J. RAHIMZADEH-HANACHI

PH. D.

J. RAHIMZADEH-HANACHI PH. D.

J. RAHIMZADEH-HANACHI

PH. D.

This is to certify that the

thesis entitled

NONLINEAR ELASTIC FRAME
ANALYSIS BY FINITE ELEMENT

presented by

Jalil Rahimzadeh-Hanachi

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in C.E. - Structures

[illegible]

Major professor
Dr. R. Wen

Date July 30, 1981

| | |
|---|---|
| 14 K0[illegible] | 68 K024 |
| 49 K117 | 48 K06[illegible] |
| 7 K[illegible] | 400 A052 |
| 136 A148 | ~~NOV 0 4 1989~~ ~~JAN 04~~ FEB 6 |
| 32 K343 | CX A037 |
| 88 K010 | |

# NONLINEAR ELASTIC FRAME ANALYSIS BY FINITE ELEMENT

by

Jalil Rahimzadeh-Hanachi

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Civil and Sanitary Engineering

1981

## ABSTRACT

NONLINEAR ELASTIC FRAME ANALYSIS BY FINITE ELEMENT

By

Jalil Rahimzadeh-Hanachi

Several methods of analysis of nonlinear elastic framed structures are discussed. A method of analysis is defined to consist of three components: (a) a finite element model, (b) local coordinates (Eulerian or Lagrangian) for the element, and (c) a solution process. The finite element models are based on a linear longitudinal displacement function and a cubic transverse displacement function. However, two versions of the contribution to the axial strain by the transverse displacement are considered: one quartic and one constant.

Both the Eulerian and Lagrangian coordinates are considered for the specification of the element local displacements. In addition, two versions are employed for the Lagrangian formulation: one with a fixed coordinate system and the other a moving (updated) coordinate system.

Solution processes considered include the Newton-Raphson, the one-step Newton-Raphson and a straight incremental procedure. Past contributions are pointed out in the framework as outlined above. They include the works of Martin, Jennings, Mallet and Marcal, Powell, Holzer and Somers, Ebner and Ucciferro, Oran, Bathe, Akkoush, et al., and others. The finite element results are compared among themselves and with the numerical solutions corresponding to the "exact" beam-column formulation.

In addition, the identification of bifurcation loads is discussed. The formulation of eigenproblems and the accuracy of their solutions as estimates of bifurcation loads are also considered. Recognizing that in practical applications the number of members in a structure system is likely to be large, emphasis is placed on the effectiveness of using a single finite element to represent a beam (- column) member in a framed structure. In this regard, the results seem to indicate that a most effective method would be using the finite element with a constant axial strain (which of course includes the effects of transverse displacements), Lagrangian, fixed coordinates, and the Newton-Raphson algorithm.

## ACKNOWLEDGMENTS

The investigation reported in this dissertation has been made possible with the support of many people to whom I am greatly indebted. I would like to express special appreciation for the advice and encouragement of my dissertation advisor, Dr. Robert K. Wen, throughout the long months of research and writing. Thanks are also due to members of the writer's Guidance Committee: Dr. William Bradley, Dr. Charles Cutts, Dr. James Lubkin and Dr. Nicholas Altiero. The support and encouragement of Dr. William Taylor, Chairman of the Department of Civil Engineering, are sincerely appreciated. I would also like to express my appreciation to the National Science Foundation for their support of the research and to Nancy Hunt and Vicki Brannan for their dedicated typing.

TABLE OF CONTENTS

Page

## LIST OF TABLES

LIST OF FIGURES

# CHAPTER I

# INTRODUCTION

## 1.1 BACKGROUND

In recent years the concept of basing structural design on ultimate strength has gained increasing acceptance. The computation of the ultimate strength of a structure would generally involve load-displacement relationships that are nonlinear. In other words, nonlinear structural analysis becomes necessary.

Nonlinear behavior of structures may be the result of two sources: (a) "material nonlinearity" such as a nonlinear stress-strain relation, and (b) "geometric nonlinearity" which represents the effect of the distortion of the structure on its response.

In the present study we shall exclude the effects of "material nonlinearity" and consider only "geometric nonlinearity", although the exclusion of material nonlinearity would place a limitation on the direct application of the results to the design of many types of engineering structures. The study, however, represents a fundamental step. For slender structures such as suspension bridges and perhaps arch bridges, elastic nonlinearity is of direct concern.

Nonlinear elastic analysis of framed structures has been the subject of investigation by a number of researchers. In the following review, past works will be discussed as two groups. One group considers the basic beam element as a continuum. The "exact" method would use the correct expression for the curvature of beams, and is called the "theory of elastica" (Timoshenko [1]*). But most works have been based on the use of an approximate expression for the curvature (equal to the second derivative of the lateral deflection), and the resulting theory is referred to as "beam-column theory." (Timoshenko [1], Bleich [2]). The second group consists of those works that use finite elements to model the members of a frame.

It is instructive to note that, in addition to the basic element model discussed above, a method of analysis has two more attributes. The first is the local coordinate system used which could be either Eulerian or Lagrangian. In the "Eulerian coordinate formulation" local displacements are measured with respect to the chord of the deformed member, and in the "Lagrangian coordinate formulation" local displacements are measured with respect to the axis of the undeformed member. The second attribute is the method of solution. At least four methods have been used: the method of "Direct Substitution", Newton-Raphson, One-step Newton-Raphson, and "Straight Incremental" (Cook [3], Haisler [4]). The last three methods of solution have been used here and are described in Chapter III.

## 1.1.1 WORKS BASED ON THE BEAM-COLUMN MODEL

The essence of the beam-column theory is the inclusion of the effect of the axial force on the bending moment of a deflected beam. Discussion of this can be found in various monographs (Timoshenko [1], Bleich [2]).

Detailed expressions representing the exact solution of the

* Number in brackets refer to entries in the list of references.

beam-column problem were given by Saafan [5]. He also derived a tangent stiffness matrix [6] for use in a Newton-Raphson method of solution. However, the effect of bowing (flexural deformation) on the axial shortening was neglected in the tangent stiffness matrix.

Conner, Logcher, and Chan [7], using the principle of virtual displacement, developed the stiffness and tangent stiffness matrices in two and three dimensions. The method was based on fixed Lagrangian coordinates (the local coordinates are fixed, i.e., are not updated after each incremental loading). It is good only for deformations involving small rotations. Methods of solution discussed included that of successive iteration, Newton-Raphson and the straight incremental method.

Oran [8,9] formulated for both two and three dimensional problems the "exact" tangent stiffness matrix in "Eulerian coordinates". Subsequently, he and Kassimali [10] applied these matrices to obtain solutions to a number of numerical problems. The Newton-Raphson and straight incremental methods were used. Nonlinear load-displacement behavior as well as stability were discussed. The accuracy of the solutions was shown to be generally excellent even for very large displacements.

## 1.1.2 WORKS BASED ON FINITE ELEMENT MODEL

Martin [11] presented one of the earliest finite element formulation to deal with geometrically nonlinear problems. The method is one of incremental loading, using the well-known geometric stiffness matrix [12] based on Lagrangian coordinates and updating the geometry of the structure at every load increment. This is referred to herein as the updated-Lagrange coordinates. Although in his method of solution there is no check on equilibrium, it is a very efficient approach.

Jennings [13] highlighted his study by including the bowing effect on the axial strain in the finite element formulation. He derived stiffness and tangent stiffness matrices based on Eulerian coordinates for plane frames. Consequently the expressions may be used for very large displacements.

Mallett and Marcal [14] presented relationships between the strain energy, the total equilibrium and incremental equilibrium equation in terms of the usual stiffness matrix and two (nonlinear) incremental stiffness matrices. Lagrange coordinates were used in the formulation. Expressions for the stiffness matrices for two dimensional beam elements were derived based on the usual cubic shape function for the lateral displacement. Since the contribution of that displacement to the axial strain is in terms of the square of its derivative, this model is referred to as the "quartic axial strain model." They presented no numerical results.

Powell [15], in illustrating a general discussion of the theory of nonlinear structures, presented the stiffness and incremental stiffness matrix for two dimensional beams. He adopted the same shape functions as those by Mallett and Marcal [14] but the stiffness matrices were derived in Eulerian coordinates.

Akkoush, Toridis, Khozeimeh and Huang [16] used the concept of geometric stiffness matrix for a three dimensional beam model in updated-Lagrange coordinates. It is essentially a generalized version of Martin's method for space frames. The method was used to generate a complete load-displacement path to study post-buckling and post-limit load behavior.

Hozler and Somers [17] developed a method for the study of the nonlinear response of reinforced concrete and steel plane frames up to collapse. Material nonlinearity was also considered. Their formulation was based on a minimization of the energy function defined by generalized coordinates and forces in an Eulerian coordinate formulation.

Bathe and Bolourchi [18] developed the stiffness matrices for three dimensional beam elements subjected to large displacements and rotations for the application to elastic, elastic-plastic, static or dynamic analysis. Their formulation was quite rigorously based on the theory of continuum mechanics. A number of numerical results were given which will also be considered later in this thesis.

A theoretical and numerical comparison of methods including those of Martin [11], Jennings [13], Mallett and Marcal [14], and Powell [15] was undertaken by Ebner and Ucciferro [19]. The study was limited to two dimensional problems. They presented derivations of the stiffness matrices related to these methods from a common starting point and thus made more clear the similarities and differences among them. Part of the numerical results they obtained for comparison studies have also been reproduced and discussed here.

Most of the previous studies have dealt with two dimensional problems and structures with a small number of members. Since in practice, three dimensional and larger systems are frequently involved, this thesis is an effort to consider this class of problems. The specific objectives and scope are discussed in the following section.

## 1.2 OBJECTIVES AND SCOPE

The primary objective of this work is to search for an effective method for the elastic nonlinear analysis of three dimensional framed structures, with a view to eventually applying it to structural systems consisting of a relatively large number of members. It was anticipated that the finite element model would be more efficient than the more accurate beam-column model; that a formulation based on the Lagrangian coordinates would be more efficient than one based on the Eulerian coordinates; and that the fixed-Lagrange formulation of the solution would be more efficient than the updated-Lagrange one.

These considerations led, in the initial phase of the work, to a development of the M & M model (Mallett and Marcal) referred to previously [14] for three dimensional beam elements. However, preliminary results indicated certain basic problems for this model. That is, for very slender members, it produced grossly inaccurate results. This motivated a comperative study of the other finite element models discussed previously.

They include Martin [11], Powell [15], Jennings [13] as well as a new model [20] developed in the course of the present research of which this thesis is a part. The new model is based on an "average axial strain assumption", i.e., the axial strain due to the lateral deflection is averaged over the element as discussed in Chapter 2. It is herein referred to as the FEA (Finite Element Average) model. For the comparative studies, the beam-column model was used as a basis.

In addition to studies of load-displacement relations, this study also included the formulation of eigenvalue problems, using finite element models, from which estimates of "bifurcation load" or "limit load" can be obtained.

The scope of this report is thus as follows:

1) To prepare computer programs for two dimensional problems based on the following methods: the beam-column method, the M & M method, Jennings' method and Powell's method.

2) To develop the stiffness matrices for three dimensional beams based on the "quartic axial strain assumption".

3) To develop computer programs for three and two dimensional problems based on the FEA formulation.

4) To formulate and solve eigenvalue problems in order to obtain estimates of bifurcation or limit loads. Both linear and quadratic eigenvalue problems were considered.

5) To obtain and compare numerical results, using the developed programs.

6) To assess the relative merits of the various methods.

In the course of the study, it was found appropriate to divide the problems into three categories: problems of "Small Displacements", "Intermediate Displacements", and "Large Displacements".

The comparison indicated that, for "Large Displacement" problems, the method would have to be based on either Eulerian coordinates or updated-Lagrange coordinates. For "Small Displacement" problems (although still involving load-displacement relationships that are quite nonlinear), the fixed-Lagrange formulation considered here is satisfactory (that is, both the M & M method and the FEA method). However, for "Intermediate Displacement" problems, the FEA method still produces reliable results while the M & M method seems to fail.

The results obtained from eigenvalue problem studies indicated that, with little primary or no bending, both linear and quadratic eigenvalue solutions agreed with results obtained from complete load-displacement solutions. For problems with substantial primary bending, linear eigensolutions still generally produced acceptable results if the structure-load system is symmetric. For asymmetric systems the significance of the eigensolutions deteriorated. However, in some cases certain linear eigensolutions were shown to represent reasonable estimates for "limit loads."

## 1.3 NOTATION

| | | |
|---|---|---|
| A | = | Area of cross section; |
| A, B | = | End nodes of an element; |
| $a_1, a_2, \ldots, a_{12}$ | = | Parameters used for definition of shape functions; |
| E | = | Young's modulus; |
| FEA | = | Finite Element Average strain model; |
| G | = | Shear modulus; |
| $I_{xx}, I_{yy}$ | = | Moment of inertia of cross section (Figure 2-2); |
| $I_\eta, I_\zeta$ | = | Moment of inertia of cross section (Figure 2-2); |
| Inc. | = | Straight incremental; |
| J | = | Torsional constant; |
| [k], [K] | = | Element and structural linear stiffness matrices; |
| $[k_{\varepsilon_o}], [K_{\varepsilon_o}]$ | = | Element and structural initial strain stiffness matrices; |
| $[k_G]$ | = | $\frac{1}{\text{Axial load}}[n_1^*]$; |
| $\ell$ | = | Length of element; |

| | |
|---|---|
| M & M | = Mallett and Marcal's method; |
| $[n_1]$, $[N_1]$ | = Element and structural first order nonlinear stiffness matrices; |
| $[n_1^*]$, $[N_1^*]$ | = Element and structural first order geometric stiffness matrices; |
| $[n_2]$, $[N_2]$ | = Element and structural second order nonlinear stiffness matrices; |
| NR | = Newton-Raphson; |
| $\{P\}$ | = External load vector; |
| $\Delta p$ | = Load step (load increment); |
| ${}^iP$ | = Axial load at the end of ith load increment in the element; |
| $P_{cr}$ | = Critical value of applied load; |
| $P_{BC}$ | = Critical load corresponding to Beam-Column solution; |
| $P_N$ | = Critical load corresponding to eigenvalue solution (using $N_1$); |
| $P_{N_1}{}^*$ | = Critical load corresponding to eigenvalue solution (using $N_1$*); |
| $P_{N_1+N_2}$ | = Critical load corresponding to quadratic eigensolution; |
| $\{P_{ref}\}$ | = Reference external load vector; |
| $\{q\}$ | = $\lfloor q_1, q_2, \ldots, q_6, q_7, q_8, \ldots, q_{12}\rfloor^T$; (Element generalized displacement vector); |
| $\{Q\}$ | = Structural generalized displacement vector; |
| $\{Q_{ref}\}$ | = Reference structural generalized displacement vector; |
| $Q$ | = Symbol for exact configuration of the structure; |

| | | |
|---|---|---|
| $Q_i$ | = | Symbol for structural configuration at the ith iteration; |
| $\{\Delta R\}_i$ | = | Unbalanced force vector related to the ith iteration; |
| R | = | Radius of circle; |
| $[S_S]$ | = | Structural secant stiffness matrix; |
| $[S_T]$ | = | Structural tangent stiffness matrix; |
| u, v, w | = | Displacements along local x, y, z axes, respectively; |
| $u_1, v_1, w_1$ and $u_2, v_2, w_2$ | = | Displacements for nodes 1 and 2 of the beam element along x, y, z axes, respectively; |
| $U_\varepsilon$ | = | Strain energy of the element; |
| $U_{\varepsilon_o}$ | = | Initial strain energy of the element; |
| $U_t$ | = | Torsional strain energy of the element; |
| $U_{TOTAL}$ | = | Total strain energy of the element; |
| $U_{\varepsilon+t}$ | = | $U_\varepsilon + U_t$ |
| $U_2, U_3, U_4$ | = | Quadratic, cubic and quartic parts of strain energy; |
| V | = | Potential energy of external loads; |
| Vol. | = | Volume; |
| x, y, z | = | Local coordinate axes; |
| X, Y, Z | = | Global coordinate axes; |
| $\alpha$ | = | Angle of opening of circular arch; |
| $\alpha$ | = | Multiplier for asymmetric loading; |
| $\varepsilon$ | = | Longitudinal strain; |
| ${}^{i}\varepsilon_o$ | = | Initial strain at the beginning of ith load increment; |

| | |
|---|---|
| $\varepsilon_d$ | = Tolerance ratio for convergence check based on displacement variation; |
| $\varepsilon_f$ | = Tolerance for convergence check based on unbalanced force vector; |
| $\phi$, $\Psi$, $\theta$ | = Rotation about x, y, z axis, respectively; |
| $\phi_1$, $\Psi_1$, $\theta_1$ and $\phi_2$, $\Psi_2$, $\theta_2$ | = Rotation about x, y, z axis for nodes 1 and 2, respectively; |
| $\Phi_P$ | = Total potential energy; |
| $\theta_o$, $\Psi_o$ | = Chord rotations about z and y axis, respectively; |
| $\lambda$ | = Buckling load parameter; |
| $\Delta$ | = Incremental operator; |
| $\{\ \}$ | = Column vector; |
| $\lfloor\ \rfloor$ | = Row vector; |
| $[\ ]$ | = Rectangular matrix; |

# CHAPTER II

# FINITE ELEMENT MODELS

## 2.1 FINITE ELEMENT MODELS FOR THREE AND TWO DIMENSIONAL BEAM ELEMENTS

### 2.1.1 GENERAL

In this chapter the strain-displacement relations for three and two dimensional beam elements are presented. Then the stiffness matrices (including the linear and nonlinear parts) are derived, and finally the equilibrium equations are written.

### 2.1.2 STRAIN ENERGY OF THREE DIMENSIONAL BEAM ELEMENTS BASED ON QUARTIC AXIAL STRAIN FUNCTION

Consider a beam element in space as shown in Figure 2-1. The x-, y-, z-axes, a right-handed coordinate system, represent the local or member coordinates. The displacements and rotations corresponding to these axes are denoted by u, v, w and $\phi$, $\Psi$, $\theta$, respectively.

The initial position of the element is AB. The length of AB is equal to $\ell$. The displaced position is $A_1B_1$ and the projections of $A_1B_1$ on the x-y plane and x-z plane are denoted by $A_1'B_1'$ and $A_1''B_1''$.

It should be noted that the following assumptions have been used in our derivation.

a) The material of the beam element is linearly elastic.

b) Plane sections remain plane after deformation.

c) The cross section of the beam is constant and has two axes of symmetry.

d) The effect of torsional deformation on normal strain is negligible.

For a finite element analysis we assume linear shape functions for u and $\phi$ and cubic shape functions for v and w, i.e.,

$$
\begin{aligned}
u &= a_1+a_2x \\
v &= a_3+a_4x+a_5x^2+a_6x^3 \\
w &= a_7+a_8x+a_9x^2+a_{10}x^3 \\
\phi &= a_{11}+a_{12}x
\end{aligned}
\qquad (2\text{-}1)
$$

The boundary conditions are:

at x=o.

$$u = u_1,\ v = v_1,\ w = w_1$$

$$\frac{dv}{dx} = \theta_1,\ \frac{dw}{dx} = -\Psi_1,\ \phi = \phi_1$$

at $n=\ell$ (2-2)

$$u = u_2,\ v = v_2,\ w = w_2$$

$$\frac{dv}{dx} = \theta_2,\ \frac{dw}{dx} = -\Psi_2,\ \phi = \phi_2$$

Substituting Equation (2-1) into Equation (2-2), we obtain a system of linear equations for the unknowns $a_1$, $a_2$, ..., $a_{12}$.

Solving the equations and substituting the results back into Equation (2-1) we have

$$
\begin{aligned}
u &= u_1 + \frac{u_2-u_1}{\ell}\,x \\
v &= v_1+\theta_1x+\frac{1}{\ell}(-2\theta_1-\theta_2+3\theta_o)x^2+\frac{1}{\ell^2}(\theta_1+\theta_2-2\theta_o)x^3 \\
w &= w_1-\Psi_1x+\frac{1}{\ell}(2\Psi_1+\Psi_2-3\Psi_o)x^2+\frac{1}{\ell^2}(-\Psi_1-\Psi_2+2\Psi_o)x^3 \\
\phi &= \phi_1 + \frac{\phi_2-\phi_1}{\ell}\,x
\end{aligned}
\qquad (2\text{-}3)
$$

in which

$$\theta_o = \frac{v_2 - v_1}{\ell}$$

and (2-4)

$$\psi_o = \frac{-(w_2 - w_1)}{\ell}$$

Following the usual beam theory assumption of plane sections remaining plane, the longitudinal strain at each point of the beam element may be written as:

$$\varepsilon(x,\eta,\zeta) = \varepsilon_a(x) + \eta \frac{d^2v}{dx^2} + \zeta \frac{d^2w}{dx^2} \qquad (2\text{-}5)$$

in which $\varepsilon_a(x)$ is the axial strain at the centroid, and $\eta$ and $\zeta$ are the coordinates of the point with respect to the principal axes of the cross section plane as shown in Figure 2-2.

The axial strain at the centroid is

$$\varepsilon_a(x) = \frac{du}{dx} + \frac{1}{2}\left(\frac{dv}{dx}\right)^2 + \frac{1}{2}\left(\frac{dw}{dx}\right)^2 \qquad (2\text{-}6)$$

in which the last two terms represent the nonlinear effects of bending. Thus it is seen that when v and w are cubic functions of x, $\varepsilon_a(x)$ is quartic. Using Equation (2-6), Equation (2-5) becomes:

$$\varepsilon(x,\eta,\zeta) = \frac{du}{dx} + \frac{1}{2}\left(\frac{dv}{dx}\right)^2 + \frac{1}{2}\left(\frac{dw}{dx}\right)^2 + \eta \frac{d^2v}{dx^2} + \zeta \frac{d^2w}{dx^2} \qquad (2\text{-}7)$$

From equations (2-3) we obtain:

$$\frac{du}{dx} = \frac{u_2-u_1}{\ell}$$

$$\frac{dv}{dx} = \theta_1 + \frac{2x}{\ell}(-2\theta_1-\theta_2+3\theta_o) + \frac{3x^2}{\ell^2}(\theta_1+\theta_2-2\theta_o)$$

$$\frac{dw}{dx} = -\Psi_1 + \frac{2x}{\ell}(2\Psi_1+\Psi_2-3\Psi_o) + \frac{3x^2}{\ell^2}(-\Psi_1-\Psi_2+2\Psi_o) \qquad (2\text{-}8)$$

$$\frac{d^2v}{dx^2} = \frac{2}{\ell}(-2\theta_1-\theta_2+3\theta_o) + \frac{6x}{\ell^2}(\theta_1+\theta_2-2\theta_o)$$

$$\frac{d^2w}{dx^2} = \frac{2}{\ell}(2\Psi_1+\Psi_2-3\Psi_o) + \frac{6x}{\ell^2}(-\Psi_1-\Psi_2+2\Psi_o)$$

Using Equation (2-8), Equation (2-7) may be written as:

$$\varepsilon(x,\eta,\zeta) = a + \frac{1}{2}\left(b+\frac{c}{\ell}x+\frac{d}{\ell^2}x^2\right)^2 + \frac{1}{2}\left(e+\frac{f}{\ell}x+\frac{g}{\ell^2}x^2\right)^2 + \eta\left(\frac{c}{\ell}+\frac{2d}{\ell^2}x\right) + \zeta\left(\frac{f}{\ell}+\frac{2gx}{\ell^2}\right) \qquad (2\text{-}9)$$

in which:

$$a = \frac{u_2-u_1}{\ell}, \quad b = \theta_1, \quad c = 2\,(-2\theta_1-\theta_2+3\theta_o)$$

$$d = 3\,(\theta_1+\theta_2-2\theta_o), \quad e = -\Psi_1$$

$$f = 2\,(2\Psi_1+\Psi_2-3\Psi_o), \quad g = 3\,(-\Psi_1-\Psi_2+2\Psi_o)$$

The strain energy of the beam element due to normal strain is:

$$U_\varepsilon = \int_{vol.}\frac{1}{2}E[\varepsilon(x,\eta,\zeta)]^2 dVol. = \int_o^\ell \int_A \frac{1}{2}E[\varepsilon(x,\eta,\zeta)]^2 dAdx \qquad (2\text{-}10)$$

in which E is the modulus of elasticity and A is the cross sectional area.

By substituting Equation (2-9) into Equation (2-10) we have:

$$U_\varepsilon = \frac{1}{2}E\Big\{\int_o^\ell \int_A \left[a+\frac{1}{2}\left(b+\frac{c}{\ell}x+\frac{d}{\ell^2}x^2\right)^2+\frac{1}{2}\left(e+\frac{f}{\ell}x+\frac{g}{\ell^2}x^2\right)^2\right]^2 dAdx + \int_o^\ell \int_A\left[\eta\left(\frac{c}{\ell}+\frac{2d}{\ell^2}x\right)+\zeta\left(\frac{f}{\ell}+\frac{2g}{\ell^2}x\right)\right]^2 dAdx\Big\} \qquad (2\text{-}11)$$

The strain energy due to torsion $U_t$ may be written as:

$$U_t = \frac{1}{2}\int_0^{\ell} GJ \left(\frac{d\phi}{dx}\right)^2 dx \tag{2-12}$$

in which, G is the shear modulus, J is the torsion constant, and from Equation (2-3):

$$\frac{d\phi}{dx} = \frac{\phi_2-\phi_1}{\ell} \tag{2-13}$$

In the absence of initial strain the total strain energy is the sum of $U_\varepsilon$ and $U_t$:

$$U_{TOTAL}=\frac{1}{2}E\{A\int_0^{\ell}[a+\frac{1}{2}(b+\frac{c}{\ell}x+\frac{d}{\ell^2}x^2)^2+\frac{1}{2}(e+\frac{f}{\ell}x+\frac{g}{\ell^2}x^2)^2]^2 dx$$

$$+\int_0^{\ell}\int_A[\eta^2(\frac{c}{\ell}+\frac{2d}{\ell^2}x)^2+\zeta^2(\frac{f}{\ell}+\frac{2g}{\ell^2}x)^2]\,dAdx\}$$

$$+\frac{1}{2}\int_0^{\ell} GJ(\frac{\phi_2-\phi_1}{\ell})^2 dx \tag{2-14}$$

By integrating (2-14), the expression for the total strain energy for a three dimensional beam element is obtained as follows:

$$U_{TOTAL} = \frac{EA\ell}{2}\{a^2+ab^2+ae^2+\frac{1}{4}(b^4+e^4)+\frac{1}{2}b^2e^2+abc+aef$$

$$+\frac{1}{2}b^3c+\frac{1}{2}e^3f+\frac{1}{2}bce^2+\frac{1}{2}efb^2+\frac{1}{3}ac^2+\frac{2}{3}abd+\frac{1}{3}af^2+\frac{2}{3}aeg$$

$$+\frac{1}{5}b^2c^2+\frac{1}{3}b^3d+\frac{1}{2}e^2f^2+\frac{1}{3}e^3g+\frac{2}{3}bcef+\frac{1}{6}b^2f^2+\frac{1}{6}c^2e^2$$

$$+\frac{1}{3}egb^2+\frac{1}{3}bde^2+\frac{1}{2}acd+\frac{1}{2}afg+\frac{3}{4}b^2cd+\frac{1}{4}bc^3+\frac{3}{4}e^2fg+\frac{1}{4}ef^3$$

$$+\frac{1}{4}efc^2+\frac{1}{4}bcf^2+\frac{1}{4}fgb^2+\frac{1}{4}cde^2+\frac{1}{2}bceg+\frac{1}{2}bdef+\frac{1}{5}ad^2$$

$$+\frac{1}{5}ag^2+\frac{1}{20}c^4+\frac{3}{10}b^2d^2+\frac{3}{5}bc^2d+\frac{1}{20}f^4+\frac{3}{10}e^2g^2+\frac{3}{5}ef^2g+\frac{1}{10}g^2b^2$$

$$+\frac{1}{10}d^2e^2+\frac{1}{10}c^2f^2+\frac{1}{5}bdf^2+\frac{1}{5}egc^2+\frac{2}{5}bdeg+\frac{2}{5}bcfg+\frac{4}{5}cdef+\frac{1}{6}c^3d$$

$$+\frac{1}{2}bcd^2+\frac{1}{6}f^3g+\frac{1}{2}efg^2+\frac{1}{6}efd^2+\frac{1}{6}bcg^2+\frac{1}{6}egc^2+\frac{1}{3}fgbd$$

$$+\frac{1}{6}cdf^2+\frac{1}{6}cdeg+\frac{3}{14}c^2d^2+\frac{1}{7}bd^3+\frac{3}{14}f^2g^2+\frac{1}{7}eg^3+\frac{1}{14}c^2g^2+\frac{1}{7}bdg^2$$

$$+\frac{1}{14}d^2f^2+\frac{1}{7}egd^2+\frac{2}{7}cdfg+\frac{1}{8}cd^3+\frac{1}{8}fg^3+\frac{1}{8}cdg^2+\frac{1}{8}fgd^2$$

$$+\frac{1}{36}d^4+\frac{1}{36}g^4+\frac{1}{18}d^2g^2\} +$$

$$\frac{1}{2\ell}\,E\,(c^2+\frac{4}{3}d^2+2cd)\,I_\eta+(f^2+\frac{4}{3}g^2+2fg)\,I_\zeta\ +\frac{1}{2\ell}GJ(\phi_2-\phi_1)^2 \qquad (2\text{-}15)$$

in which $I_\eta = \int_A \eta^2\, dA$ and $I_\zeta = \int_A \zeta^2\, dA$ are the principal moments of inertia.

It is noted that the total strain energy is a quartic function of end displacements and rotations.

## 2.1.3 STIFFNESS MATRICES OF A THREE DIMENSIONAL BEAM ELEMENT

The total energy expression derived in the previous section may be divided into three parts, i.e.,

$$U_{TOTAL} = U_2 + U_3 + U_4$$

in which $U_2$ contains only quadratic terms (in terms of degrees of freedom of a, b, c, d, e, f, g), and similarly $U_3$ and $U_4$ contain cubic and quartic terms, respectively.

It is well-known that the stiffness matrices can be obtained from the strain energy expression as follows:

$$[k] = [(k)_{i,j}] = [\frac{\partial^2 U_2}{\partial q_i \; \partial q_j}]$$

$$[n_1] = [(n_1)_{i,j}] = [\frac{\partial^2 U_3}{\partial q_i \; \partial q_j}] \qquad (2\text{-}16)$$

$$[n_2] = [(n_2)_{i,j}] = [\frac{\partial^2 U_4}{\partial q_i \; \partial q_j}]$$

in which $q_i$, $q_j$ represent the generalized coordinates such as $u_1$, $v_1$, ..., etc. It should be noted that $[k]$ is the usual linear stiffness matrix, while $[n_1]$ and $[n_2]$ contain, respectively, linear and quadratic terms of the displacements.

The calculations of $[k]$, $[n_1]$, $[n_2]$ in Equation (2-16) are very lengthy, but straightforward. The intermediate computations are not presented here and expressions for each of the above matrices are given in Appendix A.

It is of interest to note that if the terms containing rotational displacements are dropped from $[n_1]$, i.e., only terms involving the relative axial displacement $(u_2-u_1)$ are kept, the resulting matrix is:

$$[n_1{}^*] = \frac{AE(u_2-u_1)}{\ell} [k_G] \qquad (2\text{-}17)$$

in which $[n_1{}^*]$ is the usual "geometric stiffness matrix"; $\frac{(u_2-u_1)}{\ell}$ has been interpreted to be the axial strain of the member, and

$$P = AE(\frac{u_2-u_1}{\ell})$$

is the axial load (Przemieniecki [12]). The matrix $[n_1{}^*]$ used in the eigenvalue problem considered here, is given in Appendix C.

## 2.1.4 STIFFNESS MATRICES OF A TWO DIMENSIONAL BEAM ELEMENT

Since we already have [k], $[n_1]$, $[n_2]$ for three dimensional problems, by eliminating the terms corresponding to a third dimension (e.g., $w_1$, $\phi_1$, $\Psi_1$, $w_2$, $\phi_2$, $\Psi_2$) the expressions for the two dimensional case (e.g., an element in x-y plane) can be obtained easily.

These expressions are shown in Appendix A. They check with those reported by Mallett and Marcal [14].

## 2.1.5 STIFFNESS MATRICES BASED ON "AVERAGE AXIAL STRAIN"

The preceding stiffness matrices were based on a quartic expression for the axial strain as given by Equations (2-6) and (2-7). An alternative to this expression is to use the average of the non-linear strains over the length [20]. In this case the expression for axial strain is written as:

$$\varepsilon_a = \frac{du}{dx} + \frac{1}{\ell}\int_0^{\ell} \frac{1}{2}\left(\frac{dv}{dx}\right)^2 dx + \frac{1}{\ell}\int_0^{\ell} \frac{1}{2}\left(\frac{dw}{dx}\right)^2 dx \qquad (2\text{-}18)$$

Therefore, using Equation (2-7) we obtain the strain at each point of a section as:

$$\begin{aligned}\varepsilon(x,\eta,\zeta) = \varepsilon_a + \eta\frac{d^2v}{dx^2} + \zeta\frac{d^2w}{dx^2} &= \frac{u_2-u_1}{\ell} \\ &+ \frac{1}{30}(2\theta_1^{\,2}+2\theta_2^{\,2}-\theta_1\theta_2-3\theta_1\theta_0-3\theta_2\theta_0+18\theta_0^{\,2}) \\ &+ \frac{1}{30}(2\Psi_1^{\,2}+2\Psi_2^{\,2}-\Psi_1\Psi_2-3\Psi_1\Psi_0-3\Psi_2\Psi_0+18\Psi_0^{\,2}) \\ &+ \eta\left[\frac{2}{\ell}(-2\theta_1-\theta_2+3\theta_0) + \frac{6x}{\ell^2}(\theta_1+\theta_2-2\theta_0)\right] \\ &+ \zeta\left[\frac{2}{\ell}(2\Psi_1+\Psi_2-3\Psi_0) + \frac{6x}{\ell^2}(-\Psi_1-\Psi_2+2\Psi_0)\right] \qquad (2\text{-}19)\end{aligned}$$

The strain energy in this case is given by:

$$U_{TOTAL} = U_{\varepsilon}+U_{t} = \frac{1}{2} E \int_{0}^{\ell} A \left[\frac{u_2-u_1}{\ell} + \frac{1}{30} \times \right.$$

$$(2\theta_1{}^2+2\theta_2{}^2-\theta_1\theta_2-3\theta_1\theta_0-3\theta_2\theta_0+18\theta_0{}^2)$$

$$\left. + \frac{1}{30} (2\Psi_1{}^2+2\Psi_2{}^2-\Psi_1\Psi_2-3\Psi_1\Psi_0-3\Psi_2\Psi_0+18\Psi_0{}^2\right]dx$$

$$+ \frac{E}{2} I_\eta \int_{0}^{\ell} \left[\frac{2}{\ell} (-2\theta_1-\theta_2+3\theta_0) + \frac{6x}{\ell^2} (\theta_1+\theta_2-2\theta_0)\right]^2 dx$$

$$+ \frac{E}{2} I_\zeta \int_{0}^{\ell} \left[\frac{2}{\ell} (2\Psi_1+\Psi_2-3\Psi_0) + \frac{6x}{\ell^2} (-\Psi_1-\Psi_2+2\Psi_0)\right]^2 dx$$

$$+ \frac{1}{2} GJ \int_{0}^{\ell} \left(\frac{d\zeta}{dx}\right)^2 dx \qquad (2\text{-}20)$$

By using exactly the same procedure as described in section 2.1.3, expressions for [k], [$n_1$], and [$n_2$] have been obtained. Since the expressions for [k] and [$n_1$] turn out to be the same as for the quartic cases only the [$n_2$] terms are shown in Appendix A. By appropriately deleting certain terms in the [$n_2$] matrix, its two dimensional version is obtained and is shown in Appendix A also.

## 2.1.6 GLOBAL EQUILIBRIUM EQUATIONS

In the preceding sections we have derived the stiffness matrices [k], [$n_1$], [$n_2$] for each element in local coordinates. If for each element we transform these matrices to global coordinates and assemble them in the usual fashion of the finite element method, the structural linear and nonlinear stiffness matrices [K], [$N_1$] and [$N_2$], are obtained.

For an elastic and conservative system, the potential energy is:

$$\Phi_P = U_{TOTAL} + V \qquad (2\text{-}21)$$

in which $U_{TOTAL}$ is the total strain energy of the structure and V is the potential of the external loads. Denoting by $\{Q\}$ and $\{P\}$ the generalized displacement vector and the corresponding external load vector, we may write (see Mallett and Marcal [14]).

$$U_{TOTAL} = \lfloor Q \rfloor \left[\frac{1}{2}[K] + \frac{1}{6}[N_1] + \frac{1}{12}[N_2]\right]\{Q\} \tag{2-22}$$

$$V = -\lfloor Q \rfloor \{P\} \tag{2-23}$$

and,

$$\Phi_P = \lfloor Q \rfloor \left[\frac{1}{2}[K] + \frac{1}{6}[N_1] + \frac{1}{12}[N_2]\right]\{Q\} - \lfloor Q \rfloor \{P\} \tag{2-24}$$

The first variation of the potential energy gives the total equilibrium equation, [14]

$$[S_S]\{Q\} = \{P\} \tag{2-25}$$

in which $[S_S]$ is the secant stiffness matrix, i.e.,

$$[S_S] = [K] + \frac{1}{2}[N_1] + \frac{1}{3}[N_2] \tag{2-26}$$

The second variation of potential energy gives the incremental equilibrium equation, [14]

$$[S_T]\{\Delta Q\} = \{\Delta p\} \tag{2-27}$$

in which $\{\Delta Q\}$ and $\{\Delta p\}$ denote the incremental displacement and load vectors, respectively. $[S_T]$ is the tangent stiffness, given by

$$[S_T] = ([K] + [N_1] + [N_2])_{\{\bar{Q}\}} \tag{2-28}$$

so: $$([K] + [N_1] + [N_2])_{\{\bar{Q}\}}\{\Delta Q\} = \{\Delta p\} \tag{2-29}$$

in which $\{\bar{Q}\}$ denotes the displacement vector at which the incremental vector $\{\Delta Q\}$ is to be measured.

Equation (2-29) may be used to formulate the eigenvalue problem for buckling analysis. Both Equations (2-25) and (2-29) will be used for studies of geometrically nonlinear behavior.

## 2.2 INITIAL STRAIN STIFFNESS MATRIX

### 2.2.1 GENERAL

The stiffness matrices derived in the preceding sections were based on the assumption of no initial strain in the structural system; that is, the total strain energy depends only on the displacements.

As will be shown in the next chapter, for some methods of solution, it is necessary to consider the strain energy with reference to a deformed state, i.e., a structure with initial strain. This initial strain would result in an "initial strain stiffness matrix" in the analysis.

Let us use a two dimensional beam element as shown in Figure 2-3. The X and Y axes represent the global coordinate system, the $x_i$, and $y_i$ axes denote the member coordinates and $C_i$ the member configuration at the beginning of the ith load level.

The current strain energy $U_{TOTAL}$ during the ith load increment is formed of two parts:

a) $\overset{i}{U}$ the strain energy at the beginning of this increment.

b) $\overset{i\to i+1}{U}$ the strain energy due to change of the geometry with reference to configuration $C_i$

so:

$$U_{TOTAL} = \overset{i}{U} + \overset{i\to i+1}{U} \qquad (2\text{-}30)$$

Since in Equation (2-30) ${}^{i}U$ is independent of the generalized coordinates it does not enter in the derivation of stiffness matrices of the system.

$\overset{i\to i+1}{U}$ may be written as:

$$\overset{i\to i+1}{U} = \int_{Vol} (\frac{1}{2} E\, \varepsilon^2 + E\, \varepsilon\, {}^{i}\varepsilon_o)\, dVol = U_\varepsilon + {}^{i}U_{\varepsilon_o} \tag{2-31}$$

in which

$$U_\varepsilon = \int_{Vol} \frac{1}{2} E\, \varepsilon^2\, dVol \tag{2-32}$$

$${}^{i}U_{\varepsilon_o} = \int_{Vol} E\, \varepsilon\, {}^{i}\varepsilon_o\, dVol \tag{2-33}$$

$\varepsilon$ is the current strain and ${}^{i}\varepsilon_o$ is the physical initial strain at the beginning of the ith configuration. It should be noticed that both $\varepsilon$ and ${}^{i}\varepsilon_o$ are measured with reference to the chord configuration and not to the deformed beam element.

As in the previous sections, taking the derivatives of $U_\varepsilon$ in Equation (2-32) with respect to the generalized coordinates we can obtain the usual tangent stiffness matrix (see Equation (2-28)).

In the same manner the initial strain stiffness matrix could be derived from (2-33) if we substitute the expression for $\varepsilon$ in terms of generalized coordinates.

In this section three initial strain stiffness matrices are derived. The first two follow directly from the "quartic" and "average" axial strain assumptions. They are used in the updated Lagrange method of solution. A third one which corresponds to the usual geometric stiffness matrix is used in the straight incremental method of solution.

## 2.2.2 INITIAL STRAIN STIFFNESS MATRIX BASED ON QUARTIC AXIAL STRAIN ASSUMPTION

In this case the contribution of each increment to the initial strain is a quartic function of x, as in Equation (2·7). At the beginning of the ith increment:

$$ {}^{i}\varepsilon_o\,(x,\zeta,\eta) = \sum_{j=1}^{i-1} {}^{j}\left[\frac{du}{dx}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2+\frac{1}{2}\left(\frac{dw}{dx}\right)^2+\eta\frac{d^2v}{dx^2}+\zeta\frac{d^2w}{dx^2}\right] \tag{2-34}$$

in which j denotes the stage of the configuration.

Since the axial strain evaluated at the end of the jth configuration is regarded as a scaler physical quantity, the effect of successive increments has been added for j = 1 to j = i - 1.

By substituting Equations (2-34) and (2-7) in Equation (2-33) we have:

$$ {}^{i}U_{\varepsilon_o} = E\int_{Vol}\left\{\sum_{j=1}^{i-1} {}^{j}\left[\frac{du}{dx}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2+\frac{1}{2}\left(\frac{dw}{dx}\right)^2+\eta\frac{d^2v}{dx^2}+\zeta\frac{d^2w}{dx^2}\right]\right\}\times$$

$$\left[\frac{du}{dx}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2+\frac{1}{2}\left(\frac{dw}{dx}\right)^2+\eta\frac{d^2v}{dx^2}+\zeta\frac{d^2w}{dx^2}\right] d\,Vol \tag{2-35}$$

$$ {}^{i}[k_{\varepsilon_o}] = [({}^{i}k_{\varepsilon_o})_{m,n}] = \left[\frac{\partial^2\, {}^{i}U_{\varepsilon_o}}{\partial q_m\,\partial q_n}\right] \tag{2-36}$$

The intermediate computations are not shown here. The major steps and final expressions for $[k_{\varepsilon_o}]$ are given in Appendix B for both the three and two dimensional cases.

## 2.2.3 INITIAL STRAIN STIFFNESS MATRIX BASED ON THE AVERAGE STRAIN ASSUMPTION

In this case the contribution of each load increment to the initial strain is based on the previously-mentioned average strain assumption. Thus at the beginning of the ith increment,

$$ {}^{i}\varepsilon_o\ (x,\zeta,\eta) = \sum_{j=1}^{i-1} {}^{j}[\frac{u_2-u_1}{\ell}+\frac{1}{2\ell}\int_o^\ell(\frac{dv}{dx})^2 dx+\frac{1}{2\ell}\int_o^\ell(\frac{dw}{dx})^2 dx $$

$$ + \eta\,\frac{d^2v}{dx^2} + \zeta\,\frac{d^2w}{dx^2}] \qquad (2\text{-}37) $$

Since the right hand side of Equation (2-37) is independent of x by using Equations (2-37) and (2-7) in Equation (2-33) we have:

$$ {}^{i}U_{\varepsilon_o} = EA\left\{\sum_{d=1}^{i-1} {}^{j}[\frac{u_2-u_1}{\ell}+\frac{1}{2\ell}\int_o^\ell(\frac{dv}{dx})^2 dx+\frac{1}{2\ell}\int_o^\ell(\frac{dw}{dx})^2 dx]+\eta\frac{d^2v}{dx^2}+\right. $$

$$ \left.\zeta\frac{d^2w}{dx^2}\right\}\int_o^\ell [\frac{du}{dx}+\frac{1}{2}(\frac{dv}{dx})^2+\frac{1}{2}(\frac{dw}{dx})^2 + \eta\,\frac{d^2v}{dx^2} + \zeta\,\frac{d^2w}{dx^2}]\,dx \qquad (2\text{-}38) $$

in which ${}^{i}U_{\varepsilon_o}$ finally can be shown as a function of generalized coordinates.

Similar to the previous case, by using Equation (2-36) we have:

$$ {}^{i}[k_{\varepsilon_o}] = {}^{i}P\ [k_G] \qquad (2\text{-}39) $$

in which

$$ {}^{i}P = EA\sum_{j=1}^{i-1} {}^{j}[\frac{u_2-u_1}{\ell}+\frac{1}{2}(\frac{v_2-v_1}{\ell})^2+\frac{1}{2}(\frac{w_2-w_1}{\ell})^2 $$

$$ + \frac{\ell}{30}(2\theta_1^2-\theta_1\theta_2+2\theta_2^2) + \frac{\ell}{30}(2\Psi_1^2-\Psi_1\Psi_2+2\Psi_2^2)] \qquad (2\text{-}40) $$

for three dimensional and

$$ {}^{i}P = EA \sum_{j=1}^{i-1} {}^{j}\left[\frac{u_2-u_1}{\ell} + \frac{1}{2}\left(\frac{v_2-v_1}{\ell}\right)^2 + \frac{\ell}{30}(2\theta_1{}^2-\theta_1\theta_2+2\theta_2{}^2)\right] \qquad (2\text{-}41) $$

for two dimensional beam elements, and $[k_G]$ is shown in Appendix C.

## 2.2.4 INITIAL STRAIN STIFFNESS MATRIX BASED ON LONGITUDINAL DISPLACEMENTS ONLY

The geometric stiffness matrix that appears in the literature cited previously (Martin [11] and Przemieniecki [12]) and given in Appendix C may be regarded as an initial strain stiffness matrix and derived as follows.

In this case we take:

$$ {}^{i}\varepsilon_o\,(x,\eta,\zeta) = \sum_{j=1}^{i-1} {}^{j}\left[\frac{u_2-u_1}{\ell}\right] \qquad (2\text{-}42) $$

Using Equation (2-42) and (2-7) in Equation (2-33) we have:

$$ {}^{i}U_{\varepsilon_o} = EA\left\{\sum_{j=1}^{i-1} {}^{j}\left[\frac{u_2-u_1}{\ell}\right]\right\}\int_o^{\ell}\left[\frac{du}{dx}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2+\frac{1}{2}\left(\frac{dw}{dx}\right)^2\right] dx \qquad (2\text{-}43) $$

By following a similar procedure:

$$ {}^{i}[k_{\varepsilon_o}] = {}^{i}P\,[k_G] \qquad (2\text{-}44) $$

in which

$$ {}^{i}P = EA \sum_{j=1}^{i-1} {}^{j}\left[\frac{u_2-u_1}{\ell}\right] \qquad (2\text{-}45) $$

## 2.3 EIGENVALUE PROBLEMS FOR BUCKLING LOAD ANALYSIS

In Section 2.1.6 we introduced the linear incremental equilibrium equation (Equation (2-29)). In the following section we are going to use these equations to formulate certain eigenvalue problems for the calculation of buckling loads.

One usual way to evaluate the critical load of a structure is to set the incremental load vector $\{P\}$ to null in Equation (2-29). This leads us to the following equation.

$$([K] + [N_1] + [N_2])_{\{\bar{Q}\}} \{\Delta Q\} = \{0\} \tag{2-46}$$

For a buckling load analysis we look for a point $(\{\bar{P}\}, \{\bar{Q}\})$ on the load displacement curve (Figure 2-4) which satisfies the above Equation (2-46). That $\{\bar{P}\}$ would be the buckling or critical load.

The exact solution of (2-46) in general is complicated because of its nonlinear nature. But if we assume that the displacement of the structure is a linear function of applied loads just up to the point at which buckling occurs, then we have:

$$\{P_{ref}\} = [K] \{Q_{ref}\} \tag{2-47}$$

and

$$\{Q_{ref}\} = [K]^{-1} \{P_{ref}\} \tag{2-48}$$

In (2-47) $\{P_{ref}\}$ is an arbitrary reference load vector. Since $[N_1]$ and $[N_2]$ are linear and quadratic functions of displacements, with $\{\bar{P}\} = \lambda \{P_{ref}\}$:

$$[N_1(\{\bar{Q}\})] = [N_1(\{Q_{ref}\})]\lambda \qquad (2\text{-}49)$$

and

$$[N_2(\{\bar{Q}\})] = [N_2(\{Q_{ref}\})]\lambda^2 \qquad (2\text{-}50)$$

in which $\lambda$ is a parameter.

Since Equations (2-49) and (2-50) are supposed to be valid until buckling, we have

$$[N_1(\{\bar{Q}\})] = [N_1(\{Q_{ref}\})]\ \lambda_{cr}$$

$$[N_2(\{\bar{Q}\})] = [N_2(\{Q_{ref}\})]\ \lambda_{cr}^2$$

Thus Equation (2-46) can be written as:

$$([K] + \lambda_{cr}\ [N_1] + \lambda_{cr}^2\ [N_2])_{\{Q_{ref}\}}\ \{\Delta Q\} = \{0\} \qquad (2\text{-}51)$$

Equation (2-51) is a quadratic eigenvalue equation. For sufficiently small displacements, matrix $[N_2]$ may be neglected and Equation (2-51) reduces to a linear eigenvalue equation:

$$([K] + \lambda_{cr}\ [N_1])_{\{Q_{ref}\}}\ \{\Delta Q\} = \{0\} \qquad (2\text{-}52)$$

Solution of Equation (2-51) or Equation (2-52) would yield $\lambda_{cr}$ and, of course, the critical load vector is $\lambda_{cr}\ \{P_{ref}\}$.

# CHAPTER III

# METHODS OF SOLUTION

## 3.1 GENERAL

As mentioned previously, a method of analysis for the non-linear elastic behavior of framed structures may be regarded as consisting of three parts: (i) model, (ii) local coordinates, and (iii) method of solution. In the preceding chapter several finite element models have been formulated in Lagrange coordinates. In this chapter, the methods of solution that will be applied for the solution of these models are described.

## 3.2 NEWTON-RAPHSON METHOD

### 3.2.1 CONCEPT

Consider a structure subjected to a predefined external load vector $\{P\}$. Let $Q$ be symbolically the so called exact deformed configuration of the structure. If we assume an iterative process, and in the ith iteration the approximate configuration $Q_i$ is known, we are interested in improving $Q_i$ in such a way that it would get sufficiently close to $Q$.

We write the load displacement relation as:

$$\{P\} = \{f(Q)\} \tag{3-1}$$

using a first order Taylor series expansion about $Q_i$ we have:

$$\{P\} = \{f(Q_i)\} + \{\frac{\partial f}{\partial Q_j}\}_{Q_i} \{\Delta Q\}_i$$

in which, $\{f(Q_i)\}$ may be interpreted as representing the elastic resistance of the structure corresponding to $Q_i$, and $\{\frac{\partial f}{\partial Q_j}\}_{Q_i}$ as the tangent stiffness at $Q_i$. Then the modification to $Q_i$ is:

$$\{\Delta Q_i\} = \{\frac{\partial f}{\partial Q_j}\}_{Q_i}^{-1} \{P-f(Q_i)\} = \{\frac{\partial f}{\partial Q_j}\}_{Q_i}^{-1} \{\Delta R_i\}$$

in which, $\{\Delta R_i\}$ is the "unbalanced force vector" at stage $Q_i$.

The modified displacement is:

$$Q_{i+1} = Q_i + \Delta Q_i$$

The process may be repeated until either $\Delta Q_{i+k}$ or $\Delta R_{i+k}$ is sufficiently small. This process is graphically illustrated in Figure 3-1 for a one degree of freedom system.

The preceding discussion was for the load applied as a single load increment. For many problems greater accuracy in the solution may be obtained by applying the load in increments (i.e., $\Delta P$, $2\Delta P$, ..., etc.). For each increment the concept described previously applies, provided the stress state of the structure at the beginning of load increment is properly taken into account.

At the beginning of the increment the geometry of structure may or may not be updated. Both cases are considered in the following sections.

## 3.2.2 NEWTON-RAPHSON METHODS FOR FIXED COORDINATES

In this case the geometry of the structure is not updated. The steps of the calculation are as follows:

1) Set load increment (and check if the intended total load has been applied).

2) Form the structural tangent stiffness matrix as:

$$\text{Tangent stiffness matrix} = [K] + [N_1(\{Q\})] + [N_2(\{Q\})]$$

3) Solve for $\{\Delta Q\}$ from:

$$\{\Delta Q\} = [\text{tangent stiffness matrix}]^{-1}\{\text{load increment vector}\}$$

4) Add $\{\Delta Q\}$ to the latest $\{Q\}$ to obtain a new $\{Q\}$

5) If convergence check is based on displacement and $\{\Delta Q\}$ is sufficiently small, return to 1.

6) Based on the new $\{Q\}$ from step 4 evaluate $N_1(\{Q\})$ and $N_2(\{Q\})$.

7) Form the tangent and secant stiffness matrices and resistance force vector as:

$$\text{Tangent stiffness matrix} = [K] + [N_1(\{Q\})] + [N_2(\{Q\})]$$

$$\text{Secant stiffness matrix} = [K] + \frac{1}{2}[N_1(\{Q\})] + \frac{1}{3}[N_2(\{Q\})]$$

$$\text{Resistance force vector} = [\text{Secant stiffness matrix}] \times \{Q\}$$

8) Evaluate the unbalanced force vector as:

$$\text{Unbalanced force vector} = \text{Increment load vector} - \text{Resistance force vector.}$$

9) If convergence check is based on unbalanced force vector and it is sufficiently small, return to 1.

10) Return to 2 but use the unbalanced force vector for the load increment vector.

## 3.2.3 NEWTON-RAPHSON METHOD FOR UPDATED COORDINATES

This procedure is to be used to implement the theory as discussed in Section 3.2.1. The loads are applied in increments. At the end of each increment the geometry of structure is updated. In addition to the usual stiffness matrices $[k]$, $[n_1]$, $[n_2]$ there is the initial strain matrix (resulting from initial strain energy) as explained previously.

The steps of calculation are as follows:

1) Set load increment (and check if the intended total load has been applied).

2) Determine the most up-to-date geometry of the structure by using the latest joint displacements, and update the linear stiffness matrix.

3) Form the tangent stiffness matrix according to one of the following cases:

   a) For the first load increment:

      Tangent stiffness matrix = $[K] + [N_1(\{Q\})] + [N_2(\{Q\})]$

   b) For other load increments:

      Tangent stiffness matrix = $[K] + [K_{\varepsilon_o}] + [N_1(\{Q\})]$
      $+ [N_2(\{Q\})]$

      in which $[K_{\varepsilon_o}]$ is the initial strain stiffness matrix.

4) Solve for $\{\Delta Q\}$ from:

   $\{\Delta Q\}$ = [tangent stiffness matrix]$^{-1}$\{load increment vector\}

5) If convergence check is based on displacement and $\{\Delta Q\}$ is sufficiently small, return to 1.

6) Add $\{\Delta Q\}$ to the latest $\{Q\}$ to obtain a new $\{Q\}$.

7) Based on the new $\{Q\}$ evaluate $[N_1(\{Q\})]$ and $[N_2(\{Q\})]$.

8) Form tangent and secant stiffness matrices and resistance force vector as:

   a) For the first load increment:

      Tangent stiffness matrix = $[K] + [N_1(\{Q\})] + [N_2(\{Q\})]$

      Secant stiffness matrix = $[K] + \frac{1}{2}[N_1(\{Q\})] + \frac{1}{3} \times [N_2(\{Q\})]$

   b) For other load increments:

      Tangent stiffness matrix = $[K] + [K_{\varepsilon_o}] + [N_1(\{Q\})] + [N_2(\{Q\})]$

      Secant stiffness matrix = $[K] + [K_{\varepsilon_o}] + \frac{1}{2}[N_1(\{Q\})] + \frac{1}{3}[N_2(\{Q\})]$

      Resistance force vector = $[\text{Secant stiffness matrix}] * \{Q\}$

9) Evaluate unbalanced force vector from:

   Unbalanced force vector = Incremental load vector - Resistance force vector

10) If convergence check is based on unbalanced force vector and it is sufficiently small, return to 1.

11) Return to 4 but use the unbalanced force vector as the load increment vector.

## 3.2.4 CONVERGENCE CRITERIA

### 3.2.4.1 GENERAL

In implementing the above Newton-Raphson method a convergence criterion is needed. In this report, two convergence criteria have been used. The first one is based on the unbalanced force vector and the second one is based on the incremental displacement vector.

### 3.2.4.2 CONVERGENCE CHECK BASED ON UNBALANCED FORCE VECTOR

In this type of convergence check, a reasonable tolerance (which has the unit of force or moment) is prescribed first for each group of components (i.e., force or moment) of the unbalanced force vector.

After the evaluation of the unbalanced force vector in each iteration the absolute value of each component of the vector is independently compared with the prescribed tolerance. Convergence is considered achieved if, for each of the components, this absolute value is less than or equal to the tolerance.

The feature of this convergence criterion is that it represents a real test of the equilibrium of the structure and it is an absolute check. The tolerance for this convergence criterion is denoted by $\varepsilon_f$ times unit force or unit moment.

### 3.2.4.3 CONVERGENCE CHECK BASED ON INCREMENTAL DISPLACEMENT VECTOR

In the displacement convergence check used herein, for each group of displacement components (i.e., translations or rotations) a reasonable tolerance ratio is defined. If we denote the incremental displacement vector by $\{\Delta x\}$ and the total displacement vector by $\{x\}$, convergence is considered achieved if for both groups the following is simultaneously satisfied.

$$\left[\frac{\sum_i (\Delta x_i)^2}{\sum_i (x_i)^2}\right]^{\frac{1}{2}} \leqslant \text{Tolerance ratio} = \varepsilon_d$$

in which i varies from 1 to the number of translation or rotation components of the displacement vector, and $\varepsilon_d$ is the tolerance ratio.

It should be noted that this convergence criterion does not directly deal with equilibrium of the structure. Furthermore, its absolute tolerance would decrease as the total displacement increases.

A comparison of the use of the two convergence criteria will be presented in Chapter IV on numerical results.

## 3.3 "ONE-STEP" NEWTON-RAPHSON METHOD

This approach in general is the same as what was described in Section 3.2. The only difference is that we do not iterate more than once for each load increment. Thus there is no convergence check.

Obviously the advantage of this approach, when compared to the Newton-Raphson method presented previously,is that it takes less computation. It should be noted that whenever this method or the straight incremental method (as described in next section) is used for beam-column models, the iteration process on the axial load of each element should be continued until convergence is satisfied.

## 3.4 "STRAIGHT INCREMENTAL" METHOD

This approach is the same as the One-Step Newton-Raphson method except that not even one iteration would be used. Hence, there is no need to evaluate the secant stiffness matrix, resistance and unbalanced force vectors. Obviously the accuracy of this method would depend on the size of the load increment more than the previously mentioned methods.

## 3.5 SOLUTION OF EIGENVALUE PROBLEMS

### 3.5.1 LINEAR EIGENVALUE PROBLEM

In this report the inverse vector iteration technique as described by Bathe and Wilson [21] is used for solutions of the linear eigenvalue problems.The technique may be regarded as a mathematical formulation of the Stodola method [22] in structural mechanics.

The basic equation (2-52) could be written as:

$$Aq = \lambda Bq \tag{3-2}$$

in which for simplicity symbols ([ ] and { }) have been dropped and $A = [K]$, $B = -[N_1]$. It is assumed that A is positive definite and B may be a diagonal matrix with or without zero diagonal terms.

The technique used for computer implementation is as follows:

(a) Start with a trial vector $X_1$ for the first eigenvector $q$, $(X_1^T B q_1 \neq 0.)$

(b) For i=1, 2, ..., etc. evaluate

$$A\bar{X}_{i+1} = y_i$$

$$\bar{y}_{i+1} = B\bar{X}_{i+1}$$

$$\rho(\bar{X}_{i+1}) = \frac{\bar{X}_{i+1}^T y_i}{\bar{X}_{i+1}^T \bar{y}_{i+1}} \tag{3-3}$$

$$y_{i+1} = \frac{\bar{y}_{i+1}}{(\bar{X}^T_{i+1}\ \bar{y}_{i+1})^{\frac{1}{2}}}$$

in which $\rho$ is the Rayleigh quotient

(c) The iterative process is considered to have converged if:

$$\frac{\rho(\bar{X}_{i+1})-\rho(\bar{X}_i)}{\rho(\bar{X}_{i+1})} \leqslant \varepsilon psi \qquad (3\text{-}4)$$

$\varepsilon psi$ in Equation (3-4) should be less than or equal to $10^{-2S}$ if the answer is required to be accurate up to 2S digits. If Equation (3-4) is satisfied for i=n the smallest eigenvalue will be taken to be:

$$\lambda_1 = \rho(\bar{X}_{n+1}) \qquad (3\text{-}5)$$

and the corresponding eigenvector is:

$$q_n = \frac{\bar{X}_{n+1}}{(\bar{X}^T_{n+1}\ \bar{y}_{n+1})^{\frac{1}{2}}} \qquad (3\text{-}6)$$

The computer implementation of this technique (Ref. [23]) is contained in the subroutine EIGENVL listed in Appendix D.

## 3.5.2 QUADRATIC EIGENVALUE PROBLEM

Using Equation (2-51) the solution for a quadratic eigenvalue equation may be obtained by finding $\lambda$ for which:

$$\det \left| [K] + \lambda [N_1] + \lambda^2 [N_2] \right|_{\{Q_{ref}\}} = 0 \qquad (3\text{-}7)$$

Since we are looking for the lowest buckling mode the smallest value of

$\lambda$ is required.

The solution is carried out by evaluating the left-hand side of equation (3-7) using increasing values of $\lambda$, starting from zero with small increments as shown in Figure 3-2. If for $\lambda_A$, det $(\lambda_A) > 0$, but for $\lambda_B = \lambda_A + \Delta\lambda$, det $(\lambda_B) < 0$, then the solution $\lambda = \bar{\lambda}$ lies in the interval $[\lambda_A, \lambda_B]$.

A modified Regula-Falsi iteration technique is used to obtain a closer estimate of the root $\bar{\lambda}$ and the computer implementation of the quadratic eigenvalue solution [23] is given in subroutine NLEIGNP of the computer program in Appendix D.

## 3.6 COMPUTER PROGRAMS

### 3.6.1 GENERAL

In this section a general description of the programs developed for this study is presented. For the Lagrangian coordinate formulations two versions (for three and two dimensional problems) have been prepared. For Eulerian coordinate formulation only the two dimensional problem has been programmed for solution. A complete listing of the programs is given in Appendix D.

### 3.6.2 PROGRAMS FOR PROBLEMS IN LAGRANGIAN FORMULATION

#### 3.6.2.1 PROGRAM NFRAL3D

The program solves three dimensional problems formulated in Lagrangian coordinates, discussed in Chapter II, by using the various methods of solution as presented in Chapter III.

In addition to the usual required data input such as the physical properties of the system, the input should include the following:

(a) Coordinates used: Fixed-Lagrange or updated-Lagrange.

(b) Problem type specification (i.e., eigenvalue or incremental load-displacement).

(c) If (b) is eigenvalue problem, specify either linear or quadratic.

(d) If (c) is linear, specify whether $[N_1]$ or $[N_1{}^*]$ is to be used.

(e) If (b) is incremental load-displacement problem:

1) Type of solution, either Newton-Raphson or "straight incremental". (The successive substitution method of solution can also be handled by the program, but it was not used in this report.);

2) Maximum number of iterations;

3) Type of convergence check and tolerance;

4) Parameters which specify whether both $[N_1]$ and $[N_2]$ are to be used, or $[N_1]$ only, or neither of them in the solution method using updated coordinates;

5) If $[N_2]$ is to be included, specify whether it is based on the average strain or quartic strain formulation.

In the program, the linear stiffness for each element is computed, transformed into structural coordinates, and assembled into the linear structural stiffness matrix. A linear analysis of the structure is performed to obtain the displacements.

The structural displacement vector is transformed back into element end displacements. Now for each element $[n_1]$, $[n_2]$ and $[k_{\varepsilon_o}]$, (depending on the type of solution), are computed if needed and the matrices $[N_1]$, $[N_2]$ and $[K_{\varepsilon_o}]$ for the structure are assembled. All of the structural stiffness matrices have been assembled in banded format. Due to symmetry only the upper semi-band is computed.

#### 3.6.2.2 PROGRAM NFRAL2D

The general features of this program are similar to program NFRAL3D except that it is specifically prepared for two dimensional problems, and hence more efficient than using NFRAL3D for those problems. The options available to NFRAL3D are also available in this program except for linear eigenvalue solutions.

### 3.6.3 PROGRAM NFRAE2D FOR TWO DIMENSIONAL PROBLEMS IN EULERIAN COORDINATES

In this program three different models have been used. The first one is that of the beam-column continuum. It has been studied by Oran and Kassimali [10], among others. The second and third are finite element models that have been developed by Powell [15] and Jennings [13] respectively. No eigenvalue problem has been formulated for these models.

It should be noted that some of the subroutines which have been used in these programs are the same. However, since we wish each program to be self-contained the same subroutine is repeated as often as necessary in each program in Appendix D.

# CHAPTER IV

# NUMERICAL RESULTS

## 4.1 GENERAL

In this chapter we are going to consider a number of numerical problems of nonlinear load-displacement behavior and buckling (eigen-value problems) for both two and three dimensional cases. For the first group we divide the problems into "Large," "Small" and "Inter-mediate" displacement categories. This is a relative classification. What we mean by a "Large Displacement" problem is the case in which the deflection is of the order of the length of the member. By "Small Displacement" we mean it is less than about 2% of the member length. "Intermediate Displacement" lies in between.

For eigenvalue problems, two types of loading (symmetric and asymmetric) will be used for different arches and frames.

## 4.2 NONLINEAR LOAD-DISPLACEMENT BEHAVIOR

### 4.2.1 LARGE DISPLACEMENT PROBLEMS

#### 4.2.1.1 CANTILEVER BEAM WITH TWO LATERAL LOADS

The geometry, physical properties and loading for this problem are shown in Figure 4-1. This problem was chosen because it had been solved by other investigators using many of the different methods dis-cussed previously [19,10].

This system is also used to consider the effect of the step size (load increment) on convergence criterion and to illustrate the limitation of the one-step Newton-Raphson method of solution (1-step-NR).

4.2.1.1.1 COMPARISON OF RESULTS FROM DIFFERENT SOLUTIONS

The displacements under the loads as computed by the various methods are listed in Table 4-1. Row 1 gives the elastica solution (Frisch-Fay [24]) and is taken to be the exact solution. Rows 2 and 3 list results of the beam-column (continuum) theory using, respectively, the Newton-Raphson (beam-col-NR) and the straight incremental(beam-col-Inc) method of solution [10]. Rows 4 and 5 are, respectively, results of Jennings' formulation [13] using the Newton-Raphson and the straight incremental methods (Jennings'-NR or Inc). The numerical results in these two rows were taken from Ebner and Ucciferro's report [19].

For the incremental solutions the results obtained by use of the program written for the Jenning's formulation for this study (Program NFRE2D) are given in parentheses. It is seen that the latter results are much closer to the elastica solution.

Rows 6 and 7 correspond, respectively, to the Newton-Raphson and straight incremental method of solution using Powell's (Powell's-NR or Inc) formulation [15]. Row 8 corresponds to Martin's method [11]. In Row 9 is given solution corresponding to Mallett and Marcal's [14] model. The results based on the same model but using the updated Lagrange coordinate formulation are contained in Row 10. Finally in Rows 11 and 12 are listed solutions using the FEA model for the fixed and updated formulations.

The number of elements and number of increments of loading used are listed in columns 2 and 3 of the table. A consistent convergence criterion has been used for all the Newton-Raphson methods.

From a comparison of the results in Table 4-1, it is reasonable to rank five methods that produce sufficiently accurate results for this large deflection problem in the following order: (1) Beam-col-NR; (2) Beam-col-Inc; (3) Jennings'-NR; (4) FEA-updated; and (5) Martin's method.

The results obtained from the straight incremental solution of Jennings' model are not as good as those given by the five methods. However, when larger number of elements and increments were used, the results may be regarded as acceptable. The results given by all the other methods are so much off the mark that they are unacceptable.

Out of the preceding five accurate methods, the first three (the beam-column and Jennings' formulations) have used Eulerian coordinates which require a geometric transformation in every iteration. This requirement made them less efficient than the last two formulations, i.e., the FEA-updated method and Martin's method. A further comparison of these two will be given in the next section.

#### 4.2.1.1.2 COMPARISON OF MARTIN'S METHOD AND FEA-UPDATED METHODS

For the same problem considered above in Figure 4-1 are plotted the load-displacement curves obtained by Martin's method and FEA-updated method. Also shown is a curve obtained by the beam-col-NR method. In the comparison below, the beam-col-NR solution with a suitable number of elements and convergence criterion would be regarded as the "exact" one. This is because the elastica solution is not conveniently obtainable, and for the range of behavior considered herein, the beam-column theory results have been shown to be very close to the elastica solution [19,10].

