GEOMETRICALLY NONLINEAR FINITE ELEMENT ANALYSIS

OF SPACE FRAMES

by

Jih Jih Jau

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Civil Engineering

APPROVED:

S. M. Holzer, Chairman

R. M. Barker

T. Kuppusamy

M. P. Singh

A. E. Somers

February, 1985

Blacksburg, Virginia

Geometrically Nonlinear Finite Element Analysis

of Space Frames

by

Jih Jih Jau

(ABSTRACT)

The displacement method of the finite element is adopted. Both the updated Lagrangian formulation and total Lagrangian formulation of a three-dimensional beam element is employed for large displacement and large rotation, but small strain analysis.

A beam-column element or finite element can be used to model geometrically nonlinear behavior of space frames. The two element models are compared on the basis of their efficiency, accuracy, economy and limitations.

An iterative approach, either Newton-Raphson iteration or modified Riks/Wempner iteration, is employed to trace the nonlinear equilibrium path. The latter can be used to perform postbuckling anaylsis.

ACKNOWLEDGEMENTS

The author would like to express his sincere appreciation to Dr. S. M. Holzer for his guidance, encouragement and infinite patience. The author is also very much obliged to Dr. R. M. Barker, Dr. T. Kuppusamy, Dr. M. P. Singh and Dr. A. E. Somers for reviewing this dissertation, giving advice and serving on the committee.

Finally, the author gives deep gratitude to his beloved family for their complete devotion and encouragement.

TABLE OF CONTENTS

Appendix

page

LIST OF FIGURES

Chapter I

INTRODUCTION

1.1 PURPOSE AND SCOPE

The primary purpose of this dissertation is to implement and compare three formulations in geometrically nonlinear finite element analysis of space frames for static analysis

1. The updated Lagrangian (U.L.) formulation: all variables are referred to the current deformed configuration at time t.
2. The total Lagrangian (T.L.) formulation: all variables are referred to the initial undeformed configuration at time 0.
3. The convected coordinate formulation: all variables are referred to the new incremented configuration at time $t+\Delta t$. This formulation utilizes a set of moving rigid convected coordinates that rotate and translate with the element, but do not deform with the element.

To derive the element deformation displacements, four types of coordinate systems are defined (section 3.2). The element may undergo large translations and large rotations, but is restricted to small strains which means the cross sectional area does not change.

The second purpose is to compare two models of geometrically nonlinear space frames. To predict the structural response accurately, it is necessary to select the proper mathematical models, either finite element model or a beam-column model can be used. The finite element model is formulated by the principle of virtual work with Lagrangian and Hermitian interpolation functions used for discretization. The beam-column model, developed by Oran [1], [2], [3], is based on the conventional beam-column theory.

The solution procedure is iterative as well as incremental. The Newton-Raphson method and the modified Riks/Wempner method [27] are employed. The Newton-Raphson method cannot be used to trace equilibrium path beyond the limit point; for this reason the modified Riks/Wempner method was developed.

The final purpose is to develop a computer program for geometrically nonlinear static analysis of space frames. The U.L. formulation was implemented in this program. Several test examples were investigated using the computer program. The results compared well with available data in the literature.

1.2 SURVEY OF LITERATURE

The analysis of geometrically nonlinear framed structures has attracted considerable attention during the last two decades. An early paper by Connor [16] presented a nonlinear formulation for a rigid-jointed space frame with small rotations subjected to loads applied only at the joints. Oran [1], [2], [3] derived a tangent stiffness matrix for elastic frame structures based on the conventional beam-column theory [23] in which small relative deformations of the members were assumed, however the rotations and translations of the joints were considered to be arbitrarily large. Belytschko and coworkers [5],[17] employed a convected coordinate system in which the deformation displacements were seperated from the rigid body motion, and node orientations were described by unit vectors that only three components of two unit vectors were stored. Mikkola [4] combined Oran's and Belytschko's formulations in which joint displacements and element deformations were described, and derived the tangent stiffness matrix in the different form.

A large displacement problem in structural analysis can be analyzed in three types of formulations: U.L., T.L. and convected coordinates formulations.

References [25],[26],[39],[41] adopted the U.L. formulation: Murray and Wilson [25],[41] investigated the response of thin elastic plates and employed in-plane displacement functions and plate bending displacement functions which maintained boundary compatibility for the in-plane and bending problems, respectively, but violated boundary compatibility when superimposed in the large deflection problem. Yang [26] applied a linearized midpoint tangent incremental approach to predict the nonlinear equilibrium path. Chu and coworkers [39] developed the constant load method to determine buckling loads of space frames based on the large deflection theory in which the iteration may start at any load level and the stiffness matrix developed for small deflection theory could be used directly.

References [7],[12],[19],[20],[32] adopted the T.L. formulation: Rajasekaran and Murray [7] showed that the equilibrium equation and the linear incremental equilibrium equation did not necessarily follow from the total potential energy in the form introduced by Mallett and Marcal [8], and derived the particular forms of the incremental stiffness matrices. Hibbitt and coworkers [12] developed a large displacement, large strain formulation by introducing an additional initial load stiffness matrix into the large displacement, small strain formulation, and took the material to be elastic-plastic. Remseth [19],[32] presented nonlinear static and dynamic analysis of space frames in which the node rotations were limited to 12 to 15 degree, and higher order axial interpolation polynomials were included in order to obtain an appropriate coupling between axial forces and bending. Wood and Zienkiewicz [20] presented the geometrically nonlinear analysis of the two dimensional inplane structures, e.g. beams,frames and arches, which omitted midside nodes in the "thickness" direction, thereby reducing the number of degrees of freedom, and employed a paralinear isoparametric element.

Cook [33] gave an basic introduction to the geometric nonlinear problem. Bathe [9],[10],[38] presented an U.L. and a T.L. formulation, derived from the continuum mechanics, of space frame element for large displacement analysis. Tang [18] described these three formulations and explained that in the convected coordinate formulation an incremental concept was impracticable, thus the tangent stiffness matrix was difficult to be established.

Wood and Schrefler [6] gave a correlation between the so-called N-notation and the B-notation of the T.L. formulation of geometrically nonlinear problems. Mallet [8] and Rajasekaran [7] adopted the N-notation, Zienkiewicz and co-workers [31], [20] adopted the B-notation.

Katzenberger [48] derived the secant stiffness matrices for the plane frame element from which element forces can be obtained. Butler [29] compared two models for geometrically nonlinear finite element analysis of plane frames.

To trace nonlinear equilibrium paths into the postbuckling range, Holzer [27] employed the modified Riks/Wempner method. Bathe and Cimento [49] described the practical procedures for the incremental solution of nonlinear finite element equations and proposed specific ways to measure convergence.

Papadrakakis [21] employed the vector iteration methods to study the post-buckling behavior of spatial structures in which there is no need to compute or formulate the tangent stiffness matrix.

Chapter II

UPDATED AND TOTAL LAGRANGIAN FORMULATIONS IN GEOMATRICALLY NONLINEAR FINITE ELEMENT ANALYSIS

2.1 INTRODUCTION

Large displacement analysis may be formulated in three types of coordinate systems (Fig. 2.1):

1. The total Lagranginan (T.L.) formulation which refers to the initial undeformed equilibrium configuration ${}^{0}C$ at time 0.
2. The updated Lagrangian (U.L.) formulation which refers to the current deformed equilibrium configuration ${}^{t}C$ at time t.
3. The convected coordinate (Eulerian) formulation which refers to the new incremented configuration ${}^{t+\Delta t}C$ at time t+Δt.

In Fig. 2.1 ${}^{0}X_i$, ${}^{t}X_i$, ${}^{t+\Delta t}X_i$ are the global coordinate systems in the configuration at time 0, t, t+Δt respectively, i=1, 2, 3; ${}^{0}x_i$, ${}^{t}x_i$, ${}^{t+\Delta t}x_i$ are the local coordinate systems at time 0, t, t+Δt respectively; ${}^{t}u_i$, ${}^{t+\Delta t}u_i$ are the displacement components from initial position at time 0 to configuration at time t, t+Δt respectively.

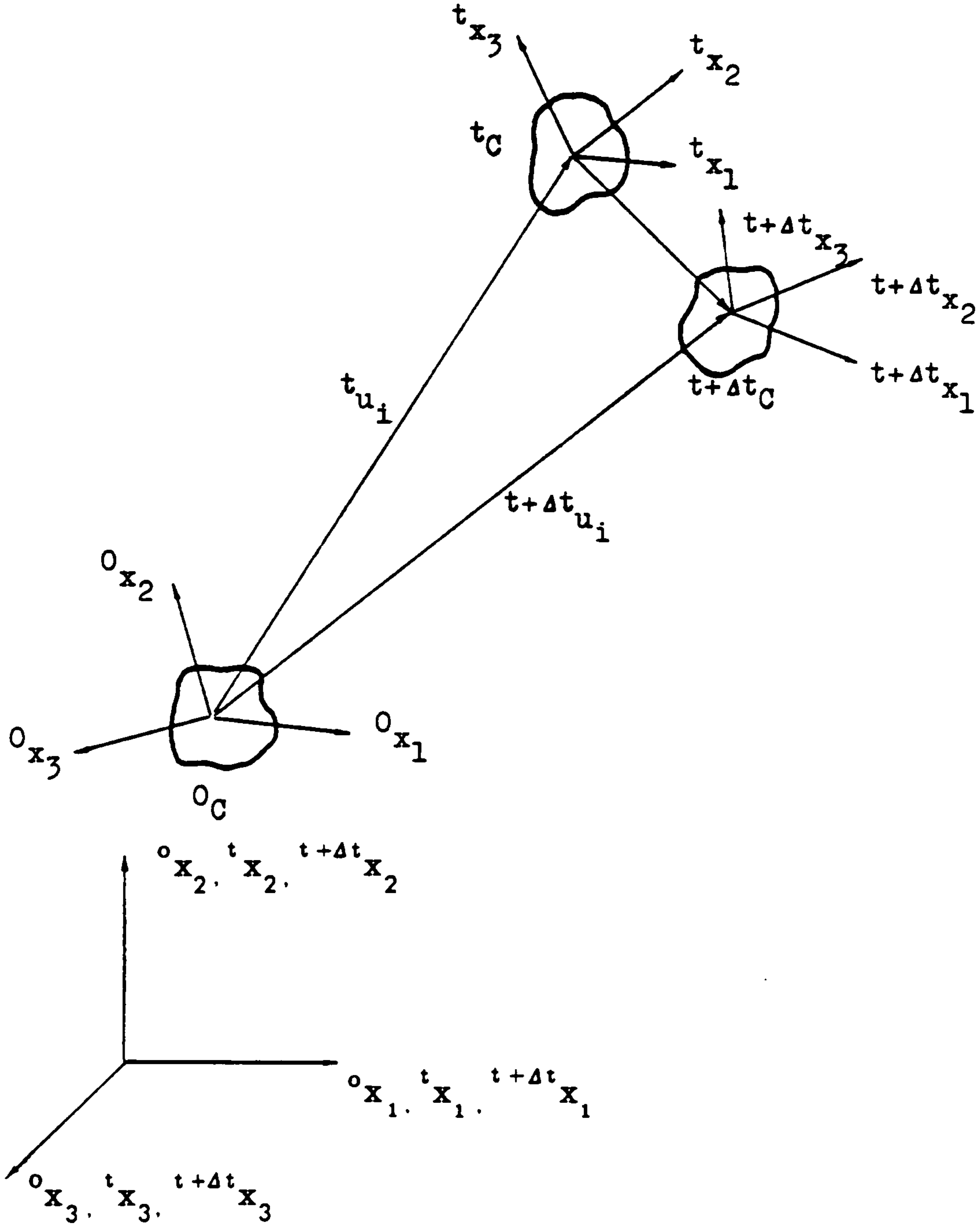


Fig. 2.1 Motion of an Element in Cartesian Coordinate System

Tensor notation is adopted in incremental U.L. and T.L. continuum mechanics formulation. The Green-Lagrange strain tensor used with the 2nd Piola-Kirchhoff stress tensor is defined as

$$ {}_{0}^{t}\varepsilon_{ij} = 1/2\left({}_{0}^{t}u_{i,j} + {}_{0}^{t}u_{j,i} + {}_{0}^{t}u_{k,i}\ {}_{0}^{t}u_{k,j}\right) \qquad (2.0)$$

in which ${}_{0}^{t}u_{i,j} = \partial^{t}u_{i} / \partial^{0}x_{j}$. Using the principle of virtual work, we can express the equilibrium equation in the configuration at time t+Δt as [9], [10], [38]

$$\int_{{}^{t}V} {}_{t}^{t+\Delta t}S_{ij}\ \delta\, {}_{t}^{t+\Delta t}\varepsilon_{ij}\ {}^{t}dV = {}^{t+\Delta t}R$$

in U.L. formulation (2.1)

$$\int_{{}^{0}V} {}_{0}^{t+\Delta t}S_{ij}\ \delta\, {}_{0}^{t+\Delta t}\varepsilon_{ij}\ {}^{0}dV = {}^{t+\Delta t}R$$

in T.L. formulation (2.2)

where a left superscript indicates in which configuration the quantity occurs; a left subscript indicates the configuration to which the quantity is referred; a right subscript indicates the components of a tensor.

${}_{t}^{t+\Delta t}S_{ij}$, ${}_{0}^{t+\Delta t}S_{ij}$ = component of 2nd Piola-Kirchhoff stress tensor

δ = 'variation in '

${}_{t}^{t+\Delta t}\varepsilon_{ij}$, ${}_{0}^{t+\Delta t}\varepsilon_{ij}$ = component of Green-Lagrange strain tensor

${}^{t+\Delta t}R$ = external virtual work corresponding to configuration at time $t+\Delta t$

Since Eqs. (2.1) and (2.2) are nonlinear in the displacement increments u_i, i.e. $u_i = {}^{t+\Delta t}u_i - {}^{t}u_i$, they cannot be solved directly. Approximate solutions can be obtained by referring all variables to a previously calculated known equilibrium configuration, and linearizing the resulting equation; then the solution can be improved by iteration. Assuming

$${}_t\varepsilon_{ij} = {}_te_{ij} \qquad \text{in U.L. formulation} \tag{2.3}$$

$${}_0\varepsilon_{ij} = {}_0e_{ij} \qquad \text{in T.L. formulation} \tag{2.4}$$

where

${}_t\varepsilon_{ij}$, ${}_0\varepsilon_{ij}$ = component of strain increment tensor (Green-Lagrange) referred to configuration at time t and 0 respectively

${}_te_{ij}$, ${}_0e_{ij}$ = linear part of strain increment ${}_t\varepsilon_{ij}$, ${}_0\varepsilon_{ij}$ respectively

In addition, since in Eqs. (2.1) and (2.2) the 2nd Piola-Kirchhoff stresses and Green-Lagrange strains are unknown, they must be decomposed into

In U.L. formulation:

$${}^{t+\Delta t}_{\ \ t}S_{ij} = {}^{t}\tau_{ij} + {}_tS_{ij} \quad (\text{note: } {}^{t}_{t}S_{ij} = {}^{t}\tau_{ij}) \tag{2.5}$$

$$ {}^{t+\Delta t}_{t}\varepsilon_{ij} = {}_{t}\varepsilon_{ij} \tag{2.6} $$

In T.L. formulation:

$$ {}^{t+\Delta t}_{0}S_{ij} = {}^{t}_{0}S_{ij} + {}_{0}S_{ij} \tag{2.7} $$

$$ {}^{t+\Delta t}_{0}\varepsilon_{ij} = {}^{t}_{0}\varepsilon_{ij} + {}_{0}\varepsilon_{ij} \tag{2.8} $$

where

$^{t}\tau_{ij} = {}^{t}_{t}S_{ij}$ = component of Cauchy stress tensor

${}_{t}S_{ij}$, ${}_{0}S_{ij}$ = component of 2nd Piola-Kirchhoff stress increment at time t

${}_{t}\varepsilon_{ij}$, ${}_{0}\varepsilon_{ij}$ = component of strain increment tensor (Green-Lagrange)

2.2 INCREMENTAL EQUILIBRIUM EQUATION IN U.L. FORMULATION

2.2.1 Incremental U.L. Continuum Mechanics Formulation

In the U.L. formualtion all variables, i.e. displacement, area, volume, stress, strain, differentiations and integrations, etc., refer to the current deformed configuration ^{t}C at time t (Fig. 2.1).

Since the Green-Lagrange strain tensor ${}^{t+\Delta t}_{t}\varepsilon_{ij}$ uses the displacements from the configuration at time t to the configuration at time t+Δt, from Eq. (2.6) we have

$$\delta\,{}^{t+\Delta t}_{\ \ t}\varepsilon_{ij} = \delta\,{}_t\varepsilon_{ij} \tag{2.9}$$

Substituting Eqs. (2.5), (2.9) into Eq. (2.1) yields

$$\int_{{}^tV} {}^t\tau_{ij}\ \delta\,{}_t\varepsilon_{ij}\ {}^t dV + \int_{{}^tV} {}_tS_{ij}\ \delta\,{}_t\varepsilon_{ij}\ {}^t dV = {}^{t+\Delta t}R \tag{2.10}$$

The strain increment components can be separated into linear and nonlinear parts

$$ {}_t\varepsilon_{ij} = {}_te_{ij} + {}_t\eta_{ij} \tag{2.11}$$

where

$$ {}_te_{ij} = 1/2\ ({}_tu_{i,j} + {}_tu_{j,i}) \tag{2.12}$$

$$ {}_t\eta_{ij} = 1/2\ {}_tu_{k,i}\ {}_tu_{k,j} \tag{2.13}$$

$$ {}_tu_{i,j} = \partial u_i \,/\, \partial\,{}^tx_j \tag{2.14}$$

$$ u_i = {}^{t+\Delta t}u_i - {}^tu_i\ ; \qquad i=1,2,3 \tag{2.15}$$

${}_t\eta_{ij}$ is the nonlinear part of strain increment ${}_t\varepsilon_{ij}$; ${}_tu_{i,j}$ is the derivative of displacement increment with respect to local coordinate ${}^tx_j$; u_i are the increments in the displacements from time t to time t+Δt; ${}^tu_i$, ${}^{t+\Delta t}u_i$ are the displacement components in the local coordinate system from the initial configuration at time 0 to the deformed configuration at time t and time t+Δt respectively.

$$ {}^{t}u_i = {}^{t}x_i - {}^{0}x_i \tag{2.15a} $$

$$ {}^{t+\Delta t}u_i = {}^{t+\Delta t}x_i - {}^{0}x_i \tag{2.15b} $$

where ${}^{t}u_i$ and ${}^{t+\Delta t}u_i$ are the functions of position coordinates (x_1, x_2, x_3) respectively.
The constitutive law is [10]

$$ {}_{t}S_{ij} = {}_{t}C_{ijrs}\ {}_{t}\varepsilon_{rs} \tag{2.16} $$

where ${}_{t}C_{ijrs}$ is the component of incremental material property tensor at time t referred to the configuration at time t.
Using Eqs. (2.11), (2.16), Eq. (2.10) can be transformed to

$$ \int_{{}^{t}V} {}_{t}C_{ijrs}\ {}_{t}\varepsilon_{rs}\ \delta_{t}\varepsilon_{ij}\ {}^{t}dV + \int_{{}^{t}V} {}^{t}\tau_{ij}\ \delta_{t}\eta_{ij}\ {}^{t}dV = {}^{t+\Delta t}R - \int_{{}^{t}V} {}^{t}\tau_{ij}\ \delta_{t}e_{ij}\ {}^{t}dV \tag{2.17} $$

Eq. (2.17) is nonlinear in the incremental displacements u_i, and it can be linearized by using the approximations

$$ \delta_{t}\varepsilon_{ij} = \delta_{t}e_{ij} \tag{2.18} $$

$$ {}_{t}S_{ij} = {}_{t}C_{ijrs}\ {}_{t}e_{rs} \tag{2.19} $$

Therefore Eq. (2.17) becomes

$$ \int_{{}^{t}V} {}_{t}C_{ijrs}\ {}_{t}e_{rs}\ \delta_{t}e_{ij}\ {}^{t}dV + \int_{{}^{t}V} {}^{t}\tau_{ij}\ \delta_{t}\eta_{ij}\ {}^{t}dV = {}^{t+\Delta t}R - \int_{{}^{t}V} {}^{t}\tau_{ij}\ \delta_{t}e_{ij}\ {}^{t}dV \tag{2.20} $$

Eq. (2.20) is the incremental equilibrium equation of a deformed element, linear in the incremental displacements u_i, corresponding to the local coordinate system.

In three dimensional beam element for our problem, small deformation and uniaxial state of strain (i.e. ${}_t\varepsilon_{11}$ only) are assumed, and torsion is treated independently from bending and axial force so that it can be obtained from linear theory; in this situation Eq. (2.20) can be specialized as

$$E\int_{{}^tV} {}_te_{11}\ \delta\,{}_te_{11}\ {}^tdV + \int_{{}^tV} {}^t\sigma\ \delta\,{}_t\eta_{11}\ {}^tdV$$
$$= {}^{t+\Delta t}R - \int_{{}^tV} {}^t\sigma\ \delta\,{}_te_{11}\ {}^tdV \tag{2.21}$$

where

$$E = {}_tC_{1111} \tag{2.22}$$

$${}^t\sigma = {}^t\tau_{11} \tag{2.23}$$

E is the Young's modulus and ${}^t\sigma$ is the axial Cauchy stress.

Eq. (2.21) can be expressed in matrix form. By using interpolation functions for incremental displacements to evaluate the derivatives of displacements, we will obtain the linear and nonlinear strain-displacement transformation matrices. For a beam element it is more effective to first evaluate the finite element matrices in the local coordinate axes x_i, and then transform them to the global coordinate axes X_i prior to the element assemblage process.

2.2.2 Incremental Strain

For our uniaxial strain problem, the u_i in Eq. (2.15) means the displacement increments along the centroid axis of elements and is only the function of x_1-axis. By using interpolation function, u_i can be expressed in the nodal displacement increments.

$$\begin{Bmatrix} {}_tu_1 \\ {}_tu_2 \\ {}_tu_3 \end{Bmatrix} = \left[\begin{array}{cccccc|cccccc} {}_th_1 & 0 & 0 & 0 & 0 & 0 & {}_th_7 & 0 & 0 & 0 & 0 & 0 \\ 0 & {}_th_2 & 0 & 0 & 0 & {}_th_6 & 0 & {}_th_8 & 0 & 0 & 0 & {}_th_{12} \\ 0 & 0 & {}_th_3 & 0 & {}_th_5 & 0 & 0 & 0 & {}_th_9 & 0 & {}_th_{11} & 0 \end{array}\right] \begin{Bmatrix} \Delta^t d_1 \\ \Delta^t d_2 \\ \cdot \\ \cdot \\ \cdot \\ \Delta^t d_{11} \\ \Delta^t d_{12} \end{Bmatrix} \tag{2.24}$$

where

${}_tu_i$ = increment in displacement component of element from t to t+Δt measured in the local axes ${}^{t}x_i$; i = 1,2,3

${}_{t}h_k$ = finite element interpolation function corresponding to $\Delta^{t}d_k$, k = 1 to 12

$\Delta^{t}d_k$ = increment in nodal displacement component of element from t to t+Δt measured in the local axes ${}^{t}x_i$

In Eq. (2.24) we use the Hermitian interpolation functions to describe bending deformations and linear interpolation functions to describe axial and torsional displacements; however, the torsional end displacements do not effect the local element displacements u_1, u_2, u_3 of the centroid axis.

Assuming the cross section of element do not change during deformation for small strain analysis, thus the distances from the centroid axis in the local x_2, x_3-axes directions respectively, say y,z, are constant. The term ${}_{t}u_{1,1}$ in Eq. (2.13) is always small compared to unity, and the square of ${}_{t}u_{1,1}$ is negligible in comparison with ${}_{t}u_{1,1}$. Therefore, from Eq. (2.11) the incremental uniaxial strain along the beam element for small deformation is

$$ {}_{t}\varepsilon_{11} = {}_{t}e_{11} + {}_{t}\eta_{11} \qquad (2.25) $$

where

$$ {}_te_{11} = \underbrace{{}_tu_{1,1}}_{\substack{\text{due to} \\ \text{axial force}}} \underbrace{- y\ {}_tu_{2,11} - z\ {}_tu_{3,11}}_{\text{due to bending}} \tag{2.26} $$

$$ {}_t\eta_{11} = \frac{1}{2}\left({}_tu_{2,1}\right)^2 + \frac{1}{2}\left({}_tu_{3,1}\right)^2 \tag{2.27} $$

$$ {}_tu_{i,jj} = \partial^2 u_i / \partial\, {}^tx^2_j \tag{2.28} $$

2.2.3 Incremental Equilibrium Equation

Eqs. (2.25), (2.26), (2.27) can be expressed in matrix form introduced by Wood and Schrefler [6]

$$ {}_t\varepsilon = {}_te + {}_t\eta \tag{2.29} $$

where

$$ {}_te = {}_tL^T\ {}_t\theta \tag{2.30} $$

$$ {}_t\eta = 1/2\ {}_t\theta^T\ H\ {}_t\theta \tag{2.31} $$

${}_tL^T$ is the row vector defining linear strains ${}_te$ from displacement gradients given by

$$ {}_tL^T = [\ 1 \quad 0 \quad 0 \quad -y \quad -z\ ] \tag{2.32} $$

${}_t\theta$ is the column vector of displacement gradient contributing to the strain ${}_t\varepsilon$ given by

$$ {}_t\theta = [\ {}_tu_{1,1} \quad {}_tu_{2,1} \quad {}_tu_{3,1} \quad {}_tu_{2,11} \ {}_tu_{3,11}\ ]^T \tag{2.33} $$

H is the symmetric matrix containing arrangements of unity and zero given by

$$ H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.34} $$

Employing the finite element discretization of Eq. (2.24) into Eq. (2.33), θ can be expressed in terms of the nodal incremental displacements as

$$ {}_t\theta = {}^t_tB_{NL}\ \Delta^t d \tag{2.35} $$

where

$$ {}^t_tB_{NL} = \left[\begin{array}{cccccc|cccccc} {}_th_{1,1} & 0 & 0 & 0 & 0 & 0 & {}_th_{7,1} & 0 & 0 & 0 & 0 & 0 \\ 0 & {}_th_{2,1} & 0 & 0 & 0 & {}_th_{6,1} & 0 & {}_th_{8,1} & 0 & 0 & 0 & {}_th_{12,1} \\ 0 & 0 & {}_th_{3,1} & 0 & {}_th_{5,1} & 0 & 0 & 0 & {}_th_{9,1} & 0 & {}_th_{11,1} & 0 \\ 0 & {}_th_{2,11} & 0 & 0 & 0 & {}_th_{6,11} & 0 & {}_th_{8,11} & 0 & 0 & 0 & {}_th_{12,11} \\ 0 & 0 & {}_th_{3,11} & 0 & {}_th_{5,11} & 0 & 0 & 0 & {}_th_{9,11} & 0 & {}_th_{11,11} & 0 \end{array} \right] \tag{2.36} $$

in which

$$ {}_t h_{i,j} = \partial\, {}_t h_i \,/\, \partial\, {}^t x_j \tag{2.37} $$

$$ {}_t h_{i,jj} = \partial^2_t h_i \,/\, \partial\, {}^t x_j^2 \tag{2.38} $$

Eq. (2.21) in matrix form is

$$ E \int_{{}^tV} \delta_t e^T \; {}_t e \; {}^t dV + \int_{{}^tV} \delta_t \eta^T \; {}^t\sigma \; {}^t dV $$
$$ = {}^{t+\Delta t}R - \int_{{}^tV} \delta_t e^T \; {}^t\sigma \; {}^t dV \tag{2.39} $$

Taking the variation of Eq. (2.30) and from Eq. (2.35) we have

$$ \delta_t e = {}^t_t B_L \quad \delta\Delta\, {}^t d \qquad \text{where } {}^t_t B_L = {}_t L^T \; {}^t_t B_{NL} \tag{2.40} $$

Taking the variation of Eq. (2.31) and from Eq. (2.35) we have

$$ \delta_t \eta = 1/2 \; (\; \delta_t \theta^T \;\; H \;\; {}_t\theta \;\; + \; {}_t\theta^T \;\; H \;\; \delta_t\theta \;) $$
$$ = {}_t\theta^T \;\; H \;\; \delta_t\theta $$
$$ = {}_t\theta^T \;\; H \;\; {}^t_t B_{NL} \;\; \delta\Delta\, {}^t d \tag{2.41} $$

Substituting Eqs. (2.30),(2.35),(2.40),(2.41) into Eq. (2.39) and eliminating $\delta\Delta\, {}^t d^T$ on both sides of equation, we have

$$\int_{{}^tV} {}^t_tB_L^T \, E \, {}^t_tB_L^T \, \Delta^t d \; {}^t dV + \int_{{}^tV} {}^t_tB_{NL}^T \; {}^t\sigma \; H \; {}^t_tB_{NL} \; \Delta^t d \; {}^t dV$$

$$= {}^{t+\Delta t}r - \int_{{}^tV} {}^t_tB_L^T \; {}^t\sigma \; {}^t dV \tag{2.42}$$

Where ${}^{t+\Delta t}r$ is the vector of externally applied element nodal loads at time $t+\Delta t$ in the local coordinate system.

Let

$${}^t\tau = {}^t\sigma \; H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & {}^t\sigma & 0 & 0 & 0 \\ 0 & 0 & {}^t\sigma & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.43}$$

which is a symmetric Cauchy stress matrix in configuration at time t.

Substituting Eq. (2.43) into Eq. (2.42) we obtain the incremental equilibrium equation

$$({}^t_tk_L + {}^t_tk_{NL})\Delta^t d^k$$

$$= {}^{t+\Delta t}r - {}^{t+\Delta t}_{t+\Delta t}f^{k-1} \tag{2.44}$$

where

$${}^t_tk_L = \int_{{}^tV} {}^t_tB_L^T \; E \; {}^t_tB_L \; {}^t dV \tag{2.45}$$

$${}^t_tk_{NL} = \int_{{}^tV} {}^t_tB_{NL}^T \; {}^t\tau \; {}^t_tB_{NL} \; {}^t dV \tag{2.46}$$

$$\Delta^t d^k = {}^{t+\Delta t}_{t}d^k - {}^{t+\Delta t}_{t}d^{k-1} \tag{2.47}$$

$$ {}^{t+\Delta t}_{t+\Delta t}f^{k-1} = \int_{{}^{t}V} {}^{t}_{t}B_L^T \; {}^{t}\sigma \; {}^{t}dV \tag{2.48} $$

$$ {}^{t}_{t}B_L = {}_{t}L^T \; {}^{t}_{t}B_{NL} $$

$$ = [\; {}_{t}h_{1,1} \quad -y\,{}_{t}h_{2,11} \quad -z\,{}_{t}h_{3,11} \quad 0 \quad \vdots $$

$$ \vdots \; -z\,{}_{t}h_{5,11} \quad -y\,{}_{t}h_{6,11} \quad {}_{t}h_{7,1} \quad -y\,{}_{t}h_{8,11} \; \vdots $$

$$ \vdots \; -z\,{}_{t}h_{9,11} \quad 0 \quad -z\,{}_{t}h_{11,11} \quad -y\,{}_{t}h_{12,11} \;] \tag{2.49} $$

${}^{t}_{t}k_L$ is the linear strain incremental stiffness matrix, ${}^{t}_{t}k_{NL}$ is the nonlinear strain incremental (geometric or initial stress) stiffness matrix, $\Delta^{t}d^{k}$ is the vector of incremental nodal displacements in iteration k, ${}^{t+\Delta t}r$ is the vector of externally applied element nodal loads (given new load level) at time t+Δt, ${}^{t+\Delta t}_{t+\Delta t}f^{k-1}$ is the vector of nodal equivalent element forces corresponding to ${}^{t+\Delta t}_{t+\Delta t}d^{k-1}$, ${}^{t}_{t}B_L$ is the linear strain-displacement transformation matrix, ${}^{t}_{t}B_{NL}$ is the nonlinear strain-displacement transformation matrix defined by Eq. (2.36), ${}^{t}\tau$ is the Cauchy stress matrix in configuration at time t defined by Eq. (2.43).

For simplicity of presentation, Fig. 2.2 shows the local element incremental displacements in U. L. formulation for a

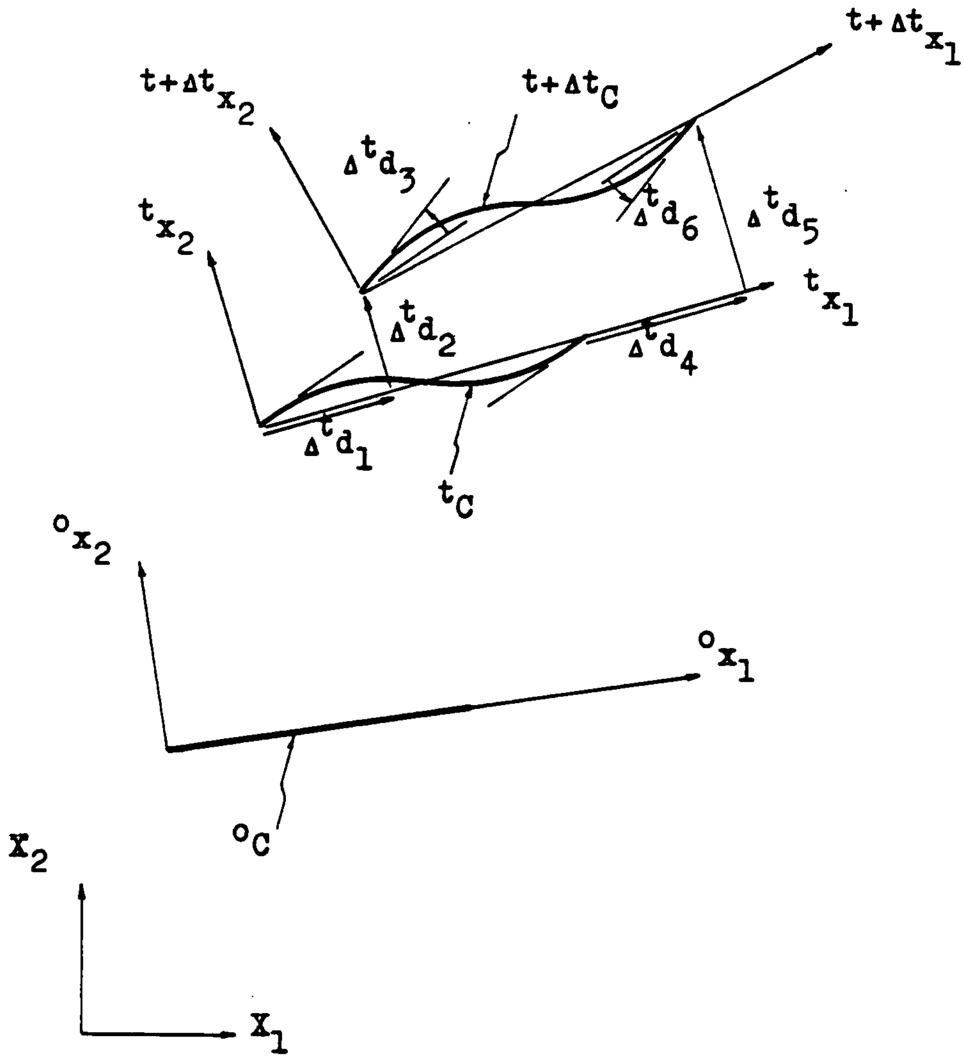


Fig. 2.2 Local Element Displacements in U.L. Formulation for Plane Frame

plane frame, although the theoretical development is carried out for a space frame.

The finite element matrices in Eq. (2.44) are transformed to the global coordinate system for element i

$$ {}^{t}_{t}K^{i} = {}^{t}_{t}\Lambda^{T} \; {}^{t}_{t}k \; {}^{t}_{t}\Lambda \tag{2.50} $$

$$ {}^{t}_{t}F^{i} = {}^{t}_{t}\Lambda^{T} \; {}^{t}_{t}f \tag{2.51} $$

$$ \Delta^{t}D = {}^{t}_{t}\Lambda^{T} \; \Delta^{t}d \qquad \text{(see sec. 2.2.4)} \tag{2.52} $$

in which

$$ {}^{t}_{t}k = {}^{t}_{t}k_{L} + {}^{t}_{t}k_{NL} \tag{2.53} $$

$$ {}^{t}_{t}\Lambda = \left[\begin{array}{cc|cc} {}^{t}_{t}\lambda & 0 & 0 & 0 \\ 0 & {}^{t}_{t}\lambda & 0 & 0 \\ \hline 0 & 0 & {}^{t}_{t}\lambda & 0 \\ 0 & 0 & 0 & {}^{t}_{t}\lambda \end{array} \right] \tag{2.54} $$

and ${}^{t}_{t}\lambda$ is the element orthogonal orientation matrix in configuration ${}^{t}C$, same as in Eq. (3.17).

By employing the member code technique [28] (i.e. direct stiffness procedure), the incremental equilibrium equation in U.L. formulation of the whole structure is

$$ {}_{t}^{t}K \, \Delta q^{k} = {}^{t+\Delta t}Q - {}_{t+\Delta t}^{t+\Delta t}F^{k-1} \qquad (2.55) $$

where

${}_{t}^{t}K$ = structural strain incremental stiffness matrix corresponding to q^{k-1}

Δq^{k} = vector of incremental nodal displacements at k^{th} iteration in configuration at time t

${}^{t+\Delta t}Q$ = vector of given new load level in configuration configuration at time $t+\Delta t$

${}_{t+\Delta t}^{t+\Delta t}F^{k-1}$ = vector of nodal equivalent element forces corresponding to q^{k-1}

q^{k-1} = vector of nodal displacements at k-1 iteration

Here the structural tangent stiffness matrix K, is a function of displacements q, since the problem is nonlinear. In Eq. (2.55) the response of a nonlinear structure may be approximated for incremental nodal displacements by a linear relationship. Because the final configuration is based on equilibrium balance between the nodal element forces and the applied nodal load, the stiffness used to solve Eq. (2.55) need not be exact [41].

2.2.4 Transformation Matrix

The global element displacements are transformed to the local element displacements by the transformation (Fig. 2.2 to Fig. 2.4)

$$ {}_{t}^{t}d = {}_{t}^{t}\Lambda \; {}_{t}^{t}D \tag{2.56} $$

Similarly, the local element forces are transformed to the global element forces by equation

$$ {}_{t}^{t}F^{i} = {}_{t}^{t}\Lambda^{T} \; {}_{t}^{t}f \tag{2.57} $$

where ${}_{t}^{t}\Lambda$ is transformation matrix defined by Eq. (2.54).

Because all variables refer to the current deformed configuration ${}^{t}C$ in the U.L. formulation, the transformation matrix ${}_{t}^{t}\Lambda$ in Eq. (2.54) remains constant from configuration ${}^{t}C$ to ${}^{t+\Delta t}C$. Eqs. (2.56) and (2.57) yield the increments of local element displacements and global element forces

$$ \Delta{}^{t}d = {}_{t}^{t}\Lambda \;\; \Delta{}^{t}D \tag{2.58} $$

$$ \Delta{}^{t}F^{i} = {}_{t}^{t}\Lambda^{T} \; \Delta{}^{t}f \tag{2.59} $$

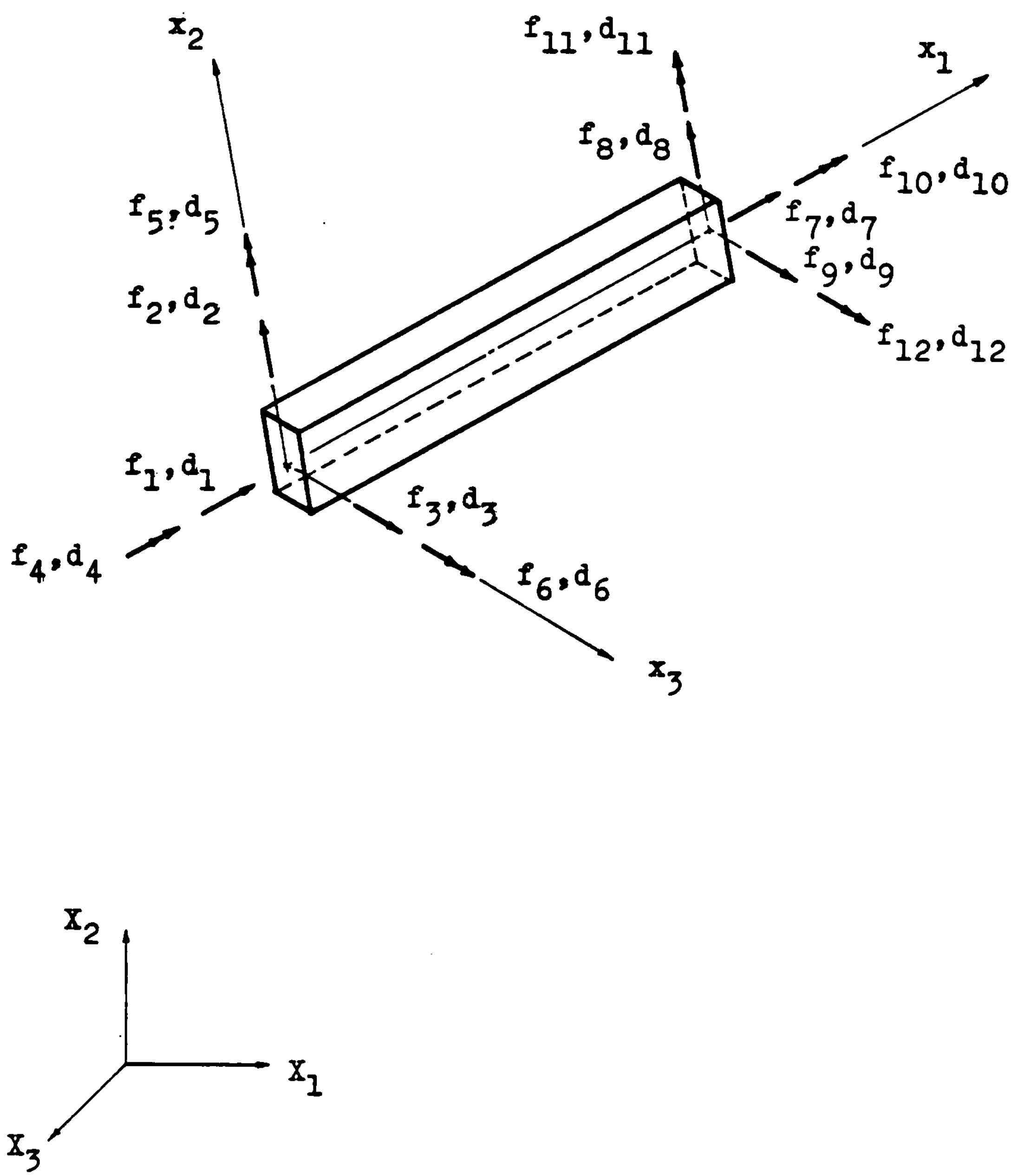


Fig. 2.3 Local Element Forces and Displacements in Space Frame

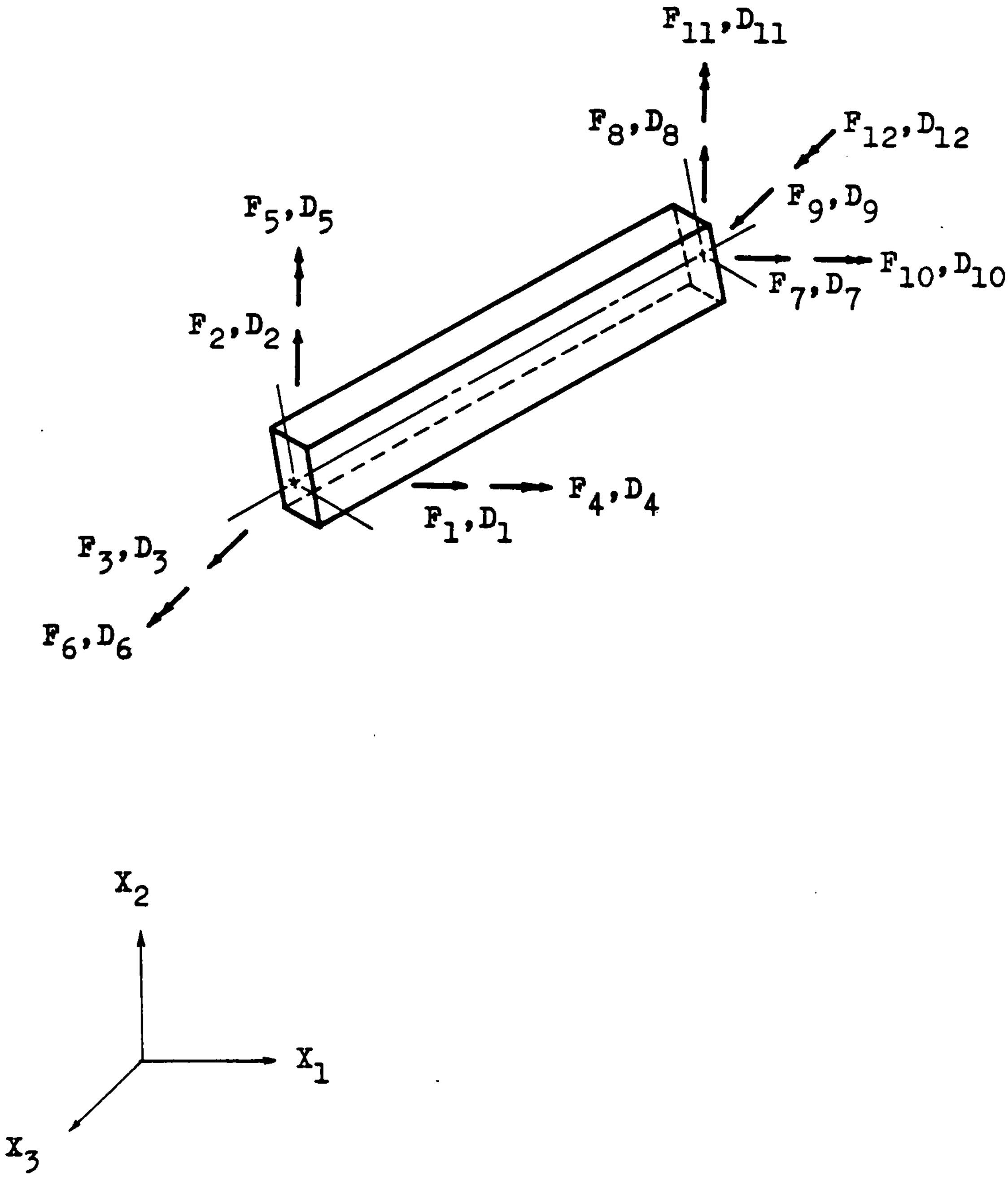


Fig. 2.4 Global Element Forces and Displacements in Space Frame

where $\Delta^{t}f$ and $\Delta^{t}F^{i}$ are the vectors of the local and global incremental element forces from time t to t+Δt respectively, referred to configuration at time t.

2.3 INCREMENTAL EQUILIBRIUM EQUATION IN T.L. FORMULATION

2.3.1 Incremental T.L. Continuum Mechanics Formulation

The T.L. formulation is based on the same procedures that are used in the U.L. formulation, but all variables refer to the initial undeformed configuration ${}^{0}C$ at time 0 (Fig. 2.1).

Taking the variation of Eq. (2.8), we have

$$\delta\,{}^{t+\Delta t}_{\;\;0}\varepsilon_{ij} = \delta\,{}_{0}\varepsilon_{ij} \tag{2.60}$$

Substituting Eqs. (2.7), (2.60) into Eq. (2.2)

$$\int_{{}^{0}V} {}^{t}_{0}S_{ij}\; \delta\,{}_{0}\varepsilon_{ij}\; {}^{0}dV + \int_{{}^{0}V} {}_{0}S_{ij}\; \delta\,{}_{0}\varepsilon_{ij}\; {}^{0}dV = {}^{t+\Delta t}R \tag{2.61}$$

The strain increment components can be separated into linear and nonlinear parts

$${}_{0}\varepsilon_{ij} = {}_{0}e_{ij} + {}_{0}\eta_{ij} \tag{2.62}$$

where

$$\begin{aligned} {}_{0}e_{ij} = 1/2\; (& {}_{0}u_{i,j} + {}_{0}u_{j,i} + {}^{t}_{0}u_{k,i}\;\, {}_{0}u_{k,j} \\ & + {}_{0}u_{k,i}\;\, {}^{t}_{0}u_{k,j}) \end{aligned} \tag{2.63}$$

$$ {}_0\eta_{ij} = 1/2 \; {}_0u_{k,i} \; {}_0u_{k,j} \tag{2.64} $$

$$ {}_0u_{i,j} = \partial u_i \,/\, \partial\, {}^0x_j \tag{2.65} $$

$$ {}_0^tu_{i,j} = \partial\, {}^tu_i \,/\, \partial\, {}^0x_j \tag{2.66} $$

${}_0\eta_{ij}$ is the nonlinear part of strain increment ${}_0\varepsilon_{ij}$, ${}_0u_{i,j}$ is the derivative of displacement increment with respect to local coordinate ${}^0x_j$, ${}_0^tu_{i,j}$ is the derivative of dislacement component in configuration at time t with respect to local coordinate ${}^0x_j$.

The constitutive law is [10]

$$ {}_0S_{ij} = {}_0C_{ijrs} \; {}_0\varepsilon_{rs} \tag{2.67} $$

where ${}_0C_{ijrs}$ is the component of incremental material property tensor at time 0 referred to the configuration at time 0.

Using Eqs. (2.62), (2.67), Eq. (2.61) can be transformed to

$$ \int_{{}^0V} {}_0C_{ijrs} \; {}_0\varepsilon_{rs} \; \delta\, {}_0\varepsilon_{ij} \; {}^0dV + \int_{{}^0V} {}_0^tS_{ij} \; \delta\, {}_0\eta_{ij} \; {}^0dV $$
$$ = {}^{t+\Delta t}R - \int_{{}^0V} {}_0^tS_{ij} \; \delta\, {}_0e_{ij} \; {}^0dV \tag{2.68} $$

Eq. (2.68) is nonlinear in the incremental displacements u_i, and it can be linearized by using the approximations

$$ \delta\, {}_0\varepsilon_{ij} = \delta\, {}_0e_{ij} \tag{2.69} $$

$$ {}_0S_{ij} = {}_0C_{ijrs}\ {}_0e_{rs} \tag{2.70}$$

Then Eq. (2.68) becomes

$$\int_{{}^0V} {}_0C_{ijrs}\ {}_0e_{rs}\ \delta_0e_{ij}\ {}^0dV + \int_{{}^0V} {}_0^tS_{ij}\ \delta_0\eta_{ij}\ {}^0dV = {}^{t+\Delta t}R - \int_{{}^0V} {}_0^tS_{ij}\ \delta_0e_{ij}\ {}^0dV \tag{2.71}$$

which is linear equation in the incremental displacements u_i, corresponding to the local coordinate system.

For three dimensional beam element with small deformation and uniaxial state of strain (i.e. ${}_0\varepsilon_{11}$ only), in which torsion is treated independently, Eq. (2.71) can be specialized as

$$E\int_{{}^0V} {}_0e_{11}\ \delta_0e_{11}\ {}^0dV + \int_{{}^0V} {}_0^t\sigma\ \delta_0\eta_{11}\ {}^0dV = {}^{t+\Delta t}R - \int_{{}^0V} {}_0^t\sigma\ \delta_0e_{11}\ {}^0dV \tag{2.72}$$

where

$$ {}_0^t\sigma = {}_0^tS_{11} \tag{2.73}$$

is the 2nd Piola-Kirchhoff stress.

2.3.2 Incremental Strain

The incremental displacement components along the centroid axis of elements are interpolated as

$$\begin{Bmatrix} {}_0u_1 \\ {}_0u_2 \\ {}_0u_3 \end{Bmatrix} = \left[\begin{array}{cccccc|cccccc} {}_0h_1 & 0 & 0 & 0 & 0 & 0 & {}_0h_7 & 0 & 0 & 0 & 0 & 0 \\ 0 & {}_0h_2 & 0 & 0 & 0 & {}_0h_6 & 0 & {}_0h_8 & 0 & 0 & 0 & {}_0h_{12} \\ 0 & 0 & {}_0h_3 & 0 & {}_0h_5 & 0 & 0 & 0 & {}_0h_9 & 0 & {}_0h_{11} & 0 \end{array}\right] \begin{Bmatrix} \Delta^0 d_1 \\ \Delta^0 d_2 \\ \cdot \\ \cdot \\ \cdot \\ \Delta^0 d_{11} \\ \Delta^0 d_{12} \end{Bmatrix} \quad (2.74)$$

where

${}_0u_i$ = increment in displacement component of element from t to t+Δt measured in the local axes ${}^0x_i$; i=1,2,3

${}_0h_k$ = finite element interpolation function corresponding to Δ^0d_k, k = 1 to 12

Δ^0d_k = increment in nodal displacement component of element from t to t+Δt measures in the local axes ${}^0x_i$

Neglecting the 2nd order terms of ${}^t_0u_{1,1}$, ${}_0u_{1,1}$, from Eq. (2.62) the incremental uniaxial strain along the beam element for small deformation is

$${}_0\varepsilon_{11} = {}_0e_{11} + {}_0\eta_{11} \quad (2.75)$$

where

$$ {}_0e_{11} = \underbrace{{}_0u_{1,1}}_{\substack{\text{due to}\\ \text{axial force}}} + \underbrace{{}^t_0u_{2,1}\ {}_0u_{2,1} + {}^t_0u_{3,1}\ {}_0u_{3,1}}_{\substack{\text{due to initial}\\ \text{displacement}}} $$

$$ \underbrace{- y\ {}_0u_{2,11} - z\ {}_0u_{3,11}}_{\text{due to bending}} \qquad (2.76) $$

$$ {}_0\eta_{11} = 1/2\ ({}_0u_{2,1})^2 + 1/2\ ({}_0u_{3,1})^2 \qquad (2.77) $$

and

$$ {}^t_0u_{i,j} = \partial\, {}^tu_i \ /\ \partial\, {}^0x_j \qquad (2.78) $$

$$ {}_0u_{i,jj} = \partial^2 u_i \ /\ \partial\, {}^0x_j^2 \qquad (2.79) $$

2.3.3 Incremental Equilibrium Equation

Expressing Eqs. (2.75), (2.76), (2.77) in matrix form introduced by Wood and Schrefler [6]

$$ {}_0\varepsilon = {}_0e + {}_0\eta \qquad (2.80) $$

where

$$ {}_0e = {}_0L^T\ {}_0\theta \qquad (2.81) $$

$$ {}_0\eta = 1/2\ {}_0\theta^T\ H\ {}_0\theta \qquad (2.82) $$

$${}_0L^T = [\ 1 \quad {}^t_0u_{2,1} \quad {}^t_0u_{3,1} \quad -y \quad -z\ ] \tag{2.83}$$

$${}_0\theta = [\ {}_0u_{1,1} \quad {}_0u_{2,1} \quad {}_0u_{3,1} \quad {}_0u_{2,11} \quad {}_0u_{3,11}\ ]^T \tag{2.84}$$

${}_0L^T$ is the row vector defining linear strains ${}_0e$ from displacement gradients, ${}_0\theta$ is the column vector of displacement gradient contributing to the strain ${}_0\varepsilon$, H is defined as in Eq. (2.34).

Employing Eq. (2.74) into Eq. (2.84), we express θ in terms of the nodal incremental displacements as

$${}_0\theta = {}^t_0B_{NL}\ \Delta^0d \tag{2.85}$$

where

$${}^t_0B_{NL} = \left[\begin{array}{cccccc|cccccc}
{}_0h_{1,1} & 0 & 0 & 0 & 0 & 0 & {}_0h_{7,1} & 0 & 0 & 0 & 0 & 0 \\
0 & {}_0h_{2,1} & 0 & 0 & 0 & {}_0h_{6,1} & 0 & {}_0h_{8,1} & 0 & 0 & 0 & {}_0h_{12,1} \\
0 & 0 & {}_0h_{3,1} & 0 & {}_0h_{5,1} & 0 & 0 & 0 & {}_0h_{9,1} & 0 & {}_0h_{11,1} & 0 \\
0 & {}_0h_{2,11} & 0 & 0 & 0 & {}_0h_{6,11} & 0 & {}_0h_{8,11} & 0 & 0 & 0 & {}_0h_{12,11} \\
0 & 0 & {}_0h_{3,11} & 0 & {}_0h_{5,11} & 0 & 0 & 0 & {}_0h_{9,11} & 0 & {}_0h_{11,11} & 0
\end{array}\right] \tag{2.86}$$

in which

$${}_0h_{i,j} = \partial\,{}_0h_i \,/\, \partial\,{}^0x_j \tag{2.87}$$

$$ {}_0h_{i,jj} = \partial^2_0 h_i \,/\, \partial\,{}^0x_j^2 \tag{2.88}$$

Eq. (2.72) in matrix form is

$$E \int_{{}^0V} \delta_0 e^T \, {}_0e \, {}^0dV + \int_{{}^0V} \delta_0 \eta^T \, {}^t_0\sigma \, {}^0dV = {}^{t+\Delta t}R - \int_{{}^0V} \delta_0 e^T \, {}^t_0\sigma \, {}^0dV \tag{2.89}$$

Taking the variation of Eq. (2.81) and using Eq. (2.85) we obtain

$$\delta_0 e = {}^t_0B_L \, \delta\Delta\,{}^0d \tag{2.90}$$

where

$${}^t_0B_L = {}_0L^T \, {}^t_0B_{NL} = {}^t_0B_{L0} + {}^t_0B_{L1} \tag{2.91a}$$

and

$$\begin{aligned} {}^t_0B_{L0} = \Big[\, & {}_0h_{1,1} \quad -y\,{}_0h_{2,11} \quad -z\,{}_0h_{3,11} \quad 0 \quad -z\,{}_0h_{5,11} \quad -y\,{}_0h_{6,11} \;\vdots \\ \vdots\; & {}_0h_{7,1} \quad -y\,{}_0h_{8,11} \quad -z\,{}_0h_{9,11} \quad 0 \quad -z\,{}_0h_{11,11} \quad -y\,{}_0h_{12,11} \Big] \end{aligned} \tag{2.91b}$$

$$\begin{aligned} {}^t_0B_{L1} = \Big[\, & 0 \quad {}^t_0u_{2,1}\,{}_0h_{2,1} \quad {}^t_0u_{3,1}\,{}_0h_{3,1} \quad 0 \quad {}^t_0u_{3,1}\,{}_0h_{5,1} \quad {}^t_0u_{2,1}\,{}_0h_{6,1} \;\vdots \\ \vdots\; & 0 \quad {}^t_0u_{2,1}\,{}_0h_{8,1} \quad {}^t_0u_{3,1}\,{}_0h_{9,1} \quad 0 \quad {}^t_0u_{3,1}\,{}_0h_{11,1} \quad {}^t_0u_{2,1}\,{}_0h_{12,1} \Big] \end{aligned} \tag{2.91c}$$

Taking the variation of Eq. (2.82) and from Eq. (2.85) we have

$$\begin{aligned}\delta_0\eta &= \frac{1}{2}\left(\delta_0\theta^T H\, {}_0\theta + {}_0\theta^T H\, \delta_0\theta\right)\\ &= {}_0\theta^T H\, \delta_0\theta\\ &= {}_0\theta^T H\, {}^t_0B_{NL}\, \delta\Delta{}^0d\end{aligned} \tag{2.92}$$

Substituting Eqs. (2.81),(2.85),(2.90),(2.92) into Eq. (2.89), we have

$$\int_{{}^0V} {}^t_0B_L^T\, E\, {}^t_0B_L\, \Delta{}^0d\, {}^0dV + \int_{{}^0V} {}^t_0B_{NL}^T\, {}^t_0\sigma\, H\, {}^t_0B_{NL}\, \Delta{}^0d\, {}^0dV = {}^{t+\Delta t}r - \int_{{}^0V} {}^t_0B_L^T\, {}^t_0\sigma\, {}^0dV \tag{2.93}$$

Let

$${}^t_0S = {}^t_0\sigma\, H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0\\ 0 & {}^t_0\sigma & 0 & 0 & 0\\ 0 & 0 & {}^t_0\sigma & 0 & 0\\ 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\end{bmatrix} \tag{2.94}$$

which is the 2nd Piola-Kirchhoff stress matrix.

Substituting Eq. (2.94) into Eq. (2.93), we obtain the incremental equilibrium equation

$$({}^t_0k_L + {}^t_0k_{NL})\, \Delta{}^0d^k = {}^{t+\Delta t}r - {}^{t+\Delta t}_0f^{k-1} \tag{2.95}$$

where

$${}^{t}_{0}k_{L} = \int_{{}^{0}V} {}^{t}_{0}B_{L}^{T}\ E\ {}^{t}_{0}B_{L}\ {}^{0}dV \tag{2.96}$$

$${}^{t}_{0}k_{NL} = \int_{{}^{0}V} {}^{t}_{0}B_{NL}^{\ T}\ {}^{t}_{0}S\ {}^{t}_{0}B_{NL}\ {}^{0}dV \tag{2.97}$$

$\Delta^{0}d^{k}$ = vector of incremental nodal displacement at time t in iteraion k; i.e.

$${}^{t+\Delta t}_{0}d^{k} = {}^{t+\Delta t}_{0}d^{k-1} + \Delta^{0}d^{k} \tag{2.98}$$

${}^{t+\Delta t}r$ = vector of externally applied element nodal loads at time t+Δt

$${}^{t+\Delta t}_{0}f^{k-1} = \int_{{}^{0}V} {}^{t}_{0}B_{L}^{T}\ {}^{t}_{0}\sigma\ {}^{0}dV \tag{2.99}$$

For the simplicity of presentation, Fig. 2.5 shows the local element displacements in T.L. formulation for a plane frame.

