} \bar{u}_2\right) - \left(\bar{u}_3 - \bar{u}_2\right)\right] \\ & + EAa \left[\nu - \frac{4}{L} \bar{u}_4 + \frac{1}{8} \bar{u}_3 \left(\bar{u}_2 + \bar{u}_3\right)\right] \\ & + \frac{EAL}{15} \left(\frac{2}{L} \bar{u}_1 + \mu\right) \left(2 \bar{u}_3 - \frac{1}{2} \bar{u}_2\right) \\ & + \frac{EAL}{24} \left(\nu - \frac{4}{L} \bar{u}_4\right) \bar{u}_3\end{aligned} \tag{3.49}$$

$$\frac{\partial \pi_j}{\partial \bar{u}_4} = -\frac{4}{3} EA \left(\nu - \frac{4}{L} \bar{u}_4\right) - \frac{4EA}{L} a \left(\bar{u}_2 + \bar{u}_3\right) \qquad (3.50)$$

Values of the derivatives of the local deformation vector with respect to the global deformation vector for an element are given in Table 3.1.

Function Minimization

Once the equations of equilibrium and the objective function are formulated, the objective function is minimized to yield the values of the generalized coordinates at an equilibrium state. In this study two minimization routines are used: the Davidon-Fletcher-Powell method [13] and the hybrid method of Powell [43].

Davidon-Fletcher-Powell Method - Gradients of the objective function are used to minimize the objective function by the Davidon-Fletcher-Powell method. Convergence is guaranteed only for a quadratic objective function, but higher order functions may be minimized by the Davidon-Fletcher-Powell method since, near the minimum, the second order terms of a Taylor series expansion dominate [13].

The objective function is minimized by a series of one-dimensional searches. This may be expressed as

$$x^{i+1} = x^i + \alpha^i S^i \qquad (3.51)$$

where

x^{i+1}, x^i = vector of unknowns at the i+1, ith iteration

α^i = step length for the ith iteration, a positive constant

S^i = ith directional search vector

Table 3.1. Partial Derivatives of Element Deformation Components with Respect to Global Displacements (Refer to Fig. 3.1)

	$\frac{\partial \bar{u}_1}{\partial (\ )}$	$\frac{\partial \bar{u}_2}{\partial (\ )}$	$\frac{\partial \bar{u}_3}{\partial (\ )}$	$\frac{\partial \bar{u}_4}{\partial (\ )}$
U_{i1}	$-\frac{1}{2}\frac{\lvert\Delta X_1^*\rvert}{\lvert\Delta X^*\rvert}$	AB · AC	AB · AC	0
U_{i2}	$-\frac{1}{2}\frac{\lvert\Delta X_2^*\rvert}{\lvert\Delta X^*\rvert}$	-AB · AD	-AB · AD	0
U_{i3}	0	1	0	0
U_{j1}	$+\frac{1}{2}\frac{\lvert\Delta X_1^*\rvert}{\lvert\Delta X^*\rvert}$	-AB · AC	-AB · AC	0
U_{j2}	$+\frac{1}{2}\frac{\lvert\Delta X_2^*\rvert}{\lvert\Delta X^*\rvert}$	AB · AD	AB · AD	0
U_{j3}	0	0	1	0
$\bar{u}_4$	0	0	0	1

$$AA = |\Delta X_1||\Delta X_2^*| - |\Delta X_2||\Delta X_1^*| \qquad \text{(Eq. 3.20)}$$

$$BB = |\Delta X_1||\Delta X_1^*| + |\Delta X_2||\Delta X_2^*| \qquad \text{(Eq. 3.21)}$$

$$AB = -\frac{1}{AA^2 + BB^2}$$

$$AC = AA|\Delta X_1| + BB|\Delta X_2|$$

$$AD = BB|\Delta X_1| - AA|\Delta X_2|$$

The search vector, S, is a function of the initial gradient vector for the first iteration and a function of the current gradient vector and all previous gradients for subsequent iterations. Convergence is obtained when each component of the search vector is less than a prescribed accuracy and the absolute distance from the minimum is less than a prescribed amount [13].

Powell Method - Powell combines the ideas of steepest descent and Newton-Raphson to form a hybrid method for finding the minimum of a function, $F(x)$ where

$$F(x) = \sum_{i=1}^{n} [f_i(x)]^2 \tag{3.52}$$

To begin an iteration, the method requires a vector of the variables and the corresponding function values, an approximation to the Jacobian matrix, the inverse of the approximation to the Jacobian matrix, a direction matrix, and the maximum step length for this iteration. The Jacobian matrix is evaluated as

$$J_{ij} = \frac{\partial f_i(x)}{\partial x_j} \tag{3.53}$$

where

$$\frac{\partial f_i(x)}{\partial x_j} = \frac{f_i(x_1, x_2, \ldots, x_j+\delta, \ldots, x_n) - f_i(x_1, x_2, \ldots, x_j, \ldots, x_n)}{\delta} \tag{3.54}$$

The term, δ, in Eq. 3.54 represents the step length and $f_i(x)$ is the ith function value. The method proceeds as follows:

1. The step length, δ, is computed from the gradients and Newton steps such that $||\delta||$ is less than the maximum step length.
2. The step length, δ, is checked to determine if it is "sufficiently independent" to ensure that the directions which are generated in the solution process are separated by a substantial amount and that the directions span the full space of the variables. This check is necessary since a property of the improvement to the Jacobian matrix is that the results of applying both the old and the new Jacobian approximations to any vector orthogonal to δ are the same. If the vectors δ are linearly dependent, there is some nonzero vector γ (orthogonal to all vectors δ) such that $J\gamma$ is the same for all Jacobian approximations. This is unsatisfactory since the true Jacobian changes with x, and the true value of $J\gamma$ changes as well [43].
3. If δ is sufficiently independent, the function values $f_i(x + \delta)$ are evaluated and the maximum step length is revised. If the change, δ, in the variables improves the objective function, the values of the unknowns are changed; if no improvement in the objective function occurs, no changes are made to the variable.
4. The step length is checked to see if it is less than a specified value.

5. If the step length is less than the specified value or if the step length is not sufficiently independent, the step length is revised.
6. The Jacobian matrix is revised and the process is repeated until convergence is obtained.

IV. DEMONSTRATION PROBLEMS

Five demonstration problems are presented which illustrate the applicability and limitations of the objective function G^TG. The results of a computer program using this objective function are compared with the results of a computer program which was written by Stinnett [47] and which minimizes the potential energy for static problems. For all test problems, equilibrium is formulated for the deformed state, and linear elastic material behavior is assumed. Only the static response of nonlinear plane frame and truss systems is studied. For reference purposes, algorithm 1 is the objective function G^TG minimized by the Davidon-Fletcher-Powell method [13] (subroutine DFMFP), algorithm 2 is the objective function G^TG minimized by the hybrid method of Powell [43] (subroutine NSO1AD), and algorithm 3 is the computer program of Stinnett [47].

Two Member Symmetric Truss

A two member symmetric truss (Fig. 4.1) is analyzed in order to test the gradients of the objective function for algorithm 1 and to compare solution times as a measure of efficiency. The two elements of the truss have equal stiffnesses (k = 200 lbs/in). Three load conditions with six load steps per load condition are used in the analysis. The load conditions are defined as follows:

1. $Q_1 \neq 0, \quad Q_2 = 0$

2. $Q_1 = 0, \quad Q_2 \neq 0$

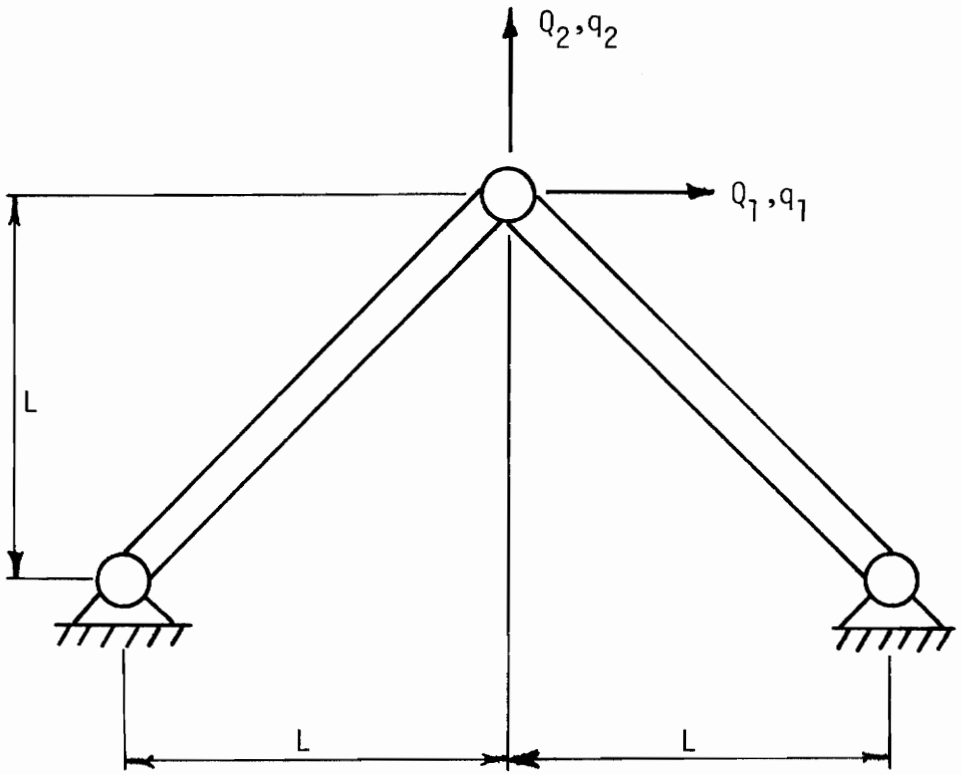


Fig. 4.1. Two Member Symmetric Truss

Table 4.1. Comparison of Execution Times, Errors and Gradients, Problem 1

Algorithm	Closure Tolerance	Solution Time (Sec.)	Error	Gradients
1	10^{-8}	102.36	10^{-8}	10^{-7}
1	10^{-5}	15.76	10^{-5}	10^{-3}
1	10^{-4}	14.80	10^{-4}	10^{-3}
2	10^{-8}	4.60	10^{-5}	--
3	10^{-8}	8.17	10^{-7}	10^{-7}

3. $Q_1 \neq 0$, $Q_2 \neq 0$, and $Q_1 \neq Q_2$

The closure tolerance, EPS, is adjusted in order to reduce the amount of computer time used by algorithm 1 to analyze the truss under the given loading conditions. Since the loads are kept small, the response is essentially linear, and the displacements predicted by all three algorithms are in close agreement with the displacements predicted by a linear model (e.g. a direct stiffness solution). A comparison of the solution times, error tolerances, gradients, and errors is presented in Table 4.1. The error is defined as the Euclidean norm of the equilibrium residuals.

Asymmetric Truss

A two member asymmetric truss (Fig. 4.2) is also analyzed to test the gradients of the objective function for algorithm 1 and to compare solution times. The two elements have equal stiffnesses (k = 400 lbs/in). Three load conditions with four load increments per load condition are used in the analysis. The load conditions are defined as follows:

1. $Q_1 \neq 0$, $Q_2 = 0$
2. $Q_1 = 0$, $Q_2 \neq 0$
3. $Q_1 \neq 0$, $Q_2 \neq 0$, and $Q_1 = Q_2$

Solution time for algorithm 2 for all load increments and load conditions was 6.08 seconds; solution time for algorithm 3 was 4.11 seconds. Algorithm 1 failed to analyze this truss for any of the

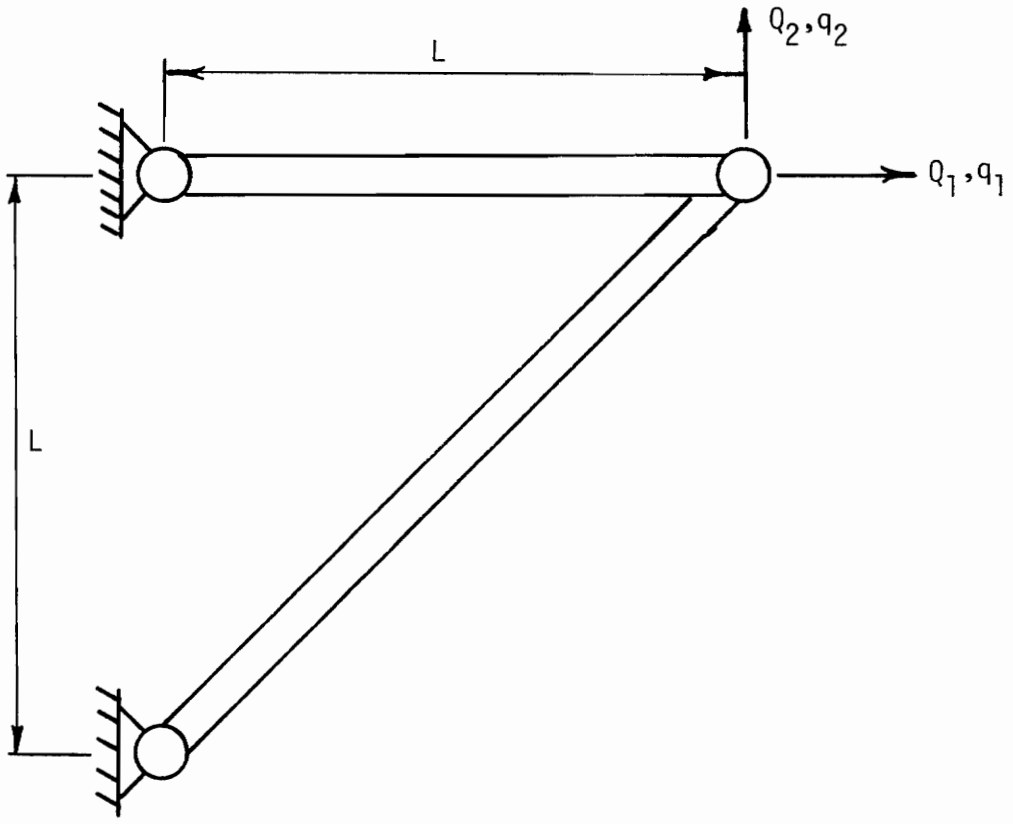


Fig. 4.2. Two Member Asymmetric Truss

load conditions, and its use was discontinued. The loads are kept small, and the response predicted by algorithms 2 and 3 is essentially linear. The displacements predicted by algorithms 2 and 3 are in close agreement with the displacements predicted by a linear model (e.g. a direct stiffness solution).

Fixed-Fixed Beam

The third demonstration problem is used to examine future the efficiency and accuracy of algorithms 2 and 3. The fixed-fixed beam (Fig. 4.3) is 20 inches long and has a square cross-section, three inches on a side. Four finite elements (see Chapter III) of equal length are used to model the response of the beam. Four load increments are used. Algorithm 2 needed 47.06 seconds to determine the response of the beam for all load steps; algorithm 3 required less time, 22.37 seconds, for the same analysis. Midspan transverse deflections are compared in Table 4.2.

Beam-Column

A geometric nonlinear beam-column (Fig. 4.4) is analyzed to evaluate the ability of the second algorithm to predict the response of structures which undergo large displacements. The axial load is applied concentrically. The initial load is slightly above the Euler load (P_{cr} = 21.055 lbs) of the column, and the load is incremented to approximately 1.5 times the Euler load.

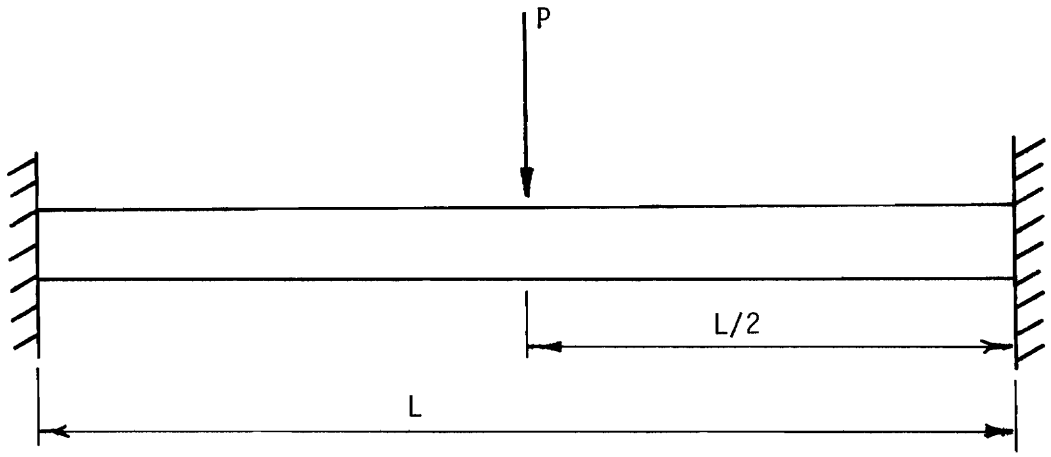
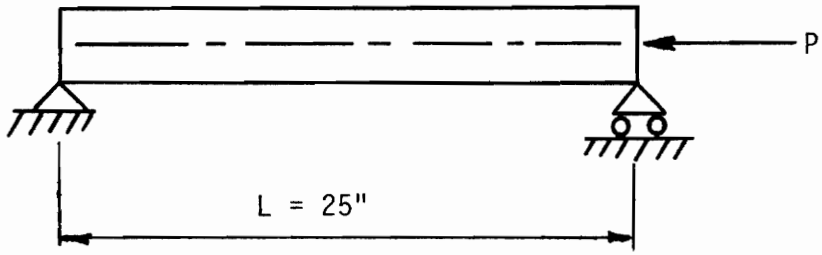


Fig. 4.3. Fixed-Fixed Beam, Problem 3

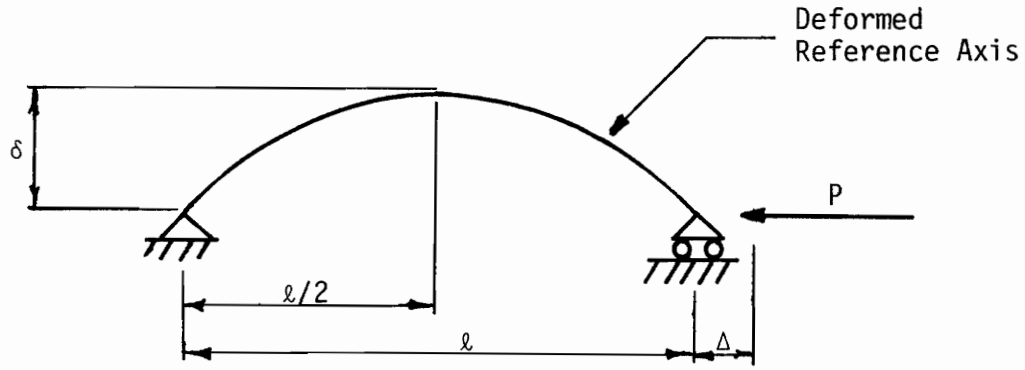
Table 4.2. Transverse Midspan Deflection[1]
Fixed-Fixed Beam

Q	Linear Model	Algorithm 2	Algorithm 3
-15.0	-0.6481481	-0.6371532	-0.6371535
-20.0	-0.6790123	-0.6664436	-0.6664436
-25.0	-0.7098765	-0.6955985	-0.6955985
-30.0	-0.7407407	-0.7246143	-0.7246143

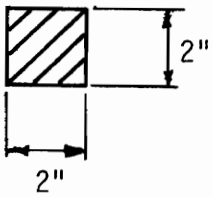
[1]Deflections include the effect of the weight of the beam.



a. Concentrically Loaded Beam Column



b. Deformation of the Reference Axis



c. Cross-Section

Fig. 4.4. Buckling of a Column, Problem 4

This corresponds to the elastica theory of thin rods. The results are compared with the exact solution expressed in the form of the elliptical integral by Timoshenko and Gere [53]. The displacements predicted by algorithm 2 and the displacements given by the elliptical integral solution are in good agreement and are plotted in Fig. 4.5 and Fig. 4.6.

The same beam-column (Fig. 4.4) is also analyzed with a small initial curvature due to the application of small moments (0.5 in-lbs) at the ends of the beam-column. The moments, which are held constant throughout the analysis, cause an initial transverse midspan deflection of 0.029297 inches. The initial axial load is zero, and the load is incremented until the load is approximately 1.5 times the Euler load. For loads above the Euler load, the displacements predicted by the second algorithm are in accord with the displacements predicted for a perfect column and by the elliptical integral solution. Results are plotted in Fig. 4.5 and Fig. 4.6.

For the deformed beam-column, algorithm 2 requires a total of 41 load steps and a solution time of approximately 25.80 minutes to determine the response of the beam-column. Although algorithm 3 requires smaller load increments and therefore, a greater number of load increments (the number of load increments is 96), much less time (approximately 17.75 minutes) is required to determine the response of the beam-column.

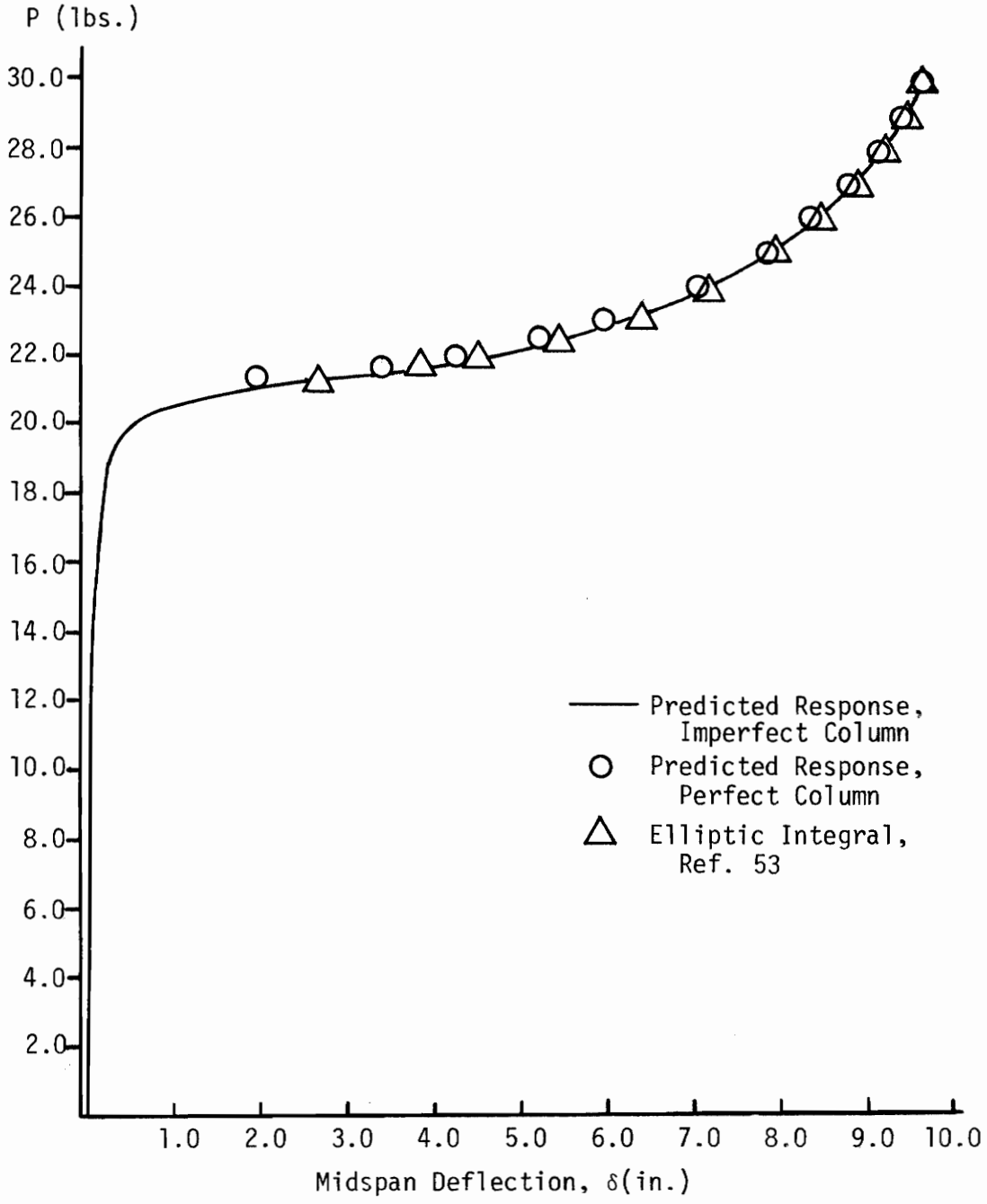


Fig. 4.5. Four Element Beam Column - Midspan Deflection

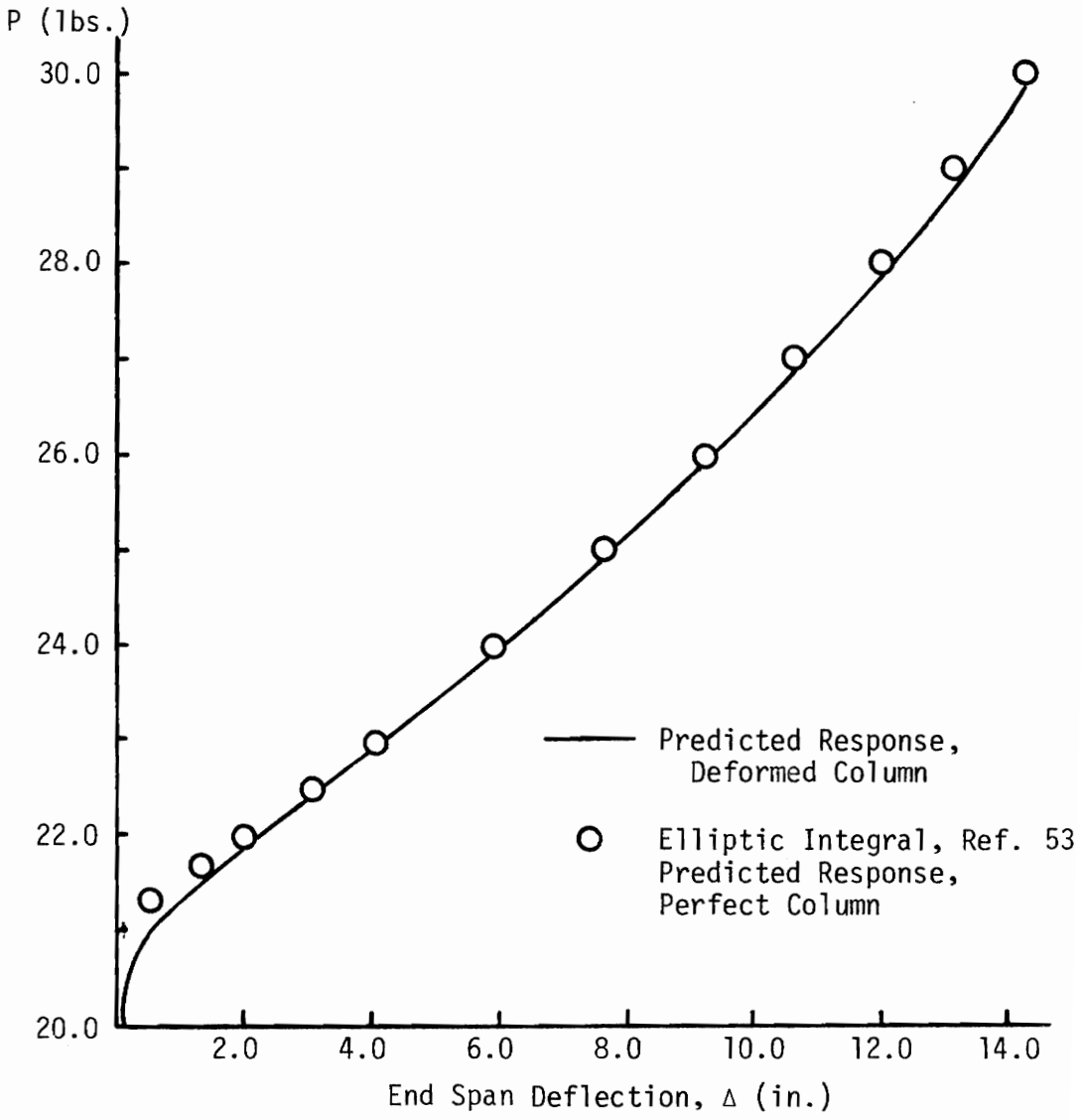


Fig. 4.6 Four Element Beam Column - Axial End Displacement

Stability of a Truss

The response of a five element symmetric truss (Fig. 4.7) is investigated in order to demonstrate the ability of algorithm 2 to analyze structures that have unstable equilibrium states. Element 2 has a stiffness (k_2) of 40.00 lbs/in; members 1 and 3 each have a stiffness of 19.40 lbs/in; members 4 and 5 each have a stiffness of 3.288 lbs/in. The truss is loaded symmetrically at the second and third joints. The truss is unstable when the determinant of one of the principal minors of the inverse of the approximate Jacobian matrix becomes less than zero. The load-deflection curve (Fig. 4.8) indicates that both limit and bifurcation points exist. As a limit point is approached, the size of the load increment must be kept very small. When a limit point is reached, the minimization routine NSOlAD gives an error return since a nearby stationary point of the equilibrium path is predicted. By incrementing the displacements just past those of the limit point and by unloading the structure, an unstable equilibrium path may be traced.

