Title

A Computer Program for the Static and Dynamic Finite Element Analysis of Axisymmetric Thin Shells

Authors

Chow, H.

Lu, Zung

Abel, John

et al.

Publication Date

1966-10-01

SESM 66-18a

STRUCTURES AND MATERIALS RESEARCH
DEPARTMENT OF CIVIL ENGINEERING

A COMPUTER PROGRAM FOR THE STATIC AND DYNAMIC FINITE ELEMENT ANALYSIS OF AXISYMMETRIC THIN SHELLS

by

H. Y. CHOW, Z. A. LU, J. F. ABEL
Research Assistants

E. P. POPOV, J. PENZIEN
Faculty Investigators

Report to
National Aeronautics and Space Administration
NASA Research Grant No. NsG 274 S-2

OCTOBER 1966

STRUCTURAL ENGINEERING LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY CALIFORNIA

A COMPUTER PROGRAM FOR
THE STATIC AND DYNAMIC FINITE ELEMENT ANALYSIS
OF AXISYMMETRIC THIN SHELLS

by

H. Y. Chow, Z. A. Lu, J. F. Abel
Research Assistants

E. P. Popov, J. Penzien
Faculty Investigators

University of California
Berkeley, California

October 1966

PREFACE

The investigations summarized in this report were developed over a considerable length of time. In the initial phases of the investigation dealing with the static aspects of the analysis, Dr. Z. A. Lu, a former graduate student, performed the theoretical derivations and wrote the corresponding computer program. The investigation of the dynamic analysis and the writing of the expanded computer program were carried out by Mr. H. Y. Chow, a Graduate Student. Both phases of the above research have been published in detail previously [References (1) and (4)]. Mr. John Abel, a Graduate Student, was responsible for assembling the material into the present unified form for practical application.

The work was carried out under the supervision and technical responsibility of Egor P. Popov, and during the initial phases also of Joseph Penzien, both Professors of Civil Engineering, Division of Structural Engineering and Structural Mechanics, University of California, Berkeley, California.

All the work is a part of the research sponsored by the National Aeronautics and Space Administration under NASA Grants No. NsG 274-62 and No. NsG 274 S-1 and S-2.

Mr. M. Khojasteh-Bakht and Mr. S. Yaghmai, Graduate Students, read the manuscript and made helpful suggestions.

Mr. B. Kot prepared the drawings and Mrs. M. French typed the final report.

SYNOPSIS

A FORTRAN IV listing and a flow chart of a computer program for the static and dynamic small deflection analysis of axisymmetric thin shells of revolution is presented in this report. The program permits free and forced vibration investigation as well as static analysis. Although the program is not completely general as far as shell geometry and boundary conditions are concerned, enough theoretical information is presented to enable a potential user to assemble a program general enough for his particular needs. This information includes an entire chapter summarizing the theoretical bases for various aspects of the problem and a set of appendices giving useful formulae.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF APPENDICES

SYMBOLS USED IN REPORT

A	= force transformation matrix
a	= radius of sphere or cylinder
B	= displacement transformation matrix
b	= slope length along cone from apex to the centroidal circle of an element
C	= transformation matrix
c	= vector of constants
E	= modulus of elasticity
$E(s)$	= matrix kernel for integration leading to element mass matrix
e	= change in meridional length of a cylindrical or conical element or in height of an annular ring element
f	= element flexibility matrix in local co-ordinates
H,N	= stress resultants per unit nodal length in local coordinates
i,j	= subscripts designating upper and lower ends of element respectively
K	= overall assemblage stiffness matrix; bending stiffness of shell section
K.E.	= kinetic energy of a shell element at a given time
k	= element stiffness matrix in local co-ordinates
$\bar{k}$	= element stiffness matrix in global co-ordinates
k_{lm}	= partitioned portion of K matrix $(l,m = 1,2)$
$\bar{k}_{lm}$	= partitioned portion of $\bar{k}$ matrix $(l,m = i,j)$
ℓ	= length of element; $\ell = (at)^{1/2}/(12(1-\nu^2))^{1/4}$ for a spherical cap
M	= overall assemblage mass matrix; moment per nodal length in local co-ordinates

m = element mass matrix in local co-ordinates; unit mass of element

$\bar{m}$ = element mass matrix in overall co-ordinates

m_{kl} = partitioned portion of M matrix (k,l = 1,2); partitioned portion of m matrix (k,l = i,j)

$\bar{m}_{kl}$ = partitioned portion of $\bar{m}$ matrix (k,l = i,j)

n = superscript designating the n^{th} element; subscript designating the n^{th} node

o = subscript indicating the spherical cap or disc for end closure

P = vector of nodal load amplitudes

P^* = vector of generalized forces

P_h, P_v = total horizontal and vertical forces at node (global co-ordinates)

$p(s)$ = vector of distributed moment, normal load and meridional load on an element

Q = stress resultants per unit nodal length in global co-ordinates

Q_o = vertical force per unit nodal length for cap or disc

q = vector of element displacements in local co-ordinates

R = vector of all total nodal forces; vector of known total nodal forces at unrestrained nodes; radius of sphere

r = vector of all nodal displacements; vector of known displacements at supports

$\bar{r}$ = s_j/s_i for conical elements, 1 for cylindrical elements and r_j/r_i for annular flat plates

$r(s)$ = horizontal radius of element as function of slope length

r_i, r_j = horizontal radii at top and bottom of element

r_o = horizontal radius of disc or of cap node

S = vector of element stress resultants per unit nodal length in local co-ordinates

S_W = vector of element stress resultants per unit nodal length due to inertial joint loads (local co-ordinates)

s = distance from apex of cone to point of question on element

s_i, s_j = distance along cone from apex to top and bottom of element

T = total moment at node in global co-ordinates

t = thickness of shell element

U = vector of unknown nodal displacements

v = vector of element displacements in local co-ordinates; meridional displacement of element in local co-ordinates

W = vector of inertial joint loads in global co-ordinates

w = normal displacement of element in local co-ordinates

X = vector of unknown total reactions at supports

x_o = r_o/ℓ for spherical cap

Y = matrix whose rows represent linearly independent coefficients of a displacement state of an element

$\bar{Y}$ = boundary condition matrix

y = $2(3(1-\nu^2))^{1/4} (2 \tan \alpha/t)^{1/2} s^{1/2}$

I = subscript indicating portion of dynamic response due to static effects

II = subscript indicating portion of dynamic response due to acceleration

α = angle of inclination of element

β = angle from the horizontal of a line joining the center of the arc and the top of the first element

Δ, δ = translation of nodes

η = normal co-ordinates

θ, χ	= rotation of nodes
κ	= $(3(1-\nu^2)a^2/t^2)^{1/4}$
λ	= κ^{ℓ}/a for cylindrical element
ν	= Poisson's ratio
ρ_A	= radius of gyration of section of shell element
ϕ	= matrix whose columns are the normal modes (eigenvectors)
ϕ_o	= semi-angle of spherical cap or disc
ψ	= rotation of element at a given point in local co-ordinates
ω	= natural frequency in radians per unit time (square root of eigenvalue)
{ }	= column vector
< >	= row vector
[ ]	= rectangular matrix
⌈ ⌋	= square matrix with non-zero terms occurring only on the principal diagonal

INTRODUCTION

This report presents a computer program which has been developed in the course of a continuing investigation into the structural analysis of thin shells of revolution. More detailed presentations of the theoretical development of procedures have been published under separate cover [References (1), (3) and (4)]. The purpose of the present report is to give the results of the investigations to date in a form suitable for practical application.

No attempt has been made to assemble a computer program that will account for all possibilities or all options of structural analysis. Rather it is hoped that a potential user, employing the program herein as a model and the summary of theory as a tool, will be able to assemble a program suited to his particular needs. The program has been written in FORTRAN IV for the IBM 7094 computer at the University of California. However, since it is accompanied by a flow chart, there should be no difficulty to the user caused by this restriction.

The first chapter of the following is a summary of the theory, accompanied by tabulation of the relevant equations in the appendices. The second chapter contains a description of the program, a glossary of terms used in the program, the flow chart and the program listing.

CHAPTER I. SUMMARY OF THEORY

A. General

Many of the shells of revolution used in flight structures and other applications are continuous assemblages of rings, cylinders, conical segments, spherical segments and other axisymmetric shapes, often of varying thickness. It has been found that the displacement (or stiffness) method of analysis employing matrix methods with an electronic digital computer is a satisfactory approach for structural analysis of such assemblages.

Of the various displacement techniques available, the computer program herein utilizes the finite element approach in which the three displacements of selected circular nodes are the unknowns. The features of this particular finite element technique are:

(1) Axisymmetric loading, small displacements and isotropic linear elastic materials are assumed. (In principle, extension to non-symmetric loading is possible but has not been carried out in this investigation.)

(2) The shape of the assemblage is approximated by simple axisymmetric elements connecting the circular nodes. Each element is of constant thickness but different elements may have different thicknesses to account for continuous or discontinuous variations in the assemblage.

(3) The basic finite element chosen is the truncated conical segment and in the limiting cases this element becomes a spherical cap or flat plate at one extreme and a

cylindrical segment at the other.

(4) The approximate stiffness matrix for the overall assemblage is constructed from exact formulations of the structural stiffnesses of the assumed simple finite elements. The overall mass matrix is constructed from element mass matrices derived from assumed approximate displacement fields.

The finite element method is advantageous because it is easy to formulate geometrically and because the formulation is readily adaptable to the matrix algebra required for solution. In addition, boundary conditions, elastic restraints and actual geometric discontinuities (e.g., discontinuity in curvature) often occur on transverse circular sections, leading to a natural corresponding choice of circular nodes. Moreover the method not only admits modifications for local changes of properties but also allows adaptations permitting dynamic, thermal and large displacement analysis as well as the basic static small displacement analysis. Finally, the particular advantage of the use of exact stiffnesses for individual elements is that for portions of the overall assemblage where a basic element is the geometric duplicate of the structure the number of elements required is minimized.

Static solution by the stiffness method utilizes the force displacement equation

$$\{R\} = [K]\ \{r\} \tag{1}$$

Here $\{R\}$ is the vector of total nodal static forces, $\{r\}$ is the vector of nodal displacements and $[K]$ is the overall stiffness matrix which is singular. The stiffness matrix can be constructed from the element stiffnesses by requiring equilibrium and compatibility conditions to be satisfied at the nodes. The nodal force vector can be approximated or calculated from energy considerations. Construction of these two matrices will be treated later. In order to carry out the solution the matrices of Equation (1) are partitioned to get

$$\left\{\begin{array}{c} R \\ \hline X \end{array}\right\} = \left[\begin{array}{c|c} k_{11} & k_{12} \\ \hline k_{21} & k_{22} \end{array}\right] \left\{\begin{array}{c} U \\ \hline r \end{array}\right\} \tag{2}$$

where $\{R\}$ are the known total loads at the unrestrained nodes, $\{X\}$ are the total unknown reactions at the nodes of support, $\{U\}$ are the unknown nodal displacements and $\{r\}$ are the known displacements at supports. Here $[k_{11}]$ is the non-singular stiffness matrix for the nodal degrees of freedom of the assemblage so we may solve for all unknowns by writing

$$\{U\} = [k_{11}]^{-1} \left\{ \{R\} - [k_{12}] \{r\} \right\} \tag{3a}$$

and

$$\{X\} = [k_{21}] \{U\} + [k_{22}] \{r\} \tag{3b}$$

When all nodal displacements and forces are known, the static problem is essentially solved.

The dynamic analysis by the stiffness method employs the matrix formulation of the equations of motion

$$[m_{11}] \{\ddot{U}(t)\} + [k_{11}] \{U(t)\} = \{R(t)\} \tag{4}$$

where the above notation applies. Here $[m_{11}]$ and $[k_{11}]$, the mass and stiffness matrices for the nodal degrees of freedom, are symmetric and positive definite. The solution of these differential equations is carried out using the normal-mode superposition approach. This method not only gives accurate results for forced vibration problems, but also entails solution for the free vibration characteristics which are a desirable supplement to dynamic solutions if not an end in themselves.

Equations (4) must be rewritten in terms of normal co-ordinates. This can be accomplished if the mode shapes and frequencies are established such that

$$[\phi]^T [m_{11}] [\phi] = [I] \tag{5a}$$

$$[\phi]^T [k_{11}] [\phi] = \lceil \omega^2 \rfloor \tag{5b}$$

where $[\phi]$ is the square matrix whose columns are the mode shapes (eigenvectors) corresponding to the lowest eigenvalues, ω^2, in ascending order of value. $[I]$ is an identity matrix and $\lceil \omega^2 \rfloor$ is a diagonal matrix of the eigenvalues. The results in Equations (5) can be attained by first finding the eigenvalues and eigenvectors of $[m_{11}]$ so that

$$[\bar{\phi}]^T [m_{11}] [\bar{\phi}] = \lceil \bar{\omega}^2 \rfloor \tag{6}$$

and then modifying $[k_{11}]$ as follows

$$[\bar{K}] = \lceil \bar{\omega}^{-1} \rfloor \, [\bar{\Phi}]^T \, [k_{11}] \, [\bar{\Phi}] \, \lceil \bar{\omega}^{-1} \rfloor \; . \tag{7}$$

Then the natural frequencies are obtained by calculating the eigenvalues and eigenvectors of $[\bar{K}]$ thus

$$[\bar{\bar{\Phi}}]^T \, [\bar{K}] \, [\bar{\bar{\Phi}}] = \lceil \omega^2 \rfloor . \tag{8}$$

Finally, the mode shapes are calculated using the relation

$$[\Phi] = [\bar{\Phi}] \, \lceil \bar{\omega}^{-1} \rfloor \, [\bar{\bar{\Phi}}] \; . \tag{9}$$

The derivation of these results is detailed in Reference (4) and can be verified by application of standard numerical analysis procedures.

Since the columns of $[\Phi]$ are the normal modes, the unknown displacements in overall (or global) co-ordinates can be expressed in terms of the normal co-ordinates $\{\eta(t)\}$ by

$$\{U(t)\} = [\Phi] \, \{\eta(t)\} . \tag{10}$$

Hence substituting Equation (10) into the equations of motion (4) and premultiplying by $[\Phi]^T$, one gets

$$\{\ddot{\eta}(t)\} + \lceil \omega^2 \rfloor \, \{\eta(t)\} = [\Phi]^T \, \{R(t)\} = \{P^*(t)\} \tag{11}$$

which are the uncoupled equations of motion in normal co-ordinates and where $\{P^*(t)\}$ is the generalized load vector. This vector may be expressed as the product of a nodal-load amplitude vector and a function of time

$$\{P^*(t)\} = \{P\} \, f(t) . \tag{12}$$

The uncoupled equations represented in Equation (11) can be integrated directly with respect to time as an initial value problem. As many modes as desired can be used since it may not be necessary to use as many shapes as there are degrees of freedom. In any case, the final result calculated will be the superposition of the responses in all the modes chosen. The integration formulae used in this program are given in Appendix A. It should be noted that the time increment for integration for a particular mode must be sufficiently smaller than the period for that mode in order to get accurate results. The accompanying computer program automatically reduces that increment if necessary. Once the displacements are known at any given time, the reactions can also be found using Equation (3b) and the dynamic response is essentially solved.

With all the nodal displacements and forces known in both static and dynamic problems, it is an easy matter to utilize the known elastic and geometry properties of the elements to solve for the internal stresses and/or strains. The equations for such calculation are given in Appendix B.

B. Construction of Element Stiffness Matrices

For the construction of the required element stiffness matrix the known homogeneous bending and membrane solutions of a uniform thin shell of corresponding configuration are used to formulate the element flexibility matrix. The element stiffness is then merely the inverse of the flexibility. In each case, before inverting the flexibility, it is modified to ensure that correct symmetry properties are exhibited. The stiffnesses generated by this computer program are referenced to a local co-ordinate system oriented to the particular element.

Computation of the conical element flexibility and of the spherical cap flexibility requires the use of Thomson functions. The necessary series and asymptotic formulae for evaluating these functions are given in Appendix C and the criteria for various methods of computation are given in Appendix D. Although the program provides correct values of the Thomson functions, it should be noted that for some arguments the magnitude of the Thomson function values exceeds the capacity of the computer. To counteract this, equal negative or positive powers of ten must be taken from the appropriate groups of functions so that the capacity is not exceeded. The resulting stiffnesses are not impaired since each flexibility co-efficient involves a product of a ber-group function and a ker-group function and thus the opposite powers of ten cancel.

1. Conical Element Stiffnesses

For the conical element shown in Figure 1, the element displacements are related to the element forces by the flexibility matrix as follows

$$\begin{Bmatrix} \chi_i \\ \chi_j \\ \delta_i \\ \delta_j \\ e \end{Bmatrix} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{15} \\ f_{21} & \cdots & & f_{25} \\ f_{31} & & & f_{35} \\ f_{41} & & & f_{45} \\ f_{51} & \cdots & & f_{55} \end{bmatrix} \begin{Bmatrix} M_i \\ M_j \\ H_i \\ H_j \\ N_j \end{Bmatrix} \qquad (13)$$

or in matrix notation

$$\{v\} = [f]\ \{S\} \tag{14}$$

where $\{S\}$ are the nodal forces per unit nodal length. The formulae for the elements of the 5x5 flexibility matrix are given in Appendix E and are derived in Reference (1). Thus the stiffness matrix is the inverse of the flexibility matrix and satisfies

$$\{S\} = [f]^{-1}\ \{v\} = [k]\ \{v\}\ . \tag{15}$$

Since the element forces are all quantities per unit length, the flexibility matrix is not symmetrical. Elements of the matrix representing displacements due to forces on opposite edges of the element will bear a ratio of $\bar{r} = s_j/s_i$ to each other, reflecting the ratio of circumferences. However, the matrix does satisfy Betti's law and since engineers commonly use expressions involving action per unit length, this terminology is maintained. However, since the overall stiffness matrix must be symmetric for eigenvalue calculations, the element stiffnesses are made symmetrical during the construction of the overall stiffness. This will be discussed below.

Investigations have shown that the first 4x4 portion of the flexibility matrix is nearly perfect for almost any case provided the length of the element exceeds the thickness, but as α approaches 90° (as tan α becomes infinite) the cylindrical element should be used. In addition the fifth row or fifth column diverges from proper values as α approaches 90° or 0° respectively. To reduce discrepancies the program was written such that for $\alpha < 30^\circ$ or $\alpha > 150^\circ$ quantities of the fifth row are used to establish those of the fifth column (by multiplication with appropriate factors) and that for $60^\circ < \alpha < 120^\circ$ the opposite is done.

For conical elements with $90^\circ < \alpha < 180^\circ$, by taking the larger slant distance as s_i and the smaller as s_j we can use the same formulae to construct [f]. However, a sign change must be applied to all third and fourth column quantities (f_{ij} for i = 1 to 5, j = 3,4) and to f_{35} and f_{45} to counteract the sign change of cos α and also to f_{55} to counteract the sign change in ln $\bar{r}$.(see Appendix E).

2. Cylindrical Element Stiffness

Similarly, for the cylindrical element shown in Figure 2 the displacement-force equation is

$$\{v\} = [f]\ \{S\}\ . \qquad (14)$$

The formulae for the elements of this 5x5 symmetrical flexibility matrix are given in Appendix F and are derived in Reference (1).

3. Spherical Cap Stiffness

A spherical cap element may be used to approximate the end closure of a shell of revolution. A cap is shown in Figure 3. Depending on the manner in which the cap is held in vertical equilibrium, there are two cases of interest.

The Case with Singularity. If the cap is supported by a concentrated force P at the apex, then with the condition that the apex is a fixed reference point, δ_{v_o} represents the increase in height of the cap.

A 3x3 flexibility matrix is appropriate in this case satisfying the relation

$$\begin{Bmatrix} \chi_o \\ \delta_{h_o} \\ \delta_{v_o} \end{Bmatrix} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{Bmatrix} M_o \\ H_o \\ Q_o \end{Bmatrix} \tag{16}$$

or in matrix notation

$$\{v_o\} = [f_o] \{S_o\} \ . \tag{17}$$

The formulae for the elements of this symmetrical 3x3 flexibility matrix may be found in Appendix G and are derived in Reference (1). The use of the stiffness matrix obtained from inverting this flexibility yields satisfactory results except for local inaccuracies near the singularity.

<u>The Case Without Singularity</u>. When the vertical edge force of the cap is balanced by some distributed pressure rather than by a single force, only the first 2x2 portion of the matrix should be retained. It is possible to invert this 2x2 matrix and use the result as the cap stiffness, treating the vertical force and displacement components separately. Thus the displacement-force equation is

$$\left\{\begin{array}{c} \chi_o \\ \delta_{h_o} \\ \hline \delta_{v_o} \end{array}\right\} = \left[\begin{array}{cc|c} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ \hline f_{31} & f_{32} & f_{33} \end{array}\right] \left\{\begin{array}{c} M_o \\ H_o \\ \hline Q_o \end{array}\right\} \tag{18}$$

where the upper left submatrix of $[f_o]$ is inverted for the 2x2 stiffness matrix. The formulae for the elements of the 2x2 flexibility matrix may be found in Appendix G and are derived in Reference (1).

4. Flat Plate Element Stiffness

There are two possible flat plate elements -- an annular ring element and a disc for end closure.

Annular Ring Element. This element as shown in Figure 4 might be suitable for an open end of an assemblage or for any intermediate section of the assemblage where α approaches 0° or 180° for the conical element. Like the conical and cylindrical elements, this annular ring element has a 5x5 flexibility matrix and the displacement-force equation is

$$\{v\} = [f]\ \{S\}. \tag{14}$$

The formulae for the elements of this non-symmetrical flexibility matrix are given in Appendix H and are derived from References (5) and (6). The ratio relating displacements due to forces on opposite edges of the element is $\bar{r} = r_j/r_i$. Note that the case given in Figure 4 is for $\alpha = 0^\circ$. For $\alpha = 180^\circ$ an analogous set of formulae are necessary to construct the flexibility matrix. These are also given in Appendix H.

Disc for End Closure. For end closures that are flat or nearly flat, a circular disc element as shown in Figure 5 can be used. For the case with singularity, i.e., when the vertical edge forces are balanced by a concentrated load at the center, the result is a 3x3 symmetrical flexibility matrix and the displacement-force equation is

$$\{v_o\} = [f_o]\ \{S_o\} \tag{17}$$

The formulae for the elements of this flexibility matrix are given in Appendix H and are derived from References (5) and (6). For the non-singular case, the first 2x2 portion of Equation (17) is used.

C. Construction of Element Mass Matrices

It is possible to express the displacements at any point in a shell element by the relationship

$$\{q(s)\} = \begin{Bmatrix} \psi(s) \\ w(s) \\ v(s) \end{Bmatrix} = [Y(s)]\ \{c\} \tag{19}$$

Here the displacement at any value of s ($s_i \le s \le s_j$) in local co-ordinates are $\{q(s)\}$ as indicated in Figures 1, 2, 4 and 5. $[Y(s)]$ is the matrix whose rows represent linearly independent coefficients of a displacement state and $\{c\}$ is a vector of constants. The vector of constants can be determined from the six boundary conditions by evaluating Equation (19) at both ends of the element as follows

$$\{c\} = \begin{bmatrix} Y(s_i) \\ Y(s_j) \end{bmatrix}^{-1} \begin{Bmatrix} q(s_i) \\ q(s_j) \end{Bmatrix} = [\bar{Y}]\ \{q\}\ . \tag{20}$$

Substituting this expression for the vector of constants into Equation (19), one obtains

$$\{q(s)\} = [Y(s)]\ [\bar{Y}]\ \{q\} \tag{21}$$

In the dynamic problem the displacements are of course time dependent so Equation (21) would be rewritten to get

$$\{q(s,t)\} = [Y(s)]\ [\bar{Y}]\ \{q(t)\}\ . \tag{22}$$

Then at any instant of time the kinetic energy of the shell element can be written as

$$K.E. = \frac{1}{2} \int_s \left[m\ \rho_A^2\ \dot{\psi}^2(s,t) + m\dot{v}^2(s,t) + m\dot{w}^2(s,t) \right] 2\pi r(s)\ ds \tag{23}$$

where m is the mass per unit surface area of the element, ρ_A is the radius of gyration of the section of a shell element and r(s) is the shell radius measured normal to the axis. Substituting the displacement formulation (22) into (23), the expression for the kinetic energy becomes

$$K.E. = \frac{1}{2} < \dot{q}(t) > [\bar{Y}]^T \left(\int_s 2\pi\ [E(s)]\ r(s)\ ds \right) [\bar{Y}]\ \{\dot{q}(t)\} \tag{24}$$

where

$$[E_{ij}(s)] = m\rho_A^2\ \{Y_{1i}(s)\} < Y_{1j}(s) > + m\ \{Y_{2i}(s)\} < Y_{2j}(s) >$$

$$+ m\ \{Y_{3i}(s)\} < Y_{3j}(s) >, \text{ for } i,j = 1,2,\ldots 6 \tag{25}$$

$< Y_{ij}(s) >$, $< Y_{2j}(s) >$ and $< Y_{3j}(s) >$ are the row vectors identical to the first, second and third rows of the matrix $[Y(s)]$. By comparing Equation (24) with the usual expression for kinetic energy

$$K.E. = \frac{1}{2} < \mathring{q}(t) > [m] \{\mathring{q}(t)\} \tag{26}$$

it is apparent that the element mass matrix is given by

$$[m] = [\bar{Y}]^T \int_s 2\pi \, [E(s)] \, r(s) \, ds \, [\bar{Y}]. \tag{27}$$

The question remains what displacement state, $[Y(s)]$, to use in formulating the mass matrix for the element. It would be possible to employ the static homogeneous (exact) solution such as used to determine the element flexibility matrix above. However, for conical elements these involve very complicated expressions in terms of Thomson functions and their derivatives, and the integration in Equation (27) is prohibitive. It has been found that satisfactory results can be obtained from an assumed displacement field in the simplest possible polynomial form, neglecting rotational inertia. The resulting "consistent" mass matrix, is superior to the mass formulation based on the tributary areas.

1. <u>Open-ended Elements</u>

For conical, cylindrical and annular plate elements, the assumed polynomial displacement field used to construct the distributed mass matrix is

$$[Y(s)] = \begin{bmatrix} 0 & 1 & 2s & 3s^2 & 0 & 0 \\ 1 & s & s^2 & s^3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & s \end{bmatrix} \tag{28}$$

When this matrix is evaluated at both ends of the element, the result is

$$[\bar{Y}]^{-1} = \begin{bmatrix} Y(s_i) \\ \hline Y(s_j) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 2s_i & 3s_i^2 & 0 & 0 \\ 1 & s_i & s_i^2 & s_i^3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & s_i \\ \hline 0 & 1 & 2s_j & 3s_j^2 & 0 & 0 \\ 1 & s_j & s_j^2 & s_j^3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & s_j \end{bmatrix} \tag{29}$$

The formulae for the element mass matrix, neglecting rotational inertia, are given in Appendix I.

2. <u>End Closure Elements</u>

For end-closure elements, i.e., elements having only a single node, the distributed mass matrix of a disc shaped element is used. Here the appropriate displacement field is given by

$$[Y(s)] = \begin{bmatrix} 0 & 2s & 0 \\ 1 & s^2 & 0 \\ 0 & 0 & s \end{bmatrix} \tag{30}$$

and the boundary condition matrix in this case is

$$[\bar{Y}]^{-1} = [Y(r_o)] = \begin{bmatrix} 0 & 2r_o & 0 \\ 1 & r_o^2 & 0 \\ 0 & 0 & r_o \end{bmatrix} . \tag{31}$$

The formulae for the element mass matrix, neglecting rotational inertia, are given in Appendix J.

D. Construction of Transformation Matrices

In order to relate forces and displacements in both local and global systems and to enable the construction of overall mass and stiffness matrices by the direct stiffness method, certain transformation matrices are necessary.

1. Displacement Transformation Matrices

The displacement transformation matrix $[B]$ relates the displacements in the local co-ordinate system of the element $\{v\}$ to those of the overall assemblage (global) co-ordinate system $\{r\}$ according to the relation

$$\{v\} = [B]\,\{r\} . \tag{32}$$

By comparing the displacements $\{v\}$ indicated in Figures 3 and 5 with $\{r\}$ in Figure 7, it is apparent that for the disc and spherical cap, Equation (32) may be written as

$$\begin{Bmatrix} \chi_o \\ \delta_{h_o} \\ \delta_{v_o} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{Bmatrix} \theta_o \\ \Delta_{h_o} \\ \Delta_{v_o} \end{Bmatrix}. \tag{33}$$

Similarly, by comparing Figures 1 and 2 with Figure 7, for the cylindrical and conical cases Equation (13) may be rewritten as

$$\begin{Bmatrix} \chi_i \\ \chi_j \\ \delta_i \\ \delta_j \\ e \end{Bmatrix} = [B_i \mid B_j] \begin{Bmatrix} r_i \\ \hline r_j \end{Bmatrix} = \left[\begin{array}{ccc|ccc} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -\cos\alpha & \sin\alpha & 0 & \cos\alpha & -\sin\alpha \end{array}\right] \begin{Bmatrix} \theta_i \\ \Delta_{hi} \\ \Delta_{vi} \\ \hline \theta_j \\ \Delta_{hj} \\ \Delta_{vj} \end{Bmatrix} \tag{34}$$

Finally, comparing Figure 4 with Figure 7, for the annular element with $\alpha = 0^o$

$$\begin{Bmatrix} \chi_i \\ \chi_j \\ \delta_i \\ \delta_j \\ e \end{Bmatrix} = [B_i \mid B_j] \begin{Bmatrix} r_i \\ \hline r_j \end{Bmatrix} = \left[\begin{array}{ccc|ccc} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{array}\right] \begin{Bmatrix} \theta_i \\ \Delta_{hi} \\ \Delta_{vi} \\ \hline \theta_j \\ \Delta_{hj} \\ \Delta_{vj} \end{Bmatrix} \tag{35}$$

For the case $\alpha = 180^{\circ}$, the signs of elements B_{32} and B_{45} must be changed. Application of this displacement transformation for construction of the overall assemblage stiffness matrix is described below.

The transformation matrix [C] relates the displacements in the local co-ordinate system {q} to those of the overall assemblage co-ordinate system {r} according to the relation

$$\{q\} = [C]\ \{r\} \tag{36}$$

By comparing the displacements {q} indicated in Figures 1, 2 and 4 with the displacements {r} in Figure 7, it is apparent that Equation (36) may be rewritten as

$$\{q\} = \left\{\begin{array}{c} d_i \\ \hline d_j \end{array}\right\} = \left[\begin{array}{c|c} C_i & 0 \\ \hline 0 & C_i \end{array}\right] \left\{\begin{array}{c} r_i \\ \hline r_j \end{array}\right\} \tag{37}$$

where

$$[C_i] = \begin{bmatrix} -1 & 0 & 0 \\ 0 & \sin\alpha & \cos\alpha \\ 0 & -\cos\alpha & \sin\alpha \end{bmatrix} . \tag{38}$$

From Figure 5 it is apparent that for the disc element {q} and {r} are identical. Application of this transformation matrix for construction of the overall assemblage mass matrix and of the load vector is described below.

3. Force Transformation Matrix

The force transformation matrix [A] relates the nodal forces per unit nodal lengh in global co-ordinates {Q} to the element forces {S} as follows

$$\{Q\} = [A]\ \{S\}\quad . \tag{39}$$

Analogous to the derivation of Equations (33), (34) and (35), Equation (39) may be rewritten for the three relevant cases as

$$\frac{1}{2\pi r_o} \begin{Bmatrix} T_o \\ P_{ho} \\ P_{vo} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{Bmatrix} M_o \\ H_o \\ Q_o \end{Bmatrix} \tag{40}$$

and

$$\begin{Bmatrix} Q_i \\ \hline Q_j \end{Bmatrix} = \frac{1}{2\pi} \begin{Bmatrix} T_i/r_i \\ P_{hi}/r_i \\ P_{vi}/r_i \\ \hline T_j/r_j \\ P_{hj}/r_j \\ P_{vj}/r_j \end{Bmatrix} = \begin{bmatrix} A_i \\ \hline A_j \end{bmatrix} \{S\} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -\bar{r}\cos\alpha \\ 0 & 0 & 0 & 0 & \bar{r}\sin\alpha \\ \hline 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \cos\alpha \\ 0 & 0 & 0 & 0 & -\sin\alpha \end{bmatrix} \begin{Bmatrix} M_i \\ M_j \\ H_i \\ H_j \\ N_j \end{Bmatrix} \tag{41}$$

and

$$\begin{Bmatrix} Q_i \\ \hline Q_j \end{Bmatrix} = \begin{bmatrix} A_i \\ \hline A_j \end{bmatrix} \{S\} \quad \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \bar{r} \\ \hline 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{Bmatrix} M_i \\ M_j \\ H_i \\ H_j \\ N_j \end{Bmatrix} . \tag{42}$$

For the case $\alpha = 180^{\circ}$, Equation (42) must be adapted by changing the sign on elements A_{23}, A_{35}, A_{54} and A_{55}. Application of the force transformation matrix for construction of the overall assemblages stiffness matrix is described below.