The beam-col-NR Curve $C_1$ in the figure shows a pattern of "zig-zag" shape in the middle portion. This is because of the relatively large tolerance used in the convergence check. For a smaller tolerence, as we will see later, smoother curves would be obtained. In any case, the last point of $C_1$ agrees closely with the elastica solution.

A comparison of Curves $C_2$ and $C_3$ would indicate that for the same accuracy the FEA-updated method used five steps (load increments) while Martin's method used twenty steps. The number of iterations per step in the FEA-updated method being about three, the total number of iterations for the method was 15.

Although geometry updating is required only once in a load step the FEA-updated method involves more computation per iteration because of the use of the incremental stiffness matrices. Therefore, on the whole, the two methods appear competitive in efficiency. The FEA-updated method, however, does have a check on equilibrium which Martin's method lacks.

The preceding discussion also applies to Curves $C_4$ and $C_5$.

### 4.2.1.1.3 <u>CONVERGENCE CRITERION</u>

To consider the effect of the convergence criterion on the accuracy of solution the cantilever beam of the preceding problem was used. In Figure 4-2 Curve $C_1$, as before, is regarded as the "exact" result. It is interesting to note from Curve $C_2$ that, with the number of elements doubled but the same tolerance used in the solution, the results obtained deteriorated from $C_1$.

Now if for four elements we use a smaller tolerance, $\varepsilon_d = .001$, very good results are obtained, Curve $C_3$. Thus it seems that when using a convergence check on displacement, increasing the number of elements could be detrimental if a sufficiently small tolerance is not used. Curve $C_6$, which was obtained with six elements and $\varepsilon_d = .01$, again shows the same pattern as Curve $C_2$.

For Curves $C_4$ and $C_5$, the FEA-updated approach with unbalanced force tolerance $\varepsilon_f$ = .01 has been used. It is seen that in this case both solutions are very reasonable. Thus it would appear that greater caution is required when the displacement convergence criterion is used.

It is also worth noting that the curves corresponding to solutions using an unbalanced force check were much smoother than the others.

### 4.2.1.1.4 COMPARISON OF ONE-STEP NEWTON-RAPHSON METHOD WITH STRAIGHT INCREMENTAL METHOD OF SOLUTION

It has been pointed out in the literature [4] that the one-step Newton-Raphson method of solution could be a very effective one in the sense that by using a single iteration the results would be improved materially over those obtained by the straight incremental method.

For a comparison, in Figure 4-3 Curves $C_1$, $C_2$ and $C_3$ represent the beam-col-NR (considered "exact" for comparison), beam-col-1 step (beam-column one step Newton-Raphson) and beam-col-Inc solution respectively. It is seen that the incremental solution $C_3$ is quite close to $C_1$. On the other hand, the results from the one-step Newton-Raphson method not only do not show any improvement over those of the straight incremental, but contrary to expectation they appear grossly inaccurate. This method is not used further in this report.

### 4.2.1.2 CANTILEVER BEAM WITH A SINGLE TIP LOAD

The problem considered is illustrated in Figure 4-4. For this example again we have used for reference the beam-col-NR solution

(Curve $C_1$) as the "exact" solution. Included in this figure are also results as obtained by the M & M-updated formulation (Curve $C_2$), Martin's method (Curve $C_3$) and the FEA-updated formulation (Curve $C_4$).

It is seen that $C_2$ is grossly inaccurate. The result is very similar to that using Powell's formulation [15] in the example discussed in Section 4.2.1.1.1 in which the structure responded with an unusually large stiffness.

Both Curves $C_3$ and $C_4$ appear acceptable, with $C_4$ a little better. Comparison of $C_4$ (FEA-updated) and $C_2$ (M & M-updated) shows here, as in Table 4-1, the very important positive effect which the average axial strain assumption has on the solution in comparison with the quartic axial strain assumption for the finite element model.

It is of interest to note that the FEA-updated solution converged faster than the beam-col-NR solution, especially for the lower load levels (e.g., 3 and 7 iterations were needed for the first increment, respectively, for the two methods).

### 4.2.1.3 CANTILEVER BEAM WITH BOTH LATERAL AND AXIAL LOADS

The system considered is illustrated in Figure 4-5. The lateral load is 10% of the axial load. Two methods have been used, beam-col-NR and FEA-updated. It is seen that the load-displacement relations become nonlinear at the early stage of loading. The two curves agree very well until 90% of the Euler load at which point the corresponding deflection is on the order of half of the span length.

### 4.2.1.4 CANTILEVER BEAM SUBJECTED TO END MOMENT

This problem was considered in [18]. It involves gross distortions of the beam. For comparison, solutions corresponding to five methods are shown in Figure 4-6, 4-7, and 4-8. These cover lateral deflections,longitudinal deflections and end rotations, respectively.

For lateral displacements all solutions agree quite well up to approximately $.7\ell$. It is interesting to note that the displacement reverses its direction beyond this point as loading increases.

Comparison of longitudinal displacement and end rotation indicate similarly good agreement. For comparable accuracy it took only one element for the beam-col-NR [10] and Jennings' formulation [13], but five elements were needed for the FEA-updated formulation and 20 elements for the ADINA [18] model (which also needed 90 steps versus the 20 steps used by all other formulations).

It should be emphasized again that this comparison is based on unusually large distortions of the structure, as illustrated by the dotted curve in the figures representing the final configuration of the structure.

### 4.2.1.5 CURVED BEAM SUBJECTED TO A LATERAL LOAD

This example is also taken from[18]. It deals with the three dimensional structure illustrated in Figure 4-9, which also contains three sets of solutions obtained by ADINA, MARTIN'S method [11] and the FEA-updated method.

It is seen that the three sets of solutions are quite close to each other. The ADINA [18] solutions, obtained by use of a large number of elements and load steps, should be regarded as the most correct one.

The curves corresponding to the FEA-updated method are closer to the ADINA curves than those by Martin's method.

### 4.2.1.6 DISCUSSION

From the preceding numerical examples involving large displacements, the following observations may be made:

a) The convergence criterion based on the unbalanced force vector (i.e., equilibrium check) is more reliable than a convergence check based on the displacement vector.

b) Of the three procedures, the straight incremental, one-step Newton-Raphson, and Newton-Raphson method, the first one is efficient and provides reasonable results, the second one is not reliable and the last one is accurate, but relatively less efficient.

c) The fixed-Lagrange coordinate formulation and the M & M updated method should not be used.

d) Martin's approach gives very good results.

e) The FEA-updated method gives slightly more accurate results and is somewhat more effective than Martin's method.

f) As expected Jennings'-NR results (because of its Eulerian formulation) produces more accurate solutions than the FEA-updated ones. Jennings' incremental formulation is not effective. (The judgment on the effectiveness of a method is based on both accuracy and efficiency).

Except for the fixed coordinate formulations all the methods require updating of the geometry. This is not unexpected, because of the large deflections and rotations involved.

However, for some nonlinear problems, displacements may not be very large and sufficiently accurate results may be obtained without updating the geometry (thus saving computation time). In the following sections on "Small" and "Intermediate" displacement problems, we continue to include the fixed-Lagrangian methods as well as updated geometry approaches in the investigation.

## 4.2.2 "SMALL DISPLACEMENT" PROBLEMS

### 4.2.2.1 ONE SPAN PORTAL FRAME

As the first example for the small deflection class of problems, a one span portal frame is considered (see Figure 4 10). To initiate nonlinear behavior, a small horizontal load equal to .1% of each of the vertical loads is applied. For this structure as well as all the other frames subsequently considered in this report, each member is represented by a single finite element.

Load-displacement curves corresponding to five methods are shown in Figure 4-10. It is seen that although the behavior is quite nonlinear, the displacements are small (i.e., on the order of 1% of the linear dimension of the structure).

As before, the beam-col-NR solution is regarded as the "exact" one. It is seen that, except for the M & M-updated results, all other solutions including the M & M-fixed are very close to the "exact". They also check very well with the results presented by Conner et al. [7]. The values for $P_{cr}$ (critical load) shown in the figure will be discussed later when we consider eigenvalue problems.

#### 4.2.2.2 TWO STORY FRAME

This example deals with a larger structure than the preceding one. In this section we are mainly interested in examining the effectiveness of the FEA-fixed method relative to the FEA-updated and the beam-column "exact" solutions. Therefore, only these three methods are used in the following "Small Deflection" problems.

As shown in Figure 4-11, the order of magnitude of displacement is about 1.5% of the length of a member at the maximum load. The nonlinear behavior is, however, conspicuous. It is seen that the results given by all three methods are very close to each other.

#### 4.2.2.3 TWO BAY FRAME

This example is very similar to the previous one, except that it is a two bay frame, instead of a two story frame. Figure 4-12 shows the properties of the structure as well as a comparison of three solutions for this example. Again the solutions agree very well with one another.

#### 4.2.2.4 PLANE ARCH FRAME WITH HINGED SUPPORTS

Shown in Figure 4-13(a) is a three member symmetric arch frame subjected to two vertical loads. The new aspect in this example is the existence of bending in the structure due to the inclination of the columns,even with no lateral load on it.

It is seen from the load-displacement plots that the behavior is essentially linear. Although there is no sign of instability from the load-displacement curves, at 2378 kips and 2186 kips the determinant of the tangent stiffness matrix of the system vanished, respectively, in the FEA-updated and beam-col-NR solutions. It did not vanish for the FEA-fixed approach.

In Figure 4-13(b) are shown the same arch frame and vertical loading. In addition, a small horizontal load equal to .001 of a vertical load is also applied. In this case, the load-displacement behavior began to show nonlinearity at P ≅ 1500 kips.

For Curve $C_1$ (beam-col-NR) there is again a change of sign in the determinant at P = 2190 kips. For the FEA method no such change of sign was indicated. However, the iteration failed to converge (after 20 cycles) at P = 2000 kips and P = 2250 kips, respectively, for both the updated and the fixed versions.

Although theoretically no bifurcation load is expected for this loading, the lack of convergence, like the vanishing of the determinant, could be taken as a sign of instability.

### 4.2.2.5 SPACE ARCH FRAME

For this example, the properties of the symmetric structure and loading are shown in Figure 4-14. Since we did not have the program for a three dimensional version of the beam-col-NR method only the results of the FEA methods are shown in Figure 4-15. It is seen that the two curves are very close to each other.

These solutions will be referred to again in a later discussion of the buckling load of the structure.

### 4.2.3 "INTERMEDIATE DISPLACEMENT" PROBLEMS

For convenience, displacements which are neither "small" nor "large" as defined previously are referred to as "intermediate". The arch problem considered in the following falls in this category.

### 4.2.3.1 HINGED HALF CIRCULAR ARCH WITH A CONCENTRATED LOAD AT CROWN

This example is considered mainly to show the effect of the order of deflection on the results of the M & M-fixed and updated methods and on those of the FEA-fixed and updated methods.

The properties of the structure and the solution curves are shown in Figure 4-16 in which the beam-col-NR solution is presented for reference as the "exact" solution.

From a comparison of the curves, it is seen that the FEA-fixed and updated results agree quite well with the beam-col-NR result.

The curve corresponding to the M & M-fixed method shows excessive stiffening while that of the M & M-updated method shows excessive softening.

The order of the maximum deflection is approximately 10% of the arch span and 25% of the length of each element. At this level of displacements, the corresponding load approaches the bifurcation load of the arch. This will be discussed further later.

### 4.2.3.2 CANTILEVER BEAM WITH TWO LATERAL LOADS

The system considered is identical with that shown previously in Figure 4-1. We have seen previously that the "FEA-fixed" and "M & M-fixed" methods did not produce accurate results for problems involving "large displacements". On the other hand, good results were obtained if the displacements were small (although the behavior was nevertheless quite nonlinear). An interesting question would be: what would be the largest displacement at which the FEA-fixed or the M & M-fixed method could be considered valid?

To obtain an approximate answer to this question, we refer to the results of the study presented previously for the cantilever beam in Figure 4-1. In Table 4-2 are listed displacements up to values equal to approximately 15% of the beam length. The load displacements for the problem are from the beam-col-NR and the two finite element methods. An examination of the table leads to the following observations. The results obtained by the M & M-fixed method are unacceptable. The FEA-fixed method produced reasonably accurate results (in comparison with the beam-col-NR solution) for the range of deflection considered, i.e., equal to approximately 15% of the beam length. It is of some interest to note that in this range the lateral and rotational displacements are essentially linear, while the longitudinal displacement is not.

## 4.3 BUCKLING LOAD STUDIES

### 4.3.1 GENERAL

To consider the stability of a framed structure we refer first to Figure 2-4. Here is shown a typical nonlinear load deflection behavior, Curve OACD, which is called the "fundamental path". The laod at C is known as the "limit load" which in practice may not be reached because the bifurcation load may be reached sooner.

The bifurcation load (if it exists) may be obtained by checking the determinant of the tangent stiffness matrix (det $[K_T]$); i.e., it is defined as that point along the fundamental path (e.g., Point A in Figure 2-3) at which det $[K_T]$ vanishes.

If the only objective for analysis is to evaluate the bifurcation load, this approach would not be efficient because of the amount of computation needed for the load-displacement curve, and checking the determinant. It is well known in the classical theory of elastic stability that for some systems the bifurcation load may be obtained in a simpler manner by the formulation and solution of an eigenvalue problem.

As will be pointed out later, for some other systems for which bifurcation loads either are not obtainable by an eigenvalue analysis or simply do not exist, eigenproblems may still be formulated. The solutions for such problems can be obtained as rough estimates of the maximum load capacity. These estimates may also be useful in calculating the nonlinear load-displacement curve, e.g., in the selection of load increment.

We will consider four types of eigenvalue problems formulated for finite element models:

a) Linear eigenvalue problems using the regular first order nonlinear stiffness matrix (i.e., $[n_1]$ in Equation 2-16)

b) The same as (a) but using $[n_1*]$ as it is defined in Equation (2-17)

c) Quadratic eigenvalue problems using $[N_1]$ and $[N_2]$ which is the second order nonlinear stiffness matrix. Based on the assumption used in the derivation of $[N_2]$ two combinations exist:

(1) $[N_1]$ and $[N2_Q]$

(2) $[N_1]$ and $[N2_A]$

in which $[N2_Q]$ and $[N2_A]$ are based on the quartic and average axial strain assumption, respectively.

In order to judge the accuracy and usefulness of the solution of the eigenproblem we have also obtained the load-displacement curves, as well as the det $[K_T]$ as a function of load for the numerical examples considered here. The beam-col-NR method has been used (except for a three dimensional case for which no beam-column solution is available) to obtain the load-displacement and the load-determinant curves.

In Figure 4-17 are illustrated three typical load-determinant curves. In the beginning as the load increases, the determinant of the tangent stiffness decreases. There are three possibilities for subsequent behavior [25]:

a) The curve crosses the load axis at point A with an angle $\neq 90^\circ$. The load at A is the bifurcation load.

b) The curve crosses the load axis at point B at $90^\circ$, the load at B is the limit load.

c) The det $[K_T]$ reaches a minimum at point C without crossing the load axis and increases in value afterwards. In this case, there is no critical load. That is, the structure can continue to take more load increments. However, at that point, the displacement and its rate of increase usually are already very large. In general, for the problems discussed here , the numerical solution was stopped after the minimum had been detected.

In the following, numerical problems involving both symmetric and asymmetric loading will be considered.

### 4.3.2 PROBLEMS INVOLVING SYMMETRIC LOADING

#### 4.3.2.1 ONE SPAN PORTAL FRAME

The geometry and properties of the structure, as well as the load-displacement curve, are shown in Figure 4-18. On the curve are marked the critical loads. The critical load obtained from checking the determinant of the tangent stiffness matrix of the beam-col-NR [10] solution is denoted by $P_{BC}$, and those obtained from linear eigenvalue solutions are denoted by $P_{N_1}$ and $P_{N_1}*$ (similarly critical loads obtained from quadratic eigenproblem will be denoted by $P_{N_1+N_{2_A}}$ and $P_{N_1} + N_{2_Q}$).

Neither of the quadratic eigensolutions converged, but as indicated on the curve, $P_{N_1}*$ is very close to $P_{BC}$ while $P_{N_1}$ is not. So using $N_1*$ provides a better approximation for the bifurcation load in this problem. The buckling modes corresponding to $P_{N_1}*$ and $P_{N_1}$ are antisymmetric.

#### 4.3.2.2 ARCH PROBLEM WITH A CONCENTRATED LOAD AT CROWN

In Figure 4-19 are shown the vertical and horizontal displacements at the crown of a 135°-arch subjected to a concentrated load. $P_{BC}$ was found to be equal to 8.56 pounds. The quadratic eigenvalue solutions did not converge. As in the preceding example, $P_{N_1}*$ agrees with $P_{BC}$ but $P_{N_1}$ does not.

It is of interest to note from the load-displacement curve that the displacement components increase abruptly near $P_{BC}$. The abrupt increase happened after a finite load increment over $P_{BC}$ at $P = \overline{P}$.

Of course, the equilibrium state on the fundamental path after bifurcation is unstable. After the abrupt increase in displacement, the state would be stable as seen from the load-determinant plot shown in Figure 4-19(b). The values of det $[K_T]$ for $P_{BC} < P < \bar{P}$ is negative, but it turns positive when $P > \bar{P}$.

In Table 4-3 are shown the numerical results for the above arch as well as arches with a 90° and 180° opening angles. Again the values of $P_{N_1}$ are very close to $P_{BC}$. The difference between $P_{N_1}$ and $P_{N_1}*$ seems to increase with the opening angle, i.e., the comparison improves in the case of the 90°-arch but deteriorates with the 180°-arch.

The behavior of the load displacement and load-determinant curves for the 90° and 180° angles are similar to the ones presented for the 135° angle. The buckling modes corresponding to $P_{N_1}*$ and $P_{N_1}$ for all cases in this example are antisymmetric.

4.3.2.3 SPACE ARCH FRAME

The finite element eigensolutions of the three dimensional space frame as described in Figure 4-14 have been obtained. For these calculations the horizontal load Q was set equal to zero.

$P_{N_1}$ was found to be 87 kips which corresponds to a lateral buckling with a mode shape which is antisymmetric and normal to the planes of either arch ribs.

The same solution was obtained using the quadratic eigenproblem formulation. From the load-displacement curves plotted in Figure 4-15 it may be noted that the eigensolutions represent a good estimate of the limit load of the system. In fact, in the FEA-fixed solution the determinant of the tangent stiffness matrix did change sign (vanish) at $P = 89.7$ kips.

## 4.3.3 PROBLEMS INVOLVING ASYMMETRIC LOADING

### 4.3.3.1 GENERAL

For this class of problems no bifurcation load exists in beam-column theory. However, eigenproblems may still be formulated using finite element models. It is of interest to study the significance (or lack of it) of their solutions. Again judgment should be based on a comparison with the load-displacement, load-determinant curves of the beam-col-NR method.

### 4.3.3.2 HORIZONTAL AND VERTICAL LOADING

#### 4.3.3.2.1 ONE SPAN PORTAL FRAME

This problem involves a square portal frame subjected to two vertical loads and a small horizontal load. It has been considered previously (for nonlinear load-displacement behavior study) in Section 4.3.2.1. The load-displacement curves are shown in Figure 4-10.

As expected, det $[K_T]$ did not vanish in the beam-col-NR solution (i.e., there is no bifurcation load). The load determinant plot belongs to Type (C) in Figure 4-17.

However, solutions for all four types of eigenproblems have been obtained. They are as follows:

$P_{N1}$ = 4759 kips, $P_{N1}{}^*$ = 4758 kips, $P_{N1+N2_\Omega}$ = 4764 kips and

$P_{N1+N2_A}$ = 4759 kips.

We can also note from the load-displacement curves shown in Figure 4-10 that any of these critical load values may be regarded as a good estimate of the limit load of the system. This is not unexpected because the horizontal load is very small and the exact bifurcation load for this frame (in the absence of the horizontal load) is 4750 kips [7], which is extremely close to all the critical loads obtained from the eigensolutions.

#### 4.3.3.2.2 TWO BAY FRAME

This system is similar to the previous one, but larger. The load-displacement curve is shown in Figure 4-12. The following eigensolutions have been obtained.

$$P_{N_1} = 4940 \text{ kips}, \; P_{N_1}{}^* = 4940 \text{ kips and } P_{N_1+N_{2A}} = 4939 \text{ kips}$$

From the load-displacement curve it is seen that these results may be regarded as a limit load. By checking the det $[K_T]$ we obtained:

$P_{BC}$ = 4965 kips, and from the FEA-updated solution, the critical load is 5028 kips. Both of these are very close to the eigensolutions given above.

### 4.3.3.3 ASYMMETRIC VERTICAL LOADING

#### 4.3.3.3.1 ONE SPAN PORTAL FRAME SUBJECTED TO ASYMMETRIC VERTICAL LOADS

In Figure 4-20 the load displacement plots are shown for four cases of asymmetric vertical loading as illustrated therein. It should be noted that for clarity the curves begin at different points on the displacement axis, and for purposes of comparison the case of symmetric loading has been replotted from Figure 4-18.

For asymmetric loading, no bifurcation load is expected. Indeed the load-displacement plots obtained from the beam-col-NR solution (not shown) belong to Type (C) of Figure 4-17. As before, the quadratic eigensolutions did not converge, but the linear eigenvalue solutions have been obtained and marked on Figure 4-20.

It is seen that at $P_{N_1}*$ the structure has gone substantially into the nonlinear range, while at $P_{N_1}$ the displacements would appear to begin their higher rate of increase. It would seem that one could use $P_{N_1}$ or the average of $P_{N_1}$ and $P_{N_1}*$ as an index of a "limit load" of the structure-load system.

## 4.3.3.3.2 A 90°-ARCH SUBJECTED TO TWO VERTICAL LOADS

Similar to the case considered in the preceding example, the load-displacement curves for an arch with an opening angle equal to 90° (approximated by four elements), subjected to two symmetrically placed loads, are shown in Figure 4-21.

Again, except for the case of two equal loads, there is no bifurcation load for the system considered. The load-displacement curves were terminated at points beyond which the number of cycles of iteration for convergence increased drastically and the converged results did not appear physically reasonable.

As before, the finite element linear eigenvalue solutions have been noted. It is of interest to note that while $P_{N_1}$ values would provide a rough measure of the "limit load", $P_{N_1}*$ values noted along the load axis were too high to be of any significance.

## 4.3.3.3.3 A HALF CIRCULAR ARCH SUBJECTED TO AN ASYMMETRIC LOADING

The geometry of the structure and the load-displacement curves are shown in Figure 4-22. It may be seen that the behavior is highly nonlinear. At P = 5.2 lbs., the arch "snaps" and it is so grossly distorted that part of it now lies below the chord.

The values of $P_{N_1}$ and $P_{N_1}*$ were found to be equal to .94 lbs. and 68.70 lbs., respectively.

In this case $P_{N_1}*$ is totally meaningless, and $P_{N_1}$ is too small to be of significance.

# CHAPTER V

# DISCUSSION AND CONCLUSIONS

A comparative study involving a number of existing and some new methods was presented in the preceding chapter. From the numerical results obtained an assessment of the methods may be given as follows:

## 5.1 ASSESSMENT OF METHODS

1) Martin's method [11], a "straight incremental" type, is very efficient and generally quite accurate for all types of problems. This method is very sensitive to the step size, but gives good results even with very few elements (even one element per beam or column). Unfortunately, there is no equilibrium check (or convergence check of any kind) involved in his procedure. The only way to judge the results is by comparing them with known accurate solutions or by decreasing the step size and/or increasing the number of elements until a pattern of converging results emerges.

2) Jennings' formulation [13], when used with the Newton-Raphson procedure, produces very good results for all classes of problems with a small number of steps and a small number of elements. Because of the Eulerian formulation it requires coordinate transformation in every iteration, which tends to be time consuming. The straight incremental version is very sensitive to the step size and number of elements. It is ineffective for "large displacement" problems.

3) Mallett and Marcal's method [14], based on fixed-Lagrange coordinates, is effective for "small displacement" problems. It requires

small numbers of elements and number of load steps for an acceptable solution. However, it is totally inaccurate for "large displacement" problem.

The version based on the updated-Lagrange coordinates developed here did not result in any improvement.

4) Powell's method [15] is good only for "small displacement" problems. In general it is very sensitive to the step size and number of elements both in the iterative and straight incremental versions. The method is not efficient because of the Eulerian formulation.

5) Based on continuum mechanics, Bathe's formulation [18] of three dimensional beam finite element should be very accurate. However, comparison indicated that simpler models used here with less computation requirements can produce results that are of the same order of accuracy.

6) The FEA-updated and FEA-fixed methods which have been considered throughout this study are quite efficient. For the "large displacement" problems, the FEA-updated method is competitive in accuracy with all approaches used above. They are not very sensitive to the step size and number of elements. In general, for the framed structures considered, one element per beam or column was enough. The FEA-fixed method is even more efficient as it involves no coordinate transformation. However, it should be used only for "small and intermediate displacement" problems.

## 5.2 CONCLUDING REMARKS

The objective of this study as stated in Chapter I was to search for an effective method which could be used for nonlinear behavior study of relatively large space frames. From the preceding chapters, it seems evident that there is no single method that is most useful for all structural load systems.

The choice would depend on whether the displacemnet is "large", "small" or "intermediate". For "small displacement" (say of the order of 2% or less of the length of a typical member), it appears that the FEA-fixed is attractive. For "intermediate displacements" (approximately 2-15% of the length of a member), the FEA-fixed method is still good. For "large displacements" (over 15% of member length) a number of methods are effective. They include the Jennings' method [13] and Martin's method [11]. However, the FEA-updated method is competitive with these two.

It is appropriate to comment on the continuum beam-column method which has been used as a reference throughout this study. When using Oran's tangent stiffness matrix [10], this method is quite efficient as far as a continuum model goes. However, it is less efficient than the finite element models because it requires iteration for the calculation of the axial force and computations involving transcendental functions. Of course, it also would be difficult to extend the formulation to elements that do not have a constant cross section.

In lieu of a complete but time consuming load-displacement analysis, some index values useful for engineering purposes may be obtained by the solution of eigenvalue probelms. From the eigenproblems considered here , it appears that solutions of the quadratic eigenproblems have little merit in the sense that whenever they are meaningful they are also very close to the solution of the simpler linear eigenproblems.

For the latter eigenproblems, the use of the usual geometrical stiffness matrix provides good results ($P_{N_1}$*) for classical problems of elastic stability, i.e., problems involving little or no primary bending. The use of the first order incremental stiffness matrix, $N_1$, in the eigenproblems overemphasizes bending effects in the system. However, it appears that in the same cases the critical load $P_{N_1}$ thus obtained could be taken as a rough estimate of the "limit load". This possibility seems to deserve further study.

As mentioned previously, the present study is limited to geometric nonlinearity. For many practical problems, when geometric nonlinearity becomes significant, effects of material nonlinearity would become important at the same time. Thus, future studies of finite element analysis of frame structures should include these effects.

TABLE 4-1 Comparison of Solutions for Cantilever Beam with Two Lateral Loads

| Row | Method | No. of Elem. | No. of Inc. | Horiz. defl. pt. B (in.) | Vert. defl. pt. B (in.) | Horiz. defl. pt. C (in.) | Vert. defl. pt. C (in.) |
|---|---|---|---|---|---|---|---|
| 1 | Elastica [24] | - | - | 8.28 | 25.14 | 31.01 | 67.32 |
| 2 | Beam-Col-NR [10] | 2 | 20 | 8.04 | 24.68 | 30.57 | 66.75 |
| 3 | Beam-Col-Inc [10] | 2 | 20 | 7.89 | 25.26 | 30.17 | 68.88 |
| 4 | Jennings'-NR [19] | 2 | 20 | 8.08 | 24.68 | 30.63 | 66.76 |
| 5 | Jennings'-Inc [19] | 2 | 20 | .01(.06) | 2.19(3.05) | .03(.54) | 6.39(10.99) |
|  |  | 20 | 100 | 4.94(8.06) | 20.60(24.89) | 15.01(30.71) | 51.14(67.37) |
| 6 | Powell's-NR [19] | 10 | 20 | 1.48 | 10.84 | 7.86 | 35.31 |
| 7 | Powell's-Inc [19] | 2 | 20 | .01 | 2.08 | .02 | 6.19 |
|  |  | 20 | 100 | 3.40 | 16.94 | 11.43 | 44.49 |

TABLE 4-1 (continued)

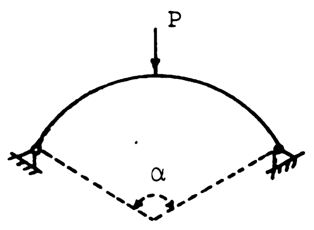
| Row | Method | No. of Elem. | No. of Inc | Horiz. defl. pt. B (in.) | Vert. defl. pt. B (in.) | Horiz. defl. pt. C (in.) | Vert. defl. pt. C (in.) |
|---|---|---|---|---|---|---|---|
| 8 | Martin [19] | 2 | 20 | 8.97 | 27.49 | 36.02 | 74.18 |
| | | 20 | 100 | 9.11 | 26.29 | 34.37 | 70.62 |
| 9 | M&M-fixed | 2 | 20 | .0056 | .709 | .0167 | 1.77 |
| 10 | M&M-updated | 2 | 20 | .0007 | .246 | .021 | 1.67 |
| 11 | FEA-fixed | 2 | 20 | 18.52 | 39.19 | 78.91 | 117.28 |
| 12 | FEA-updated | 2 | 20 | 8.13 | 27.73 | 35.41 | 72.70 |

TABLE 4-2 Numerical Results for Cantilever Beam Subjected to Two Lateral Loads as shown in Figure 4-1

| load | Beam-col-NR | | | FEA-fixed | | | M & M - fixed | | |
|---|---|---|---|---|---|---|---|---|---|
| | u | v | θ | u | v | θ | u | v | θ |
| .0085 | -.0079 | -1.1724 | -.0165 | -.0079 | -1.1728 | -.0165 | -.0006 | -.3416 | -.0043 |
| .0255 | -.0707 | -3.5100 | -.0495 | -.0710 | -3.5185 | -.0497 | -.0015 | -.5285 | -.0064 |
| .0425 | -.1949 | -5.8271 | -.0822 | -.1972 | -5.8642 | -.0828 | -.0022 | -.6387 | -.0076 |
| .0595 | -.3741 | -8.0709 | -.1138 | -.3866 | -8.2099 | -.1159 | -.0028 | -.7210 | -.0085 |
| .0765 | -.6062 | -10.2697 | -.1448 | -.6391 | -10.5556 | -.1490 | -.0033 | -.7884 | -.0092 |
| .0935 | -.9065 | -12.5464 | -.1772 | -.9548 | -12.9012 | -.1821 | -.0038 | -.8462 | -.0099 |
| .1105 | -1.2734 | -14.8506 | -.2102 | -1.3335 | -15.2469 | -.2152 | -.0043 | -.8972 | -.0105 |

u = Longitudinal Deflection at c, v = Lateral Deflection at c, θ = End Rotation

TABLE 4-3 Comparison of Eigensolutions and Load-Determinant Results for a Symmetric Arch Subjected to Concentrated Load at Crown



$E = 10000.$ psi

$A = .1875$ in$^2$

$I = .008789$ in$^4$

$R = 10.$ in

4 equal elements

| $\alpha$ | $P_{BC}$ | $P_{N_1}^*$ | $P_{N_1}$ | $\frac{P_{N_1}^*}{P_{BC}}$ | $\frac{P_{N_1}}{P_{BC}}$ |
|---|---|---|---|---|---|
| $90^o$ | 12.70 | 13.47 | 10.23 | 1.0606 | .8055 |
| $135^o$ | 8.56 | 8.56 | 4.33 | 1.00 | .5058 |
| $180^o$ | 4.71 | 5.62 | 2.06 | 1.1932 | .3665 |

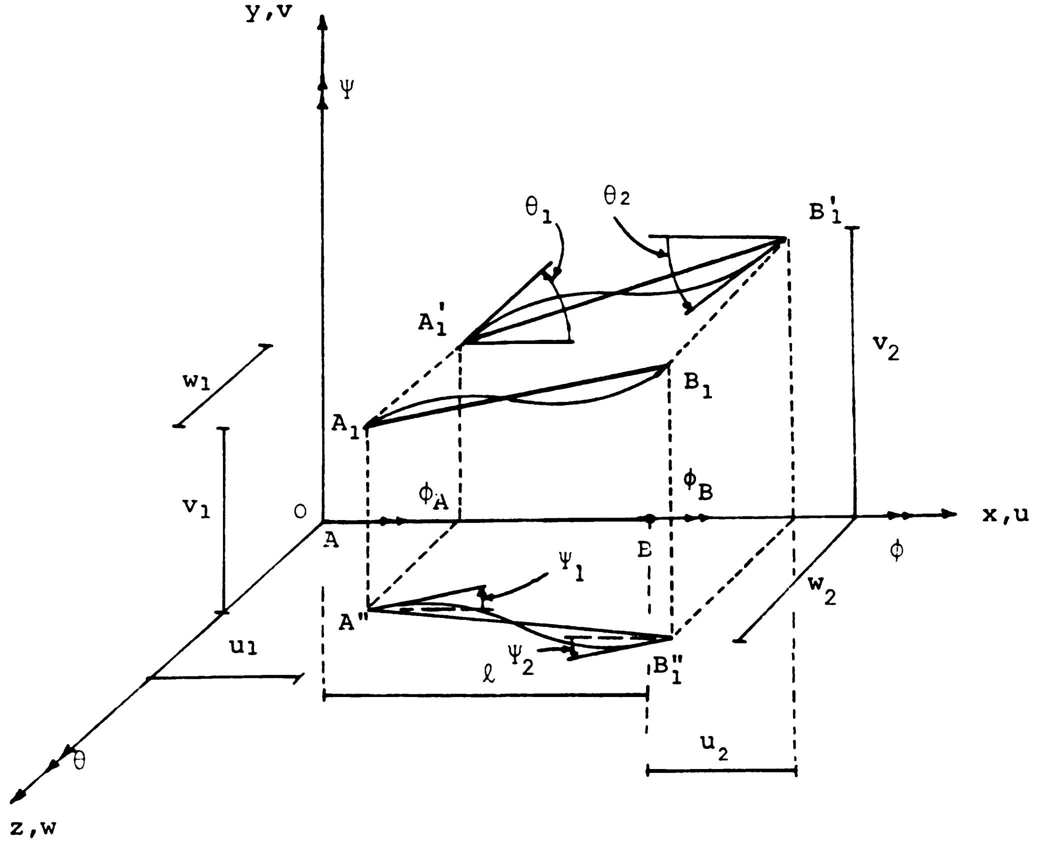


FIGURE 2-1 End Displacements of Three Dimensional Beam Element

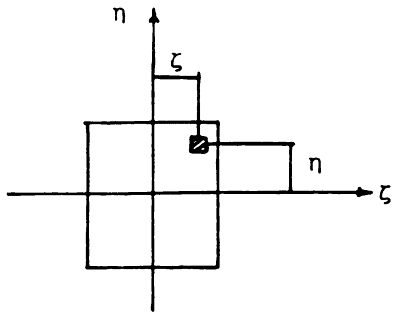


FIGURE 2-2 Cross Section of Beam Element

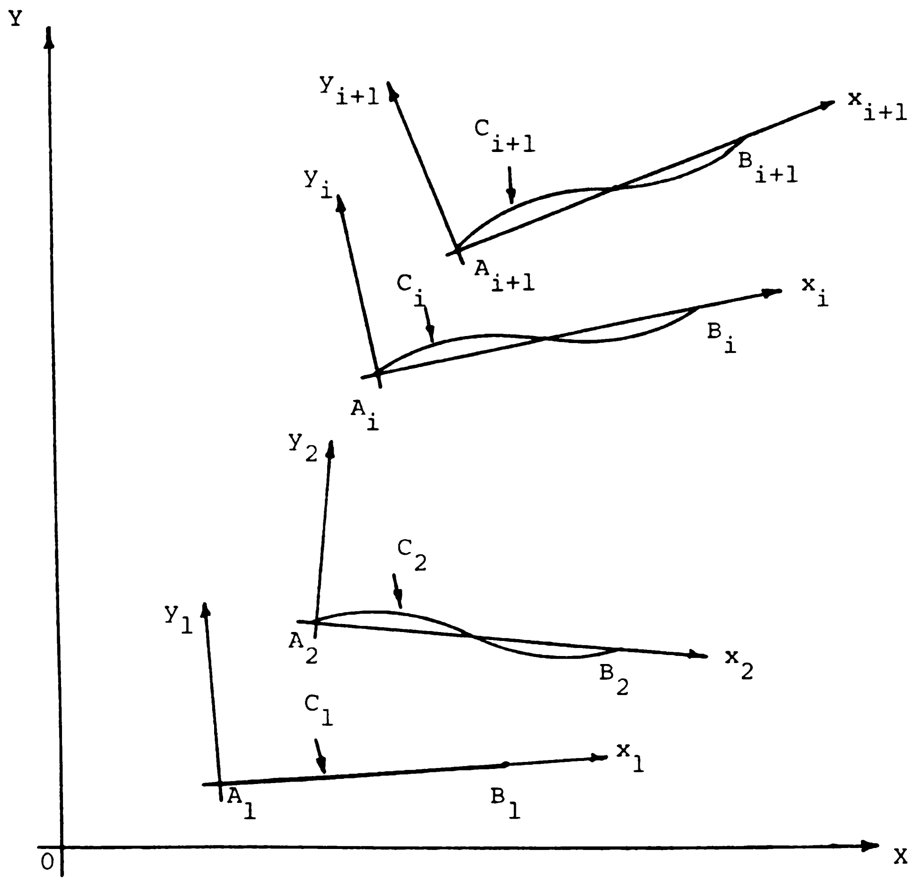


FIGURE 2-3 Configuration of a Two Dimensional Beam Element at Successive Load Increments in Updated-Lagrange Formulation

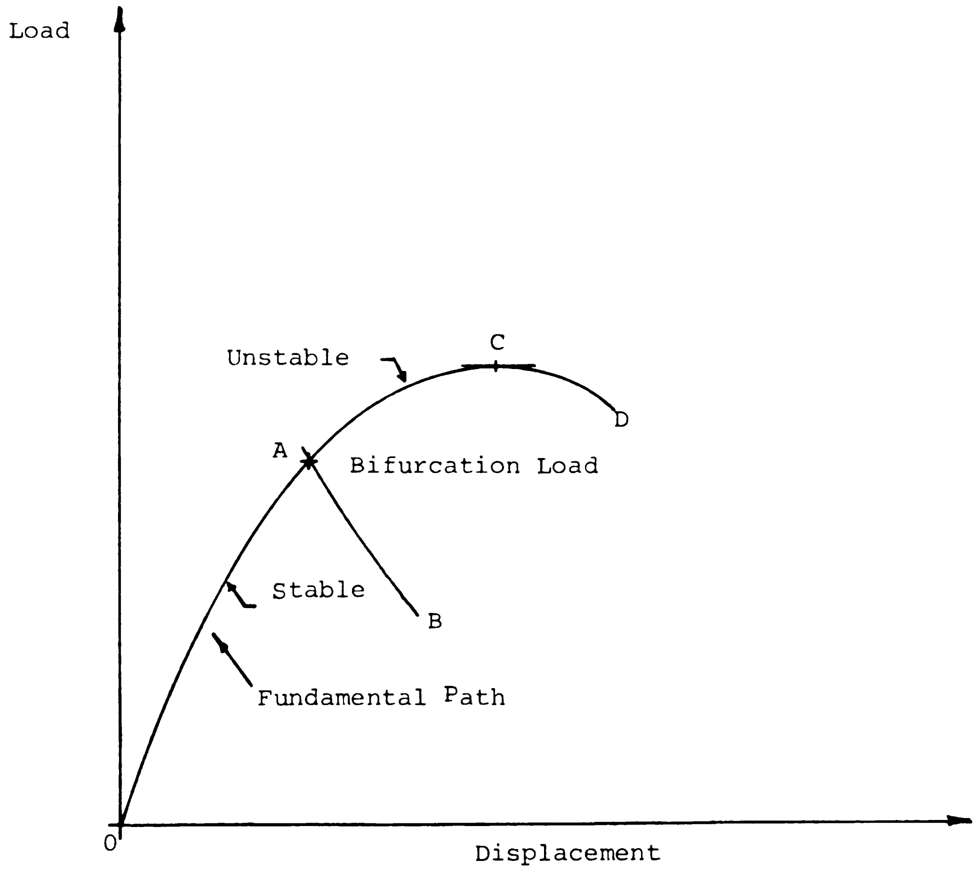


FIGURE 2-4 Nonlinear Load Deflection Relation

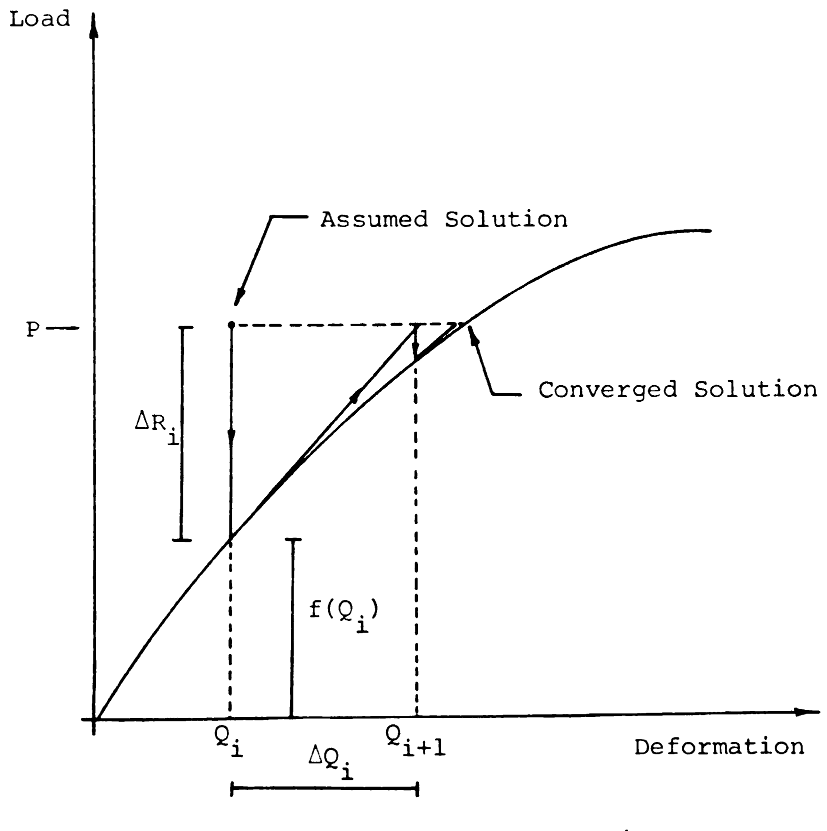


FIGURE 3-1 Newton-Raphson Iteration

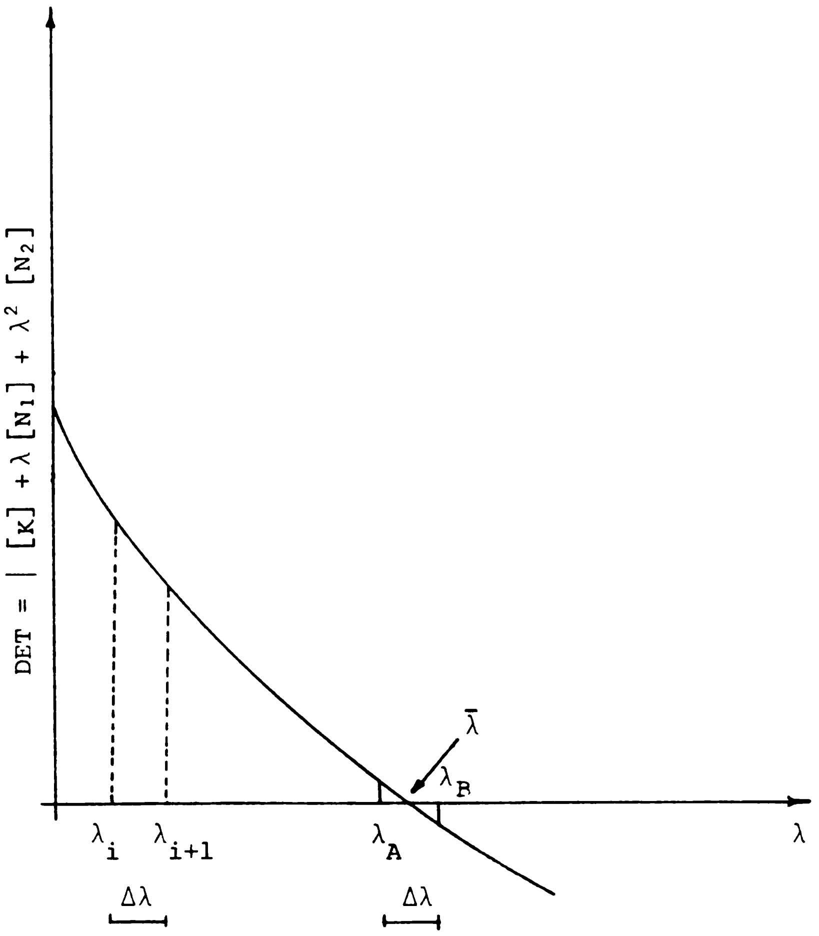


FIGURE 3-2  Determinant Search Method

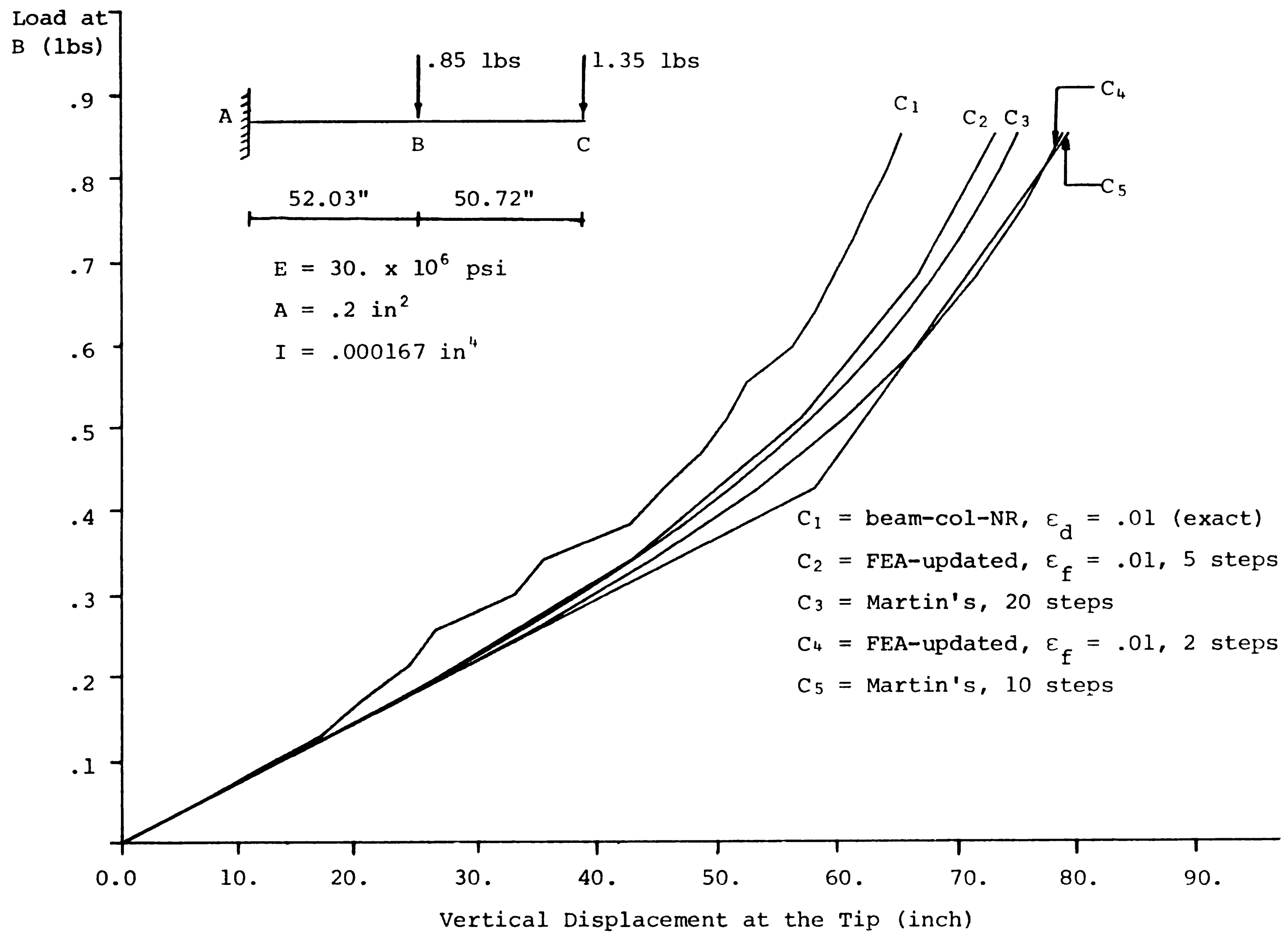


FIGURE 4-1 Comparison of Martin's Method and FEA-Updated Method

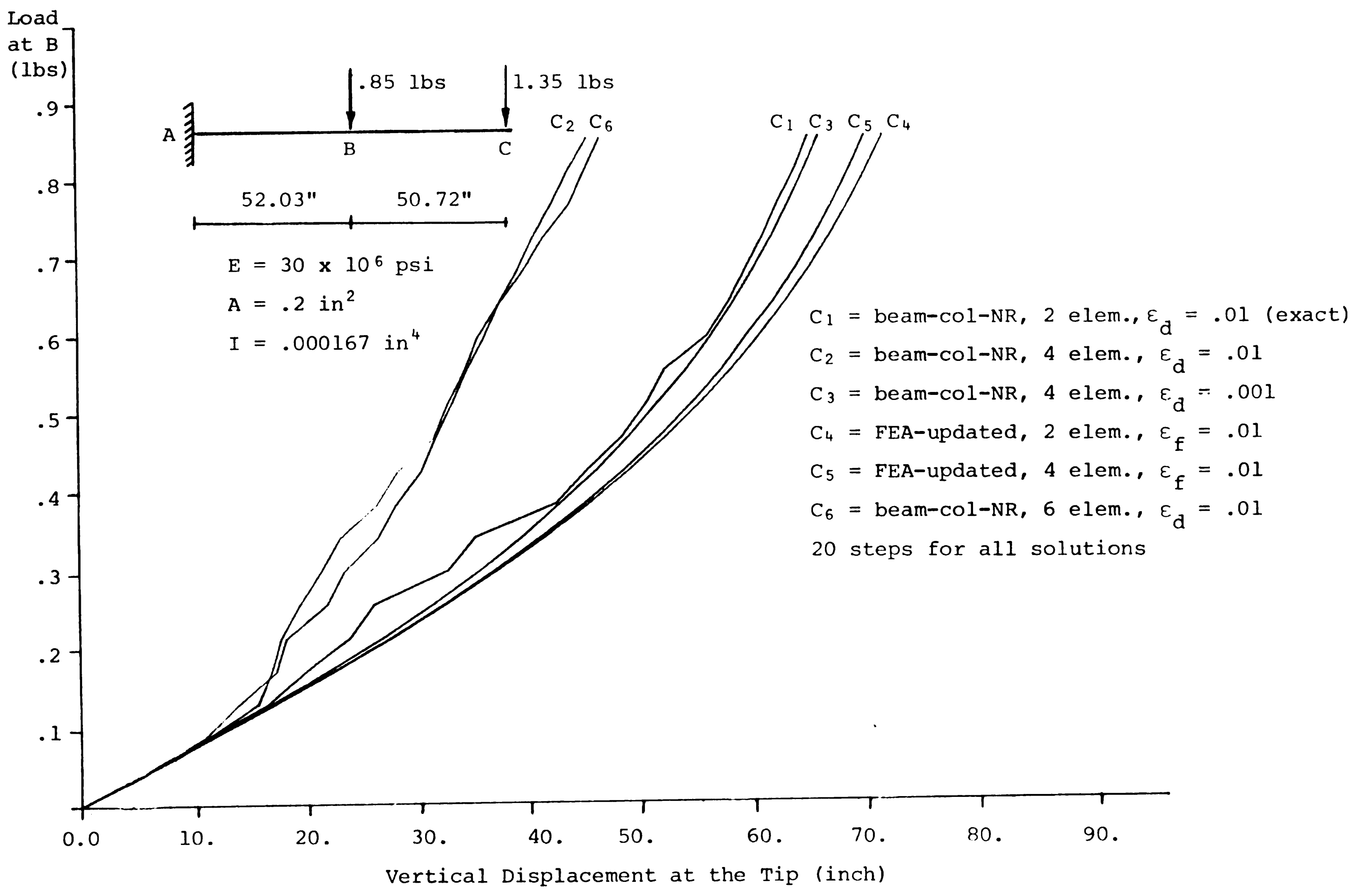


FIGURE 4-2 Effect of Convergence Criterion

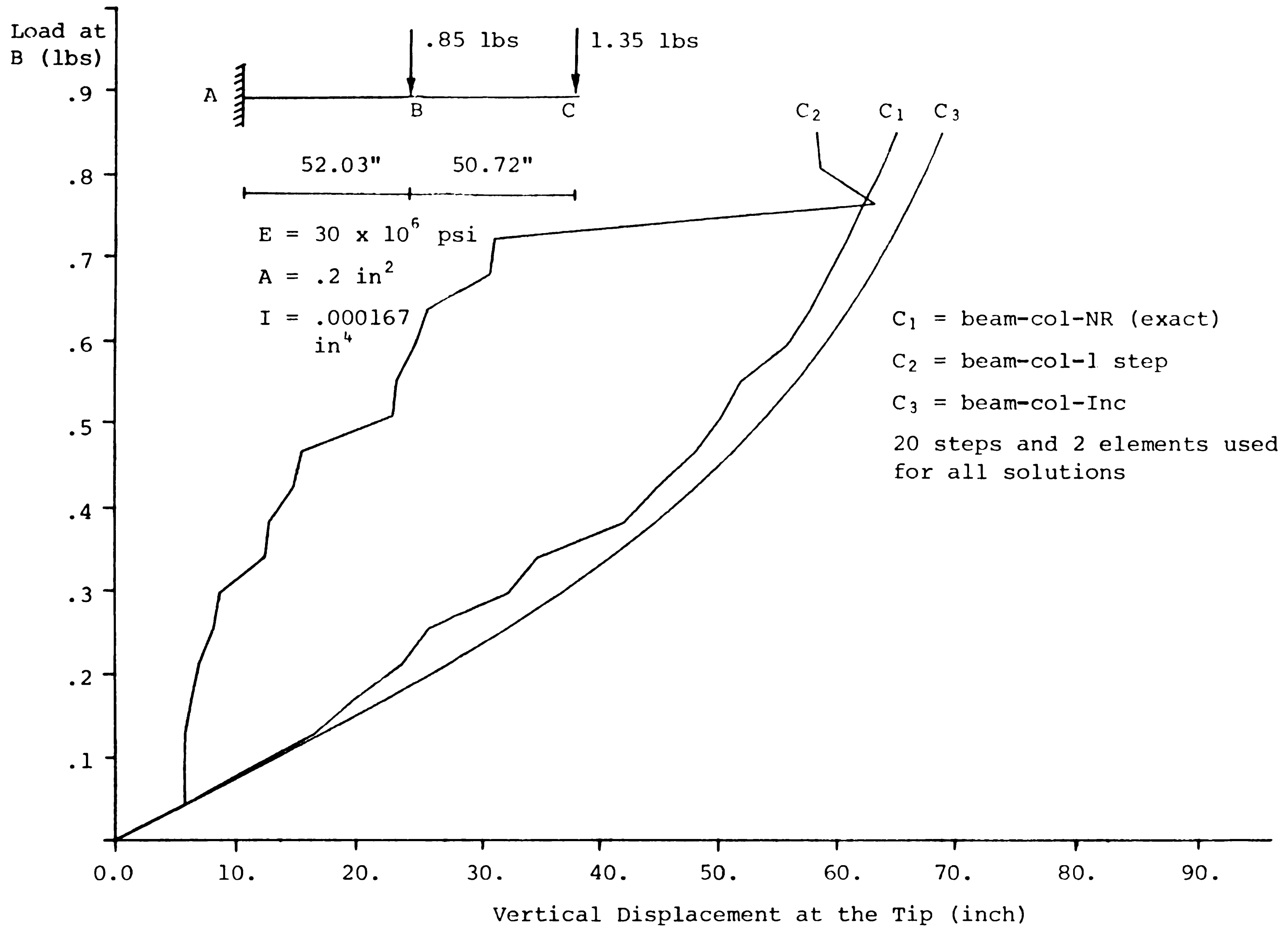


FIGURE 4-3 One-step-NR versus Straight Incremental

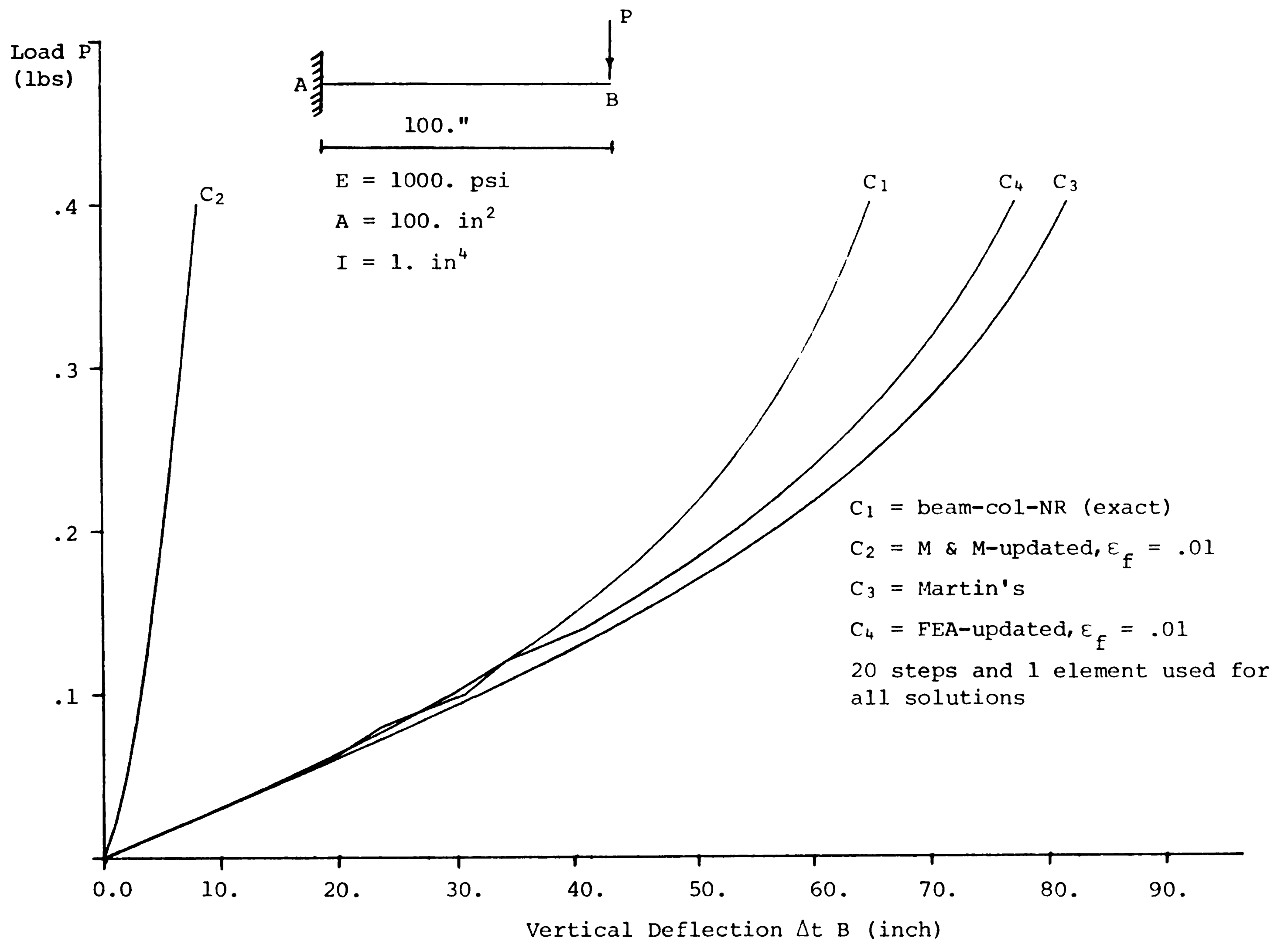


FIGURE 4-4 Comparison of Solutions for Cantilever Beam with a Single Tip Load

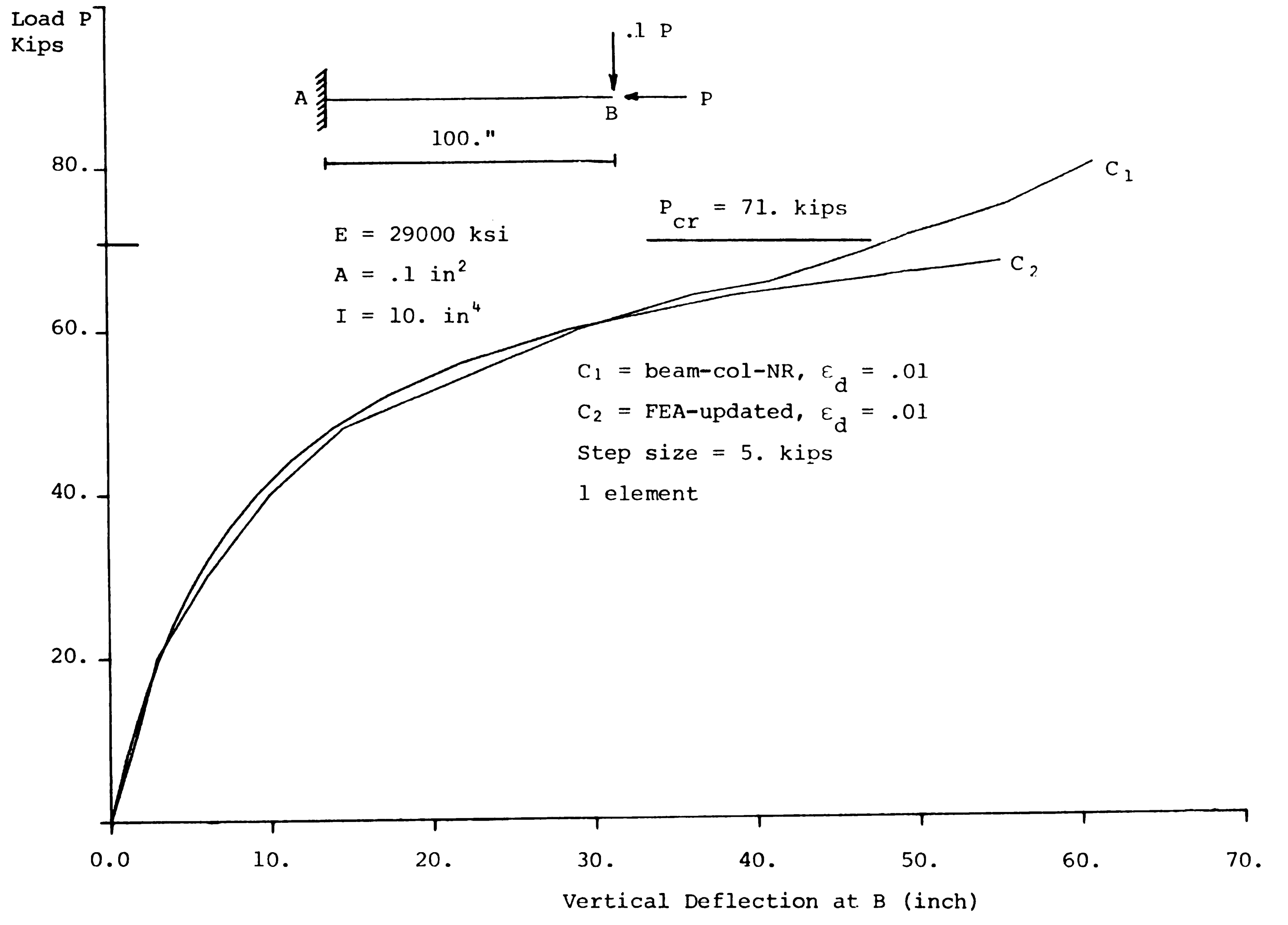


FIGURE 4-5 Comparison of Solutions for Cantilever Beam with Both Axial and Lateral Load

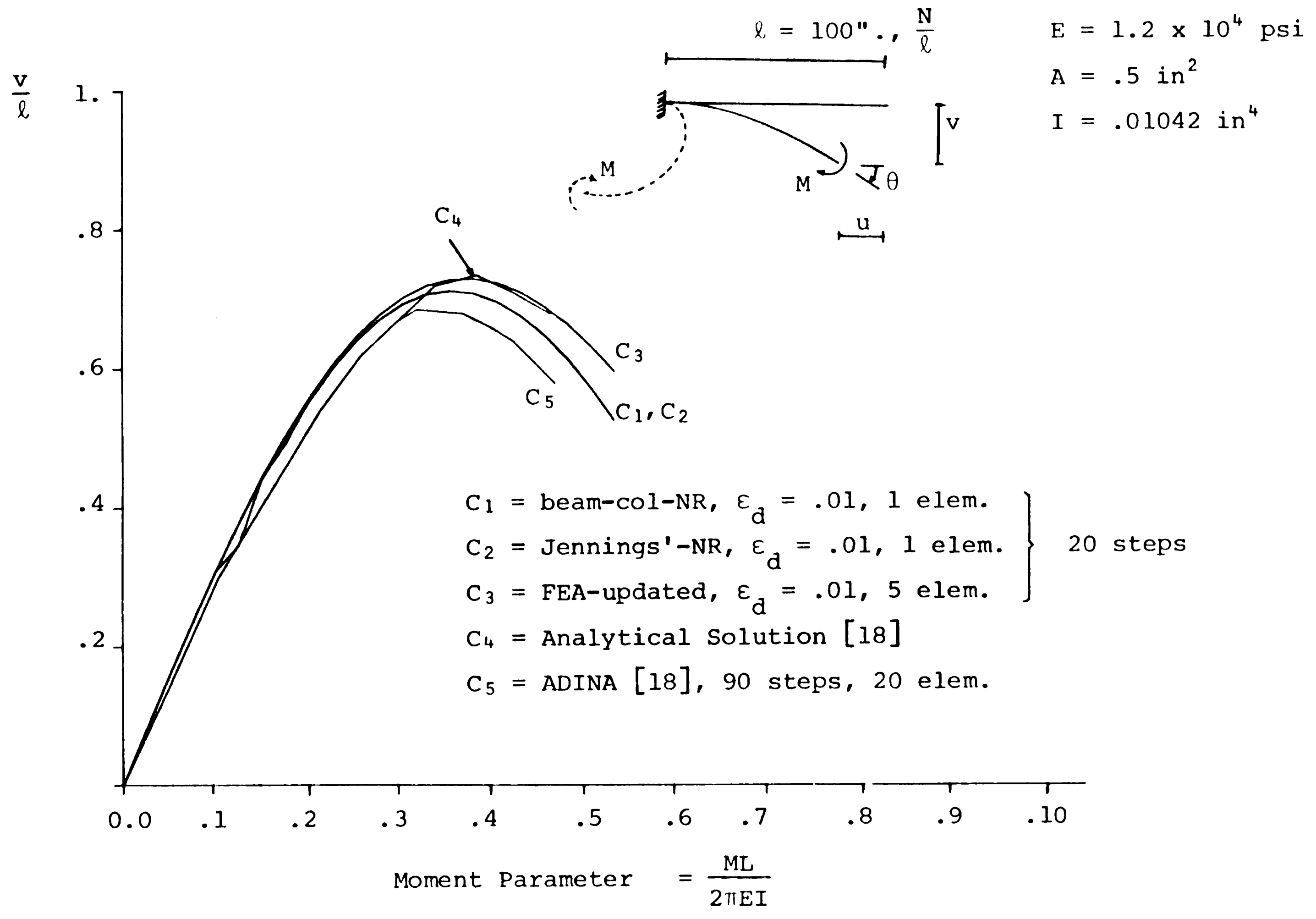


FIGURE 4-6 Comparison of Solutions for Cantilever Beam Subjected to End Moment (v-Component)