Bifurcation failure occurs when the determinant or one of the principal minors of the inverse of the approximate Jacobian matrix becomes less than zero. Once a starting point on the unstable equilibrium path which passes through a bifurcation point has been determined, the entire path may be traced. A starting point for this problem was determined as follows:

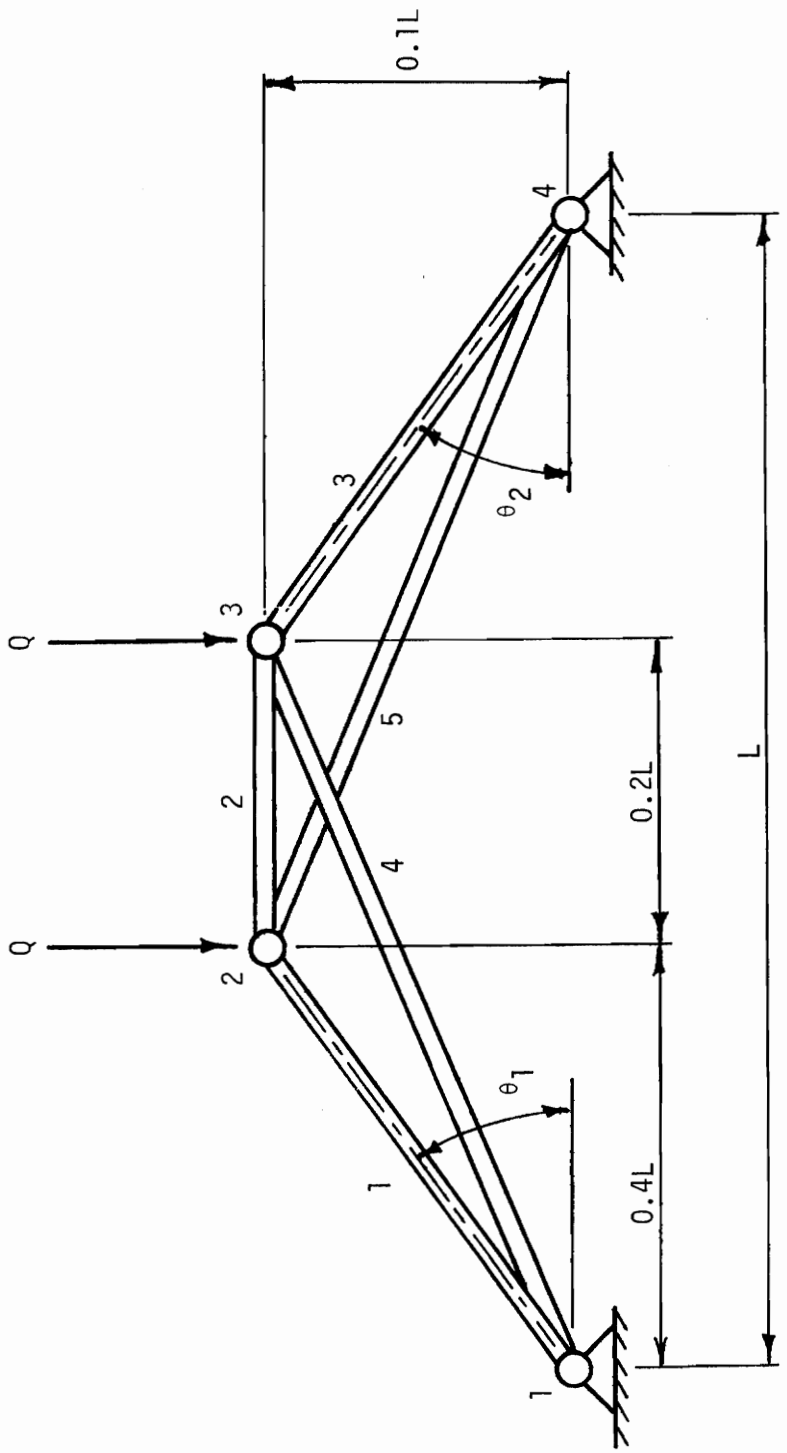


Fig. 4.7. Five Member Symmetric Truss, Problem 5

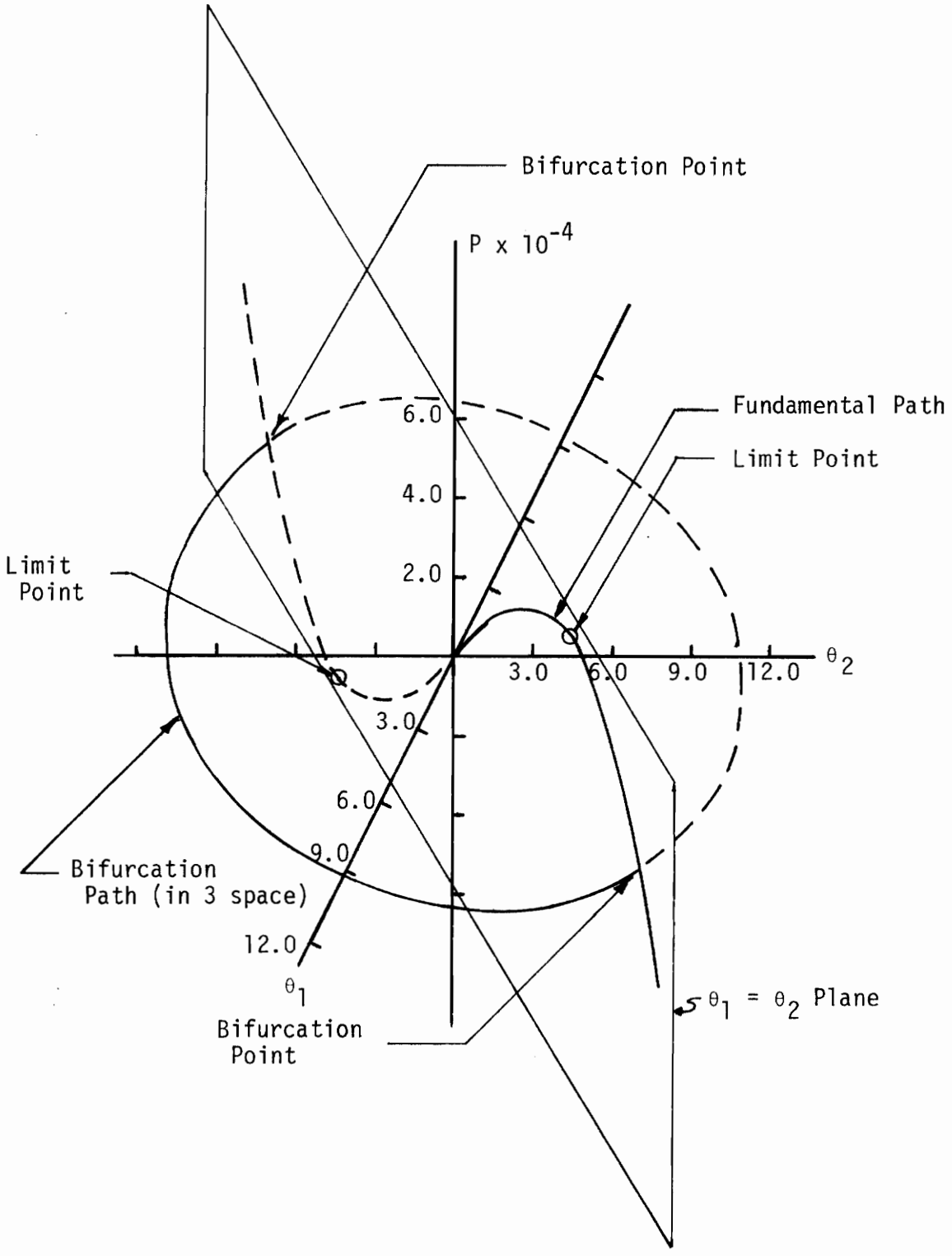


Fig. 4.8. Load Deflection Curve, Problem 5

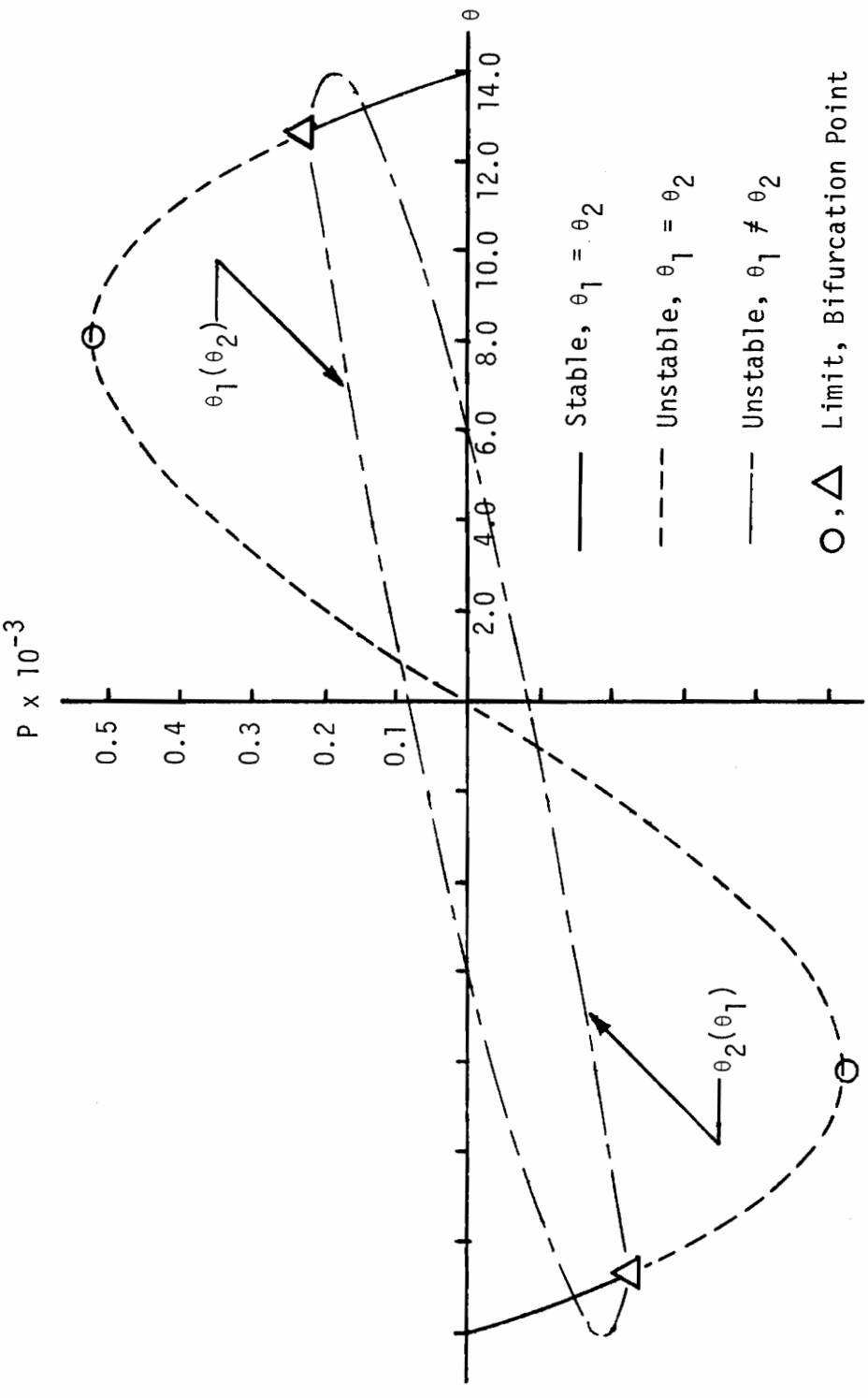


Fig. 4.9. Projection of Load Deflection Curve on $\theta_1 = \theta_2$ Plane

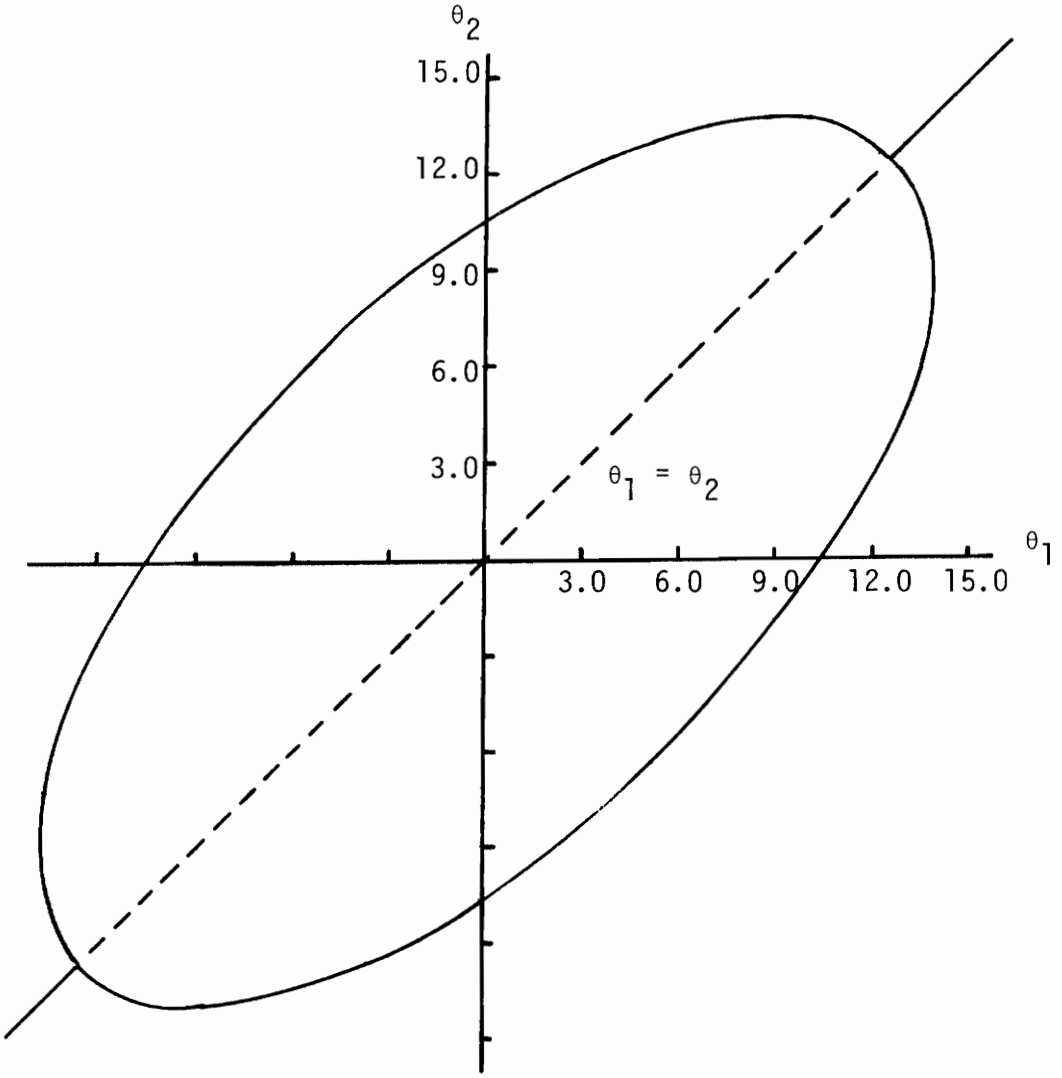


Fig. 4.10. Projection of Load Deflection Curve on $\theta_1 \sim \theta_2$ Axes

1. A straight line was drawn through the bifurcation points, and a load and corresponding angles ($\theta_1 = \theta_2$) were selected as initial guesses.
2. The angles were modified to be

$$\theta_1^{K+1} = \theta_1^{k} - \delta^{k} \tag{4.1}$$

$$\theta_2^{k+1} = \theta_2^{k} + \delta^{k} \tag{4.2}$$

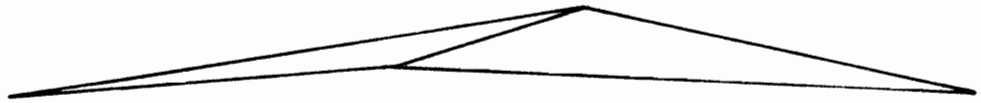
   This was done manually and the load was kept constant.
3. The loads and new guess for an equilibrium configuration are input and the results are checked for convergence to an equilibrium path with $\theta_1 \neq \theta_2$.

The projection of the load deplection curve on the $\theta_1 = \theta_2$ plane is presented in Fig. 4.9. Fig. 4.10 depicts the projection of the load-deflection curve on the $\theta_1 = \theta_2$ axes.

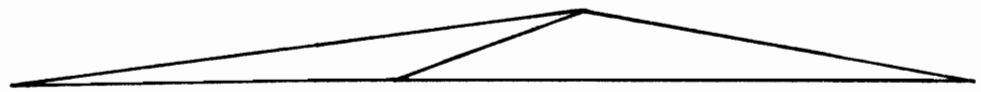
For selected points on the bifurcation path (Fig. 4.8), sketches of the deflected shape of the truss are presented in Fig. 4.11.

a. undeformed, Q=0

b. Q = -0.036

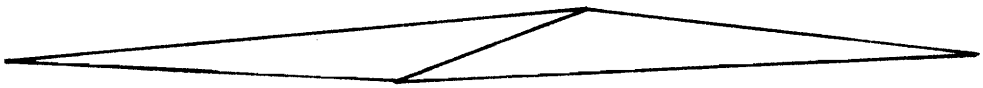


c. Q = -0.026

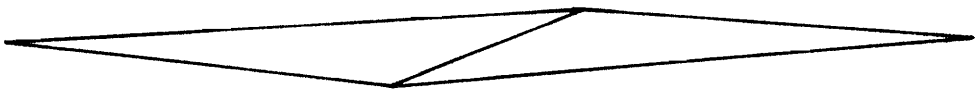


d. Q = -0.016

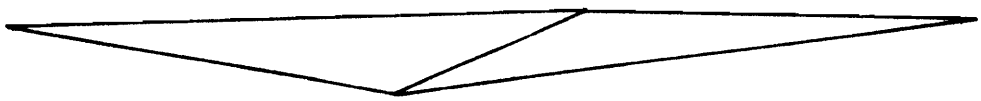
Fig. 4.11. Deflected Shape of Truss



e. Q = -0.006

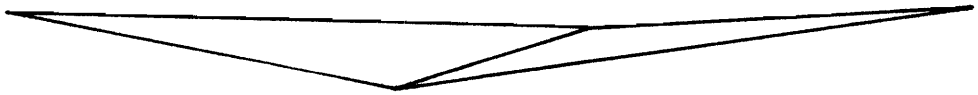


f. Q = 0.004



g. Q = 0.014

Fig. 4.11. (Continued)



h. Q = 0.024

i. Q = 0.034

j. Q = 0.044

Fig. 4.11. (Continued)

V. RECOMMENDATIONS AND CONCLUSIONS

On the basis of the experience with the demonstration problems presented in Chapter IV, it is recommended that the objective G^TG be minimized by the hybrid method of Powell [43]. The inability of the Davidon-Fletcher-Powell method [13] to minimize the objective function is due primarily to the form of the objective function. The objective function, when it is minimized by the hybrid method of Powell, is suitable for the analysis of structures with both stable and unstable equilibrium states. The fifth problem in Chapter IV demonstrates the ability of the objective function G^TG to predict limit and bifurcation points. As a limit point is approached, the size of the load step must be kept very small. When a limit point is reached, the minimization routine NSOlAD gives an error return since a nearby stationary point of the equilibrium path is predicted. Bifurcation failure occurs when the determinant or one of the principal minors of the inverse of the approximate Jacobian matrix becomes less than zero. In the fifth demonstration problem, a bifurcation point exists, and the attempt to trace this unstable path proved to be difficult. In order to trace an unstable equilibrium path passing through a bifurcation point, one must start very near to an unstable equilibrium point on the path or the minimization routine predicts points on a fundamental equilibrium path (see Fig. 4.8).

For elastic static analysis, algorithms 2 and 3 predict virtually the same equilibrium states and have the same order of magnitude of error. Structural analysis by minimizing the potential energy function (algorithm 3) proves to be more efficient than structural analysis by minimizing the objective function G^TG. Since static analysis and dynamic analysis are performed by essentially the same procedure (for static analysis, loads are incremented; for dynamic analysis, time is incremented), the results of the fourth demonstration problem in Chapter IV indicate that algorithm 2 would still be more inefficient than algorithm 3 for dynamic analysis. It is therefore recommended that the objective function G^TG (algorithm 2) be used stability analysis since it can predict the existence of limit and bifurcation points and predict post-buckled equilibrium states.

The efficiency of the objective function G^TG may be improved by minimizing the objective function by a gradient method such as Marquardt's method [29]. Stability analysis under either static or dynamic loading with more than two degrees of freedom should be attempted to determine the ability of the objective function G^TG to predict post-buckled states of more complex structures.

APPENDIX A

REFERENCES

REFERENCES

1. Anand, S. C., Weisgerber, F. E., and Shaw, H-H., "Direct Solution vs. Quadratic Programming Technique in Elastic-Plastic Finite Element Analysis," Computers and Structures, Vol. 7, No. 2, April, 1977, pp. 221-228.

2. Argyris, J. H., Dunne, P. C., and Angelopoulos, T., "Nonlinear Oscillations Using the Finite Element Technique," Computer Methods in Applied Mechanics and Engineering, Vol. 2, 1973, pp. 203-250.

3. Bathe, K. J. and Ozdemir, H., "Elastic-Plastic Large Deformation Static and Dynamic Analysis," Computers and Structures, Vol. 6, 1976, pp. 81-92.

4. Bathe, K. J., Ramm, E., and Wilson, E. L., "Finite Element Formulation for Large Deformation Dynamic Analysis," International Journal for Numerical Methods in Engineering, Vol. 9, 1975, pp. 353-386.

5. Bathe, K. J. and Wilson, E. L., "Stability and Accuracy Analysis of Direct Integration Methods," Earthquake Engineering and Structural Dynamics, Vol. 1, 1973, pp. 283-291.

6. Belytschko, T., Chiapetta, R. L., and Bartel, H. D., "Efficient Large Scale Non-linear Transient Analysis by Finite Elements," International Journal for Numerical Methods in Engineering, Vol. 10, 1976, pp. 579-596.

7. Belytschko, T., and Hsieh, B. J., "Non-linear Transient Finite Element Analysis with Convected Coordinates," International Journal for Numerical Methods in Engineering, Vol. 7, 1973, pp. 255-271.

8. Belytschko, T. and Shoeberle, D. F., "On the Unconditional Stability of an Implicit Algorithm for Nonlinear Structural Dynamics," Journal of Applied Mechanics, Vol. 42, Dec., 1975, pp. 865-869.

9. Chan, S. P., Cox, H. L., and Benfield, W. A., "Transient Analysis of Forced Vibrations of Complex Structural-Mechanical Systems," Journal of the Royal Aeronautical Society, Vol. 66, July, 1962, pp. 457-460.

10. Danay, A., Gluck, J., and Gellert, M., "A Generalized Continuum Method for Dynamic Analysis of Asymmetric Tall Buildings," Earthquake Engineering and Structural Dynamics, Vol. 4, 1975, pp. 179-203.

11. Dunham, R. S., Nickell, R. E., and Stickler, D. C., "Integration Operators for Transient Structural Response," Computers and Structures, Vol. 2, 1972, pp. 1-15.

12. Felippa, C. A., and Sharifi, P., "Computer Implementation of Nonlinear Finite Element Analysis," Symposium on Numerical Solution of Nonlinear Structural Problems, ASME, Detroit, Michigan, Dec., 1973, pp. 31-49.

13. Fletcher, R., and Powell, M. J. D., "A Rapidly Convergent Descent Method for Minimization," The Computer Journal, Vol. 6, 1963, pp. 163-170.

14. Fox, R. L., and Stanton, E. L., "Developments in Structural Analysis by Direct Energy Minimization," AIAA Journal, Vol. 6, June, 1968, pp. 1036-1042.

15. Goudreau, G. L., and Taylor, R. L., "Evaluation of Numerical Integration Methods in Elastodynamics," Computer Methods in Applied Mechanics and Engineering, Vol. 2, 1972, pp. 69-97.

16. Himmelblau, D. M., Applied Nonlinear Analysis, McGraw Hill, New York, 1972.

17. Holzer, S. M., Melosh, R. J., Barker, R. M., and Somers, Jr., A. E., "SINGER: A Computer Code for General Analysis of Two-Dimensional Concrete Structures," AFWL TR 74-228, Vol. 1, Air Force Weapons Laboratory, Kirtland Air Force Base, N.M., May, 1975.

18. Holzer, S. M. and Somers, Jr., A. E., "Nonlinear Model/Solution Process: An Energy Approach," Journal of the Engineering Mechanics Division, ASCE, August, 1977, pp. 629-647.

19. Houbolt, J. C., "A Recurrence Matrix Solution for the Dynamic Response of Elastic Air Craft," Journal of the Aeronautical Sciences, Vol. 17, Sept., 1950, pp. 540-550.

20. Hughes, T. J. R., "Stability, Convergence, and Growth and Decay of Energy of the Average Acceleration Method in Nonlinear Structural Dynamics," Computers and Structures, Vol. 6, 1976, pp. 313-324.

21. Jones, R. F., Jr., and Costello, M. G., "A Solution Procedure for Nonlinear Structural Problems," Sumposium on Numerical Solution of Nonlinear Structural Problems, ASME, Detroit, Michigan, Dec., 1973, pp. 157-169.

22. Kalev, I. and Gluck, J., "Cyclic Elastic-Plastic Dynamic Analysis by the Finite Element Method," Computer Methods in Applied Mechanics and Engineering, Vol. 10, 1977, pp. 63-74.

23. Kamat, M. P., "Sensitivity of Inelastic Response to Numerical Integration of Strain Energy," International Journal for Numerical Methods in Engineering, Vol. 10, 1976, pp. 1369-1377.

24. Key, S. W. and Beisinger, Z. E., "The Transient Dynamic Analysis of Thin Shells by the Finite Element Method," Third Conference on Matrix Methods in Structural Mechanics, Wright-Patterson Air Force Base, Ohio, October, 1971.

25. Krieg, R. D., "Unconditional Stability in Numerical Time Integration," Journal of Applied Mechanics, Vol. 40, June, 1973, pp. 417-421.

26. Krieg, R. D. and Key, S. W., "Transient Shell Response by Numerical Time Integration," International Journal for Numerical Methods in Engineering, Vol. 7, 1973, pp. 273-286.

27. Leech, J. W., Hsu, P. T., and Mack, E. W., "Stability of a Finite Difference Method for Solving Matrix Equations," AIAA Journal, Vol. 3, 1965, pp. 2172-2173.

28. Mallett, R. H., and Schmit, L. A., Jr., "Nonlinear Structural Analysis by Energy Search," Journal of the Structural Division, ASCE, Vol. 93, No. ST3, June, 1967, pp. 221-234.

29. Marquardt, D. W., "An Algorithm for Least Squares Estimation of Non-linear Parameters," Journal SIAM, Vol. 11, 1963, p. 431.

30. McNamara, J. F., and Marcal, P. V., "Incremental Stiffness Method for Finite Element Analysis of the Nonlinear Dynamic Problem," Proceedings of the International Symposium on Numerical and Computer Methods in Structural Mechanics, Urbana, Illinois, Sept., 1971.

31. Meirovitch, L., ELements of Vibration Analysis, McGraw Hill, New York, N.Y., 1975.

32. Melosh, R. J., and Kelley, D. M., "Prediction of Nonlinear Transient Response of Structures," ASME/AIAA 10th Structures, Structural Dynamics, and Materials Conference, New Orleans, La., April, 1969.

33. Mikkola, M. J., Tuomala, M. T. E., and Sinisalo, H., "Comparison of Numerical Integration Methods in the Analysis of Impulsively Loaded Elasto-Plastic and Viscoplastic Structures," to be published.

34. Mondkar, D. P. and Powell, G. H., "Finite Element Analysis of Non-linear Static and Dynamic Response," International Journal for Numerical Methods in Engineering, Vol. 11, 1977, pp. 499-520.

35. Morris, N. F., "The Use of Modal Superposition in Nonlinear Dynamics," Computers and Structures, Vol. 7, No. 1, Feb., 1977, pp. 65-72.

36. Nagarajan, S. and Popov, E. P., "Non-linear Dynamic Analysis of Axisymmetric Shells," International Journal for Numerical Methods in Engineering, Vol. 9, 1975, pp. 535-550.

37. Newmark, N. M., "A Method of Computation for Structural Dynamics," Journal of the Engineering Mechanics Division, ASCE, July, 1959, pp. 67-94.

38. Nickell, R. E., "Nonlinear Dynamics by Mode Superposition," Computer Methods in Applied Mechanics and Engineering, Vol. 7, 1976, pp. 107-129.

39. Nickell, R. E., "Direct Integration Methods in Structural Dynamics," Journal of the Engineering Mechanics Division, ASCE, Vol. 99, April, 1973, pp. 303-317.

40. Nickell, R. E., "On the Stability of Approximation Operators in Problems of Structural Dynamics," International Journal of Solids and Structures, Vol. 7, 1971, pp. 301-319.

41. Oden, J. T., Finite Elements of Nonlinear Continua, McGraw-Hill, New York, New York, 1972, p. 262.

42. Park, K. C., "Practical Aspects of Numerical Time Integration," Computers and Structures, Vol. 7, 1977, pp. 343-353.

43. Powell, M. J. D., "A Fortran Subroutine for Solving Systems of Nonlinear Algebraic Equations," Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz, ed., Gordon and Breach, London, 1970, pp. 115-161.

44. Schmit, L. A., Jr., Bogner, F. K., and Fox, R. L., "Finite Deflection Structural Analysis Using Plate and Cylindrical Shell Discrete Elements," AIAA/ASME 8th Structures, Structural Dynamics and Materials Conference, Palm Springs California, March, 1967, pp. 197-211.

45. Shifflett, R. B., "Analysis of Nonlinear Beam Columns," Thesis presented to Virginia Polytechnic Institute and State University, Blacksburg, Virginia, in 1977, in partial fulfillment of the requirements for the degree of Master of Science.

46. Stanton, E. L., and McGovern, D. J., "The Application of Gradient Minimization Methods and Higher Order Discrete Elements to Shell Buckling and Vibration Eigen Problems," Computers and Structures, Vol. 1, 1971, pp. 413-434.