The finite element matrices in Eq. (2.95) are transformed to the global coordinate system for element i

$${}^{t}_{0}K^{i} = {}^{t}_{0}\Lambda^{T}\ {}^{t}_{0}k\ {}^{t}_{0}\Lambda \tag{2.100}$$

$${}^{t}_{0}F^{i} = {}^{t}_{0}\Lambda^{T}\ {}^{t}_{0}f \tag{2.101}$$

$$\Delta^{0}D = {}^{t}_{0}\Lambda^{T}\ \Delta^{0}d \qquad \text{(see sec. 2.3.4)} \tag{2.102}$$

in which

$${}^{t}_{0}k = {}^{t}_{0}k_{L} + {}^{t}_{0}k_{NL} \tag{2.103}$$

${}^{t}_{0}\Lambda$ = transformation matrix (see sec. 2.3.4)

Employing the member code technique [28], the incremental equilibrium equation in T.L. formulation of the whole structure is

$$ {}^{t}_{0}K \, \Delta q^{k} = {}^{t+\Delta t}Q - {}^{t+\Delta t}_{0}F^{k-1} \tag{2.104} $$

2.3.4 Transformation Matrix

Similar to Eqs. (2.56) to (2.59), we have (Fig. 2.3 to Fig. 2.5)

$$ {}^{t}_{0}d = {}^{t}_{0}\Lambda \, {}^{t}_{0}D \tag{2.105} $$

$$ {}^{t}_{0}F^{i} = {}^{t}_{0}\Lambda^{T} \, {}^{t}_{0}f \tag{2.106} $$

where

$$ {}^{t}_{0}\Lambda = \begin{bmatrix} {}^{t}_{0}\lambda & 0 & 0 & 0 \\ 0 & {}^{t}_{0}\lambda & 0 & 0 \\ 0 & 0 & {}^{t}_{0}\lambda & 0 \\ 0 & 0 & 0 & {}^{t}_{0}\lambda \end{bmatrix} \tag{2.107} $$

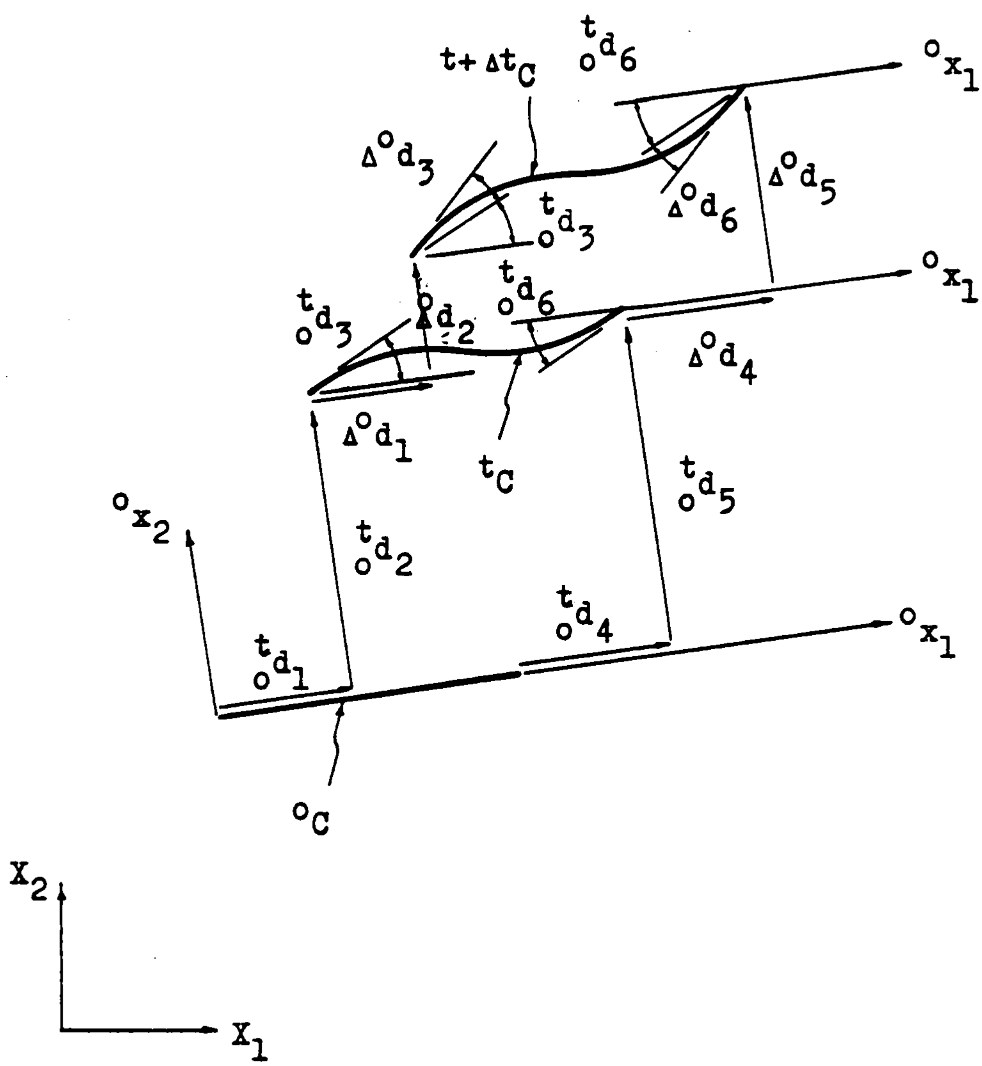


Fig. 2.5 Local Element Displacements in T.L. Formulation for Plane Frame

and ${}^t_0\lambda$ is the orthogonal orientation matrix of element in configuration at time t referred to configuration at time 0. Because all variables refer to the initial undeformed configuration ${}^0C$ in the T.L. formulation, the transformation matrix ${}^t_0\Lambda$ in Eq. (2.107) always not change from time 0 to t and t to t+Δt, i.e.

$$
{}^t_0\lambda = {}^0_0\lambda \tag{2.108}
$$

The initial orientation matrix of space element in initial configuration ${}^0C$ is [28]

$$
{}^0_0\lambda = \begin{bmatrix} {}^0c_1 & {}^0c_2 & {}^0c_3 \\ -\dfrac{{}^0c_1{}^0c_2}{{}^0\ell}\cos\phi - \dfrac{{}^0c_3}{{}^0\ell}\sin\phi & {}^0\ell\cos\phi & -\dfrac{{}^0c_2{}^0c_3}{{}^0\ell}\cos\phi + \dfrac{{}^0c_1}{{}^0\ell}\sin\phi \\ \dfrac{{}^0c_1{}^0c_2}{{}^0\ell}\sin\phi - \dfrac{{}^0c_3}{{}^0\ell}\cos\phi & -{}^0\ell\sin\phi & \dfrac{{}^0c_2{}^0c_3}{{}^0\ell}\sin\phi + \dfrac{{}^0c_1}{{}^0\ell}\cos\phi \end{bmatrix} \tag{2.109}
$$

where

${}^0c_i$ = direction cosines of the local x_1-axis at time 0 with respect to the global coordinate system X_i -axis; i = 1,2,3

ϕ = roll angle between the local x_2-axis and global X_2-axis

$$ {}^{0}\ell = ({}^{0}c_1^2 + {}^{0}c_3^2)^{1/2} \tag{2.110} $$

Eqs. (2.105), (2.106) yield the increments of the local element displacements and global element forces in T.L. formulation

$$ \Delta {}^{0}d = {}^{0}_{0}\Lambda \; \Delta {}^{0}D \tag{2.111} $$

$$ \Delta {}^{0}F^{i} = {}^{0}_{0}\Lambda^{T} \; \Delta {}^{0}f \tag{2.112} $$

where $\Delta {}^{0}f$ and $\Delta {}^{0}F^{i}$ are the vectors of the lcoal and global incremental element forces from time t to t+Δt respectively, referred to configuration at time 0.

2.4 CONVECTED COORDINATE FORMULATION

In the convected coordiante formulation developed by Belytschko [5], [17], all variables refer to the new incremented configuration at time t+Δt. The convected coordinate system means that each element is associated with a rigid cartesian coordinate system that rotates and translates with the element but does not deform with the element. Because the coordinate systems corresponding to the configurations at time t and t+Δt are independent of each other, an incremented concept in this formulation cannot be directly applied [18]. Hence, in the convected coordinate formulation the displacements of each element at time $t+\Delta t$ are decomposed into rigid body displacements and deformation displacements as shown in Fig. 2.6.

$$ {}^{t+\Delta t}_{t+\Delta t}d = {}^{t+\Delta t}_{t+\Delta t}d^{rig} + {}^{t+\Delta t}_{t+\Delta t}d^{def} \tag{2.113} $$

where

$$ {}^{t+\Delta t}_{t+\Delta t}d = [{}^{t+\Delta t}_{t+\Delta t}d_1 \quad {}^{t+\Delta t}_{t+\Delta t}d_2 \quad {}^{t+\Delta t}_{t+\Delta t}d_3 \ldots\ldots\ldots\ldots\ldots {}^{t+\Delta t}_{t+\Delta t}d_{10} \quad {}^{t+\Delta t}_{t+\Delta t}d_{11} \quad {}^{t+\Delta t}_{t+\Delta t}d_{12}]^T \tag{2.114} $$

$$ {}^{t+\Delta t}_{t+\Delta t}d^{rig} = [{}^{t+\Delta t}_{t+\Delta t}d_1^{rig} \quad {}^{t+\Delta t}_{t+\Delta t}d_2^{rig} \quad {}^{t+\Delta t}_{t+\Delta t}d_3^{rig} \ldots\ldots {}^{t+\Delta t}_{t+\Delta t}d_{10}^{rig} \quad {}^{t+\Delta t}_{t+\Delta t}d_{11}^{rig} \quad {}^{t+\Delta t}_{t+\Delta t}d_{12}^{rig}]^T \tag{2.115} $$

$$ {}^{t+\Delta t}_{t+\Delta t}d^{def} = [{}^{t+\Delta t}_{t+\Delta t}d_1^{def} \quad {}^{t+\Delta t}_{t+\Delta t}d_2^{def} \quad {}^{t+\Delta t}_{t+\Delta t}d_3^{def} \ldots\ldots {}^{t+\Delta t}_{t+\Delta t}d_{10}^{def} \quad {}^{t+\Delta t}_{t+\Delta t}d_{11}^{def} \quad {}^{t+\Delta t}_{t+\Delta t}d_{12}^{def}]^T \tag{2.116} $$

The displacements in each element u can be similarly decomposed into rigid body displacements u^{rig} and deformation displacements u^{def}, i.e.

$$ {}^{t+\Delta t}u = {}^{t+\Delta t}u^{rig} + {}^{t+\Delta t}u^{def} \tag{2.117} $$

where u is measured from the initial position at time 0. For our problem small deformation and uniaxial state of strain are assumed; thus for space frame element u is 3x1 column matrix and can be represented by interpolation functions h so that

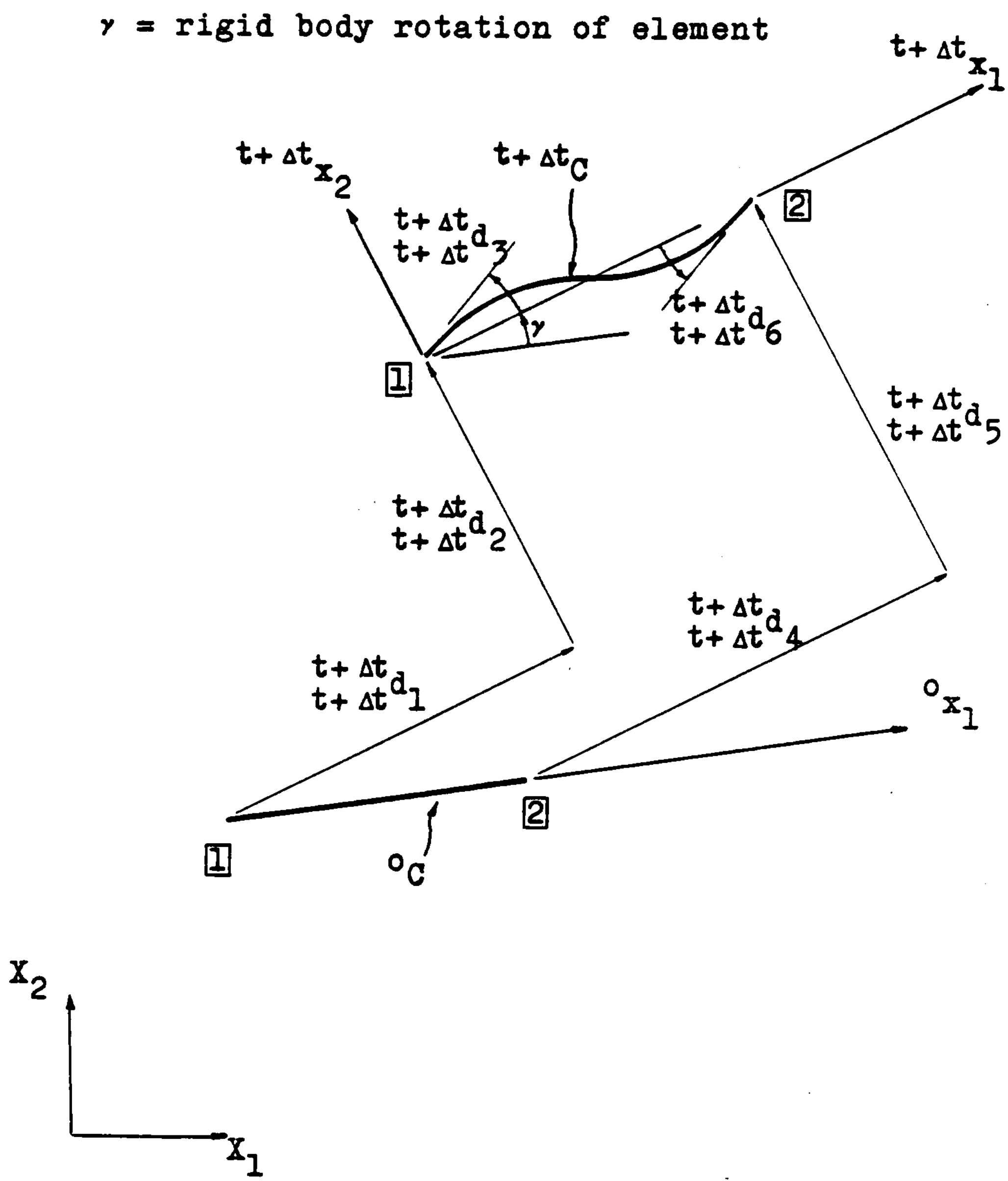


Fig. 2.6 Local Element Displacements in Convected Coordinate Formulation for Plane Frame

$$\begin{Bmatrix} {}^{t+\Delta t}u_1 \\ {}^{t+\Delta t}u_2 \\ {}^{t+\Delta t}u_3 \end{Bmatrix} = \left[\begin{array}{cccccc|cccccc} h_1 & 0 & 0 & 0 & 0 & 0 & h_7 & 0 & 0 & 0 & 0 & 0 \\ 0 & h_2 & 0 & 0 & 0 & h_6 & 0 & h_8 & 0 & 0 & 0 & h_{12} \\ 0 & 0 & h_3 & 0 & h_5 & 0 & 0 & 0 & h_9 & 0 & h_{11} & 0 \end{array}\right] \begin{Bmatrix} {}^{t+\Delta t}_{t+\Delta t}d_1 \\ {}^{t+\Delta t}_{t+\Delta t}d_2 \\ \vdots \\ {}^{t+\Delta t}_{t+\Delta t}d_{11} \\ {}^{t+\Delta t}_{t+\Delta t}d_{12} \end{Bmatrix}$$

or

$${}^{t+\Delta t}u = h \; {}^{t+\Delta t}_{t+\Delta t}d \tag{2.118}$$

Substituting Eq. (2.113) into Eq. (2.118) and from Eq. (2.117) we obtain

$${}^{t+\Delta t}u^{rig} = h \; {}^{t+\Delta t}_{t+\Delta t}d^{rig} \tag{2.119}$$

and

$${}^{t+\Delta t}u^{def} = h \; {}^{t+\Delta t}_{t+\Delta t}d^{def} \tag{2.120}$$

In this formulation the global element displacements at time $t+\Delta t$ are transformed to the local element displacements by the transformation (Figs. 2.3 and 2.4)

$${}^{t+\Delta t}_{t+\Delta t}d = {}^{t+\Delta t}_{t+\Delta t}\Lambda \; {}^{t+\Delta t}_{t+\Delta t}D \tag{2.121}$$

From the Appendix of reference [5], the relationships between the strains and deformation displacements in the convected coordinate system of each element are linear so that

$$ {}^{t+\Delta t}_{t+\Delta t}\varepsilon = B_L \; {}^{t+\Delta t}_{t+\Delta t}d^{def} \tag{2.122}$$

where B_L is the linear strain displacement transformation matrix and can be obtained from the Eq. (2.120) and appropriate linear strain displacement equations.

As shown in the appendix of reference [5], consider two successive configurations at time t and t+Δt, the global internal nodal forces of each element can be derived from the principle of virtual work and it yields

$$ {}^{t+\Delta t}_{t+\Delta t}F = {}^{t+\Delta t}_{t+\Delta t}\Lambda^T \int_{{}^{t+\Delta t}V} B_L^T \; {}^{t+\Delta t}\sigma \; {}^{t+\Delta t}dV \tag{2.123}$$

$$ {}^{t}_{t}F = {}^{t}_{t}\Lambda^T \int_{{}^{t}V} B_L^T \; {}^{t}\sigma \; {}^{t}dV \tag{2.124}$$

in which ${}^{t}\sigma$ and ${}^{t+\Delta t}\sigma$ are the axial Cauchy stresses at time t and t+Δt respectively, V is the element volume which the change of volume can be neglected in the volume integration for small deformations, Λ is the transformation matrix from the global coordinate system to the convected coordinate system, defining

$$ \Delta^{t}\Lambda = {}^{t+\Delta t}_{t+\Delta t}\Lambda - {}^{t}_{t}\Lambda \tag{2.125}$$

$$ \Delta^{t}\sigma = {}^{t+\Delta t}\sigma - {}^{t}\sigma \tag{2.126}$$

and

$$\Delta^{t}F = {}^{t+\Delta t}_{t+\Delta t}F - {}^{t}_{t}F \tag{2.127}$$

From Eqs. (2.123) to (2.127), we obtain

$$\Delta^{t}F = {}^{t}_{t}\Lambda^{T}\int_{V} B_{L}^{T}\,\Delta^{t}\sigma\,dV + \Delta^{t}\Lambda^{T}\int_{V} B_{L}^{T}\,{}^{t+\Delta t}\sigma\,dV \tag{2.128}$$

To establish the global tangent stiffness matrix for an element, we must express $\Delta^{t}\sigma$ and $\Delta^{t}\Lambda$ in Eq. (2.128) in terms of the nodal incremental displacements $\Delta^{t}D$ defined as

$$\Delta^{t}D = {}^{t+\Delta t}_{t+\Delta t}D - {}^{t}_{t}D \tag{2.129}$$

Because in this convected coordinate system the nodal element dsiplacements between the configurations at time t and t+Δt are independent of each other, the nodal incremental displacements in Eq. (2.129) are not vector quantity. Hence, it is inappropriate to express $\Delta^{t}\sigma$ and $\Delta^{t}\Lambda$ in Eq. (2.128) in terms of $\Delta^{t}D$. Therefore, using the finite element model in the convected coordinate formulation to establish the local element tangent stiffness matrix is inconvenient.

2.5 COMPARISON OF U.L. AND T.L. FORMULATIONS

The main difference between the U.L. and T.L. formulations lies in the coordinate system referred to formulate the incremental equilibrium equations; the former refers to the current deformed configuration ${}^{t}C$; the latter refers to the initial undeformed configuration ${}^{0}C$.

Regardless of which formulation used, we will show that the incremental equilibrium Eqs. (2.55) and (2.104) are identical in both formulations [9]. From Eqs. (2.50) and (2.100) the global element tangent stiffness matrices are

$$ {}^{t}_{t}K^{i} = {}^{t}_{t}\Lambda^{T} \, {}^{t}_{t}k \, {}^{t}_{t}\Lambda \qquad \text{for U.L. formulation} \tag{2.130} $$

$$ {}^{t}_{0}K^{i} = {}^{t}_{0}\Lambda^{T} \, {}^{t}_{0}k \, {}^{t}_{0}\Lambda \qquad \text{for T.L. formulation} \tag{2.131} $$

in which

${}^{t}_{t}\Lambda$, ${}^{t}_{0}\Lambda$ = transformation matrix between the local coordinate axes in configuration at time t and the global coordinate axes

Let

$$ {}^{t}_{t}\Lambda = R \, {}^{t}_{0}\Lambda \tag{2.132} $$

where R is the transformation matrix from the local coordinates ${}^0x_i$ to ${}^tx_i$ in space (Fig. 2.1); it has been derived by Bathe [9] and will not be described here.

Substituting Eq. (2.132) into (2.130)

$$ {}^t_tK^i = {}^t_0\Lambda^T \; {}^t_0k \; {}^t_0\Lambda \tag{2.133} $$

where

$$ {}^t_0k = R^T \; {}^t_tk \; R \tag{2.134} $$

From Eqs. (2.131) and (2.133), we have

$$ {}^t_tK^i = {}^t_0K^i \tag{2.135} $$

Therefore, the global element tangent stiffness matrices K^i are identical in both formulations.

As a result of Eq. (2.135), the assembled system tangent stiffness matrices K in Eqs. (2.55) and (2.104) are identical in both formulations.

$$ {}^t_tK = {}^t_0K \tag{2.136} $$

Similarly, from Eqs. (2.51),(2.101) the global element forces are

$$ {}^{t+\Delta t}_{t+\Delta t}F^{i\,k-1} = {}^{t+\Delta t}_{t+\Delta t}\Lambda^{k-1\,T} \; {}^{t+\Delta t}_{t+\Delta t}f^{k-1} $$

for U.L. formulation (2.137)

$$ {}^{t+\Delta t}_{0}F^{i\,k-1} = {}^{t+\Delta t}_{0}\Lambda^{k-1\,T}\ {}^{t+\Delta t}_{0}f^{k-1} $$

for T.L. formulation (2.138)

Substituting Eq. (2.132) into Eq. (2.137)

$$ {}^{t+\Delta t}_{t+\Delta t}F^{i\,k-1} = {}^{t+\Delta t}_{0}\Lambda^{k-1\,T}\ {}^{t+\Delta t}_{0}f^{k-1} \tag{2.139} $$

where

$$ {}^{t+\Delta t}_{0}f^{k-1} = R^{T}\ {}^{t+\Delta t}_{t+\Delta t}f^{k-1} \tag{2.140} $$

From Eqs. (2.138),(2.139), we have

$$ {}^{t+\Delta t}_{t+\Delta t}F^{i\,k-1} = {}^{t+\Delta t}_{0}F^{i\,k-1} \tag{2.141} $$

Therefore, the global element force vectors are identical in both formulations.

As a result of Eq. (2.141), the vectors of equivalent nodal element forces F in Eqs. (2.55),(2.104) are identical in both formulations.

$$ {}^{t+\Delta t}_{t+\Delta t}F^{k-1} = {}^{t+\Delta t}_{0}F^{k-1} \tag{2.142} $$

From Eqs. (2.136),(2.142) we may conclude that the incremental equilibrium Eqs. (2.55) and (2.104) are identical in both formulations.

As a result of Eqs. (2.55) and (2.104), we are led to the conclusion that

The main advantages of the U.L. formulation are:

1. Since the rotation referred to the current configuration ${}^{t}C$ is infinitesimal, it can be treated as vector in space frame [2], [18].
2. The linear incremental strains in Eq. (2.12), referred to the current configuration ${}^{t}C$, not include the initial displacement effect; it results that the linear strain incremental stiffness matrix ${}_{t}^{t}K_{L}$ in Eq. (2.55) is not including the initial displacement effect.
3. Because the strains, referred to the current configuration ${}^{t}C$, are so infinitesimal that sometimes we can neglect the nonlinear part of incremental strains in Eq. (2.11); it results that even the nonlinear strain incremental stiffness matrix ${}_{t}^{t}K_{NL}$ in Eq. (2.55) can be omited. In fact, as we pointed out earlier, the stiffness used to solve Eq. (2.55) need not be exact.

4. Items 2 and 3 result that simple and approximate formulations of the U.L. in Eq. (2.55) may be developed, which make the U.L. formulation very efficient and clearly superior to the T.L. formulation.
5. Especially more suitable for the large displacement but small strain problems which are very common in practice for many types of problems [36].
6. If the relative member deformations are small enough that we can directly employ the beam-column model which uses convected coordinate system to formulate the tangent stiffness matrix and nodal equivalent force in Eq. (2.55), in addition to finite element model.

The disadvantages of the U.L. formulation are:

1. The transformation matrix ${}^{t}_{t}\Lambda$ in Eq. (2.54), referred to the current configuration ${}^{t}C$, must be updated at each time step (iteration).
2. The stresses at time t+Δt as in Eq. (2.5)

$$ {}^{t+\Delta t}_{t}S_{ij} = {}^{t}\tau_{ij} + {}_{t}S_{ij} $$

where the stress increments ${}_{t}S_{ij}$ from time t to t+Δt are the 2nd Piola-Kirchhoff stresses referred to time t, which must be transformed into Cauchy stresses.

Therefore, it is slightly more complicated than in T.L. formulation to compute the stresses [36].

3. Mass matrix would be updated at each iteration which leads to complexity in the U.L. formulation to dynamic problems.
4. As to economic consideration, items 1 to 3 result in more computational effort than T.L. formulation.

On the other hand, the advantages of the T.L. formulation are:

1. The transformation matrix ${}_0^t\Lambda$ in Eq. (2.107) remains unchanged throughout each iteration as in Eq. (2.108).
2. The use of a unique type of the 2nd Piola-Kirchhoff stresses in Eq. (2.7).
3. Because the mass matrix would then be constant in the extension to dynamic problems throughout each iteration, it leads to simplification in the T.L. formulation [12], [3].
4. Items 1 to 3 result in less computional effort than U.L. formulation which saves computer time and money.

The disadvantages of the T.L. formulation are:

1. Because the rotation referred to the undeformed configuration ${}^{0}C$ is finite, it cannot be treated as vector in space problems; therefore, the merit of matrix operation will be lost [3], [18].
2. The linear incremental strains in Eq. (2.63), referred to the initial configuration ${}^{0}C$, include the initial displacement effect; it results that the linear strain incremental stiffness matrix ${}^{t}_{0}K_{L}$ in Eq. (2.104) include the initial displacement effect.
3. Because the strains referred to the initial configuration ${}^{0}C$ is finite, as displacements become larger and larger, nonlinear term of incremental strains in Eq. (2.62) are significant; in other words, the nonlinear strain incremental stiffness matrix ${}^{t}_{0}K_{NL}$ in Eq. (2.104) must be taken into account to obtain the exact stiffness for large displacements.
4. Employing the beam-column model, which uses convected coordinate system, in T.L. formulation, the local element stiffness matrix and force vector must be transformed to the initial coordinate system.

In conclusion, no matter which formulation we use, the same structural stiffness matrices and nodal equivalent element forces should be obtained. Therefore, the solutions using different formulations must be the same, if the same number of elements are employed [9]. Whether to use the U.L. or the T.L. formulations depends largely on the program design and the practical problems. In beam analysis the U.L. formulation is more effective than the T.L. formulation in which the additional ${}_0^tB_{L1}$ matrix must be evaluated as in Eq. (2.91c). Based on this concept, in next chapters we will only employ the U.L. formualtion to solve the incremental equilibrium equation as shown in Eq. (2.55).

Chapter III

DEFORMATION DISPLACEMENTS OF SPACE FRAME ELEMENT

3.1 INTRODUCTION

The element displacements are decomposed into rigid body displacements and deformation dispacements. In this chapter the deformation displacements are derived for a space frame element with large displacements and rotations of nodes and members. However, the strain in each element is assumed to be small; that is, the element deformations are restricted to be small. Four types of coordinate systems are employed to derive the element deformation displacements.

The beam element is assumed to be straight and so slender that shear deformations can be neglected in comparison with bending deformations. It is assumed that the cross section of the beam remains plane and normal to the centroidal axis during deformation. The cross section is doubly symmetric, and the torsional stiffness will be treated independently from bending and axial stiffnesses. However, interaction of axial force and bending is considered [21].

3.2 COORDINATE SYSTEM OF SPACE FRAME

To derive the element deformation displacements, four types of coordinate systems are defined (Fig. 3.1):

1. The structural global coordinate system X_1, X_2, X_3.
2. The nodal coordinate system y_1, y_2, y_3; which is rigidly connected to each node. The initial directions of the y_1, y_2, y_3-axes are chosen to coincide with the global axes.
3. The element convected coordinate system x_1, x_2, x_3; which is associated with each element . The x_1-axis always moves with the beam element and passes through the centroids of the two end sections of the element; in the initial state the x_2, x_3-axes coincide with the principal axes of the cross section, and in the deformed state they are taken to be the average of the rotations of the two ends about the x_1-axis.
4. The end cross section (body [5],[17]) coordinate system $\bar{x}_1$, $\bar{x}_2$, $\bar{x}_3$; which rotates with the end cross section. The $\bar{x}_1$-axis is tangent to the deformed axis of the member, and the $\bar{x}_2$, $\bar{x}_3$-axes coincide with the principal axes of the cross section.

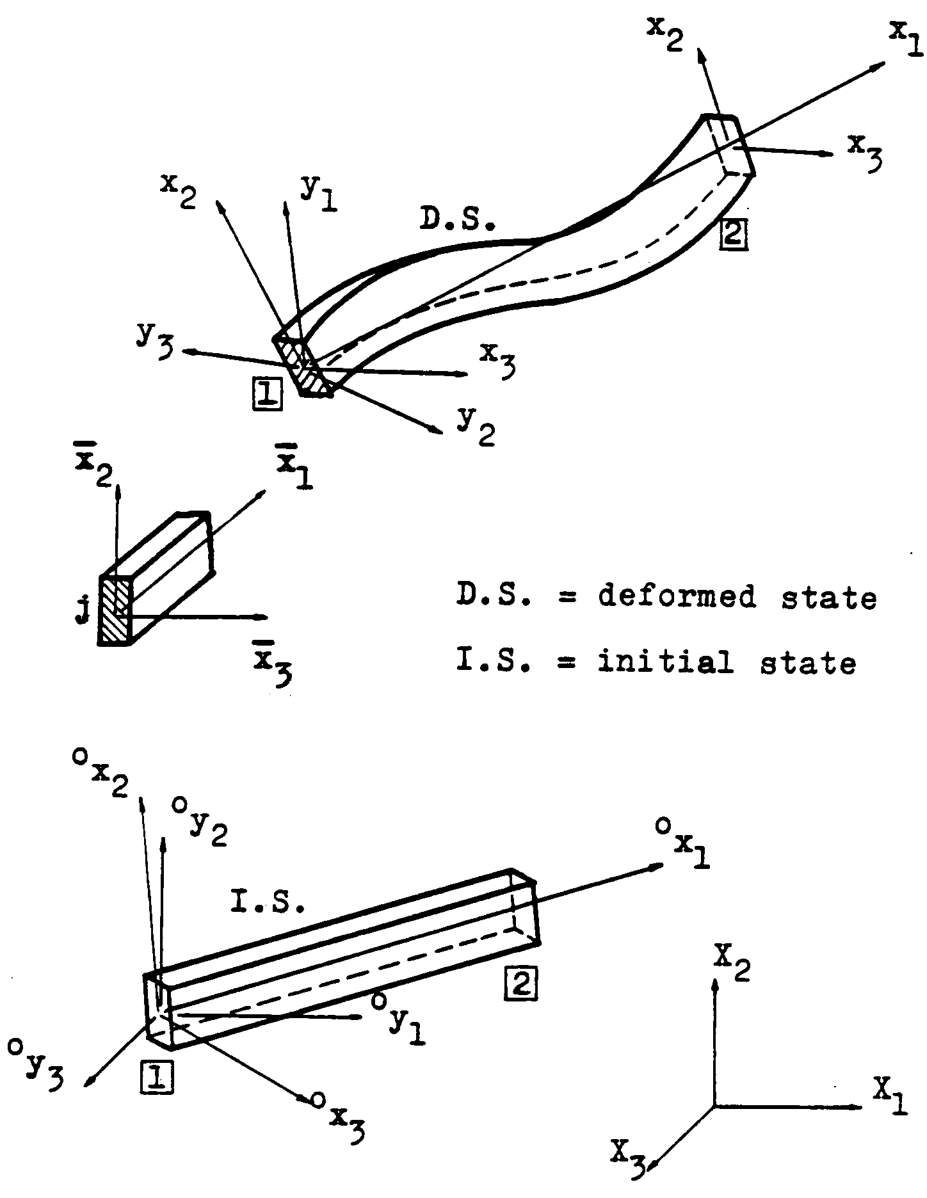


Fig. 3.1 Coordinate System of Space Frame

3.3 NODAL DISPLACEMENTS

The orientation of each deformed node j can be described in terms of the direction cosines of three node axes y_1, y_2, y_3 relative to to the global coordinates:

$$
{}^t\begin{Bmatrix} y_1 \\ y_2 \\ y_3 \end{Bmatrix}_j = {}^t\begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} \end{bmatrix}_j \begin{Bmatrix} X_1 \\ X_2 \\ X_3 \end{Bmatrix} \tag{3.1}
$$

or

$$
{}^t y_j = {}^t\alpha_j \; X
$$

where ${}^t\alpha_j$ is the nodal orientation matrix at time t. The orthogonal nodal orientation matrix can be partitioned as

$$
\alpha_j = [i_1 \; i_2 \; i_3]^T \tag{3.2}
$$

where

$$
i_\ell = [\alpha_{\ell 1} \; \alpha_{\ell 2} \; \alpha_{\ell 3}] \; ; \qquad \ell=1,2,3 \tag{3.3}
$$

and

$$
\alpha_{\ell m} = \cos \phi_{\ell m} \; ; \qquad m=1,2,3 \tag{3.4}
$$

$\alpha_{\ell m}$ is the direction cosine and $\phi_{\ell m}$ is the direction angle between the nodal y_ℓ-axis and the global X_m-axis.

In analyzing large displacements of a space frame, the deformed configuration of each node j can be represented in terms of a translation vector $[D_{j1}\ D_{j2}\ D_{j3}]^T$ relative to the global coordinate system and an node orientation matrix α_j.

Consider the increments of nodal displacements which are assumed to be small during a load step. The incremental displacement vector of node j is

$$\Delta U_j = \left\{ \begin{array}{c} \Delta U_{jt} \\ \hline \Delta U_{jr} \end{array} \right\} \tag{3.5}$$

in which ΔU_{jt}, ΔU_{jr} are the translational and rotational incremental vectors of node j in the global coordinate system:

$$\Delta U_{jt} = [\ \Delta D_{j1}\ \ \Delta D_{j2}\ \ \Delta D_{j3}\ ]^T \tag{3.6}$$

$$\Delta U_{jr} = [\ \Delta D_{j4}\ \ \Delta D_{j5}\ \ \Delta D_{j6}\ ]^T \tag{3.7}$$

where ΔD_{j1}, ΔD_{j2}, ΔD_{j3} are the incremental deflections at node j and ΔD_{j4}, ΔD_{j5}, ΔD_{j6} are the incremental rotations at node j.

According to appendix A, the node transformation from the configuration ${}^{t}C$ to the configuration ${}^{t+\Delta t}C$ is (Fig. 3.2)

$$ {}^{t+\Delta t}y_j = (I + {}^{t}R_j)\ {}^{t}y_j \tag{3.8}$$

where I is identity matrix and ${}^{t}R_j$ is the rotation matrix at node j, defined as

$$ {}^{t}R_j = \begin{bmatrix} 0 & \Delta^t\omega_{j3} & -\Delta^t\omega_{j2} \\ -\Delta^t\omega_{j3} & 0 & \Delta^t\omega_{j1} \\ \Delta^t\omega_{j2} & -\Delta^t\omega_{j1} & 0 \end{bmatrix} \tag{3.9}$$

where $\Delta\omega_{ji}$ is the incremental nodal rotation at node j about ${}^{t}y_i$-axis (Fig. 3.2), i= 1, 2, 3; defining

$$ \Delta^t\omega_j = \begin{bmatrix} \Delta^t\omega_{j1} \\ \Delta^t\omega_{j2} \\ \Delta^t\omega_{j3} \end{bmatrix} \tag{3.10}$$

where $\Delta^t\omega_j$ can be obtained from Eq. (3.1) as

$$ \Delta^t\omega_j = {}^{t}\alpha_j\ \Delta^t U_{jr} \tag{3.11}$$

Substituting Eq. (3.1) into Eq. (3.8) yields

$$ {}^{t+\Delta t}y_j = (I + {}^{t}R_j)\ {}^{t}\alpha_j\ X \tag{3.12}$$

$$ = \underbrace{{}^{t}\alpha_j\ X}_{{}^{t}y_j} + \underbrace{\Delta^t\alpha_j\ X}_{\Delta^t y_j} \tag{3.13}$$

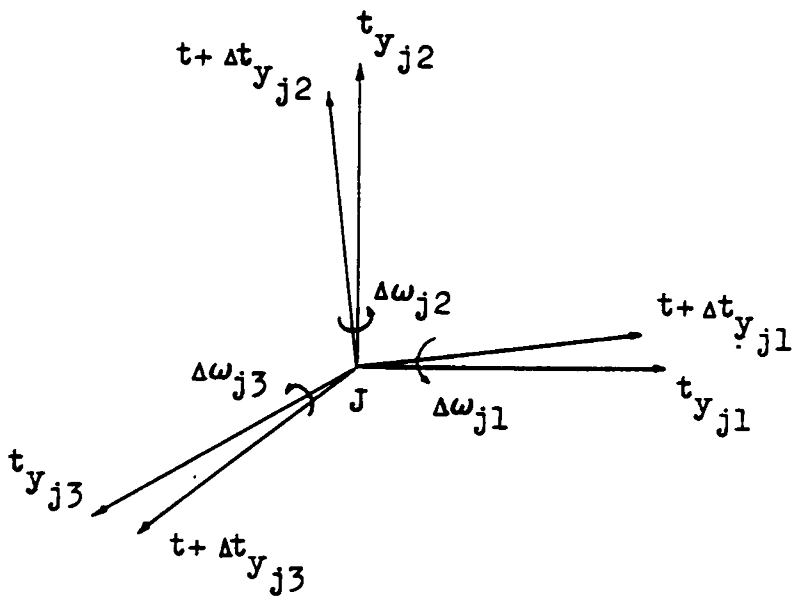


Fig. 3.2 Incremental Nodal Rotations

or

$$ {}^{t+\Delta t}y_j = {}^{t+\Delta t}\alpha_j \; X \tag{3.14} $$

where

$$ {}^{t+\Delta t}\alpha_j = {}^{t}\alpha_j + \Delta^{t}\alpha_j \tag{3.15} $$

$\Delta^{t}\alpha_j$ is the change of the node orientation matrix due to incremental rotations of node j, defined as

$$ \Delta^{t}\alpha_j = {}^{t}R_j \; {}^{t}\alpha_j \tag{3.16} $$

3.4 ELEMENT DEFORMATION DISPLACEMENTS

The relative deformation displacements of an element are referred to the rigid-convected (corotational) coordinate system x_1, x_2, x_3. Each discrete element is referred to a cartesian coordinate system that rotates and translates with the element but does not deform with the element (Fig. 3.1).

The element orientation is defined by the element orientation matrix, λ. The rows of λ are the direction cosines of x_1, x_2, x_3-axes relative to the global coordinates X_1, X_2, X_3, respectively.

$$
{}^{t}\begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = {}^{t}\begin{bmatrix} \lambda_{11} & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & \lambda_{22} & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & \lambda_{33} \end{bmatrix} \begin{Bmatrix} X_1 \\ X_2 \\ X_3 \end{Bmatrix} \tag{3.17}
$$

or

$$
{}^{t}x = {}^{t}\lambda \; X
$$

The orthogonal element orientation matrix can be partitioned as

$$
\lambda = \begin{bmatrix} j_1 \\ j_2 \\ j_3 \end{bmatrix} \tag{3.18}
$$

where

$$
j_\ell = [\; c_1 \; c_2 \; c_3 \;] \; ; \qquad \ell=1 \tag{3.19}
$$

$$
j_\ell = [\; \lambda_{\ell 1} \; \lambda_{\ell 2} \; \lambda_{\ell 3} \;] \; ; \qquad \ell=2,3 \tag{3.20}
$$

c_i = direction cosines of the local x_1-axis in any configuration C with respect to the global coordinates X_i

and

$$\lambda_{\ell m} = \cos \omega_{\ell m} \; ; \quad m=1,2,3 \tag{3.21}$$

$\lambda_{\ell m}$ is the direction consine and $\omega_{\ell m}$ is the direction angle between the local element x_ℓ-axis and the global X_m-axis.

The element orientation matrix of a space element depends on the orientations of the principal plane of the element defined by the local x_1, x_2-axes. The initial orientation matrix of the space element in configuration ^0C is defined in Eq. (2.109).

In the formulation developed here, no restrictions are made on the rotations and translations of the node. However, the relative deformations of the members are small such that we can apply the conventional beam-column theory to the member-force deformation relations.

The deformation displacements of a space frame element is represented by the end angles e_{13}, e_{23}, e_{12}, e_{22}; total twist ϕ_t, and relative axial displacemnet u (Fig. 3.3).

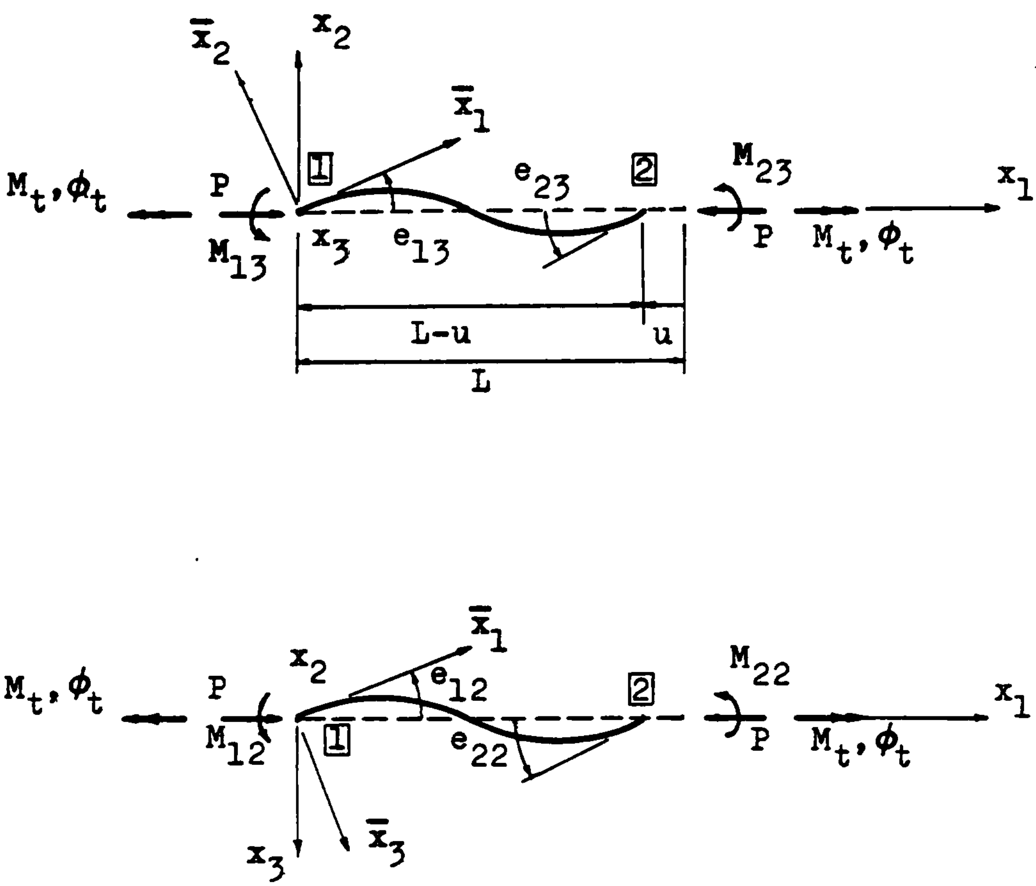


Fig. 3.3 Element Deformation Displacements and Associated Forces

Consider the increments of nodal displacements and element deformations which are assumed to be small during a load step. The element transformation from the configuration ${}^{t}C$ to the configuration ${}^{t+\Delta t}C$ is

$$ {}^{t+\Delta t}x = (I + {}^{t}R)\ {}^{t}x \tag{3.22} $$

where

$$ {}^{t}R = \begin{bmatrix} 0 & \Delta\psi_3 & -\Delta\psi_2 \\ -\Delta\psi_3 & 0 & \Delta\psi_1 \\ \Delta\psi_2 & -\Delta\psi_1 & 0 \end{bmatrix} \tag{3.23} $$

$$ \Delta\psi_1 = (\Delta d_{10} - \Delta d_4)/2 \tag{3.24} $$

$$ \Delta\psi_2 = -(\Delta d_9 - \Delta d_3)/{}^{t}\ell \tag{3.25} $$

$$ \Delta\psi_3 = (\Delta d_8 - \Delta d_2)/{}^{t}\ell \tag{3.26} $$

and the local element displacements are numbered as shown in Fig. 2.3. Substituting Eq. (3.17) into Eq. (3.22) yield

$$ \begin{aligned} {}^{t+\Delta t}x &= (I + {}^{t}R)\ {}^{t}\lambda\ X \\ &= \underbrace{{}^{t}\lambda\ X}_{{}^{t}x} + \underbrace{\Delta{}^{t}\lambda\ X}_{\Delta{}^{t}x} \end{aligned} \tag{3.27} $$

or

$$ {}^{t+\Delta t}x = {}^{t+\Delta t}\lambda \quad X \tag{3.28} $$

where

$$ {}^{t+\Delta t}\lambda = {}^{t}\lambda + {}_{\Delta}{}^{t}\lambda \tag{3.29} $$

${}_{\Delta}{}^{t}\lambda$ is the change of element orientation matrix due to incremental rotations of element, defined as

$$ {}_{\Delta}{}^{t}\lambda = {}^{t}R \; {}^{t}\lambda \tag{3.30} $$

The nodal coordinates y_1, y_2, y_3 are initially parallel to the global coordinates X_1, X_2, X_3. Thus, from Eqs. (3.1), (3.17) one obtains

$$ {}^{0}y_j = X = {}^{0}\lambda^T \; {}^{0}x \tag{3.31} $$

where left superscript '0' refers to the initial undeformed configuration ${}^{0}C$.

Because the member-end sections are rigidly connected to the respective nodes, Eq. (3.31) may be written in the deformed configuration ${}^{t+\Delta t}C$ [4] as

$$ {}^{t+\Delta t}y_j = {}^{0}\lambda^T \quad {}^{t+\Delta t}\bar{x}_j \tag{3.32} $$

where ${}^{t+\Delta t}\bar{x}_j$ is the coordinate vector of end section at node j in the ${}^{t+\Delta t}C$ configuration.

From Eqs. (3.32),(3.14),(3.17) one obtains

$$ {}^{t+\Delta t}\bar{x}_j = {}^{0}\lambda\ {}^{t+\Delta t}y_j = {}^{0}\lambda\ {}^{t+\Delta t}\alpha_j\ X = \left[{}^{0}\lambda\ {}^{t+\Delta t}\alpha_j\ {}^{t+\Delta t}\lambda^T\right]\ {}^{t+\Delta t}x \tag{3.33} $$

Since deformations are small

$$ \sin e_{jn} \approx e_{jn}\ ,\ \cos e_{jn} \approx 1 \tag{3.34} $$

where e_{jn} is the relative element rotation at node j about x_n-axis.

From Figs. 3.1 and 3.3 one obtains

$$ {}^{t+\Delta t}\bar{x}_j = \begin{bmatrix} 1 & \cos(90° - e_{j3}) & \cos(90° + e_{j2}) \\ \cos(90° + e_{j3}) & 1 & \cos(90° \pm \phi_t/2) \\ \cos(90° - e_{j2}) & \cos(90° \mp \phi_t/2) & 1 \end{bmatrix} {}^{t+\Delta t}x \tag{3.35} $$

where j is 1 for the left end and 2 for the right end; e_{j2}, e_{j3} are the relative end rotations at node j about the x_2, x_3-axes respectively, ϕ_t is the relative angle of twist of the element ends about the x_1-axis, and the upper and lower signs apply to nodes 1 and 2, respectively.

Substituting Eq. (3.34) into Eq. (3.35) yields

$$
{}^{t+\Delta t}\bar{x}_j = \begin{bmatrix} 1 & e_{j3} & -e_{j2} \\ -e_{j3} & 1 & \mp \phi_t/2 \\ e_{j2} & \pm \frac{\phi_t}{2} & 1 \end{bmatrix} {}^{t+\Delta t}x \tag{3.36}
$$

Comparison of Eqs. (3.33) and (3.36) yields

$$
{}^{t+\Delta t}e_{j2} = [{}^{o}\lambda \; {}^{t+\Delta t}\alpha_j \; {}^{t+\Delta t}\lambda^T]_{31} = - [{}^{o}\lambda \; {}^{t+\Delta t}\alpha_j \; {}^{t+\Delta t}\lambda^T]_{13} \tag{3.37}
$$

$$
{}^{t+\Delta t}e_{j3} = - [{}^{o}\lambda \; {}^{t+\Delta t}\alpha_j \; {}^{t+\Delta t}\lambda^T]_{21} = [{}^{o}\lambda \; {}^{t+\Delta t}\alpha_j \; {}^{t+\Delta t}\lambda^T]_{12} \tag{3.38}
$$

$$
{}^{t+\Delta t}\phi_t/2 = \mp [{}^{o}\lambda \; {}^{t+\Delta t}\alpha_j \; {}^{t+\Delta t}\lambda^T]_{23} = \pm [{}^{o}\lambda \; {}^{t+\Delta t}\alpha_j \; {}^{t+\Delta t}\lambda^T]_{32} \tag{3.39}
$$

where $[\quad]_{mn}$ denotes the entry in the m^{th} row and n^{th} column of the matrix.

From Eqs. (3.1) and (3.32) we obtain

$$
{}^{t+\Delta t}\bar{x}_j = {}^{t+\Delta t}p_j \; X \tag{3.40}
$$

where

$$
{}^{t+\Delta t}p_j = {}^{O}\lambda \; {}^{t+\Delta t}\alpha_j \tag{3.41}
$$

${}^{t+\Delta t}p_j$, ${}^{t+\Delta t}\alpha_j$ are the end section and node orientation matrices at node j in the configuration at time t+Δt, respectively. The rows of p_j represent the direction cosines of the normal and principal directions of the corresponding end section at joint j with respect to the global coordinates X_1, X_2, X_3.

The deformation desplacements can be expressed directly from Eqs. (3.37),(3.38),(3.39),(3.17) and (2.109) as

$$ {}^{t+\Delta t}e_{j2} = \left[\frac{-{}^0c_3}{{}^0\ell} \quad 0 \quad \frac{{}^0c_1}{{}^0\ell}\right] {}^{t+\Delta t}\alpha_j \begin{bmatrix} {}^{t+\Delta t}c_1 \\ {}^{t+\Delta t}c_2 \\ {}^{t+\Delta t}c_3 \end{bmatrix} \tag{3.42} $$

$$ {}^{t+\Delta t}e_{j3} = -\left[\frac{-{}^0c_1\,{}^0c_2}{{}^0\ell} \quad {}^0\ell \quad \frac{-{}^0c_2\,{}^0c_3}{{}^0\ell}\right] {}^{t+\Delta t}\alpha_j \begin{bmatrix} {}^{t+\Delta t}c_1 \\ {}^{t+\Delta t}c_2 \\ {}^{t+\Delta t}c_3 \end{bmatrix} \tag{3.43} $$

$$ {}^{t+\Delta t}\phi_t = \mp 2\left[\frac{-{}^0c_1\,{}^0c_2}{{}^0\ell} \quad {}^0\ell \quad \frac{-{}^0c_2\,{}^0c_3}{{}^0\ell}\right] {}^{t+\Delta t}\alpha_j \begin{bmatrix} {}^{t+\Delta t}\lambda_{31} \\ {}^{t+\Delta t}\lambda_{32} \\ {}^{t+\Delta t}\lambda_{33} \end{bmatrix} \tag{3.44} $$

and

$$ u = {}^0L - {}^{t+\Delta t}L \tag{3.45} $$

Chapter IV

FINITE ELEMENT MODEL

4.1 INTRODUCTION

To solve the incremental equilibrium equation in the U.L. formulation we must to evaluate the finite element matrices in Eq. (2.55). The local element secant stiffness matrix of the plane frame element, developed by Katzenberger [48], is extended to form the local element secant stiffness matrix of the space frame element in which the torsional forces are treated independently and obtained by linear theory.

For a straight small strain beam of constant cross section in the convected coordinate system, Hermitian interpolation functions are employed to interpolate the transverse bending displacements, and Lagrange interpolation functions are used to interpolate the axial and torsional displacements [9],[33],[34].

For convenience, the left superscripts and subscripts are not shown in this formulation.

4.2 INTERPOLATION FUNCTIONS FOR INCREMENTAL DISPLACEMENTS

The natural coordinate system in Fig. 4.1 is used. The interpolation matrix of Eq. (2.24) contains the entries [9], [33], [34]:

$$
\begin{aligned}
h_1 &= 1 - \xi \\
h_2 &= 1 - 3\xi^2 + 2\xi^3 \\
h_3 &= 1 - 3\xi^2 + 2\xi^3 \\
h_5 &= -L\xi + 2L\xi^2 - L\xi^3 \\
h_6 &= L\xi - 2L\xi^2 + L\xi^3 \\
h_7 &= \xi \\
h_8 &= 3\xi^2 - 2\xi^3 \\
h_9 &= 3\xi^2 - 2\xi^3 \\
h_{11} &= L\xi^2 - L\xi^3 \\
h_{12} &= -L\xi^2 + L\xi^3
\end{aligned}
\tag{4.1}
$$

where

$$\xi = x_1 / L \tag{4.2}$$

Eqs. (4.1) yield the 1st and 2nd derivatives of the interpolation functions with respect to ξ-axis:

$$
\begin{aligned}
h_{1,1} &= -1 \\
h_{2,1} &= -6\xi + 6\xi^2 \\
h_{3,1} &= -6\xi + 6\xi^2 \\
h_{5,1} &= -L + 4L\xi - 3L\xi^2
\end{aligned}
$$

$$
\begin{aligned}
h_{6,1} &= L - 4L\xi + 3L\xi^2 \\
h_{7,1} &= 1 \\
h_{8,1} &= 6\xi - 6\xi^2 \\
h_{9,1} &= 6\xi - 6\xi^2 \\
h_{11,1} &= 2L\xi - 3L\xi^2 \\
h_{12,1} &= -2L\xi + 3L\xi^2
\end{aligned}
\tag{4.3}
$$

and

$$
\begin{aligned}
h_{2,11} &= -6 + 12\xi \\
h_{3,11} &= -6 + 12\xi \\
h_{5,11} &= 4L - 6L\xi \\
h_{6,11} &= -4L + 6L\xi \\
h_{8,11} &= 6 - 12\xi \\
h_{9,11} &= 6 - 12\xi \\
h_{11,11} &= 2L - 6L\xi \\
h_{12,11} &= -2L + 6L\xi
\end{aligned}
\tag{4.4}
$$

Using Eqs. (4.1),(4.3), and (4.4), the linear and nonlinear strain-displacement transformation matrices that are required to evaluate the tangent stiffness matrix and nodal force vector of an element can directly be evaluated.

ξ = natural coordinate = x_1 / L

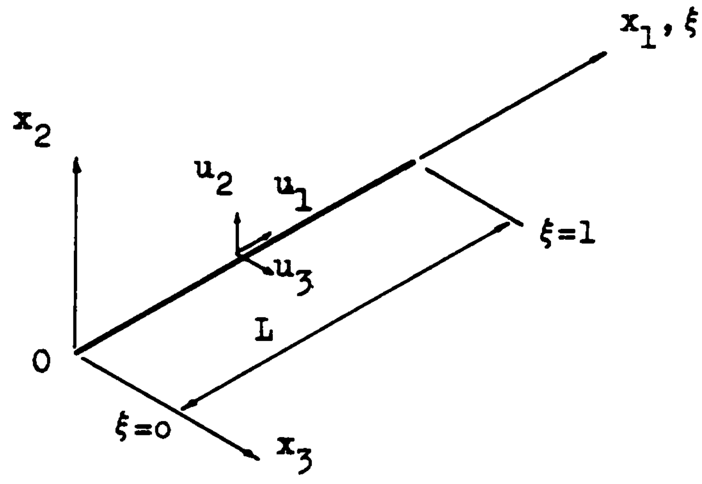


Fig. 4.1 Local and Natural Coordinate Systems

4.3 LINEAR STRAIN INCREMENTAL STIFFNESS MATRIX

Employing Eqs. (4.3),(4.4) and substituting Eq. (2.49) into Eq. (2.45) one obtains the linear strain incremental stiffness matrix of the local element model with the coordinate system in Fig. 2.3 [28]

$$k_L = \int_V B_L^T E B_L \, dV = \begin{array}{c} \begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{array} \\ \left[\begin{array}{cccccc|cccccc}
\frac{EA}{L} & 0 & 0 & 0 & 0 & 0 & \frac{-EA}{L} & 0 & 0 & 0 & 0 & 0 \\
 & \frac{12EI_3}{L^3} & 0 & 0 & 0 & \frac{6EI_3}{L^2} & 0 & \frac{-12EI_3}{L^3} & 0 & 0 & 0 & \frac{6EI_3}{L^2} \\
 & & \frac{12EI_2}{L^3} & 0 & \frac{-6EI_2}{L^2} & 0 & 0 & 0 & \frac{-12EI_2}{L^3} & 0 & \frac{-6EI_2}{L^2} & 0 \\
 & & & \frac{GJ}{L} & 0 & 0 & 0 & 0 & 0 & \frac{-GJ}{L} & 0 & 0 \\
 & & & & \frac{4EI_2}{L} & 0 & 0 & 0 & \frac{6EI_2}{L^2} & 0 & \frac{2EI_2}{L} & 0 \\
 & & & & & \frac{4EI_3}{L} & 0 & \frac{-6EI_3}{L^2} & 0 & 0 & 0 & \frac{2EI_3}{L} \\
\hline
 & & & & & & \frac{EA}{L} & 0 & 0 & 0 & 0 & 0 \\
 & & & & & & & \frac{12EI_3}{L^3} & 0 & 0 & 0 & \frac{-6EI_3}{L^2} \\
 & \text{symmetric} & & & & & & & \frac{12EI_2}{L^3} & 0 & \frac{6EI_2}{L^2} & 0 \\
 & & & & & & & & & \frac{GJ}{L} & 0 & 0 \\
 & & & & & & & & & & \frac{4EI_2}{L} & 0 \\
 & & & & & & & & & & & \frac{4EI_3}{L}
\end{array}\right] \end{array} \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \end{array} \quad (4.5)$$

where

A = area of cross section

L = undeformed element length

I_n = moment of inertia about the x_n-axis

G = shear modulus of elasticity

J = polar moment of inertia

4.4 NONLINEAR STRAIN INCREMENTAL STIFFNESS MATRIX

Similarly, employing Eqs. (4.3),(4.4) and substituting Eqs. (2.36), (2.43) into Eq. (2.46) one obtains the nonlinear strain incremental stiffness matrix of the local element model with coordinate system in Fig. 2.3 [18]

$$
k_{NL}=\int_V B_{NL}^T \tau\, B_{NL}\, dV = T
\begin{array}{c}
\begin{array}{cccccccccccc}
\scriptstyle 1 & \scriptstyle 2 & \scriptstyle 3 & \scriptstyle 4 & \scriptstyle 5 & \scriptstyle 6 & \scriptstyle 7 & \scriptstyle 8 & \scriptstyle 9 & \scriptstyle 10 & \scriptstyle 11 & \scriptstyle 12
\end{array} \\
\left[\begin{array}{cccccc|cccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & \frac{6}{5L} & 0 & 0 & 0 & \frac{1}{10} & 0 & \frac{-6}{5L} & 0 & 0 & 0 & \frac{1}{10} \\
 & & \frac{6}{5L} & 0 & \frac{-1}{10} & 0 & 0 & 0 & \frac{-6}{5L} & 0 & \frac{-1}{10} & 0 \\
 & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & & & & \frac{2L}{15} & 0 & 0 & 0 & \frac{1}{10} & 0 & \frac{-L}{30} & 0 \\
 & & & & & \frac{2L}{15} & 0 & \frac{-1}{10} & 0 & 0 & 0 & \frac{-L}{30} \\
\hline
 & & & & & & 0 & 0 & 0 & 0 & 0 & 0 \\
 & & & & & & & \frac{6}{5L} & 0 & 0 & 0 & \frac{-1}{10} \\
 & & \text{symmetric} & & & & & & \frac{6}{5L} & 0 & \frac{1}{10} & 0 \\
 & & & & & & & & & 0 & 0 & 0 \\
 & & & & & & & & & & \frac{2L}{15} & 0 \\
 & & & & & & & & & & & \frac{2L}{15}
\end{array}\right]
\end{array}
\begin{array}{c}
\\ \scriptstyle 1 \\ \scriptstyle 2 \\ \scriptstyle 3 \\ \scriptstyle 4 \\ \scriptstyle 5 \\ \scriptstyle 6 \\ \scriptstyle 7 \\ \scriptstyle 8 \\ \scriptstyle 9 \\ \scriptstyle 10 \\ \scriptstyle 11 \\ \scriptstyle 12
\end{array}
\qquad (4.6)
$$

where $T = \sigma A$ is positive in tension.

The nonlinear strain incremental stiffness matrix k_{NL} is independent of elastic properties. It depends only on the element's geometry, displacement field, and the current state of stress level.