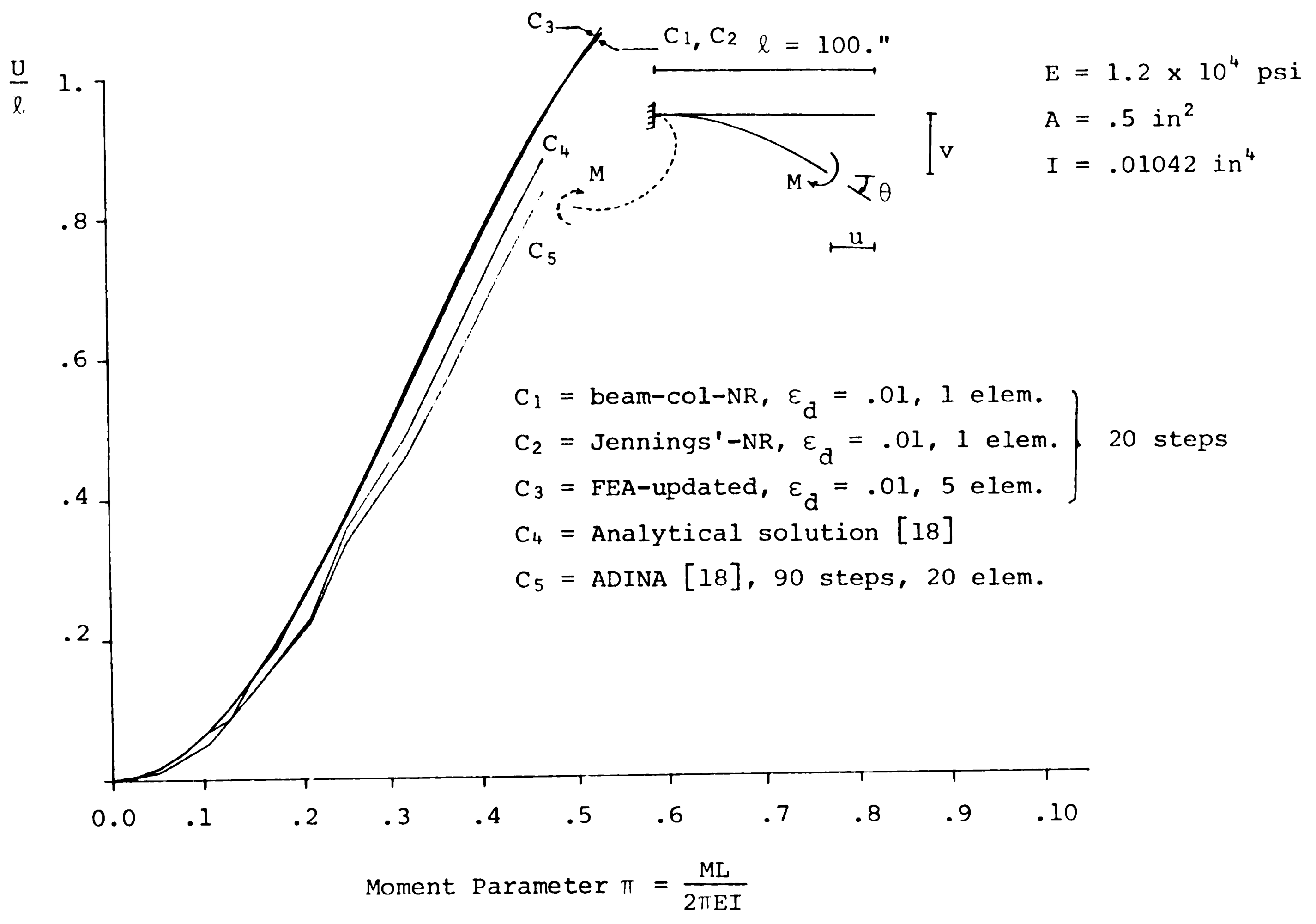


FIGURE 4-7 Comparison of Solutions for Cantilever Beam Subjected to End Moment (u component)

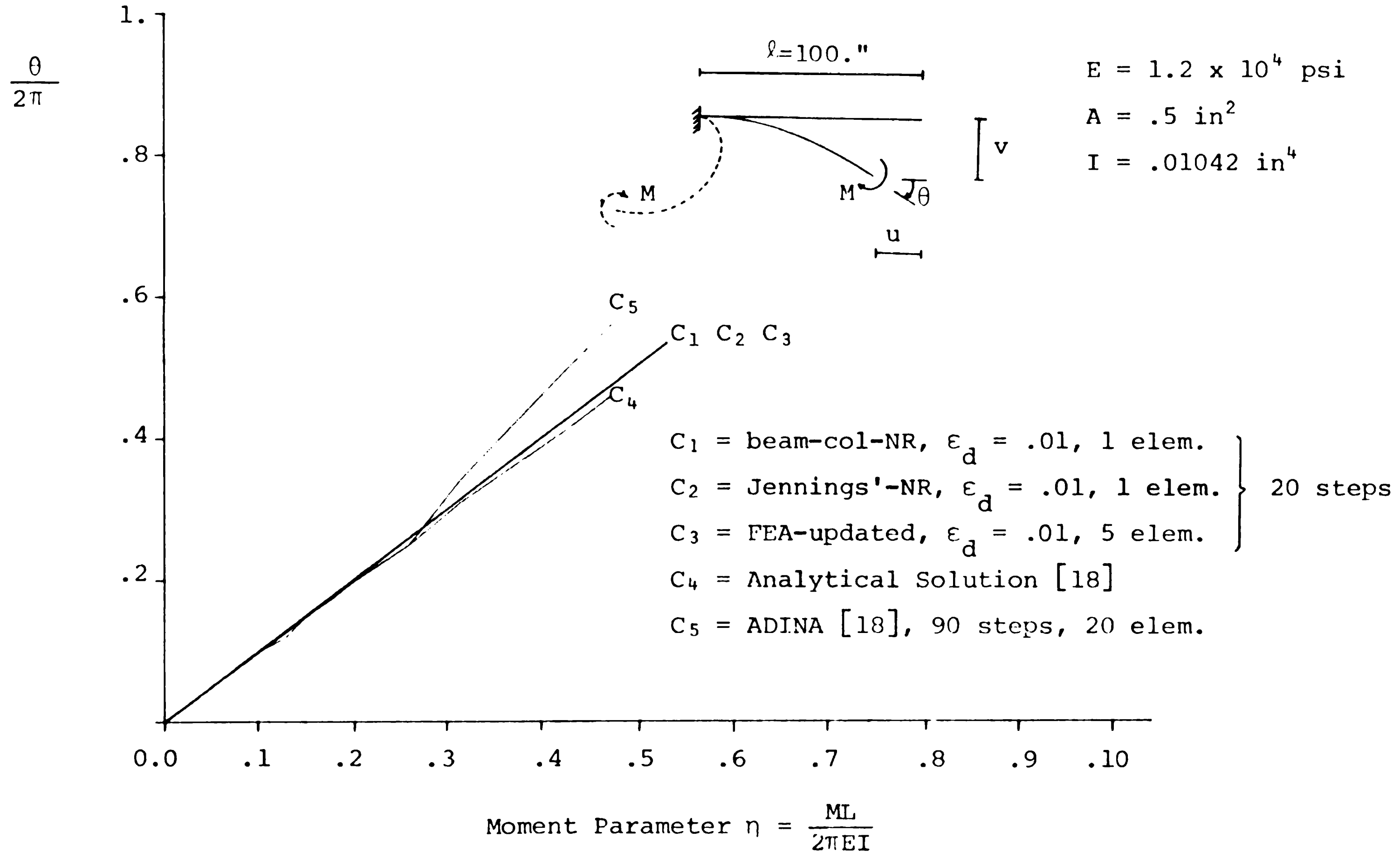


FIGURE 4-8 Comparison of Solutions for Cantilever Beam Subjected to End Moment ($\theta$-Component)

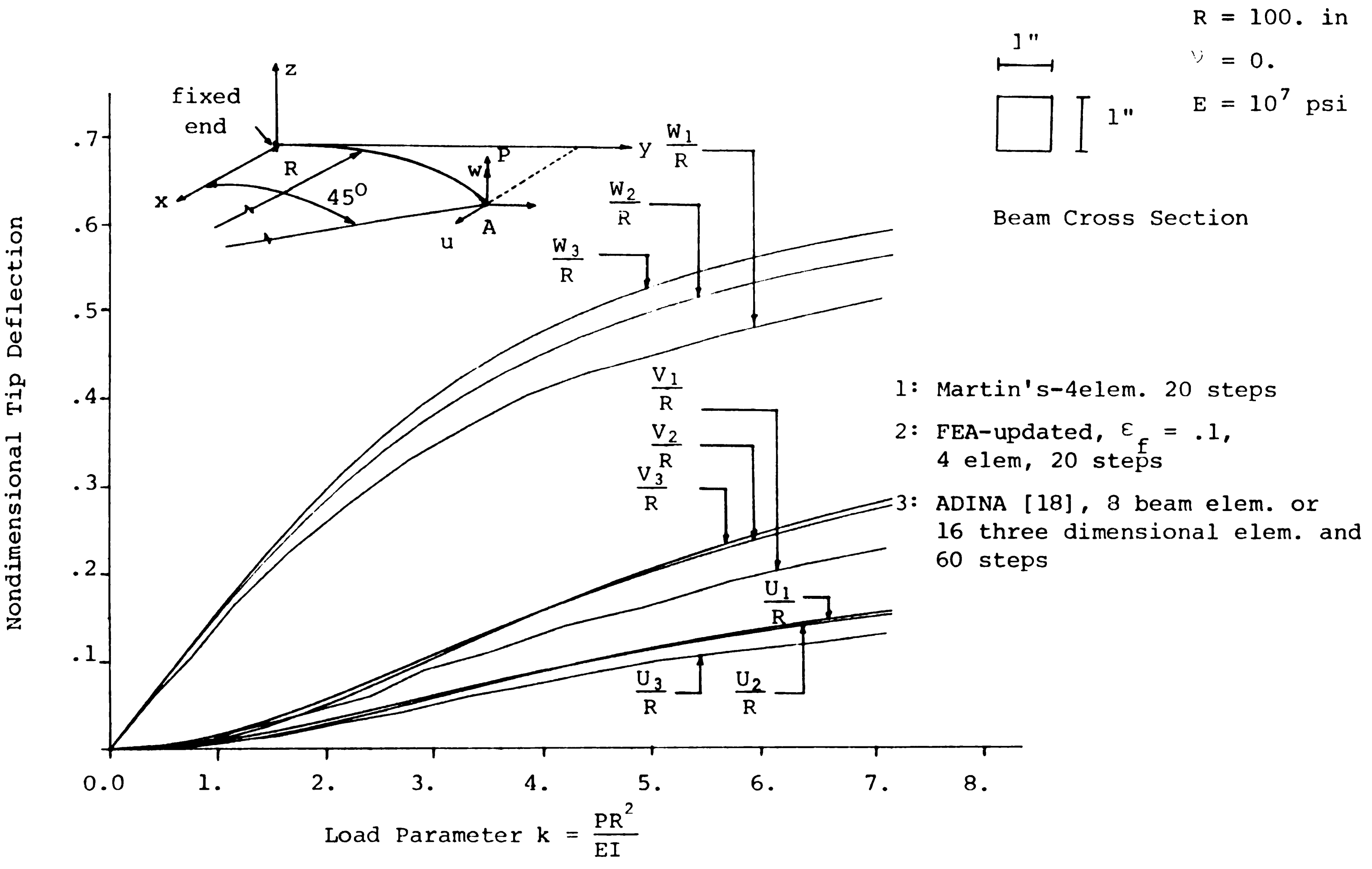


FIGURE 4-9 Comparison of Solutions for A Three Dimensional Beam Curved in Space Subjected to a Lateral Load

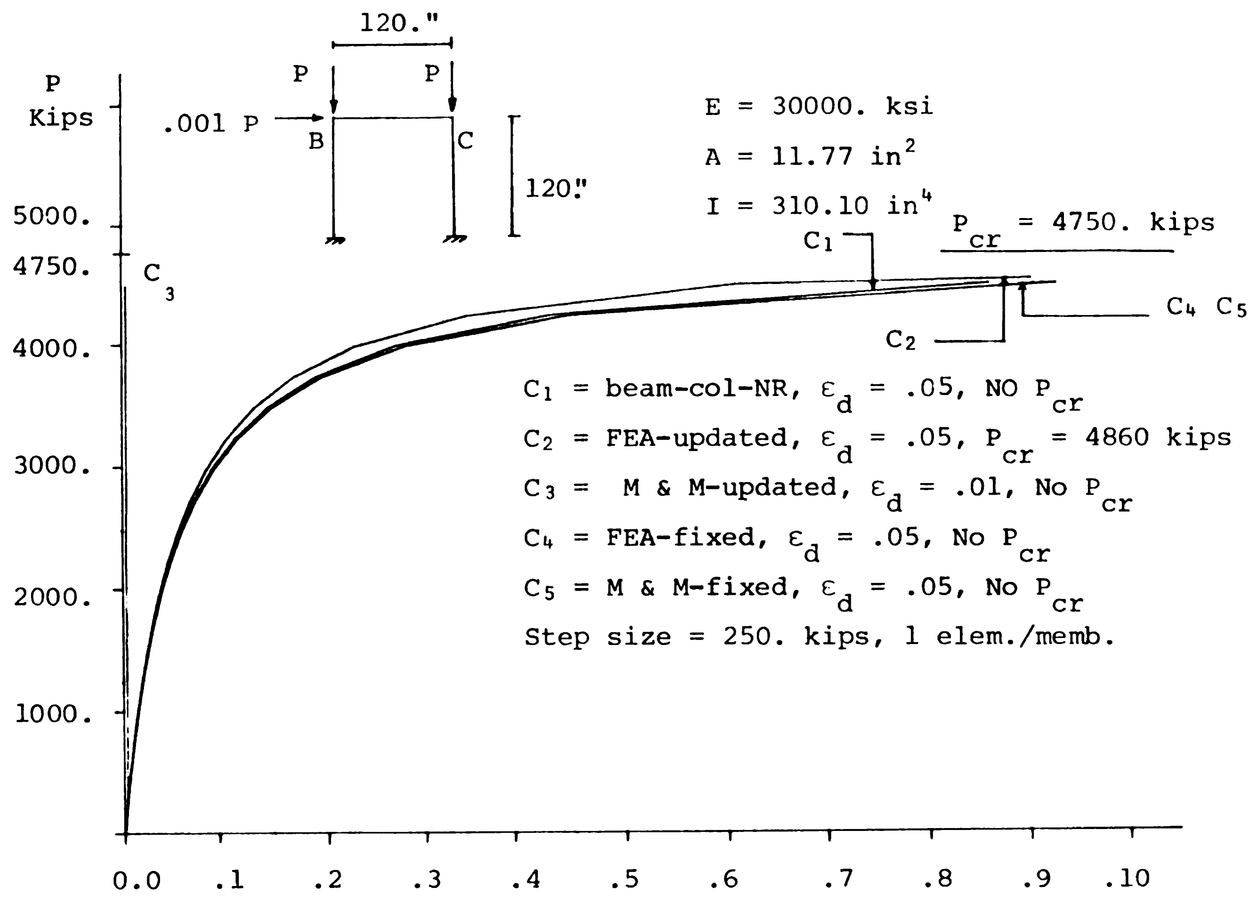


FIGURE 4-10 Comparison of Solutions of A One Span Portal Frame

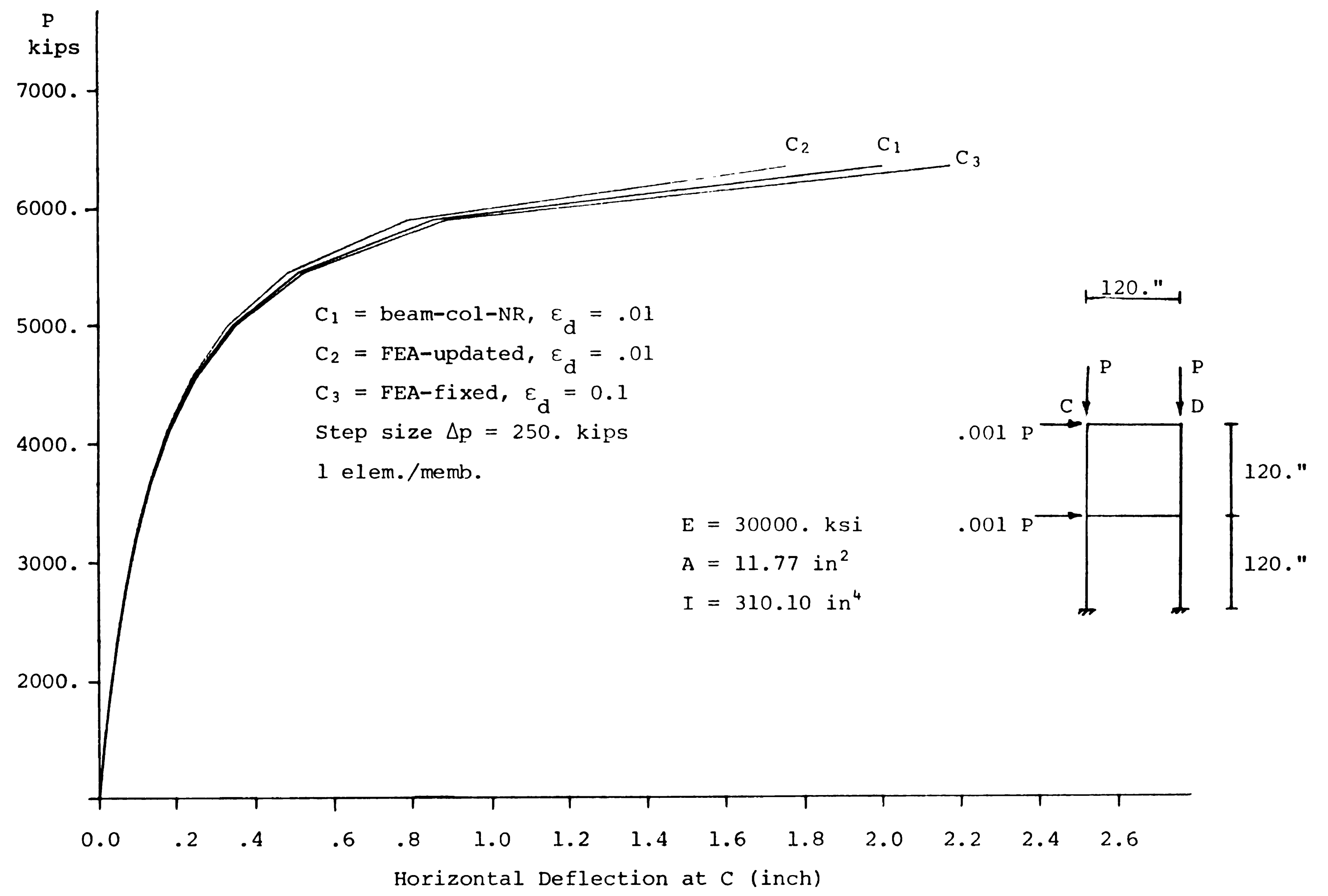


FIGURE 4-11 Comparison of Solutions of A Two-Story Frame

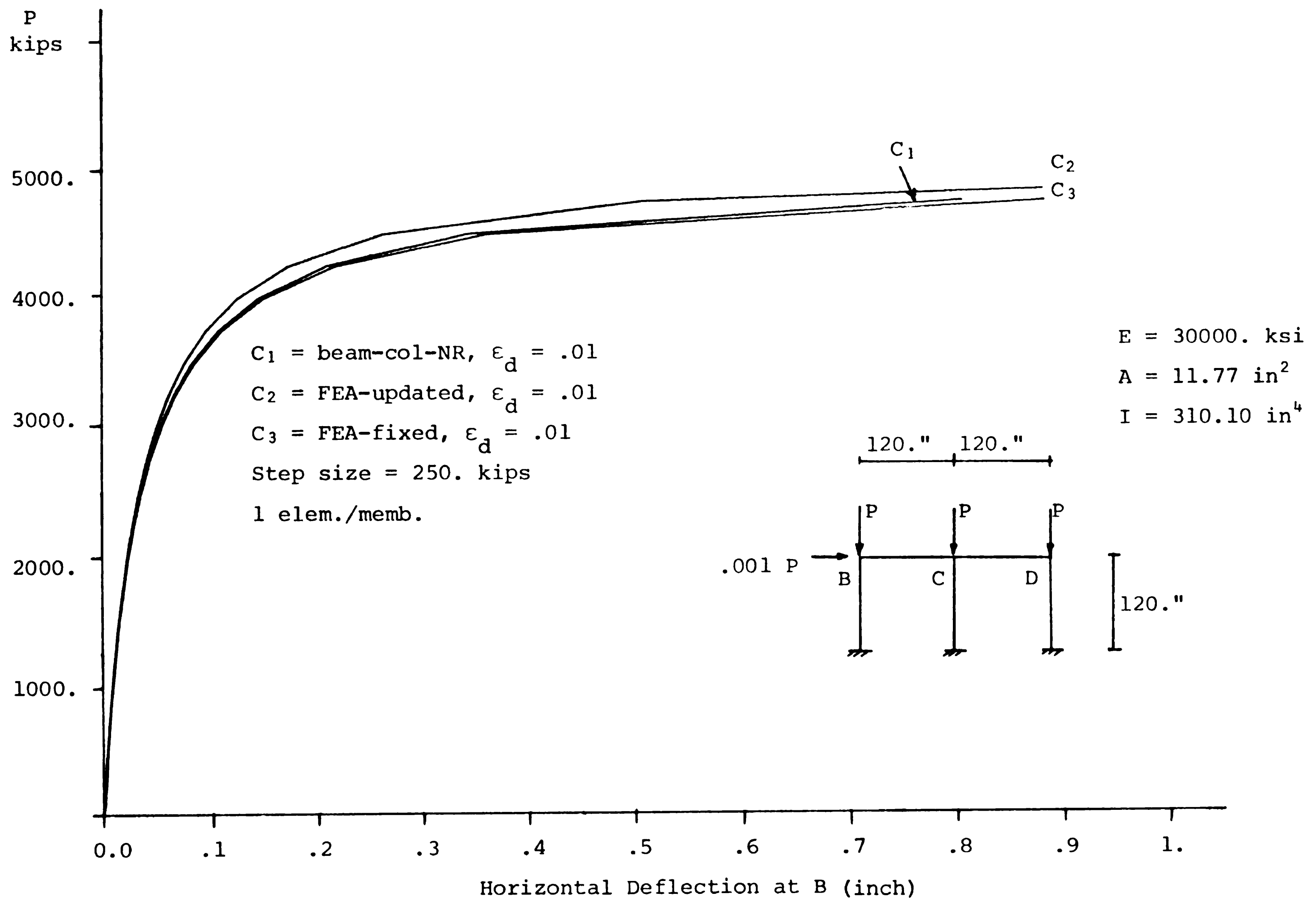


FIGURE 4-12 Comparison of Solutions of A Two Bay Frame

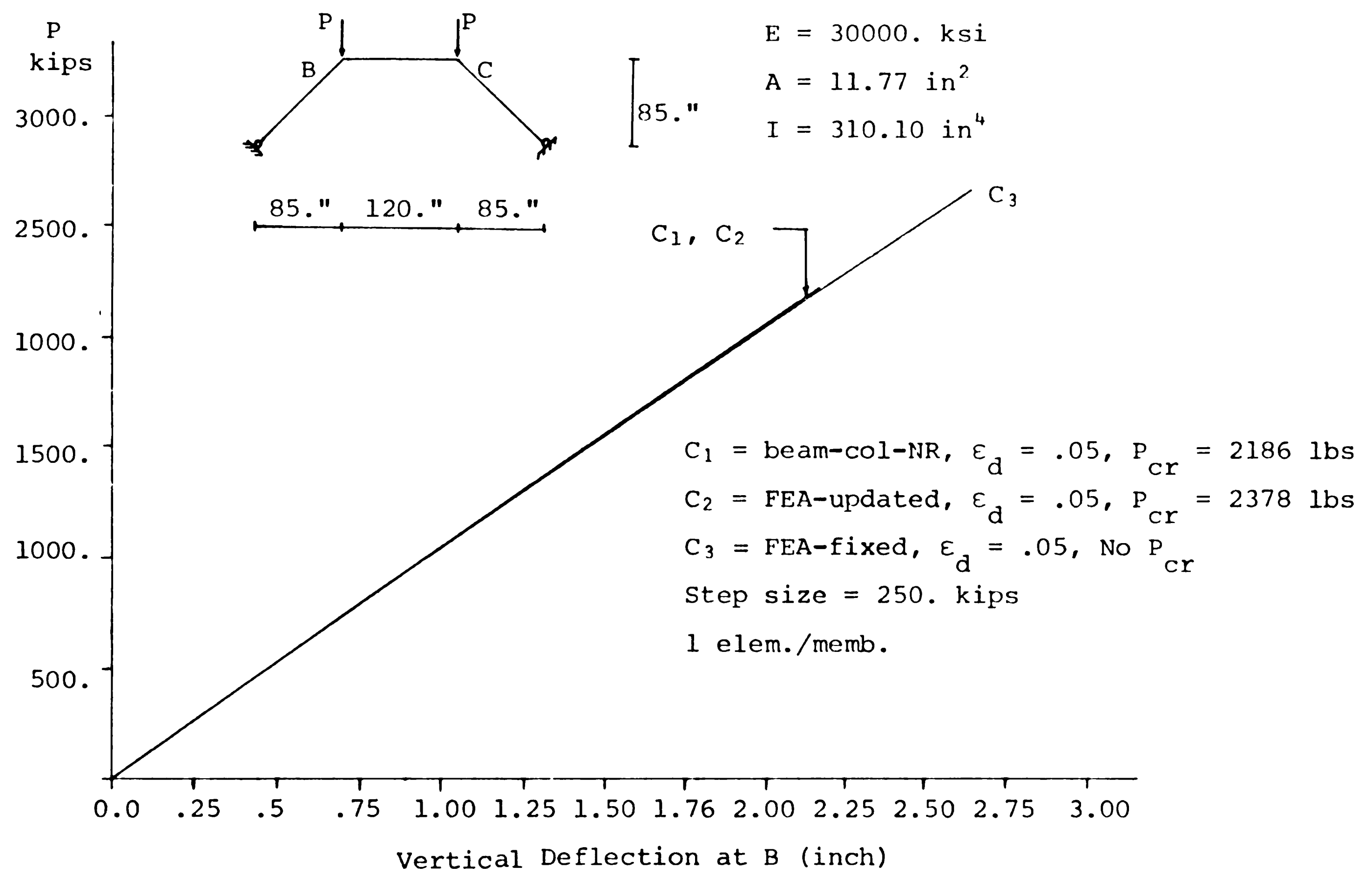


FIGURE 4-13(a) Comparison of Solutions for an Arch Frame (No Horizontal Load)

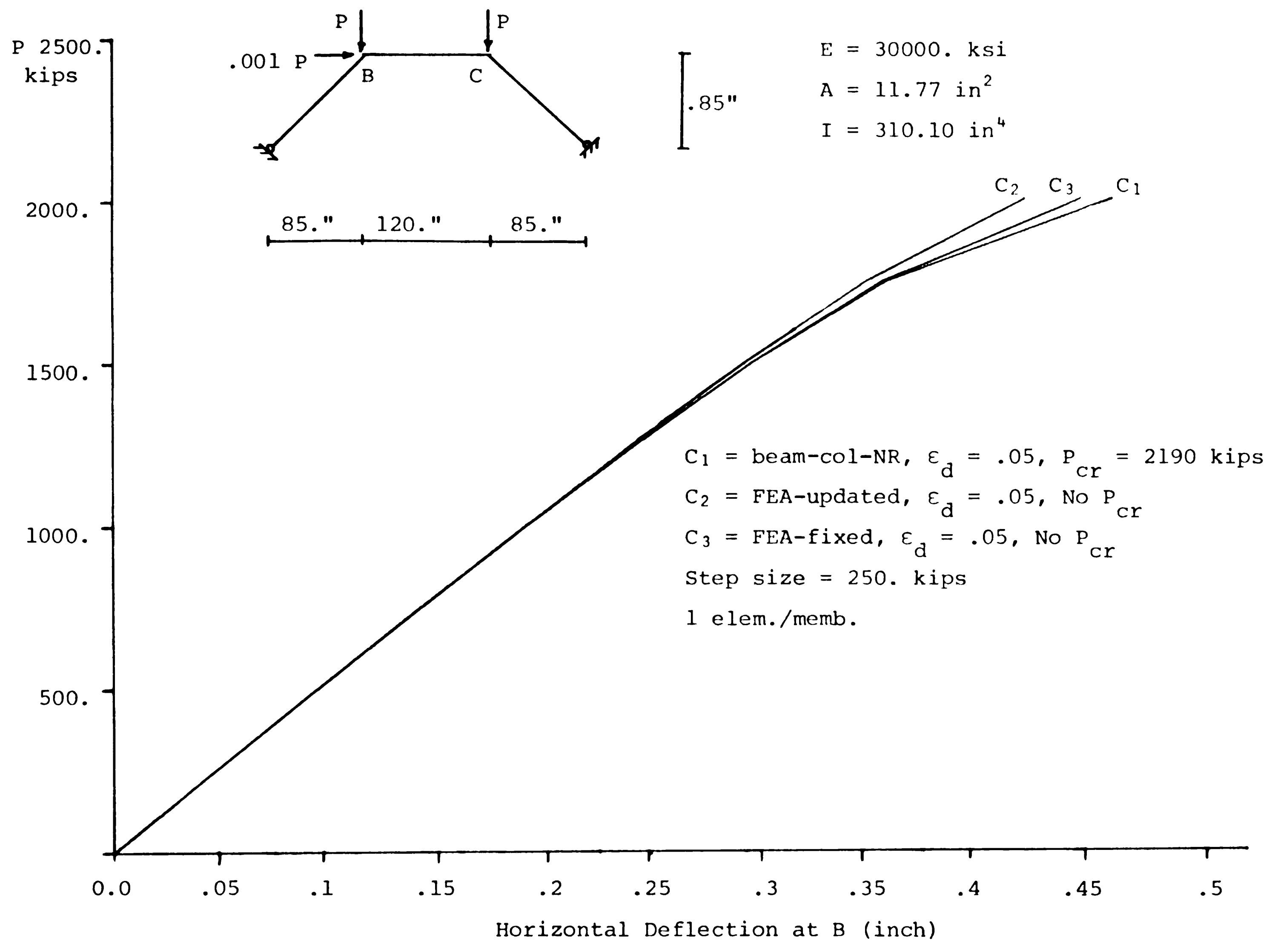
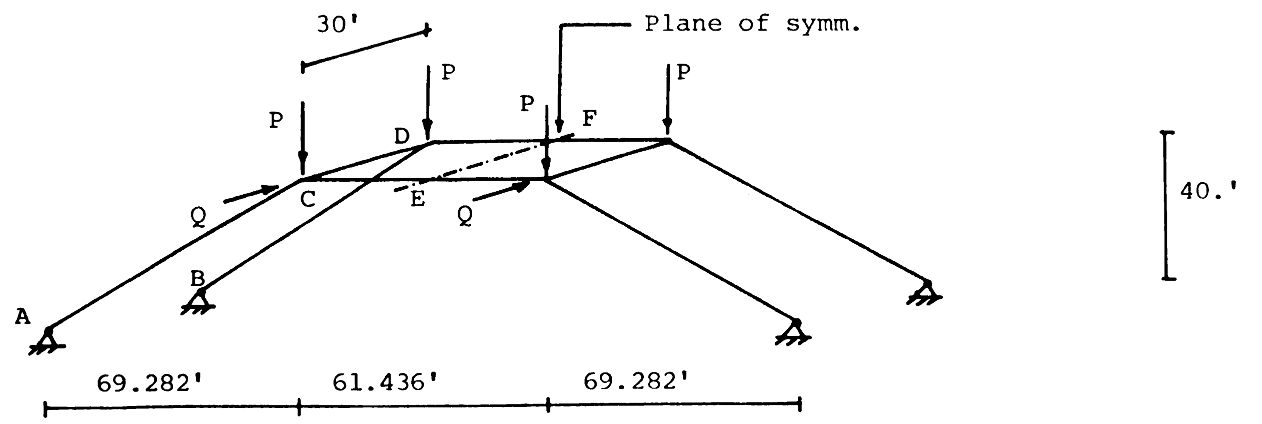
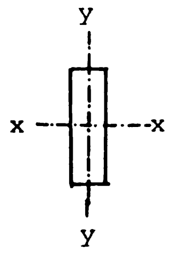


FIGURE 4-13(b) Comparison of Solutions for an Arch Frame (with Horizontal Load)



$Q = .001\ P,\ E = 4320 \times 10^3 \text{ksf}$



$$\left.\begin{array}{l} A = .5\ \text{ft}^2 \\ I_{xx} = .4\ \text{ft}^4 \\ I_{yy} = .133\ \text{ft}^4 \end{array}\right\} \text{For AC,BD,CE,DF}$$

$$\left.\begin{array}{l} A = .1\ \text{ft}^2 \\ I_{xx} = .05\ \text{ft}^4 \\ I_{yy} = .05\ \text{ft}^4 \end{array}\right\} \text{For CD}$$

y-y is in vertical
plane for all members

FIGURE 4-14 Properties of and Loading on A Space Arch Frame

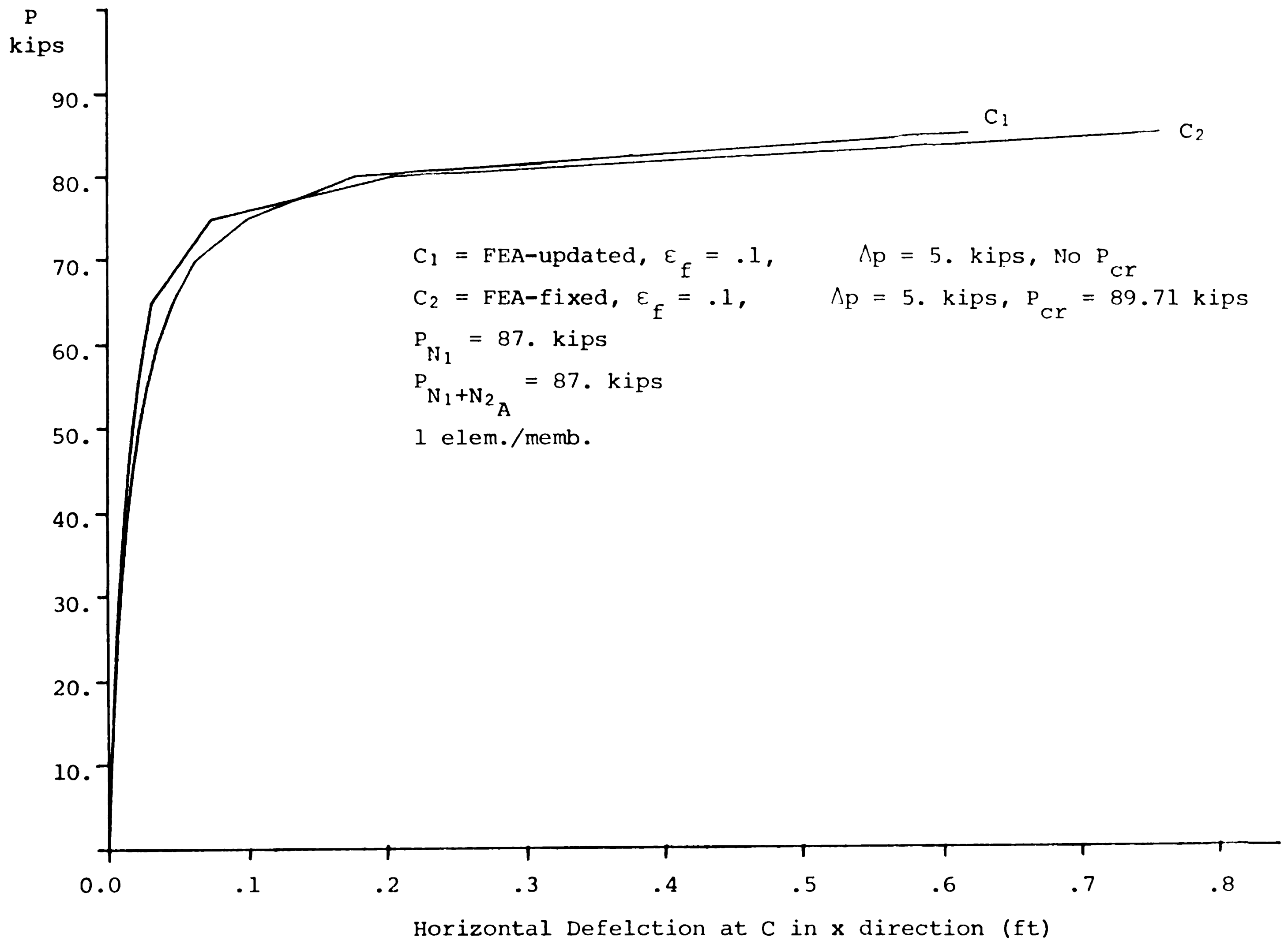


FIGURE 4-15 Comparison of Solutions for A Space Arch Frame

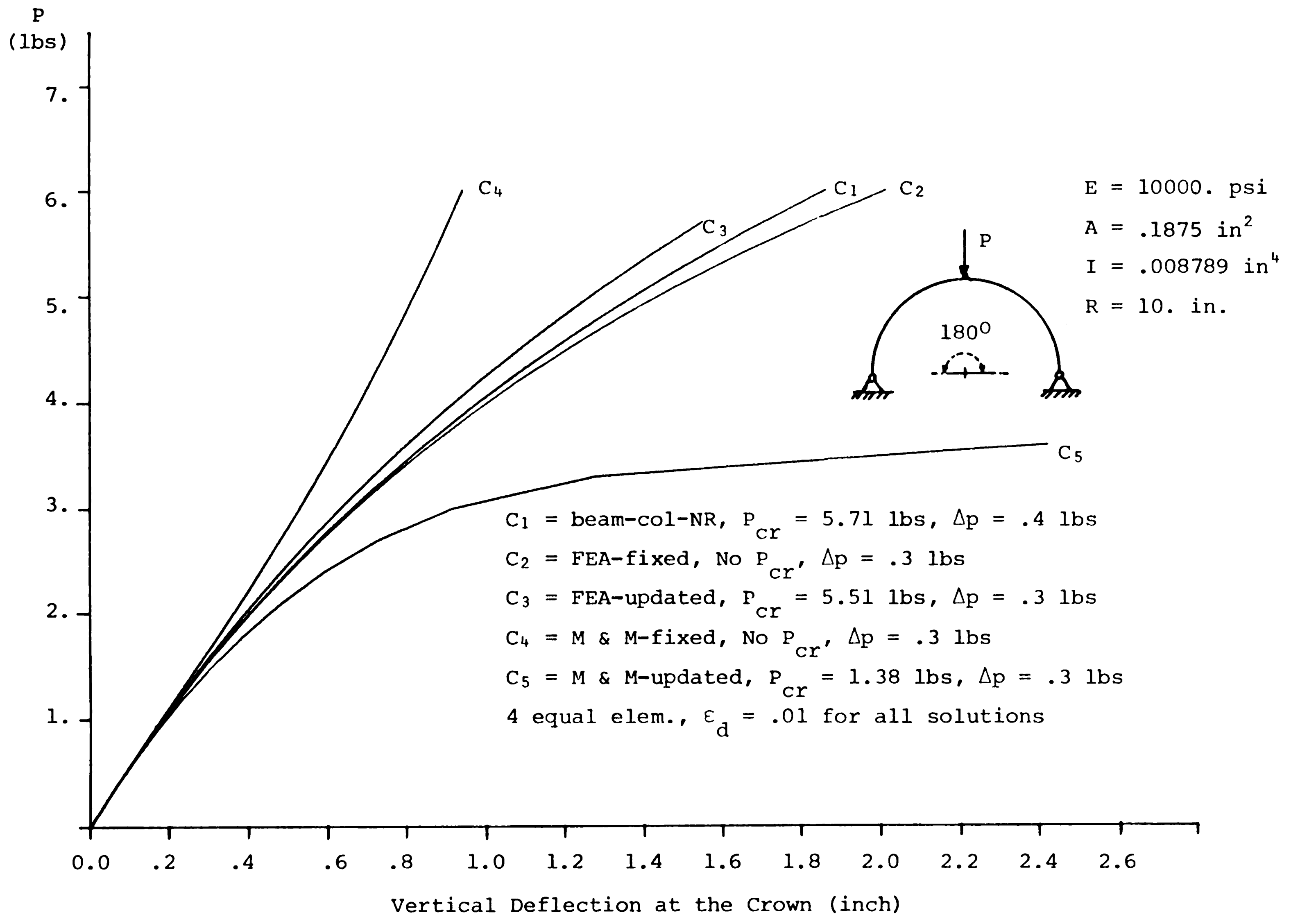


FIGURE 4-16 Solutions of a Half Circular Arch

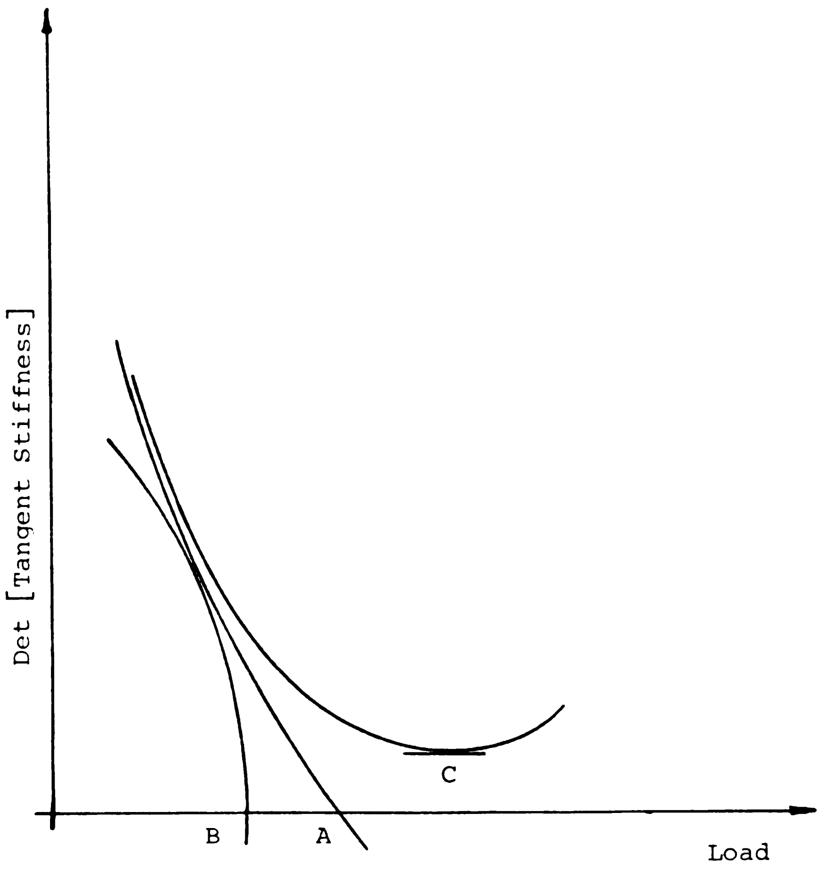


FIGURE 4-17 Load-Determinant Relation for Different Cases of Nonlinear Behavior

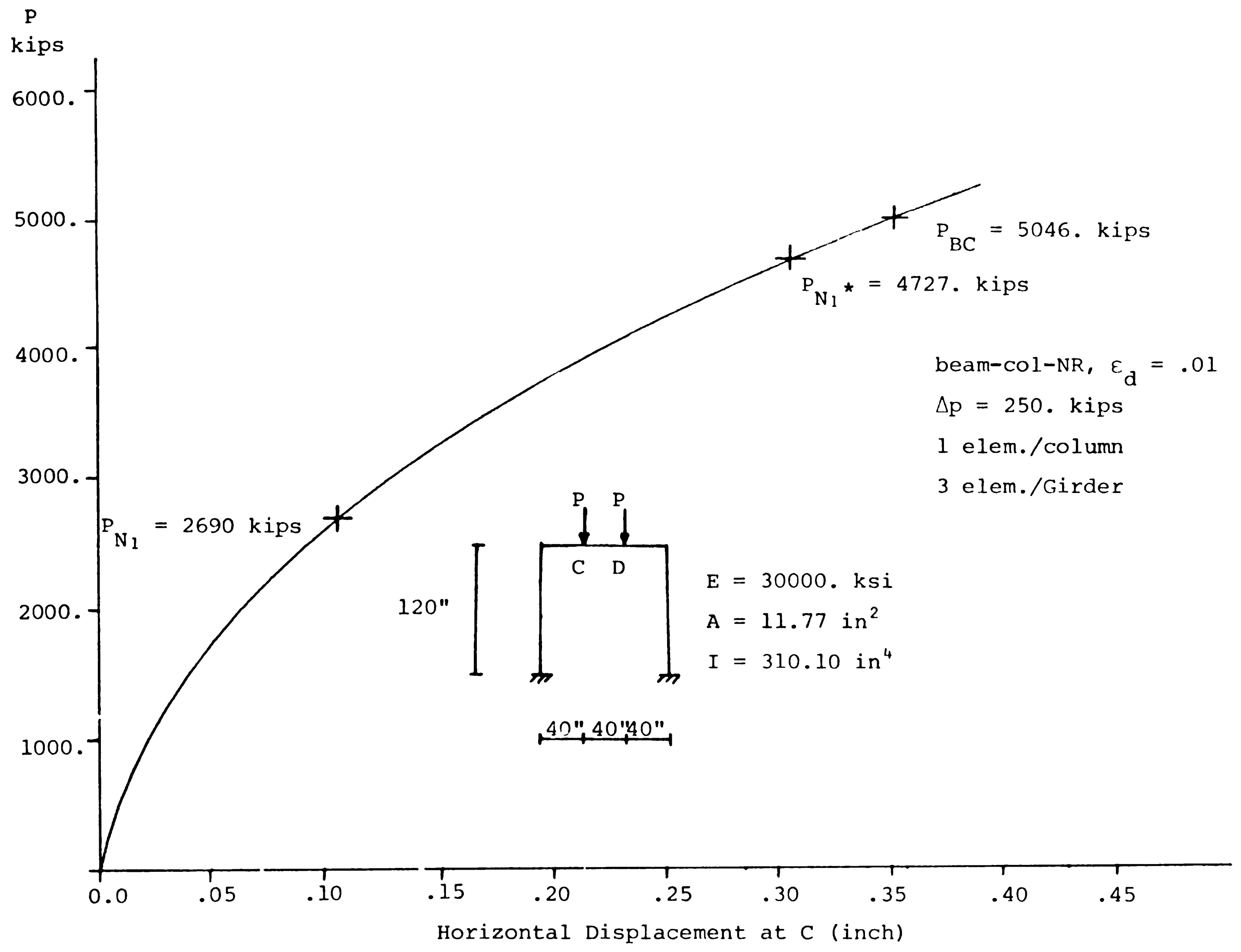


FIGURE 4-18 Load-Displacement and Stability of A Symmetrically Loaded Portal Frame

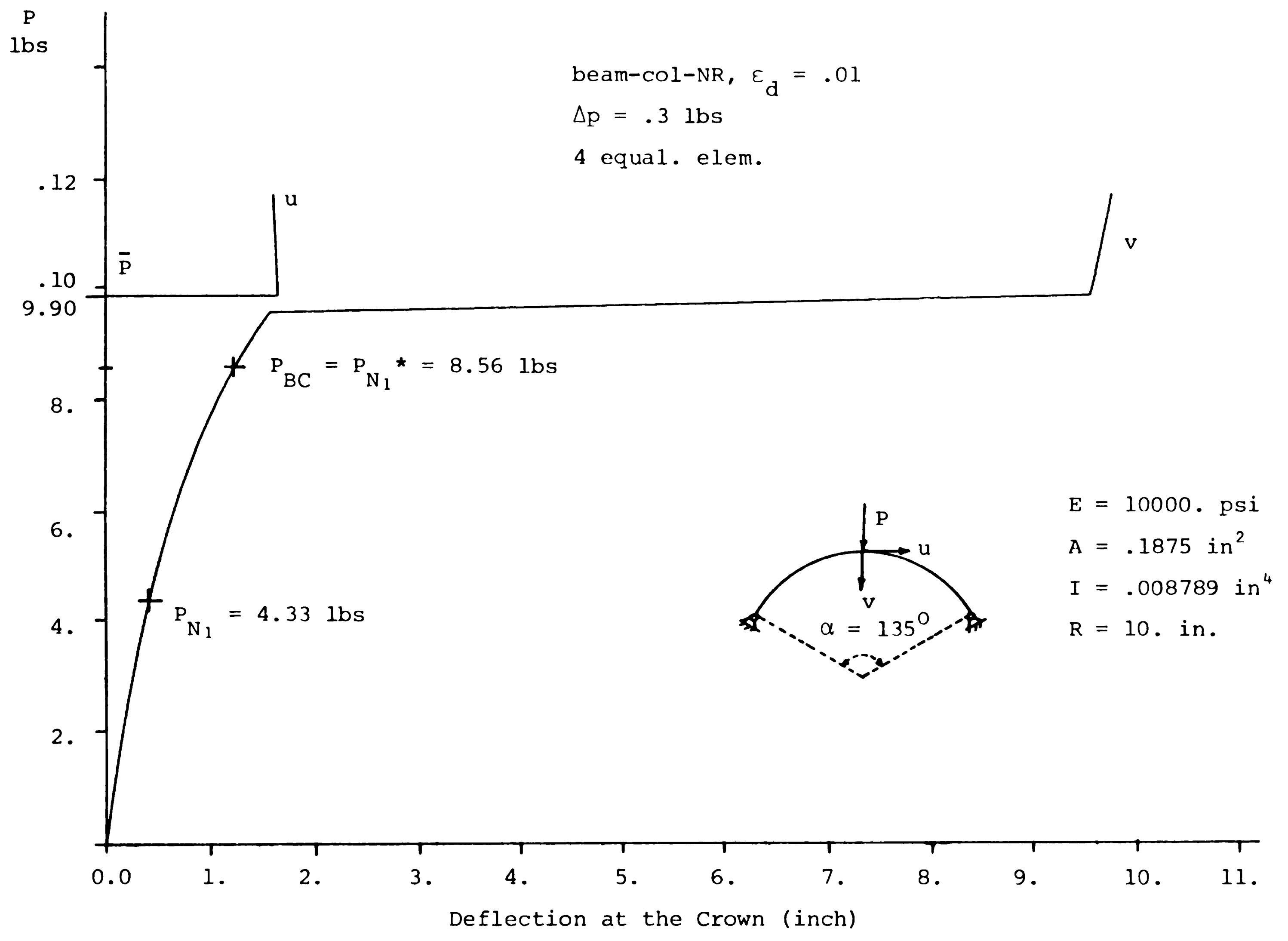


FIGURE 4-19(a) Behavior of a $135^o$-Arch Subjected to A Concentrated Load at the Crown

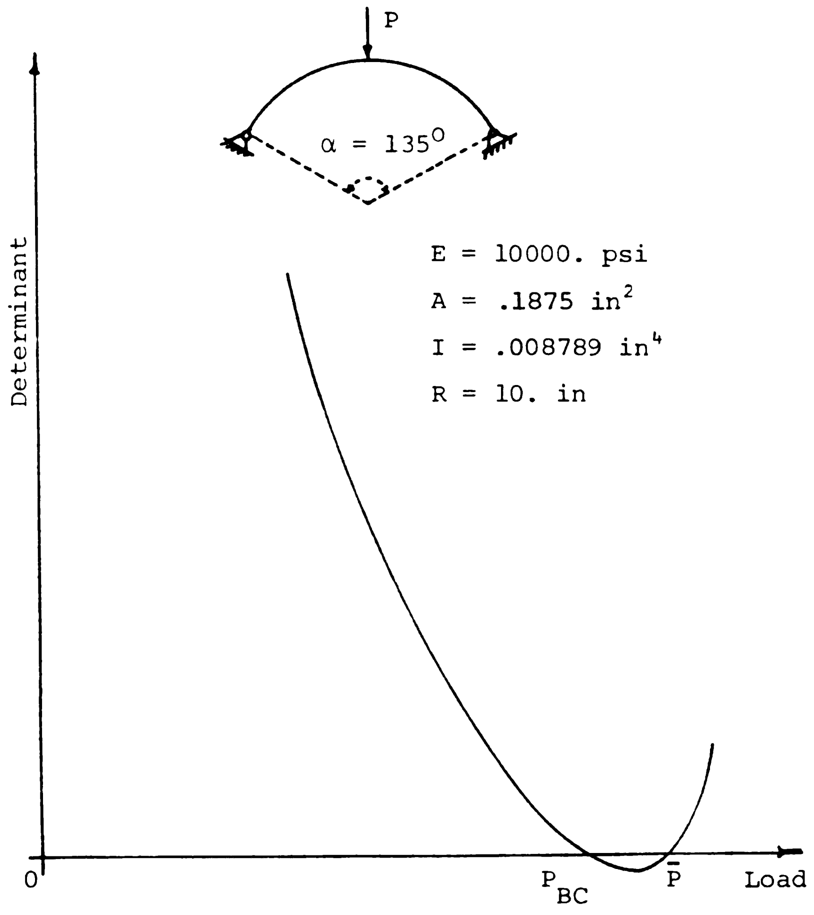
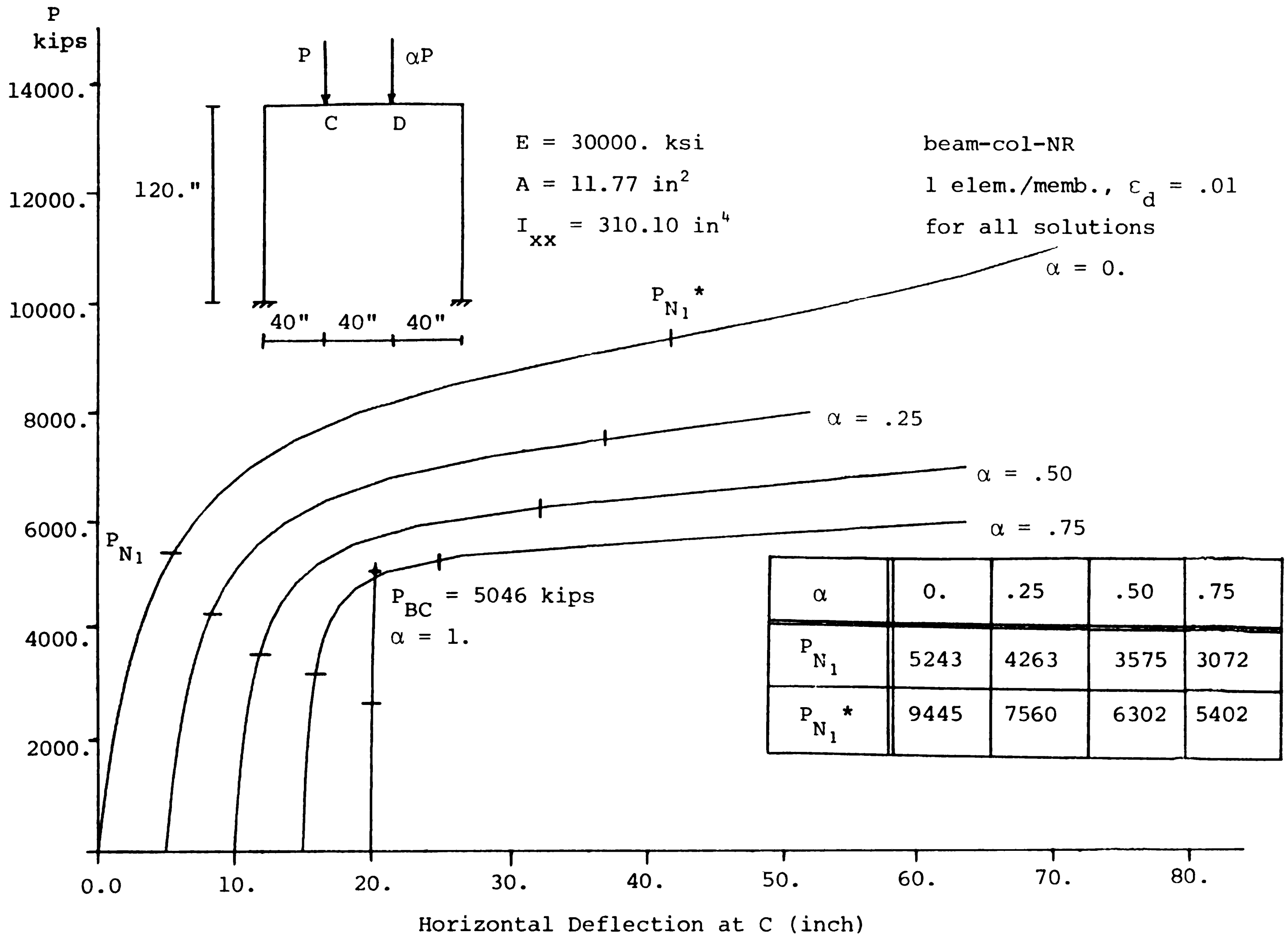


FIGURE 4-19(b) Variation of Determinant of $[K_T]$ with Load



| α | 0. | .25 | .50 | .75 |
|---|---|---|---|---|
| $P_{N_1}$ | 5243 | 4263 | 3575 | 3072 |
| $P_{N_1}$* | 9445 | 7560 | 6302 | 5402 |

FIGURE 4-20 One Story Frame Subjected to Asymmetric Vertical Loads

| $\alpha$ | 1. | .75 | .50 | .25 | 0. |
|---|---|---|---|---|---|
| $P_{N_1}$ | 7.96 | 7.83 | 6.73 | 5.69 | 4.87 |
| $P_{N_1}*$ | 9.67 | 11.05 | 12.89 | 15.46 | 19.29 |

beam-col-NR, $\varepsilon_d$ = .01, 4 equal elem.

$P_{BC}$ = 9.35 lbs for $\alpha$ = 1.

E = 10000. psi

A = .1875 in$^2$

I = .008789 in$^4$

R = 10. in

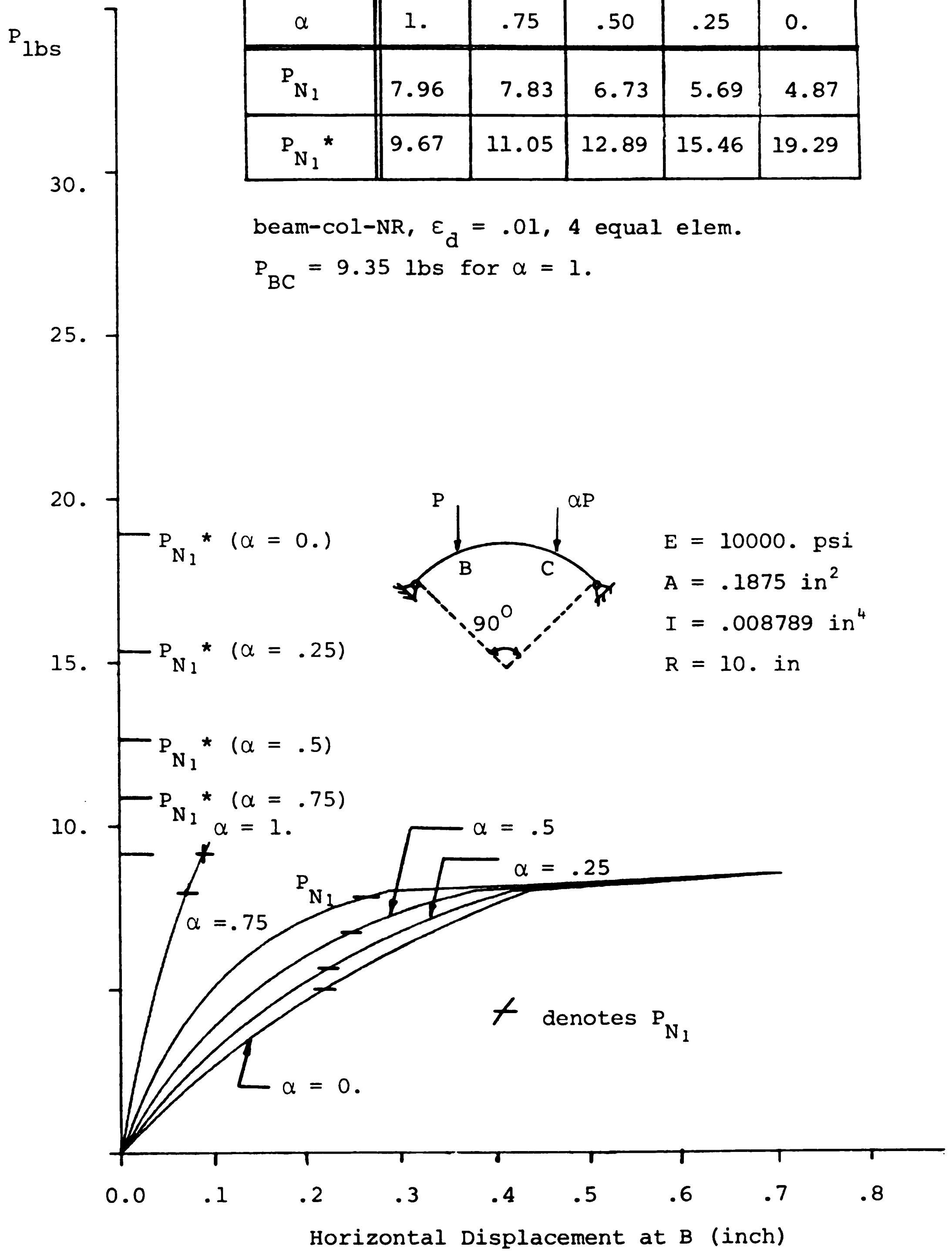


FIGURE 4-21 A 90°-Arch Subjected to Two Vertical Loads

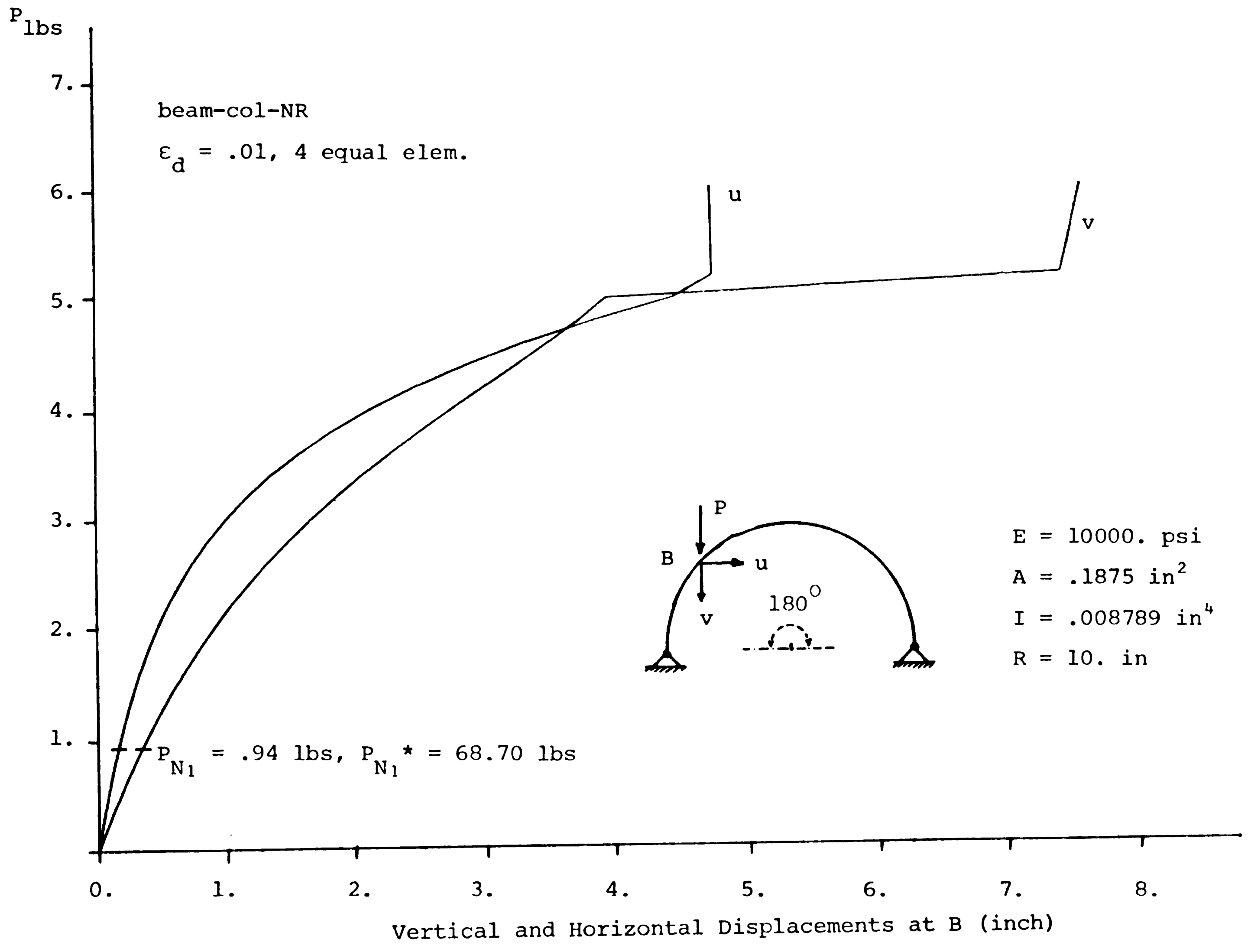


FIGURE 4-22 A Half Circular Arch Subjected to An Asymmetric Loading

## LIST OF REFERENCES

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

2. Bleich, F. "Buckling Strength of Metal Structures," McGraw-Hill Book Co., Inc., New York, N.Y., 1952.

3. Cook. R. D., Concepts and Applications of Finite Element Analysis, John Wiley and Sons, Inc., New York, N.Y., 1974.

4. Haisler, W. E., Stricklin, J. A., and Stebbins, F., "Development and Evaluation of Solution Procedures for Geometrically Nonlinear Structural Analysis by the Direct Stiffness Method," proceedings of AIAA/ASME 12th Structures, Structural Dynamics, and Materials Conference, Anaheim, California, March 1972. Vol. 10, No. 3, pp. 264-272.

5. Saafan, S. A., "Nonlinear Behavior of Structural Plane Frames," Journal of the Structural Division, ASCE, Vol. 89, No. ST4, Proc. Paper 3615, August, 1963, pp. 557-579.

6. Saafan, S. A., "A Theoretical Analysis of Suspension Bridges," Journal of the Structural Division, ASCE, Vol. 92, No. ST4, Proc., Paper 4885, August, 1966, pp. 1-11.

7. Conner, J., Jr., Logcher, R. D., and Chan, S. C., "Nonlinear Analysis of Elastic Framed Structures," Journal of the Structural Division, ASCE, Vol. 94, No. ST6, Proc. Paper 6011, June, 1968, pp. 1525-1547.

8. Oran, C. "Tangent Stiffness in Plane Frames," Journal of the Structural Division, ASCE, Vol. 99, No. ST6, Proc. Paper 9810, June, 1973, pp. 973-985.

9. Oran, C. "Tangent Stiffness in Space Frames," Journal of the Structural Division, ASCE, Vol. 99, No. ST6, Proc. Paper 9813, June, 1973, pp. 987-1001.

10. Kassimali, A. "Nonlinear Static and Dynamic Analysis of Frames". Ph.D. Thesis, Department of Civil Engineering, University of Missouri-Columbia, August, 1976.

11. Martin, H. C., "A Survey of Finite Element Formulation of Geometrically Nonlinear Problems," Recent Advances in Matrix Methods of Structural Analysis and Design, Edited by Gallagher, R. H., Yameda, Y., and Oden, J. T., The University of Alabama Press, 1971, pp. 343-381.

12. Przemieniecki, J. S., Theory of Matrix Structural Analysis, McGraw-Hill Book Co., New York, N.Y., 1968.

13. Jennings, A., "Frame Analysis Including Change of Geometry," Journal of the Structural Division, ASCE, Vol. 94, No. ST3, Proc. Paper 5839, March, 1968, pp. 627-643.

14. Mallet, R. H. and Marcal, P. V., "Finite Element Analysis of Nonlinear Structures," Journal of the Structural Division, ASCE, Vol. 94, No. ST9, Proc. Paper 6115, Sept., 1968, pp. 2081-2105.

15. Powell, G. H., "Theory of Nonlinear Elastic Structures," Journal of the Structural Division, ASCE, Vol. 95, No. ST12, Proc. Paper 6943, Dec., 1969, pp. 2687-2701.

16. Akkoush, E. A., Toridis, T. G., Khozeimeh, K., and Huang, H. K. "Bifurcation, Pre- and Post-Buckling ANalysis of Frame Structures," Computer & Structures, Vol. 8, June 1978, pp. 667-678.

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

18. Bathe, K. J. and Bolourchi, S. "Large Displacement Analysis of Three-Dimensional Beam Structures", International Journal for Numerical Methods in Engineering, Vol. 14, April 1979, pp. 961-986.

19. Ebner, A. M. and Ucciferro, J. J., "A Theoretical and Numerical Comparison of Elastic Nonlinear Finite Element Methods," Computers and Structures, Vol. 2 Nos. 5/6, 1972, pp. 1043-1061.

20. Wen, R. K., Unpublished Research Notes on Work Done Under NSF Grant Eng-7822478, College of Engineering, Michigan State University, 1980.

21. Bathe, K. J., and Wilson, E. L., Numerical Methods in Finite Element Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

22. Hurty, W. C., and Rubinstein, M. F., Dynamics of Structures, Prentice-Hall, Inc., 1964.

23. Lange, J. S. "Elastic Buckling of Arches by Finite Element Method" Ph.D. Thesis, Department of Civil and Sanitary Engineering, Michigan State University, 1980.

24. Frisch-Fay, R., "A New Approach to the Analysis of the Deflection of Thin Cantilevers," Journal of Applied Mechanics, Transactions of the American Society of Mechanical Engineers, Vol. 28, Series E, March, 1961, pp. 87-90.

25. Oran, C. and Kassimali, A., "Large Deformations of Framed Structures Under Static and Dynamic Loads", Computer and Structures, Vol. 6, 1976, pp. 539-547.

26. Gere, J. M. and Weaver, W.J.,"Analysis of Framed Structures", D. Van Nostrand Company, 1965, page 291 - Figure 4-46.

# APPENDIX A

## MATRICES [k], $[n_1]$, AND $[n_2]$

All matrices contained in this appendix and Appendices B and C are symmetric. Only non-zero entries are given here.