E. Calculation and Construction of Joint Load Vectors

In order to solve for the unknown nodal displacements and forces, it is necessary to construct the vector of nodal loads $\{R\}$ as employed in Equations (2) and (3). This section presents the necessary procedures to accomplish this construction for static loads. For dynamic loads the difference is merely that all loads are time dependent.

1. Joint Loads

If there are axisymmetric concentrated loads on the assemblage, it is convenient and desirable to select nodal circles corresponding to these loads, which then can be taken directly as joint loads by resolving them into the proper components. Often however, a distributed load exists and in this case there are two methods of calculating the necessary nodal loads.

Tributary Joint Loads. Using the tributary areas adjacent to each node, distributed loads can be converted to the appropriate components of nodal loads. Thus, for the n^{th} element shown in Figure 6, the contribution to the nodal loads as shown in Figure 7 are given by

$$\{R^{(n)}\} = \left\{\begin{array}{c} R_i^{(n)} \\ \hline R_j^{(n)} \end{array}\right\} = [C]^T \left\{\begin{array}{c} \int_{s_i}^{b} 2\pi r(s) \ \{p(s)\} ds \\ \hline \int_{b}^{s_j} 2\pi r(s) \ \{p(s)\} ds \end{array}\right\} \tag{43}$$

where $\{p(s)\}$ is the 3x1 vector of distributed moments, normal loads and meridional loads in the ψ, w and v directions respectively. For a disc or spherical cap, loads are assumed to be entirely concentrated at the single adjoining node.

Consistent Joint Loads. When the element sizes are relatively small, tributary loads give satisfactory results. However, as the element sizes increase it becomes more advantageous to use consistent joint loads. These are constructed so that in a virtual displacement the work done by the actual load is equal to the work done by the consistent joint loads. The final result for the n^{th} element as derived in Reference (8) is given by

$$\{R^{(n)}\} = \left\{\begin{array}{c} R_i^{(n)} \\ \hline R_j^{(n)} \end{array}\right\} = \left\{\begin{array}{c} T_i^{(n)} \\ P_{hi}^{(n)} \\ P_{vi}^{(n)} \\ \hline T_j^{(n)} \\ P_{hj}^{(n)} \\ P_{vj}^{(n)} \end{array}\right\} = [C]^T [\bar{Y}]^T \int_s [Y(s)]^T \{p(s)\} 2\pi r(s) \ ds \tag{44}$$

where the notation is the same as in Equation (43) above. These are "consistent" joint loads because their calculation employs the assumed displacement field.

2. Load Vector

The contributions of the various elements must be combined to get the vector of nodal loads for the overall assemblage. In both cases of load calculation there is a contribution from each of the two neighboring elements to the joint loads at each node. Using subscripts to indicate the node and superscripts to indicate the element in the assemblage, from Figure 7 it is apparent that the total nodal load is

$$\{R\}_n = \begin{Bmatrix} T_i^{(n)} + T_j^{(n-1)} \\ P_{hi}^{(n)} + P_{hj}^{(n-1)} \\ P_{vi}^{(n)} + P_{vj}^{(n-1)} \end{Bmatrix} = \{R_i^{(n)}\} + \{R_j^{(n-1)}\} \tag{45}$$

Then the load vector can be assembled by combining the nodal loads directly as follows

$$\{R\} = \begin{Bmatrix} R_1 \\ R_2 \\ \cdot \\ \cdot \\ \cdot \\ R_{n+1} \end{Bmatrix} \tag{46}$$

F. Construction of Overall Assemblage Stiffness and Mass Matrices

The overall stiffness and mass matrices are assembled according to the "direct stiffness" method. First the element stiffnesses and mass matrices are transformed from the local to the global co-ordinate system. Then these transformed matrices are partitioned and the submatrices are added to the appropriate parts of the overall matrices. Finally these overall matrices are rearranged so they may be partitioned as in Equation (2).

Substituting Equation (32) into Equation (15) gives

$$\{S\} = [k]\ [B]\ \{r\} \tag{47}$$

and inserting this equation into Equation (39) and converting from forces per unit length to total forces produces

$$\{R\} = 2\pi \begin{bmatrix} r_i & & & & & \\ & r_i & & & & \\ & & r_i & & & \\ & & & r_j & & \\ & & & & r_j & \\ & & & & & r_j \end{bmatrix} [A]\ [k]\ [B]\ \{r\} = [\bar{k}]\ \{r\}\ . \tag{48}$$

Comparing the result with Equation (1) it is apparent that Equation (48) is the force-displacement equation for a single element in the overall co-ordinates and that $[\bar{k}]$ is the symmetric element stiffness in the global co-ordinates. This formulation provides the basic building blocks for constructing the overall stiffness matrix. Using the partitioned forms of Equations (34), (35), (41) and (42), it is possible to rewrite the element stiffness in Equation (48) as

$$\left[\begin{array}{c|c} \bar{k}_{ii} & \bar{k}_{ij} \\ \hline \bar{k}_{ji} & \bar{k}_{jj} \end{array}\right] = 2\pi \left[\begin{array}{ccc|ccc} r_i & & & & & \\ & r_i & & & & \\ & & r_i & & & \\ \hline & & & r_j & & \\ & & & & r_j & \\ & & & & & r_j \end{array}\right] \left[\begin{array}{c} A_i \\ \hline A_j \end{array}\right] [k] \, [B_i \mid B_j] \; . \tag{49}$$

It is evident from Equation (49) how the force transformation and displacement transformation matrices are used to formulate the 3x3 portions of $[\bar{k}]$

$$[\bar{k}_{lm}] = 2\pi r_l \; [A_l] \; [k] \; [B_m]; \quad l,m = i,j \; . \tag{50}$$

In a completely analogous procedure, substituting Equation (36) into Equation (26) gives

$$K.E. = \frac{1}{2} < \dot{r}(t) > [C]^T \; [m] \; [C] \; \{\dot{r}(t)\}. \tag{51}$$

It is obvious from Equation (51) that the element mass matrix in global co-ordinates is

$$[\bar{m}] = [C]^T \; [m] \; [C] \tag{52}$$

and from Equations (27) and (37) that this may be partitioned as follows

$$[\bar{m}] = \left[\begin{array}{c|c} \bar{m}_{ii} & \bar{m}_{ij} \\ \hline \bar{m}_{ji} & \bar{m}_{jj} \end{array}\right] = \left[\begin{array}{c|c} C_i & 0 \\ \hline 0 & C_i \end{array}\right]^T \left[\begin{array}{c|c} m_{ii} & m_{ij} \\ \hline m_{ji} & m_{jj} \end{array}\right] \left[\begin{array}{c|c} C_i & 0 \\ \hline 0 & C_i \end{array}\right] . \qquad (53)$$

Thus the 3x3 submatrices of $[\bar{m}]$, which are the basic building blocks for the overall mass matrix, are gotten using the transformation matrix [C]

$$[\bar{m}_{kl}] = [C_i]^T [m_{kl}] [C_i]; \quad k,l = i,j . \qquad (54)$$

To see how the mass and stiffness submatrices are to be added to the overall matrices, the equation of motion for the n^{th} element can be written in partitioned form

$$\left[\begin{array}{c|c} \bar{m}_{ii}^{(n)} & \bar{m}_{ij}^{(n)} \\ \hline \bar{m}_{ji}^{(n)} & \bar{m}_{jj}^{(n)} \end{array}\right] \left\{\begin{array}{c} \ddot{r}_i^{(n)} \\ \hline \ddot{r}_j^{(n)} \end{array}\right\} + \left[\begin{array}{c|c} \bar{k}_{ii}^{(n)} & \bar{k}_{ij}^{(n)} \\ \hline \bar{k}_{ji}^{(n)} & \bar{k}_{jj}^{(n)} \end{array}\right] \left\{\begin{array}{c} r_i^{(n)} \\ \hline r_j^{(n)} \end{array}\right\} = \left\{\begin{array}{c} R_i^{(n)} \\ \hline R_j^{(n)} \end{array}\right\} . \qquad (55)$$

Also, maintaining the convention of superscripts and subscripts to indicate elements and nodes respectively, the compatability requirement at the n^{th} node will be

$$\{r\}_n = \{r_i^{(n)}\} = \{r_j^{(n-1)}\} . \qquad (56)$$

Thus from Equations (45), (55), and (56) the following recursion formula can be obtained:

$$[\bar{m}_{ji}^{(n-1)}]\ \{\ddot{r}\}_{n-1} + \left[[\bar{m}_{ii}^{(n)}] + [\bar{m}_{jj}^{(n-1)}]\right] \{\ddot{r}\}_n +$$

$$+ [\bar{m}_{ij}^{(n)}]\ \{\ddot{r}\}_{n+1} + [\bar{k}_{ji}^{(n-1)}]\ \{r\}_{n-1} + \left[[\bar{k}_{ii}^{(n)}] + [\bar{k}_{jj}^{(n-1)}]\right] \{r\}_n + \qquad (57)$$

$$+ [\bar{k}_{ij}^{(n)}]\ \{r\}_{n+1} = \{R\}_n \ .$$

Using this recursion relation, the overall assemblage mass and stiffness matrices are constructed. For example, dropping the intermediate brackets, the overall stiffness matrix may be written as

$$[K] = \begin{bmatrix} [\bar{k}^{(0)} + \bar{k}_{ii}^{(1)}] & [\bar{k}_{ij}^{(1)}] & & & \\ [\bar{k}_{ji}^{(1)}] & [\bar{k}_{jj}^{(1)} + \bar{k}_{ii}^{(2)}] & [\bar{k}_{ij}^{(2)}] & & \\ & [\bar{k}_{ji}^{(2)}] & [\bar{k}_{jj}^{(2)} + \bar{k}_{ii}^{(3)}] & & \\ & & \cdots\cdots & & \\ & & \cdots\cdots & & \\ & & [\bar{k}_{jj}^{(n-1)} + \bar{k}_{ii}^{(n)}] & [\bar{k}_{ij}^{(n)}] \\ & & [\bar{k}_{ji}^{(n)}] & [\bar{k}_{jj}^{(n)}] \end{bmatrix} . \qquad (58)$$

The overall mass matrix [M] is identical to the form in Equation (58) with each $\bar{k}$ replaced by $\bar{m}$.

G. Solutions for Internal Stress Response

In the static problem the element stress resultants {S} are easily gotten from the element stiffness matrix once the nodal displacement are known:

$$\{S\} = [k]\ \{v\} = [k]\ [B]\ \{r\}\ . \tag{47}$$

However, in dynamic problems Equation (47) yields highly inaccurate results since acceptable computational errors in the displacement respone $\{r(t)\}$ are amplified by the element stiffness. Thus an alternate approach proposed in Reference (9) must be adopted so that the degree of accuracy of the stress resultants $\{S(t)\}$ will be of the same order as that of the displacements $\{r(t)\}$. This approach is also detailed in Reference (4).

The essence of this alternate approach is to calculate the internal stress response in two parts - - the response due to static application of the loads $\{S(t)\}_I$ and the response due to the acceleration of the system $\{S(t)\}_{II}$. Since a limited number of approximate modes are used in calculating the response, the static effect would not be completely accounted for if Equation (47) were used. By considering the static portion of the response separately, that portion is certain to be accurately determined. The superposition of static and acceleration responses is expressed by

$$\{S(t)\} = \{S(t)\}_I + \{S(t)\}_{II}\ . \tag{59}$$

$\{S(t)\}_I$ is easily determined by using the static displacement response $\{r(t)\}_I$ in Equation (47). To obtain $\{S(t)\}_{II}$ the uncoupled equation of motion for the i^{th} mode is considered

$$\ddot{\eta}_i(t) + \omega_i^2\ \eta_i(t) = P_i^*\ (t)\ . \tag{60}$$

This equation can be rewritten in the form

$$\frac{-\ddot{\eta}_i(t)}{\omega_i^2} = \eta_i(t) - \frac{P_i^*(t)}{\omega_i^2} . \qquad (61)$$

This quantity represents the response due to the acceleration of the system when it is vibrating in its i^{th} mode. It can be accurately calculated once $\eta_i(t)$, the displacement response is normal co-ordinates for the particular mode, is known. Furthermore, for each normal mode there is an inertial joint load system which, when applied to the system, will cause it to deform with a unit amplitude in that mode. For the i^{th} mode this inertial joint load vector is given by

$$\{W\}_i = \omega_i^2 [M] \{\phi\}_i \qquad (62)$$

where $\{\phi\}_i$ is the i^{th} normal mode of the system, i.e., the i^{th} column of $[\phi]$. Using this set of inertial joint loads the corresponding internal stress resultants $\{S_W(t)\}_i$ are computed for each mode. These are then amplified by the values of the acceleration response of the corresponding modes to obtain the total response due to the acceleration of the system

$$\{S(t)\}_{II} = - \sum_i \frac{\ddot{\eta}_i(t)}{\omega_i^2} \{S_W(t)\}_i , \qquad (63)$$

where the summation is over the total number of modes used in calculating the dynamic response.

CHAPTER II. COMPUTER PROGRAM

A. General

The computer program that follows is an example of the unification of the principles outlined in Chapter I into a complete shell analysis routine with various analysis options. The described program is restricted to certain types of shell configurations and boundary conditions. A more general program could be written. The possible modifications are left for completion by a potential user. The appendices give the additional equations necessary for assembling other programs.

A complete description of the program, including all input and output quantities, follows. Then a flow chart of the program is given followed by a FORTRAN listing. No examples of results are given here since they may be found in References (1) and (4).

B. Description of Program

1. Purpose and Scope

The program will perform static, free vibration or forced vibration small deflection analysis of axisymmetrically loaded thin shells of revolution. Structural configuration is limited to shells open at both ends or closed at only one end. Boundary conditions can occur only at one nodal circle, but this may be at any of the nodes, interior or exterior. The possible structural supports are three: a node completely fixed against all three possible displacements, a hinged node or a hinged node free to roll in a direction perpendicular to the axis of symmetry.

The program employs only conical elements except when the shell has a closed end. In that case, the closure is provided by a disc. The maximum number of conical elements is 25. There are various options on methods of inputing element geometry. One alternative is to provide x and y co-ordinates for each node. Another is to provide the element angle α and the slope length co-ordinates for the element's two nodes. Finally, if all the shell elements fit a circular arc with either positive or negative Gaussian curvature, special provision is made to use input geometry that utilizes this fact.

Subroutines are included that calculate the element stiffness and mass matrices, assemble the overall assemblage stiffness and mass, compute the frequencies and node shapes and find the static or dynamic nodal displacement and internal stress resultant response. No provision is made to calculate internal stresses or strains but see Appendix B for these formulae. Also, the joint load vector is not computed internally, rather is read in as part of the input data. This input may take the form of either applied loads or applied accelerations. See Chapter I, Section E for discussion of methods for formulating the joint loads.

2. Subroutines

The principal program, called MAIN, controls most of the inputing of data and, depending upon the options indicated by the user, utilizes the several subroutines to carry out the shell analysis. The primary subroutines perform the remainder of the data input and virtually all of the output of computed quantities. They also control the necessary tape storage. Following is a list of the primary subroutines and their functions:

L1 Computation of disc stiffness for end closure

L2 Computation of conical element stiffnesses

L3 Computation of element mass matrices; assembly and rearrangement of overall stiffness and mass matrices

L4 Calculation of frequencies and mode shapes

L5 Numerical integration to get dynamic displacement response

L6 Inversion of overall stiffness to get static displacement and internal stress responses

L7 Solution for dynamic internal stress response

In turn, the primary subroutines employ various secondary subroutines to carry out repeated basic calculation procedures. Following is a list of the secondary subroutines and their functions:

THØ Calculation of Thomson functions for two arguments

FLEKCØ Calculation of a single conical element stiffness

SHKMT Computation of element mass matrices; assembly of overall stiffness and mass matrices

SHEKXM Rearrangement of overall mass and stiffness matrices according to boundary conditions

MULT1 Multiplication of two conformable matrices

MULT2 Premultiplication of a matrix by the transpose of another matrix

RESPØN Numerical integration for dynamic displacement response

SRES Calculation of the static internal stress response.

Finally, there are three standardized secondary subroutines which are drawn from the SHARE library. Flow charts or FØRTRAN listings will not be given for these subroutines, since programs of their type are generally available. These subroutines are:

INVERT Inversion of a square matrix

HØWF Computation of eigenvalues and eigenvectors of a matrix

PRINTM Printing a matrix

3. Input

Following are the input quantities necessary to indicate desired options and to provide the required data. Included is the format in which the data is to be provided and a brief description of the quantities. For a more detailed definition, see the glossary below. Conditions on the need for various input quantities are underlined.

N: number of conical elements (maximum 25)

NS: cone geometry input code
- NS = 0 for input of s_i, s_j and α
- NS = 2 for input of x_i and y_j of nodes
- NS = 1 for circular arc, positive Gaussian curvature
- NS = -1 for circular arc, negative Gaussian curvature

NSØ: end closure code
- NSØ = 1 for end closure
- NSØ = 0 for no closure

NSD: static/dynamic code
- NSD = 1 for static problem
- NSD = 2 for dynamic problem

NFRSC: number of degrees of freedom of restrained node
- NFRSC = 0 for fixed node
- NFRSC = 1 for hinged node
- NFRSC = 2 for hinged horizontal roller

NAS: node number at which NFRSC applies ($\leq$ N+1)

} 6I4

(XPOIS(I),I = 1,N): conical element Poisson ratios		9F8.4
(XE(I),I = 1,N): conical element Young's moduli		4E18.8
(HH(I),I = 1,N1): conical element thicknesses at nodes		9F8.4
<u>If NS = ± 1:</u>	HD, BETA, R: (see Figure 8)	3E18.8
	(XA(I),I = 1,N): element arcs (^{o})	9F8.4
	(XALPH(I),I = 1,N): element angles (^{o})	9F8.4
<u>If NS = 2:</u>	(XN(I),I = 1, N1): x co-ordinates of nodes	4E18.8
	(YN(I),I = 1,N1): y co-ordinates of nodes	4E18.8
<u>If NS = 0:</u>	(XSI(I), XSJ(I), I = 1,N): s_i,s_j	4E18.8
	(XALPH(I), I = 1,N): α in degrees	9F8.4
<u>If NSØ ≠ 0:</u>	RR, TØ, ALPHØ, PØISØ, EØ: disc data	5E14.7
<u>If NSØ ≠ 0 and NSD ≠ 1:</u>	XWØ: unit weight of disc	E18.8
<u>If NSD ≠ 1:</u>	GA: acceleration of gravity	E18.8
	(XW(I), I = 1,N): unit weight of cones	4E18.8
	NEP: number of modes to be used in mode superposition (≤ 3N + NFRSC)	
	NAF: dynamic force type code NAF = 0 for free vibration NAF = 1 for applied forces NAF = 2 for applied accelerations	2I4
	MM, NT: integration data	2I4
	TIME: total time for integration	E18.8
	(AF(I), I = 1,NN): nodal load amplitudes	4E18.8
	(FA(I), I = 1,MM1): force time factors	10F7.4
<u>If NSD = 1:</u>	(AF(I), I = 1,NN): nodal loads	4E18.8

4. Output

The output quantities include echo prints of all the input data as well as printed intermediate and final results. In each case, the output is identified when printed. Any difficulties with interpreting the identification terminology can be resolved using the glossary below and by studying the flow chart and program listing. Therefore, a listing of all output data will not be provided here.

C. Glossary of FORTRAN Variable Names

Following is a partial glossary of the terms used in the computer program. All input and output quantities are included and also those terms directly relevant to the basic theory of the proposed method. Intermediate variable names are not all defined since they are often mere computational entities or temporary storage devices. The symbols used in the definitions are those used in Chapter I and the Illustrations and are listed at the beginning of the report.

A	= portion of increment to velocity during integration
ACEL	= acceleration at a given time during response
AF( )	= amplitude vector for joint loads; static nodal loads
AI	= floating value of index I
AI(,),AJ(,)	= force transformation matrices
AIS( ),AJS( )	= nodal forces at top and bottom nodes in global co-ordinates for element in question (in L7, these are for a particular time)
AIX	= floating value of IX
AL	= $\ell = \lvert (r_j - r_i)/\cos\alpha \rvert$

ALPH	= angle in degrees for element in question
ALPHA	= angle in radians for element in question
ALPHØ	= semi-angle of closure element in degrees
AM	= $2\pi t$ w/g for cone
AMØ	= $\pi t_o \, w_o/g$ for closure element
ANT	= floating valut of NT
ATALP	= $\lvert \tan \alpha \rvert$
B	= portion of increment to displacement during integration
BE	= $y/ \, 2^{1/2}$ for Thomson function computation
BERYI,BEIYI, BERYJ,BEIYJ	= ber y_i etc. for conical element
BER( )	= scaled down magnitudes of Thomson functions; first computed value of Thomson functions (8x1)
BETA	= angle in degrees from the horizontal of a line joining the center of the arc and the top of the first element (Figure 8)
BI(,),BJ(,)	= displacement transformation matrices
C	= non-integer index in Thomson function computation
CALP	= $\cos \alpha$
CI(,)	= transformation matrix $[C_i]$
CID( ),CJD( )	= nodal displacements for top and bottom of element in question in local co-ordinates (in L7, these are for a given time) (3x1)
CIS( ),CJS( )	= nodal forces for top and bottom of element in question in local co-ordinates (in L7, these are for a given time) (3x1)
DBERYI,DBERYJ, DBEIYI,DBEIYJ	= ber'y_i etc. for conical element

DDØ	= $Et^3/12(1-\nu^2)$ for closure element
DER(,)	= Thomson function array (2x8)
DI( ),DJ( )	= displacements at top and bottom of cone in global co-ordinates (3x1)
DISP	= displacement at a given time during response
DT	= adjustable time increment for time integration
DXERYI,DXERYJ, DXEIYI,DXEIYJ	= ker'y_i etc. for conical element
DXP	= increment of load time factor
DY	= horizontal distance from shell axis to center of arc (Figure 8)
E	= Young's modulus for element in question
EK(,)	= overall stiffness matrix; storage for eigenvectors $[\phi]$; inverted overall stiffness matrix; array of joint inertial loads $[M]$ $[\phi]$ $[\omega^2]$; storage for displacement response of each mode at each time
EKII(,),EKIJ(,), EKJI(,),EKJJ(,)	= $\bar{k}_{ii}$ etc. for conical element in question
EKØ	= end closure element flexibility or stiffness
EKT(,)	= matrix of eigenvectors $[\bar{\Phi}]$; storage for $[\phi]^T$ $[K]$; storage for $[\phi]$ [XDEF]; storage for mode shapes $[\phi]$; storage for inverse of overall stiffness
EØ	= Young's modulus for closure element
EVL( )	= vector of eigenvalues ω^2
EVL1( )	= vector of eigenvalues $\bar{\omega}^{-2}$; storage for vector of $\bar{\omega}^{-1}$
EVT(,)	= matrix of eigenvectors $[\bar{\bar{\Phi}}]$; storage for $[\bar{\omega}^{-1}]$ $[\bar{\bar{\Phi}}]$
F(,)	= element flexibility or stiffness
FA( )	= time factor of joint loads

FREQ	= arbitrary fraction (1/32nd) of period
FREQ( )	= vector of natural frequencies in cycles per second
GA	= acceleration due to gravity
HD	= horizontal radius at top of first conical element (Figure 8)
HH( )	= thicknesses of conical elements at nodes
I, II	= indices
INS	= NS^2 = code for cone geometry input
IX	= counter for change in time increment for integration
J	= index
JØB	= index
K	= index
L	= index
MA(,)	= modified power of 10 for scaled Thomson functions
MC(,)	= power of ten by which Thomson functions are scaled
MJ,MJM	= indices for selecting proper displacements for conical element
MM	= number of time intervals at which response is to be calculated
MN	= MN/NT = number of time intervals at which response is to be recorded for output; number of Thomson function arguments (MN = 1 or 2)
N	= number of conical elements
NAF	= nodal force/acceleration code
NAS	= location (node number) of restrained node
NASA	= 3(NAS − 1) + NFRSC = nodal degree of freedom number after which there are no restraints

NEP	= number of modes to be used in the mode superposition method
NFRSC	= number of degrees of freedom at restrained node
NI, NJ	= indices for selection of restraint location
NN	= 3N + NFRSC = total number of degrees of freedom
NØUT	= counter to indicate when response should be recorded for output
NS	= cone geometry input code
NSD	= static/dynamic problem code
NSØ	= end closure code
NT	= frequency of time intervals at which response should be recorded for output (e.g., NT = 3 for every third interval calculated)
P( )	= vector of natural frequencies in radians per second, ω
PØIS	= Poisson ratio for element in question
PØISØ	= Poisson ration for end closure element
P1(,)	= array of element forces in local co-ordinates due to static loading (Nx5)
P2(,)	= array of element forces in local co-ordinates for a particular unit mode due to inertial loading (Nx5)
P3(,)	= array of amplified static element forces in local co-ordinates at a particular time (Nx5)
QKQ(,)	= $[\phi]^T[K][\phi] = [\omega^2]$, as a check on eigenvalue calculations
QT(,)	= matrix of eigenvectors (normal mode shapes)
R	= radius of element arc
RA	= $(y_i - y_j)/(x_i - x_j) = \tan\alpha$ for element in question

RI, RJ	=	horizontal radii at top and bottom of element, r_i and r_j
RØ	=	horizontal radius r_o for end closure element
RR	=	radius of curvature of end closure element (Figures 3 and 5)
S( )	=	element forces in local co-ordinates (5x1)
SALP	=	$\sin \alpha$
SI, SJ	=	slope lengths to top and bottom of element, s_i and s_j
SKØ	=	$2\pi r_o$ times end closure stiffness
SØ( )	=	$[k_o]\ \{u_o\}$ = end closure node moment and horizontal force (2x1)
SØ1( )	=	end closure node moment and horizontal force due to static load (2x1)
SØ2( )	=	end closure node moment and horizontal forces for a particular unit mode due to joint inertial loads (2x1)
SØ3( )	=	amplified end closure node moment and horizontal forces due to static effects at a particular time (2x1)
SS	=	slope length to the middle of the conical element in question
SSØ2(,)	=	array of SØ2 vectors for various unit modes (NEPx2)
T( )	=	vector of times at which response is given
TIFS	=	amplification factor for static response at a particular time
TIME	=	total time for which response is desired
TØ	=	thickness of end closure element
TP(,)	=	for a given time, the total element force response in local co-ordinates due to inertial loads; for a given time, the element force response in local co-ordinates due to both static and inertial loads

TSØ( )	= force response at end closure node at a particular time due to both static and inertial loads (2x1)
TSØ2( )	= total force response at end closure node at a particular time due to inertial loads (2x1)
TT	= average thickness of element in question = $(t_i + t_j)/2$; time increment
TU( )	= nodal displacements in global co-ordinates at a particular time due to both static and inertial loads (NNx1)
TU2( )	= total nodal displacements in global co-ordinates at a particular time due to inertial loads (NNx1)
U1( )	= nodal displacements in global co-ordinates due to static loads (NNx1)
U2( )	= nodal displacements in global co-ordinates for a particular unit mode due to inertial forces (NNx1)
U3( )	= amplified static displacement response in global co-ordinates at a particular time (NNx1)
V( )	= conical element displacements in local co-ordinates (5x1)
VEL	= velocity at given time during integration
W	= weight density for element in question
X	= $\ell/2$ = semi slope length of element
XA( )	= arcs of cones in degrees
XALPH( )	= cone angles in degrees
XBER( )	= Thomson functions (16x1)
XCALP( )	= cos α for conical elements
XDEF(,)	= displacement response of each mode at each time
XE( )	= Young's modulus for cones
XERYI, XERYJ, XEIYI, XEIYJ	= ker y_i etc. for conical elements

XEK(,)	= element stiffness
XEKT(,)	= symmetrized element stiffness in local co-ordinates
XM(,)	= overall mass matrix; storage for $[\bar{K}]$
XMAF( )	= [M] {AF} where {AF} is acceleration input
XMII(,),XMIJ(,) XMJI(,),XMJJ(,)	= $\bar{m}_{ii}$ etc. for element
XMØ(,)	= end closure element mass matrix
XMTII(,),XMTIJ(,), XMTJI(,),XMTJJ(,)	= m_{ii} etc. for element
XN( )	= x co-ordinate of node
XNR( )	= displacement response of all modes in normal co-ordinates at a given time, i.e., $\{\eta(t)\}$
XNR1( )	= generalized force amplified and modified for a given time, i.e., $\{P^*(t)/\omega^2\}$
XN2( )	= $-\{\ddot{\eta}(t)/\omega^2\}$ for a given time, i.e., XNR - XNR1
XP	= time/factor of load at a given time during integration
XPØIS( )	= Poisson ratios for conical elements
XQFT( )	= generalized force vector $([\phi]^T$ {XMAF} if {AF} is acceleration input; $[\phi]^T$ {AF} if {AF} is force input
XRI( ),XRJ( )	= r_i and r_j for conical elements
XSALP( )	= sin α for conical elements
XSI( ),XSJ( )	= s_i and s_j for conical elements
XW( )	= unit weights of conical elements
XWØ	= unit weight of end closure element
XXEKT(,,)	= array of element stiffnesses
XX	= $x_i - x_j$ for element in question

XXYY = length of element in question = $((x_i - x_j)^2 + (y_i - y_j)^2)^{\frac{1}{2}}$

Y( ) = Thomson function arguments (1x2)

YI,YJ = Thomson function arguments for element in question

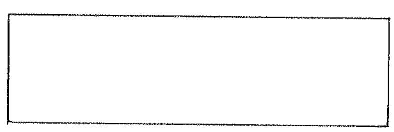
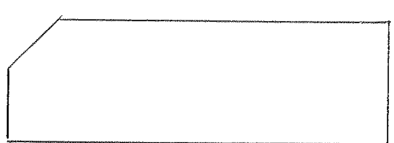
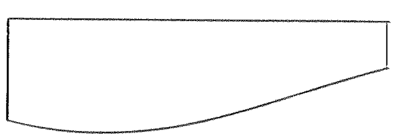
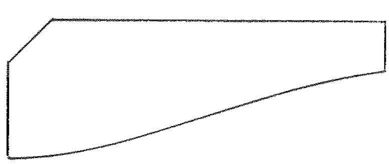
YN( ) = y co-ordinate of node

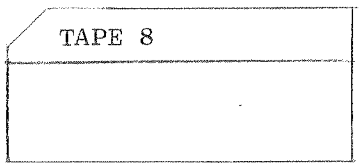
YY = $y_i - y_j$ for element in question

YS = floating value of NS

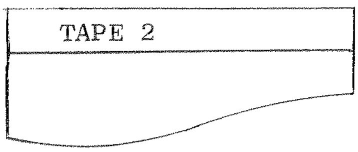
D. Flow Chart Symbols

Following are the conventions used for the flow chart in this report. An attempt has been made to keep the flow chart as simple as possible without loss of detail.