4.5 LOCAL STRAIN INCREMENTAL STIFFNESS MATRIX

The local strain incremental stiffness matrix in Eq. (2.53) is

$$k = k_L + k_{NL} = \begin{array}{c} \begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{array} \\ \left[\begin{array}{cccccc|cccccc} T_1 & 0 & 0 & 0 & 0 & 0 & -T_1 & 0 & 0 & 0 & 0 & 0 \\ & T_2 & 0 & 0 & 0 & T_3 & 0 & -T_2 & 0 & 0 & 0 & T_3 \\ & & T_4 & 0 & T_5 & 0 & 0 & 0 & -T_4 & 0 & T_5 & 0 \\ & & & T_6 & 0 & 0 & 0 & 0 & 0 & -T_6 & 0 & 0 \\ & & & & T_7 & 0 & 0 & 0 & -T_5 & 0 & T_8 & 0 \\ & & & & & T_9 & 0 & -T_3 & 0 & 0 & 0 & T_{10} \\ \hline & & & & & & T_1 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & T_2 & 0 & 0 & 0 & -T_3 \\ & & \text{symmetric} & & & & & & T_4 & 0 & -T_5 & 0 \\ & & & & & & & & & T_6 & 0 & 0 \\ & & & & & & & & & & T_7 & 0 \\ & & & & & & & & & & & T_9 \end{array}\right] \end{array} \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \end{array} \tag{4.7}$$

where

$$
\begin{aligned}
T_1 &= EA / L \\
T_2 &= 12EI_3 / L^3 + 6T / 5L \\
T_3 &= 6EI_3 / L^2 + T / 10 \\
T_4 &= 12EI_2 / L^3 + 6T / 5L \\
T_5 &= -6EI_2 / L^2 - T / 10 \\
T_6 &= GJ / L \\
T_7 &= 4EI_2 / L + 2TL / 15 \\
T_8 &= 2EI_2 / L - TL / 30 \\
T_9 &= 4EL_3 / L + 2TL / 15 \\
T_{10} &= 2EI_3 / L - TL / 30
\end{aligned}
\tag{4.8}
$$

In Eqs. (4.8) the first terms are contributed by k_L and the second terms by k_{NL}.

4.6 GLOBAL STRAIN INCREMENTAL STIFFNESS MATRIX

The global strain incremental stiffness matarix for element i may be obtained by the standard transformation in Eq. (2.50) [28]

$$K^i = \Lambda^T k \Lambda \tag{4.9}$$

where Λ is the transformation matrix defined in Eq. (2.54). Substituting Eqs. (4.7),(3.17) into Eq. (4.9), one can exppress K^i in terms of coefficient functions g_i, i = 1, 2,.... 42. The following index matrix locates g_i in the stiffness matrix storing the subscripts and negative signs of g_i [28].

$$\mathrm{INDEX} = \left[\begin{array}{cccccc|cccccc}
1 & 2 & 3 & 4 & 5 & 6 & -1 & -2 & -3 & 7 & 8 & 9 \\
2 & 10 & 11 & 12 & 13 & 14 & -2 & -10 & -11 & 15 & 16 & 17 \\
3 & 11 & 18 & 19 & 20 & 21 & -3 & -11 & -18 & 22 & 23 & 24 \\
4 & 12 & 19 & 25 & 26 & 27 & -4 & -12 & -19 & 28 & 29 & 30 \\
5 & 13 & 20 & 26 & 31 & 32 & -5 & -13 & -20 & 29 & 33 & 34 \\
6 & 14 & 21 & 27 & 32 & 35 & -6 & -14 & -21 & 30 & 34 & 36 \\
\hline
-1 & -2 & -3 & -4 & -5 & -6 & 1 & 2 & 3 & -7 & -8 & -9 \\
-2 & -10 & -11 & -12 & -13 & -14 & 2 & 10 & 11 & -15 & -16 & -17 \\
-3 & -11 & -18 & -19 & -20 & -21 & 3 & 11 & 18 & -22 & -23 & -24 \\
7 & 15 & 22 & 28 & 29 & 30 & -7 & -15 & -22 & 37 & 38 & 39 \\
8 & 16 & 23 & 29 & 33 & 34 & -8 & -16 & -23 & 38 & 40 & 41 \\
9 & 17 & 24 & 30 & 34 & 36 & -9 & -17 & -24 & 39 & 41 & 42
\end{array}\right] \tag{4.10}$$

the coefficient functions are

$$\begin{aligned}
g_1 &= T_1\, c_1^2 + T_2\, \lambda_{21}^2 + T_4\, \lambda_{31}^2 \\
g_2 &= T_1\, c_1\, c_2 + T_2\, \lambda_{21}\, \lambda_{22} + T_4\, \lambda_{31}\, \lambda_{32} \\
g_3 &= T_1\, c_1\, c_3 + T_2\, \lambda_{21}\, \lambda_{23} + T_4\, \lambda_{31}\, \lambda_{33}
\end{aligned} \tag{4.11}$$

$$g_4 = T_3 \lambda_{21} \lambda_{31} + T_5 \lambda_{31} \lambda_{21}$$

$$g_5 = T_3 \lambda_{21} \lambda_{32} + T_5 \lambda_{31} \lambda_{22}$$

$$g_6 = T_3 \lambda_{21} \lambda_{33} + T_5 \lambda_{31} \lambda_{23}$$

$$g_7 = T_3 \lambda_{21} \lambda_{31} + T_5 \lambda_{31} \lambda_{21}$$

$$g_8 = T_3 \lambda_{21} \lambda_{32} + T_5 \lambda_{31} \lambda_{22}$$

$$g_9 = T_3 \lambda_{21} \lambda_{33} + T_5 \lambda_{31} \lambda_{23}$$

$$g_{10} = T_1 c_2^2 + T_2 \lambda_{22}^2 + T_4 \lambda_{32}^2$$

$$g_{11} = T_1 c_2 c_3 + T_2 \lambda_{22} \lambda_{23} + T_4 \lambda_{32} \lambda_{33}$$

$$g_{12} = T_3 \lambda_{22} \lambda_{31} + T_5 \lambda_{32} \lambda_{21}$$

$$g_{13} = T_3 \lambda_{22} \lambda_{32} + T_5 \lambda_{32} \lambda_{22} \tag{4.11}$$

$$g_{14} = T_3 \lambda_{22} \lambda_{33} + T_5 \lambda_{32} \lambda_{23}$$

$$g_{15} = T_3 \lambda_{22} \lambda_{31} + T_5 \lambda_{32} \lambda_{21}$$

$$g_{16} = T_3 \lambda_{22} \lambda_{32} + T_5 \lambda_{32} \lambda_{22}$$

$$g_{17} = T_3 \lambda_{22} \lambda_{33} + T_5 \lambda_{32} \lambda_{23}$$

$$g_{18} = T_1 c_3^2 + T_2 \lambda_{23}^2 + T_4 \lambda_{33}^2$$

$$g_{19} = T_3 \lambda_{23} \lambda_{31} + T_5 \lambda_{33} \lambda_{21}$$

$$g_{20} = T_3 \lambda_{23} \lambda_{32} + T_5 \lambda_{33} \lambda_{22}$$

$$g_{21} = T_3 \lambda_{23} \lambda_{33} + T_5 \lambda_{33} \lambda_{23}$$

$$g_{22} = T_3 \lambda_{23} \lambda_{31} + T_5 \lambda_{33} \lambda_{21}$$

$$
\begin{aligned}
g_{23} &= T_3 \lambda_{23} \lambda_{32} + T_5 \lambda_{33} \lambda_{22} \\
g_{24} &= T_3 \lambda_{23} \lambda_{33} + T_5 \lambda_{33} \lambda_{23} \\
g_{25} &= T_6 c_1^2 + T_9 \lambda_{31}^2 + T_7 \lambda_{21}^2 \\
g_{26} &= T_6 c_1 c_2 + T_9 \lambda_{31} \lambda_{32} + T_7 \lambda_{21} \lambda_{22} \\
g_{27} &= T_6 c_1 c_3 + T_9 \lambda_{31} \lambda_{33} + T_7 \lambda_{21} \lambda_{23} \\
g_{28} &= - T_6 c_1^2 + T_{10} \lambda_{31}^2 + T_8 \lambda_{21}^2 \\
g_{29} &= -T_6 c_1 c_2 + T_{10} \lambda_{31} \lambda_{32} + T_8 \lambda_{21} \lambda_{22} \\
g_{30} &= -T_6 c_1 c_3 + T_{10} \lambda_{31} \lambda_{33} + T_8 \lambda_{21} \lambda_{23} \\
g_{31} &= T_6 c_2^2 + T_9 \lambda_{32}^2 + T_7 \lambda_{22}^2 \\
g_{32} &= T_6 c_2 c_3 + T_9 \lambda_{32} \lambda_{33} + T_7 \lambda_{22} \lambda_{23} \\
g_{33} &= -T_6 c_2^2 + T_{10} \lambda_{32}^2 + T_8 \lambda_{22}^2 \\
g_{34} &= -T_6 c_2 c_3 + T_{10} \lambda_{32}\lambda_{33} + T_8 \lambda_{22} \lambda_{23} \\
g_{35} &= T_6 c_3^2 + T_9 \lambda_{33}^2 + T_7 \lambda_{23}^2 \\
g_{36} &= -T_6 c_3^2 + T_{10} \lambda_{33}^2 + T_8 \lambda_{23}^2 \\
g_{37} &= T_6 c_1^2 + T_9 \lambda_{31}^2 + T_7 \lambda_{21}^2 \\
g_{38} &= T_6 c_1 c_2 + T_9 \lambda_{31} \lambda_{32} + T_7 \lambda_{21} \lambda_{22} \\
g_{39} &= T_6 c_1 c_3 + T_9 \lambda_{31} \lambda_{33} + T_7 \lambda_{21} \lambda_{23} \\
g_{40} &= T_6 c_2^2 + T_9 \lambda_{32}^2 + T_7 \lambda_{22}^2 \\
g_{41} &= T_6 c_2 c_3 + T_9 \lambda_{32} \lambda_{33} + T_7 \lambda_{22} \lambda_{23} \\
g_{42} &= T_6 c_3^2 + T_9 \lambda_{33}^2 + T_7 \lambda_{23}^2
\end{aligned}
\qquad (4.11)
$$

4.7 LOCAL ELEMENT FORCES

The relative deformation displacements and forces can be defined by Figs. 4.2 and 4.3, corresponding to the local x_1-x_2 and x_1-x_3 planes, respectively.

The deformation forces in the x_1-x_2 plane (Fig. 4.2) can be obtained by the equilibrium equation

$$p' = \hat{k}'_s \; e' \tag{4.12}$$

where

$$p' = [\; T \quad M_{13} \quad M_{23} \;]^T \tag{4.13}$$

$$e' = [\; e_1 \quad e_{13} \quad e_{23} \;]^T \tag{4.14}$$

p' is the deformation force vector in the x_1-x_2 plane, $\hat{k}'_s$ is the local secant stiffness matrix of the element in the x_1-x_2 plane, e' is the deformation displacement vector in the x_1-x_2 plane, T is the axial force (positive in tension), e_{mn} are the relative end rotations (Figs. 4.2 and 4.3), M_{mn} are the bending moments corresponding to e_{mn}, and e_1 is the relative axial elongation.

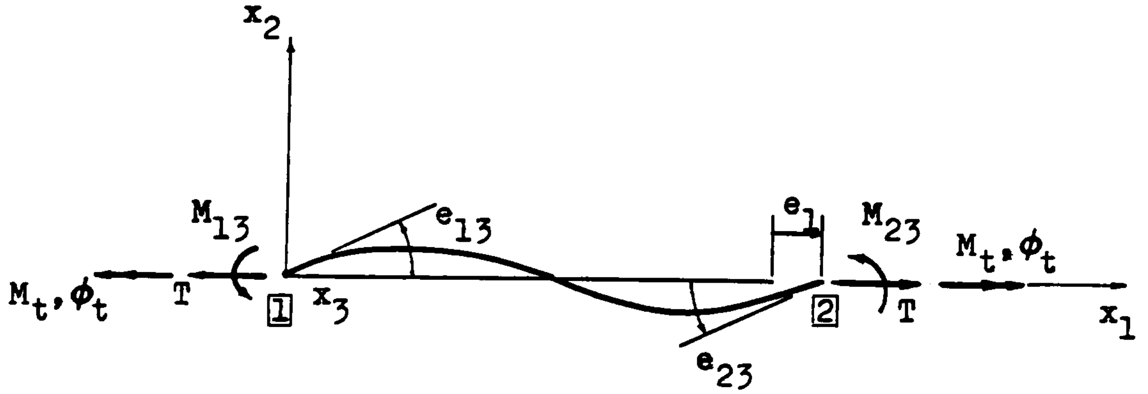


Fig. 4.2 Deformation Displacements and Forces in x_1–x_2 Plane

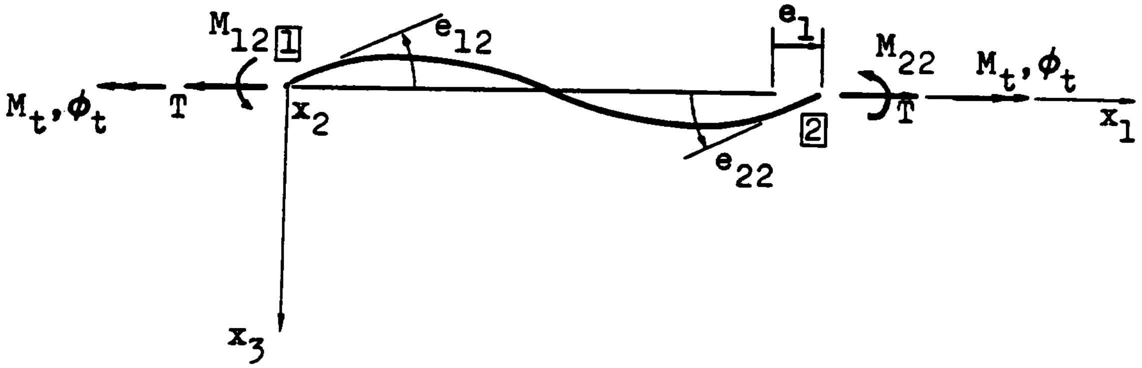


Fig. 4.3 Deformation Displacements and Forces in x_1–x_3 Plane

Katzenberger [48] derived the local element secant stiffness matrix and the deformation forces for the plane frame element. The symmetric form of the secant stiffness matrix in the x_1-x_2 plane is

$$\hat{k}_s' = \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \left[\begin{array}{ccc} \frac{EA}{L} & \frac{EA}{60}(4e_{13} - e_{23}) & \frac{EA}{60}(4e_{23} - e_{13}) \\ & \frac{4EI_3}{L} + \frac{EA}{15}e_1 + \frac{EAL}{420}(12e_{13}^2 - 3e_{13}e_{23} + e_{23}^2) & \frac{2EI_3}{L} - \frac{EA}{60}e_1 + \frac{EAL}{840}(-3e_{13}^2 + 4e_{13}e_{23} - 3e_{23}^2) \\ \text{symm} & & \frac{4EI_3}{L} + \frac{EA}{15}e_1 + \frac{EAL}{420}(e_{13}^2 - 3e_{13}e_{23} + 12e_{23}^2) \end{array} \right] \end{array} \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}$$

(4.15)

Similarly, the deformation forces in the x_1-x_3 plane (Fig. 4.3) are

$$p'' = \hat{k}_s'' \, e'' \qquad (4.16)$$

where

$$p'' = [\; T \quad M_{12} \quad M_{22} \;]^T \qquad (4.17)$$

$$e'' = [\; e_1 \quad e_{12} \quad e_{22} \;]^T \qquad (4.18)$$

p'' is the deformation force vector in x_1-x_3 plane, $\hat{k}_s''$ is the local secant stiffness matrix of the element in the x_1-x_3 plane, and e'' is the deformation displacement vector in the x_1-x_3 plane.

Comparing Fig. 4.2 with Fig. 4.3, we obtain the local element secant stiffness matrix in x_1-x_3 plane, similar to Eq. (4.15):

$$\hat{k}_s^{*} = \begin{array}{c} \begin{array}{ccc} 1 & 4 & 5 \end{array} \\ \left[\begin{array}{ccc} \frac{EA}{L} & \frac{EA}{60}(4e_{12} - e_{22}) & \frac{EA}{60}(4e_{22} - e_{12}) \\ & \frac{4EI_2}{L} + \frac{EA}{15} e_1 + \frac{EAL}{420}(12e_{12}^2 - 3e_{12}e_{22} + e_{22}^2) & \frac{2EI_2}{L} - \frac{EA}{60} e_1 + \frac{EAL}{840}(-3e_{12}^2 + 4e_{12}e_{22} - 3e_{22}^2) \\ \text{symm} & & \frac{4EI_2}{L} + \frac{EA}{15} e_1 + \frac{EAL}{420}(e_{12}^2 - 3e_{12}e_{22} + 12e_{22}^2) \end{array} \right] \end{array} \begin{array}{c} \\ 1 \\ 4 \\ 5 \end{array}$$

(4.19)

Combining Eqs. (4.12) and (4.16), one obtains the deformation forces of the space frame element

$$p = \hat{k}_s \, e \tag{4.20}$$

where

$$p = [\ T \quad M_{13} \quad M_{23} \quad M_{12} \quad M_{22}\ ]^T \tag{4.21}$$

$$e = [\ e_1 \quad e_{13} \quad e_{23} \quad e_{12} \quad e_{22}\ ]^T \tag{4.22}$$

p, e are the deformation force and displacement vectors of the space frame element, respectively.

The relative element forces and displacements are as shown in Fig. 4.4. The superposition of Eqs. (4.15) and (4.19) in accordance with the sequence in Eqs. (4.21) and

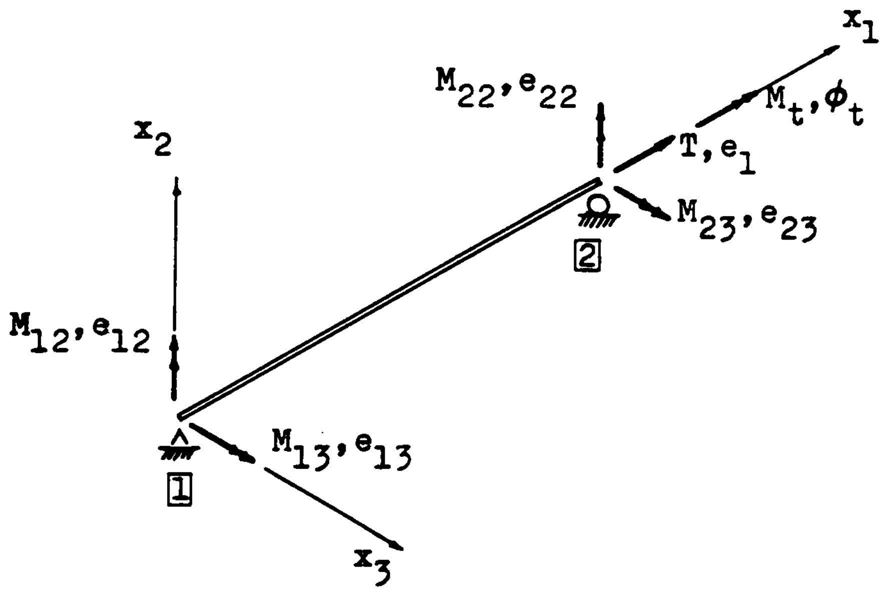


Fig. 4.4 Relative Element Forces and Displacements in Space Frame Element

(4.22) yields the local element secant stiffness matrix of space frame

$$\hat{k}_s = \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[\begin{array}{ccccc} EA/L & \frac{EA}{60}(4e_{13}-e_{23}) & \frac{EA}{60}(4e_{23}-e_{13}) & \frac{EA}{60}(4e_{12}-e_{22}) & \frac{EA}{60}(4e_{22}-e_{12}) \\ \frac{EA}{60}(4e_{13}-e_{23}) & \frac{4EI_3}{L}+\frac{EA}{15}e_1+\frac{EAL}{420}(12e_{13}^2-3e_{13}e_{23}+e_{23}^2) & \frac{2EI_3}{L}-\frac{EA}{60}e_1+\frac{EAL}{840}(-3e_{13}^2+4e_{13}e_{23}-3e_{23}^2) & 0 & 0 \\ \frac{EA}{60}(4e_{23}-e_{13}) & \frac{2EI_3}{L}-\frac{EA}{60}e_1+\frac{EAL}{840}(-3e_{13}^2+4e_{13}e_{23}-3e_{23}^2) & \frac{4EI_3}{L}+\frac{EA}{15}e_1+\frac{EAL}{420}(e_{13}^2-3e_{13}e_{23}+12e_{23}^2) & 0 & 0 \\ \frac{EA}{60}(4e_{12}-e_{22}) & 0 & 0 & \frac{4EI_2}{L}+\frac{EA}{15}e_1+\frac{EAL}{420}(12e_{12}^2-3e_{12}e_{22}+e_{22}^2) & \frac{2EI_2}{L}-\frac{EA}{60}e_1+\frac{EAL}{840}(-3e_{12}^2+4e_{12}e_{22}-3e_{22}^2) \\ \frac{EA}{60}(4e_{22}-e_{12}) & 0 & 0 & \frac{2EI_2}{L}-\frac{EA}{60}e_1+\frac{EAL}{840}(-3e_{12}^2+4e_{12}e_{22}-3e_{22}^2) & \frac{4EI_2}{L}+\frac{EA}{15}e_1+\frac{EAL}{420}(e_{12}^2-3e_{12}e_{22}+12e_{22}^2) \end{array}\right] \end{array} \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \tag{4.23}$$

From equilibrium conditions one obtains the local element forces corresponding to Fig. 2.3:

$$f_1 = -T$$

$$
\begin{aligned}
f_2 &= (M_{13}+M_{23}) \;/\; L \\
f_3 &= -(M_{12}+M_{22}) \;/\; L \\
f_5 &= M_{12} \\
f_6 &= M_{13} \\
f_7 &= -f_1 \\
f_8 &= -f_2 \\
f_9 &= -f_3 \\
f_{11} &= M_{22} \\
f_{12} &= M_{23}
\end{aligned}
\tag{4.24}
$$

The torsional forces are treated independently and can be obtained from the linear theory [28]

$$
\begin{aligned}
f_4 &= -(GJ/L)\,\phi_t \\
f_{10} &= -f_4
\end{aligned}
\tag{4.25}
$$

where ϕ_t is the relative rotation of element ends about the element axis.

4.8 GLOBAL ELEMENT FORCES

The global element forces for element i can be obtained from Eq. (2.51):

$$F^i = \Lambda^T f \tag{4.26}$$

where Λ is defined in Eq. (3.17). The components of F^i are

$$F_1 = c_1 f_1 + \lambda_{21} f_2 + \lambda_{31} f_3$$

$$F_2 = c_2 f_1 + \lambda_{22} f_2 + \lambda_{32} f_3$$

$$F_3 = c_3 f_1 + \lambda_{23} f_2 + \lambda_{33} f_3$$

$$F_4 = c_1 f_4 + \lambda_{21} f_5 + \lambda_{31} f_6$$

$$F_5 = c_2 f_4 + \lambda_{22} f_5 + \lambda_{32} f_6$$

$$F_6 = c_3 f_4 + \lambda_{23} f_5 + \lambda_{33} f_6$$

$$F_7 = c_1 f_7 + \lambda_{21} f_8 + \lambda_{31} f_9$$

$$F_8 = c_2 f_7 + \lambda_{22} f_8 + \lambda_{32} f_9$$

$$F_9 = c_3 f_7 + \lambda_{23} f_8 + \lambda_{33} f_9$$

$$F_{10} = c_1 f_{10} + \lambda_{21} f_{11} + \lambda_{31} f_{12}$$

$$F_{11} = c_2 f_{10} + \lambda_{22} f_{11} + \lambda_{32} f_{12}$$

$$F_{12} = c_3 f_{10} + \lambda_{23} f_{11} + \lambda_{33} f_{12} \quad (4.27)$$

Chapter V

BEAM-COLUMN MODEL

5.1 INTRODUCTION

This chapter presents the three-dimensional beam-column model in U.L. formulation [2],[4], which is based on the conventional beam-column theory [23]. Because using the actural solution to the differential equation, the beam-column model can trace equilibrium path accurately. In this model the behavior of the element is referred to the convected coordinate system, and then a transformation is applied from a local to a global coordinate system.

The rotations and translations of the nodes are considered to be arbitrarily large, but the relative deformations of the element are assumed to be small such that the conventional beam-column theory can be applied. In the beam-column model, the effect of length shortening due to bending is considered, and the external loads are supposed to be applied at the nodes only.

5.2 ELEMENT END FORCE-DEFORMATION RELATIONS

The relationships between element-end forces and deformations (Fig. 5.1) based on the conventional beam-column theory are [2],[3],[23]:

$$M_{1n} = EI_n / L \, (c_{1n} e_{1n} + c_{2n} e_{2n}) \tag{5.1}$$

$$M_{2n} = EI_n / L \, (c_{2n} e_{1n} + c_{1n} e_{2n}) \tag{5.2}$$

$$M_t = (GJ / L) \, \phi_t \tag{5.3}$$

$$P = EA \, (u / L - c_{b2} - c_{b3}) \tag{5.4}$$

in which

e_{mn} = relative end rotations (Fig. 5.1); the first subscript refers to the node where the angle is measured (1 for left node and 2 for right node); the second subscript indicates the axis about which the rotation takes place

M_{mn} = bending moment corresponding to e_{mn}

M_t = torgue

P = axial force, positive in compression

c_{mn} = stability functions [1],[4] associated with bending moments about the x_n-axis (Appendix B)

L = element length

u = relative axial displacement

c_{bn} is length correction factor resulting from the flexural deformations about the x_n-axis [1],[2] defined as

$$c_{bn} = b_{1n} (e_{1n} + e_{2n})^2 + b_{2n} (e_{1n} - e_{2n})^2 \tag{5.5}$$

b_{1n}, b_{2n} are the bowing functions given by

$$b_{1n} = (c_{1n}+c_{2n})(c_{2n}-2) / 8 \pi^2 p_n$$

$$b_{2n} = c_{2n} / 8(c_{1n}+c_{2n}) \tag{5.6}$$

in which

$$p_n = P / P_{En} \tag{5.7}$$

p_n is the dimensionless axial force parameter and

$$P_{En} = \pi^2 E I_n / L^2 \tag{5.8}$$

Thus

$$p_n = PL^2 / \pi^2 E I_n \tag{5.9}$$

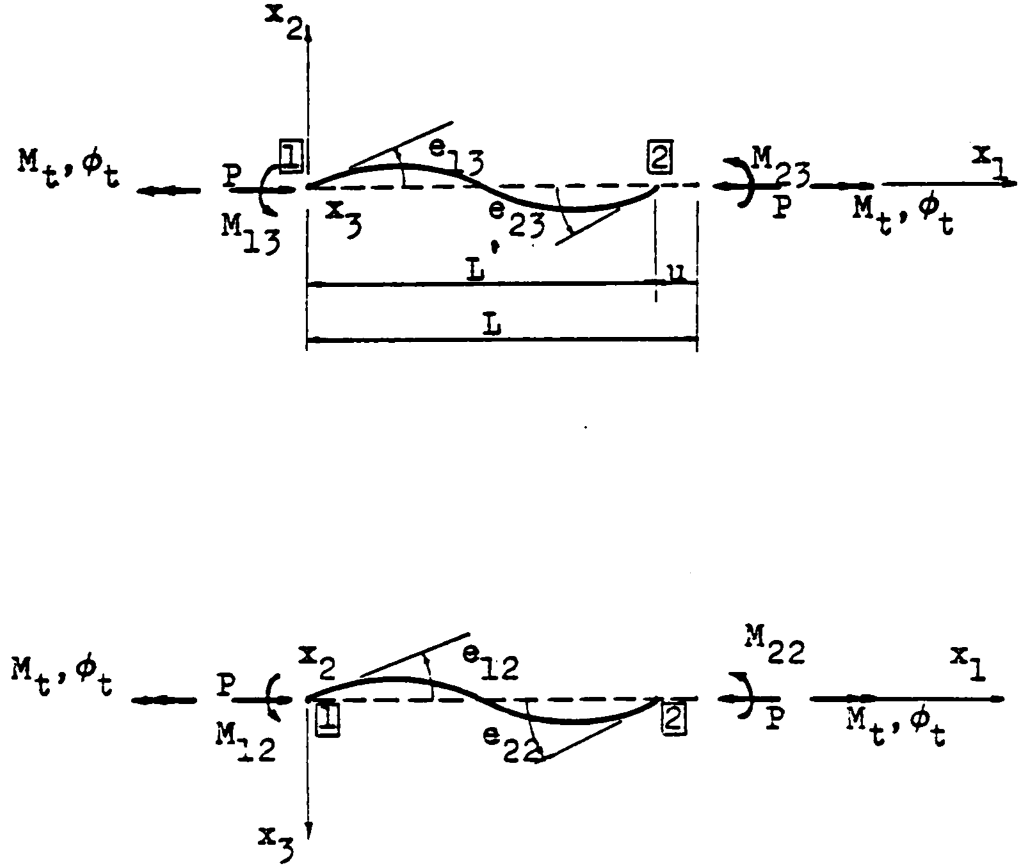


Fig. 5.1 Relative Member End Deformations and Associated Forces in Space Frame

5.3 TANGENT STIFFNESS MATRIX FOR RELATIVE DEFORMATIONS

The relation between incremental vectors, the relative end forces and deformations is

$$\Delta f_{bc} = \hat{k}\ \Delta e \tag{5.10}$$

where

$$\begin{aligned} \Delta f_{bc} &= [\ \Delta M_{13}\ \Delta M_{23}\ \Delta M_{12}\ \Delta M_{22}\ \Delta M_t\ \Delta PL\ ]^T \\ \Delta e &= [\ \Delta e_{13}\ \Delta e_{23}\ \Delta e_{12}\ \Delta e_{22}\ \Delta \phi_t\ \Delta u/L\ ]^T \end{aligned} \tag{5.11}$$

$\hat{k}$ is the tangent stiffness matrix for relative deformations of the beam-column model and has been derived by Oran [2]:

$$\hat{k} = \frac{EI}{L} \begin{bmatrix} \xi_3 c_{13} + \frac{G_{13}^2}{\pi^2 H} & \xi_3 c_{23} + \frac{G_{13}G_{23}}{\pi^2 H} & 0 & 0 & 0 & \frac{G_{13}}{H} \\ \xi_3 c_{23} + \frac{G_{13}G_{23}}{\pi^2 H} & \xi_3 c_{13} + \frac{G_{23}^2}{\pi^2 H} & 0 & 0 & 0 & \frac{G_{23}}{H} \\ 0 & 0 & \xi_2 c_{12} + \frac{G_{12}^2}{\pi^2 H} & \xi_2 c_{22} + \frac{G_{12}G_{22}}{\pi^2 H} & 0 & \frac{G_{12}}{H} \\ 0 & 0 & \xi_2 c_{22} + \frac{G_{12}G_{22}}{\pi^2 H} & \xi_2 c_{12} + \frac{G_{22}^2}{\pi^2 H} & 0 & \frac{G_{22}}{H} \\ 0 & 0 & 0 & 0 & \eta & 0 \\ \frac{G_{13}}{H} & \frac{G_{23}}{H} & \frac{G_{12}}{H} & \frac{G_{22}}{H} & 0 & \frac{\pi^2}{H} \end{bmatrix} \tag{5.12}$$

where

$$G_{1n} = c'_{1n}\, e_{1n} + c'_{2n}\, e_{2n} \tag{5.13}$$

$$G_{2n} = c'_{2n}\, e_{1n} + c'_{1n}\, e_{2n} \tag{5.14}$$

$$H = \pi^2/\zeta^2 + \sum_{n=2,3} 1/\xi_n \,[b'_{1n}(e_{1n}+e_{2n})^2 + b'_{2n}(e_{1n}-e_{2n})^2] \tag{5.15}$$

$$\eta = GJ/EI \tag{5.16}$$

$$\zeta = L \,/\, (I/A)^{1/2} \tag{5.17}$$

$$\xi_n = I_n \,/\, I \tag{5.18}$$

I is the reference moment of inertia, a prime superscript on c_{mn} or b_{mn} indicates one differentiation with respect to p_n.

5.4 LOCAL ELEMENT TANGENT STIFFNESS MATRIX

From Fig. 2.3 and Fig. 5.2 the relationship between local element forces f and relative local element forces f_{bc} is

$$f = C^T f_{bc} \tag{5.19}$$

where

$$f = [\ f_1\ f_2\ f_3\ \ldots.\ f_{12}\ ]^T \tag{5.20}$$

$$f_{bc} = [\ M_{13}\ M_{23}\ M_{12}\ M_{22}\ M_t\ PL\ ]^T \tag{5.21}$$

and C is the local instantaneous transformation matrix defined as

$$C = \begin{bmatrix} 0 & 1/L' & 0 & 0 & 0 & 1 & 0 & -1/L' & 0 & 0 & 0 & 0 \\ 0 & 1/L' & 0 & 0 & 0 & 0 & 0 & -1/L' & 0 & 0 & 0 & 1 \\ 0 & 0 & -1/L' & 0 & 1 & 0 & 0 & 0 & 1/L' & 0 & 0 & 0 \\ 0 & 0 & -1/L' & 0 & 0 & 0 & 0 & 0 & 1/L' & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1/L' & 0 & 0 & 0 & 0 & 0 & -1/L' & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.22}$$

L' is the deformed length of element given by

$$L' = L(1+\delta) \tag{5.23}$$

The length correction factor δ is a function of d, since δ is small in comparison with unity, L' may be approximated by L; i.e., one can set $\delta = 0$. However, $\Delta\delta$ which is a function of Δd must be considered to determine ΔC [2], [29].

The incremental form of Eq. (5.19) is

$$\Delta f = C^T\ \Delta f_{bc} + \Delta C^T\ f_{bc} \tag{5.24}$$

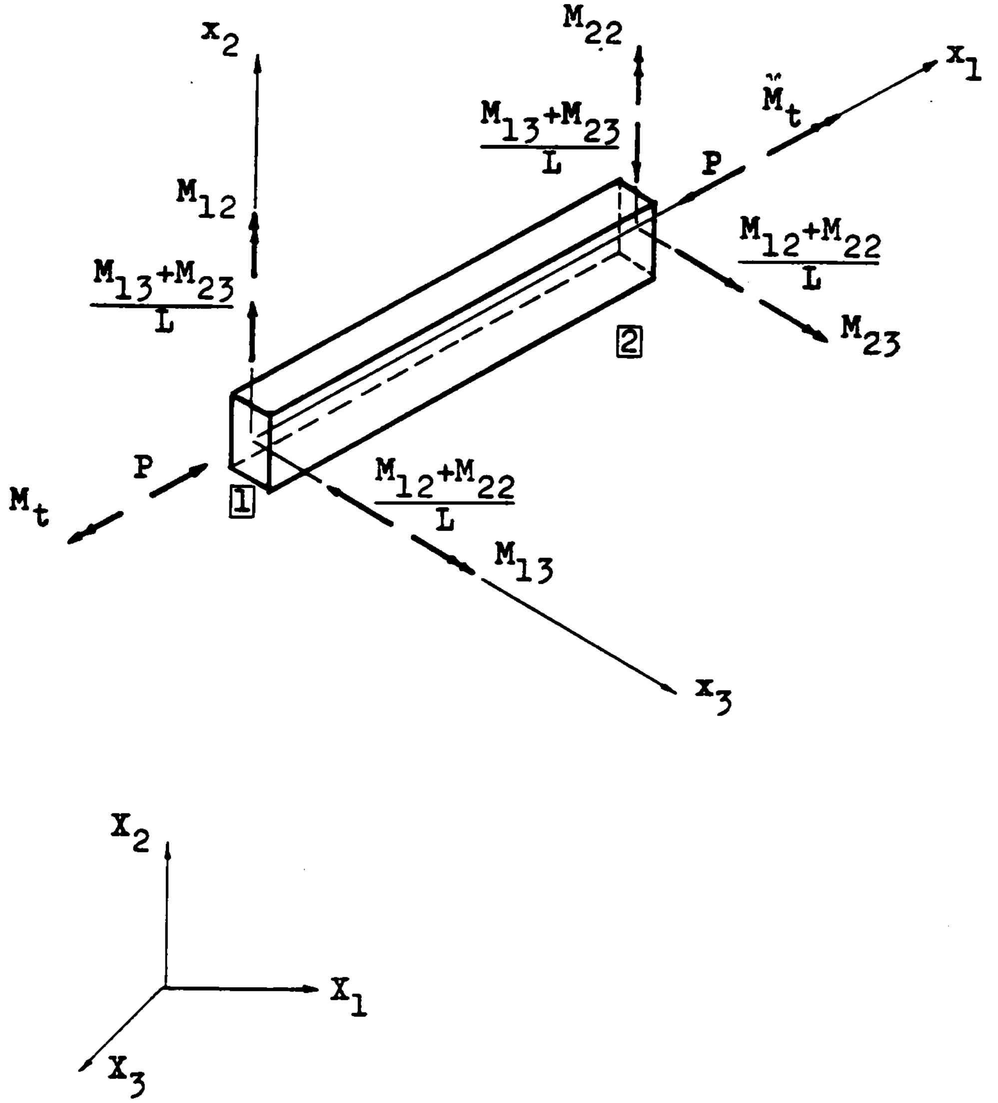


Fig. 5.2 Basic Local Element Forces Associated with Oran's Beam–Column Model in Space Frame

where ΔC is the change in C resulting from Δd (see Appendix C)

By contragredience from Eq. (5.19)

$$\Delta e = C \, \Delta d \tag{5.25}$$

Substituting Eqs. (5.10), (5.25), (C.1) into Eq. (5.24), one obtain

$$\Delta f = k \, \Delta d \tag{5.26}$$

where k is the local tangent stiffness matrix of element defined as

$$k = k_L + k_{NL} \tag{5.27}$$

where

$$k_L = C^T \hat{k} \, C \tag{5.28}$$

$$k_{NL} = \sum_{i=1}^{6} f_{bc_i} \, \bar{g}^{(i)} \tag{5.29}$$

k_L, k_{NL} are the linear and nonlinear tangent stiffness matrix respectively, $\bar{g}^{(i)}$ are the geometric matarices defined in appendix C.

Substituting Eqs. (5.12),(5.22) into Eq. (5.28), one can express the k_L in terms of coefficient functions g_i', i = 1,....20:

$$
k_L = \begin{array}{c}
\begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{array} \\
\left[\begin{array}{cccccc|cccccc}
g_1' & g_2' & g_3' & 0 & g_4' & g_5' & -g_1' & -g_2' & -g_3' & 0 & g_6' & g_7' \\
g_2' & g_8' & 0 & 0 & 0 & g_9' & -g_2' & -g_8' & 0 & 0 & 0 & g_{10}' \\
g_3' & 0 & g_{11}' & 0 & g_{12}' & 0 & -g_3' & 0 & -g_{11}' & 0 & g_{13}' & 0 \\
0 & 0 & 0 & g_{14}' & 0 & 0 & 0 & 0 & 0 & -g_{14}' & 0 & 0 \\
g_4' & 0 & g_{12}' & 0 & g_{15}' & 0 & -g_4' & 0 & -g_{12}' & 0 & g_{16}' & 0 \\
g_5' & g_9' & 0 & 0 & 0 & g_{17}' & -g_5' & -g_9' & 0 & 0 & 0 & g_{18}' \\
\hline
-g_1' & -g_2' & -g_3' & 0 & -g_4' & -g_5' & g_1' & g_2' & g_3' & 0 & -g_6' & -g_7' \\
-g_2' & -g_8' & 0 & 0 & 0 & -g_9' & g_2' & g_8' & 0 & 0 & 0 & -g_{10}' \\
-g_3' & 0 & -g_{11}' & 0 & -g_{12}' & 0 & g_3' & 0 & g_{11}' & 0 & -g_{13}' & 0 \\
0 & 0 & 0 & -g_{14}' & 0 & 0 & 0 & 0 & 0 & g_{14}' & 0 & 0 \\
g_6' & 0 & g_{13}' & 0 & g_{16}' & 0 & -g_6' & 0 & -g_{13}' & 0 & g_{19}' & 0 \\
g_7' & g_{10}' & 0 & 0 & 0 & g_{18}' & -g_7' & -g_{10}' & 0 & 0 & 0 & g_{20}'
\end{array}\right]
\end{array}
\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \end{array}
\tag{5.30}
$$

where

$$g_1' = \alpha \; (\pi^2 / L^2 H)$$

$$g_2' = \alpha \; (G_{13} + G_{23}) / L^2 H$$

$$g_3' = \alpha \; (-G_{12} - G_{22}) / L^2 H$$

$$g_4' = \alpha \; (G_{12} / L H)$$

$$g_5' = \alpha \; (G_{13} / L H)$$

$$g_6' = \alpha \; (G_{22} / L H)$$

$$g_7' = \alpha \; (\; G_{23} \; / \; L \; H \;)$$

$$g_8' = \alpha \; [\; 2(\xi_3 c_{13} + \xi_3 c_{23}) \; / \; L^2 \; + \; (G_{13} + G_{23})^2 \; / \; L^2 \; \pi^2 \; H \;]$$

$$g_9' = \alpha \; [\; \xi_3 (c_{13} + c_{23}) \; / \; L \; + \; (G_{13}^2 + G_{13} G_{23}) \; / \; L \; \pi^2 \; H \;]$$

$$g_{10}' = \alpha \; [\; \xi_3 (c_{13} + c_{23}) \; / \; L \; + \; (G_{13} G_{23} + G_{23}^2) \; / \; L \; \pi^2 \; H \;]$$

$$g_{11}' = \alpha \; [2(\xi_2 c_{12} + \xi_2 c_{22}) / L^2 \; + \; (G_{12} + G_{22})^2 \; / \; L^2 \; \pi^2 \; H \;]$$

$$g_{12}' = \alpha \; [\; -\xi_2 (c_{12} + c_{22}) \; / \; L \; - \; (G_{12}^2 + G_{12} G_{22}) \; / \; L \pi^2 \; H \;]$$

$$g_{13}' = \alpha \; [\; -\xi_2 (c_{12} + c_{22}) \; / \; L \; - \; (G_{12} G_{22} + G_{22}^2) \; / \; L \; \pi^2 \; H \;]$$

$$g_{14}' = \alpha \; \eta$$

$$g_{15}' = \alpha \; (\; \xi_2 c_{12} \; + \; G_{12}^2 \; / \; \pi^2 \; H \;)$$

$$g_{16}' = \alpha \; (\; \xi_2 c_{22} \; + \; G_{12} G_{22} \; / \; \pi^2 \; H \;)$$

$$g_{17}' = \alpha \; (\; \xi_3 c_{13} \; + \; G_{13}^2 \; / \; \pi^2 \; H \;)$$

$$g_{18}' = \alpha \; (\; \xi_3 c_{23} \; + \; G_{13} G_{23} \; / \; \pi^2 \; H \;)$$

$$g_{19}' = \alpha \; (\; \xi_2 c_{12} \; + \; G_{22}^2 \; / \; \pi^2 \; H \;)$$

$$g_{20}' = \alpha \; (\; \xi_3 c_{13} \; + \; G_{23}^2 \; / \; \pi^2 \; H \;)$$

and

$$\alpha = EI/L \tag{5.31}$$

5.5 GLOBAL ELEMENT TANGENT STIFFNESS MATRIX

From Eq. (2.59)

$$\Delta F = \Lambda^T \Delta f \tag{5.32}$$

Substituting Eqs. (5.26),(2.58) into Eq. (5.32) yields

$$\Delta F = K^i \Delta D \tag{5.33}$$

where

$$K^i = \Lambda^T k \Lambda \tag{5.34}$$

is the global strain incremental stiffness matrix of element i. Substituting Eqs. (5.27),(5.28),(5.29) into Eq. (5.34) yields

$$K^i = K_L^i + K_{NL}^i \tag{5.35}$$

where

$$K_L^i = \Lambda^T k_L \Lambda \tag{5.36}$$

$$K_{NL}^i = \Lambda^T k_{NL} \Lambda \tag{5.37}$$

Sustituting Eq. (5.28) into Eq. (5.37) yields

$$K_{NL}^i = \sum_{i=1}^{6} f_{bc_i} g^{(i)} \tag{5.38}$$

$$= (M_{13}+M_{23})g^{(1)} + (M_{12}+M_{22})g^{(3)} + PL\ g^{(6)} \tag{5.39}$$

where

$$g^{(i)} = \Lambda^T \, \bar{g}^{(i)} \, \Lambda \tag{5.40}$$

$\bar{g}^{(i)}$ and $g^{(i)}$ are the geometric matrices defined in the appendices C and D respectively. K_L^i, K_{NL}^i are the linear and nonlinear tangent stiffness matrices in the global coordinates respectively, f_{bc_i} is the i^{th} component of f_{bc}.

Sustituting Eqs. (2.54),(5.30) into Eq. (5.36), one can express K_L^i, similar to Eq. (4.10), in terms of coefficient functions g_i, i = 1, 2,.... 42. The index matrix of g_i is [28]

$$\text{INDEX} = \left[\begin{array}{cccccc|cccccc}
1 & 2 & 3 & 4 & 5 & 6 & -1 & -2 & -3 & 7 & 8 & 9 \\
2 & 10 & 11 & 12 & 13 & 14 & -2 & -10 & -11 & 15 & 16 & 17 \\
3 & 11 & 18 & 19 & 20 & 21 & -3 & -11 & -18 & 22 & 23 & 24 \\
4 & 12 & 19 & 25 & 26 & 27 & -4 & -12 & -19 & 28 & 29 & 30 \\
5 & 13 & 20 & 26 & 31 & 32 & -5 & -13 & -20 & 29 & 33 & 34 \\
6 & 14 & 21 & 27 & 32 & 35 & -6 & -14 & -21 & 30 & 34 & 36 \\
\hline
-1 & -2 & -3 & -4 & -5 & -6 & 1 & 2 & 3 & -7 & -8 & -9 \\
-2 & -10 & -11 & -12 & -13 & -14 & 2 & 10 & 11 & -15 & -16 & -17 \\
-3 & -11 & -18 & -19 & -20 & -21 & 3 & 11 & 18 & -22 & -23 & -24 \\
7 & 15 & 22 & 28 & 29 & 30 & -7 & -15 & -22 & 37 & 38 & 39 \\
8 & 16 & 23 & 29 & 33 & 34 & -8 & -16 & -23 & 38 & 40 & 41 \\
9 & 17 & 24 & 30 & 34 & 36 & -9 & -17 & -24 & 39 & 41 & 42
\end{array}\right] \tag{5.41}$$

which is consistent with Eq. (4.10) and the coefficient functions are

$$g_1 = \alpha(c_1^2\ g_1' + 2c_1\ \lambda_{21}\ g_2' + 2c_1\ \lambda_{31}\ g_3' + \lambda_{21}^2\ g_8' + \lambda_{31}^2\ g_{11}')$$

$$g_2 = \alpha[c_1\ c_2\ g_1' + (c_1\ \lambda_{22} + c_2\ \lambda_{21})g_2' + (c_1\ \lambda_{32} + c_2\ \lambda_{31})g_3'$$
$$+ \lambda_{21}\ \lambda_{22}\ g_8' + \lambda_{31}\lambda_{32}g_{11}']$$

$$g_3 = \alpha[c_1\ c_3\ g_1' + (c_1\ \lambda_{23} + c_3\ \lambda_{21})g_2' + (c_1\ \lambda_{33} + c_3\ \lambda_{31})g_3'$$
$$+ \lambda_{21}\ \lambda_{23}\ g_8' + \lambda_{31}\lambda_{33}g_{11}'] \tag{5.42}$$

$$g_4 = \alpha(c_1\ \lambda_{21}\ g_4' + c_1\ \lambda_{31}\ g_5' + \lambda_{21}\ \lambda_{31}\ g_9' + \lambda_{31}\ \lambda_{21}\ g_{12}')$$

$$g_5 = \alpha(c_1\ \lambda_{22}\ g_4' + c_1\ \lambda_{32}\ g_5' + \lambda_{21}\ \lambda_{32}\ g_9' + \lambda_{31}\ \lambda_{22}\ g_{12}')$$

$$g_6 = \alpha(c_1\ \lambda_{23}\ g_4' + c_1\ \lambda_{33}\ g_5' + \lambda_{21}\ \lambda_{33}\ g_9' + \lambda_{31}\ \lambda_{23}\ g_{12}')$$

$$g_7 = \alpha(c_1\ \lambda_{21}\ g_6' + c_1\ \lambda_{31}\ g_7' + \lambda_{21}\ \lambda_{31}\ g_{10}' + \lambda_{31}\ \lambda_{21}\ g_{13}')$$

$$g_8 = \alpha(c_1\ \lambda_{22}\ g_6' + c_1\ \lambda_{32}\ g_7' + \lambda_{21}\ \lambda_{32}\ g_{10}' + \lambda_{31}\ \lambda_{22}\ g_{13}')$$

$$g_9 = \alpha(c_1\ \lambda_{23}\ g_6' + c_1\ \lambda_{33}\ g_7' + \lambda_{21}\ \lambda_{33}\ g_{10}' + \lambda_{31}\ \lambda_{23}\ g_{13}')$$

$$g_{10} = \alpha(c_2^2\ g_1' + 2c_2\ \lambda_{22}\ g_2' + 2c_2\ \lambda_{32}\ g_3' + \lambda_{22}^2\ g_8' + \lambda_{32}^2\ g_{11}')$$

$$g_{11} = \alpha[c_2\ c_3\ g_1' + (c_2\ \lambda_{23} + c_3\ \lambda_{22})g_2' + (c_2\ \lambda_{33} + c_3\ \lambda_{32})g_3'$$
$$+ \lambda_{22}\ \lambda_{23}\ g_8' + \lambda_{32}\lambda_{33}g_{11}']$$

$$g_{12} = \alpha(c_2\ \lambda_{21}\ g_4' + c_2\ \lambda_{31}\ g_5' + \lambda_{22}\ \lambda_{31}\ g_9' + \lambda_{32}\ \lambda_{21}\ g_{12}')$$

$$g_{13} = \alpha(c_2 \lambda_{22} g_4' + c_2 \lambda_{32} g_5' + \lambda_{22} \lambda_{32} g_9' + \lambda_{32} \lambda_{22} g_{12}')$$

$$g_{14} = \alpha(c_2 \lambda_{23} g_4' + c_2 \lambda_{33} g_5' + \lambda_{22} \lambda_{33} g_9' + \lambda_{32} \lambda_{23} g_{12}')$$

$$g_{15} = \alpha(c_2 \lambda_{21} g_6' + c_2 \lambda_{31} g_7' + \lambda_{22} \lambda_{31} g_{10}' + \lambda_{32} \lambda_{21} g_{13}')$$

$$g_{16} = \alpha(c_2 \lambda_{22} g_6' + c_2 \lambda_{32} g_7' + \lambda_{22} \lambda_{32} g_{10}' + \lambda_{32} \lambda_{22} g_{13}')$$

$$g_{17} = \alpha(c_2 \lambda_{23} g_6' + c_2 \lambda_{33} g_7' + \lambda_{22} \lambda_{33} g_{10}' + \lambda_{32} \lambda_{23} g_{13}')$$

$$g_{18} = \alpha(c_3^2 g_1' + 2c_3 \lambda_{23} g_2' + 2c_3 \lambda_{33} g_3' + \lambda_{23}^2 g_8' + \lambda_{33}^2 g_{11}')$$

$$g_{19} = \alpha(c_3 \lambda_{21} g_4' + c_3 \lambda_{31} g_5' + \lambda_{23} \lambda_{31} g_9' + \lambda_{33} \lambda_{21} g_{12}')$$

$$g_{20} = \alpha(c_3 \lambda_{22} g_4' + c_3 \lambda_{32} g_5' + \lambda_{23} \lambda_{32} g_9' + \lambda_{33} \lambda_{22} g_{12}')$$

$$g_{21} = \alpha(c_3 \lambda_{23} g_4' + c_3 \lambda_{33} g_5' + \lambda_{23} \lambda_{33} g_9' + \lambda_{33} \lambda_{23} g_{12}')$$

$$g_{22} = \alpha(c_3 \lambda_{21} g_6' + c_3 \lambda_{31} g_7' + \lambda_{23} \lambda_{31} g_{10}' + \lambda_{33} \lambda_{21} g_{13}')$$

$$g_{23} = \alpha(c_3 \lambda_{22} g_6' + c_3 \lambda_{32} g_7' + \lambda_{23} \lambda_{32} g_{10}' + \lambda_{33} \lambda_{22} g_{13}')$$

$$g_{24} = \alpha(c_3 \lambda_{23} g_6' + c_3 \lambda_{33} g_7' + \lambda_{23} \lambda_{33} g_{10}' + \lambda_{33} \lambda_{23} g_{13}')$$

$$g_{25} = \alpha(c_1^2 g_{14}' + \lambda_{21}^2 g_{15}' + \lambda_{31}^2 g_{17}') \tag{5.42}$$

$$g_{26} = \alpha(c_1 c_2 g_{14}' + \lambda_{21} \lambda_{22} g_{15}' + \lambda_{31} \lambda_{32} g_{17}')$$

$$g_{27} = \alpha(c_1 c_3 g_{14}' + \lambda_{21} \lambda_{23} g_{15}' + \lambda_{31} \lambda_{33} g_{17}')$$

$$g_{28} = \alpha(-c_1^2 g_{14}' + \lambda_{21}^2 g_{16}' + \lambda_{31}^2 g_{18}')$$

$$g_{29} = \alpha(-c_1 c_2 g_{14}' + \lambda_{21} \lambda_{22} g_{16}' + \lambda_{31} \lambda_{32} g_{18}')$$

$$g_{30} = \alpha(-c_1 c_3 g_{14}' + \lambda_{21} \lambda_{23} g_{16}' + \lambda_{31} \lambda_{33} g_{18}')$$

$$g_{31} = \alpha(c_2^2 g_{14}' + \lambda_{22}^2 g_{15}' + \lambda_{32}^2 g_{17}')$$

$$g_{32} = \alpha(c_2\ c_3\ g'_{14} + \lambda_{22}\ \lambda_{23}\ g'_{15} + \lambda_{32}\ \lambda_{33}\ g'_{17})$$

$$g_{33} = \alpha(-c_2^2\ g'_{14} + \lambda_{22}^2\ g'_{16} + \lambda_{32}^2\ g'_{18})$$

$$g_{34} = \alpha(-c_2\ c_3\ g'_{14} + \lambda_{22}\ \lambda_{23}\ g'_{16} + \lambda_{32}\ \lambda_{33}\ g'_{18})$$

$$g_{35} = \alpha(c_3^2\ g'_{14} + \lambda_{23}^2\ g'_{15} + \lambda_{33}^2\ g'_{17})$$

$$g_{36} = \alpha(-c_3^2\ g'_{14} + \lambda_{23}^2\ g'_{16} + \lambda_{33}^2\ g'_{18})$$

$$g_{37} = \alpha(c_1^2\ g'_{14} + \lambda_{21}^2\ g'_{19} + \lambda_{31}^2\ g'_{20}) \tag{5.42}$$

$$g_{38} = \alpha(c_1\ c_2\ g'_{14} + \lambda_{21}\ \lambda_{22}\ g'_{19} + \lambda_{31}\ \lambda_{32}\ g'_{20})$$

$$g_{39} = \alpha(c_1\ c_3\ g'_{14} + \lambda_{21}\ \lambda_{23}\ g'_{19} + \lambda_{31}\ \lambda_{33}\ g'_{20})$$

$$g_{40} = \alpha(c_2^2\ g'_{14} + \lambda_{22}^2\ g'_{19} + \lambda_{32}^2\ g'_{20})$$

$$g_{41} = \alpha(c_2\ c_3\ g'_{14} + \lambda_{22}\ \lambda_{23}\ g'_{19} + \lambda_{32}\ \lambda_{33}\ g'_{20})$$

$$g_{42} = \alpha(c_3^2\ g'_{14} + \lambda_{23}^2\ g'_{19} + \lambda_{33}^2\ g'_{20})$$

where

$\alpha = EI/L$

Chapter VI

SOLUTION ALGORITHMS

6.1 INTRODUCTION

The response of a nonlinear structure may be solved approximately for incremental nodal displacements by taking a series of linear steps. Many different solution schemes have been proposed to solve the nonlinear equilibrium Eqs. (2.55) and (2.104). Here the numerical solutions are obtained by applying either the Newton-Raphson method or the modified Riks/Wempner method to the nonlinear equilibrium equation. The former, efficient for low convergence tolerance [16], is popularly used to trace nonlinear prebuckling paths of structures; however, it cannot trace the response beyond the limit point. The latter has been especially proposed to overcome this problem and can trace nonlinear post critical response [27]. Both solution techniques are described briefly in this chapter.

6.2 NEWTON-RAPHSON METHOD

The Newton-Raphson method for solving the nonlinear equilibrium equations (2.55) and (2.104) may be stated as [43],[45]

$$K^{k-1}\,\Delta q^{k} = R^{k-1} \qquad k=1,2,\ldots.n \tag{6.1}$$

in which

$$K^{k-1} = \sum_{i=1}^{NE} K^{(i)^{k-1}} \tag{6.2}$$

$$R^{k-1} = {}^{t+\Delta t}Q - F^{k-1} \tag{6.3}$$

$$q^{k} = q^{k-1} + \Delta q^{k} \tag{6.4}$$

q^k is the k^{th} trial solution corresponding to a given load level ${}^{t+\Delta t}Q$, K^{k-1} is the structural tangent stiffness matrix corresponding to q^{k-1}, $K^{(i)k-1}$ is the generalized tangent stiffness matrix of element i corresponding to q^{k-1}, F^{k-1} is the equilibrating nodal force vector corresponding to q^{k-1}, R^{k-1} is the unbalanced force vector corresponding to q^{k-1}.

The procedure, illustrated for a one-degree-of-freedom system in Fig. 6.1, is as follows. For a given new load level ${}^{t+\Delta t}Q$

1. Establish system tangent stiffness matrix K^{k-1}.
2. Evaluate the equilibrating nodal force vector F^{k-1}.
3. Compute the vector of unbalanced nodal force R^{k-1} using Eq. (6.3).
4. Solve Eq. (6.1) for the vector of incremental nodal displacements Δq^{k}.
5. Update the vector of nodal displacements by Eq. (6.4).
6. Test for convergence.
7. If the process has not converged return to step 1. Otherwise increment the load vector and seek a new solution.

To reduce the amount of computations per iteration, the Newton-Raphson method is often modified by using the same tangent stiffness matrix during several iterative cycles (Fig. 6.2). It is updated only at every load level or only when the convergence rate becomes poor. This method requires more steps to reach a new equilibrium point. The choice between both methods depends on the closeness of the initial vector to the true solution.

Using Newton-Raphson method in the neighborhood of a limit point, the tangent stiffness matrix approaches singulari-

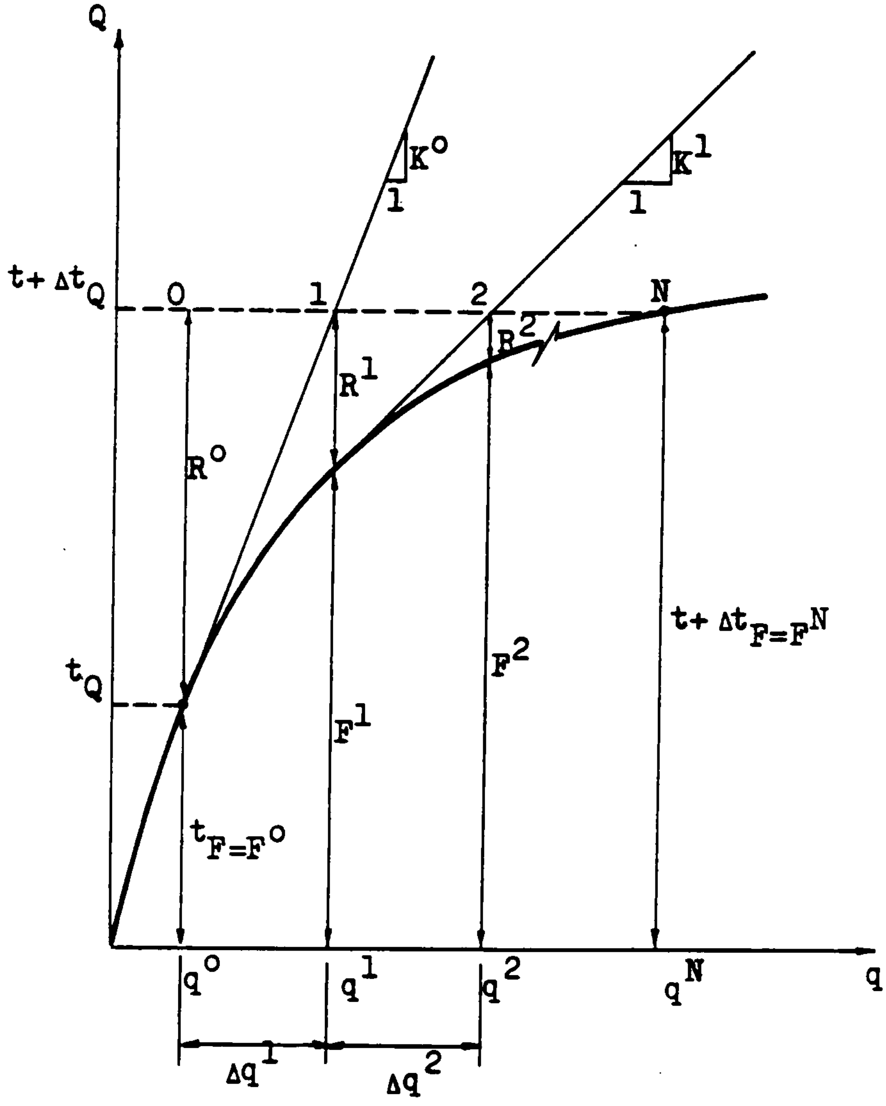


Fig. 6.1 Newton-Raphson Iteration

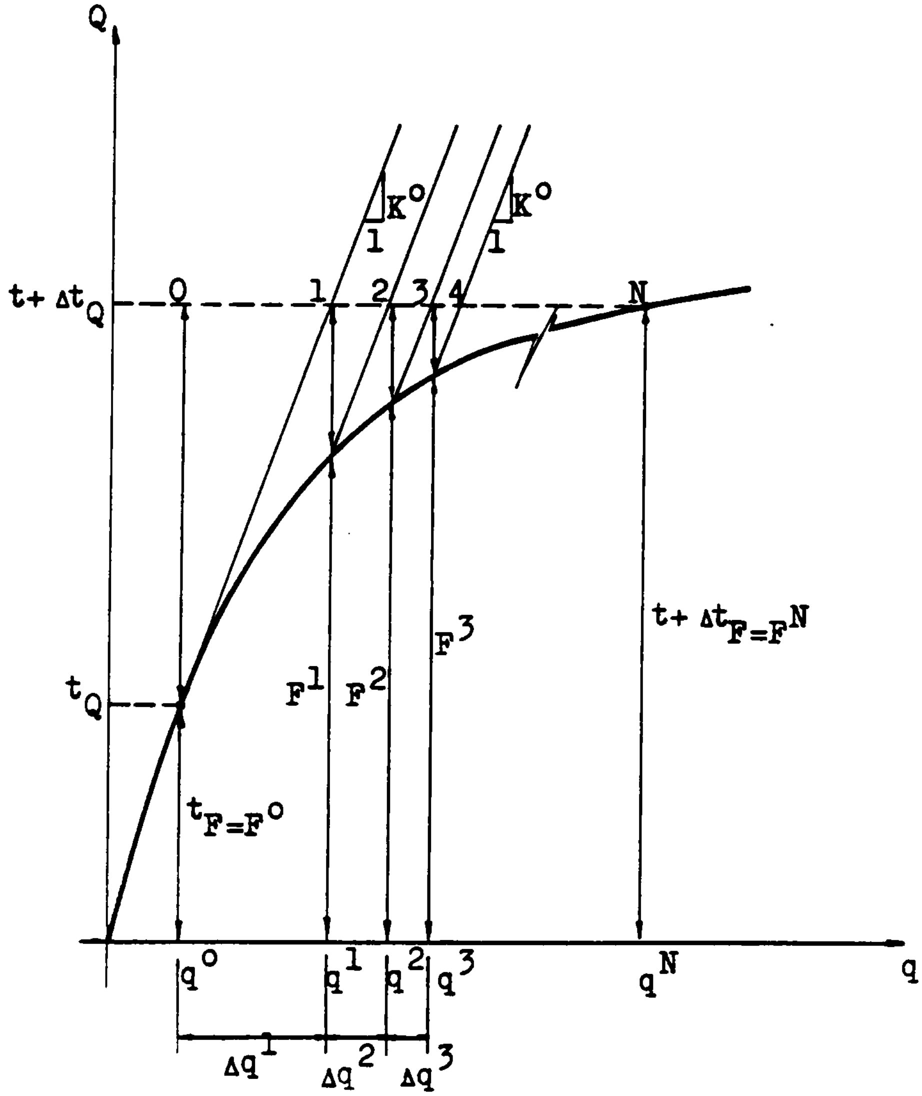


Fig. 6.2 Modified Newton-Raphson Iteration

ty resulting in an increasing number of iterations and smaller and smaller load step; finally the solution diverges. Therefore, it cannot trace the response beyond the limit point. To solve this problem, the modified Riks/Wempner method is recommended [27].