## A.1 [k] MATRIX

### A.1.1 THREE DIMENSIONAL

$$k\,(1,7) = -k(1,1) \quad k(1,1) = k(7,7) = -k(1,7) = \frac{EA}{\ell}$$

$$k(2,2) = k(8,8) = \frac{12EI_\eta}{\ell^3}$$

$$k(2,8) = -k(2,2)$$

$$k(3,3) = k(9,9) = \frac{12EI_\zeta}{\ell^3}$$

$$k(3,9) = -k(303)$$

$$k(10,10) = k(4,4) = \frac{GJ}{\ell}$$

$$k(4,10) = -k(10,10)$$

$$k(6,6) = k(12,12) = \frac{4EI_\eta}{\ell}$$

$$k(5,5) = k(11,11) = \frac{4EI_\zeta}{\ell}$$

$$k(2,12) = k(2,6) = \frac{6EI_\eta}{\ell^2}$$

$$k(6,8) = k(8,12) = -k(2,12)$$

$$k(3,11) = k(3,5) = \frac{-6EI_\zeta}{\ell^2}$$

$$k(5,9) = k(9,11) = -k(3,11)$$

$$k(6,12) = \frac{2EI_\eta}{\ell}$$

$$k(5,11) = \frac{2EI_\zeta}{\ell}$$

A.1.2 TWO DIMENSIONAL

$$k(1,1) = k(4,4) = \frac{EA}{\ell}$$

$$k(1,4) = -k(1,1)$$

$$k(2,2) = k(5,5) = \frac{12EI_\eta}{\ell^3}$$

$$k(2,5) = -k(2,2)$$

$$k(2,6) = k(2,3) = \frac{6EI_\eta}{\ell^2}$$

$$k(3,5) = k(5,6) = -k(2,3)$$

$$k(6,6) = k(3,3) = \frac{4EI_\eta}{\ell}$$

$$k(3,6) = \frac{2EI_\eta}{\ell}$$

A.2 $\underline{[n_1]\text{ MATRIX}}$

A.2.1 THREE DIMENSIONAL

$$n_1(1,2) = n_1(7,8) = \frac{-F_4}{10\ell}\,EA$$

$$n_1(1,8) = n_1(2,7) = -n_1(1,2)$$

$$n_1(2,2) = n_1(3,3) = n_1(8,8) = n_1(9,9) = \frac{6(u_2-u_1)}{5\ell^2}\,EA$$

$$n_1(2,8) = n_1(3,9) = -n_1(2,2)$$

$$n_1(5,5) = n_1(6,6) = n_1(11,11) = n_1(12,12) = \frac{2(u_2-u_1)}{15}\,EA$$

$$n_1(1,3) = n_1(7,9) = \frac{G_4}{10\ell}\,EA$$

$$n_1(1,9) = n_1(3,7) = -n_1(1,3)$$

$$n_1(1,6) = \frac{-F_{51}}{30}\,EA$$

$$n_1(6,7) = -n_1(1,6)$$

$$n_1(1,5) = \frac{-G_{51}}{30}\,EA$$

$$n_1(5,7) = -n_1(1,5)$$

$$n_1(1,12) = \frac{-F_{52}}{30}\,EA$$

$$n_1(7,12) = -n_1(1,12)$$

$$n_1(1,11) = \frac{-G_{52}}{30}\,EA$$

$$n_1(7,11) = -n_1(1,11)$$

$$n_1(2,6) = n_1(2,12) = n_1(5,9) = n_1(9,11) = \frac{u_2-u_1}{10\ell}\,EA$$

$$n_1(3,5) = n_1(3,11) = n_1(8,12) = n_1(6,8) = -n_1(2,6)$$

$$n_1(5,11) = n_1(6,12) = -\,\frac{u_2-u_1}{30}\,EA$$

in which:

$$F_4 = \theta_1+\theta_2-12\theta_o$$

$$G_4 = \Psi_1+\Psi_2-12\Psi_o$$

$$F_{51} = 4\theta_1-\theta_2-3\theta_o$$

$$F_{52} = 4\theta_2-\theta_1-3\theta_o$$

$$G_{51} = 4\Psi_1-\Psi_2-3\Psi_o$$

$$G_{52} = 4\Psi_2-\Psi_1-3\Psi_o$$

$$\theta_o = \frac{v_2-v_1}{\ell}$$

$$\Psi_o = \frac{w_1-w_2}{\ell}$$

## A.2.2 TWO DIMENSIONAL

$$n_1(4,5) = n_1(1,2) = [-\frac{\theta_1+\theta_2}{10} + \frac{6(v_2-v_1)}{5\ell}]\frac{EA}{\ell}$$

$$n_1(2,4) = n_1(1,5) = -n_1(1,2)$$

$$n_1(1,3) = [\theta_2-4\theta_1+3(\frac{v_2-v_1}{\ell})]\frac{EA}{30}$$

$$n_1(1,6) = [\theta_1-4\theta_2+3(\frac{v_2-v_1}{\ell})]\frac{EA}{30}$$

$$n_1(4,6) = -n_1(1,6)$$

$$n_1(3,4) = -n_1(1,3)$$

$$n_1(5,5) = n_1(2,2) = 6(u_2-u_1)\frac{EA}{5\ell^2}$$

$$n_1(2,5) = -n_1(2,2)$$

$$n_1(2,3) = n_1(2,6) = (u_2-u_1)\frac{EA}{10\ell}$$

$$n_1(3,5) = n_1(5,6) = -n_1(2,3)$$

$$n_1(3,3) = n_1(6,6) = 2(u_2-u_1)\frac{EA}{15}$$

$$n_1(3,6) = -(u_2-u_1)\frac{EA}{30}$$

## A.3 $[n_2]$ MATRIX

### A.3.1 THREE DIMENSIONAL BASED ON QUARTIC STRAIN FUNCTION

$$n_2(2,2) = n_2(8,8) = (6B_3-6B_4+2B_5)\frac{EA}{\ell}$$

$$n_2(2,8) = -n_2(2,2)$$

$$n_2(3,3) = n_2(9,9) = (6B_8-6B_9+2B_{10})\frac{EA}{\ell}$$

$$n_2(3,9) = -n_2(3,3)$$

$$n_2(2,3) = (6B_{13}-6B_{14}+2B_{15})\frac{EA}{\ell}$$

$$n_2(2,9) = -n_2(2,3)$$

$$n_2(2,6) = (-3B_2+10B_3-7B_4+2B_5)\frac{EA}{2}$$

$$n_2(6,8) = -n_2(2,6)$$

$$n_2(3,6) = n_2(5,8) = (-3B_{12}+10B_{13}-7B_{14}+2B_{15})\frac{EA}{2}$$

$$n_2(6,9) = n_2(2,5) = -n_2(3,6)$$

$$n_2(6,6) = (B_1-4B_2+\frac{22}{3}B_3-4B_4+B_5)\frac{EA\ell}{2}$$

$$n_2(5,5) = (B_6-4B_7+\frac{22}{3}B_8-4B_9+B_{10})\frac{EA\ell}{2}$$

$$n_2(12,12) = (\frac{4}{3}B_3-2B_4+B_5)\frac{EA\ell}{2}$$

$$n_2(11,11) = (\frac{4}{3}B_8-2B_9+B_{10})\frac{EA\ell}{2}$$

$$n_2(3,5) = (3B_7-10B_8+7B_9-2B_{10})\frac{EA}{2}$$

$$n_2(5,9) = -n_2(3,5)$$

$$n_2(5,6) = -\left(B_{11}-4B_{12}+\frac{22}{3}B_{13}-4B_{14}+B_{15}\right)\frac{EA\ell}{2}$$

$$n_2(3,8) = -\left(6B_{13}-6B_{14}+2B_{15}\right)\frac{EA}{\ell}$$

$$n_2(8,9) = -n_2(3,8)$$

$$n_2(2,12) = \left(4B_3-5B_4+2B_5\right)\frac{EA}{2}$$

$$n_2(8,12) = -n_2(2,12)$$

$$n_2(3,12) = n_2(8,11) = \left(4B_{13}-5B_{14}+2B_{15}\right)\frac{EA}{2}$$

$$n_2(9,12) = n_2(2,11) = -n_2(3,12)$$

$$n_2(3,11) = \left(-4B_8+5B_9-2B_{10}\right)\frac{EA}{2}$$

$$n_2(9,11) = -n_2(3,11)$$

$$n_2(6,12) = \left(-B_2+\frac{11}{3}B_3-3B_4+B_5\right)\frac{EA\ell}{2}$$

$$n_2(6,11) = n_2(5,12) = \left(B_{12}-\frac{11}{3}B_{13}+3B_{14}-B_{15}\right)\frac{EA\ell}{2}$$

$$n_2(5,11) = \left(-B_7+\frac{11}{3}B_8-3B_9+B_{10}\right)\frac{EA\ell}{2}$$

$$n_2(11,12) = \left(\frac{-4}{3}B_{13}+2B_{14}-B_{15}\right)\frac{EA\ell}{2}$$

in which:

$$B_1 = \frac{1}{5}\left[2(\theta_1{}^2+\theta_2{}^2)+18\theta_0{}^2-3\theta_0(\theta_1+\theta_2)-\theta_1\theta_2-\frac{68}{3}\Psi_1{}^2+\frac{2}{3}\Psi_2{}^2\right.$$

$$\left.+\,6\Psi_0{}^2+39\Psi_1\Psi_0-\Psi_2\Psi_0-17\Psi_1\Psi_2\right]$$

$$B_2 = \frac{1}{5}\left(\theta_1{}^2+3\theta_2{}^2+18\theta_0{}^2-6\theta_0\theta_1-\theta_1\theta_2-104\Psi_1{}^2+\Psi_2{}^2+6\Psi_0{}^2+58\Psi_1\Psi_0-76\Psi_1\Psi_2\right)$$

$$B_3 = \frac{1}{35}(6\theta_1^2+27\theta_2^2+108\theta_0^2-45\theta_0\theta_1+18\theta_0\theta_2-9\theta_1\theta_2-292\Psi_1^2+9\Psi_2^2$$

$$+36\Psi_0^2+489\Psi_0\Psi_1+6\Psi_0\Psi_2-213\Psi_1\Psi_2)$$

$$B_4 = \frac{33}{140}\theta_1^2 + \frac{29}{28}\theta_2^2 - \frac{1107}{7}\theta_0^2 - \frac{9}{5}\theta_0\theta_1 + \frac{9}{5}\theta_0\theta_2 - \frac{33}{70}\theta_1\theta_2$$

$$- \frac{1949}{140}\Psi_1^2 + \frac{13}{28}\Psi_2^2 + \frac{9}{7}\Psi_0^2 + \frac{117}{5}\Psi_0\Psi_1 + \frac{339}{7}\Psi_0\Psi_2 - \frac{711}{70}\Psi_1\Psi_2$$

$$B_5 = \frac{9}{35}\theta_1^2 + \frac{27}{14}\theta_2^2 + \frac{27}{7}\theta_0^2 - \frac{27}{14}\theta_0\theta_1 + \frac{27}{14}\theta_0\theta_2 - \frac{9}{14}\theta_1\theta_2$$

$$- \frac{627}{35}\Psi_1^2 + \frac{9}{14}\Psi_2^2 + \frac{9}{7}\Psi_0^2 + \frac{423}{14}\Psi_0\Psi_1 + \frac{9}{14}\Psi_0\Psi_2 - \frac{183}{14}\Psi_1\Psi_2$$

$$B_6 = \frac{1}{5}\left[2(\Psi_1^2+\Psi_2^2)+18\Psi_0^2-3\Psi_0(\Psi_1+\Psi_2)-\Psi_1\Psi_2- \frac{68}{3}\theta_1^2+ \frac{2}{3}\theta_2^2\right.$$

$$\left.+ 6\theta_0^2+39\theta_1\theta_0-\theta_2\theta_0-17\theta_1\theta_2\right]$$

$$B_7 = \frac{1}{5}(\Psi_1^2+3\Psi_2^2+18\Psi_0^2-6\Psi_0\Psi_1-\Psi_1\Psi_2-104\theta_1^2+\theta_2^2+6\theta_0^2+58\theta_1\theta_0-76\theta_1\theta_2)$$

$$B_8 = \frac{1}{35}(6\Psi_1^2+27\Psi_2^2+108\Psi_0^2-45\Psi_0\Psi_1+18\Psi_0\Psi_2-9\Psi_1\Psi_2-292\theta_1^2+9\theta_2^2$$

$$+36\theta_0^2+489\theta_0\theta_1+6\theta_0\theta_2-213\theta_1\theta_2)$$

$$B_9 = \frac{33}{140}\Psi_1^2 + \frac{39}{28}\Psi_2^2 - \frac{1107}{7}\Psi_0^2 - \frac{9}{5}\Psi_0\Psi_1 + \frac{9}{5}\Psi_0\Psi_2 - \frac{33}{70}\Psi_1\Psi_2$$

$$- \frac{1949}{140}\theta_1^2 + \frac{13}{28}\theta_2^2 + \frac{9}{7}\theta_0^2 + \frac{117}{5}\theta_0\theta_1 + \frac{339}{7}\theta_0\theta_2 - \frac{711}{70}\theta_1\theta_2$$

$$B_{10} = \frac{9}{35}\Psi_1^2 + \frac{27}{14}\Psi_2^2 + \frac{27}{7}\Psi_0^2 - \frac{27}{14}\Psi_0\Psi_1 + \frac{27}{14}\Psi_0\Psi_2 - \frac{9}{14}\Psi_1\Psi_2$$

$$+ \frac{627}{35}\theta_1^2 + \frac{9}{14}\theta_2^2 + \frac{9}{7}\theta_0^2 + \frac{423}{14}\theta_0\theta_1 + \frac{9}{14}\theta_0\theta_2 - \frac{183}{14}\theta_1\theta_2$$

$$B_{11} = \frac{-4}{15}\,\theta_1\Psi_1 + \frac{\theta_1\Psi_0}{5} + \frac{\theta_1\Psi_2}{15} + \frac{\Psi_1\theta_0}{5} + \frac{\theta_2\Psi_1}{5} - \frac{12}{5}\,\theta_0\Psi_0 + \frac{1}{5}\,\theta_0\Psi_2$$

$$+ \frac{1}{5}\,\theta_2\Psi_0 - \frac{4}{15}\,\theta_2\Psi_2$$

$$B_{12} = \frac{-2}{15}\,\theta_1\Psi_1 + \frac{2}{5}\,\theta_1\Psi_0 + \frac{1}{15}\,\theta_1\Psi_2 + \frac{2}{5}\,\Psi_1\theta_0 + \frac{1}{15}\,\theta_2\Psi_1 - \frac{12}{5}\,\theta_0\Psi_0 - \frac{2}{5}\,\theta_2\Psi_2$$

$$B_{13} = \frac{-4}{35}\,\theta_1\Psi_1 + \frac{3}{7}\,\theta_1\Psi_0 + \frac{3}{35}\,\theta_1\Psi_2 + \frac{3}{7}\,\Psi_1\theta_0 + \frac{3}{35}\,\theta_2\Psi_1 - \frac{72}{35}\,\theta_0\Psi_0$$

$$- \frac{6}{35}\,\theta_0\Psi_2 - \frac{6}{35}\,\theta_2\Psi_0 - \frac{18}{35}\,\theta_2\Psi_2$$

$$B_{14} = \frac{-11}{70}\,\theta_1\Psi_1 + \frac{3}{5}\,\theta_1\Psi_0 + \frac{11}{70}\,\theta_1\Psi_2 + \frac{3}{5}\,\Psi_1\theta_0 + \frac{11}{70}\,\theta_2\Psi_1 - \frac{18}{7}\,\theta_0\Psi_0$$

$$- \frac{3}{7}\,\theta_0\Psi_2 - \frac{3}{7}\,\theta_2\Psi_0 - \frac{13}{14}\,\theta_2\Psi_2$$

$$B_{15} = \frac{-6}{35}\,\theta_1\Psi_1 + \frac{9}{14}\,\theta_1\Psi_0 + \frac{3}{14}\,\theta_1\Psi_2 + \frac{9}{14}\,\Psi_1\theta_0 + \frac{3}{14}\,\theta_2\Psi_1 - \frac{18}{7}\,\theta_0\Psi_0$$

$$- \frac{9}{14}\,\theta_0\Psi_2 - \frac{9}{14}\,\theta_2\Psi_0 - \frac{9}{7}\,\theta_2\Psi_2$$

## A.3.2 TWO DIMENSIONAL BASED ON QUARTIC STRAIN FUNCTION

$$n_2(3,3) = [12\ell\theta_1^2 + \ell\theta_2^2 - 3\ell\theta_1\theta_2 + \frac{18}{\ell}\,(v_2-v_1)^2 + 3(v_2-v_1)\,(\theta_1-\theta_2)]\,\frac{EA}{140}$$

$$n_2(2,3) = [-3\theta_1^2 + 3\theta_2^2 + 6\theta_1\theta_2 + \frac{108}{\ell^2}\,(v_2-v_1)^2 - \frac{72}{\ell}\,\theta_1(v_2-v_1)]\,\frac{EA}{280}$$

$$n_2(3,5) = -n_2(2,3)$$

$$n_2(3,6) = [-3\ell\theta_1^2 - 3\ell\theta_2^2 + 4\ell\theta_1\theta_2 - 6(v_2-v_1)(\theta_1+\theta_2)]\,\frac{EA}{280}$$

$$n_2(6,6) = [\ell\theta_1^2 + 12\ell\theta_2^2 - 3\ell\theta_1\theta_2 + \frac{18}{\ell}\,(v_2-v_1)^2 + 3(v_2-v_1)(\theta_2-\theta_1)]\,\frac{EA}{140}$$

$$n_2(2,6) = [3\theta_1^2 - 3\theta_2^2 + 6\theta_1\theta_2 + \frac{108}{\ell^2}\,(v_2-v_1)^2 - \frac{72}{\ell}\,\theta_2(v_2-v_1)]\,\frac{EA}{280}$$

$n_2(5,6) = -n_2(2,6)$

$$n_2(5,5) = n_2(2,2) = [\frac{18}{\ell}\,\theta_1{}^2 + \frac{18}{\ell}\,\theta_2{}^2 + \frac{432}{\ell^3}\,(v_2-v_1)^2 - \frac{108}{\ell^2}\,(v_2-v_1)(\theta_1+\theta_2)]\,\frac{EA}{140}$$

$n_2(2,5) = -n_2(2,2)$

A.3.3 THREE DIMENSIONAL BASED ON AVERAGE STRAIN

$$n_2(2,2) = n_2(8,8) = (\frac{F_1}{100} + \frac{G_2}{25})\,\frac{EA}{\ell}$$

$n_2(2,8) = -n_2(2,2)$

$$n_2(2,3) = n_2(8,9) = -F_4G_4\,\frac{EA}{100\ell}$$

$n_2(2,9) = n_2(3,8) = -n_2(2,3)$

$$n_2(3,3) = n_2(9,9) = (\frac{G_1}{100} + \frac{F_2}{25})\,\frac{EA}{\ell}$$

$n_2(3,9) = -n_2(3,3)$

$$n_2(2,6) = (F_{31} + G_2)\,\frac{EA}{300}$$

$n_2(6,8) = -n_2(2,6)$

$$n_2(2,5) = F_4G_{51}\,\frac{EA}{300}$$

$$n_2(3,6) = -G_4F_{51}\,\frac{EA}{300}$$

$n_2(6,9) = -n_2(3,6)$

$$n_2(3,5) = -(G_{31} + F_2)\,\frac{EA}{300}$$

$n_2(5,9) = -n_2(3,5)$

$n_2(6,6) = (\frac{F_{61}}{300} + \frac{G_2}{225})\ EA\ell$

$n_2(5,6) = F_{51}G_{51}\frac{EA\ell}{900}$

$n_2(5,5) = (\frac{G_{61}}{300} \div \frac{F_2}{225})\ EA\ell$

$n_2(5,8) = -F_4G_{51}\ \frac{EA}{300}$

$n_2(2,12) = (F_{32} + G_2)\ \frac{EA}{300}$

$n_2(8,12) = -n_2(2,12)$

$n_2(2,11) = F_4G_{52}\ \frac{EA}{300}$

$n_2(8,11) = -n_2(2,11)$

$n_2(3,12) = -G_4F_{52}\ \frac{EA}{300}$

$n_2(9,12) = -n_2(3,12)$

$n_2(3,11) = -\ (G_{32} + F_2)\ \frac{EA}{300}$

$n_2(9,11) = -n_2(3,11)$

$n_2(6,12) = (F_7 - \frac{G_2}{3})\ \frac{EA\ell}{300}$

$n_2(6,11) = F_{51}G_{52}\ \frac{EA\ell}{900}$

$n_2(5,12) = F_{52}G_{51}\ \frac{EA\ell}{900}$

$n_2(5,11) = (G_7 - \frac{F_2}{3})\ \frac{EA\ell}{300}$

$n_2(12,12) = (\frac{F_{62}}{300} + \frac{G_2}{225})\ EA\ell$

$n_2(11,12) = F_{52}G_{52}\frac{EA\ell}{900}$

$n_2(11,11) = (\frac{G_{62}}{300} + \frac{F_2}{225})\ EA\ell$

in which:

$F_1 = 9\theta_1{}^2+9\theta_2{}^2-2\theta_1\theta_2-36\theta_1\theta_0-36\theta_2\theta_0+216\theta_0{}^2$

$G_1 = 9\Psi_1{}^2+9\Psi_2{}^2-2\Psi_1\Psi_2-36\Psi_1\Psi_0-36\Psi_2\Psi_0+216\Psi_0{}^2$

$F_2 = 2\theta_1{}^2+2\theta_2{}^2-\theta_1\theta_2-3\theta_1\theta_0-3\theta_2\theta_0+18\theta_0{}^2$

$G_2 = 2\Psi_1{}^2+2\Psi_2{}^2-\Psi_1\Psi_2-3\Psi_1\Psi_0-3\Psi_2\Psi_0+18\Psi_0{}^2$

$F_4 = \theta_1+\theta_2 -12\theta_0$

$G_4 = \Psi_1+\Psi_2 -12\Psi_0$

$F_7 = -2\theta_1{}^2-2\theta_2{}^2+6\theta_1\theta_2-2\theta_1\theta_0-2\theta_2\theta_0-3\theta_0{}^2$

$G_7 = -2\Psi_1{}^2-2\Psi_2{}^2+6\Psi_1\Psi_2-2\Psi_1\Psi_0-2\Psi_2\Psi_0-3\Psi_0{}^2$

$F_{31} = 6\theta_1{}^2+\theta_2{}^2+2\theta_1\theta_2-54\theta_1\theta_0+6\theta_2\theta_0+54\theta_0{}^2$

$F_{32} = 6\theta_2{}^2+\theta_1{}^2+2\theta_1\theta_2-54\theta_2\theta_0+6\theta_1\theta_0+54\theta_0{}^2$

$G_{31} = 6\Psi_1{}^2+\Psi_2{}^2+2\Psi_1\Psi_2-54\Psi_1\Psi_0+6\Psi_2\Psi_0+54\Psi_0{}^2$

$G_{32} = 6\Psi_2{}^2+\Psi_1{}^2+2\Psi_1\Psi_2-54\Psi_2\Psi_0+6\Psi_1\Psi_0+54\Psi_0{}^2$

$F_{51} = 4\theta_1-\theta_2-3\theta_0$

$F_{52} = 4\theta_2-\theta_1-3\theta_0$

$G_{51} = 4\Psi_1-\Psi_2-3\Psi_0$

$G_{52} = 4\Psi_2-\Psi_1-3\Psi_0$

$F_{61} = 8\theta_1{}^2+3\theta_2{}^2-4\theta_1\theta_2-12\theta_1\theta_0-2\theta_2\theta_0+27\theta_0{}^2$

$F_{62} = 8\theta_2{}^2+3\theta_1{}^2-4\theta_1\theta_2-12\theta_2\theta_0-2\theta_1\theta_0+27\theta_0{}^2$

$G_{61} = 8\Psi_1{}^2+3\Psi_2{}^2-4\Psi_1\Psi_2-12\Psi_1\Psi_0-2\Psi_2\Psi_0+27\Psi_0{}^2$

$G_{62} = 8\Psi_2{}^2+3\Psi_1{}^2-4\Psi_1\Psi_2-12\Psi_2\Psi_0-2\Psi_1\Psi_0+27\Psi_0{}^2$

## A.3.4 TWO DIMENSIONAL BASED ON AVERAGE STRAIN

$n_2(2,2) = n_2(5,5) = (9\theta_1{}^2+9\theta_2{}^2-2\theta_1\theta_2-36\theta_1\theta_0-36\theta_2\theta_0+216\theta_0{}^2)\ \frac{EA}{100\ell}$

$n_2(2,5) = -n_2(2,2)$

$n_2(2,3) = (6\theta_1{}^2+\theta_2{}^2+2\theta_1\theta_2-54\theta_1\theta_0+6\theta_2\theta_0+54\theta_0{}^2)\ \frac{EA}{300}$

$n_2(3,5) = -n_2(2,3)$

$n_2(2,6) = (6\theta_2{}^2+\theta_1{}^2+2\theta_1\theta_2-54\theta_2\theta_0+6\theta_1\theta_0+54\theta_0{}^2)\ \frac{EA}{300}$

$n_2(5,6) = -n_2(2,6)$

$n_2(3,3) = (8\theta_1{}^2+3\theta_1{}^2-4\theta_1\theta_2-12\theta_1\theta_0-2\theta_2\theta_0+27\theta_0{}^2)\ \frac{EA\ell}{300}$

$n_2(3,6) = (-2\theta_1{}^2-2\theta_2{}^2+6\theta_1\theta_2-2\theta_1\theta_0-2\theta_2\theta_0-3\theta_0{}^2)\ \frac{EA\ell}{300}$

$n_2(6,6) = (8\theta_2{}^2+3\theta_1{}^2-4\theta_1\theta_2-12\theta_2\theta_0-2\theta_1\theta_0+27\theta_o{}^2)\ \frac{EA\ell}{300}$

in which

$\theta_0 = \frac{v_2-v_1}{\ell}$

# APPENDIX B

## $[k_{\varepsilon_o}]$ INITIAL STRAIN STIFFNESS MATRIX FOR QUARTIC STRAIN FUNCTION

### B.1 THREE DIMENSIONAL

$$k_{\varepsilon_o}(2,2) = k_{\varepsilon_o}(3,3) = k_{\varepsilon_o}(8,8) = k_{\varepsilon_o}(9,9) = (\rho_3 - 2\rho_4 + \rho_5)\ \frac{36EA}{\ell}$$

$$k_{\varepsilon_o}(2,8) = k_{\varepsilon_o}(3,9) = -k_{\varepsilon_o}(2,2)$$

$$k_{\varepsilon_o}(3,5) = k_{\varepsilon_o}(6,8) = 6\ (\rho_2 - 5\rho_3 + 7\rho_4 - 3\rho_5)\ EA$$

$$k_{\varepsilon_o}(2,6) = k_{\varepsilon_o}(5,9) = -k_{\varepsilon_o}(3,5)$$

$$k_{\varepsilon_o}(6,6) = k_{\varepsilon_o}(5,5) = (\rho_1 - 8\rho_2 + 22\rho_3 - 24\rho_4 + 9\rho_5)\ EA\ell$$

$$k_{\varepsilon_o}(11,11) = k_{\varepsilon_o}(12,12) = (4\rho_3 - 12\rho_4 + 9\rho_5)\ EA\ell$$

$$k_{\varepsilon_o}(9,11) = k_{\varepsilon_o}(2,12) = 6\ (2\rho_3 - 5\rho_4 + 3\rho_5)\ EA$$

$$k_{\varepsilon_o}(3,11) = k_{\varepsilon_o}(8,12) = -k_{\varepsilon_o}(9,11)$$

$$k_{\varepsilon_o}(5,11) = k_{\varepsilon_o}(6,12) = (-2\rho_2 + 11\rho_3 - 18\rho_4 + 9\rho_5)\ EA\ell$$

in which $\rho_1$, $\rho_2$, ..., $\rho_5$ are evaluated from the following steps:

1) Initialize $BTO_i = 0.0$ for $i = 1,5$

2) Save $BTO_i$ in $BOL_i$ as:

$$BLO_i = BTO_i \quad i = 1,5$$

3) Evaluate $\alpha_1$, $\alpha_2$, $\alpha_3$, $\beta_1$, $\beta_2$, $\beta_3$ as:

$$\alpha_1 = \theta_1$$

$$\alpha_2 = \frac{2}{\ell} (-3v_1-2\theta_1\ell+3v_2-\theta_2\ell)$$

$$\alpha_3 = \frac{3}{\ell} (2v_1+\theta_1\ell-2v_2+\theta_2\ell)$$

$$\beta_1 = -\Psi_1$$

$$\beta_2 = \frac{2}{\ell} (-3w_1-2\Psi_1\ell+3w_2+\Psi_2\ell)$$

$$\beta_3 = \frac{3}{\ell} (2w_1-\Psi_1\ell-2w_2-\Psi_2\ell)$$

4) Evaluate $b_1, b_2$ , ..., $b_5$ as:

$$b_1 = \frac{u_2-u_1}{\ell} + \frac{1}{2} (\alpha_1{}^2+\beta_1{}^2)$$

$$b_2 = \alpha_1\alpha_2+\beta_1\beta_2$$

$$b_3 = \frac{1}{2} (\alpha_2{}^2+\beta_2{}^2) + \alpha_1\alpha_3+\beta_1\beta_3$$

$$b_4 = \alpha_2\alpha_3+\beta_2\beta_3$$

$$b_5 = \frac{1}{2} (\alpha_3{}^2+\beta_3{}^2)$$

5) Updated $BTO_i$ as:

$$BTO_i = BOL_i + b_i \qquad i = 1,5$$

6) Evaluate $\rho_i$ as:

$$\rho_i = \frac{1}{i} BTO_1 + \frac{1}{i+1} BTO_2 + \frac{1}{i+2} BTO_3 + \frac{1}{i+3} BTO_4 + \frac{1}{i+4} BTO_5 \quad i=1,5$$

## B.2 TWO DIMENSIONAL

$$k_{\varepsilon_o}(2,2) = k_{\varepsilon_o}(5,5) = (\rho_3-2\rho_4+\rho_5)\,\frac{36EA}{\ell}$$

$$k_{\varepsilon_o}(2,5) = -k_{\varepsilon_o}(2,2)$$

$$k_{\varepsilon_o}(3,5) = 6\,(\rho_2-5\rho_3+7\rho_4-3\rho_5)\,EA$$

$$k_{\varepsilon_o}(2,3) = -k_{\varepsilon_o}(3,5)$$

$$k_{\varepsilon_o}(3,3) = k_{\varepsilon_o}(2,2) = (\rho_1-8\rho_2+22\rho_3-24\rho_4+9\rho_5)\,EA\ell$$

$$k_{\varepsilon_o}(6,6) = (4\rho_3-12\rho_4+9\rho_5)\,EA\ell$$

$$k_{\varepsilon_o}(2,6) = 6\,(2\rho_3-5\rho_4+3\rho_5)\,EA$$

$$k_{\varepsilon_o}(5,6) = -k_{\varepsilon_o}(2,6)$$

$$k_{\varepsilon_o}(3,6) = (-2\rho_2+11\rho_3-18\rho_4+9\rho_5)\,EA\ell$$

in which $\rho_1$, $\rho_2$, ..., $\rho_5$ are evaluated as following steps:

1. & 2. The same as the three dimensional case.

3. Evaluate $\alpha_1$, $\alpha_2$, $\alpha_3$ as:

$$\alpha_1 = \theta_1$$

$$\alpha_2 = \frac{2}{\ell}\,(-3v_1-2\theta_1\ell+3v_2-\theta_2\ell)$$

$$\alpha_3 = \frac{3}{\ell}(2v_1+\theta_1\ell-2v_2+\theta_2\ell)$$

4. Evaluate $b_1$, $b_2$, ..., $b_5$ as:

$$b_1 = \frac{u_2-u_1}{\ell} + \frac{1}{2}\alpha_1^2$$

$$b_2 = \alpha_1\alpha_2$$

$$b_3 = \frac{1}{2}\alpha_2^2+\alpha_1\alpha_3$$

$$b_4 = \alpha_2\alpha_3$$

$$b_5 = \frac{\alpha_3^2}{2}$$

5. & 6. The same as the three dimensional case.

# APPENDIX C

## $[n_1^*]$ GEOMETRIC STIFFNESS MATRIX

### C.1 THREE DIMENSIONAL

$$n_1^*(2,2) = n_1^*(3,3) = n_1^*(8,8) = n_1^*(9,9) = \frac{6EA}{5\ell^2}(u_2-u_1)$$

$$n_1^*(3,9) = n_1^*(2,8) = -n_1^*(2,2)$$

$$n_1^*(5,5) = n_1^*(6,6) = n_1^*(11,11) = n_1^*(12,12) = \frac{2EA}{15}(u_2-u_1)$$

$$n_1^*(2,6) = n_1^*(5,9) = n_1^*(9,11) = n_1^*(2,12) = \frac{EA}{10\ell}(u_2-u_1)$$

$$n_1^*(3,5) = n_1^*(6,8) = n_1^*(8,12) = n_1^*(3,11) = -n_1^*(2,6)$$

$$n_1^*(6,12) = n_1^*(5,11) = \frac{-EA}{30}(u_2-u_1)$$

### C.2 TWO DIMENSIONAL

$$n_1^*(2,2) = n_1^*(5,5) = \frac{6EA}{5\ell^2}(u_2-u_1)$$

$$n_1^*(2,3) = n_1^*(2,6) = \frac{EA}{10\ell}(u_2-u_1)$$

$$n_1^*(2,5) = n_1^*(3,5) = n_1^*(5,6) = -n_1^*(2,2)$$

$$n_1^*(3,3) = n_1^*(6,6) = \frac{2EA}{15}(u_2-u_1)$$

$$n_1^*(3,6) = \frac{-EA}{30}(u_2-u_1)$$

### C.3 $[k_G]$ MATRIX

$[k_G] = \dfrac{\ell}{EA(u_2-u_1)}[n_1^*]$ for both two and three dimensional cases.

# APPENDIX D

# COMPUTER PROGRAMS

## D.1 DESCRIPTION OF SUBROUTINES

A general description of the computer programs is given in Section 3.6. The listing of programs are presented at the end of this appendix with appropriate comment statements. In the following a brief description of the subroutines is given.

The main programs (NFRAL3D, NFRAL2D, NFRAE2D) direct the flow of computation by calling the appropriate subroutines for each step of the solution procedure. Subroutine NODDATA reads data regarding the overall geometry of the structure including coordinates and degrees of freedom. Coordinates for plane circular or parabolic arches may be generated. The equation numbers are generated by this subroutine. The subroutine ELEMENT reads data related to the element properties and node numbers. The subroutine BAND computes the semibandwidth, MBAND, that the stiffness matrix of the structure will have.

Subroutines BEAM and TRUSS evaluate the linear stiffness matrices of the beam and truss elements, respectively. Subroutines TRANSFM and INVTRNS are used for geometric transformation from local coordinates to global coordinates and vice versa. Subroutines SBEAME1, SBEAME2, and KEPSI01, respectively, evaluate the non-zero entries of $[n_1]$, $[n_2]$, and $[K_{\varepsilon_o}]$. The assembly of $[k]$, $[n_1]$, $[n_2]$ and $[K_{\varepsilon_o}]$ into the appropriate global stiffness matrices is accomplished with subroutine ASEMBLE. Subroutine LINSOLN solves the system of linear equations by Gauss elimination. Subroutine STCONDN condenses the structural linear stiffness matrix and load vector into the degrees of freedom which have been established in subroutine NODDATA. Subroutine RECOVER recovers the internal degrees of freedom of the structure after using subroutine LINSOLN. Subroutine IDENT identifies the displacements obtained from LINSOLN with the nodal displacements similarly for those found in the recovery process.

The solution of the linear eigenvalues problem and the quadratic eigenvalue problem is obtained with subroutines EIGENVL and NLEIGNP, respectively. The subroutine EIGENVL uses inverse vector iteration with Rayleigh quotient to obtain the lowest eigenvalue and corresponding eigenvector of the linear problem. For the solution of the quadratic problem, the subroutine NLEIGNP uses the modified regula falsi method of iteration by calling subroutine MRGFLS and the function sunprogram DET. Subroutine MULT is used for matrix multiplication and the function subprogram DET1 evaluates the determinant of the structural tangent stiffness matrix. Finally, subroutines ENDFORC and STRESS evaluate the element end forces and stresses, respectively.

## D.2 VARIABLES USED IN THE COMPUTER PROGRAMS

The variable names used in the programs are listed below in alphabetical order:

### MAIN PROGRAMS NFRAL3D, NFRAL2D, NFRAE2D

| | | |
|---|---|---|
| A(M) | = | The cross-sectional area of element M; |
| A7OLD(M), A7TOT(M) | = | parameters related to element M for evaluation of the initial strain stiffness matrix; |

| | | |
|---|---|---|
| BOL(M,J), BTO(M,J), BE(J) | = | Intermediate parameters for the evaluation of initial strain stiffness matrix; |
| D(I) | = | Displacement vector, found from the solution of the system S*D=R. I varies from 1 to NEQ; |
| DTOT(I), DACTUAL(I) | = | The same as D(I) but for total displacement measured with reference to the beginning of each load increment or initial geometry, respectively; |
| DETER, DETERMNT | = | Determinant of the structural secant or tangent stiffness matrices; |
| DN(I,1) | = | End forces in global coordinates for each element. I varies from 1 to 6; |
| E(N) | = | Modulus of elasticity of element group N; |
| ES(I,M) | = | End forces in local coordinates for element M. I varies from 1 to 3; |
| G(N) | = | Shear modulus of element group N; |
| IA(N,I) | = | "Boundary condition code" of node N for its Ith degree of freedom. Initially it is defined as follows:<br>IA(N,I) = 1 if constrained;<br>= 0 if free<br>After processing,<br>IA(N,I) = 0 if initially = 1;<br>= equation number for the D.O.F. if initially = 0; |
| IB(N,I) | = | "Additional boundary condition codes."<br>IB(N,I) = 0 if free<br>= N if slave to node N;<br>= -1 if to be condensed.<br>After processing, IB(N,I) is unchanged except,<br>IB(N,I) = -(condensation number of the D.O.F. if initially IB(N,I) = -1); |

| | |
|---|---|
| ICAL1, ICAL2, ICAL3 | = Variables controlling print-out (more details are indicated by "comment statement" in the listing of programs); |
| ICHECK | = Parameter used for Newton-Raphson approach in Lagrangian coordinates to control the type of computation needed in each load increment; |
| IDET | = Parameter used for evaluation of the determinant of the secant or tangent stiffness matrices either before or after Gauss elimination process; |
| IGOPTIN | = Parameter used to specify type of the geometry for plane frames (i.e., circular, parabolic arch or arbitrary geometry); |
| IPAR | = Variable identifying appropriate "Tape" for storage of different structural stiffness matrices (i.e., [K], [$K_{\varepsilon_o}$], [$N_1$], [$N_2$]); |
| ISTRESS | = If EQ. 1, compute nodal forces and stresses in the structure. If EQ. 0, skip; |
| IXX(M) | = Moment of inertia about the $\zeta$-axis of the cross section of element M; |
| IZZ(M) | = Moment of inertia about the $\eta$-axis of the cross section of element M; |
| KT(M) | = Torsion constant of element M; |
| L(N,K) | = Variable identifying the Kth element in the element group N; |
| LE(M) | = Length of element M; |
| MBAND | = Semibandwidth of structure stiffness matrix; |
| NCOND | = Total number of degrees of freedom to be condensed out; |
| NCOUNT | = The order of load increment in incremental approaches; |

| | |
|---|---|
| NE | = Total number of elements in the structure; |
| NEQ | = Total number of equations; |
| NODEI(M) | = Variable identifying the number of node I of element M; |
| NODEJ(M) | = Variable identifying the number of node J of element M; |
| NSIZE | = Total number of degrees of freedom, condensed and free, of the system. (NSIZE = NEQ + NCOND); |
| NUMEG | = Total number of element groups; |
| NUMEL(I) | = Total number of elements in element group I (in NFRAL3D); |
| NUMEL | = Total number of elements (in NFRAL2D and NFRAE2D); |
| NUMITER | = Number of iterations at each stage of computation; |
| NUMNP | = Total number of nodal points; |
| PI(N,I) | = Load applied at node N, in the Ith direction; |
| PACTUAL(I) | = Applied load related to the Ith D.O.F. in the structural load vector at each stage; |
| R(I) | = Load vector of the system; |
| ROT(I,J), ROTRAN(I,J) | = Rotation and inverse rotation matrix for each element (I = 1, 6, J = 1, 6), respectively; |
| S(I,J) | = Tangent stiffness matrix of the system; |
| SCALE | = Scale factor in the evaluation of the determinant of the structural stiffness matrix; |
| SE(I,J), SEI(I,J), SE2(I,J) | = Element stiffness matrices (i.e., [k], [$n_1$], [$n_2$], respectively); |

| | |
|---|---|
| SXX(M) | = Section modulus about the $\zeta$-axis of the cross section of element M; |
| ULOC(M,I) | = Identifies local displacement in the Ith direction of element M (I varies from 1 to 12 for three dimensional case and from 1 to 6 for two dimensional); |
| USTAR(I,M) | = Identifies the end displacement for the Ith direction of element M in Eulerian coordinates; |
| W(I,J), WCHK(I,J) | = Incremental recovered displacements (used in iterative process) related to node I in the Jth direction; |
| WTOT(I,J) | = The same as W(I,J) but for total displacements; |
| X(N), Y(N), Z(N) | = Global X, Y, Z-coordinates of node N; |
| YPGM(M), ZPGM(M) | = $Y_{p\gamma}$ and $Z_{p\gamma}$, respectively; See Ref. [26]; |

SUBROUTINE NODDATA

| | |
|---|---|
| ALFZERO | = Opening angle of circular arch; |
| RADIUS | = Radius of circular arch; |
| RISE | = Rise of parabolic arch; |
| SPAN | = Span of parabolic arch; |

SUBROUTINE TRANSFM

| | |
|---|---|
| Rcol(I) | = Identifies the entries of rotation matrix for three dimensional beam element. I varies from 1 to 9; |

SUBROUTINE INVTRNS

| | |
|---|---|
| V(NP,I) | = Identifies the element local displacements for nodal point NP and Ith direction (I varies from 1 to 6); |

## SUBROUTINE STCNDN

| | | |
|---|---|---|
| RC(I) | = | Condensed structural load vector (I = 1, NEQ); |
| SC(I,J) | = | Condensed structure linear tangent stiffness matrix; |

## SUBROUTINE EIGENVL (EIGEN, IDATA)

| | | |
|---|---|---|
| EIGEN | = | Eigenvalue; |
| EIGNVTR | = | Eigenvector corresponding to EIGEN; |
| EPSI | = | Tolerance; |
| MAX | = | Maximum number of iterations allowed; |
| RHO | = | Rayleigh quotient; |
| XB | = | Vector that stores the approximation to the eigenvector after each iteration; |

## SUBROUTINE ENDFORC

| | | |
|---|---|---|
| DN(I) | = | Stress resultants on the nodes of each element; |

## SUBROUTINE STRESS

| | | |
|---|---|---|
| SIGMA(M) | = | Maximum stress of element M; |
| STRAIN(M) | = | Maximum strain of element M; |

## SUBROUTINE NLEIGNP

| | | |
|---|---|---|
| A,B | = | Variables defining the interval in which the eigenvalue is enclosed; |
| ERROR | = | Upper bound on the computation of the eigenvalue after convergence; |
| FL | = | Value of the determinant of the matrix $S = K + L*N_1 + L*L*N_2$ at the converged value of the eigenvalue; |

| | |
|---|---|
| FTOL | = Convergence criterion for sufficiently small value of the determinant of eigenvalue; |
| L | = Converged value of the eigenvalue; L = (A+B)/2; |
| NTOL | = Maximum number of iterations allowed; |
| XTOL | = Tolerance; |

<u>SUBROUTINE MRGFLS</u>

| | |
|---|---|
| IFLAG | = Variable defining the status of the iteration. If EQ. 1, convergence was successful. If EQ. 2, no convergence after NTOL iterations. If EQ. 3, both endpoints, A,B, are on the same side of the root, hence method of iteration cannot be used; |
| FA | = Value of the determinant of matrix S at interval endpoint A; |
| FB | = Value of the determinant of matrix S at interval endpoint B; |
| W | = Weighted values of the root between interval endpoints A and B; |
| FW | = Value of the determinant of matrix S at the weighted value W; |

<u>FUNCTION DET</u>

| | |
|---|---|
| DET | = Value of the determinant of the matrix $S = K + L*N_1 + L*L*N_2$ at a particular value of L; |
| K(I,J) | = Part of element S(I,J) corresponding to linear stiffness K(I,J); |
| L | = Load parameter; |

|  |  |
|---|---|
| $N_1(I,J)$ | = Part of element S(I,J) corresponding to matrix $N_1$ (I,J); |
| $N_2(I,J)$ | = Part of element S(I,J) corresponding to matrix $N_2$(I,J). |