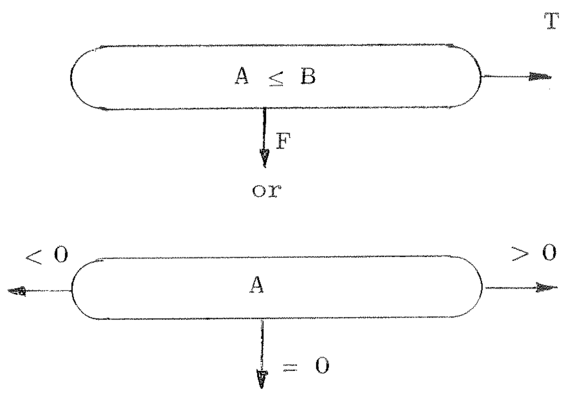
Symbol	Meaning
	Substitution statement; subroutine call; basic computational step; or tape control statement
	Input statement
	Output statement
	Read and echo print



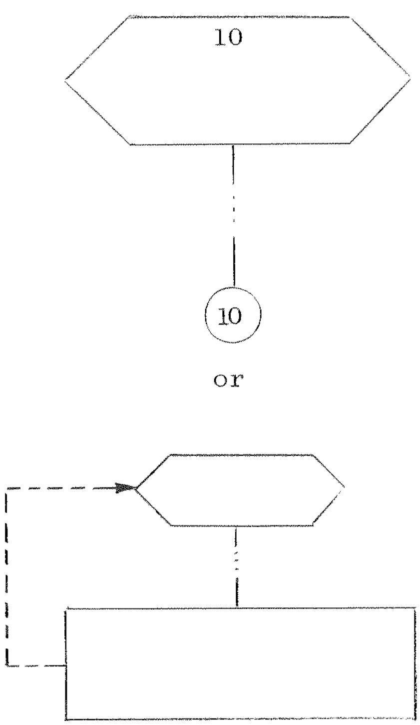
Input from tape



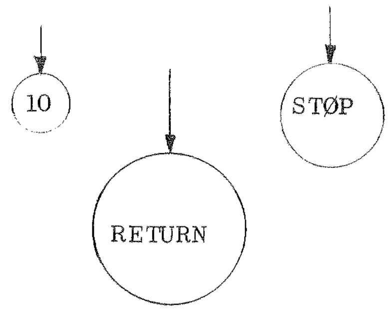
Output onto tape



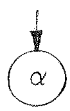
Conditional control statement



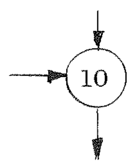
Iterative control statement
(do loop)



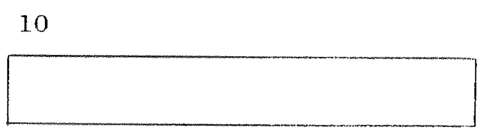
Unconditional control statements



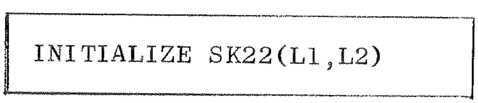
Drawing link



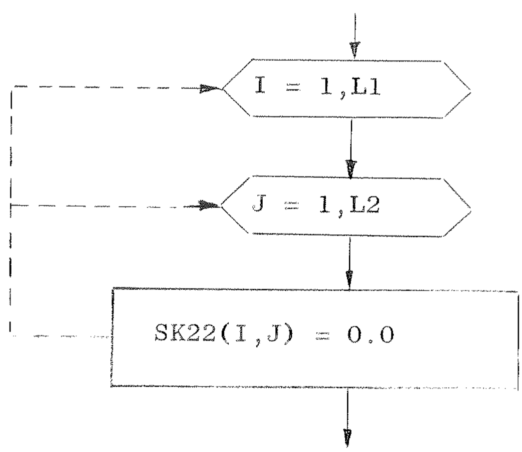
Statement label at junction point



Statement label at substitution statement

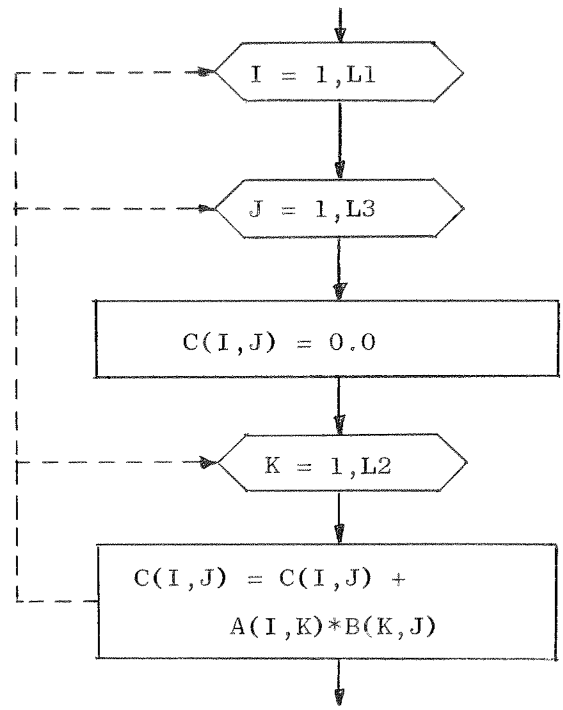


Set all elements of the L1xL2 SK22 array equal to zero, i.e.,



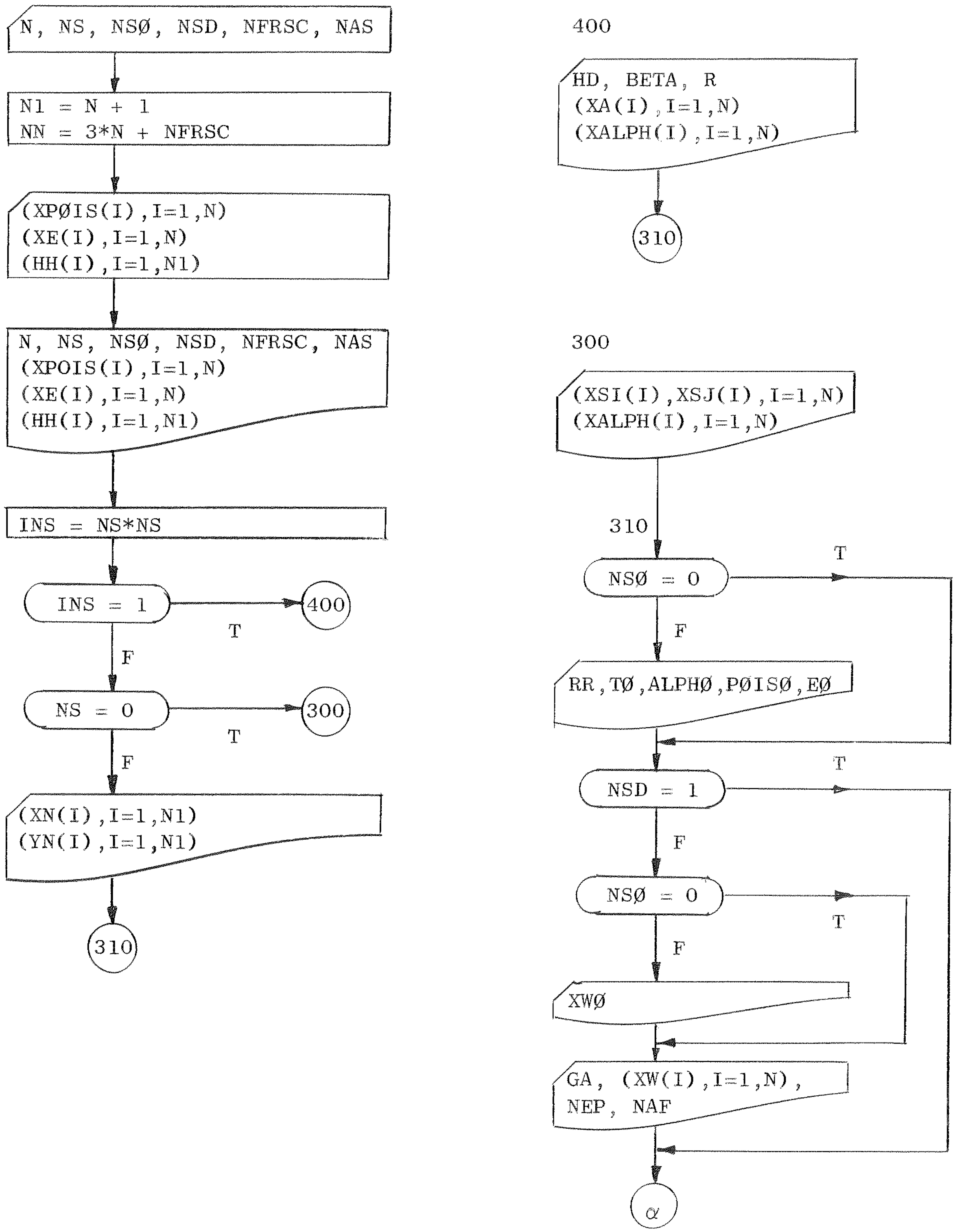
C(L1,L3)=A(L1,L2)*B(L2,L3)

Perform the matrix (or vector) multiplication [C] = [A][B], i.e.,

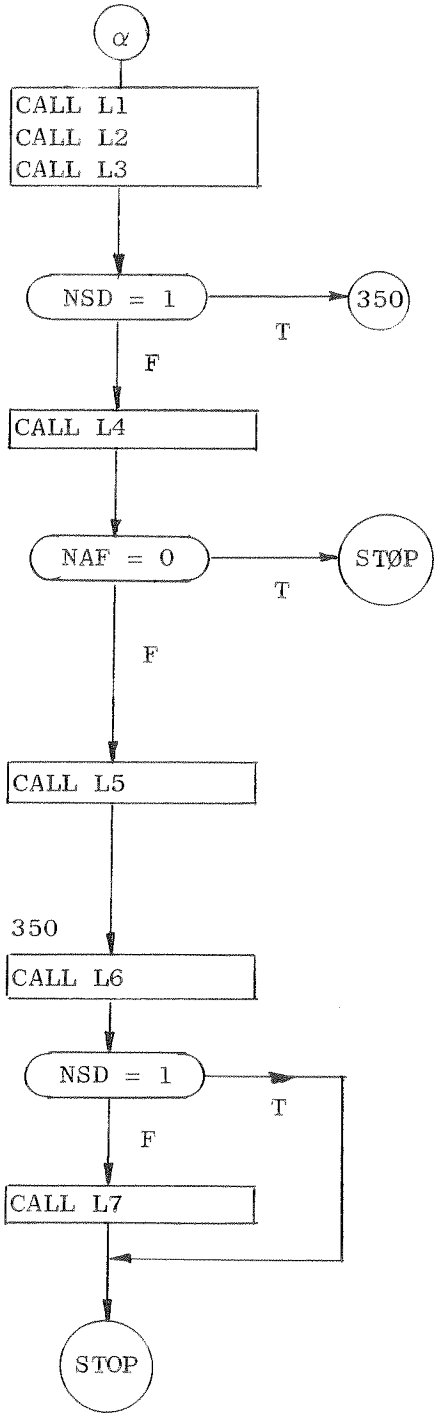


E. Flow Chart

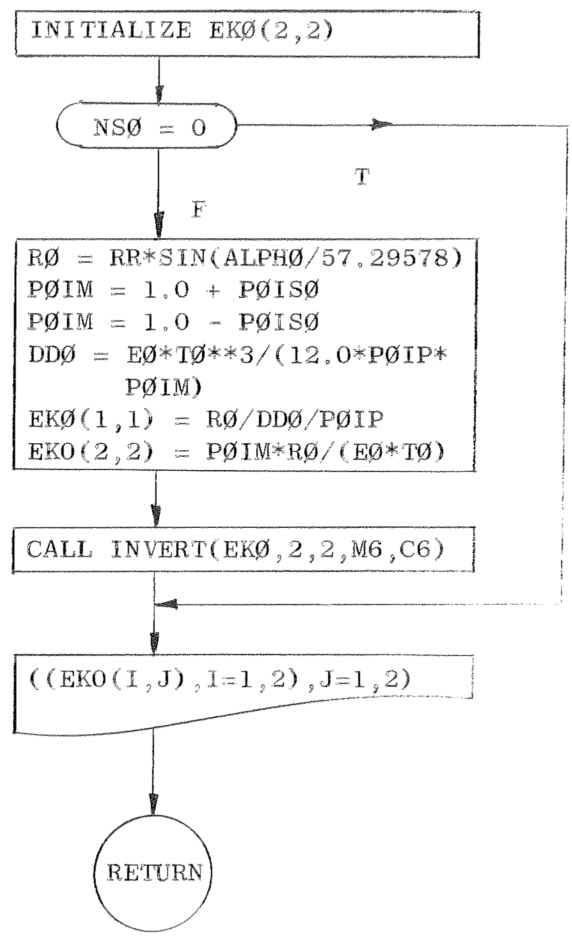
MAIN



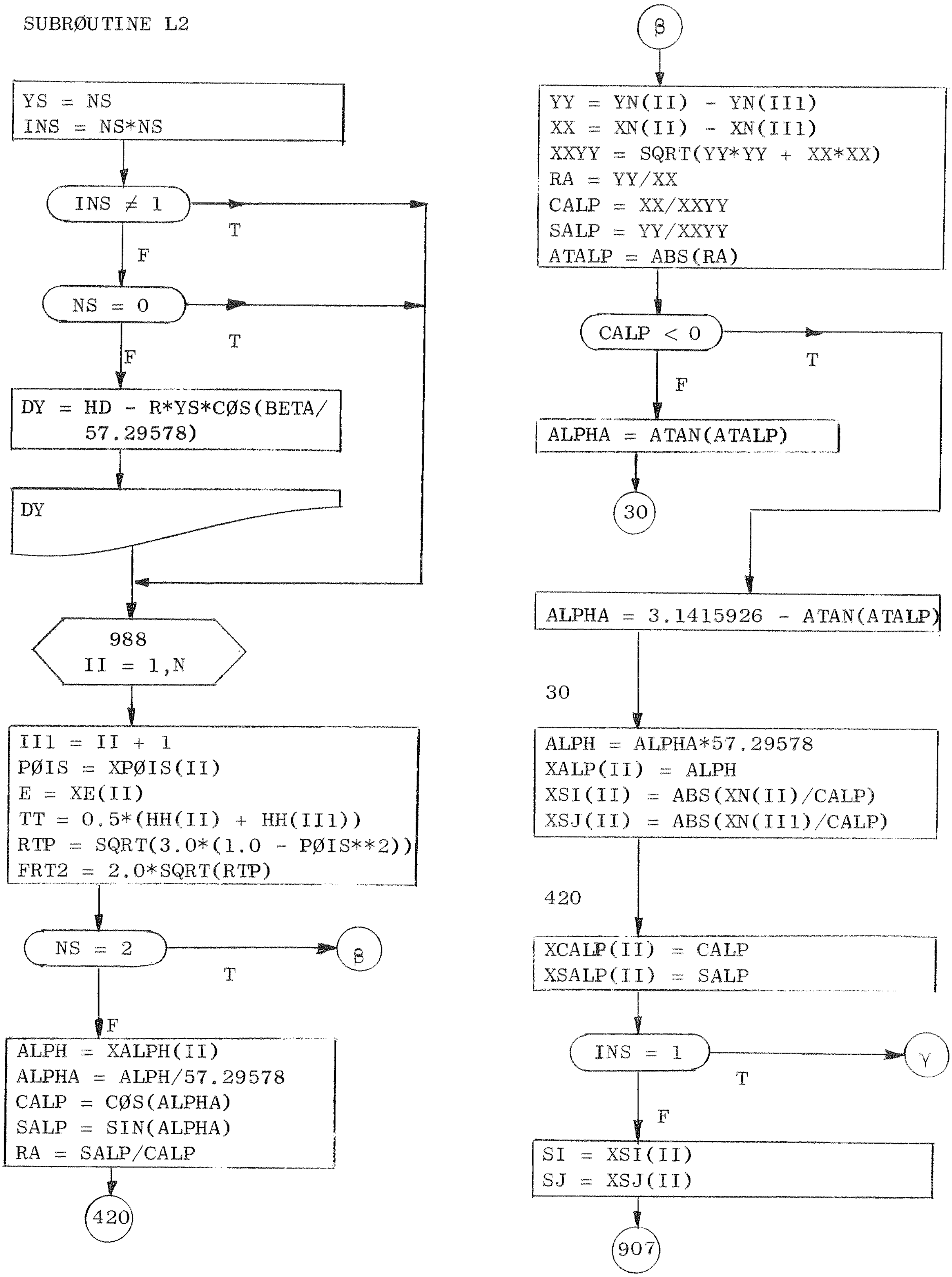
MAIN(continued)



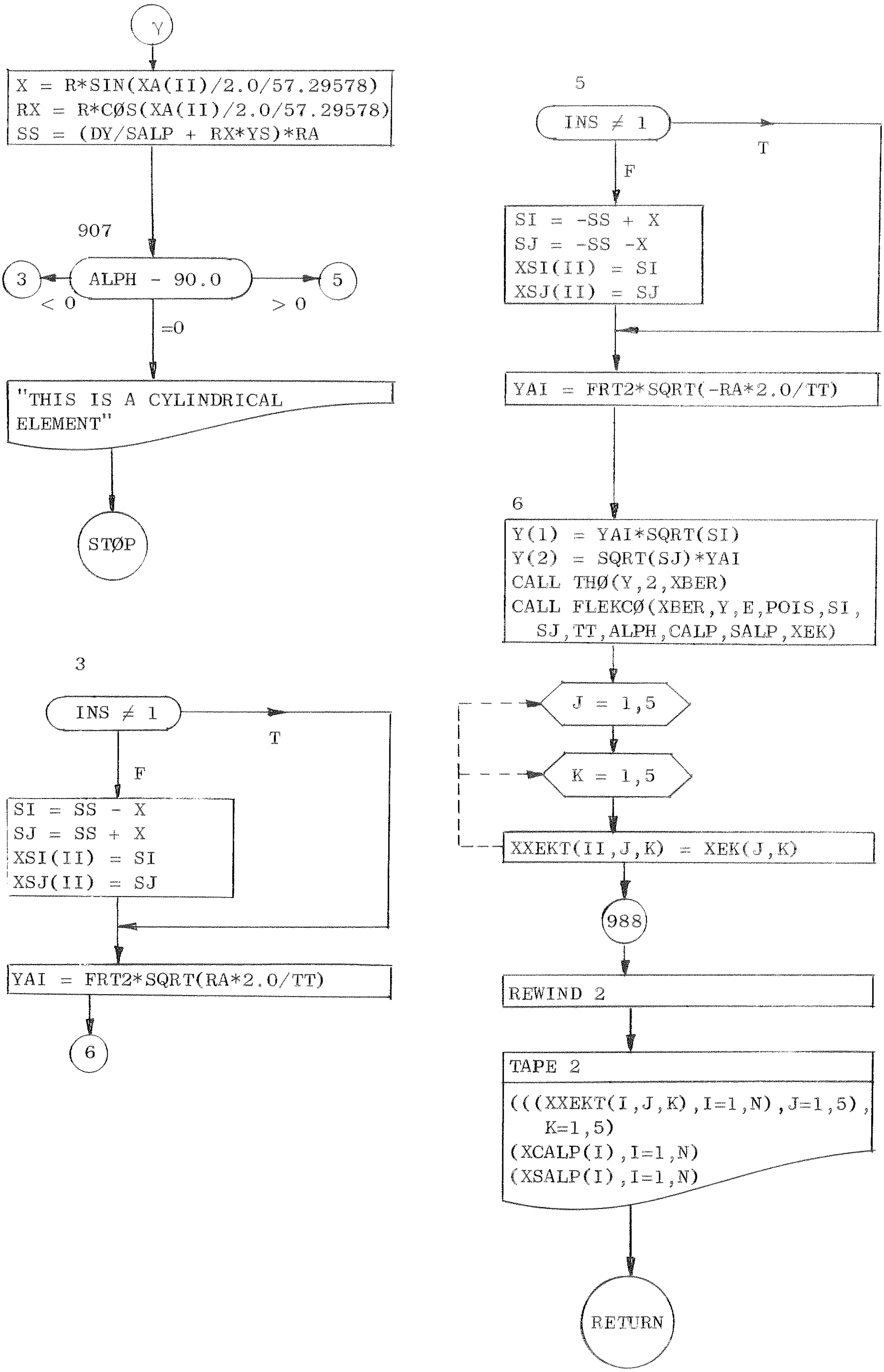
SUBRØUTINE L1

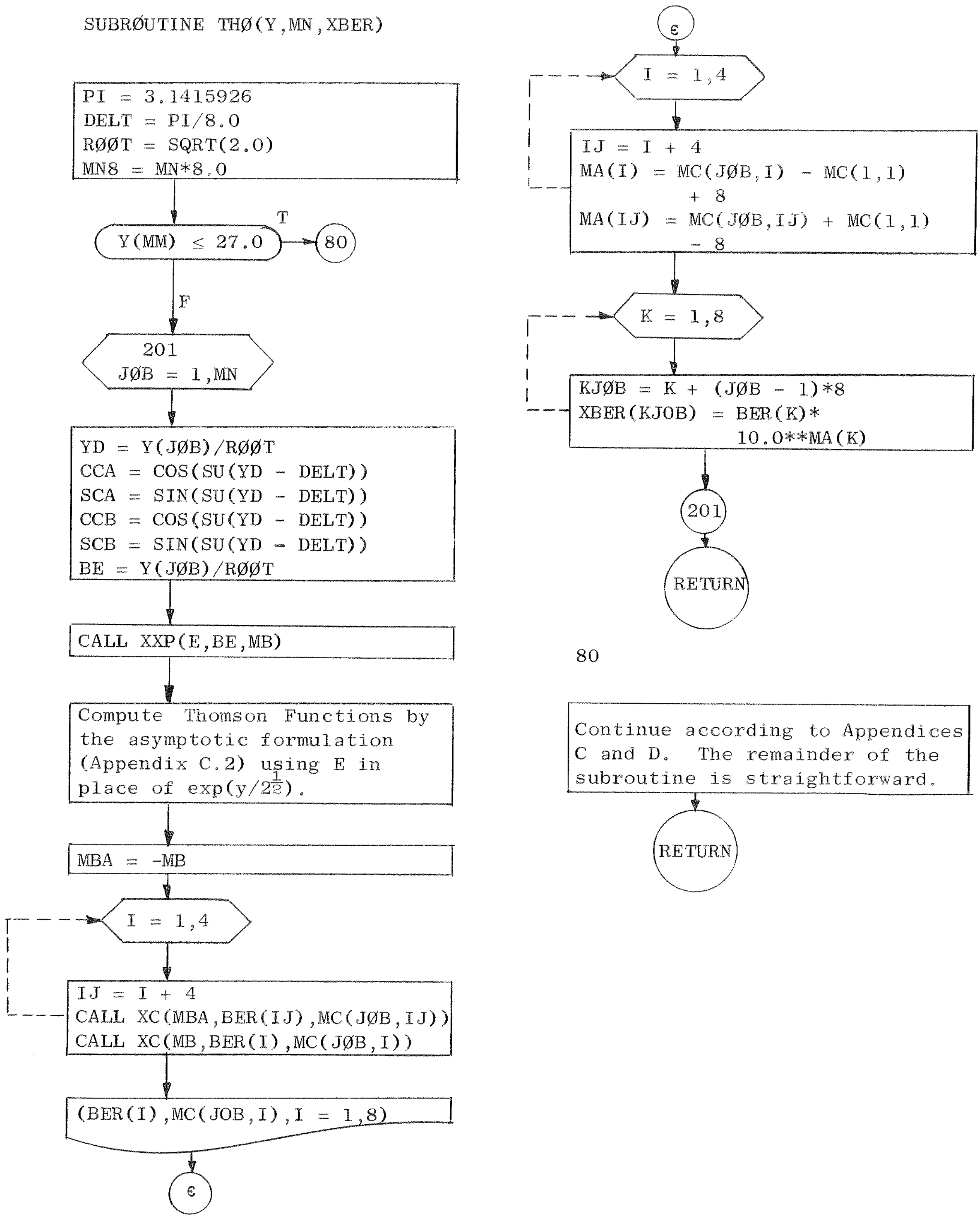


SUBRØUTINE L2

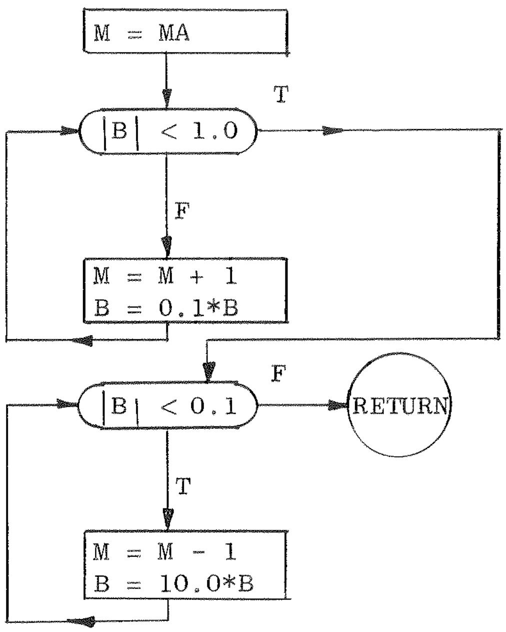


L2(continued)

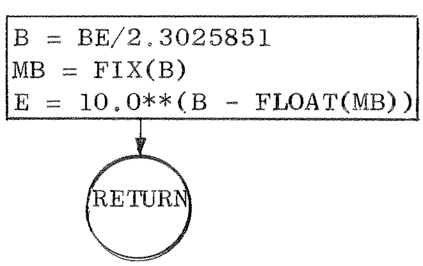



SUBRØUTINE THØ(Y,MN,XBER)
PI = 3.1415926
DELT = PI/8.0
RØØT = SQRT(2.0)
MN8 = MN*8.0
Y(MM) ≤ 27.0
T
80
F
201
JØB = 1,MN
YD = Y(JØB)/RØØT
CCA = COS(SU(YD - DELT))
SCA = SIN(SU(YD - DELT))
CCB = COS(SU(YD - DELT))
SCB = SIN(SU(YD - DELT))
BE = Y(JØB)/RØØT
CALL XXP(E,BE,MB)
Compute Thomson Functions by the asymptotic formulation (Appendix C.2) using E in place of exp(y/2½).
MBA = -MB
I = 1,4
IJ = I + 4
CALL XC(MBA,BER(IJ),MC(JØB,IJ))
CALL XC(MB,BER(I),MC(JØB,I))
(BER(I),MC(JOB,I),I = 1,8)
ε
ε
I = 1,4
IJ = I + 4
MA(I) = MC(JØB,I) - MC(1,1) + 8
MA(IJ) = MC(JØB,IJ) + MC(1,1) - 8
K = 1,8
KJØB = K + (JØB - 1)*8
XBER(KJOB) = BER(K)* 10.0**MA(K)
201
RETURN
80
Continue according to Appendices C and D. The remainder of the subroutine is straightforward.
RETURN

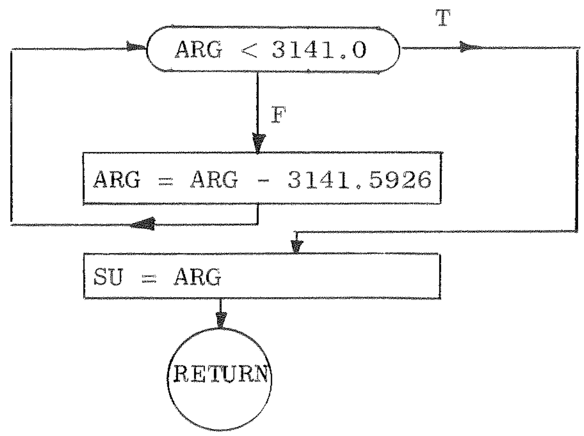
SUBRØUTINE XC(MA,B,M)

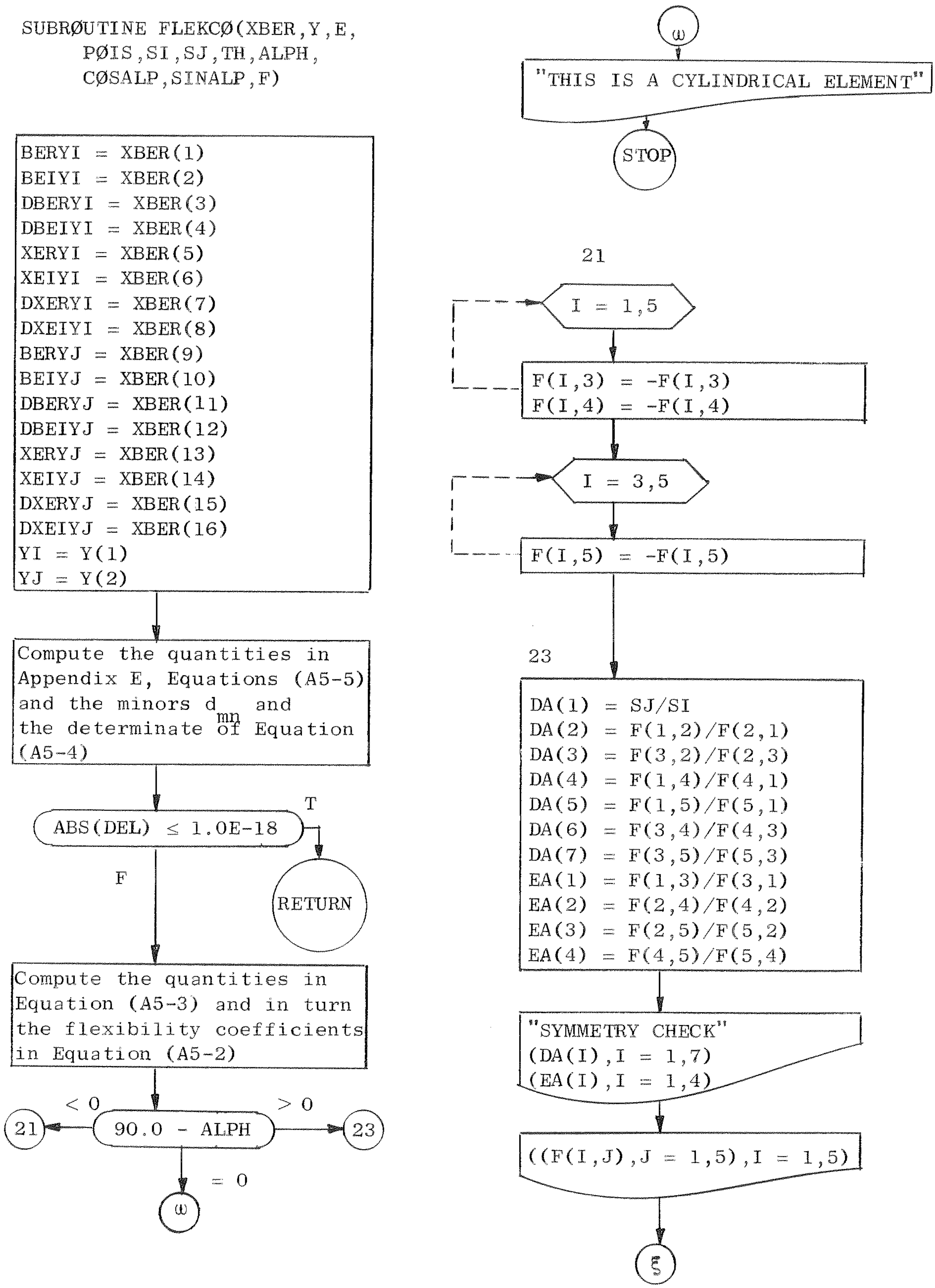


SUBRØUTINE XXP(E,BE,MB)

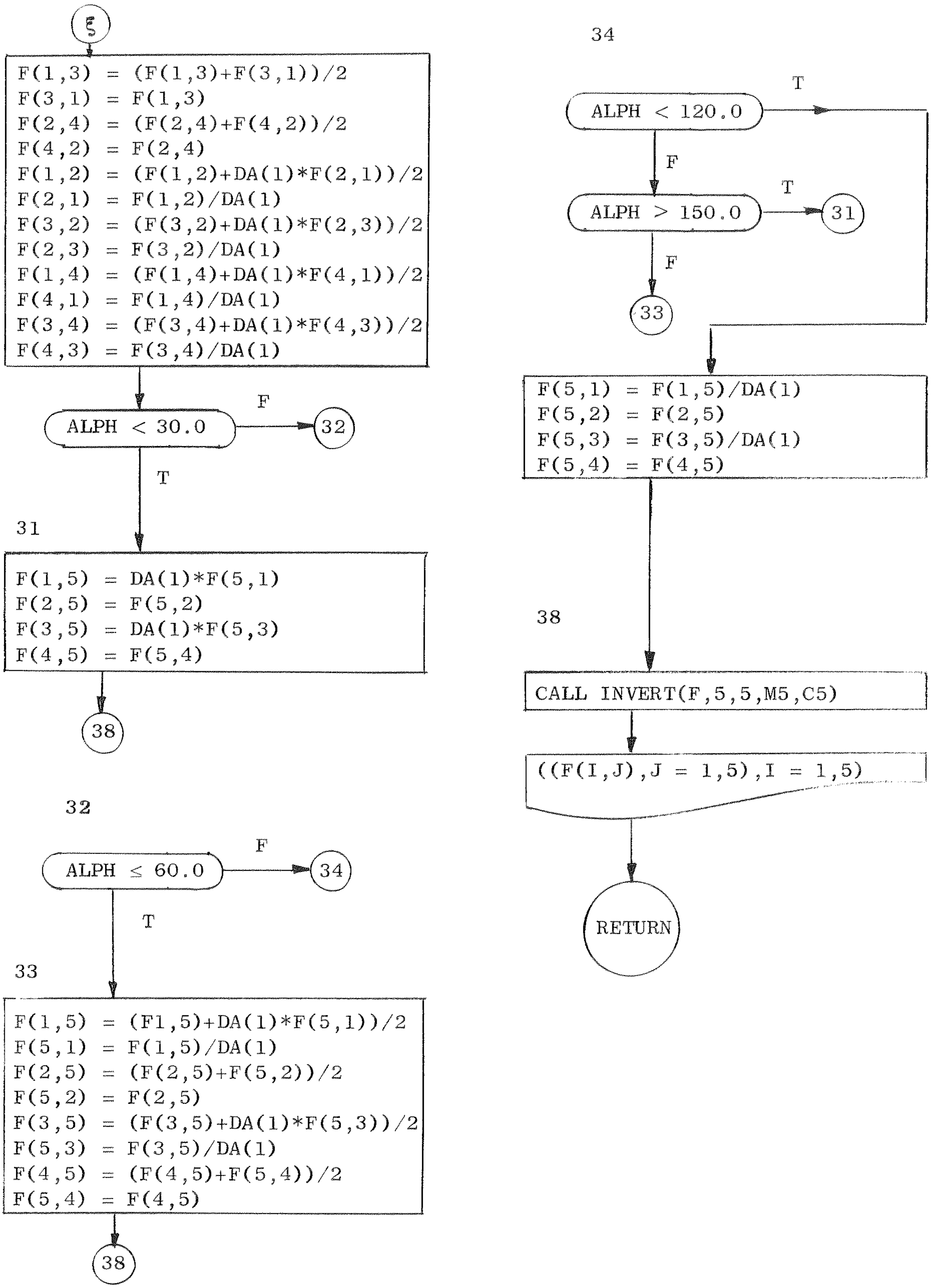


FUNCTIØN SU(ARG)