47. Stinnett, D. B., "Dynamic Analysis of Inelastic and Nonlinear Plane Frames by Direct Energy Minimization," Thesis presented to Virginia Polytechnic Institute and State University, Blacksburg, Virginia, in 1977, in partial fulfillment of the requirements for the degree of Master of Science.

48. Stricklin, J. A., and Haisler, W. E., "Formulation and Solution Procedures for Nonlinear Structural Analysis," Computers and Structures, Vol. 7, 1977, pp. 125-136.

49. Stricklin, J. A., Martinez, J. E., Tillerson, J. R., Hong, J. H., and Haisler, W. E., "Nonlinear Dynamic Analysis of Shells of Revolution by Matrix Displacement Method," AIAA Journal, Vol. 9, 1971, pp. 629-636.

50. Stricklin, J. A. Haisler, W. E., and Von Riesemann, W. A., "Large Deflection Elastic-Plastic Dynamic Response of Stiffened Shells of Revolution," Journal of Pressure Vessel Technology, ASME, Vol. 96, No. 2, May, 1974, pp. 87-95.

51. Suryoutomo, H., Gould, P. L., and Prodyst, K. B., "Direct Dynamic Analysis of Shells of Revolution Using High Precision Finite Elements," Computers and Structures, Vol. 7, 1977, pp. 425-433.

52. Tillerson, J. R., Stricklin, J. A., and Haisler, W. E., "Numerical Methods for the Solution of Nonlinear Problems in Structural Analysis," Symposium on Numerical Solution of Nonlinear Structural Problems, ASME, Detroit, Michigan, Dec., 1973, pp. 67-101.

53. Timoshenko, S. P. and Gere, J. M., Theory of Elastic Stability, McGraw-Hill, New York, 1961.

54. Weeks, G., "Temporal Operators for Nonlinear Structural Dynamics Problems," Journal of the Engineering Mechanics Division, ASCE, Vol. 98, 1972, pp. 1087-1104.

55. Wilson, E. L., Farhomand, I., and Bathe, K. J., "Nonlinear Dynamic Analysis of Complex Structures," Earthquake Engineering and Structural Dynamics, 1973, pp. 241-252.

56. Wu, R. W., and Wittmer, E. A., "Nonlinear Transient Responses of Structures by the Spatial Finite Element Method," AIAA Journal, Vol. 11, 1973, pp. 1110-1117.

57. Zienkiewicz, O. C., The Finite Element Method in Engineering Science, McGraw-Hill, London, 2nd Edition, 1971.

APPENDIX B

USER'S GUIDE

USER'S GUIDE

The computer code in Appendix C is for the static and dynamic analysis of nonlinear plane frame and truss systems by algorithm 2. Only geometric nonlinearities are included in the formulation. Since material nonlinearities are not considered in the formulation of the computer code, the internal energy and the variation of the internal energy with respect to the generalized coordinates are obtained in closed form. Minimization of the objective function is accomplished by the hybrid method of Powell [43].

The computer code, written in the WATFIV computer language, requires approximately 450 kilo-bytes of computer storage space. An input guide for the data is included in the program listing. The format for the data for this program and the format for the data needed for the program written by Stinnett [47] are virtually the same. The only difference in the data sets is the definition and use of three variables, EPS, EST, and IPR. The variable EPS differs only in its usage; EST for the program in Appendix C is defined as an estimate of the step length used to approximate derivatives while, in Stinnett's program, EST is an estimate of the value of the objective function; IPR is not used in Stinnett's program. Following a listing and a description of the subroutines used in the computer programs, flow charts of the programs are presented.

Subroutine	Purpose
MAIN	To initialize variables, to set problem parameters, and to increment loads and time.
FUNCT	To form the objective function and, if needed, the gradients of the objective function.
SRAN	To compute the strain at a point in an element given the point and the local deformation vector. To uncouple the strain-displacement relation, set ALFA and BETA to zero.
DEFO	To compute the local deformation vector of an element. To simulate truss behavior set the element rotation components, $\bar{u}_2$ and $\bar{u}_3$, equal to zero and constrain all joints against rotation.
STNG	To compute the internal energy of an element and the variation of the internal energy with respect to the local deformation vector.
GRAD	To transform the variation of the internal energy with respect to the local deformation vector into the variation of the internal energy with respect to the global deformation vector.
STENDN	To compute the internal-energy density at a point. This subroutine is needed only for nonlinear material behavior.
MASS	To compute the nodal force due to the weight of an element and to compute the mass contribution of an element at the external nodes.
IMBAL	To compute the errors at the end of a time (load) step.
INRTIA	To compute the inertial force contribution to an energy function. This subroutine is not needed for the objective function G^TG.
POTE	To compute the contribution of the generalized forces to an energy function. This subroutine is not needed for the objective function G^TG.

Subroutine	Purpose
DFMFP	To minimize the objective function by the Davidon-Fletcher-Powell method.
NS01AD	To minimize the objective function by the hybrid method of Powell. This subroutine calls the subroutine MB01AD which inverts the approximate Jacobian matrix.
OSIRIS	To call the minimization routine and to write out data following the minimization for a load (time) step.

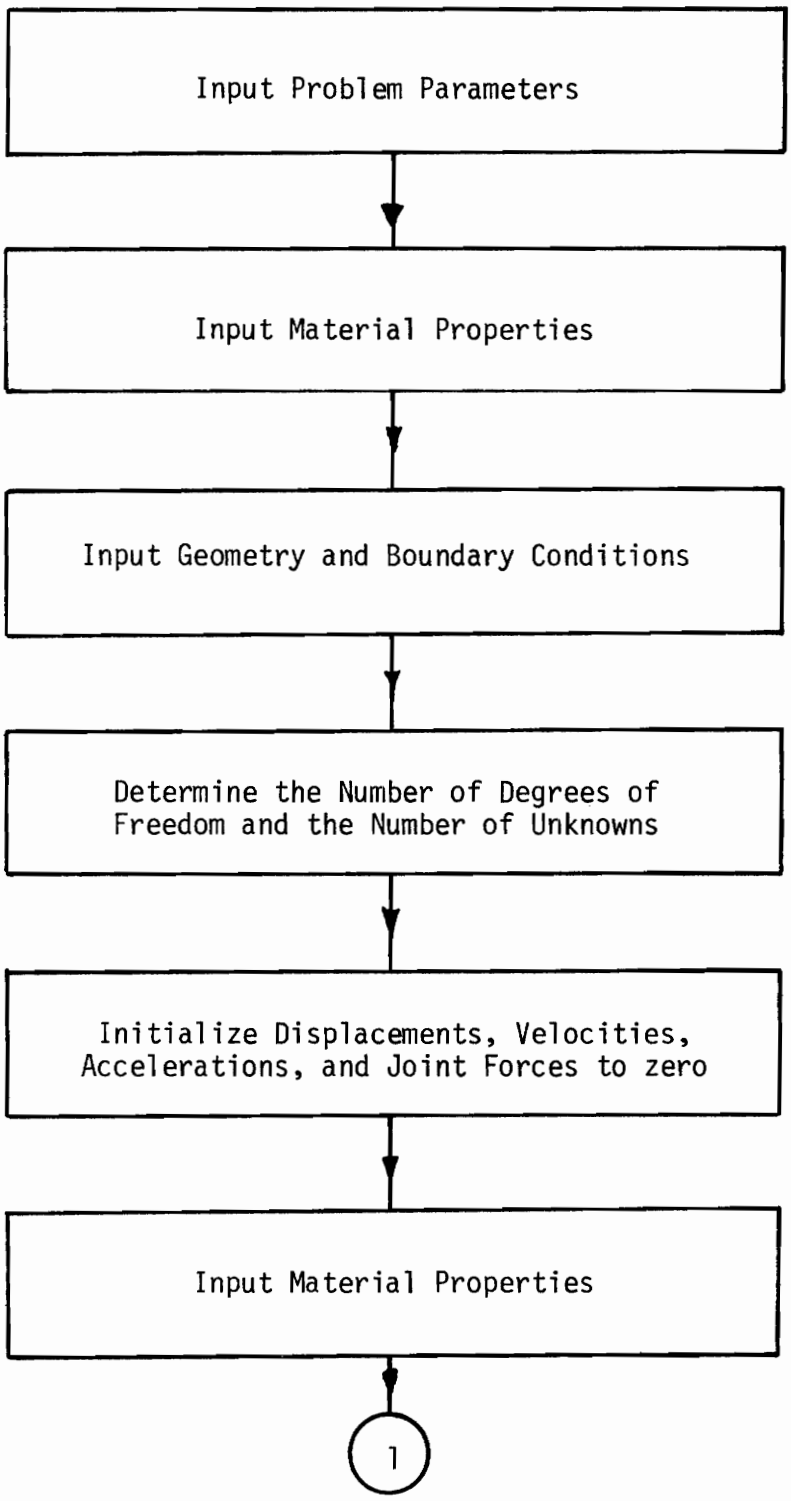


Fig. B.1. Flow Chart of MAIN Program

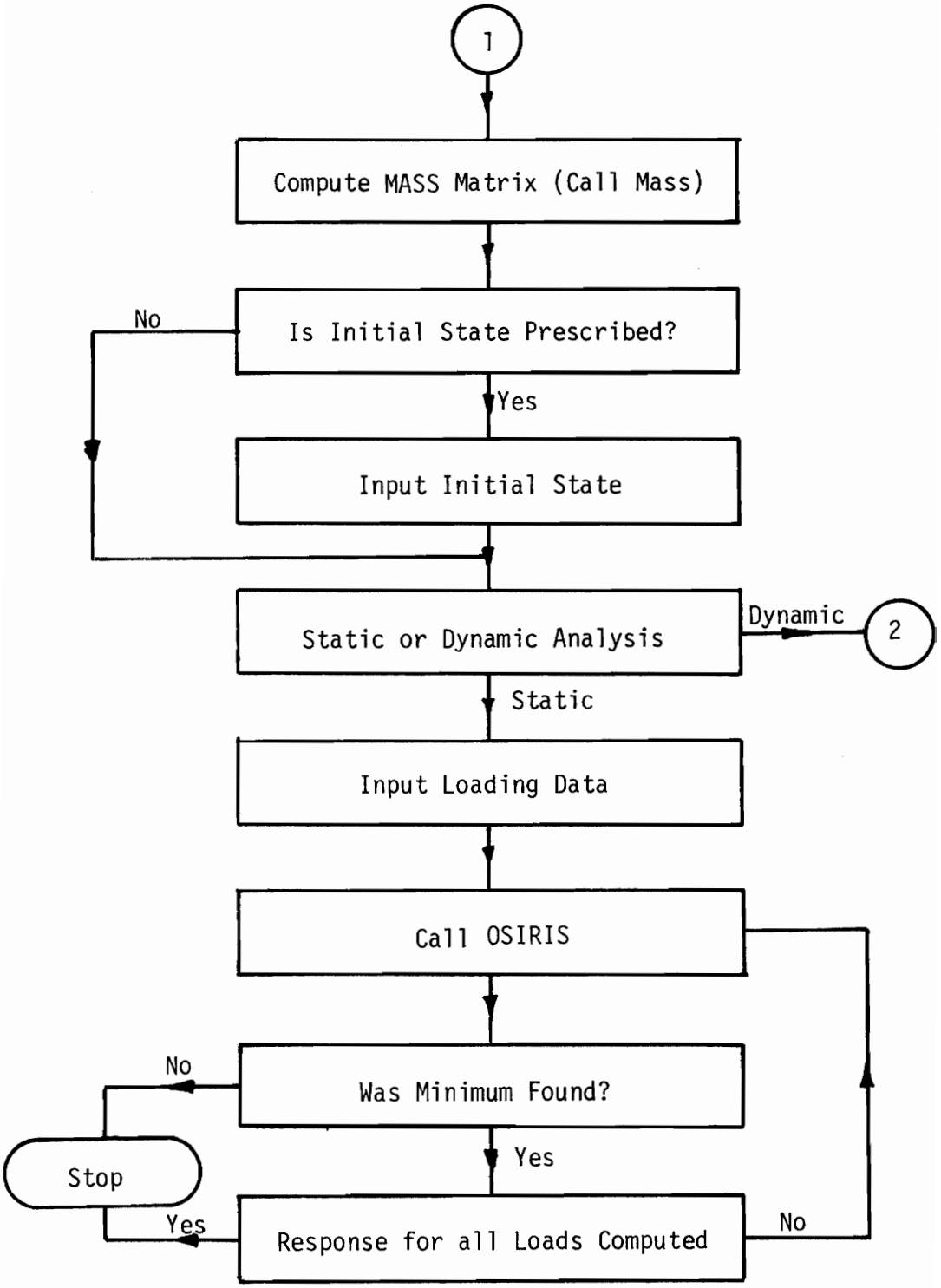


Fig. B.1. (Continued)

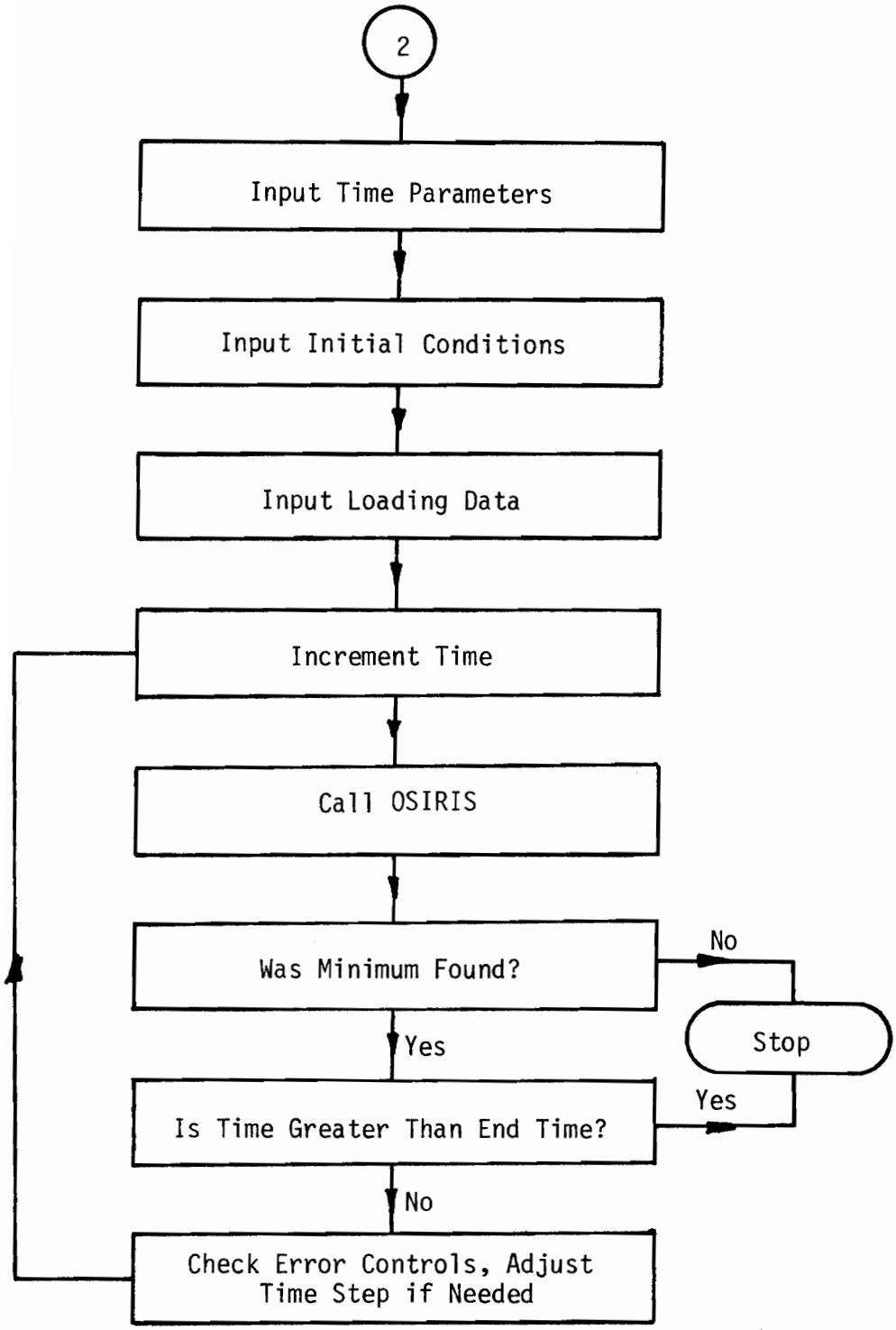


Fig. B.1. (Continued)

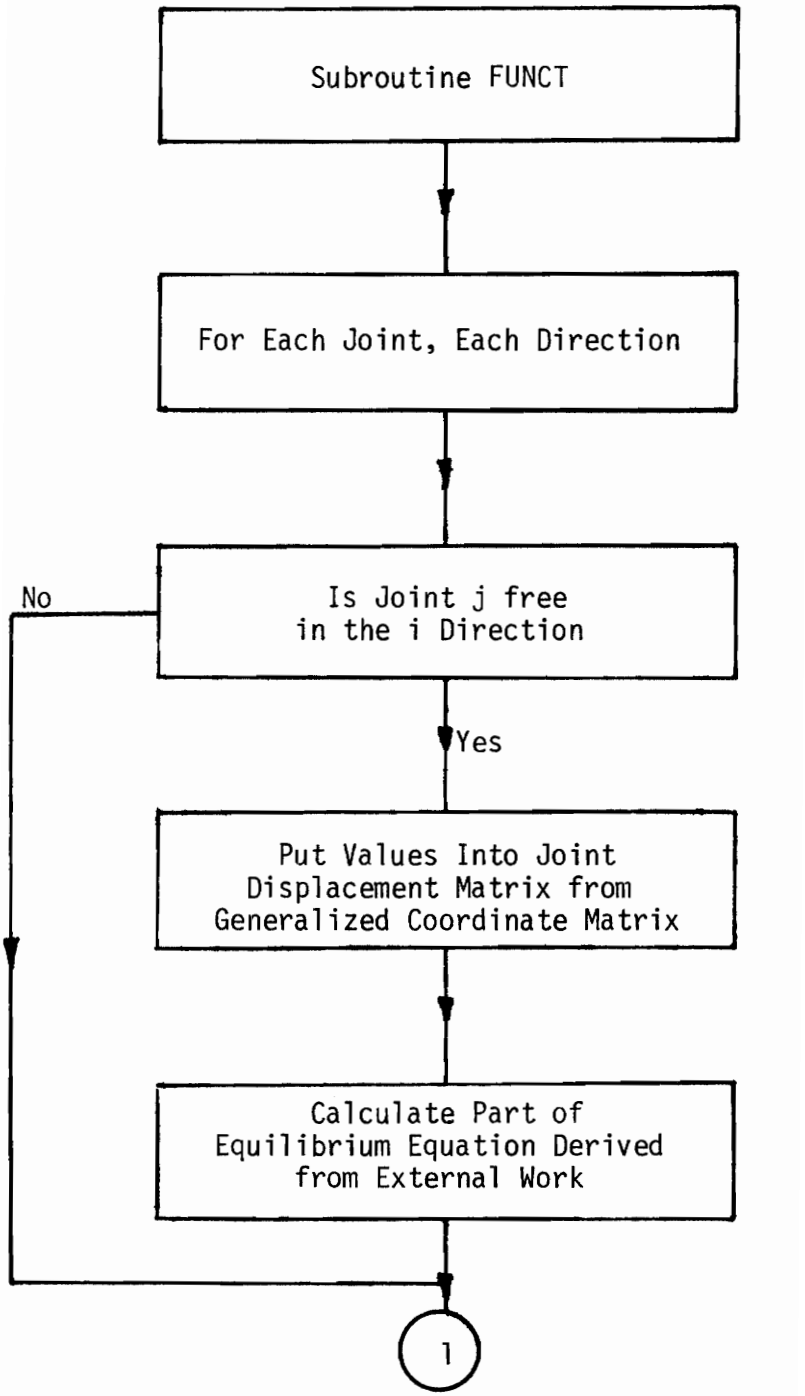


Fig. B.2. Flow Chart of Subroutine FUNCT

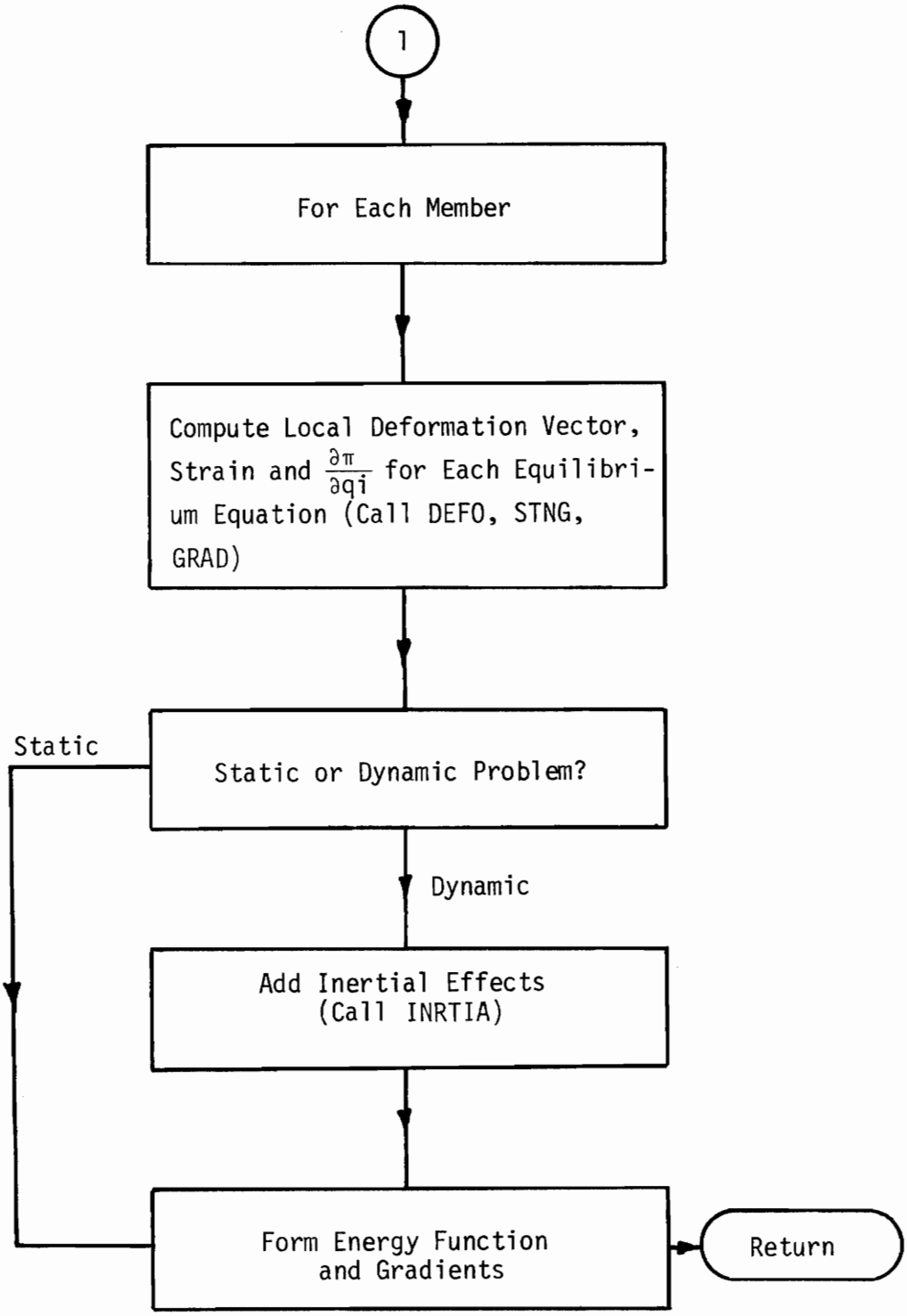


Fig. B.2. (Continued)