6.3 MODIFIED RIKS/WEMPNER METHOD

The theoretical development was recently summarized in reference [27]. Therefore, details are omitted here. The basic idea of the Riks/Wempner method is to choose a generalized arc length to facilitate the procedure of seeking a new solution [45],[46]. For a given basic load incremental $\Delta\lambda^1$, the generalized arc legngh ΔS can be computed from the constraint equation. Then the iteration path follows a normal plane to the tangent, and the new equilibrium point will be the intersection of the normal plane with the equilibrium path. Iteration along a circle (or sphere in space) will not be considered here [27],[47]. In this section the notation of reference [27] is adopted.

The algorithm of the modified Riks/Wempner method is briefly summarized as follows (Figs. 6.3 and 6.4):

For the First Step

1. Choose a basic load increment $\Delta\lambda^1$
2. Establish the system tangent stiffness matrix K^0 in the current configuration q^0
3. Solve equation

$$K^0 \ \Delta q^{1I} = \bar{Q} \quad \text{for } \Delta q^{1I} \tag{6.5}$$

where $\bar{Q}$ is the vector of constant load distribution

4. Compute the generalized arc length Δs from the constraint equation

$$\Delta S = \Delta\lambda^1 \ (\Delta q^{1I} \ . \ \Delta q^{1I} + 1)^{1/2} \tag{6.6}$$

or scale ΔS to control the number of future iterations

$$\Delta\hat{S} = \Delta S \ (\hat{I}/I)^{1/2} \tag{6.7}$$

where $\hat{I}$ is the desired number of iterations and I is the required number of iterations in the previous step.

5. Compute the incremental nodal displacements

$$\Delta q^1 = \Delta\lambda^1 \ \Delta q^{1I} \tag{6.8}$$

6. Update the nodal displacements and the load parameter

$$q^1 = q^0 + \Delta q^1 \tag{6.9}$$

$$\lambda^1 = \lambda^0 + \Delta\lambda^1 \tag{6.10}$$

<u>During the k^{th} iteration on normal plane</u>, $k=2,3,\ldots$

7. Compute the nodal force vector F^{k-1} corresponding to q^{k-1}

8. Update the load vector

$$Q^{k-1} = \lambda^{k-1} \bar{Q} \tag{6.11}$$

9. Compute the unbalanced force vector

$$R^{k-1} = Q^{k-1} - F^{k-1} \tag{6.12}$$

10. Update the system tangent stiffness matrix K^{k-1} if desirable

11. Solve for Δq^{kI} and Δq^{kII} from the two sets of equilibrium equations

$$K^{k-1} \Delta q^{kI} = \bar{Q} \tag{6.13}$$

$$K^{k-1} \Delta q^{kII} = R^{k-1} \tag{6.14}$$

12. Compute the incremental load parameter

$$\Delta\lambda^{k} = -(\Delta q^{1} . \Delta q^{kII})/(\Delta q^{1} . \Delta q^{kI} + \Delta\lambda^{1}) \tag{6.15}$$

13. Compute the incremental nodal displacement vector

$$\Delta q^{k} = \Delta\lambda^{k} \Delta q^{kI} + \Delta q^{kII} \tag{6.16}$$

14. Update the nodal displacement vector and the load parameter

$$q^k = q^{k-1} + \Delta q^k \tag{6.17}$$

$$\lambda^k = \lambda^{k-1} + \Delta\lambda^k \tag{6.18}$$

15. Repeat steps 7 to 14 until process has converged
16. Start a new step by returning to step 2.

In conclusion, the modified Riks/Wempner method has been successfully applied to a variety problems [27],[43].

6.4 CONVERGENCE CRITERIA

The incremental solution at the end of each iteration should be checked to see whether it has converged within preset tolerances. Two displacement criteria and one force criterion are discussed in this section.

Cook [33] adopted the displacement criterion based on the infinity vector norm:

$$\| \Delta q^k \|_\infty \, / \, \| q^k \|_\infty \leqslant \text{CPDC} \tag{6.19}$$

where Δq^k is the maximum incremental displacement, q^k is the maximum total displacement of same type, and CPDC is the

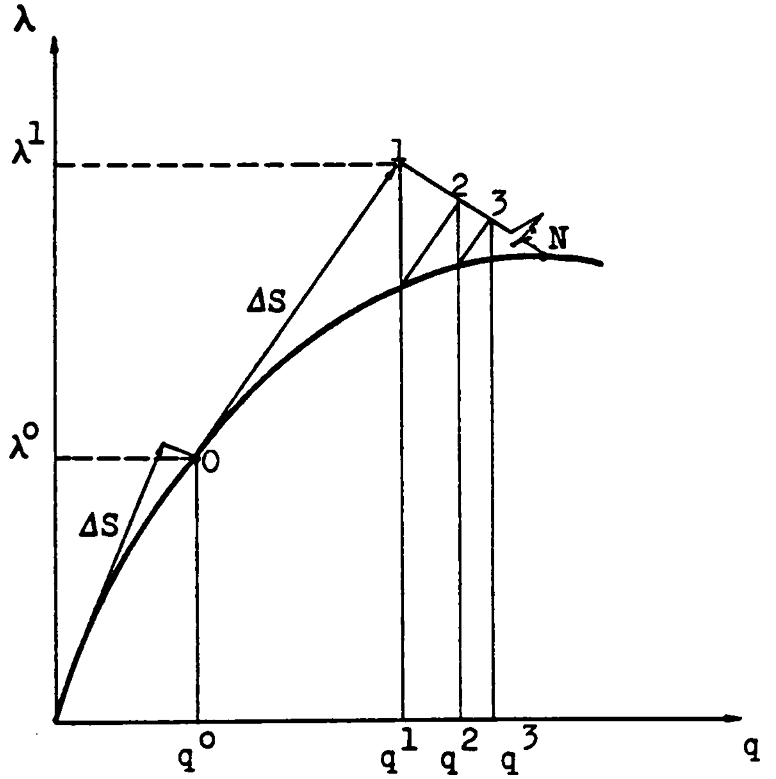


Fig. 6.3 Modified Riks/Wempner Iteration Along the Normal Plane

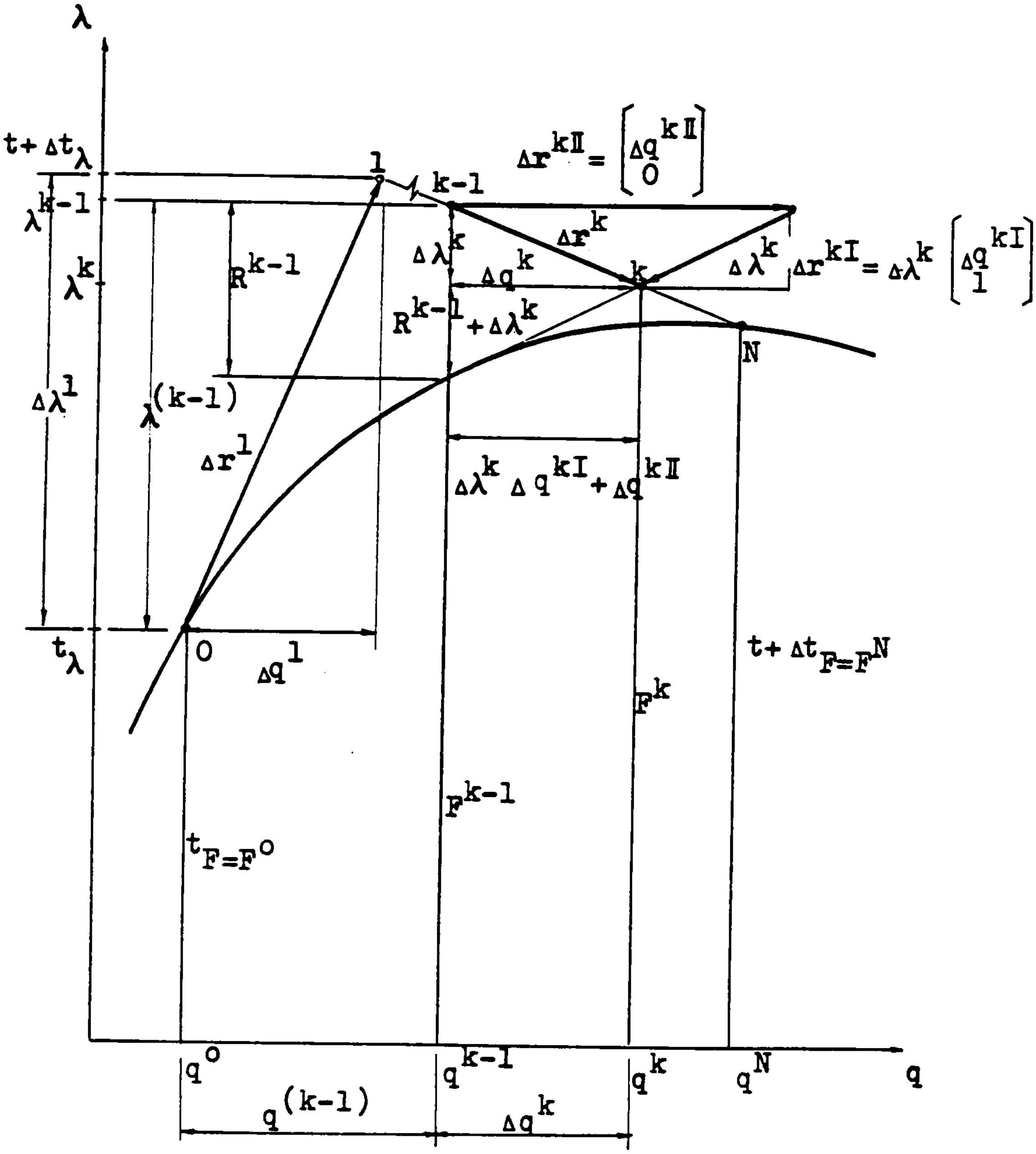


Fig. 6.4 Iteration Procedure along Normal Plane with Updating K

displacement convergence parameter adopted by Cook with the following range:

$$10^{-6} \leq CPDC \leq 10^{-2} \qquad (6.20)$$

Bathe [38],[49] recommends the displacement criterion based on the Euclidean vector norm:

$$\| \Delta q^{k} \|_2 \ / \ \| q^{k} \|_2 \leq CPDB \qquad (6.21)$$

where CPDB is the displacement convergence parameter adopted by Bathe with the value

$$CPDB = 0.001 \qquad (6.22)$$

A third convergence criterion could be defined as the ratio of the norm of the residual load to the norm of the original load increment [38],[49]:

$$\| {}^{t+\Delta t}Q - {}^{t+\Delta t}_{t+\Delta t}F^{k} \|_2 \ / \ \| {}^{t+\Delta t}Q - F^{0} \|_2 \leq CPF \qquad (6.23)$$

where CPF is the convergence parameter for the unbalanced force criterion with a convergence tolerance of 0.1.

Chapter VII

SAMPLE ANALYSIS

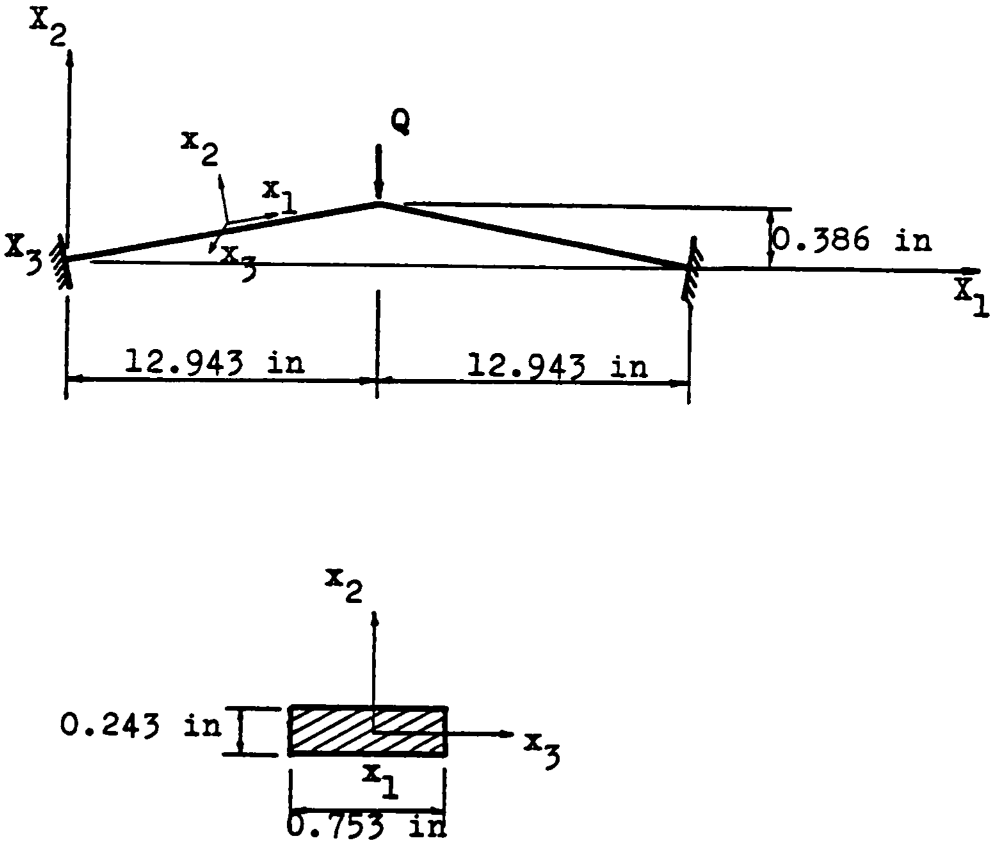
7.1 INTRODUCTION

The U.L formulation described in Chapter 2 has been employed in the computer program. Four examples were investigated in this study and comparisons between the finite element model and the beam-column model were made. All of the test structures were treated as space frames and analyzed by the modified Riks/Wempner method.

7.2 EXAMPLE 1: WILLIAMS' TOGGLE FRAME

Williams [24], Wood and Zienkiewicz [20], and Papadrakakis [21] investigated this problem. Fig. 7.1 shows the configuration and properties.

The finite element solution is presented in Fig. 7.2 to compare the effects of mesh refinement. One, four, six, and twelve elements per member are used . The responses of the different meshes are in close agreement up to near the limit point. Beyond this region,the single element is too stiff, but four elements are adequate for modeling the behavior of the toggle frame.



$A = 0.183 \text{ in}^2$

$E = 10{,}300. \text{ ksi}$

$I_3 = 0.00090039 \text{ in}^4$

Fig. 7.1 William's Toggle Frame

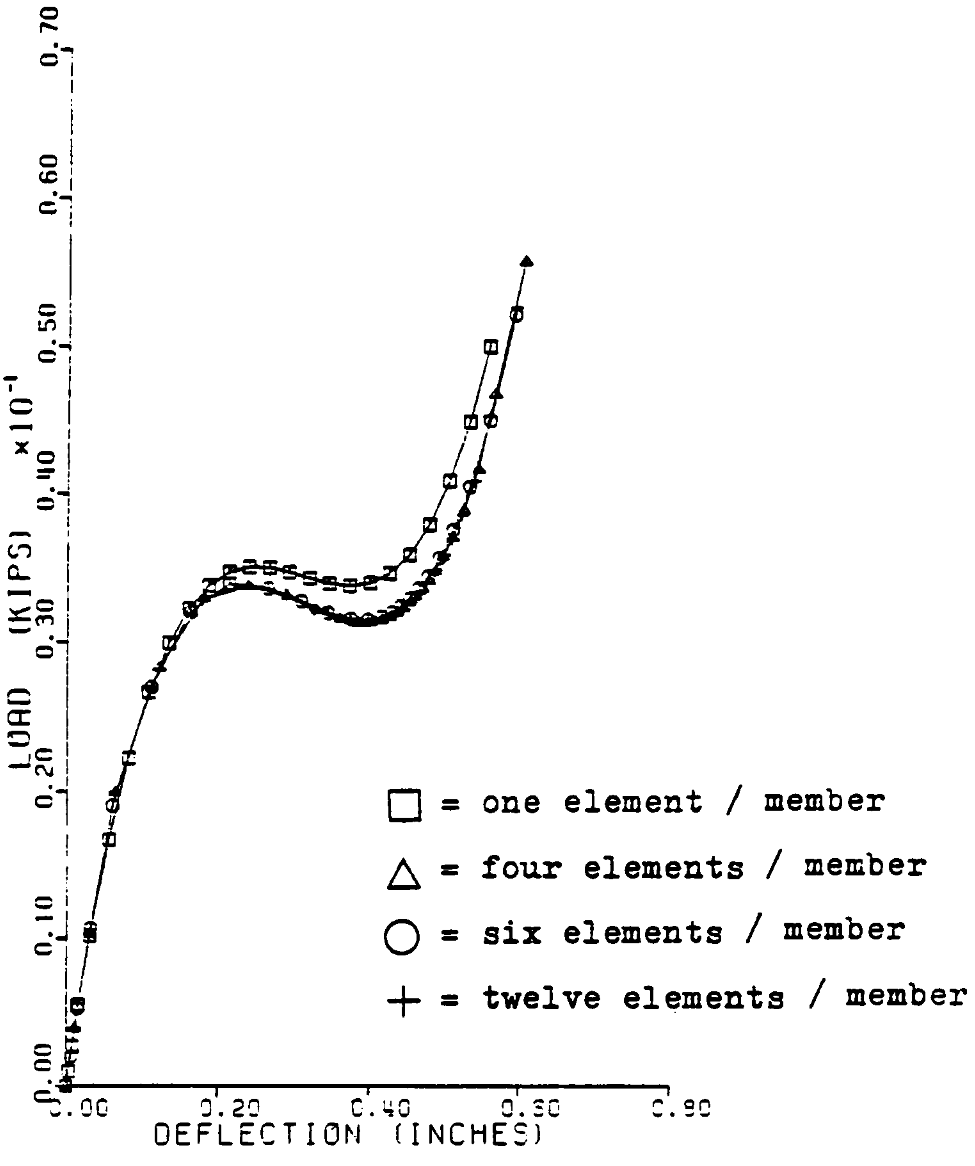


Fig. 7.2 Finite Element Model

In Fig. 7.3 a comparison of twelve finite elements/member to one beam-column element/member is presented. It shows that the accuracy of one beam-column element is comparable to that of twelve finite elements. The execution times for both elements are as follows:

one finite element	:	0.43 sec
four finite elements	:	1.12 sec
six finite elements	:	1.37 sec
twelve finite elements	:	2.40 sec
one beam-column element	:	0.49 sec

7.3 EXAMPLE 2 : THREE DIMENSIONAL CANTILEVER BEAM OF A 45-DEGREE BEND

A tip loaded cantilever beam of a 45-degree bend, undergoing large displacements (Fig. 7.4), was investigated by Bathe [9]. The beam lies in the X_1-X_3 plane. The average radius of the bend is 100 inch. The concentrated end load is applied in the X_2-direction.

Using both models, the tip deflections in the X_2-direction are shown in Figs. 7.5 and 7.6 respectively. Fig. 7.5 presents the finite element solution in which the

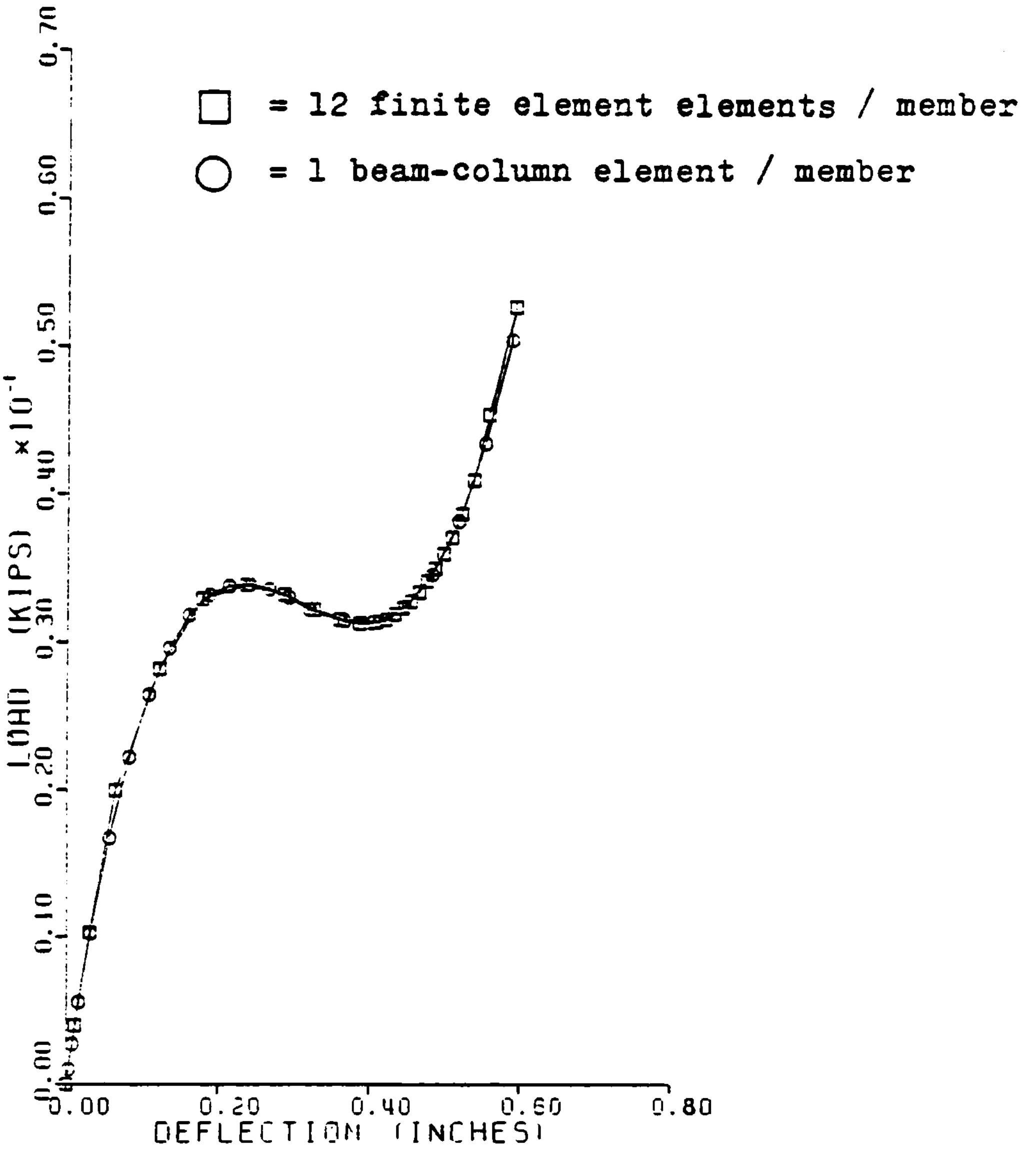
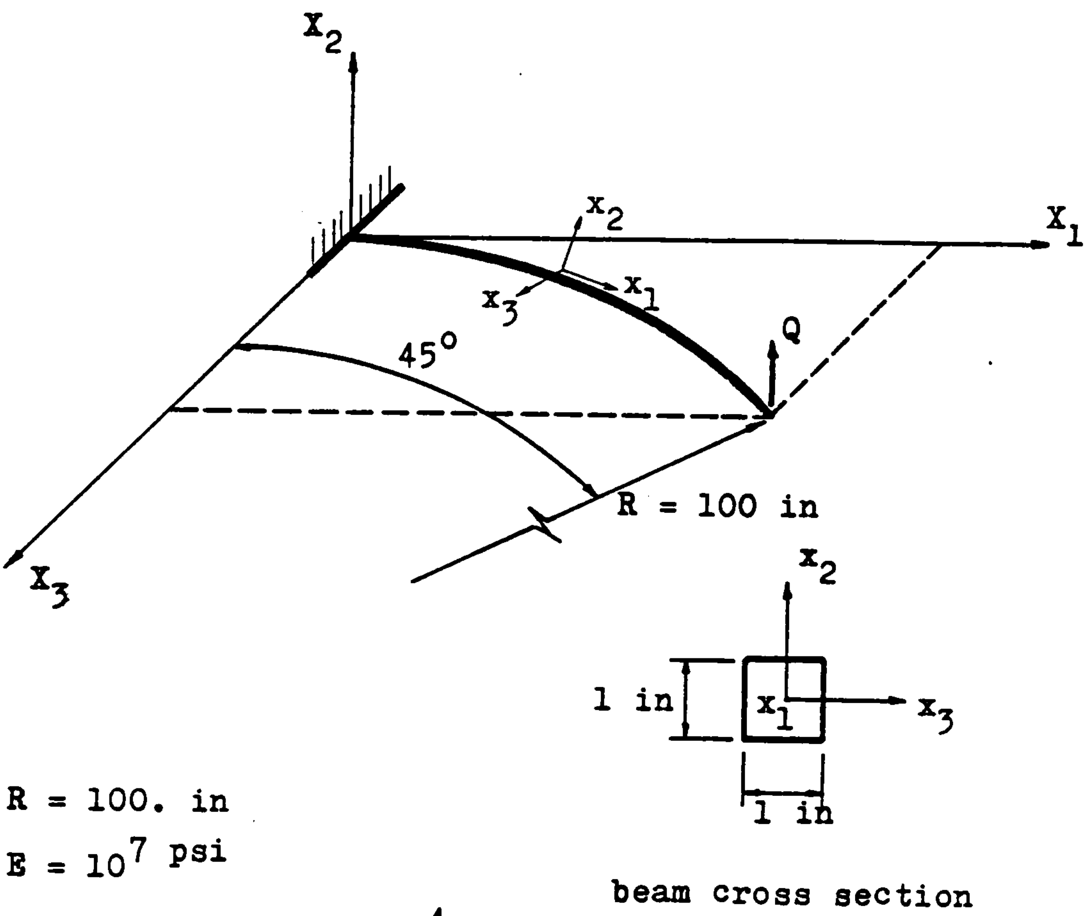


Fig. 7.3 Comparison of Models



$R = 100.$ in

$E = 10^7$ psi

$I_2 = I_3 = 0.0833333\ \text{in}^4$

$\nu = 0.$

$G = \frac{E}{2(1+\nu)}$

$J = 0.141a^4 = 0.141\ \text{in}^4$

Fig. 7.4 Three Dimensional Cantilevel Beam of 45^o Bend effects of different mesh refinements are compared; it shows that two and four elements fail (solution procedure breaks down), but eight elements are adequate to model the behavior of the curved beam. Fig. 7.6 presents the beam-column solution obtained using four and eight elements; it gives similar results for both meshes. Fig. 7.7 presents a comparison of four beam-column elements to eight finite elements; the response predicted by eight finite elements is slightly stiffer and less accurate than the response predicted by four beam-column elements.

The result obtained using four beam-column elements is also compared to the Bathe's solution [9] using eight finite elements (Fig. 7.8); it can be seen that the beam-column solution produces a similar level of accuracy using a much coarser mesh. The deflected shapes of the cantilever beam at various load levels are depicted in Fig. 7.9. The execution times for both models are as follows:

four elements:

finite element model: 5.66 sec

beam-column model: 7.19 sec

eight elements:

finite element model: 37.27 sec

beam-column model: 37.35 sec

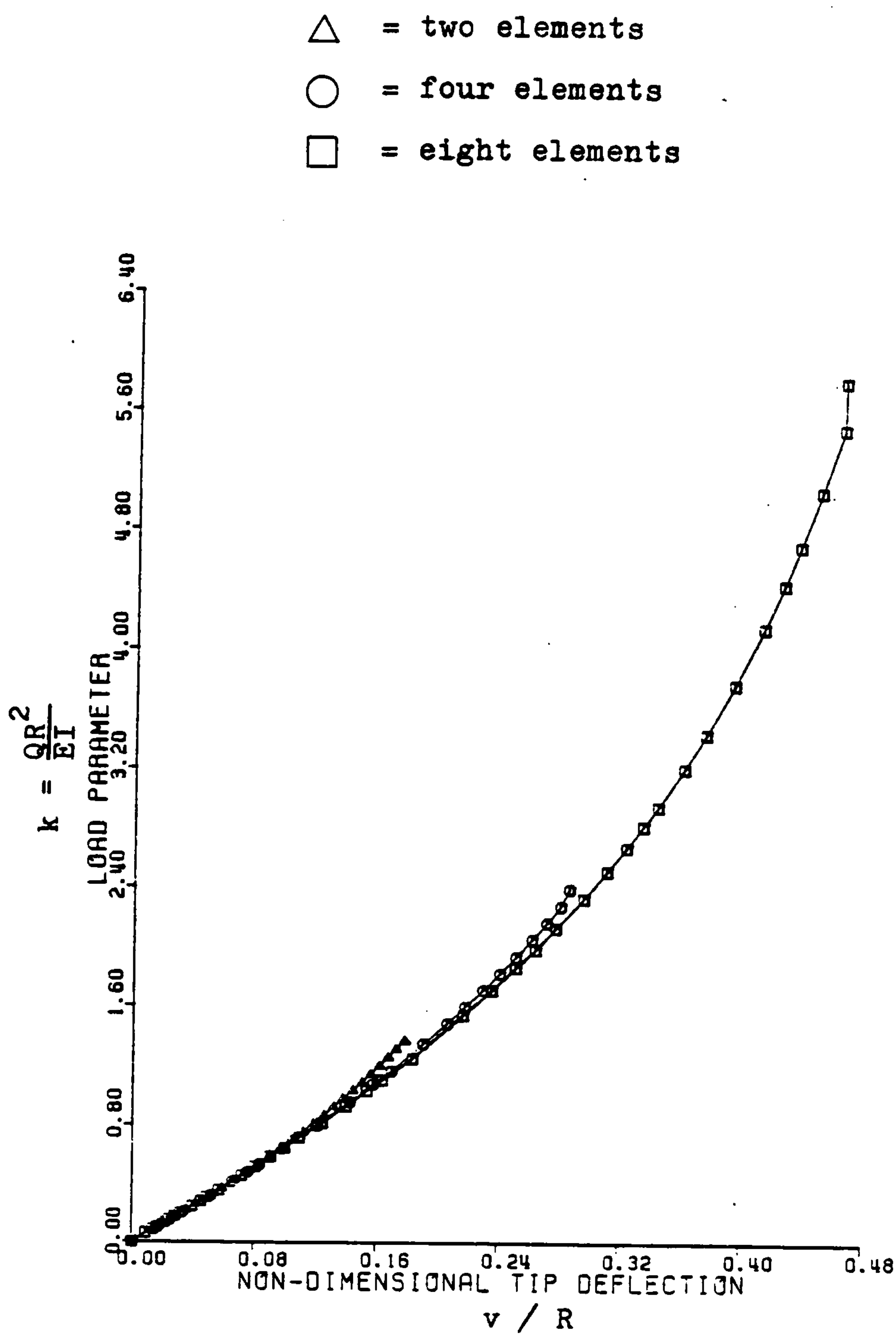


Fig. 7.5 Finite Element Model

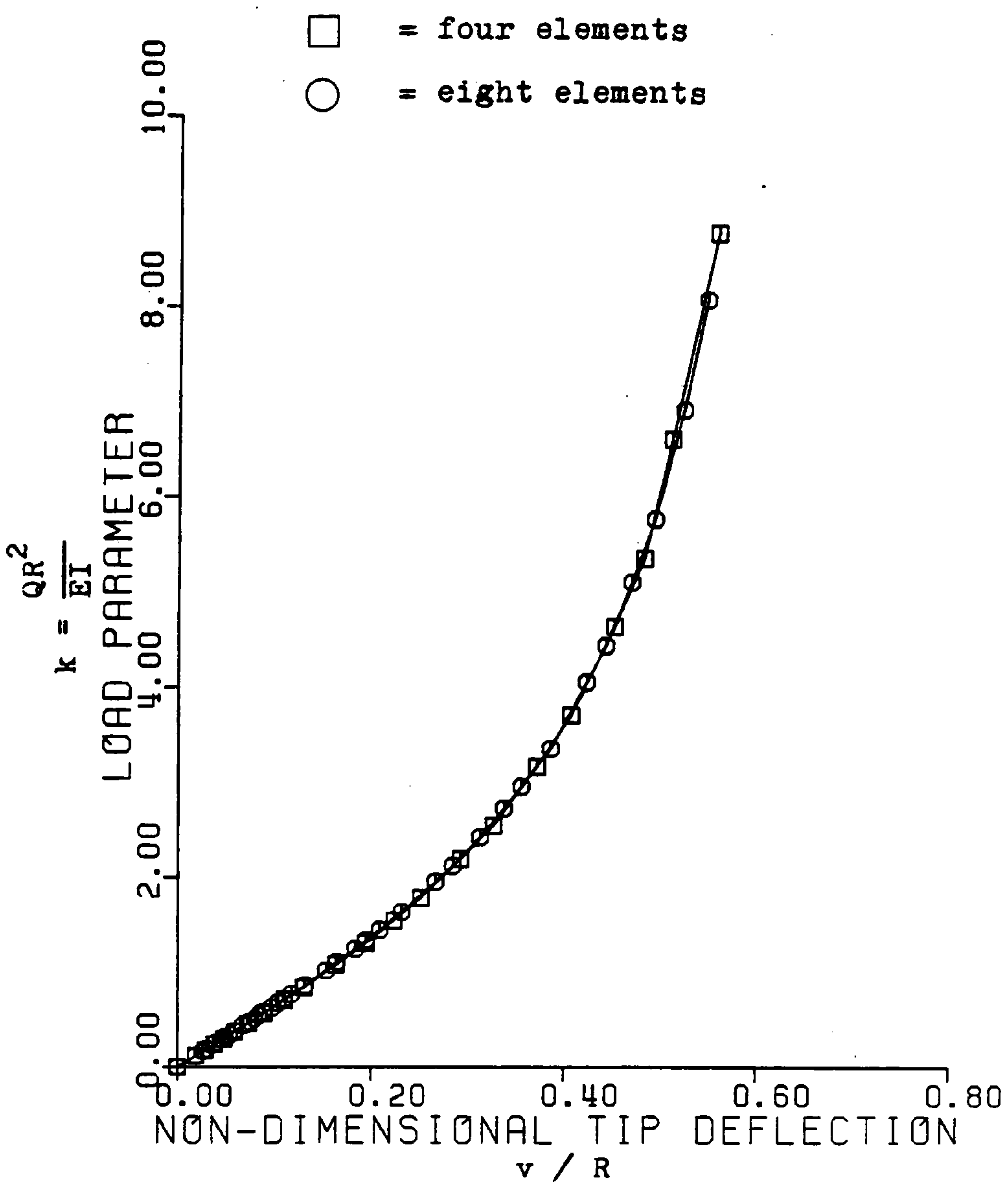


Fig. 7.6 Beam-Column Model

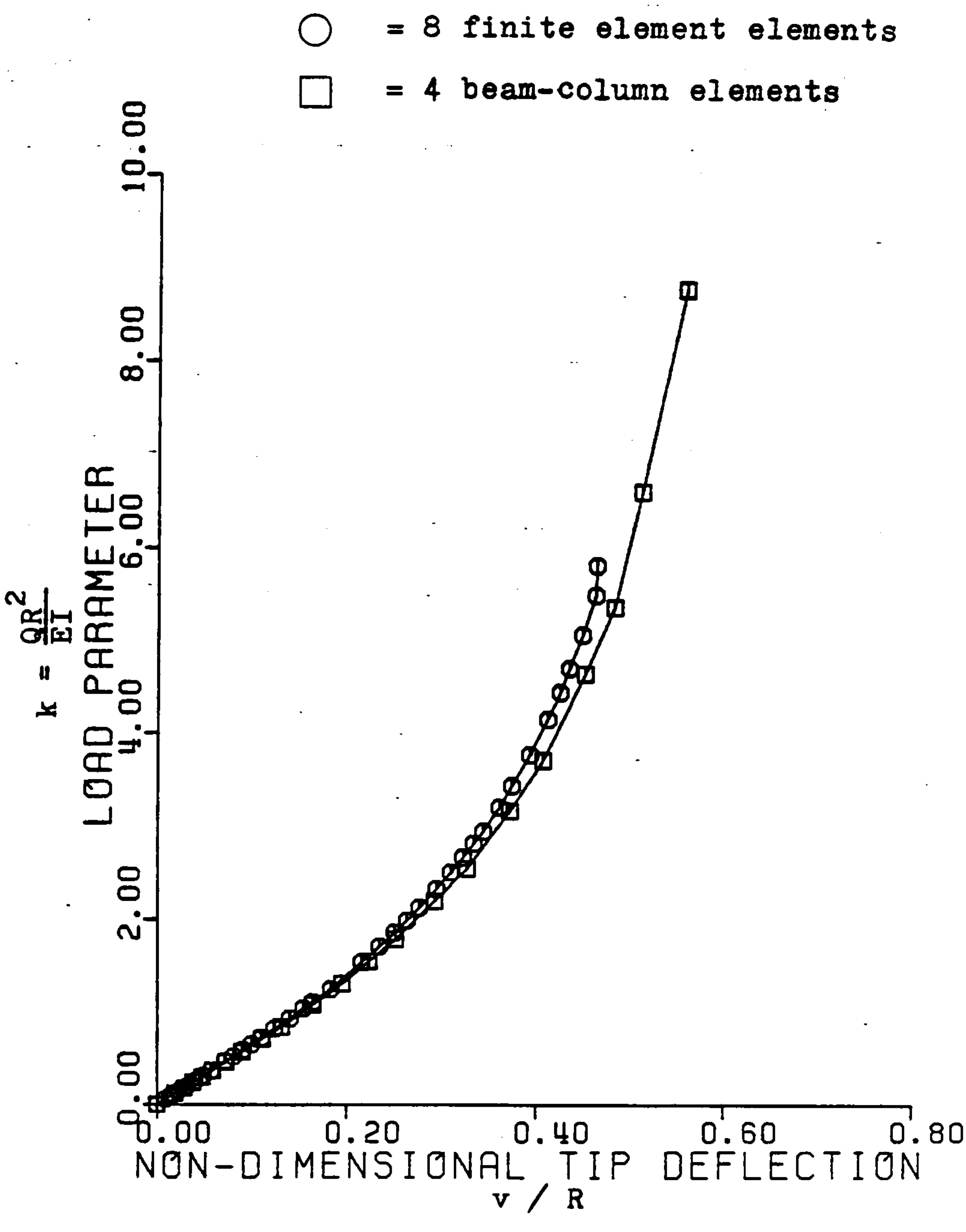


Fig. 7.7 Comparison of Models

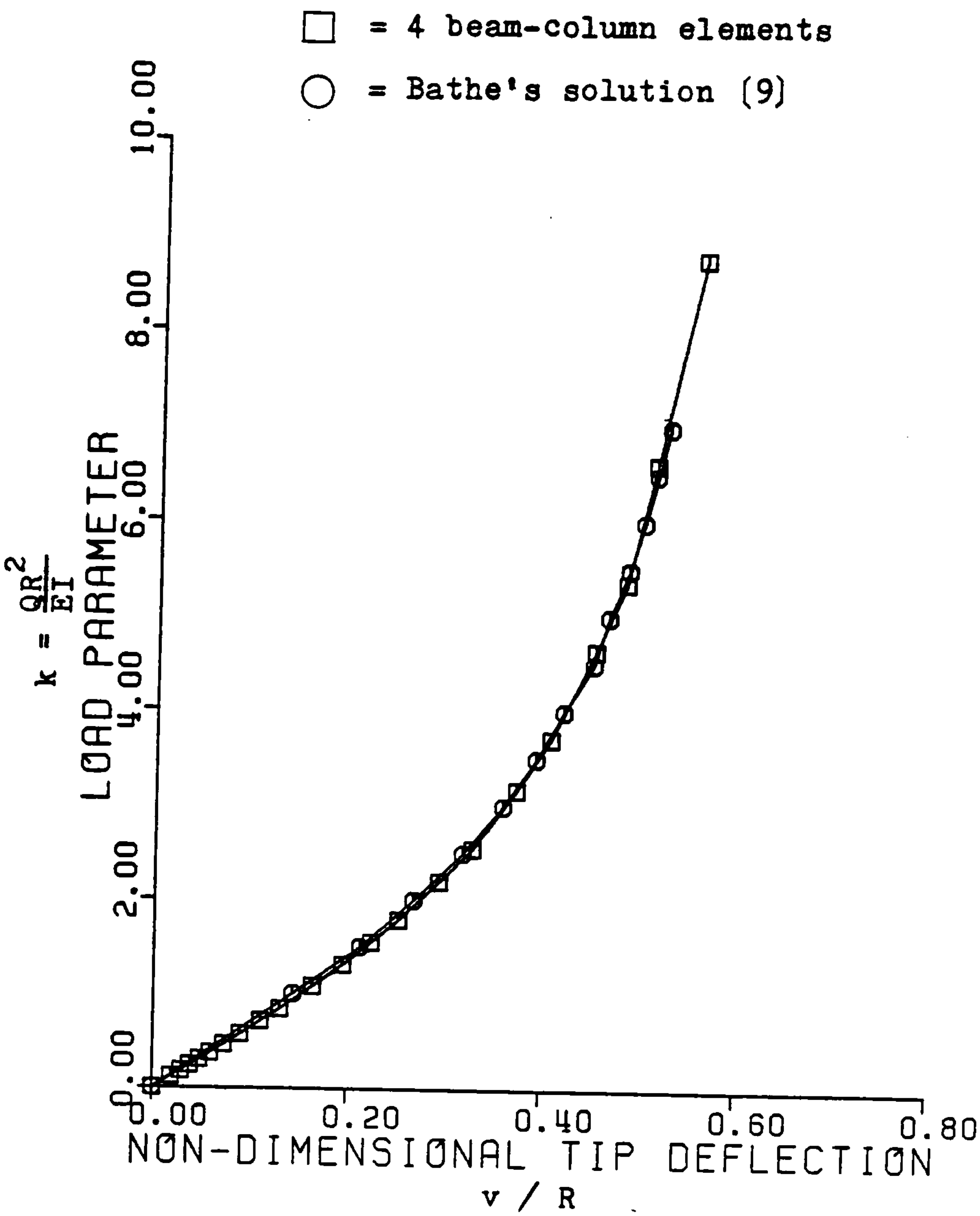
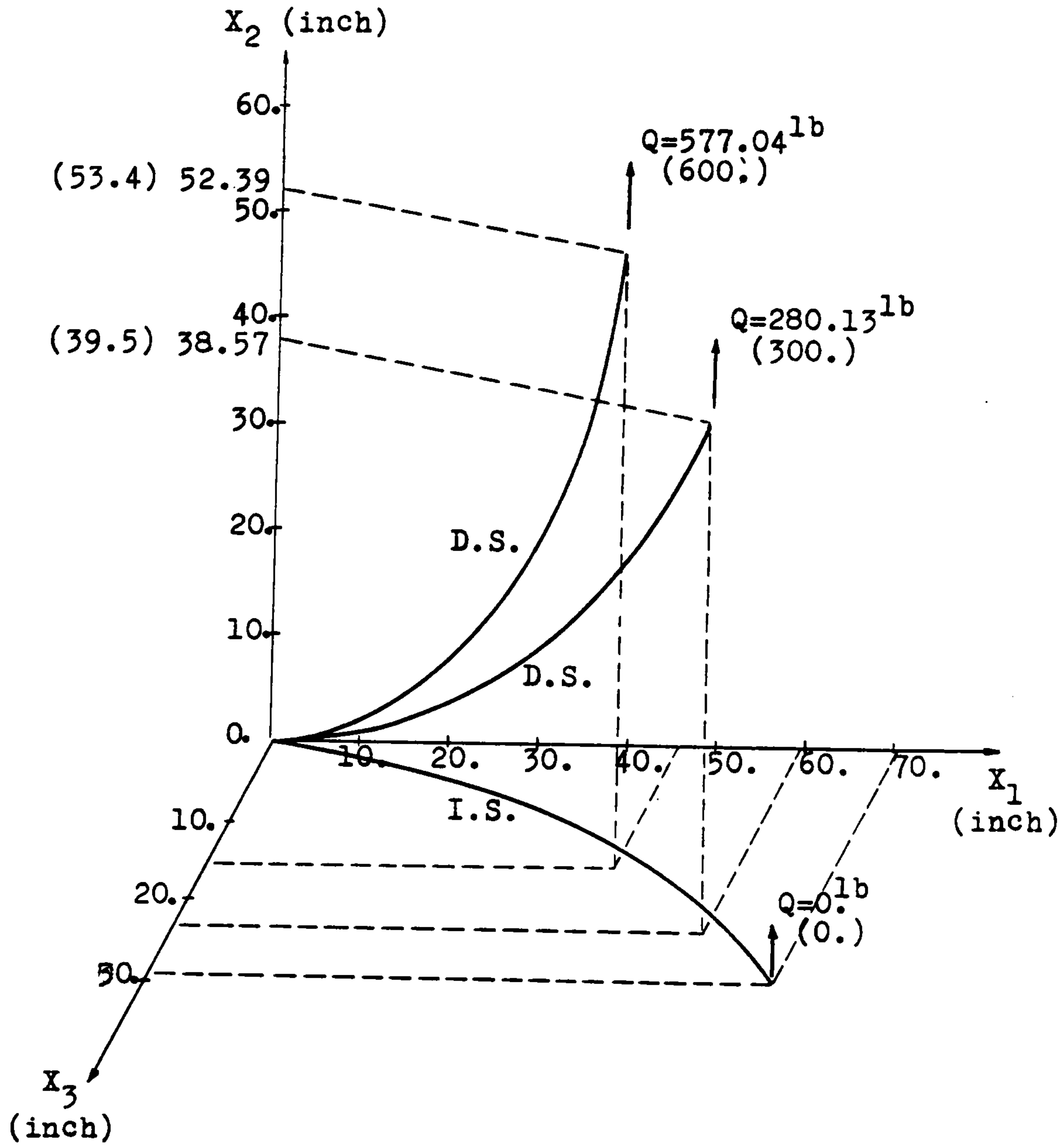


Fig. 7.8 Load Deflection Curves



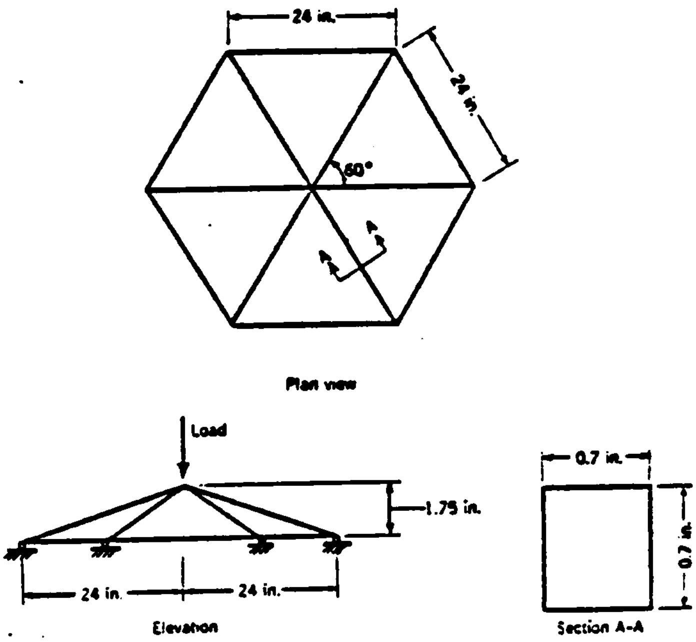
( ) = Bathe's results [9]

Fig. 7.9 Deflected Shapes of a 45° Circular Bend using the Beam-Column Model

7.4 EXAMPLE 3: 12 MEMBER MODEL FRAME

The model frame shown in Fig. 7.10 was analyzed by Connor [16], Papadrakakis [21],and Chu [39], one and two elements per member are employed in this study. Fig. 7.11 shows the finite element solution; it can be seen that one element cannot accurately represent the behavior of the structure. Fig. 7.12 presents the beam-column solution; it gives similar results for both meshes. In Fig. 7.13 a comparison of equilibrium paths up to the limit point of the finite element model with the beam-column model is presented; both paths agree quite closely. The numerical solutions of others workers [16],[21] and [39] are given in Fig. 7.14. In Fig. 7.15 a comparison of Papadrakakis solution with the two beam-column elements/member solution is presented; the results are in close agreement.

Combining Fig. 7.12,7.13 and 7.15, it is seen that the beam-column solution obtained using one element per member give satisfactory result in which two elements of finite element solution are needed, to predict the structural response.



$E = 439{,}800.$ psi

$G = 159{,}000.$ psi

$J = 0.141\ a^4 = 0.0344382\ \text{in}^4$

Fig. 7.10 12 Member Model Frame

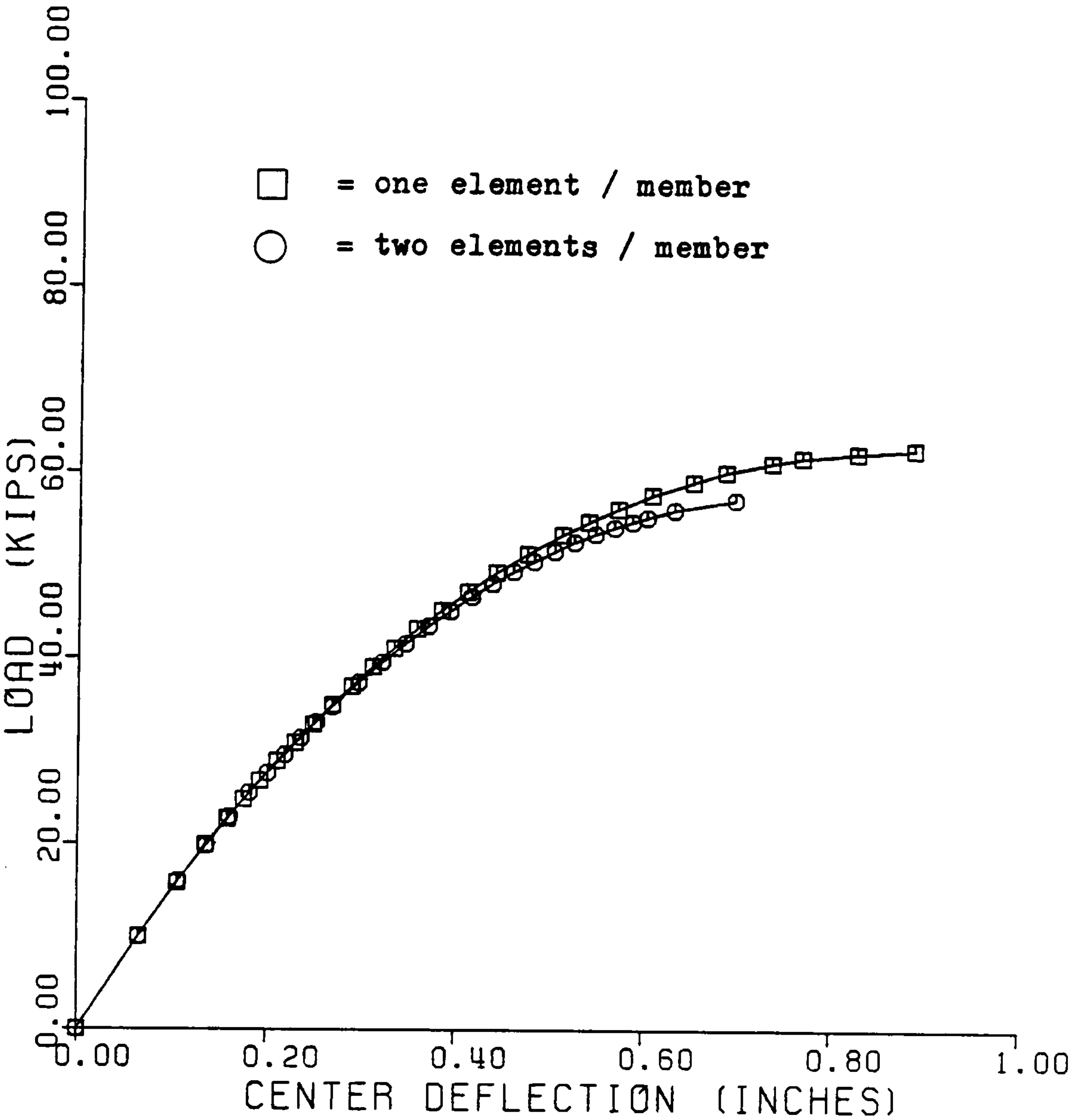


Fig. 7.11 Finite Element Model

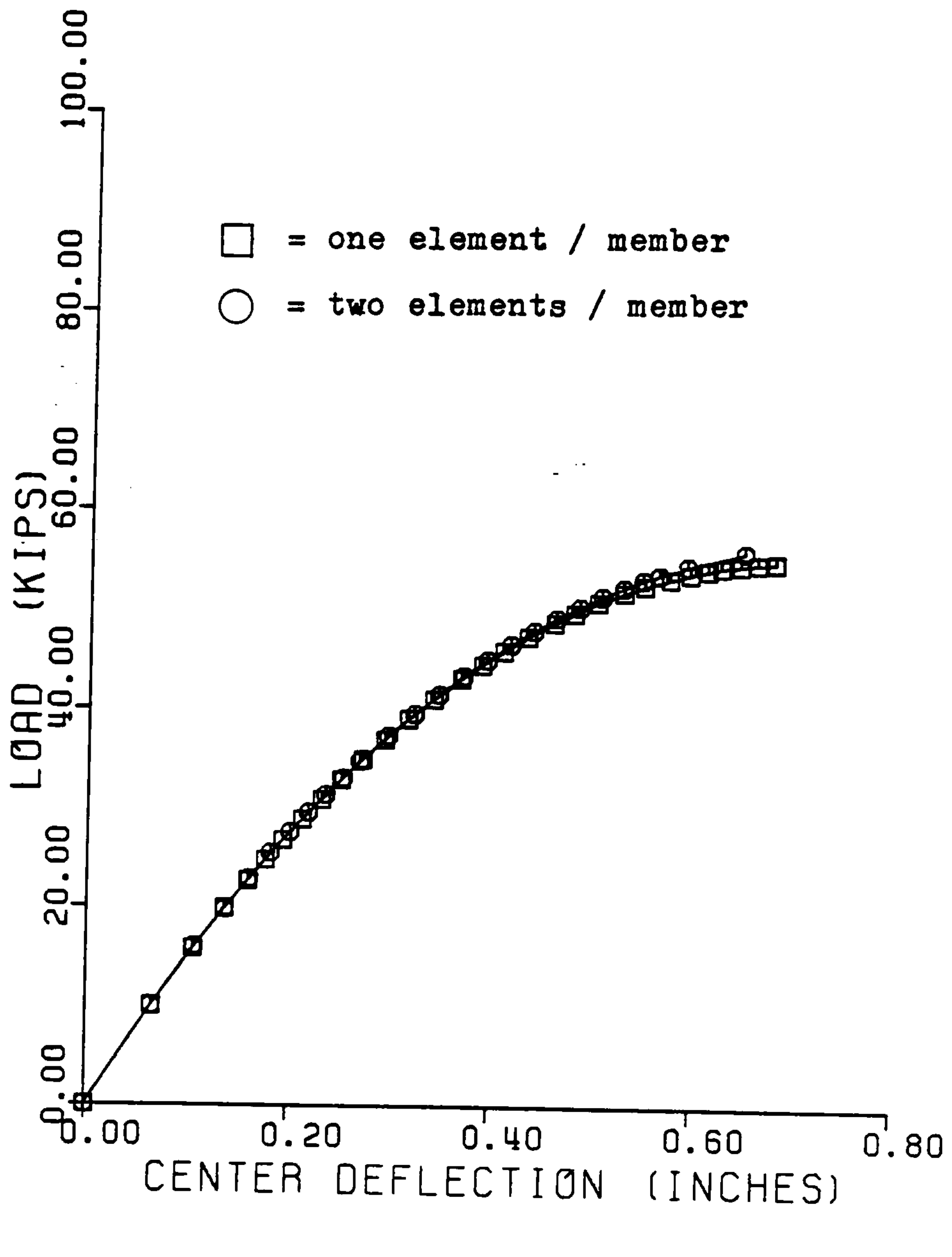


Fig. 7.12 Beam-Column Model

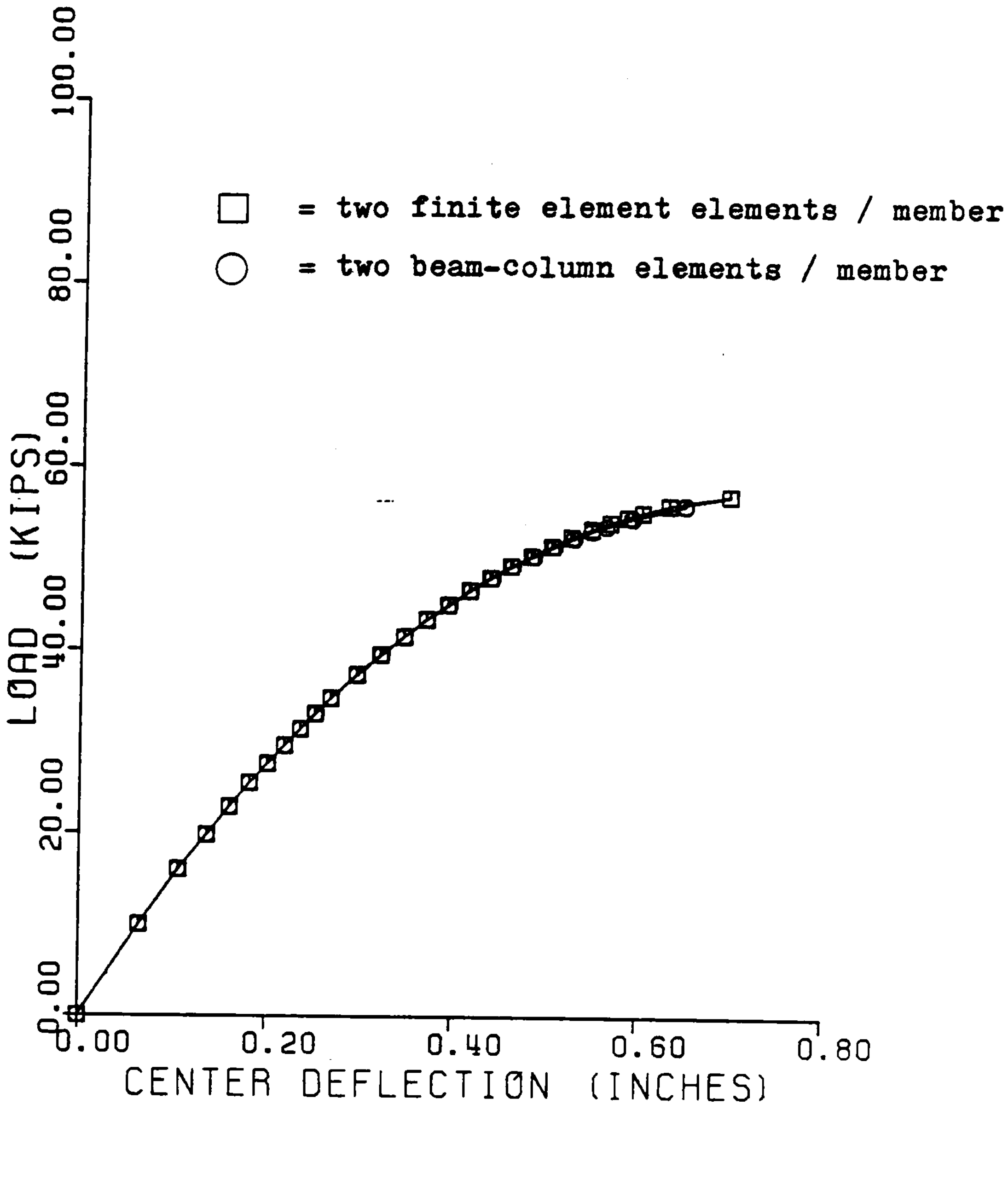


Fig. 7.13 Comparison of Models

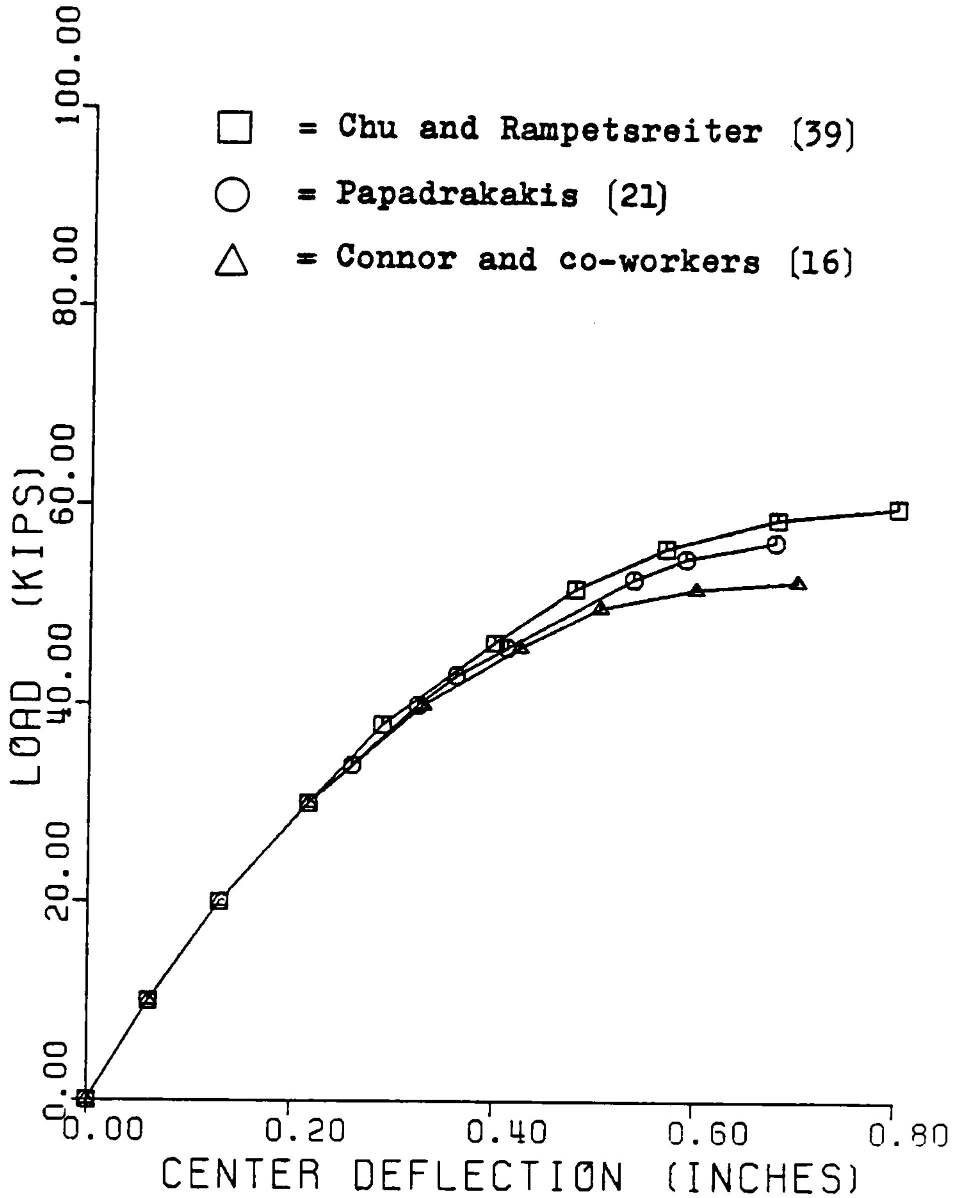


Fig. 7.14 Load Deflection Curves

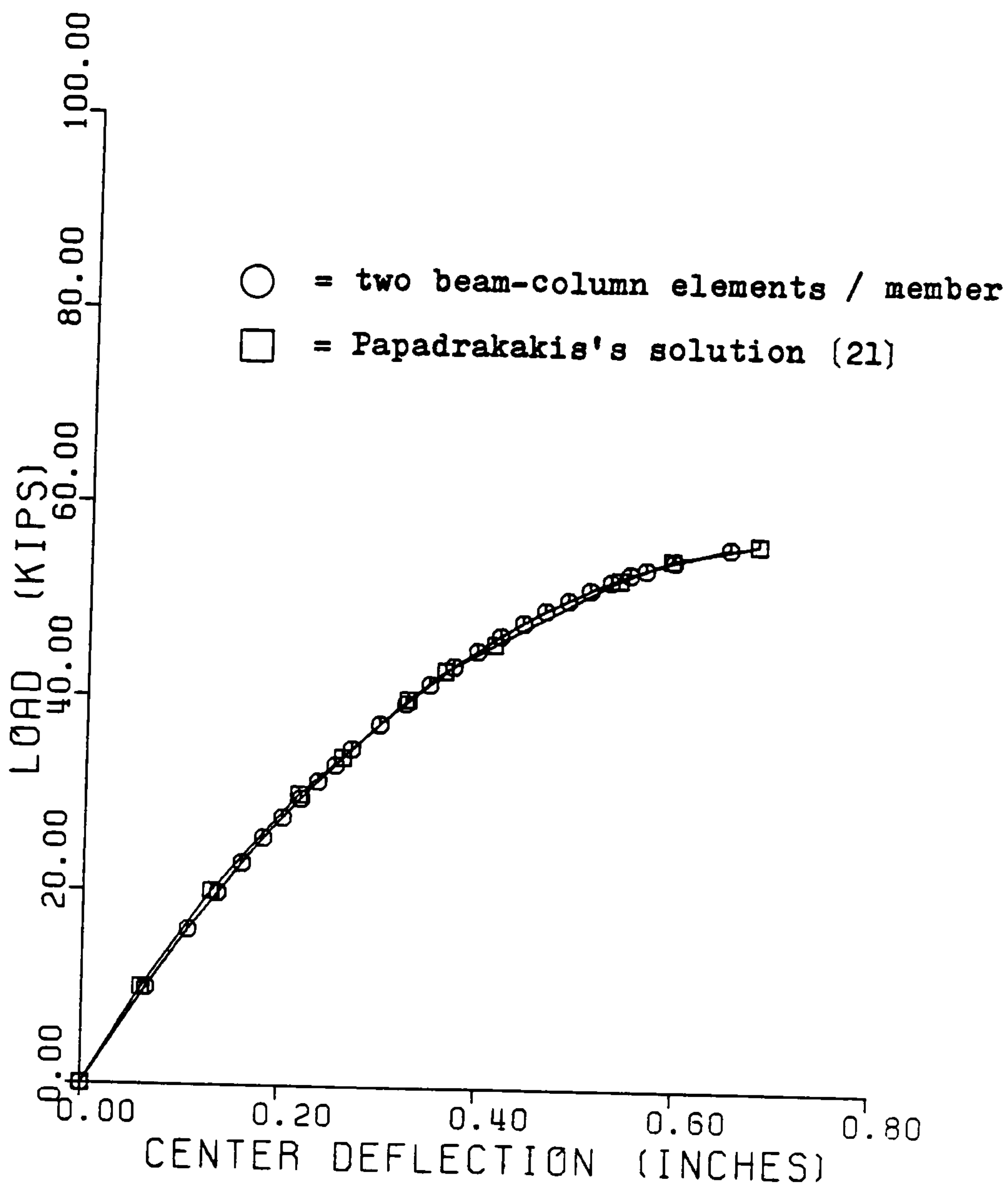


Fig. 7.15 Load Deflection Curves

Employing two elements per member, the buckling load obtained by the beam-column model is 56.14 lb and by the finite element model is 57.24 lb. The execution time for both models are as follows:

one element:

finite element model:	10.55 sec
beam-column model:	14.55 sec

two elements:

finite element model:	71.53 sec
beam-column model:	75.59 sec

7.5 EXAMPLE 4: RETICULATED DOME

Shallow truss domes were investigated by Holzer [27],[51], Hangai [52], Paradiso [53]. In this study a shallow frame dome designed in accordance with the Specification of the Aluminum Association [54] is considered. The geometry of the reticulated frame dome is given in Fig. 7.16. The dome is subjected to a single vertical load at the center.