## D.3 PROGRAM NFRAL3D

```
      PROGRAM NFRAL3D(INPUT,OUTPUT=65,TAPE60=INPUT,TAPE61=OUTPUT,
     +TAPE1,TAPE2,TAPE3,TAPE4,TAPE5,TAPE6,TAPE7,TAPE8,TAPE9,TAPE10,
     +TAPE11,TAPE12,TAPE13,TAPE14,TAPE15,TAPE16)
C
C     ******************************************************************
C         THIS PROGRAM USES THE FINITE ELEMENT METHOD TO ANALYZE
C         STRUCTURE MADE UP OF STRAIGHT BEAM ELEMENTS IN THREE
C         DIMENSIONAL SPACE
C         OTHER ELEMENTS MAY BE ANALYZED BY ADDING A SUBROUTINE
C         FOR EACH NEW TYPE OF ELEMENT BEING USED.
C         GEOMETRIC NONLINEARITIES ARE CONSIDERED.
C     ******************************************************************
C
      REAL IXX,IYY,IZZ,KT,II,JJ,LE,N1STTOT
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/4/SE(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36),
     +L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/8/PI(37,6),PN(37,6),R(107)
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/10/D(60),G1(60),G2(60),G3(60),G4(60),RC(60),
     +SC(60,20),IGAUS
      COMMON/11/DN(12),W(37,6),V(37,6)
      COMMON/12/JLOC(35,12),U(12),RCOL(9),MSUOPTN,N1GOPTN
      COMMON/16/PRIOPTN
      COMMON/17/A7TOT(36),A7OLD(36),BOL(36,5),BTO(36,5),BE(5)
      DIMENSION  DTEMP(60),PTEMP(60),PSTART(60),DTOT(60),PACTUAL(60)
      DIMENSION  PSAVE(60),DACTUAL(60),N1STTOT(60,20)
      DIMENSION  SOLD(60,20),SRK(60,20),SRN1(60,15)
      DIMENSION  REFSTRT(37,6),REFPTMP(37,6),SRN2(37,20)
      INTEGER  PROTYPE,EIGVALU,PRIOPTN,DETOPTN
C     ******************************************************************
C         TAPES 8,7,4  FOR K BEAM ELEMENT
C         TAPES 10,11,4  FOR K TRUSS ELEMENT
C         TAPE 9  FOR (K+N1) STRUCTURAL
C         TAPE 13 FOR (-K)
C         TAPES 1,2,6 FOR N1
C         TAPES 3,5,12  FOR N2
C         TAPES 14,15,16  FOR KEPSID
C         N1OPTIN=1    N1 SHOULD BE INCLUDED
C         N1OPTIN=0    N1 SHOULDN,T BE INCLUDED
C         THE SAME AS ABOVE FOR N2OPTIN
C         ITERCHK=1 FOR ITERATION APPROACH
C         ITERCHK=0 FOR STRAIGHT INCREMENTAL APPROACH AND EIGENSOLUTION
C         PRIOPTN=0  IF WE JUST WANT THE RESULTS TO BE PRINTED
C         PRIOPTN=1  IF WE WANT INTERMEDIATE COMPUTATIONS PRINTED
C         IFIX=1 FOR FIXED LAGRANGIAN  APPROACH.
C         IFIX=0 FOR UPDATED LAGRANGIAN  APPROACH.
C         MSUOPTN=1  CONSTANT STRAIN(AVERAGE) FOR EACH ELEMENT
C         MSUOPTN=2  STRAIN IS A QUADRATIC FUNCTION OF SLOPE AT
C         EACH POINT OF ELEMENT.
C         N1GOPTN=0  FOR LINEAR EIGENVALUE SOLUTION N1 IS USED.
C         N1GOPTN=1  FOR LINEAR EIGENVALUE SOLUTION N1STAR IS USED
C         JUSTK=1  ONLY UPDATED-LINEAR STIFFNESS MATRIX IS USED
C         OTHERWISE JUSTK=0
C         TOLER=TOLERANCE  FOR SUCCESSIVE ITERATION CONVERGENCE
C         FOR OTHER CASES TOLER=0.
C         DETOPTN=0  NO CONTROL ON THE DETERMINANT OF THE TANGENT
C         STIFFNESS MATRIX
C         DETOPTN=1 EXECUTION WOULD BE TERMINATED IF DETERMINANT OF
C         THE TANGENT STIFFNESS MATRIX IS NEGATIVE
C         IN THE FOLLOWING INPUT FOR EIGENVALUE PROBLEM ONLY
C         PRIOPTN,MSUOPTN ARE NEEDED(OTHERS MAY BE SET EQUAL
C         TO ZERO)
C     ******************************************************************
      READ(60,6971)PRIOPTN,N2OPTIN,N1OPTIN,ITERCHK,
     +MSUOPTN,N1GOPTN,IFIX,JUSTK,TOLER,DETOPTN
 6971 FORMAT(8I5,F10.5,I5)
      WRITE(61,6972)PRIOPTN,N2OPTIN,N1OPTIN,ITERCHK,
     +MSUOPTN,N1GOPTN,IFIX,JUSTK,TOLER,DETOPTN
 6972 FORMAT(10X,8HPRIOPTN=,I2/10X,8HN2OPTIN=,I2/
     +10X,*N1OPTIN=*,I2/10X,*ITERCHK=*,I2,/10X,*MSUOPTN=*,I2,10X,
     +*N1GOPTN=*,I2,10X,*IFIX=*,I2,10X,*JUSTK=*,I2,
     +/10X,*TOLER=*,F10.5,10X,*DETOPTN=*,I2)
      IF(ITERCHK.EQ.1) READ(60,6948)  DELTA1,DELTA2
C     ******************************************************************
C         DELTA1=ALLOWABLE TOLERANCE FOR FORCE COMPONENTS OF
C         UNBALANCED FORCE VECTOR
```

```
C         DELTA2=ALLOWABLE TOLERANCE FOR MOMENT COMPONENTS OF
C         UNBALANCED FORCE VECTOR
C     ******************************************************************
6948  FORMAT(2E21.15)
      IF(ITERCHK.EQ.1) WRITE(61,6931) DELTA1,DELTA2
6931  FORMAT(10X,6HEPSI1=,E21.15/10X,6HEPSI2=,E21.15,/)
      READ(60,1) PROTYPE,EIGVALU,LODPON1,LODPON2,LODPON3,LODPON4,LODPON
     +5,LODPON6,ISTRESS
C     ******************************************************************
C         PROTYPE=1 IS FOR INCREMENTAL LOADING IN FIXED COORDINATES
C         SECANT STIFFNESS APPROACH(SUCCESIVE ITERATIONS)
C         PROTYPE=2 IS  FOR EIGENVALUE PROBLEM
C         PROTYPE=3   FOR INCREMENTAL LOADING IN UPDATED-COORDINATES
C         PROTYPE=3   FOR NEWTON-RAPHSON FIXED COORDINATE APPROACH
C         PROTYPE=3,ITERCHK=0 AND JUSTK=1 FOR THE UPDATED-COORDINATE
C         APPROACH BY UPDATING ONLY LINEAR STIFFNESS MATRIX WITH
C         NO ITERATION.
C         EIGVALU=1 IS FOR LINEAR EIGENVALUE PROBLEM
C         EIGVALU=2 IS FOR QUADRATIC EIGENVALUE PROBLEM
C         EIGVALU=3 IS FOR INCREMENTAL LOADING IN FIXED COORDINATES
C         EIGVALU=4 FOR INCREMENTAL LOADING IN UPDATED-COORDINATES(OR
C         FIXED COORDINATES)
C         LODPONI(I=1,6) EQUAL TO ZERO FOR EIGENVALUE SOLUTION
C         LODPON1 UP TO LODPON6 ARE THE ORDER OF D.O.F.(IN LINEAR
C         SYSTEM OF EQUATIONS) RELATED TO EXTERNAL CONCENTRATED
C         LOADS OR MOMENTS APPLIED ON THE STRUCTURE
C         IF ISTRESS=1 ELEMENT END FORCES SHOULD BE EVALUATED.
C         IF ISTRESS=0 ELEMENT END FORCES SHOULDN,T BE
C         EVALUATED(EFFICIENT CODING)
C     ******************************************************************
C
      IF(PROTYPE.EQ.2) GO TO 501
C
C     ******************************************************************
C         PINIT1 UP TO PINIT6,PINC1 UP TO PINC6 AND PTOT1 UP TO PTOT6
C         ARE THE INITIAL,INCREMENTAL AND MAXIMUM DEFINED EXTERNAL
C         LOAD COMPONENTS.
C         AT LEAST PINIT1 SHOULDN,T BE EQUAL TO ZERO
C     .   MAXITER=MAXIMUM NUMBER OF ITERATIONS
C     ******************************************************************
      READ(60,699)    PINIT1,PINC1,PTOT1,PINIT2,PINC2,PTOT2,MAXITER
      WRITE(61,799)   PINIT1,PINC1,PTOT1,PINIT2,PINC2,PTOT2,MAXITER
      READ(60,1699)   PINIT3,PINC3,PTOT3,PINIT4,PINC4,PTOT4
1699  FORMAT(6F10.6)
      WRITE(61,1799) PINIT3,PINC3,PTOT3,PINIT4,PINC4,PTOT4
1799  FORMAT(10X,*PINIT3=*,F10.6,10X,*PINC3=*,F10.6,
     +10X,*PTOT3=*,F10.6,10X,*PINIT4=*,F10.6/
     +10X,*PINC4=*,F10.6,10X,*PTOT4=*,F10.6,/)
      READ(60,1699)   PINIT5,PINC5,PTOT5,PINIT6,PINC6,PTOT6
      WRITE(61,1837)  PINIT5,PINC5,PTOT5,PINIT6,PINC6,PTOT6
1837  FORMAT(10X,*PINIT5=*,F10.6,10X,*PINC5=*,F10.6,
     +10X,*PTOT5=*,F10.6,10X,*PINIT6=*,F10.6/
     +10X,*PINC6=*,F10.6,10X,*PTOT6=*,F10.6,/)
501   CONTINUE
C
C     ******************************************************************
C         IGOPTIN=1 FOR CIRCULAR ARCH,IGOPTIN=2 FOR PARABOLIC
C         AND IGOPTIN=0 FOR OTHER GEOMETRIES
C     ******************************************************************
      READ(60,10)      IGOPTIN
      WRITE(61,5287)   IGOPTIN
5287  FORMAT(//,10X,*IGOPTIN=*,I2,/)
      WRITE(61,8765) PROTYPE,LODPON1,LODPON2,LODPON3,LODPON4,LODPON5,
     +LODPON6,ISTRESS
8765  FORMAT(10X,*PROTYPE=*,I2,10X,*LODPON1=*,I2,
     +10X,*LODPON2=*,I2,10X,*LODPON3=*,I2/
     +10X,*LODPON4=*,I2,10X,*LODPON5=*,I2,10X,*LODPON6=*,I2,/
     +10X,*ISTRESS=*,I5,/)
C
      READ(60,1010) TITLE1,TITLE2,TITLE3,NE,NUMNP,NUMEG,IDATA,ICAL1,
     +              ICAL2,ICAL3
C     ******************************************************************
C         ICAL1=0 FOR LOAD VECTOR AND STRUCTURAL LINEAR STIFFNESS
C         MATRIX TO BE PRINTED(ICAL1=1 SKIP)
C         ICAL2=0 FOR DISPLACEMENT VECTOR TO BE PRINTED
C         (ICAL2=1 SKIP)
C         ICAL3=0 FOR LINEAR STIFFNESS MATRIX IN LOCAL
C         OR GLOBAL TO BE PRINTED,ALSO FOR DETAILS OF EIGENVALUE
C         SOLUTION(ICAL3=1 SKIP)
C     ******************************************************************
      WRITE(61,2010)TITLE1,TITLE2,TITLE3,NE,NUMNP,NUMEG,IDATA,ICAL1,
     +          ICAL2,ICAL3
C
C          READ NODAL POINT DATA
C     ****************************************************************
C
      CALL NODDATA(IGOPTIN)
C
C          READ AND STORE INITIAL LOAD DATA
C     ****************************************************************
C
      WRITE(61,2015)
500   READ(60,1015)  N,(PI(N,I),I=1,6)
      WRITE(61,2020) N,(PI(N,I),I=1,6)
      IF(N.NE.NUMNP) GO TO 500
C
      IF(PROTYPE.NE.3) GO TO 3021
      SCALE=10000.
      DO 5001 I=1,NEQ
5001  PSAVE(I)=DACTUAL(I)=DTOT(I)=0.0
      PLOAD1=PINIT1
      PLOAD2=PINIT2
      PLOAD3=PINIT3
      IF(NEQ.GE.4) PLOAD4=PINIT4
      IF(NEQ.GE.5) PLOAD5=PINIT5
      IF(NEQ.GE.6) PLOAD6=PINIT6
      DO 3010 I=1,NUMNP
      DO 3010 J=1,6
3010  W(I,J)=0.0
      ICHECK=1
1001  DO 3020 I=1,NUMNP
      IF(IFIX.EQ.0)   X(I)=X(I)+W(I,1)
      IF(IFIX.EQ.0)   Y(I)=Y(I)+W(I,2)
      IF(IFIX.EQ.0)   Z(I)=Z(I)+W(I,3)
3020  CONTINUE
      IF(PRIOPTN.EQ.0)     GO TO 4994
      WRITE(61,4995)
4995  FORMAT(/,10X,*NODE*,10X,*X(I)*,10X,*Y(I)*,10X,*Z(I)*,/)
      DO 4996  I=1,NUMNP
      WRITE(61,4997)  I,X(I),Y(I),Z(I)
4997  FORMAT(/,10X,I5,3F15.8)
4996  CONTINUE
4994  CONTINUE
3021  CONTINUE
C          READ AND STORE ELEMENT DATA
C     ****************************************************************
C
      IPAR=NUMITER=1
      N=0
5927  N=N+1
      IF(PROTYPE.NE.3)   GO TO 2222
      IF(ICHECK.NE.1)    GO TO 4444
2222  READ(60,1020)  NUMEL(N)
4444  IF(NUMEL(N).EQ.0)   GO TO 100
      IF(PROTYPE.NE.3)   ICHECK=0
      CALL ELEMENT (N,ICHECK,PROTYPE)
100   CONTINUE
      IF(N.EQ.NUMEG)   GO TO 5928
      GO TO 5927
5928  CONTINUE
C
      IF(PROTYPE.NE.3) GO TO 3335
      IF(ICHECK.NE.1)  GO TO 3333
3335  CONTINUE
C          COMPUTE SEMIBANDWIDTH OF STRUCTURE STIFFNESS MATRIX
C     ****************************************************************
      CALL BAND
      IF(JUSTK.EQ.1)      GO TO 2110
      IF(PROTYPE.NE.3)    GO TO 2110
      DO 5745  NN=1,NUMEG
      IF(NUMEL(NN).EQ.0)     GO TO 5745
      NAME=NUMEL(NN)
      DO 5341  K=1,NAME
      M=L(NN,K)
      IF(MSUOPTN.EQ.1)   A7OLD(M)=0.0
      DO 5341 I=1,5
      IF(MSUOPTN.EQ.2)   BOL(M,I)=0.0
5341  CONTINUE
5745  CONTINUE
2110  CONTINUE
C
C
```

```
C
C        ASSEMBLE INITIAL LOADS AND NODAL LOADS INTO LOAD VECTOR
C        SET ARRAYS -S- AND -R- EQUAL TO ZERO
C     ******************************************************************
C
C
      CALL ASEMBLE (M)
3333  CONTINUE
      IF(PROTYPE.NE.3)  GO TO 3336
      IF(ICHECK.EQ.1)   GO TO 2111
      GO TO 2112
2111  DO 5003  I=1,NEQ
5003  PSTART(I)=0.0
      PSTART(LODPON1)=PINIT1
      PSTART(LODPON2)=PINIT2
      PSTART(LODPON3)=PINIT3
      IF(NEQ.GE.4) PSTART(LODPON4)=PINIT4
      IF(NEQ.GE.5) PSTART(LODPON5)=PINIT5
      IF(NEQ.GE.6) PSTART(LODPON6)=PINIT6
C
2112  IF(JUSTK.EQ.1)  ICHECK=2
      IF(JUSTK.EQ.1)  GO TO 3336
      IF(ICHECK.EQ.1) GO TO 3336
      IPAR=2
      DO 2113  I=1,NSIZE
      DO 2113  J=1,MBAND
2113  S(I,J)=0.0
      CALL ELEMENT(N,ICHECK,PROTYPE)
1901  IF(ITERCHK.NE.0) CALL INVTRNS
      IF(ICHECK.EQ.3)  GO TO 4988
      IF(N1OPTIN.EQ.0) GO TO 4987
      DO 1801 I=1,NSIZE
      DO 1801 J=1,MBAND
1801  S(I,J)=0.0
      IPAR=3
      DO 2101 N=1,NUMEG
      IF(NUMEL(N).EQ.0) GO TO 2101
      CALL SBEAME1(N)
2101  CONTINUE
4987  CONTINUE
      IF(ICHECK.EQ.3)  GO TO 4988
      IF(N2OPTIN.EQ.0) GO TO 4988
      DO 7691 I=1,NSIZE
      DO 7691 J=1,MBAND
7691  S(I,J)=0.0
      IPAR=4
      DO 7692 N=1,NUMEG
      IF(NUMEL(N).EQ.0) GO TO 7692
      CALL SBEAME2(N)
7692  CONTINUE
4988  CONTINUE
      IF(ICHECK.EQ.2.OR.PSAVE(LODPON1).EQ.0.) GO TO 5010
      DO 9436 I=1,NSIZE
      DO 9436 J=1,MBAND
9436  S(I,J)=0.0
      IPAR=7
      DO  9437 N=1,NUMEG
      IF(NUMEL(N).EQ.0) GO TO 9437
      CALL KEPSI01(N)
9437  CONTINUE
      IF(ICHECK.EQ.2)   GO TO 5010
      DO 3071 I=1,NEQ
      DO 3081 J=1,MBAND
      READ(4,11) RK
      SRK(I,J)=RK
      READ(16,11) RN1STAR
      SRN1(I,J)=RN1STAR
      N1STTOT(I,J)=RN1STAR
      IF(IFIX.EQ.1)  S(I,J)=SOLD(I,J)=RK
      IF(IFIX.EQ.0)  S(I,J)=SOLD(I,J)=RK+N1STTOT(I,J)
3081  CONTINUE
3071  CONTINUE
      REWIND 4
      REWIND 16
      IF(PRIOPTN.EQ.0) GO TO 7233
      WRITE(61,8005)
8005  FORMAT(///,10X,*KEPSIO MATRIX*,/)
      WRITE(61,8002)  ((SRN1(I,J),J=1,MBAND),I=1,NEQ)
      WRITE(61,8008)
8008  FORMAT(///,10X,*K LINEAR STIFFNESS MATRIX*,/)
      WRITE(61,8002)  ((SRK(I,J),J=1,MBAND),I=1,NEQ)
      WRITE(61,8009)
 8009 FORMAT(///,10X,*S(I,J)    MATRIX*,/)
      WRITE(61,8002)  ((S(I,J),J=1,MBAND),I=1,NEQ)
 7233 CONTINUE
      GO TO 5011
 5010 DO 4071 I=1,NEQ
      DO 4081 J=1,MBAND
      IF(N1OPTIN.EQ.1)READ(6,11) RN1
      IF(PSAVE(LODPON1).EQ.0.)    READ(4,11)   RK
      IF(N2OPTIN.EQ.1) READ(12,11) RN2
      IF(N1OPTIN.EQ.1)SRN1(I,J)=RN1
      IF(PSAVE(LODPON1).EQ.0..AND.N2OPTIN.EQ.1)SP(I,J)=RK+.5*RN1+RN2/3.
      IF(PSAVE(LODPON1).NE.0..AND.N2OPTIN.EQ.1)
     +SP(I,J)=SOLD(I,J)+.5*RN1+RN2/3.
      IF(PSAVE(LODPON1).EQ.0..AND.N1OPTIN.EQ.0)SP(I,J)=S(I,J)=RK
      IF(PSAVE(LODPON1).EQ.0..AND.N1OPTIN.EQ.1.AND.N2OPTIN.EQ.0)SP(I,J)=
     +RK+.5*RN1
      IF(PSAVE(LODPON1).NE.0..AND.N1OPTIN.EQ.0)SP(I,J)=S(I,J)=SOLD(I,J)
      IF(PSAVE(LODPON1).NE.0..AND.N1OPTIN.EQ.1.AND.N2OPTIN.EQ.0)SP(I,J)=
     +SOLD(I,J)+.5*RN1
      IF(PSAVE(LODPON1).EQ.0..AND.N2OPTIN.EQ.1)S(I,J)=RK+RN1+RN2
      IF(PSAVE(LODPON1).NE.0..AND.N2OPTIN.EQ.1)S(I,J)=SOLD(I,J)+RN1+RN2
      IF(PSAVE(LODPON1).EQ.0..AND.N1OPTIN.EQ.1.AND.N2OPTIN.EQ.0)
     +S(I,J)=RK+RN1
      IF(PSAVE(LODPON1).NE.0..AND.N1OPTIN.EQ.1.AND.N2OPTIN.EQ.0)
     +S(I,J)=SOLD(I,J)+RN1
 4081 CONTINUE
 4071 CONTINUE
      IF(N1OPTIN.EQ.1) REWIND 6
      IF(N2OPTIN.EQ.1) REWIND 12
      IF(PSAVE(LODPON1).EQ.0.)  REWIND 4
      IF(PRIOPTN.EQ.0) GO TO 5011
      IF(N2OPTIN.EQ.0) GO TO 4989
      WRITE(61,7693)
 7693 FORMAT(///,10X,11HN2   MATRIX,/)
      DO 7694 I=1,NEQ
      DO 7695 J=1,MBAND
      READ(12,11) RN2
      SRN2(I,J)=RN2
 7695 CONTINUE
 7694 CONTINUE
 4989 CONTINUE
      IF(N2OPTIN.EQ.1) REWIND 12
      IF(N2OPTIN.EQ.1) WRITE(61,8002)((SRN2(I,J),J=1,MBAND),I=1,NEQ)
      IF(N1OPTIN.EQ.1) WRITE(61,8004)
 8004 FORMAT(//,10X,*N1 NONLINEAR STIFFNESS MATRIX*,/)
      IF(N1OPTIN.EQ.1) WRITE(61,8002)   ((SRN1(I,J),J=1,MBAND),I=1,NEQ)
      IF(PSAVE(LODPON1).NE.0.) WRITE(61,8010)
 8010 FORMAT(///,10X,*SOLD(I,J)  MATRIX*,/)
      IF(PSAVE(LODPON1).NE.0.) WRITE(61,8002)
     +((SOLD(I,J),J=1,MBAND),I=1,NEQ)
      WRITE(61,8018)
 8018 FORMAT(///,10X,*SP(I,J) MATRIX*,/)
      WRITE(61,8002)  ((SP(I,J),J=1,MBAND),I=1,NEQ)
      WRITE(61,8009)
      WRITE(61,8002)  ((S(I,J),J=1,MBAND),I=1,NEQ)
 5011 IF(ICHECK.NE.3)  GO TO 7001
      GO TO 6001
 7001 ICHECK=3
      GO TO 3339
 3336 CONTINUE
C
C        COMPUTE ELEMENT LINEAR STIFFNESS AND ASSEMBLE INTO STRUCTURE
C        LINEAR STIFFNESS
C     ******************************************************************
C
C
C
      DO 2114  I=1,NSIZE
      DO 2114  J=1,MBAND
 2114 S(I,J)=0.0
      IPAR=2
      DO 110 N=1,NUMEG
      IF(NUMEL(N).EQ.0) GO TO 110
      CALL ELEMENT (N,ICHECK,PROTYPE)
  110 CONTINUE
C
      IF(PROTYPE.NE.3)  GO TO 3337
      DO 1071 I=1,NEQ
      DO 1081 J=1,MBAND
      READ(4,11) RK
      S(I,J)=SP(I,J)=PK
1081  CONTINUE
1071  CONTINUE
      REWIND 4
      IF(NCOND.EQ.0) GO TO 1809
      CALL STCONON
1809  CONTINUE
      IF(PRIOPTN.EQ.0) GO TO 9431
      IF(ICHECK.EQ.1) WRITE(61,8008)
      IF(ICHECK.EQ.1) WRITE(61,8002)((S(I,J),J=1,MBAND),I=1,NEQ)
9431  CONTINUE
6001  IDET=1
8002  FORMAT(1X,6(2X,E19.13),/)
      CALL LINSOLN
      DETRMNT=DET1(SCALE)
      DO 5005 I=1,NEQ
      DTOT(I)=DTOT(I)+D(I)
      DACTUAL(I)=DACTUAL(I)+D(I)
      IF(IFIX.EQ.0)  D(I)=DTOT(I)
      IF(IFIX.EQ.1)  D(I)=DACTUAL(I)
5005  CONTINUE
      IF(NCOND.NE.0)  CALL RECOVER
      CALL IDENT
      IF(JUSTK.EQ.1)  GO TO 3339
      IF(ITERCHK.EQ.0) CALL INVTRNS
      IF(ITERCHK.NE.0) GO TO 8537
      DO 5763  NN=1,NUMEG
      IF(NUMEL(NN).EQ.0.)  GO TO 5763
      NAME=NUMEL(NN)
      DO 5002  K=1,NAME
      M=L(NN,K)
5002  A7TOT(M)=A7OLD(M)+ULOC(M,7)-ULOC(M,1)
5763  CONTINUE
8537  CONTINUE
      ICHECK=2
      IF(ITERCHK.NE.0) GO TO 2120
      DO 2121 I=1,NEQ
      R(I)=0.0
2121  PACTUAL(I)=PSAVE(I)+PSTART(I)
      GO TO 3342
2120  CONTINUE
      GO TO 1901
3339  DO 2001 I=1,NEQ
      PTEMP(I)=0.0
      IM=I+1
      IF(IM.GT.NEQ) GO TO 2001
      DO 3901 J=2,MBAND
      IF(SP(I,J).EQ.0.)  GO TO 1804
      PTEMP(I)=PTEMP(I)+SP(I,J)*D(IM)
1804  IM=IM+1
      IF(IM.GT.NEQ)  GO TO 2001
3901  CONTINUE
2001  CONTINUE
      DO 2301 I=1,NEQ
      IM=I
      JM=1
2108  IF(SP(IM,JM).EQ.0.) GO TO 2201
      PTEMP(I)=PTEMP(I)+SP(IM,JM)*D(IM)
2201  IM=IM-1
      JM=JM+1
      IF(IM.EQ.0)        GO TO 2301
      IF(JM.GT.MBAND)  GO TO 2301
      GO TO 2108
2301  CONTINUE
      DO 5006 I=1,NEQ
      IF(IFIX.EQ.0)  PACTUAL(I)=PTEMP(I)+PSAVE(I)
      IF(IFIX.EQ.1)  PACTUAL(I)=PTEMP(I)
5006  CONTINUE
      IF(ITERCHK.EQ.0) GO TO 6975
      IF(PRIOPTN.EQ.1) WRITE(61,8011)
8011  FORMAT(15X,*I*,5X,*PACTUAL(I)*,10X,*PTEMP(I)*,10X,*PSAVE(I)*,//)
      IF(PRIOPTN.EQ.0) GO TO 4990
      DO 8012 I=1,NEQ
      WRITE(61,8013)  I,PACTUAL(I),PTEMP(I),PSAVE(I)
8012  CONTINUE
4990  CONTINUE
8013  FORMAT(10X,I5,5X,E21.15,10X,E21.15,10X,E21.15,/)
      WRITE(61,8547)
8547  FORMAT(/,10X,*DTOT(I)*,/)
      DO 8541 MM=1,NEQ
8541  WRITE(61,8542) DACTUAL(MM)
```

```
8542  FORMAT(10X,E21.15)
      WRITE(61,9001)  PACTUAL(LODPON1),DACTUAL(LODPON1)
9001  FORMAT(10X,8HPACTUAL=,F15.9/10X,13HDISPLACEMENT=,F15.9,//)
6975  CONTINUE
      DO 2115 I=1,NEQ
2115  R(I)=PSTART(I)-PTEMP(I)
      IF(PRIOPTN.EQ.0) GO TO 6976
      WRITE(61,9731)
9731  FORMAT(//,20X,*R(I)*,15X,*PSTART(I)*,15X,*PTEMP(I)*,/)
      DO 9732  IMM=1,3
9732  WRITE(61,9733)  R(IMM),PSTART(IMM),PTEMP(IMM)
9733  FORMAT(//,10X,E21.15,10X,E21.15,10X,E21.15,/)
6976  CONTINUE
      IF(ITERCHK.EQ.0)     GO TO 3342
      DO 2451  NN=1,NUMEG
      IF(NUMEL(NN).EQ.0)  GO TO 2451
      NAME=NUMEL(NN)
      DO 2351  K=1,NAME
      M=L(NN,K)
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 2351  K1=1,2
      IF(K1.EQ.1)  NP=NI
      IF(K1.EQ.2)  NP=NJ
      DO 2251  I=1,6
      IF(IA(NP,I))  1651,1551,1571
1571  NL=IA(NP,I)
      REFSTRT(NP,I)=PSTART(NL)
      REFPTMP(NP,I)=PTEMP(NL)
      GO TO 2251
1551  REFSTRT(NP,I)=0.0
      REFPTMP(NP,I)=0.0
      GO TO 2251
1651  IF(IB(NP,I).LT.0)  GO TO 1751
      NM=IB(NP,I)
      GO TO 1851
1751  NL=-IB(NP,I)+NEQ
      REFSTRT(NP,I)=PSTART(NL)
      REFPTMP(NP,I)=PTEMP(NL)
      GO TO 2251
1851  IF(IA(NM,I)) 1951,2051,2151
1951  NL=-IB(NM,I)+NEQ
      REFSTRT(NP,I)=PSTART(NL)
      REFPTMP(NP,I)=PTEMP(NL)
      GO TO 2251
2051  REFSTRT(NP,I)=0.0
      REFPTMP(NP,I)=0.0
      GO TO 2251
2151  NL=IA(NM,I)
      REFSTRT(NP,I)=PSTART(NL)
      REFPTMP(NP,I)=PTEMP(NL)
2251  CONTINUE
2351  CONTINUE
2451  CONTINUE
      DO 6949 NP=1,NUMNP
      DO 6949 J=1,3
      KJJ=J+3
      PART1=ABS(REFSTRT(NP,J)-REFPTMP(NP,J))
      PART2=ABS(REFSTRT(NP,KJJ)-REFPTMP(NP,KJJ))
      IF(PART1.GT.DELTA1.OR.PART2.GT.DELTA2)  GO TO 6950
      WRITE(61,2116)  PART1,PART2
6949  CONTINUE
      GO TO 3342
6950  WRITE(61,2116)  PART1,PART2
2116  FORMAT(10X,6HPART1=,E21.15,10X,6HPART2=,E21.15)
      NUMITER=NUMITER+1
      IF(NUMITER.LE.MAXITER)  GO TO  2117
      GO TO 900
2117  GO TO 6001
3342  CONTINUE
      IF(ITERCHK.NE.1)  GO TO 8945
      IF(MSUOPTN.EQ.2)  GO TO 8945
      DO 8538  NN=1,NUMEG
      IF(NUMEL(NN).EQ.0)  GO TO 8538
      NAME=NUMEL(NN)
      DO 8539 K=1,NAME
      M=L(NN,K)
      T0=(ULOC(M,8)-ULOC(M,2))/LE(M)
      SI0=(ULOC(M,3)-ULOC(M,9))/LE(M)
      TA=ULOC(M,6)-T0
      TB=ULOC(M,12)-T0
      SIA=ULOC(M,5)-SIO
      SIB=ULOC(M,11)-SIO
 8539 A7TOT(M)=A7OLD(M)+ULOC(M,7)-ULOC(M,1)
     ++.5*(TO**2+SIO**2)*LE(M)
     ++LE(M)*(2.*TA**2-TA*TB+2.*TB**2)/30.
     ++LE(M)*(2.*SIA**2-SIA*SIB+2.*SIB**2)/30.
 8538 CONTINUE
 3945 IF(ITERCHK.NE.1)  GO TO 4993
      IF(MSUOPTN.EQ.1)  GO TO 4993
      DO 4992  NN=1,NUMEG
      IF(NUMEL(NN).EQ.0)   GO TO 4992
      NAME=NUMEL(NN)
      DO 5344  K=1,NAME
      M=L(NN,K)
      ALFA1=ULOC(M,6)
      ALFA2=2.*(-3.*ULOC(M,2)-2.*ULOC(M,6)*LE(M)+3.*ULOC(M,8)-
     +ULOC(M,12)*LE(M))/LE(M)
      ALFA3=3.*(2.*ULOC(M,2)+ULOC(M,6)*LE(M)-2.*ULOC(M,8)+ULOC(M,12)
     +*LE(M))/LE(M)
      BETA1=-ULOC(M,5)
      BETA2=2.*(-3.*ULOC(M,3)+2.*ULOC(M,5)*LE(M)+3.*ULOC(M,9)
     ++ULOC(M,11)*LE(M))/LE(M)
      BETA3=3.*(2.*ULOC(M,3)-ULOC(M,5)*LE(M)-2.*ULOC(M,9)-
     +ULOC(M,11)*LE(M))/LE(M)
      BE(1)=(-ULOC(M,1)+ULOC(M,7))/LE(M)+(ALFA1**2+BETA1**2)/2.
      BE(2)=ALFA1*ALFA2+BETA1*BETA2
      BE(3)=(ALFA2**2+BETA2**2)/2.+ALFA1*ALFA3+BETA1*BETA3
      BE(4)=ALFA2*ALFA3+BETA2*BETA3
      BE(5)=(ALFA3**2+BETA3**2)/2.
      DO 5343  I=1,5
 5343 BTO(M,I)=BOL(M,I)+BE(I)
 5344 CONTINUE
 4992 CONTINUE
 4993 CONTINUE
      WRITE(61,8649)
 8649 FORMAT(/,10X,*DACTUAL(I)*,/)
      DO 8653  I=1,NEQ
 8653 WRITE(61,8654)    DACTUAL(I)
 8654 FORMAT(10X,E21.15)
      WRITE(61,399) PACTUAL(LODPON1),DACTUAL(LODPON1),DACTUAL(LODPON2)
     +,DACTUAL(LODPON3),DETRMNT,NUMITER
      IF(DETRMNT.LE.0..AND.DETOPTN.EQ.1)        GO TO 900
      IF(ABS(PACTUAL(LODPON1)).GE.ABS(PTOT1))  GO TO 900
      DO 5007 I=1,NEQ
      PSAVE(I)=PACTUAL(I)
      DTOT(I)=0.0
 5007 CONTINUE
      IF(JUSTK.EQ.1)  GO TO 2118
      DO 5281  NN=1,NUMEG
      IF(NUMEL(NN).EQ.0)   GO TO 5281
      NAME=NUMEL(NN)
      DO 5342  K=1,NAME
      M=L(NN,K)
      IF(MSUOPTN.EQ.1)   A7OLD(M)=A7TOT(M)
      DO 5342  I=1,5
      IF(MSUOPTN.EQ.2)   BOL(M,I)=BTO(M,I)
 5342 CONTINUE
 5281 CONTINUE
 2118 CONTINUE
      R(LODPON1)=R(LODPON1)+PINC1
      R(LODPON2)=R(LODPON2)+PINC2
      R(LODPON3)=R(LODPON3)+PINC3
      IF(NEQ.GE.4) R(LODPON4)=R(LODPON4)+PINC4
      IF(NEQ.GE.5) R(LODPON5)=R(LODPON5)+PINC5
      IF(NEQ.GE.6) R(LODPON6)=R(LODPON6)+PINC6
      DO  2119  I=1,NEQ
      IF(IFIX.EQ.0)  PSTART(I)=R(I)
      IF(IFIX.EQ.1)  PSTART(I)=R(I)+PSAVE(I)
 2119 CONTINUE
      IF(PRIOPTN.EQ.0) GO TO 6977
      WRITE(61,9735)
 9735 FORMAT(///,10X,*R(I)*,/)
      DO 9736  IMM=1,NEQ
 9736 WRITE(61,9737)  R(IMM)
 9737 FORMAT(//,10X,E21.15,/)
 6977 CONTINUE
      ICHECK=3
      GO TO 1001
 3337 CONTINUE
C
C         CONDENSE LINEAR STIFFNESS AND LOAD VECTOR OF STRUCTURE
```

```
C     ******************************************************************
      IF(NCOND.EQ.0) GO TO 801
      CALL STCONDN
801   CONTINUE
C
C
C         SOLVE SYSTEM OF LINEAR EQUATIONS S*D=R
C     ******************************************************************
      IF(PROTYPE.NE.1) GO TO 601
      SCALE=10000.
      PLOAD1=PINIT1
      PLOAD2=PINIT2
      PLOAD3=PINIT3
      IF(NEQ.GE.4) PLOAD4=PINIT4
      IF(NEQ.GE.5) PLOAD5=PINIT5
      IF(NEQ.GE.6) PLOAD6=PINIT6
299   R(LODPON1)=PLOAD1
      R(LODPON2)=PLOAD2
      R(LODPON3)=PLOAD3
      IF(NEQ.GE.4) R(LODPON4)=PLOAD4
      IF(NEQ.GE.5) R(LODPON5)=PLOAD5
      IF(NEQ.GE.6) R(LODPON6)=PLOAD6
      IF(PLOAD1.NE.PINIT1)  GO TO 777
      IDET=1
601   CALL LINSOLN
      IF(PROTYPE.EQ.2)  GO TO 1778
      IF(R(LODPON1).EQ.PINIT1) CALL IDENT
      IF(R(LODPON1).EQ.PINIT1) CALL INVTRNS
C     ******************************************************************
C         IN CASE WHICH WE WANT THE END FORCES DUE TO
C         THE LINEAR SOLUTION SUBROUTINE ENDFORC MAY BE CALLED
C         AT THIS STAGE(THE FIRST ITERATION OF THE FIRST
C         LOAD INCREMENT)
C     ******************************************************************
      IF(R(LODPON1).EQ.PINIT1.AND.ISTRESS.EQ.1) CALL ENDFORC
      IF(PROTYPE.NE.1) GO TO 1778
      DETER=DET1(SCALE)
      WRITE(61,399)  PLOAD1,D(LODPON1),D(LODPON2),D(LODPON3),DETER,
     *NUMITER
      GO TO 709
C
777   NUMITER=0
C         RECOVER INTERNAL D.O.F."S OF STRUCTURE
C     ******************************************************************
C
      DO 1555 I=1,NEQ
1555  DTEMP(I)=D(I)
      IF(PROTYPE.NE.1) GO TO 715
778   NUMITER=NUMITER+1
715   CONTINUE
      NN=MAXITER+1
      IF(NUMITER.EQ.NN)  GO TO 9999
1778  IF(NCOND.NE.0)  CALL RECOVER
C
C         IDENTIFY DISPLACEMENTS FOUND FROM SOLUTION OF S*D=R AND FROM
C         THE RECOVERY PROCESS
C     ******************************************************************
      CALL IDENT
C         TO HAVE NODAL DEGREES OF FREEDOM IN LOCAL COORDINATES
C     ******************************************************************
      CALL INVTRNS
C
C
C
      DO 180 I=1,NSIZE
      DO 180 J=1,MBAND
180   S(I,J)=0.0
      IPAR=3
      DO 210 N=1,NUMEG
      IF(NUMEL(N).EQ.0) GO TO 210
      CALL SBEAME1(N)
210   CONTINUE
      IF(NCOND.EQ.0) GO TO 802
      CALL STCONDN
802   CONTINUE
      IF(N2OPTIN.EQ.0)  GO TO 4991
      DO 190 I=1,NSIZE
      DO 190 J=1,MBAND
190   S(I,J)=0.0
      IPAR=4
      DO 310 N=1,NUMEG
      IF(NUMEL(N).EQ.0) GO TO 310
      CALL SBEAME2(N)
310   CONTINUE
4991  CONTINUE
      IF(NCOND.EQ.0) GO TO 899
      CALL STCONDN
899   CONTINUE
      IF(PROTYPE.EQ.1) GO TO 222
      IF(PROTYPE.EQ.2.AND.EIGVALU.EQ.1)   CALL EIGENVL(EIGEN,IDATA)
      IF(EIGVALU.EQ.2) GO TO 444
      GO TO 900
222   CONTINUE
      DO 107 I=1,NEQ
      DO 108 J=1,MBAND
      READ(4,11) RK
      READ(6,11) RN1
      IF(N2OPTIN.EQ.1) READ(12,11) RN2
      IF(N2OPTIN.EQ.0) S(I,J)=RK+.5*RN1
      IF(N2OPTIN.EQ.1) S(I,J)=RK+.5*RN1+RN2/3.
      IF(N2OPTIN.EQ.0) SP(I,J)=RK+RN1
      IF(N2OPTIN.EQ.1) SP(I,J)=RK+RN1+RN2
108   CONTINUE
107   CONTINUE
      REWIND 4
      REWIND 6
      IF(N2OPTIN.EQ.1) REWIND 12
      IF(NUMITER.EQ.1) GO TO 701
      GO TO 702
9999  PINC1=PINC1/2.
      PINC2=PINC2/2.
      PINC3=PINC3/2.
      IF(NEQ.GE.4) PINC4=PINC4/2.
      IF(NEQ.GE.5) PINC5=PINC5/2.
      IF(NEQ.GE.6) PINC6=PINC6/2.
      PLOAD1=PLOAD1-PINC1
      PLOAD2=PLOAD2-PINC2
      PLOAD3=PLOAD3-PINC3
      IF(NEQ.GE.4) PLOAD4=PLOAD4-PINC4
      IF(NEQ.GE.5) PLOAD5=PLOAD5-PINC5
      IF(NEQ.GE.6) PLOAD6=PLOAD6-PINC6
      DO 333 I=1,NEQ
333   D(I)=DTEMP(I)
      WRITE(61,1899)   PLOAD1,PINC1
1899  FORMAT(15X,23HLOADINCREMENT IS HALVED/
     +15X,6HPLOAD=,F10.5/15X,5HPINC=,F10.5)
      GO TO 1777
701   UOLD=D(LODPON1)
      GO TO 778
702   IF(ABS((UOLD-D(LODPON1))/D(LODPON1)).LE.TOLER)  GO TO 708
      UOLD=D(LODPON1)
      GO TO 778
708   DO 2555 I=1,NEQ
2555  DTEMP(I)=D(I)
      IDET=3
      WRITE(61,1399) PLOAD1,D(LODPON1),NUMITER
1399  FORMAT(//,10X,5HLOAD=,F10.5/10X,7HDEFLEC=,F15.10/
     +10X,11HITERATIONS=,I5)
      DETER=DET1(SCALE)
      WRITE(61,399)  PLOAD1,D(LODPON1),D(LODPON2),D(LODPON3),DETER,
     +NUMITER
      IF(DETER.LE.0..AND.DETOPTN.EQ.1)    GO TO 900
709   PLOAD1=PLOAD1+PINC1
      PLOAD2=PLOAD2+PINC2
      PLOAD3=PLOAD3+PINC3
      IF(NEQ.GE.4) PLOAD4=PLOAD4+PINC4
      IF(NEQ.GE.5) PLOAD5=PLOAD5+PINC5
      IF(NEQ.GE.6) PLOAD6=PLOAD6+PINC6
1777  IF(ABS(PLOAD1).GT.ABS(PTOT1)) GO TO 900
      GO TO 299
C
444   CONTINUE
C     ******************************************************************
C        TO HAVE EIGENVALUE SOLUTION USING DETERMINANT SEARCH METHOD
C        INTHE CASE OF IEIGEN=1  (N1) STIFFNESS MATRIX WOULD
C        BE CONSIDERED IN SUBROUTINE NLEIGNP FOR NONLINEAR
C        EIGENVALUE PROBLEM.
C        FOR IEIGEN=2  (N1+K)  WOULD BE CONSIDERED
C     ******************************************************************
      IEIGEN=1
      SCALE=10000.
      CALL NLEIGNP(SCALE)
```

```
900   CONTINUE
C
1     FORMAT(9I5)
10    FORMAT(I5)
11    FORMAT(E21.15)
399   FORMAT(///,10X,5HLOAD=,F15.9/10X,*D(LODPON1)=*,F15.10/
     +10X,*D(LODPON2)=*,F15.10/10X,*D(LODPON3)=*,F15.10/
     +10X,12HDETERMINANT=,E25.15/10X,11HITERATIONS=,I5)
699   FORMAT(6F10.6,I5)
799   FORMAT(10X,*PINIT1=*,F15.8,10X,*PINC1=*,F15.8/
     +10X,*PTOT1=*,F15.8,10X,*PINIT2=*,F15.8,10X,*PINC2=*,F15.8/
     +10X,*PTOT2=*,F15.8,10X,*MAXITER=*,I5)
1010  FORMAT(A10,A10,A10,7I5)
1015  FORMAT(I5,6F10.1)
1020  FORMAT(I5)
2010  FORMAT(*1*,A10,A10,A10//7X,6HNE    =,I3//7X,6HNUMNP=,I3//7X,
     +        6HNUMEG=,I3//7X,6HIDATA=,I3//7X,6HICAL1=,I3//7X,6HICAL2=,
     +        I3//7X,6HICAL3=,I3)
2015  FORMAT(*1*,15H  INITIAL LOADS//7H   NODE,27X,14HLOAD DIRECTION//
     +        7H NUMBER,13X,1HX,9X,1HY,9X,1HZ,9X,2HTX,8X,2HTY,8X,2HTZ//)
2020  FORMAT(I5,6X,6F10.5)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE NODDATA(IGOPTIN)
C
C     ******************************************************************
C        TO READ AND PRINT NODAL POINT DATA
C        TO CALCULATE EQUATION NUMBERS AND CONDENSATION NUMBERS AND
C        STORE THEM IN ARRAYS  -IA- AND -IB- RESPECTIVELY
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
C
C     ******************************************************************
C        READ NODAL POINT DATA
C        EXPRESSIONS FOR X(N) AND Y(N) SHOULD BE CHANGED
C        ACCORDING TO CURVE DEFINING THE ARCH.
C     ******************************************************************
C
      WRITE(61,2000)
      WRITE(61,2010)
      WRITE(61,2015)
      IF(IGOPTIN.EQ.0) GO TO 100
      IF(IGOPTIN.EQ.2) GO TO 202
      READ(60,10)  ALFZERO,RADIUS
      WRITE(61,20) ALFZERO,RADIUS
      ALFINC=ALFZERO/NE
      DO 201 I=1,NUMNP
      Z(I)=0.0
      X(I)=RADIUS*SIN((-ALFZERO/2)+(I-1)*ALFINC)
      Y(I)=SQRT(RADIUS**2-X(I)**2)
201   CONTINUE
      GO TO 204
202   READ(60,10) RISE,SPAN
      WRITE(61,30) RISE,SPAN
      DO 203 I=1,NUMNP
      Z(I)=0.0
      X(I)=-SPAN/2+(I-1)*SPAN/NE
      Y(I)=RISE-4*RISE*X(I)**2/(SPAN**2)
203   CONTINUE
204   CONTINUE
90    READ(60,1001)  N,(IA(N,I),I=1,6),(IB(N,I),I=1,6)
      WRITE(61,2020) N,(IA(N,I),I=1,6),(IB(N,I),I=1,6),X(N),Y(N),Z(N)
      IF(N.NE.NUMNP) GO TO 90
      GO TO 101
100   READ(60,1000)  N,(IA(N,I),I=1,6),(IB(N,I),I=1,6),X(N),Y(N),Z(N)
      WRITE(61,2020) N,(IA(N,I),I=1,6),(IB(N,I),I=1,6),X(N),Y(N),Z(N)
      IF(N.NE.NUMNP) GO TO 100
C
C     ******************************************************************
C        PROCESS ARRAYS -IA- AND -IB- TO FIND EQUATION NUMBERS AND
```

```
C         CONDENSATION NUMBERS. STORE NEQ"S AND NCOND"S IN ARRAYS IA AND
C         IB RESPECTIVELY.
C      *****************************************************************
C
101    NEQ=0
       NCOND=0
       DO 125 N=1,NUMNP
       DO 120 I=1,6
       IF(IA(N,I).NE.1) GO TO 105
       IA(N,I)=0
       GO TO 120
105    IA(N,I)=-1
       IF(IB(N,I)) 110,115,120
110    NCOND=NCOND+1
       IB(N,I)=-NCOND
       GO TO 120
115    NEQ=NEQ+1
       IA(N,I)=NEQ
120    CONTINUE
125    CONTINUE
       NSIZE=NEQ+NCOND
C
C         WRITE GENERATED NODAL POINT DATA
C      *****************************************************************
C
       WRITE(61,2030)
       WRITE(61,2040)
       WRITE(61,2050)(N,(IA(N,I),I=1,6),(IB(N,I),I=1,6),N=1,
      +NUMNP)
       WRITE(61,2060) NSIZE,NEQ,NCOND
       RETURN
C
10     FORMAT(2F15.10)
20     FORMAT(//,10X,5HALFA=,F15.10,10X,7HRADIUS=,F15.10,//)
30     FORMAT(//,10X,5HRISE=,F15.10,10X,5HSPAN=,F15.10,//)
1000   FORMAT(I5,12I3,3F10.5)
1001   FORMAT(I5,12I3)
2000   FORMAT(*1*,33H N O D A L   P O I N T   D A T A   //)
2010   FORMAT(18H INPUT NODAL DATA //)
2015   FORMAT(7H   NODE,26X,35HNODAL POINT BOUNDARY CONDITION CODES,33X,
      +         23HNODAL POINT COORDINATES/7H NUMBER,21X,7HIA(N,I),33X,
      +7HIB(N,I)/11X,2(4X,1HX,4X,1HY,4X,1HZ,4X,2HTX,3X,2HTY,3X,
      +2HTZ,10X),9X,4HX(N),8X,4HY(N),8X,4HZ(N))
2020   FORMAT(I5,6X,6I5,11X,6I5,14X,3F12.3)
2030   FORMAT(///22H GENERATED NODAL DATA //)
2040   FORMAT(7H   NODE,16X,16HEQUATION NUMBERS,22X,
      +         20HCONDENSATION NUMBERS/7H NUMBER,21X,7HIA(N,I),33X,
      +         7HIB(N,I)/11X,2(4X,1HX,4X,1HY,4X,1HZ,4X,2HTX,3X,2HTY,3X,2HT
      +Z,10X))
2050   FORMAT(I5,6X,6I5,11X,6I5)
2060   FORMAT(*-*,6HNSIZE=,I3,3X,4HNEQ=,I3,3X,6HNCOND=,I3)
C
       END
C
C
C
C
C
C
C
       SUBROUTINE ELEMENT (N,ICHECK,PROTYPE)
C
C      *****************************************************************
C         TO CALL THE APPROPRIATE ELEMENT SUBROUTINE
C      *****************************************************************
C
       COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
      +ISTRESS
       INTEGER PROTYPE
C
       IF(N.EQ.1)  CALL BEAM(N,ICHECK,PROTYPE)
       IF(N.EQ.2)  CALL TRUSS(N,ICHECK,PROTYPE)
       RETURN
C
       END
C
C
C
C
C
C
```

```
C
      SUBROUTINE BAND
C
C     ****************************************************************
C        TO COMPUTE SEMIBANDWIDTH OF STRUCTURE STIFFNESS MATRIX
C        DONE BY FINDING THE MAXIMUM DIFFERENCE BETWEEN THE
C        EQUATION NUMBERS ASSOTIATED WITH THE NODES OF A
C        PARTICULAR ELEMENT
C     ****************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36)
     +,L(3,36),IZZ(36),YPGM(36),ZPGM(36)
C
      MBAND=0
      ICONTRL=0
      DO 900 M=1,NE
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 800 I=1,6
      IF(ICONTRL.EQ.1)  GO TO 1001
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0.
     +AND.IA(NI,4).LE.0.AND.IA(NI,5).LE.0.AND.IA(NI,6).LE.0) GO TO 199
 1001 CONTINUE
      IF(IA(NI,I).LE.0) GO TO 800
      N1=IA(NI,I)
      GO TO 99
  199 ICONTRL=1
      N1=0
   99 DO 700 J=1,6
      IF(ICONTRL.EQ.1)  GO TO 1002
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0.
     +AND.IA(NJ,4).LE.0.AND.IA(NJ,5).LE.0.AND.IA(NJ,6).LE.0) GO TO 399
 1002 CONTINUE
      GO TO 499
  399 ICONTRL=1
      MB=N1
      GO TO 299
  499 IF(IA(NJ,J).LE.0) GO TO 700
      N2=IA(NJ,J)
      MB=IABS(N2-N1)
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0.
     +AND.IA(NI,4).LE.0.AND.IA(NI,5).LE.0.AND.IA(NI,6).LE.0) GO TO 299
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0.
     +AND.IA(NJ,4).LE.0.AND.IA(NJ,5).LE.0.AND.IA(NJ,6).LE.0) GO TO 299
      MB=MB+1
  299 IF(MB.GT.MBAND)  MBAND=MB
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0.
     +AND.IA(NJ,4).LE.0.AND.IA(NJ,5).LE.0.AND.IA(NJ,6).LE.0) GO TO 800
  700 CONTINUE
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0.
     +AND.IA(NI,4).LE.0.AND.IA(NI,5).LE.0.AND.IA(NI,6).LE.0) GO TO 900
  800 CONTINUE
  900 CONTINUE
      WRITE(61,2000) MBAND
      RETURN
C
 2000 FORMAT(*1*,20HSEMIBANDWIDTH MBAND=,I3)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE BEAM(N,ICHECK,PROTYPE)
C
C     ****************************************************************
C        BEAM ELEMENT SUBROUTINE
C        ONLY STRAIGHT BEAM ELEMENTS ARE CONSIDERED
C     ****************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/4/SE(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36)
     +,L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/8/PI(37,6),PN(37,6),R(107)
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/10/D(60),G1(60),G2(60),G3(60),G4(60),RC(60),
     +SC(60,20),IGAUS
      REAL IXX,IYY,IZZ,KT,LE
      INTEGER PROTYPE
      IF(IPAR.EQ.2)  GO TO 200
C
      IF(PROTYPE.NE.3)  GO TO 1001
      IF(ICHECK.EQ.1)   GO TO 1001
      DO 5555 M=1,NE
      XI=X(NODEI(M))
      YI=Y(NODEI(M))
      ZI=Z(NODEI(M))
      XJ=X(NODEJ(M))
      YJ=Y(NODEJ(M))
      ZJ=Z(NODEJ(M))
      LE(M)=SQRT((XJ-XI)**2+(YJ-YI)**2+(ZJ-ZI)**2)
5555  CONTINUE
      GO TO 120
C
C        READ MATERIAL INFORMATION
C     ****************************************************************
1001  WRITE(61,2000) N
      READ(60,1010) E(N),G(N)
      WRITE(61,2020) NUMEL(N),E(N),G(N)
C
C        READ ELEMENT AND CROSS SECTION INFORMATION
C     ****************************************************************
      WRITE(61,2021)
      K=0
105   READ(60,1020) M,NODEI(M),NODEJ(M),A(M),YPGM(M),ZPGM(M),
     +IXX(M),IZZ(M),KT(M)
      WRITE(61,2022) M,NODEI(M),NODEJ(M),A(M),YPGM(M),ZPGM(M),
     +IXX(M),IZZ(M),KT(M)
      K=K+1
      L(N,K)=M
      NI=NODEI(M)
      NJ=NODEJ(M)
      AA=(X(NJ)-X(NI))**2
      B=(Y(NJ)-Y(NI))**2
      C=(Z(NJ)-Z(NI))**2
      LE(M)=SQRT(AA+B+C)
      IF(K.NE.NUMEL(N)) GO TO 105
120   CONTINUE
      RETURN
C
200   CONTINUE
C
C        CALCULATE AND STORE LINEAR STIFFNESS MATRIX OF BEAM ELEMENTS
C     ****************************************************************
      NAME=NUMEL(N)
      DO 220 K=1,NAME
      M=L(N,K)
C        TERMS OF STIFFNESS MATRIX OF BEAM ELEMENTS IN LOCAL COORDINATES
C     ****************************************************************
      DO 400 I=1,12
      DO 500 J=1,I
      SE(I,J)=0.0
500   CONTINUE
400   CONTINUE
      SE(1,1)=SE(7,7)=E(N)*A(M)/LE(M)
      SE(7,1)=-SE(1,1)
      SE(2,2)=SE(8,8)=12.*IXX(M)*E(N)/LE(M)**3
      SE(8,2)=-SE(2,2)
      SE(3,3)=SE(9,9)=12.*IZZ(M)*E(N)/LE(M)**3
      SE(9,3)=-SE(3,3)
      SE(10,10)=SE(4,4)=G(N)*KT(M)/LE(M)
      SE(10,4)=-SE(10,10)
      SE(6,6)=SE(12,12)=4.*IXX(M)*E(N)/LE(M)
      SE(5,5)=SE(11,11)=4.*IZZ(M)*E(N)/LE(M)
      SE(12,2)=SE(6,2)=6.*IXX(M)*E(N)/LE(M)**2
      SE(8,6)=SE(12,8)=-SE(12,2)
      SE(11,3)=SE(5,3)=-6.*IZZ(M)*E(N)/LE(M)**2
      SE(9,5)=SE(11,9)=-SE(11,3)
      SE(12,6)=2.*IXX(M)*E(N)/LE(M)
      SE(11,5)=2.*IZZ(M)*E(N)/LE(M)
C
```

```
C          FILL-IN UPPER HALF OF MATRIX BY SYMMETRY
C       ***************************************************************
        DO 210 I=1,12
        DO 210 J=I,12
210     SE(I,J)=SE(J,I)
        IF(ICAL3.EQ.0) WRITE(7,10) ((SE(I,J),J=1,12),I=1,12)
        IF(ISTRESS.EQ.0)  GO TO 601
        WRITE(7,10) ((SE(I,J),J=1,12),I=1,12)
601     CONTINUE
C
C
C          TO TRANSFORM SE(12,12) FROM LOCAL TO GLOBAL COORDINATE
C       ***************************************************************
C          SE(12,12) IS THE  STIFFNESS  MATRIX IN GLOBAL
        CALL TRANSFM(M)
        IF(ICAL3.EQ.0) WRITE(8,10) ((SE(I,J),J=1,12),I=1,12)
        IF(ISTRESS.EQ.0)  GO TO 602
        WRITE(8,10) ((SE(I,J),J=1,12),I=1,12)
602     CONTINUE
C
C          ASSEMBLE STIFFNESS OF EACH ELEMENT INTO STRUCTURAL
C       ***************************************************************
C          LINEAR STIFFNESS
        CALL ASEMBLE(M)
220     CONTINUE
        IF(ICAL3.EQ.0) REWIND 7
        IF(ICAL3.EQ.0) REWIND 8
        IF(ISTRESS.EQ.1) REWIND 7
        IF(ISTRESS.EQ.1) REWIND 8
        WRITE(4,10) ((S(I,J),J=1,MBAND),I=1,NSIZE)
        REWIND 4
C          TO PRINT ENTRIES OF EACH ELEMENT
C          STIFFNESS MATRIX IN LOCAL
C       ***************************************************************
        IF(ICAL3.NE.0) GO TO 251
        WRITE(61,2054)
        NAME=NUMEL(N)
        DO 250 M=1,NAME
        WRITE(61,2053) M
        READ(7,10) ((SE(I,J),J=1,12),I=1,12)
        WRITE(61,2030)
        WRITE(61,2032) ((SE(I,J),J=1,6),I=1,6)
        WRITE(61,2034)
        WRITE(61,2032) ((SE(I,J),J=7,12),I=1,6)
        WRITE(61,2036)
        WRITE(61,2032) ((SE(I,J),J=1,6),I=7,12)
        WRITE(61,2038)
        WRITE(61,2032) ((SE(I,J),J=7,12),I=7,12)
250     CONTINUE
251     CONTINUE
        REWIND 7
C          TO PRINT ENTRIES OF EACH ELEMENT
C          STIFFNESS MATRIX IN GLOBAL
C       ***************************************************************
        IF(ICAL3.NE.0) GO TO 230
        WRITE(61,2052)
        NAME=NUMEL(N)
        DO 242 M=1,NAME
        WRITE(61,2053) M
        READ(8,10)    ((SE(I,J),J=1,12),I=1,12)
        WRITE(61,2030)
        WRITE(61,2032) ((SE(I,J),J=1,6),I=1,6)
        WRITE(61,2034)
        WRITE(61,2032) ((SE(I,J),J=7,12),I=1,6)
        WRITE(61,2036)
        WRITE(61,2032) ((SE(I,J),J=1,6),I=7,12)
        WRITE(61,2038)
        WRITE(61,2032) ((SE(I,J),J=7,12),I=7,12)
242     CONTINUE
        REWIND 8
230     CONTINUE
        RETURN
C
10      FORMAT(E21.15)
1010    FORMAT(2E10.2)
1020    FORMAT(3I5,3F10.6,3F10.4)
2000    FORMAT(*1*,23HG R O U P  N U M B E R ,I2//2X,6HNUMBER,6X,7HMODULUS
     +       ,11X,5HSHEAR/4X,2HOF,11X,2HOF,12X,7HMODULUS/1X
     +           ,8HELEMENTS,4X,10HELASTICITY)
2020    FORMAT(I6,2E17.6)
2021    FORMAT(//8H ELEMENT,3X,8HNODEI(M),3X,8HNODEJ(M),12X,4HA(M),10X,
     +7HYPGM(M),9X,7HZPGM(M),7X,6HIXX(M),8X,6HIZZ(M)
     +,10X,5HKT(M))
2022  FORMAT(I6,5X,I5,6X,I5,6X,3F15.6,3E15.4)
2030  FORMAT(*1*,33HSTIFFNESS MATRIX OF BEAM ELEMENTS///9H BLOCK II)
2032  FORMAT(//1X,6F20.6)
2034  FORMAT(///9H BLOCK IJ)
2036  FORMAT(*1*//9H BLOCK JI)
2038  FORMAT(///9H BLOCK JJ)
2052  FORMAT(///,5X,*ENTRIES OF EACH ELEMENT
     + STIFFNESS MATRIX IN GLOBAL*)
2053  FORMAT(///,10X,15HELEMENT NUMBER ,I2)
2054  FORMAT(///,5X,*ENTRIES OF EACH ELEMENT STIFFNESS
     +MATRIX IN LOCAL *)
      END
C
C
C
C
C
C
C
      SUBROUTINE TRANSFM(M)
C
C     ******************************************************************
C
C        TO TRANSFER FROM LOCAL TO GLOBAL COORDINATES
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/4/SE(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36)
     +,L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/12/JLOC(36,12),U(12),RCOL(9),MSUOPTN,N1GOPTN
      DIMENSION SENEW(12,12)
      REAL  LE,LAMBDA
C
C
C        TO EVALUATE ROTATION MATRIX FOR ELEMENT(M)
C     ******************************************************************
      XI=X(NODEI(M))
      YI=Y(NODEI(M))
      ZI=Z(NODEI(M))
      XJ=X(NODEJ(M))
      YJ=Y(NODEJ(M))
      ZJ=Z(NODEJ(M))
      LE(M)=SQRT((XJ-XI)**2+(YJ-YI)**2+(ZJ-ZI)**2)
      RCOL(1)=CX=(XJ-XI)/LE(M)
      RCOL(2)=CY=(YJ-YI)/LE(M)
      RCOL(3)=CZ=(ZJ-ZI)/LE(M)
      CALF=YPGM(M)/SQRT(ZPGM(M)**2+YPGM(M)**2)
      SALF=ZPGM(M)/SQRT(ZPGM(M)**2+YPGM(M)**2)
      IF(CX.EQ.0..AND.CZ.EQ.0.)  GO TO 699
      LAMBDA=SQRT(CX**2+CZ**2)
      RCOL(4)=-(CX*CY*CALF+CZ*SALF)/LAMBDA
      RCOL(5)=CALF*LAMBDA
      RCOL(6)=(-CY*CZ*CALF+CX*SALF)/LAMBDA
      RCOL(7)=(CX*CY*SALF-CZ*CALF)/LAMBDA
      RCOL(8)=-LAMBDA*SALF
      RCOL(9)=(CY*CZ*SALF+CX*CALF)/LAMBDA
      GO TO 499
699   IF(CY.EQ.1.) GO TO 799
      RCOL(1)=RCOL(3)=RCOL(5)=RCOL(8)=0.0
      RCOL(4)=RCOL(9)=CALF
      RCOL(6)=SALF
      RCOL(7)=-RCOL(6)
      RCOL(2)=-1.
      GO TO 499
799   RCOL(1)=RCOL(3)=RCOL(5)=RCOL(8)=0.0
      RCOL(4)=-CALF
      RCOL(9)=-RCOL(4)
      RCOL(6)=RCOL(7)=SALF
      RCOL(2)=1.
499   DO 101 MM=1,4
      NAME1=3*MM-2
      NAME2=3*MM
      DO 102 JJ=NAME1,NAME2
      JK=JJ-(MM-1)*3
      DO 103 I=1,12
      SENEW(I,JJ)=0.0
      DO 104 LM=1,3
      LL=LM+(MM-1)*3
      SENEW(I,JJ)=SENEW(I,JJ)+SE(I,LL)*RCOL((LM-1)*3+JK)
 104  CONTINUE
 103  CONTINUE
 102  CONTINUE
 101  CONTINUE
      DO 201 MM=1,4
      NAME1=3*MM-2
      NAME2=3*MM
      DO 202 II=NAME1,NAME2
      IK=II-(MM-1)*3
      DO 203 J=1,12
      SE(II,J)=0.0
      DO 204 LM=1,3
      LL=LM+(MM-1)*3
      SE(II,J)=SE(II,J)+SENEW(LL,J)*RCOL((LM-1)*3+IK)
 204  CONTINUE
 203  CONTINUE
 202  CONTINUE
 201  CONTINUE
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE INVTRNS
C
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     *ISTRESS
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36)
     *,L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/11/DN(12),W(37,6),V(37,6)
      COMMON/12/JLOC(36,12),U(12),RCOL(9),MSUOPTN,N1GOPTN
      COMMON/16/PRIOPTN
      INTEGER PRIOPTN
      REAL  LE,LAMBDA
C        TO STORE DISPLACEMENTS OF EACH MEMBER IN U(12) ARRAY
C     ******************************************************************
      DO 100 M=1,NE
      DO 210 I=1,12
      NI=NODEI(M)
      NJ=NODEJ(M)
      IF(I.LE.6) GO TO 250
      IF(I.GT.6) GO TO 300
 250  NP=NI
      GO TO 350
 300  NP=NJ
      GO TO 400
 350  U(I)=W(NP,I)
      GO TO 210
 400  U(I)=W(NP,I-6)
 210  CONTINUE
      XI=X(NODEI(M))
      YI=Y(NODEI(M))
      ZI=Z(NODEI(M))
      XJ=X(NODEJ(M))
      YJ=Y(NODEJ(M))
      ZJ=Z(NODEJ(M))
      LE(M)=SQRT((XJ-XI)**2+(YJ-YI)**2+(ZJ-ZI)**2)
      RCOL(1)=CX=(XJ-XI)/LE(M)
      RCOL(2)=CY=(YJ-YI)/LE(M)
      RCOL(3)=CZ=(ZJ-ZI)/LE(M)
      CALF=YPGM(M)/SQRT(ZPGM(M)**2+YPGM(M)**2)
      SALF=ZPGM(M)/SQRT(ZPGM(M)**2+YPGM(M)**2)
      IF(CX.EQ.0..AND.CZ.EQ.0.)  GO TO 699
      LAMBDA=SQRT(CX**2+CZ**2)
      RCOL(4)=-(CX*CY*CALF+CZ*SALF)/LAMBDA
      RCOL(5)=CALF*LAMBDA
      RCOL(6)=(-CY*CZ*CALF+CX*SALF)/LAMBDA
      RCOL(7)=(CX*CY*SALF-CZ*CALF)/LAMBDA
      RCOL(8)=-LAMBDA*SALF
      RCOL(9)=(CY*CZ*SALF+CX*CALF)/LAMBDA
      GO TO 499
```

```
699   IF(CY.EQ.1.) GO TO 799
      RCOL(1)=RCOL(3)=RCOL(5)=RCOL(8)=0.0
      RCOL(4)=RCOL(9)=CALF
      RCOL(6)=SALF
      RCOL(7)=-RCOL(6)
      RCOL(2)=-1.
      GO TO 499
799   RCOL(1)=RCOL(3)=RCOL(5)=RCOL(8)=0.0
      RCOL(4)=-CALF
      RCOL(9)=-RCOL(4)
      RCOL(6)=RCOL(7)=SALF
      RCOL(2)=1.
499   DO 101 MM=1,4
      NAME1=3*MM-2
      NAME2=3*MM
      DO 102 JJ=NAME1,NAME2
      JK=JJ-(MM-1)*3
      ULOC(M,JJ)=0.0
      DO 103 KL=1,3
      LL=KL+(MM-1)*3
      ULOC(M,JJ)=ULOC(M,JJ)+U(LL)*RCOL(3*(JK-1)+KL)
103   CONTINUE
102   CONTINUE
101   CONTINUE
100   CONTINUE
C         TO HAVE PRINT OUT OF NODAL DISPLACEMENTS IN LOCAL COORDINATES
C     *****************************************************************
      IF(PRIOPTN.EQ.0) GO TO 299
      WRITE(61,104)
104   FORMAT(//,10X,*NODAL DISPLACEMENTS ON EACH ELEMENT IN
     +LOCAL COORDINATES*,//)
299   CONTINUE
      DO 105 NN=1,NUMEG
      IF(NUMEL(NN).EQ.0) GO TO 105
      NAME=NUMEL(NN)
      DO 104 K=1,NAME
      M=L(NN,K)
      IF(PRIOPTN.EQ.0) GO TO 399
      WRITE(61,117) M
399   CONTINUE
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 106 K1=1,2
      IF(K1.EQ.1) NP=NI
      IF(K1.EQ.2) NP=NJ
      NAME1=6*(K1-1)+1
      NAME2=6*K1
      DO 106 I=NAME1,NAME2
      IF(K1.EQ.1) GO TO 91
      J=I-6
      V(NP,J)=ULOC(M,I)
      IF(PRIOPTN.EQ.0) GO TO 99
      WRITE(61,109)  M,NP,J,V(NP,J)
99    CONTINUE
      GO TO 106
91    V(NP,I)=ULOC(M,I)
      IF(PRIOPTN.EQ.0) GO TO 199
      WRITE(61,109) M,NP,I,V(NP,I)
199   CONTINUE
106   CONTINUE
105   CONTINUE
117   FORMAT(*-*,7HELEMENT,I3//)
109   FORMAT(* *,5X,2HV(,I2,1H,,I2,1H,,I1,2H)=,E25.15)
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE TRUSS(N,ICHECK,PROTYPE)
C
C     *****************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/4/SE(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36)
     +,L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/8/PI(37,6),PN(37,6),R(107)
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/10/D(60),G1(60),G2(60),G3(60),G4(60),RC(60),
     +SC(60,20),IGAUS
      REAL IXX,IYY,IZZ,KT,LE
      INTEGER PROTYPE
      IF(IPAR.EQ.2)  GO TO 200
      IF(PROTYPE.NE.3)  GO TO 1001
      IF(ICHECK.EQ.1)   GO TO 1001
      DO 5555 M=1,NE
      XI=X(NODEI(M))
      YI=Y(NODEI(M))
      ZI=Z(NODEI(M))
      XJ=X(NODEJ(M))