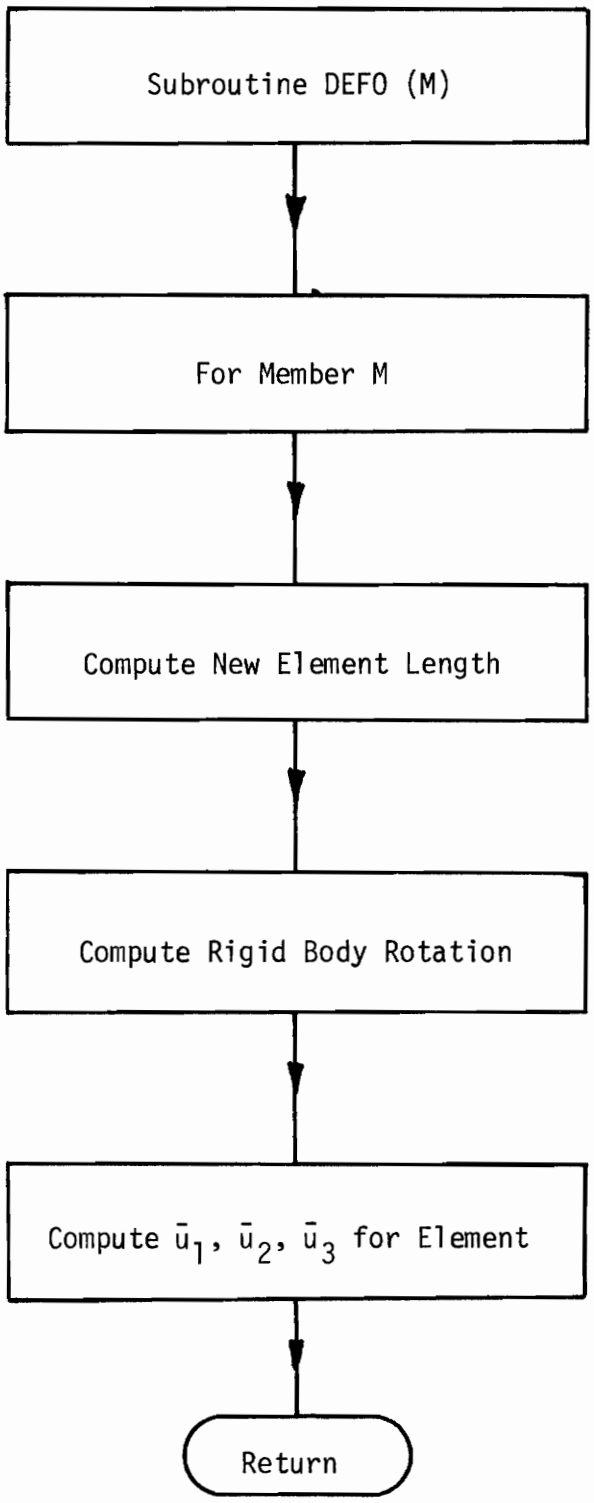


Fig. B.3. Flow Chart of Subroutine DEFO

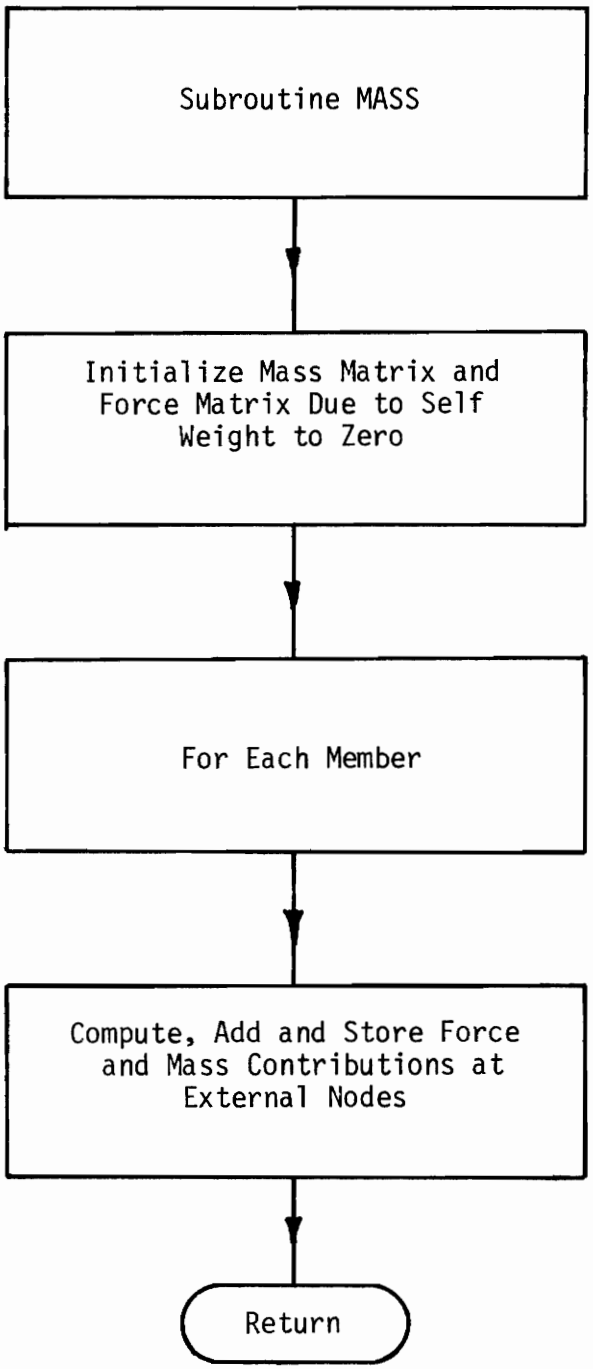


Fig. B.4. Flow Chart of Subroutine MASS

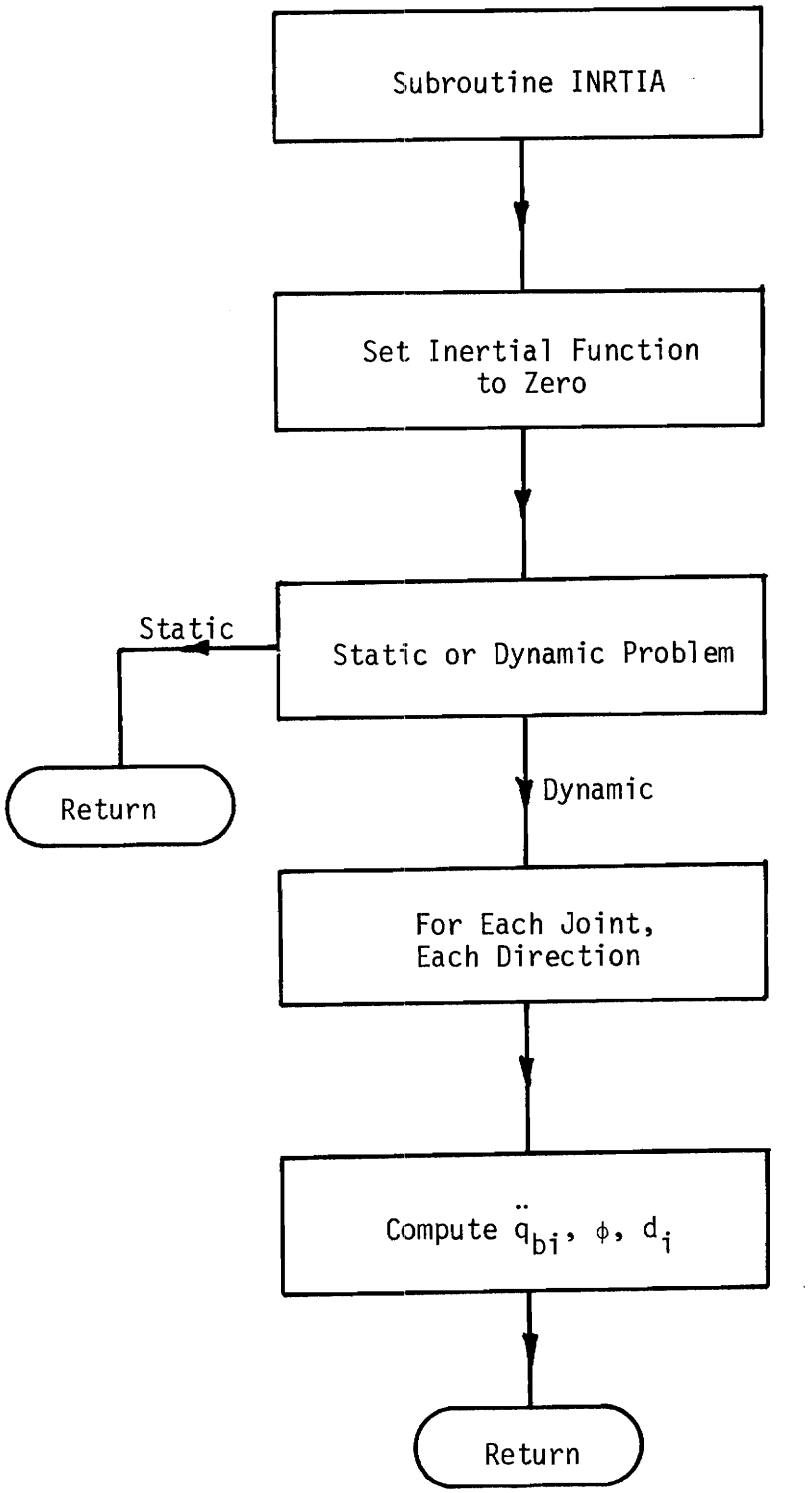


Fig. B.5. Flow Chart of Subroutine INRTIA

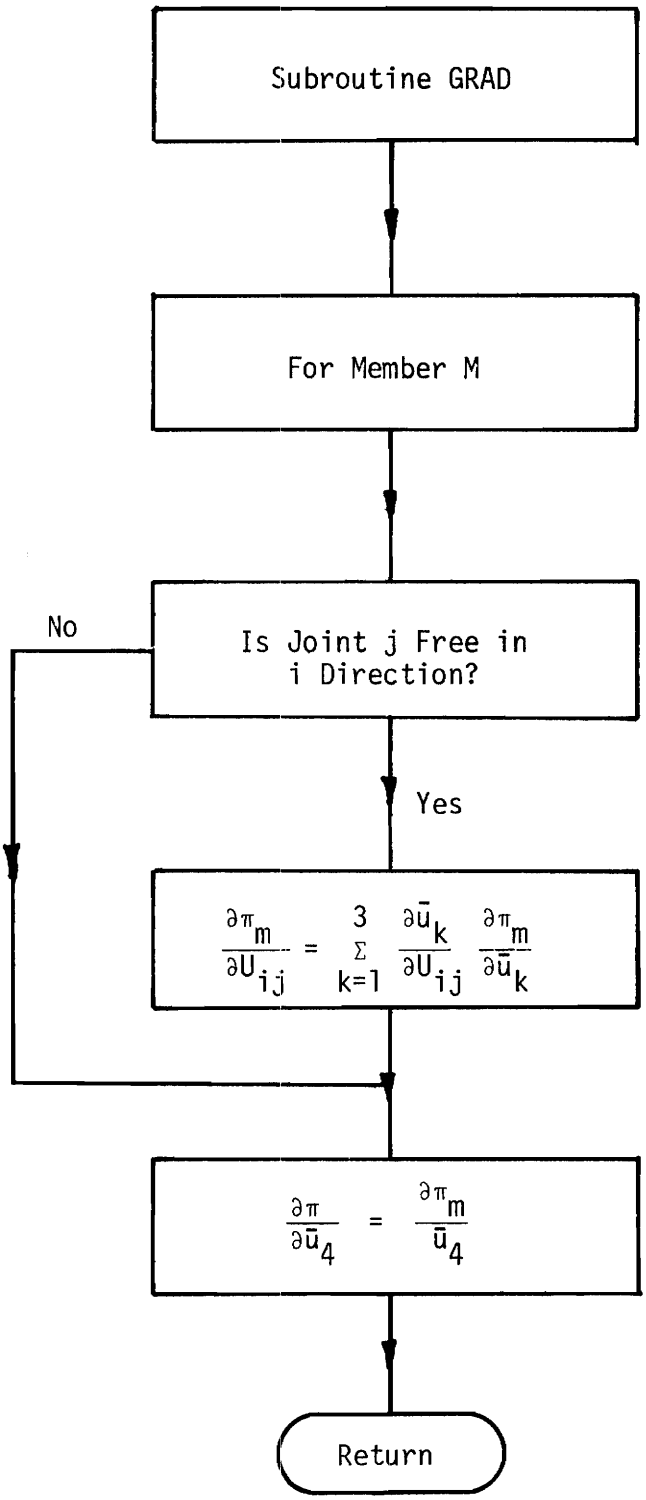


Fig. B.6. Flow Chart of Subroutine GRAD

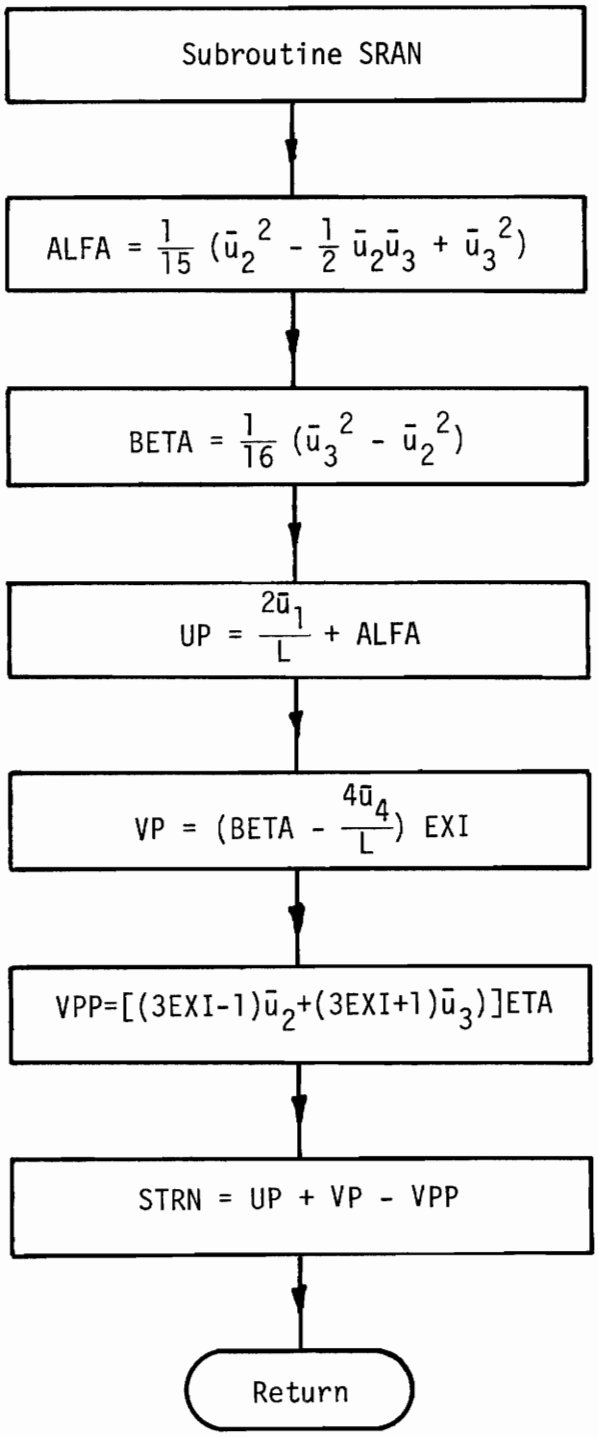


Fig. B.7. Flow Chart of Subroutine SRAN

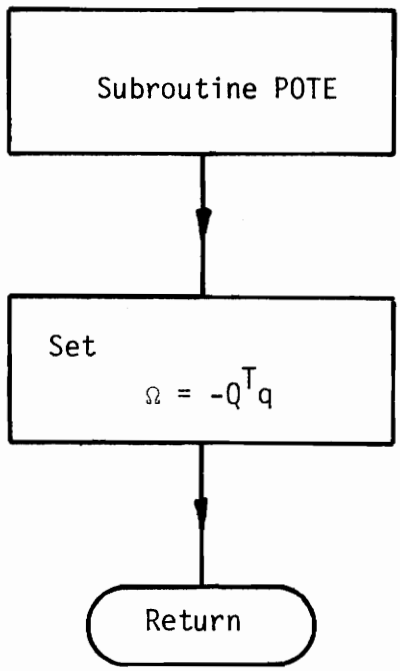


Fig. B.8. Flow Chart of Subroutine POTE

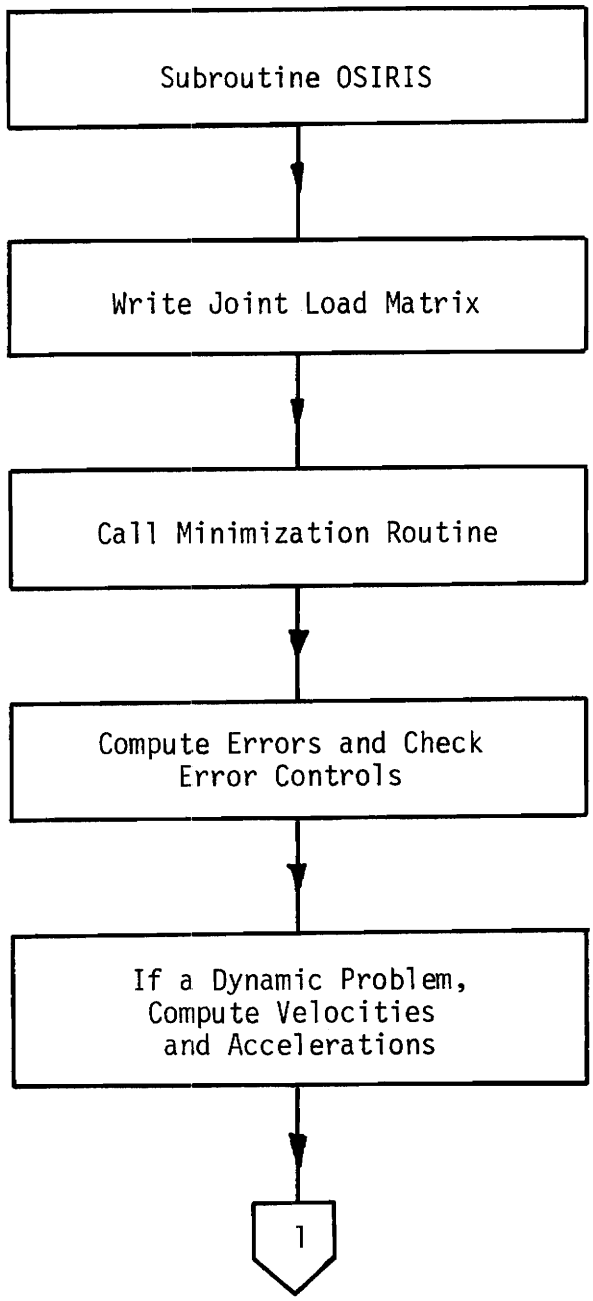


Fig. B.9. Flow Chart of Subroutine OSIRIS

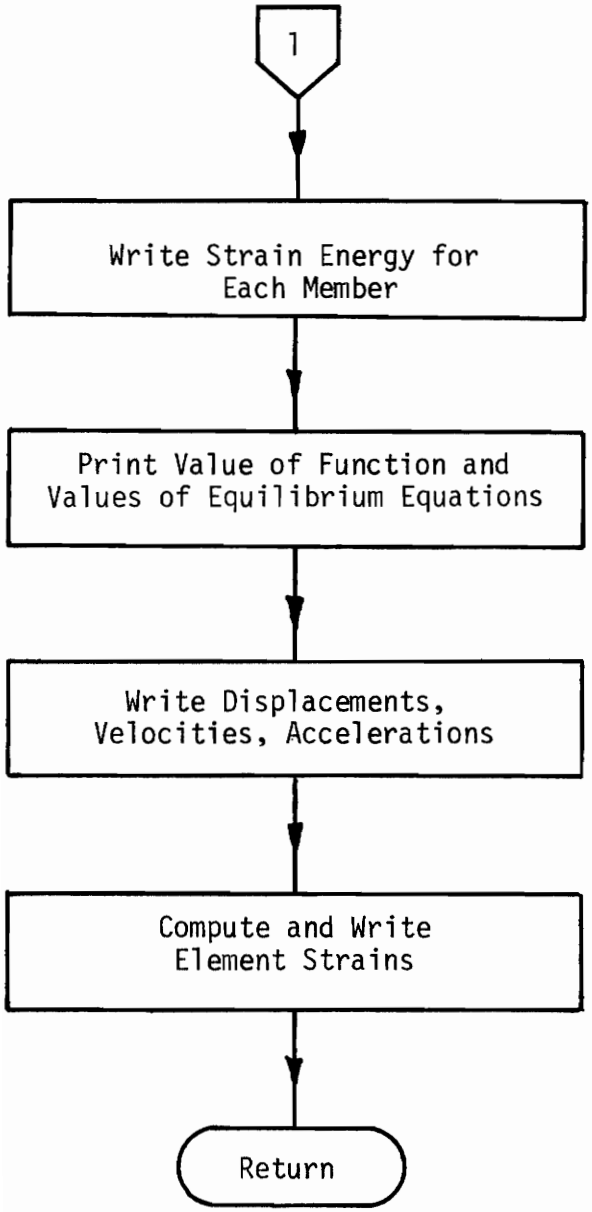


Fig. B.9. (Continued)

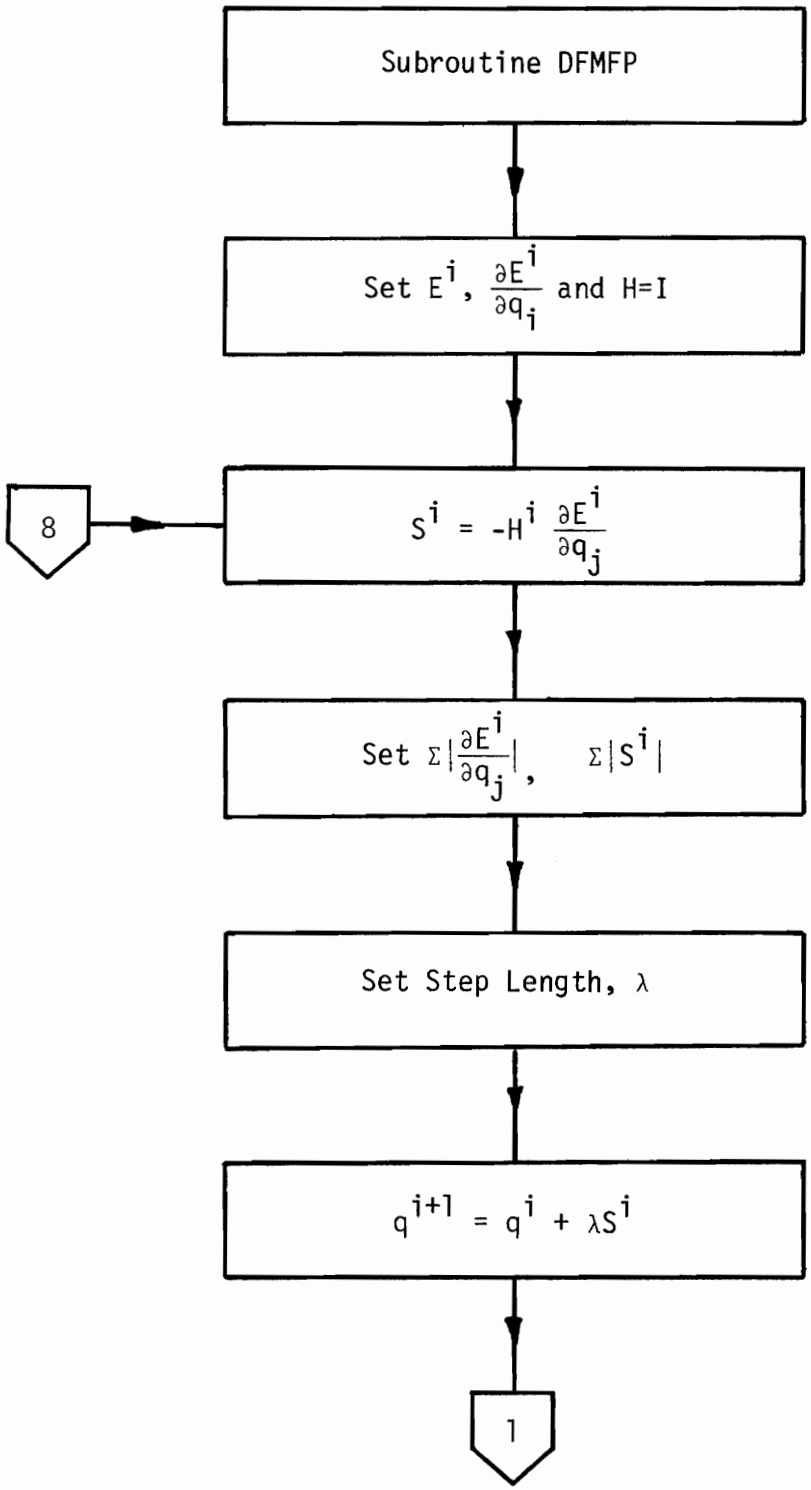


Fig. B.10. Flow Chart of Subroutine DFMFP

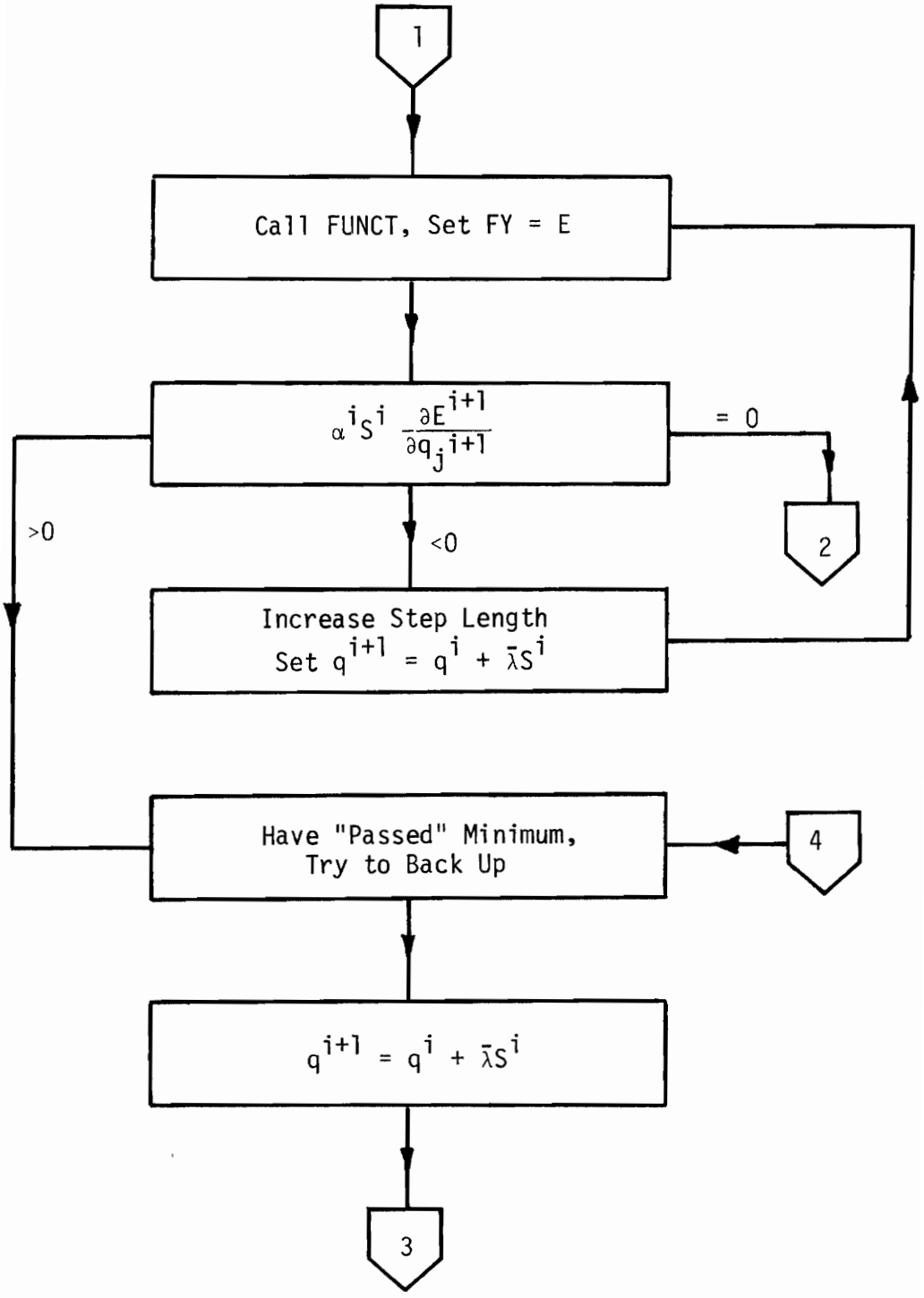


Fig. B.10. (Continued)

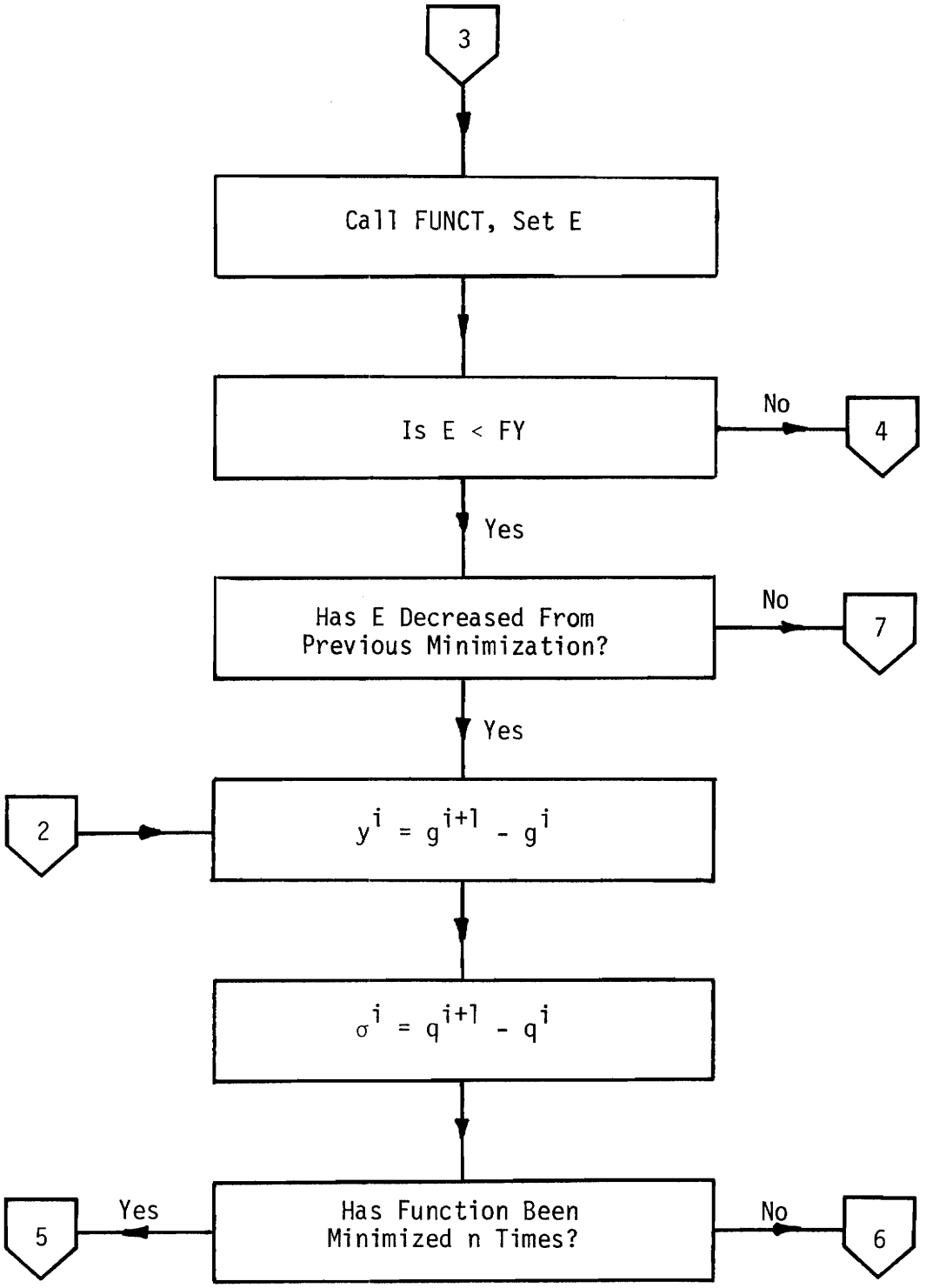


Fig. B.10. (Continued)

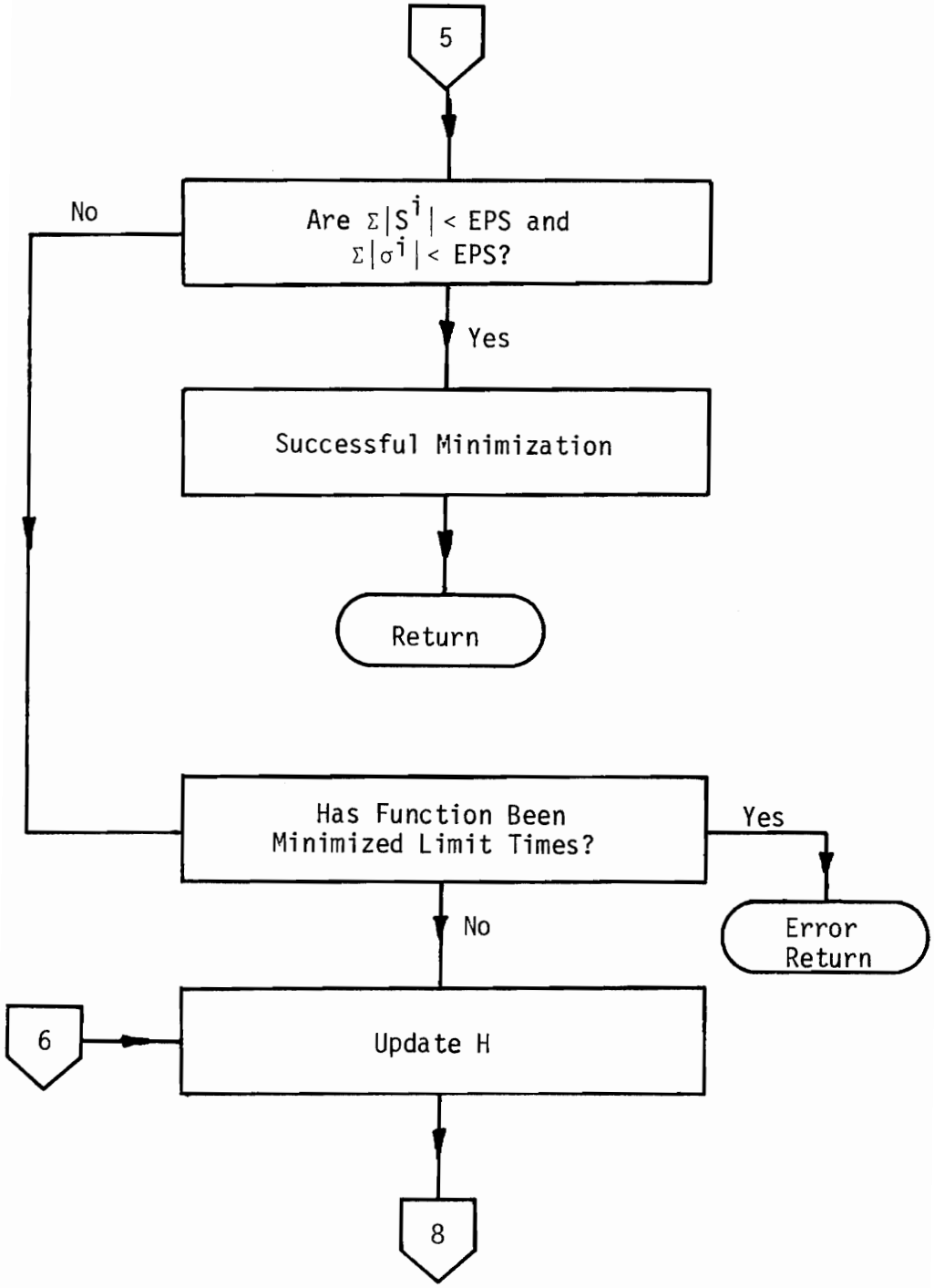


Fig. B.10. (Continued)