Fig. 7.17 presents the finite element solution obtained using one and two elements per member; it is seen that single element is inadequate to model the structural behavior.

Fig. 7.18 presents the beam-column solution obtained using one and two elements per member; similar results for both meshes are obtained. Fig. 7.19 presents a comparison of two finite elements to one beam-column element; it shows again that fewer elements in the beam column model are often needed than in the finite element model for satisfactory accuracy.

In addition Fig. 7.19 shows a comparison of two extreme cases, the shallow truss dome, considered by Uliana [55], with the present frame dome, is presented; it shows that the behavior of the frame dome remains linear when the truss dome reaches the limit point due to a central, concentrated load. Clearly, the frame dome is much stronger than the truss dome. The execution time for both models are given as follows:

one element:

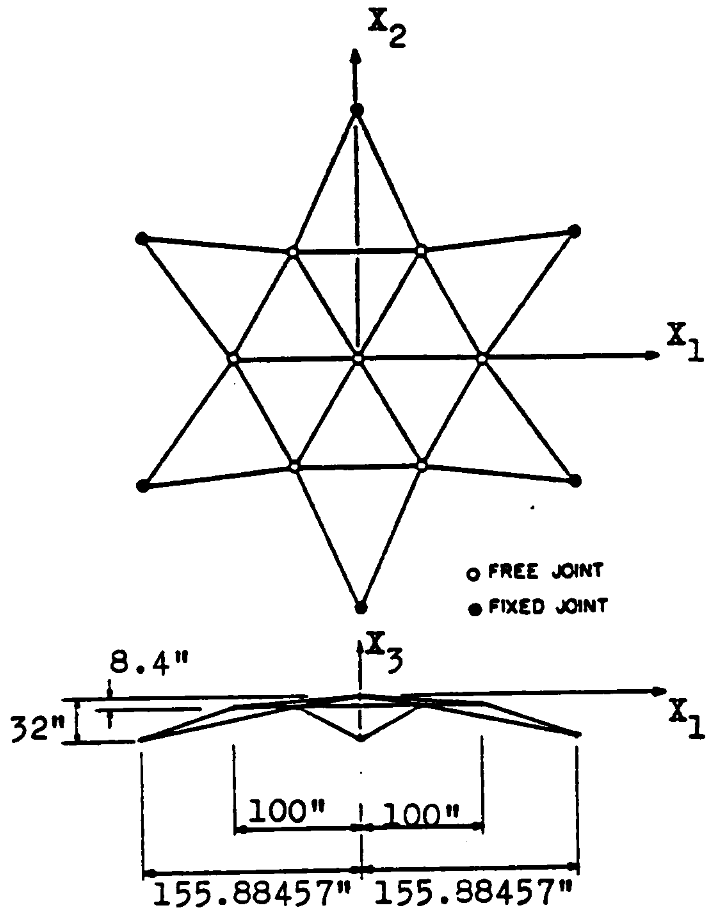
finite element model: 52.71 sec

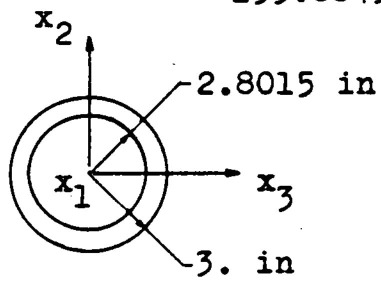
beam-column model: 60.53 sec

two elements:

finite element model: 339.24 sec

beam-column model: 459.41 sec





tubular section

$A = 3.615 \text{ in}^2$

$E = 10{,}300. \text{ ksi}$

$I_2 = I_3 = 15.34 \text{ in}^4$

$J = I_2 + I_3 = 30.68 \text{ in}^4$

$G = \dfrac{E}{2(1+\nu)}$

$\nu = 0.3$

Fig. 7.16 Reticulated Dome

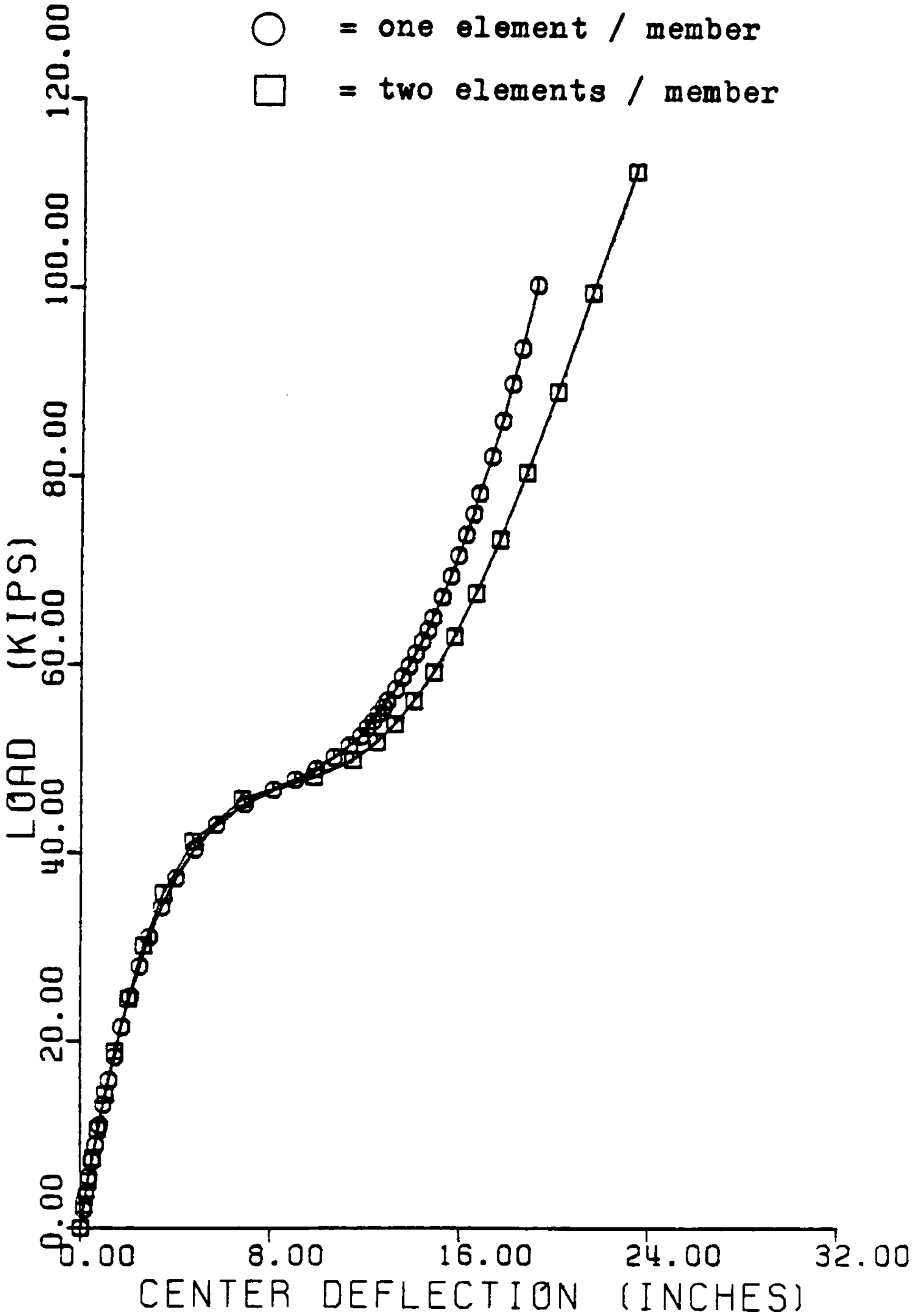


Fig. 7.17 Finite Element Model

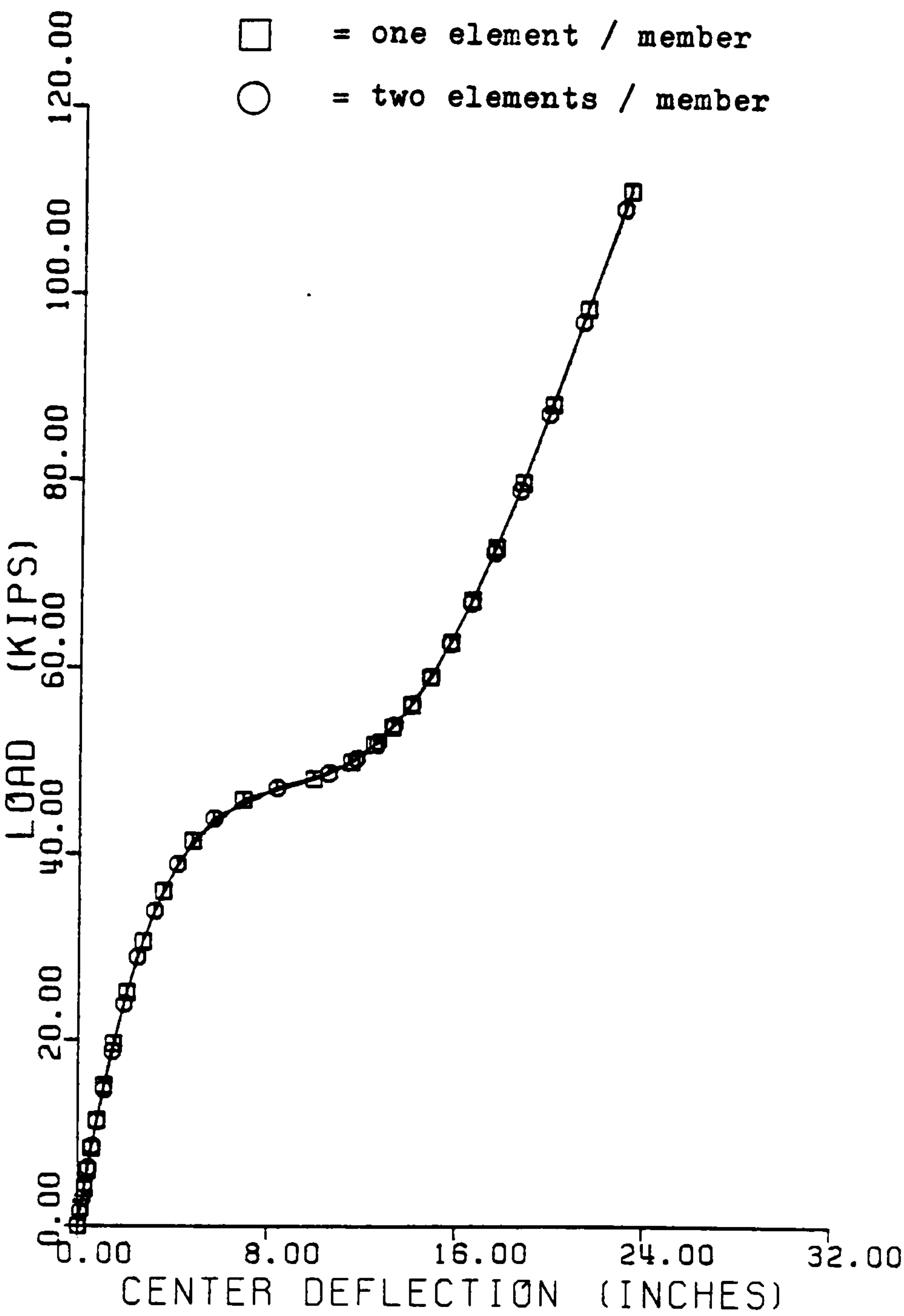


Fig. 7.18 Beam-Column Model

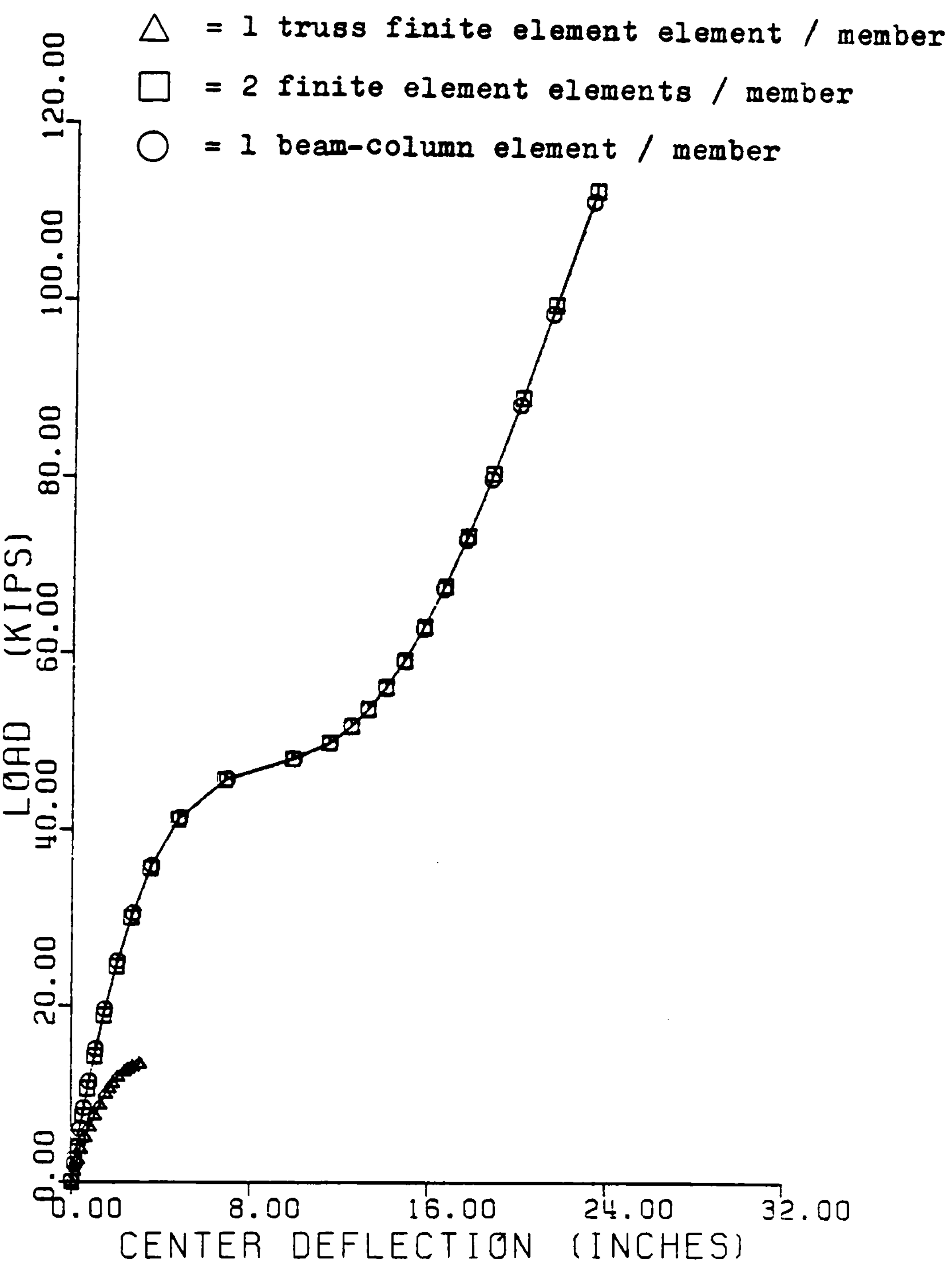


Fig. 7.19 Comparison of Frame Dome with Truss Dome

Chapter VIII

PROGRAM DEVELOPMENT

8.1 INTRODUCTION

The program developed by Butler [29], for the geometrically nonlinear static analysis of plane frames, has been modified and extended to the GNSSF (Geometrically Nonlinear Static Analysis of Space Frames) program using U.L. formulation as in appendix F. The modified program has the capability to analyze space frames with arbitrary large displacement. It is restricted to continuous, elastic structures, i.e., only geometrical nonlinearity is considered.

The GNSSF computer program is written in the WATFIV version of FORTRAN code having the following options:

1. Newton-Raphson method or modified Riks/Wempner method
2. beam-column model or finite element model

Subroutines FORCES and STIFF can be approached by using either the beam-column model or the finite element model. Subroutines FACTOR, REDUCE, and BACSUB, adapted from Bathe [38] and Subroutines SOLVE, TEST, DISPLC, DISPLB, UNBALF, UPDATE, DOTPRD, adapted from Butler [29], will not be described here.

8.2 PROGRAM STRUCTURE

Fig. 8.1 presents the tree chart of the GNSSF computer program. The MAIN program and each subroutine are described by Nassi-Schneiderman (N-S) diagrams in this section [28].

8.2.1 Main Program

Function: Initialize and read in control parameters; select the mathematical model, INDTAN; determine the desired number of iterations in the future step; call DATA; initialize the joint and element orientation matrices to the undeformed state; determine the type of solution algorithm, call either NEWRAP or RIKWEM.

N-S diagram:

Initialize the parameters of solution tolerance CPDB, CPDC, CPF, the parameters of maximum limit MX, MXNA, MXNEQ, the number of iteration limits, IMAX, ITMAX; determine the mathematical model to be used, INDTAN, the type of solution algorithm, ITIND, the desired number of iterations in the future step, ITDES
Read IMP, SIGN, KT

Echo input data	
Call DATA	
Initialize F, FP, FPI, D, DD, DDO, Z, to zero; let Q=QJ	
Initialize the joint orientation matrix at time zero to identity martrix	
Calculate the components of the initial orientation matrix of element I	
If ITIND=0	
then	else
Call NEWRAP	Call RIKWEM

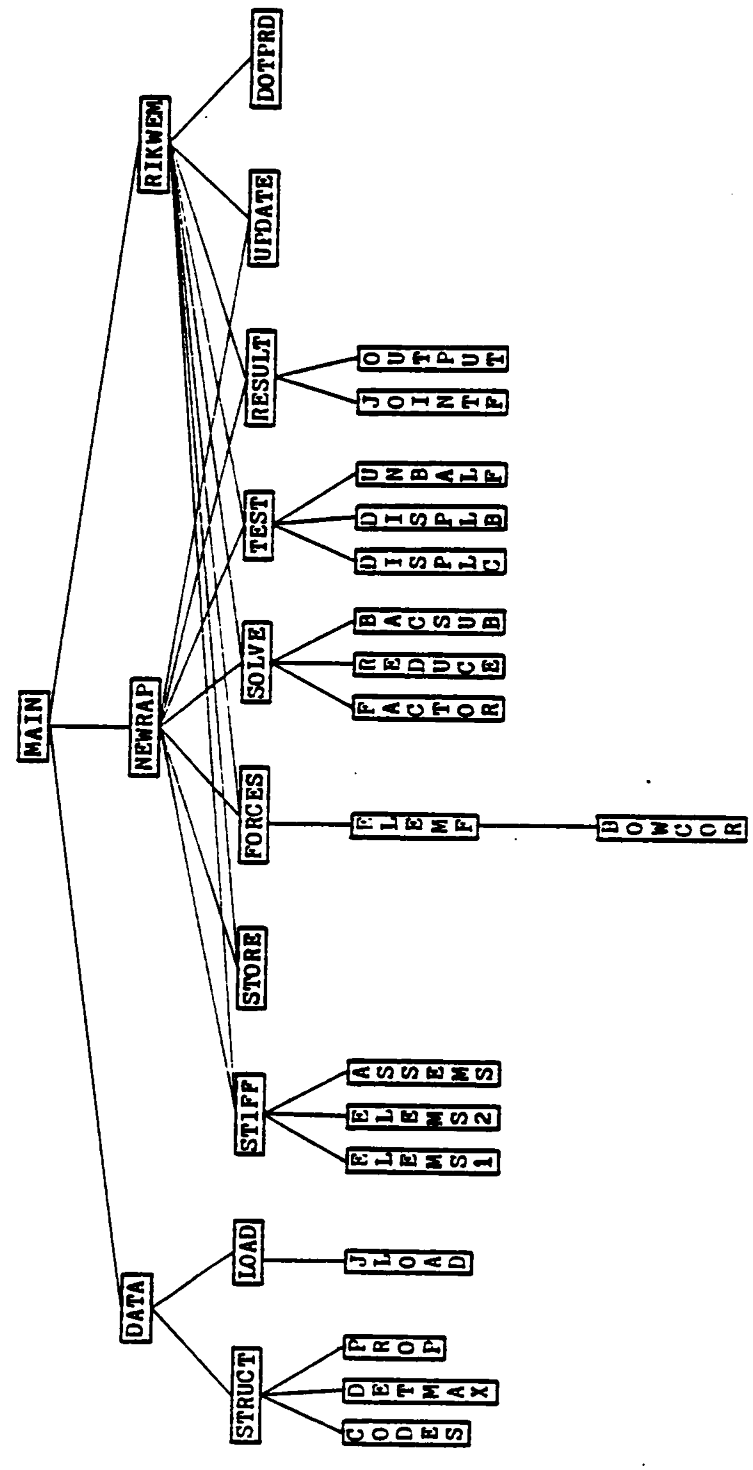


Fig. 8.1 Program Structure

8.2.2 Subroutine DATA

Function: Read and echo NE, NJ; call STRUCT and LOAD.

Input arguments: MX, MXNA, MXNEQ

Output arguments: MINC, JCODE, MCODE, NEQ, MAXA, NKT, X, AREA, ZI2, ZI3, ZJ, EMOD, GMOD, ZJ, NE, NJ, Q, CO1, CO2, CO3, C1PI, C2PI, C3PI, C1, C2, C3, ELENGO, ELENG

N-S diagram:

<table>
<tr><td colspan="2">Read and echo NE and NJ</td></tr>
<tr><td colspan="2">If NE $\leqslant$ MX and NJ $\leqslant$ MX</td></tr>
<tr><td>then</td><td>else</td></tr>
<tr><td>Call STRUCT</td><td rowspan="2">Print error message</td></tr>
<tr><td>Call LOAD</td></tr>
</table>

8.2.3 Subroutine STRUCT

Function: Read echo, and process the structural data.

Input arguments: NE, NJ, MXNEQ

Output arguments: MINC, JCODE, MCODE, NEQ, MAXA, NKT, X, AREA, ZI2, ZI3, ZJ, EMOD, GMOD, ELENGO, ELENG, CO1, CO2, CO3, C1PI, C2PI, C3PI, C1, C2, C3

N-S diagram:

```
Read, echo MINC(1,I), MINC(2,I)   for I=1 to NE
Initialize JCODE(L,J)=1   for J=1 to NJ, L=1 to 6
Read joint constraint:  JNUM, JDIR
While JNUM ≠ 0
        Print JNUM, JDIR
        JCODE(JDIR,JNUM) = 0
        Read joint constraint JNUM, JDIR
Call CODES
Call DETMAX
Call PROP
```

8.2.4 <u>Subroutine CODES</u>

Function: Generate the JCODE and MCODE.

Input arguments: JCODE, MINC, NE, NJ, MXNEQ

Output arguments: MCODE, JCODE, NEQ

N-S diagram:

```
NEQ = 0
Do for J = 1 to NJ
    Do for L = 1 to 6
        If JCODE(L,J) ≠ 0
        then
            NEQ = NEQ+1
            JCODE(L,J) = NEQ
        else
Do for I = 1 to NE
    J = MINC(1,I)
    K = MINC(2,I)
    Do for L = 1 to 6
        MCODE(L,I) = JCODE(L,J)
        MCODE(L+6,I) = JCODE(L,K)
```

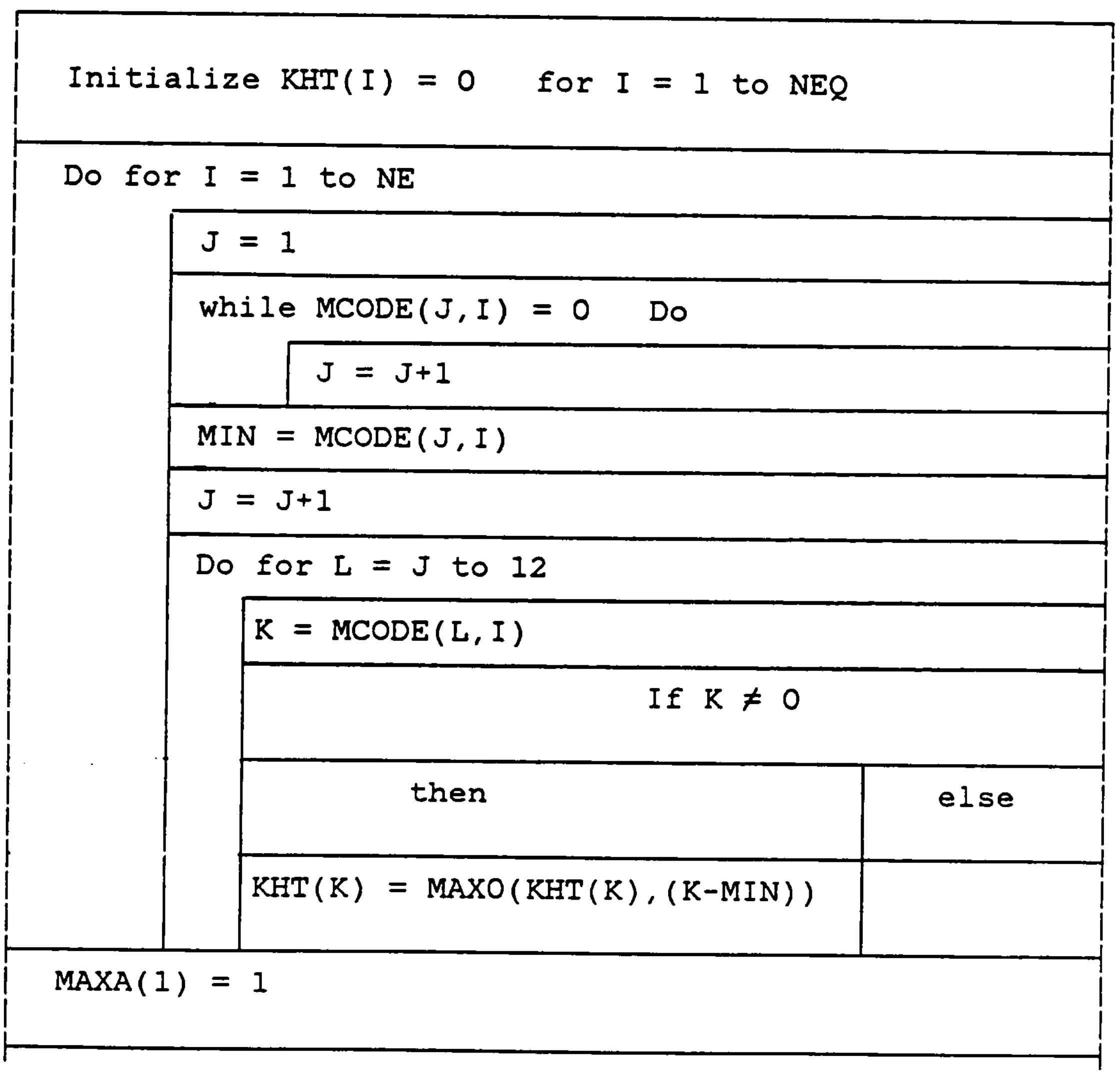
8.2.5 Subroutine DETMAX

Function: To calculate column heights, KHT(NEQ); addresses of diagonal elements in banded matrix K (column heights known), MAXA(NEQ+1); and the number of elements below skyline of matrix K, NKT.

Input arguments: NEQ, NE, MCODE

Output arguments: MAXA, NKT

N-S diagram:



```
DO for I = 1 to NEQ
        MAXA(I+1) = MAXA(I) + KHT(I)+ 1
NKT = MAXA(NEQ+1) - 1
```

8.2.6 Subroutine PROP

Function: Read and echo the joint coordiates, X; compute and echo the undeformed element length and direction cosines, ELENGO, CO1, CO2, CO3; store ELENGO, CO1, CO2, CO3 in the previous iteration state, ELENG, C1PI, C2PI, C3PI; read and print element properties: AREA, ELENGO, EMOD, GMOD, ZI2, ZI3, ZJ.

Input arguments: NJ, NE, MINC

Output arguments: X, ELENGO, ELENG, CO1, CO2, CO3, C1PI, C2PI, C3PI, AREA, EMOD, GMOD, ZI2, ZI3, ZJ

N-S diagram

```
Read and echo joint coordinates, X(I,J) for I=1 to
3, J=1 to NJ

Do for I = 1 to NE
    J = MINC(1,I)
    K = MINC(2,I)
    EL1 = X(1,K) - X(1,J)
    EL2 = X(2,K) - X(2,J)
    EL3 = X(3,K) - X(3,J)
    ELENGO(I) = DSQRT(EL1**2 + EL2**2 + EL3**2)
    ELENG(I) = ELENGO(I)
    CO1(I) = EL1 / ELENGO(I)
    CO2(I) = EL2 / ELENGO(I)
```

CO3(I) = EL3 / ELENGO(I)
C1PI(I) = CO1(I)
C2PI(I) = CO2(I)
C3PI(I) = CO3(I)
Read AREA(I), EMOD(I), GMOD(I), ZI2(I), ZI3(I), ZJ(I)
Echo AREA(I), ELENGO(I), EMOD(I), GMOD(I), ZI2(I), ZI3(I), ZJ(I)

8.2.7 Subroutine LOAD

Function: Initialize to zero the joint load vector, Q; Call JLOAD and MACT.

Input arguments: NEQ, JCODE

Output arguments: Q

N-S diagram:

Initialize joint load Q to zero
Call JLOAD
Call MACT

8.2.8 Subroutine JLOAD

Function: Read JNUM, JDIR, and FORCE; while JNUM = 0, print JNUM, JDIR, FORCE; store FORCE in Q, and read JNUM, JDIR, FORCE.

Input arguments: JCODE

Output arguments: Q

N-S diagram:

```
Read JNUM, JDIR, FORCE
while  JNUM ≠ 0
        Print JNUM, JDIR, FORCE
        K = JCODE(JDIR,JNUM)
        Q(K) = FORCE
        Read JNUM, JDIR, FORCE
```

8.2.9 Subroutine NEWRAP

Function: Perform the Newton-Raphson or modified Newton-Raphson method to trace the prelimit equilibrium path. Call STIFF, STORE, SOLVE, FORCES, TEST, RESULT, and UPDATE.

Input arguments: AREA, C01, C02, C03, C1PI, C2PI, C3PI, C1, C2, C3, ELENGO, ELENG, EMOD, GMOD, JCODE, MAXA, MCODE, MINC, ITIND, INDTAN, KT, IMP, Q, QJ, CPDB, CPDC, CPF, DD, DDO, DQQR, FQR, IMAX, ITMAX, NE, NJ, NEQ, NKT, NUPD, QI, QIMAX, DQI, NPRINT, IELS, AP, OLAMD

Output arguments: none

N-S diagram:

```
Initialize the total relative deformations of element I to zero

Do I = 1 to NE
    TR12(I) = 0.DO
    TR22(I) = 0.DO
    TR13(I) = 0.DO
    TR23(I) = 0.DO
    TTORS(I)= 0.DO
    TUL(I)  = 0.DO
```

```
        TCB2(I) = 0.DO
        TCB3(I) = 0.DO
        TU(I)   = 0.DO
NN = 1
Call FORCES
Call UPDATE
While QI ≤ QIMAX
    NC = NUPD
    ITCT = 0
    ICI = 1
    While ICI ≠ 0 and ITCT ≤ ITMAX
            If NC ≥ NUPD
        then                | else
        Call STIFF          |
        NC = 0              |
        Call STORE
        Call SOLVE
        Do for I = 1 to NEQ
            D(I) = D(I)+DD(I)
        Call FORCES
        Call TEST
        NC = NC+1
```

```
        ITCT = ITCT+1
        Do for I = 1 to NEQ
            FPI(I) = F(I)
    Do for I = 1 to NEQ
        FP(I) = F(I)
    Call RESULT
    QI = QI+DQI
    If ICI ≠ 0
    then
        Print 'last solution is not converged'
        Stop
    else
Call UPDATE
```

8.2.10 Subroutine RIKWEM

Function: Perform the Riks/Wempner or modified Riks/Wempner iteration on the normal plane. Call STIFF, STORE, SOLVE, FORCES, TEST, RESULT, and UPDATE.

Input arguments: AREA, CO1, CO2, CO3, C1PI, C2PI, C3PI, C1, C2, C3, ELENGO, ELENG, EMOD, GMOD, JCODE, MAXA, MCODE, MINC, ITIND, INDIAN, KT, IMP, Q, QJ, CPDB, CPDC, CPF, DD, DDO, DQQR, FQR, IMAX, ITMAX, NE, NJ, NEQ, NKT, NUPD, QI, QIMAX, DQI, NPRINT, SIGN, IELS, AP, OLAMD

Output arguments: none

N-S diagram:

```
ITCT = 0  ; NEGPIV = 0

Initialize the total relative deformations of ele-
ment I to zero

Do I = 1 to NE
    TR12(I) = 0.DO
    TR22(I) = 0.DO
    TR13(I) = 0.DO
    TR23(I) = 0.DO
    TTORS(I)= 0.DO
    TUL(I)  = 0.DO
```

```
        TCB2(I) = 0.D0
        TCB3(I) = 0.D0
        TU(I)   = 0.D0
NN = 1
Call FORCES
While QI ≤ QIMAX and ITCT ≤ ITMAX
    Call STIFF
    Call STORE
    Call SOLVE : SKT DD01 = Q  for DD01
    If ITCT = 0
      then:
        DS=DQI*[DOTPRD(DD01.DD01)+1]^1/2
      else:
        DQI=SIGN(NEGPIV+1)*DS/[DOTPRD(DD01.DD01)+1]^1/2
    DQI1 = DQI
    Do for I = 1 to NEQ
        DD0(I) = DQI*DD01(I)
        D(I) = D(I)+DD0(I)
    QI = QI+DQI
    NN = 1
    Call FORCES
    ICI = 1 ; NC = NUPD ; IT = 0
    While ICI ≠ 0 and IT ≤ IMAX
        Call UPDATE
```

If NC ≥ NUPD

then	else
Call STIFF	
NC = 0	

Call STORE

Call SOLVE: SKT DD1 = Q for DD1

Call SOLVE: SKT DD2 = QT-F for DD2

DQI =

-DOTPRD(DD0.DD2)/(DOTPRD(DD0.DD1)+DQI1)

Do for I = 1 to NEQ

- DD(I) = DQI*DD1(I)+DD2(I)
- D(I) = D(I) + DD(I)

NN = 2

Call FORCES

QI = QI + DQI

Call TEST

NC = NC + 1 ; IT = IT + 1

ITCT = ITCT + 1

Do for I = 1 to NEQ

- FP(I) = F(I)

Call RESULT

If ICI ≠ 0

then	else

Print 'last solution is not converged'

Stop

$DS = DS*(ITDES/IT)^{1/2}$

8.2.11 Subroutine STIFF

Function: Initialize the system tangent stiffness matrix, SKT, to zero; for each element call either ELEMS1 for INDTAN = 1, or ELEMS2 for INDTAN = 2, and call ASSEMS.

Input arguments: AREA, BP12, BP22, BP13, BP23, C1, C2, C3, ELENGO, EMOD, GMOD, MAXA, MCODE, TR12, TR22, TR13, TR23, SP12, SP22, SP13, SP23, ST12, ST22, ST13, ST23, RM32, RM33, QL, TU, INDTAN, KT, ZI2, ZI3, ZJ, IELS, NE, NKT, NPRINT, OLAMD

Output arguments: SKT

N-S diagram:

Initialize SKT(I) = 0.D0 for I = 1 to NKT

I = 1

While I ≤ NE

If INDTAN = 1

then | else

Call ELEMS1 |

If INDTAN = 2

then | else

	Call ELEMS2	
	Call ASSEMS	
	I = I + 1	
If NPRINT = 3		
then		else
Print SKT(K) for K = 1 to NKT		

8.2.12 Subroutine ELEMS1

Function: Compute the components of the element tangent stiffness matrix for the beam-column model.

Input arguments: AREA, BP12, BP22, BP13, BP23, C1, C2, C3, EMOD, GMOD, TR12, TR22, TR13, TR23, SP12, SP22, SP13, SP23, ST12, ST22, ST13, ST23, ZI2, ZI3, ZJ, I, IELS, NE, RM32, RM33, QL, NPRINT, KT, OLAMD, ELENGO

Output arguments: G

N-S diagram:

<table>
<tr><td colspan="3" align="center">If IELS = 0</td></tr>
<tr><td colspan="2">then</td><td align="center">else</td></tr>
<tr><td colspan="2">Compute G_{12}, G_{22}, G_{13}, G_{23}, H for undeformed state by Eqs. (5.13),(5.14) and (5.15)</td><td rowspan="3">Compute G_{12}, G_{22}, G_{13}, G_{23}, H for deformed state by Eqs. (5.13), (5.14) and (5.15)</td></tr>
<tr><td colspan="2" align="center">If I = NE</td></tr>
<tr><td>then</td><td align="right">else</td></tr>
<tr><td>IELS = 1</td><td></td><td></td></tr>
<tr><td colspan="3">Compute the components of the linear strain incre-</td></tr>
</table>

mental stiffness matrix k_L in the local coordinate system g'(1) to g'(20), by Eq. (5.31)

Compute the components of the linear strain incremental stiffness matrix K_L in the global coordinate system, g(1) to g(42), by Eqs.(5.41) and (5.42)

If KT(2) > 0

then	else
Add the contribution of the nonlinear strain incremental stiffness matrix in the global coordinate system, K_{NL}, to G(1),G(2),G(3),G(10),G(11),G(18) by Eq. (5.38)	

If NPRINT = 3

then	else
Print G(KI) of element I, for KI = 1 to 42	

8.2.13 Subroutine ELEMS2

Function: Compute the components of the element tangent stiffness matrix for the finite element model.

Input arguments: AREA, C1, C2, C3, ELENGO, EMOD, GMOD, I, ZI2, ZI3, ZJ, KT, QL, OLAMD

Output arguments: G

N-S diagram:

<table>
<tr><td colspan="2">Initialize the components of the lcoal element tangant stiffness matrix to zero</td></tr>
<tr><td colspan="2">If KT(1) > 0</td></tr>
<tr><td>then</td><td>else</td></tr>
<tr><td>Compute the components of the linear strain incremental stiffness matrix in the local coordinate system, k_L</td><td></td></tr>
<tr><td colspan="2">If KT(2) > 0</td></tr>
<tr><td>then</td><td>else</td></tr>
<tr><td>Compute the components of the nonlinear strain incremental stiffness matrix in the local coordinate system, k_{NL}</td><td></td></tr>
</table>

Compute the components of the global element tangent stiffness matrix, K^i

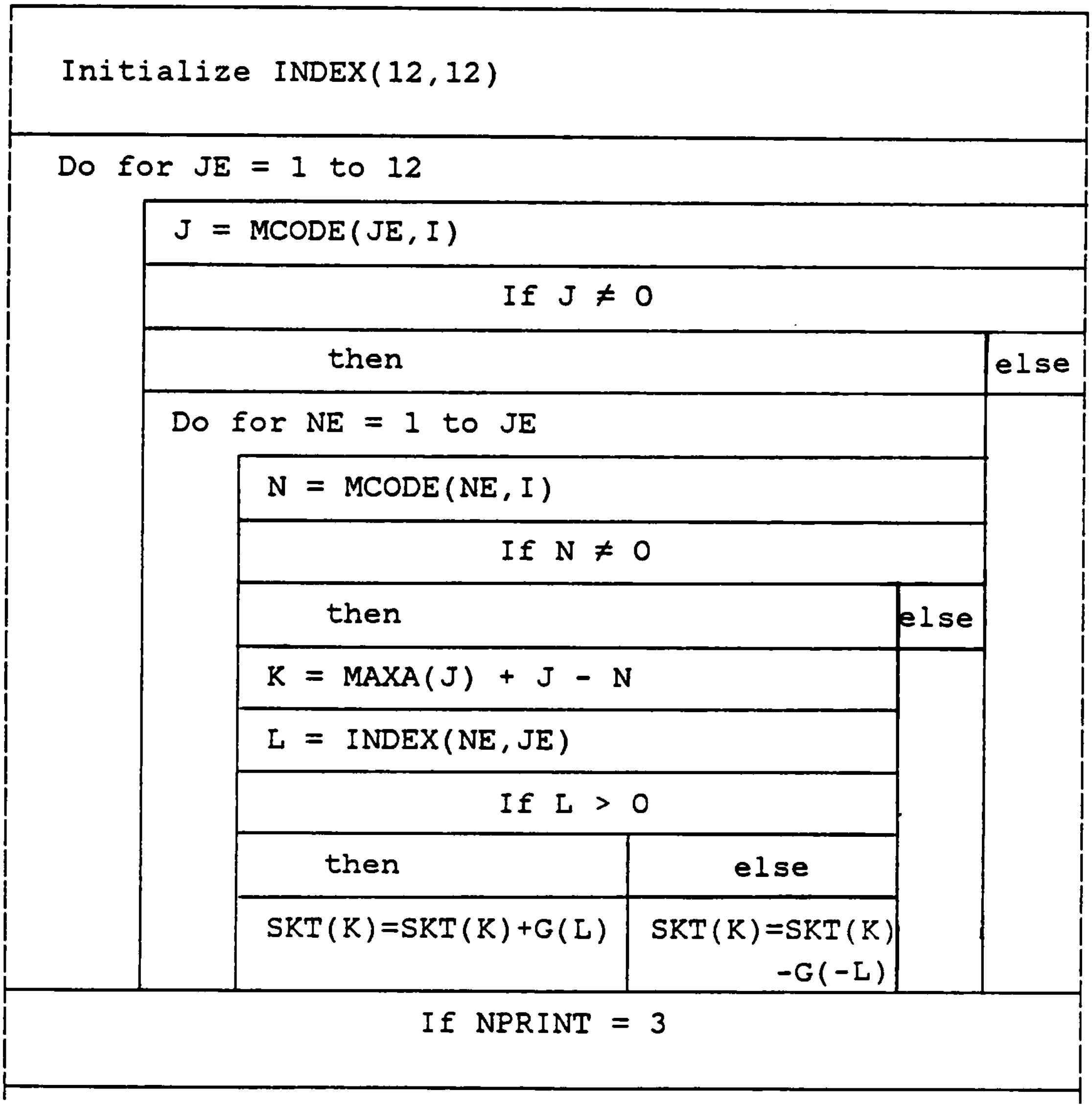
8.2.14 Subroutine ASSEMS

Function: Initialize INDEX; assign the global stiffness coefficients, G(L), of element to the system stiffness column matrix, SKT, by INDEX, MAXA.

Input arguments: G, I, MAXA, MCODE, NKT, NPRINT

Output arguments: SKT

N-S diagram:



then	else
Print SKT(KI) from element I, for KI = 1 to NKT	

8.2.15 <u>Subroutine</u> <u>STORE</u>

Function: Store the previous iteration length and direction cosines of element I.

Input arguments: NE, RL1, RL2, RL3, RL, C1, C2, C3

Output arguments: RL1PI, RL2PI, RL3PI, ELENG, C1PI, C2PI, C3PI

N-S diagram:

Do for I=1 to NE	
	RL1PI(I) = RL1(I)
	RL2PI(I) = RL2(I)
	RL3PI(I) = RL3(I)
	ELENG(I) = RL(I)
	C1PI(I) = C1(I)
	C2PI(I) = C2(I)
	C3PI(I) = C3(I)

8.2.16 Subroutine FORCES

Function: Generate the incremental joint displacement matrix DDJ; calculate the incremental nodal rotations at joint J and yield the components of the rotation matrix of joint J; for each iteration employing the last joint orientation matrix, AP, and rotation matrix of joint J, RJ, to generate the new joint orientation matrix; initialize the vector of the equivalent nodal element force, F, to zero; for each element call ELEMF.

Input arguments: AREA, CO1, CO2, CO3, C1PI, C2PI, C3PI, OL, DD, DDO, NN, ELENGO, ELENG, EMOD, GMOD, MCODE, ZI2, ZI3, ZJ, DQQR, FQR, IMAX, NE, NJ, NEQ, INDTAN, AP, NPRINT, OLAMD

Output arguments: BP12, BP22, BP13, BP23, C1, C2, C3, F, FG, FL, TR12, TR22, TR13, TR23, TTORS, SP12, SP22, SP13, SP23, ST12, ST22, ST13, ST23, RM32, RM33, QL, OLAMD

N-S diagram:

```
Do for J = 1 to NJ
    Do for IDIR = 1 to 6
        DDJ(IDIR,J) = 0.DO
        N = JCODE(IDIR,J)
```

If N ≠ 0

then

If NN=1 and ITIND=1

then

DDJ(IDIR,J) = DDO(N)

else

DDJ(IDIR,J) =DD(N)

else

Calculate the incremental nodal rotations at joint J, OME1, OME2, OME3, by Eq. (3.11)

Yield the incremental nodal rotation matrix at joint J, RJ, by Eq. (3.9)

Generate the current joint orientation matrix, AP, by Eq. (3.12)

Do for L = 1 to 3, M = 1 to 3

APNEW(L,M,J) = 0.DO

Do for K = 1 to 3

APNEW(L,M,J) = RJ(L,K)*AP(K,M,J)+APNEW(L,M,J)

Do for L = 1 to 3, K = 1 to 3

AP(L,K,J) = APNEW (L,K,J)

Do for I = 1 to NEQ

F(I) = 0.DO

I = 1

While I ≤ NE

Call ELEMF

```
I = I + 1
```

8.2.17 Subroutine ELEMF

Function: Employing the MCODE to get the global incremental displacements and direction cosines of element I; generate the increments of the local displacements and the components of the rotation matrix of element I; obtain the current element orientation matrix; employing the MINC to calculate the total relative deformations of element I; compute the local internal element forces by either beam-column model or finite element model, due to the deformations; transform them to the global element forces, and compute the equivalent nodal element forces.

Input arguments: AREA, C1PI, C2PI, C3PI, DD, DD0, NN, ELENG0, ELENG, EMOD, GMOD, MCODE, ZI2, ZI3, ZJ, DQQR, FQR, I, IMAX, INDTAN, OLAMD, CO1, CO2, CO3, OL, AP

Output arguments: BP12, BP22, BP13, BP23, C1, C2, C3, F, FL, FG, TR12, TR22, TR13, TR23, TTORS, SP12, SP22, SP13, SP23, ST12, ST22, ST13, ST23, RM32, RM33, QL, OLAMD

N-S diagram:

```
Do for J = 1 to 12
    DE(J) = 0.D0
```

<table>
<tr><td colspan="3">K = MCODE(J,I)</td></tr>
<tr><td colspan="3">If K ≠ 0</td></tr>
<tr><td colspan="2">then</td><td>else</td></tr>
<tr><td colspan="2">If NN=1 and ITIND=1</td><td rowspan="3"></td></tr>
<tr><td>then</td><td>else</td></tr>
<tr><td>DE(J)=DDO(K)</td><td>DE(J)=DD(K)</td></tr>
</table>

```
RL1PI(I) = ELENG(I)*C1PI(I)
RL2PI(I) = ELENG(I)*C2PI(I)
RL3PI(I) = ELENG(I)*C3PI(I)
RL1(I) = RL1PI(I)+DE(7)-DE(1)
RL2(I) = RL2PI(I)+DE(8)-DE(2)
RL3(I) = RL3PI(I)+DE(9)-DE(3)
RL(I) = DSQRT(RL1(I)**2+RL2(I)**2+RL3(I)**2)
C1(I)=RL1(I)/RL(I)
C2(I)=RL2(I)/RL(I)
C3(I)=RL3(I)/RL(I)
```

Calulate the local incremental element displacements by Eq. (2.56)

```
Yield the incremental element rotation matrix, R,
by Eqs. (3.23) to (3.25)

Generate the current element orientation matrix,
OLANEW, by Eq. (3.26)

Do for L=1 to 3, J=1 to 3
    OLANEW(L,J,I) = 0.DO
    Do for K=1 to 3
        OLANEW(L,J,I)=R(L,K,I)*OLAMD(K,J,I)
                          +OLANEW(L,J,I)

Do for L=1 to 3, J=1 to 3
        OLAMD(L,J,I)=OLANEW(L,J,I)

TUL = (ELENGO(I) - RL(I)) / ELENGO(I)

L = MINC(1,I)

K = MINC(2,I)

Compute the total relative element deformations,
TR12, TR13, TR22, TR23, TTORS

If INDTAN = 1
```

then	else
Call BOWCOR	Compute the RM12, RM22, RM13, RM23, FL(1,I) using the secant stiff-
Compute RM12, RM22, RM13, RM23, FL(1,I) by Eqs.	

(5.1),(5.2) and (5.4)	ness matrix by Eq. (4.20)

Compute the remaining local element forces
Transform the local element forces, FL, to the global element forces, FG, by Eq. (4.27)
Do for J = 1 to 12

	K = MCODE(J,I)	
	If K ≠ 0	
	then	else
	Compute the equivalent nodal element forces F(K)=F(K)+FG(J,I)	

8.2.18 <u>Subroutine</u> <u>BOWCOR</u>

Function: Calculate stability functions, ST12, ST22, ST13, ST23; bowing functions, BW12, BW22, BW13, BW23; length correction factors for the bowing actions, TCB2, TCB3; axial force, QPRINT.

Input arguments: AREA, ELENGO, EMOD, GMOD, ZI2, ZI3, ZJ, I, TUL,

Output arguments: BP12, BP22, BP13, BP23, TR12, TR22, TR13,

TR23, SP12, SP22, SP13, SP23, ST12, ST22,
ST13, ST23, TCB2, TCB3

N-S Diagram:

QR=TUL(I)*AREA(I)*ELENGO(I)**2/(Z12(I)*PI**2)

IC = 0

While IC < IMAX

If QR > 0.DO

then	else
Compute stability functions ST12, ST22, ST13, ST23, for compression force	

If QR = 0.DO

then	else
Compute stability functions ST12, ST22, ST13, ST23, for no axial force	
SP12(I)=SP22(I)=SP13(I)=SP23(I)=0.DO	
BP12(I)=BP22(I)=BP13(I)=BP23(I)=0.DO	
TCB2(I)=TCB3(I)=TUL(I)/2.DO	

If QR < 0.DO

then	else
Compute stability functions ST12, ST22, ST13, ST23, for tension force	

Compute BW12, BW22, BW13, BW23, SP12, SP22,
SP13, SP23, BP12, BP22, BP13, BP23, TCB2,

<table>
<tr><td rowspan="12"></td><td colspan="2">TCB3, TCB2P, TCB3P</td></tr>
<tr><td colspan="2">Compute FQRC and DQ (ref. [29])</td></tr>
<tr><td colspan="2">QR = QR + DQ</td></tr>
<tr><td colspan="2">DQQRC = DABS(DQ/QR)</td></tr>
<tr><td colspan="2">IC = IC + 1</td></tr>
<tr><td colspan="2">If DQ/QR ≤ DQQR and
FQRC ≤ FQR</td></tr>
<tr><td>then</td><td>else</td></tr>
<tr><td>IC = IMAX + 1</td><td></td></tr>
<tr><td colspan="2">If DQ/QR > DQQR or
FQRC > FQR</td></tr>
<tr><td>then</td><td>else</td></tr>
<tr><td>Stop</td><td></td></tr>
</table>

8.2.19 Subroutine RESULT

Function: Initialize the joint force matrix, P, to zero;
call JOINTF and OUTPUT.

Input arguments: NJ, NE, MINC, JCODE, FG, D, NPRINT,
QI, QT, NEQ, FL, IMP

Output arguments: none

N-S diagram:

Initialize P(I,J)=0.D0 for J=1 to NJ, I=1 to 6
Call JOINTF
Call OUTPUT

8.2.20 Subroutine JOINTF

Function: Assign the global element forces, FG, to the joint force matrix, P, via MINC.

Input arguments: NE, MINC, FG

Output arguments: P

N-S diagram:

```
Do for I = 1 to NE
    J = MINC(1,I)
    K = MINC(2,I)
    Do for L = 1 to 6
        P(L,J) = P(L,J)+FG(L,I)
        P(L,K) = P(L,K)+FG(L+6,I)
```

8.2.21 Subroutine OUTPUT

Function: Assign the generalized displacements, D, to the joint displacements, DJ, via JCODE; print the loading parameter, QI, generalized displacements, D; print the current generalized external force vector, QT; print the global joint displacements (including joint constraints), DJ; print the local element forces, FL; print the joint forces, P.

Input arguments: NJ, JCODE, D, NPRINT, QI, QT, NEQ, FL, NE, P, IMP

Output arguments: none

N-S diagram:

```
Do for J = 1 to NJ
    Do for I = 1 to 6
        DJ(I,J) = 0.D0
        K = JCODE(I,J)
        If K ≠ 0
            then                    | else
            DJ(I,J) = D(K)          |
If NPRINT ≤ 3
    then                            | else
    Print QI, D(IMP)                |
```

If NPRINT = 4	
then	else
Print QI	
Print QT(I) for I = 1 to NEQ	
Print DJ(I,J) for I=1 to 6, J=1 to NJ	
Print FL(I,J) for I=1 to 12, J=1 to NE	
Print P(I,J) for I=1 to 6, J=1 to NJ	

Chapter IX

CONCLUSION

9.1 CONCLUSION

Updated Lagrangian and a total Lagrangian formulations of a geometrically nonlinear three-dimensional beam element, undergoing large displacements with small strain, have been developed. In the T.L. formulation the incremental linear strains ${}_0e_{ij}$ (Eqs. (2.63) and (2.76)) contain an initial displacement effect that leads to a more complex strain-displacement matrix than in the U.L. formulation. The main difference between the U.L. and the T.L. formulations is that in the T.L. formulation the interpolation functions in Eq. (2.74) are obtained by referring the displacements to the initial configuration and the ${}^t_0B_{L1}$ matrix (Eq. 2.91c) is included in the calculations.