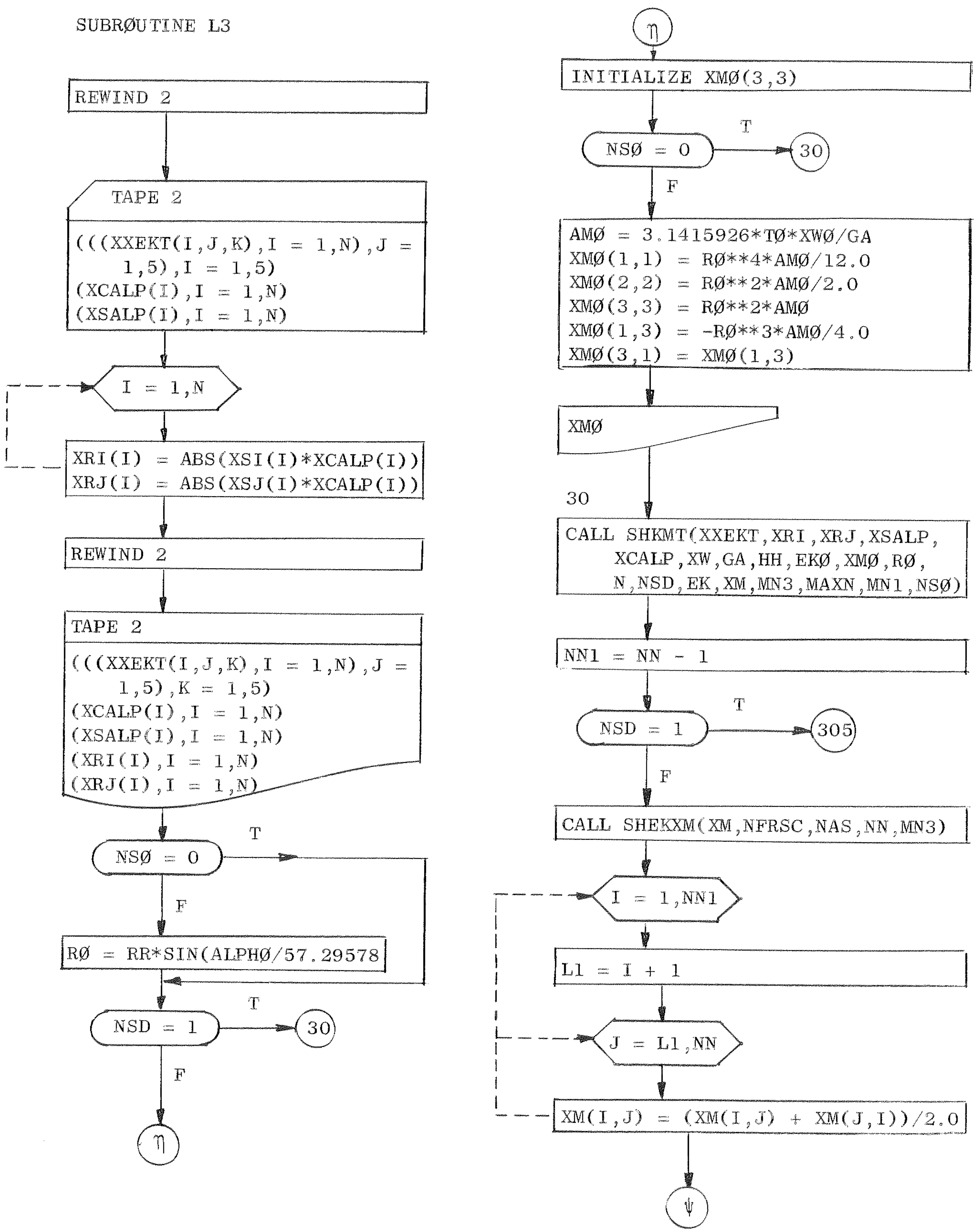



SUBRØUTINE FLEKCØ(XBER,Y,E,
PØIS,SI,SJ,TH,ALPH,
CØSALP,SINALP,F)
BERYI = XBER(1)
BEIYI = XBER(2)
DBERYI = XBER(3)
DBEIYI = XBER(4)
XERYI = XBER(5)
XEIYI = XBER(6)
DXERYI = XBER(7)
DXEIYI = XBER(8)
BERYJ = XBER(9)
BEIYJ = XBER(10)
DBERYJ = XBER(11)
DBEIYJ = XBER(12)
XERYJ = XBER(13)
XEIYJ = XBER(14)
DXERYJ = XBER(15)
DXEIYJ = XBER(16)
YI = Y(1)
YJ = Y(2)
Compute the quantities in Appendix E, Equations (A5-5) and the minors d mn of the determinate of Equation (A5-4)
ABS(DEL) ≤ 1.0E-18
T
RETURN
F
Compute the quantities in Equation (A5-3) and in turn the flexibility coefficients in Equation (A5-2)
< 0
> 0
21
90.0 - ALPH
23
= 0
ω
ω
"THIS IS A CYLINDRICAL ELEMENT"
STOP
21
I = 1,5
F(I,3) = -F(I,3)
F(I,4) = -F(I,4)
I = 3,5
F(I,5) = -F(I,5)
23
DA(1) = SJ/SI
DA(2) = F(1,2)/F(2,1)
DA(3) = F(3,2)/F(2,3)
DA(4) = F(1,4)/F(4,1)
DA(5) = F(1,5)/F(5,1)
DA(6) = F(3,4)/F(4,3)
DA(7) = F(3,5)/F(5,3)
EA(1) = F(1,3)/F(3,1)
EA(2) = F(2,4)/F(4,2)
EA(3) = F(2,5)/F(5,2)
EA(4) = F(4,5)/F(5,4)
"SYMMETRY CHECK"
(DA(I),I = 1,7)
(EA(I),I = 1,4)
((F(I,J),J = 1,5),I = 1,5)
ξ

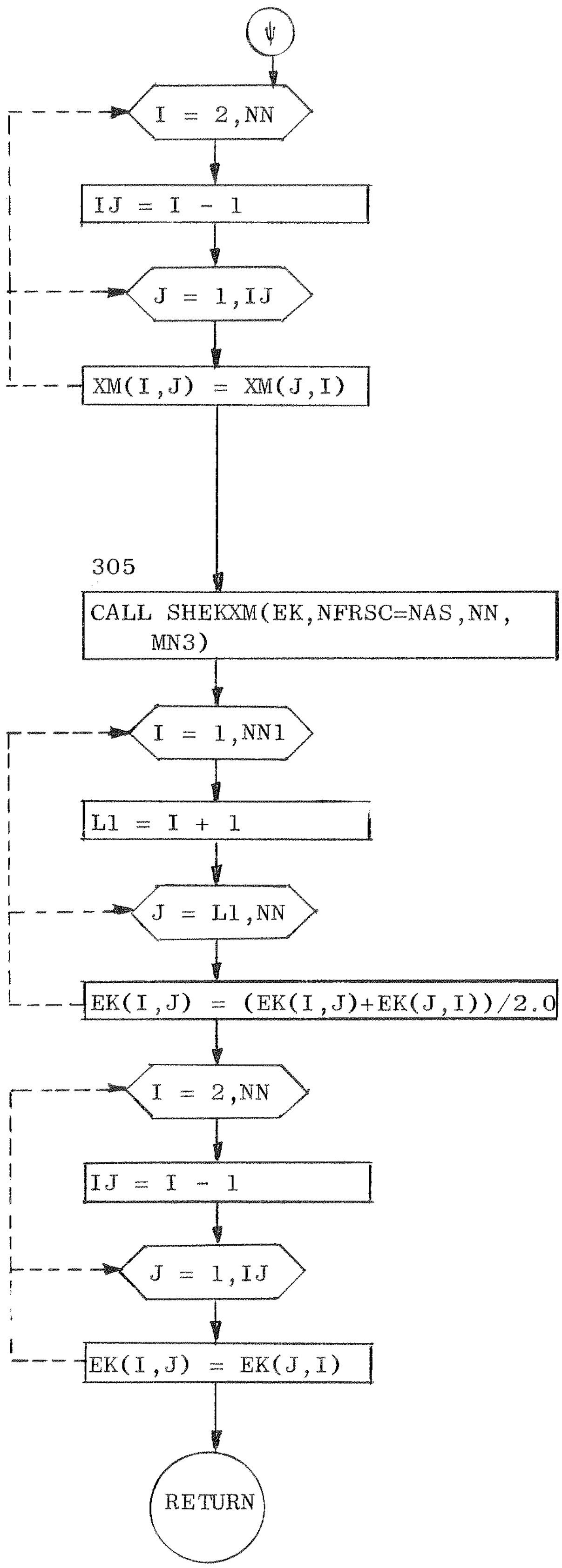
FLEKCØ (continued)



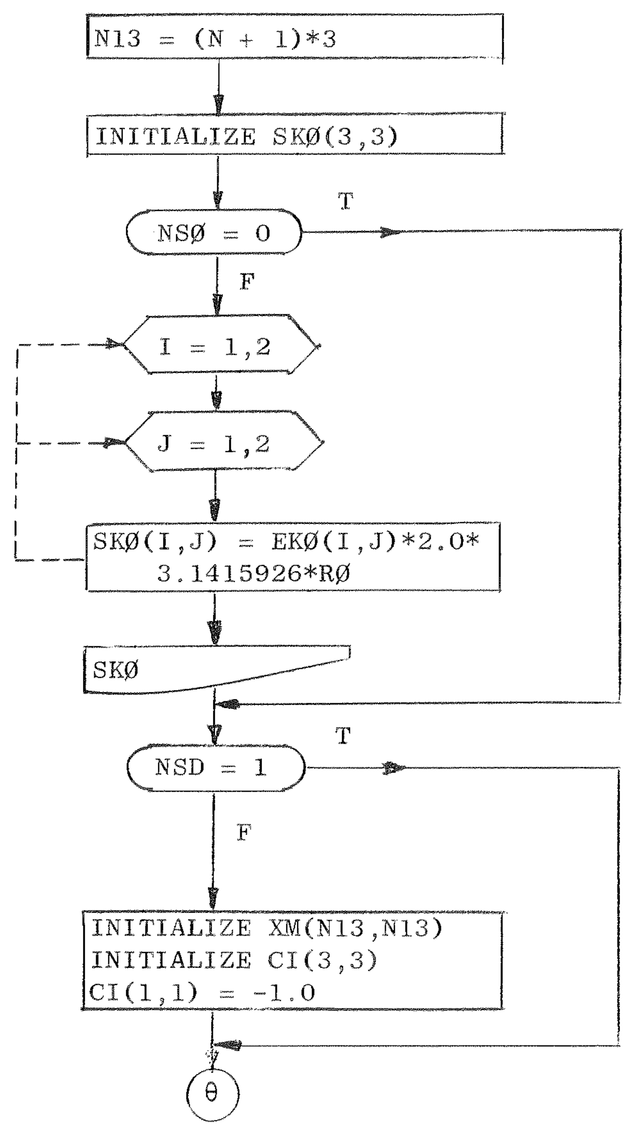
SUBRØUTINE L3



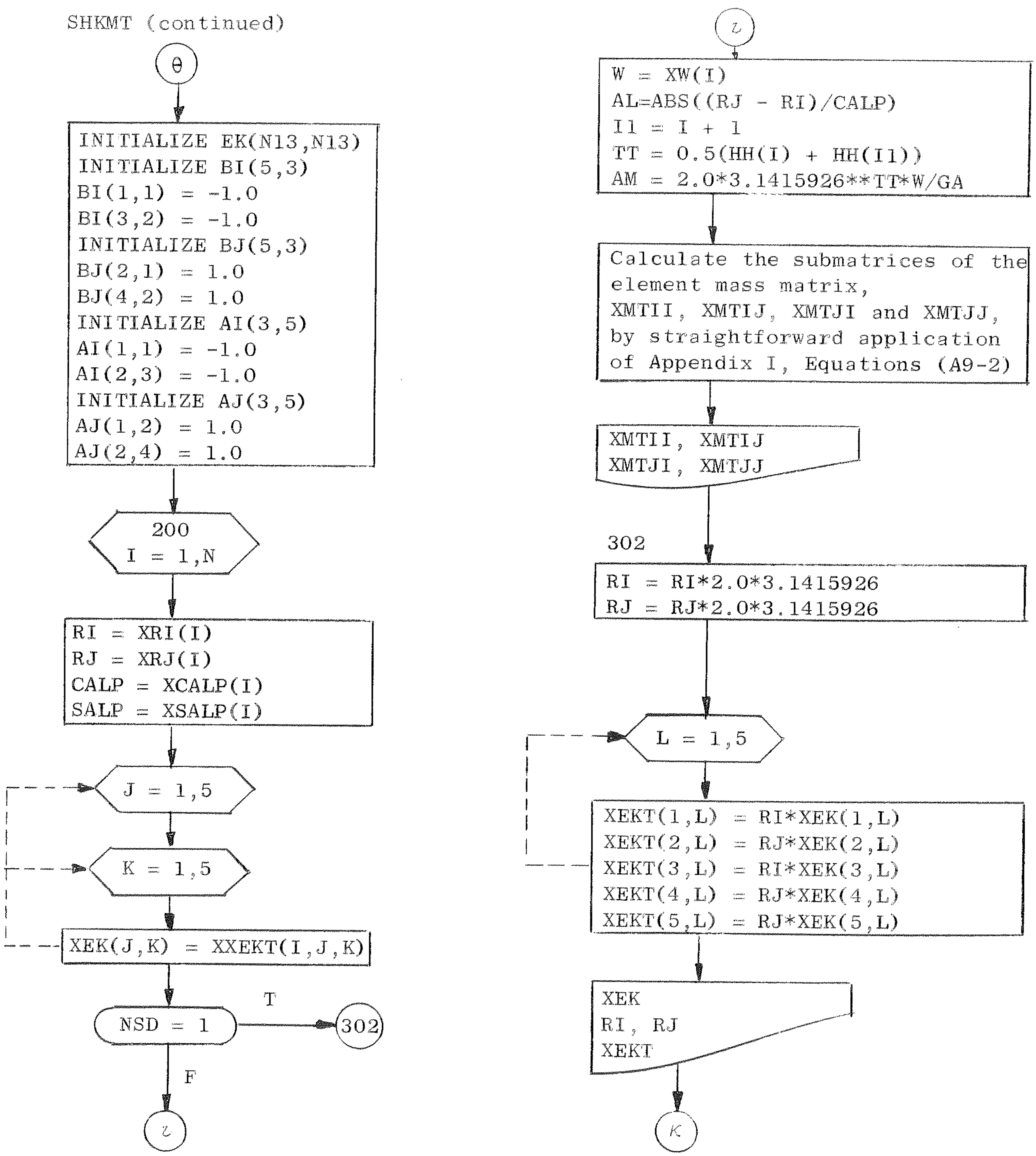
L3 (continued)



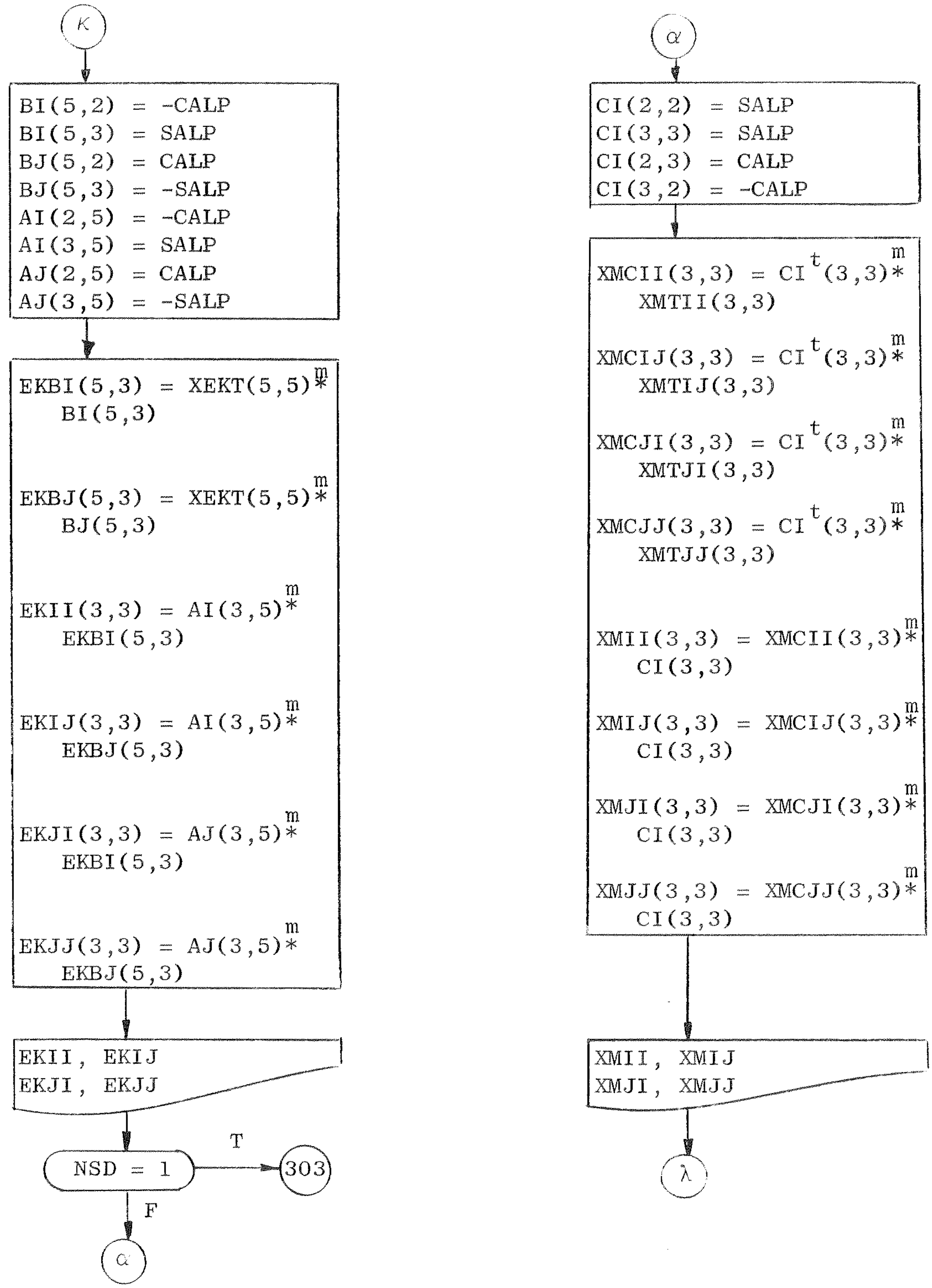
SUBRØUTINE SHKMT(XXEKT,XRI,
XRJ,XSALP,XCALP,XW,GA,HH
EKØ,XMØ,RØ,N,NSD,EK,XM,
MN3,MAXN,MN1,NSØ)



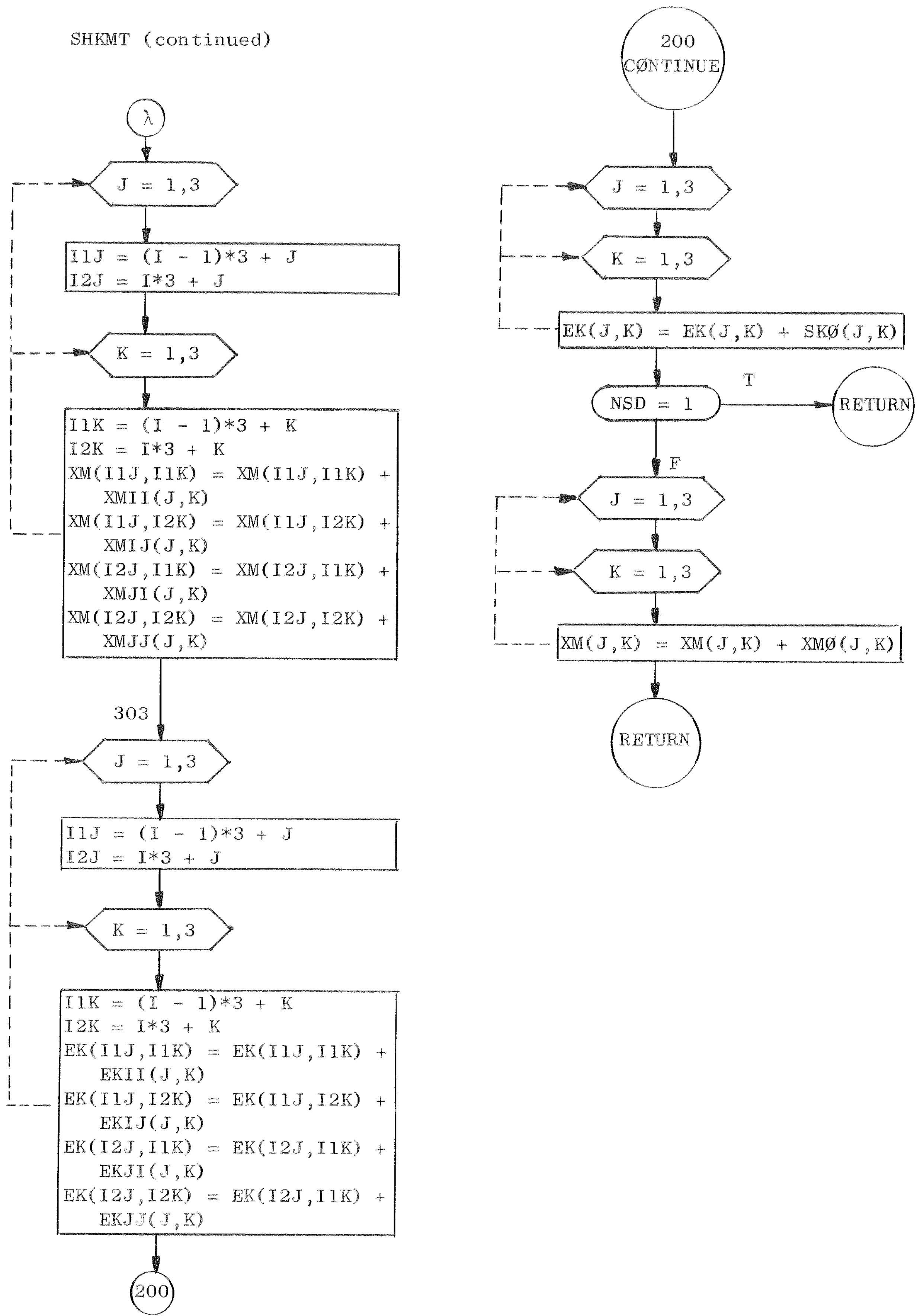
SHKMT (continued)



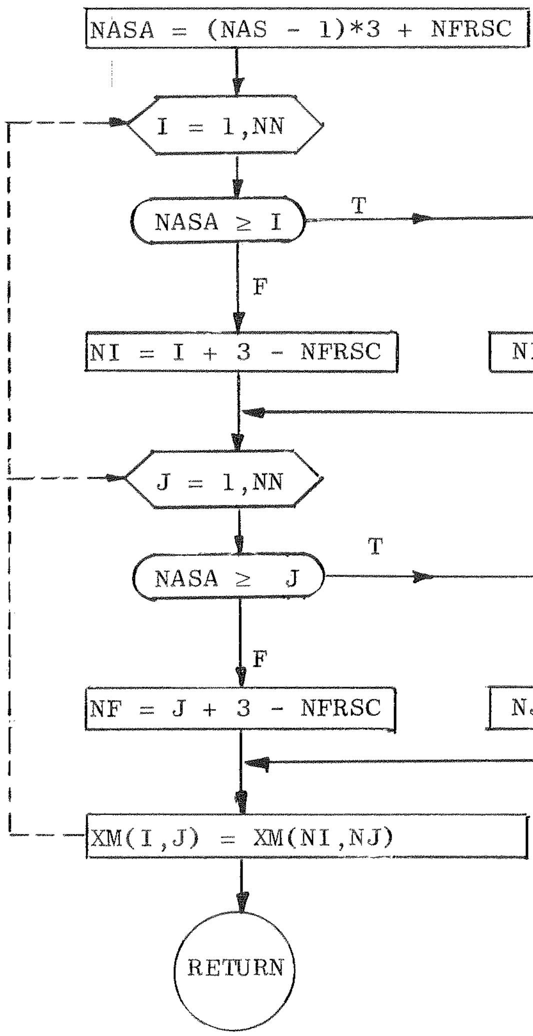
SHKMT (continued)



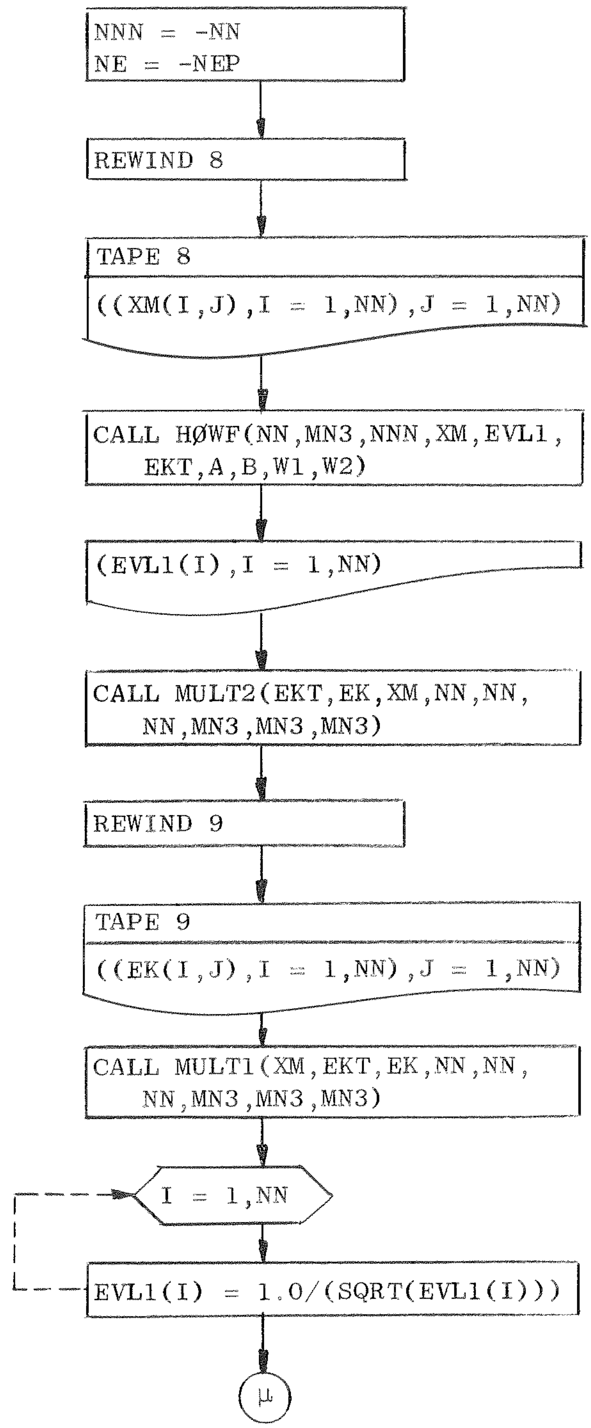
SHKMT (continued)



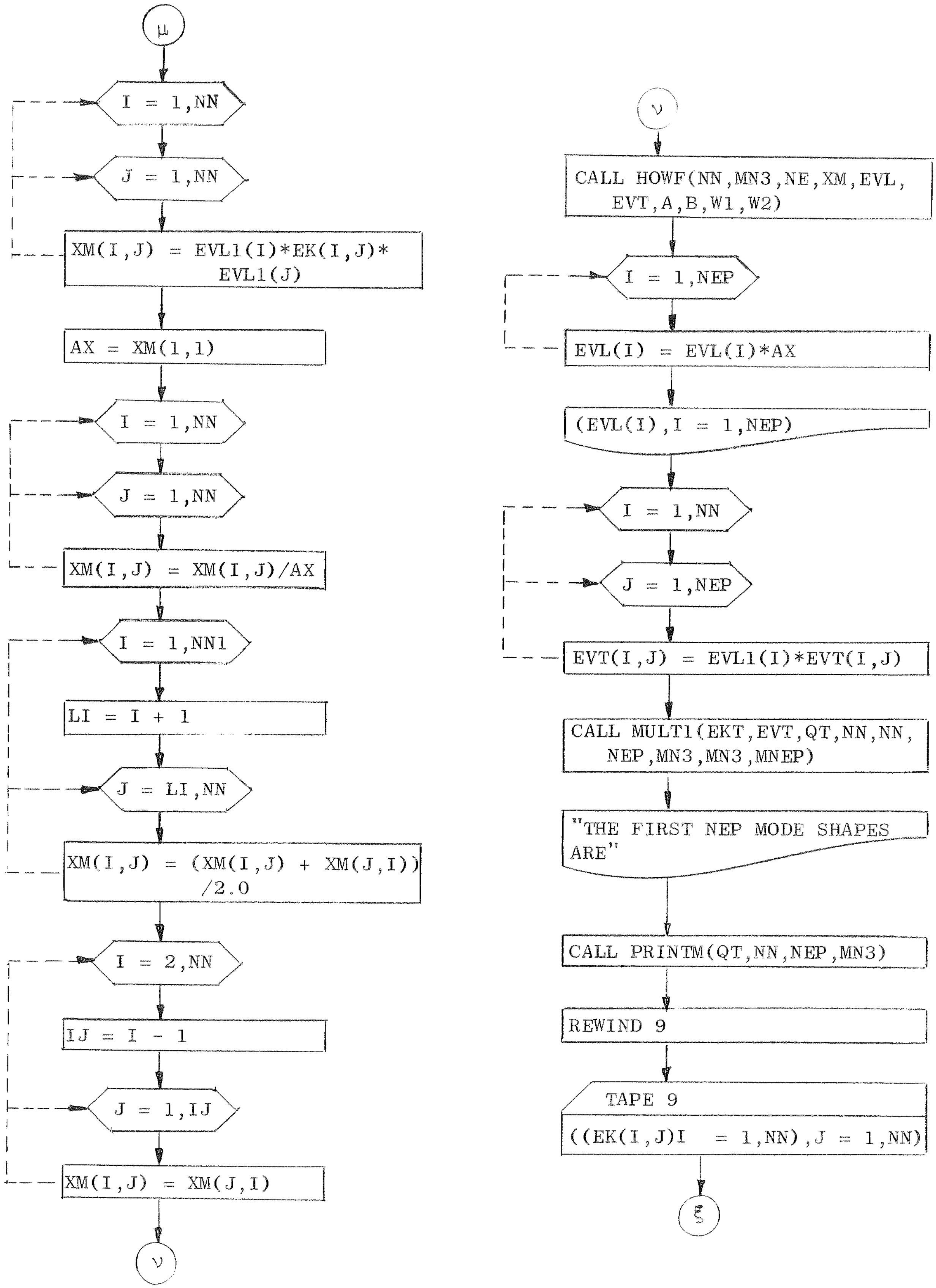
SUBROUTINE SHEKXM(XM,NFRSC,
NAS,NN,MN3)