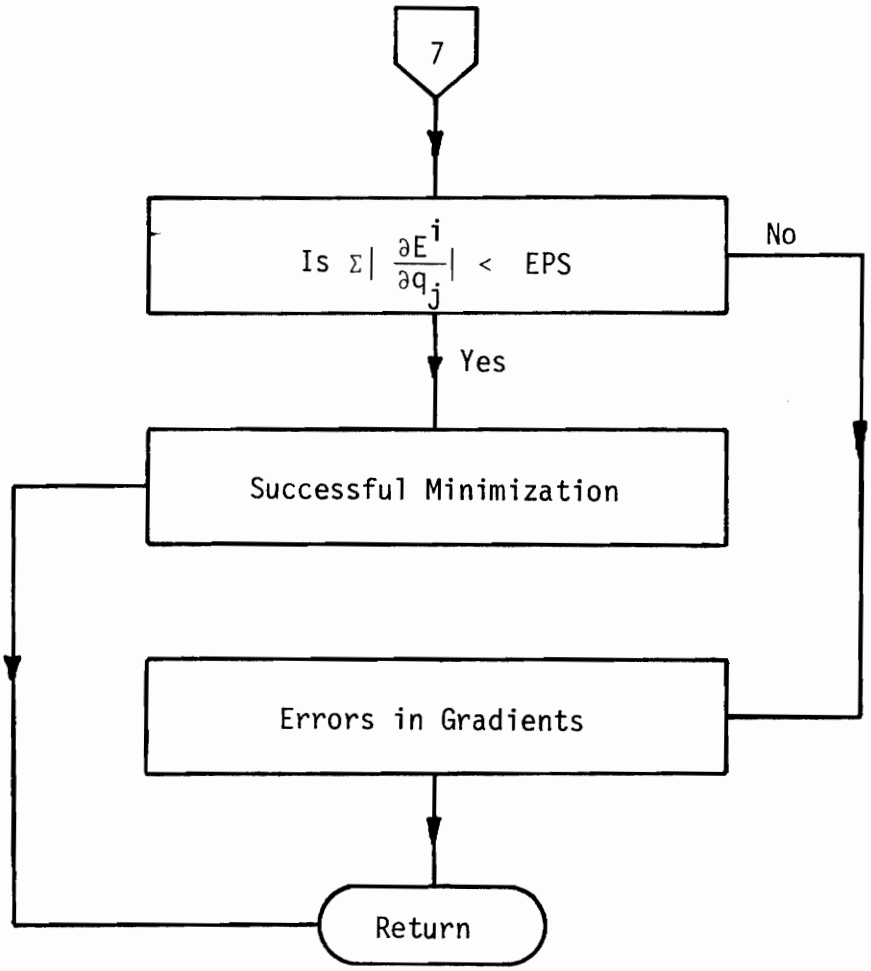


Fig. B.10. (Continued)

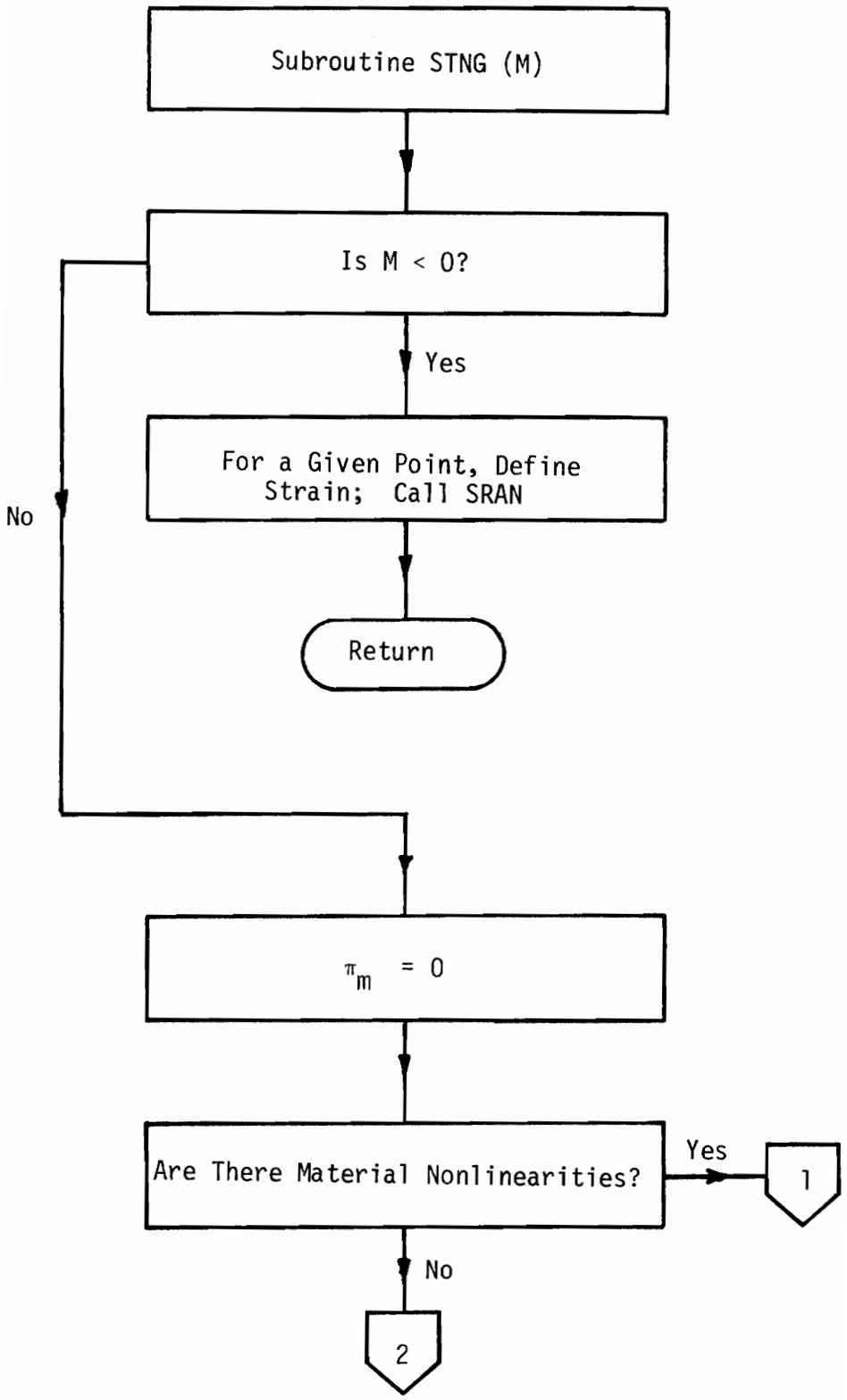


Fig. B.11. Flow Chart of Subroutine STNG

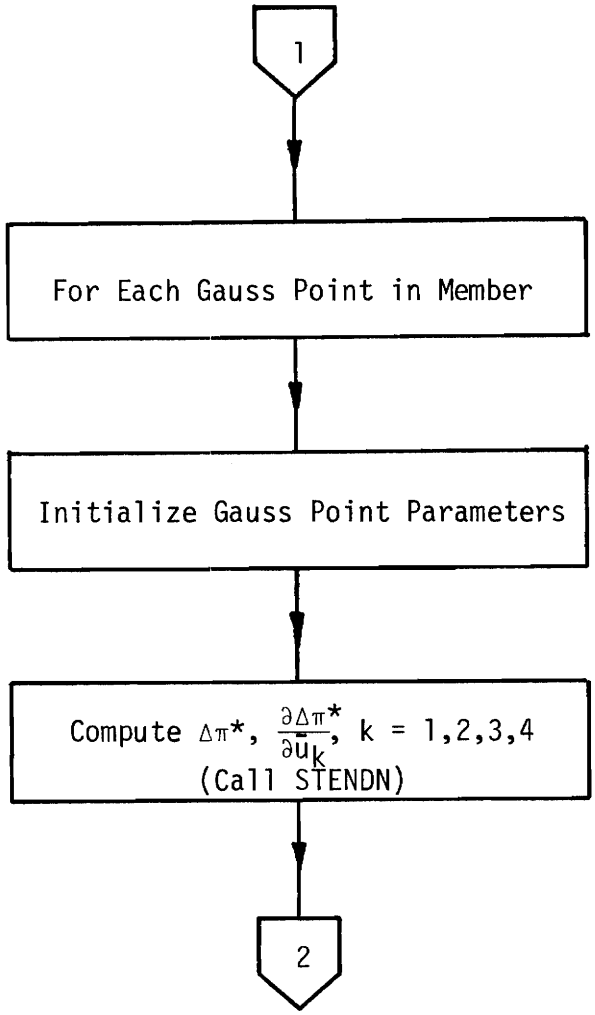


Fig. B.11. (Continued)

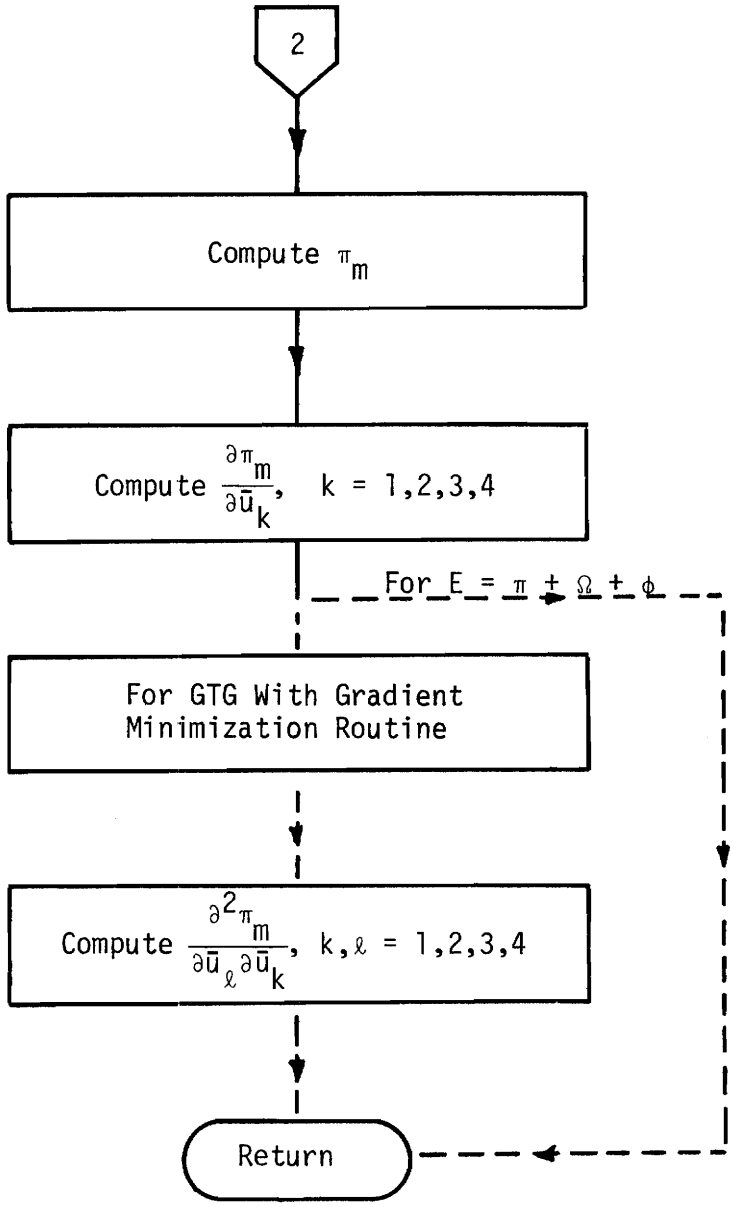


Fig. B.11. (Continued)

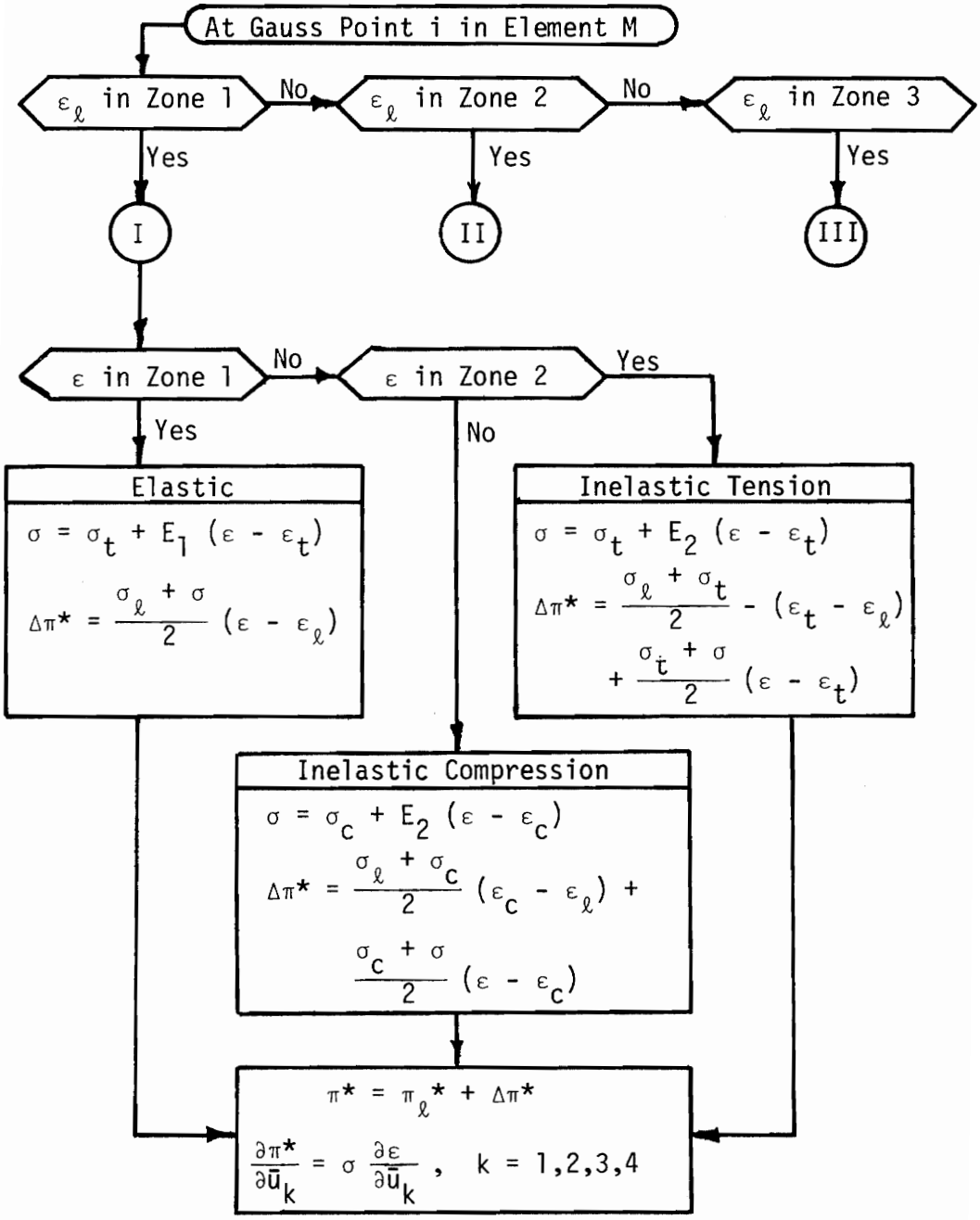


Fig. B.12. Flow Chart of Subroutine STENDN
(Reproduced from Ref. 45)

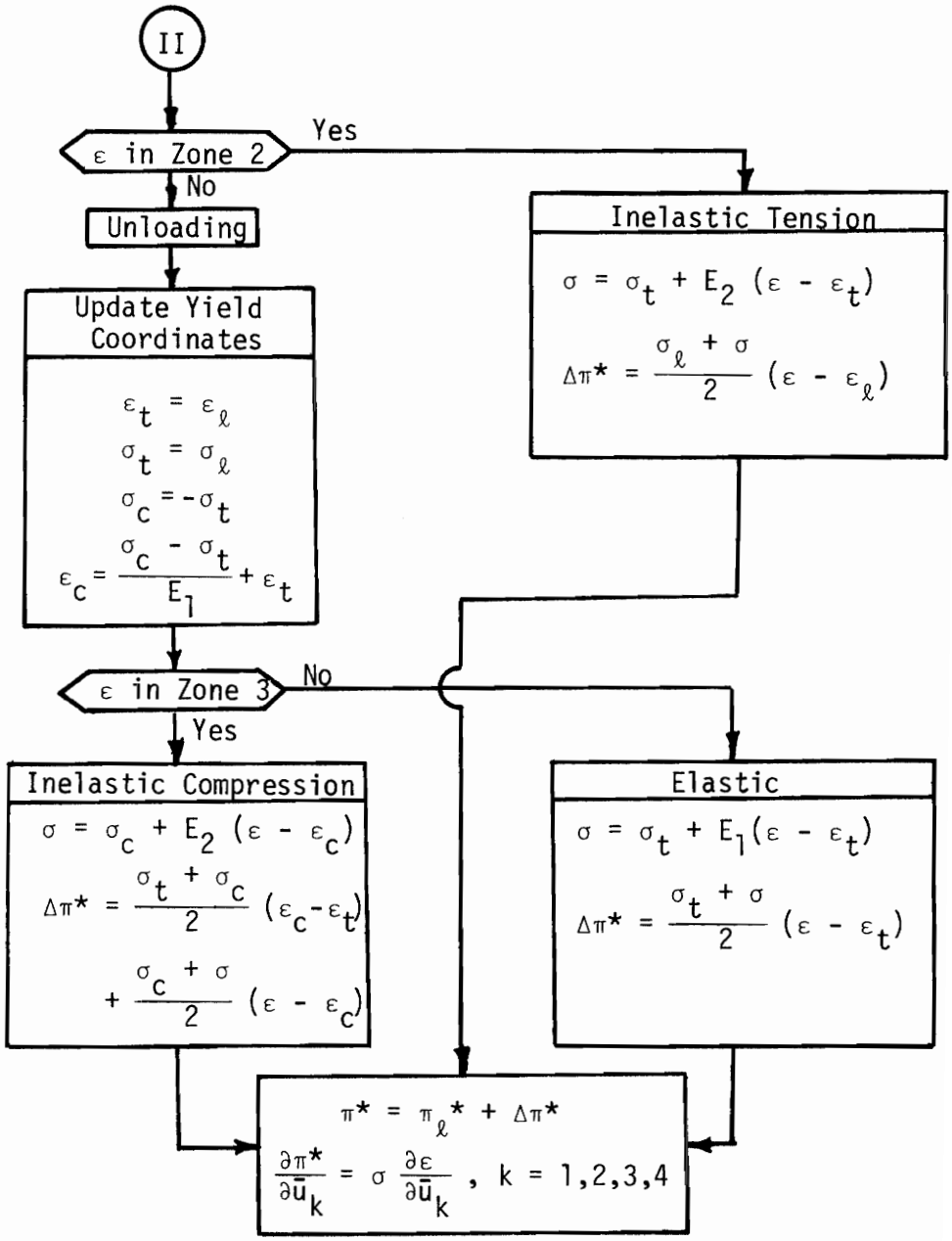


Fig. B.12. (Continued)

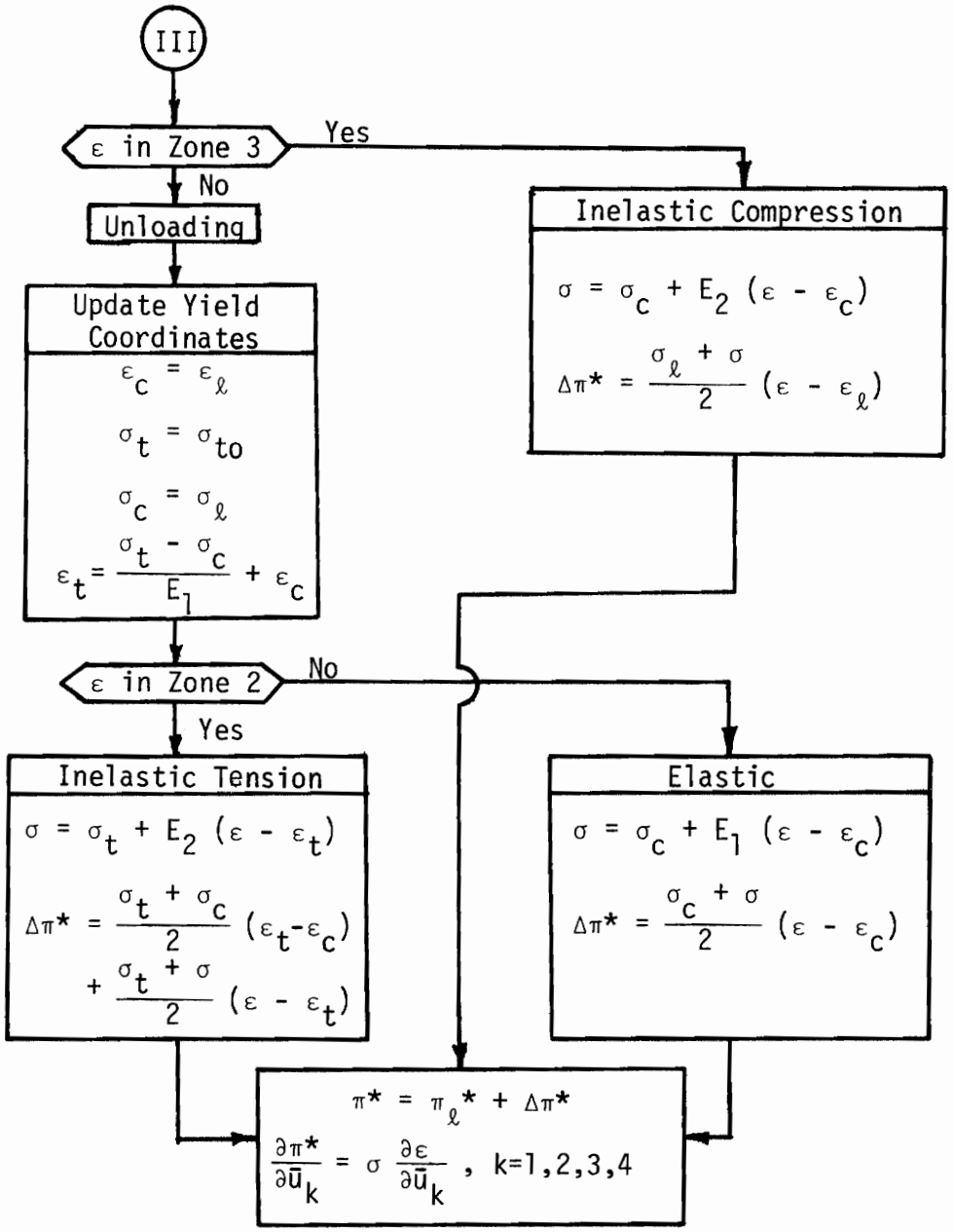


Fig. B.12. (Continued)