      YJ=Y(NODEJ(M))
      ZJ=Z(NODEJ(M))
      LE(M)=SQRT((XJ-XI)**2+(YJ-YI)**2+(ZJ-ZI)**2)
5555  CONTINUE
      GO TO 120
C        READ MATERIAL INFORMATION
C     ****************************************************************
1001  WRITE(61,2000)N
      READ(60,1010) E(N),G(N)
      WRITE(61,2020) NUMEL(N),E(N),G(N)
C
C        READ ELEMENT AND CROSS SECTION INFORMATION
C     ****************************************************************
      WRITE(61,2021)
      K=0
105   READ(60,1020) M,NODEI(M),NODEJ(M),A(M),YPGM(M),ZPGM(M),
     +IXX(M),IZZ(M),KT(M)
      WRITE(61,2022) M,NODEI(M),NODEJ(M),A(M),YPGM(M),ZPGM(M),
     +IXX(M),IZZ(M),KT(M)
      K=K+1
      L(N,K)=M
      NI=NODEI(M)
      NJ=NODEJ(M)
      AA=(X(NJ)-X(NI))**2
      B=(Y(NJ)-Y(NI))**2
      C=(Z(NJ)-Z(NI))**2
      LE(M)=SQRT(AA+B+C)
      IF(K.NE.NUMEL(N)) GO TO 105
120   CONTINUE
      RETURN
C
200   CONTINUE
C
C
C        CALCULATE AND STORE LINEAR STIFFNESS MATRIX OF TRUSS ELEMENTS
C     ****************************************************************
      NAME=NUMEL(N)
      DO 220 K=1,NAME
      M=L(N,K)
C
C        TERMS OF STIFFNESS MATRIX OF TRUSS ELEMENTS IN LOCAL COORDINATE
C     ****************************************************************
      DO 400 I=1,12
      DO 500 J=1,I
      SE(I,J)=0.0
500   CONTINUE
400   CONTINUE
      SE(1,1)=SE(7,7)=E(N)*A(M)/LE(M)
      SE(7,1)=-SE(1,1)
:
:        FILL IN UPPER HALF OF MATRIX BY SYMMETRY
:     ****************************************************************
      DO 210 I=1,12
      DO 210 J=1,12
      SE(I,J)=SE(J,I)
 10   CONTINUE
      IF(ISTRESS.EQ.0)  GO TO 601
      WRITE(10,10) ((SE(I,J),J=1,12),I=1,12)
 01   CONTINUE



         TO TRANSFORM SE(12,12) FROM LOCAL TO GLOBAL COORDINATE
         SE(12,12) IS THE  STIFFNESS  MATRIX IN GLOBAL
      ****************************************************************
      CALL TRANSFM(M)
      IF(ISTRESS.EQ.0)  GO TO 602
      WRITE(11,10)  ((SE(I,J),J=1,12),I=1,12)
 12   CONTINUE
C        ASSEMBLE STIFFNESS OF EACH TRUSS ELEMENT INTO
C        STRUCTURAL LINEAR STIFFNESS
      ****************************************************************
      CALL ASEMBLE(M)
 10   CONTINUE
      IF(ISTRESS.EQ.1) REWIND 10
      IF(ISTRESS.EQ.1) REWIND 11
      WRITE(4,10) ((S(I,J),J=1,MBAND),I=1,NSIZE)
      REWIND 4
         TO PRINT ENTRIES OF EACH ELEMENT
C        STIFFNESS MATRIX IN LOCAL
      ****************************************************************
      IF(ICAL3.NE.0) GO TO 251
      WRITE(61,2054)
      NAME=NUMEL(N)
      DO 250 M=1,NAME
      WRITE(61,2053) M
      READ(10,10) ((SE(I,J),J=1,12),I=1,12)
      WRITE(61,2030)
      WRITE(61,2032)  ((SE(I,J),J=1,6),I=1,6)
      WRITE(61,2034)
      WRITE(61,2032)  ((SE(I,J),J=7,12),I=1,6)
      WRITE(61,2036)
      WRITE(61,2032)  ((SE(I,J),J=1,6),I=7,12)
      WRITE(61,2038)
      WRITE(61,2032)  ((SE(I,J),J=7,12),I=7,12)
  0   CONTINUE
  1   CONTINUE
      REWIND 10
         TO PRINT ENTRIES OF EACH ELEMENT
         STIFFNESS MATRIX IN GLOBAL
      ****************************************************************
      IF(ICAL3.NE.0) GO TO 230
      WRITE(61,2052)
      NAME=NUMEL(N)
      DO 242 M=1,NAME
      WRITE(61,2053) M
      READ(11,10)  ((SE(I,J),J=1,12),I=1,12)
      WRITE(61,2030)
      WRITE(61,2032)  ((SE(I,J),J=1,6),I=1,6)
      WRITE(61,2034)
      WRITE(61,2032)  ((SE(I,J),J=7,12),I=1,6)
      WRITE(61,2036)
      WRITE(61,2032)  ((SE(I,J),J=1,6),I=7,12)
      WRITE(61,2038)
      WRITE(61,2032)  ((SE(I,J),J=7,12),I=7,12)
 42   CONTINUE
      REWIND 11
 30   CONTINUE
      RETURN
C
 10   FORMAT(E21.15)
1010  FORMAT(2E10.2)
1020  FORMAT(3I5,3F10.6,3F10.4)
2000  FORMAT(*1*,23HG R O U P   N U M B E R ,I2//2X,6HNUMBER,6X,7HMODULUS
     *,11X,5HSHEAR/4X,2HOF,11X,2HOF,12X,7HMODULUS/1X
     *,8HELEMENTS,4X,10HELASTICITY)
2020  FORMAT(I6,2E17.6)
2021  FORMAT(//8H ELEMENT,3X,8HNODEI(M),3X,8HNODEJ(M),12X,2HLE,11X,
     *4HA(M),9X,6HIXX(M),9X,6HIZZ(M),10X,5HKT(M))
2022  FORMAT(I6,5X,I5,6X,I5,6X,3F15.6,3E15.4)
2030  FORMAT(*1*33HSTIFFNESS MATRIX OF TRUSS ELEMENT///9H BLOCK II)
2032  FORMAT(//11X,6F20.6)
2034  FORMAT(///9H BLOCK IJ)
2036  FORMAT(*1*//9H BLOCK JI)
2038  FORMAT(///9H BLOCK JJ)
2052  FORMAT(///,5X,*ENTRIES OF EACH ELEMENT
     * STIFFNESS MATRIX IN GLOBAL*)
2053  FORMAT(///,10X,15HELEMENT NUMBER ,I2)
2054  FORMAT(///,5X,*ENTRIES OF EACH ELEMENT STIFFNESS
     *MATRIX IN LOCAL *)
C
      END
C
C
C
C
C
```

```
C
C
C
      SUBROUTINE SBEAME1(N)
C
C     ******************************************************************
C        EVALUATION OF ENTRIES OF N1 USING DISPLACEMENTS AND
C        ROTATIONS IN LOCAL COORDINATE SYSTEM
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/4/SE(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36),
     +L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/12/ULOC(36,12),U(12),RCOL(9),MSUOPTN,N1GOPTN
      REAL LE
      NAME=NUMEL(N)
      DO 220 K=1,NAME
      M=L(N,K)
      DO 105  I=1,12
      DO 105  J=I,12
  105 SE(I,J)=0.0
      IF(N1GOPTN.EQ.0)  GO TO 99
      SE(2,2)=SE(3,3)=SE(8,8)=SE(9,9)=6.*(ULOC(M,7)-ULOC(M,1))
     +*E(1)*A(M)/(5.*LE(M)**2)
      SE(2,8)=SE(3,9)=-SE(2,2)
      SE(5,5)=SE(6,6)=SE(11,11)=SE(12,12)=
     +2.*(ULOC(M,7)-ULOC(M,1))*E(1)*A(M)/15.
      SE(2,6)=SE(2,12)=SE(5,9)=SE(9,11)=(ULOC(M,7)-ULOC(M,1))*
     +E(1)*A(M)/(10.*LE(M))
      SE(3,5)=SE(3,11)=SE(8,12)=SE(6,8)=-SE(2,6)
      SE(5,11)=SE(6,12)=-(ULOC(M,7)-ULOC(M,1))*E(1)*A(M)/30.
   99 IF(N1GOPTN.EQ.1)  GO TO 199
      T0=(ULOC(M,8)-ULOC(M,2))/LE(M)
      SI0=(ULOC(M,3)-ULOC(M,9))/LE(M)
      F4=ULOC(M,6)+ULOC(M,12)-12.*T0
      G4=ULOC(M,5)+ULOC(M,11)-12.*SI0
      F51=4.*ULOC(M,6)-ULOC(M,12)-3.*T0
      G51=4.*ULOC(M,5)-ULOC(M,11)-3.*SI0
      F52=4.*ULOC(M,12)-ULOC(M,6)-3.*T0
      G52=4.*ULOC(M,11)-ULOC(M,5)-3.*SI0
      SE(1,2)=SE(7,8)=(-F4/10.)*E(1)*A(M)/LE(M)
      SE(1,8)=SE(2,7)=-SE(1,2)
      SE(2,2)=SE(3,3)=SE(8,8)=SE(9,9)=6.*(ULOC(M,7)-ULOC(M,1))
     +*E(1)*A(M)/(5.*LE(M)**2)
      SE(2,8)=SE(3,9)=-SE(2,2)
      SE(5,5)=SE(6,6)=SE(11,11)=SE(12,12)=
     +2.*(ULOC(M,7)-ULOC(M,1))*E(1)*A(M)/15.
      SE(1,3)=SE(7,9)=(G4/10.)*E(1)*A(M)/LE(M)
      SE(1,9)=SE(3,7)=-SE(1,3)
      SE(1,6)=-F51*E(1)*A(M)/30.
      SE(6,7)=-SE(1,6)
      SE(1,5)=-G51*E(1)*A(M)/30.
      SE(5,7)=-SE(1,5)
      SE(1,12)=-F52*E(1)*A(M)/30.
      SE(7,12)=-SE(1,12)
      SE(1,11)=-G52*E(1)*A(M)/30.
      SE(7,11)=-SE(1,11)
      SE(2,6)=SE(2,12)=SE(5,9)=SE(9,11)=(ULOC(M,7)-ULOC(M,1))*
     +E(1)*A(M)/(10.*LE(M))
      SE(3,5)=SE(3,11)=SE(8,12)=SE(6,8)=-SE(2,6)
      SE(5,11)=SE(6,12)=-(ULOC(M,7)-ULOC(M,1))*E(1)*A(M)/30.
  199 CONTINUE
C
C        FILL IN LOWER HALF OF MATRIX BY SYMMETRY
C     ******************************************************************
      DO 106  I=1,12
      DO 106  J=1,I
  106 SE(I,J)=SE(J,I)
      IF(ISTRESS.EQ.0)  GO TO 601
      WRITE(1,10) ((SE(I,J),J=1,12),I=1,12)
  601 CONTINUE
C
C
C        TO TRANSFER SE(12*12) FROM LOCAL TO GLOBAL COORDINATE
C        SE(12*12) IS THE (N1) STIFFNESS MATRIX IN GLOBAL
C     ******************************************************************
      CALL TRANSFM(M)
      IF(ISTRESS.EQ.0)  GO TO 602
      WRITE(2,10)  ((SE(I,J),J=1,12),I=1,12)
602   CONTINUE
C
C        ASSEMBLE STIFFNESS OF EACH ELEMENT INTO STRUCTURAL
C        STIFFNESS
C     ****************************************************************
      CALL ASEMBLE(M)
220   CONTINUE
      IF(ISTRESS.EQ.1) REWIND 1
      IF(ISTRESS.EQ.1) REWIND 2
      WRITE(6,10) ((S(I,J),J=1,MBAND),I=1,NSIZE)
      REWIND 6
C
C
      RETURN
10    FORMAT(E21.15)
      END
C
C
C
C
C
C
C
      SUBROUTINE KEPSI01(N)
C
C     ****************************************************************
C         TO HAVE THE INITIAL STRAIN STIFFNESS MATRIX
C     ****************************************************************
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     *ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/4/SE(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36),
     *L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/12/ULOC(36,12),U(12),RCOL(9),MSUOPTN,N1GOPTN
      COMMON/17/A7TOT(36),A7OLD(36),BOL(36,5),BTO(36,5),BE(5)
      REAL LE
      INTEGER PROTYPE
      NAME=NUMEL(N)
      DO 220 K=1,NAME
      M=L(N,K)
      IF(MSUOPTN.EQ.1)  A7=A7TOT(M)
      DO 104 I=1,12
      DO 105 J=1,I
      SE(I,J)=0.0
105   CONTINUE
104   CONTINUE
      IF(MSUOPTN.EQ.1)  GO TO 99
      RO1=BTO(M,1)+BTO(M,2)/2.+BTO(M,3)/3.+BTO(M,4)/4.+BTO(M,5)/5.
      RO2=BTO(M,1)/2.+BTO(M,2)/3.+BTO(M,3)/4.+BTO(M,4)/5.+BTO(M,5)/6.
      RO3=BTO(M,1)/3.+BTO(M,2)/4.+BTO(M,3)/5.+BTO(M,4)/6.+BTO(M,5)/7.
      RO4=BTO(M,1)/4.+BTO(M,2)/5.+BTO(M,3)/6.+BTO(M,4)/7.+BTO(M,5)/8.
      RO5=BTO(M,1)/5.+BTO(M,2)/6.+BTO(M,3)/7.+BTO(M,4)/8.+BTO(M,5)/9.
      SE(2,2)=SE(3,3)=SE(8,8)=SE(9,9)=(RO3-2.*RO4+RO5)*36.*A(M)*
     *E(1)/LE(M)
      SE(8,2)=SE(9,3)=-SE(2,2)
      SE(5,3)=SE(8,6)=(RO2-5.*RO3+7.*RO4-3.*RO5)*6.*E(1)*A(M)
      SE(6,2)=SE(9,5)=-SE(5,3)
      SE(6,6)=SE(5,5)=(RO1-8.*RO2+22.*RO3-24.*RO4+9.*RO5)*
     *E(1)*A(M)*LE(M)
      SE(11,11)=SE(12,12)=(4.*RO3-12.*RO4+9.*RO5)*E(1)*A(M)*LE(M)
      SE(11,9)=SE(12,2)=(2.*RO3-5.*RO4+3.*RO5)*6.*E(1)*A(M)
      SE(11,3)=SE(12,8)=-SE(11,9)
      SE(11,5)=SE(12,6)=(-2.*RO2+11.*RO3-18.*RO4+9.*RO5)*E(1)*A(M)*LE(M)
99    IF(MSUOPTN.EQ.2)  GO TO 199
      SE(2,2)=SE(3,3)=SE(8,8)=SE(9,9)=6.*A7*A(M)*E(1)/(5.*LE(M)**2)
      SE(9,3)=SE(8,2)=-SE(2,2)
      SE(5,5)=SE(6,6)=SE(11,11)=SE(12,12)=A7*2.*A(M)*E(1)/15.
      SE(6,2)=SE(9,5)=SE(11,9)=A(M)*A7*E(1)/(10.*LE(M))
      SE(5,3)=SE(8,6)=SE(12,8)=-SE(6,2)
      SE(12,2)=A7*A(M)*E(1)/(10.*LE(M))
      SE(11,3)=-SE(12,2)
      SE(12,6)=SE(11,5)=-A7*A(M)*E(1)/(30.)
199   CONTINUE
C
C        FILL IN UPPER HALF OF MATRIX BY SYMMETRY
C     ****************************************************************
      DO 106 I=1,12
      DO 107 J=I,12
      SE(I,J)=SE(J,I)
107   CONTINUE
106   CONTINUE
      IF(ISTRESS.EQ.0)  GO TO 601
      WRITE(14,10) ((SE(I,J),J=1,12),I=1,12)
601   CONTINUE
C
C
C        TO TRANSFER SE(12*12) FROM LOCAL TO GLOBAL COORDINATE
C        SE(12*12) IS THE (N1) STIFFNESS MATRIX IN GLOBAL
C     *****************************************************************
      CALL TRANSFM(M)
      IF(ISTRESS.EQ.0)  GO TO 602
      WRITE(15,10)  ((SE(I,J),J=1,12),I=1,12)
602   CONTINUE
C
C        ASSEMBLE STIFFNESS OF EACH ELEMENT INTO STRUCTURAL
C        STIFFNESS
C     *****************************************************************
      CALL ASEMBLE(M)
120   CONTINUE
      WRITE(16,10) ((S(I,J),J=1,MBAND),I=1,NSIZE)
      IF(ISTRESS.EQ.1) REWIND 14
      IF(ISTRESS.EQ.1) REWIND 15
      REWIND 16
C
C
      RETURN
10    FORMAT(E21.15)
      END
C
C
C
C
C
C
C
C
      SUBROUTINE SBEAME2(N)
C
C     *****************************************************************
C        EVALUATION OF ENTRIES OF N2 USING DISPLACEMENTS
C        AND ROTATIONS IN LOCAL COORDINATES
C     *****************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/4/SE(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36),
     +L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/12/ULOC(36,12),U(12),RCOL(9),MSUOPTN,N1GOPTN
      REAL LE
      NAME=NUMEL(N)
      DO 220 K=1,NAME
      M=L(N,K)
      DO 105  I=1,12
      DO 105  J=I,12
105   SE(I,J)=0.0
      T0=(ULOC(M,8)-ULOC(M,2))/LE(M)
      SI0=(ULOC(M,3)-ULOC(M,9))/LE(M)
      T1=ULOC(M,6)
      T2=ULOC(M,12)
      SI1=ULOC(M,5)
      SI2=ULOC(M,11)
      IF(MSUOPTN.EQ.1)  GO TO 99
      B1=.4*(T1**2+T2**2)+18.*T0**2/5.-3.*T0*(T1+T2)/5.
     +-T1*T2/5.-63.*SI1**2/15.+2.*SI2**2/15.+6.*SI0**2/5.
     ++39.*SI1*SI0/5.-SI2*SI0/5.-17.*SI1*SI2/5.
      B2=.2*T1**2+.6*T2**2+3.6*T0**2-1.2*T0*T1-.2*T1*T2
     +-20.8*SI1**2+.2*SI2**2+1.2*SI0**2+11.6*SI1*SI0-15.2*SI1*SI2
      B3=(6.*T1**2+27.*T2**2+108.*T0**2-45.*T1*T0+15.*T2*T0
     +-9.*T1*T2-292.*SI1**2+9.*SI2**2+36.*SI0**2+489.*SI1*SI0
     ++6.*SI0*SI2-213.*SI1*SI2)/35.
      B4=33.*T1**2/140.+39.*T2**2/28.-1107.*T0**2/7.-1.8*T0*T1
     ++1.8*T0*T2-33.*T1*T2/70.-1949.*SI1**2/140.+13.*SI2**2/35.
     ++9.*SI0**2/7.+117.*SI0*SI1/5.+339.*SI0*SI2/7.-711.*SI1*SI2/70.
      B5=9.*T1**2/35.+27.*T2**2/14.+27.*T0**2/7.-27.*T0*T1/14.
     ++27.*T0*T2/14.-9.*T1*T2/14.-627.*SI1**2/35.+9.*SI2**2/14.
     ++9.*SI0**2/7.+423.*SI0*SI1/14.+9.*SI0*SI2/14.-183.*SI1*SI2/14.
      B6=.4*(SI1**2+SI2**2)+18.*SI0**2/5.-3.*SI0*(SI1+SI2)/5.
     +-SI1*SI2/5.-68.*T1**2/15.+2.*T2**2/15.+6.*T0**2/5.
     ++39.*T1*T0/5.-T2*T0/5.-17.*T1*T2/5.
      B7=.2*SI1**2+.6*SI2**2+3.6*SI0**2-1.2*SI0*SI1-.2*SI1*SI2
     +-20.8*T1**2+.2*T2**2+1.2*T0**2+11.6*T1*T0-15.2*T1*T2
      B8=(6.*SI1**2+27.*SI2**2+108.*SI0**2-45.*SI1*SI0+18.*SI2*SI0
     +-9.*SI1*SI2-292.*T1**2+9.*T2**2+36.*T0**2+489.*T1*T0
     ++6.*T0*T2-213.*T1*T2)/35.
      B9=33.*SI1**2/140.+39.*SI2**2/28.-1107.*SI0**2/7.-1.8*SI0*SI1
     ++1.8*SI0*SI2-33.*SI1*SI2/70.-1949.*T1**2/140.+13.*T2**2/28.
     ++9.*T0**2/7.+117.*T0*T1/5.+339.*T0*T2/7.-711.*T1*T2/70.
      B10=9.*SI1**2/35.+27.*SI2**2/14.+27.*SI0**2/7.-27.*SI0*SI1/14.
     ++27.*SI0*SI2/14.-9.*SI1*SI2/14.-627.*T1**2/35.+9.*T2**2/14.
     ++9.*T0**2/7.+423.*T0*T1/14.+9.*T0*T2/14.-183.*T1*T2/14.
      B11=-4.*T1*SI1/15.+T1*SI0/5.+T1*SI2/15.+SI1*T0/5.+T2*SI1/5.
     +-12.*T0*SI0/5.+T0*SI2/5.+T2*SI0/5.-4.*T2*SI2/15.
      B12=-2.*T1*SI1/15.+2.*T1*SI0/5.+T1*SI2/15.+2.*SI1*T0/5.
     ++T2*SI1/15.-12.*T0*SI0/5.-2.*T2*SI2/5.
      B13=-4.*T1*SI1/35.+3.*T1*SI0/7.+3.*T1*SI2/35.+3.*SI1*T0/7.
     ++3.*T2*SI1/35.-72.*T0*SI0/35.-6.*T0*SI2/35.-6.*T2*SI0/35.
     +-18.*T2*SI2/35.
      B14=-11.*T1*SI1/70.+3.*T1*SI0/5.+11.*T1*SI2/70.+
     +3.*SI1*T0/5.+11.*T2*SI1/70.
     +-18.*T0*SI0/7.-3.*T0*SI2/7.-3.*T2*SI0/7.-13.*T2*SI2/14.
      B15=-6.*T1*SI1/35.+9.*T1*SI0/14.+3.*T1*SI2/14.+9.*SI1*T0/14.
     ++3.*T2*SI1/14.-18.*T0*SI0/7.-9.*T0*SI2/14.
     +-9.*T2*SI0/14.-9.*T2*SI2/7.
      SE(2,2)=SE(8,8)=A(M)*E(1)*(6.*B3-6.*B4+2.*B5)/LE(M)
      SE(2,8)=-SE(2,2)
      SE(3,3)=SE(9,9)=A(M)*E(1)*(6.*B8-6.*B9+2.*B10)/LE(M)
      SE(3,9)=-SE(3,3)
      SE(2,3)=A(M)*E(1)*(6.*B13-6.*B14+2.*B15)/LE(M)
      SE(2,9)=-SE(2,3)
      SE(2,6)=(-3.*B2+10.*B3-7.*B4+2.*B5)*A(M)*E(1)/2.
      SE(6,8)=-SE(2,6)
      SE(3,6)=SE(5,8)=(-3.*B12+10.*B13-7.*B14+2.*B15)*A(M)*E(1)/2.
      SE(6,9)=SE(2,5)=-SE(3,6)
      SE(6,6)=(B1-4.*B2+22.*B3/3.-4.*B4+B5)*A(M)*E(1)*LE(M)/2.
      SE(5,5)=(B6-4.*B7+22.*B8/3.-4.*B9+B10)*A(M)*E(1)*LE(M)/2.
      SE(12,12)=(4.*B3/3.-2.*B4+B5)*A(M)*E(1)*LE(M)/2.
      SE(11,11)=(4.*B8/3.-2.*B9+B10)*A(M)*E(1)*LE(M)/2.
      SE(3,5)=(3.*B7-10.*B8+7.*B9-2.*B10)*A(M)*E(1)/2.
      SE(5,9)=-SE(3,5)
      SE(5,6)=-(B11-4.*B12+22.*B13/3.-4.*B14+B15)*A(M)*E(1)*LE(M)/2.
      SE(3,8)=-(6.*B13-6.*B14+2.*B15)*A(M)*E(1)/LE(M)
      SE(8,9)=-SE(3,8)
      SE(2,12)=(4.*B3-5.*B4+2.*B5)*A(M)*E(1)/2.
      SE(8,12)=-SE(2,12)
      SE(3,12)=SE(8,11)=(4.*B13-5.*B14+2.*B15)*A(M)*E(1)/2.
      SE(9,12)=SE(2,11)=-SE(3,12)
      SE(3,11)=(-4.*B8+5.*B9-2.*B10)*A(M)*E(1)/2.
      SE(9,11)=-SE(3,11)
      SE(6,12)=(-B2+11.*B3/3.-3.*B4+B5)*A(M)*E(1)*LE(M)/2.
      SE(6,11)=SE(5,12)=(B12-11.*B13/3.+3.*B14-B15)*A(M)*E(1)*LE(M)/2.
      SE(5,11)=(-B7+11.*B8/3.-3.*B9+B10)*A(M)*E(1)*LE(M)/2.
      SE(11,12)=(-4.*B13/3.+2.*B14-B15)*A(M)*E(1)*LE(M)/2.
99    IF(MSUOPTN.EQ.2)  GO TO 199
      F4=ULOC(M,6)+ULOC(M,12)-12.*T0
      G4=ULOC(M,5)+ULOC(M,11)-12.*SI0
      F51=4.*ULOC(M,6)-ULOC(M,12)-3.*T0
      G51=4.*ULOC(M,5)-ULOC(M,11)-3.*SI0
      F52=4.*ULOC(M,12)-ULOC(M,6)-3.*T0
      G52=4.*ULOC(M,11)-ULOC(M,5)-3.*SI0
      F1=9.*(T1**2+T2**2)-2.*T1*T2-36.*T0*(T1+T2)+216.*T0**2
      G1=9.*(SI1**2+SI2**2)-2.*SI1*SI2-36.*SI0*(SI1+SI2)+216.*SI0**2
      F2=2.*(T1**2+T2**2)-T1*T2-3.*T0*(T1+T2)+18.*T0**2
      G2=2.*(SI1**2+SI2**2)-SI1*SI2-3.*SI0*(SI1+SI2)+18.*SI0**2
      F7=-2.*(T1**2+T2**2)+6.*T1*T2-2.*T0*(T1+T2)-3.*T0**2
      G7=-2.*(SI1**2+SI2**2)+6.*SI1*SI2-2.*SI0*(SI1+SI2)-3.*SI0**2
      F31=6.*T1**2+T2**2+2.*T1*T2-54.*T1*T0+6.*T2*T0+54.*T0**2
      F32=6.*T2**2+T1**2+2.*T1*T2-54.*T2*T0+6.*T1*T0+54.*T0**2
      G31=6.*SI1**2+SI2**2+2.*SI1*SI2-54.*SI1*SI0+6.*SI2*SI0+54.*SI0**2
      G32=6.*SI2**2+SI1**2+2.*SI1*SI2-54.*SI2*SI0+6.*SI1*SI0+54.*SI0**2
      F61=8.*T1**2+3.*T2**2-4.*T1*T2-12.*T1*T0-2.*T2*T0+27.*T0**2
      F62=8.*T2**2+3.*T1**2-4.*T1*T2-12.*T2*T0-2.*T1*T0+27.*T0**2
      G61=8.*SI1**2+3.*SI2**2-4.*SI1*SI2-12.*SI1*SI0-2.*SI2*SI0+27.
     +*SI0**2
      G62=8.*SI2**2+3.*SI1**2-4.*SI1*SI2-12.*SI2*SI0-2.*SI1*SI0+27.
     +*SI0**2
      SE(2,2)=SE(8,8)=E(1)*A(M)*(F1/100.+G2/25.)/LE(M)
      SE(2,8)=-SE(2,2)
      SE(2,3)=SE(8,9)=-E(1)*A(M)*F4*G4/(100.*LE(M))
      SE(2,9)=SE(3,8)=-SE(2,3)
      SE(3,3)=SE(9,9)=E(1)*A(M)*(G1/100.+F2/25.)/LE(M)
      SE(3,9)=-SE(3,3)
      SE(2,6)=E(1)*A(M)*(F31+G2)/300.
      SE(6,8)=-SE(2,6)
      SE(2,5)=E(1)*A(M)*F4*G51/300.
      SE(3,6)=-E(1)*A(M)*G4*F51/300.
      SE(6,9)=-SE(3,6)
      SE(3,5)=-E(1)*A(M)*(G31+F2)/300.
      SE(5,9)=-SE(3,5)
      SE(6,6)=E(1)*A(M)*LE(M)*(F61/300.+G2/225.)
      SE(5,6)=E(1)*A(M)*LE(M)*F51*G51/900.
      SE(5,5)=E(1)*A(M)*LE(M)*(G61/300.+F2/225.)
      SE(5,8)=-E(1)*A(M)*F4*G51/300.
      SE(2,12)=E(1)*A(M)*(F32+G2)/300.
      SE(8,12)=-SE(2,12)
      SE(2,11)=E(1)*A(M)*F4*G52/300.
      SE(8,11)=-SE(2,11)
      SE(3,12)=-E(1)*A(M)*G4*F52/300.
      SE(9,12)=-SE(3,12)
      SE(3,11)=-E(1)*A(M)*(G32+F2)/300.
      SE(9,11)=-SE(3,11)
      SE(6,12)=E(1)*A(M)*LE(M)*(F7-G2/3.)/300.
      SE(6,11)=E(1)*A(M)*LE(M)*F51*G52/900.
      SE(5,12)=E(1)*A(M)*LE(M)*F52*G51/900.
      SE(5,11)=E(1)*A(M)*LE(M)*(G7-F2/3.)/300.
      SE(12,12)=E(1)*A(M)*LE(M)*(F62/300.+G2/225.)
      SE(11,12)=E(1)*A(M)*LE(M)*F52*G52/900.
      SE(11,11)=E(1)*A(M)*LE(M)*(G62/300.+F2/225.)
199   CONTINUE
C
C        FILL IN LOWER HALF OF MATRIX BY SYMMETRY
C     *****************************************************************
      DO 111  I=1,12
      DO 111  J=1,I
111   SE(I,J)=SE(J,I)
      IF(ISTRESS.EQ.0)  GO TO 601
      WRITE(3,10) ((SE(I,J),J=1,12),I=1,12)
601   CONTINUE
C
C
C        TO TRANSFER SE(12*12) FROM LOCAL TO GLOBAL COORDINATE
C        SE(12*12) IS THE (N2) STIFFNESS MATRIX IN GLOBAL
C     *****************************************************************
      CALL TRANSFM(M)
      IF(ISTRESS.EQ.0)  GO TO 602
      WRITE(5,10) ((SE(I,J),J=1,12),I=1,12)
602   CONTINUE
C
C        ASSEMBLE STIFFNESS OF EACH ELEMENT INTO STRUCTURAL
C        STIFFNESS
C     *****************************************************************
      CALL ASEMBLE(M)
220   CONTINUE
      IF(ISTRESS.EQ.1) REWIND 3
      IF(ISTRESS.EQ.1) REWIND 5
      WRITE(12,10)  ((S(I,J),J=1,MBAND),I=1,NSIZE)
      REWIND 12
C
C
C
      RETURN
10    FORMAT(E21.15)
      END
C
C
C
C
C
C
C
      SUBROUTINE ASEMBLE(M)
C
C     *****************************************************************
C        TO PROCESS AND ASSEMBLE ELEMENT STIFFNESS MATRICES AND NODAL
C        LOAD VECTORS INTO THEIR CORRESPONDING STRUCTURE ARRAYS.
C     *****************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/4/SE(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36),
     +L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/8/PI(37,6),PN(37,6),R(107)
      COMMON/9/S(60,20),SP(60,20),IDET
C
C        SET STRUCTRUE STIFFNESS ARRAY AND LOAD VECTOR
C        ARRAY EQUAL TO ZERO
C     ******************************************************************
      IF(IPAR.NE.1) GO TO 90
      DO 5 I=1,NSIZE
      R(I)=0.0
      DO 5 J=1,MBAND
      S(I,J)=0.0
5     CONTINUE
C
C        PROCESSING OF INITIAL LOADS AND NODAL LOADS
C     ******************************************************************
      IF(ICAL1.EQ.0) WRITE(61,2000)
      DO 80 N=1,NUMNP
      DO 70 I=1,6
      IF(IA(N,I)) 20,70,10
10    II=IA(N,I)
      GO TO 60
20    IF(IB(N,I).LT.0) GO  TO 30
      NN=IB(N,I)
      GO TO 35
30    II=-IB(N,I)+NEQ
      GO TO 60
35    IF(IA(NN,I)) 40,70,50
40    II=-IB(NN,I)+NEQ
      GO TO 60
50    II=IA(NN,I)
60    R(II)=PI(N,I)
      IF(ICAL1.EQ.0) WRITE(61,2010) II,N,I,R(II)
70    CONTINUE
80    CONTINUE
      RETURN
C
C        ASSEMBLE ELEMENT STIFFNESS AND NODAL LOAD VECTORS
C     ******************************************************************
90    NI=NODEI(M)
      NJ=NODEJ(M)
      DO 165 K1=1,2
      IF(K1.EQ.1) NP=NI
      IF(K1.EQ.2) NP=NJ
      DO 160 I=1,6
      IF(IA(NP,I)) 105,160,100
100   II=IA(NP,I)
      GO TO 115
105   IF(IB(NP,I).LT.0) GO TO 110
      NN=IB(NP,I)
      GO TO 111
110   II=-IB(NP,I)+NEQ
      GO TO 115
111   IF(IA(NN,I)) 112,160,113
112   II=-IB(NN,I)+NEQ
      GO TO 115
113   II=IA(NN,I)
115   CONTINUE
      DO 155 K2=1,2
      IF(K2.EQ.1) ND=NI
      IF(K2.EQ.2) ND=NJ
      DO 150 J=1,6
      IF(IA(ND,J)) 125,150,120
120   JJ=IA(ND,J)
      GO TO 145
125   IF(IB(ND,J).LT.0) GO TO 130
      NN=IB(ND,J)
      GO TO 132
130   JJ=-IB(ND,J)+NEQ
      GO TO 145
132   IF(IA(NN,J)) 135,150,140
135   JJ=-IB(NN,J)+NEQ
      GO TO 145
140   JJ=IA(NN,I)
145   CONTINUE
C
```

```
C         FILL IN STRUCTURE STIFFNESS MATRIX IN BANDED FORMAT
C         ONLY LOWER TRIANGLE INCLUDING MAIN DIAGONAL
C     ******************************************************************
      IF(JJ.LT.II) GO TO 150
      IF(K1.EQ.1) IE=I
      IF(K1.EQ.2) IE=I+6
      IF(K2.EQ.1) JE=J
      IF(K2.EQ.2) JE=J+6
C
C         CHANGE -JJ-SUBSCRIPT OF FULL MATRIX TO -JJ- SUBSCRIPT
C         OF BANDED FORMAT. LOOP OVER TERMS OUTSIDE OF BAND
C     ******************************************************************
      JJ=JJ-II+1
      S(II,JJ)=S(II,JJ)+SE(IE,JE)
      IF(JJ.GT.MBAND) MBAND=JJ
150   CONTINUE
155   CONTINUE
160   CONTINUE
165   CONTINUE
C
      RETURN
C
2000  FORMAT(*1*,43HINITIAL AND NODAL LOADS PROCESSED INTO LOAD,
     +      12H VECTOR P(I)//)
2010  FORMAT(*0*,2HR(,I3,4H)=P(,I2,1H,,I2,2H)=,F16.6)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE STCONDN
C
C     ******************************************************************
C         TO CONDENSE STRUCTURE STIFFNESS ACCORDING TO D. O.F.*S IN
C         ARRAYS IA AND IB. ALSO TO CONDENSE LOAD VECTOR OF STRUCTURE.
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/8/PI(37,6),PN(37,6),R(107)
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/10/D(60),G1(60),G2(60),G3(60),G4(60),RC(60),
     +SC(60,20),IGAUS
      IF(ICAL1.EQ.0) WRITE(61,2050)
      IF(ICAL1.EQ.0) WRITE(61,2060) (I,R(I),I=1,NSIZE)
C
C         WRITE UNCONDENSED STRUCTURE LINEAR STIFFNESS
C     ******************************************************************
      IF(ICAL1.NE.0) GO TO 90
      IF(IPAR.EQ.2) WRITE(61,2030)
      K1=1
      K2=8
      K3=MBAND-K1
      IF(K3.LE.7) GO TO 60
50    WRITE(61,2015) K1,K2
      WRITE(61,2020) ((S(I,J),J=K1,K2),I=1,NSIZE)
      K1=K1+8
      K2=K2+8
      K3=MBAND-K1
      IF(K3.LE.7) GO TO 60
      GO TO 50
60    WRITE(61,2015) K1,MBAND
      IF(K3.EQ.0) WRITE(61,2027) ((S(I,J),J=K1,MBAND),I=1,NSIZE)
      IF(K3.EQ.1) WRITE(61,2021) ((S(I,J),J=K1,MBAND),I=1,NSIZE)
      IF(K3.EQ.2) WRITE(61,2022) ((S(I,J),J=K1,MBAND),I=1,NSIZE)
      IF(K3.EQ.3) WRITE(61,2023) ((S(I,J),J=K1,MBAND),I=1,NSIZE)
      IF(K3.EQ.4) WRITE(61,2024) ((S(I,J),J=K1,MBAND),I=1,NSIZE)
      IF(K3.EQ.5) WRITE(61,2025) ((S(I,J),J=K1,MBAND),I=1,NSIZE)
      IF(K3.EQ.6) WRITE(61,2026) ((S(I,J),J=K1,MBAND),I=1,NSIZE)
      IF(K3.EQ.7) WRITE(61,2020) ((S(I,J),J=K1,MBAND),I=1,NSIZE)
90    CONTINUE
C
      IF(NCOND.EQ.0) GO TO 115
      DO 112 K=1,NCOND
      LL=NSIZE-K
      KK=LL+1
      DO 110 L=1,LL
      J=L-KK+MBAND
      IF(J.LE.0) GO TO 110
      IF(S(KK,J).EQ.0) GO TO 110
      DUM=S(KK,J)/S(KK,MBAND)
      DO 100 MM=1,L
      JJ=MM-L+MBAND
      IF(JJ.LE.0) GO TO 100
      II=MM-KK+MBAND
      IF(II.LE.0) GO TO 100
      S(L,JJ)=S(L,JJ)-S(KK,II)*DUM
100   CONTINUE
110   CONTINUE
112   CONTINUE
115   CONTINUE
C
C        STORE CONDENSATION DATA
C     ******************************************************************
      IF(NCOND.EQ.0) GO TO 150
      IF(IPAR.NE.2) GO TO 150
      DO 140 I=1,NCOND
      K=NSIZE-NCOND+I
      DO 130 J=1,MBAND
130   SC(I,J)=S(K,J)
140   RC(I)=R(K)
150   CONTINUE
C
C        CHECK DATA GENERATION
C     ******************************************************************
      IF(ICAL1.EQ.0) WRITE(61,2070)
      IF(ICAL1.EQ.0) WRITE(61,2080) (I,R(I),I=1,NEQ)
      IF(ICAL1.NE.0)GO TO 185
      IF(IPAR.EQ.2) WRITE(61,2000)
      K1=1
      K2=8
      K3=MBAND-K1
      IF(K3.LE.7) GO TO 180
160   WRITE(61,2015) K1,K2
      WRITE(61,2020) ((S(I,J),J=K1,K2),I=1,NEQ)
      K1=K1+8
      K2=K2+8
      K3=MBAND-K1
      IF(K3.LE.7) GO TO 180
      GO TO 160
180   WRITE(61,2015)K1,MBAND
      IF(K3.EQ.0) WRITE(61,2027) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.1) WRITE(61,2021) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.2) WRITE(61,2022) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.3) WRITE(61,2023) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.4) WRITE(61,2024) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.5) WRITE(61,2025) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.6) WRITE(61,2026) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.7) WRITE(61,2020) ((S(I,J),J=K1,MBAND),I=1,NEQ)
185   CONTINUE
C
C        STORE CONDENSED LINEAR STIFFNESS OF STRUCTURE
C     ******************************************************************
      WRITE(4,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      REWIND 4
C
      RETURN
C
10    FORMAT(E21.15)
2000  FORMAT(*1*,43HCONDENSED LINEAR STIFFNESS OF STRUCTURE (S))
2015  FORMAT(*-*,7HCOLUMNS,I4,7HTHROUGH,I4)
2020  FORMAT(*0*,8E16.5)
2021  FORMAT(*0*,2E16.5)
2022  FORMAT(*0*,3E16.5)
2023  FORMAT(*0*,4E16.5)
2024  FORMAT(*0*,5E16.5)
2025  FORMAT(*0*,6E21.6)
2026  FORMAT(*0*,7E16.5)
2027  FORMAT(*0*,E16.5)
2030  FORMAT(*1*,45HUNCONDENSED LINEAR STIFFNESS OF STRUCTURE (S))
2050  FORMAT(*1*,28HUNCONDENSED LOAD VECTOR R(I)//)
2060  FORMAT(* *,2HR(,I2,2H)=,F16.6)
2070  FORMAT(*1*,26HCONDENSED LOAD VECTOR R(I)//)
2080  FORMAT(* *,2HR(,I2,2H)=,F16.6)
C
      END
C
```

```
C
C
C
C
C
C
      SUBROUTINE LINSOLN
C
C     *****************************************************************
C        TO SOLVE SYSTEM OF LINEAR EQUATIONS S*D=R BY CALLING THE
C        APPROPRIATE SUBROUTINE
C           S= STUCTURE"S LINEAR STIFFNESS
C           D= VECTOR OF D.O.F."S
C           R= LOAD VECTOR
C        GAUSS ELIMINATION EQUATION SOLVER, BANDED FORMAT
C        FROM BOOK BY ROBERT D. COOK, FIG. 2.8.1., PAGE 45
C        CONCEPTS AND APPLICATIONS OF FINITE ELEMENT ANALYSIS
C     *****************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/8/PI(37,6),PN(37,6),R(107)
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/10/D(60),G1(60),G2(60),G3(60),G4(60),RC(60),
     +SC(60,20),IGAUS
      COMMON/16/PRIOPTN
      INTEGER PRIOPTN
C
      IF(IGAUS.EQ.1) GO TO 99
C        FILL-IN ARRAY D(I) WITH VALUES OF LOAD VECTOR R(I)
C        AFTER SOLUTION D(I) WILL CONTAIN THE DISPLACEMENT VALUES
C     *****************************************************************
C
      DO 110 I=1,NEQ
110   D(I)=R(I)
C
C        CHECK DATA GENERATION FOR SOLUTION OF EQUATIONS
C     *****************************************************************
C
      IF(PRIOPTN.EQ.1) ICAL2=0
      IF (ICAL2.EQ.0) WRITE(61,2020)
      IF (ICAL2.EQ.0) WRITE(61,2010) (I,D(I),I=1,NEQ)
C
C
C        SOLVE SYSTEM OF -NEQ- LINEAR EQUATIONS
C
C
C        FORWARD REDUCTION OF MATRIX (GAUSS ELIMINATION)
C     *****************************************************************
C
99    DO 790 N=1,NEQ
      DO 780 L=2,MBAND
      IF (S(N,L).EQ.0.) GO TO 780
      I=N+L-1
      C=S(N,L)/S(N,1)
      J=0
      DO 750 K=L,MBAND
      J=J+1
750   S(I,J)=S(I,J)-C*S(N,K)
      S(N,L)=C
780   CONTINUE
790   CONTINUE
C
C        FORWARD REDUCTION OF CONSTANTS (GAUSS ELIMINATION)
C     *****************************************************************
C
      DO 830 N=1,NEQ
      DO 820 L=2,MBAND
      IF (S(N,L).EQ.0.) GO TO 820
      I=N+L-1
      D(I)=D(I)-S(N,L)*D(N)
820   CONTINUE
830   D(N)=D(N)/S(N,1)
C
C        SOLVE FOR UNKNOWNS BY BACK SUBSTITUTION
C     *****************************************************************
C
      DO 860 M=2,NEQ
      N=NEQ+1-M
      DO 850 L=2,MBAND
      IF (S(N,L).EQ.0.)GO TO 850
      K=N+L-1
      D(N)=D(N)-S(N,L)*D(K)
850   CONTINUE
860   CONTINUE
      IF(IGAUS.EQ.1) GO TO 140
C
C        CHECK DATA GENERATION
C     ****************************************************************
C
      IF(PRIOPTN.EQ.1) ICAL2=0
      IF (ICAL2.NE.0) GO TO 140
      WRITE(61,2000)
      WRITE(61,2010) (I,D(I),I=1,NEQ)
      ICAL2=1
140   RETURN
C
2000  FORMAT(* *,34HDISPLACEMENTS FROM LINEAR SOLUTION//)
2010  FORMAT(* *,2HD(,I3,2H)=,E25.15)
2020  FORMAT(* *,31HLOAD VECTOR FOR LINEAR SOLUTION//)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE RECOVER
C
C     ****************************************************************
C        TO RECOVER THE INTERNAL D.O.F.*S OF THE STRUCTURE AFTER
C        SOLVING THE SYSTEM OF EQUATINS
C     ****************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/10/D(60),G1(60),G2(60),G3(60),G4(60),RC(60),
     +SC(60,20),IGAUS
      COMMON/16/PRIOPTN
      INTEGER PRIOPTN
C
      IF(PRIOPTN.EQ.1) ICAL2=0
      IF(NCOND.EQ.0) GO TO 120
      DO 110 J=1,NCOND
      JJ=NSIZE-NCOND+J
      DUM=0.0
      K=JJ-1
      DO 100 I=1,K
      II=I-JJ+MBAND
      IF(II.LE.0) GO TO 100
      DUM=DUM+SC(J,II)*D(I)
100   CONTINUE
110   D(JJ)=(RC(J)-DUM)/SC(J,MBAND)
C
      IF(ICAL2.NE.0) GO TO 120
      WRITE(61,2000)
      N=NEQ+1
      WRITE(61,2010) (I,D(I),I=N,NSIZE)
      ICAL2=1
120   RETURN
C
2000  FORMAT(*1*,15HINTERNAL D.O.F.)
2010  FORMAT(* *,2HD(,I3,2H)=,E25.15)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE IDENT
C
C     ****************************************************************
C        TO IDENTFY THE DISPLACEMENTS FOUND IN THE SOLUTION OF
C        EQUATIONS S*D=R AND THE ONES FOUND IN THE RECOVERY PROCESS
C     ****************************************************************
```

```
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36),
     +L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/10/D(60),G1(60),G2(60),G3(60),G4(60),RC(60),
     +SC(60,20),IGAUS
      COMMON/11/DN(12),W(37,6),V(37,6)
      COMMON/16/PRIOPTN
      INTEGER PRIOPTN
C
C        IDENTIFICATION OF DISPLACEMENTS
C     ******************************************************************
C
      IF(PRIOPTN.EQ.1) ICAL2=0
      IF(ICAL2.EQ.0) WRITE(61,2000)
      DO 240 NN=1,NUMEG
      IF(NUMEL(NN).EQ.0) GO TO 240
      NAME=NUMEL(NN)
      DO 230 K=1,NAME
      M=L(NN,K)
      IF(ICAL2.EQ.0) WRITE(61,2010) M
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 230 K1=1,2
      IF(K1.EQ.1) NP=NI
      IF(K1.EQ.2) NP=NJ
      DO 220 I=1,6
      IF(IA(NP,I)) 160,155,150
150   NL=IA(NP,I)
      W(NP,I)=D(NL)
      IF(ICAL2.EQ.0) WRITE(61,2020)     M,NP,I,W(NP,I)
      GO TO 220
155   W(NP,I)=0.0
      IF(ICAL2.EQ.0) WRITE(61,2020)     M,NP,I,W(NP,I)
      GO TO 220
160   IF (IB(NP,I).LT.0) GO TO 170
      NM=IB(NP,I)
      GO TO 180
170   NL=-IB(NP,I)+NEQ
      W(NP,I)=D(NL)
      IF(ICAL2.EQ.0) WRITE(61,2020)     M,NP,I,W(NP,I)
      GO TO 220
180   IF (IA(NM,I)) 190,200,210
190   NL=-IB(NM,I)+NEQ
      W(NP,I)=D(NL)
      IF(ICAL2.EQ.0) WRITE(61,2020)     M,NP,I,W(NP,I)
      GO TO 220
200   W(NP,I)=0.0
      IF(ICAL2.EQ.0) WRITE(61,2020)     M,NP,I,W(NP,I)
      GO TO 220
210   NL=IA(NM,I)
      W(NP,I)=D(NL)
      IF(ICAL2.EQ.0) WRITE(61,2020)     M,NP,I,W(NP,I)
220   CONTINUE
230   CONTINUE
240   CONTINUE
      ICAL2=1
      RETURN
C
2000  FORMAT(*1*,35HNODAL DISPLACEMENTS ON EACH ELEMENT)
2010  FORMAT(*-*,7HELEMENT,I3//)
2020  FORMAT(* *,5X,2HU(,I2,1H,,I2,1H,,I1,2H)=,E25.15)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE EIGENVL(EIGEN,IDATA)
C
C     ******************************************************************
C        TO SOLVE EIGENVALUE PROBLEM S*X=-(LAMBDA)*S1*X
C        WILL OBTAIN ONLY THE LOWEST EIGENVALUE AND CORRESPONDING
C        EIGENVECTOR. USES INVERSE VECTOR ITERATION WITH THE
C        RAYLEIGH QUOTIENT
```

```
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/9/S(60,20),SP(60,20),IDET
      COMMON/10/XB(60),YB(60),X(60),Y(60),EIGNVTR(60)
     +,RC(60),SC(60,20),IGAUS
C
C        ASSUME STARTING SHIFT ,STARTING VECTOR,AND
C        MAXIMUM NUMBER OF ITERATIONS ALLOWED.
C     ******************************************************************
      WRITE(61,2010)
      READ(60,1000)  MAX,EPSI,RHO
      WRITE(61,2000) MAX,EPSI,RHO
      DO 100 I=1,NEQ
  100 X(I)=1.
C
C        OBTAIN VECTOR Y(I) FROM Y(I)=S1(I,J)*X(I)
C        FIRST CHANGE SIGN OF MATRIX S1
C     ******************************************************************
C
      READ(6,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      REWIND 6
      DO 107 I=1,NEQ
      DO 105 J=1,MBAND
      S(I,J)=-S(I,J)
  105 CONTINUE
  107 CONTINUE
      WRITE(13,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      REWIND 13
C
C        HORIZONTAL SWEEP OF S1(I,J)*X(I),DIAGONAL NOT INCLUDED
C     ******************************************************************
C
      DO 130 I=1,NEQ
      Y(I)=0.0
      II=I+1
      IF(II.GT.NEQ) GO TO 130
      DO 120 J=2,MBAND
      IF(S(I,J).EQ.0.) GO TO  110
      Y(I)=Y(I)+S(I,J)*X(II)
  110 II=II+1
      IF(II.GT.NEQ) GO TO 130
  120 CONTINUE
  130 CONTINUE
C
C        DIAGONAL SWEEP OF S1(I,J)*X(I)
C     ******************************************************************
C
      DO 160 I=1,NEQ
      II=I
      JJ=1
  140 IF(S(II,JJ).EQ.0.) GO TO 150
      Y(I)=Y(I)+S(II,JJ)*X(II)
  150 II=II-1
      JJ=JJ+1
      IF(II.EQ.0) GO TO 160
      IF(JJ.GT.MBAND) GO TO 160
      GO TO 140
  160 CONTINUE
C
C        EQUATIONS S(I,J)*XB(I)=Y(I) AND SOLVING FOR XB(I).
C        STORE VALUES OF Y(I) INTO XB(I) FOR GAUSS SOLUTION
C     ******************************************************************
C
      DO 300 K=1,MAX
      DO 165 I=1,NEQ
  165 XB(I)=Y(I)
C        IDATA=0 AND TAPE 4 FOR K ONLY
C        IDATA=1 AND TAPE 9 FOR (K+N1)
C     ******************************************************************
C
      IF(IDATA.EQ.0) READ(4,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      IF(IDATA.EQ.1) READ(9,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      REWIND 4
      REWIND 9
      IF(ICAL3.NE.0) GO TO 176
C
C        PRINT DATA SENT TO SUBROUTINE GAUSSOL
C     ******************************************************************
```

```
C
      IF(K.NE.1) GO TO 178
      WRITE(61,2100) K
      K1=1
      K2=8
      K3=MBAND-K1
      IF(K3.LE.7) GO TO 174
  172 WRITE (61,2110) K1,K2
      WRITE (61,2115) ((S(I,J),J=K1,K2),I=1,NEQ)
      K1=K1+8
      K2=K2+8
      K3=MBAND-K1
      IF(K3.LE.7) GO TO 174
      GO TO 172
  174 WRITE(61,2110) K1,MBAND
      IF(K3.EQ.0) WRITE(61,2120) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.1) WRITE(61,2121) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.2) WRITE(61,2122) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.3) WRITE(61,2123) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.4) WRITE(61,2124) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.5) WRITE(61,2125) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.6) WRITE(61,2126) ((S(I,J),J=K1,MBAND),I=1,NEQ)
      IF(K3.EQ.7) WRITE(61,2115) ((S(I,J),J=K1,MBAND),I=1,NEQ)
  178 WRITE(61,2130)
      WRITE(61,2135) (XB(I),I=1,NEQ)
  176 CONTINUE
C
C        SOLVE SYSTEM OF EQUATIONS S(I,J)*XB(I)=Y(I)
C     ******************************************************************
C
      IGAUS=1
      CALL LINSOLN
      IF(ICAL3.EQ.0) WRITE(61,2030)
C
C        OBTAIN VECTOR YB(I) FROM YB(I)=S1(I,J)*XB(I)
C     ******************************************************************
C
      READ(13,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      REWIND 13
C
C        HORIZONTAL SWEEP OF S1(I,J)*XB(I),DIAGONAL NOT INCLUDED
C     ******************************************************************
C
      DO 200 I=1,NEQ
      YB(I)=0.0
      II=I+1
      IF(II.GT.NEQ) GO TO 200
      DO 190 J=2,MBAND
      IF(S(I,J).EQ.0.) GO TO 180
      YB(I)=YB(I)+S(I,J)*XB(II)
  180 II=II+1
      IF(II.GT.NEQ) GO TO 200
  190 CONTINUE
  200 CONTINUE
C
C        DIAGONAL SWEEP OF S1(I,J)*XB(I)
C     ******************************************************************
C
      DO 230 I=1,NEQ
      II=I
      JJ=1
  210 IF(S(II,JJ).EQ.0.) GO TO 220
      YB(I)=YB(I)+S(II,JJ)*XB(II)
  220 II=II-1
      JJ=JJ+1
      IF(II.EQ.0) GO TO 230
      IF(JJ.GT.MBAND)  GO TO 230
      GO TO 210
  230 CONTINUE
C
C        COMPUTE RAYLEIGH QUOTIENT
C     ******************************************************************
C
      RQ=RHO
      Q1=Q2=0.
      DO 240 I=1,NEQ
      Q1=Q1+XB(I)*Y(I)
  240 Q2=Q2+XB(I)*YB(I)
      RHO=Q1/Q2
      DO 250 I=1,NEQ
  250 Y(I)=YB(I)/(ABS(Q2)**.5)
```

```
C
C        CHECK CONVERGENCE TO DESIRED EIGENVALUE
C     *****************************************************************
C
      CHECK=ABS((RHO-RQ)/RHO)
      IF(CHECK.LE.EPSI) GO TO 310
      EIGEN=RHO
      DO 260 I=1,NEQ
260   EIGNVTR(I)=XB(I)/(ABS(Q2)**.5)
      IF(ICAL3.EQ.0) WRITE(61,2035) K,EIGEN
      IF(ICAL3.NE.0) GO TO 300
      WRITE(61,2040) K,RHO,CHECK,EIGEN
      WRITE(61,2050)(XB(I),YB(I),Y(I),EIGNVTR(I),I=1,NEQ)
300   CONTINUE
C
C        OBTAIN EIGENVALUE AND CORRESPONDING EIGENVECTOR
C     *****************************************************************
C
310   EIGEN=RHO
      DO 320 I=1,NEQ
320   EIGNVTR(I)=XB(I)/(ABS(Q2)**.5)
      ILAST=K
      WRITE(61,2070) ILAST
      WRITE(61,2080) EIGEN
      WRITE(61,2090) (EIGNVTR(I),I=1,NEQ)
      RETURN
C
10    FORMAT(E21.15)
1000  FORMAT(I5,2F20.15)
2000  FORMAT(*-*,4HMAX=,I3//6H EPSI=,F20.15///6H  RHO=,F10.6)
2010  FORMAT(*1*,45HLINEAR EIGENVALUE PROBLEM (INVERSE ITERATION)//)
2030  FORMAT(*1*,38HINVERSE VECTOR ITERATION WITH SHIFTING///2X,1HK,9X,
     +    2HXB,16X,2HYB,16X,3HRHO,14X,5HCHECK,15X,1HY,15X,5HEIGEN,
     +    12X,7HEIGNVTR///)
2035  FORMAT(*-*,2HK=,I3,5X,6HEIGEN=,E15.9)
2040  FORMAT(*-*,I3,39X,E15.9,3X,E15.9,21X,E15.9)
2050  FORMAT(* *,6X,E15.9,3X,E15.9,39X,E15.9,21X,E15.9)
2100  FORMAT(*1*,34HDATA FOR GAUSSOL S(I,J) AND XB(I)//1X,2HK=,I3//)
2110  FORMAT(*-*,7HCOLUMNS,I4,10H   THROUGH,I4)
2115  FORMAT(*0*,8E16.8)
2120  FORMAT(*0*,E16.8)
2121  FORMAT(*0*,2E16.8)
2122  FORMAT(*0*,3E16.8)
2123  FORMAT(*0*,4E16.8)
2124  FORMAT(*0*,5E16.8)
2125  FORMAT(*0*,6E16.8)
2126  FORMAT(*0*,7E16.8)
2130  FORMAT(*-*,37HVECTOR Y(I), SENT TO GAUSSOL AS XB(I)//)
2135  FORMAT(*0*,10X,E15.9)
2070  FORMAT(*1*,5X,10HEIGENVALUE,9X,11HEIGENVECTOR,5X,6HILAST=,I3)
2080  FORMAT(*-*,E15.9)
2090  FORMAT(* *,20X,E15.9)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE ENDFORC
C
C     *****************************************************************
C           TO COMPUTE ELEMENT END FORCES
C     *****************************************************************
C
      COMMON/1/NE,NUMNP,NUMEG,LE(36),NUMEL(3),IPAR,ICAL1,ICAL2,ICAL3,
     +ISTRESS
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/3/IA(37,6),IB(37,6),X(37),Y(37),Z(37)
      COMMON/4/SF(12,12)
      COMMON/5/E(3),G(3),NODEI(36),NODEJ(36),A(36),IXX(36),KT(36),
     +L(3,36),IZZ(36),YPGM(36),ZPGM(36)
      COMMON/8/PI(37,6),PN(37,6),R(107)
      COMMON/11/DN(12),W(37,6),V(37,6)
      COMMON/12/JLOC(36,12),U(12),RCOL(9),MSUOPTN,N1GOPTN
C
      WRITE(61,2000)
      DO 100 LN=1,NUMNP
      DO 100 I=1,6
```

```
100   PI(LN,I)=0.
C
C        PROCESS EVERY ELEMENT OF EACH ELEMENT GROUP
C     *****************************************************************
C
      DO 200 K=1,NUMEG
      IF(NUMEL(K) .EQ.0) GO TO 200
      NAME=NUMEL(K)
      DO 190 KK=1,NAME
      M=L(K,KK)
      NI=NODEI(M)
      NJ=NODEJ(M)
      IF(K.EQ.2) GO TO 150
      READ(7,10)    ((SE(I,J),J=1,12),I=1,12)
      GO TO 151
150   READ(10,10)   ((SE(I,J),J=1,12),I=1,12)
151   CONTINUE
C
C
C        OBTAIN RESULTANT LOADS
C     *****************************************************************
C
      LN=NI
      DO 145 I=1,6
      DN(I)=0.
      DO 140 J=1,12
140   DN(I)=DN(I)+SE(I,J)*ULOC(M,J)
145   PI(LN,I)=PI(LN,I)+DN(I)
      LN=NJ
      DO 160 I=7,12
      DN(I)=0.
      DO 155 J=1,12
155   DN(I)=DN(I)+SE(I,J)*ULOC(M,J)
160   CONTINUE
C
C        WRITE RESULTANT LOADS OF THE NODES OF EACH ELEMENT
C     *****************************************************************
C
      WRITE(61,2010) M,(DN(I),I=1,6),(DN(I),I=7,12)
C
C
190   CONTINUE
200   CONTINUE
C
      REWIND 7
      REWIND 10
      RETURN
C
10    FORMAT(E21.15)
2000  FORMAT(10X,31HRESULTANT LOADS ON EACH ELEMENT///)
2010  FORMAT(*-*,8H ELEMENT,I3//4X,6HNODE-I,6E15.9//
     +4X,6HNODE-J,6E15.9)
C
      END
C
C
C
C
C
C
      SUBROUTINE NLEIGNP(SCALE)
C
C     *****************************************************************
C        THIS ROUTINE WILL COMPUTE THE EIGENVALUE OF THE
C        QUADRATIC EIGENVALUE PROBLEM (K+L*N1+L*L*N2)*X=0
C        IT USES THE MODIFIED REGULA FALSI METHOD
C     *****************************************************************
C
      EXTERNAL DET
      REAL L
C
      READ(60,1000)    XTOL,FTOL,NTOL,DINCR
      WRITE(61,2010)   XTOL,FTOL,NTOL,DINCR
      WRITE(61,2030)
      A=0.
100   FA=DET(A,SCALE)
      WRITE(61,2020)   A,FA
      IF(FA.LT.0.)     GO TO 110
      A=A+DINCR
      GO TO 100
110   CONTINUE
```

```
      B=A
      A=A-DINCR
      CALL MRGFLS(DET,A,B,XTOL,FTOL,NTOL,IFLAG,SCALE)
      IF(IFLAG.GT.2)  GO TO 500
      L=(A+B)/2.
      ERROR=ABS(B-A)/2.
      FL=DET(L,SCALE)
      WRITE(61,2000)  L,ERROR,FL
500   CONTINUE
      RETURN
C
1000  FORMAT(2F10.7,I10,F10.7)
2000  FORMAT(////14H  THE ROOT IS ,E25.15,10X,12H PLUS/MINUS ,E25.15//
     +    15H  DETERMINANT =,E25.15)
2010  FORMAT(*1*,28HQUADRATIC EIGENVALUE PROBLEM///6H XTOL=,F10.7///
     +   6H FTOL=,F10.7///6H NTOL=,I3///7H DINCR=,F10.7)
2020  FORMAT(*-*,E25.15,5X,E25.15//)
2030  FORMAT(////13X,6HLAMBDA,17X,11HDETERMINANT//)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE MRGFLS(F,A,B,XTOL,FTOL,NTOL,IFLAG,SCALE)
C
C     ******************************************************************
C        ITERATES TO A SUFFICIENTLY SMALL VALUE OF THE DETERMINANT
C        OR TO A SUFFICIENTLY SMALL INTERVAL WHERE THE ROOTS MAY
C        BE FOUND
C     ******************************************************************
C
      IFLAG=0
      FA=F(A,SCALE)
      SIGNFA=FA/ABS(FA)
      FB=F(B,SCALE)
C
C        CHECK FOR SIGN CHANGE
C     ******************************************************************
C
      IF(SIGNFA*FB.LE.0.)  GO TO 100
      IFLAG=3
      WRITE(61,2010) A,B
      RETURN
C
100   W=A
      FW=FA
      DO 400 N=1,NTOL
C
C        CHECK FOR SUFFICIENTLY SMALL INTERVAL
C     ******************************************************************
C
      IF(ABS(B-A)/2..LE.XTOL)  RETURN
C
C        CHECK FOR SUFFICIENTLY SMALL DETERMINANT VALUE
C PROTYPE=3   FOR INCREMENTAL LOADING IN MOVING COORDINATES
C
      IF(ABS(FW).GT.FTOL)  GO TO 200
      A=W
      B=W
      IFLAG=1
      RETURN
200   W=(FA*B-FB*A)/(FA-FB)
      PREVFW=FW/ABS(FW)
      FW=F(W,SCALE)
C
C        TEMPORARY PRINT OUT
C     ******************************************************************
C
      NM1=N-1
      WRITE(61,2020) NM1,A,W,B,FA,FW,FB
C
C        CHANGE TO NEW INTERVAL
C     ******************************************************************
C
      IF(SI GNFA*FW.LT.0.) GO TO 300
      A=W
      FA=FW
      IF(FW*PREVFW.GT.0.)  FB=FB/2.
      GO TO 400
300   B=W
      FB=FW
      IF(FW*PREVFW.GT.0.)  FA=FA/2.
400   CONTINUE
      IFLAG=2
      WRITE(61,2030) NTOL
      RETURN
C
2010  FORMAT(////43H F(X) IS OF SAME SIGN AT THE TWO ENDPOINTS ,
     +2E25.15)
2020  FORMAT(*-*,I3,9H L-VALUES,3E25.15//4X,9H F-VALUES,3E25.15//)
2030  FORMAT(////19H  NO CONVERGENCE IN,I5,11H ITERATIONS)
C
      END
C
C
C
C
C
C
C
      FUNCTION DET1(SCALE)
C
C     ****************************************************************
C        THIS FUNCTION COMPUTES THE VALUE OF THE DETERMINANT OF
C        THE MATRIX S=K+N1+N2
C     ****************************************************************
C
      COMMON/2/ NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/9/ S(60,20),SP(60,20),IDET
C
      IF(IDET.EQ.1) GO TO 250
      IF(IDET.EQ.2) GO TO 450
      DO 490 I=1,NEQ
      DO 490 J=1,MBAND
490   S(I,J)=SP(I,J)
C        FORWARD REDUCTION OF MATRIX(GAUSS ELEMINATION)
C     ****************************************************************
450   DO 390 LN=1,NEQ
      DO 380 LL=2,MBAND
      IF(S(LN,LL).EQ.0.) GO TO 380
      I=LN+LL-1
      C=S(LN,LL)/S(LN,1)
      J=0
      DO 350 KK=LL,MBAND
      J=J+1
350   S(I,J)=S(I,J)-C*S(LN,KK)
      S(LN,LL)=C
380   CONTINUE
390   CONTINUE
250   CONTINUE
C
C        COMPUTE DETERMINANT OF MATRIX S
C        SCALE DOWN"DET1" BY A "SCALE" VALUE AFTER EACH STEP
C     ****************************************************************
C
      DT=1.
      DO 400 I=1,NEQ
      DT=DT*S(I,1)/SCALE
400   CONTINUE
      DET1=DT
      RETURN
C
      END
C
C
C
C
C
C
C
      FUNCTION DET(L,SCALE)
C
C     ****************************************************************
C
      COMMON/2/NSIZE,NEQ,NCOND,MBAND,IEIGEN
      COMMON/9/S(60,20),SP(60,20),IDET
      REAL K,L,N1,N2
      IF(L.EQ.0.)  GO TO 220
      DO 210  I=1,NEQ
      DO 200  J=1,MBAND
      READ(4,10)  K
      IF(IEIGEN.EQ.1)  READ(6,10)  N1
      IF(IEIGEN.EQ.2)  READ(9,10)  N1
      READ(12,10)  N2
      S(I,J)=K+L*N1+L*L*N2
200   CONTINUE
210   CONTINUE
      GO TO 230
220   READ(4,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
230   REWIND  4
      REWIND  6
      REWIND  9
      REWIND  12
      DO 390  LN=1,NEQ
      DO 380  LL=2,MBAND
      IF(S(LN,LL).EQ.0.)  GO TO 380
      I=LN+LL-1
      C=S(LN,LL)/S(LN,1)
      J=0
      DO 350  KK=LL,MBAND
      J=J+1
350   S(I,J)=S(I,J)-C*S(LN,KK)
      S(LN,LL)=C
380   CONTINUE
390   CONTINUE
      DT=1.
      DO 400  I=1,NEQ
      DT=DT*S(I,1)/SCALE
400   CONTINUE
      DET=DT
      RETURN
10    FORMAT(E21.15)
      END
```

```
 14:48:13 07/13/81    SL11124     3033 LINES PRINT.
  40 PAGES PRINT.  COST AT RG3 IS   $    3.99
```