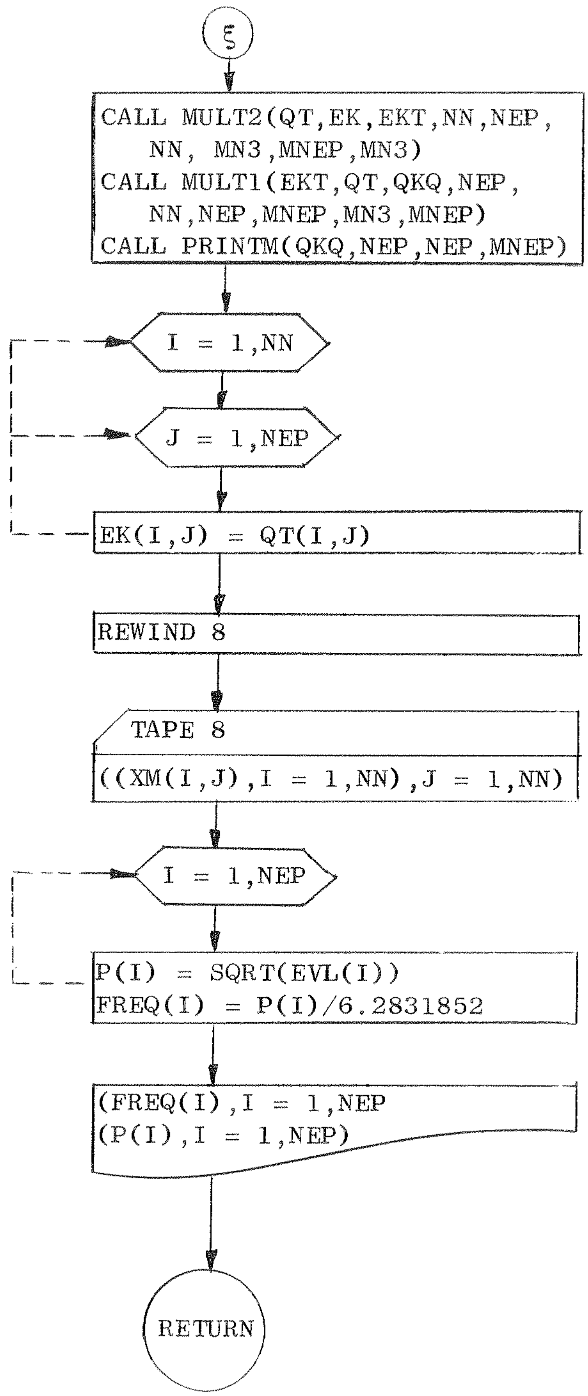
SUBROUTINE L4



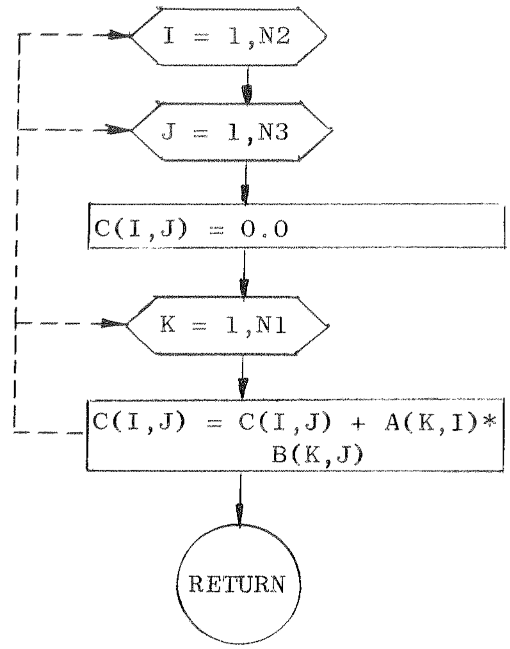
L4(continued)



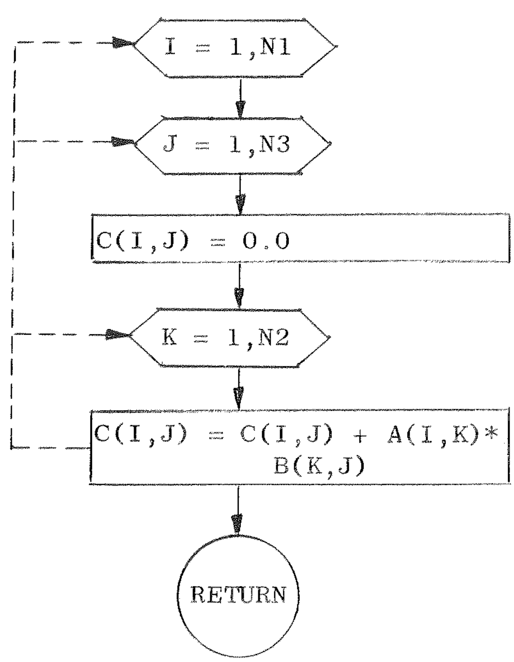
L4(continued)



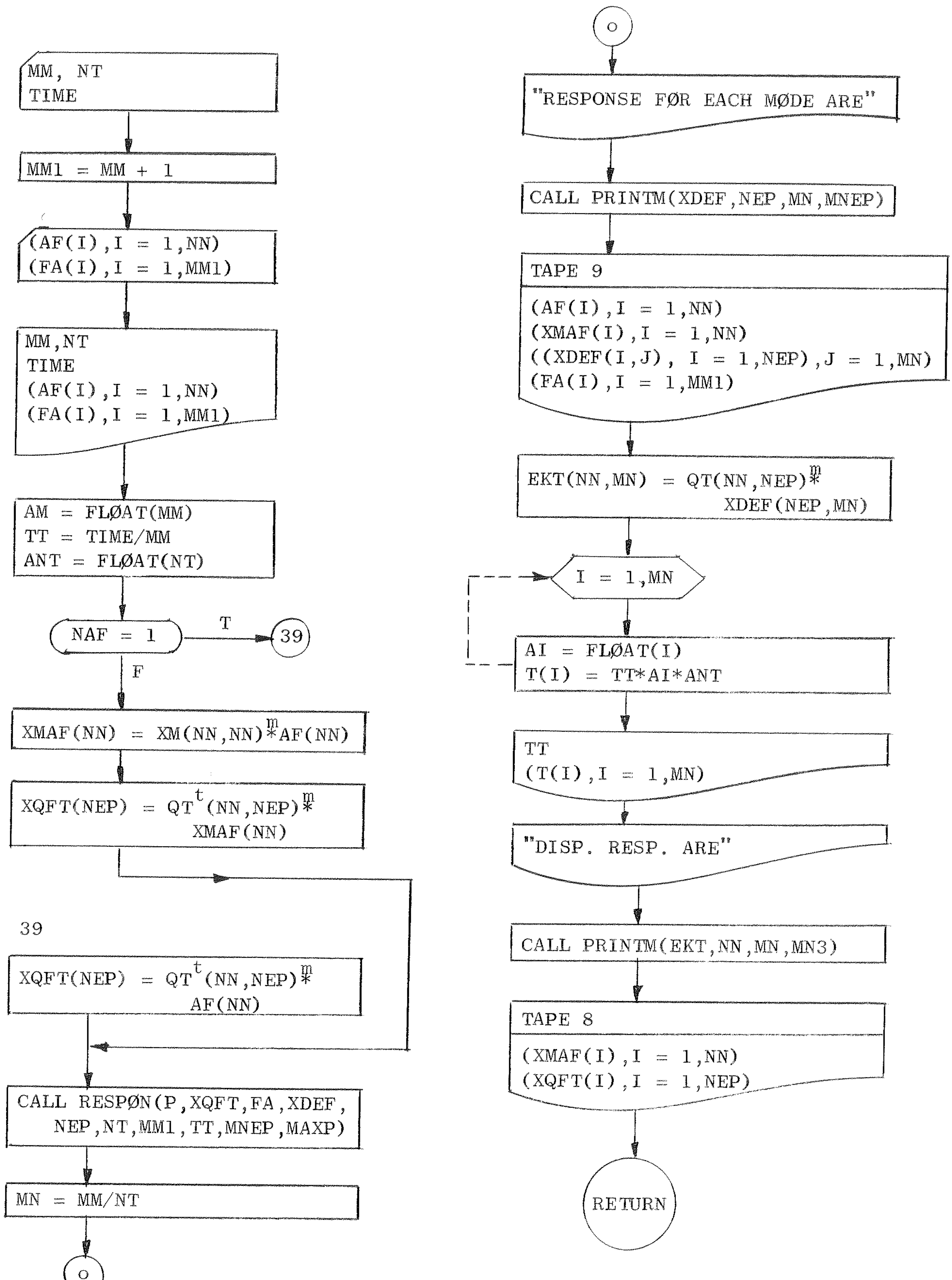
SUBRØUTINE MULT2(A,B,C,N1N2,
N3,MN1,MN2,MN3)



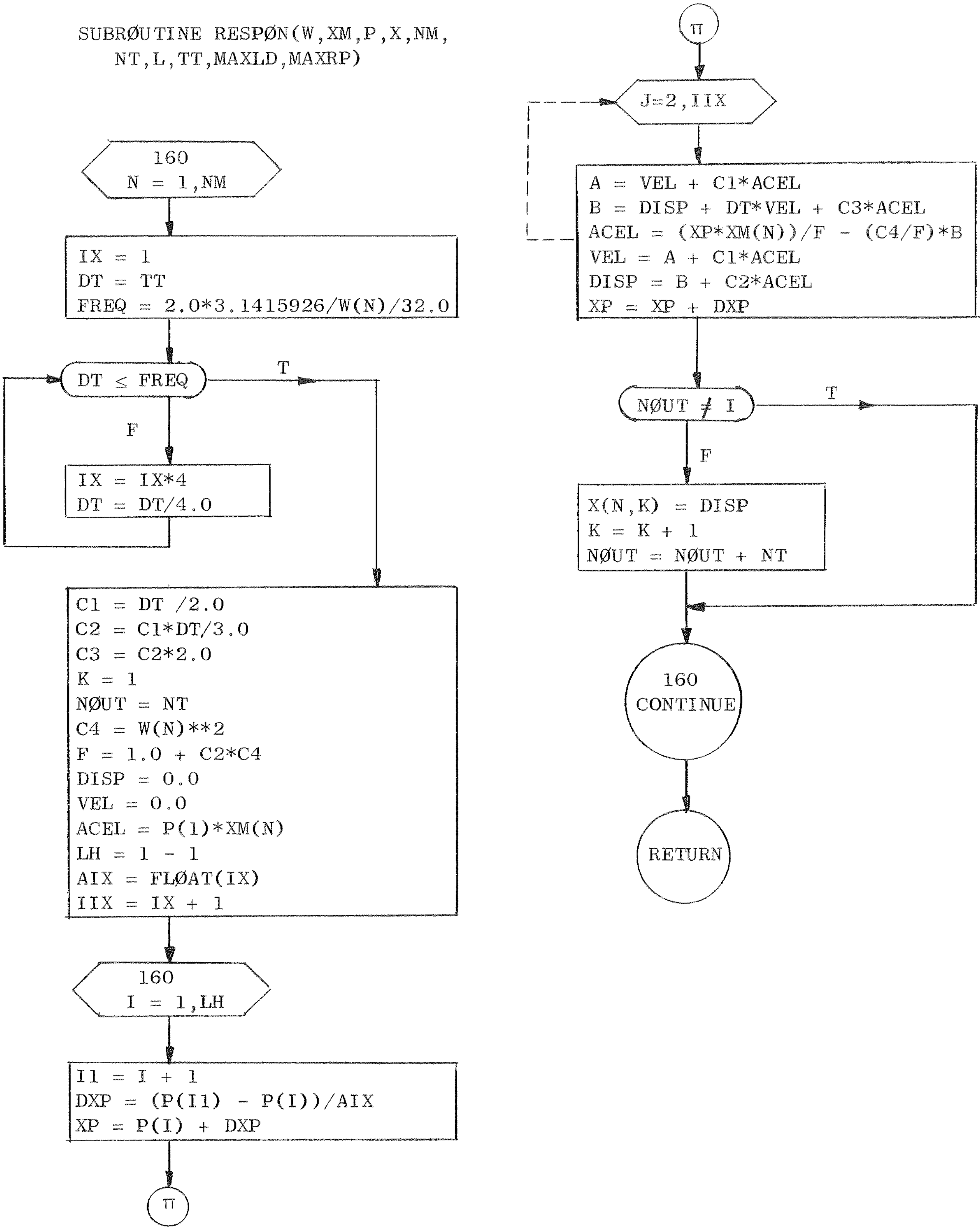
SUBRØUTINE MULT1(A,B,C,N1,N2,
N3,MN1,MN2,MN3)



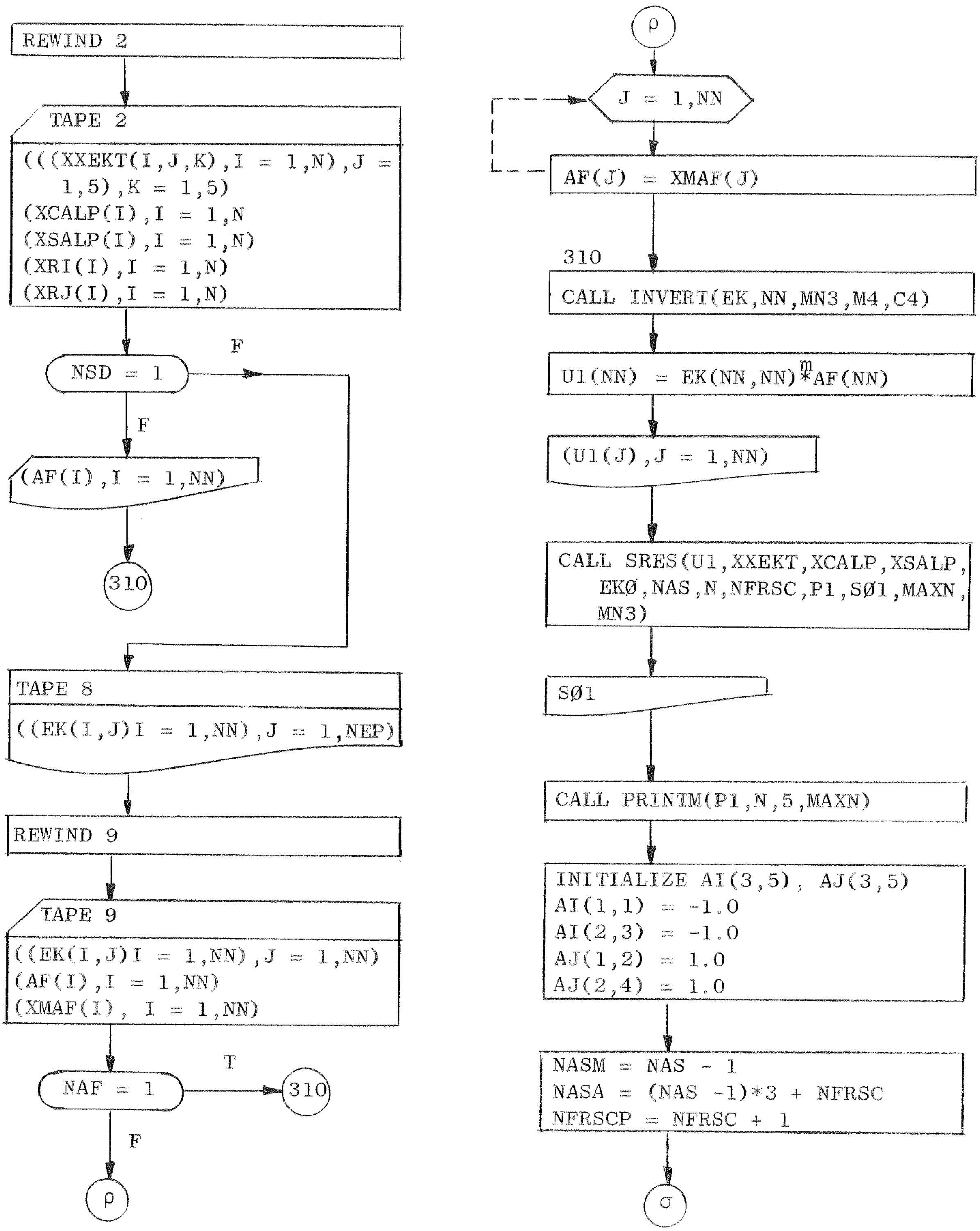
SUBROUTINE L5



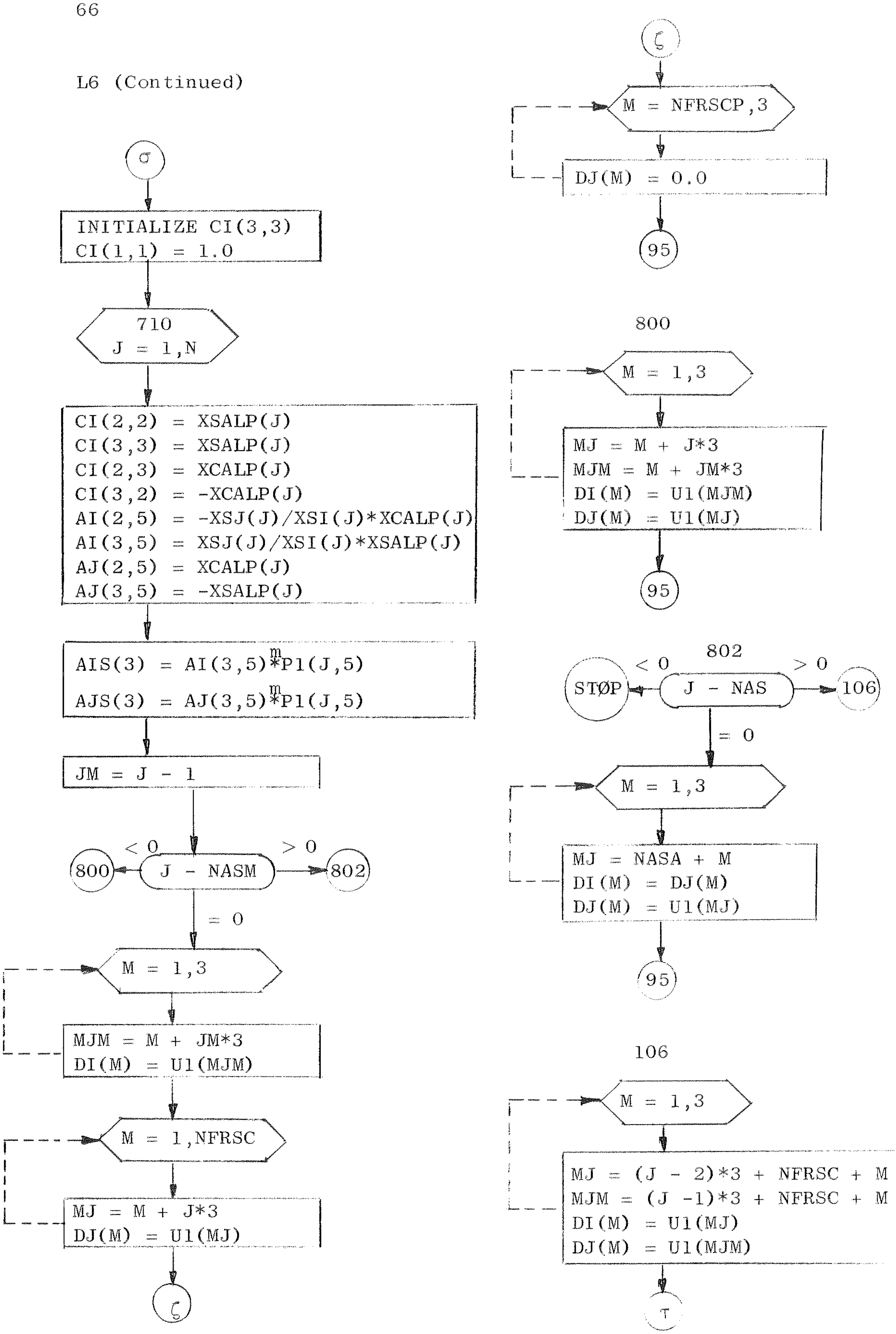
SUBRØUTINE RESPØN(W,XM,P,X,NM,
NT,L,TT,MAXLD,MAXRP)



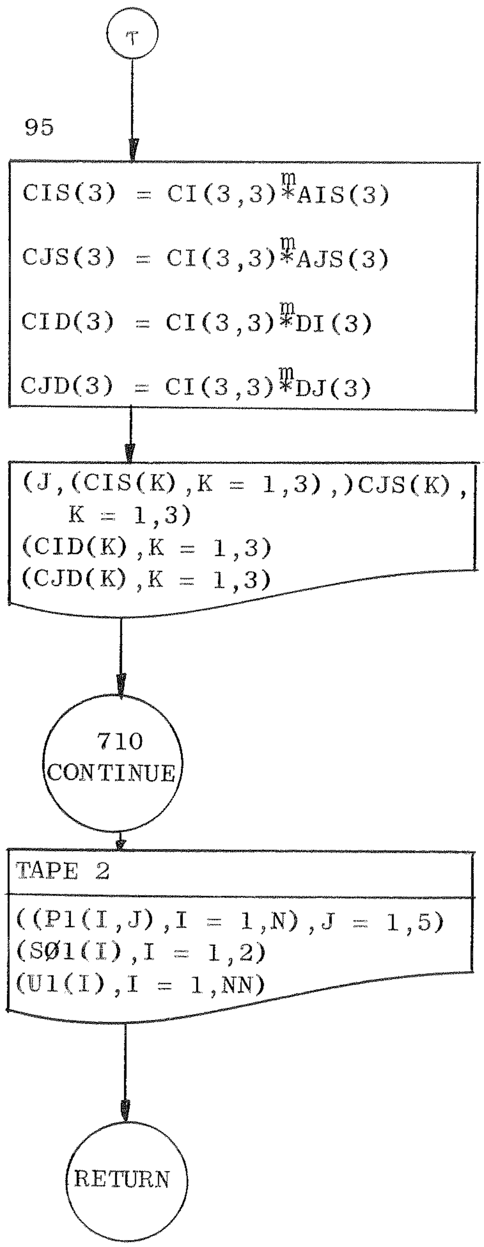
SUBROUTINE L6



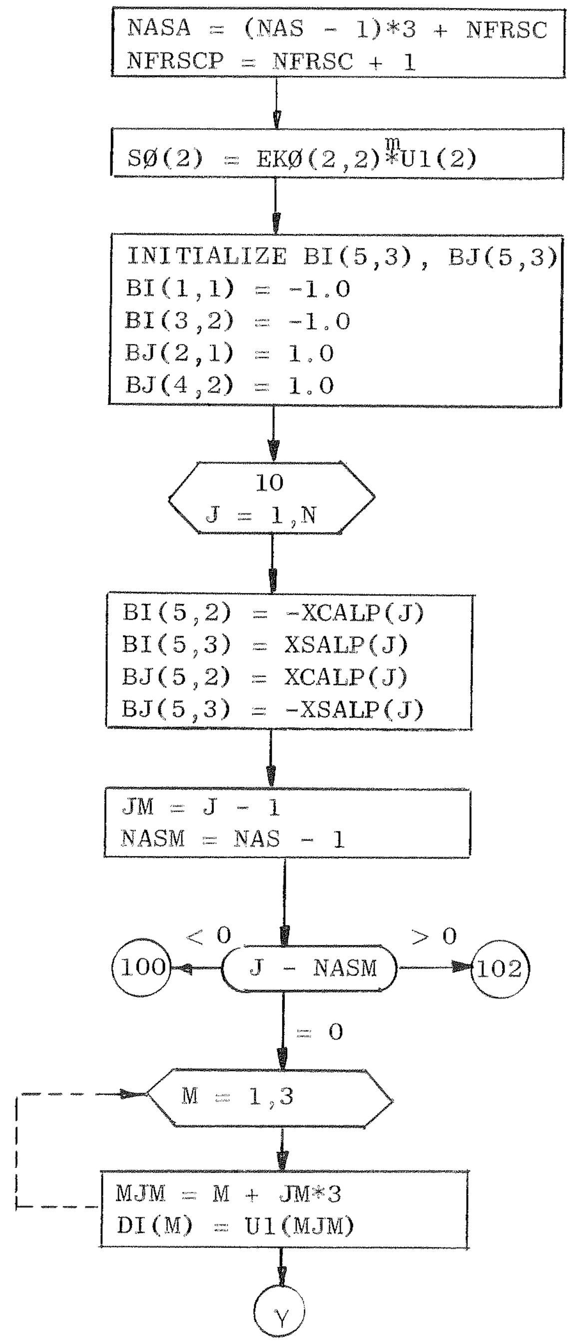
L6 (Continued)



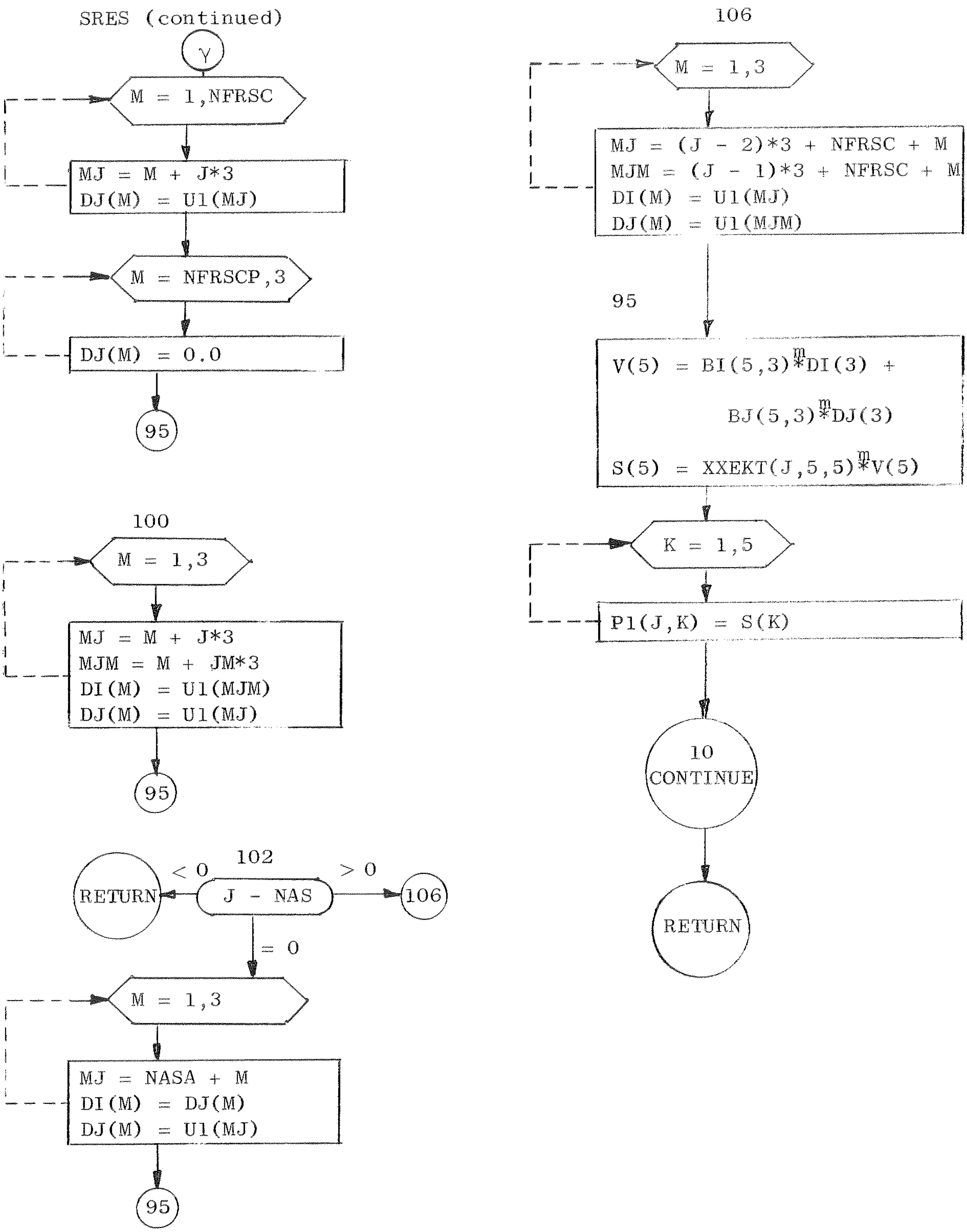
L6 (continued)



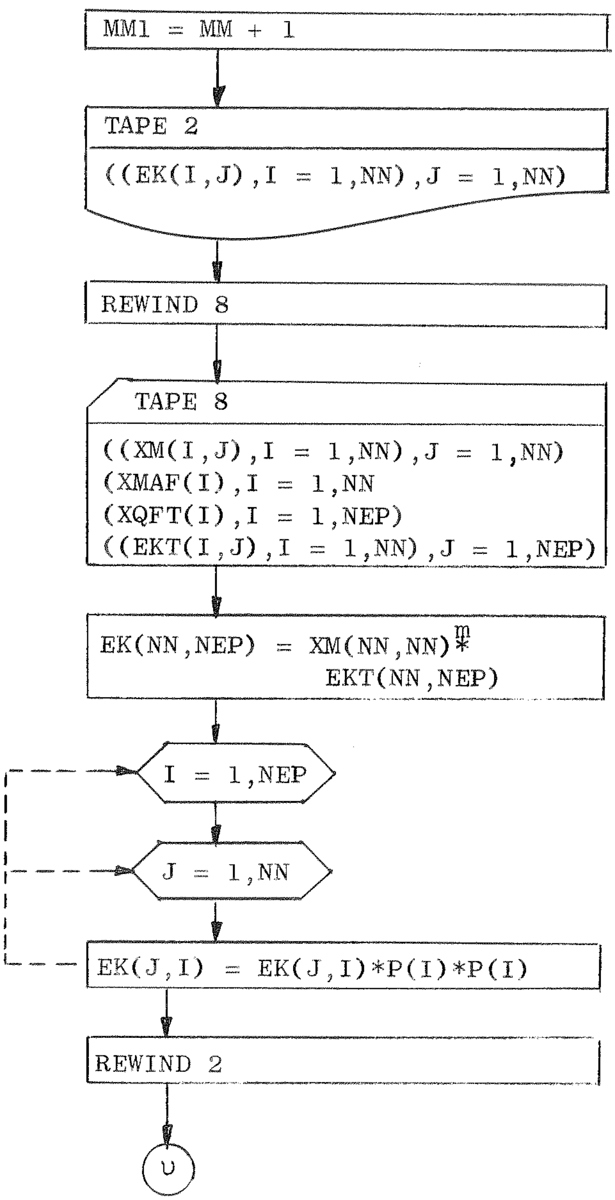
SUBRØUTINE SRES(U1,XXEKT, XCALP,XSALP,EKØ,NAS,N, NFRSC,P1,SØ,MAXN,MN3)

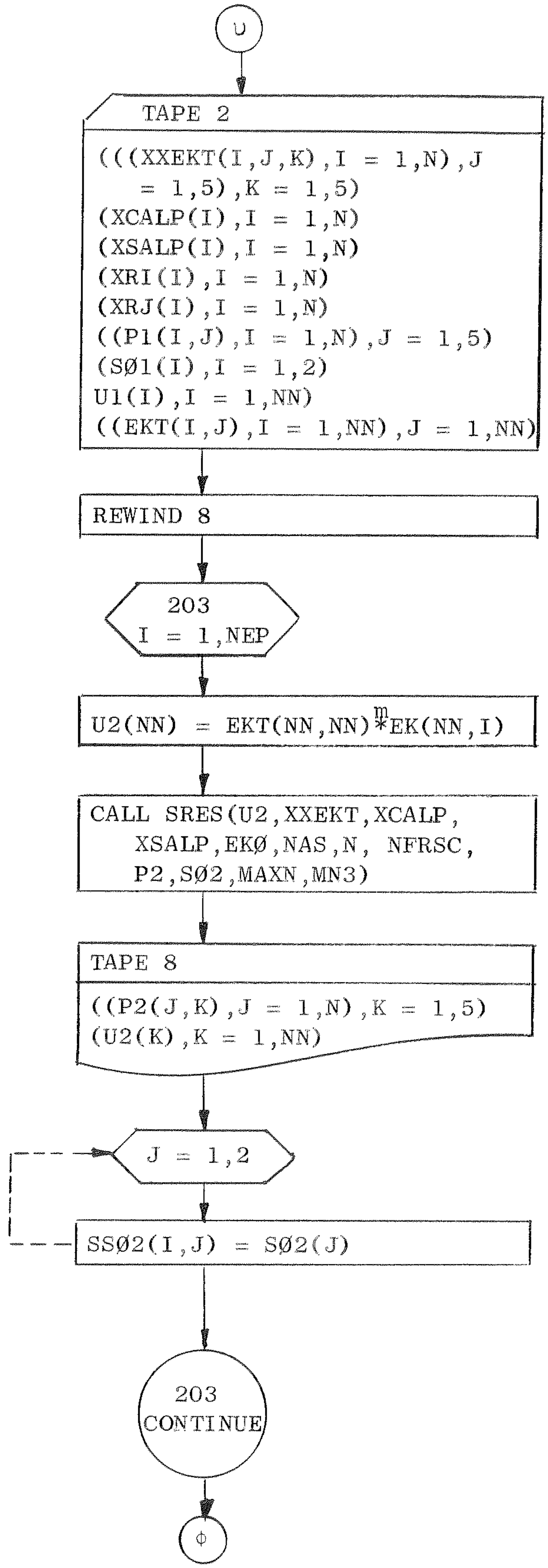


SRES (continued)