APPENDIX C

PROGRAM LISTING

```
C     CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM
C                             INPUT GUIDE                           NSAM
C                                                                   NSAM
C                                                                   NSAM
C     CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM
C     FIRST CARD:  (20A1)                                           NSAM
C      TITLE -ANY INFORMATION BY WHICH THE USER MAY IDENTIFY THE PROBLEMNSAM
C                                                                   NSAM
C     SECOND CARD:  (A4)                                            NSAM
C                                                                   NSAM
C     CODE EITHER STATIC OR DYNAMIC - MUST BE LEFT JUSTIFIED WITH COLUMNNSAM
C     ONE.                                                          NSAM
C      THIS CARD DETERMINES WHETHER STATIC OR DYNAMIC ANAL. IS PERFORMEDNSAM
C                                                                   NSAM
C     THIRD CARD:  (6I5,F20.16,F10.5,I5)                            NSAM
C      NJ,NM,NF,LIMIT,NCONF,INICON,EPS,EST,IPR                      NSAM
C                                                                   NSAM
C        NJ= NUMBER OF JOINTS                                       NSAM
C        NM= NUMBER OF MEMBERS                                      NSAM
C        NF= 4 FOR DYNAMIC ANALYSIS                                 NSAM
C          = NUMBER OF LOAD STEPS FOR STATIC ANALYSIS, MAXIMUM = 80 NSAM
C        LIMIT= MAXIMUM NUMBER OF MINIMIZATIONS ALLOWED FOR ONE TIME OR NSAM
C                LOAD STEP  (DEFAULT = 40)                          NSAM
C        NCONF= 0,1 FOR STATIC ANALYSIS                             NSAM
C                 0= GIVEN THE INITIAL LOAD AND A LOAD INCREMENT, NF NSAM
C                    LOADS WILL BE COMPUTED                         NSAM
C                 1= ALL NF LOADS MUST BE INPUTTED BY USER          NSAM
C                                                                   NSAM
C                  = BLANK FOR DYNAMIC ANALYSIS                     NSAM
C                                                                   NSAM
C        INICON= 0 IF NO PRESCRIBED CONFIGURATION FOR STRUCTURE INPUTTEDNSAM
C              = 1 IF PRESCRIBED INITIAL CONFIGURATION INPUTTED     NSAM
```

```
C                                                                       NSAM  33
C        EPS= DESIRED ACCURACY FOR MINIMIZATION OF OBJECTIVE FUNCTION   NSAM  34
C             (DEFAULT = .00000001 )                                    NSAM  35
C                                                                       NSAM  36
C        EST= ESTIMATE OF STEP LENGTH TO APPROXIMATE DERIVATIVES        NSAM  37
C            (DEFAULT = 0.00001 )                                       NSAM  38
C                                                                       NSAM  39
C        IPR = CONTROLS FREQUENCY OF OUTPUT FROM NSO1AD                 NSAM  40
C            = 0 NO OUTPUT FROM NSO1AD UNLESS IN CASE OF AN ERROR       NSAM  41
C            =+K, INTERMEDIATE OUTPUT FROM NSO1AD EVERY K ITERATIONS    NSAM  42
C            =-K,  OUTPUT FROM NSO1AD ONLY UPON COMPLETION OF ROUTINE   NSAM  43
C                                                                       NSAM  44
C     FOURTH CARD:  (F10.0)                                             NSAM  45
C                                                                       NSAM  46
C          E = MODULUS OF ELASTICITY                                    NSAM  47
C                                                                       NSAM  48
C                                                                       NSAM  49
C     NEXT NJ CARDS (I5,5X,2F10.5,2X,3A1,5X,3F10.5)                     NSAM  50
C          I,X(I),Y(I),IR(I,1),IR(I,2),IR(I,3),XDJ(I,1),XDJ(I,2),XDJ(I,3NSAM  51
C                                                                       NSAM  52
C          WHERE:                                                       NSAM  53
C                                                                       NSAM  54
C            X= X-COORDINATE OF I-TH JOINT                              NSAM  55
C            Y= Y-COORDINATE OF I-TH JOINT                              NSAM  56
C            I= JOINT NUMBER                                            NSAM  57
C            IR(I,J) = CONSTRAINT CODE FOR I-TH JOINT, J-TH DIRECTION   NSAM  58
C                                                                       NSAM  59
C                   CODE R FOR CONSTRAINED                              NSAM  60
C                   CODE 0 FOR UNCONSTRAINED                            NSAM  61
C                                                                       NSAM  62
C            XDJ(I,J)= CONSTRAINED DISPLACEMENT OF I-TH JOINT, J-TH DIR.NSAM  63
C                                                                       NSAM  64
```

```
C                                                                       NSAM  65
C     NEXT NM CARDS (3I5,5X,4F10.4)                                     NSAM  66
C          M,IP(M),IQ(M),B(M),H(M),A(M),DENS(M)                         NSAM  67
C          WHERE:                                                       NSAM  68
C                                                                       NSAM  69
C                                                                       NSAM  70
C            M=MEMBER                                                   NSAM  71
C                                                                       NSAM  72
C                                                                       NSAM  73
C           IP(M) = JOINT WHERE M-TH MEMBER BEGINS                      NSAM  74
C           IQ(M) = JOINT WHERE M-TH MEMBER ENDS                        NSAM  75
C           B(M)  = WIDTH OF M-TH MEMBER                                NSAM  76
C           H(M)  = HEIGHT OF M-TH MEMBER                               NSAM  77
C           A(M)  = DISTANCE FROM REFERENCE AXIS TO CENTROIDAL AXIS FOR  NSAM  78
C                   M-TH MEMBER                                         NSAM  79
C                                                                       NSAM  80
C                                                                       NSAM  81
C     CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM  82
C     IF INICON = 0, SKIP NEXT SECTION                                  NSAM  83
C     CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM  84
C                 INPUT PRESCRIBED INITIAL CONFIGURATION                NSAM  85
C                 (THIS SECTION PROVIDES AN INITIAL GUESS FOR           NSAM  86
C                  THE MINIMIZATION PROCESS)                            NSAM  87
C     ONE CARD FOR EACH JOINT DISPLACEMENT INITIALIZED (I5,5X,3F20.16)  NSAM  88
C                                                                       NSAM  89
C         J,Q1,Q2,Q3                                                    NSAM  90
C                                                                       NSAM  91
C         WHERE                                                         NSAM  92
C                                                                       NSAM  93
C           J = NUMBER OF JOINT BEING INITIALIZED                       NSAM  94
C           Q1= DISPLACEMENT IN X-DIRECTION                             NSAM  95
C           Q2=DISPLACEMENT IN Y-DIRECTION                              NSAM  96
```

```
C            Q3= DISPLACEMENT IN Z-DIRECTION                                NSAM  97
C                                                                           NSAM  98
C       - PLACE IN BLANK CARD AFTER LAST JOINT INITIALIZED -                NSAM  99
C                                                                           NSAM 100
C      ONE CARD FOR EACH INTERNAL NODE INITIALIZED (I5,5X,F20.10)           NSAM 101
C                                                                           NSAM 102
C            M,Q                                                            NSAM 103
C                                                                           NSAM 104
C            M= NUMBER OF MEMBER IN WHICH INTERNAL NODE LIES                NSAM 105
C            Q= DISPLACEMENT OF INTERNAL NODE                               NSAM 106
C                                                                           NSAM 107
C       - PLACE IN BLANK CARD AFTER LAST INTERNAL NODE INITIALIZED -        NSAM 108
C                                                                           NSAM 109
C                                                                           NSAM 110
C      CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM 111
C                  DYNAMIC ANALYSIS INPUT SECTION                           NSAM 112
C                                                                           NSAM 113
C      CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM 114
C                                                                           NSAM 115
C      IF THIS PROBLEM IS FOR STATIC ANALYSIS, SKIP THIS SECTION            NSAM 116
C      NEXT CARD (3F10.5,I5)                                                NSAM 117
C         TBEG,TEND,T,NTMAX                                                 NSAM 118
C                                                                           NSAM 119
C         WHERE                                                             NSAM 120
C                                                                           NSAM 121
C            TBEG = TIME INTEGRATION BEGINS (SECONDS)                       NSAM 122
C            TEND = TIME INTEGRATION ENDS (SECONDS)                         NSAM 123
C            NTMAX=MAXIMUM NUMBER OF TIME STEPS ALLOWED                     NSAM 124
C                                                                           NSAM 125
C      ONE CARD FOR EACH INITIAL CONDITION (DISP., VEL., OR ACC.)           NSAM 126
C                                                                           NSAM 127
C            (2I5,4X,A1,5X,D20.16)                                          NSAM 128
```

```
C                                                                       NSAM 129
C          I,ID,TYPE,XINCON                                             NSAM 130
C                                                                       NSAM 131
C          WHERE                                                        NSAM 132
C            I = JOINT INITIAL CONDITION APPLIED                        NSAM 133
C            ID= DIRECTION OF INITIAL CONDITION                         NSAM 134
C            TYPE = CODE D FOR DISPLACEMENT                             NSAM 135
C                 = CODE V FOR VELOCITY                                 NSAM 136
C                 = CODE A FOR ACCELERATION                             NSAM 137
C            XINCON= VALUE OF INITIAL CONDITION                         NSAM 138
C         - PLACE IN BLANK CARD AFTER LAST INITIAL CONDITION            NSAM 139
C                                                                       NSAM 140
C      ONE CARD FOR EACH APPLIED FORCE (ONLY LINEARLY VARYING FORCES)   NSAM 141
C                                                                       NSAM 142
C          (2I5,2F10.2)                                                 NSAM 143
C          IJOI,IDIR,AFOR,BFOR                                          NSAM 144
C                                                                       NSAM 145
C          WHERE                                                        NSAM 146
C                                                                       NSAM 147
C            IJOI= JOINT AT WHICH FORCE IS APPLIED                      NSAM 148
C            IDIR= DIRECTION OF FORCE (CODE 1,2,3)                      NSAM 149
C            AFOR= MAGNITUDE OF FORCE AT TIME = TBEG                    NSAM 150
C            BFOR= MAGNITUDE OF FORCE AT TIME = TEND                    NSAM 151
C                                                                       NSAM 152
C         - PLACE IN BLANK CARD AFTER LAST FORCE INPUTTED               NSAM 153
C                                                                       NSAM 154
C                                                                       NSAM 155
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM 156
C                    STATIC ANALYSIS INPUT SECTION                      NSAM 157
C                                                                       NSAM 158
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM 159
C                                                                       NSAM 160
```

```
C      IF THIS PROBLEM IS FOR DYNAMIC ANALYSIS, SKIP THIS SECTION          NSAM 161
C      IF NCONF=1                                                          NSAM 162
C          FOR EACH JOINT AND DIRECTION LOADED:                            NSAM 163
C            ONE CARD (2I5)                                                NSAM 164
C              IJOI,IDIR                                                   NSAM 165
C              WHERE                                                       NSAM 166
C                IJOI = JOINT LOADED                                       NSAM 167
C                IDIR = DIRECTION OF LOAD (CODE 1,2,3)                     NSAM 168
C            NEXT NF CARDS  (F20.10)                                       NSAM 169
C              F(IKIR,IJOI,J)                                              NSAM 170
C              WHERE J= NUMBER OF LOAD ( J=1 TO NF)                        NSAM 171
C                                                                          NSAM 172
C         - PLACE IN BLANK CARD AFTER LAST JOINT LOADING -                 NSAM 173
C                                                                          NSAM 174
C                                                                          NSAM 175
C      IF NCONF = 0                                                        NSAM 176
C          FOR EACH JOINT AND DIRECTION LOADED                             NSAM 177
C            ONE CARD (2I5,3F10.2)                                         NSAM 178
C                                                                          NSAM 179
C                                                                          NSAM 180
C            IJOI,IDIR,BEGLOD,ADD,DS                                       NSAM 181
C                                                                          NSAM 182
C            WHERE                                                         NSAM 183
C                                                                          NSAM 184
C              IJOI = JOINT LOADED                                         NSAM 185
C              IDIR = DIRECTION LOADED                                     NSAM 186
C              BEGLOD = LOAD FOR FIRST LOAD STEP                           NSAM 187
C              ADD = INCREMENT OF LOAD ADDED TO EACH SUBSEQUENT LOAD       NSAM 188
C                    STEP                                                  NSAM 189
C            DS = SCALING FACTOR, DEFAULT=1                                NSAM 190
C                                                                          NSAM 191
C         - PLACE BLANK CARD AFTER LAST JOINT LOADING -                    NSAM 192
```

```
C                                                                       NSAM 193
C                                                                       NSAM 194
C     CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM 195
C        ADD THREE BLANK CARDS AFTER THE LAST PROBLEM RUN FOR SUCCESSFULNSAM 196
C        TERMINATION.                                                   NSAM 197
C                                                                       NSAM 198
C                                                                       NSAM 199
C     CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCNSAM 200
C           THE PURPOSE OF THE MAIN PROGRAM IS TO INITIALIZE VARIABLES,NSAM 201
C        TO SET THE VARIOUS PARAMETERS FOR A PROBLEM, AND TO INCREMENT  NSAM 202
C        LOADS AND TIME.                                                NSAM 203
C                                                                       NSAM 204
C     LINEAR ELASTIC CONSTITUTIVE LAW, BETA=1/4.                        NSAM 205
C                                                                       NSAM 206
      IMPLICIT REAL*8 (A-H,O-Z)                                         NSAM 207
      COMMON /PLOTS/ TIME(500),NF,IOPT                                  NSAM 208
      COMMON /TRANS/ U1,U2,U3,U4,DX1,DX2,DU1,DU2                        NSAM 209
      COMMON /MEMB/ STRAIN(20),XLEN(20),B(20),A(20),H(20),DENS(20),OLDENNSAM 210
     1G,IP(20),IQ(20),NM                                                NSAM 211
      COMMON /JOINTS/ X(21),Y(21),XDJ(3,21),F(3,21,80),IJ(3,21),NDF,NJ  NSAM 212
      COMMON /MAINBK/ U(80),G(80)                                       NSAM 213
      COMMON /CONBK/ E                                                  NSAM 214
      COMMON /SAVBK/ ETOP1,ETOP2,EBOT1,EBOT2,EUQP1,EMID1,ELQP1,EUQP2,EMINSAM 215
     1D2,ELQP2                                                          NSAM 216
      COMMON /TAB/ ILOAD,NMIN,KOUNT                                     NSAM 217
      COMMON /OSIR/ EPS,EST,LIMIT,N,IER,IPR,NH                          NSAM 218
      COMMON /INERT/ ACC(3,21),VEL(3,21),FORCE(3,21),XMASS(3,21),DIS(3,2NSAM 219
     11),GACC(3,21),DISM(20),T,TIM                                      NSAM 220
      COMMON /EXTRA/ TIMEL,TWOTIM                                       NSAM 221
      DIMENSION TITLE(20), IR(3,21)                                     NSAM 222
      DATA IRES/1HR/,OPT/4HDYNA/,BC1/1HD/,BC2/1HV/,BC3/1HA/,BC4/1HF/    NSAM 223
20    CONTINUE                                                          NSAM 224
```

```
      CALL TRAPS (0,9999,9999)                                                  NSAM 225
C                                                                               NSAM 226
C             INPUT PROBLEM PARAMETERS AND CONSTANTS                            NSAM 227
C                                                                               NSAM 228
      READ 530, TITLE                                                           NSAM 229
      READ 350, OPTION                                                          NSAM 230
      READ 360, NJ,NM,NF,LIMIT,NCONF,INICON,EPS,EST,IPR                         NSAM 231
      IF (NJ.LE.0) GO TO 340                                                    NSAM 232
      IF (NF.EQ.0) NF=4                                                         NSAM 233
      IF (LIMIT.EQ.0) LIMIT=40                                                  NSAM 234
      IF (EPS.EQ.0.D0) EPS=1.D-8                                                NSAM 235
      IF (EST.EQ.0.D0) EST=1.D-05                                               NSAM 236
      NDF=0                                                                     NSAM 237
      OLDENG=0.D0                                                               NSAM 238
      IOPT=0                                                                    NSAM 239
      IF (OPTION.EQ.OPT) IOPT=1                                                 NSAM 240
      PRINT 520                                                                 NSAM 241
      PRINT 560, TITLE                                                          NSAM 242
      READ (5,370) E                                                            NSAM 243
      PRINT 380, E,EPS                                                          NSAM 244
      READ 540, (I,X(I),Y(I),IR(1,I),IR(2,I),IR(3,I),XDJ(1,I),XDJ(2,I),XNSAM 245
     1DJ(3,I),K=1,NJ)                                                           NSAM 246
      PRINT 670                                                                 NSAM 247
      PRINT 570                                                                 NSAM 248
C                                                                               NSAM 249
C     DETERMINE THE JOINT STATUS, AND INDEX FOR ASSEMBLY.                       NSAM 250
C                                                                               NSAM 251
      DO 50 J=1,NJ                                                              NSAM 252
      DO 40 I=1,3                                                               NSAM 253
      IJ(I,J)=0                                                                 NSAM 254
      IF (IR(I,J).EQ.IRES) GO TO 30                                             NSAM 255
      NDF=NDF+1                                                                 NSAM 256
```

```
      IJ(I,J)=NDF                                                       NSAM 257
30    CONTINUE                                                          NSAM 258
C                                                                       NSAM 259
C     INITIALIZE THE FORCE MATRIX.                                      NSAM 260
C                                                                       NSAM 261
      DO 40 K=1,NF                                                      NSAM 262
      F(I,J,K)=0.D0                                                     NSAM 263
40    CONTINUE                                                          NSAM 264
      PRINT 580, J,X(J),Y(J),IJ(1,J),IJ(2,J),IJ(3,J)                    NSAM 265
50    CONTINUE                                                          NSAM 266
      N=NDF+NM                                                          NSAM 267
      NH=N*(N+N+5)                                                      NSAM 268
      DO 60 I=1,N                                                       NSAM 269
      G(I)=0.D0                                                         NSAM 270
      U(I)=0.D0                                                         NSAM 271
60    CONTINUE                                                          NSAM 272
C                                                                       NSAM 273
C     INITIALIZE ACCELERATION, VELOCITY, AND DISPLACEMENT TERMS TO ZERO NSAM 274
C                                                                       NSAM 275
      DO 70 I=1,21                                                      NSAM 276
      DO 70 J=1,3                                                       NSAM 277
      GACC(J,I)=0.D0                                                    NSAM 278
      ACC(J,I)=0.D0                                                     NSAM 279
      VEL(J,I)=0.D0                                                     NSAM 280
      DIS(J,I)=0.D0                                                     NSAM 281
70    CONTINUE                                                          NSAM 282
      DO 80 I=1,NM                                                      NSAM 283
      DISM(I)=0.D0                                                      NSAM 284
80    CONTINUE                                                          NSAM 285
      PRINT 670                                                         NSAM 286
      PRINT 590                                                         NSAM 287
      DO 90 K=1,NM                                                      NSAM 288
```

```
      READ 390, M,IP(M),IQ(M),B(M),H(M),A(M),DENS(M)                    NSAM 289
      XL=(X(IQ(M))-X(IP(M)))**2+(Y(IQ(M))-Y(IP(M)))**2                  NSAM 290
      XL=DSQRT(XL)                                                      NSAM 291
      AX=B(M)*H(M)                                                      NSAM 292
      ZI=B(M)*H(M)**3/12.D0+AX*A(M)**2                                  NSAM 293
      PRINT 600, M,IP(M),IQ(M),B(M),H(M),A(M),XL,AX,ZI                  NSAM 294
      XLEN(M)=XL                                                        NSAM 295
90    CONTINUE                                                          NSAM 296
      CALL MASS                                                         NSAM 297
      PRINT 400                                                         NSAM 298
      DO 100 M=1,NJ                                                     NSAM 299
      PRINT 410, M,(XMASS(I,M),I=1,3)                                   NSAM 300
100   CONTINUE                                                          NSAM 301
C                                                                       NSAM 302
C     INPUT PRESCRIBED CONFIGURATION IF THERE IS ONE                    NSAM 303
C                                                                       NSAM 304
      IF (INICON.EQ.0) GO TO 160                                        NSAM 305
      PRINT 420                                                         NSAM 306
      PRINT 610                                                         NSAM 307
110   READ 430, J,Q1,Q2,Q3                                              NSAM 308
      IF (J.EQ.0) GO TO 140                                             NSAM 309
      PRINT 620, J,Q1,Q2,Q3                                             NSAM 310
      IF (IJ(1,J).EQ.0) GO TO 120                                       NSAM 311
      U(IJ(1,J))=Q1                                                     NSAM 312
120   IF (IJ(2,J).EQ.0) GO TO 130                                       NSAM 313
      U(IJ(2,J))=Q2                                                     NSAM 314
130   IF (IJ(3,J).EQ.0) GO TO 110                                       NSAM 315
      U(IJ(3,J))=Q3                                                     NSAM 316
      GO TO 110                                                         NSAM 317
140   PRINT 630                                                         NSAM 318
150   READ 440, I,Q                                                     NSAM 319
      IF (I.EQ.0) GO TO 160                                             NSAM 320
```

```
      PRINT 640, I,Q                                                    NSAM 321
      U(NDF+I)=Q                                                        NSAM 322
      GO TO 150                                                         NSAM 323
160   CONTINUE                                                          NSAM 324
      IF (IOPT.EQ.1) GO TO 230                                          NSAM 325
C                                                                       NSAM 326
C            INPUT LOADING DATA                                         NSAM 327
C                                                                       NSAM 328
      IF (NCONF.EQ.0) GO TO 190                                         NSAM 329
170   READ 450, IJOI,IDIR                                              NSAM 330
      IF (IJOI.EQ.0) GO TO 210                                          NSAM 331
      DO 180 ILOAD=1,NF                                                 NSAM 332
180   READ 460, F(IDIR,IJOI,ILOAD)                                      NSAM 333
      GO TO 170                                                         NSAM 334
190   CONTINUE                                                          NSAM 335
      READ 550, IJOI,IDIR,BEGLOD,ADD,DS                                 NSAM 336
      IF (IJOI.EQ.0) GO TO 210                                          NSAM 337
      IF (DS.EQ.0.D0) DS=1.D0                                           NSAM 338
      ADD=ADD*DS                                                        NSAM 339
      F(IDIR,IJOI,1)=BEGLOD*DS                                          NSAM 340
      DO 200 ILOAD=2,NF                                                 NSAM 341
      ILOADL=ILOAD-1                                                    NSAM 342
200   F(IDIR,IJOI,ILOAD)=F(IDIR,IJOI,ILOADL)+ADD                        NSAM 343
      GO TO 190                                                         NSAM 344
210   CONTINUE                                                          NSAM 345
      DO 220 IL=1,NF                                                    NSAM 346
      PRINT 670                                                         NSAM 347
      PRINT 650, IL                                                     NSAM 348
      ILOAD=IL                                                          NSAM 349
      CALL OSIRIS (IL,IERROR)                                           NSAM 350
      IF (IER.NE.0) WRITE (6,680)                                       NSAM 351
      IF (IER.NE.0) GO TO 330                                           NSAM 352
```

```
220   CONTINUE                                                          NSAM 353
      GO TO 330                                                         NSAM 354
C                                                                       NSAM 355
C             INPUT PARAMETERS FOR DYNAMIC ANALYSIS                     NSAM 356
C                                                                       NSAM 357
230   READ 470, TBEG,TEND,T,TMIN,NTMAX                                  NSAM 358
      TWOTIM=T+T+TBEG                                                   NSAM 359
      NSTEP=0                                                           NSAM 360
      TIM=TEND-TBEG                                                     NSAM 361
      TIME(1)=TBEG                                                      NSAM 362
C                                                                       NSAM 363
C     INPUT INITIAL CONDITIONS                                          NSAM 364
C                                                                       NSAM 365
240   READ (5,480) I,ID,TYPE,XINCON                                     NSAM 366
      IF (I.EQ.0) GO TO 250                                             NSAM 367
      IF (TYPE.EQ.BC1) DIS(ID,I)=DIS(ID,I)+XINCON                       NSAM 368
      IF (TYPE.EQ.BC2) VEL(ID,I)=VEL(ID,I)+XINCON                       NSAM 369
      IF (TYPE.EQ.BC3) ACC(ID,I)=ACC(ID,I)+XINCON                       NSAM 370
      IF (TYPE.EQ.BC4) FORCE(ID,I)=FORCE(ID,I)+XINCON                   NSAM 371
      GO TO 240                                                         NSAM 372
C                                                                       NSAM 373
C     PERFORM TIME ZERO STATIC ANALYSIS                                 NSAM 374
C                                                                       NSAM 375
250   WRITE (6,490)                                                     NSAM 376
      ILOAD=3                                                           NSAM 377
      IOPT=0                                                            NSAM 378
      CALL OSIRIS (1,IERROR)                                            NSAM 379
      IOPT=1                                                            NSAM 380
C                                                                       NSAM 381
C     ADD DISPLACEMEMTS FROM TIME ZERO STATIC ANALYSIS TO INITIAL DISP. NSAM 382
C                                                                       NSAM 383
      DO 260 I=1,3                                                      NSAM 384
```

```
      DO 260 J=1,NJ                                                    NSAM 385
      IF (IJ(I,J).EQ.0) GO TO 260                                      NSAM 386
      DIS(I,J)=DIS(I,J)+U(IJ(I,J))                                     NSAM 387
260   CONTINUE                                                         NSAM 388
270   READ 500, IJOI,IDIR,AFOR,BFOR                                    NSAM 389
      IF (IJOI.EQ.0) GO TO 280                                         NSAM 390
      F(IDIR,IJOI,1)=AFOR                                              NSAM 391
      F(IDIR,IJOI,2)=BFOR                                              NSAM 392
      GO TO 270                                                        NSAM 393
280   DO 290 MM=1,NJ                                                   NSAM 394
      DO 290 NN=1,3                                                    NSAM 395
      IF (XMASS(NN,MM).EQ.0.D0) GO TO 290                              NSAM 396
      IF (IJ(NN,MM).EQ.0) GO TO 290                                    NSAM 397
      ACC(NN,MM)=ACC(NN,MM)+F(NN,MM,1)/XMASS(NN,MM)                    NSAM 398
290   CONTINUE                                                         NSAM 399
      I=1                                                              NSAM 400
300   I=I+1                                                            NSAM 401
      NSTEP=NSTEP+1                                                    NSAM 402
      IF (NSTEP.GT.NTMAX) GO TO 330                                    NSAM 403
310   DELT=T                                                           NSAM 404
      TIME(I)=TIME(I-1)+DELT                                           NSAM 405
      IF (TIME(I).GT.TEND) GO TO 330                                   NSAM 406
      DO 320 NN=1,NJ                                                   NSAM 407
      DO 320 MM=1,3                                                    NSAM 408
      F(MM,NN,3)=F(MM,NN,1)+(TIME(I)/TIM)*(F(MM,NN,2)-F(MM,NN,1))      NSAM 409
320   CONTINUE                                                         NSAM 410
      PRINT 670                                                        NSAM 411
      PRINT 510, TIME(I)                                               NSAM 412
      IL=I                                                             NSAM 413
      TIMEL=TIME(I)                                                    NSAM 414
      CALL OSIRIS (IL,IERROR)                                          NSAM 415
      IF (IER.NE.0) WRITE (6,680)                                      NSAM 416
```

```
      IF (IER.NE.0) GO TO 330                                          NSAM 417
      IF (IERROR.EQ.2) T=T/.5D0                                        NSAM 418
      IF (IERROR.EQ.3) T=T*.50D0                                       NSAM 419
      IF (IERROR.EQ.3) GO TO 310                                       NSAM 420
      GO TO 300                                                        NSAM 421
330   CONTINUE                                                         NSAM 422
      GO TO 20                                                         NSAM 423
340   CONTINUE                                                         NSAM 424
      PRINT 520                                                        NSAM 425
      PRINT 660                                                        NSAM 426
C                                                                      NSAM 427
      RETURN                                                           NSAM 428
C                                                                      NSAM 429
350   FORMAT (A4)                                                      NSAM 430
360   FORMAT (6I5,F20.16,F10.5,I5)                                     NSAM 431
370   FORMAT (F10.0)                                                   NSAM 432
380   FORMAT (////1X,22HMODULUS OF ELASTICITY=,F10.0,//1X,21HERROR TOLERNSAM 433
     1ANCE, EPS=,D10.3)                                               NSAM 434
390   FORMAT (3I5,5X,4F10.5)                                           NSAM 435
400   FORMAT (1H0,T24,18HLUMPED MASS MATRIX//T7,5HJOINT,12X,2HDX,19X,2HDNSAM 436
     1Y,19X,2HDZ//)                                                   NSAM 437
410   FORMAT (1H ,T9,I2,6X,3(D16.7,5X)/)                               NSAM 438
420   FORMAT (1X,75H*** THE FOLLOWING INITIAL EQUILIBRIUM CONFIGURATION NSAM 439
     1HAS BEEN PRESCRIBED ***//)                                      NSAM 440
430   FORMAT (I5,5X,3F20.16)                                           NSAM 441
440   FORMAT (I5,5X,F20.16)                                            NSAM 442
450   FORMAT (2I5)                                                     NSAM 443
460   FORMAT (F20.10)                                                  NSAM 444
470   FORMAT (4F10.5,I5)                                               NSAM 445
480   FORMAT (I5,I5,4X,A1,5X,D20.16)                                   NSAM 446
490   FORMAT (///1X,37HRESULTS FOR TIME ZERO STATIC ANALYSIS)          NSAM 447
500   FORMAT (2I5,2F10.2)                                              NSAM 448
```

```
510   FORMAT (///1X,5HTIME=,F15.10,8H SECONDS)                                  NSAM 449
520   FORMAT (1H1)                                                              NSAM 450
530   FORMAT (20A4)                                                             NSAM 451
540   FORMAT (I5,5X,2F10.5,2X,3A1,5X,3F10.5)                                    NSAM 452
550   FORMAT (2I5,3F10.2)                                                       NSAM 453
560   FORMAT (1HO,T10,20A4//)                                                   NSAM 454
570   FORMAT (1HO,T20,28HJOINT COORDINATES AND STATUS//T7,5HJOINT,9X,1HXNSAM 455
     1,12X,1HY,9X,2HDX,3X,2HDY,3X,2HDZ//)                                       NSAM 456
580   FORMAT (1H ,T9,I2,6X,2(F8.3,5X),3(I3,2X)/)                                NSAM 457
590   FORMAT (1HO,T29,18HELEMENT PROPERTIES//T6,7HELEMENT,4X,5HSTART,4X,NSAM 458
     13HEND,3X,5HWIDTH,6X,6HHEIGHT,9X,1HA,7X,6HLENGTH,7X,4HAREA,6X,7HINENSAM 459
     2RTIA//)                                                                  NSAM 460
600   FORMAT (1H ,T9,I2,7X,I3,4X,I3,6(4X,F8.4)/)                                NSAM 461
610   FORMAT (1HO,T24,19HJOINT DISPLACEMENTS//T7,5HJOINT,12X,2HDX,19X,2HNSAM 462
     1DY,19X,2HDZ//)                                                           NSAM 463
620   FORMAT (1H ,T9,I2,6X,3(D16.7,5X)/)                                        NSAM 464
630   FORMAT (1HO,T17,26HINTERNAL NODE DISPLACEMENT//T18,7HELEMENT,12X,2NSAM 465
     1HU4//)                                                                   NSAM 466
640   FORMAT (1H ,T21,I2,7X,D16.8/)                                             NSAM 467
650   FORMAT (1H ,T10,10HLOAD STEP ,I2/)                                        NSAM 468
660   FORMAT (1HO,T10,46H:::::  NORMAL COMPLETION OF THE PROGRAM  :::::/NSAM 469
     1/)                                                                       NSAM 470
670   FORMAT (1HO,60(1H-)//)                                                    NSAM 471
680   FORMAT (1HO,1X,41H****  IER NE 0 - PROBLEM TERMINATED  ****)             NSAM 472
      END                                                                       NSAM 473
```

```
      SUBROUTINE FUNCT (N,U,G)                                          FUNC    1
      IMPLICIT REAL*8 (A-H,O-Z)                                         FUNC    2
      COMMON /JOINTS/ X(21),Y(21),XDJ(3,21),F(3,21,80),IJ(3,21),NDF,NJ  FUNC    3
      COMMON /PLOTS/ TIME(500),NF,IOPT                                  FUNC    4
      COMMON /MEMB/ STRAIN(20),XLEN(20),B(20),A(20),H(20),DENS(20),OLDENFUNC    5
     1G,IP(20),IQ(20),NM                                                FUNC    6
      COMMON /TAB/ ILOAD,NMIN,KOUNT                                     FUNC    7
      COMMON /TRANS/ U1,U2,U3,U4,DX1,DX2,DU1,DU2                        FUNC    8
      COMMON /INERT/ ACC(3,21),VEL(3,21),FORCE(3,21),XMASS(3,21),DIS(3,2FUNC    9
     11),GACC(3,21),DISM(20),T,TIM                                      FUNC   10
      DIMENSION U(N), G(N)                                              FUNC   11
C                                                                       FUNC   12
C     THE PURPOSE OF THIS SUBROUTINE IS TO CALCULATE THE VALUE OF THE   FUNC   13
C     EQUILIBRIUM EQUATION (G(I) OR R(I)), GIVEN A SET OF GENERALIZED   FUNC   14
C     COORDINATES.  THIS VALUE IS ALSO CALLED THE RESIDUAL.  IDEALLY    FUNC   15
C     THIS VALUE SHOULD BE ZERO.                                        FUNC   16
C                                                                       FUNC   17
      NMIN=NMIN+1                                                       FUNC   18
      DO 20 J=1,NJ                                                      FUNC   19
      DO 20 I=1,3                                                       FUNC   20
      IF (IJ(I,J).EQ.0) GO TO 20                                        FUNC   21
C                                                                       FUNC   22
C     PUT VALUES INTO THE JOINT DISPLACEMENT MATRIX FROM THE            FUNC   23
C     GENERALIZED COORDINATES MATRIX.                                   FUNC   24
C                                                                       FUNC   25
      XDJ(I,J)=U(IJ(I,J))                                               FUNC   26
C                                                                       FUNC   27
C     CALCULATE THE PART OF EACH GRADIENT WHICH IS DERIVED FROM         FUNC   28
C     THE EXTERNAL WORK.                                                FUNC   29
C                                                                       FUNC   30
      G(IJ(I,J))=-F(I,J,ILOAD)-FORCE(I,J)                               FUNC   31
20    CONTINUE                                                          FUNC   32
```

```
      DO 30 M=1,NM                                                      FUNC  33
      U4=U(NDF+M)                                                       FUNC  34
      CALL DEFO (M)                                                     FUNC  35
      CALL STNG (M)                                                     FUNC  36
C     THE DERIVATIVES OF THE SHAPE FUNCTIONS :                          FUNC  37
      CALL GRAD (G,N,M)                                                 FUNC  38
30    CONTINUE                                                          FUNC  39
      IF (IOPT.EQ.0) GO TO 50                                           FUNC  40
      DO 40 J=1,NJ                                                      FUNC  41
      DO 40 I=1,3                                                       FUNC  42
      IF (IJ(I,J).EQ.0) GO TO 40                                        FUNC  43
      ALBAR=DIS(I,J)+VEL(I,J)*T+ACC(I,J)*T*T/4.D0                       FUNC  44
      GACC(I,J)=4.D0*(XDJ(I,J)-ALBAR)/(T*T)                             FUNC  45
      G(IJ(I,J))=G(IJ(I,J))+XMASS(I,J)*GACC(I,J)                        FUNC  46
40    CONTINUE                                                          FUNC  47
50    CONTINUE                                                          FUNC  48
      RETURN                                                            FUNC  49
      END                                                               FUNC  50
```

```
      SUBROUTINE SRAN (STRN)                                            SRAN   2
C                                                                       SRAN   3
C           THE PURPOSE OF SRAN IS TO COMPUTE THE STRAIN AT A POINT,    SRAN   4
C        (EXI,ETA), GIVEN  THE ELELENT DEFORMATION VECTOR.              SRAN   5
C                                                                       SRAN   6
      IMPLICIT REAL*8 (A-H,O-Z)                                         SRAN   7
      REAL*8 L                                                          SRAN   8
      COMMON /SHAPE/ UP,VP,VPP,EXI,ETA,L                                SRAN   9
      COMMON /TRANS/ U1,U2,U3,U4,DX1,DX2,DU1,DU2                        SRAN  10
      ALFA=(U2**2-U2*U3/2.D0+U3**2)/15.D0                               SRAN  11
      BETA=(U3**2-U2**2)/16.D0                                          SRAN  12
      UP=2.D0*U1/L+ALFA                                                 SRAN  13
      VP=(BETA-4.D0*U4/L)*EXI                                           SRAN  14
      VPP=((3.D0*EXI-1.D0)*U2+(3.D0*EXI+1.D0)*U3)*ETA                   SRAN  15
      STRN=UP+VP-VPP                                                    SRAN  16
      RETURN                                                            SRAN  17
      END                                                               SRAN  18
```

```
      SUBROUTINE DEFO (M)                                                DEFO   2
C                                                                        DEFO   3
C            THE PURPOSE OF DEFO IS TO COMPUTE THE LOCAL DEFORMATION     DEFO   4
C         VECTOR OF AN ELEMENT                                           DEFO   5
C                                                                        DEFO   6
      IMPLICIT REAL*8 (A-H,O-Z)                                          DEFO   7
      COMMON /JOINTS/ X(21),Y(21),XDJ(3,21),F(3,21,80),IJ(3,21),NDF,NJ   DEFO   8
      COMMON /MEMB/ STRAIN(20),XLEN(20),B(20),A(20),H(20),DENS(20),OLDENDEFO   9
     1G,IP(20),IQ(20),NM                                                 DEFO  10
      COMMON /TRANS/ U1,U2,U3,U4,DX1,DX2,DU1,DU2                         DEFO  11
      I=IP(M)                                                            DEFO  12
      J=IQ(M)                                                            DEFO  13
      DX1=(X(J)-X(I))                                                    DEFO  14
      DX2=(Y(J)-Y(I))                                                    DEFO  15
      DU1=(XDJ(1,J)-XDJ(1,I))                                            DEFO  16
      DU2=(XDJ(2,J)-XDJ(2,I))                                            DEFO  17
      DXX2=DX2+DU2                                                       DEFO  18
      DXX1=DX1+DU1                                                       DEFO  19
      XL1=DSQRT(DXX1**2+DXX2**2)                                         DEFO  20
      DELTA=XL1-XLEN(M)                                                  DEFO  21
      AA=DX1*DXX2-DXX1*DX2                                               DEFO  22
      BB=DXX1*DX1+DXX2*DX2                                               DEFO  23
      GAMMA=DATAN2(AA,BB)                                                DEFO  24
      U1=DELTA/2.DO                                                      DEFO  25
      U2=XDJ(3,I)-GAMMA                                                  DEFO  26
      U3=XDJ(3,J)-GAMMA                                                  DEFO  27
      RETURN                                                             DEFO  28
      END                                                                DEFO  29
```

```
      SUBROUTINE STNG (M)                                                STNG   2
C                                                                        STNG   3
C           STNG COMPUTES THE INTERNAL ENERGY OF AN ELEMENT AND THE      STNG   4
C        DERIVATIVE OF THE STRAIN ENERGY WITH RESPECT TO THE LOCAL       STNG   5
C        DEFORMATION VECTOR                                              STNG   6
C                                                                        STNG   7
      IMPLICIT REAL*8 (A-H,O-Z)                                          STNG   8
      REAL*8 L                                                           STNG   9
      COMMON /SHAPE/ UP,VP,VPP,EXI,ETA,L                                 STNG  10
      COMMON /DERIV/ DU(4)                                               STNG  11
      COMMON /SAVBK/ ETOP1,ETOP2,EBOT1,EBOT2,EUQP1,EMID1,ELQP1,EUQP2,EMISTNG  12
     1D2,ELQP2                                                           STNG  13
      COMMON /MEMB/ STRAIN(20),XLEN(20),B(20),A(20),H(20),DENS(20),OLDENSTNG  14
     1G,IP(20),IQ(20),NM                                                 STNG  15
      COMMON /CONBK/ E                                                   STNG  16
      COMMON /TRANS/ U1,U2,U3,U4,DX1,DX2,DU1,DU2                         STNG  17
      L=XLEN(IABS(M))                                                    STNG  18
      IF (M.LT.0) GO TO 20                                               STNG  19
      STRAIN(M)=0.D0                                                     STNG  20
      ALPHA=(U2**2-U2*U3/2.D0+U3**2)/15.D0                               STNG  21
      BETA=(U3**2-U2**2)/16.D0                                           STNG  22
      DALF2=(2.D0*U2-U3/2.D0)/15.D0                                      STNG  23
      DALF3=(2.D0*U3-U2/2.D0)/15.D0                                      STNG  24
      DBET2=-U2/8.D0                                                     STNG  25
      DBET3=U3/8.D0                                                      STNG  26
      EAL=E*B(M)*H(M)*L                                                  STNG  27
      EA=E*B(M)*H(M)                                                     STNG  28
      STRAIN(M)=.5D0*EAL*(2.D0*U1/L+ALPHA)**2+EAL*((BETA-4.D0*U4/L)**2)/STNG  29
     16.D0+2.D0*EA*(H(M)*H(M)/4.D0+3.D0*A(M)*A(M))*(U2**2+U3**2+U2*U3)/(STNG  30
     23.D0*L)+EA*A(M)*(2.D0*U1/L+ALPHA)*(U3-U2)+EA*A(M)*(BETA-4.D0*U4/L)STNG  31
     3*(U2+U3)                                                           STNG  32
      DU(1)=2.D0*EA*(2.D0*U1/L+ALPHA)+2.D0*EA*A(M)*(U3-U2)/L             STNG  33
```

```
      DU(2)=2.D0*EA*(H(M)*H(M)/4.D0+3.D0*A(M)*A(M))*(U2+U2+U3)/(3.D0*L)-STNG  34
     1EA*A(M)*(2.D0*U1/L+ALPHA-(U3-U2)*DALF2)+EA*A(M)*(BETA-4.D0*U4/L+(USTNG  35
     22+U3)*DBET2)+EAL*(2.D0*U1/L+ALPHA)*DALF2+EAL*(BETA-4.D0*U4/L)*DBETSTNG  36
     32/3.D0                                                             STNG  37
      DU(3)=EAL*(2.D0*U1/L+ALPHA)*DALF3+EAL*(BETA-4.D0*U4/L)*DBET3/3.D0+STNG  38
     12.D0*EA*(H(M)*H(M)/4.D0+3.D0*A(M)*A(M))*(2.D0*U3+U2)/(3.D0*L)+EA*ASTNG  39
     2(M)*(2.D0*U1/L+ALPHA+(U3-U2)*DALF3)+EA*A(M)*(BETA-4.D0*U4/L+(U2+U3STNG  40
     3)*DBET3)                                                           STNG  41
      DU(4)=-4.D0*EA*(BETA-4.D0*U4/L)/3.D0-4.D0*EA*A(M)*(U2+U3)/L        STNG  42
      GO TO 30                                                          STNG  43
20    CONTINUE                                                          STNG  44
      M=IABS(M)                                                         STNG  45
      EXI=-.5D0                                                         STNG  46
      ETA=(-A(M)+H(M)/2.D0)/L                                           STNG  47
      CALL SRAN (ETOP1)                                                 STNG  48
      ETA=(-A(M)+H(M)/4.D0)/L                                           STNG  49
      CALL SRAN (EUQP1)                                                 STNG  50
      ETA=-A(M)/L                                                       STNG  51
      CALL SRAN (EMID1)                                                 STNG  52
      ETA=-(A(M)+H(M)/4.D0)/L                                           STNG  53
      CALL SRAN (ELQP1)                                                 STNG  54
      ETA=-(A(M)+H(M)/2.D0)/L                                           STNG  55
      CALL SRAN (EBOT1)                                                 STNG  56
      EXI=.5D0                                                          STNG  57
      CALL SRAN (EBOT2)                                                 STNG  58
      ETA=-(A(M)+H(M)/4.D0)/L                                           STNG  59
      CALL SRAN (ELQP2)                                                 STNG  60
      ETA=-A(M)/L                                                       STNG  61
      CALL SRAN (EMID2)                                                 STNG  62
      ETA=(-A(M)+H(M)/4.D0)/L                                           STNG  63
      CALL SRAN (EUQP2)                                                 STNG  64
      ETA=(-A(M)+H(M)/2.D0)/L                                           STNG  65
```

```
      CALL SRAN (ETOP2)                                          STNG  66
30    CONTINUE                                                   STNG  67
      RETURN                                                     STNG  68
      END                                                        STNG  69