## D.4 PROGRAM NFRAL2D

```
      PROGRAM NFRAL2D(INPUT,OUTPUT=65,TAPE60=INPUT,TAPE61=OUTPUT,
     +TAPE1,TAPE2,TAPE3,TAPE4,TAPE5,TAPE6,TAPE7,TAPE8,TAPE9,TAPE10,
     +TAPE11,TAPE12)
C
C     ****************************************************************
C        THIS PROGRAM ANALYSIS TWO DIMENSIONAL FRAMED
C        STRUCTURES USING LAGRANGIAN COORDINATES.
C        THE FOLLOWING METHODS MAY BE SPECIFIED:
C        NEWTON-RAPHSON,STRAIGHT INCREMENTAL AND SUCCESSIVE
C        ITERATION.
C     ****************************************************************
C
      REAL IXX,LE,KEPSIO
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(11,3),X(11),Y(11)
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/8/PI(11,3),R(35)
      COMMON/9/S(35,12),SP(35,12),IDET
      COMMON/10/D(35)
      COMMON/11/W(11,3),WTOT(11,3),WCHK(11,3)
      COMMON/12/JLOC(10,6),U(6,1),MSUOPTN
      COMMON/13/PLOAD1,PINIT1,PLOAD2,PINIT2,PLOAD3,PINIT3,LODPON1,
     +LODPON2,LODPON3
      COMMON/17/A7TOT(10),A7OLD(10),BOL(10,5),BTO(10,5),BE(5)
      DIMENSION PTEMP(35),PSTART(35),DTOT(35),PACTUAL(35)
      DIMENSION PSAVE(35),DACTUAL(35),KEPSIO(35,12),PLOAD(35)
      DIMENSION SOLD(35,12),SRK(35,12),SRN1(35,12)
      DIMENSION REFSTRT(11,3),REFPTMP(11,3),SRN2(35,12),DTEMP(35)
      INTEGER PRIOPTN,HALFOPT,DETOPTN
C     ****************************************************************
C        TAPES 7,8,4   FOR K
C        TAPES 1,2,6   FOR N1
C        TAPES 3,5,9   FOR N2
C        TAPES 10,11,12  FOR KEPSIO
C     ****************************************************************
C
      READ(60,4000)  MTDTYPE,IEIGVAL,DETOPTN
4000  FORMAT(3I5)
C     ****************************************************************
C        MTDTYPE=1     FOR SECANT STIFFNESS OPTION
C        MTDTYPE=2     FOR FIXED LAGRANGE OPTION
C        MTDTYPE=3     FOR UPDATED LAGRANGE OPTION
C        IEIGVAL=1,MTDTYPE=1     FOR NONLINEAR EIGENVALUE PROBLEM
C        IEIGVAL=0     FOR THE CASE WE DON,T WANT EIGENVALUE SOLUTION
C        DETOPTN=0     NO CONTROL ON THE DETERMINANT OF  THE
C        TANGENT STIFFNESS MATRIX
C        DETOPTN=1 EXECUTION WOULD BE TERMINATED IF DETERMINANT OF
C        THE TANGENT STIFFNESS MATRIX IS NEGATIVE
C     ****************************************************************
C
      WRITE(61,4001)  MTDTYPE,IEIGVAL,DETOPTN
4001  FORMAT(/,10X,*MTDTYPE=*,I2,10X,*IEIGVAL=*,I2,/,10X,*DETOPTN=*,I2)
      IF(MTDTYPE.EQ.1)  READ(60,4002) TOLRANC,HALFOPT
4002  FORMAT(F15.10,I5)
      IF(MTDTYPE.EQ.1) WRITE(61,4003) TOLRANC,HALFOPT
4003  FORMAT(/,10X,*TOLERANCE=*,F15.10,/,10X,*HALFOPT=*,I5)
C     ****************************************************************
C        HALFOPT=1     FOR USING HALF STEP SIZE IF REQUIRED FOR
C        CONVERGENCE IN PREDEFINED NUMBER OF ITERATIONS
C        HALFOPT=0     TO USE THE SAME STEP SIZE EVERYWHERE
C        TOLRANC=ALLOWABLE TOLERANCE FOR CONVERGENCE CHECK
C        N1OPTIN=1  N1 SHOULD BE INCLUDED
C        N1OPTIN=0  N1 SHOULDN,T BE INCLUDED
C        THE SAME AS ABOVE FOR N2OPTIN
C        IF ISTRESS=1  STRESS SHOULD BE EVALUATED
C        IF ISTRESS=0  STRESS SHOULDN,T BE EVALUATED(EFFICIENT CODING)
C        PRIOPTN=0 IF WE JUST WANT THE RESULTS TO BE PRINTED
C        PRIOPTN=1 IF WE WANT INTERMEDIATE COMPUTATIONS PRINTED
C        ITERCHK=0  FOR STRAIGHT INCREMENTAL METHOD
C        ITERCHK=2  FOR NEWTON RAPHSON METHOD
C        MSUOPTN=1  CONSTANT STRAIN(AVERAGE) FOR EACH ELEMENT
C        MSUOPTN=2  STRAIN IS A QUADRATIC FUNCTION OF SLOPE AT EACH
C        POINT OF ELEMENT
C     ****************************************************************
C
      READ(60,6971)PRIOPTN,N2OPTIN,N1OPTIN,ITERCHK,MSUOPTN,ISTRESS
6971  FORMAT(6I5)
      WRITE(61,6972)PRIOPTN,N2OPTIN,N1OPTIN,ITERCHK,MSUOPTN,ISTRESS
6972  FORMAT(10X,8HPRIOPTN=,I2/10X,8HN2OPTIN=,I2/
     +10X,8HNIOPTIN=,I2/10X,8HITERCHK=,I2/10X,8HMSUCPTN=,I2,10X,*ISTRESS
     +=*,I2)
      IF(MTDTYPE.EQ.1)  GO TO 4024
C     ******************************************************************
C        DELTA1 AND DELTA2 ARE ALLOWABLE TOLERANCES FOR CONVERGENCE
C        CHECK.
C        ICHKOPT=1 FOR CONVERGENCE CHECK ON UNBALANCED FORCE COMPONENTS
C        ICHKOPT=2 FOR CONVERGENCE CHECK ON DISPLACEMENT COMPONENTS
C     ******************************************************************
C
      IF(ITERCHK.EQ.2) READ(60,6948)  DELTA1,DELTA2,ICHKOPT
 6948 FORMAT(2E21.15,I5)
      IF(ITERCHK.EQ.2) WRITE(61,6931) DELTA1,DELTA2,ICHKOPT
C
 4024 CONTINUE
 6931 FORMAT(10X,6HEPSI1=,E21.15/10X,6HEPSI2=,E21.15,9X,8HICHKOPT=,I5,/)
      READ(60,1) LODPON1,LODPON2,LODPON3,LODPON4,LODPON5,LODPON6
C     ******************************************************************
C        LODPON1 UP TO LODPON6 ARE THE ORDER OF D.O.F.(IN LINEAR
C        SYSTEM OF EQUATIONS) RELATED TO EXTERNAL CONCENTRATED
C        LOADS OR MOMENTS APPLIED ON THE STRUCTURE.
C        FOR EIGENVALUE PROBLEM SET LODPONI=0 (I=1,6)
C        PINIT1 UP TO PINIT6,PINC1 UP TO PINC6 AND PTOT1 UP TO PTOT6
C        ARE THE INITIAL,INCREMENTAL AND MAXIMUM DEFINED EXTERNAL
C        LOAD COMPONENTS.
C        MAXITER=MAXIMUM ALLOWABLE NUMBER OF ITERATIONS
C        FOR STRAIGHT INCCREMENTAL METHOD PUT MAXITER=1
C     ******************************************************************
C
      READ(60,699)  PINIT1,PINC1,PTOT1,PINIT2,PINC2,PTOT2,MAXITER
      WRITE(61,799) PINIT1,PINC1,PTOT1,PINIT2,PINC2,PTOT2,MAXITER
      READ(60,1699) PINIT3,PINC3,PTOT3,PINIT4,PINC4,PTOT4
 1699 FORMAT(6F10.6)
      WRITE(61,1799) PINIT3,PINC3,PTOT3,PINIT4,PINC4,PTOT4
 1799 FORMAT(10X,*PINIT3=*,F15.8,10X,10X,*PINC3=*,F15.8,/
     +10X,*PTOT3=*,F15.8,10X,*PINIT4=*,F15.8,/
     +10X,*PINC4=*,F15.8,10X,*PTOT4=*,F15.8,/)
      READ(60,1699)  PINIT5,PINC5,PTOT5,PINIT6,PINC6,PTOT6
      WRITE(61,1899) PINIT5,PINC5,PTOT5,PINIT6,PINC6,PTOT6
 1899 FORMAT(10X,*PINIT5=*,F15.8,10X,*PINC5=*,F15.8,/
     +10X,*PTOT5=*,F15.8,10X,*PINIT6=*,F15.8,
     +10X,*PINC6=*,F15.8,10X,*PTOT6=*,F15.8,/)
C
C     ******************************************************************
C        AT LEAST PINIT1 SHOULDN,T BE EQUAL TO ZERO
C        IGOPTIN=1 FOR CIRCULAR ARCH IGOPTIN=2 FOR PARABOLIC
C        AND IGOPTIN=0 FOR OTHER GEOMETRIES
C     ******************************************************************
C
      READ(60,10) IGOPTIN
      WRITE(61,8765) LODPON1,LODPON2,LODPON3,LODPON4,LODPON5,LODPON6
 8765 FORMAT(10X,8HLODPON1=,I2,10X,8HLODPON2=,I2,10X,8HLODPON3=,I2,
     +10X,*LODPON4=*,I2,10X,*LODPON5=*,I2,10X,*LODPON6=*,I2/)
      READ(60,1010) TITLE1,TITLE2,TITLE3,NE,NUMNP,ICAL1,ICAL2,ICAL3
C     ******************************************************************
C        ICAL1=0 FOR DISPLACEMENT VECTOR TO BE PRINTED
C        (ICAL1=1 SKIP)
C        ICAL2=0 FOR LINEAR MEMBER AND STRUCTURAL STIFFNESS MATRIX
C        TO BE PRINTED(ICAL2=1 SKIP)
C        ICAL3=0 FOR LOAD VECTOR TO BE PRINTED
C        (ICAL3=1 SKIP)
C     ******************************************************************
C
      WRITE(61,2010)TITLE1,TITLE2,TITLE3,NE,NUMNP,ICAL1,ICAL2,ICAL3
C
C        READ NODAL POINT DATA
C     ******************************************************************
C
      CALL NODDATA(IGOPTIN)
C
C        READ AND STORE INITIAL LOAD DATA
C     ******************************************************************
C
      WRITE(61,2015)
  500 READ(60,1015)  N,(PI(N,I),I=1,3)
      WRITE(61,2020) N,(PI(N,I),I=1,3)
      IF(N.NE.NUMNP) GO TO 500
C
C
C
C
```

```
      IF(MTDTYPE.NE.1)  GO TO 4004
      IPAR=1
      CALL ASEMBLE(M)
      CALL ELEMENT
      CALL BAND
      IPAR=2
      DO 4025 I=1,NEQ
      DO 4025 J=1,MBAND
4025  S(I,J)=0.0
      CALL BEAM
      SCALE=10000.
      PLOAD1=PINIT1
      PLOAD2=PINIT2
      PLOAD3=PINIT3
      IF(NEQ.GE.4) PLOAD4=PINIT4
      IF(NEQ.GE.5) PLOAD5=PINIT5
      IF(NEQ.GE.6) PLOAD6=PINIT6
      NN=MAXITER+1
4005  R(LODPON1)=PLOAD1
      R(LODPON2)=PLOAD2
      R(LODPON3)=PLOAD3
      IF(NEQ.GE.4) R(LODPON4)=PLOAD4
      IF(NEQ.GE.5) R(LODPON5)=PLOAD5
      IF(NEQ.GE.6) R(LODPON6)=PLOAD6
      IF(PLOAD1.EQ.PINIT1)  GO TO 4006
      ICOUNTR=0
      NUMITER=0
      DO 4007  I=1,NEQ
4007  DTEMP(I)=D(I)
4008  NUMITER=NUMITER+1
      IF(NUMITER.EQ.NN)  GO TO 4009
      CALL IDENT
      CALL INVTRNS
      DO 4010  I=1,NEQ
      DO 4010 J=1,MBAND
4010  S(I,J)=0.0
      IPAR=3
      CALL SBEAME1
      DO 4011  I=1,NEQ
      DO 4011  J=1,MBAND
4011  S(I,J)=0.0
      IPAR=4
      CALL SBEAME2
      DO 4012  I=1,NEQ
      DO 4012  J=1,MBAND
      READ(4,11)  RK
      READ(6,11)  RN1
      READ(9,11)  RN2
      S(I,J)=RK+.5*RN1+RN2/3.
      SP(I,J)=RK+RN1+RN2
4012  CONTINUE
      REWIND 4
      REWIND 6
      REWIND 9
      IF(IEIGVAL.EQ.0)   GO TO 4356
      CALL NLEIGNP(SCALE)
      GO TO 900
4356  CONTINUE
      CALL  LINSOLN
      IF(ITERCHK.EQ.0)  GO TO 4016
      IF(NUMITER.NE.1)  GO TO 4013
      UOLD=D(LODPON1)
      GO TO 4008
4013  CONTINUE
C        CONVERGENCE CHECK FOR THE LARGEST DISPLACEMENT COMPONENT
C     ************************************************************
      IF(ABS((UOLD-D(LODPON1))/D(LODPON1)).LE.TOLRANC) GO TO 4014
      UOLD=D(LODPON1)
      GO TO 4008
4014  CONTINUE
      IF(ISTRESS.NE.1)  GO TO 4238
C        TO HAVE ELEMENTS,INTERNAL FORCES AND STRESSES
C     ************************************************************
      DO 4107 M=1,NUMEL
      READ(7,11)((SE(I,J),J=1,6),I=1,6)
      READ(1,11)((SE1(I,J),J=1,6),I=1,6)
      READ(3,11)((SE2(I,J),J=1,6),I=1,6)
      DO 4108 I=1,6
      DO 4108 J=1,6
4108  SE(I,J)=SE(I,J)+SE1(I,J)/2.+SE2(I,J)/3.
      CALL STRESS(M)
```

```
4107  CONTINUE
      REWIND 7
      REWIND 1
      REWIND 3
4238  CONTINUE
      DO 4015 I=1,NEQ
4015  DTEMP(I)=D(I)
      WRITE(61,4115)
      DO 4119 I=1,NEQ
4119  WRITE(61,4117)  D(I)
4016  IDET=3
      WRITE(61,4017) PLOAD1,D(LODPON1),PLOAD2,D(LODPON2),
     +PLOAD3,D(LODPON3),PLOAD4,D(LODPON4),PLOAD5,D(LODPON5),PLOAD6,
     +D(LODPON6),NUMITER
4017  FORMAT(/,10X,*PLOAD1=*,E21.15,/
     +,10X,*DEFLEC1=*,E21.15,/
     +,10X,*PLOAD2=*,E21.15,/
     +,10X,*DEFLEC2=*,E21.15,/
     +,10X,*PLOAD3=*,E21.15,/
     +,10X,*DEFLEC3=*,E21.15,/
     +,10X,*PLOAD4=*,E21.15,/
     +,10X,*DEFLEC4=*,E21.15,/
     +,10X,*PLOAD5=*,E21.15,/
     +,10X,*DEFLEC5=*,E21.15,/
     +,10X,*PLOAD6=*,E21.15,/
     +,10X,*DEFLEC6=*,E21.15,/
     +,10X,*NO OF ITERATIONS=*,I5,/)
      DETER=DET1(SCALE)
      WRITE(61,4018) DETER
4018  FORMAT(/,10X,*DETERMINANT=*,E21.15,/)
      IF(DETER.LT.0.)  GO TO 900
      GO TO 4019
4006  IDET=1
      NUMITER=1
      CALL LINSOLN
      IF(ISTRESS.NE.1)  GO TO 4123
      IF(NUMITER.NE.1)  GO TO 4123
      CALL IDENT
      CALL INVTRNS
C        TO HAVE ELEMENTS, INTERNAL FORCES AND STRESSES
C     *********************************************************************
      DO 4109 M=1,NUMEL
      READ(7,11)((SE(I,J),J=1,6),I=1,6)
      CALL STRESS(M)
4109  CONTINUE
      REWIND 7
4123  CONTINUE
      DETER=DET1(SCALE)
      WRITE(61,4018) DETER
      IF(DETER.LE.0..AND.DETOPTN.EQ.1)   GO TO 900
      WRITE(61,4017) PLOAD1,D(LODPON1),PLOAD2,D(LODPON2),
     +PLOAD3,D(LODPON3),PLOAD4,D(LODPON4),PLOAD5,D(LODPON5),PLOAD6,
     +D(LODPON6),NUMITER
4019  PLOAD1=PLOAD1+PINC1
      PLOAD2=PLOAD2+PINC2
      PLOAD3=PLOAD3+PINC3
      IF(NEQ.GE.4) PLOAD4=PLOAD4+PINC4
      IF(NEQ.GE.5) PLOAD5=PLOAD5+PINC5
      IF(NEQ.GE.6) PLOAD6=PLOAD6+PINC6
      IF(ABS(PLOAD1).GT.ABS(PTOT1)) GO TO 900
      GO TO 4005
4009  ICOUNTR=ICOUNTR+1
      WRITE(61,4129) ICOUNTR,NUMITER
4129  FORMAT(/,10X,*ICOUNTR=*,I2,3X,*NUMITER=*,I2,/)
      IF(HALFOPT.EQ.0)  WRITE(61,4130)
      IF(HALFOPT.EQ.0)  GO TO 900
4130  FORMAT(9X,*NO WAY FOR CONVERGENCE WITH THIS PINC AND MAXITER*,)
      IF(ICOUNTR.LE.4)  GO TO 4021
      WRITE(61,4022)
4022  FORMAT(/,10X,*STEP SIZE HAS BEEN HALVED FOUR TIMES*,/)
      GO TO 900
4021  PINC1=PINC1/2.
      PINC2=PINC2/2.
      PINC3=PINC3/2.
      IF(NEQ.GE.4) PINC4=PINC4/2.
      IF(NEQ.GE.5) PINC5=PINC5/2.
      IF(NEQ.GE.6) PINC6=PINC6/2.
      WRITE(61,4131) PINC1,PINC2,PINC3
4131  FORMAT(9X,*PINC1=*,F15.9,3X,*PINC2=*,F15.9,3X,*PINC3=*,F15.9,)
      PLOAD1=PLOAD1-PINC1
      PLOAD2=PLOAD2-PINC2
      PLOAD3=PLOAD3-PINC3
      IF(NEQ.GE.4) PLOAD4=PLOAD4-PINC4
      IF(NEQ.GE.5) PLOAD5=PLOAD5-PINC5
      IF(NEQ.GE.6) PLOAD6=PLOAD6-PINC6
      DO 4023 I=1,NEQ
      DO 4023 J=1,MBAND
 4023 D(I)=DTEMP(I)
      GO TO 4005
C
C
C
 4004 CONTINUE
C
C
C
C
C
C
      IF(MTDTYPE.NE.2)  GO TO 4026
      IPAR=NUMITER=1
      CALL ASEMBLE(M)
      SCALE=10000.
      DO 4027 I=1,NEQ
 4027 DTOT(I)=0.0
      CALL ELEMENT
      CALL BAND
      PLOAD(LODPON1)=PINIT1
      PLOAD(LODPON2)=PINIT2
      PLOAD(LODPON3)=PINIT3
      IF(NEQ.GE.4) PLOAD(LODPON4)=PINIT4
      IF(NEQ.GE.5) PLOAD(LODPON5)=PINIT5
      IF(NEQ.GE.6) PLOAD(LODPON6)=PINIT6
      DO 4035  I=1,NEQ
      IF(I.NE.LODPON1.AND.I.NE.LODPON2.AND.I.NE.LODPON3
     *.AND.I.NE.LODPON4.AND.I.NE.LODPON5.AND.I.NE.LODPON6)  PLOAD(I)=0.
 4035 CONTINUE
      IPAR=2
      DO 4028 I=1,NEQ
      DO 4028 J=1,MBAND
 4028 S(I,J)=0.0
      CALL BEAM
 4029 IDET=1
      CALL LINSOLN
      IF(ITERCHK.EQ.0)  GO TO 4289
      IF(ICHKOPT.EQ.1)  GO TO 4289
      DO 4827 M=1,NUMEL
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 4827  K1=1,2
      IF(K1.EQ.1)  NP=NI
      IF(K1.EQ.2)  NP=NJ
      DO 4828  I=1,3
      IF(IA(NP,I)) 4828,4829,4830
 4830 NL=IA(NP,I)
      WCHK(NP,I)=D(NL)
      GO TO 4828
 4829 WCHK(NP,I)=0.0
 4828 CONTINUE
 4827 CONTINUE
 4289 CONTINUE
      IF(ITERCHK.NE.0)  GO TO 4269
      IF(ISTRESS.NE.1)  GO TO 4269
      IF(IPAR.EQ.2) CALL IDENT
      IF(IPAR.EQ.2) CALL INVTRNS
C         TO HAVE ELEMENTS, INTERNAL FORCES AND STRESSES
C     ******************************************************************
      DO 4110 M=1,NUMEL
      READ(7,11) ((SE(I,J),J=1,6),I=1,6)
      IF(IPAR.NE.2)    READ(1,11)((SE1(I,J),J=1,6),I=1,6)
      IF(IPAR.NE.2)    READ(3,11)((SE2(I,J),J=1,6),I=1,6)
      DO 4111 I=1,6
      DO 4111 J=1,6
      IF(IPAR.NE.2)    SE(I,J)=SE(I,J)+SE1(I,J)/2.+SE2(I,J)/3.
 4111 CONTINUE
      CALL STRESS(M)
 4110 CONTINUE
      REWIND 7
      IF(IPAR.NE.2) REWIND 1
      IF(IPAR.NE.2) REWIND 3
 4269 CONTINUE
      DO 4030 I=1,NEQ
      DTOT(I)=DTOT(I)+D(I)
 4030 D(I)=DTOT(I)
      IF(ITERCHK.EQ.0)  GO TO 4290
      IF(ICHKOPT.EQ.1)  GO TO 4290
      DO 4831  M=1,NUMEL
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 4831  K1=1,2
      IF(K1.EQ.1)  NP=NI
      IF(K1.EQ.2)  NP=NJ
      DO 4832  I=1,3
      IF(IA(NP,I))  4832,4833,4834
 4834 NL=IA(NP,I)
      WTOT(NP,I)=DTOT(NL)
      GO TO 4832
 4833 WTOT(NP,I)=0.0
 4832 CONTINUE
 4831 CONTINUE
      SUM1=SUM2=SUM3=SUM4=0.0
      DO 4825  I=1,NUMNP
      SUM1=SUM1+WCHK(I,1)**2+WCHK(I,2)**2
      SUM2=SUM2+WCHK(I,3)**2
      SUM3=SUM3+WTOT(I,1)**2+WTOT(I,2)**2
 4825 SUM4=SUM4+WTOT(I,3)**2
      ERROR1=SQRT(SUM1/SUM3)
      ERROR2=SQRT(SUM2/SUM4)
      IF(PRIOPTN.EQ.0)  GO TO 4826
      WRITE(61,9038)    ERROR1,ERROR2
 4826 CONTINUE
 4290 CONTINUE
      IF(ITERCHK.NE.0)  GO TO 4132
      DETER=DET1(SCALE)
      WRITE(61,4031) DETER
 4031 FORMAT(/,10X,*DETERMINANT=*,E21.15,/)
      IF(DETER.LE.0..AND.DETOPTN.EQ.1)    GO TO 900
 4132 CONTINUE
      CALL IDENT
      CALL INVTRNS
      IPAR=3
      DO 4032  I=1,NEQ
      DO 4032 J=1,MBAND
 4032 S(I,J)=0.0
      CALL SBEAME1
      IPAR=4
      DO 4033 I=1,NEQ
      DO 4033 J=1,MBAND
 4033 S(I,J)=0.0
      CALL SBEAME2
      DO 4034 I=1,NEQ
      DO 4034 J=1,MBAND
      READ(4,11)  RK
      READ(6,11)  RN1
      READ(9,11)  RN2
      S(I,J)=RK+RN1+RN2
      IF(ITERCHK.NE.0)  SP(I,J)=RK+.5*RN1+RN2/3.
 4034 CONTINUE
      REWIND 4
      REWIND 6
      REWIND 9
      IF(ITERCHK.EQ.0)  GO TO 4057
C        TO HAVE PTEMP(I)=SP(I,J)*D(I)
C     ***************************************************************
      DO 4036 I=1,NEQ
      PTEMP(I)=0.0
      IM=I+1
      IF(IM.GT.NEQ)  GO TO 4036
      DO 4037 J=2,MBAND
      IF(SP(I,J).EQ.0.)  GO TO 4038
      PTEMP(I)=PTEMP(I)+SP(I,J)*D(IM)
 4038 IM=IM+1
      IF(IM.GT.NEQ)  GO TO 4036
 4037 CONTINUE
 4036 CONTINUE
      DO 4039 I=1,NEQ
      IM=I
      JM=1
 4040 IF(SP(IM,JM).EQ.0.) GO TO 4041
      PTEMP(I)=PTEMP(I)+SP(IM,JM)*D(IM)
 4041 IM=IM-1
      JM=JM+1
      IF(IM.EQ.0)       GO TO 4039
      IF(JM.GT.MBAND)  GO TO 4039
      GO TO 4040
 4039 CONTINUE
      DO 4043 I=1,NEQ
 4043 R(I)=PLOAD(I)-PTEMP(I)
C         TO CHECK CONVERGENCE
C     *****************************************************************
      IF(ICHKOPT.EQ.1)  GO TO 4292
      IF(ERROR1.GT.DELTA1.OR.ERROR2.GT.DELTA2)  GO TO 4293
      GO TO 4961
 4292 CONTINUE
      DO 4047 M=1,NUMEL
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 4048 K1=1,2
      IF(K1.EQ.1) NP=NI
      IF(K1.EQ.2) NP=NJ
      DO 4049 I=1,3
      IF(IA(NP,I))4049,4050,4051
 4051 NL=IA(NP,I)
      REFSTRT(NP,I)=PLOAD(NL)
      REFPTMP(NP,I)=PTEMP(NL)
      GO TO 4049
 4050 REFSTRT(NP,I)=0.0
      REFPTMP(NP,I)=0.0
 4049 CONTINUE
 4048 CONTINUE
 4047 CONTINUE
      DO 4052 NP=1,NUMNP
      PART1=ABS(REFSTRT(NP,1)-REFPTMP(NP,1))
      PART2=ABS(REFSTRT(NP,2)-REFPTMP(NP,2))
      PART3=ABS(REFSTRT(NP,3)-REFPTMP(NP,3))
      IF(PART1.GT.DELTA1.OR.PART2.GT.DELTA1.OR.PART3.GT.DELTA2)
     +GO TO 4053
      WRITE(61,4060) PART1,PART2,PART3
 4052 CONTINUE
 4961 CONTINUE
      IF(ISTRESS.NE.1)  GO TO 4233
C         TO HAVE ELEMENTS, INTERNAL FORCES AND STRESSES
C     *****************************************************************
      DO 4133 M=1,NUMEL
      READ(7,11) ((SE(I,J),J=1,6),I=1,6)
      READ(1,11) ((SE1(I,J),J=1,6),I=1,6)
      READ(3,11) ((SE2(I,J),J=1,6),I=1,6)
      DO 4134 I=1,6
      DO 4134 J=1,6
 4134 SE(I,J)=SE(I,J)+SE1(I,J)/2.+SE2(I,J)/3.
      CALL STRESS(M)
 4133 CONTINUE
      REWIND 7
      REWIND 1
      REWIND 3
 4233 CONTINUE
      DETER=DET1(SCALE)
      WRITE(61,4031) DETER
      IF(DETER.LE.0..AND.DETOPTN.EQ.1)   GO TO 900
      WRITE(61,4115)
      DO 4118 I=1,NEQ
 4118 WRITE(61,4117) DTOT(I)
      WRITE(61,4055) PTEMP(LODPON1),PTEMP(LODPON2),PTEMP(LODPON3)
     +,PTEMP(LODPON4),PTEMP(LODPON5),PTEMP(LODPON6)
      WRITE(61,4056)DTOT(LODPON1),DTOT(LODPON2),DTOT(LODPON3),
     +DTOT(LODPON4),DTOT(LODPON5),DTOT(LODPON6),NUMITER
 4055 FORMAT(/,10X,*PTEMP(LODPON1)=*,F15.8,/
     +10X,*PTEMP(LODPON2)=*,F15.8,/
     +10X,*PTEMP(LODPON3)=*,F15.8,/
     +10X,*PTEMP(LODPON4)=*,F15.8,/
     +10X,*PTEMP(LODPON5)=*,F15.8,/
     +10X,*PTEMP(LODPON6)=*,F15.8,/
     +10X,*PTEMP(LODPON6)=*,F15.8,/)
 4056 FORMAT(/,10X,*DTOT(LODPON1)=*,F15.10,/
     +10X,*DTOT(LODPON2)=*,F15.10,/
     +10X,*DTOT(LODPON3)=*,F15.10,/
     +10X,*DTOT(LODPON4)=*,F15.10,/
     +10X,*DTOT(LODPON5)=*,F15.10,/
     +10X,*DTOT(LODPON6)=*,F15.8,/
     +10X,*NO OF ITERATIONS=*,I5,/)
      GO TO 4054
 4053 WRITE(61,4060) PART1,PART2,PART3
 4060 FORMAT(10X,*PART1=*,E21.15,10X,*PART2=*,E21.15,10X,
     +*PART3=*,E21.15,/)
 4293 NUMITER=NUMITER+1
```

```
      IF(NUMITER.LT.MAXITER) GO TO 4029
      DETER=DET1(SCALE)
      WRITE(61,4018)  DETER
      GO TO 900
4057  R(LODPON1)=PINC1
      R(LODPON2)=PINC2
      R(LODPON3)=PINC3
      IF(NEQ.GE.4) R(LODPON4)=PINC4
      IF(NEQ.GE.5) R(LODPON5)=PINC5
      IF(NEQ.GE.6) R(LODPON6)=PINC6
      DO 4058 I=1,NEQ
      IF(I.NE.LODPON1.AND.I.NE.LODPON2.AND.I.NE.LODPON3
     +.AND.I.NE.LODPON4.AND.I.NE.LODPON5.AND.I.NE.LODPON6)    R(I)=0.
4058  CONTINUE
      WRITE(61,4055)PLOAD(LODPON1),PLOAD(LODPON2),PLOAD(LODPON3)
     +,PLOAD(LODPON4),PLOAD(LODPON5),PLOAD(LODPON6)
      WRITE(61,4056)DTOT(LODPON1),DTOT(LODPON2),DTOT(LODPON3),
     +DTOT(LODPON4),DTOT(LODPON5),DTOT(LODPON6),NUMITER
4054  IF(ABS(PLOAD(LODPON1)).LT.ABS(PTOT1))    GO TO 4059
      GO TO 900
4059  IF(ITERCHK.EQ.0)           GO TO 4061
      R(LODPON1)=R(LODPON1)+PINC1
      R(LODPON2)=R(LODPON2)+PINC2
      R(LODPON3)=R(LODPON3)+PINC3
      IF(NEQ.GE.4) R(LODPON4)=R(LODPON4)+PINC4
      IF(NEQ.GE.5) R(LODPON5)=R(LODPON5)+PINC5
      IF(NEQ.GE.6) R(LODPON6)=R(LODPON6)+PINC6
4061  PLOAD(LODPON1)=PLOAD(LODPON1)+PINC1
      PLOAD(LODPON2)=PLOAD(LODPON2)+PINC2
      PLOAD(LODPON3)=PLOAD(LODPON3)+PINC3
      IF(NEQ.GE.4) PLOAD(LODPON4)=PLOAD(LODPON4)+PINC4
      IF(NEQ.GE.5) PLOAD(LODPON5)=PLOAD(LODPON5)+PINC5
      IF(NEQ.GE.6) PLOAD(LODPON6)=PLOAD(LODPON6)+PINC6
      NUMITER=1
      GO TO 4029
C
C
C
C
4026  CONTINUE
      SCALE=10000.
      DO 5001 I=1,NEQ
5001  PSAVE(I)=DACTUAL(I)=DTOT(I)=0.0
      PLOAD1=PINIT1
      PLOAD2=PINIT2
      PLOAD3=PINIT3
      IF(NEQ.GE.4) PLOAD4=PINIT4
      IF(NEQ.GE.5) PLOAD5=PINIT5
      IF(NEQ.GE.6) PLOAD6=PINIT6
      DO 3010  I=1,NUMNP
      DO 3010  J=1,3
3010  W(I,J)=0.0
      ICHECK=1
1001  DO 3020 I=1,NUMNP
      X(I)=X(I)+W(I,1)
      Y(I)=Y(I)+W(I,2)
3020  CONTINUE
      IF(PRIOPTN.EQ.0)   GO TO 4956
      WRITE(61,4957)
4957  FORMAT(/,10X,*NODE*,10X,*X(I)*,10X,*Y(I)*,/)
      DO 4958 I=1,NUMNP
      WRITE(61,4959)  I,X(I),Y(I)
4959  FORMAT(/,10X,I5,2F15.8)
4958  CONTINUE
4956  CONTINUE
C        READ AND STORE ELEMENT DATA
C     *****************************************************************
C
      IPAR=NUMITER=1
      IF(ICHECK.NE.1) GO TO 4093
      CALL ELEMENT
      GO TO 4094
4093  DO 5555 M=1,NE
      XI=X(NODEI(M))
      YI=Y(NODEI(M))
      XJ=X(NODEJ(M))
      YJ=Y(NODEJ(M))
5555  LE(M)=SQRT((XJ-XI)**2+(YJ-YI)**2)
C
4094  IF(ICHECK.NE.1) GO TO 3333
C        COMPUTE SEMIBANDWIDTH OF STRUCTURE STIFFNESS MATRIX
```

```
C      ******************************************************************
       CALL BAND
       DO 5341 M=1,NUMEL
       IF(MSUOPTN.EQ.1)  A7OLD(M)=0.0
       DO 5341 I=1,5
       IF(MSUOPTN.EQ.2)  BOL(M,I)=0.0
5341   CONTINUE
C
C
C
C         ASSEMBLE INITIAL LOADS AND NODAL LOADS INTO LOAD VECTOR
C      ******************************************************************
       CALL ASEMBLE(M)
3333   CONTINUE
       IF(ICHECK.EQ.1) GO TO 4095
       GO TO 4096
4095   DO 5003  I=1,NEQ
5003   PSTART(I)=0.0
       PSTART(LODPON1)=PINIT1
       PSTART(LODPON2)=PINIT2
       PSTART(LODPON3)=PINIT3
       IF(NEQ.GE.4) PSTART(LODPON4)=PINIT4
       IF(NEQ.GE.5) PSTART(LODPON5)=PINIT5
       IF(NEQ.GE.6) PSTART(LODPON6)=PINIT6
4096   CONTINUE
       IF(ICHECK.EQ.1)   GO TO 3336
       IPAR=2
       DO 4097  I=1,NEQ
       DO 4097  J=1,MBAND
4097   S(I,J)=0.0
       CALL BEAM
1901   IF(ITERCHK.NE.0) CALL INVTRNS
       IF(ICHECK.EQ.3)  GO TO 7692
       IF(N1OPTIN.EQ.0) GO TO 2101
       DO 1801 I=1,NEQ
       DO 1801 J=1,MBAND
1801   S(I,J)=0.0
       IPAR=3
       CALL SBEAME1
2101   CONTINUE
       IF(ICHECK.EQ.3)  GO TO 7692
       IF(N2OPTIN.EQ.0) GO TO 7692
       DO 7691 I=1,NEQ
       DO 7691 J=1,MBAND
7691   S(I,J)=0.0
       IPAR=4
       CALL SBEAME2
7692   CONTINUE
       IF(ICHECK.EQ.2.OR.PSAVE(LODPON1).EQ.0.)  GO TO 5010
       IPAR=7
       DO 4065 I=1,NEQ
       DO 4065 J=1,MBAND
4065   S(I,J)=0.0
       CALL KEPSI01
       IF(ISTRESS.NE.1)  GO TO 4239
C         TO HAVE ELEMENTS, INTERNAL FORCES AND STRESSES
C      ******************************************************************
       DO 4105 M=1,NUMEL
       READ(7,11) ((SE(I,J),J=1,6),I=1,6)
       IF(PSAVE(LODPON1).NE.0.) READ(10,11)
      +((SE1(I,J),J=1,6),I=1,6)
       DO 4106 I=1,6
       DO 4106 J=1,6
       IF(PSAVE(LODPON1).NE.0.)SE(I,J)=SE(I,J)+SE1(I,J)
4106   CONTINUE
       CALL STRESS(M)
4105   CONTINUE
       REWIND 7
       IF(PSAVE(LODPON1).NE.0.) REWIND 10
4239   CONTINUE
       DO 3071 I=1,NEQ
       DO 3081 J=1,MBAND
       READ(4,11) RK
       SRK(I,J)=RK
       READ(12,11) RN1STAR
       SRN1(I,J)=RN1STAR
       KEPSI0(I,J)=RN1STAR
       S(I,J)=SOLD(I,J)=RK+KEPSI0(I,J)
3081   CONTINUE
3071   CONTINUE
       REWIND 4
      REWIND 12
      IF(PRIOPTN.EQ.0) GO TO 7233
      WRITE(61,8005)
8005  FORMAT(///,10X,*KEPSIO MATRIX*,/)
      WRITE(61,8002)  ((SRN1(I,J),J=1,MBAND),I=1,NEQ)
      WRITE(61,8008)
8008  FORMAT(///,10X,*K LINEAR STIFFNESS MATRIX*,/)
      WRITE(61,8002)  ((SRK(I,J),J=1,MBAND),I=1,NEQ)
      WRITE(61,8009)
8009  FORMAT(///,10X,*S(I,J)   MATRIX*,/)
      WRITE(61,8002)  ((S(I,J),J=1,MBAND),I=1,NEQ)
7233  CONTINUE
      GO TO 5011
5010  DO 4071 I=1,NEQ
      DO 4081 J=1,MBAND
      IF(N1OPTIN.EQ.1) READ(6,11) RN1
      IF(PSAVE(LODPON1).EQ.0.)  READ(4,11)  RK
      IF(N2OPTIN.EQ.1) READ(9,11) RN2
      IF(N1OPTIN.EQ.1) SRN1(I,J)=RN1
      IF(PSAVE(LODPON1).EQ.0..AND.N2OPTIN.EQ.1)SP(I,J)=RK+.5*RN1+RN2/3.
      IF(PSAVE(LODPON1).NE.0..AND.N2OPTIN.EQ.1)
     +SP(I,J)=SOLD(I,J)+.5*RN1+RN2/3.
      IF(PSAVE(LODPON1).EQ.0..AND.N1OPTIN.EQ.0)SP(I,J)=S(I,J)=RK
      IF(PSAVE(LODPON1).EQ.0..AND.N1OPTIN.EQ.1.AND.N2OPTIN.EQ.0)SP(I,J)=
     +RK+.5*RN1
      IF(PSAVE(LODPON1).NE.0..AND.N1OPTIN.EQ.0)SP(I,J)=S(I,J)=SOLD(I,J)
      IF(PSAVE(LODPON1).NE.0..AND.N1OPTIN.EQ.1.AND.N2OPTIN.EQ.0)SP(I,J)=
     +SOLD(I,J)+.5*RN1
      IF(PSAVE(LODPON1).EQ.0..AND.N2OPTIN.EQ.1)S(I,J)=RK+RN1+RN2
      IF(PSAVE(LODPON1).NE.0..AND.N2OPTIN.EQ.1)S(I,J)=SOLD(I,J)+RN1+RN2
      IF(PSAVE(LODPON1).EQ.0..AND.N1OPTIN.EQ.1.AND.N2OPTIN.EQ.0)
     +S(I,J)=RK+RN1
      IF(PSAVE(LODPON1).NE.0..AND.N1OPTIN.EQ.1.AND.N2OPTIN.EQ.0)
     +S(I,J)=SOLD(I,J)+RN1
4081  CONTINUE
4071  CONTINUE
      IF(N1OPTIN.EQ.1) REWIND 6
      IF(N2OPTIN.EQ.1) REWIND 9
      IF(PSAVE(LODPON1).EQ.0.)  REWIND 4
      IF(PRIOPTN.EQ.0) GO TO 5011
      IF(N2OPTIN.EQ.0) GO TO 8075
      WRITE(61,7693)
7693  FORMAT(///,10X,11HN2   MATRIX,/)
      DO 7694 I=1,NEQ
      DO 7695 J=1,MBAND
      READ(9,11) RN2
      SRN2(I,J)=RN2
7695  CONTINUE
7694  CONTINUE
8075  CONTINUE
      IF(N2OPTIN.EQ.1) REWIND 9
      IF(N2OPTIN.EQ.1) WRITE(61,8002)((SRN2(I,J),J=1,MBAND),I=1,NEQ)
      IF(N1OPTIN.EQ.1) WRITE(61,8004)
8004  FORMAT(//,10X,*N1 NONLINEAR STIFFNESS MATRIX*,/)
      IF(N1OPTIN.EQ.1) WRITE(61,8002)  ((SRN1(I,J),J=1,MBAND),I=1,NEQ)
      IF(PSAVE(LODPON1).NE.0.) WRITE(61,8010)
8010  FORMAT(///,10X,*SOLD(I,J)  MATRIX*,/)
      IF(PSAVE(LODPON1).NE.0.) WRITE(61,8002)
     +((SOLD(I,J),J=1,MBAND),I=1,NEQ)
      WRITE(61,8018)
8018  FORMAT(///,10X,*SP(I,J) MATRIX*,/)
      WRITE(61,8002)  ((SP(I,J),J=1,MBAND),I=1,NEQ)
      WRITE(61,8009)
      WRITE(61,8002)  ((S(I,J),J=1,MBAND),I=1,NEQ)
5011  IF(ICHECK.NE.3)  GO TO 7001
      GO TO 6001
7001  ICHECK=3
      GO TO 3339
3336  CONTINUE
C
C     COMPUTE ELEMENT LINEAR STIFFNESS AND ASSEMBLE INTO STRUCTURE
C        LINEAR STIFFNESS
C
C     ***************************************************************
C
C
      DO 4098  I=1,NEQ
      DO 4098  J=1,MBAND
4098  S(I,J)=0.0
      IPAR=2
      CALL BEAM
```

```
C
      DO 1071 I=1,NEQ
      DO 1081 J=1,MBAND
      READ(4,11) RK
      S(I,J)=SP(I,J)=RK
1081  CONTINUE
1071  CONTINUE
      REWIND 4
      IF(PRIOPTN.EQ.0) GO TO 9431
      IF(ICHECK.EQ.1) WRITE(61,8008)
      IF(ICHECK.EQ.1) WRITE(61,8002)((S(I,J),J=1,MBAND),I=1,NEQ)
9431  CONTINUE
6001  IDET=1
8002  FORMAT(1X,6(2X,E19.13),/)
      CALL LINSOLN
      IF(ITERCHK.NE.2) GO TO 4063
      DO 230 M=1,NUMEL
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 230 K1=1,2
      IF(K1.EQ.1)  NP=NI
      IF(K1.EQ.2)  NP=NJ
      DO 220 I=1,3
      IF(IA(NP,I))  220,155,150
150   NL=IA(NP,I)
      WCHK(NP,I)=D(NL)
      GO TO 220
155   WCHK(NP,I)=0.0
220   CONTINUE
230   CONTINUE
4063  CONTINUE
      DETRMNT=DET1(SCALE)
      DO 5005 I=1,NEQ
      DTOT(I)=DTOT(I)+D(I)
      DACTUAL(I)=DACTUAL(I)+D(I)
5005  D(I)=DTOT(I)
      IF(ITERCHK.NE.2)  GO TO 4064
      DO 9230 M=1,NUMEL
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 9230 K1=1,2
      IF(K1.EQ.1)  NP=NI
      IF(K1.EQ.2)  NP=NJ
      DO 9230 I=1,3
      IF(IA(NP,I)) 9220,9155,9150
9150  NL=IA(NP,I)
      WTOT(NP,I)=DACTUAL(NL)
      GO TO 9220
9155  WTOT(NP,I)=0.0
9220  CONTINUE
9230  CONTINUE
4064  CONTINUE
      CALL IDENT
      IF(ITERCHK.EQ.0)  CALL INVTRNS
      IF(ITERCHK.NE.0)  GO TO 4120
      DO 4092 M=1,NUMEL
4092  A7TOT(M)=A7OLD(M)+ULOC(M,4)-ULOC(M,1)
4120  CONTINUE
      ICHECK=2
      IF(ITERCHK.NE.0)  GO TO 4066
      DO 4067 I=1,NEQ
      R(I)=0.0
4067  PACTUAL(I)=PSAVE(I)+PSTART(I)
      GO TO 3342
4066  CONTINUE
      GO TO 1901
3339  DO 2001 I=1,NEQ
      PTEMP(I)=0.0
      IM=I+1
      IF(IM.GT.NEQ) GO TO 2001
      DO 3901 J=2,MBAND
      IF(SP(I,J).EQ.0.)  GO TO 1804
      PTEMP(I)=PTEMP(I)+SP(I,J)*D(IM)
1804  IM=IM+1
      IF(IM.GT.NEQ)  GO TO 2001
3901  CONTINUE
2001  CONTINUE
      DO 2301 I=1,NEQ
      IM=I
      JM=1
2108  IF(SP(IM,JM).EQ.0.) GO TO 2201
```

```
      PTEMP(I)=PTEMP(I)+SP(IM,JM)*D(IM)
 2201 IM=IM-1
      JM=JM+1
      IF(IM.EQ.0)  GO TO 2301
      IF(JM.GT.MBAND)  GO TO 2301
      GO TO 2108
 2301 CONTINUE
      DO 5006 I=1,NEQ
 5006 PACTUAL(I)=PTEMP(I)+PSAVE(I)
      IF(ITERCHK.EQ.0) GO TO 6975
      IF(PRIOPTN.EQ.1) WRITE(61,8011)
 8011 FORMAT(15X,*I*,5X,*PACTUAL(I)*,10X,*PTEMP(I)*,10X,*PSAVE(I)*,//)
      IF(PRIOPTN.EQ.0) GO TO 8068
      DO 8012 I=1,NEQ
      WRITE(61,8013)  I,PACTUAL(I),PTEMP(I),PSAVE(I)
 8012 CONTINUE
 8068 CONTINUE
 8013 FORMAT(10X,I5,5X,E21.15,10X,E21.15,10X,E21.15,/)
      WRITE(61,9001) PACTUAL(LODPON1),DACTUAL(LODPON1)
 9001 FORMAT(10X,*PLOAD1=*,F15.9/10X,*DEFLEC1=*,F15.9)
      WRITE(61,9002) PACTUAL(LODPON2),DACTUAL(LODPON2)
 9002 FORMAT(10X,*PLOAD2=*,F15.9/10X,*DEFLEC2=*,F15.9)
      WRITE(61,9003) PACTUAL(LODPON3),DACTUAL(LODPON3)
 9003 FORMAT(10X,*PLOAD3=*,F15.9/10X,*DEFLEC3=*,F15.9)
 6975 CONTINUE
      DO 4099 I=1,NEQ
 4099 R(I)=PSTART(I)-PTEMP(I)
      IF(PRIOPTN.EQ.0) GO TO 6976
      WRITE(61,9731)
 9731 FORMAT(//,20X,*R(I)*,15X,*PSTART(I)*,15X,*PTEMP(I)*,/)
      DO 9732  IMM=1,NEQ
 9732 WRITE(61,9733)  R(IMM),PSTART(IMM),PTEMP(IMM)
 9733 FORMAT(10X,E21.15,10X,E21.15,10X,E21.15)
 6976 CONTINUE
      IF(ITERCHK.NE.2)  GO TO 3342
      IF(ICHKOPT.EQ.2)  GO TO 4101
      DO 2451 M=1,NUMEL
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 2351  K1=1,2
      IF(K1.EQ.1)  NP=NI
      IF(K1.EQ.2)  NP=NJ
      DO 2251  I=1,3
      IF(IA(NP,I))  2251,1551,1571
 1571 NL=IA(NP,I)
      REFSTRT(NP,I)=PSTART(NL)
      REFPTMP(NP,I)=PTEMP(NL)
      GO TO 2251
 1551 REFSTRT(NP,I)=0.0
      REFPTMP(NP,I)=0.0
 2251 CONTINUE
 2351 CONTINUE
 2451 CONTINUE
      DO 6949 NP=1,NUMNP
      PART1=ABS(REFSTRT(NP,1)-REFPTMP(NP,1))
      PART2=ABS(REFSTRT(NP,2)-REFPTMP(NP,2))
      PART3=ABS(REFSTRT(NP,3)-REFPTMP(NP,3))
      IF(PART1.GT.DELTA1.OR.PART2.GT.DELTA1.OR.PART3.GT.DELTA2)
     *GO TO 6950
      WRITE(61,4100)  PART1,PART2,PART3
 6949 CONTINUE
      GO TO 3342
 6950 WRITE(61,4100)  PART1,PART2,PART3
 4100 FORMAT(10X,*PART1=*,E21.15,10X,*PART2=*,E21.15,10X,
     **PART3=*,E21.15)
      GO TO 4102
 4101 SUM1=SUM2=SUM3=SUM4=0.0
      DO 1400 I=1,NUMNP
      SUM1=SUM1+WCHK(I,1)**2+WCHK(I,2)**2
      SUM2=SUM2+WCHK(I,3)**2
      SUM3=SUM3+WTOT(I,1)**2+WTOT(I,2)**2
 1400 SUM4=SUM4+WTOT(I,3)**2
      ERROR1=SQRT(SUM1/SUM3)
      ERROR2=SQRT(SUM2/SUM4)
      IF(PRIOPTN.EQ.0) GO TO 9037
      WRITE(61,9038) ERROR1,ERROR2
 9038 FORMAT(/,10X,*ERROR1=*,F15.8,10X,*ERROR2=*,F15.8,/)
 9037 CONTINUE
      IF(ERROR1.GT.DELTA1.OR.ERROR2.GT.DELTA2) GO TO 4102
      GO TO 3342
 4102 NUMITER=NUMITER+1
```

```
      IF(NUMITER.LE.MAXITER)  GO TO  4103
      WRITE(61,4018)  DETRMNT
      GO TO 900
4103  GO TO 6001
3342  CONTINUE
      IF(ITERCHK.NE.2)  GO TO 4126
      IF(MSUOPTN.EQ.2)  GO TO 4126
      DO 4127 M=1,NUMEL
      TA=ULOC(M,3)-(ULOC(M,5)-ULOC(M,2))/LE(M)
      TB=ULOC(M,6)-(ULOC(M,5)-ULOC(M,2))/LE(M)
4127  A7TOT(M)=A7OLD(M)+ULOC(M,4)-ULOC(M,1)
     *+.5*(ULOC(M,5)-ULOC(M,2))**2/LE(M)
     *+LE(M)*(2.*TA**2-TA*TB+2.*TB**2)/30.
4126  IF(ITERCHK.NE.2)  GO TO 5344
      IF(MSUOPTN.EQ.1)  GO TO 5344
      DO 5343 M=1,NUMEL
      ALFA1=ULOC(M,3)
      ALFA2=2.*(-3.*ULOC(M,2)-2.*ULOC(M,3)*LE(M)+3.*ULOC(M,5)-
     *ULOC(M,6)*LE(M))/LE(M)
      ALFA3=3.*(2.*ULOC(M,2)+ULOC(M,3)*LE(M)-2.*ULOC(M,5)+ULOC(M,6)
     **LE(M))/LE(M)
      BE(1)=(-ULOC(M,1)+ULOC(M,4))/LE(M)+ALFA1**2/2.
      BE(2)=ALFA1*ALFA2
      BE(3)=ALFA2**2/2.+ALFA1*ALFA3
      BE(4)=ALFA2*ALFA3
      BE(5)=ALFA3**2/2.
      DO 5343  I=1,5
5343  BTO(M,I)=BOL(M,I)+BE(I)
5344  CONTINUE
      WRITE(61,4115)
4115  FORMAT(/,10X,*DTOT(I)*,/)
      DO 4116 I=1,NEQ
4116  WRITE(61,4117)  DACTUAL(I)
4117  FORMAT(10X,F15.10)
      WRITE(61,399)PACTUAL(LODPON1),DACTUAL(LODPON1),PACTUAL(LODPON2)
     *,DACTUAL(LODPON2),PACTUAL(LODPON3),DACTUAL(LODPON3),
     *PACTUAL(LODPON4),DACTUAL(LODPON4),PACTUAL(LODPON5),DACTUAL(LODPON5
     *),PACTUAL(LODPON6),DACTUAL(LODPON6),DETRMNT,NUMITER
      IF(DETRMNT.LE.0..AND.DETOPTN.EQ.1)  GO TO 900
      IF(ABS(PACTUAL(LODPON1)).GT.ABS(PTOT1)) GO TO 900
      DO 5007 I=1,NEQ
      PSAVE(I)=PACTUAL(I)
      DTOT(I)=0.0
5007  CONTINUE
      DO 5342 M=1,NUMEL
      IF(MSUOPTN.EQ.1)  A7OLD(M)=A7TOT(M)
      DO 5342  I=1,5
      IF(MSUOPTN.EQ.2)  BOL(M,I)=BTO(M,I)
5342  CONTINUE
      R(LODPON1)=R(LODPON1)+PINC1
      R(LODPON2)=R(LODPON2)+PINC2
      R(LODPON3)=R(LODPON3)+PINC3
      IF(NEQ.GE.4)  R(LODPON4)=R(LODPON4)+PINC4
      IF(NEQ.GE.5)  R(LODPON5)=R(LODPON5)+PINC5
      IF(NEQ.GE.6)  R(LODPON6)=R(LODPON6)+PINC6
      DO  4104  I=1,NEQ
4104  PSTART(I)=R(I)
      IF(PRIOPTN.EQ.0) GO TO 6977
      WRITE(61,9735)
9735  FORMAT(///,10X,*R(I)*,/)
      DO 9736  IMM=1,NEQ
9736  WRITE(61,9737)  R(IMM)
9737  FORMAT(10X,E21.15)
6977  CONTINUE
      ICHECK=3
      GO TO 1001
900   CONTINUE
C
1     FORMAT(6I5)
10    FORMAT(I5)
11    FORMAT(E21.15)
399   FORMAT(//,10X,7HPLOAD1=,F15.9/10X,8HDEFLEC1=,F15.10/
     *10X,7HPLOAD2=,F15.9/10X,8HDEFLEC2=,F15.10/
     *10X,*PLOAD3=*,F15.9/10X,*DEFLEC3=*,F15.10/
     *10X,*PLOAD4=*,F15.9/10X,*DEFLEC4=*,F15.10/
     *10X,*PLOAD5=*,F15.9/10X,*DEFLEC5=*,F15.10/
     *10X,*PLOAD6=*,F15.9/10X,*DEFLEC6=*,F15.10/
     *10X,12HDETERMINANT=,E25.15/10X,11HITERATIONS=,I5)
699   FORMAT(5F10.6,I5)
799   FORMAT(10X,7HPINIT1=,F15.8,10X,6HPINC1=,F15.8/
     *10X,6HPTOT1=,F15.8,10X,7HPINIT2=,F15.8,10X,6HPINC2=,F15.8/
     +10X,6HPTOT2=,F15.8,10X,8HMAXITER=,I5)
1010  FORMAT(A10,A10,A10,5I5)
1015  FORMAT(I5,3F10.1)
2010  FORMAT(*1*,A10,A10,A10//7X,6HNE    =,I3//7X,6HNUMNP=,I3//7X,
     +//7X,6HICAL1=,I3//7X,6HICAL2=,
     +I3//7X,6HICAL3=,I3)
2015  FORMAT(*1*,15H  INITIAL LOADS//7H   NODE,10X,*LOAD DIRECTION*,//
     +7H NUMBER,13X,1HX,9X,1HY,9X,2HTZ//)
2020  FORMAT(I5,6X,3F10.6)
C
      END
C
C
C
C
C
C
C
      SUBROUTINE NODDATA(IGOPTIN)
C     ****************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(11,3),X(11),Y(11)
      WRITE(61,2010)
      WRITE(61,2015)
      IF(IGOPTIN.EQ.0) GO TO 100
      IF(IGOPTIN.EQ.2) GO TO 202
      READ(60,10)  ALFZERO,RADIUS
      WRITE(61,20) ALFZERO,RADIUS
      ALFINC=ALFZERO/NE
      DO 201 I=1,NUMNP
      X(I)=RADIUS*SIN((-ALFZERO/2)+(I-1)*ALFINC)
      Y(I)=SQRT(RADIUS**2-X(I)**2)
201   CONTINUE
      GO TO 204
202   READ(60,10)  RISE,SPAN
      WRITE(61,30) RISE,SPAN
      DO 203 I=1,NUMNP
      X(I)=-SPAN/2+(I-1)*SPAN/NE
      Y(I)=RISE-4.*RISE*X(I)**2/(SPAN**2)
203   CONTINUE
204   CONTINUE
90    READ(60,1001)N,(IA(N,I),I=1,3)
      WRITE(61,2020)N,(IA(N,I),I=1,3),X(N),Y(N)
      IF(N.NE.NUMNP)  GO TO 90
      GO TO 101
100   READ(60,1000) N,(IA(N,I),I=1,3),X(N),Y(N)
      WRITE(61,2020)N,(IA(N,I),I=1,3),X(N),Y(N)
      IF(N.NE.NUMNP) GO TO 100
101   NEQ=0
      DO 125 N=1,NUMNP
      DO 120 I=1,3
      IF(IA(N,I).NE.1) GO TO 105
      IA(N,I)=0
      GO TO 120
105   IA(N,I)=-1
      NEQ=NEQ+1
      IA(N,I)=NEQ
120   CONTINUE
125   CONTINUE
      WRITE(61,2030)
      WRITE(61,2040)
      WRITE(61,2050) (N,(IA(N,I),I=1,3),N=1,NUMNP)
      WRITE(61,2060) NEQ
      RETURN
10    FORMAT(2F15.10)
20    FORMAT(//,10X,5HALFA=,F15.10,10X,7HRADIUS=,F15.10,//)
30    FORMAT(//,10X,5HRISE=,F15.10,10X,5HSPAN=,F15.10,//)
1000  FORMAT(I5,3I3,2F10.5)
1001  FORMAT(I5,3I3)
2010  FORMAT(/,10X,*INPUT NODAL DATA*,/)
2015  FORMAT(7H   NODE,3X,36HNODAL POINT BOUNDARY CONDITION CODES,3X,
     +23HNODAL POINT COORDINATES/7H NUMBER,10X,7HIA(N,I)/
     +15X,1HX,4X,1HY,4X,2HTZ,20X,4HX(N),8X,4HY(N))
2020  FORMAT(I5,6X,3I5,14X,2F12.3)
2030  FORMAT(///,10X,*GENERATED NODAL DATA*)
2040  FORMAT(7H   NODE,7X,*EQUATION NUMBERS*,22X,
     +/*   NUMBER*,8X,*IA(N,I)*,33X,/
     +15X,1HX,4X,1HY,4X,2HTX,//)
2050  FORMAT(I5,6X,3I5)
```

```
2060  FORMAT(*-*,4HNEQ=,I3)
      END
C
C
C
C
C
C
C
      SUBROUTINE ELEMENT
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/3/IA(11,3),X(11),Y(11)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      REAL IXX,LE
      READ(60,1010)  NUMEL,E(1)
      WRITE(61,2021)
      WRITE(61,2020) NUMEL,E(1)
      K=0
      WRITE(61,2025)
2025  FORMAT(///5X,*ELEMENT*,3X,*NODEI(M)*,3X,*NODEJ(M)*,12X,
     +*A(M)*,10X,*IXX(M)*,3X,*SXX(M)*,/)
105   READ(60,1020)M,NODEI(M),NODEJ(M),A(M),IXX(M),SXX(M)
      WRITE(61,2022)M,NODEI(M),NODEJ(M),A(M),IXX(M),SXX(M)
      K=K+1
      L(1,K)=M
      IF(K.NE.NUMEL) GO TO 105
      RETURN
1010  FORMAT(I5,E10.2)
1020  FORMAT(3I5,3F10.6)
2020  FORMAT(6X,I6,2E17.6)
2021  FORMAT(///,10X,*NUMEL*,10X,*E(1)*,/)
2022  FORMAT(I6,5X,I5,6X,I5,6X,3F15.6)
      END
C
C
C
C
C
C
C
      SUBROUTINE BAND
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(11,3),X(11),Y(11)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      MBAND=0
      ICONTRL=0
      DO 900 M=1,NE
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 800 I=1,3
      IF(ICONTRL.EQ.1) GO TO 1001
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0) GO TO 199
1001  CONTINUE
      IF(IA(NI,I).LE.0) GO TO 800
      N1=IA(NI,I)
      GO TO 99
199   ICONTRL=1
      N1=0
99    DO 700 J=1,3
      IF(ICONTRL.EQ.1) GO TO 1002
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0) GO TO 399
1002  CONTINUE
      GO TO 499
399   ICONTRL=1
      MB=N1
      GO TO 299
499   IF(IA(NJ,J).LE.0) GO TO 700
      N2=IA(NJ,J)
      MB=IABS(N2-N1)
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0) GO TO 299
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0) GO TO 299
      MB=MB+1
299   IF(MB.GT.MBAND) MBAND=MB
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0) GO TO 800
700   CONTINUE
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0) GO TO 900
```

```
800   CONTINUE
900   CONTINUE
      WRITE(61,2000) MBAND
      RETURN
2000  FORMAT(/////,*SEMIBANDWIDTH MBAND=*,I3)
      END
C
C
C
C
C
C
C
      SUBROUTINE TRANSFM(M)
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/3/IA(11,3),X(11),Y(11)
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/12/ULOC(10,6),U(6,1),MSUOPTN
      REAL LE
      XI=X(NODEI(M))
      YI=Y(NODEI(M))
      XJ=X(NODEJ(M))
      YJ=Y(NODEJ(M))
      LE(M)=SQRT((XJ-XI)**2+(YJ-YI)**2)
      CX=(XJ-XI)/LE(M)
      CY=(YJ-YI)/LE(M)
      DO 501 I=1,6
      DO 501 J=1,6
501   ROT(I,J)=0.0
      ROT(1,1)=ROT(2,2)=ROT(4,4)=ROT(5,5)=CX
      ROT(1,2)=ROT(4,5)=CY
      ROT(2,1)=ROT(5,4)=-CY
      ROT(3,3)=ROT(6,6)=1.
      DO 502  I=1,6
      DO 502  J=1,6
502   ROTRAN(I,J)=ROT(J,I)
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE INVTRNS
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/3/IA(11,3),X(11),Y(11)
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/11/W(11,3),WTOT(11,3),WCHK(11,3)
      COMMON/12/ULOC(10,6),U(6,1),MSUOPTN
      DIMENSION UTEM(6,1)
      REAL LE
C        STORE DISPLACEMENTS OF EACH MEMBER IN U(6,1) ARRAY
C     ******************************************************************
      DO 100 M=1,NE
      DO 210 I=1,6
      NI=NODEI(M)
      NJ=NODEJ(M)
      IF(I.LE.3) GO TO 250
      IF(I.GT.3) GO TO 300
250   NP=NI
      GO TO 350
300   NP=NJ
      GO TO 400
350   U(I,1)=W(NP,I)
      GO TO 210
400   U(I,1)=W(NP,I-3)
210   CONTINUE
      CALL TRANSFM(M)
      CALL MULT(6,6,1,ROT,U,UTEM)
      DO 1001 I=1,6
1001  ULOC(M,I)=UTEM(I,1)
100   CONTINUE
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE BEAM
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/3/IA(11,3),X(11),Y(11)
      COMMON/2/NEQ,MBAND
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/8/PI(11,3),R(35)
      COMMON/9/S(35,12),SP(35,12),IDET
      COMMON/10/D(35)
      DIMENSION SEROT(6,6)
      REAL IXX,LE
      K=0
105   CONTINUE
      K=K+1
      M=L(1,K)
      NI=NODEI(M)
      NJ=NODEJ(M)
      AA=(X(NJ)-X(NI))**2
      B=(Y(NJ)-Y(NI))**2
      LE(M)=SQRT(AA+B)
      C=(X(NJ)-X(NI))/LE(M)
      Z=(Y(NJ)-Y(NI))/LE(M)
      SE(1,1)=SE(4,4)=E(1)*A(M)*C**2/LE(M)+12.*E(1)*IXX(M)*Z**2/LE(M)**3
      SE(1,4)=-SE(1,1)
      SE(1,2)=SE(4,5)=(E(1)*A(M)/LE(M)-12.*E(1)*IXX(M)/LE(M)**3)*C*Z
      SE(2,4)=SE(1,5)=-SE(1,2)
      SE(2,2)=SE(5,5)=E(1)*A(M)*Z**2/LE(M)+12.*E(1)*IXX(M)*C**2/LE(M)**3
      SE(2,5)=-SE(2,2)
      SE(4,6)=SE(3,4)=6.*E(1)*IXX(M)*Z/LE(M)**2
      SE(1,6)=SE(1,3)=-SE(3,4)
      SE(2,6)=SE(2,3)=6.*E(1)*IXX(M)*C/LE(M)**2
      SE(3,5)=SE(5,6)=-SE(2,3)
      SE(6,6)=SE(3,3)=4.*E(1)*IXX(M)/LE(M)
      SE(3,6)=SE(6,6)/2.
      DO 210 I=1,6
      DO 210 J=I,6
210   SE(J,I)=SE(I,J)
      IF(ICAL2.EQ.1) GO TO 9024
      WRITE(61,9025) M
9025  FORMAT(/,10X,*M=*,I2,10X,*SE(I,J),LINEAR*,/)
      WRITE(61,503) ((SE(I,J),J=1,6),I=1,6)
9024  CONTINUE
      IF(ISTRESS.EQ.0)  GO TO 601
      WRITE(8,10) ((SE(I,J),J=1,6),I=1,6)
601   CONTINUE
      CALL ASEMBLE(M)
      IF(ISTRESS.EQ.0)  GO TO 602
      CALL TRANSFM(M)
      CALL MULT(6,6,6,SE,ROTRAN,SEROT)
      CALL MULT(6,6,6,ROT,SEROT,SE)
      WRITE(7,10) ((SE(I,J),J=1,6),I=1,6)
602   CONTINUE
      IF(K.NE.NUMEL) GO TO 105
      IF(ISTRESS.EQ.1)  REWIND 8
      IF(ISTRESS.EQ.1)  REWIND 7
      WRITE(4,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      REWIND 4
      IF(ICAL2.EQ.1) GO TO 9026
      WRITE(61,9027)
9027  FORMAT(/,10X,*S(I,J) LINEAR*,/)
      WRITE(61,503) ((S(I,J),J=1,MBAND),I=1,NEQ)
9026  CONTINUE
      RETURN
10    FORMAT(E21.15)
503   FORMAT(/1X,6F20.10)
      END
C
C
C
C
C
```

```
C
C
      SUBROUTINE SBEAME1
      ******************************************************************
C
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/9/S(35,12),SP(35,12),IDET
      COMMON/12/ULOC(10,6),U(6,1),MSUOPTN
      DIMENSION SEROT(6,6)
      REAL LE
      DO 220 M=1,NUMEL
      DO 104 I=1,6
      DO 104 J=I,6
  104 SE(I,J)=0.0
      SE(4,5)=SE(1,2)=(-(ULOC(M,3)+ULOC(M,6))/10.+6.*(ULOC(M,5)-
     +ULOC(M,2))/(5.*LE(M)))*E(1)*A(M)/LE(M)
      SE(2,4)=SE(1,5)=-SE(1,2)
      SE(1,3)=((ULOC(M,6)-4.*ULOC(M,3)+3.*(ULOC(M,5)-ULOC(M,2))/
     +LE(M))/30.)*E(1)*A(M)
      SE(1,6)=((ULOC(M,3)-4.*ULOC(M,6)+3.*(ULOC(M,5)-ULOC(M,2))/
     +LE(M))/30.)*E(1)*A(M)
      SE(4,6)=-SE(1,6)
      SE(3,4)=-SE(1,3)
      SE(5,5)=SE(2,2)=6.*(ULOC(M,4)-ULOC(M,1))*E(1)*A(M)/(5.*LE(M)**2)
      SE(2,5)=-SE(2,2)
      SE(2,3)=SE(2,6)=(ULOC(M,4)-ULOC(M,1))*E(1)*A(M)/(10.*LE(M))
      SE(3,5)=SE(5,6)=-SE(2,3)
      SE(3,3)=SE(6,6)=2.*(ULOC(M,4)-ULOC(M,1))*E(1)*A(M)/15.
      SE(3,6)=-(ULOC(M,4)-ULOC(M,1))*E(1)*A(M)/30.
      DO 106 I=1,6
      DO 106 J=1,I
  106 SE(I,J)=SE(J,I)
      IF(ISTRESS.EQ.0)  GO TO 601
      WRITE(1,10) ((SE(I,J),J=1,6),I=1,6)
  601 CONTINUE
      CALL TRANSFM(M)
      CALL MULT(6,6,6,SE,ROT,SEROT)
      CALL MULT(6,6,6,ROTRAN,SEROT,SE)
      IF(ISTRESS.EQ.0)  GO TO 602
      WRITE(2,10) ((SE(I,J),J=1,6),I=1,6)
  602 CONTINUE
      CALL ASEMBLE(M)
  220 CONTINUE
      WRITE(6,10)((S(I,J),J=1,MBAND),I=1,NEQ)
      IF(ISTRESS.EQ.1)  REWIND 1
      IF(ISTRESS.EQ.1)  REWIND 2
      REWIND 6
      RETURN
   10 FORMAT (E21.15)
      END
C
C
C
C
C
C
C
      SUBROUTINE KEPSI01
      ******************************************************************
C
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/9/S(35,12),SP(35,12),IDET
      COMMON/12/ULOC(10,6),U(6,1),MSUOPTN
      COMMON/17/A7TOT(10),A7OLD(10),BOL(10,5),BTO(10,5),BE(5)
      DIMENSION SEROT(6,6)
      REAL LE
      DO 220 M=1,NUMEL
      DO 104 I=1,6
      DO 104 J=1,I
  104 SE(I,J)=0.0
      IF(MSUOPTN.EQ.1)  GO TO 99
      R01=BTO(M,1)+BTO(M,2)/2.+BTO(M,3)/3.+BTO(M,4)/4.+BTO(M,5)/5.
      R02=BTO(M,1)/2.+BTO(M,2)/3.+BTO(M,3)/4.+BTO(M,4)/5.+BTO(M,5)/6.
      R03=BTO(M,1)/3.+BTO(M,2)/4.+BTO(M,3)/5.+BTO(M,4)/6.+BTO(M,5)/7.
      R04=BTO(M,1)/4.+BTO(M,2)/5.+BTO(M,3)/6.+BTO(M,4)/6.+BTO(M,5)/8.
      R05=BTO(M,1)/5.+BTO(M,2)/6.+BTO(M,3)/7.+BTO(M,4)/8.+BTO(M,5)/9.
      SE(2,2)=SE(5,5)=(R03-2.*R04+R05)*36.*A(M)*E(1)/LE(M)
      SE(5,2)=-SE(2,2)
      SE(5,3)=(R02-5.*R03+7.*R04-3.*R05)*6.*E(1)*A(M)
      SE(3,2)=-SE(5,3)
      SE(3,3)=SE(2,2)=(R01-8.*R02+22.*R03-24.*R04+9.*R05)*
     +E(1)*A(M)*LE(M)
      SE(6,6)=(4.*R03-12.*R04+9.*R05)*E(1)*A(M)*LE(M)
      SE(6,2)=(2.*R03-5.*R04+3.*R05)*6.*E(1)*A(M)
      SE(6,5)=-SE(6,2)
      SE(6,3)=(-2.*R02+11.*R03-18.*R04+9.*R05)*E(1)*A(M)*LE(M)
99    IF(MSUOPTN.EQ.2)  GO TO 199
      SE(2,2)=SE(5,5)=6./(5.*LE(M))
      SE(3,2)=SE(6,2)=1/10.
      SE(5,2)=-SE(2,2)
      SE(5,3)=SE(6,5)=-1/10.
      SE(3,3)=SE(6,6)=2.*LE(M)/15.
      SE(6,3)=-LE(M)/30.
      DO 1041 I=1,6
      DO 1041 J=1,I
1041  SE(I,J)=SE(I,J)*E(1)*A(M)*A7TOT(M)/LE(M)
199   CONTINUE
      DO 106 I=1,6
      DO 106 J=I,6
106   SE(I,J)=SE(J,I)
      IF(ISTRESS.EQ.0)  GO TO 601
      WRITE(10,10) ((SE(I,J),J=1,6),I=1,6)
601   CONTINUE
      CALL TRANSFM(M)
      CALL MULT(6,6,6,SE,ROT,SEROT)
      CALL MULT(6,6,6,ROTRAN,SEROT,SE)
      IF(ISTRESS.EQ.0)  GO TO 602
      WRITE(11,10) ((SE(I,J),J=1,6),I=1,6)
602   CONTINUE
      CALL ASEMBLE(M)
220   CONTINUE
      WRITE(12,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      IF(ISTRESS.EQ.1)  REWIND 10
      IF(ISTRESS.EQ.1)  REWIND 11
      REWIND 12
      RETURN
10    FORMAT(E21.15)
      END
C
C
C
C
C
C
C
      SUBROUTINE SBEAME2
C     ****************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/9/S(35,12),SP(35,12),IDET
      COMMON/12/ULOC(10,6),U(6,1),MSUOPTN
      DIMENSION SEROT(6,6)
      REAL LE
      DO 220 M=1,NUMEL
      DO 104  I=1,6
      DO 104  J=I,6
104   SE(I,J)=0.0
      IF(MSUOPTN.EQ.1) GO TO 399
      TA=ULOC(M,3)
      TB=ULOC(M,6)
      VA=ULOC(M,2)
      VB=ULOC(M,5)
      SE(3,3)=(12.*LE(M)*TA**2+LE(M)*TB**2-3.*LE(M)*TA*TB+
     +18.*(VB-VA)**2/LE(M)
     ++3.*(VB-VA)*(TA-TB))/(140.)*E(1)*A(M)
      SE(2,3)=(-3.*TA**2+3.*TB**2+6.*TA*TB+108.*(VB-VA)**2/
     +(LE(M)**2)-72.*TA*(VB-VA)/LE(M))*E(1)*A(M)/280.
      SE(3,5)=-SE(2,3)
      SE(3,6)=(-3.*LE(M)*TA**2-3.*LE(M)*TB**2+4.*LE(M)*TA*TB
     +-6.*(VB-VA)*(TA+TB))/(280.)*E(1)*A(M)
      SE(6,6)=(LE(M)*TA**2+12.*LE(M)*TB**2-3.*LE(M)*TA*TB
     ++18.*(VB-VA)**2/LE(M)
     ++3.*(VB-VA)*(TB-TA))/(140.)*E(1)*A(M)
```

```
      SE(2,6)=(3.*TA**2-3.*TB**2+6.*TA*TB+108.*(VB-VA)**2/LE(M)**2
     +-72.*TB*(VB-VA)/LE(M))/(280.)*E(1)*A(M)
      SE(5,6)=-SE(2,6)
      SE(5,5)=SE(2,2)=(18.*TA**2/LE(M)+18.*TB**2/LE(M)+432.*(VB-VA)**2
     +/LE(M)**3-108.*(VB-VA)*(TA+TB)/LE(M)**2)/(140.)*E(1)*A(M)
      SE(2,5)=-SE(2,2)
      GO TO 199
399    RHO=(ULOC(M,5)-ULOC(M,2))/LE(M)
      SE(2,2)=SE(5,5)=((9.*ULOC(M,3)**2+9.*ULOC(M,6)**2-2.
     +*ULOC(M,3)*ULOC(M,6)-36.*ULOC(M,3)*RHO-36.*ULOC(M,6)
     +*RHO+216.*RHO**2)/100.)*E(1)*A(M)/LE(M)
      SE(2,5)=-SE(2,2)
      SE(2,3)=(6.*ULOC(M,3)**2+ULOC(M,6)**2+2.*ULOC(M,3)*ULOC(M,6)-54.
     +*ULOC(M,3)*RHO+6.*ULOC(M,6)*RHO+54.*RHO**2)*E(1)*A(M)/300.
      SE(3,5)=-SE(2,3)
      SE(2,6)=(6.*ULOC(M,6)**2+ULOC(M,3)**2+2.*ULOC(M,3)*ULOC(M,6)-54.
     +*ULOC(M,6)*RHO+6.*ULOC(M,3)*RHO+54.*RHO**2)*E(1)*A(M)/300.
      SE(5,6)=-SE(2,6)
      SE(3,3)=(8.*ULOC(M,3)**2+3.*ULOC(M,6)**2-4.*ULOC(M,3)*ULOC(M,6)-12
     +.*ULOC(M,3)*RHO-2.*ULOC(M,6)*RHO+27.*RHO**2)*E(1)*A(M)*LE(M)/300.
      SE(3,6)=(-2.*ULOC(M,3)**2-2.*ULOC(M,6)**2+6.*ULOC(M,3)*ULOC(M,6)-2
     +.*ULOC(M,3)*RHO-2.*ULOC(M,6)*RHO-3.*RHO**2)*E(1)*A(M)*LE(M)/300.
      SE(6,6)=(8.*ULOC(M,6)**2+3.*ULOC(M,3)**2-4.*ULOC(M,3)*ULOC(M,6)-12
     +.*ULOC(M,6)*RHO-2.*ULOC(M,3)*RHO+27.*RHO**2)*E(1)*A(M)*LE(M)/300.
199   DO 106 I=1,6
      DO 106 J=1,I
106   SE(I,J)=SE(J,I)
      IF(ISTRESS.EQ.0)  GO TO 601
      WRITE(3,10) ((SE(I,J),J=1,6),I=1,6)
601   CONTINUE
      CALL TRANSFM(M)
      CALL MULT(6,6,6,SE,ROT,SEROT)
      CALL MULT(6,6,6,ROTRAN,SEROT,SE)
      IF(ISTRESS.EQ.0)  GO TO 602
      WRITE(5,10) ((SE(I,J),J=1,6),I=1,6)
602   CONTINUE
      CALL ASEMBLE(M)
220   CONTINUE
      WRITE(9,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      IF(ISTRESS.EQ.1)  REWIND 3
      IF(ISTRESS.EQ.1)  REWIND 5
       REWIND 9
      RETURN
10    FORMAT(E21.15)
      END
C
C
C
C
C
C
C
      SUBROUTINE ASEMBLE(M)
C     ***************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(11,3),X(11),Y(11)
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/8/PI(11,3),P(35)
      COMMON/9/S(35,12),SP(35,12),IDET
      IF(IPAR.NE.1) GO TO 90
      DO 5 I=1,NEQ
5     R(I)=0.0
      DO 80 N=1,NUMNP
      DO 70 I=1,3
      IF(IA(N,I)) 70,70,10
10    II=IA(N,I)
      R(II)=PI(N,I)
70    CONTINUE
80    CONTINUE
      RETURN
90    NI=NODEI(M)
      NJ=NODEJ(M)
      DO 165 K1=1,2
      IF(K1.EQ.1) NP=NI
      IF(K1.EQ.2) NP=NJ
      DO 160 I=1,3
      IF(IA(NP,I)) 115,160,100
100   II=IA(NP,I)
```

```
115   CONTINUE
      DO 155 K2=1,2
      IF(K2.EQ.1) ND=NI
      IF(K2.EQ.2) ND=NJ
      DO 150 J=1,3
      IF(IA(ND,J)) 145,150,120
120   JJ=IA(ND,J)
145   CONTINUE
      IF(JJ.LT.II) GO TO 150
      IF(K1.EQ.1) IE=I
      IF(K1.EQ.2) IE=I+3
      IF(K2.EQ.1) JE=J
      IF(K2.EQ.2) JE=J+3
      JJ=JJ-II+1
      S(II,JJ)=S(II,JJ)+SE(IE,JE)
      IF(JJ.GT.MBAND) MBAND=JJ
150   CONTINUE
155   CONTINUE
160   CONTINUE
165   CONTINUE
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE LINSOLN
C     ***************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/8/PI(11,3),R(35)
      COMMON/9/S(35,12),SP(35,12),IDET
      COMMON/10/D(35)
      DO 110 I=1,NEQ
110   D(I)=R(I)
      IF(ICAL1.EQ.0) WRITE(61,2020)
      IF(ICAL1.EQ.0) WRITE(61,2010) (I,D(I),I=1,NEQ)
      DO 790 N=1,NEQ
      DO 780 L=2,MBAND
      IF(S(N,L).EQ.0.) GO TO 780
      I=N+L-1
      C=S(N,L)/S(N,1)
      J=0
      DO 750 K=L,MBAND
      J=J+1
750   S(I,J)=S(I,J)-C*S(N,K)
      S(N,L)=C
780   CONTINUE
790   CONTINUE
      DO 830 N=1,NEQ
      DO 820 L=2,MBAND
      IF(S(N,L).EQ.0.) GO TO 820
      I=N+L-1
      D(I)=D(I)-S(N,L)*D(N)
820   CONTINUE
830   D(N)=D(N)/S(N,1)
      DO 860 M=2,NEQ
      N=NEQ+1-M
      DO 850 L=2,MBAND
      IF(S(N,L).EQ.0.) GO TO 850
      K=N+L-1
      D(N)=D(N)-S(N,L)*D(K)
850   CONTINUE
860   CONTINUE
      IF(ICAL3.EQ.0) WRITE(61,2000)
      IF(ICAL3.EQ.0) WRITE(61,2010) (I,D(I),I=1,NEQ)
      RETURN
2000  FORMAT(/,10X,*DISPLACEMENT FROM LINEAR SOLUTION*,/)
2010  FORMAT(/,10X,*D(*,I3,*)=*,E21.15)
2020  FORMAT(/,10X,*LOAD VECTOR FOR LINEAR SOLUTION*,/)
      END
C
C
C
C
C
C
```

```
C
      SUBROUTINE IDENT
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(11,3),X(11),Y(11)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/10/D(35)
      COMMON/11/W(11,3),WTOT(11,3),WCHK(11,3)
      DO 230 K=1,NUMEL
      M=L(1,K)
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 230 K1=1,2
      IF(K1.EQ.1) NP=NI
      IF(K1.EQ.2) NP=NJ
      DO 220 I=1,3
      IF(IA(NP,I)) 220,155,150
150   NL=IA(NP,I)
      W(NP,I)=D(NL)
      GO TO 220
155   W(NP,I)=0.0
220   CONTINUE
230   CONTINUE
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE MULT(M,K,N,A,B,C)
C     ******************************************************************
C
      DIMENSION A(M,K),B(K,N),C(M,N)
      DO 100 I=1,M
      DO 100 J=1,N
      C(I,J)=0.0
      DO 100 MM=1,K
100   C(I,J)=C(I,J)+A(I,MM)*B(MM,J)
      RETURN
      END
C
C
C
C
C
C
C
      FUNCTION DET1(SCALE)
C     ******************************************************************
C
      COMMON/2/NEQ,MBAND
      COMMON/9/S(35,12),SP(35,12),IDET
      IF(IDET.EQ.1) GO TO 250
      DO 490 I=1,NEQ
      DO 490 J=1,MBAND
490   S(I,J)=SP(I,J)
      DO 390 LN=1,NEQ
      DO 380 LL=2,MBAND
      IF(S(LN,LL).EQ.0.) GO TO 380
      I=LN+LL-1
      C=S(LN,LL)/S(LN,1)
      J=0
      DO 350 KK=LL,MBAND
      J=J+1
350   S(I,J)=S(I,J)-C*S(LN,KK)
      S(LN,LL)=C
380   CONTINUE
390   CONTINUE
250   CONTINUE
      DT=1.
      DO 400 I=1,NEQ
      DT=DT*S(I,1)/SCALE
400   CONTINUE
      DET1=DT
      RETURN
      END
```

```
C
C
C
C
C
C
C
      SUBROUTINE STRESS(M)
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(10),NUMEL,IPAR,ICAL1,ICAL2,ICAL3,ISTRESS
      COMMON/4/SE(6,6),ROT(6,6),ROTRAN(6,6),SE1(6,6),SE2(6,6)
      COMMON/5/E(1),NODEI(10),NODEJ(10),A(10),IXX(10),L(1,10),SXX(10)
      COMMON/12/ULOC(10,6),U(6,1),MSUOPTN
      DIMENSION SIGMA(10),STRAIN(10),ENDFORC(6,1),UTEMP(6,1)
      REAL LE
C        TO HAVE END FORCES
C     ******************************************************************
      DO 100 I=1,6
100   UTEMP(I,1)=ULOC(M,I)
      CALL MULT(6,6,1,SE,UTEMP,ENDFORC)
      IF(ABS(ENDFORC(1,1)).GT.ABS(ENDFORC(4,1)))RPMAX=ABS(ENDFORC(1,1))
      IF(ABS(ENDFORC(4,1)).GE.ABS(ENDFORC(1,1)))RPMAX=ABS(ENDFORC(4,1))
      IF(ABS(ENDFORC(3,1)).GT.ABS(ENDFORC(6,1)))RMMAX=ABS(ENDFORC(3,1))
      IF(ABS(ENDFORC(6,1)).GE.ABS(ENDFORC(3,1)))RMMAX=ABS(ENDFORC(6,1))
      SIGMA(M)=RPMAX/A(M)+RMMAX/SXX(M)
      STRAIN(M)=SIGMA(M)/E(1)
      WRITE(61,200) M,STRAIN(M)
200   FORMAT(10X,/,10X,*FOR ELEMENT NUMBER*,3X,I2,3X,*WE HAVE*,/
     +10X,*STRAIN=*,E21.15)
      DO 300 I=1,6
300   WRITE(61,400) I,ENDFORC(I,1)
400   FORMAT(10X,*END FORCE  *,I2,*=*,2X,F15.10)
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE NLEIGNP(SCALE)
C     ******************************************************************
C
      EXTERNAL  DET
      REAL L
      READ(60,1000)   XTOL,FTOL,NTOL,DINCR
      WRITE(61,2010)  XTOL,FTOL,NTOL,DINCR
      WRITE(61,2030)
      A=0.
100   FA=DET(A,SCALE)
      WRITE(61,2020)  A,FA
      IF(FA.LT.0.)  GO TO 110
      A=A+DINCR
      GO TO 100
110   CONTINUE
      B=A
      A=A-DINCR
      CALL MRGFLS(DET,A,B,XTOL,FTOL,NTOL,IFLAG,SCALE)
      IF(IFLAG.GT.2)  GO TO 500
      L=(A+B)/2.
      ERROR=ABS(B-A)/2.
      FL=DET(L,SCALE)
      WRITE(61,2000)  L,ERROR,FL
500   CONTINUE
      RETURN
1000  FORMAT(2F10.7,I10,F10.7)
2000  FORMAT(////,*     THE ROOT IS *,E21.15,10X,*PLUS/MINUS*,
     +E21.15//,*   DETERMINANT=*,E21.15)
2010  FORMAT(*1*,*      QUADRATIC EIGENVALUE PROBLEM*,//,*  XTOL=*,
     +F10.7,///,*     FTOL=*,F10.7,///,*     NTOL=*,I3///,*   DINCR=*,
     +F10.7,/)
2020  FORMAT(*-*,E21.15,5X,E21.15,//)
2030  FORMAT(////,13X,*LAMBDA*,17X,*DETERMINANT*,//)
      END
C
C
C
C
C
```

```
C
C
      SUBROUTINE MRGFLS(F,A,B,XTOL,FTOL,NTOL,IFLAG,SCALE)
C     ******************************************************************
C
      IFLAG=0
      FA=F(A,SCALE)
      SIGNFA=FA/ABS(FA)
      FB=F(B,SCALE)
      IF(SIGNFA*FB.LE.0.)  GO TO 100
      IFLAG=3
      WRITE(61,2010)  A,B
      RETURN
100   W=A
      FW=FA
      DO 400 N=1,NTOL
      IF(ABS(B-A)/2..LE.XTOL)   RETURN
      IF(ABS(FW).GT.FTOL)  GO TO 200
      A=W
      B=W
      IFLAG=1
      RETURN
200   W=(FA*B-FB*A)/(FA-FB)
      PREVFW=FW/ABS(FW)
      FW=F(W,SCALE)
      NM1=N-1
      WRITE(61,2020)  NM1,A,W,B,FA,FW,FB
      IF(SIGNFA*FW.LT.0.)  GO TO 300
      A=W
      FA=FW
      IF(FW*PREVFW.GT.0.)  FB=FB/2.
      GO TO 400
300   B=W
      FB=FW
      IF(FW*PREVFW.GT.0.)  FA=FA/2.
400   CONTINUE
      IFLAG=2
      WRITE(61,2030)  NTOL
      RETURN
2010  FORMAT(////,*    F(X) IS OF SAME SIGN AT THE TWO ENDPOINTS*
     +,2E21.15)
2020  FORMAT(*-*,I3,*     L-VALUES*,3E21.15//,4X,*   F-VALUES*
     +,3E21.15,//)
2030  FORMAT(////,*    NO CONVERGENCE IN*,I5,*ITERATIONS*,/)
      END
C
C
C
C
C
C
C
      FUNCTION DET(L,SCALE)
C     ******************************************************************
C
      COMMON/2/NEQ,MBAND
      COMMON/3/S(35,12),SP(35,12),IDET
      REAL K,N1,N2,L
      IF(L.EQ.0.)  GO TO 220
      DO 210  I=1,NEQ
      DO 200  J=1,MBAND
      READ(4,10)  K
      READ(6,10)  N1
      READ(9,10)  N2
      S(I,J)=K+L*N1+L*L*N2
200   CONTINUE
210   CONTINUE
      GO TO 230
220   READ(4,10)  ((S(I,J),J=1,MBAND),I=1,NEQ)
230   REWIND  4
      REWIND  6
      REWIND  9
      DO 390  LN=1,NEQ
      DO 380  LL=2,MBAND
      IF(S(LN,LL).EQ.0.)  GO TO 380
      I=LN+LL-1
      C=S(LN,LL)/S(LN,1)
      J=0
      DO 350  KK=LL,MBAND
      J=J+1
350   S(I,J)=S(I,J)-C*S(LN,KK)
```

```
      S(LN,LL)=C
380   CONTINUE
390   CONTINUE
      DT=1.
      DO 400 I=1,NEQ
      DT=DT*S(I,1)/SCALE
400   CONTINUE
      DET=DT
      RETURN
10    FORMAT(E21.15)
      END
```

```
 08:41:25 07/13/81    SL09017     1956 LINES PRINT.
  27 PAGES PRINT.  COST AT RG3 IS   $    2.59
```