It was been shown in Chapter 2 that two formulations yield identical element stiffness matrices and vectors of nodal element forces. In fact, if the appropriate constitutive relations are employed, same results are obtained. In this study for three dimensional beam element with small deformations and uniaxial strain, the same Young's modulus used in the T.L. and U.L. formulations yields practically the same results.

The choice of using either the U.L. or the T.L. formulation depends on their numerical efficiency, i.e., economy. Which formulation is most effective depends on the program design and the actual structures.

The finite element model and the beam-column model have been compared for space frame problems. Both the linear and nonlinear strain incremental stiffness matrices must be considered in both models in order to predict the response accurately and to detect instability of large displacement problems. The finite element model is simple in theory, but quite a number of elements are often needed for satisfactory accuracy. The beam-column model uses the conventional beam-column theory to determine the member force-deformation relations so that fewer elements are often needed than in the case of a finite element model to give the same accuracy.

Because the stability functions c_{mn} in the beam-column model depend on whether the axial force is compressive, zero, or tensile, the axial force must be determined by an iterative procedure (i.e. Newton method). Consequently, more computational effort is required in the beam-column model than in the finite element model for the same mesh.

From the sample analyses (chapter 7) it is seen that both element models are reliable for studying the behavior of space frames. In conclusion the following observations are presented:

The only disadvantage of the finite element model is that a fine mesh is often needed for satisfactory accuracy. The main advantages of the finite element model are:

1. The formulations are simple without recourse to complex differential equations.
2. It can be extended to the elements undergoing large deformations in which the cross-sectional area would be updated.
3. The material properties can be adjusted at each element integration point. Combinations of material nonlinearity and geometric nonlinearity is particularly simple if the increment of the material properties may be obtained. The operations required in the solution of problems of material and geometric nonlinearity are similar.
4. The external loads can be configuration-dependent, i.e., nonconservative forces, like follower forces can be involved [48].

On the other hand, the main advantage of the beam-column model is that it yields higher accuracy than the finite element model for the same mesh. However, the disadvantages of the beam-column model are:

1. The formulations are complicated by the fact that the stability coefficients (appendix B) are functions of the axial force which must be obtained by an iterative procedure.
2. The basic element force-deformation relations (sec. 5.2) are derived from the conventional beam-column theory [23]; thus the relative deformations of the element are limited to be small.
3. Since the formulations are based on elastic behavior (sec. 5.2), extensions to material nonlinearity is impratical.
4. The external loads are assumed to be configuration-independent, i.e., conservative forces, and applied at the nodes only.

9.2 SUGGESTIONS FOR FUTURE DEVELOPMENT

Some interesting extensions of the present study are suggested:

1. Implement the T.L. formulation in the computer program using both models by transformation of the interpolation functions [9].
2. Use numerical integration, that is, Newton-Cotes or Gauss integration [38], to evaluate the vectors of nodal element forces.
3. Include the material nonlinearity in the finite element model.
4. The displacements in the U.L. formulation referred to the current coordinates at time t are usually small enough that K_{NL} can be omitted [33]. The effects of neglecting the nonlinear strain incremental stiffness matrix in the U.L. formulation can be studied.
5. Different convergence criteria will effect the accuracy and computer cost. The effects of using different convergence criteria in nonlinear analysis are of interest.
6. A comparative study of different numerical solution techniques in nonlinear analysis.

7. The final goal is to develop a nonlinear dynamic analysis program to analyze space frames under earthquake loads.

REFERENCES

1. Oran, C., "Tangent Stiffness in Plane Frames," Journal of the Structural Division, ASCE, vol. 99 (ST6), no. 4/6, 1973, pp. 973-985.

2. Oran, C., "Tangent Stiffness in Space Frames," Journal of the Structural Division, ACSE, vol. 99 (ST6), no. 4/6, 1973, pp. 987-1001.

3. Oran, C. and Kassimali, A., "Large Deformations of Framed Structures under Static and Dynamic Loads," Computers & Structures, vol. 6, 1976, pp. 539-547.

4. Mikkola, M. J. "Modelling of Geometrically Nonlinear Problems of Framed Structures," Notes, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, August, 1981.

5. Belytschlo, T. and Hsieh, B. J., "Nonlinear Transient Finite Element Analysis with Convected Co-ordinates," International Journal for Numerical Methods in Engineering, vol. 7, 1973, pp. 255-271.

6. Wood, R. D. and Schrefler, B., "Geometrically Nonlinear Analysis - A Correlation of Finite Element Notations," International Journal for Numerical Methods in Engineering, vol. 12, 1978, pp. 635-642.

7. Rajasekaran, S. and Murray, D. W., "Incremental Finite Element Matrices," Journal of the Structural Division, ASCE, vol. 99 (ST12), December 1973, pp. 2423-2438.

8. Mallett, R. H. and Marcal, P. V., "Finite Element Analysis of Nonlinear Structures," Journal of the Structural Division, ASCE, vol. 94 (ST9), September 1968, pp. 2081-2105.

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

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

11. Carey, G. F., "A Unified Approach to Three Finite Element Theories for Geometric Nonlinearity," Computer Methods in Applied Mechanics and Engineering, vol. 4, 1974, pp. 69-79.

12. Hibbitt, H. D., Marcal, P. V. and Rice, J. R., "A Finite Element Formulation for Problems of Large Strain and Large Displacement," Int. J. of Solids And Structures, vol. 6, 1970, pp. 1069-1086.

13. Holden, J. T., "On the Finite Deflections of Thin Beams," Int. J. of Solids and Structures, vol. 8, 1972, pp. 1051-1055.

14. Felippa, C. A., "Procedures for Computer Analysis of Large Nonlinear Structural Systems," Large Engineering Systems, Proceedings of the International Symposium Held at the University of Manitoba, Winnipeg, Manitoba, Canada, August 1976, pp. 60-100.

15. Becker, Eric B., Carey, Graham F. and Oden, Tinsley J., "Finite Elements, An Introduction," vol. 1, pp. 87-131.

16. Connor, J. J., Logcher, R. D. and Chan, S. C., "Nonlinear Analysis of Elastic Framed Structures," Journal of the Structural Division, ASCE, vol. 94 (ST6), June 1968, pp. 1525-1545.

17. Belytschko, T. and Schwer, L., "Large Displacement, Transient Analysis of Space Frames," International Journal for Numerical Methods in Engineering, vol. 11, 1977, pp. 65-84.

18. Tang, S. C., Yeung, K. S. and Chon, C. T., "On the Tangent Stiffness Matrix in a Convected Coordinates System," Computers & Structures, vol. 12, 1980, pp. 849-856.

19. Remseth, S. N., "Nonlinear Static and Dynamic Analysis of Framed Structures," Computers & Structures, vol. 10, pp. 879-897.

20. Wood, R. D. and Zienkiewicz, O. C., "Geometrically Nonlinear Finite Element Analysis of Beams, Frames, Arches and Axisymmetric Shells," Computers & Structures, vol. 7, 1977, pp. 725-735.

21. Papadrakakis, M., "Post-Buckling Analysis of Spatial Structures by Vector Iteration Methods," Computers & Structures, vol. 14, 1981, pp. 393-402.

22. Felippa, C. A. and Sharifi, P., "Numerical Solution of Nonlinear Structureal Problems," the Winter Annual Meeting of the American Society of Mechanical Engineers, AMD-vol. 6, November 11-15, 1973, pp. 31-49.

23. Timoshenko, S. P. and Gere, J. M., Theory of Elastic Stability, 2nd edition, Chapter One, McGraw-Hill, New York, 1961.

24. Williams, F. W., "An Approach to the Non-Linear Behavior of the Members of a Rigid Jointed Plane Framework with Finite Deflections," Quarterly Journal of Mechanics and Applied Mathematics, vol. 17, Part 4, 1964, pp. 451-469.

25. Murray, D. W. and Wilson, E. L., "Finite-Element Postbuckling Analysis of Thin Elastic Plates," AIAA Journal, vol. 7, no. 10, October 1969, pp. 1915-1920.

26. Yang, T. Y., "Matrix Displacement Solution to Elastic Problems of Beams and Frames," International Journal of Solids and Structures, vol. 9, 1973, pp. 829-842.

27. Holzer, S. M., Watson, L. T. and Vu, P. D., "Stability Analysis of Lamella Domes," Proceedings of ASCE 1981 Annal Convention and Exhibit, St. Louis, Missouri, October 26-30, 1981, pp. 179-209.

28. Holzer, S. M., Computer Analysis of Structures, Elsevier.

29. Butler, M. J., "A Comparison of Two Models for Geometrically Nonlinear Finite Element Analysis of Plane Frames," M.S. Thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, December 1983.

30. Bathe, K. J. and Ozdemir, Haluk, "Elastic-Plastic Large Deformation Static and Dynamic Analysis," Computers & Structures, vol. 6, Feb-June, 1976, pp. 81-92.

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

32. Remseth, S. N., "Nonlinear Static and Dynamic Analysis of Space Structures," Ph.D Dissertation, Division of Structural Mechanics, Norwegian Institute of Technology, April 1978.

33. Cook, R. D., Concepts and Applications of Finite Element Analysis, 2nd edition, John-Wiley] Sons, London, 1981.

34. Desai, C. S., Elementary Finite Element Method, Prentice-Hall Inc., N.J., 1979.

35. Young, J. W., "A Computer Simulation of Nonlinear Transient Response of Structures," Automobile Manufacturers' Association and the National Highway Traffic Safety Administration, DOT-HS-091-1-125-B, March 1972.

36. Glowinski, R., Rodin, E. Y. and Zienkiewicz, O. C., Energy Methods in Finite Element Analysis, 1979.

37. Fung, Y. C., Foundations of Solid Mechanics, Prentice-Hall Inc., N.J., 1965.

38. Bathe, K. J., Finite Element Procedures in Engineering Analysis, Prentice-Hall Inc., N.J., 1982.

39. Chu, Kuang-Han and Rampetsreiter, R. H., "Large Deflection Buckling of Space Frames," Journal of the Structural Division, ASCE, vol. 98 (ST12), December 1972, pp. 2701-2722.

40. Dickie, J. F. and Broughton, "Geometric Nonlinearity," Journal of Structural Mechanics, 1(2), 1972, pp. 249-266.

41. Murray, D. W. and Wilson, E. L., "Finite-Element Large Deflection Analysis of Plates," Engineering Mechanics Division, ASCE, vol. 95, (EM1), 1969, pp. 143-165.

42. Yaghmai, S. and Popov, E. P., "Incremental Analysis of Large Deflections of Shells of Revolution, " International Journal of Solids and Structures, vol. 7, no. 10, 1971, pp. 1375-1393.

43. Ramm, E., "Strategies for Tracing Nonlinear Response Near Limit Points," Europe-U.S.-Workshop: Nonlinear Finite Element Analysis in Structural Mechanics, July 28-31, 1980, Bochum, G. F. R..

44. Gallagher, R. H., Yamada, Y. and Oden, T., Recent Advances in Matrix Methods of Structural Analysis and Design, The University of Alabama in Huntsville, Alabama, August 1969.

45. Riks, E., "The Application of Newton's Method to the Problem of Elastic Stability, " Journal of Applied Mechanics, vol. 39, 1972, pp. 1060-1066.

46. Wempner, G. A., "Discrete Approximatons Related to Nonlinear Theories of Solids," International Journal of Solids And Structures, vol. 7, 1971, pp. 1581-1599.

47. Crisfield, M. A., "A Fast Incremental/Iterative Solution Procedure That Handles Snap-Through," Computers & Structures, vol. 13, 1981, pp. 55-62.

48. Katzenberger, G. S.,"Geometrically Nonlinear Analysis of Pressure-Loaded Arches, Rings, and Frames Using a Follower Force Algorithm", M. S. Thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, August, 1983.

49. Bathe, K. J. and Cimento, A. P., "Some Practical Procedures for the Solution of Nonlinear Finite Element Equations," Journal of Computer Methods in Applied Mechanics and Engineering, vol. 22, 1980, pp. 59-85.

50. Bergan, P. G. and Soreide, T. H., "Solution of Large Displacement and Instability Problems Using the Current Stiffness Parameter," Finite Elements in Nonlinear Mechanics, vol. 2, 1977, pp. 647-669.

51. Holzer, S. M., Plaut, R. H., Somers, A. E. and White, W. S., "Stability of Lattice Structures under Combined Loads," Journal of the Engineering Mechanics Division, ASCE, vol. 106, No. EM2, April, 1980, pp. 289-305.

52. Hangai, Y. and Kawamata, S., "Nonlinear Analysis of Space Frames and Snap-Through Buckling of Reticulated Shell Structures," IASS Pacific Symposium Part II on Tension Structures and Space Frames, Tokyo and Kyoto, Paper No. 9-4, 1972, pp. 803-816.

53. Paradiso, M. and Tempesta, G., "Member Buckling Effects in Non-linear Analysis of Space Trusses," Numerical Methods for Non-linear Problems, vol. 1, September, 1980, pp. 395-405.

54. The Aluminum Association, "Specifications for Aluminum Structures," Construction Manual Series, Section 1, New York, N.Y., April,1982.

55. Uliana, D., "The Effects of Earthquake Excitations on Reticulated Domes," M. S. Thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, March,1985.

Appendix A

JOINT ORIENTATION MATRIX α_j FOR SMALL JOINT ROTATIONS

In this appendix Eq. (3.9) will be derived for small joint rotation. A small change in the orientation of the joint can be represented by rotation components ρ_1, ρ_2, ρ_3, in the global coordinates X_1, X_2, X_3 (Fig. A.1). Let

$$\rho = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \end{bmatrix} \tag{A.1}$$

To derive Eq. (3.9) we can formulate the joint orientation matrix in three steps:

First, consider Fig. A.2 which shows a small joint rotation ρ_2 about X_2-axis, similar to Eq. (3.1)

$$b = \alpha_{j2}\, X \tag{A.2}$$

where

$$\alpha_{j2} = \begin{bmatrix} \cos \rho_2 & \cos 90^\circ & \cos(90^\circ+\rho_2) \\ \cos 90^\circ & \cos 0^\circ & \cos 90^\circ \\ \cos(90^\circ-\rho_2) & \cos 90^\circ & \cos \rho_2 \end{bmatrix} \tag{A.3}$$

For infinitesimal ρ_2:

$$\rho_2 \approx 0 , \quad \cos \rho_2 \approx 1 , \ \sin \rho_2 \approx \rho_2 \tag{A.4}$$

Substituting Eq. (A.4) into (A.3), we have

$$\alpha_{j2} = \begin{bmatrix} 1 & 0 & -\rho_2 \\ 0 & 1 & 0 \\ \rho_2 & 0 & 1 \end{bmatrix} \tag{A.5}$$

Second, we rotate a small joint rotation ρ_3 about b_3-axis (Fig. A.3). In a similar manner we have

$$a = \alpha_{j3}\ b \tag{A.6}$$

where

$$\alpha_{j3} = \begin{bmatrix} 1 & \rho_3 & 0 \\ -\rho_3 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.7}$$

Third, we rotate a small joint rotation ρ_1 about a_1-axis (Fig. A.4). Similarly

$$y = \alpha_{j1}\ a \tag{A.8}$$

where

$$\alpha_{j1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \rho_1 \\ 0 & -\rho_1 & 1 \end{bmatrix} \tag{A.9}$$

Combining Eqs. (A.2),(A.6) and (A.8), we obtain

$$y = \alpha_j X \tag{A.10}$$

where

$$\alpha_j = \alpha_{j1}\,\alpha_{j3}\,\alpha_{j2} = \begin{bmatrix} 1 & \rho_3 & -\rho_2 \\ -\rho_3+\rho_1\rho_2 & 1 & \rho_1+\rho_2\rho_3 \\ \rho_2+\rho_1\rho_3 & -\rho_1 & 1-\rho_1\rho_2\rho_3 \end{bmatrix} \tag{A.11}$$

Neglecting the higher order terms, we obtain Eq. (3.9)

$$\alpha_j = \begin{bmatrix} 1 & \rho_3 & -\rho_2 \\ -\rho_3 & 1 & \rho_1 \\ \rho_2 & -\rho_1 & 1 \end{bmatrix} \tag{A.12}$$

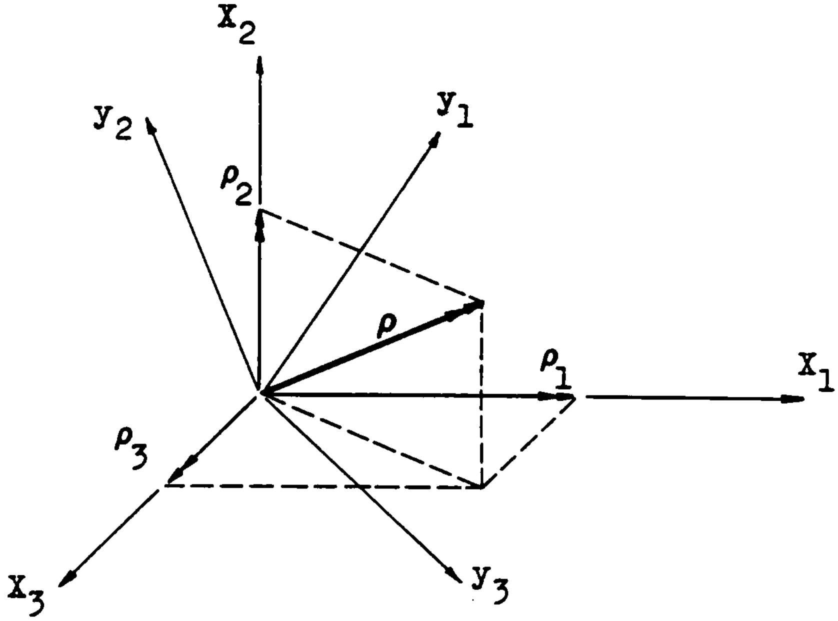


Fig. A.1 Small Joint Rotation

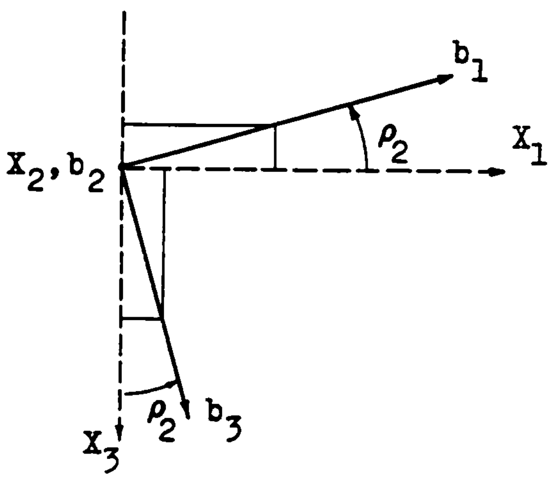


Fig. A.2 Small joint rotation ρ_2 about X_2-axis

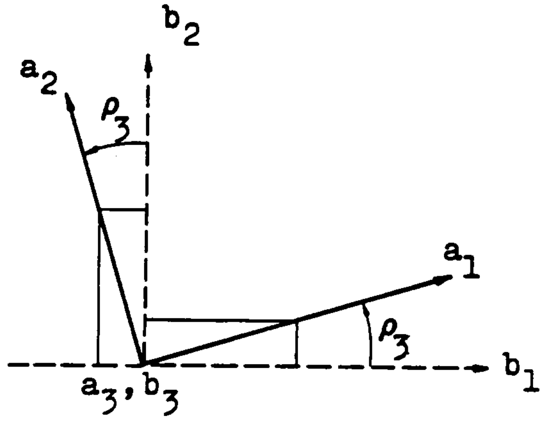


Fig. A.3 Small Joint rotation ρ_3 about b_3-axis

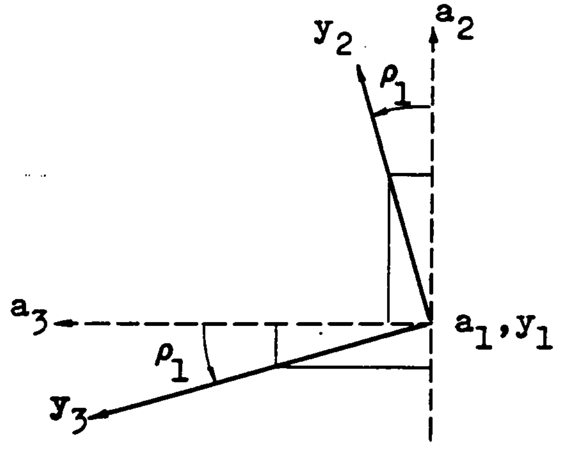


Fig. A.4 Small joint rotation ρ_1 about a_1-axis

Appendix B

STABILITY FUNCTIONS

1. Compression ($p_n > 0$)

$$c_{1n} = \frac{(k_n {}^tL)[\sin(k_n {}^tL) - (k_n {}^tL)\cos(k_n {}^tL)]}{2[1-\cos(k_n {}^tL)] - (k_n {}^tL)\sin(k_n {}^tL)} \tag{B.1}$$

$$c_{2n} = \frac{(k_n {}^tL)[(k_n {}^tL) - \sin(k_n {}^tL)]}{2[1-\cos(k_n {}^tL)] - (k_n {}^tL)\sin(k_n {}^tL)} \tag{B.2}$$

with

$$k_n = (P/EI_n)^{1/2} \tag{B.3}$$

2. No Axial Force ($p_n = 0$)

$$c_{1n} = 4 \tag{B.4}$$

$$c_{2n} = 2 \tag{B.5}$$

3. Tension ($p_n < 0$)

$$c_{1n} = \frac{k_n {}^tL[(k_n {}^tL)\cosh(k_n {}^tL) - \sinh(k_n {}^tL)]}{2[(1-\cosh(k_n {}^tL)] + (k_n {}^tL)\sinh(k_n {}^tL)} \tag{B.6}$$

$$c_{2n} = \frac{(k_n {}^tL)[\sinh(k_n {}^tL) - (k_n {}^tL)]}{2[(1-\cosh(k_n {}^tL)] + (k_n {}^tL)\sinh(k_n {}^tL)} \tag{B.7}$$

with

$$k_n = (-P/EI_n)^{1/2} \tag{B.8}$$

Appendix C

GEOMETRIC MATRICES $\bar{g}^{(i)}$

The geometric matrices $\bar{g}^{(i)}$ can be defined as [2], [29]

$$\Delta C^{T^{(i)}} = \bar{g}^{(i)} \, \Delta d \qquad \text{for } i = 1 \text{ to } 6 \tag{C.1}$$

in which

$$\Delta C^{T^{(i)}} = \text{the } i^{th} \text{ column of } \Delta C^T$$

such that

$$\bar{g}^{(1)} = \bar{g}^{(2)} = 1/L^2 \begin{bmatrix} \delta^{(1)} & -\delta^{(1)} \\ -\delta^{(1)} & \delta^{(1)} \end{bmatrix}$$

$$\bar{g}^{(3)} = \bar{g}^{(4)} = 1/L^2 \begin{bmatrix} \delta^{(3)} & -\delta^{(3)} \\ -\delta^{(3)} & \delta^{(3)} \end{bmatrix} \tag{C.2}$$

$$\bar{g}^{(5)} = 0$$

$$\bar{g}^{(6)} = 1/L^2 \begin{bmatrix} \delta^{(6)} & -\delta^{(6)} \\ -\delta^{(6)} & \delta^{(6)} \end{bmatrix}$$

in which the size of matrices $\delta^{(1)}$, $\delta^{(3)}$, $\delta^{(6)}$ are 6x6, the values of their element are defined as:

$$\delta_{12}^{(1)} = \delta_{21}^{(1)} = 1 \text{ , other } \delta_{ij}^{(1)} = 0 \tag{C.3}$$

$$\delta_{13}^{(3)} = \delta_{31}^{(3)} = -1, \text{ other } \delta_{ij}^{(3)} = 0 \tag{C.4}$$

$$\delta_{22}^{(6)} = \delta_{33}^{(6)} = -1, \text{ other } \delta_{ij}^{(6)} = 0 \tag{C.5}$$

Appendix D

$$g^{(i)} = \Lambda^T \bar{g}^{(i)} \Lambda$$

Sustituting Eqs. (2.54) and appendix C into Eq. (5.40), one obtains $g^{(i)}$, corresponding to the coordinate system in Fig. 2.4:

$$g^{(1)} = g^{(2)} = \frac{1}{L^2} \left[\begin{array}{c|c} a^{(1)} & -a^{(1)} \\ \hline -a^{(1)} & a^{(1)} \end{array} \right] \tag{D.1}$$

where

$$a^{(1)} = \left[\begin{array}{ccc|c} 2c_1\lambda_{21} & c_2\lambda_{21}+c_1\lambda_{22} & c_3\lambda_{21}+c_1\lambda_{23} & \\ c_1\lambda_{22}+c_2\lambda_{21} & 2c_2\lambda_{22} & c_3\lambda_{22}+c_2\lambda_{23} & \underset{\sim}{0} \\ c_1\lambda_{23}+c_3\lambda_{21} & c_2\lambda_{23}+c_3\lambda_{22} & 2c_3\lambda_{23} & \\ \hline & \underset{\sim}{0} & & \underset{\sim}{0} \end{array} \right] \tag{D.2}$$

$\underset{\sim}{0}$ is a null matrix.

$$g^{(3)} = g^{(4)} = \frac{1}{L^2} \left[\begin{array}{c|c} b^{(3)} & -b^{(3)} \\ \hline -b^{(3)} & b^{(3)} \end{array} \right] \tag{D.3}$$

where

$$
b^{(3)} = \left[\begin{array}{ccc:c}
-2c_1\lambda_{31} & -c_2\lambda_{31} - c_1\lambda_{32} & -c_3\lambda_{31} - c_1\lambda_{33} & \\
-c_1\lambda_{32} - c_2\lambda_{31} & -2c_2\lambda_{32} & -c_3\lambda_{32} - c_2\lambda_{33} & \underset{\sim}{0} \\
-c_1\lambda_{33} - c_3\lambda_{31} & -c_2\lambda_{33} - c_3\lambda_{32} & -2c_3\lambda_{33} & \\
\hdashline
 & \underset{\sim}{0} & & \underset{\sim}{0}
\end{array}\right]
\tag{D.4}
$$

$$
g^{(5)} = 0 \tag{D.5}
$$

$$
g^{(6)} = \frac{1}{L^2}\left[\begin{array}{c:c}
i^{(6)} & -i^{(6)} \\
\hdashline
-i^{(6)} & i^{(6)}
\end{array}\right]
\tag{D.6}
$$

where

$$
i^{(6)} = \left[\begin{array}{ccc:c}
-\lambda_{21}^2 - \lambda_{31}^2 & -\lambda_{21}\lambda_{22} - \lambda_{31}\lambda_{32} & -\lambda_{21}\lambda_{23} - \lambda_{31}\lambda_{33} & \\
-\lambda_{22}\lambda_{21} - \lambda_{32}\lambda_{31} & -\lambda_{22}^2 - \lambda_{32}^2 & -\lambda_{22}\lambda_{23} - \lambda_{32}\lambda_{33} & \underset{\sim}{0} \\
-\lambda_{23}\lambda_{21} - \lambda_{33}\lambda_{31} & -\lambda_{23}\lambda_{22} - \lambda_{33}\lambda_{32} & -\lambda_{23}^2 - \lambda_{33}^2 & \\
\hdashline
 & \underset{\sim}{0} & & \underset{\sim}{0}
\end{array}\right]
\tag{D.7}
$$

Appendix E

NOTATION

A = area of cross section

${}^{t}_{0}B_L$, ${}^{t}_{t}B_L$ =

linear strain-displacement matrix at time t referred to configuration at time 0, t

${}^{t}_{0}B_{NL}$, ${}^{t}_{t}B_{NL}$ =

nonlinear strain-displacement matrix at time t referred to configuration at time 0, t

b_{mn} = bowing functions

$= (c_{1n}+c_{2n})(c_{2n}-2) / 8\pi^2 q_n$ for m = 1

$= c_{2n} / 8(c_{1n}+c_{2n})$ for m = 2

C = local transformation matrix defined in Eq. (5.22)

${}_{0}C_{ijrs}$, ${}_{t}C_{ijrs}$ =

component of incremental material property tensor at time t referred to configuration at time 0, t

${}^{0}C$, ${}^{t}C$, ${}^{t+\Delta t}C$ = element configuration at time 0, t, t+Δt

CPDB = displacement convergence parameter adopted by Bathe

CPDC = displacement convergence parameter adopoted by Cook

CPF = convergence parameter for the unbalanced force criterion

c_{bn} = length correction factors for bending action

$= b_{1n} (e_{1n}+e_{2n})^2 + b_{2n}(e_{1n}-e_{2n})^2$

${}^{0}c_i$ =

direction cosines of the local x_i-axis at time 0 with respect to the global coordinate system X_i-axis; i=1,2,3

c_{mn} =

stability functions corresponing to e_{mn} defined in appendix B [1], [4]

D_{jk} =

component of node displacement at node j in the global coordinate system, k=1 to 6

${}^{t}_{0}D$, ${}^{t}_{t}D$, ${}^{t+\Delta t}_{t+\Delta t}D$ =

vector of nodal element displacements at time t, t, t+Δt referred to configuration at time 0, t, t+Δt in global coordinate system

ΔD_{ji} =

incremental component of nodal displacement at node j in the global coordinate system; i=1 to 6

$\Delta^{0}D$, $\Delta^{t}D$ =

incremental nodal displacement vector of element from t to t+Δt measured in global axis ${}^{0}X_i$, ${}^{t}X_i$

${}^{t}_{0}d$, ${}^{t}_{t}d$, ${}^{t+\Delta t}_{t+\Delta t}d$ =

vector of nodal element displacements at time t, t, t+Δt referred to configuration at time 0, t, t+Δt in local coordinate system

$\Delta^{0}d$, $\Delta^{t}d$ =

incremental nodal displacement vector of element from t to t+Δt measured in local axis ${}^{0}x_i$, ${}^{t}x_i$

$\Delta{}^{0}d_k$, $\Delta{}^{t}d_k$ =

increment in nodal displacement component of element from t to t+Δt measured in local axis ${}^{0}x_i$, ${}^{t}x_i$, k=1 to 12

E = Young's modulus of elasticity

e = vector of deformation displacements in space

e_1 = relative axial elongation of element

e_1, e_2, e_3 =

relative deformation displacements of element for plane frame

e_{mn} =

member relative end rotations: the first subscript m=1,2 referred to the nodal point (1 for left and 2 for right); the second subscript n=2,3 indicates the member axis, x_n, about which the rotation takes place (Fig. 3.3)

e', e" =

vector of deformation displacements in x_1-x_2 and x_1-x_3 plane

${}_{0}e$, ${}_{t}e$ =

linear part of incremental strain vector (Green-Lagrange) ${}_{0}\varepsilon$, ${}_{t}\varepsilon$

${}_{0}e_{ij}$, ${}_{t}e_{ij}$ = linear part of strain increment ${}_{0}\varepsilon_{ij}$, ${}_{t}\varepsilon_{ij}$

${}^{t}_{0}F^{i}$, ${}^{t}_{t}F^{i}$ =

vector of global end forces for element i at time t referred to configuration at time 0, t

${}^{t+\Delta t}_{0}F^{k-1}$, ${}^{t+\Delta t}_{t+\Delta t}F^{k-1}$ =

vector of nodal equivalent element force at time t+Δt referred to configuration at time 0, t+Δt, during (k-1)th iteration (see Figs. 6.2 and 6.4)

$\Delta{}^{0}F^{i}$, $\Delta{}^{t}F^{i}$ =

vector of global incremental end forces for elment i from time t to t+Δt referred to configuration at time 0, t

f_{bc} =

vector of relative local element forces in beam-column model

f_i, F_i = component of local and global nodal element force

${}^{t}_{0}f$, ${}^{t}_{t}f$, ${}^{t+\Delta t}_{t+\Delta t}f$ =

vector of local nodal element forces at time t, t, t+Δt referred to configuration at time 0, t, t+Δt

${}^{t+\Delta t}_{0}f^{k-1}$, ${}^{t+\Delta t}_{t+\Delta t}f^{k-1}$ =

vector of nodal equivalent element forces corresponding to ${}^{t+\Delta t}_{0}d^{k-1}$, ${}^{t+\Delta t}_{t+\Delta t}d^{k-1}$

$\Delta{}^{0}f$, $\Delta{}^{t}f$ =

vector of local incremental element forces from time t to t+Δt referred to configuration at time 0, t

G = shear modulus of elasticity

GJ = torsional rigidity

g_i =

coefficient function defined in Eqs. (4.11) and (5.42), $i = 1$ to 42

$\mathbf{g}^{(i)} = \Lambda^T \, g^{(i)} \, \Lambda$

g_i = coefficient functions defined in Eq. (5.31)

$g^{(i)}$ = geometric matrix defined in Eq. (D.2)

H = symmetric matrix containing entries of 1 and 0

h_k =

interpolation function corresponding to nodal direction k, $k = 1$ to 12.

${}_0h_{i,j}$, ${}_th_{i,j}$ =

derivative of component of interpolation function h_i with respect to coordinate ${}^0x_j$, ${}^tx_j$

${}_0h_{i,jj}$, ${}_th_{i,jj}$ =

2nd order derivative of component of interpolation function h_i with respect to coordinate ${}^0x_j$, ${}^tx_j$

I = identity matrix or reference moment of inertia

I_n = moment of inertia about the x_n-axis

INDEX =

location matrix of the components in the stiffness matrix, which stores the subscripts and negative signs of these components [28]

J = polar moment of inertia (torsional stiffness)

${}^t_0K$, ${}^t_tK$ =

structural strain incremental stiffness matrix at time t referred to configuration at time 0, t

${}_{0}^{t}K^{i}$, ${}_{t}^{t}K^{i}$ =

strain incremental stiffness matrix at time t referred to configuration at time 0, t for element i in global coordinate system

${K_q^{(i)}}^{k-1}$ =

generalized tangent stiffness matrix of element i corresponding to q^{k-1}

${}_{0}^{t}k$, ${}_{t}^{t}k$ =

element strain incremental stiffness matrix at time t referred to configuration at time 0, t, in local coordinate system

${}_{0}^{t}k_{L}$, ${}_{t}^{t}k_{L}$ =

linear strain incremental stiffness matrix at time t referred to configuration at time 0, t, in local coordinate system

${}_{0}^{t}k_{NL}$, ${}_{t}^{t}k_{NL}$ =

nonlinear strain incremental stiffness matrix at time t referred to cofiguration at time 0, t, in local coordinate system

$\hat{k}$ =

local tangent stiffness matrix of element for relative deformations in space

$\hat{k}_s$ =

local secant stiffness matrix of element for relative deformations in space

$\hat{k}_s'$, $\hat{k}_s''$ =

local secant stiffness matrix of element for relative deformations in x_1-x_2, x_1-x_3 plane

L = undeformed element length

L' = new deformed element length

${}_0L$, ${}_tL$ =

column vector defining linear strain ${}_oe$, ${}_te$ from displacement gradients

${}^0L$, ${}^{t+\Delta t}L$ = element length at time 0, t+Δt

${}^0\ell = ({}^0c_1^2 + {}^0c_3^2)^{1/2}$

M_{mn} = bending moment corresponding to e_{mn}

M_t = torque

P = axial compression force

P_{En} = classical Euler buckling load = $\pi^2 EI_n / L^2$

p =

vector of deformation forces in space defined in Eq. (4.21)

p_n = dimensionless axial force parameter = P / P_{En}

${}^{t+\Delta t}p_j$ = end section matrix at node j at time t+Δt

p', p" =

vector of deformation forces in x_1-x_2 and x_1-x_3 plane

${}^{t+\Delta t}Q$ =

vector of given new load level in configuration at time $t+\Delta t$

q^{k-1} = vector of nodal displacements at k-1 iteration

Δq^{k} =

vector of incremental nodal displacements at k^{th} iteration in configuration at time t

R =

transformation matrix from local coordinates ${}^{0}x_{i}$ to ${}^{t}x_{i}$ in space

R^{k-1} = unbalanced force vector corresponding to q^{k-1}

${}^{t}R$ = rotation matrix defined in Eq. (3.23)

${}^{t}R_{j}$ = rotation matrix at node j at time t

${}^{t+\Delta t}R$ = external virtual work at time $t+\Delta t$

${}^{t+\Delta t}r$ =

vector of externally applied element nodal loads at time $t+\Delta t$ in local cordinate system

${}^{t+\Delta t}_{t}S_{ij}$ =

component of 2nd Piola-Kirchhoff stress tensor at time $t+\Delta t$ referred to configuration at time t

${}^{t}_{0}S_{ij}$, ${}^{t+\Delta t}_{0}S_{ij}$ =

component of 2nd Piola-Kirchhoff stress tensor at time t, $t+\Delta t$ referred to configuration at time 0

${}_{0}S_{ij}$, ${}_{t}S_{ij}$ =

component of 2nd Piola-Kirchhoff stress increment at time t

T = axial force (positive in tension)

T_i =

entry of local strain incremental stiffness matrix defined in Eq. (4.8), i=1 to 10

ΔU_j =

increment vector of node displacements at node j in the global coordinate system

ΔU_{jr} =

rotational increment vector of node displacements at node j in the global coordinate system

ΔU_{jt} =

translational increment vector of node displacements at node j in the global coordinate system

u = relative axial displacement

u_i = increment in displacement component, $u_i = {}^{t+\Delta t}u_i - {}^{t}u_i$

${}^{t}_{0}u_{i,j}$ =

derivative of displacement component at time t with respect to coordinate ${}^{0}x_j$

${}^{t+\Delta t}u$ =

displacement vector in each element at time t+Δt measured from initial position at time 0

${}_{0}u_i$, ${}_{t}u_i$ =

increment in displacement component of element from t to t+Δt measured in local coordinate system at time 0, t

${}^{t}u_i$, ${}^{t+\Delta t}u_i$ =

displacement component from initial position at time 0 to configuration at time t, t+Δt

${}_0u_{i,j}$, ${}_tu_{i,j}$ =

derivative of displacement increment with respect to coordinate ${}^0x_j$, ${}^tx_j$

${}_0u_{i,jj}$, ${}_tu_{i,jj}$ =

2nd order derivative of displacement increment with respect to coordinate ${}^0x_j$, ${}^tx_j$

${}^0V$, ${}^tV$, ${}^{t+\Delta t}V$ = volume at time 0, t, t+Δt

${}^0X_i$, ${}^tX_i$, ${}^{t+\Delta t}X_i$ =

global coordinate system at 0, t, t+Δt respectively

x_i =

end cross section (body) coordinate system of element associated with each node, i=1,2,3

${}^0x_i$, ${}^tx_i$, ${}^{t+\Delta t}x_i$ =

local coordinate system at time 0, t, t+Δt respectively

y_i =

node coordinate system of element which is rigidly connected to each node, i=1,2,3

$\Delta^t y_j$ =

increment vector of node coordinate vector of node j at time t

y, z =

distance from centroid axis in local x_2, x_3-axes direction

$\alpha = EI / L$

${}^{t}\alpha_j$, ${}^{t+\Delta t}\alpha_j$ =

orthogonal node orientation matrix at node j at time t, t+Δt

$\Delta^{t}\alpha_j$ =

incremental node orientation matrix of node j at time t

δ = 'variation in' or length correction factor

${}_0\varepsilon$, ${}_t\varepsilon$ =

incremental Green-Lagrange strain vector referred to configuration at time 0, t

${}^{t+\Delta t}_{t}\varepsilon_{ij}$ =

component of Green-Lagrange strain tensor at time t+Δt referred to configuration at time t (i.e. using displacements from the configuration at time t to the configuration at time t+Δt)

${}^{t}_{0}\varepsilon_{ij}$, ${}^{t+\Delta t}_{0}\varepsilon_{ij}$ =

component of Green-Lagrange strain tensor at time t+Δt, t, referred to configuration at time 0.

${}_0\varepsilon_{ij}$, ${}_t\varepsilon_{ij}$ =

component of strain increment tensor (Green-Lagrange) referred to configuration at time 0, t

η = GJ / EI

${}_0\eta$, ${}_t\eta$ =

nonlinear part of incremental strain vector (Green-Lagrange) ${}_0\varepsilon$, ${}_t\varepsilon$

${}_0\eta_{ij}$, ${}_t\eta_{ij}$ =

nonlinear part of strain increment ${}_0\varepsilon_{ij}$, ${}_t\varepsilon_{ij}$

γ = rigid body rotation of element for plane frame

${}^t_0\Lambda$, ${}^t_t\Lambda$, ${}^{t+\Delta t}_{t+\Delta t}\Lambda$ =

coordinate transformation matrix at time 0, t, t+Δt defined in Eqs. (2.54) and (2.107)

λ_{ij} = component of element orientation matrix

${}^t_0\lambda$, ${}^t_t\lambda$ =

element orientation matrix at time 0, t defined in Eqs. (2.109) and (3.17)

$\Delta^t\lambda$ =

change of element orientation matrix due to incremental element rotations

$\omega_{\ell m}$ =

direction angle between the local element x_ℓ-axis and the global X_m-axis

$\Delta^t\omega_j$ = increment vector of nodal rotations at node j

ϕ = roll angle between the local x_2-axis and global X_2-axis

$\phi_{\ell m}$ =

direction angle between the local node y_ℓ-axis and the global X_m-axis

ϕ_t = member relative end torsion about x_1-axis

$\Delta\psi_i$ = incremental element rotation in local x_i-axis direction

ρ = vector of small joint rotation

${}_0^t\sigma$ = 2nd Piola-Kirchhoff stress = ${}_0^tS_{11}$

${}^t\sigma$, ${}^{t+\Delta t}\sigma$ = axial Cauchy stress at time t, t+Δt

${}^t\tau$ = Cauchy stress matrix at time t

${}^t\tau_{ij}$ = component of Cauchy stress tensor at time t = ${}_t^tS_{ij}$

${}_0\theta$, ${}_t\theta$ = column vector of displacement gradient contributing to strain ${}_0\varepsilon$, ${}_t\varepsilon$