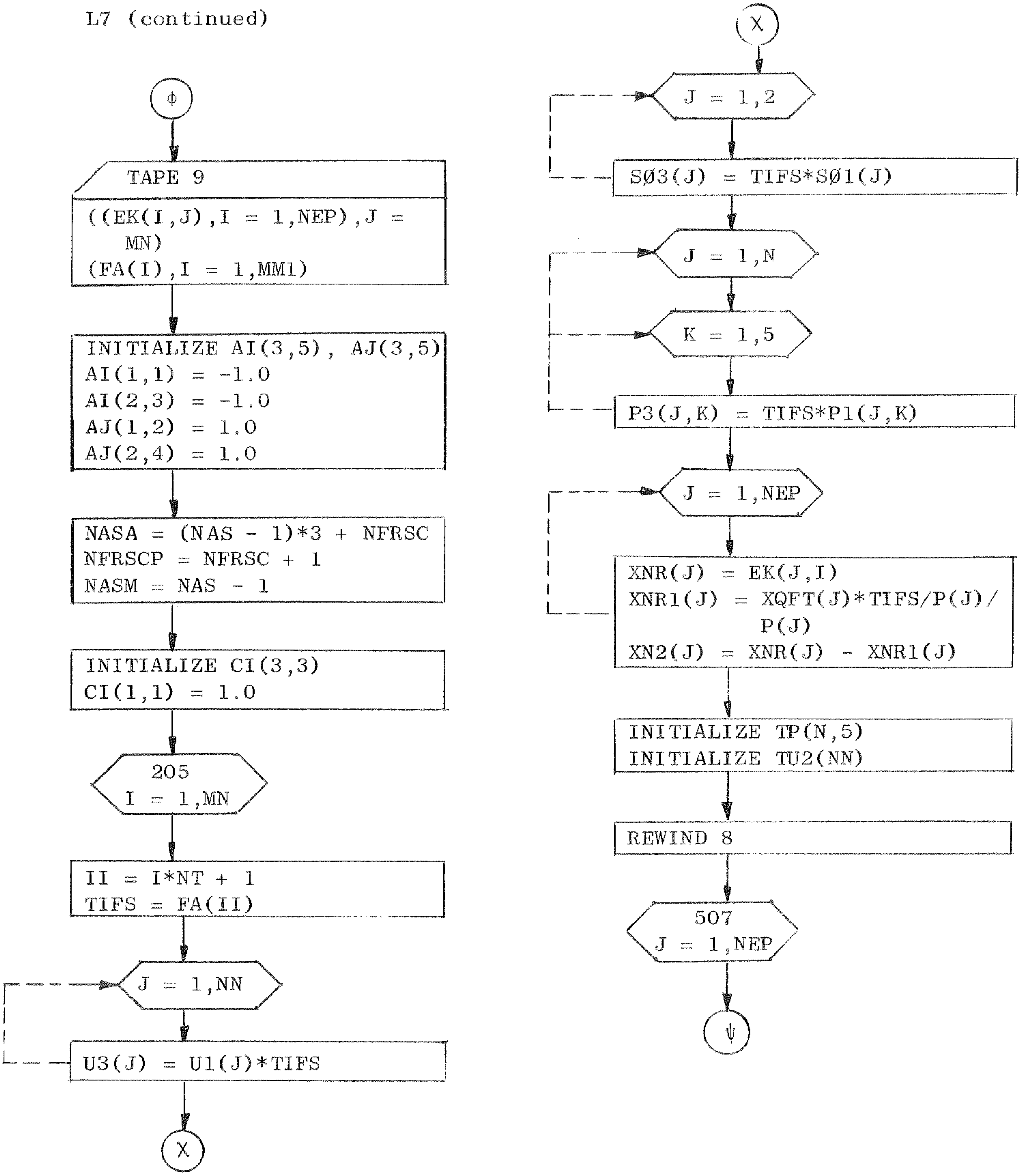


SUBRØUTINE L7

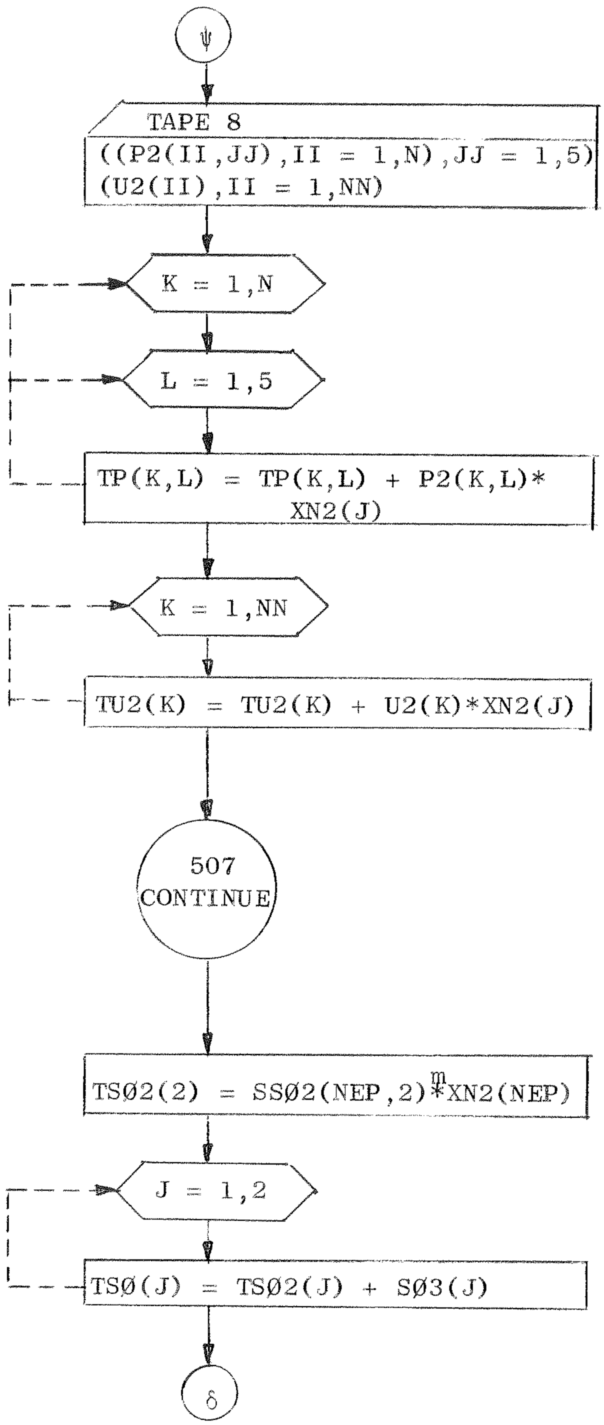


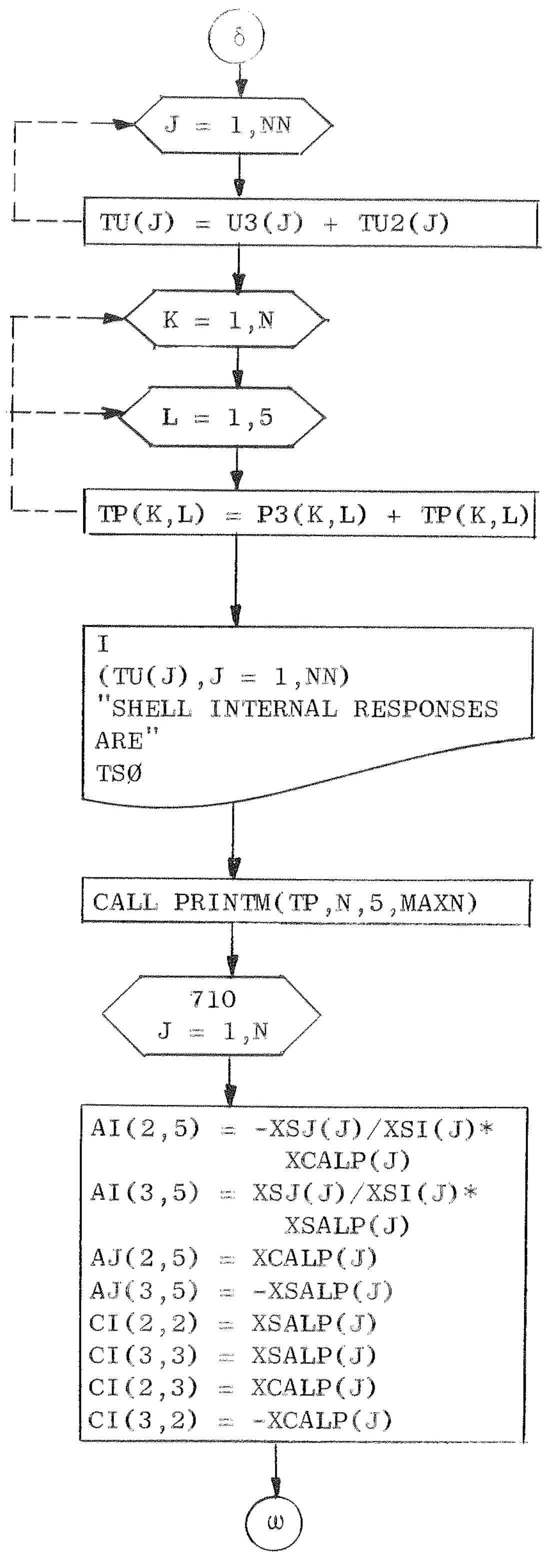


L7 (continued)

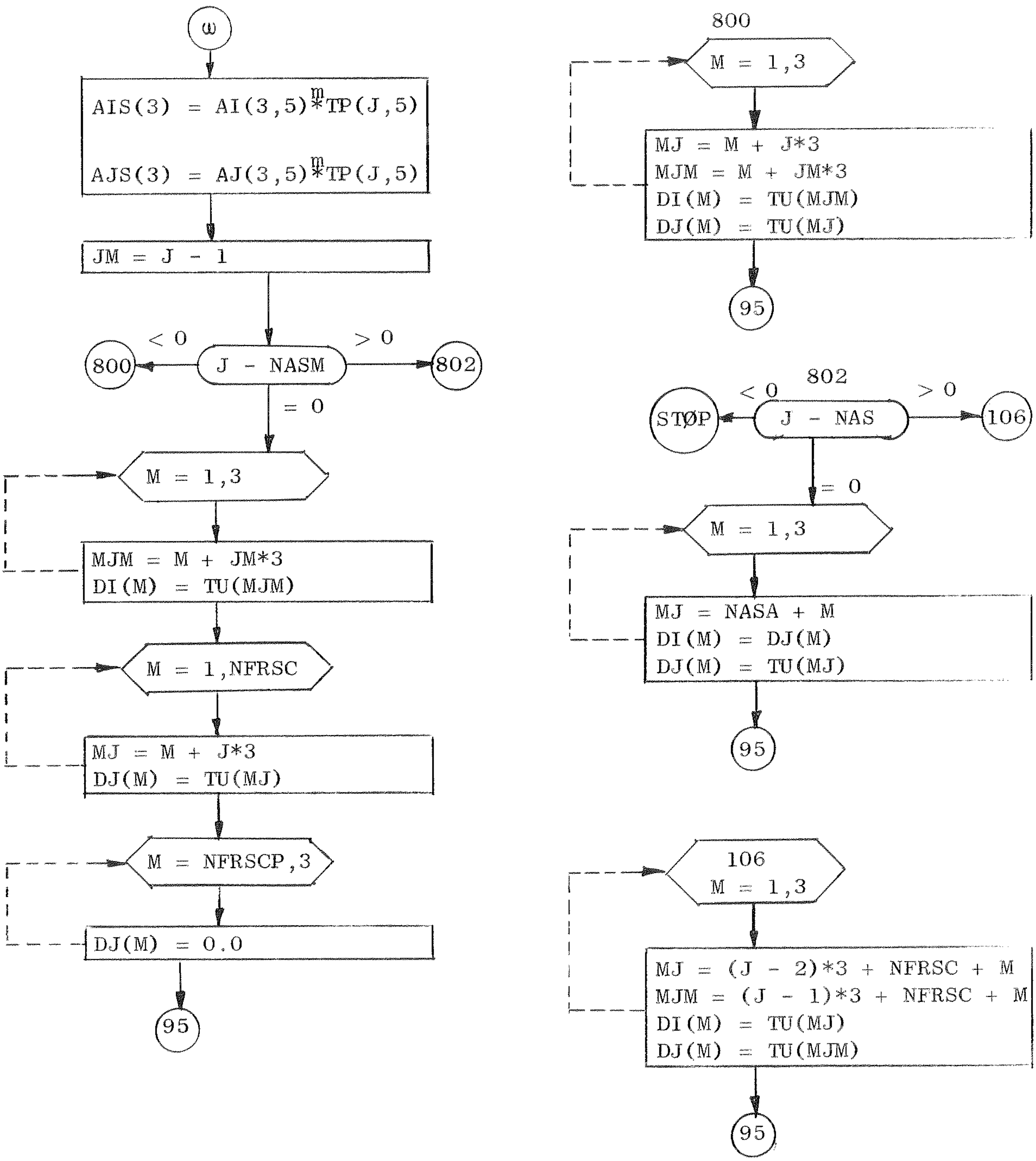


L7 (continued)

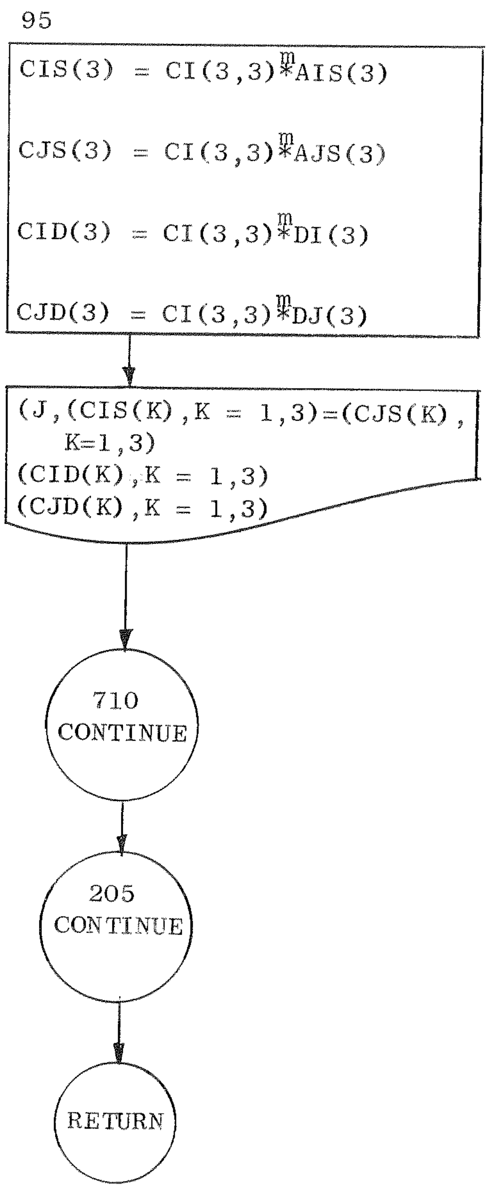




L7 (continued)



L7 (continued)