## D.5 PROGRAM NFRAE2D

```
      PROGRAM NFRAE2D(INPUT,OUTPUT=65,TAPE60=INPUT,TAPE61=OUTPUT,
     +TAPE1,TAPE2,TAPE3,TAPE4)
C
C     **************************************************************
C        THIS PROGRAM ANALYSIS TWO DIMENSIONAL FRAMED
C        STRUCTURES USING EULERIAN COORDINATES.THE FOLLOWING
C        METHODS MAY BE SPECIFIED: BEAM-COLUMN,JENNINGS,
C        AND POWELL,S WITH THE OPTION OF NEWTON-RAPHSON,
C        ONE-STEP NEWTON-RAPHSON AND STRAIGHT INCREMENTAL.
C     **************************************************************
C
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(21,3),X(21),Y(21)
      COMMON/4/SE(6,6),STSTAR(3,3),STSTARM(3,3,20),TSMAL(3,3,20),
     +TTSML(3,3)
      COMMON/5/E(1),NODEI(20),NODEJ(20),A(20),IXX(20),L(1,20),SXX(20)
      COMMON/8/PI(21,3),PII(21,3),R(65)
      COMMON/9/S(65,12),IDET,ES(3,20)
      COMMON/10/U(65),TETC(20),ITERCHK,PROTYPE
      COMMON/11/V(21,3),DN(6,1),WTOT(21,3)
      COMMON/13/PLOAD1,PINIT1,PLOAD2,PINIT2,LODPON1,LODPON2,LODPON3
      DIMENSION USTAR(3,20),ESS(3,1),DTOT(65),COLD(20),ZOLD(20)
      DIMENSION DELOLD(20),DELU(3,1),DELS(3,1)
      DIMENSION T1(3,6),T1TRAN(6,3),RK(3,3),N1(3,3),N2(3,3),TBTRAN(3,6)
      DIMENSION B(6,3),BTRAN(3,6)
      DIMENSION SECSTAR(3,3),T21(6,6),T22(6,6)
      DIMENSION PART1(6,6),PART2(6,6),STT1(3,6),UNBLANC(21,3)
      DIMENSION GS1(6,6),GS2(6,6),PART21(6,6),PART22(6,6),PART23(6,6)
      INTEGER PROTYPE,DETOPTN
C     **************************************************************
C        TAPE1  FOR ELEMENT LINEAR STIFFNESS MATRIX IN GLOBAL
C        TAPE2  FOR STRUCTURAL LINEAR STIFFNESS MATRIX IN GLOBAL
C        TAPE4  FOR ELEMENT NONLINEAR STIFFNESS MATRIX IN GLOBAL
C        TAPE4  FOR STRUCTURAL NONLINEAR STIFFNESS MATRIX IN GLOBAL
C     **************************************************************
      REAL IXX,LE,N1,N2,LAMBDA,KO
      READ(60,21) PINIT1,PINC1,PMAX1,LODPON1
      READ(60,21) PINIT2,PINC2,PMAX2,LODPON2
      READ(60,21) PINIT3,PINC3,PMAX3,LODPON3
      READ(60,21) PINIT4,PINC4,PMAX4,LODPON4
      READ(60,21) PINIT5,PINC5,PMAX5,LODPON5
      READ(60,21) PINIT6,PINC6,PMAX6,LODPON6
21    FORMAT(3F15.8,I5)
C     **************************************************************
C        LODPON1 UP TO LODPON6 ARE THE ORDER OF D.O.F(IN LINEAR
C        SYSTEM OF EQUATIONS)RELATED TO EXTERNAL CONCENTRATED
C        LOADS OR MOMENTS APPLIED ON THE STRUCTURE.
C        PINIT1 UP TO PINIT6,PINC1 UP TO PINC6 AND PTOT1 UP TO
C        PTOT6 ARE THE INITIAL,INCREMENTAL AND MAXIMUM DEFINED
C        EXTERNAL LOAD COMPONENTS
C     **************************************************************
      READ(60,20)    IGOPTIN,DETOPTN
20    FORMAT(2I5)
      WRITE(61,30)   IGOPTIN,DETOPTN
30    FORMAT(/,10X,*IGOPTIN=*,I2,10X,*DETOPTN=*,I2)
C        IGOPTIN=1   FOR CIRCULAR ARCH,IGOPTIN=2  FOR PARABOLIC
C        ARCH AND IGOPTIN=0  FOR OTHER GEOMETRIES.
C        DETOPTN=0 NO CONTROL ON THE DETERMINANT OF TANGENT STIFFNESS
C        MATRIX
C        DETOPTN=1 EXECUTION WOULD BE TERMINATED IF DETERMINANT
C        OF TANGENT STIFFNESS MATRIX IS NEGATIVE
C     **************************************************************
      READ(60,22) EPSI1,EPSI2,MAX1,MAX2,N2OPTIN,ITERCHK,PROTYPE,ISTRESS
22    FORMAT(2F15.8,6I5)
C        FOR EULER FEM FORMULATION NO NEED TO HAVE EPSI2
C        ,MAX2(SET THEM EQUAL TO ZERO)
C     **************************************************************
      WRITE(61,2311) EPSI1,EPSI2,MAX1,MAX2
2311  FORMAT(/,10X,*EPSI1=*,F10.6/
     +,10X,*EPSI2=*,F10.6/
     +10X,*MAX1=*,I3/
     +,10X,*MAX2=*,I3/)
C     **************************************************************
C        EPSI1=ALLOWABLE TOLERANCE FOR VARIATION OF
C        DISPLACEMENT VECTOR
C        EPSI2=ALLOWABLE TOLERANCE IN ITERATIVE PROCESS
C        FOR AXIAL LOAD
C        MAX1=MAXIMUM NUMBER OF ITERATIONS FOR CONVERGENCE
C        ON DISPLACEMENT VECTOR
C        MAX2=MAXIMUM NUMBER OF ITERATIONS FOR MEMBER
```

```
C           AXIAL LOAD
C           PROTYPE=1 FOR EULER FINITE ELEMENT FORMULATION
C           PROTYPE=2 FOR BEAM COLUMN FORMULATION
C           N2OPTIN=1  FOR JENNINGS, FORMULATION
C           N2OPTIN=2  FOR POWELL,S FORMULATION
C           IF  ISTRESS=1  STRESS SHOULD BE EVALUATED
C           IF  ISTRESS=0  STRESS SHOULDN,T BE EVALUATED
C           ITERCHK=0  FOR STRAIGHT INCREMENTAL METHOD
C           ITERCHK=1  FOR ONE STEP NEWTON RAPHSON METHOD
C           ITERCHK=2  FOR NEWTON RAPHSON METHOD
C           ICHKOPT=1  FOR CONVERGENCE CHECK ON UNBALANCED FORCE
C           ICHKOPT=2  FOR CONVERGENCE CHECK ON DISPLACEMENT
C           EPSI3,EPSI4 ALLOWABLE TOLERANCE FOR UNBALANCED FORCE OR
C           MOMENT AND WE DON,T NEED THEM IF ICHKOPT=2(SET THEM EQUAL TO
C           ZERO)
C        ****************************************************************
         IF(ITERCHK.EQ.2) READ(60,4375)  ICHKOPT,EPSI3,EPSI4
4375     FORMAT(I5,2F15.10)
         IF(ITERCHK.EQ.2) WRITE(61,4376)  ICHKOPT,EPSI3,EPSI4
4376     FORMAT(/,10X,*ICHKOPT=*,I5,10X,*EPSI3=*,E21.15,10X,
        +*EPSI4=*,E21.15)
         WRITE(61,2222) N2OPTIN,ITERCHK,PROTYPE,ISTRESS
2222     FORMAT(10X,*N2OPTIN=*,I2,10X,*ITERCHK=*,I2,/
        +,10X,*PROTYPE=*,I2,10X,*ISTRESS=*,I2,/)
         WRITE(61,23)     PINIT1,PINC1,PMAX1,LODPON1
         WRITE(61,231)    PINIT2,PINC2,PMAX2,LODPON2
         WRITE(61,2319)   PINIT3,PINC3,PMAX3,LODPON3
         WRITE(61,2320)   PINIT4,PINC4,PMAX4,LODPON4
         WRITE(61,2321)   PINIT5,PINC5,PMAX5,LODPON5
         WRITE(61,2322)   PINIT6,PINC6,PMAX6,LODPON6
23       FORMAT(/,10X,7HPINIT1=,F15.8,10X,6HPINC1=,F15.8/
        +10X,6HPMAX1=,F15.8,10X,8HLODPON1=,I5)
231      FORMAT(/,10X,7HPINIT2=,F15.8,10X,6HPINC2=,F15.8/
        +10X,6HPMAX2=,F15.8,10X,8HLODPON2=,I5)
2319     FORMAT(/,10X,7HPINIT3=,F15.8,10X,6HPINC3=,F15.8/
        +10X,6HPMAX3=,F15.8,10X,8HLODPON3=,I5)
2320     FORMAT(/,10X,7HPINIT4=,F15.8,10X,6HPINC4=,F15.8/
        +10X,6HPMAX4=,F15.8,10X,8HLODPON4=,I5)
2321     FORMAT(/,10X,7HPINIT5=,F15.8,10X,6HPINC5=,F15.8/
        +10X,6HPMA5=,F15.8,10X,8HLODPON5=,I5)
2322     FORMAT(/,10X,7HPINIT6=,F15.8,10X,6HPINC6=,F15.8/
        +10X,6HPMAX5=,F15.8,10X,8HLODPON6=,I5)
         READ(60,1010) TITLE1,TITLE2,TITLE3,NE,NUMNP,ICAL1,
        +ICAL2,ICAL3
C        ****************************************************************
C           ICAL1=0 FOR DISPLACEMENT VECTOR PRINT OUT AND DETAILS
C           PRINT OUT(ICAL1=1 SKIP)
C           ICAL2=0 FOR ELEMENT TANGENT STIFFNESS MATRIX
C           PRINT OUT (ICAL2=1 SKIP)
C           ICAL3=0 FOR LOAD VECTOR PRINT OUT (ICAL3=1 SKIP)
C        ****************************************************************
         WRITE(61,2010) TITLE1,TITLE2,TITLE3,NE,NUMNP,ICAL1,
        +ICAL2,ICAL3
1010     FORMAT(A10,A10,A10,5I5)
2010     FORMAT(* *,A10,A10,A10//7X,6HNE    =,I3//7X,6HNUMNP=,I3//
        +7X,6HICAL1=,I3//7X,6HICAL2=,I3//7X,6HICAL3=,I3)
         CALL NADDATA(IGOPTIN)
         NCOUNT=1
         NUMITER=0
         PLOAD1=PINIT1
         PLOAD2=PINIT2
         PLOAD3=PINIT3
         PLOAD4=PINIT4
         PLOAD5=PINIT5
         PLOAD6=PINIT6
         SCALE=1000.
         WRITE(61,2015)
2015     FORMAT(///,15H  INITIAL LOADS//7H   NODE,17X,14HLOAD DIRECTION//
        +7H NUMBER,13X,1HX,9X,1HY,9X,2HTZ/)
1015     FORMAT(I5,3F10.5)
2020     FORMAT(I5,6X,3F10.5)
500      READ(60,1015) N,(PI(N,I),I=1,3)
         WRITE(61,2020) N,(PI(N,I),I=1,3)
         IF(N.NE.NUMNP)  GO TO 500
         DO 147 N=1,NUMNP
         DO 147 I=1,3
147      PII(N,I)=PI(N,I)
         DO 501  I=1,NEQ
501      DTOT(I)=0.0
         DO 502 I=1,NUMNP
         DO 502 J=1,3
```

```
502   W(I,J)=WTOT(I,J)=0.0
1001  DO 1503 I=1,NUMNP
      X(I)=X(I)+W(I,1)
1503  Y(I)=Y(I)+W(I,2)
      IF(ICAL1.EQ.1) GO TO 9001
      WRITE(61,9002)
9002  FORMAT(/,10X,*NODE*,10X,*X(I)*,10X,*Y(I)*,/)
      DO 9003 I=1,NUMNP
9003  WRITE(61,9004) I,X(I),Y(I)
9004  FORMAT(/,10X,I5,2F15.8)
9001  CONTINUE
      IF(NUMITER.EQ.0.AND.NCOUNT.EQ.1) GO TO 99
      PIE=3.1415926535898
      IPAR=3
      DO 1504 I=1,NEQ
      DO 1504 J=1,MBAND
1504  S(I,J)=0.0
      DO 504  M=1,NUMEL
      NI=NODEI(M)
      NJ=NODEJ(M)
      X0=X(NJ)-X(NI)
      Y0=Y(NJ)-Y(NI)
      ELO=SQRT(X0**2+Y0**2)
      C=X0/ELO
      Z=Y0/ELO
      IF(C.GE.0..AND.Z.GE.0.)   TET=ASIN(ABS(Y0/ELO))
      IF(C.LE.0..AND.Z.GE.0.)   TET=PIE-ASIN(ABS(Y0/ELO))
      IF(C.LE.0..AND.Z.LE.0.)   TET=PIE+ASIN(ABS(Y0/ELO))
      IF(C.GE.0..AND.Z.LT.0.)   TET=2.*PIE-ASIN(ABS(Y0/ELO))
      DEL=(ELO-LE(M))/LE(M)
      IF(ITERCHK.EQ.0) GO TO 8001
      ALFA=TETO(M)-TET
      IF(ABS(ALFA).GE.PIE)  ALFA=2.*PIE-ABS(ALFA)
      USTAR(1,M)=ALFA+WTOT(NI,3)
      USTAR(2,M)=ALFA+WTOT(NJ,3)
      USTAR(3,M)=ELO-LE(M)
      GO TO 8002
8001  CONTINUE
      DELU(1,1)=(ZOLD(M)*(W(NJ,1)-W(NI,1))+COLD(M)*(W(NI,2)-W(NJ,2)))/
     +(LE(M)*(1.+DELOLD(M)))+W(NI,3)
      DELU(2,1)=(ZOLD(M)*(W(NJ,1)-W(NI,1))+COLD(M)*(W(NI,2)-W(NJ,2)))/
     +(LE(M)*(1.+DELOLD(M)))+W(NJ,3)
      IF(PROTYPE.EQ.1)
     +DELU(3,1)=-COLD(M)*(W(NI,1)-W(NJ,1))-ZOLD(M)*(W(NI,2)-W(NJ,2))
      IF(PROTYPE.EQ.2)
     +DELU(3,1)=(COLD(M)*(W(NI,1)-W(NJ,1))+
     +ZOLD(M)*(W(NI,2)-W(NJ,2)))/LE(M)
C         TO HAVE USTAR(I)
C     ***************************************************************
      USTAR(1,M)=USTAR(1,M)+DELU(1,1)
      USTAR(2,M)=USTAR(2,M)+DELU(2,1)
      USTAR(3,M)=USTAR(3,M)+DELU(3,1)
8002  CONTINUE
      IF(ICAL1.EQ.1) GO TO 9005
      WRITE(61,9006)
9006  FORMAT(/,10X,*M*,10X,*USTAR(1)*,10X,*USTAR(2)*,10X,*USTAR(3)*,/)
      WRITE(61,9007) M,USTAR(1,M),USTAR(2,M),USTAR(3,M)
9007  FORMAT(5X,I5,5X,F15.8,5X,F15.8,5X,F15.8)
9005  CONTINUE
C
C
C
C
C
      IF(PROTYPE.EQ.1)  GO TO 8020
      IF(ITERCHK.NE.0)  GO TO 7011
C         TO HAVE DELS(I,1)
C     ***************************************************************
      DO 4929 I=1,3
      DO 4929 J=1,3
4929  TTSML(I,J)=TSMAL(I,J,M)
      CALL MULT(3,3,1,TTSML,DELU,DELS)
      ES(1,M)=ES(1,M)+DELS(1,1)
      ES(2,M)=ES(2,M)+DELS(2,1)
      ES(3,M)=ES(3,M)+DELS(3,1)
      Q=ES(3,M)/LE(M)
7011  CONTINUE
      IOITER=0
      IF(ITERCHK.NE.0) Q=QSMAL=0.0
      IF(ITERCHK.EQ.0) QSMAL=Q*LE(M)**2/(PIE**2*E(1)*IXX(M))
1999  IF(ABS(QSMAL).LE..1) GO TO 4999
```

```
      IF(Q.GT.0.) FE=SQRT(QSMAL*PIE**2)
      IF(Q.GT.0.) C1=(FE*SIN(FE)-COS(FE)*FE**2)/
     +(2.-2.*COS(FE)-FE*SIN(FE))
      IF(Q.GT.0.) C2=(FE**2-FE*SIN(FE))/
     +(2.-2.*COS(FE)-FE*SIN(FE))
      IF(Q.LT.0.) SI=SQRT(-QSMAL*PIE**2)
      IF(Q.LT.0.) C1=(COSH(SI)*SI**2-SI*SINH(SI))/
     +(2.-2.*COSH(SI)+SI*SINH(SI))
      IF(Q.LT.0.) C2=(SI*SINH(SI)-SI**2)/
     +(2.-2.*COSH(SI)+SI*SINH(SI))
      IF(Q.NE.0.) B1=+(C1+C2)*(C2-2.)/
     +(8.*QSMAL*PIE**2)
      B2=C2/(8.*(C1+C2))
      CPRIM1=-2.*PIE**2*(B1+B2)
      CPRIM2=-2.*PIE**2*(B1-B2)
      IF(Q.NE.0.)BPRIM1=-((B1-B2)*(C1+C2)+2.*C2*B1)/(4.*QSMAL)
      BPRIM2=PIE**2*(16.*B1*B2-B1+B2)/(4.*(C1+C2))
      GO TO 3999
4999  V=QSMAL
      IF(Q.EQ.0.) C1=4.
      IF(Q.EQ.0.) C2=2.
      IF(Q.EQ.0.) B1=1./40.
      IF(Q.EQ.0.) B2=1./24.
      IF(Q.EQ.0.) BPRIM1=PIE**2/2800.
      IF(Q.EQ.0.) BPRIM2=PIE**2/720.
      IF(Q.EQ.0.) CPRIM1=-2.*PIE**2/15.
      IF(Q.EQ.0.) CPRIM2=PIE**2/30.
      IF(Q.EQ.0.) GO TO 3999
      C1=4.-2.*PIE**2*V/15.-11.*PIE**4*V**2/6300.-PIE**6*V**3/27000.
      C2=2.+PIE**2*V/30.+13.*PIE**4*V**2/12600.+11.*PIE**6*V**3/378000.
      B1=1./40.+PIE**2*V/2800.+PIE**4*V**2/168000.+37.*PIE**6*V**3/
     +388080000.
      BPRIM1=PIE**2/2800.+PIE**4*V/84000.+37.*PIE**6*V**2/129360000.
      B2=1./24.+PIE**2*V/720.+PIE**4*V**2/20160.+PIE**6*V**3/604800.
      CPRIM1=-2.*PIE**2/15.-11.*PIE**4*V/3150.-PIE**6*V**2/9000.
      CPRIM2=PIE**2/30.+13.*PIE**4*V/6300.+11.*PIE**6*V**2/126000.
      BPRIM2=PIE**2/720.+PIE**4*V/10080.+PIE**6*V**2/201600.
3999  CB=B1*(USTAR(1,M)+USTAR(2,M))**2+B2*(USTAR(1,M)-USTAR(2,M))**2
      LAMBDA=LE(M)/SQRT(IXX(M)/A(M))
      H=PIE**2/LAMBDA**2+BPRIM1*(USTAR(1,M)+USTAR(2,M))**2+
     +BPRIM2*(USTAR(1,M)-USTAR(2,M))**2
      IF(ITERCHK.EQ.0) GO TO 7003
      KQ=PIE**2*QSMAL/LAMBDA**2+CB+USTAR(3,M)/LE(M)
      DELQ=-KQ/H
      IQITER=IQITER+1
      IF(IQITER.GT.MAX2)  GO TO 900
      QSMAL=QSMAL+DELQ
      Q=QSMAL*(PIE**2*E(1)*IXX(M))/LE(M)**2
      IF(ABS(DELQ).LT.EPSI2)  GO TO 2999
      GO TO 1999
2999  CONTINUE
C        TO HAVE S1,S2,S3
C     ******************************************************************
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2)  GO TO 5891
      IF(ICAL1.EQ.1) GO TO 5891
      WRITE(61,9009)
9009  FORMAT(/,10X,*M*,10X,*IQITER*,10X,*Q*,10X,*QSMAL*,10X,*DELQ*,/)
      WRITE(61,9010) M,IQITER,Q,QSMAL,DELQ
9010  FORMAT(/,7X,I5,10X,I5,3F20.10)
5891  CONTINUE
      ES(1,M)=E(1)*IXX(M)*(C1*USTAR(1,M)+C2*USTAR(2,M))/LE(M)
      ES(2,M)=E(1)*IXX(M)*(C2*USTAR(1,M)+C1*USTAR(2,M))/LE(M)
      ES(3,M)=Q*LE(M)
7003  IF(ICAL1.EQ.1) GO TO 9011
      WRITE(61,9012)
9012  FORMAT(/,14X,*M*,10X,*S1*,14X,*S2*,14X,*S3*,/)
      WRITE(61,9013) M,ES(1,M),ES(2,M),ES(3,M)
9013  FORMAT(/,10X,I5,3F20.10)
9011  CONTINUE
      G1=CPRIM1*USTAR(1,M)+CPRIM2*USTAR(2,M)
      G2=CPRIM2*USTAR(1,M)+CPRIM1*USTAR(2,M)
C        TO HAVE TSMAL(3.3)
C     ******************************************************************
      TSMAL(1,1,M)=(C1+G1**2/(PIE**2*H))*E(1)*IXX(M)/LE(M)
      TSMAL(2,2,M)=(C1+G2**2/(PIE**2*H))*E(1)*IXX(M)/LE(M)
      TSMAL(3,3,M)=(PIE**2/H)*E(1)*IXX(M)/LE(M)
      TSMAL(1,2,M)=TSMAL(2,1,M)=(C2+G1*G2/(PIE**2*H))*E(1)*IXX(M)/LE(M)
      TSMAL(1,3,M)=TSMAL(3,1,M)=(G1/H)*E(1)*IXX(M)/LE(M)
      TSMAL(2,3,M)=TSMAL(3,2,M)=(G2/H)*E(1)*IXX(M)/LE(M)
      DO 4928  I=1,3
      DO 4928  J=1,3
```

```
4928  TTSML(I,J)=TSMAL(I,J,M)
      IF(ICAL1.EQ.1) GO TO 9015
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2)  GO TO 9015
      WRITE(61,521)
521   FORMAT(/,10X,*TSMAL(I,J)*,/)
      WRITE(61,402) ((TSMAL(I,J,M),J=1,3),I=1,3)
402   FORMAT(/1X,3F20.6)
9015  CONTINUE
C        TO HAVE B MATRIX FROM ORAN,S PAPER 2 DIMENSIONAL CASE
C     ****************************************************************
      B(6,1)=B(3,2)=B(3,3)=B(6,3)=0.0
      B(1,1)=B(1,2)=-Z/(1.+DEL)
      B(4,1)=B(4,2)=-B(1,1)
      B(2,1)=B(2,2)=C/(1.+DEL)
      B(5,1)=B(5,2)=-B(2,1)
      B(1,3)=C
      B(2,3)=Z
      B(4,3)=-C
      B(5,3)=-Z
      B(3,1)=B(6,2)=1.
      DO 5061 I=1,6
      DO 5061 J=1,3
5061  BTRAN(J,I)=B(I,J)
      IF(ICAL1.EQ.1) GO TO 9016
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2)  GO TO 9016
      WRITE(61,506)
506   FORMAT(10X,*B(I,J)*,/)
      WRITE(61,402) ((B(I,J),J=1,3),I=1,6)
      WRITE(61,507)
507   FORMAT(10X,*BTRAN(I,J)*,/)
      WRITE(61,503) ((BTRAN(I,J),J=1,6),I=1,3)
503   FORMAT(/1X,6F20.10)
9016  CONTINUE
      IF(ITERCHK.EQ.0)  GO TO 9019
C        TO HAVE END FORCES FOR EACH MEMBER IN GLOBAL DN(6,1)
C     ****************************************************************
      DO 7004  I=1,3
7004  ESS(I,1)=ES(I,M)
      CALL MULT(6,3,1,B,ESS,DN)
      DN(1,1)=DN(1,1)/LE(M)
      DN(2,1)=DN(2,1)/LE(M)
      DN(4,1)=DN(4,1)/LE(M)
      DN(5,1)=DN(5,1)/LE(M)
      IF(ICAL1.EQ.1) GO TO 9017
      WRITE(61,9018)M,DN(1,1),DN(2,1),DN(3,1),DN(4,1),DN(5,1),DN(6,1)
9018  FORMAT(/,10X,*M=*,I5,5X,*DN(1)=*,F15.8,5X,*DN(2)=*,F15.8,/
     +,10X,*DN(3)=*,F15.8,*DN(4)=*,F15.8,5X,*DN(5)=*,F15.8,
     +10X,*DN(6)=*,F15.8,/)
9017  CONTINUE
C        TO HAVE UNBALANCED FORCES
C     ****************************************************************
      DO 145 I=1,3
145   PI(NI,I)=PI(NI,I)-DN(I,1)
      DO 146 I=4,6
146   PI(NJ,I-3)=PI(NJ,I-3)-DN(I,1)
      IF(ICAL1.EQ.1) GO TO 9019
      WRITE(61,9020)M,PI(NI,1),PI(NI,2),PI(NI,3),PI(NJ,1),PI(NJ,2)
     +,PI(NJ,3)
9020  FORMAT(/,10X,*M=*,I5,5X,*PI(NI,1)=*,F15.8,5X,*PI(NI,2)=*,F15.8,/
     +,10X,*PI(NI,3)=*,F15.8,5X,*PI(NJ,1)=*,F15.8,
     +10X,*PI(NJ,2)=*,F15.8,*PI(NJ,3)=*,F15.8,/)
9019  CONTINUE
C        TO HAVE TBTRAN=TTSML*BTRANSPOSE
C     ****************************************************************
      CALL MULT(3,3,6,TTSML,BTRAN,TBTRAN)
C        TO HAVE PART1=B*TSMAL*BTRAN
C     ****************************************************************
      CALL MULT(6,3,6,B,TBTRAN,PART1)
      IF(ICAL1.EQ.1) GO TO 9021
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2)  GO TO 9021
      WRITE(61,525)
525   FORMAT(/,10X,*TBTRAN*,/)
      WRITE(61,503) ((TBTRAN(I,J),J=1,6),I=1,3)
      WRITE(61,526)
526   FORMAT(/,10X,*PART1(I,J)*,/)
      WRITE(61,503) ((PART1(I,J),J=1,6),I=1,6)
9021  CONTINUE
C        TO HAVE GSMAL1,GSMAL2
C     ****************************************************************
      DO 527 I=1,6
      DO 527 J=I,6
```

```
527   GS1(I,J)=GS2(I,J)=0.0
      GS1(2,5)=GS1(1,1)=GS1(4,4)=-2.*C*Z/(1.+DEL)**2
      GS1(1,4)=GS1(2,2)=GS1(5,5)=-GS1(1,1)
      GS1(1,2)=GS1(4,5)=(C**2-Z**2)/(1.+DEL)**2
      GS1(1,5)=GS1(2,4)=-GS1(1,2)
      GS2(1,1)=GS2(4,4)=-Z**2/(1.+DEL)
      GS2(1,4)=-GS2(1,1)
      GS2(1,2)=GS2(4,5)=C*Z/(1.+DEL)
      GS2(1,5)=GS2(2,4)=-GS2(1,2)
      GS2(2,2)=GS2(5,5)=-C**2/(1.+DEL)
      GS2(2,5)=-GS2(2,2)
      DO 528 I=1,6
      DO 528 J=1,I
      GS1(I,J)=GS1(J,I)
528   GS2(I,J)=GS2(J,I)
      IF(ICAL1.EQ.1) GO TO 9022
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2)  GO TO 9022
      WRITE(61,531)
531   FORMAT(/,10X,*GS1(I,J)*,/)
      WRITE(61,503) ((GS1(I,J),J=1,6),I=1,6)
      WRITE(61,530)
530   FORMAT(/,10X,*GS2(I,J)*,/)
      WRITE(61,503) ((GS2(I,J),J=1,6),I=1,6)
9022  CONTINUE
      IF(ITERCHK.NE.0)  GO TO 7005
      DELOLD(M)=DEL
      COLD(M)=C
      ZOLD(M)=Z
7005  CONTINUE
C        TO HAVE G1*S1+G1*S2+G2*S3
C        AND TANGENT STIFFNESS MATRIX IN GLOBAL FOR EACH MEMBER
C     **********************************************************************
      DO 535 I=1,6
      DO 535 J=1,6
      PART21(I,J)=ES(1,M)*GS1(I,J)
      PART22(I,J)=ES(2,M)*GS1(I,J)
535   PART23(I,J)=ES(3,M)*GS2(I,J)
      IF(ICAL1.EQ.1) GO TO 9023
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2)  GO TO 9023
      WRITE(61,536)
536   FORMAT(/,10X,*PART21(I,J)*,/)
      WRITE(61,503) ((PART21(I,J),J=1,6),I=1,6)
      WRITE(61,537)
537   FORMAT(/,10X,*PART22(I,J)*,/)
      WRITE(61,503) ((PART22(I,J),J=1,6),I=1,6)
      WRITE(61,538)
538   FORMAT(/,10X,*PART23(I,J)*,/)
      WRITE(61,503) ((PART23(I,J),J=1,6),I=1,6)
9023  CONTINUE
      DO 539 I=1,6
      DO 539 J=I,6
539   SE(I,J)=PART1(I,J)+PART21(I,J)+PART22(I,J)+PART23(I,J)
C     **********************************************************************
      SE(1,1)=SE(1,1)/LE(M)**2
      SE(2,2)=SE(2,2)/LE(M)**2
      SE(1,2)=SE(1,2)/LE(M)**2
      SE(1,4)=SE(1,4)/LE(M)**2
      SE(1,5)=SE(1,5)/LE(M)**2
      SE(2,4)=SE(2,4)/LE(M)**2
      SE(2,5)=SE(2,5)/LE(M)**2
      SE(1,3)=SE(1,3)/LE(M)
      SE(1,6)=SE(1,6)/LE(M)
      SE(2,3)=SE(2,3)/LE(M)
      SE(2,6)=SE(2,6)/LE(M)
      SE(3,4)=SE(3,4)/LE(M)
      SE(3,5)=SE(3,5)/LE(M)
      SE(4,4)=SE(4,4)/LE(M)**2
      SE(4,5)=SE(4,5)/LE(M)**2
      SE(5,5)=SE(5,5)/LE(M)**2
      SE(4,6)=SE(4,6)/LE(M)
      SE(5,6)=SE(5,6)/LE(M)
      DO 540 I=1,6
      DO 540 J=1,I
540   SE(I,J)=SE(J,I)
      GO TO 8021
C
C
C
C
C
8020  CONTINUE
```

```
C     *****************************************************************
C        TO HAVE T1 AND T1TRANSPOSE
C     *****************************************************************
      T1(1,1)=T1(2,1)=-Z/ELO
      T1(1,4)=T1(2,4)=-T1(1,1)
      T1(1,2)=T1(2,2)=C/ELO
      T1(1,5)=T1(2,5)=-T1(1,2)
      T1(3,1)=-C
      T1(3,4)=C
      T1(3,2)=-Z
      T1(3,5)=Z
      T1(1,3)=T1(2,6)=1.
      T1(2,3)=T1(3,3)=T1(1,6)=T1(3,6)=0.0
      DO 106 I=1,3
      DO 106 J=1,6
106   T1TRAN(J,I)=T1(I,J)
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2) GO TO 916
      IF(ICAL1.EQ.1) GO TO 916
      WRITE(61,203)
203   FORMAT(10X,*T1(I,J)*,/)
      WRITE(61,202) ((T1(I,J),J=1,6),I=1,3)
202   FORMAT(/1X,6F20.6)
      WRITE(61,405)
405   FORMAT(10X,*T1TRAN(I,J)*,/)
      WRITE(61,406) ((T1TRAN(I,J),J=1,3),I=1,6)
406   FORMAT(/1X,3F20.6)
916   CONTINUE
      IF(ITERCHK.NE.0)  GO TO 8003
C        TO HAVE DELS(I,1)
C     *****************************************************************
      DO 5929 I=1,3
      DO 5929 J=1,3
5929  STSTAR(I,J)=STSTARM(I,J,M)
      CALL MULT(3,3,1,STSTAR,DELU,DELS)
      ES(1,M)=ES(1,M)+DELS(1,1)
      ES(2,M)=ES(2,M)+DELS(2,1)
      ES(3,M)=ES(3,M)+DELS(3,1)
8003  CONTINUE
C        TO HAVE LINEAR STIFFNESS MATRIX K
C     *****************************************************************
      RK(1,1)=RK(2,2)=4.*E(1)*IXX(M)/LE(M)
      RK(1,2)=RK(2,1)=2.*E(1)*IXX(M)/LE(M)
      RK(3,3)=E(1)*A(M)/LE(M)
      RK(1,3)=RK(2,3)=RK(3,1)=RK(3,2)=0.0
C        TO HAVE N1
C     *****************************************************************
C
C
      N1(1,1)=N1(2,2)=2.*USTAR(3,M)*E(1)*A(M)/15.
      N1(1,2)=N1(2,1)=-USTAR(3,M)*E(1)*A(M)/30.
      N1(1,3)=N1(3,1)=(4.*USTAR(1,M)-USTAR(2,M))*E(1)*A(M)/30.
      N1(2,3)=N1(3,2)=(-USTAR(1,M)+4.*USTAR(2,M))*E(1)*A(M)/30.
      N1(3,3)=0.0
C        TO HAVE N2
C     *****************************************************************
C
C
C
      IF(N2OPTIN.EQ.1)N2(1,1)=LE(M)*(8.*USTAR(1,M)**2-4.*USTAR(1,M)*
     +USTAR(2,M)+3.*USTAR(2,M)**2)*E(1)*A(M)/300.
      IF(N2OPTIN.EQ.1)N2(2,2)=LE(M)*(3.*USTAR(1,M)**2-4.*USTAR(1,M)*
     +USTAR(2,M)+8.*USTAR(2,M)**2)*E(1)*A(M)/300.
      IF(N2OPTIN.EQ.1)N2(1,2)=N2(2,1)=LE(M)*(-2.*USTAR(1,M)**2+6.*
     +USTAR(1,M)*USTAR(2,M)-2.*USTAR(2,M)**2)*E(1)*A(M)/300.
C
C
      IF(N2OPTIN.EQ.2) N2(1,1)=LE(M)*(12.*USTAR(1,M)**2-3.*USTAR(1,M)*
     +USTAR(2,M)+USTAR(2,M)**2)*E(1)*A(M)/140.
      IF(N2OPTIN.EQ.2) N2(2,2)=LE(M)*(USTAR(1,M)**2-3.*USTAR(1,M)*
     +USTAR(2,M)
     ++12.*USTAR(2,M)**2)*E(1)*A(M)/140.
      IF(N2OPTIN.EQ.2) N2(1,2)=N2(2,1)=-LE(M)*(3.*USTAR(1,M)**2-4.*
     +USTAR(1,M)*USTAR(2,M)+3.*USTAR(2,M)**2)*E(1)*A(M)/280.
C
C
C
C
      N2(1,3)=N2(2,3)=N2(3,1)=N2(3,2)=N2(3,3)=0.0
C        TO HAVE SSECANT*,AND S TANGENT*
C     *****************************************************************
      DO 105 I=1,3
      DO 105 J=1,3
      STSTAR(I,J)=RK(I,J)+N1(I,J)+N2(I,J)
      STSTARM(I,J,M)=STSTAR(I,J)
105   SECSTAR(I,J)=RK(I,J)+.5*N1(I,J)+N2(I,J)/3.
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2) GO TO 917
      IF(ICAL1.EQ.1) GO TO 917
      WRITE(61,409)
409   FORMAT(10X,*RK(I,J)*,/)
      WRITE(61,410)
410   FORMAT(/1X,3F20.6)
      WRITE(61,411)
411   FORMAT(10X,*N1(I,J)*,/)
      WRITE(61,410) ((N1(I,J),J=1,3),I=1,3)
      WRITE(61,412)
412   FORMAT(10X,*N2(I,J)*,/)
      WRITE(61,410) ((N2(I,J),J=1,3),I=1,3)
      WRITE(61,413)
413   FORMAT(10X,*STSTAR(I,J)*,/)
      WRITE(61,410) ((STSTAR(I,J),J=1,3),I=1,3)
      WRITE(61,414)
414   FORMAT(10X,*SECSTAR(I,J)*,/)
      WRITE(61,410) ((SECSTAR(I,J),J=1,3),I=1,3)
917   CONTINUE
      IF(ITERCHK.EQ.0) GO TO 8004
      DO 8916 I=1,3
8916  USTAR(I,1)=USTAR(I,M)
C         TO HAVE ESS=SECSTAR*USTAR
C     ******************************************************************
      CALL MULT(3,3,1,SECSTAR,USTAR,ESS)
      DO 8005 I=1,3
8005  ES(I,M)=ESS(I,1)
8004  CONTINUE
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2) GO TO 918
      IF(ICAL1.EQ.1) GO TO 918
      WRITE(61,415)
415   FORMAT(10X,*ES(I)*,/)
      WRITE(61,408) (ES(I,M),I=1,3)
408   FORMAT(10X,F20.10)
918   CONTINUE
      IF(ITERCHK.EQ.0)  GO TO 9109
C         TO HAVE END FORCES FOR EACH MEMBER IN GLOBAL DN(6,1)
C     ******************************************************************
      CALL MULT(6,3,1,T1TRAN,ESS,DN)
      IF(ICAL1.EQ.1) GO TO 7109
      WRITE(61,8109)M,DN(1,1),DN(2,1),DN(3,1),DN(4,1),DN(5,1),DN(6,1)
8109  FORMAT(/,10X,*M=*,I5,5X,*DN(1)=*,F15.8,5X,*DN(2)=*,F15.8,/
     +,10X,*DN(3)=*,F15.8,*DN(4)=*,F15.8,5X,*DN(5)=*,F15.8,
     +10X,*DN(6)=*,F15.8,/)
7109  CONTINUE
C         TO HAVE UNBALANCED FORCES
C     ******************************************************************
      DO 1145 I=1,3
1145  PI(NI,I)=PI(NI,I)-DN(I,1)
      DO 1146 I=4,6
1146  PI(NJ,I-3)=PI(NJ,I-3)-DN(I,1)
      IF(ICAL1.EQ.1) GO TO 9109
      WRITE(61,1209)M,PI(NI,1),PI(NI,2),PI(NI,3),PI(NJ,1),PI(NJ,2)
     +,PI(NJ,3)
1209  FORMAT(/,10X,*M=*,I5,5X,*PI(NI,1)=*,F15.8,5X,*PI(NI,2)=*,F15.8,/
     +,10X,*PI(NI,3)=*,F15.8,5X,*PI(NJ,1)=*,F15.8,
     +10X,*PI(NJ,2)=*,F15.8,*PI(NJ,3)=*,F15.8,/)
9109  CONTINUE
C         TO HAVE T21=T22 AND T23(SAY T22 IN CODING)
C     ******************************************************************
      DO 112 I=1,6
      DO 112 J=I,6
112   T21(I,J)=T22(I,J)=0.0
      T21(1,1)=T21(2,5)=T21(4,4)=-2.*Z*C
      T21(1,4)=T21(2,2)=T21(5,5)=-T21(1,1)
      T21(1,5)=T21(2,4)=Z**2-C**2
      T21(4,5)=T21(1,2)=-T21(1,5)
      T22(1,1)=T22(4,4)=Z**2
      T22(1,4)=-T22(1,1)
      T22(1,5)=T22(2,4)=Z*C
      T22(1,2)=T22(4,5)=-T22(1,5)
      T22(2,2)=T22(5,5)=C**2
      T22(2,5)=-T22(2,2)
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2) GO TO 983
      IF(ICAL1.EQ.1) GO TO 983
      WRITE(61,1983)
1983  FORMAT(/,10X,*T21(I,J)*,/)
```

```
      WRITE(61,202) ((T21(I,J),J=1,6),I=1,6)
      WRITE(61,1984)
1984  FORMAT(/,10X,*T22(I,J)*,/)
      WRITE(61,202) ((T22(I,J),J=1,6),I=1,6)
983   CONTINUE
C        TO HAVE SE(I,J)=PART1(I,J)+PART2(I,J) EQ -10
C        TO HAVE STT1=ST*.T1
C     ******************************************************************
      CALL MULT(3,3,6,STSTAR,T1,STT1)
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2) GO TO 919
      IF(ICAL1.EQ.1) GO TO 919
      WRITE(61,421)
421   FORMAT(10X,*STT1*,/)
      WRITE(61,202) ((STT1(I,J),J=1,6),I=1,3)
919   CONTINUE
C        TO HAVE PART1(I,J)=T1TRAN.ST*.T1
C     ******************************************************************
      CALL MULT(6,3,6,T1TRAN,STT1,PART1)
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2) GO TO 982
      IF(ICAL1.EQ.1) GO TO 982
      WRITE(61,1982)
1982  FORMAT(10X,*PART1(I,J)*,/)
      WRITE(61,202) ((PART1(I,J),J=1,6),I=1,6)
982   CONTINUE
      IF(ITERCHK.NE.0) GO TO 8006
      DELOLD(M)=DEL
      COLD(M)=C
      ZOLD(M)=Z
8006  CONTINUE
C        TO HAVE PART2(I,J)=U*TRANSPOSE.SSECANT*TRANSPOSE.T2
C     ******************************************************************
      DO 113 I=1,6
      DO 113 J=I,6
      PART2(I,J)=(ES(1,M)+ES(2,M))*T21(I,J)/
     +(ELO**2)+ES(3,M)*T22(I,J)/ELO
113   PART2(J,I)=PART2(I,J)
      IF(NCOUNT.GT.2.OR.NUMITER.GT.2.OR.M.GT.2) GO TO 981
      IF(ICAL1.EQ.1) GO TO 981
      WRITE(61,1981)
1981  FORMAT(10X,*PART2(I,J)*,/)
      WRITE(61,202)((PART2(I,J),J=1,6),I=1,6)
981   CONTINUE
C        TO HAVE TANGENT STIFFNESS FOR EACH MEMBER (NONLINEAR)
C     ******************************************************************
      DO 114I=1,6
      DO 114 J=1,6
114   SE(I,J)=PART1(I,J)+PART2(I,J)
8021  CONTINUE
C
C
C
C
C
      WRITE(3,10) ((SE(I,J),J=1,6),I=1,6)
      IF(ICAL2.EQ.1) GO TO 9024
      WRITE(61,9025) M
9025  FORMAT(/,10X,*M=*,I5,10X,*SE(I,J),NONLINEAR*,/)
      WRITE(61,503) ((SE(I,J),J=1,6),I=1,6)
9024  CONTINUE
C        TO HAVE STRUCTURAL TANGENT STIFFNESS IN GLOBAL
C     ******************************************************************
      IPAR=3
      CALL ASEMBLE(M)
504   CONTINUE
      IF(ITERCHK.EQ.0.AND.ISTRESS.EQ.1) CALL STRESS
      REWIND 3
      IF(ICAL2.EQ.1) GO TO 9026
      WRITE(61,9027)
9027  FORMAT(/,10X,*S(I,J) NONLINEAR*,/)
      WRITE(61,503) ((S(I,J),J=1,MBAND),I=1,NEQ)
9026  CONTINUE
      WRITE(4,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      REWIND 4
C        TO HAVE UNBALANCED FORCE VECTOR ASSEMBLED
C     ******************************************************************
      IF(ITERCHK.NE.0)  GO TO 1591
      DO 1598 I=1,NEQ
      IF(I.EQ.LODPON1)  R(I)=PINC1
      IF(I.EQ.LODPON2)  R(I)=PINC2
      IF(I.EQ.LODPON3)  R(I)=PINC3
      IF(I.EQ.LODPON4)  R(I)=PINC4
      IF(I.EQ.LODPON5)  R(I)=PINC5
      IF(I.EQ.LODPON6)  R(I)=PINC6
      IF(I.NE.LODPON1.AND.I.NE.LODPON2.AND.I.NE.LODPON3
     *.AND.I.NE.LODPON4.AND.I.NE.LODPON5.AND.I.NE.LODPON6)   R(I)=0.
 1598 CONTINUE
      GO TO 1593
 1591 IPAR=1
      CALL ASEMBLE(M)
 1593 IF(ICAL3.EQ.1)  GO TO 9028
      WRITE(61,9029)
 9029 FORMAT(/,10X,*R(I)-UNBALANCED FORCE VECTOR*,/)
      WRITE(61,9030) (R(I),I=1,NEQ)
 9030 FORMAT(5X,F20.10)
 9028 CONTINUE
      GO TO 399
C        TO HAVE ELEMENT PROPERTIES
C     ****************************************************************
 99   CALL ELEMENT
      IF(ITERCHK.NE.0) GO TO 8007
      DO 9083 I=1,3
      DO 9083 J=1,NUMEL
 9083 ES(I,J)=USTAR(I,J)=0.0
      DO 9084 I=1,NUMEL
 9084 DELOLD(I)=0.0
 8007 CONTINUE
C        TO HAVE SEMIBANDWIDTH
C     ****************************************************************
      CALL BAND
C        TO HAVE EXTERNAL LOAD VECTOR ASSEMBLED
C     ****************************************************************
      IPAR=1
      CALL ASEMBLE(M)
      IF(ICAL3.EQ.1) GO TO 9031
      WRITE(61,9032)
 9032 FORMAT(/,10X,*R(I)-EXTERNAL LOAD VECTOR*,/)
      WRITE(61,9030) (R(I),I=1,NEQ)
 9031 CONTINUE
C        TO HAVE LINEAR STIFFNESS FOR MEMBERS AND STRUCTURE
C        IN GLOBAL
C     ****************************************************************
      IPAR=2
      DO 5044 I=1,NEQ
      DO 5044 J=1,MBAND
 5044 S(I,J)=0.0
      CALL BEAM
      IF(ITERCHK.NE.0) GO TO 8008
      DO 9085 I=1,NUMEL
      COLD(I)=COS(TETO(I))
 9085 ZOLD(I)=SIN(TETO(I))
 8008 CONTINUE
 399  CALL LINSOLN
      DO 115 I=1,NEQ
 115  DTOT(I)=DTOT(I)+D(I)
C        TO HAVE TOTAL DISPLACEMENTS AND ROTATIONS FOR EACH NODE
C        IN GLOBAL
C     ****************************************************************
      WRITE(61,9033)
 9033 FORMAT(/,10X,*DTOT(I)*,/)
      WRITE(61,9030) (DTOT(I),I=1,NEQ)
      CALL IDENT
      DO 1399 I=1,NUMNP
      DO 1399 J=1,3
 1399 WTOT(I,J)=WTOT(I,J)+W(I,J)
      IF(ICAL1.EQ.1) GO TO 9034
      WRITE(61,9035)
 9035 FORMAT(/,36X,*WTOT(I,J)*,53X,*W(I,J)*,/)
      WRITE(61,9036)((WTOT(I,J),J=1,3),(W(I,J),J=1,3),I=1,NUMNP)
 9036 FORMAT(/,1X,6F22.10)
 9034 CONTINUE
      IF(ITERCHK.NE.2)   GO TO 3727
      IF(ICHKOPT.EQ.1)   GO TO 4377
C        TO CHECK CONVERGENCE
C     ****************************************************************
      SUM1=SUM2=SUM3=SUM4=0.0
      DO 1400 I=1,NUMNP
      SUM1=SUM1+W(I,1)**2+W(I,2)**2
      SUM2=SUM2+W(I,3)**2
      SUM3=SUM3+WTOT(I,1)**2+WTOT(I,2)**2
 1400 SUM4=SUM4+WTOT(I,3)**2
      ERROR1=SQRT(SUM1/SUM3)
      ERROR2=SQRT(SUM2/SUM4)
      IF(ICAL1.EQ.1) GO TO 9037
      WRITE(61,9038) ERROR1,ERROR2
9038  FORMAT(/,10X,*ERROR1=*,F15.8,10X,*ERROR2=*,F15.8,/)
9037  IF(ERROR1.GT.EPSI1.OR.ERROR2.GT.EPSI1)  GO TO 520
      IF(ISTRESS.EQ.1)  CALL STRESS
4377  CONTINUE
      IF(ICHKOPT.EQ.2)   GO TO 4378
C     TO CHECK CONVERGENCE
C     ***************************************************************
      DO 4379  K=1,NUMEL
      M=L(1,K)
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 4379  K1=1,2
      IF(K1.EQ.1)  NP=NI
      IF(K1.EQ.2)  NP=NJ
      DO 4380  I=1,3
      IF(IA(NP,I))  4380,4381,4382
4382  NL=IA(NP,I)
      UNBLANC(NP,I)=R(NL)
      GO TO 4380
4381  UNBLANC(NP,I)=0.0
4380  CONTINUE
4379  CONTINUE
      DO 4383  NP=1,NUMNP
      PARTT1=ABS(UNBLANC(NP,1))
      PARTT2=ABS(UNBLANC(NP,2))
      PARTT3=ABS(UNBLANC(NP,3))
      IF(PARTT1.GT.EPSI3.OR.PARTT2.GT.EPSI3.OR.PARTT3.GT.EPSI4)
     +GO TO 4384
      WRITE(61,4385)  PARTT1,PARTT2,PARTT3
4383  CONTINUE
      IF(ISTRESS.EQ.1)  CALL STRESS
      GO TO 4378
4384  WRITE(61,4385)  PARTT1,PARTT2,PARTT3
4385  FORMAT(10X,6HPART1=,E21.15,10X,6HPART2=,E21.15,10X,6HPART3=
     +,E21.15)
      GO TO 520
4378  CONTINUE
3727  IDET=1
      DETER=DET1(SCALE)
      WRITE(61,1401) DETER,PLOAD1,PLOAD2,PLOAD3,PLOAD4,PLOAD5,PLOAD6
1401  FORMAT(/,10X,*DETERMINANT=*,E21.15/,10X,*PLOAD1=*,F20.10,/
     +10X,*PLOAD2=*,F20.10,10X,*PLOAD3=*,F20.10,10X,*PLOAD4=*,F20.10,/
     +10X,*PLOAD5=*,F20.10,10X,*PLOAD6=*,F20.10,/)
      PLOAD1=PLOAD1+PINC1
      PLOAD2=PLOAD2+PINC2
      PLOAD3=PLOAD3+PINC3
      PLOAD4=PLOAD4+PINC4
      PLOAD5=PLOAD5+PINC5
      PLOAD6=PLOAD6+PINC6
      IF(ABS(PLOAD1).GT.ABS(PMAX1).OR.ABS(PLOAD2).GT.ABS(PMAX2)
     +.OR.ABS(PLOAD3).GT.ABS(PMAX3))
     +GO TO 900
      IF(DETOPTN.EQ.1.AND.DETER.LE.0.)    GO TO 900
      NCOUNT=NCOUNT+1
      NUMITER=0
      DO 801 N=1,NUMNP
      DO 1701 I=1,3
      IF(IA(N,I)) 1701,1701,1004
1004  II=IA(N,I)
      IF(II.EQ.LODPON1)  PII(N,I)=PLOAD1
      IF(II.EQ.LODPON2)  PII(N,I)=PLOAD2
      IF(II.EQ.LODPON3)  PII(N,I)=PLOAD3
      IF(II.EQ.LODPON4)  PII(N,I)=PLOAD4
      IF(II.EQ.LODPON5)  PII(N,I)=PLOAD5
      IF(II.EQ.LODPON6)  PII(N,I)=PLOAD6
      IF(II.NE.LODPON1.AND.II.NE.LODPON2.AND.II.NE.LODPON3.
     +AND.II.NE.LODPON4.AND.II.NE.LODPON5.AND.II.NE.LODPON6)PII(N,I)=0.0
1701  CONTINUE
801   CONTINUE
      DO 375 N=1,NUMNP
      DO 375 I=1,3
375   PI(N,I)=PII(N,I)
      GO TO 1001
520   IF(NUMITER.GT.MAX1)  GO TO 900
      NUMITER=NUMITER+1
      WRITE(61,9039) NUMITER,NCOUNT,DTOT(LODPON1),DTOT(LODPON2)
     +,DTOT(LODPON3),DTOT(LODPON4),DTOT(LODPON5),DTOT(LODPON6)
9039  FORMAT(/,10X,*NUMITER=*,I5,5X,*NCOUNT=*,I5,/
     +,10X,*DTOT(LODPON1)=*,F20.10,10X,*DTOT(LODPON2)=*,F20.10,/
     *,10X,*DTOT(LODPON3)=*,F20.10,10X,*DTOT(LODPON4)=*,F20.10/
     *,10X,*DTOT(LODPON5)=*,F20.10,10X,*DTOT(LODPON6)=*,F20.10/)
      DO 573 N=1,NUMNP
      DO 573 I=1,3
573   PI(N,I)=PII(N,I)
      GO TO 1001
900   CONTINUE
10    FORMAT(E21.15)
      END
C
C
C
C
C
C
C
      SUBROUTINE NADDATA(IGOPTIN)
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(21,3),X(21),Y(21)
      WRITE(61,2010)
      WRITE(61,2015)
      IF(IGOPTIN.EQ.0)  GO TO 100
      IF(IGOPTIN.EQ.2)  GO TO 202
      RE AD(60,10)  ALFZERO,RADIUS
      WRITE(61,20)  ALFZERO,RADIUS
      ALFINC=ALFZERO/NE
      DO 201  I=1,NUMNP
      X(I)=RADIUS*SIN((-ALFZERO/2.)+(I-1)*ALFINC)
      Y(I)=SQRT(RADIUS**2-X(I)**2)
201   CONTINUE
      GO TO 204
202   READ(60,10)   RISE,SPAN
      WRITE(61,30)  RISE,SPAN
      DO 203  I=1,NUMNP
      X(I)=-SPAN/2.+(I-1)*SPAN/NE
      Y(I)=RISE-4.*RISE*X(I)**2/(SPAN**2)
203   CONTINUE
204   CONTINUE
90    READ(60,1001)N,(IA(N,I),I=1,3)
      WRITE(61,2020)N,(IA(N,I),I=1,3),X(N),Y(N)
      IF(N.NE.NUMNP)  GO TO 90
      GO TO 101
100   READ(60,1000) N,(IA(N,I),I=1,3),X(N),Y(N)
      WRITE(61,2020)N,(IA(N,I),I=1,3),X(N),Y(N)
      IF(N.NE.NUMNP) GO TO 100
101   NEQ=0
      DO 125 N=1,NUMNP
      DO 120 I=1,3
      IF(IA(N,I).NE.1) GO TO 105
      IA(N,I)=0
      GO TO 120
105   IA(N,I)=-1
      NEQ=NEQ+1
      IA(N,I)=NEQ
120   CONTINUE
125   CONTINUE
      WRITE(61,2030)
      WRITE(61,2040)
      WRITE(61,2050)(N,(IA(N,I),I=1,3),N=1,NUMNP)
      WRITE(61,2060) NEQ
      RETURN
C
C
10    FORMAT(2F15.10)
20    FORMAT(//,10X,*ALFA=*,F15.10,10X,*RADIUS=*,F15.10,//)
30    FORMAT(//,10X,*RISE=*,F15.10,10X,*SPAN=*,F15.10,//)
1000  FORMAT(I5,3I3,2F10.5)
1001  FORMAT(I5,3I3)
2010  FORMAT(/,10X,*INPUT NODAL DATA*,/)
2015  FORMAT(7H   NODE,3X,36HNODAL POINT BOUNDARY CONDITION CODES,5X,
     +23HNODAL POINT COORDINATES/7H NUMBER,11X,7HIA(N,I)/
     +15X,1HX,4X,1HY,4X,2HTZ,20X,4HX(N),8X,4HY(N))
2020  FORMAT(I5,6X,3I5,14X,2F12.3)
2030  FORMAT(///,10X,*GENERATED NODAL DATA*)
2040  FORMAT(7H   NODE,8X,*EQUATION NUMBERS*,22X,
     +/*   NUMBER*,11X,*IA(N,I)*,33X,/
     +15X,1HX,4X,1HY,4X,2HTX)
2050  FORMAT(I5,6X,3I5)
```

```
2060  FORMAT(*-*,4HNEQ=,I3)
      END
C
C
C
C
C
C
C
      SUBROUTINE ELEMENT
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/3/IA(21,3),X(21),Y(21)
      COMMON/5/E(1),NODEI(20),NODEJ(20),A(20),IXX(20),L(1,20),SXX(20)
      REAL IXX,LE
      READ(60,1010)  NUMEL,E(1)
      WRITE(61,2021)
      WRITE(61,2020) NUMEL,E(1)
      K=0
      WRITE(61,2025)
2025  FORMAT(//,3X,*ELEMENT*,3X,*NODEI(M)*,3X,*NODEJ(M)*,12X,*A(M)*,10X,
     +*IXX(M)*,8X,*SXX(M)*,/)
105   READ(60,1020) M,NODEI(M),NODEJ(M),A(M),IXX(M),SXX(M)
      WRITE(61,2022)M,NODEI(M),NODEJ(M),A(M),IXX(M),SXX(M)
      K=K+1
      L(1,K)=M
      IF(K.NE.NUMEL) GO TO 105
      RETURN
1010  FORMAT(I5,E10.2)
1020  FORMAT(3I5,3F10.6)
2020  FORMAT(I6,8X,2E17.6)
2021  FORMAT(/,3X,*NO OF ELEMENTS*,10X,*E(1)*,/)
2022  FORMAT(I6,6X,I5,6X,I5,6X,3F15.6)
      END
C
C
C
C
C
C
C
      SUBROUTINE BAND
C     ******************************************************************
C
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(21,3),X(21),Y(21)
      COMMON/5/E(1),NODEI(20),NODEJ(20),A(20),IXX(20),L(1,20),SXX(20)
      MBAND=0
      ICONTRL=0
      DO 900 M=1,NE
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 800 I=1,3
      IF(ICONTRL.EQ.1) GO TO 1001
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0) GO TO 199
1001  CONTINUE
      IF(IA(NI,I).LE.0) GO TO 800
      N1=IA(NI,I)
      GO TO 99
199   ICONTRL=1
      N1=0
99    DO 700 J=1,3
      IF(ICONTRL.EQ.1) GO TO 1002
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0) GO TO 399
1002  CONTINUE
      GO TO 499
399   ICONTRL=1
      MB=N1
      GO TO 299
499   IF(IA(NJ,J).LE.0) GO TO 700
      N2=IA(NJ,J)
      MB=IABS(N2-N1)
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0) GO TO 299
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0) GO TO 299
      MB=MB+1
299   IF(MB.GT.MBAND) MBAND=MB
      IF(IA(NJ,1).LE.0.AND.IA(NJ,2).LE.0.AND.IA(NJ,3).LE.0) GO TO 800
700   CONTINUE
      IF(IA(NI,1).LE.0.AND.IA(NI,2).LE.0.AND.IA(NI,3).LE.0) GO TO 900
```

```
800   CONTINUE
900   CONTINUE
      WRITE(61,2000) MBAND
      RETURN
2000  FORMAT(/////,*SEMIBANDWIDTH MBAND=*,I3,///)
      END
C
C
C
C
C
C
C
      SUBROUTINE BEAM
C     *****************************************************************
C
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/3/IA(21,3),X(21),Y(21)
      COMMON/2/NEQ,MBAND
      COMMON/4/SE(6,6),STSTAR(3,3),STSTARM(3,3,20),TSMAL(3,3,20),
     *TTSML(3,3)
      COMMON/5/E(1),NODEI(20),NODEJ(20),A(20),IXX(20),L(1,20),SXX(20)
      COMMON/8/PI(21,3),PII(21,3),R(65)
      COMMON/9/S(65,12),IDET,ES(3,20)
      COMMON/10/D(65),TETO(20),ITERCHK,PROTYPE
      REAL IXX,LE
      INTEGER PROTYPE
      K=0
105   CONTINUE
      K=K+1
      M=L(1,K)
      NI=NODEI(M)
      NJ=NODEJ(M)
      AA=(X(NJ)-X(NI))**2
      B=(Y(NJ)-Y(NI))**2
      LE(M)=SQRT(AA+B)
      C=(X(NJ)-X(NI))/LE(M)
      Z=(Y(NJ)-Y(NI))/LE(M)
      PIE=3.14159265358979
      IF(C.GE.0..AND.Z.GE.0.)    TETO(M)=ASIN(SQRT(B)/LE(M))
      IF(C.LE.0..AND.Z.GE.0.)    TETO(M)=PIE-ASIN(SQRT(B)/LE(M))
      IF(C.LE.0..AND.Z.LE.0.)    TETO(M)=PIE+ASIN(SQRT(B)/LE(M))
      IF(C.GE.0..AND.Z.LT.0.)    TETO(M)=2.*PIE-ASIN(SQRT(B)/LE(M))
      SE(1,1)=SE(4,4)=E(1)*A(M)*C**2/LE(M)+12.*E(1)*IXX(M)*Z**2/LE(M)**3
      SE(1,4)=-SE(1,1)
      SE(1,2)=SE(4,5)=(E(1)*A(M)/LE(M)-12.*E(1)*IXX(M)/LE(M)**3)*C*Z
      SE(2,4)=SE(1,5)=-SE(1,2)
      SE(2,2)=SE(5,5)=E(1)*A(M)*Z**2/LE(M)+12.*E(1)*IXX(M)*C**2/LE(M)**3
      SE(2,5)=-SE(2,2)
      SE(4,6)=SE(3,4)=6.*E(1)*IXX(M)*Z/LE(M)**2
      SE(1,6)=SE(1,3)=-SE(3,4)
      SE(2,6)=SE(2,3)=6.*E(1)*IXX(M)*C/LE(M)**2
      SE(3,5)=SE(5,6)=-SE(2,3)
      SE(6,6)=SE(3,3)=4.*E(1)*IXX(M)/LE(M)
      SE(3,6)=SE(6,6)/2.
      IF(ITERCHK.NE.0) GO TO 8009
      IF(PROTYPE.EQ.2) GO TO 8009
      STSTARM(1,1,M)=STSTARM(2,2,M)=4.*E(1)*IXX(M)/LE(M)
      STSTARM(1,2,M)=STSTARM(2,1,M)=2.*E(1)*IXX(M)/LE(M)
      STSTARM(3,3,M)=E(1)*A(M)/LE(M)
      STSTARM(1,3,M)=STSTARM(2,3,M)=STSTARM(3,1,M)=STSTARM(3,2,M)=0.0
8009  CONTINUE
      IF(ITERCHK.NE.0) GO TO 8010
      IF(PROTYPE.EQ.1) GO TO 8010
      TSMAL(1,1,M)=TSMAL(2,2,M)=4.*E(1)*IXX(M)/LE(M)
      TSMAL(1,2,M)=TSMAL(2,1,M)=2.*E(1)*IXX(M)/LE(M)
      TSMAL(3,3,M)=E(1)*A(M)*LE(M)
      TSMAL(1,3,M)=TSMAL(2,3,M)=TSMAL(3,1,M)=TSMAL(3,2,M)=0.0
8010  CONTINUE
      DO 210 I=1,6
      DO 210 J=I,6
210   SE(J,I)=SE(I,J)
      IF(ICAL2.EQ.1) GO TO 9024
      WRITE(61,9025) M
9025  FORMAT(/,10X,*M=*,I5,10X,*SE(I,J),LINEAR*,/)
      WRITE(61,503) ((SE(I,J),J=1,6),I=1,6)
9024  CONTINUE
      WRITE(1,10) ((SE(I,J),J=1,6),I=1,6)
      CALL ASEMBLE(M)
      IF(K.NE.NUMEL) GO TO 105
      REWIND 1
      WRITE(2,10) ((S(I,J),J=1,MBAND),I=1,NEQ)
      REWIND 2
      IF(ICAL2.EQ.1) GO TO 9026
      WRITE(61,9027)
9027  FORMAT(/,10X,*S(I,J) LINEAR*,/)
      WRITE(61,503) ((S(I,J),J=1,MBAND),I=1,NEQ)
9026  CONTINUE
      RETURN
10    FORMAT(E21.15)
503   FORMAT(/1X,6F20.10)
      END
C
C
C
C
C
C
C
      SUBROUTINE ASEMBLE(M)
C     ****************************************************************
C
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(21,3),X(21),Y(21)
      COMMON/4/SE(6,6),STSTAR(3,3),STSTARM(3,3,20),TSMAL(3,3,20),
     +TTSML(3,3)
      COMMON/5/E(1),NODEI(20),NODEJ(20),A(20),IXX(20),L(1,20),SXX(20)
      COMMON/8/PI(21,3),PII(21,3),R(65)
      COMMON/9/S(65,12),IDET,ES(3,20)
      IF(IPAR.NE.1) GO TO 90
      DO 5 I=1,NEQ
5     R(I)=0.0
      DO 80 N=1,NUMNP
      DO 70 I=1,3
      IF(IA(N,I)) 70,70,10
10    II=IA(N,I)
      R(II)=PI(N,I)
70    CONTINUE
80    CONTINUE
      RETURN
90    NI=NODEI(M)
      NJ=NODEJ(M)
      DO 165 K1=1,2
      IF(K1.EQ.1) NP=NI
      IF(K1.EQ.2) NP=NJ
      DO 160 I=1,3
      IF(IA(NP,I)) 115,160,100
100   II=IA(NP,I)
115   CONTINUE
      DO 155 K2=1,2
      IF(K2.EQ.1) ND=NI
      IF(K2.EQ.2) ND=NJ
      DO 150 J=1,3
      IF(IA(ND,J)) 145,150,120
120   JJ=IA(ND,J)
145   CONTINUE
      IF(JJ.LT.II) GO TO 150
      IF(K1.EQ.1) IE=I
      IF(K1.EQ.2) IE=I+3
      IF(K2.EQ.1) JE=J
      IF(K2.EQ.2) JE=J+3
      JJ=JJ-II+1
      S(II,JJ)=S(II,JJ)+SE(IE,JE)
      IF(JJ.GT.MBAND) MBAND=JJ
150   CONTINUE
155   CONTINUE
160   CONTINUE
165   CONTINUE
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE LINSOLN
C     ****************************************************************
C
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/2/NEQ,MBAND
      COMMON/8/PI(21,3),PII(21,3),R(65)
      COMMON/9/S(65,12),IDET,ES(3,20)
      COMMON/10/D(65),TETO(20),ITERCHK,PROTYPE
      DO 110 I=1,NEQ
110   D(I)=R(I)
      IF(ICAL3.EQ.0) WRITE(61,2020)
      IF(ICAL3.EQ.0) WRITE(61,2010) (I,D(I),I=1,NEQ)
      DO 790 N=1,NEQ
      DO 780 L=2,MBAND
      IF(S(N,L).EQ.0.) GO TO 780
      I=N+L-1
      C=S(N,L)/S(N,1)
      J=0
      DO 750 K=L,MBAND
      J=J+1
750   S(I,J)=S(I,J)-C*S(N,K)
      S(N,L)=C
780   CONTINUE
790   CONTINUE
      DO 830 N=1,NEQ
      DO 820 L=2,MBAND
      IF(S(N,L).EQ.0.) GO TO 820
      I=N+L-1
      D(I)=D(I)-S(N,L)*D(N)
820   CONTINUE
830   D(N)=D(N)/S(N,1)
      DO 860 M=2,NEQ
      N=NEQ+1-M
      DO 850 L=2,MBAND
      IF(S(N,L).EQ.0.) GO TO 850
      K=N+L-1
      D(N)=D(N)-S(N,L)*D(K)
850   CONTINUE
860   CONTINUE
      IF(ICAL1.EQ.0) WRITE(61,2000)
      IF(ICAL1.EQ.0) WRITE(61,2010) (I,D(I),I=1,NEQ)
      RETURN
2000  FORMAT(/,10X,*DISPLACEMENT FROM LINEAR SOLUTION*,/)
2010  FORMAT(/,10X,*D(*,I3,*)=*,E21.15)
2020  FORMAT(/,10X,*LOAD VECTOR FOR LINEAR SOLUTION*,/)
      END
C
C
C
C
C
C
C
      SUBROUTINE IDENT
C     ****************************************************************
C
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/2/NEQ,MBAND
      COMMON/3/IA(21,3),X(21),Y(21)
      COMMON/5/E(1),NODEI(20),NODEJ(20),A(20),IXX(20),L(1,20),SYX(20)
      COMMON/10/D(65),TETO(20),ITERCHK,PROTYPE
      COMMON/11/W(21,3),DN(6,1),WTOT(21,3)
      DO 230 K=1,NUMEL
      M=L(1,K)
      NI=NODEI(M)
      NJ=NODEJ(M)
      DO 230 K1=1,2
      IF(K1.EQ.1) NP=NI
      IF(K1.EQ.2) NP=NJ
      DO 220 I=1,3
      IF(IA(NP,I)) 220,155,150
150   NL=IA(NP,I)
      W(NP,I)=D(NL)
      GO TO 220
155   W(NP,I)=0.0
220   CONTINUE
230   CONTINUE
      RETURN
      END
C
C
C
C
C
C
```

```
C
      SUBROUTINE MULT(M,K,N,A,B,C)
C     ****************************************************************
C
      DIMENSION A(M,K),B(K,N),C(M,N)
      DO 100 I=1,M
      DO 100 J=1,N
      C(I,J)=0.0
      DO 100 MM=1,K
100   C(I,J)=C(I,J)+A(I,MM)*B(MM,J)
      RETURN
      END
C
C
C
C
C
C
C
      FUNCTION DET1(SCALE)
C     ****************************************************************
C
      COMMON/2/NEQ,MBAND
      COMMON/9/S(65,12),IDET,ES(3,20)
      IF(IDET.EQ.1) GO TO 250
      DO 390 LN=1,NEQ
      DO 380 LL=2,MBAND
      IF(S(LN,LL).EQ.0.) GO TO 380
      I=LN+LL-1
      C=S(LN,LL)/S(LN,1)
      J=0
      DO 350 KK=LL,MBAND
      J=J+1
350   S(I,J)=S(I,J)-C*S(LN,KK)
      S(LN,LL)=C
380   CONTINUE
390   CONTINUE
250   CONTINUE
      DT=1.
      DO 400 I=1,NEQ
      DT=DT*S(I,1)/SCALE
400   CONTINUE
      DET1=DT
      RETURN
      END
C
C
C
C
C
C
C
      SUBROUTINE STRESS
C     ****************************************************************
C
      DIMENSION SIGMA(20),STRAIN(20)
      COMMON/1/NE,NUMNP,LE(20),NUMEL,IPAR,ICAL1,ICAL2,ICAL3
      COMMON/5/E(1),NODEI(20),NODEJ(20),A(20),IXY(20),L(1,20),SXX(20)
      COMMON/9/S(65,12),IDET,ES(3,20)
      COMMON/10/D(65),TETO(20),ITERCHK,PROTYPE
      INTEGER PROTYPE
      DO 100 M=1,NUMEL
      WRITE(61,9012)
9012  FORMAT(/,10X,*M*,10X,*S1*,10X,*S2*,10X,*S3*,/)
      WRITE(61,9013) M,ES(1,M),ES(2,M),ES(3,M)
9013  FORMAT(/,10X,I5,3F20.10)
      IF(ABS(ES(1,M)).GT.ABS(ES(2,M))) RMMAX=ABS(ES(1,M))
      IF(ABS(ES(1,M)).LT.ABS(ES(2,M))) RMMAX=ABS(ES(2,M))
      IF(PROTYPE.EQ.1) SIGMA(M)=ABS(ES(3,M)/A(M))+RMMAX/SXX(M)
      IF(PROTYPE.EQ.2) SIGMA(M)=ABS(ES(3,M)/(LE(M)*A(M)))+RMMAX/SXX(M)
      STRAIN(M)=SIGMA(M)/E(1)
      WRITE(61,110) M,STRAIN(M)
110   FORMAT(/,10X,*FOR ELEMENT NO *,I2,10X,*STRAIN IS*,F20.10)
100   CONTINUE
      RETURN
      END
```

```
  08:36:57 07/13/81    SL09017     1373 LINES PRINT.
   19 PAGES PRINT.  COST AT RG3 IS    $   1.81
```