```

```
      SUBROUTINE GRAD (G,N,M)                                           GRAD   2
C           GRAD TRANSFORMS THE VARIATION OF THE INTERNAL ENERGY WRT     GRAD   4
C        THE LOCAL DEFORMATION VECTOR INTO THE   VARIATION OF THE INTERNGRAD   5
C        ENERGY WRT THE GLOBAL DEFORMATION VECTOR.                       GRAD   6
      IMPLICIT REAL*8 (A-H,O-Z)                                         GRAD   8
      COMMON /DERIV/ DU(4)                                              GRAD   9
      COMMON /JOINTS/ X(21),Y(21),XDJ(3,21),F(3,21,80),IJ(3,21),NDF,NJ  GRAD  10
      COMMON /MEMB/ STRAIN(20),XLEN(20),B(20),A(20),H(20),DENS(20),OLDENGRAD  11
     1G,IP(20),IQ(20),NM                                                GRAD  12
      COMMON /TRANS/ U1,U2,U3,U4,DX1,DX2,DU1,DU2                        GRAD  13
      DIMENSION G(N)                                                    GRAD  14
      I=IP(M)                                                           GRAD  15
      J=IQ(M)                                                           GRAD  16
      DXX2=DX2+DU2                                                      GRAD  17
      DXX1=DX1+DU1                                                      GRAD  18
      XL1=DSQRT(DXX1**2+DXX2**2)                                        GRAD  19
      AA=DX1*(DX2+DU2)-DX2*(DX1+DU1)                                    GRAD  20
      BB=DX1*(DX1+DU1)+DX2*(DX2+DU2)                                    GRAD  21
      AB=-1.D0/((1.D0+(AA/BB)**2)*BB**2)                                GRAD  22
      IF (IJ(1,I).NE.0) G(IJ(1,I))=G(IJ(1,I))-(DX1+DU1)*DU(1)/(2.D0*XL1)GRAD  23
     1+AB*(BB*DX2+AA*DX1)*(DU(2)+DU(3))                                 GRAD  24
      IF (IJ(2,I).NE.0) G(IJ(2,I))=G(IJ(2,I))-(DX2+DU2)*DU(1)/(2.D0*XL1)GRAD  25
     1+AB*(AA*DX2-BB*DX1)*(DU(2)+DU(3))                                 GRAD  26
      IF (IJ(3,I).NE.0) G(IJ(3,I))=G(IJ(3,I))+DU(2)                     GRAD  27
      IF (IJ(1,J).NE.0) G(IJ(1,J))=G(IJ(1,J))+(DX1+DU1)*DU(1)/(2.D0*XL1)GRAD  28
     1-AB*(BB*DX2+AA*DX1)*(DU(2)+DU(3))                                 GRAD  29
      IF (IJ(2,J).NE.0) G(IJ(2,J))=G(IJ(2,J))+(DX2+DU2)*DU(1)/(2.D0*XL1)GRAD  30
     1+AB*(BB*DX1-AA*DX2)*(DU(2)+DU(3))                                 GRAD  31
      IF (IJ(3,J).NE.0) G(IJ(3,J))=G(IJ(3,J))+DU(3)                     GRAD  32
      G(NDF+M)=DU(4)                                                    GRAD  33
      RETURN                                                            GRAD  34
      END                                                               GRAD  35
```

```
      SUBROUTINE MASS                                                   MASS    2
      IMPLICIT REAL*8 (A-H,O-Z)                                         MASS    3
      COMMON /INERT/ ACC(3,21),VEL(3,21),FORCE(3,21),XMASS(3,21),DIS(3,2MASS    4
     11),GACC(3,21),DISM(20),T,TIM                                      MASS    5
      COMMON /JOINTS/ X(21),Y(21),XDJ(3,21),F(3,21,80),IJ(3,21),NDF,NJ  MASS    6
      COMMON /MEMB/ STRAIN(20),XLEN(20),B(20),A(20),H(20),DENS(20),OLDENMASS    7
     1G,IP(20),IQ(20),NM                                                MASS    8
      DO 20 I=1,NJ                                                      MASS    9
      DO 20 J=1,3                                                       MASS   10
      XMASS(J,I)=0.D0                                                   MASS   11
      FORCE(J,I)=0.D0                                                   MASS   12
20    CONTINUE                                                          MASS   13
C                                                                       MASS   14
C     COMPUTE MASS AND WEIGHT CONTRIBUTING TO EACH NODE OF ELEMENT      MASS   15
C                                                                       MASS   16
      DO 30 I=1,NM                                                      MASS   17
      XM=XLEN(I)*B(I)*H(I)*DENS(I)/(2.D0*386.088D0)                     MASS   18
      XI=XLEN(I)**2*B(I)*H(I)*DENS(I)/(24.D0*386.088D0)                 MASS   19
      IPM=IP(I)                                                         MASS   20
      IQM=IQ(I)                                                         MASS   21
      XMASS(1,IPM)=XMASS(1,IPM)+XM                                      MASS   22
      XMASS(2,IPM)=XMASS(2,IPM)+XM                                      MASS   23
      XMASS(3,IPM)=XMASS(3,IPM)+XI                                      MASS   24
      XMASS(1,IQM)=XMASS(1,IQM)+XM                                      MASS   25
      XMASS(2,IQM)=XMASS(2,IQM)+XM                                      MASS   26
      XMASS(3,IQM)=XMASS(3,IQM)+XI                                      MASS   27
      FORCE(2,IPM)=FORCE(2,IPM)-XM*386.088D0                            MASS   28
      FORCE(2,IQM)=FORCE(2,IQM)-XM*386.088D0                            MASS   29
30    CONTINUE                                                          MASS   30
      XMASS(2,NJ)=XMASS(2,NJ)*2.D0                                      MASS   31
      RETURN                                                            MASS   32
      END                                                               MASS   33
```

```
      SUBROUTINE IMBAL (G,U,FUN,EB,N,IEPROR,IOPT)                       IMBL   1
C                                                                       IMBL   2
C             IMBAL COMPUTES THE ERRORS AT THE END OF A TIME(LOAD) STEP IMBL   3
C                                                                       IMBL   4
      IMPLICIT REAL*8 (A-H,O-Z)                                         IMBL   5
      COMMON /INERT/ ACC(3,21),VEL(3,21),FORCE(3,21),XMASS(3,21),DIS(3,2IMBL   6
     11),GACC(3,21),DISM(20),T,TIM                                      IMBL   7
      COMMON /MEMB/ STRAIN(20),XLEN(20),B(20),A(20),H(20),DENS(20),GLDENIMBL   8
     1G,IP(20),IQ(20),NM                                                IMBL   9
      COMMON /JOINTS/ X(21),Y(21),XDJ(3,21),F(3,21,80),IJ(3,21),NDF,NJ  IMBL  10
      COMMON /EXTRA/ TIMEL,TWOTIM                                       IMBL  11
      DIMENSION G(N), U(N)                                              IMBL  12
      EAB=0.D0                                                          IMBL  13
      EB=0.D0                                                           IMBL  14
      IF (FUN.EQ.0.D0) GO TO 30                                         IMBL  15
      DO 20 I=1,N                                                       IMBL  16
      EB=EB+G(I)*G(I)                                                   IMBL  17
20    CONTINUE                                                          IMBL  18
      EB=DSQRT(EB)                                                      IMBL  19
30    IF (IOPT.EQ.0) GO TO 120                                          IMBL  20
      BEGKIN=0.D0                                                       IMBL  21
      ENGNEW=0.D0                                                       IMBL  22
      DELWE=0.D0                                                        IMBL  23
      DELKIN=0.D0                                                       IMBL  24
      DO 40 I=1,3                                                       IMBL  25
      DO 40 J=1,NJ                                                      IMBL  26
      IF (IJ(I,J).EQ.0) GO TO 40                                        IMBL  27
      DELWE=DELWE+(XDJ(I,J)-DIS(I,J))*(F(I,J,3)+F(I,J,4)+FORCE(I,J))/2.DIMBL  28
     10                                                                 IMBL  29
      ALBAR=DIS(I,J)+VEL(I,J)*T+ACC(I,J)*T*T/4.D0                       IMBL  30
      AC=4.D0*(XDJ(I,J)-ALBAR)/(T*T)                                    IMBL  31
      V=VEL(I,J)+T*(ACC(I,J)+AC)/2.D0                                   IMBL  32
```

```
      BEGKIN=BEGKIN+0.5D0*XMASS(I,J)*VEL(I,J)**2                        IMBL  33
      DELKIN=DELKIN+XMASS(I,J)*(V*V-VEL(I,J)**2)                        IMBL  34
40    CONTINUE                                                          IMBL  35
      DO 50 M=1,NM                                                      IMBL  36
      ENGNEW=ENGNEW+STRAIN(M)                                           IMBL  37
50    CONTINUE                                                          IMBL  38
      DELWI=ENGNEW-OLDENG                                               IMBL  39
      IF (OLDENG+BEGKIN) 90,90,60                                       IMBL  40
60    CONTINUE                                                          IMBL  41
      EAB=(DELWE-DELWI-DELKIN)/(OLDENG+BEGKIN)                          IMBL  42
      EAB=DABS(EAB)                                                     IMBL  43
      IF (EAB.GT.EB) EB=EAB                                             IMBL  44
      IERROR=3                                                          IMBL  45
      IF (TWOTIM.GE.TIMEL) GO TO 90                                     IMBL  46
      IF (EB.LT.1.0D-04) IERROR=1                                       IMBL  47
      IF (EB.LT.1.0D-15) IERROR=2                                       IMBL  48
      IF (IERROR.NE.3) GO TO 100                                        IMBL  49
      DO 70 I=1,3                                                       IMBL  50
      DO 70 J=1,NJ                                                      IMBL  51
      IF (IJ(I,J).EQ.0) GO TO 70                                        IMBL  52
      U(IJ(I,J))=DIS(I,J)                                               IMBL  53
70    CONTINUE                                                          IMBL  54
      DO 80 M=1,NM                                                      IMBL  55
      U(NDF+M)=DISM(M)                                                  IMBL  56
80    CONTINUE                                                          IMBL  57
      GO TO 100                                                         IMBL  58
90    IERROR=1                                                          IMBL  59
100   CONTINUE                                                          IMBL  60
      IF (IERROR.EQ.1) OLDENG=ENGNEW                                    IMBL  61
      IF (IERROR.NE.1) GO TO 120                                        IMBL  62
      DO 110 I=1,NJ                                                     IMBL  63
      DO 110 J=1,3                                                      IMBL  64
```

```
      F(J,I,4)=F(J,I,3)                                              IMBL   65
110   CONTINUE                                                       IMBL   66
120   IF (IOPT.EQ.0) IERROR=1                                        IMBL   67
      RETURN                                                         IMBL   68
      END                                                            IMBL   69
```

```
      SUBROUTINE OSIRIS (IL,IERROR)                                      OSRS   2
      IMPLICIT REAL*8 (A-H,O-Z)                                          OSRS   3
      COMMON /JOINTS/ X(21),Y(21),XDJ(3,21),F(3,21,80),IJ(3,21),NDF,NJ   OSRS   4
      COMMON /CONBK/ E                                                   OSRS   5
      COMMON /OSIR/ EPS,EST,LIMIT,N,IER,IPR,NH                           OSRS   6
      COMMON /MAINBK/ U(80),G(80)                                        OSRS   7
      COMMON /MEMB/ STRAIN(20),XLEN(20),B(20),A(20),H(20),DENS(20),OLDENOSRS   8
     1G,IP(20),IQ(20),NM                                                 OSRS   9
      COMMON /PLOTS/ TIME(500),NF,ICPT                                   OSRS  10
      COMMON /TRANS/ U1,U2,U3,U4,DX1,DX2,DU1,DU2                         OSRS  11
      COMMON /SAVBK/ ETOP1,ETOP2,EBOT1,EBOT2,EUQP1,EMID1,ELQP1,EUQP2,EMIOSRS  12
     1D2,ELQP2                                                           OSRS  13
      COMMON /TAB/ ILOAD,NMIN,KOUNT                                      OSRS  14
      COMMON /INERT/ ACC(3,21),VEL(3,21),FORCE(3,21),XMASS(3,21),DIS(3,2OSRS  15
     11),GACC(3,21),DISM(20),T,TIM                                       OSRS  16
      DIMENSION AINV(80,80)                                              OSRS  17
      DIMENSION HN(13200)                                                OSRS  18
      NMIN=0                                                             OSRS  19
C                                                                        OSRS  20
C     PRINT OUT THE JOINT LOAD MATRIX.                                   OSRS  21
C                                                                        OSRS  22
      PRINT 260                                                          OSRS  23
      DO 20 J=1,NJ                                                       OSRS  24
      PRINT 250, J,(F(I,J,ILOAD),I=1,3)                                  OSRS  25
20    CONTINUE                                                           OSRS  26
C                                                                        OSRS  27
C     CALL THE MINIMIZATION ROUTINE FOR EACH LOAD STEP.                  OSRS  28
C                                                                        OSRS  29
      CALL NSOIAD (N,U,G,AINV,EST,1.D0,EPS,LIMIT,IPR,HN,FUN,IER,NH)      OSRS  30
      CALL IMBAL (G,U,FUN,EB,N,IERROR,ICPT)                              OSRS  31
      WRITE (6,160) EB                                                   OSRS  32
      KOUNT=NMIN                                                         OSRS  33
```

```
C                                                                       OSRS  34
C     PRINT OUT THE RESULTS FROM THE MINIMIZATION PROCESS.              OSRS  35
C                                                                       OSRS  36
      IF (IER.EQ.0) PRINT 210                                           OSRS  37
      PRINT 220, KOUNT,NMIN                                             OSRS  38
      IF (IERROR.EQ.1) GO TO 50                                         OSRS  39
      IF (IERROR.EQ.2) GO TO 30                                         OSRS  40
      IF (IERROR.EQ.3) GO TO 40                                         OSRS  41
30    WRITE (6,320)                                                     OSRS  42
      GO TO 50                                                          OSRS  43
40    WRITE (6,330)                                                     OSRS  44
      GO TO 150                                                         OSRS  45
50    CONTINUE                                                          OSRS  46
      IF (IOPT.EQ.0) GO TO 80                                           OSRS  47
      DO 60 I=1,NJ                                                      OSRS  48
      DO 60 J=1,3                                                       OSRS  49
      ALBAR=DIS(J,I)+VEL(J,I)*T+ACC(J,I)*T*T/4.D0                       OSRS  50
      AC=4.D0*(XDJ(J,I)-ALBAR)/(T*T)                                    OSRS  51
      DIS(J,I)=XDJ(J,I)                                                 OSRS  52
      VEL(J,I)=VEL(J,I)+ACC(J,I)*T/2.D0+AC*T/2.D0                       OSRS  53
      ACC(J,I)=AC                                                       OSRS  54
60    CONTINUE                                                          OSRS  55
      DO 70 I=1,NM                                                      OSRS  56
      DISM(I)=U(NDF+I)                                                  OSRS  57
70    CONTINUE                                                          OSRS  58
80    CONTINUE                                                          OSRS  59
      PRINT 170                                                         OSRS  60
      PRINT 180, (STRAIN(I),I=1,NM)                                     OSRS  61
      PRINT 230, FUN,(G(I),I=1,N)                                       OSRS  62
      PRINT 240                                                         OSRS  63
      DO 90 J=1,NJ                                                      OSRS  64
      PRINT 250, J,(XDJ(I,J),I=1,3)                                     OSRS  65
```

```
90    CONTINUE                                                          OSRS  66
      PRINT 270                                                         OSRS  67
      DO 100 M=1,NM                                                     OSRS  68
      PRINT 280, M,U(NDF+M)                                             OSRS  69
100   CONTINUE                                                          OSRS  70
      IF (IOPT.EQ.0) GO TO 130                                          OSRS  71
      PRINT 190                                                         OSRS  72
      DO 110 J=1,NJ                                                     OSRS  73
      PRINT 250, J,(VEL(I,J),I=1,3)                                     OSRS  74
110   CONTINUE                                                          OSRS  75
      PRINT 200                                                         OSRS  76
      DO 120 J=1,NJ                                                     OSRS  77
      PRINT 250, J,(ACC(I,J),I=1,3)                                     OSRS  78
120   CONTINUE                                                          OSRS  79
130   PRINT 290                                                         OSRS  80
      DO 140 M=1,NM                                                     OSRS  81
      CALL DEFO (M)                                                     OSRS  82
      U4=U(NDF+M)                                                       OSRS  83
      MNEG=-M                                                           OSRS  84
      CALL STNG (MNEG)                                                  OSRS  85
      PRINT 310, M,ETOP2,EUQP2,EMID2,ELQP2,EBOT2                        OSRS  86
      PRINT 300, M,ETOP1,EUQP1,EMID1,ELQP1,EBOT1                        OSRS  87
140   CONTINUE                                                          OSRS  88
150   RETURN                                                            OSRS  89
C                                                                       OSRS  90
160   FORMAT (5X,13HENERGY ERRORS//10X,18HFULL TIME ERROR=   ,D16.8)    OSRS  91
170   FORMAT (////1X,29HSTRAIN ENERGY FOR EACH MEMBER/)                 OSRS  92
180   FORMAT (1X,5(D23.16,2X))                                          OSRS  93
190   FORMAT (1H0,T24,16HJOINT VELOCITIES//T7,5HJOINT,12X,2HDX,19X,2HDY,OSRS  94
     119X,2HDZ//)                                                       OSRS  95
200   FORMAT (1H0,T24,19HJOINT ACCELERATIONS//T7,5HJOINT,12X,2HDX,19X,2HOSRS  96
     1DY,19X,2HDZ//)                                                    OSRS  97
```

```
210   FORMAT (1HO,T10,24HCONVERGENCE WAS OBTAINED/)                          OSRS  98
220   FORMAT (1HO,T10,I4,29H MINIMIZATIONS HAVE BEEN MADE/T10,I4,27H FUNOSRS  99
     1CTION SUBROUTINE CALLS.///)                                           OSRS 100
230   FORMAT (1HO,T10,17HFUNCTION VALUE = ,D16.8,//T10,34HVALUES OF EQUIOSRS 101
     1LIBRIUM EQUATIONS  ://20(T10,6(D16.8,2X)/))                           OSRS 102
240   FORMAT (1HO,T24,19HJOINT DISPLACEMENTS//T7,5HJOINT,12X,2HDX,19X,2HOSRS 103
     1DY,19X,2HDZ//)                                                        OSRS 104
250   FORMAT (1H ,T9,I2,6X,3(D16.7,5X)/)                                     OSRS 105
260   FORMAT (1HO,T28,11HJOINT LOADS//T7,5HJOINT,13X,1HX,20X,1HY,20X,1HZOSRS 106
     1//)                                                                   OSRS 107
270   FORMAT (1HO,T17,26HINTERNAL NODE DISPLACEMENT//T18,7HELEMENT,12X,20SRS 108
     1HU4//)                                                                OSRS 109
280   FORMAT (1H ,T21,I2,7X,D16.8/)                                          OSRS 110
290   FORMAT (1HO/T20,20HELEMENT STRAIN STATE//T6,7HELEMENT,2X,7HSECTIONOSRS 111
     1,8X,3HTOP,14X,12HUPPER QTR PT,11X,8HCENTROID,11X,12HLOWER QTR PT,10SRS 112
     22X,6HBOTTOM/)                                                         OSRS 113
300   FORMAT (1H ,T9,I2,7X,1HL,5(5X,D16.8)/)                                 OSRS 114
310   FORMAT (1H ,T9,I2,7X,1HR,5(5X,D16.8)/)                                 OSRS 115
320   FORMAT (//1X,46H*** TIME STEP INCREASED, PROBLEM CONTINUED ***)        OSRS 116
330   FORMAT (//1X,69H*** TIME STEP DECREASED, MINIMIZATION REPEATED WITOSRS 117
     1H NEW TIME STEP ***)                                                  OSRS 118
      END                                                                   OSRS 119
```

```
      SUBROUTINE NSO1AD (N,X,F,AJINV,DSTEP,DMAX,ACC,MAXFUN,IPR,W,FSQ,IERPOWL    2
     1R,NH)                                                              POWL    3
C                                                                        POWL    4
C              NSO1AD MINIMIZES THE OBJECTIVE FUNCTION BY THE HYBRID METHOPOWL    5
C         OF POWELL.                                                     POWL    6
C                                                                        POWL    7
      IMPLICITREAL*8(A-H,O-Z)                                            POWL    8
      COMMON /NSO1BD/ LP,LPD                                             POWL    9
      DIMENSION X(N), F(N), AJINV(N,N), W(NH)                            POWL   10
      DATA LP,LPD/6,6/                                                   POWL   11
C     SET VARIOUS PARAMETERS                                             POWL   12
      MAXC=0                                                             POWL   13
      IPRINT=IPR                                                         POWL   14
      IERR=0                                                             POWL   15
      ITER=0                                                             POWL   16
      IF (LP.LE.0) IPRINT=0                                              POWL   17
C     'MAXC' COUNTS THE NUMBER OF CALLS OF FUNCT                         POWL   18
      NT=N+4                                                             POWL   19
      NTEST=NT                                                           POWL   20
C     'NT' AND 'NTEST' CAUSE AN ERROR RETURN IF F(X) DOES NOT DECREASE   POWL   21
      DTEST=DFLOAT(N+N)-0.5D0                                            POWL   22
C     'DTEST' IS USED TO MAINTAIN LINEAR INDEPENDENCE                    POWL   23
      NX=N*N                                                             POWL   24
      NF=NX+N                                                            POWL   25
      NW=NF+N                                                            POWL   26
      MW=NW+N                                                            POWL   27
      NDC=MW+N                                                           POWL   28
      ND=NDC+N                                                           POWL   29
C     THESE PARAMETERS SEPARATE THE WORKING SPACE ARRAY W                POWL   30
      FMIN=0.0D0                                                         POWL   31
C     USUALLY 'FMIN' IS THE LEAST CALCULATED VALUE OF F(X),              POWL   32
C     AND THE BEST X IS IN W(NX+1) TO W(NX+N)                            POWL   33
```

```
      DD=0.0D0                                                          POWL  34
C     USUALLY DD IS THE SQUARE OF THE CURRENT STEP LENGTH               POWL  35
      DSS=DSTEP*DSTEP                                                   POWL  36
      DM=DMAX*DMAX                                                      POWL  37
      DMM=4.0D0*DM                                                      POWL  38
      IS=5                                                              POWL  39
C     'IS' CONTROLS A 'GO TO' STATEMENT FOLLOWING A CALL OF FUNCT       POWL  40
      TINC=1.0D0                                                        POWL  41
C     'TINC' IS USED IN THE CRITERION TO INCREASE THE STEP LENGTH       POWL  42
C     START A NEW PAGE FOR PRINTING                                     POWL  43
      IF (IPRINT.NE.0) WRITE (LP,890)                                   POWL  44
C     CALL THE SUBROUTINE FUNCT                                         POWL  45
20    MAXC=MAXC+1                                                       POWL  46
      CALL FUNCT (N,X,F)                                                POWL  47
C     TEST FOR CONVERGENCE                                              POWL  48
      FSQ=0.0D0                                                         POWL  49
      DO 30 I=1,N                                                       POWL  50
      FSQ=FSQ+F(I)*F(I)                                                 POWL  51
30    CONTINUE                                                          POWL  52
      IF (FSQ-ACC) 40,40,60                                             POWL  53
C     PROVIDE PRINTING OF FINAL SOLUTION IF REQUESTED                   POWL  54
40    IF (IPRINT.EQ.0) GO TO 50                                         POWL  55
      WRITE (LP,900) MAXC                                               POWL  56
      WRITE (LP,910) (I,X(I),F(I),I=1,N)                                POWL  57
      WRITE (LP,920) FSQ                                                POWL  58
50    RETURN                                                            POWL  59
C     TEST FOR ERROR RETURN BECAUSE F(X) DOES NOT DECREASE              POWL  60
60    GO TO (70,150,150,70,150), IS                                     POWL  61
70    IF (FSQ-FMIN) 140,80,80                                           POWL  62
80    IF (DD-DSS) 90,90,150                                             POWL  63
90    NTEST=NTEST-1                                                     POWL  64
      IF (NTEST) 130,100,150                                            POWL  65
```

```
100   IF (LPD.GT.0) WRITE (LPD,930) NT                                  POWL  66
      IERR=1                                                            POWL  67
110   DO 120 I=1,N                                                      POWL  68
      X(I)=W(NX+I)                                                      POWL  69
      F(I)=W(NF+I)                                                      POWL  70
120   CONTINUE                                                          POWL  71
      FSQ=FMIN                                                          POWL  72
      GO TO 40                                                          POWL  73
C     ERROR RETURN BECAUSE A NEW JACOBIAN IS UNSUCCESSFUL               POWL  74
130   IF (LPD.GT.0) WRITE (LPD,940)                                     POWL  75
      IERR=2                                                            POWL  76
      GO TO 110                                                         POWL  77
140   NTEST=NT                                                          POWL  78
C     TEST WHETHER THERE HAVE BEEN MAXFUN CALLS OF FUNCT                POWL  79
150   IF (MAXFUN-MAXC) 160,160,170                                      POWL  80
160   IF (LPD.GT.0) WRITE (LPD,950) MAXC                                POWL  81
      IERR=3                                                            POWL  82
      IF (FSQ-FMIN) 40,110,110                                          POWL  83
C     PROVIDE PRINTING IF REQUESTED                                     POWL  84
170   IF (IPRINT.LE.0) GO TO 180                                        POWL  85
      IF (MOD(MAXC,IPRINT).NE.0) GO TO 180                              POWL  86
      WRITE (LP,960) MAXC                                               POWL  87
      WRITE (LP,910) (I,X(I),F(I),I=1,N)                                POWL  88
      WRITE (LP,920) FSQ                                                POWL  89
180   GO TO (720,790,230,760,190), IS                                   POWL  90
C     STORE THE RESULT OF THE INITIAL CALL OF FUNCT                     POWL  91
190   FMIN=FSQ                                                          POWL  92
      DO 200 I=1,N                                                      POWL  93
      W(NX+I)=X(I)                                                      POWL  94
      W(NF+I)=F(I)                                                      POWL  95
200   CONTINUE                                                          POWL  96
C     CALCULATE A NEW JACOBIAN APPROXIMATION                            POWL  97
```

```
210   IC=0                                                                  POWL  98
      IS=3                                                                  POWL  99
220   IC=IC+1                                                               POWL 100
      X(IC)=X(IC)+DSTEP                                                     POWL 101
      GO TO 20                                                              POWL 102
230   K=IC                                                                  POWL 103
      DO 240 I=1,N                                                          POWL 104
      W(K)=(F(I)-W(NF+I))/DSTEP                                             POWL 105
      K=K+N                                                                 POWL 106
240   CONTINUE                                                              POWL 107
      X(IC)=W(NX+IC)                                                        POWL 108
      IF (IC-N) 220,250,250                                                 POWL 109
C     CALCULATE THE INVERSE OF THE JACOBIAN AND SET THE DIRECTION MATRIX    POWL 110
250   K=0                                                                   POWL 111
      DO 270 I=1,N                                                          POWL 112
      DO 260 J=1,N                                                          POWL 113
      K=K+1                                                                 POWL 114
      AJINV(I,J)=W(K)                                                       POWL 115
      W(ND+K)=0.0D0                                                         POWL 116
260   CONTINUE                                                              POWL 117
      W(NDC+K+I)=1.0D0                                                      POWL 118
      W(NDC+I)=1.0D0+DFLOAT(N-I)                                            POWL 119
270   CONTINUE                                                              POWL 120
      CALL MB01CD (AJINV,N,N,X,F)                                           POWL 121
C     START ITERATION BY PREDICTING THE DESCENT AND NEWTON MINIMA           POWL 122
280   DS=0.0D0                                                              POWL 123
      DN=0.0D0                                                              POWL 124
      SP=0.0D0                                                              POWL 125
      ITER=ITER+1                                                           POWL 126
      IF (IPRINT.GE.0) GO TO 290                                            POWL 127
      IF (MOD(ITER,-IPRINT).NE.0) GO TO 290                                 POWL 128
      WRITE (LP,970) ITER                                                   POWL 129
```

```
      WRITE (LP,910) (I,W(NX+I),W(NF+I),I=1,N)                          POWL 130
      WRITE (LP,920) FMIN                                               POWL 131
290   DO 310 I=1,N                                                      POWL 132
      X(I)=0.0D0                                                        POWL 133
      F(I)=0.0D0                                                        POWL 134
      K=I                                                               POWL 135
      DO 300 J=1,N                                                      POWL 136
      X(I)=X(I)-W(K)*W(NF+J)                                            POWL 137
      F(I)=F(I)-AJINV(I,J)*W(NF+J)                                      POWL 138
      K=K+N                                                             POWL 139
300   CONTINUE                                                          POWL 140
      DS=DS+X(I)*X(I)                                                   POWL 141
      DN=DN+F(I)*F(I)                                                   POWL 142
      SP=SP+X(I)*F(I)                                                   POWL 143
310   CONTINUE                                                          POWL 144
C     TEST WHETHER A NEARBY STATIONARY POINT IS PREDICTED               POWL 145
      IF (FMIN*FMIN-DMM*DS) 360,360,320                                 POWL 146
C     IF SO THEN RETURN OR REVISE JACOBIAN                              POWL 147
320   GO TO (340,340,330), IS                                           POWL 148
330   IF (LPD.GT.0) WRITE (LPD,980)                                     POWL 149
      IERR=4                                                            POWL 150
      GO TO 110                                                         POWL 151
340   NTEST=0                                                           POWL 152
      DO 350 I=1,N                                                      POWL 153
      X(I)=W(NX+I)                                                      POWL 154
350   CONTINUE                                                          POWL 155
      GO TO 210                                                         POWL 156
C     TEST WHETHER TO APPLY THE FULL NEWTON CORRECTION                  POWL 157
360   IS=2                                                              POWL 158
      IF (DN-DD) 370,370,390                                            POWL 159
370   DD=DMAX1(DN,DSS)                                                  POWL 160
      DS=2.5D-1*DN                                                      POWL 161
```

```
      TINC=1.0D0                                                        POWL 162
      IF (DN-DSS) 380,540,540                                           POWL 163
380   IS=4                                                              POWL 164
      GO TO 690                                                         POWL 165
C     CALCULATE THE LENGTH OF THE STEEPEST DESCENT STEP                 POWL 166
390   K=0                                                               POWL 167
      DMULT=0.0D0                                                       POWL 168
      DO 410 I=1,N                                                      POWL 169
      DW=0.0D0                                                          POWL 170
      DO 400 J=1,N                                                      POWL 171
      K=K+1                                                             POWL 172
      DW=DW+W(K)*X(J)                                                   POWL 173
400   CONTINUE                                                          POWL 174
      DMULT=DMULT+DW*DW                                                 POWL 175
410   CONTINUE                                                          POWL 176
      DMULT=DS/DMULT                                                    POWL 177
      DS=DS*DMULT*DMULT                                                 POWL 178
C     TEST WHETHER TO USE THE STEEPEST DESCENT DIRECTION                POWL 179
      IF (DS-DD) 450,420,420                                            POWL 180
C     TEST WHETHER THE INITIAL VALUE OF DD HAS BEEN SET                 POWL 181
420   IF (DD) 430,430,440                                               POWL 182
430   DD=DMAX1(DSS,DMIN1(DM,DS))                                       POWL 183
      DS=DS/(DMULT*DMULT)                                               POWL 184
      GO TO 360                                                         POWL 185
C     SET THE MULTIPLIER OF THE STEEPEST DESCENT DIRECTION              POWL 186
440   ANMULT=0.0D0                                                      POWL 187
      DMULT=DMULT*DSQRT(DD/DS)                                          POWL 188
      GO TO 460                                                         POWL 189
C     INTERPOLATE BETWEEN THE STEEPEST DESCENT AND THE NEWTON DIRECTIONSPOWL 190
450   SP=SP*DMULT                                                       POWL 191
      ANMULT=(DD-DS)/((SP-DS)+DSQRT((SP-DD)**2+(DN-DD)*(DD-DS)))        POWL 192
      DMULT=DMULT*(1.0D0-ANMULT)                                        POWL 193
```

```
C     CALCULATE THE CHANGE IN X AND ITS ANGLE WITH THE FIRST DIRECTION  POWL 194
460   DN=0.0D0                                                          POWL 195
      SP=0.0D0                                                          POWL 196
      DO 470 I=1,N                                                      POWL 197
      F(I)=DMULT*X(I)+ANMULT*F(I)                                       POWL 198
      DN=DN+F(I)*F(I)                                                   POWL 199
      SP=SP+F(I)*W(ND+I)                                                POWL 200
470   CONTINUE                                                          POWL 201
      DS=2.5D-1*DN                                                      POWL 202
C     TEST WHETHER AN EXTRA STEP IS NEEDED FOR INDEPENDENCE             POWL 203
      IF (W(NDC+1)-DTEST) 540,540,480                                   POWL 204
480   IF (SP*SP-DS) 500,540,540                                         POWL 205
C     TAKE THE EXTRA STEP AND UPDATE THE DIRECTION MATRIX               POWL 206
490   IS=2                                                              POWL 207
500   DO 510 I=1,N                                                      POWL 208
      X(I)=W(NX+I)+DSTEP*W(ND+I)                                        POWL 209
      W(NDC+I)=W(NDC+I+1)+1.0D0                                         POWL 210
510   CONTINUE                                                          POWL 211
      W(ND)=1.0D0                                                       POWL 212
      DO 530 I=1,N                                                      POWL 213
      K=ND+I                                                            POWL 214
      SP=W(K)                                                           POWL 215
      DO 520 J=2,N                                                      POWL 216
      W(K)=W(K+N)                                                       POWL 217
      K=K+N                                                             POWL 218
520   CONTINUE                                                          POWL 219
      W(K)=SP                                                           POWL 220
530   CONTINUE                                                          POWL 221
      GO TO 20                                                          POWL 222
C     EXPRESS THE NEW DIRECTION IN TERMS OF THOSE OF THE DIRECTION      POWL 223
C     MATRIX, AND UPDATE THE COUNTS IN W(NDC+1) ETC.                    POWL 224
540   SP=0.0D0                                                          POWL 225
```

```
      K=ND                                                              POWL 226
      DO 600 I=1,N                                                      POWL 227
      X(I)=DW                                                           POWL 228
      DW=0.0D0                                                          POWL 229
      DO 550 J=1,N                                                      POWL 230
      K=K+1                                                             POWL 231
      DW=DW+F(J)*W(K)                                                   POWL 232
550   CONTINUE                                                          POWL 233
      GO TO (580,560), IS                                               POWL 234
560   W(NDC+I)=W(NDC+I)+1.0D0                                           POWL 235
      SP=SP+DW*DW                                                       POWL 236
      IF (SP-DS) 600,600,570                                            POWL 237
570   IS=1                                                              POWL 238
      KK=I                                                              POWL 239
      X(1)=DW                                                           POWL 240
      GO TO 590                                                         POWL 241
580   X(I)=DW                                                           POWL 242
590   W(NDC+I)=W(NDC+I+1)+1.0D0                                         POWL 243
600   CONTINUE                                                          POWL 244
      W(ND)=1.0D0                                                       POWL 245
C     REORDER THE DIRECTIONS SO THAT KK IS FIRST                        POWL 246
      IF (KK-1) 640,640,610                                             POWL 247
610   KS=NDC+KK*N                                                       POWL 248
      DO 630 I=1,N                                                      POWL 249
      K=KS+I                                                            POWL 250
      SP=W(K)                                                           POWL 251
      DO 620 J=2,KK                                                     POWL 252
      W(K)=W(K-N)                                                       POWL 253
      K=K-N                                                             POWL 254
620   CONTINUE                                                          POWL 255
      W(K)=SP                                                           POWL 256
630   CONTINUE                                                          POWL 257
```

```
C     GENERATE THE NEW ORTHOGONAL DIRECTION MATRIX                      POWL 258
640   DO 650 I=1,N                                                      POWL 259
      W(NW+I)=0.0D0                                                     POWL 260
650   CONTINUE                                                          POWL 261
      SP=X(1)*X(1)                                                      POWL 262
      K=ND                                                              POWL 263
      DO 670 I=2,N                                                      POWL 264
      DS=DSQRT(SP*(SP+X(I)*X(I)))                                       POWL 265
      DW=SP/DS                                                          POWL 266
      DS=X(I)/DS                                                        POWL 267
      SP=SP+X(I)*X(I)                                                   POWL 268
      DO 660 J=1,N                                                      POWL 269
      K=K+1                                                             POWL 270
      W(NW+J)=W(NW+J)+X(I-1)*W(K)                                       POWL 271
      W(K)=DW*W(K+N)-DS*W(NW+J)                                         POWL 272
660   CONTINUE                                                          POWL 273
670   CONTINUE                                                          POWL 274
      SP=1.0D0/DSQRT(DN)                                                POWL 275
      DO 680 I=1,N                                                      POWL 276
      K=K+1                                                             POWL 277
      W(K)=SP*F(I)                                                      POWL 278
680   CONTINUE                                                          POWL 279
C     CALCULATE THE NEXT VECTOR X, AND PREDICT THE RIGHT HAND SIDES     POWL 280
690   FNP=0.0D0                                                         POWL 281
      K=0                                                               POWL 282
      DO 710 I=1,N                                                      POWL 283
      X(I)=W(NX+I)+F(I)                                                 POWL 284
      W(NW+I)=W(NF+I)                                                   POWL 285
      DO 700 J=1,N                                                      POWL 286
      K=K+1                                                             POWL 287
      W(NW+I)=W(NW+I)+W(K)*F(J)                                         POWL 288
700   CONTINUE                                                          POWL 289
```

```
      FNP=FNP+W(NW+I)**2                                                POWL 290
  710 CONTINUE                                                          POWL 291
C     CALL CALFUN USING THE NEW VECTOR OF VARIABLES                     POWL 292
      GO TO 20                                                          POWL 293
C     UPDATE THE STEP SIZE                                              POWL 294
  720 DMULT=9.0D-1*FMIN+1.0D-1*FNP-FSQ                                  POWL 295
      IF (DMULT) 730,740,740                                            POWL 296
  730 DD=DMAX1(DSS,2.5D-1*DD)                                           POWL 297
      TINC=1.0D0                                                        POWL 298
      IF (FSQ-FMIN) 770,790,790                                         POWL 299
C     TRY THE TEST TO DECIDE WHETHER TO INCREASE THE STEP LENGTH        POWL 300
  740 SP=0.0D0                                                          POWL 301
      SS=0.0D0                                                          POWL 302
      DO 750 I=1,N                                                      POWL 303
      SP=SP+DABS(F(I)*(F(I)-W(NW+I)))                                   POWL 304
      SS=SS+(F(I)-W(NW+I))**2                                           POWL 305
  750 CONTINUE                                                          POWL 306
      PJ=1.+DMULT/(SP+DSQRT(SP*SP+DMULT*SS))                            POWL 307
      SPNS=4.0D0                                                        POWL 308
      SP=DMIN1(SPNS,TINC,PJ)                                            POWL 309
      TINC=PJ/SP                                                        POWL 310
      DD=DMIN1(DM,SP*DD)                                                POWL 311
      GO TO 770                                                         POWL 312
C     IF F(X) IMPROVES STORE THE NEW VALUE OF X                         POWL 313
  760 IF (FSQ-FMIN) 770,490,490                                         POWL 314
  770 FMIN=FSQ                                                          POWL 315
      DO 780 I=1,N                                                      POWL 316
      SP=X(I)                                                           POWL 317
      X(I)=W(NX+I)                                                      POWL 318
      W(NX+I)=SP                                                        POWL 319
      SP=F(I)                                                           POWL 320
      F(I)=W(NF+I)                                                      POWL 321
```

```
      W(NF+I)=SP                                                        POWL 322
      W(NW+I)=-W(NW+I)                                                  POWL 323
  780 CONTINUE                                                          POWL 324
      IF (IS-1) 790,790,490                                             POWL 325
C     CALCULATE THE CHANGES IN F AND IN X                               POWL 326
  790 DO 800 I=1,N                                                      POWL 327
      X(I)=X(I)-W(NX+I)                                                 POWL 328
      F(I)=F(I)-W(NF+I)                                                 POWL 329
  800 CONTINUE                                                          POWL 330
C     UPDATE THE APPROXIMATIONS TO J AND TO AJINV                       POWL 331
      K=0                                                               POWL 332
      DO 820 I=1,N                                                      POWL 333
      W(MW+I)=X(I)                                                      POWL 334
      W(NW+I)=F(I)                                                      POWL 335
      DO 810 J=1,N                                                      POWL 336
      W(MW+I)=W(MW+I)-AJINV(I,J)*F(J)                                   POWL 337
      K=K+1                                                             POWL 338
      W(NW+I)=W(NW+I)-W(K)*X(J)                                         POWL 339
  810 CONTINUE                                                          POWL 340
  820 CONTINUE                                                          POWL 341
      SP=0.0D0                                                          POWL 342
      SS=0.0D0                                                          POWL 343
      DO 840 I=1,N                                                      POWL 344
      DS=0.0D0                                                          POWL 345
      DO 830 J=1,N                                                      POWL 346
      DS=DS+AJINV(J,I)*X(J)                                             POWL 347
  830 CONTINUE                                                          POWL 348
      SP=SP+DS*F(I)                                                     POWL 349
      SS=SS+X(I)*X(I)                                                   POWL 350
      F(I)=DS                                                           POWL 351
  840 CONTINUE                                                          POWL 352
      DMULT=1.0D0                                                       POWL 353
```

```
      IF (DABS(SP)-0.1*SS) 850,860,860                                  POWL 354
850   DMULT=8.0D-1                                                      POWL 355
860   PJ=DMULT/SS                                                       POWL 356
      PA=DMULT/(DMULT*SP+(1.-DMULT)*SS)                                 POWL 357
      K=0                                                               POWL 358
      DO 880 I=1,N                                                      POWL 359
      SP=PJ*W(NW+I)                                                     POWL 360
      SS=PA*W(MW+I)                                                     POWL 361
      DO 870 J=1,N                                                      POWL 362
      K=K+1                                                             POWL 363
      W(K)=W(K)+SP*X(J)                                                 POWL 364
      AJINV(I,J)=AJINV(I,J)+SS*F(J)                                     POWL 365
870   CONTINUE                                                          POWL 366
880   CONTINUE                                                          POWL 367
      GO TO 280                                                         POWL 368
C                                                                       POWL 369
890   FORMAT (1H1)                                                      POWL 370
900   FORMAT (///5X,47HTHE FINAL SOLUTION CALCULATED BY NS01A REQUIRED,IPOWL 371
     15,24H CALLS OF FUNCT , AND IS)                                    POWL 372
910   FORMAT (//4X,1HI,7X,4HX(I),12X,4HF(I)//(I5,2E17.8))              POWL 373
920   FORMAT (/5X,21HTHE SUM OF SQUARES IS,E17.8)                       POWL 374
930   FORMAT (///5X,31HERROR RETURN FROM NS01A BECAUSE,I5,48H CALLS OF FPOWL 375
     1UNCT  FAILED TO IMPROVE THE RESIDUALS)                            POWL 376
940   FORMAT (///5X,37HERROR RETURN FROM NS01A BECAUSE F(X) ,39HFAILED TPOWL 377
     1O DECREASE USING A NEW JACOBIAN)                                  POWL 378
950   FORMAT (///5X,47HERROR RETURN FROM NS01A BECAUSE THERE HAVE BEEN,IPOWL 379
     15,16H CALLS OF FUNCT )                                            POWL 380
960   FORMAT (///5X,6HAT THE,I5,26H TH CALL OF FUNCT  WE HAVE)          POWL 381
970   FORMAT (///5X,6HAT THE,I5,63H TH ITERATION THE LEAST CALCULATED SUPOWL 382
     1M OF SQUARES IS AS FOLLOWS)                                       POWL 383
980   FORMAT (///5X,41HERROR RETURN FROM NS01A BECAUSE A NEARBY ,37HSTATPOWL 384
     1IONARY POINT OF F(X) IS PREDICTED)                                POWL 385
```