F. FORTRAN Listing

```
C       MAIN
        COMMON
       1N,NS,NSO,NSD,NFRSC,NAS,NN,NEP,NAF,MAXN,MN3,MN1,MNEP,MAXP,MN,MM,NT,
       2HD,BETA,R,RR,TO,ALPHO,POISO,EO,XWO,GA,TIME,
       3XM(78,78),EK(78,78),P(78),                        XALPH(25),XPOIS(25),
       4XE(25),HH(26),XSI(25),XSJ(25),XW(25),EKO(2,2),XA(25),XN(26),YN(26)
    10 FORMAT (6I4)
    20 FORMAT (4E18.8)
    30 FORMAT(9F8.4)
    40 FORMAT (3E18.8)
    50 FORMAT (5E14.7)
    60 FORMAT (E18.8)
    70 FORMAT (2I4)
    90 FORMAT (5X,23HN,NS,NSO,NSD,NFRSC,NAS=6I4)
   100 FORMAT(5X,6HXALPH=6E16.8/(11X,6E16.8))
   110 FORMAT(5X,6HXPOIS=6E16.8/(11X,6E16.8))
   120 FORMAT(5X,3HXE=6E16.8/(8X,6E16.8))
   130 FORMAT(5X,3HHH=12F8.4/(8X,12F8.4))
   140 FORMAT(5X,10HHD,BETA,R=3E20.8)
   150 FORMAT(5X,3HXA=6E18.8/(8X,6E18.8))
   160 FORMAT(8HXSI,XSJ=6E18.8/(8X,6E18.8))
   170 FORMAT (5X,21HRR,TO,ALPHO,POISO,EO=/5X,5E20.8)
   180 FORMAT (5X,4HXWO=E20.8)
   190 FORMAT(5X,3HGA=E20.8)
   200 FORMAT(5X,3HXW=6E18.8/(8X,6E18.8))
   210 FORMAT (5X,8HNEP,NAF=2I4)
   220 FORMAT(5X,3HXN=6E18.8/(8X,6E18.8))
   230 FORMAT(5X,3HYN=6E18.8/(8X,6E18.8))
   240 FORMAT(1H1)
   250 PRINT 240
       MAXN=25
       MN3=78
       MN1=26
       MNEP=78
       MAXP=78
       READ 10, N,NS,NSO,NSD,NFRSC,NAS
       N1=N+1
       NN=N*3+NFRSC
       READ  30, (XPOIS(I),I=1,N)
       READ  20, (XE(I),I=1,N)
       READ  30, (HH(I),I=1,N1)
       PRINT 90, N,NS,NSO,NSD,NFRSC,NAS
       PRINT 110, (XPOIS(I),I=1,N)
       PRINT 120, (XE(I),I=1,N)
       PRINT 130, (HH(I),I=1,N1)
       INS=NS*NS
       IF (INS .EQ. 1) GO TO 400
       IF (NS .EQ. 0) GO TO 300
       READ 20, (XN(I),I=1,N1)
       READ 20, (YN(I),I=1,N1)
       PRINT 220, (XN(I),I=1,N1)
       PRINT 230, (YN(I),I=1,N1)
       GO TO 310
   300 READ  20, (XSI(I),XSJ(I),I=1,N)
       PRINT 160, (XSI(I),XSJ(I),I=1,N)
       READ  30, (XALPH(I),I=1,N)
       PRINT 100, (XALPH(I),I=1,N)
```

```
      GO TO 310
  400 READ  40, HD,BETA,R
      READ  30,(XA(I),I=1,N)
      PRINT 140, HD,BETA,R
      PRINT 150, (XA(I),I=1,N)
      READ  30, (XALPH(I),I=1,N)
      PRINT 100, (XALPH(I),I=1,N)
  310 IF (NSO .EQ. 0) GO TO 320
      READ  50, RR,TO,ALPHO,POISO,EO
      PRINT 170,RR,TO,ALPHO,POISO,EO
  320 IF (NSD .EQ. 1) GO TO 340
      IF (NSO .EQ. 0) GO TO 330
      READ 60, XWO
      PRINT 180, XWO
  330 READ  60, GA
      READ  20, (XW(I),I=1,N)
      READ 70, NEP,NAF
      PRINT 190, GA
      PRINT 200, (XW(I),I=1,N)
      PRINT 210, NEP,NAF
  340 CALL L1
      CALL L2
      CALL L3
      IF (NSD .EQ. 1) GO TO 350
      CALL L4
      IF (NAF .EQ. 0) GO TO  360
      CALL L5
  350 CALL L6
      IF (NSD .EQ. 1) GO TO 360
      CALL L7
  360 GO TO 250
      END
C
C
C
      SUBROUTINE L1
C     COMPUTATION OF DISC STIFFNESS FOR END CLOSURE
      COMMON
     1N,NS,NSO,NSD,NFRSC,NAS,NN,NEP,NAF,MAXN,MN3,MN1,MNEP,MAXP,MN,MM,NT,
     2HD,BETA,R,RR,TO,ALPHO,POISO,EO,XWO,GA,TIME,
     3XM(78,78),EK(78,78),P(78),                   XALPH(25),XPOIS(25),
     4XE(25),HH(26),XSI(25),XSJ(25),XW(25),EKO(2,2),XA(25),XN(26),YN(26)
      DIMENSION M6(2),C6(2)
      DO 100 J=1,2
      DO 100 K=1,2
  100 EKO(J,K)=0.0
      IF (NSO .EQ. 0) GO TO 991
      RO=RR*SIN(ALPHO/57.29578)
      POIP=1.0+POISO
      POIM=1.0-POISO
      DDO=EO*TO**3/(12.0*POIP*POIM)
      EKO(1,1)=RO/DDO/POIP
      EKO(2,2)=POIM*RO/(EO*TO)
      CALL  INVERT(EKO,2,2,M6,C6)
  991 PRINT 110, ((EKO(I,J),I=1,2),J=1,2)
```

```
  110 FORMAT (5X,4HEKO=4E20.8)
      RETURN
      END
C
C
C
      SUBROUTINE L2
C     COMPUTATION OF CONICAL ELEMENT STIFFNESSES
      DIMENSION XCALP(25),XSALP(25),Y(2),XEK(5,5),XBER(16),XXEKT(25,5,5)
      COMMON
     1N,NS,NSO,NSD,NFRSC,NAS,NN,NEP,NAF,MAXN,MN3,MN1,MNEP,MAXP,MN,MM,NT,
     2HO,BETA,R,RR,TO,ALPHO,POISO,EO,XWO,GA,TIME,
     3XM(78,78),EK(78,78),P(78),                   XALPH(25),XPOIS(25),
     4XE(25),HH(26),XSI(25),XSJ(25),XW(25),EKO(2,2),XA(25),XN(26),YN(26)
      YS=NS
      INS=NS*NS
      IF (INS .NE. 1) GO TO 902
      IF (NS .EQ. 0) GO TO 902
      DY=HO-R*YS*COS(BETA/57.29578)
      PRINT 904, DY
  904 FORMAT (5X,3HDY=E20.8)
  902 DO 988 II=1,N
      III=II+1
      POIS=XPOIS(II)
      E=XE(II)
      TT=0.5*(HH(II)+HH(III))
      RTP=SQRT(3.*(1.-POIS**2))
  908 YAI=FRT2*SQRT(RA*2.0/TT)
      GO TO 6
    4 PRINT 7
    7 FORMAT (5X,29HTHIS IS A CYLINDRICAL ELEMENT)
      STOP
    5 IF (INS .NE. 1) GO TO 909
      SI=-SS+X
      SJ=-SS-X
      XSI(II)=SI
      XSJ(II)=SJ
  909 YAI=FRT2*SQRT(-RA*2.0/TT)
      FRT2=2.*SQRT(RTP)
      IF (NS-2) 410,400,410
  400 YY=YN(II)-YN(III)
      XX=XN(III)-XN(II)
      XXYY=SQRT(YY*YY+XX*XX)
      RA=YY/XX
      CALP=XX/XXYY
      SALP=YY/XXYY
      ATALP=ABS(RA)
      IF (CALP) 10,20,20
   10 ALPHA=3.1415926-ATAN(ATALP)
      GO TO 30
   20 ALPHA=ATAN(ATALP)
   30 ALPH=ALPHA*57.29578
      XALPH(II)=ALPH
      XSI(II)=ABS(XN(II)/CALP)
      XSJ(II)=ABS(XN(III)/CALP)
      GO TO 420
```

```
410 ALPH=XALPH(II)
    ALPHA=ALPH/57.29578
    CALP=COS(ALPHA)
    SALP=SIN(ALPHA)
    RA=SALP/CALP
420 XCALP(II)=CALP
    XSALP(II)=SALP
    IF (INS-1) 905,901,905
901 X=R*SIN(XA(II)/2./57.29578)
    RX=R*COS(XA(II)/2./57.29578)
    SS=(DY/SALP+RX*YS)*RA
    GO TO 907
905 SI=XSI(II)
    SJ=XSJ(II)
907 IF (ALPH-90.0) 3,4,5
  3 IF (INS .NE. 1) GO TO 908
    SI=SS-X
    SJ=SS+X
    XSI(II)=SI
    XSJ(II)=SJ
  6 Y(1)=YAI*SQRT(SI)
    Y(2)=SQRT(SJ)*YAI
    CALL THO(Y,2,XBER)
    CALL FLEKCO(XBER,Y,E,POIS,SI,SJ,TT,ALPH,CALP,SALP,XEK)
    DO 300 J=1,5
    DO 300 K=1,5
300 XXEKT(II,J,K)=XEK(J,K)
988 CONTINUE
    REWIND 2
    WRITE (2) (((XXEKT(I,J,K),I=1,N),J=1,5),K=1,5)
    WRITE (2) (XCALP(I),I=1,N)
    WRITE (2) (XSALP(I),I=1,N)
    RETURN
    END
C
C
    SUBROUTINE THO (Y,MN,XBER)
C   CALCULATION OF THOMPSON FUNCTIONS FOR TWO ARGUEMENTS
    DIMENSION BER(8),MC(2,8),Y(2),MA(8),DER(2,8),XBER(16)
    PI=3.1415926
    DELT=3.1415926/8.
    ROOT=SQRT(2.0)
    MN8=MN*8
    IF (Y(MN)-27.) 80,80,62
 62 DO 201 JOB=1,MN
    YD=Y(JOB)/ROOT
    CCA=COS (SU (YD-DELT))
    SCA=SIN (SU (YD-DELT))
    CCB=COS (SU (YD+DELT))
    SCB=SIN (SU (YD+DELT))
    BE=Y(JOB)/ROOT
    CALL XXP(E,BE,MB)
    C=0.0
    VYI=1.0
    VNI=1.0
    XV=1.0
```

```
      XNI=1.0
      XW=-1.0
      XNW=-1.0
      WYI=0.0
      WNI=0.0
      XS=-1.0
      XNS=-1.0
      SYI=1.0
      SNI=1.0
      XT=1.0
      XNT=1.0
      TYI=0.0
      TNI=0.0
   63 C=C+1.0
      CS=COS (C*PI/4.)
      SS=SIN (C*PI/4.)
      XV=XV*(-1.)*(-(2.*C-1.)**2)*CS/(C*8.*Y(JOB))
      XNI=XNI*(-1.)*(-(2.*C-1.)**2)*CS/(C*8.*(-Y(JOB)))
      VYI=VYI+XV
      VNI=VNI+XNI
      XV=XV/CS
      XNI=XNI/CS
      XW=XW*(-1.)*(-(2.*C-1.)**2)*SS/(C*8.*Y(JOB))
      XNW=XNW*(-1.)*(-(2.*C-1.)**2)*SS/(C*8.*(-Y(JOB)))
      WYI=WYI+XW
      WNI=WNI+XNW
      XW=XW/SS
      XNW=XNW/SS
      XS=XS*(-1.)*(-(2.*C-1.)*(2.*C+1.))*CS/(C*8.*Y(JOB))
      XNS=XNS*(-1.)*(-(2.*C-1.)*(2.*C+1.))*CS/(C*8.*(-Y(JOB)))
      SYI=SYI+XS
      SNI=SNI+XNS
      XS=XS*(2.*C-1.)/(CS*(2.*C+1.))
      XNS=XNS*(2.*C-1.)/(CS*(2.*C+1.))
      XT=XT*(-1.)*(-(2.*C-1.)*(2.*C+1.))*SS/(C*8.*Y(JOB))
      XNT=XNT*(-1.)*(-(2.*C-1.)*(2.*C+1.))*SS/(C*8.*(-Y(JOB)))
      TYI=TYI+XT
      TNI=TNI+XNT
      XT=XT*(2.*C-1.)/(SS*(2.*C+1.))
      XNT=XNT*(2.*C-1.)/(SS*(2.*C+1.))
      IF (Y(JOB)-92.0) 71,71,72
   71 IF (C-15.0) 63,121,121
   72 IF (C-7.0) 63,121,121
  121 BYI=SQRT (PI/(2.*Y(JOB)))/E
      BER(5)=BYI*VNI*CCB+BYI*WNI*SCB
      BER(6)=BYI*(WNI*CCB-VNI*SCB)
      BER(7)=(-BYI)*(SNI*CCA+TNI*SCA)
      BER(8)=(-BYI)*(TNI*CCA-SNI*SCA)
      AYI=E/SQRT (2.*PI*Y(JOB))
      BER(1)=AYI*(VYI*CCA-WYI*SCA)
      BER(2)=AYI*(WYI*CCA+VYI*SCA)
      BER(3)=AYI*(SYI*CCB-TYI*SCB)
      BER(4)=AYI*(TYI*CCB+SYI*SCB)
      MBA=-MB
      DO 17 I=1,4
      IJ=I+4
```

```
      CALL XC  (MBA,BER(IJ),MC(JOB,IJ))
   17 CALL XC  (MB,BER(I),MC(JOB,I))
      PRINT 200,(BER(I),MC(JOB,I),I=1,8)
  200 FORMAT (4(F20.8,1HE,I9))
      DO 19 I=1,4
      IJ=I+4
      MA(I)=MC(JOB,I)-MC(1,1)+8
   19 MA(IJ)=MC(JOB,IJ)+MC(1,1)-8
      DO 297 K=1,8
      KJOB=K+(JOB-1)*8
  297 XBER(KJOB)=BER(K)*10.0**MA(K)
  201 CONTINUE
      GO TO 800
   80 DO 88 JOB=1,MN
      IF (Y(JOB)-5.) 31,61,61
   61 YD=Y(JOB)/ROOT
      CCA=COS (SU (YD-DELT))
      SCA=SIN (SU (YD-DELT))
      CCB=COS (SU (YD+DELT))
      SCB=SIN (SU (YD+DELT))
      BE=Y(JOB)/ROOT
      E=EXP (BE)
      C=0.0
      VYI=1.0
      VNI=1.0
      XV=1.0
      XNI=1.0
      XW=-1.0
      XNW=-1.0
      WYI=0.0
      WNI=0.0
      XS=-1.0
      XNS=-1.0
      SYI=1.0
      SNI=1.0
      XT=1.0
      XNT=1.0
      TYI=0.0
      TNI=0.0
      IF (Y(JOB)-15.0) 81,133,133
  133 C=C+1.0
      CS=COS (C*PI/4.)
      SS=SIN (C*PI/4.)
      XV=XV*(-1.)*(-(2.*C-1.)**2)*CS/(C*8.*Y(JOB))
      XNI=XNI*(-1.)*(-(2.*C-1.)**2)*CS/(C*8.*(-Y(JOB)))
      VYI=VYI+XV
      VNI=VNI+XNI
      XV=XV/CS
      XNI=XNI/CS
      XW=XW*(-1.)*(-(2.*C-1.)**2)*SS/(C*8.*Y(JOB))
      XNW=XNW*(-1.)*(-(2.*C-1.)**2)*SS/(C*8.*(-Y(JOB)))
      WYI=WYI+XW
      WNI=WNI+XNW
      XW=XW/SS
      XNW=XNW/SS
      XS=XS*(-1.)*(-(2.*C-1.)*(2.*C+1.))*CS/(C*8.*Y(JOB))
```

```
      XNS=XNS*(-1.)*(-(2.*C-1.)*(2.*C+1.))*CS/(C*8.*(-Y(JOB)))
      SYI=SYI+XS
      SNI=SNI+XNS
      XS=XS*(2.*C-1.)/(CS*(2.*C+1.))
      XNS=XNS*(2.*C-1.)/(CS*(2.*C+1.))
      XT=XT*(-1.)*(-(2.*C-1.)*(2.*C+1.))*SS/(C*8.*Y(JOB))
      XNT=XNT*(-1.)*(-(2.*C-1.)*(2.*C+1.))*SS/(C*8.*(-Y(JOB)))
      TYI=TYI+XT
      TNI=TNI+XNT
      XT=XT*(2.*C-1.)/(SS*(2.*C+1.))
      XNT=XNT*(2.*C-1.)/(SS*(2.*C+1.))
      IF (Y(JOB)-20.0) 131,131,134
  131 IF (C-(2.*Y(JOB))) 133,135,135
  134 IF (C-30.0) 133,135,135
  135 BYI=SQRT (PI/(2.*Y(JOB)))/E
      BER(5)=BYI*VNI*CCB+BYI*WNI*SCB
      BER(6)=BYI*(WNI*CCB-VNI*SCB)
      BER(7)=(-BYI)*(SNI*CCA+TNI*SCA)
      BER(8)=(-BYI)*(TNI*CCA-SNI*SCA)
      AYI=E/SQRT (2.*PI*Y(JOB))
      BER(1)=AYI*(VYI*CCA-WYI*SCA)-BER(6)/PI
      BER(2)=AYI*(WYI*CCA+VYI*SCA)+BER(5)/PI
      BER(3)=AYI*(SYI*CCB-TYI*SCB)-BER(8)/PI
      BER(4)=AYI*(TYI*CCB+SYI*SCB)+BER(7)/PI
      PRINT 57, (BER(I),I=1,8)
      DO 103 K=1,8
      KJOB=K+(JOB-1)*8
  103 XBER(KJOB)=BER(K)
      GO TO 88
   81 XX=Y(JOB)**4
      XA=1.
      EN=1.
      DER(JOB,1)=XA
   82 XA=-XA*XX/(16.*((2.*EN)*(2.*EN-1.))**2)
      DER(JOB,1)=DER(JOB,1)+XA
      EN=EN+1.
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,1)))) 83,83,82
   83 XX=Y(JOB)**4
      XA=(Y(JOB)**2)/4.
      EN=1.
      DER(JOB,2)=XA
   84 XA=-XA*XX/(16.*((2.*EN+1.)*(2.*EN))**2)
      DER(JOB,2)=DER(JOB,2)+XA
      EN=EN+1.
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,2)))) 86,86,84
   86 XZ=Y(JOB)**4
      XA=-Y(JOB)**3/16.
      EN=2.
      DER(JOB,3)=XA
   87 XA=-XA*XZ/(16.*(2.*EN-2.)*(2.*EN)*((2.*EN-1.)**2))
      DER(JOB,3)=DER(JOB,3)+XA
      EN=EN+1.
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,3)))) 91,91,87
   91 XZ=Y(JOB)**4
      XA=Y(JOB)/2.
      EN=2.
```

```
      DER(JOB,4)=XA
   92 XA=-XA*XZ/(16.*(2.*EN-1.)*(2.*EN-3.)*((2.*EN-2.)**2))
      DER(JOB,4)=DER(JOB,4)+XA
      EN=EN+1.
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,4)))) 85,85,92
   85 C=C+1.0
      CS=COS (C*PI/4.)
      SS=SIN (C*PI/4.)
      XNI=XNI*(-1.)*(-(2.*C-1.)**2)*CS/(C*8.*(-Y(JOB)))
      VNI=VNI+XNI
      XNI=XNI/CS
      XNW=XNW*(-1.)*(-(2.*C-1.)**2)*SS/(C*8.*(-Y(JOB)))
      WNI=WNI+XNW
      XNW=XNW/SS
      XNS=XNS*(-1.)*(-(2.*C-1.)*(2.*C+1.))*CS/(C*8.*(-Y(JOB)))
      SNI=SNI+XNS
      XNS=XNS*(2.*C-1.)/(CS*(2.*C+1.))
      XNT=XNT*(-1.)*(-(2.*C-1.)*(2.*C+1.))*SS/(C*8.*(-Y(JOB)))
      TNI=TNI+XNT
      XNT=XNT*(2.*C-1.)/(SS*(2.*C+1.))
      IF (C-(2.*Y(JOB))) 85,99,99
   99 BYI=SQRT (PI/(2.*Y(JOB)))/E
      DER(JOB,5)=BYI*VNI*CCB+BYI*WNI*SCB
      DER(JOB,6)=BYI*(WNI*CCB-VNI*SCB)
      DER(JOB,7)=(-BYI)*(SNI*CCA+TNI*SCA)
      DER(JOB,8)=(-BYI)*(TNI*CCA-SNI*SCA)
      PRINT 57,(DER(JOB,I),I=1,8)
      GO TO 888
   31 XX=Y(JOB)**4
      XA=1.
      EN=1.
      DER(JOB,1)=XA
   41 XA=-XA*XX/(16.*((2.*EN)*(2.*EN-1.))**2)
      DER(JOB,1)=DER(JOB,1)+XA
      EN=EN+1.
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,1)))) 42,42,41
   42 XX=Y(JOB)**4
      XA=(Y(JOB)**2)/4.
      EN=1.
      DER(JOB,2)=XA
   43 XA=-XA*XX/(16.*((2.*EN+1.)*(2.*EN))**2)
      DER(JOB,2)=DER(JOB,2)+XA
      EN=EN+1.
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,2)))) 44,44,43
   44 XY=DER(JOB,1)*(ALOG(2./Y(JOB))-0.577215665)+DER(JOB,2)*PI/4.
      EN=1.
      YA=1.5
      XZ=Y(JOB)**4
      XX=-XZ/64.
      XA=YA*XX
      DER(JOB,5)=XY+XA
   45 EN=EN+1.
      YA=YA+(4.*EN-1.)/(4.*(EN**2)-2.*EN)
      XX=-XX*XZ/(16.*((2.*EN)*(2.*EN-1.))**2)
      XA=YA*XX
      DER(JOB,5)=DER(JOB,5)+XA
```

```
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,5)))) 46,46,45
   46 XY=DER(JOB,2)*(ALOG(2./Y(JOB))-0.577215665)-DER(JOB,1)*PI/4.
      EN=1.
      YA=1.
      XZ=Y(JOB)**4
      XX=(Y(JOB)**2)/4.
      XA=YA*XX
      DER(JOB,6)=XY+XA
   47 EN=EN+1.
      YA=YA+(4.*EN-3.)/((2.*EN-1.)*(2.*EN-2.))
      XX=-XX*XZ/(16.*((2.*EN-1.)*(2.*EN-2.))**2)
      XA=YA*XX
      DER(JOB,6)=DER(JOB,6)+XA
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,6)))) 48,48,47
   48 XZ=Y(JOB)**4
      XA=-Y(JOB)**3/16.
      EN=2.
      DER(JOB,3)=XA
   49 XA=-XA*XZ/(16.*(2.*EN-2.)*(2.*EN)*((2.*EN-1.)**2))
      DER(JOB,3)=DER(JOB,3)+XA
      EN=EN+1.
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,3)))) 50,50,49
   50 XZ=Y(JOB)**4
      XA=Y(JOB)/2.
      EN=2.
      DER(JOB,4)=XA
   51 XA=-XA*XZ/(16.*(2.*EN-1.)*(2.*EN-3.)*((2.*EN-2.)**2))
      DER(JOB,4)=DER(JOB,4)+XA
      EN=EN+1.
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,4)))) 52,52,51
  520XY=DER(JOB,3)*(ALOG(2./Y(JOB))-0.577215665)-DER(JOB,1)/Y(JOB)+
     1DER(JOB,4)*PI/4.
      EN=1.
      YA=1.5
      XX=-2.*Y(JOB)**3/32.
      XA=YA*XX
      DER(JOB,7)=XY+XA
   53 EN=EN+1.
      XZ=EN*Y(JOB)**4
      YA=YA+(4.*EN-1.)/(4.*(EN**2)-2.*EN)
      XX=-XX*XZ/(16.*((2.*EN)*(2.*EN-1.))**2)
      XA=YA*XX
      DER(JOB,7)=DER(JOB,7)+XA
      XX=XX/EN
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,7)))) 54,54,53
  540XY=DER(JOB,4)*(ALOG(2./Y(JOB))-0.577215665)-DER(JOB,2)/Y(JOB)-
     1DER(JOB,3)*PI/4.
      EN=1.
      YA=1.
      XX=Y(JOB)/2.
      XA=YA*XX
      DER(JOB,8)=XY+XA
   55 EN=EN+1.
      XZ=(2.*EN-1.)*Y(JOB)**4
      YA=YA+(4.*EN-3.)/((2.*EN-1.)*(2.*EN-2.))
      XX=-XX*XZ/(16.*((2.*EN-1.)*(2.*EN-2.))**2)
```

```
      XA=YA*XX
      DER(JOB,8)=DER(JOB,8)+XA
      XX=XX/(2.*EN-1.)
      IF (ABS (XA)-(0.00000001*ABS (DER(JOB,8)))) 56,56,55
   56 PRINT 57, (DER(JOB,I),I=1,8)
   57 FORMAT (4E30.8)
  888 DO 333 K=1,8
      KJOB=K+(JOB-1)*8
  333 XBER(KJOB)=DER(JOB,K)
   88 CONTINUE
      PRINT 97, (XBER(I),I=1,MN8)
   97 FORMAT(5X,5HTHOM=4E20.8/(10X,4E20.8))
  800 RETURN
      END
C
C
      SUBROUTINE XC (MA,B,M)
C     SCALES SUCH THAT B BECOMES  B*10**M WHERE (0.1 .LE. B .LT. 1.0)
      M=MA
    2 IF (ABS(B)-1.) 3,1,1
    1 M=M+1
      B=0.1*B
      GO TO 2
    3 IF (ABS(B)-0.1) 5,4,4
    5 M=M-1
      B=10.*B
      GO TO 3
    4 RETURN
      END
C
C
      SUBROUTINE XXP(E,BE,MB)
C     SCALES SUCH THAT BE = E*10**MB WHERE (1.0 .LE. E .LT. 10.0)
      B=BE/2.3025851
      MB=B
      E=10.**(B-FLOAT(MB))
      RETURN
      END
C
C
      FUNCTION SU (ARG)
C     MAKES ARGUEMENT SMALL ENOUGH TO USE COS AND SIN FUNCTIONS
    2 IF (ARG-3141.) 4,4,3
    3 ARG=ARG-3141.5926
      GO TO 2
    4 SU=ARG
      RETURN
      END
C
C
      SUBROUTINE FLEKCO(XBER,Y,E,POIS,SI,SJ,TH,ALPH,COSALP,SINALP,F)
C     CALCULATION OF A SINGLE CONICAL ELEMENT STIFFNESS
      DIMENSION F(5,5),DA(7),EA(4),XBER(16),Y(2),M5(5),C5(5)
      BERYI=XBER(1)
      BEIYI=XBER(2)
      DBERYI=XBER(3)
```

```
DBEIYI=XBER(4)
XERYI=XBER(5)
XEIYI=XBER(6)
DXERYI=XBER(7)
DXEIYI=XBER(8)
BERYJ=XBER(9)
BEIYJ=XBER(10)
DBERYJ=XBER(11)
DBEIYJ=XBER(12)
XERYJ=XBER(13)
XEIYJ=XBER(14)
DXERYJ=XBER(15)
DXEIYJ=XBER(16)
YI=Y(1)
YJ=Y(2)
B1YI=BERYI   -2.0*DBEIYI   /YI
B2YI=BEIYI   +2.0*DBERYI   /YI
X1YI=XERYI   -2.0*DXEIYI   /YI
X2YI=XEIYI   +2.0*DXERYI   /YI
B3YI=YI   *DBERYI   -2.0*(1.0-POIS)*B1YI
B4YI=YI   *DBEIYI   -2.0*(1.0-POIS)*B2YI
X3YI=YI   *DXERYI   -2.0*(1.0-POIS)*X1YI
X4YI=YI   *DXEIYI   -2.0*(1.0-POIS)*X2YI
B5YI=-0.5*YI   *DBERYI   +(1.0+POIS)*B1YI
B6YI=-0.5*YI   *DBEIYI   +(1.0+POIS)*B2YI
X5YI=-0.5*YI   *DXERYI   +(1.0+POIS)*X1YI
X6YI=-0.5*YI   *DXEIYI   +(1.0+POIS)*X2YI
B7YI=BERYI   -0.5*YI   *DBERYI
B8YI=BEIYI   -0.5*YI   *DBEIYI
X7YI=XERYI   -0.5*YI   *DXERYI
X8YI=XEIYI   -0.5*YI   *DXEIYI
B9YI=POIS*BERYI   -2.0*(1.0+POIS)*DBEIYI   /YI
B10YI=POIS*BEIYI   +2.0*(1.0+POIS)*DBERYI   /YI
X9YI=POIS*XERYI   -2.0*(1.0+POIS)*DXEIYI   /YI
X10YI=POIS*XEIYI   +2.0*(1.0+POIS)*DXERYI   /YI
B1YJ=BERYJ   -2.0*DBEIYJ   /YJ
B2YJ=BEIYJ   +2.0*DBERYJ   /YJ
X1YJ=XERYJ   -2.0*DXEIYJ   /YJ
X2YJ=XEIYJ   +2.0*DXERYJ   /YJ
B3YJ=YJ   *DBERYJ   -2.0*(1.0-POIS)*B1YJ
B4YJ=YJ   *DBEIYJ   -2.0*(1.0-POIS)*B2YJ
X3YJ=YJ   *DXERYJ   -2.0*(1.0-POIS)*X1YJ
X4YJ=YJ   *DXEIYJ   -2.0*(1.0-POIS)*X2YJ
B5YJ=-0.5*YJ   *DBERYJ   +(1.0+POIS)*B1YJ
B6YJ=-0.5*YJ   *DBEIYJ   +(1.0+POIS)*B2YJ
X5YJ=-0.5*YJ   *DXERYJ   +(1.0+POIS)*X1YJ
X6YJ=-0.5*YJ   *DXEIYJ   +(1.0+POIS)*X2YJ
B7YJ=BERYJ   -0.5*YJ   *DBERYJ
B8YJ=BEIYJ   -0.5*YJ   *DBEIYJ
X7YJ=XERYJ   -0.5*YJ   *DXERYJ
X8YJ=XEIYJ   -0.5*YJ   *DXEIYJ
B9YJ=POIS*BERYJ   -2.0*(1.0+POIS)*DBEIYJ   /YJ
B10YJ=POIS*BEIYJ   +2.0*(1.0+POIS)*DBERYJ   /YJ
X9YJ=POIS*XERYJ   -2.0*(1.0+POIS)*DXEIYJ   /YJ
X10YJ=POIS*XEIYJ   +2.0*(1.0+POIS)*DXERYJ   /YJ
SP1=-X1YJ*X3YJ-X2YJ*X4YJ
```

```
      SP2=-X4YI*X3YJ+X3YI*X4YJ
      SP3=-X1YI*X3YJ-X2YI*X4YJ
      SP4=X4YI*X2YJ+X3YI*X1YJ
      SP5=X1YI*X2YJ-X2YI*X1YJ
      SP6=-X1YI*X3YI-X2YI*X4YI
      SP7=-B1YJ*B3YJ-B2YJ*B4YJ
      SP8=-B4YI*B3YJ+B3YI*B4YJ
      SP9=-B1YI*B3YJ-B2YI*B4YJ
      SP10=B4YI*B2YJ+B3YI*B1YJ
      SP11=B1YI*B2YJ-B2YI*B1YJ
      SP12=-B1YI*B3YI-B2YI*B4YI
      DM11=-B3YI*SP1-B2YJ*SP2-B3YJ*SP4
      DM21=B2YI*SP1-B2YJ*SP3-B3YJ*SP5
      DM31=B2YI*SP2+B3YI*SP3-B3YJ*SP6
      DM41=B2YI*SP4+B3YI*SP5+B2YJ*SP6
      DM12=B4YI*SP1-B1YJ*SP2+B4YJ*SP4
      DM22=B1YI*SP1-B1YJ*SP3+B4YJ*SP5
      DM32=B1YI*SP2-B4YI*SP3+B4YJ*SP6
      DM42=B1YI*SP4-B4YI*SP5+B1YJ*SP6
      DM13=-X3YI*SP7-X2YJ*SP8-X3YJ*SP10
      DM23=X2YI*SP7-X2YJ*SP9-X3YJ*SP11
      DM33=X2YI*SP8+X3YI*SP9-X3YJ*SP12
      DM43=X2YI*SP10+X3YI*SP11+X2YJ*SP12
      DM14=X4YI*SP7-X1YJ*SP8+X4YJ*SP10
      DM24=X1YI*SP7-X1YJ*SP9+X4YJ*SP11
      DM34=X1YI*SP8-X4YI*SP9+X4YJ*SP12
      DM44=X1YI*SP10-X4YI*SP11+X1YJ*SP12
      DEL=B1YI*DM11-B4YI*DM21+B1YJ*DM31-B4YJ*DM41
      IF(ABS (DEL)-1.0E-18) 30,40,40
   40 FSSDI=SI    *SINALP    /DEL
      FSSDJ=SJ    *SINALP    /DEL
      FYSDI=0.5*YI    **2.0/DEL
      FYSDJ=0.5*YJ    **2.0/DEL
      A11=-FSSDI*DM11
      A12=-FYSDI*DM21
      A13=-FSSDJ*DM31
      A14=-FYSDJ*DM41
      A21=FSSDI*DM12
      A22=FYSDI*DM22
      A23=FSSDJ*DM32
      A24=FYSDJ*DM42
      B11=-FSSDI*DM13
      B12=-FYSDI*DM23
      B13=-FSSDJ*DM33
      B14=-FYSDJ*DM43
      B21=FSSDI*DM14
      B22=FYSDI*DM24
      B23=FSSDJ*DM34
      B24=FYSDJ*DM44
      CP2=COSALP    /SINALP    /(E*TH)
      CP1=2.0*(3.0*(1.0-POIS**2))**0.5*CP2/TH
      CP3=CP2*COSALP
      CLOG=ALOG(SJ    /SI    )
      F(1,1)=-CP1*(A12*B2YI-A22*B1YI+B12*X2YI-B22*X1YI)
      F(1,2)=-CP1*(A14*B2YI-A24*B1YI+B14*X2YI-B24*X1YI)
      F(1,3)=-CP1*(A11*B2YI-A21*B1YI+B11*X2YI-B21*X1YI)
```

```
      F(1,4)=-CP1*(A13*B2YI-A23*B1YI+B13*X2YI-B23*X1YI)
      F(1,5)=CP2*SJ    /SI
      F(2,1)=CP1*(A12*B2YJ-A22*B1YJ+B12*X2YJ-B22*X1YJ)
      F(2,2)=CP1*(A14*B2YJ-A24*B1YJ+B14*X2YJ-B24*X1YJ)
      F(2,3)=CP1*(A11*B2YJ-A21*B1YJ+B11*X2YJ-B21*X1YJ)
      F(2,4)=CP1*(A13*B2YJ-A23*B1YJ+B13*X2YJ-B23*X1YJ)
      F(2,5)=-CP2
      F(3,1)=-CP3*(A12*B5YI+A22*B6YI+B12*X5YI+B22*X6YI)
      F(3,2)=-CP3*(A14*B5YI+A24*B6YI+B14*X5YI+B24*X6YI)
      F(3,3)=-CP3*(A11*B5YI+A21*B6YI+B11*X5YI+B21*X6YI)
      F(3,4)=-CP3*(A13*B5YI+A23*B6YI+B13*X5YI+B23*X6YI)
      F(3,5)=POIS*SJ    *COSALP    /(E*TH)
      F(4,1)=CP3*(A12*B5YJ+A22*B6YJ+B12*X5YJ+B22*X6YJ)
      F(4,2)=CP3*(A14*B5YJ+A24*B6YJ+B14*X5YJ+B24*X6YJ)
      F(4,3)=CP3*(A11*B5YJ+A21*B6YJ+B11*X5YJ+B21*X6YJ)
      F(4,4)=CP3*(A13*B5YJ+A23*B6YJ+B13*X5YJ+B23*X6YJ)
      F(4,5)=-POIS*SJ    *COSALP    /(E*TH)
      DFB9=B9YJ-B9YI
      DFB10=B10YJ-B10YI
      DFX9=X9YJ-X9YI
      DFX10=X10YJ-X10YI
      F(5,1)=CP2*(A12*DFB9+A22*DFB10+B12*DFX9+B22*DFX10)
      F(5,2)=CP2*(A14*DFB9+A24*DFB10+B14*DFX9+B24*DFX10)
      F(5,3)=CP2*(A11*DFB9+A21*DFB10+B11*DFX9+B21*DFX10)
      F(5,4)=CP2*(A13*DFB9+A23*DFB10+B13*DFX9+B23*DFX10)
      F(5,5)=SJ    *CLOG/(E*TH)
      IF (90.0-ALPH) 21,22,23
   21 DO 25 I=1,5
      F(I,3)=-F(I,3)
   25 F(I,4)=-F(I,4)
      DO 26 I=3,5
   26 F(I,5)=-F(I,5)
      GO TO 23
   22 PRINT 27
   27 FORMAT (5X,46HTHIS ELEMENT NEEDS THE USE OF SUBROUTINE CYSTI)
      GO TO 30
C     MODIFY FLEXIBILITY TO ENSURE PROPER SYMMETRY PROPERTIES
   23 DA(1)=SJ    /SI
      DA(2)=F(1,2)/F(2,1)
      DA(3)=F(3,2)/F(2,3)
      DA(4)=F(1,4)/F(4,1)
      DA(5)=F(1,5)/F(5,1)
      DA(6)=F(3,4)/F(4,3)
      DA(7)=F(3,5)/F(5,3)
      EA(1)=F(1,3)/F(3,1)
      EA(2)=F(2,4)/F(4,2)
      EA(3)=F(2,5)/F(5,2)
      EA(4)=F(4,5)/F(5,4)
      TI=SJ    -SI
   95 PRINT 50,((F(I,J),J=1,5),I=1,5)
   50 FORMAT (5E20.8)
      F(1,3)=(F(1,3)+F(3,1))/2.
      F(3,1)=F(1,3)
      F(2,4)=(F(2,4)+F(4,2))/2.
      F(4,2)=F(2,4)
      F(1,2)=(F(1,2)+DA(1)*F(2,1))/2.
```

```
      F(2,1)=F(1,2)/DA(1)
      F(3,2)=(F(3,2)+DA(1)*F(2,3))/2.
      F(2,3)=F(3,2)/DA(1)
      F(1,4)=(F(1,4)+DA(1)*F(4,1))/2.
      F(4,1)=F(1,4)/DA(1)
      F(3,4)=(F(3,4)+DA(1)*F(4,3))/2.
      F(4,3)=F(3,4)/DA(1)
      IF (ALPH-30.0) 31,32,32
   31 F(1,5)=DA(1)*F(5,1)
      F(2,5)=F(5,2)
      F(3,5)=DA(1)*F(5,3)
      F(4,5)=F(5,4)
      GO TO 38
   32 IF(ALPH-60.0) 33,33,34
   33 F(1,5)=(F(1,5)+F(5,1)*DA(1))/2.
      F(5,1)=F(1,5)/DA(1)
      F(2,5)=(F(2,5)+F(5,2))/2.
      F(5,2)=F(2,5)
      F(3,5)=(F(3,5)+DA(1)*F(5,3))/2.
      F(5,3)=F(3,5)/DA(1)
      F(4,5)=(F(4,5)+F(5,4))/2.
      F(5,4)=F(4,5)
      GO TO 38
   34 IF (120.0-ALPH) 35,35,36
   36 F(5,1)=F(1,5)/DA(1)
      F(5,2)=F(2,5)
      F(5,3)=F(3,5)/DA(1)
      F(5,4)=F(4,5)
      GO TO 38
   35 IF (150.0-ALPH) 31,33,33
   38 CALL INVERT(F,5,5,M5,C5)
      PRINT 50,((F(I,J),J=1,5),I=1,5)
   30 RETURN
      END
C
C
C
      SUBROUTINE L3
C     COMPUTATION OF ELEMENT MASS MATRICES AND ASSEMBLY AND REARRANGE-
C     MENT OF OVERALL STIFFNESS AND MASS MATRICES
      DIMENSION XRI(25),XRJ(25),XXEKT(25,5,5),XMO(3,3),XCALP(25),
     1XSALP(25)
      COMMON
     1N,NS,NSO,NSD,NFRSC,NAS,NN,NEP,NAF,MAXN,MN3,MN1,MNEP,MAXP,MN,MM,NT,
     2HD,BETA,R,RR,TO,ALPHO,POISO,EO,XWO,GA,TIME,
     3XM(78,78),EK(78,78),P(78),                        XALPH(25),XPOIS(25),
     4XE(25),HH(26),XSI(25),XSJ(25),XW(25),EKO(2,2),XA(25),XN(26),YN(26)
      REWIND 2
      READ (2) (((XXEKT(I,J,K),I=1,N),J=1,5),K=1,5)
      READ (2) (XCALP(I),I=1,N)
      READ (2) (XSALP(I),I=1,N)
      DO 10 I=1,N
      XRI(I)=ABS(XSI(I)*XCALP(I))
   10 XRJ(I)=ABS(XSJ(I)*XCALP(I))
      REWIND 2
      WRITE (2) (((XXEKT(I,J,K),I=1,N),J=1,5),K=1,5)
```

```
      WRITE (2) (XCALP(I),I=1,N)
      WRITE (2) (XSALP(I),I=1,N)
      WRITE (2) (XRI(I),I=1,N)
      WRITE (2) (XRJ(I),I=1,N)
      IF (NSO .EQ. 0) GO TO 210
      RO=RR*SIN(ALPHO/57.29578)
  210 IF (NSD .EQ. 1) GO TO 30
C     COMPUTE DISC MASS MATRIX
      DO 201 I=1,3
      DO 201 J=1,3
  201 XMO(I,J)=0.0
      IF (NSO .EQ. 0) GO TO 30
      AMO=3.1415926*TO*XWO/GA
      XMO(1,1)=RO**4*AMO/12.0
      XMO(2,2)=RO**2*AMO/2.0
      XMO(3,3)=RO**2*AMO
      XMO(1,3)=-RO**3*AMO/4.0
      XMO(3,1)=XMO(1,3)
      PRINT 202, XMO
  202 FORMAT (5X,4HXMO=3E20.8/(9X,3E20.8))
   30 CALL SHKMT(XXEKT,XRI,XRJ,XSALP,XCALP,XW,GA,HH,EKO,XMO,RO,N,NSD,EK,
     1XM,MN3,MAXN,MN1,NSO)
      NN1=NN-1
      IF (NSD .EQ. 1) GO TO 305