υ = Poisson's ratio

ξ = natural coordinate = x_1 / L

$\xi_n = I_n / I$

$\zeta = L / (I/A)^{1/2}$

Appendix F

PROGRAM LISTING

```
C      **************************************************************
C      *                         GNSSF                                *
C      **************************************************************
C
C
C
C      PARAMETERS
C      **********
C
C   *                   TOLERANCES MAY BE DEFINED BY USER AT THE
C   *                   BEGINNING OF THE PROGRAM. SET A CONVERGENCE
C   *                   TOLERANCE GREATER THAN 1 TO NOT PERFORM THE
C   *                   CORRESPONDING TEST.
C
C   CPDB                CONVERGENCE TOLERANCE FOR 2-NORM DISPLACEMENT
C                       TEST
C   CPDC                CONVERGENCE TOLERANCE FOR INFINITY-NORM
C                       DISPLACEMENT TEST
C   CPF                 CONVERGENCE TOLERANCE FOR UNBALANCED FORCE TEST
C   DQQR                CONVERGENCE TOLERANCE FOR ITERATION IN BOWCOR
C   IMAX                MAX NO. OF ITERATIONS PERMITTED IN BOWCOR
C   IMP                 DEGREE OF FREEDOM FOR WHICH RESULTS ARE PRINTED
C                       FOR NPRINT = 0, 1, 2
C   ITMAX               MAX NO. OF ITERATIONS PERMITTED FOR NEWTON-
C                       RAPHSON OR MODIFIED RIKS/WEMPNER METHOD
C   MX                  MAX NUMBER OF ELEMENTS OR NODES
C   MXNEQ               MAX NUMBER OF EQUATIONS (D.O.F.)
C   NPRINT              PRINT INDICATOR  =  0 FOR EQUILIBRIUM PATH PLOT
C                                             DATA (FOR ONE D.O.F. )
C                                             (NO WORDS ARE PRINTED)
C                                    =  1 FOR EQUILIBRIUM PATH PLOT
C                                             DATA WITH TANGENTS
C                                    =  2 FOR EQUILIBRIUM PATH
C                                             RESULTS INCLUDING EXPLANA-
C                                             TIONS OF RESULTS
C                                    =  3 FOR FULL DEBUGGING OUTPUT
C                                    =  4 FOR FULL EQUILIBRIUM CON-
C                                             FIGURATION OUTPUT
C   NUPD                NUMBER OF ITERATIONS BETWEEN UPDATING TANGENT
C                       STIFFNESS MATRIX.  NUPD= 1 FOR UPDATING AT
C                       EVERY ITERATION. (MUST BE .GE. 1 )
C
C
C
C      VARIABLES
C      *********
C
C   AREA(MX)            CROSS SECTIONAL AREA OF ELEMENT I
C   AP(3,3,MX)          CURRENT NODAL ORIENTATION MATRIX
C   APNEW(3,3,MX)       NEW NODAL ORIENTATION MATRIX
C   BP12(MX),BP22(MX),BP13(MX),BP23(MX)
C                       FIRST DERIVATIVE OF BW12,BW22,BW13,BW23
C                       RESPECTIVELY WITH RESPECT TO QR
C   BW12,BW22,BW13,BW23
C                       BOWING FUNCTIONS
C   C                   CONVERGENCE RATIO TO BE COMPARED TO CONVERGENCE
C                       PARAMETERS
C   C01(MX),C02(MX),C03(MX)
C                       INITIAL DIRECTION COSINES OF ELEMENT I
C   C1PI(MX),C2PI(MX),C3PI(MX)
C                       CURRENT DIRECTION COSINES OF ELEMENT I
C   C1(MX),C2(MX),C3(MX)
C                       NEW DIRECTION COSINES OF ELEMENT I
C   CD,CN               DENOMINATOR, NUMERATOR OF C
C   CPR,CPT             CONVERGENCE RATIOS FOR ROTATION,TRANSLATION TO
C                       BE COMPARED TO CONV. PARAMETER
```

```
C   CR,CT              MAXIMUM CHANGE IN ROTATION, TRANSLATION
C   D(MXNEQ)           TOTAL GLOBAL DISPLACEMENT VECTOR
C   DD(MXNEQ)          CHANGE IN D ("DELTA-D")
C   DDO(MXNEQ)         DD FOR FIRST STEP OF MOD. R/W ITERATION TO A
C                      NEW EQUILIBRIUM POINT
C   DDO1(MXNEQ)        FOR MOD. R/W METHOD: DDO CONTRIBUTION
C                      (DDO=DQI*DDO1)
C   DD1(MXNEQ)         FOR N-R METHOD: DD FOR FIRST ITERATION
C                      FOR MOD. R/W METHOD: DD CONTRIBUTION
C                      (DD=DQI*DD1+DD2)
C   DD2(MXNEQ)         FOR MOD. R/W METHOD: DD CONTRIBUTION
C                      (DD=DQI*DD1+DD2)
C   DE(12)             GLOBAL DISPLACEMENTS OF ELEMENT BEING CONSIDERED
C   DJ(6,MX)           GLOBAL JOINT DISPLACEMENT MATRIX
C   DDJ(6,MX)          INCREMENTAL GLOBAL NODAL DISPLACEMENT MATRIX
C   DL(12)             LOCAL DISPLACEMENTS OF ELEMENT BEING CONSIDERED
C   DOT1(N),DOT2(N)    VECTORS FOR WHICH DOT PRODUCT IS OBTAINED IN
C                      DOTPRD
C   DQ                 CHANGE IN QR USED TO CONVERGE TO QR IN BOWCOR
C   DQI                CHANGE IN LOAD INCREMENT
C   ELENGO(MX)         INITIAL LENGTH OF ELEMENT I
C   ELENG(MX)          CURRENT LENGTH OF ELEMENT I
C   EMOD(MX)           MODULUS OF ELASTICITY OF ELEMENT I
C   F(MXNEQ)           ELEMENT FORCE VECTOR CAUSED BY DEFORMATIONS
C   FG(12,MX)          GLOBAL ELEMENT FOCE MATRIX
C                         F(L,I) = LTH GLOBAL FORCE OF MEMBER I
C   FL(12,MX)          LOCAL ELEMENT FORCE MATRIX
C                         F(L,I)= LTH LOCAL FORCE OF ELEMENT I
C   FP(MXNE)           F FROM PREVIOUS LOAD INCREMENT
C   FPI(MXNE)          F FROM PREVIOUS N-R ITERATION
C   G(42)              GLOBAL ELEMENT STIFFNESS COEFFICIENTS
C   G12,G22,G13,G23    INTERMEDIATE FUNCTIONS USED TO COMPUTE SKT FOR
C                      BEAM-COLUMN MODEL
C   GP1-GP20           INTERMEDIATE FUNCTION USED TO COMPUTE SKT
C   GMOD(MX)           SHEAR MODULUS OF ELASTICITY OF ELEMENT I
C   H                  INTERMEDIATE FUNCTION USED TO COMPUTE SKT
C   IC                 ITERATION COUNTER IN BOWCOR
C   ICI                CONVERGENCE INDICATOR
C                         = 0      AFTER TEST FOR CONVERGED
C                         .NE.0    AFTER TEST FOR NOT CONVERGED
C                         N-R OR MOD. R/W ITERATION PROCEDES AS LONG AS
C                         ICI DOES NOT = 0 (UNTIL MAX NO. ITERATIONS)
C                         ICI IS SET = 0 AT THE BEGINNING OF TEST SO
C                         THAT SOLUTION IS ASSUMED TO BE CONVERGED UNTIL
C                         IT IS PROVEN OTHERWISE. ICI IS ONLY CHANGED
C                         FROM ITS ZERO VALUE IF A CONVERGENCE TEST
C                         IS FAILED. AFTER TEST THE VALUE OF ICI MAY BE
C                         PRINTED TO INDICATE WHICH TESTS WERE FAILED:
C                            ICI=ICI+1      IF DISPLC WAS FAILED
C                            ICI=ICI+10     IF DISPLB WAS FAILED
C                            ICI=ICI+100    IF UNBALF WAS FAILED
C   INDEX(12,12)       MATRIX DEFINING LOCATION OF G IN SKT
C                         SKT(I,J)=G(INDEX(I,J))
C                         THE SIGN OF INDEX(I,J) IS THE SIGN OF SKT(I,J)
C   ITCT               ITERATION COUNTER FOR N-R AND MOD. R/W ITERATIONS
C   ITDES              NUMBER OF ITERATIONS DESIRED FOR MOD. R/W METHOD
C                      TO OBTAIN NEXT EQUILIB. CONFIGURATION
C   JCODE(6,MX)        NODAL CODE MATRIX: JCODE(I,J)= THE D.O.F. NUMBER
C                      AT NODE J IN THE GLOBAL I-DIRECTION
C   JDIR               NODAL DIRECTION OF APPLIED NODAL FORCE
C   JNUM               NODAL NUMBER THAT FORCE IS APPLIED TO
C   KHT(I)             COLUMN HEIGHT OF FULL TANGENT STIFFNESS FOR
C                      DEGREE OF FREEDOM I
C   KT(5)              KT(I) IS GREATER THAN ZERO TO INCLUDE THE
C                      CORRESPONDING CONTRIBUTION TO SKT
C   MAXA(I)            STORES ADDRESSES OF DIAGONAL TERMS IN THE COLUMN
```

```
C                       VECTOR REPRESENTATION OF THE STIFFNESS MATRIX
C   MCODE(12,MX)        MEMBER CODE MATRIX:
C                          MCODE(I,J) IS THE DEGREE OF FREEDOM NUMBER IN
C                          THE ITH GLOBAL DIRECTION OF MEMBER J
C   MINC(2,MX)          MEMBER INCIDENCE MATRIX = NODAL NUMBERS AT EACH
C                       END OF MEMBER
C   NC                  COUNTER FOR NUMBER OF ITERATIONS BETWEEN UPDATING
C                       SKT
C   NE                  NUMBER OF ELEMENTS
C   NEGPIV              NUMBER OF NEGATIVE PIVOTS ENCOUNTERED IN THE
C                       FACTORIZATION OF SKT = NO. OF NEGATIVE
C                       EIGENVALUES
C   NEQ                 NUMBER OF EQUATIONS (D.O.F.)
C   NJ                  NUMBER OF NODES
C   OL(I)               (C01(I)**2+C03(I)**2)**0.5 OF ELEMENT I
C   OLAMD(3,3,MX)       CURRENT ELEMENT ORIENTATION MATRIX
C   OLANEW(3,3,MX)      NEW ELEMENT ORIENTATION MATRIX
C   P(6,MX)             NODAL FORCE MATRIX:
C                          P(I,J)= FORCE AT NODE J IN DIRECTION I
C   Q(NEQ)              APPLIED FORCE DISTRIBUTION (UNFACTORED)
C   QI                  MULTIPLIER OF Q FOR CURRENT LOAD LEVEL OR
C                       INCREMENT (SOME REFERENCES CALL THIS LAMBDA)
C   QIMAX               MAXIMUM ALLOWED LOAD LEVEL
C   QL                  AXIAL FORCE / ELENGO
C   QT(NEQ)             APPLIED LOAD VECTOR FOR CURRENT LOAD INCREMENT
C   QR                  RATIO OF AXIAL FORCE TO EULER BUCKLING LOAD
C   R(3,3,MX)           ROTATION MATRIX OF ELEMENT I
C   RJ(3,3)             ROTATION MATRIX OF NODE J
C   RL(MX)              NEW LENGTH OF ELEMENT I
C   RL1(MX),RL2(MX),RL3(MX)
C                       PROJECTION OF NEW ELEMENT LENGTH ON TO THE
C                       DIRECTION OF X1,X2,X3-AXIS
C   RL1PI(MX),RL2PI(MX),RL3PI(MX)
C                       PROJECTION OF CURRENT ELEMENT LENGTH ON TO THE
C                       DIRECTION OF X1,X2,X3-AXIS
C   RM12,RM22,RM13,RM23
C                       MOMENTS AT A- AND B-END OF MEMBER ABOUT X2 AND
C                       X3-AXIS RESPECTIVELY
C   RM32(MX)            RM12+RM22 / ELENGO
C   RM33(MX)            RM13+RM23 / ELENGO
C   RMR,RMT             MAXIMUM ROTATION, TRANSLATION IN STRUCTURE
C   SIGN(NEGPIV+1)      +1.DO FOR A LOADING REGION
C                       -1.DO FOR AN UNLOADING REGION
C   SKT(I)              SYSTEM TANGENT STIFFNESS MATRIX
C   SP12(MX),SP22(MX),SP13(MX),SP23(MX)
C                       FIRST DERIVATIVE OF ST12,ST22,ST13,ST23 W.R.T. QR
C   ST12(MX),ST22(MX),ST13(MX),ST23(MX)
C                       STABILITY FUNCTION CORRESPONDING TO TR12,TR22
C                       TR13,TR23
C   TR12(MX),TR22(MX),TR13(MX),TR23(MX)
C                       TOTAL ROTATION AT A- AND B-END OF MEMBER ABOUT
C                       X2 AND X3-AXIS RESPECTIVELY
C   TTORS(I)            TOTAL RELATIVE TORSION AT B-END ABOUT X1-AXIS
C                       OF ELEMENT I
C   TCB2,TCB3           LENGTH CORRECTION FACTOR FOR BENDING ACTION
C   TCB2P,TCB3P         DERIVATIVES OF TCB2,TCB3 WITH RESPECT TO QR
C   TUL(I)              TOTAL AXIAL SHORTENING DIVIDED BY INITIAL LENGTH
C                       OF MEMBER I (POS FOR SHORTENING)
C   U(I)                INCREMENTAL AXIAL LENGTHENING OF MEMBER I
C   TU(I)               TOTAL AXIAL LENGTHENING OF MEMBER I
C   X(1,J),X(2,J),X(3,J)
C                       GLOBAL X1,X2,X3-COORDINATE OF NODE J
C   ZI2(I),ZI3(I)       MOMENT OF INERTIA ABOUT LOCAL X2 AND
C                       X3-AXIS OF ELEM. I
C   ZJ(I)               POLAR MOMENT OF INERTIA (TORSIONAL STIFFNESS)
C                       OF ELEMENT I
```

```
C
C
C
C     DATA CARDS
C     **** *****
C
C *NOTE:  ALL VARIABLES USE STANDARD WATFIV DEFAULT TYPING;
C         I.E.  IF THE FIRST LETTER OF THE VARIABLE NAME IS BETWEEN
C         I AND N INCLUSIVE, THE VARIABLE IS AN INTEGER.
C
C    QI,QIMAX,DQI
C    IMP
C    SIGN(I)
C     :  :
C     :  :
C    0.D0
C    KT(I),FOR I=1 TO 5
C    NE,NJ
C    MINC(1,I),MINC(2,I),FOR I=1 TO NE
C     NODAL NUMBER, CONSTRAINT GLOBAL DIRECTION
C           :                :
C           :                :
C          (FOR ALL CONSTRAINTS)
C    0,0
C    X(1,J),X(2,J),X(3,J) FOR J=1 TO NJ
C    AREA(I),EMOD(I),GMOD(I)
C       :         :
C       :         :
C     (FOR ALL MEMBERS)
C    ZI2(I),ZI3(I),ZJ(I)
C       :         :
C       :         :
C     (FOR ALL MEMBERS)
C    JNUM , JDIR , FORCE
C      :            :
C      :            :
C   (FOR ALL NODAL FORCES)
C    0,0,0.D0
C
C
C
C      *****************************************************************
C      *                              MAIN                             *
C      *****************************************************************
C      INITIALIZE PARAMETERS CPDB,CPDC,CPF,CPE,DP,IMAX,INDTAN,ITDES,
C      ITIND,ITMAX,MX,MXNA,MXNEQ,NUPD,NPRINT,IELS;  READ QI,QIMAX,
C      DQI,IMP,SIGN,KT;  INITIALIZE THE CONFIGURATION TO THE
C      UNDEFORMED STATE;  CALL DATA AND EITHER NEWRAP OR RIKWEM.
       IMPLICIT REAL*8(A-H,O-Z)
       DIMENSION AREA(60),BP12(60),BP22(60),BP13(60),BP23(60),
     *           C1(60),C2(60),C3(60),D(360),DD(360),DD1(360),
     *           DE(12),DL(12),DJ(6,60),ELENG(60),EMOD(60),
     *           GMOD(60),F(360),FG(12,60),FL(12,60),FP(360),
     *           FPI(360),G(42),INDEX(12,12),JCODE(6,60),KHT(361),
     *           MAXA(361),MCODE(12,60),MINC(2,60),P(6,60),
     *           Q(360),QT(360),R12(60),R22(60),R13(60),R23(60),
     *           TORS(60),TR12(60),TR22(60),TR13(60),TR23(60),
     *           TTORS(60),SKT(65160),SP12(60),SP22(60),SP13(60),
     *           SP23(60),ST12(60),ST22(60),ST13(60),ST23(60),
     *           ZI2(60),ZI3(60),ZJ(60),C1PI(60),C2PI(60),C3PI(60),
     *           DD2(360),Z(360),SIGN(10),DOT1(360),DOT2(360),
     *           DD0(360),DD01(360),RM32(60),RM33(60),QL(60),
     *           U(60),KT(5),RL1PI(60),RL2PI(60),RL3PI(60),RL1(60),
     *           RL2(60),RL3(60),RL(60),TUL(60),TCB2(60),TCB3(60),
     *           TU(60),DDJ(6,60),AP(3,3,60),APNEW(3,3,60),
     *           RJ(3,3),OL(60),CO1(60),CO2(60),CO3(60),R(3,3,60),
```

```
     *           OLAMD(3,3,60),OLANEW(3,3,60),ELENGO(60),
     *           X(3,60)
C     CPDB : ICI+10
      CPDB=1.D-03
C     1E-06 < CPDC < 1E-02
C     CPDC : ICI+1
C     CPDC=3.D-01
      CPDC=1.D-02
C     CPF : ICI+100
      CPF=1.D-01
      DQQR=1.D-03
      FQR=1.D-05
      IMAX=30
C     INDTAN = 1 FOR BEAM-COLUMN TANGENT STIFFNESS
C            = 2 FOR FINITE ELEMENT TANGENT STIFFNESS
      INDTAN=1
C     ITDES = NUMBER OF ITERATIONS DESIRED FOR CONVERGENCE IN
C             MODIFIED RIKS WEMPNER METHOD
      ITDES=3
C     ITIND = 0 FOR NEWTON RAPHSON ITERATION
C           = 1 FOR MODIFIED RIKS WEMPNER METHOD
      ITIND=1
      ITMAX=200
      MX=60
      MXNA=60
      MXNEQ=6*MX
      NUPD=300
C     NPRINT = 0 FOR EQUILIBRIUM PATH PLOT
C              1 FOR EQUILIBRIUM PATH PLOT WITH TANGENTS
C              2 FOR EQUILIBRIUM PATH RESULTS
C              3 FOR FULL DEBUGGING OUTPUT
C              4 FOR FULL FINAL OUTPUT
      NPRINT=0
      READ,QI,QIMAX,DQI
C     IMP = IMPORTANT DEGREE OF FREEDOM (D.O.F. FOR WHICH PRINTOUT
C           IS DESIRED)
      READ,IMP
      PRINT 100,NUPD,CPDB,CPDC,CPF,IMAX,ITMAX,QI,QIMAX,
     *          DQI,ITDES,ITIND,INDTAN,IMP,NPRINT
 100  FORMAT('-NUPD = ',I4/' 0CPDB = ',F10.6/' 0CPDC = ',F10.6/
     *       ' CPF   = ',F10.6/' IMAX = ',I6/' ITMAX= ',I6/
     *       '0QI    = ',F15.5/' 0QIMAX= ',F15.5/' DQI  = ',
     *       F15.5/' 0ITDES= ',I6,'  ITIND= ',I6/' INDTAN=',
     *       I3/' D.O.F. PRINTED= ',I3,'   NPRINT= ',I3)
        PRINT 200
 200    FORMAT(/'0  I',10X,'SIGN(I)')
        I=1
        READ,SIGN(I)
        WHILE (SIGN(I) .NE. 0.D00) DO
           PRINT 300,I,SIGN(I)
 300       FORMAT(' ',I3,5X,F10.1)
           I=I+1
           READ,SIGN(I)
        END WHILE
        READ,(KT(J),J=1,5)
        PRINT 400,(KT(J),J=1,5)
 400    FORMAT(/' KT=',5(3X,I4))
      CALL DATA(AREA,C1,C2,C3,ELENG,EMOD,GMOD,FL,JCODE,KHT,
     *          MAXA,MCODE,MINC,Q,QT,X,ZI2,ZI3,ZJ,MX,MXNA,
     *          MXNEQ,NE,NEQ,NJ,NKT,QI,C1PI,C2PI,C3PI,
     *          C01,C02,C03,ELENGO)
      DO 10 I=1,NEQ
         F(I)=0.D00
         FP(I)=0.D00
         FPI(I)=0.D00
         D(I)=0.D00
```

```
          DD(I)=0.D00
          DDO(I)=0.D0
          Z(I)=0.D00
   10 CONTINUE
      IELS=0
C
C     INITIALIZE THE NODAL ORIENTATION MATRIX AT TIME 0 TO
C     IDENTITY MATRIX.
      DO 35 J=1,NJ
         DO 30 K=1,3
            DO 20 L=1,3
               IF (K .EQ. L) THEN DO
                  AP(K,L,J)=1.D0
               ELSE DO
                  AP(K,L,J)=0.D0
               END IF
   20       CONTINUE
   30    CONTINUE
   35 CONTINUE
C
C     CALCULATE THE COMPONENTS OF THE INITIAL ORIENTATION MATRIX
C     OF ELEMENT I.
      DO 40 I=1,NE
         OL(I)=(C01(I)**2+C03(I)**2)**0.5
         OLAMD(1,1,I)=C01(I)
         OLAMD(1,2,I)=C02(I)
         OLAMD(1,3,I)=C03(I)
         OLAMD(2,1,I)=-C01(I)*C02(I)/OL(I)
         OLAMD(2,2,I)=OL(I)
         OLAMD(2,3,I)=-C02(I)*C03(I)/OL(I)
         OLAMD(3,1,I)=-C03(I)/OL(I)
         OLAMD(3,2,I)=0.D0
         OLAMD(3,3,I)=C01(I)/OL(I)
   40 CONTINUE
C
      IF (ITIND .EQ. 0) THEN DO
         CALL NEWRAP(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,D,DD,DD1,
     *               DE,ELENG,EMOD,GMOD,F,FG,FL,FP,FPI,G,JCODE,
     *               MAXA,MCODE,QT,R12,R22,R13,R23,TORS,TR12,TR22,
     *               TR13,TR23,TTORS,SKT,SP12,SP22,SP13,SP23,ST12,
     *               ST22,ST13,ST23,ZI2,ZI3,ZJ,CPDB,CPDC,CPF,
     *               ICI,IELS,IMAX,ITMAX,NE,NEQ,NJ,NKT,NUPD,
     *               Q,QI,C1PI,C2PI,C3PI,QIMAX,DQI,DJ,MINC,
     *               P,ITIND,RM32,RM33,QL,TU,INDTAN,KT,IMP,
     *               NPRINT,QJ,RL1PI,RL2PI,RL3PI,RL1,RL2,RL3,RL,
     *               TUL,TCB2,TCB3,DDJ,AP,APNEW,RJ,OL,C01,C02,
     *               C03,DL,R,OLAMD,OLANEW,DQQR,FQR,ELENGO,DDO)
      ELSE DO
         QI=0.D0
         CALL RIKWEM(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,D,DD,DD1,
     *               DE,ELENG,EMOD,GMOD,F,FG,FL,FP,FPI,G,JCODE,
     *               MAXA,MCODE,QT,R12,R22,R13,R23,TORS,TR12,
     *               TR22,TR13,TR23,TTORS,SKT,SP12,SP22,SP13,
     *               SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,CPDB,
     *               CPDC,CPF,ICI,IELS,IMAX,ITMAX,NE,NEQ,NJ,
     *               NKT,NUPD,Q,QI,C1PI,C2PI,C3PI,DD2,Z,SIGN,
     *               ITDES,QIMAX,DQI,DJ,MINC,P,DOT1,DOT2,ITIND,
     *               DDO,DDO1,RM32,RM33,QL,TU,INDTAN,KT,IMP,NPRINT,
     *               RL1PI,RL2PI,RL3PI,RL1,RL2,RL3,RL,TUL,TCB2,TCB3,
     *               DDJ,AP,APNEW,RJ,OL,C01,C02,C03,DL,R,OLAMD,
     *               OLANEW,DQQR,FQR,ELENGO)
      END IF
      STOP
      END
C
C
```

```
C
C     ****************************************************************
C     *                         DATA                                  *
C     ****************************************************************
      SUBROUTINE DATA(AREA,C1,C2,C3,ELENG,EMOD,GMOD,FL,JCODE,
     *                KHT,MAXA,MCODE,MINC,Q,QT,X,ZI2,ZI3,
     *                ZJ,MX,MXNA,MXNEQ,NE,NEQ,NJ,NKT,QI,
     *                C1PI,C2PI,C3PI,C01,C02,C03,ELENG0)
C     READ NE,NJ; IF NE .LE. MX AND NJ .LE. MX, CALL STRUCT AND
C     LOAD.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),C1(1),C2(1),C3(1),ELENG(1),
     *          EMOD(1),GMOD(1),FL(12,1),JCODE(6,1),
     *          KHT(1),MAXA(1),MCODE(12,1),MINC(2,1),
     *          Q(1),QT(1),X(3,1),ZI2(1),ZI3(1),ZJ(1),
     *          C1PI(1),C2PI(1),C3PI(1),C01(1),C02(1),
     *          C03(1),ELENG0(1)
      READ,NE,NJ
      PRINT 100,NE,NJ
  100 FORMAT('-NE = ',I4,7X,'NJ = ',I4)
      IF (NE .LE. MX .AND. NJ .LE. MX) THEN DO
         CALL STRUCT(AREA,ELENG,EMOD,GMOD,JCODE,KHT,MAXA,MCODE,
     *               MINC,X,ZI2,ZI3,ZJ,MXNEQ,NE,NEQ,NJ,NKT,
     *               C1PI,C2PI,C3PI,C01,C02,C03,ELENG0)
         CALL LOAD(C1,C2,C3,ELENG,EMOD,GMOD,FL,JCODE,
     *             MCODE,Q,QT,ZI2,ZI3,ZJ,MXNA,NE,NEQ,QI)
      ELSE DO
         PRINT 200
  200    FORMAT('-***NE OR NJ EXCEDES MX; REDIMENSION ARRAYS')
         STOP
      END IF
      RETURN
      END
C
C
C
C     ****************************************************************
C     *                         STRUCT                                *
C     ****************************************************************
      SUBROUTINE STRUCT(AREA,ELENG,EMOD,GMOD,JCODE,KHT,MAXA,MCODE,
     *                  MINC,X,ZI2,ZI3,ZJ,MXNEQ,NE,NEQ,NJ,NKT,
     *                  C1PI,C2PI,C3PI,C01,C02,C03,ELENG0)
C     READ AND ECHO, MINC; INITIALIZE THE ELEMENTS OF JCODE
C     TO UNITY; FOR EACH NODAL CONSTRAINT,READ AND ECHO, JNUM,JDIR,
C     AND STORE A ZERO IN THE CORRESPONDING LOCATION OF JCODE;
C     CALL CODES, DETMAX, AND PROP.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),ELENG(1),EMOD(1),GMOD(1),JCODE(6,1),
     *          KHT(1),MAXA(1),MCODE(12,1),MINC(2,1),X(3,1),
     *          ZI2(1),ZI3(1),ZJ(1),C1PI(1),C2PI(1),C3PI(1),
     *          C01(1),C02(1),C03(1),ELENG0(1)
      READ,(MINC(1,I),MINC(2,I),I=1,NE)
      PRINT 100
  100 FORMAT(/'-',7X,'MINC')
      DO 10 I=1,2
         PRINT 200,(MINC(I,J),J=1,NE)
  200    FORMAT('0',22(2X,I4))
   10 CONTINUE
      DO 30 J=1,NJ
         DO 20 I=1,6
            JCODE(I,J)=1
   20    CONTINUE
   30 CONTINUE
      PRINT 300
  300 FORMAT(/'-',10X,'NODAL NUMBER',10X,'CONSTRAINT DIRECTION')
      READ,JNUM,JDIR
```

```
      WHILE (JNUM .NE. 0) DO
         PRINT 400,JNUM,JDIR
  400    FORMAT('0',11X,I4,24X,I2)
         JCODE(JDIR,JNUM)=0
         READ,JNUM,JDIR
      END WHILE
C
      CALL CODES(JCODE,MCODE,MINC,MXNEQ,NE,NEQ,NJ)
      CALL DETMAX(KHT,MAXA,MCODE,NE,NEQ,NKT)
      CALL PROP(AREA,ELENG,EMOD,GMOD,MINC,X,ZI2,ZI3,ZJ,
     *          NE,NJ,C1PI,C2PI,C3PI,C01,C02,C03,ELENG0)
      RETURN
      END
C
C
C
C     ****************************************************************
C     *                           CODES                              *
C     ****************************************************************
      SUBROUTINE CODES(JCODE,MCODE,MINC,MXNEQ,NE,NEQ,NJ)
C     GENERATE JCODE; AND GENERATE MCODE, BY USING MINC AND JCODE.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION JCODE(6,1),MCODE(12,1),MINC(2,1)
      NEQ=0
      DO 20 J=1,NJ
         DO 10 L=1,6
            IF (JCODE(L,J) .NE. 0) THEN DO
               NEQ=NEQ+1
               JCODE(L,J)=NEQ
            END IF
   10    CONTINUE
   20 CONTINUE
      IF (NEQ .GT. MXNEQ) THEN DO
         PRINT 100
  100    FORMAT('- *** NEQ EXCEDES MXNEQ ; REDIMENSION ARRAYS ***')
         STOP
      END IF
      PRINT 200
  200 FORMAT(/'-',7X,'JCODE')
      DO 30 I=1,6
         PRINT 300,(JCODE(I,J),J=1,NJ)
  300    FORMAT('0',22(2X,I4))
   30 CONTINUE
      DO 50 I=1,NE
         J=MINC(1,I)
         K=MINC(2,I)
         DO 40 L=1,6
            MCODE(L,I)=JCODE(L,J)
            MCODE(L+6,I)=JCODE(L,K)
   40    CONTINUE
   50 CONTINUE
      PRINT 400
  400 FORMAT(/'-',7X,'MCODE')
      DO 60 I=1,12
         PRINT 500,(MCODE(I,J),J=1,NE)
  500    FORMAT('0',22(2X,I4))
   60 CONTINUE
      RETURN
      END
C
C
C
C     ****************************************************************
C     *                           DETMAX                             *
C     ****************************************************************
      SUBROUTINE DETMAX(KHT,MAXA,MCODE,NE,NEQ,NKT)
```

```
C      CALCULATE COLUMN HEIGHTS, KHT; AND CALCULATE ADDRESSES OF
C      DIAGONAL ELEMENTS IN BANDED MATRIX WHOSE COLUMN HEIGHTS
C      ARE KNOWN, MAXA.
       IMPLICIT REAL*8(A-H,O-Z)
       DIMENSION KHT(1),MAXA(1),MCODE(12,1)
       DO 10 I=1,NEQ
          KHT(I)=0
   10  CONTINUE
       DO 30 I=1,NE
          J=1
          WHILE( MCODE(J,I) .EQ. 0) DO
             J=J+1
          END WHILE
          MIN=MCODE(J,I)
          J=J+1
          DO 20 L=J,12
             K=MCODE(L,I)
             IF (K .NE. 0) THEN DO
                KHT(K)=MAX0(KHT(K),(K-MIN))
             END IF
   20     CONTINUE
   30  CONTINUE
       PRINT 100
  100  FORMAT(//'-',5X,'I',10X,'KHT(I)',10X,'MAXA(I)')
       MAXA(1)=1
       DO 40 I=1,NEQ
          PRINT 200,I,KHT(I),MAXA(I)
  200     FORMAT('0',1X,I5,9X,I5,11X,I5)
          MAXA(I+1)=MAXA(I)+KHT(I)+1
   40  CONTINUE
       NKT=MAXA(NEQ+1)-1
       I=NEQ+1
       PRINT 300,I,MAXA(I),NKT
  300  FORMAT('0',1X,I5,25X,I5//1X,'NKT = ',I5)
       RETURN
       END
C
C
C
C      ******************************************************************
C      *                             PROP                               *
C      ******************************************************************
       SUBROUTINE PROP(AREA,ELENG,EMOD,GMOD,MINC,X,ZI2,ZI3,ZJ,
      *                NE,NJ,C1PI,C2PI,C3PI,C01,C02,C03,ELENG0)
C      READ AND ECHO THE NODAL COORDINATES, X(I,J), AND ELEMENT
C      PROPERTIES. FOR EACH ELEMENT COMPUTE THE INITIAL ELEMENT
C      LENGTH AND DIRECTION COSINES.
       IMPLICIT REAL*8(A-H,O-Z)
       DIMENSION AREA(1),ELENG(1),EMOD(1),GMOD(1),MINC(2,1),
      *          X(3,1),ZI2(1),ZI3(1),ZJ(1),C1PI(1),C2PI(1),
      *          C3PI(1),C01(1),C02(1),C03(1),ELENG0(1)
       READ,(X(1,J),X(2,J),X(3,J),J=1,NJ)
       PRINT 100
  100  FORMAT(//'-','GLOBAL NODAL COORDINATES',/,18X,'X1',10X,'X2',
      *       10X,'X3')
       DO 10 J=1,NJ
          PRINT 200,J,(X(I,J),I=1,3)
  200     FORMAT('0',' NODE ',I3,3(2X,F10.4))
   10  CONTINUE
       PRINT 300
  300  FORMAT(//'-','ELEMENT PROPERTIES'/,9X,'NO.',9X,'AREA',10X,
      *       'ELENG',12X,'EMOD',12X,'GMOD')
       DO 20 I=1,NE
          J=MINC(1,I)
          K=MINC(2,I)
          EL1=X(1,K)-X(1,J)
```

```
      EL2=X(2,K)-X(2,J)
      EL3=X(3,K)-X(3,J)
      ELENG0(I)=DSQRT(EL1**2+EL2**2+EL3**2)
      ELENG(I)=ELENG0(I)
      C01(I)=EL1/ELENG0(I)
      C02(I)=EL2/ELENG0(I)
      C03(I)=EL3/ELENG0(I)
      C1PI(I)=C01(I)
      C2PI(I)=C02(I)
      C3PI(I)=C03(I)
      READ,AREA(I),EMOD(I),GMOD(I)
      PRINT 400,I,AREA(I),ELENG(I),EMOD(I),GMOD(I)
 400  FORMAT('0',6X,I4,7X,F8.3,3(2X,D15.7))
  20 CONTINUE
C
     PRINT 410
 410 FORMAT(//,' ',8X,'NO.',12X,'ZI2',18X,'ZI3',17X,'ZJ')
     DO 430 I=1,NE
      READ,ZI2(I),ZI3(I),ZJ(I)
      PRINT 420,I,ZI2(I),ZI3(I),ZJ(I)
 420  FORMAT(' ',6X,I4,3X,D15.7,8X,D15.7,6X,D15.7)
 430 CONTINUE
     RETURN
     END
C
C
C
C    ****************************************************************
C    *                          LOAD                               *
C    ****************************************************************
     SUBROUTINE LOAD(C1,C2,C3,ELENG,EMOD,GMOD,FL,JCODE,
    *                MCODE,Q,QT,ZI2,ZI3,ZJ,MXNA,NE,
    *                NEQ,QI)
C    INITIALIZE THE NODAL LOAD VECTOR,Q,TO ZERO. CALL JLOAD AND
C    MACT.
     IMPLICIT REAL*8(A-H,O-Z)
     DIMENSION C1(1),C2(1),C3(1),ELENG(1),EMOD(1),GMOD(1),
    *          FL(12,1),JCODE(6,1),MCODE(12,1),Q(1),
    *          QT(1),ZI2(1),ZI3(1),ZJ(1)
     DO 10 K=1,NEQ
      Q(K)=0.D00
  10 CONTINUE
     CALL JLOAD(JCODE,Q)
     RETURN
     END
C
C
C
C    ****************************************************************
C    *                          JLOAD                              *
C    ****************************************************************
     SUBROUTINE JLOAD(JCODE,Q)
C    READ,ECHO JNUM,JDIR,AND THE APPLIED FORCE, FORCE; STORE FORCE
C    IN Q.
     IMPLICIT REAL*8(A-H,O-Z)
     DIMENSION JCODE(6,1),Q(1),QT(1)
     READ,JNUM,JDIR,FORCE
     IF (JNUM .NE. 0) THEN DO
        PRINT 100
 100    FORMAT(//'-',3X,'NODAL NUMBER',10X,'GLOBAL DIRECTION',10X,
    *          'APPLIED FORCE')
     END IF
     WHILE (JNUM .NE. 0) DO
        PRINT 200,JNUM,JDIR,FORCE
 200    FORMAT('0',6X,I4,22X,I2,13X,F16.5,/)
        K=JCODE(JDIR,JNUM)
```

```
         Q(K)=FORCE
         READ,JNUM,JDIR,FORCE
      END WHILE
      RETURN
      END
C
C
C
C     ***************************************************************
C     *                           NEWRAP                            *
C     ***************************************************************
      SUBROUTINE NEWRAP(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,D,DD,DD1,
     *                  DE,ELENG,EMOD,GMOD,F,FG,FL,FP,FPI,G,JCODE,
     *                  MAXA,MCODE,QT,R12,R22,R13,R23,TORS,TR12,
     *                  TR22,TR13,TR23,TTORS,SKT,SP12,SP22,SP13,
     *                  SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,CPDB,
     *                  CPDC,CPF,ICI,IELS,IMAX,ITMAX,NE,NEQ,NJ,
     *                  NKT,NUPD,Q,QI,C1PI,C2PI,C3PI,QIMAX,DQI,DJ,
     *                  MINC,P,ITIND,RM32,RM33,QL,TU,INDTAN,KT,IMP,
     *                  NPRINT,RL1PI,RL2PI,RL3PI,RL1,RL2,RL3,RL,
     *                  TUL,TCB2,TCB3,DDJ,AP,APNEW,RJ,OL,CO1,CO2,
     *                  CO3,DL,R,OLAMD,OLANEW,DQQR,FQR,ELENGO,DDO)
C     EMPLOY NEWTON-RAPHSON ITERATION OR MODIFIED NEWTON-RAPHSON
C     ITERATION TO ARRIVE AT THE EQUILIBRIUM POINT.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),BP12(1),BP22(1),BP13(1),BP23(1),C1(1),C2(1),
     *          C3(1),D(1),DD(1),DD1(1),DE(1),ELENG(1),EMOD(1),
     *          GMOD(1),F(1),FG(12,1),FL(12,1),FP(1),FPI(1),G(1),
     *          JCODE(6,1),MAXA(1),MCODE(12,1),Q(1),QT(1),R12(1),
     *          R22(1),R13(1),R23(1),TORS(1),TR12(1),TR22(1),
     *          TR13(1),TR23(1),TTORS(1),SKT(1),SP12(1),SP22(1),
     *          SP13(1),SP23(1),ST12(1),ST22(1),ST13(1),ST23(1),
     *          ZI2(1),ZI3(1),ZJ(1),C1PI(1),C2PI(1),C3PI(1),
     *          DJ(6,1),MINC(2,1),P(6,1),RM32(1),RM33(1),QL(1),
     *          TU(1),KT(1),RL1PI(1),RL2PI(1),RL3PI(1),RL1(1),
     *          RL2(1),RL3(1),RL(1),TUL(1),TCB2(1),TCB3(1),
     *          DDJ(6,1),AP(3,3,60),APNEW(3,3,60),RJ(3,1),
     *          OL(1),CO1(1),CO2(1),CO3(1),DL(1),R(3,3,60),
     *          OLAMD(3,3,60),OLANEW(3,3,60),ELENGO(1),
     *          DDO(1)
C
C     INITIALIZE THE TOTAL RELATIVE DEFORMATION OF ELEMENT I TO
C     ZERO.
      DO 5 I=1,NE
         TR12(I)=0.D0
         TR22(I)=0.D0
         TR13(I)=0.D0
         TR23(I)=0.D0
         TTORS(I)=0.D0
         TUL(I)=0.D0
         TCB2(I)=0.D0
         TCB3(I)=0.D0
         TU(I)=0.D0
    5 CONTINUE
C
      NN=1
      CALL FORCES(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,DD,DE,ELENG,
     *            EMOD,GMOD,F,FG,FL,MCODE,QT,R12,R22,R13,R23,TORS,
     *            TR12,TR22,TR13,TR23,TTORS,SP12,SP22,SP13,SP23,
     *            ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,IMAX,NE,NEQ,
     *            Q,QI,C1PI,C2PI,C3PI,RM32,RM33,QL,TU,INDTAN,
     *            NPRINT,RL1PI,RL2PI,RL3PI,RL1,RL2,RL3,RL,
     *            TUL,TCB2,TCB3,DDJ,JCODE,AP,APNEW,RJ,
     *            MINC,OL,CO1,CO2,CO3,DL,R,OLAMD,OLANEW,
     *            DQQR,FQR,ELENGO,NJ,DDO,ITIND,NN)
      CALL UPDATE(QT,Q,NEQ,QI)
```

```
      WHILE( QI .LE. QIMAX) DO
         NC=NUPD
         ITCT=0
         ICI=1
         WHILE(ICI .NE. 0 .AND. ITCT .LE. ITMAX) DO
            IF (NC .GE. NUPD) THEN DO
               CALL STIFF(AREA,BP12,BP22,BP13,BP23,C1PI,C2PI,C3PI,
     *                    ELENG,EMOD,GMOD,G,MAXA,MCODE,TR12,TR22,
     *                    TR13,TR23,SKT,SP12,SP22,SP13,SP23,ST12,
     *                    ST22,ST13,ST23,ZI2,ZI3,ZJ,IELS,NE,NKT,
     *                    RM32,RM33,QL,TU,INDTAN,KT,NPRINT,RL1PI,
     *                    RL2PI,RL3PI,RL1,RL2,RL3,RL,C1,C2,C3,
     *                    OLAMD,ELENGO)
               NC=0
            END IF
C
            CALL STORE(RL1PI,RL2PI,RL3PI,ELENG,RL1,RL2,RL3,RL,
     *                 C1PI,C2PI,C3PI,C1,C2,C3,NE)
            CALL SOLVE(DD,F,MAXA,QT,SKT,NC,NEQ,NEGPIV,NKT,NPRINT)
            DO 10 I=1,NEQ
               D(I)=D(I)+DD(I)
   10       CONTINUE
            IF (ITCT .EQ. 0) THEN DO
               DO 20 I=1,NEQ
                  DD1(I)=DD(I)
   20          CONTINUE
            END IF
C
            CALL FORCES(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,DD,DE,
     *                  ELENG,EMOD,GMOD,F,FG,FL,MCODE,QT,R12,R22,
     *                  R13,R23,TORS,TR12,TR22,TR13,TR23,TTORS,SP12,
     *                  SP22,SP13,SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,
     *                  ZJ,IMAX,NE,NEQ,Q,QI,C1PI,C2PI,C3PI,RM32,
     *                  RM33,QL,TU,INDTAN,NPRINT,RL1PI,RL2PI,RL3PI,
     *                  RL1,RL2,RL3,RL,TUL,TCB2,TCB3,DDJ,JCODE,
     *                  AP,APNEW,RJ,MINC,OL,CO1,CO2,CO3,DL,R,OLAMD,
     *                  OLANEW,DQQR,FQR,ELENGO,NJ,DDO,ITIND,NN)
            CALL TEST(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,D,DD,DD1,DE,
     *                ELENG,EMOD,GMOD,F,FG,FL,FP,FPI,JCODE,MCODE,
     *                QT,R12,R22,R13,R23,TORS,SP12,SP22,SP13,
     *                SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,CPDB,
     *                CPDC,CPF,ICI,IMAX,NE,NEQ,NJ,Q,QI,CI1,CI2,
     *                CI3,ITIND,NPRINT)
            IF (NPRINT .EQ. 3) THEN DO
               PRINT 100,ITCT
  100          FORMAT(//'-',7X,'NEWTON-RAPHSON ITERATION ',I5//10X,
     *                'QT',20X,'F',20X,'D')
               DO 30 I=1,NEQ
                  PRINT 200,QT(I),F(I),D(I)
  200             FORMAT('0',3(5X,D15.7))
   30          CONTINUE
            END IF
C
            NC=NC+1
            ITCT=ITCT+1
            DO 40 I=1,NEQ
               FPI(I)=F(I)
   40       CONTINUE
         END WHILE
         IF (NPRINT .EQ. 3) THEN DO
            PRINT 300,ICI,ITCT
  300       FORMAT(//'  ICI=',I4,5X,'ITCT=',I4)
         END IF
         DO 50 I=1,NEQ
            FP(I)=F(I)
   50    CONTINUE
```

```
      CALL RESULT(C1,C2,C3,D,DJ,ELENG,EMOD,GMOD,FG,FL,
     *            JCODE,MCODE,MINC,P,Q,QT,ZI2,ZI3,ZJ,
     *            NE,NEQ,NJ,QI,IMP,NPRINT)
      QI=QI+DQI
      IF (ICI .NE. 0) THEN DO
         PRINT 400
400      FORMAT(' *** LAST SOLUTION IS NOT CONVERGED ***')
         STOP
      END IF
      CALL UPDATE(QT,Q,NEQ,QI)
   END WHILE
   RETURN
   END
C
C
C
C     ******************************************************************
C     *                              RIKWEM                            *
C     ******************************************************************
      SUBROUTINE RIKWEM(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,D,DD,DD1,
     *                  DE,ELENG,EMOD,GMOD,F,FG,FL,FP,FPI,G,JCODE,
     *                  MAXA,MCODE,QT,R12,R22,R13,R23,TORS,TR12,
     *                  TR22,TR13,TR23,TTORS,SKT,SP12,SP22,SP13,
     *                  SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,CPDB,
     *                  CPDC,CPF,ICI,IELS,IMAX,ITMAX,NE,NEQ,NJ,
     *                  NKT,NUPD,Q,QI,C1PI,C2PI,C3PI,DD2,Z,SIGN,
     *                  ITDES,QIMAX,DQI,DJ,MINC,P,DOT1,DOT2,ITIND,
     *                  DD0,DD01,RM32,RM33,QL,TU,INDTAN,KT,IMP,
     *                  NPRINT,RL1PI,RL2PI,RL3PI,RL1,RL2,RL3,
     *                  RL,TUL,TCB2,TCB3,DDJ,AP,APNEW,RJ,OL,
     *                  CO1,CO2,CO3,DL,R,OLAMD,OLANEW,DQQR,
     *                  FQR,ELENGO)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),BP12(1),BP22(1),BP13(1),BP23(1),C1(1),C2(1),
     *          C3(1),D(1),DD(1),DD1(1),DE(1),ELENG(1),EMOD(1),
     *          GMOD(1),F(1),FG(12,1),FL(12,1),FP(1),FPI(1),G(1),
     *          JCODE(6,1),MAXA(1),MCODE(12,1),Q(1),QT(1),R12(1),
     *          R22(1),R13(1),R23(1),TORS(1),TR12(1),TR22(1),
     *          TR13(1),TR23(1),TTORS(1),SKT(1),SP12(1),SP22(1),
     *          SP13(1),SP23(1),ST12(1),ST22(1),ST13(1),ST23(1),
     *          ZI2(1),ZI3(1),ZJ(1),C1PI(1),C2PI(1),C3PI(1),DD2(1),
     *          Z(1),SIGN(1),DJ(6,1),MINC(2,1),P(6,1),DOT1(1),
     *          DOT2(1),DD0(1),DD01(1),RM32(1),RM33(1),QL(1),
     *          TU(1),KT(1),RL1PI(1),RL2PI(1),RL3PI(1),RL1(1),
     *          RL2(1),RL3(1),RL(1),TUL(1),TCB2(1),TCB3(1),
     *          DDJ(6,1),AP(3,3,60),APNEW(3,3,60),RJ(3,1),OL(1),
     *          CO1(1),CO2(1),CO3(1),DL(1),R(3,3,60),
     *          OLAMD(3,3,60),OLANEW(3,3,60),ELENGO(1)
      ITCT=0
      NEGPIV=0
C
C     INITIALIZE THE TOTAL RELATIVE DEFORMATION OF ELEMENT I
C     TO ZERO.
      DO 5 I=1,NE
         TR12(I)=0.D0
         TR22(I)=0.D0
         TR13(I)=0.D0
         TR23(I)=0.D0
         TTORS(I)=0.D0
         TUL(I)=0.D0
         TCB2(I)=0.D0
         TCB3(I)=0.D0
         TU(I)=0.D0
    5 CONTINUE
C
      NN=1
```

```
      CALL FORCES(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,DD,DE,ELENG,
     *            EMOD,GMOD,F,FG,FL,MCODE,QT,R12,R22,R13,R23,TORS,
     *            TR12,TR22,TR13,TR23,TTORS,SP12,SP22,SP13,SP23,
     *            ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,IMAX,NE,NEQ,
     *            Q,QI,C1PI,C2PI,C3PI,RM32,RM33,QL,TU,INDTAN,
     *            NPRINT,RL1PI,RL2PI,RL3PI,RL1,RL2,RL3,RL,
     *            TUL,TCB2,TCB3,DDJ,JCODE,AP,APNEW,RJ,
     *            MINC,OL,CO1,CO2,CO3,DL,R,OLAMD,OLANEW,
     *            DQQR,FQR,ELENGO,NJ,DDO,ITIND,NN)
C
      WHILE (QI .LE. QIMAX .AND. ITCT .LE. ITMAX) DO
         CALL STIFF(AREA,BP12,BP22,BP13,BP23,C1PI,C2PI,C3PI,ELENG,
     *              EMOD,GMOD,G,MAXA,MCODE,TR12,TR22,TR13,TR23,
     *              SKT,SP12,SP22,SP13,SP23,ST12,ST22,ST13,
     *              ST23,ZI2,ZI3,ZJ,IELS,NE,NKT,RM32,RM33,QL,
     *              TU,INDTAN,KT,NPRINT,RL1PI,RL2PI,RL3PI,
     *              RL1,RL2,RL3,RL,C1,C2,C3,OLAMD,ELENGO)
         CALL STORE(RL1PI,RL2PI,RL3PI,ELENG,RL1,RL2,RL3,RL,
     *              C1PI,C2PI,C3PI,C1,C2,C3,NE)
         CALL SOLVE(DDO1,Z,MAXA,Q,SKT,0,NEQ,NEGPIV,NKT,NPRINT)
         IF (ITCT .EQ. 0) THEN DO
            DS=DQI*DSQRT(DOTPRD(DDO1,DDO1,NEQ)+1.D0)
         ELSE DO
            DQI=SIGN(NEGPIV+1)*DS/DSQRT(DOTPRD(DDO1,DDO1,NEQ)+1.D0)
            IF (NPRINT .EQ. 3) THEN DO
               PRINT 250,SIGN(NEGPIV+1),NEGPIV,DS,DQI
  250          FORMAT(//'  SIGN=',D15.7,'   NEGPIV=',I4/' DS=',D15.7,
     *                '   DQI=',D15.7)
            END IF
         END IF
         DQI1=DQI
         DO 10 I=1,NEQ
            DDO(I)=DQI*DDO1(I)
            D(I)=D(I)+DDO(I)
   10    CONTINUE
         QI=QI+DQI
         IF (NPRINT .EQ. 1) THEN DO
            PRINT 400,QI,D(IMP)
  400       FORMAT(' ',20X,F13.7/' ',20X,F14.8)
         END IF
         IF (NPRINT .EQ. 3) THEN DO
            PRINT 500,QI,D(IMP)
  500       FORMAT(//' QI INIT = ',F13.7,'   D INIT = ',F14.8//)
         END IF
C
         NN=1
         CALL FORCES(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,DD,DE,ELENG,
     *               EMOD,GMOD,F,FG,FL,MCODE,QT,R12,R22,R13,R23,
     *               TORS,TR12,TR22,TR13,TR23,TTORS,SP12,SP22,SP13,
     *               SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,IMAX,NE,
     *               NEQ,Q,QI,C1PI,C2PI,C3PI,RM32,RM33,QL,TU,
     *               INDTAN,NPRINT,RL1PI,RL2PI,RL3PI,RL1,RL2,RL3,
     *               RL,TUL,TCB2,TCB3,DDJ,JCODE,AP,APNEW,RJ,
     *               MINC,OL,CO1,CO2,CO3,DL,R,OLAMD,OLANEW,
     *               DQQR,FQR,ELENGO,NJ,DDO,ITIND,NN)
         ICI=1
         NC=NUPD
         IT=0
C
         WHILE (ICI .NE. 0 .AND. IT .LE. IMAX) DO
            CALL UPDATE(QT,Q,NEQ,QI)
            IF (NC .GE. NUPD) THEN DO
               CALL STIFF(AREA,BP12,BP22,BP13,BP23,C1PI,C2PI,C3PI,
     *                    ELENG,EMOD,GMOD,G,MAXA,MCODE,TR12,TR22,
     *                    TR13,TR23,SKT,SP12,SP22,SP13,SP23,ST12,
     *                    ST22,ST13,ST23,ZI2,ZI3,ZJ,IELS,NE,NKT,
```

```
     *                 RM32,RM33,QL,TU,INDTAN,KT,NPRINT,RL1PI,
     *                 RL2PI,RL3PI,RL1,RL2,RL3,RL,C1,C2,C3,
     *                 OLAMD,ELENG0)
                NC=0
             END IF
             CALL STORE(RL1PI,RL2PI,RL3PI,ELENG,RL1,RL2,RL3,RL,
     *                  C1PI,C2PI,C3PI,C1,C2,C3,NE)
             CALL SOLVE(DD1,Z,MAXA,Q,SKT,NC,NEQ,NEGPIV,NKT,NPRINT)
             CALL SOLVE(DD2,F,MAXA,QT,SKT,1,NEQ,NEGPIV,NKT,NPRINT)
             DQI=-(DOTPRD(DD0,DD2,NEQ))/(DOTPRD(DD0,DD1,NEQ)+DQI1)
             DO 20 I=1,NEQ
                DD(I)=DQI*DD1(I)+DD2(I)
                D(I)=D(I)+DD(I)
   20        CONTINUE
C
             NN=2
             CALL FORCES(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,DD,DE,
     *                   ELENG,EMOD,GMOD,F,FG,FL,MCODE,QT,R12,R22,
     *                   R13,R23,TORS,TR12,TR22,TR13,TR23,TTORS,SP12,
     *                   SP22,SP13,SP23,ST12,ST22,ST13,ST23,ZI2,
     *                   ZI3,ZJ,IMAX,NE,NEQ,Q,QI,C1PI,C2PI,C3PI,RM32,
     *                   RM33,QL,TU,INDTAN,NPRINT,RL1PI,RL2PI,RL3PI,
     *                   RL1,RL2,RL3,RL,TUL,TCB2,TCB3,DDJ,JCODE,AP,
     *                   APNEW,RJ,MINC,OL,C01,C02,C03,DL,R,OLAMD,
     *                   OLANEW,DQQR,FQR,ELENG0,NJ,DD0,ITIND,NN)
             QI=QI+DQI
             CALL TEST(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,D,DD,DD1,DE,
     *                 ELENG,EMOD,GMOD,F,FG,FL,FP,FPI,JCODE,MCODE,
     *                 QT,R12,R22,R13,R23,TORS,SP12,SP22,SP13,SP23,
     *                 ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,CPDB,CPDC,
     *                 CPF,ICI,IMAX,NE,NEQ,NJ,Q,QI,CI1,CI2,CI3,
     *                 ITIND,NPRINT)
             NC=NC+1
             IT=IT+1
             IF (NPRINT .EQ. 3) THEN DO
                PRINT 600,IT,QI,D(1),ICI
  600           FORMAT(//'  ITERATION ',I3,' QI=',F13.7,'   D= ',
     *                 F13.7,'  ICI=',I5)
             END IF
          END WHILE
          ITCT=ITCT+1
          DO 50 I=1,NEQ
             FP(I)=F(I)
   50     CONTINUE
          IF (NPRINT .EQ. 3) THEN DO
             PRINT 700,IT
  700        FORMAT(//' ',I5,' ITERATIONS')
          END IF
          CALL RESULT(C1,C2,C3,D,DJ,ELENG,EMOD,GMOD,FG,FL,JCODE,
     *                MCODE,MINC,P,Q,QT,ZI2,ZI3,ZJ,NE,NEQ,NJ,
     *                QI,IMP,NPRINT)
          IF (ICI .NE. 0) THEN DO
             PRINT 800
  800        FORMAT(' *** LAST SOLUTION IS NOT CONVERGED ***')
             STOP
          END IF
          DS=DS*DSQRT(DFLOAT(ITDES)/DFLOAT(IT))
          IF (NPRINT .EQ. 3) THEN DO
             PRINT 350,DS
  350        FORMAT(//' DS= ',F14.8,'   ***')
          END IF
       END WHILE
       RETURN
       END
C
C
```

```
C
C     ******************************************************************
C     *                             STIFF                              *
C     ******************************************************************
      SUBROUTINE STIFF(AREA,BP12,BP22,BP13,BP23,C1PI,C2PI,C3PI,
     *                 ELENG,EMOD,GMOD,G,MAXA,MCODE,TR12,TR22,TR13,
     *                 TR23,SKT,SP12,SP22,SP13,SP23,ST12,ST22,ST13,
     *                 ST23,ZI2,ZI3,ZJ,IELS,NE,NKT,RM32,RM33,QL,
     *                 TU,INDTAN,KT,NPRINT,RL1PI,RL2PI,RL3PI,
     *                 RL1,RL2,RL3,RL,C1,C2,C3,OLAMD,ELENGO)
C     INITIALIZE THE SYSTEM TANGENT STIFFNESS MATRIX,SKT,TO ZERO;
C     FOR EACH ELEMENT CALL ELEMS1 (BEAM-COLUMN MODEL) OR ELEMS2
C     (FINITE ELEMENT MODEL) AND ASSEMS.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),BP12(1),BP22(1),BP13(1),BP23(1),C1PI(1),
     *          C2PI(1),C3PI(1),ELENG(1),EMOD(1),GMOD(1),G(1),
     *          MAXA(1),MCODE(12,1),TR12(1),TR22(1),TR13(1),TR23(1),
     *          SKT(1),SP12(1),SP22(1),SP13(1),SP23(1),ST12(1),
     *          ST22(1),ST13(1),ST23(1),ZI2(1),ZI3(1),ZJ(1),
     *          RM32(1),RM33(1),QL(1),TU(1),KT(1),RL1PI(1),RL2PI(1),
     *          RL3PI(1),RL1(1),RL2(1),RL3(1),RL(1),C1(1),C2(1),
     *          C3(1),OLAMD(3,3,60),ELENGO(1)
      IF (NPRINT .EQ. 3) THEN DO
         PRINT 100
  100    FORMAT(//' STIFF CALLED')
      END IF
      DO 10 I=1,NKT
         SKT(I)=0.D00
   10 CONTINUE
      I=1
C
      WHILE (I .LE. NE) DO
         IF (INDTAN .EQ. 1) THEN DO
            CALL ELEMS1(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,ELENG,
     *                  EMOD,GMOD,G,TR12,TR22,TR13,TR23,SP12,SP22,
     *                  SP13,SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,I,
     *                  IELS,NE,RM32,RM33,QL,NPRINT,KT,OLAMD,ELENGO)
         END IF
         IF (INDTAN .EQ. 2) THEN DO
            CALL ELEMS2(AREA,C1,C2,C3,ELENGO,EMOD,GMOD,G,TR12,
     *                  TR22,TR13,TR23,TU,I,ZI2,ZI3,ZJ,KT,NPRINT,QL,
     *                  OLAMD)
         END IF
         CALL ASSEMS(G,MAXA,MCODE,SKT,I,NKT,NPRINT)
         I=I+1
      END WHILE
C
      IF (NPRINT .EQ. 3) THEN DO
         PRINT 200
  200    FORMAT(//'0',7X,'STIFFNESS MATRIX')
         PRINT 300,(SKT(K),K=1,NKT)
  300    FORMAT(' ',6(5X,D15.7))
      END IF
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                            ELEMS1                              *
C     ******************************************************************
      SUBROUTINE ELEMS1(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,ELENG,
     *                  EMOD,GMOD,G,TR12,TR22,TR13,TR23,SP12,SP22,
     *                  SP13,SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,
     *                  I,IELS,NE,RM32,RM33,QL,NPRINT,KT,OLAMD,
     *                  ELENGO)
```

```
C     COMPUTE THE COMPONENTS OF THE ELEMENT TANGENT STIFFNESS MATRIX
C     FOR THE BEAM-COLUMN MODEL.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),BP12(1),BP22(1),BP13(1),BP23(1),C1(1),
     *          C2(1),C3(1),ELENG(1),EMOD(1),GMOD(1),G(1),
     *          TR12(1),TR22(1),TR13(1),TR23(1),SP12(1),SP22(1),
     *          SP13(1),SP23(1),ST12(1),ST22(1),ST13(1),ST23(1),
     *          ZI2(1),ZI3(1),ZJ(1),RM32(1),RM33(1),QL(1),KT(1),
     *          OLAMD(3,3,60),ELENGO(1)
      PI=3.141592653589793
      IF (IELS .EQ. 0) THEN DO
         ST12(I)=4.D00
         ST22(I)=2.D00
         ST13(I)=4.D00
         ST23(I)=2.D00
         G12=0.D00
         G22=0.D00
         G13=0.D00
         G23=0.D00
         H=(PI**2*ZI2(I))/(AREA(I)*ELENGO(I)**2)
         IF (I .EQ. NE) THEN DO
            IELS=1
         END IF
      ELSE DO
         G12=SP12(I)*TR12(I)+SP22(I)*TR22(I)
         G22=SP22(I)*TR12(I)+SP12(I)*TR22(I)
         G13=SP13(I)*TR13(I)+SP23(I)*TR23(I)
         G23=SP23(I)*TR13(I)+SP13(I)*TR23(I)
         H=(PI**2*ZI2(I))/(AREA(I)*ELENGO(I)**2)+BP12(I)*(TR12(I)
     *     +TR22(I))**2+BP22(I)*(TR12(I)-TR22(I))**2+ZI2(I)/ZI3(I)*
     *     (BP13(I)*(TR13(I)+TR23(I))**2
     *     +BP23(I)*(TR13(I)-TR23(I))**2)
      END IF
C
C     COMPUTE THE COMPONENTS OF THE CONVENTIONAL LINEAR STIFFNESS
C     MATRIX IN THE LOCAL COORDINATE.
      ETA=GMOD(I)*ZJ(I)/(EMOD(I)*ZI2(I))
      GP1=PI**2/(H*ELENGO(I)**2)
      GP2=(G13+G23)/(H*ELENGO(I)**2)
      GP3=(-G12-G22)/(H*ELENGO(I)**2)
      GP4=G12/(H*ELENGO(I))
      GP5=G13/(H*ELENGO(I))
      GP6=G22/(H*ELENGO(I))
      GP7=G23/(H*ELENGO(I))
      GP8=2.D0*ZI3(I)/ZI2(I)*(ST13(I)+ST23(I))/ELENGO(I)**2
     *    +(G13+G23)**2/(ELENGO(I)**2*PI**2*H)
      GP9=ZI3(I)/ZI2(I)*(ST13(I)+ST23(I))/ELENGO(I)+(G13**2
     *    +G13*G23)/(ELENGO(I)*PI**2*H)
      GP10=ZI3(I)/ZI2(I)*(ST13(I)+ST23(I))/ELENGO(I)+(G23**2
     *     +G13*G23)/(ELENGO(I)*PI**2*H)
      GP11=2.D0*(ST12(I)+ST22(I))/ELENGO(I)**2+(G12+G22)**2/
     *     (ELENGO(I)**2*PI**2*H)
      GP12=-(ST12(I)+ST22(I))/ELENGO(I)-(G12**2+G12*G22)/
     *     (ELENGO(I)*PI**2*H)
      GP13=-(ST12(I)+ST22(I))/ELENGO(I)-(G22**2+G12*G22)/
     *     (ELENGO(I)*PI**2*H)
      GP14=ETA
      GP15=ST12(I)+G12**2/(PI**2*H)
      GP16=ST22(I)+G12*G22/(PI**2*H)
      GP17=ZI3(I)/ZI2(I)*ST13(I)+G13**2/(PI**2*H)
      GP18=ZI3(I)/ZI2(I)*ST23(I)+G13*G23/(PI**2*H)
      GP19=ST12(I)+G22**2/(PI**2*H)
      GP20=ZI3(I)/ZI2(I)*ST13(I)+G23**2/(PI**2*H)
C
C     COMPUTE THE COMPONENTS OF THE CONVENTIONAL LINEAR STIFFNESS
C     MATRIX IN THE GLOBAL COORDINATE.
```

```
      ALPHA=EMOD(I)*ZI2(I)/ELENGO(I)
      G(1)=(C1(I)**2*GP1+2.D0*C1(I)*OLAMD(2,1,I)*GP2
     *     +2.D0*C1(I)*OLAMD(3,1,I)*GP3+OLAMD(2,1,I)**2*GP8
     *     +OLAMD(3,1,I)**2*GP11)*ALPHA
      G(2)=(C1(I)*C2(I)*GP1+(C1(I)*OLAMD(2,2,I)+C2(I)*
     *      OLAMD(2,1,I))*GP2+(C1(I)*OLAMD(3,2,I)+C2(I)*
     *      OLAMD(3,1,I))*GP3+OLAMD(2,1,I)*OLAMD(2,2,I)*GP8
     *     +OLAMD(3,1,I)*OLAMD(3,2,I)*GP11)*ALPHA
      G(3)=(C1(I)*C3(I)*GP1+(C1(I)*OLAMD(2,3,I)+C3(I)*
     *      OLAMD(2,1,I))*GP2+(C1(I)*OLAMD(3,3,I)+C3(I)*
     *      OLAMD(3,1,I))*GP3+OLAMD(2,1,I)*OLAMD(2,3,I)*GP8
     *     +OLAMD(3,1,I)*OLAMD(3,3,I)*GP11)*ALPHA
      G(4)=(C1(I)*OLAMD(2,1,I)*GP4+C1(I)*OLAMD(3,1,I)*GP5
     *     +OLAMD(2,1,I)*OLAMD(3,1,I)*GP9+OLAMD(3,1,I)*OLAMD(2,1,I)*
     *      GP12)*ALPHA
      G(5)=(C1(I)*OLAMD(2,2,I)*GP4+C1(I)*OLAMD(3,2,I)*GP5
     *     +OLAMD(2,1,I)*OLAMD(3,2,I)*GP9+OLAMD(3,1,I)*OLAMD(2,2,I)*
     *      GP12)*ALPHA
      G(6)=(C1(I)*OLAMD(2,3,I)*GP4+C1(I)*OLAMD(3,3,I)*GP5
     *     +OLAMD(2,1,I)*OLAMD(3,3,I)*GP9+OLAMD(3,1,I)*OLAMD(2,3,I)*
     *      GP12)*ALPHA
      G(7)=(C1(I)*OLAMD(2,1,I)*GP6+C1(I)*OLAMD(3,1,I)*GP7
     *     +OLAMD(2,1,I)*OLAMD(3,1,I)*GP10+OLAMD(3,1,I)*
     *      OLAMD(2,1,I)*GP13)*ALPHA
      G(8)=(C1(I)*OLAMD(2,2,I)*GP6+C1(I)*OLAMD(3,2,I)*GP7
     *     +OLAMD(2,1,I)*OLAMD(3,2,I)*GP10+OLAMD(3,1,I)*
     *      OLAMD(2,2,I)*GP13)*ALPHA
      G(9)=(C1(I)*OLAMD(2,3,I)*GP6+C1(I)*OLAMD(3,3,I)*GP7
     *     +OLAMD(2,1,I)*OLAMD(3,3,I)*GP10+OLAMD(3,1,I)*
     *      OLAMD(2,3,I)*GP13)*ALPHA
      G(10)=(C2(I)**2*GP1+2.D0*C2(I)*OLAMD(2,2,I)*GP2+2.D0*C2(I)*
     *       OLAMD(3,2,I)*GP3+OLAMD(2,2,I)**2*GP8+OLAMD(3,2,I)**2*
     *       GP11)*ALPHA
      G(11)=(C2(I)*C3(I)*GP1+(C2(I)*OLAMD(2,3,I)+C3(I)*
     *       OLAMD(2,2,I))*GP2+(C2(I)*OLAMD(3,3,I)+C3(I)*
     *       OLAMD(3,2,I))*GP3+OLAMD(2,2,I)*OLAMD(2,3,I)*GP8
     *      +OLAMD(3,2,I)*OLAMD(3,3,I)*GP11)*ALPHA
      G(12)=(C2(I)*OLAMD(2,1,I)*GP4+C2(I)*OLAMD(3,1,I)*GP5
     *      +OLAMD(2,2,I)*OLAMD(3,1,I)*GP9+OLAMD(3,2,I)*
     *       OLAMD(2,1,I)*GP12)*ALPHA
      G(13)=(C2(I)*OLAMD(2,2,I)*GP4+C2(I)*OLAMD(3,2,I)*GP5
     *      +OLAMD(2,2,I)*OLAMD(3,2,I)*GP9+OLAMD(3,2,I)*
     *       OLAMD(2,2,I)*GP12)*ALPHA
      G(14)=(C2(I)*OLAMD(2,3,I)*GP4+C2(I)*OLAMD(3,3,I)*GP5
     *      +OLAMD(2,2,I)*OLAMD(3,3,I)*GP9+OLAMD(3,2,I)*
     *       OLAMD(2,3,I)*GP12)*ALPHA
      G(15)=(C2(I)*OLAMD(2,1,I)*GP6+C2(I)*OLAMD(3,1,I)*GP7
     *      +OLAMD(2,2,I)*OLAMD(3,1,I)*GP10+OLAMD(3,2,I)*
     *       OLAMD(2,1,I)*GP13)*ALPHA
      G(16)=(C2(I)*OLAMD(2,2,I)*GP6+C2(I)*OLAMD(3,2,I)*GP7
     *      +OLAMD(2,2,I)*OLAMD(3,2,I)*GP10+OLAMD(3,2,I)*
     *       OLAMD(2,2,I)*GP13)*ALPHA
      G(17)=(C2(I)*OLAMD(2,3,I)*GP6+C2(I)*OLAMD(3,3,I)*GP7
     *      +OLAMD(2,2,I)*OLAMD(3,3,I)*GP10+OLAMD(3,2,I)*
     *       OLAMD(2,3,I)*GP13)*ALPHA
      G(18)=(C3(I)**2*GP1+2.D0*C3(I)*OLAMD(2,3,I)*GP2+2.D0*
     *       C3(I)*OLAMD(3,3,I)*GP3+OLAMD(2,3,I)**2*GP8
     *      +OLAMD(3,3,I)**2*GP11)*ALPHA
      G(19)=(C3(I)*OLAMD(2,1,I)*GP4+C3(I)*OLAMD(3,1,I)*GP5
     *      +OLAMD(2,3,I)*OLAMD(3,1,I)*GP9+OLAMD(3,3,I)*
     *       OLAMD(2,1,I)*GP12)*ALPHA
      G(20)=(C3(I)*OLAMD(2,2,I)*GP4+C3(I)*OLAMD(3,2,I)*GP5
     *      +OLAMD(2,3,I)*OLAMD(3,2,I)*GP9+OLAMD(3,3,I)*
     *       OLAMD(2,2,I)*GP12)*ALPHA
      G(21)=(C3(I)*OLAMD(2,3,I)*GP4+C3(I)*OLAMD(3,3,I)*GP5
     *      +OLAMD(2,3,I)*OLAMD(3,3,I)*GP9+OLAMD(3,3,I)*
```

```
     *         OLAMD(2,3,I)*GP12)*ALPHA
      G(22)=(C3(I)*OLAMD(2,1,I)*GP6+C3(I)*OLAMD(3,1,I)*GP7
     *         +OLAMD(2,3,I)*OLAMD(3,1,I)*GP10+OLAMD(3,3,I)*
     *         OLAMD(2,1,I)*GP13)*ALPHA
      G(23)=(C3(I)*OLAMD(2,2,I)*GP6+C3(I)*OLAMD(3,2,I)*GP7
     *         +OLAMD(2,3,I)*OLAMD(3,2,I)*GP10+OLAMD(3,3,I)*
     *         OLAMD(2,2,I)*GP13)*ALPHA
      G(24)=(C3(I)*OLAMD(2,3,I)*GP6+C3(I)*OLAMD(3,3,I)*GP7
     *         +OLAMD(2,3,I)*OLAMD(3,3,I)*GP10+OLAMD(3,3,I)*
     *         OLAMD(2,3,I)*GP13)*ALPHA
      G(25)=(C1(I)**2*GP14+OLAMD(2,1,I)**2*GP15+OLAMD(3,1,I)**2*
     *         GP17)*ALPHA
      G(26)=(C1(I)*C2(I)*GP14+OLAMD(2,1,I)*OLAMD(2,2,I)*GP15
     *         +OLAMD(3,1,I)*OLAMD(3,2,I)*GP17)*ALPHA
      G(27)=(C1(I)*C3(I)*GP14+OLAMD(2,1,I)*OLAMD(2,3,I)*GP15
     *         +OLAMD(3,1,I)*OLAMD(3,3,I)*GP17)*ALPHA
      G(28)=(-C1(I)**2*GP14+OLAMD(2,1,I)**2*GP16+OLAMD(3,1,I)**2*
     *         GP18)*ALPHA
      G(29)=(-C1(I)*C2(I)*GP14+OLAMD(2,1,I)*OLAMD(2,2,I)*GP16
     *         +OLAMD(3,1,I)*OLAMD(3,2,I)*GP18)*ALPHA
      G(30)=(-C1(I)*C3(I)*GP14+OLAMD(2,1,I)*OLAMD(2,3,I)*GP16
     *         +OLAMD(3,1,I)*OLAMD(3,3,I)*GP18)*ALPHA
      G(31)=(C2(I)**2*GP14+OLAMD(2,2,I)**2*GP15+OLAMD(3,2,I)**2*
     *         GP17)*ALPHA
      G(32)=(C2(I)*C3(I)*GP14+OLAMD(2,2,I)*OLAMD(2,3,I)*GP15
     *         +OLAMD(3,2,I)*OLAMD(3,3,I)*GP17)*ALPHA
      G(33)=(-C2(I)**2*GP14+OLAMD(2,2,I)**2*GP16+OLAMD(3,2,I)**2*
     *         GP18)*ALPHA
      G(34)=(-C2(I)*C3(I)*GP14+OLAMD(2,2,I)*OLAMD(2,3,I)*GP16
     *         +OLAMD(3,2,I)*OLAMD(3,3,I)*GP18)*ALPHA
      G(35)=(C3(I)**2*GP14+OLAMD(2,3,I)**2*GP15+OLAMD(3,3,I)**2*
     *         GP17)*ALPHA
      G(36)=(-C3(I)**2*GP14+OLAMD(2,3,I)**2*GP16+OLAMD(3,3,I)**2*
     *         GP18)*ALPHA
      G(37)=(C1(I)**2*GP14+OLAMD(2,1,I)**2*GP19+OLAMD(3,1,I)**2*
     *         GP20)*ALPHA
      G(38)=(C1(I)*C2(I)*GP14+OLAMD(2,1,I)*OLAMD(2,2,I)*GP19
     *         +OLAMD(3,1,I)*OLAMD(3,2,I)*GP20)*ALPHA
      G(39)=(C1(I)*C3(I)*GP14+OLAMD(2,1,I)*OLAMD(2,3,I)*GP19
     *         +OLAMD(3,1,I)*OLAMD(3,3,I)*GP20)*ALPHA
      G(40)=(C2(I)**2*GP14+OLAMD(2,2,I)**2*GP19+OLAMD(3,2,I)**2*
     *         GP20)*ALPHA
      G(41)=(C2(I)*C3(I)*GP14+OLAMD(2,2,I)*OLAMD(2,3,I)*GP19
     *         +OLAMD(3,2,I)*OLAMD(3,3,I)*GP20)*ALPHA
      G(42)=(C3(I)**2*GP14+OLAMD(2,3,I)**2*GP19+OLAMD(3,3,I)**2*
     *         GP20)*ALPHA
C
      IF (KT(2) .GT. 0) THEN DO
         G(1)=G(1)+RM33(I)*(2.D0*C1(I)*OLAMD(2,1,I))
     *                +RM32(I)*(-2.D0*C1(I)*OLAMD(3,1,I))
     *                +QL(I)*(-OLAMD(2,1,I)**2-OLAMD(3,1,I)**2)
         G(2)=G(2)+RM33(I)*(C2(I)*OLAMD(2,1,I)+C1(I)*OLAMD(2,2,I))
     *                +RM32(I)*(-C2(I)*OLAMD(3,1,I)-C1(I)*OLAMD(3,2,I))
     *                +QL(I)*(-OLAMD(2,1,I)*OLAMD(2,2,I)
     *                        -OLAMD(3,1,I)*OLAMD(3,2,I))
         G(3)=G(3)+RM33(I)*(C3(I)*OLAMD(2,1,I)+C1(I)*OLAMD(2,3,I))
     *                +RM32(I)*(-C3(I)*OLAMD(3,1,I)-C1(I)*OLAMD(3,3,I))
     *                +QL(I)*(-OLAMD(2,1,I)*OLAMD(2,3,I)
     *                        -OLAMD(3,1,I)*OLAMD(3,3,I))
         G(10)=G(10)+RM33(I)*(2.D0*C2(I)*OLAMD(2,2,I))
     *                  +RM32(I)*(-2.D0*C2(I)*OLAMD(3,2,I))
     *                  +QL(I)*(-OLAMD(2,2,I)**2-OLAMD(3,2,I)**2)
         G(11)=G(11)+RM33(I)*(C3(I)*OLAMD(2,2,I)
     *                          +C2(I)*OLAMD(2,3,I))
     *                  +RM32(I)*(-C3(I)*OLAMD(3,2,I)
     *                          -C2(I)*OLAMD(3,3,I))
```

```
     *               +QL(I)*(-OLAMD(2,2,I)*OLAMD(2,3,I)
     *                   -OLAMD(3,2,I)*OLAMD(3,3,I))
        G(18)=G(18)+RM33(I)*(2.D0*C3(I)*OLAMD(2,3,I))
     *               +RM32(I)*(-2.D0*C3(I)*OLAMD(3,3,I))
     *               +QL(I)*(-OLAMD(2,3,I)**2-OLAMD(3,3,I)**2)
      END IF
C
      IF (NPRINT .EQ. 3) THEN DO
        PRINT 100,I,(G(KI),KI=1,42)
  100   FORMAT(//'-',2X,'G(42) FOR ELE.',I3/5(5X,D15.7)/
     *          5(5X,D15.7)/5(5X,D15.7)/5(5X,D15.7)/5(5X,D15.7)/
     *          5(5X,D15.7)/5(5X,D15.7)/5(5X,D15.7)/2(5X,D15.7)/)
      END IF
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                            ELEMS2                              *
C     ******************************************************************
      SUBROUTINE ELEMS2(AREA,C1,C2,C3,ELENGO,EMOD,GMOD,G,TR12,
     *                  TR22,TR13,TR23,TU,I,ZI2,ZI3,ZJ,KT,NPRINT,
     *                  QL,OLAMD)
C     COMPUTE THE COMPONENTS OF THE ELEMENT TANGENT STIFFNESS MATRIX
C     FOR THE FINITE ELEMENT MODEL.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),C1(1),C2(1),C3(1),ELENGO(1),EMOD(1),GMOD(1),
     *          G(1),TR12(1),TR22(1),TR13(1),TR23(1),TU(1),ZI2(1),
     *          ZI3(1),ZJ(1),KT(1),QL(1),OLAMD(3,3,60)
      A=AREA(I)
      E=EMOD(I)
      GGMOD=GMOD(I)
      EL=ELENGO(I)
      Z2=ZI2(I)
      Z3=ZI3(I)
      ZZJ=ZJ(I)
      CC1=C1(I)
      CC2=C2(I)
      CC3=C3(I)
      T1=T2=T3=T4=T5=T6=T7=T8=T9=T10=0.D0
C
C     COMPUTE THE CONTRIBUTION OF THE CONVENTIONAL LINEAR STIFFNESS
C     MATRIX IN THE LOCAL COORDINATE.
      IF (KT(1) .GT. 0) THEN DO
        T1=T1+E*A/EL
        T2=T2+12.D0*E*Z3/EL**3
        T3=T3+6.D0*E*Z3/EL**2
        T4=T4+12.D0*E*Z2/EL**3
        T5=T5-6.D0*E*Z2/EL**2
        T6=T6+GGMOD*ZZJ/EL
        T7=T7+4.D0*E*Z2/EL
        T8=T8+2.D0*E*Z2/EL
        T9=T9+4.D0*E*Z3/EL
        T10=T10+2.D0*E*Z3/EL
      END IF
C
C     COMPUTE THE CONTRIBUTION OF THE INITIAL STRESS (GEOMETRIC)
C     STIFFNESS MATRIX IN THE LOCAL COORDINATE.
      IF (KT(2) .GT. 0) THEN DO
        T2=T2+1.2D0*QL(I)/EL
        T3=T3+QL(I)/10.D0
        T4=T4+1.2D0*QL(I)/EL
        T5=T5-QL(I)/10.D0
        T7=T7+2.D0*QL(I)*EL/15.D0
        T8=T8-QL(I)*EL/30.D0
```

```
          T9=T9+2.D0*QL(I)*EL/15.D0
          T10=T10-QL(I)*EL/30.D0
       END IF
C
C      COMPUTE THE COMPONENTS OF THE GLOBAL ELEMENT STIFFNESS MATRIX.
       G(1)=T1*CC1**2+T2*OLAMD(2,1,I)**2+T4*OLAMD(3,1,I)**2
       G(2)=T1*CC1*CC2+T2*OLAMD(2,1,I)*OLAMD(2,2,I)
     *      +T4*OLAMD(3,1,I)*OLAMD(3,2,I)
       G(3)=T1*CC1*CC3+T2*OLAMD(2,1,I)*OLAMD(2,3,I)
     *      +T4*OLAMD(3,1,I)*OLAMD(3,3,I)
       G(4)=T3*OLAMD(2,1,I)*OLAMD(3,1,I)+T5*OLAMD(3,1,I)*OLAMD(2,1,I)
       G(5)=T3*OLAMD(2,1,I)*OLAMD(3,2,I)+T5*OLAMD(3,1,I)*OLAMD(2,2,I)
       G(6)=T3*OLAMD(2,1,I)*OLAMD(3,3,I)+T5*OLAMD(3,1,I)*OLAMD(2,3,I)
       G(7)=T3*OLAMD(2,1,I)*OLAMD(3,1,I)+T5*OLAMD(3,1,I)*OLAMD(2,1,I)
       G(8)=T3*OLAMD(2,1,I)*OLAMD(3,2,I)+T5*OLAMD(3,1,I)*OLAMD(2,2,I)
       G(9)=T3*OLAMD(2,1,I)*OLAMD(3,3,I)+T5*OLAMD(3,1,I)*OLAMD(2,3,I)
       G(10)=T1*CC2**2+T2*OLAMD(2,2,I)**2+T4*OLAMD(3,2,I)**2
       G(11)=T1*CC2*CC3+T2*OLAMD(2,2,I)*OLAMD(2,3,I)
     *       +T4*OLAMD(3,2,I)*OLAMD(3,3,I)
       G(12)=T3*OLAMD(2,2,I)*OLAMD(3,1,I)+T5*OLAMD(3,2,I)*
     *       OLAMD(2,1,I)
       G(13)=T3*OLAMD(2,2,I)*OLAMD(3,2,I)+T5*OLAMD(3,2,I)*
     *       OLAMD(2,2,I)
       G(14)=T3*OLAMD(2,2,I)*OLAMD(3,3,I)+T5*OLAMD(3,2,I)*
     *       OLAMD(2,3,I)
       G(15)=T3*OLAMD(2,2,I)*OLAMD(3,1,I)+T5*OLAMD(3,2,I)*
     *       OLAMD(2,1,I)
       G(16)=T3*OLAMD(2,2,I)*OLAMD(3,2,I)+T5*OLAMD(3,2,I)*
     *       OLAMD(2,2,I)
       G(17)=T3*OLAMD(2,2,I)*OLAMD(3,3,I)+T5*OLAMD(3,2,I)*
     *       OLAMD(2,3,I)
       G(18)=T1*CC3**2+T2*OLAMD(2,3,I)**2+T4*OLAMD(3,3,I)**2
       G(19)=T3*OLAMD(2,3,I)*OLAMD(3,1,I)+T5*OLAMD(3,3,I)*
     *       OLAMD(2,1,I)
       G(20)=T3*OLAMD(2,3,I)*OLAMD(3,2,I)+T5*OLAMD(3,3,I)*
     *       OLAMD(2,2,I)
       G(21)=T3*OLAMD(2,3,I)*OLAMD(3,3,I)+T5*OLAMD(3,3,I)*
     *       OLAMD(2,3,I)
       G(22)=T3*OLAMD(2,3,I)*OLAMD(3,1,I)+T5*OLAMD(3,3,I)*
     *       OLAMD(2,1,I)
       G(23)=T3*OLAMD(2,3,I)*OLAMD(3,2,I)+T5*OLAMD(3,3,I)*
     *       OLAMD(2,2,I)
       G(24)=T3*OLAMD(2,3,I)*OLAMD(3,3,I)+T5*OLAMD(3,3,I)*
     *       OLAMD(2,3,I)
       G(25)=T6*CC1**2+T9*OLAMD(3,1,I)**2+T7*OLAMD(2,1,I)**2
       G(26)=T6*CC1*CC2+T9*OLAMD(3,1,I)*OLAMD(3,2,I)
     *      +T7*OLAMD(2,1,I)*OLAMD(2,2,I)
       G(27)=T6*CC1*CC3+T9*OLAMD(3,1,I)*OLAMD(3,3,I)
     *      +T7*OLAMD(2,1,I)*OLAMD(2,3,I)
       G(28)=-T6*CC1**2+T10*OLAMD(3,1,I)**2+T8*OLAMD(2,1,I)**2
       G(29)=-T6*CC1*CC2+T10*OLAMD(3,1,I)*OLAMD(3,2,I)
     *      +T8*OLAMD(2,1,I)*OLAMD(2,2,I)
       G(30)=-T6*CC1*CC3+T10*OLAMD(3,1,I)*OLAMD(3,3,I)
     *      +T8*OLAMD(2,1,I)*OLAMD(2,3,I)
       G(31)=T6*CC2**2+T9*OLAMD(3,2,I)**2+T7*OLAMD(2,2,I)**2
       G(32)=T6*CC2*CC3+T9*OLAMD(3,2,I)*OLAMD(3,3,I)
     *      +T7*OLAMD(2,2,I)*OLAMD(2,3,I)
       G(33)=-T6*CC2**2+T10*OLAMD(3,2,I)**2+T8*OLAMD(2,2,I)**2
       G(34)=-T6*CC2*CC3+T10*OLAMD(3,2,I)*OLAMD(3,3,I)
     *      +T8*OLAMD(2,2,I)*OLAMD(2,3,I)
       G(35)=T6*CC3**2+T9*OLAMD(3,3,I)**2+T7*OLAMD(2,3,I)**2
       G(36)=-T6*CC3**2+T10*OLAMD(3,3,I)**2+T8*OLAMD(2,3,I)**2
       G(37)=T6*CC1**2+T9*OLAMD(3,1,I)**2+T7*OLAMD(2,1,I)**2
       G(38)=T6*CC1*CC2+T9*OLAMD(3,1,I)*OLAMD(3,2,I)
     *      +T7*OLAMD(2,1,I)*OLAMD(2,2,I)
       G(39)=T6*CC1*CC3+T9*OLAMD(3,1,I)*OLAMD(3,3,I)
```

```
     *        +T7*OLAMD(2,1,I)*OLAMD(2,3,I)
      G(40)=T6*CC2**2+T9*OLAMD(3,2,I)**2+T7*OLAMD(2,2,I)**2
      G(41)=T6*CC2*CC3+T9*OLAMD(3,2,I)*OLAMD(3,3,I)
     *        +T7*OLAMD(2,2,I)*OLAMD(2,3,I)
      G(42)=T6*CC3**2+T9*OLAMD(3,3,I)**2+T7*OLAMD(2,3,I)**2
      RETURN
      END
C
C
C
C     ****************************************************************
C     *                           ASSEMS                              *
C     ****************************************************************
      SUBROUTINE ASSEMS(G,MAXA,MCODE,SKT,I,NKT,NPRINT)
C     ASSEMBLE THE CONTRIBUTIONS FROM EACH ELEMENT INTO THE GLOBAL
C     SYSTEM STIFFNESS MATRIX.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION G(1),MAXA(1),MCODE(12,1),SKT(1)
      INTEGER INDEX(12,12)/1,2,3,4,5,6,-1,-2,-3,7,8,9,
     *                     2,10,11,12,13,14,-2,-10,-11,15,16,17,
     *                     3,11,18,19,20,21,-3,-11,-18,22,23,24,
     *                     4,12,19,25,26,27,-4,-12,-19,28,29,30,
     *                     5,13,20,26,31,32,-5,-13,-20,29,33,34,
     *                     6,14,21,27,32,35,-6,-14,-21,30,34,36,
     *                    -1,-2,-3,-4,-5,-6,1,2,3,-7,-8,-9,
     *                    -2,-10,-11,-12,-13,-14,2,10,11,-15,-16,-17,
     *                    -3,-11,-18,-19,-20,-21,3,11,18,-22,-23,-24,
     *                     7,15,22,28,29,30,-7,-15,-22,37,38,39,
     *                     8,16,23,29,33,34,-8,-16,-23,38,40,41,
     *                     9,17,24,30,34,36,-9,-17,-24,39,41,42/
      DO 20 JE=1,12
         J=MCODE(JE,I)
         IF (J .NE. 0) THEN DO
            DO 10 NE=1,JE
               N=MCODE(NE,I)
               IF (N .NE. 0) THEN DO
                  K= MAXA(J)+J-N
                  L=INDEX(NE,JE)
                  IF (L .GT .0) THEN DO
                     SKT(K)=SKT(K)+G(L)
                  ELSE DO
                     SKT(K)=SKT(K)-G(-L)
                  END IF
               END IF
   10       CONTINUE
         END IF
   20 CONTINUE
      IF (NPRINT .EQ. 3) THEN DO
         PRINT 100,I,(SKT(KI),KI=1,NKT)
  100    FORMAT(//'  ',7X,'STIFFNESS FROM ELEMENT ',I3,/6(5X,D15.7))
      END IF
      RETURN
      END
C
C
C
C     ****************************************************************
C     *                           STORE                               *
C     ****************************************************************
      SUBROUTINE STORE(RL1PI,RL2PI,RL3PI,ELENG,RL1,RL2,RL3,RL,C1PI,
     *                 C2PI,C3PI,C1,C2,C3,NE)
C     STORE THE PREVIOUS ITERATION LENGTH AND DIRECTION COSINES OF
C     ELEMENT I
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION RL1PI(1),RL2PI(1),RL3PI(1),ELENG(1),RL1(1),RL2(1),
     *          RL3(1),RL(1),C1PI(1),C2PI(1),C3PI(1),C1(1),C2(1),
```

```
     *             C3(1)
C
      DO 30 I=1,NE
         RL1PI(I)=RL1(I)
         RL2PI(I)=RL2(I)
         RL3PI(I)=RL3(I)
         ELENG(I)=RL(I)
         C1PI(I)=C1(I)
         C2PI(I)=C2(I)
         C3PI(I)=C3(I)
   30 CONTINUE
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                               FORCES                           *
C     ******************************************************************
      SUBROUTINE FORCES(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,DD,DE,
     *                  ELENG,EMOD,GMOD,F,FG,FL,MCODE,QT,R12,R22,
     *                  R13,R23,TORS,TR12,TR22,TR13,TR23,TTORS,SP12,
     *                  SP22,SP13,SP23,ST12,ST22,ST13,ST23,ZI2,ZI3,
     *                  ZJ,IMAX,NE,NEQ,Q,QI,C1PI,C2PI,C3PI,RM32,
     *                  RM33,QL,TU,INDTAN,NPRINT,RL1PI,RL2PI,RL3PI,
     *                  RL1,RL2,RL3,RL,TUL,TCB2,TCB3,DDJ,JCODE,AP,
     *                  APNEW,RJ,MINC,OL,CO1,CO2,CO3,DL,R,OLAMD,
     *                  OLANEW,DQQR,FQR,ELENGO,NJ,DDO,ITIND,NN)
C     FOR EACH ELEMENT CALL ELEMF.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),BP12(1),BP22(1),BP13(1),BP23(1),C1(1),C2(1),
     *          C3(1),DD(1),DE(1),ELENG(1),EMOD(1),GMOD(1),F(1),
     *          FG(12,1),FL(12,1),MCODE(12,1),QT(1),R12(1),R22(1),
     *          R13(1),R23(1),TORS(1),TR12(1),TR22(1),TR13(1),
     *          TR23(1),TTORS(1),SP12(1),SP22(1),SP13(1),SP23(1),
     *          ST12(1),ST22(1),ST13(1),ST23(1),ZI2(1),ZI3(1),ZJ(1),
     *          Q(1),C1PI(1),C2PI(1),C3PI(1),RM32(1),RM33(1),QL(1),
     *          U(60),RL1PI(1),RL2PI(1),RL3PI(1),RL1(1),RL2(1),
     *          RL3(1),RL(1),TUL(1),TCB2(1),TCB3(1),TU(1),DDJ(6,1),
     *          JCODE(6,1),AP(3,3,60),APNEW(3,3,60),RJ(3,1),
     *          MINC(2,1),OL(1),CO1(1),CO2(1),CO3(1),DL(1),
     *          R(3,3,60),OLAMD(3,3,60),OLANEW(3,3,60),ELENGO(1),
     *          DDO(1)
C
C
C     GENERATE THE INCREMENTAL NODAL DISPLACEMENT MATRIX
C     DDJ(IDIR,J).
      DO 28 J=1,NJ
         DO 12 IDIR=1,6
            DDJ(IDIR,J)=0.DO
            N=JCODE(IDIR,J)
            IF (N .NE. 0) THEN DO
               IF (NN .EQ. 1 .AND. ITIND .EQ. 1) THEN DO
                  DDJ(IDIR,J)=DDO(N)
               ELSE DO
                  DDJ(IDIR,J)=DD(N)
               END IF
            END IF
   12    CONTINUE
C
C        CALCULATE THE INCREMENTAL NODAL ROTATIONS AT NODE J.
         OME1=AP(1,1,J)*DDJ(4,J)+AP(1,2,J)*DDJ(5,J)
     *        +AP(1,3,J)*DDJ(6,J)
         OME2=AP(2,1,J)*DDJ(4,J)+AP(2,2,J)*DDJ(5,J)
     *        +AP(2,3,J)*DDJ(6,J)
         OME3=AP(3,1,J)*DDJ(4,J)+AP(3,2,J)*DDJ(5,J)
```

```
     *         +AP(3,3,J)*DDJ(6,J)
C
C        YIELD THE COMPONENTS OF THE ROTATION MATRIX OF NODE J.
         RJ(1,1)=1.D0
         RJ(1,2)=OME3
         RJ(1,3)=-OME2
         RJ(2,1)=-OME3
         RJ(2,2)=1.D0
         RJ(2,3)=OME1
         RJ(3,1)=OME2
         RJ(3,2)=-OME1
         RJ(3,3)=1.D0
C
C        FOR EATCH ITERATION EMPLOYING THE LAST NODAL ORIENTATION
C        AND ROTATION MATRICES OF NODE J TO GENERATE THE NEW
C        NODAL ORIENTATION MATRIX.
         DO 25 L=1,3
            DO 20 M=1,3
               APNEW(L,M,J)=0.D0
               DO 18 K=1,3
                  APNEW(L,M,J)=RJ(L,K)*AP(K,M,J)+APNEW(L,M,J)
   18          CONTINUE
   20       CONTINUE
   25    CONTINUE
         DO 27 L=1,3
            DO 26 K=1,3
               AP(L,K,J)=APNEW(L,K,J)
   26       CONTINUE
   27    CONTINUE
   28 CONTINUE
      DO 10 I=1,NEQ
         F(I)=0.D0
   10 CONTINUE
C
      I=1
      WHILE (I .LE. NE) DO
         IF (NPRINT .EQ. 3) THEN DO
            PRINT 100,I
  100       FORMAT(/'  ELEMENT ',I3)
         END IF
         CALL ELEMF(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,DD,DE,ELENG,
     *              EMOD,GMOD,F,FG,FL,MCODE,R12,R22,R13,R23,TORS,
     *              TR12,TR22,TR13,TR23,TTORS,SP12,SP22,SP13,SP23,
     *              ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,I,IMAX,C1PI,C2PI,
     *              C3PI,RM32,RM33,QL,TU,INDTAN,NPRINT,RL1PI,RL2PI,
     *              RL3PI,RL1,RL2,RL3,RL,TUL,TCB2,TCB3,DDJ,JCODE,AP,
     *              APNEW,RJ,MINC,OL,CO1,CO2,CO3,DL,R,OLAMD,
     *              OLANEW,DQQR,FQR,ELENGO,DDO,ITIND,NN)
         I=I+1
      END WHILE
      RETURN
      END
C
C
C
C     ****************************************************************
C     *                          ELEMF                               *
C     ****************************************************************
      SUBROUTINE ELEMF(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,DD,DE,
     *                 ELENG,EMOD,GMOD,F,FG,FL,MCODE,R12,R22,
     *                 R13,R23,TORS,TR12,TR22,TR13,TR23,TTORS,
     *                 SP12,SP22,SP13,SP23,ST12,ST22,ST13,ST23,
     *                 ZI2,ZI3,ZJ,I,IMAX,C1PI,C2PI,C3PI,RM32,RM33,
     *                 QL,TU,INDTAN,NPRINT,RL1PI,RL2PI,RL3PI,RL1,
     *                 RL2,RL3,RL,TUL,TCB2,TCB3,DDJ,JCODE,AP,APNEW,
     *                 RJ,MINC,OL,CO1,CO2,CO3,DL,R,OLAMD,OLANEW,
```

```
     *                    DQQR,FQR,ELENGO,DDO,ITIND,NN)
C     COMPUTE THE LOCAL INTERNAL ELEMENT FORCES,FL, DUE TO THE
C     DEFORMATIONS;  TRANSFORM THEM TO THE GLOBAL ELEMENT FORCES,
C     FG;  AND COMPUTE THE NODAL FORCES,F, CONTRIBUTED BY ELEMENT I.
C     SUBROUTINE ELEMF IS EXECUTED FOR EACH ELEMENT,I.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),BP12(1),BP22(1),BP13(1),BP23(1),C1(1),C2(1),
     *          C3(1),DD(1),DE(1),ELENG(1),EMOD(1),GMOD(1),F(1),
     *          FG(12,1),FL(12,1),MCODE(12,1),R12(1),R22(1),R13(1),
     *          R23(1),TORS(1),TR12(1),TR22(1),TR13(1),TR23(1),
     *          TTORS(1),SP12(1),SP22(1),SP13(1),SP23(1),ST12(1),
     *          ST22(1),ST13(1),ST23(1),ZI2(1),ZI3(1),ZJ(1),C1PI(1),
     *          C2PI(1),C3PI(1),RM32(1),RM33(1),QL(1),U(60),
     *          RL1PI(1),RL2PI(1),RL3PI(1),RL1(1),RL2(1),RL3(1),
     *          RL(1),TUL(1),TCB2(1),TCB3(1),TU(1),DDJ(6,1),
     *          JCODE(6,1),AP(3,3,60),APNEW(3,3,60),RJ(3,1),
     *          MINC(2,1),OL(1),C01(1),C02(1),C03(1),DL(1),
     *          R(3,3,60),OLAMD(3,3,60),OLANEW(3,3,60),
     *          ELENGO(1),DDO(1)
      IF (NPRINT .EQ. 3) THEN DO
         PRINT 100,I
 100     FORMAT(' DE FOR ELEMENT',I3)
      END IF
      DO 10 J=1,12
         DE(J)=0.D00
         K=MCODE(J,I)
         IF (K .NE. 0) THEN DO
            IF (NN .EQ. 1 .AND. ITIND .EQ. 1) THEN DO
               DE(J)=DDO(K)
            ELSE DO
               DE(J)=DD(K)
            END IF
         END IF
         IF (NPRINT.EQ.3) THEN DO
            PRINT 200,DE(J)
 200        FORMAT(5X,D15.7)
         END IF
  10  CONTINUE
C
      RL1PI(I)=ELENG(I)*C1PI(I)
      RL2PI(I)=ELENG(I)*C2PI(I)
      RL3PI(I)=ELENG(I)*C3PI(I)
      RL1(I)=RL1PI(I)+DE(7)-DE(1)
      RL2(I)=RL2PI(I)+DE(8)-DE(2)
      RL3(I)=RL3PI(I)+DE(9)-DE(3)
      RL(I)=DSQRT(RL1(I)**2+RL2(I)**2+RL3(I)**2)
C
      C1(I)=RL1(I)/RL(I)
      C2(I)=RL2(I)/RL(I)
      C3(I)=RL3(I)/RL(I)
C
C     GENERATE THE INCREMENTS OF THE LOCAL DISPLACEMENTS OF
C     ELEMENT I.
      DL(1)=OLAMD(1,1,I)*DE(1)+OLAMD(1,2,I)*DE(2)
     *        +OLAMD(1,3,I)*DE(3)
      DL(2)=OLAMD(2,1,I)*DE(1)+OLAMD(2,2,I)*DE(2)
     *        +OLAMD(2,3,I)*DE(3)
      DL(3)=OLAMD(3,1,I)*DE(1)+OLAMD(3,2,I)*DE(2)
     *        +OLAMD(3,3,I)*DE(3)
      DL(4)=OLAMD(1,1,I)*DE(4)+OLAMD(1,2,I)*DE(5)
     *        +OLAMD(1,3,I)*DE(6)
      DL(5)=OLAMD(2,1,I)*DE(4)+OLAMD(2,2,I)*DE(5)
     *        +OLAMD(2,3,I)*DE(6)
      DL(6)=OLAMD(3,1,I)*DE(4)+OLAMD(3,2,I)*DE(5)
     *        +OLAMD(3,3,I)*DE(6)
      DL(7)=OLAMD(1,1,I)*DE(7)+OLAMD(1,2,I)*DE(8)
```

```
     *      +OLAMD(1,3,I)*DE(9)
      DL(8)=OLAMD(2,1,I)*DE(7)+OLAMD(2,2,I)*DE(8)
     *      +OLAMD(2,3,I)*DE(9)
      DL(9)=OLAMD(3,1,I)*DE(7)+OLAMD(3,2,I)*DE(8)
     *      +OLAMD(3,3,I)*DE(9)
      DL(10)=OLAMD(1,1,I)*DE(10)+OLAMD(1,2,I)*DE(11)
     *       +OLAMD(1,3,I)*DE(12)
      DL(11)=OLAMD(2,1,I)*DE(10)+OLAMD(2,2,I)*DE(11)
     *       +OLAMD(2,3,I)*DE(12)
      DL(12)=OLAMD(3,1,I)*DE(10)+OLAMD(3,2,I)*DE(11)
     *       +OLAMD(3,3,I)*DE(12)
C
C     GENERATE THE COMPONENTS OF THE ROTATION MATRIX OF
C     ELEMENT I, R.
      R(1,1,I)=1.D0
      R(1,2,I)=(DL(8)-DL(2))/ELENG(I)
      R(1,3,I)=(DL(9)-DL(3))/ELENG(I)
      R(2,1,I)=-R(1,2,I)
      R(2,2,I)=1.D0
      R(2,3,I)=(DL(10)-DL(4))/2.D0
      R(3,1,I)=-R(1,3,I)
      R(3,2,I)=-R(2,3,I)
      R(3,3,I)=1.D0
C
C     FOR EACH ITERATION EMPLOYING THE LAST ELEMENT ORIENTATION
C     AND ROTATION MATRICES TO GENERATE THE NEW ELEMENT
C     ORIENTATION MATRIX.
      DO 34 L=1,3
         DO 32 J=1,3
            OLANEW(L,J,I)=0.D0
            DO 30 K=1,3
               OLANEW(L,J,I)=R(L,K,I)*OLAMD(K,J,I)+OLANEW(L,J,I)
   30       CONTINUE
   32    CONTINUE
   34 CONTINUE
      DO 37 L=1,3
         DO 36 J=1,3
            OLAMD(L,J,I)=OLANEW(L,J,I)
   36    CONTINUE
   37 CONTINUE
C
      UL=(ELENGO(I)-RL(I))/ELENGO(I)
C
C     EMPLOYING THE MINC CODE TO OBTAIN THE ELEMENT DEFORMATIONS
C     AT BOTH ENDS, A AND B.
      L=MINC(1,I)     New  Initial direction
      K=MINC(2,I)
      R12(I)=C1(I)*(-C03(I)/OL(I)*AP(1,1,L)+C01(I)/OL(I)*AP(3,1,L))
     *       +C2(I)*(-C03(I)/OL(I)*AP(1,2,L)+C01(I)/OL(I)*AP(3,2,L))
     *       +C3(I)*(-C03(I)/OL(I)*AP(1,3,L)+C01(I)/OL(I)*AP(3,3,L))
      R13(I)=-(C1(I)*(-C01(I)*C02(I)/OL(I)*AP(1,1,L)+OL(I)*AP(2,1,L)
     *           +(-C02(I)*C03(I)/OL(I))*AP(3,1,L))
     *       +C2(I)*(-C01(I)*C02(I)/OL(I)*AP(1,2,L)+OL(I)
     *           *AP(2,2,L)+(-C02(I)*C03(I)/OL(I))*AP(3,2,L))
     *       +C3(I)*(-C01(I)*C02(I)/OL(I)*AP(1,3,L)+OL(I)
     *           *AP(2,3,L)+(-C02(I)*C03(I)/OL(I))*AP(3,3,L)))
      R22(I)=C1(I)*(-C03(I)/OL(I)*AP(1,1,K)+C01(I)/OL(I)*AP(3,1,K))
     *       +C2(I)*(-C03(I)/OL(I)*AP(1,2,K)+C01(I)/OL(I)*AP(3,2,K))
     *       +C3(I)*(-C03(I)/OL(I)*AP(1,3,K)+C01(I)/OL(I)*AP(3,3,K))
      R23(I)=-(C1(I)*(-C01(I)*C02(I)/OL(I)*AP(1,1,K)+OL(I)*AP(2,1,K)
     *           +(-C02(I)*C03(I)/OL(I))*AP(3,1,K))
     *       +C2(I)*(-C01(I)*C02(I)/OL(I)*AP(1,2,K)+OL(I)
     *           *AP(2,2,K)+(-C02(I)*C03(I)/OL(I))*AP(3,2,K))
     *       +C3(I)*(-C01(I)*C02(I)/OL(I)*AP(1,3,K)+OL(I)
     *           *AP(2,3,K)+(-C02(I)*C03(I)/OL(I))*AP(3,3,K)))
      TORS(I)=2.D0*(OLAMD(3,1,I)*(-C01(I)*C02(I)/OL(I)*AP(1,1,K)
```