END

POWL 386

```
      SUBROUTINE MB01CD (A,M,IA,IND,C)                                  MB01    1
C                                                                       MB01    2
C        MB10AD INVERTS THE APPROXIMATE JACOBIAN MATRIX BY GAUSSIAN     MB01    3
C        ELIMINATION                                                    MB01    4
C                                                                       MB01    5
      IMPLICITREAL*8(A-H,O-Z)                                           MB01    6
      DIMENSION A(IA,M), IND(M), C(M)                                   MB01    7
      IF (M-1) 370,20,30                                                MB01    8
20    IF (A(1,1).NE.0D0) A(1,1)=100/A(1,1)                              MB01    9
      GO TO 370                                                         MB01   10
30    M1=M-1                                                            MB01   11
      AMAX=0.0D0                                                        MB01   12
      DO 50 I=1,M                                                       MB01   13
      IND(I)=I                                                          MB01   14
      IF (DABS(A(I,1))-DABS(AMAX)) 50,50,40                             MB01   15
40    AMAX=A(I,1)                                                       MB01   16
      IMAX=I                                                            MB01   17
50    CONTINUE                                                          MB01   18
      ASSIGN 120 TO JUMP                                                MB01   19
      DO 160 J=1,M1                                                     MB01   20
      IF (IMAX-J) 80,80,60                                              MB01   21
60    IW=IND(IMAX)                                                      MB01   22
      IND(IMAX)=IND(J)                                                  MB01   23
      IND(J)=IW                                                         MB01   24
      DO 70 K=1,M                                                       MB01   25
      W=A(IMAX,K)                                                       MB01   26
      A(IMAX,K)=A(J,K)                                                  MB01   27
      A(J,K)=W                                                          MB01   28
70    CONTINUE                                                          MB01   29
80    J1=J+1                                                            MB01   30
      GO TO JUMP, (90,120)                                              MB01   31
90    DO 110 I=J1,M                                                     MB01   32
```

```
      LL=J-1                                                          MB01  33
      DO 100 K=1,LL                                                   MB01  34
100   A(J,I)=A(J,I)-A(J,K)*A(K,I)                                     MB01  35
110   CONTINUE                                                        MB01  36
120   DIV=AMAX                                                        MB01  37
      AMAX=0.0D0                                                      MB01  38
      ASSIGN 90 TO JUMP                                               MB01  39
      DO 150 I=J1,M                                                   MB01  40
      A(I,J)=A(I,J)/DIV                                               MB01  41
      DO 130 K=1,J                                                    MB01  42
130   A(I,J+1)=A(I,J+1)-A(I,K)*A(K,J+1)                               MB01  43
      IF (DABS(A(I,J1))-DABS(AMAX)) 150,150,140                       MB01  44
140   AMAX=A(I,J1)                                                    MB01  45
      IMAX=I                                                          MB01  46
150   CONTINUE                                                        MB01  47
160   CONTINUE                                                        MB01  48
      DO 220 I1=1,M1                                                  MB01  49
      I=M+1-I1                                                        MB01  50
      I2=I-1                                                          MB01  51
      DO 200 J1=1,I2                                                  MB01  52
      J=I2+1-J1                                                       MB01  53
      J2=J+1                                                          MB01  54
      W1=-A(I,J)                                                      MB01  55
      IF (I2-J2) 190,170,170                                          MB01  56
170   LL=I2-J2+1                                                      MB01  57
      DO 180 K=J2,LL                                                  MB01  58
180   W1=W1-A(K,J)*C(K)                                               MB01  59
190   C(J)=W1                                                         MB01  60
200   CONTINUE                                                        MB01  61
      DO 210 K=1,I2                                                   MB01  62
      A(I,K)=C(K)                                                     MB01  63
210   CONTINUE                                                        MB01  64
```

```
220   CONTINUE                                                    MB01  65
      DO 320 I1=1,M                                               MB01  66
      I=M+1-I1                                                    MB01  67
      I2=I+1                                                      MB01  68
      W=A(I,I)                                                    MB01  69
      DO 300 J=1,M                                                MB01  70
      IF (I-J) 230,240,250                                        MB01  71
230   W1=0.0D0                                                    MB01  72
      GO TO 260                                                   MB01  73
240   W1=1.0D0                                                    MB01  74
      GO TO 260                                                   MB01  75
250   W1=A(I,J)                                                   MB01  76
260   IF (I1-1) 290,290,270                                       MB01  77
270   LL=M-I2+1                                                   MB01  78
      DO 280 K=I2,LL                                              MB01  79
280   W1=W1-A(I,K)*A(K,J)                                         MB01  80
290   C(J)=W1                                                     MB01  81
300   CONTINUE                                                    MB01  82
      DO 310 J=1,M                                                MB01  83
      A(I,J)=C(J)/W                                               MB01  84
310   CONTINUE                                                    MB01  85
320   CONTINUE                                                    MB01  86
      DO 360 I=1,M                                                MB01  87
330   IF (IND(I)-I) 340,360,340                                   MB01  88
340   J=IND(I)                                                    MB01  89
      DO 350 K=1,M                                                MB01  90
      STO=A(K,I)                                                  MB01  91
      A(K,I)=A(K,J)                                               MB01  92
      A(K,J)=STO                                                  MB01  93
350   CONTINUE                                                    MB01  94
      ISTO=IND(J)                                                 MB01  95
      IND(J)=J                                                    MB01  96
```

```
      IND(I)=ISTO                                                       MB01  97
      GO TO 330                                                         MB01  98
360   CONTINUE                                                          MB01  99
370   RETURN                                                            MB01 100
      END                                                               MB01 101
```

VITA

David Brownfield Horne was born in Paducah, Kentucky, on June 22, 1953. He graduated from Pocahontas County High School, Dunmore, West Virginia, in June, 1971. He entered Virginia Polytechnic Institute and State University, Blacksburg, Virginia, in September, 1971 and while he was there participated in the cooperative education program. He graduated with a Bachelor of Science degree in Civil Engineering in June, 1976 from Virginia Polytechnic Institute and State University and shortly afterward began his graduate studies there.

David B. Horne

OBJECTIVE FUNCTIONS FOR NONLINEAR STRUCTURAL ANALYSIS

by

David Brownfield Horne

(ABSTRACT)

An objective function that is suitable for both stable and unstable equilibrium states since it is guaranteed to assume a relative minimum at any equilibrium state is presented. The objective function, which is the inner product of the equations of equilibrium (algorithm 2), is compared with another objective function which, for a static problem, is the total potential energy (algorithm 3). The method of formulating the mathematical model for algorithm 2 is presented in detail.

Algorithm 2 is found to be less efficient than algorithm 3. However, it is demonstrated that algorithm 2 is able to solve for equilibrium states on either the fundamental path or a bifurcation path. Hence algorithm 2 is a powerful tool for nonlinear structural analysis since it is able to predict the existence of limit and bifurcation points and to determine post-buckled equilibrium states.

Also addressed are the methods of formulating mathematical models for nonlinear structural analysis. A comparison of the methods of integrating the equations of motion for nonlinear dynamics problems and the selection of an appropriate time step for the time integration schemes are presented.