?

```
     *                   +OL(I)*AP(2,1,K)+(-CO2(I)*CO3(I)/OL(I))*
     *                   AP(3,1,K))+OLAMD(3,2,I)*(-CO1(I)*CO2(I)/OL(I)*
     *                   AP(1,2,K)+OL(I)*AP(2,2,K)+(-CO2(I)*CO3(I)/
     *                   OL(I))*AP(3,2,K))+OLAMD(3,3,I)*(-CO1(I)*CO2(I)/
     *                   OL(I)*AP(1,3,K)+OL(I)*AP(2,3,K)
     *                   +(-CO2(I)*CO3(I)/OL(I))*AP(3,3,K)))
C
C      COMPUTE THE TOTAL RELATIVE DEFORMATIONS OF ELEMENT I.
       TUL(I)=UL
       TR12(I)=R12(I)
       TR22(I)=R22(I)
       TR13(I)=R13(I)
       TR23(I)=R23(I)
       TTORS(I)=TORS(I)
C
C      COMPUTE THE LOCAL ELEMENT FORCES DUE TO THE RELATIVE
C      DEFORMATIONS BY BEAM-COLUMN MODEL.
       IF (INDTAN .EQ. 1) THEN DO
          CALL BOWCOR(AREA,BP12,BP22,BP13,BP23,ELENG,R12,R22,R13,R23,
     *                SP12,SP22,SP13,SP23,ST12,ST22,ST13,ST23,ZI2,
     *                ZI3,I,IMAX,TUL,EMOD,NPRINT,TCB2,TCB3,ELENGO,
     *                DQQR,FQR)
          RM12=EMOD(I)*ZI2(I)*(ST12(I)*TR12(I)+ST22(I)*TR22(I))/RL(I)
          RM22=EMOD(I)*ZI2(I)*(ST22(I)*TR12(I)+ST12(I)*TR22(I))/RL(I)
          RM32(I)=(RM12+RM22)/(ELENGO(I)**2)
C
          RM13=EMOD(I)*ZI3(I)*(ST13(I)*TR13(I)+ST23(I)*TR23(I))/RL(I)
          RM23=EMOD(I)*ZI3(I)*(ST23(I)*TR13(I)+ST13(I)*TR23(I))/RL(I)
          RM33(I)=(RM13+RM23)/(ELENGO(I)**2)
C
          FL(1,I)=EMOD(I)*AREA(I)*(TUL(I)-TCB2(I)-TCB3(I))
          QL(I)=FL(1,I)/ELENGO(I)
C
C      OR COMPUTE THE LOCAL ELEMENT FORCES DUE TO THE RELATIVE
C      DEFORMATIONS  BY FINITE ELEMENT MODEL.
       ELSE DO
          U(I)=RL(I)-ELENG(I)
          TU(I)=TU(I)+U(I) <---- TOTAL Axial Defor.
          E1=TU(I)
          E12=TR12(I)
          E22=TR22(I)
          E13=TR13(I)
          E23=TR23(I)
          A=AREA(I)
          E=EMOD(I)
          EL=RL(I)
          Z2=ZI2(I)
          Z3=ZI3(I)                                  (see (4.15)
C
C         PRINT 38,I,E1,E12,E22
C  38     FORMAT(2X,I3,3X,3(2X,D15.7))
          RM12=E*A*(4.D0*E12-E22)*E1/60.D0+2.D0*E*Z2*(2.D0*E12+E22)/
     *         EL+E*A*E1*E12/15.D0+E*A*EL*((12.D0*E12**2-3.D0*E12*E22
     *         +E22**2)*E12/420.D0+(-3.D0*E12**2+4.D0*E12*E22-3.D0*
     *         E22**2)*E22/840.D0)-E*A*E1*E22/60.D0
          RM22=E*A*(4.D0*E22-E12)*E1/60.D0+2.D0*E*Z2*(2.D0*E22+E12)/
     *         EL-E*A*E1*E12/60.D0+E*A*EL*((12.D0*E22**2-3.D0*E12*E22
     *         +E12**2)*E22/420.D0+(-3.D0*E12**2+4.D0*E12*E22-3.D0*
     *         E22**2)*E12/840.D0)+E*A*E1*E22/15.D0
          RM13=E*A*(4.D0*E13-E23)*E1/60.D0+2.D0*E*Z3*(2.D0*E13+E23)/
     *         EL+E*A*E1*E13/15.D0+E*A*EL*((12.D0*E13**2-3.D0*E13*E23
     *         +E23**2)*E13/420.D0+(-3.D0*E13**2+4.D0*E13*E23-3.D0*
     *         E23**2)*E23/840.D0)-E*A*E1*E23/60.D0
          RM23=E*A*(4.D0*E23-E13)*E1/60.D0+2.D0*E*Z3*(2.D0*E23+E13)/
     *         EL-E*A*E1*E13/60.D0+E*A*EL*((12.D0*E23**2-3.D0*E13*E23
     *         +E13**2)*E23/420.D0+(-3.D0*E13**2+4.D0*E13*E23-3.D0*
```

```
     *         E23**2)*E13/840.D0)+E*A*E1*E23/15.D0
          FL(1,I)=(-E*A)*(E1/EL+(4.D0*E13-E23)*E13/60.D0
     *            +(4.D0*E23-E13)*E23/60.D0+(4.D0*E12-E22)*E12/60.D0
     *            +(4.D0*E22-E12)*E22/60.D0)
          QL(I)=-FL(1,I)
        END IF
C
C       COMPUTE THE REMAINING LOCAL ELEMENT FORCES.
        FL(2,I)=(RM13+RM23)/RL(I)
        FL(3,I)=-(RM12+RM22)/RL(I)
        FL(4,I)=-GMOD(I)*ZJ(I)*TTORS(I)/RL(I)
        FL(5,I)=RM12
        FL(6,I)=RM13
        FL(7,I)=-FL(1,I)
        FL(8,I)=-FL(2,I)
        FL(9,I)=-FL(3,I)
        FL(10,I)=-FL(4,I)
        FL(11,I)=RM22
        FL(12,I)=RM23
C
        IF (NPRINT .EQ. 3) THEN DO
          PRINT 40,I,RL(I),RM12,RM22,RM13,RM23,ST12(I),ST22(I),
     *             ST13(I),ST23(I),TCB2(I),TCB3(I),TTORS(I),TR12(I),
     *             TR22(I),TR13(I),TR23(I),RL1PI(I),RL2PI(I),
     *             RL3PI(I),RL1(I),RL2(I),RL3(I),TU(I),ELENG(I)
  40      FORMAT(/'  ELEMENT ',I3,'          RL= ',D15.7/
     *      '         RM12= ',D15.7,'  RM22= ',D15.7/
     *      '         RM13= ',D15.7,'  RM23= ',D15.7/
     *      '         ST12= ',D15.7,'  ST22= ',D15.7/
     *      '         ST13= ',D15.7,'  ST23= ',D15.7/
     *      '         TCB2= ',D15.7,'  TCB3= ',D15.7,'  TTORS= ',D15.7/
     *      '         TR12= ',D15.7,'  TR22= ',D15.7/
     *      '         TR13= ',D15.7,'  TR23= ',D15.7/
     *      '        RL1PI= ',D15.7,' RL2PI= ',D15.7,' RL3PI= ',D15.7/
     *      '          RL1= ',D15.7,'   RL2= ',D15.7,'   RL3= ',D15.7/
     *      '           TU= ',D15.7,' ELENG= ',D15.7/)
          PRINT 50
  50      FORMAT('  LOCAL ELEMENT FORCES')
          PRINT 60,FL(1,I),FL(2,I),FL(3,I),FL(4,I),FL(5,I),FL(6,I),
     *             FL(7,I),FL(8,I),FL(9,I),FL(10,I),FL(11,I),FL(12,I)
  60      FORMAT(5X,D15.7,5X,D15.7,5X,D15.7/5X,D15.7,5X,D15.7,5X,
     *           D15.7/5X,D15.7,5X,D15.7,5X,D15.7/5X,D15.7,5X,D15.7,
     *           5X,D15.7/)
        END IF
C
C       TRANSFORM TO THE GLOBAL ELEMENT FORCES.
        FG(1,I)=C1(I)*FL(1,I)+OLAMD(2,1,I)*FL(2,I)
     *                       +OLAMD(3,1,I)*FL(3,I)
        FG(2,I)=C2(I)*FL(1,I)+OLAMD(2,2,I)*FL(2,I)
     *                       +OLAMD(3,2,I)*FL(3,I)
        FG(3,I)=C3(I)*FL(1,I)+OLAMD(2,3,I)*FL(2,I)
     *                       +OLAMD(3,3,I)*FL(3,I)
        FG(4,I)=C1(I)*FL(4,I)+OLAMD(2,1,I)*FL(5,I)
     *                       +OLAMD(3,1,I)*FL(6,I)
        FG(5,I)=C2(I)*FL(4,I)+OLAMD(2,2,I)*FL(5,I)
     *                       +OLAMD(3,2,I)*FL(6,I)
        FG(6,I)=C3(I)*FL(4,I)+OLAMD(2,3,I)*FL(5,I)
     *                       +OLAMD(3,3,I)*FL(6,I)
        FG(7,I)=C1(I)*FL(7,I)+OLAMD(2,1,I)*FL(8,I)
     *                       +OLAMD(3,1,I)*FL(9,I)
        FG(8,I)=C2(I)*FL(7,I)+OLAMD(2,2,I)*FL(8,I)
     *                       +OLAMD(3,2,I)*FL(9,I)
        FG(9,I)=C3(I)*FL(7,I)+OLAMD(2,3,I)*FL(8,I)
     *                       +OLAMD(3,3,I)*FL(9,I)
        FG(10,I)=C1(I)*FL(10,I)+OLAMD(2,1,I)*FL(11,I)
     *         +OLAMD(3,1,I)*FL(12,I)
```

```
      FG(11,I)=C2(I)*FL(10,I)+OLAMD(2,2,I)*FL(11,I)
     *         +OLAMD(3,2,I)*FL(12,I)
      FG(12,I)=C3(I)*FL(10,I)+OLAMD(2,3,I)*FL(11,I)
     *         +OLAMD(3,3,I)*FL(12,I)
C
      DO 80 J=1,12
         K=MCODE(J,I)
         IF (K .NE. 0) THEN DO
            F(K)=F(K)+FG(J,I)
            IF (NPRINT .EQ. 3) THEN DO
               PRINT 70,F(K)
   70          FORMAT('  F(K)= ',D15.7)
            END IF
         END IF
   80 CONTINUE
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                            BOWCOR                              *
C     ******************************************************************
      SUBROUTINE BOWCOR(AREA,BP12,BP22,BP13,BP23,ELENG,R12,R22,
     *                  R13,R23,SP12,SP22,SP13,SP23,ST12,ST22,
     *                  ST13,ST23,ZI2,ZI3,I,IMAX,TUL,EMOD,NPRINT,
     *                  TCB2,TCB3,ELENGO,DQQR,FQR)
C     CALCULATE THE STABILITY FUNCTIONS,ST12,ST22,ST13,ST23; THE
C     BOWING FUNCTIONS,BW12,BW22,BW13,BW23; THE LENGTH CORRECTION
C     FACTORS FOR THE BOWING ACTIONS,TCB2,TCB3; AND THE AXIAL
C     FORCE, QPRINT.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),BP12(1),BP22(1),BP13(1),BP23(1),ELENG(1),
     *          R12(1),R22(1),R13(1),R23(1),SP12(1),SP22(1),SP13(1),
     *          SP23(1),ST12(1),ST22(1),ST13(1),ST23(1),ZI2(1),
     *          ZI3(1),EMOD(1),TUL(1),TCB2(1),TCB3(1),ELENGO(1)
      PI=3.141592653589793
      QR=TUL(I)*AREA(I)*ELENGO(I)**2/(ZI2(I)*PI**2)
C     QR=0.D0
      IC=0
C
      WHILE (IC .LT. IMAX) DO
         IF (QR .GT. 0.D00) THEN DO
            W2=QR*PI**2
            W2=DSQRT(W2)
            ST12(I)=W2*(DSIN(W2)-W2*DCOS(W2))/(2.D00-2.D00*DCOS(W2)
     *              -W2*DSIN(W2))
            ST22(I)=W2*(W2-DSIN(W2))/(2.D00-2.D00*DCOS(W2)-
     *              W2*DSIN(W2))
C
            W3=QR*PI**2*ZI2(I)/ZI3(I)
            W3=DSQRT(W3)
            ST13(I)=W3*(DSIN(W3)-W3*DCOS(W3))/(2.D00-2.D00*DCOS(W3)
     *              -W3*DSIN(W3))
            ST23(I)=W3*(W3-DSIN(W3))/(2.D00-2.D00*DCOS(W3)-
     *              W3*DSIN(W3))
         END IF
C
         IF (QR .EQ. 0.D00) THEN DO
            ST12(I)=4.D00
            ST22(I)=2.D00
            SP12(I)=0.D00
            SP22(I)=0.D00
            BP12(I)=0.D00
            BP22(I)=0.D00
            TCB2(I)=TUL(I)/2.D0
```

```
C
            ST13(I)=4.D00
            ST23(I)=2.D00
            SP13(I)=0.D00
            SP23(I)=0.D00
            BP13(I)=0.D00
            BP23(I)=0.D00
            TCB3(I)=TUL(I)/2.D0
            RETURN
         END IF
C
         IF (QR .LT. 0.D00) THEN DO
            W2=-QR*PI**2
            W2=DSQRT(W2)
            ST12(I)=W2*(W2*DCOSH(W2)-DSINH(W2))/
     *              (2.D0-2.D0*DCOSH(W2)+W2*DSINH(W2))
            ST22(I)=W2*(DSINH(W2)-W2)/(2.D00-2.D00*DCOSH(W2)
     *              +W2*DSINH(W2))
C
            W3=-QR*PI**2*ZI2(I)/ZI3(I)
            W3=DSQRT(W3)
            ST13(I)=W3*(W3*DCOSH(W3)-DSINH(W3))/
     *              (2.D0-2.D0*DCOSH(W3)+W3*DSINH(W3))
            ST23(I)=W3*(DSINH(W3)-W3)/(2.D00-2.D00*DCOSH(W3)
     *              +W3*DSINH(W3))
         END IF
C
         BW12=(ST12(I)+ST22(I))*(ST22(I)-2.D00)/(8.D00*QR*PI**2)
         BW22=ST22(I)/(8.D00*(ST12(I)+ST22(I)))
         SP12(I)=-2.D00*PI**2*(BW12+BW22)
         SP22(I)=-2.D00*PI**2*(BW12-BW22)
         BP12(I)=(QR*(ST12(I)*SP22(I)+ST22(I)*SP12(I)+2.D0*ST22(I)*
     *            SP22(I)-2.D0*SP12(I)-2.D0*SP22(I))-(ST12(I)
     *            +ST22(I))*(ST22(I)-2.D0))/(8.D0*PI**2*QR**2)
         BP22(I)=(SP22(I)*(ST12(I)+ST22(I))-ST22(I)*(SP12(I)
     *            +SP22(I)))/(8.D0*(ST12(I)+ST22(I))**2)
         TCB2(I)=BW12*(R12(I)+R22(I))**2+BW22*(R12(I)-R22(I))**2
         TCB2P=BP12(I)*(R12(I)+R22(I))**2+BP22(I)*(R12(I)-R22(I))**2
C
         BW13=(ST13(I)+ST23(I))*(ST23(I)-2.D00)*ZI3(I)/
     *        (8.D0*QR*PI**2*ZI2(I))
         BW23=ST23(I)/(8.D00*(ST13(I)+ST23(I)))
         SP13(I)=-2.D00*PI**2*(BW13+BW23)
         SP23(I)=-2.D00*PI**2*(BW13-BW23)
         BP13(I)=(QR*ZI2(I)/ZI3(I)*(ST13(I)*SP23(I)+ST23(I)*SP13(I)
     *            +2.D0*ST23(I)*SP23(I)-2.D0*SP13(I)-2.D0*SP23(I))-
     *           (ST13(I)+ST23(I))*(ST23(I)-2.D0))/
     *           (8.D0*PI**2*(QR*ZI2(I)/ZI3(I))**2)
         BP23(I)=(SP23(I)*(ST13(I)+ST23(I))-ST23(I)*(SP13(I)
     *            +SP23(I)))/(8.D0*(ST13(I)+ST23(I))**2)
         TCB3(I)=BW13*(R13(I)+R23(I))**2+BW23*(R13(I)-R23(I))**2
         TCB3P=BP13(I)*(R13(I)+R23(I))**2+BP23(I)*(R13(I)-R23(I))**2
C
         RLA=ELENGO(I)/(DSQRT(ZI2(I)/AREA(I)))
         FQRC=DABS(QR-RLA**2*(TUL(I)-TCB2(I)-TCB3(I))/PI**2)
         DQ=-(QR-RLA**2*(TUL(I)-TCB2(I)-TCB3(I))/PI**2)/
     *       (1.D0+RLA**2*(TCB2P+TCB3P*ZI2(I)/ZI3(I))/PI**2)
C
         IF (NPRINT .EQ. 3) THEN DO
           PRINT 100,ST12(I),ST22(I),BW12,BW22,SP12(I),SP22(I),
     *               BP12(I),BP22(I),TCB2(I),ST13(I),ST23(I),
     *               BW13,BW23,SP13(I),SP23(I),BP13(I),BP23(I),
     *               TCB3(I)
  100      FORMAT('    ST12=',D15.7,'   ST22=',D15.7/
     *            '    BW12=',D15.7,'   BW22=',D15.7/
     *            '    SP12=',D15.7,'   SP22=',D15.7/
```

```
     *              '  BP12=',D15.7,'   BP22=',D15.7,'   TCB2=',D15.7//
     *              '  ST13=',D15.7,'   ST23=',D15.7/
     *              '  BW13=',D15.7,'   BW23=',D15.7/
     *              '  SP13=',D15.7,'   SP23=',D15.7/
     *              '  BP13=',D15.7,'   BP23=',D15.7,'   TCB3=',D15.7/)
         END IF
C
         QR=QR+DQ
         DQQRC=DABS(DQ/QR)
         IC=IC+1
         QPRINT=EMOD(I)*AREA(I)*(TUL(I)-TCB2(I)-TCB3(I))
         IF (NPRINT .EQ. 3) THEN DO
            PRINT 200,IC,QPRINT
  200       FORMAT('  ITERATION ',I4,'  Q= ',D15.7)
         END IF
         IF (DQQRC .LE. DQQR .AND. FQRC .LE. FQR) THEN DO
            IF (NPRINT .EQ. 3) THEN DO
               PRINT 300,IC,I,FQRC,DQQRC
  300          FORMAT(' BOWCOR CONVERGED IN ',I5,' ITERATIONS FOR
     *                ELEMENT',I5,'  FQRC=',D15.7,'  DQQRC=',D15.7)
            END IF
            IC=IMAX+1
         END IF
      END WHILE
C     IF (DQQRC .GT. DQQR .OR. FQRC .GT. FQR) THEN DO
C        PRINT 400,I
C 400    FORMAT(' NO CONVERGENCE IN BOWCOR FOR ELEMENT ',I3)
C        STOP
C     END IF
      RETURN
      END
C
C
C
C     ****************************************************************
C     *                           SOLVE                              *
C     ****************************************************************
      SUBROUTINE SOLVE(DD,F,MAXA,QT,SKT,NC,NEQ,NEGPIV,NKT,NPRINT)
C     CALL FACTOR IF THE STIFFNESS MATRIX HAS BEEN UPDATED; CALL
C     REDUCE AND BACSUB TO SOLVE THE SIMULTANEOUS LINEAR SYSTEM
C     EQUATIONS FOR THE NODAL DISPLACEMENTS BY GAUSS ELIMINATION.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DD(1),F(1),MAXA(1),QT(1),SKT(1)
      IF (NPRINT .EQ. 3) THEN DO
         PRINT 100
  100    FORMAT(////' SOLVE CALLED')
      END IF
C
      IF (NC .EQ. 0) THEN DO
         CALL FACTOR(MAXA,SKT,NEQ,NEGPIV,NPRINT)
         IF (NPRINT .EQ. 3) THEN DO
            PRINT 200
  200       FORMAT(/'  FACTORED STIFFNESS MATRIX')
            PRINT 300,(SKT(I),I=1,NKT)
  300       FORMAT(' ',6(5X,D15.7))
         END IF
      END IF
C
      DO 10 I=1,NEQ
         DD(I)=QT(I)-F(I)
   10 CONTINUE
      CALL REDUCE(DD,MAXA,SKT,NEQ)
      CALL BACSUB(DD,MAXA,SKT,NEQ)
      RETURN
      END
C
```

```
C
C
C     ****************************************************************
C     *                            FACTOR                            *
C     ****************************************************************
      SUBROUTINE FACTOR(MAXA,SKT,NEQ,NEGPIV,NPRINT)
C     PERFORM L*D*L(T) FACTORIZATION (DECOMPOSITION) OF STIFFNESS
C     MATRIX SKT.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION MAXA(1),SKT(1)
      NEGPIV=0
      IF (NPRINT .EQ. 3) THEN DO
         PRINT 100
  100    FORMAT(/'   FACTOR CALLED   DIAGONALS OF FACTORIZATION:')
      END IF
      DO 80 N=1,NEQ
         KN=MAXA(N)
         KL=KN+1
         KU=MAXA(N+1)-1
         KH=KU-KL
         IF (KH) 70,50,10
   10    K=N-KH
         IC=0
         KLT=KU
         DO 40 J=1,KH
            IC=IC+1
            KLT=KLT-1
            KI=MAXA(K)
            ND=MAXA(K+1)-KI-1
            IF (ND) 40,40,20
   20       KK=MIN0(IC,ND)
            C=0.D00
            DO 30 L=1,KK
   30          C=C+SKT(KI+L)*SKT(KLT+L)
            SKT(KLT)=SKT(KLT)-C
   40       K=K+1
   50    K=N
         B=0.D00
         DO 60 KK=KL,KU
            K=K-1
            KI=MAXA(K)
            C=SKT(KK)/SKT(KI)
            B=B+C*SKT(KK)
   60       SKT(KK)=C
         SKT(KN)=SKT(KN)-B
   70    IF (SKT(KN) .EQ. 0.D00) THEN DO
            PRINT 200,N,SKT(KN)
  200       FORMAT('  STIFFNESS MATRIX IS NOT POSITIVE DEFINITE'/
     *             '  PIVOT IS ZERO FOR D.O.F. ',I4/'0PIVOT = ',
     *             D15.8)
            STOP
         END IF
         IF (SKT(KN) .LT. 0.D00) THEN DO
            NEGPIV=NEGPIV+1
            IF (NPRINT .EQ. 3) THEN DO
               PRINT 300,NEGPIV
  300          FORMAT(/'   NEGATIVE PIVOT ENCOUNTERED IN FACTOR',I5)
            END IF
         END IF
         IF (NPRINT .EQ. 3) THEN DO
            PRINT 400,SKT(KN)
  400       FORMAT(/'   SKT(KN)= ',D15.7)
         END IF
   80 CONTINUE
      RETURN
      END
```

```
C
C
C
C     ******************************************************************
C     *                             REDUCE                             *
C     ******************************************************************
      SUBROUTINE REDUCE(DD,MAXA,SKT,NEQ)
C     REDUCE THE RIGHT HAND SIDE LOAD VECTOR.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DD(1),MAXA(1),SKT(1)
      DO 20 N=1,NEQ
         KL=MAXA(N)+1
         KU=MAXA(N+1)-1
         KH=KU-KL
         IF (KH .GE. 0) THEN DO
            K=N
            C=0.D00
            DO 10 KK=KL,KU
               K=K-1
   10          C=C+SKT(KK)*DD(K)
            DD(N)=DD(N)-C
         END IF
   20 CONTINUE
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                             BACSUB                             *
C     ******************************************************************
      SUBROUTINE BACSUB(DD,MAXA,SKT,NEQ)
C     BACK SUBSTITUTE TO OBTAIN THE SOLUTION.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DD(1),MAXA(1),SKT(1)
      DO 10 N=1,NEQ
         K=MAXA(N)
         DD(N)=DD(N)/SKT(K)
   10 CONTINUE
      IF (NEQ .EQ. 1) RETURN
      N=NEQ
      DO 30 L=2,NEQ
         KL=MAXA(N)+1
         KU=MAXA(N+1)-1
         KH=KU-KL
         IF (KH .GE. 0)THEN DO
            K=N
            DO 20 KK=KL,KU
               K=K-1
               DD(K)=DD(K)-SKT(KK)*DD(N)
   20       CONTINUE
         END IF
         N=N-1
   30 CONTINUE
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                              TEST                              *
C     ******************************************************************
      SUBROUTINE TEST(AREA,BP12,BP22,BP13,BP23,C1,C2,C3,D,DD,DD1,DE,
     *                ELENG,EMOD,GMOD,F,FG,FL,FP,FPI,JCODE,MCODE,
     *                QT,R12,R22,R13,R23,TORS,SP12,SP22,SP13,SP23,
     *                ST12,ST22,ST13,ST23,ZI2,ZI3,ZJ,CPDB,CPDC,
```

```
     *                 CPF,ICI,IMAX,NE,NEQ,NJ,Q,QI,CI1,CI2,CI3,
     *                 ITIND,NPRINT)
C     AT THE END OF EACH ITERATION, CHECK TO SEE IF HAS CONVERGED
C     TO AN EQUILIBRIUM POINT WITHIN PRESET TOLERANCES OR WHETHER
C     THE ITERATION IS DIVERGING.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AREA(1),BP12(1),BP22(1),BP13(1),BP23(1),C1(1),C2(1),
     *          C3(1),D(1),DD(1),DD1(1),DE(1),ELENG(1),EMOD(1),
     *          GMOD(1),F(1),FG(12,1),FL(12,1),FP(1),FPI(1),
     *          JCODE(6,1),MCODE(12,1),QT(1),R12(1),R22(1),R13(1),
     *          R23(1),TORS(1),SP12(1),SP22(1),SP13(1),SP23(1),
     *          ST12(1),ST22(1),ST13(1),ST23(1),ZI2(1),ZI3(1),ZJ(1),
     *          Q(1),CI1(1),CI2(1),CI3(1)
      ICI=0
C
      IF (CPDC .LT. 1.D00) THEN DO
         CALL DISPLC(D,DD,JCODE,CPDC,ICI,NJ)
      END IF
C
      IF (CPDB .LT. 1.D00) THEN DO
         CALL DISPLB(D,DD,CPDB,ICI,NEQ)
      END IF
C
      IF (CPF.LT. 1.D00) THEN DO
         CALL UNBALF(F,FP,QT,CPF,ICI,NEQ)
      END IF
C
      IF (NPRINT .EQ. 3) THEN DO
         PRINT 100,ICI
  100    FORMAT(//' ICI= ',I5)
      END IF
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                          DISPLC                                *
C     ******************************************************************
      SUBROUTINE DISPLC(D,DD,JCODE,CPDC,ICI,NJ)
C     CHECK TO SEE IF THE RATIO: THE MAX CHANGE IN TRANSLATION (OR
C     ROTATION) TO THE LARGEST TOTAL NODAL DISPLACEMENT (OR
C     ROTATION) IN STRUCTURE, IS WITHIN A DISPLACEMENT
C     CONVERGENCE TOLERANCE, CPDC.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION D(1),DD(1),JCODE(6,1)
      RMT=0.D00
      RMR=0.D00
      CR=0.D00
      CT=0.D00
      I=1
C
      WHILE (I .LE. 6) DO
         J=1
         WHILE (J .LE. NJ) DO
            K=JCODE(I,J)
            IF (K .NE. 0) THEN DO
               IF (I .GE. 4) THEN DO
                  RMR=DMAX1(DABS(D(K)),RMR)
                  CR=DMAX1(DABS(DD(K)),CR)
               ELSE DO
                  RMT=DMAX1(DABS(D(K)),RMT)
                  CT=DMAX1(DABS(DD(K)),CT)
               END IF
            END IF
            J=J+1
```

```
            END WHILE
            I=I+1
         END WHILE
C
         IF (RMR .NE. 0.D00) THEN DO
            CPR=CR/RMR
            IF (CPR .GT. CPDC) THEN DO
               ICI=ICI+1
            END IF
         END IF
         IF (RMT .NE. 0.D00) THEN DO
            CPT=CT/RMT
            IF (CPT .GT. CPDC) THEN DO
               ICI=ICI+1
            END IF
         END IF
C
         IF (RMR .EQ. 0.D00 .AND. RMT .EQ. 0.D00) THEN DO
            ICI=ICI+1
            PRINT 100,CR,RMR,CT,CMT
  100       FORMAT(/' MAX D IN DISPLC = 0  CR=',D15.7,'RMR=',D15.7,
     *              ' CT=',D15.7,' CMT=',D15.7)
            STOP
         END IF
         RETURN
         END
C
C
C
C        ****************************************************************
C        *                           DISPLB                             *
C        ****************************************************************
         SUBROUTINE DISPLB(D,DD,CPDB,ICI,NEQ)
C        CHECK TO SEE IF THE RATIO: THE DISPLACEMENT INCREMENT DURING
C        EACH ITERATION TO THE DISPLACEMENT AT THE END OF EACH
C        ITERATION, IS WITHIN A DISPLACEMENT CONVERGENCE
C        TOLERANCE, CPDB.
         IMPLICIT REAL*8(A-H,O-Z)
         DIMENSION D(1),DD(1)
         CN=0.D00
         CD=0.D00
         DO 10 I=1,NEQ
            CN=CN+(DD(I))**2
            CD=CD+(D(I))**2
   10    CONTINUE
         IF (CD .EQ. 0.D00) THEN DO
            ICI=ICI+10
            RETURN
         END IF
         C=(DSQRT(CN))/(DSQRT(CD))
         IF (C .GT. CPDB) THEN DO
            ICI=ICI+10
         END IF
         RETURN
         END
C
C
C
C        ****************************************************************
C        *                           UNBALF                             *
C        ****************************************************************
         SUBROUTINE UNBALF(F,FP,QT,CPF,ICI,NEQ)
C        CHECK TO SEE IF THE RATIO: THE UNBALANCED FORCE DURING EACH
C        ITERATION TO THE ORIGINAL LOAD INCREMENT, IS WITHIN A PRESET
C        TOLERANCE,CPF.
         IMPLICIT REAL*8(A-H,O-Z)
```

```
      DIMENSION F(1),FP(1),QT(1)
      CN=0.D00
      CD=0.D00
      DO 10 I=1,NEQ
         CN=CN+(QT(I)-F(I))**2
         CD=CD+(QT(I)-FP(I))**2
   10 CONTINUE
      IF (CD .EQ. 0.D00) THEN DO
         ICI=ICI+100
         RETURN
      END IF
      C=(DSQRT(CN))/(DSQRT(CD))
      IF (C .GT. CPF) THEN DO
         ICI=ICI+100
      END IF
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                          RESULT                                *
C     ******************************************************************
      SUBROUTINE RESULT(C1,C2,C3,D,DJ,ELENG,EMOD,GMOD,FG,FL,
     *                  JCODE,MCODE,MINC,P,Q,QT,ZI2,ZI3,
     *                  ZJ,NE,NEQ,NJ,QI,IMP,NPRINT)
C     INITIALIZE THE NODAL FORCE MATRIX ,P,TO ZERO; CALL JOINTF
C     AND OUTPUT.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION C1(1),C2(1),C3(1),D(1),DJ(6,1),ELENG(1),EMOD(1),
     *          GMOD(1),FG(12,1),FL(12,1),JCODE(6,1),MCODE(12,1),
     *          MINC(2,1),P(6,1),Q(1),QT(1),ZI2(1),ZI3(1),ZJ(1)
      DO 20 J=1,NJ
         DO 10 I=1,6
            P(I,J)=0.D00
   10    CONTINUE
   20 CONTINUE
      CALL JOINTF(FG,MINC,P,NE)
      CALL OUTPUT(D,DJ,FL,JCODE,P,QT,NE,NEQ,NJ,QI,IMP,NPRINT)
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                          JOINTF                                *
C     ******************************************************************
      SUBROUTINE JOINTF(FG,MINC,P,NE)
C     ASSIGN THE GLOBAL ELEMENT FORCES TO THE NODAL FORCE MATRIX,P.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION FG(12,1),MINC(2,1),P(6,1)
      DO 20 I=1,NE
         J=MINC(1,I)
         K=MINC(2,I)
         DO 10 L=1,6
            P(L,J)=P(L,J)+FG(L,I)
            P(L,K)=P(L,K)+FG(L+6,I)
   10    CONTINUE
   20 CONTINUE
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                          OUTPUT                                *
```

```
C     ******************************************************************
      SUBROUTINE OUTPUT(D,DJ,FL,JCODE,P,QT,NE,NEQ,NJ,QI,IMP,NPRINT)
C     PRINT THE NODAL DISPLACEMENTS (INCLUDING NODAL CONSTRAINTS),
C     DJ; PRINT THE LOCAL ELEMENT FORCES,FL; PRINT THE NODAL FORCES,
C     P; PRINT THE LOAD VERSUS DISPLACEMENT VECTOR,QT AND D.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION D(1),DJ(6,1),FL(12,1),JCODE(6,1),P(6,1),QT(1)
      DO 20 J=1,NJ
         DO 10 I=1,6
            DJ(I,J)=0.D00
            K=JCODE(I,J)
            IF (K .NE. 0) THEN DO
               DJ(I,J)=D(K)
            END IF
   10    CONTINUE
   20 CONTINUE
C
      IF (NPRINT .LE. 1) THEN DO
         PRINT 100,QI,D(IMP)
  100    FORMAT('   ',F17.9/'    ',F17.9)
      END IF
C
      IF (NPRINT .EQ. 2 .OR. NPRINT .EQ. 3) THEN DO
         PRINT 200,QI,D(IMP)
  200    FORMAT('   QI EQUILIB =',F17.9,'   D EQUILIB =',F17.9)
      END IF
C
      IF (NPRINT .EQ. 4) THEN DO
         PRINT 300,QI
  300    FORMAT('-',80('_')//31X,'QI=',F15.7//34X,'LOAD VECTOR'
     *      /34X,11('*'))
         DO 30 I=1,NEQ
            PRINT 400,QT(I)
  400       FORMAT('0',31X,F16.5)
   30    CONTINUE
         PRINT 500
  500    FORMAT('-',26X,'GLOBAL NODAL DISPLACEMENTS'/27X,26('*')
     *          //10X,' NODE',7X,'1-DIRECTION',8X,'2-DIRECTION',
     *          8X,'3-DIRECTION',8X,'4-DIRECTION',8X,'5-DIRECTION',
     *          8X,'6-DIRECTION')
         DO 40 J=1,NJ
            PRINT 600,J,(DJ(I,J),I=1,6)
  600       FORMAT(' ',8X,I4,6(2X,F17.9))
   40    CONTINUE
         PRINT 700
  700    FORMAT('-',29X,'LOCAL ELEMENT FORCES'/30X,20('*')/
     *          ' ELE.',30X,'A-END',60X,'B-END'/'  NUM.',3X,'1',
     *          9X,'2',9X,'3',9X,'4',9X,'5',9X,'6',9X,'7',9X,'8',9X,
     *          '9',9X,'10',9X,'11',9X,'12')
         DO 50 J=1,NE
            PRINT 800,J,(FL(I,J),I=1,12)
  800       FORMAT('0',I2,12(1X,F9.4))
   50    CONTINUE
         PRINT 900
  900    FORMAT('-',33X,'NODAL FORCES'/34X,12('*')/10X,' NODE',
     *          7X,'1-DIRECTION',8X,'2-DIRECTION',8X,'3-DIRECTION',
     *          8X,'4-DIRECTION',8X,'5-DIRECTION',8X,'6-DIRECTION')
         DO 60 J=1,NJ
            PRINT 600,J,(P(I,J),I=1,6)
   60    CONTINUE
         PRINT 1000
 1000    FORMAT('0',80('_'))
      END IF
      RETURN
      END
C
```

```
C
C
C     ******************************************************************
C     *                          DOTPRD                                *
C     ******************************************************************
      FUNCTION DOTPRD(DOT1,DOT2,N)
C     COMPUTE THE DOT PRODUCT OF DOT1 AND DOT2.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DOT1(1),DOT2(1)
      DOTPRD=0.D00
      DO 10 I=1,N
         DOTPRD=DOTPRD+DOT1(I)*DOT2(I)
   10 CONTINUE
      RETURN
      END
C
C
C
C     ******************************************************************
C     *                          UPDATE                                *
C     ******************************************************************
      SUBROUTINE UPDATE(QT,Q,NEQ,QI)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION QT(1),Q(1)
      DO 10 I=1,NEQ
         QT(I)=Q(I)*QI
   10 CONTINUE
      RETURN
      END
C
C
//DATA
```

The vita has been removed from the scanned document