      CALL SHEKXM(XM,NFRSC,NAS,NN,MN3)
C     SYMMETRIZE OVERALL STIFFNESS MATRIX
      DO 100 I=1,NN1
      LI=I+1
      DO 100 J=LI,NN
  100 XM(I,J)=(XM(I,J)+XM(J,I))/2.0
      DO 110 I=2,NN
      IJ=I-1
      DO 110 J=1,IJ
  110 XM(I,J)=XM(J,I)
  305 CALL SHEKXM(EK,NFRSC,NAS,NN,MN3)
C     SYMMETRIZE OVERALL MASS MATRIX
      DO 400 I=1,NN1
      LI=I+1
      DO 400 J=LI,NN
  400 EK(I,J)=(EK(I,J)+EK(J,I))/2.0
      DO 410 I=2,NN
      IJ=I-1
      DO 410 J=1,IJ
  410 EK(I,J)=EK(J,I)
      RETURN
      END
C
C
      SUBROUTINE SHKMT(XXEKT,XRI,XRJ,XSALP,XCALP,XW,GA,HH,EKO,XMO,RO,
C     COMPUTATION OF CONICAL ELEMENT MASS MATRICES.  ASSEMBLY OF OVERALL
C     STIFFNESS AND MASS MATRICES
     1N,NSD,EK,XM,MN3,MAXN,MN1,NSO)
      DIMENSION EK(MN3,MN3),XM(MN3,MN3),HH(MN1),XXEKT(MAXN,5,5),XRI(MAXN
     1),XRJ(MAXN),XCALP(MAXN),XSALP(MAXN),EKO(2,2)
      DIMENSION XMO(3,3),SKO(3,3),XEK(5,5),XMTII(3,3),XMTIJ(3,3)
      DIMENSION XMTJI(3,3),XMTJJ(3,3),XEKT(5,5),BI(5,3),BJ(5,3),AI(3,5)
```

```
      DIMENSION AJ(3,5),EKBI(5,3),EKBJ(5,3),EKII(3,3),EKIJ(3,3)
      DIMENSION EKJI(3,3),EKJJ(3,3),CI(3,3),XMCII(3,3),XMCIJ(3,3)
      DIMENSION XMCJI(3,3),XMCJJ(3,3),XMII(3,3),XMIJ(3,3),XMJI(3,3)
      DIMENSION XMJJ(3,3),XW(MAXN)
      N13=(N+1)*3
      DO 203 I=1,3
      DO 203 J=1,3
  203 SKO(I,J)=0.0
      IF (NSC .EQ. 0) GO TO 500
      DO 204 I=1,2
      DO 204 J=1,2
  204 SKO(I,J)=EKO(I,J)*2.0*3.1415926*RO
      PRINT 205, SKO
  205 FORMAT (5X,4HSKO=3E20.8/(9X,3E20.8))
  500 IF (NSD .EQ. 1) GO TO 301
      DO 206 I=1,N13
      DO 206 J=1,N13
  206 XM(I,J)=0.0
      DO 115 J=1,3
      DO 115 K=1,3
  115 CI(J,K)=0.0
      CI(1,1)=-1.0
  301 DO 406 I=1,N13
      DO 406 J=1.N13
  406 EK(I,J)=0.0
      DO 59 J=1,5
      DO 59 K=1,3
   59 BI(J,K)=0.0
      BI(1,1)=-1.0
      BI(3,2)=-1.0
      DO 60 J=1,5
      DO 60 K=1,3
   60 BJ(J,K)=0.0
      BJ(2,1)=1.0
      BJ(4,2)=1.0
      DO 61 J=1,3
      DO 61 K=1,5
   61 AI(J,K)=0.0
      AI(1,1)=-1.0
      AI(2,3)=-1.0
      DO 62 J=1,3
      DO 62 K=1,5
   62 AJ(J,K)=0.0
      AJ(1,2)=1.0
      AJ(2,4)=1.0
      DO 200 I=1,N
      RI=XRI(I)
      RJ=XRJ(I)
      CALP=XCALP(I)
      SALP=XSALP(I)
      DO 284 J=1,5
      DO 284 K=1,5
  284 XEK(J,K)=XXEKT(I,J,K)
      IF (NSD .EQ. 1) GO TO 302
C     COMPUTE ELEMENT MASS MATRIX IN LOCAL COORDINATES
      W=XW(I)
```

```
      AL=(RJ-RI)/CALP
      AL=ABS(AL)
      I1=I+1
      TT=0.5*(HH(I)+HH(I1))
      AM=2.0*3.1415926*TT*W/GA
      DO 207 J=1,3
      DO 207 K=1,3
  207 XMTII(J,K)=0.0
      XMTII(1,1)=(RI/105.0+CALP*AL/280.0)*AL**3*AM
      XMTII(2,2)=(13.0*RI+3.0*CALP*AL)*AL*AM/35.0
      XMTII(3,3)=(RI+CALP*AL/4.0)*AL*AM/3.0
      XMTII(1,2)=-(AL*CALP/60.0+11.0*RI/210.0)*AL**2*AM
      XMTII(2,1)=XMTII(1,2)
      DO 208 J=1,3
      DO 208 K=1,3
  208 XMTIJ(J,K)=0.0
      XMTIJ(1,1)=-(RI+CALP*AL/2.0)*AL**3*AM/140.0
      XMTIJ(2,2)=(RI+CALP*AL/2.0)*AL*AM*9.0/70.0
      XMTIJ(3,3)= (RI+CALP*AL/2.0)*AL*AM/6.0
      XMTIJ(1,2)=-(13.0*RI/7.0+CALP*AL)*AL**2*AM/60.0
      XMTIJ(2,1)=(13.0*RI/6.0+CALP*AL)*AL**2*AM/70.0
      DO 209 J=1,3
      DO 209 K=1,3
  209 XMTJI(J,K)=XMTIJ(K,J)
      DO 210 J=1,3
      DO 210 K=1,3
  210 XMTJJ(J,K)=0.0
      XMTJJ(1,1)=(RI/105.0+CALP*AL/168.0)*AL**3*AM
      XMTJJ(2,2)=(13.0*RI/5.0+2.0*CALP*AL)*AL*AM/7.0
      XMTJJ(3,3)=(RI/3.0+CALP*AL/4.0)*AL*AM
      XMTJJ(1,2)=(RI*11.0/210.0+CALP*AL/28.0)*AL**2*AM
      XMTJJ(2,1)=XMTJJ(1,2)
      PRINT 700
  700 FORMAT (83H                                      XMTII
     1                          XMTIJ)
      PRINT 701, (((XMTII(J,K),K=1,3),(XMTIJ(J,L),L=1,3)),J=1,3)
      PRINT 702
  702 FORMAT (83H                                      XMTJI
     1                          XMTJJ)
      PRINT 701, (((XMTJI(J,K),K=1,3),(XMTJJ(J,L),L=1,3)),J=1,3)
  701 FORMAT (2X,3E16.7,4H    3E16.7)
C     SYMMETRIZE ELEMENT STIFFNESS MATRIX
  302 RI=RI*2.0*3.1415926
      RJ=RJ*2.0*3.1415926
      DO 211 L=1,5
      XEKT(1,L)=RI*XEK(1,L)
      XEKT(2,L)=RJ*XEK(2,L)
      XEKT(3,L)=RI*XEK(3,L)
      XEKT(4,L)=RJ*XEK(4,L)
  211 XEKT(5,L)=RJ*XEK(5,L)
      PRINT 112, XEK
  112 FORMAT (5X,4HXEK=5E20.8/(9X,5E20.8))
      PRINT 113, RI,RJ
  113 FORMAT (7X,3HRI=E20.8,5H  RJ=E20.8)
      PRINT 114, XEKT
  114 FORMAT (5X,5HXEKT=5E20.8/(10X,5E20.8))
```

```
C     CONSTRUCT TRANSFORMATION MATRICES
      BI(5,2)=-CALP
      BI(5,3)=SALP
      BJ(5,2)=CALP
      BJ(5,3)=-SALP
      AI(2,5)=-CALP
      AI(3,5)=SALP
      AJ(2,5)=CALP
      AJ(3,5)=-SALP
C     COMPUTE CONTRIBUTIONS TO OVERALL STIFFNESS MATRIX
      DO 63 J=1,5
      DO 63 K=1,3
      EKBI(J,K)=0.0
      DO 63 L=1,5
   63 EKBI(J,K)=EKBI(J,K)+XEKT(J,L)*BI(L,K)
      DO 64 J=1,5
      DO 64 K=1,3
      EKBJ(J,K)=0.0
      DO 64 L=1,5
   64 EKBJ(J,K)=EKBJ(J,K)+XEKT(J,L)*BJ(L,K)
      DO 65 J=1,3
      DO 65 K=1,3
      EKII(J,K)=0.0
      DO 65 L=1,5
   65 EKII(J,K)=EKII(J,K)+AI(J,L)*EKBI(L,K)
      DO 66 J=1,3
      DO 66 K=1,3
      EKIJ(J,K)=0.0
      DO 66 L=1,5
   66 EKIJ(J,K)=EKIJ(J,K)+AI(J,L)*EKBJ(L,K)
      DO 67 J=1,3
      DO 67 K=1,3
      EKJI(J,K)=0.0
      DO 67 L=1,5
   67 EKJI(J,K)=EKJI(J,K)+AJ(J,L)*EKBI(L,K)
      DO 68 J=1,3
      DO 68 K=1,3
      EKJJ(J,K)=0.0
      DO 68 L=1,5
   68 EKJJ(J,K)=EKJJ(J,K)+AJ(J,L)*EKBJ(L,K)
      PRINT 703
  703 FORMAT (83H                                        EKII
     1                               EKIJ)
      PRINT 701, ((( EKII(J,K),K=1,3),( EKIJ(J,L),L=1,3)),J=1,3)
      PRINT 704
  704 FORMAT (83H                                        EKJI
     1                               EKJJ)
      PRINT 701, ((( EKJI(J,K),K=1,3),( EKJJ(J,L),L=1,3)),J=1,3)
      IF (NSD .EQ. 1) GO TO 303
C     COMPUTE CONTRIBUTIONS TO OVERALL MASS MATRIX
      CI(2,2)=SALP
      CI(3,3)=SALP
      CI(2,3)=CALP
      CI(3,2)=-CALP
      DO 116 J=1,3
      DO 116 K=1,3
```

```
      XMCII(J,K)=0.0
      XMCIJ(J,K)=0.0
      XMCJI(J,K)=0.0
      XMCJJ(J,K)=0.0
      DO 116 L=1,3
      XMCII(J,K)=XMCII(J,K)+CI(L,J)*XMTII(L,K)
      XMCIJ(J,K)=XMCIJ(J,K)+CI(L,J)*XMTIJ(L,K)
      XMCJI(J,K)=XMCJI(J,K)+CI(L,J)*XMTJI(L,K)
  116 XMCJJ(J,K)=XMCJJ(J,K)+CI(L,J)*XMTJJ(L,K)
      DO 117 J=1,3
      DO 117 K=1,3
      XMII(J,K)=0.0
      XMIJ(J,K)=0.0
      XMJI(J,K)=0.0
      XMJJ(J,K)=0.0
      DO 117 L=1,3
      XMII(J,K)=XMII(J,K)+XMCII(J,L)*CI(L,K)
      XMIJ(J,K)=XMIJ(J,K)+XMCIJ(J,L)*CI(L,K)
      XMJI(J,K)=XMJI(J,K)+XMCJI(J,L)*CI(L,K)
  117 XMJJ(J,K)=XMJJ(J,K)+XMCJJ(J,L)*CI(L,K)
      PRINT 705
  705 FORMAT (83H                                  XMII
     1                                XMIJ)
      PRINT 701, ((( XMII(J,K),K=1,3),( XMIJ(J,L),L=1,3)),J=1,3)
      PRINT 706
  706 FORMAT (83H                                  XMJI
     1                                XMJJ)
      PRINT 701, ((( XMJI(J,K),K=1,3),( XMJJ(J,L),L=1,3)),J=1,3)
C     ADD CONTRIBUTIONS TO OVERALL MASS MATRIX
      DO 82 J=1,3
      I1J=(I-1)*3+J
      DO 82 K=1,3
      I1K=(I-1)*3+K
   82 XM(I1J,I1K)=XM(I1J,I1K)+XMII(J,K)
      DO 83 J=1,3
      I1J=(I-1)*3+J
      DO 83 K=1,3
      I2K=I*3+K
   83 XM(I1J,I2K)=XM(I1J,I2K)+XMIJ(J,K)
      DO 84 J=1,3
      I2J=I*3+J
      DO 84 K=1,3
      I1K=(I-1)*3+K
   84 XM(I2J,I1K)=XM(I2J,I1K)+XMJI(J,K)
      DO 85 J=1,3
      I2J=I*3+J
      DO 85 K=1,3
      I2K=I*3+K
   85 XM(I2J,I2K)=XM(I2J,I2K)+XMJJ(J,K)
C     ADD CONTRIBUTIONS TO OVERALL STIFFNESS MATRIX
  303 DO 72 J=1,3
      I1J=(I-1)*3+J
      DO 72 K=1,3
      I1K=(I-1)*3+K
   72 EK(I1J,I1K)=EK(I1J,I1K)+EKII(J,K)
      DO 73 J=1,3
```

```
      I1J=(I-1)*3+J
      DO 73 K=1,3
      I2K=I*3+K
   73 EK(I1J,I2K)=EK(I1J,I2K)+EKIJ(J,K)
      DO 74 J=1,3
      I2J=I*3+J
      DO 74 K=1,3
      I1K=(I-1)*3+K
   74 EK(I2J,I1K)=EK(I2J,I1K)+EKJI(J,K)
      DO 75 J=1,3
      I2J=I*3+J
      DO 75 K=1,3
      I2K=I*3+K
   75 EK(I2J,I2K)=EK(I2J,I2K)+EKJJ(J,K)
  200 CONTINUE
C     ADD DISC STIFFNESS MATRIX TO OVERALLL
      DO 78 J=1,3
      DO 78 K=1,3
   78 EK(J,K)=EK(J,K)+SKO(J,K)
      IF (NSD .EQ. 1) GO TO 304
C     ADD DISC MASS MATRIX TO OVERALL
      DO 88 J=1,3
      DO 88 K=1,3
   88 XM(J,K)=XM(J,K)+XMO(J,K)
  304 RETURN
      END
C
C
      SUBROUTINE SHEKXM(XM,NFRSC,NAS,NN,MN3)
C     REARRANGEMENT OF OVERALL STIFFNESS AND MASS MATRICES ACCORDING TO
C     BOUNDARY CONDITIONS
      DIMENSION XM(MN3,MN3)
      NASA=(NAS-1)*3+NFRSC
      DO 11 I=1,NN
      IF (I-NASA) 5,5,6
    5 NI=I
      GO TO 7
    6 NI=I+3-NFRSC
    7 DO 11 J=1,NN
      IF (J-NASA) 8,8,9
    8 NJ=J
      GO TO 10
    9 NJ=J+3-NFRSC
   10 XM(I,J)=XM(NI,NJ)
   11 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE L4
C     CALCULATION OF FREQUENCIES AND MODE SHAPES
      DIMENSION EKT(78,78),EVL1(78),A(78),B(78),W1(78),W2(78),EVT(78,78)
     1,QMQ(78,78),QKQ(78,78),FREQ(78),EVL(78),QT(78,78)
      COMMON
     1N,NS,NSO,NSD,NFRSC,NAS,NN,NEP,NAF,MAXN,MN3,MN1,MNEP,MAXP,MN,MM,NT,
```

```
     2HD,BETA,R,RR,TO,ALPHO,POISO,EO,XWO,GA,TIME,
     3XM(78,78),EK(78,78),P(78),                    XALPH(25),XPOIS(25),
     4XE(25),HH(26),XSI(25),XSJ(25),XW(25),EKO(2,2),XA(25),XN(26),YN(26)
      EQUIVALENCE (EK,EVT,QKQ),(XM,QT,QMQ)
      NNN=-NN
      NE=-NEP
      REWIND 8
      WRITE (8) ((XM(I,J),I=1,NN),J=1,NN)
C     COMPUTE EIGENVALUES (EVL1) AND EIGENVECTORS (EKT) OF  XM
      CALL HOWF(NN,MN3,NNN,XM,EVL1,EKT,A,B,W1,W2)
  183 PRINT 150, (EVL1(I),I=1,NN)
  150 FORMAT (5X,5HEVL1=6E18.8/(10X,6E18.8))
      CALL MULT2(EKT,EK,XM,NN,NN,NN,MN3,MN3,MN3)
      REWIND 9
      WRITE (9) ((EK(I,J),I=1,NN),J=1,NN)
      CALL MULT1(XM,EKT,EK,NN,NN,NN,MN3,MN3,MN3)
      DO 400 I=1,NN
  400 EVL1(I)=1.0/(SQRT(EVL1(I)))
      DO 401 I=1,NN
      DO 401 J=1,NN
  401 XM(I,J)=EVL1(I)*EK(I,J)*EVL1(J)
C     NORMALIZE AND SYMMETRIZE  XM = K BAR
      AX=XM(1,1)
      DO 420 I=1,NN
      DO 420 J=1,NN
  420 XM(I,J)=XM(I,J)/AX
      DO 100 I=1,NN1
      LI=I+1
      DO 100 J=LI,NN
  100 XM(I,J)=(XM(I,J)+XM(J,I))/2.0
      DO 110 I=2,NN
      IJ=I-1
      DO 110 J=1,IJ
  110 XM(I,J)=XM(J,I)
C     COMPUTE EIGENVALUES (EVL) AND EIGENVECTORS (EVT) OF  XM = K BAR
      CALL HOWF(NN,MN3,NE,XM,EVL,EVT,A,B,W1,W2)
      DO 300 I=1,NEP
  300 EVL(I)=EVL(I)*AX
  283 PRINT 250, (EVL(I),I=1,NEP)
  250 FORMAT (5X,4HEVL=6E18.8/(9X,6E18.8))
C     CALCULATE NATURAL MODE SHAPES
      DO 415 I=1,NN
       DO 415 J=1,NEP
  415 EVT(I,J)=EVL1(I)*EVT(I,J)
      CALL MULT1(EKT,EVT,QT,NN,NN,NEP,MN3,MN3,MNEP)
      PRINT 21
   21 FORMAT(5X,25HFIRST NEP MODESHAPES ARE,)
      CALL PRINTM(QT,NN,NEP,MN3)
      REWIND 9
      READ (9) ((EK(I,J),I=1,NN),J=1,NN)
      CALL MULT2(QT,EK,EKT,NN,NEP,NN,MN3,MNEP,MN3)
      CALL MULT1(EKT,QT,QKQ,NEP,NN,NEP,MNEP,MN3,MNEP)
      CALL PRINTM(QKQ,NEP,NEP,MNEP)
      DO 91 I=1,NN
      DO 91 J=1,NEP
   91 EK(I,J)=QT(I,J)
```

```
      REWIND 8
      READ (8) ((XM(I,J),I=1,NN),J=1,NN)
      CALL MULT2(EK,XM,EKT,NN,NEP,NN,MN3,MNEP,MN3)
      CALL MULT1(EKT,EK,QMQ,NEP,NN,NEP,MNEP,MN3,MNEP)
      CALL PRINTM(QMQ,NEP,NEP,MNEP)
      REWIND 8
      READ (8) ((XM(I,J),I=1,NN),J=1,NN)
C     CALCULATE NATURAL FREQUENCIES IN RAD/SEC AND IN CPS
      DO 17 I=1,NEP
      P(I)=SQRT(EVL(I))
   17 FREQ(I)=P(I)/6.2831852
      PRINT 18, (FREQ(I),I=1,NEP)
   18 FORMAT(5X,5HFREQ=6E18.8/(10X,6E18.8))
      PRINT 19, (P(I),I=1,NEP)
   19 FORMAT (5X,2HP=6E18.8/(7X,6E18.8))
      RETURN
      END
C
C
      SUBROUTINE MULT2(A,B,C,N1,N2,N3,MN1,MN2,MN3)
C     PREMULTIPLICATION OF A MATRIX BY THE TRANSPOSE OF ANOTHER MATRIX
      DIMENSION A(MN1,MN2),B(MN1,MN3),C(MN2,MN3)
      DO 10 I=1,N2
      DO 10 J=1,N3
      C(I,J)=0.0
      DO 10 K=1,N1
   10 C(I,J)=C(I,J)+A(K,I)*B(K,J)
      RETURN
      END
C
C
      SUBROUTINE MULT1(A,B,C,N1,N2,N3,MN1,MN2,MN3)
C     MULTIPLICATION OF TWO CONFORMABLE MATRICES
      DIMENSION A(MN1,MN2),B(MN2,MN3),C(MN1,MN3)
      DO 10 I=1,N1
      DO 10 J=1,N3
      C(I,J)=0.0
      DO 10 K=1,N2
   10 C(I,J)=C(I,J)+A(I,K)*B(K,J)
      RETURN
      END
C
C
C
      SUBROUTINE L5
C     NUMERICAL INTEGRATION TO GET DYNAMIC DISPLACEMENT RESPONSE
      DIMENSION XMAF(78),XQFT(78),AF(78),FA(3901),EKT(78,78),T(78),
     1QT(78,78),XDEF(78,78)
      COMMON
     1N,NS,NSO,NSD,NFRSC,NAS,NN,NEP,NAF,MAXN,MN3,MN1,MNEP,MAXP,MN,MM,NT,
     2HD,BETA,R,RR,TO,ALPHO,POISO,EO,XWO,GA,TIME,
     3XM(78,78),EK(78,78),P(78),                   XALPH(25),XPOIS(25),
     4XE(25),HH(26),XSI(25),XSJ(25),XW(25),EKO(2,2),XA(25),XN(26),YN(26)
      EQUIVALENCE (EK,QT),(XM,XDEF),(EKT,FA)
   20 FORMAT (4E18.8)
   30 FORMAT (10F7.4)
```

```
   60 FORMAT(E18.8)
   70 FORMAT (2I4)
  220 FORMAT (5X,6HMM,NT=2I4)
  230 FORMAT(5X,5HTIME=E20.8)
  240 FORMAT(5X,3HAF=6E18.8/(8X,6E18.8))
  250 FORMAT(5X,3HFA=15F7.4/(8X,15F7.4))
      READ 70, MM,NT
      READ 60,TIME
      MM1=MM+1
      READ 20,  (AF(I),I=1,NN)
      READ 30, (FA(I),I=1,MM1)
      PRINT 220, MM,NT
      PRINT 230, TIME
      PRINT 240, (AF(I),I=1,NN)
      PRINT 250, (FA(I),I=1,MM1)
      AM=MM
      TT=TIME/AM
      ANT=NT
C     COMPUTE GENERALIZED LOADING
      IF (NAF .EQ. 1) GO TO 39
      DO 37 I=1,NN
      XMAF(I)=0.0
      DO 37 J=1,NN
   37 XMAF(I)=XMAF(I)+XM(I,J)*AF(J)
      DO 38 I=1,NEP
      XQFT(I)=0.0
      DO 38 J=1,NN
   38 XQFT(I)=XQFT(I)+QT(J,I)*XMAF(J)
      GO TO 41
   39 DO 40 I=1,NEP
      XQFT(I)=0.0
      DO 40 J=1,NN
   40 XQFT(I)=XQFT(I)+QT(J,I)*AF(J)
C     COMPUTE MODAL RESPONSES
   41 CALL RESPON(P,XQFT,FA,XDEF,NEP,NT,MM1,TT,MNEP,MAXP)
      MN=MM/NT
      PRINT 50
   50 FORMAT(5X,27HRESPONSE FOR EACH MODE ARE,)
      CALL PRINTM(XDEF,NEP,MN,MNEP)
      WRITE (9) (AF(I),I=1,NN)
      WRITE (9) (XMAF(I),I=1,NN)
      WRITE (9)  ((XDEF(I,J),I=1,NEP),J=1,MN)
      WRITE (9)  (FA(I),I=1,MM1)
C     COMPUTE TOTAL DISPLACEMENT RESPONSE IN GLOBAL COORDINATES BY
C     SUPERIMPOSING MODAL RESPONSES
      DO 44 I=1,NN
      DO 44 J=1,MN
      EKT(I,J)=0.0
      DO 44 K=1,NEP
   44 EKT(I,J)=EKT(I,J)+QT(I,K)*XDEF(K,J)
      DO 46 I=1,MN
      AI=I
   46 T(I)=TT*AI*ANT
      PRINT 51, TT
   51 FORMAT (5X,24HTIME INTERVAL FOR FA(I)=E20.8)
      PRINT 52, (T(I),I=1,MN)
```

```
   52 FORMAT (5X,48HCORRESPONDING TIME OF DISPLACEMENT RESPONSE ARE,/(5X
     1,10F11.7))
      PRINT 53
   53 FORMAT (5X,26HDISPLACEMENT RESPONSE ARE,)
      CALL PRINTM(EKT,NN,MN,MN3)
      WRITE (8) (XMAF(I),I=1,NN)
      WRITE (8) (XQFT(I),I=1,NEP)
      RETURN
      END
C
C
      SUBROUTINE RESPON(W,XM,P,X,NM,NT,L,TT,MAXLD,MAXRP)
C     NUMERICAL INTEGRATION FOR DYNAMIC DISPLACEMENT RESPONSE OF
   20 IX=IX*4
   30 IF (DT-FREQ) 10,10,20
      FREQ=2.*3.1415926/W(N)/32.0
      DT=DT/4.0
      DT=TT
      IX=1
      DO 160 N=1,NM
      DIMENSION W(MAXLD), XM(MAXLD), P(1441), X(MAXLD,MAXRP)
C     INDIVIDUAL MODES
      GO TO 30
   10 C1=DT/2.0
      C2=C1*DT/3.0
      C3=C2*2.0
      K=1
      NOUT=NT
      C4=W(N)**2
      F=1.0+C2*C4
      DISP=0.0
      VEL=0.0
      ACEL=P(1)*XM(N)
      LH=L-1
      AIX=IX
      IIX=IX+1
      DO 160 I=1,LH
      I1=I+1
      DXP=(P(I1)-P(I))/AIX
      XP=P(I)+DXP
      DO 170 J=2,IIX
      A=VEL+C1*ACEL
      B=DISP+DT*VEL+C3*ACEL
      ACEL=(XP*XM(N))/F-(C4/F)*B
      VEL=A+C1*ACEL
      DISP=B+C2*ACEL
  170 XP=XP+DXP
      IF (NOUT.NE.I)  GO TO 160
      X(N,K) = DISP
      K=K+1
      NOUT=NOUT+NT
  160 CONTINUE
      RETURN
      END
C
C
```

```
C
      SUBROUTINE L6
C     INVERSION OF OVERALL STIFFNESS TO GET STATIC DISPLACEMENT AND IN-
C     TERNAL STRESS RESPONSE
      DIMENSION CI(3,3),DI(3),DJ(3),CIS(3),CJS(3),CID(3),CJD(3)
      DIMENSION XXEKT(25,5,5),XCALP(25),XSALP(25),U1(78),P1(25,5)
     1,AI(3,5),AJ(3,5),AIS(3),AJS(3),XRI(25),XRJ(25),XMAF(78),AF(78)
     2,M4(78),C4(78),SO1(2)
      COMMON
     1N,NS,NSO,NSD,NFRSC,NAS,NN,NEP,NAF,MAXN,MN3,MN1,MNEP,MAXP,MN,MM,NT,
     2HD,BETA,R,RR,TC,ALPHC,POISO,EO,XWO,GA,TIME,
     3XM(78,78),EK(78,78),P(78),               XALPH(25),XPOIS(25),
     4XE(25),HH(26),XSI(25),XSJ(25),XW(25),EKO(2,2),XA(25),XN(26),YN(26)
      REWIND 2
      READ (2) (((XXEKT(I,J,K),I=1,N),J=1,5),K=1,5)
      READ (2) (XCALP(I),I=1,N)
      READ (2) (XSALP(I),I=1,N)
      READ (2) (XRI(I),I=1,N)
      READ (2) (XRJ(I),I=1,N)
      IF (NSD .EQ. 1) GO TO 100
      WRITE (8) ((EK(I,J),I=1,NN),J=1,NEP)
      REWIND 9
      READ (9) ((EK(I,J),I=1,NN),J=1,NN)
      READ (9)  (AF(I),I=1,NN)
      READ (9)  (XMAF(I),I=1,NN)
      IF (NAF .EQ. 1) GO TO 310
      DO 101 J=1,NN
  101 AF(J)=XMAF(J)
      GO TO 310
 100  READ 300, (AF(I),I=1,NN)
  300 FORMAT (4E18.8)
      PRINT 320, (AF(I),I=1,NN)
  320 FORMAT (5X,3HAF=6E18.8/(8X,6E18.8))
C     COMPUTE NODAL DISPLACEMENTS IN GLOBAL COORDINATES
  310 CALL INVERT(EK,NN,MN3,M4,C4)
      DO 211 J=1,NN
      U1(J)=0.0
      DO 211 K=1,NN
  211 U1(J)=U1(J)+EK(J,K)*AF(K)
      PRINT 240, (U1(J),J=1,NN)
  240 FORMAT(2X,33HSTATIC DISPLACEMENT DUE TO AF(K)=/(5X,6E16.8))
      CALL SRES(U1,XXEKT,XCALP,XSALP,EKO,NAS,N,NFRSC,P1,SO1,MAXN,MN3)
      PRINT 10, SO1
   10 FORMAT(5X,4HSO1=2E20.8)
      DO 26 J=1,3
      DO 26 K=1,5
      AI(J,K)=0.0
   26 AJ(J,K)=0.0
      AI(1,1)=-1.0
      AI(2,3)=-1.0
      AJ(1,2)=1.0
      AJ(2,4)=1.0
      NASM=NAS-1
      NASA=(NAS-1)*3+NFRSC
      NFRSCP=NFRSC+1
      DO 700 K=1,3
```

```
      DO 700 L=1,3
  700 CI(K,L)=0.0
      CI(1,1)=1.0
      DO 710 J=1,N
      CI(2,2)=XSALP(J)
      CI(3,3)=XSALP(J)
      CI(2,3)=XCALP(J)
      CI(3,2)=-XCALP(J)
      AI(2,5)=-XSJ(J)/XSI(J)*XCALP(J)
      AI(3,5)= XSJ(J)/XSI(J)*XSALP(J)
      AJ(2,5)=XCALP(J)
      AJ(3,5)=-XSALP(J)
C     COMPUTE ELEMENT NODAL FORCES IN GLOBAL COORDINATES
      DO 69 K=1,3
      AIS(K)=0.0
      DO 69 L=1,5
   69 AIS(K)=AIS(K)+AI(K,L)*P1(J,L)
      DO 70 K=1,3
      AJS(K)=0.0
      DO 70 L=1,5
   70 AJS(K)=AJS(K)+AJ(K,L)*P1(J,L)
      JM=J-1
C     SELECT ELEMENT NODAL DISPLACEMENTS IN GLOBAL COORDINATES
      IF (J-NASM) 800,801,802
  800 DO 44 M=1,3
      MJ=M+J*3
      MJM=M+JM*3
      DI(M)=U1(MJM)
   44 DJ(M)=U1(MJ)
      GO TO 95
  801 DO 20 M=1,3
      MJM=M+JM*3
   20 DI(M)=U1(MJM)
      DO 21 M=1,NFRSC
      MJ=M+J*3
   21 DJ(M)=U1(MJ)
      DO 22 M=NFRSCP,3
   22 DJ(M)=0.0
      GO TO 95
  802 IF (J-NAS) 104,105,106
  104 STOP
  105 DO 23 M=1,3
      MJ=NASA+M
      DI(M)=DJ(M)
   23 DJ(M)=U1(MJ)
      GO TO 95
  106 DO 24 M=1,3
      MJ=(J-2)*3+NFRSC+M
      MJM=(J-1)*3+NFRSC+M
      DI(M)=U1(MJ)
   24 DJ(M)=U1(MJM)
C     CALCULATE ELEMENT NODAL FORCES AND DISPLACEMENTS IN LOCAL COORDS
   95 DO 701 K=1,3
      CIS(K)=0.0
      CJS(K)=0.0
      CID(K)=0.0
```

```
      CJD(K)=0.0
      DO 701 L=1,3
      CIS(K)=CIS(K)+CI(K,L)*AIS(L)
      CJS(K)=CJS(K)+CI(K,L)*AJS(L)
      CID(K)=CID(K)+CI(K,L)*DI(L)
  701 CJD(K)=CJD(K)+CI(K,L)*DJ(L)
      PRINT 14, (J,(CIS(K),K=1,3),(CJS(K),K=1,3))
   14 FORMAT(2X,2HN=I4,9H CIS,CJS=6E16.8)
      PRINT 15, (CID(K),K=1,3),(CJD(K),K=1,3)
   15 FORMAT (8X,9H CID,CJD=6E16.8)
  710 CONTINUE
      WRITE (2) ((P1(I,J),I=1,N),J=1,5)
      WRITE (2) (SO1(I),I=1,2)
      WRITE (2) (U1(I),I=1,NN)
  200 RETURN
      END
C
C
      SUBROUTINE SRES (U1,XXEKT,XCALP,XSALP,EKO,NAS,N,NFRSC,P1,SO,MAXN,
C     CALCULATION OF THE STATIC INTERNAL STRESS RESPONSE
     1MN3)
      DIMENSION SO(2),EKO(2,2),P1(MAXN,5),U1(MN3),XXEKT(MAXN,5,5),BI(5,3
     1),BJ(5,3),XEK(5,5),V(5),S(5),AIS(3),AJS(3),DI(3),DJ(3)
     2,XCALP(MAXN),XSALP(MAXN)
      NASA=(NAS-1)*3+NFRSC
      NFRSCP=NFRSC+1
      DO 40 J=1,2
      SO(J)=0.0
      DO 40 K=1,2
   40 SO(J)=SO(J)+EKO(J,K)*U1(K)
      DO 124 J=1,5
      DO 124 K=1,3
      BI(J,K)=0.0
  124 BJ(J,K)=0.0
      BI(1,1)=-1.0
      BI(3,2)=-1.0
      BJ(2,1)=1.0
      BJ(4,2)=1.0
      DO 10 J=1,N
      BI(5,2)=-XCALP(J)
      BI(5,3)=XSALP(J)
      BJ(5,2)=XCALP(J)
      BJ(5,3)=-XSALP(J)
C     SELECT ELEMENT NODAL DISPLACEMENTS IN GLOBAL COORDINATES
      JM=J-1
      NASM=NAS-1
      IF (J-NASM) 100,101,102
  100 DO 44 M=1,3
      MJ=M+J*3
      MJM=M+JM*3
      DI(M)=U1(MJM)
   44 DJ(M)=U1(MJ)
      GO TO 95
  101 DO 20 M=1,3
      MJM=M+JM*3
   20 DI(M)=U1(MJM)
```

```
      DO 21 M=1,NFRSC
      MJ=M+J*3
   21 DJ(M)=U1(MJ)
      DO 22 M=NFRSCP,3
   22 DJ(M)=0.0
      GO TO 95
  102 IF (J-NAS) 104,105,106
  104 GO TO 145
  105 DO 23 M=1,3
      MJ=NASA+M
      DI(M)=DJ(M)
   23 DJ(M)=U1(MJ)
      GO TO 95
  106 DO 24 M=1,3
      MJ=(J-2)*3+NFRSC+M
      MJM=(J-1)*3+NFRSC+M
      DI(M)=U1(MJ)
   24 DJ(M)=U1(MJM)
C     COMPUTE ELEMENT DISPLACEMENTS IN LOCAL COORDINATES
   95 DO 89  K=1,5
      V(K)=0.0
      DO 89 L=1,3
   89 V(K)=V(K)+BI(K,L)*DI(L)
      DO 45 K=1,5
      DO 45 L=1,3
   45 V(K)=V(K)+BJ(K,L)*DJ(L)
C     COMPUTE ELEMENT FORCES IN LOCAL COORDINATES
      DO 49 K=1,5
      DO 49 L=1,5
   49 XEK(K,L)=XXEKT(J,K,L)
      DO 65 K=1,5
      S(K)=0.0
      DO 65 L=1,5
   65 S(K)=S(K)+XEK(K,L)*V(L)
      DO 71 K=1,5
   71 P1(J,K)=S(K)
   10 CONTINUE
  145 RETURN
      END
C
C
C
      SUBROUTINE L7
C     SOLUTION FOR DYNAMIC INTERNAL STRESS RESPONSE
      DIMENSION CI(3,3),DI(3),DJ(3),CIS(3),CJS(3),CID(3),CJD(3)
      DIMENSION FA(3901),XQFT(78),XXEKT(25,5,5),XCALP(25),XSALP(25),
     1XRI(25),XRJ(25),U2(78),P2(25,5),U3(78),SO3(2),P3(25,5),P1(25,5),
     2XNR(78),XNR1(78),XN2(78),TP(25,5),TU2(78),TSO2(2),SSO2(78,2),TU(78
     3),AI(3,5),AJ(3,5),AIS(3),AJS(3),U1(78),SO1(2),SO2(2),TSO(2),
     4EKT(78,78),XMAF(78)
      COMMON
     1N,NS,NSO,NSD,NFRSC,NAS,NN,NEP,NAF,MAXN,MN3,MN1,MNEP,MAXP,MN,MM,NT,
     2HD,BETA,R,RR,TO,ALPHO,POISO,EO,XWO,GA,TIME,
     3XM(78,78),EK(78,78),P(78),                    XALPH(25),XPOIS(25),
     4XE(25),HH(26),XSI(25),XSJ(25),XW(25),EKO(2,2),XA(25),XN(26),YN(26)
      EQUIVALENCE (EKT,FA)
```

```
      EQUIVALENCE (XM,XCALP),(XSALP,XCALP(26)),(XRI,XSALP(26)),
     1(XRJ,XRI(26)),(U2,XRJ(26)),(P2,U2(79)),(U3,P2(126)),(P3,U3(79)),
     2(P1,P3(126)),(XNR,P1(126)),(XNR1,XNR(79)),(XN2,XNR1(79)),
     3                (TP,XN2(79)),(TU2,TP(126)),(SSO2,TU2(79)),
     4(TU,SSO2(157)),(U1,TU(79)),(XXEKT,U1(79))
      MM1=MM+1
      WRITE (2) ((EK(I,J),I=1,NN),J=1,NN)
      REWIND 8
      READ (8) ((XM(I,J),I=1,NN),J=1,NN)
      READ (8) (XMAF(I),I=1,NN)
      READ (8) (XGFT(I),I=1,NEP)
      READ (8) ((EKT(I,J),I=1,NN),J=1,NEP)
C     COMPUTE INERTIAL JOINT LOADS
      DO 200 I=1,NN
      DO 200 J=1,NEP
      EK(I,J)=0.0
      DO 200 K=1,NN
  200 EK(I,J)=EK(I,J)+XM(I,K)*EKT(K,J)
      DO 201 I=1,NEP
      DO 201 J=1,NN
  201 EK(J,I)=EK(J,I)*P(I)*P(I)
      REWIND 2
      READ (2) (((XXEKT(I,J,K),I=1,N),J=1,5),K=1,5)
      READ (2) (XCALP(I),I=1,N)
      READ (2) (XSALP(I),I=1,N)
      READ (2) (XRI(I),I=1,N)
      READ (2) (XRJ(I),I=1,N)
      READ (2) ((P1(I,J),I=1,N),J=1,5)
      READ (2) (SO1(I),I=1,2)
      READ (2) (U1(I),I=1,NN)
      READ (2)  ((EKT(I,J),I=1,NN),J=1,NN)
      REWIND 8
C     FOR EACH MODE, COMPUTE NODAL DISPLACEMENTS IN OVERALL COORDINATES
C     DUE TO INERTIAL LOADS
      DO 203 I=1,NEP
      DO 210 J=1,NN
      U2(J)=0.0
      DO 210 K=1,NN
  210 U2(J)=U2(J)+EKT(J,K)*EK(K,I)
C     FOR EACH MODE, CALCULATE THE ELEMENT FORCES AND DISPLACEMENTS IN
C     LOCAL COORDINATES DUE TO INERTIAL LOADING
      CALL SRES(U2,XXEKT,XCALP,XSALP,EKO,NAS,N,NFRSC,P2,SO2,MAXN,MN3)
      WRITE (8) ((P2(J,K),J=1,N),K=1,5)
      WRITE (8) (U2(K),K=1,NN)
      DO 221 J=1,2
  221 SSO2(I,J)=SO2(J)
  203 CONTINUE
      READ (9) ((EK(I,J),I=1,NEP),J=1,MN)
      READ (9) (FA(I),I=1,MM1)
      DO 26 J=1,3
      DO 26 K=1,5
      AI(J,K)=0.0
   26 AJ(J,K)=0.0
      AI(1,1)=-1.0
      AI(2,3)=-1.0
      AJ(1,2)=1.0
```

```
      AJ(2,4)=1.0
      NASA=(NAS-1)*3+NFRSC
      NFRSCP=NFRSC+1
      NASM=NAS-1
      DO 700 K=1,3
      DO 700 L=1,3
  700 CI(K,L)=0.0
      CI(1,1)=1.0
      DO 205 I=1,MN
      II=I*NT+1
      TIFS=FA(II)
C     AMPLIFY STATIC RESPONSE FOR THE PARTICULAR TIME IN QUESTION
      DO 225 J=1,NN
  225 U3(J)=U1(J)*TIFS
      DO 220 J=1,2
  220 SO3(J)=TIFS*SO1(J)
      DO 208 J=1,N
      DO 208 K=1,5
  208  P3(J,K)=TIFS*P1(J,K)
C     COMPUTE  THE TOTAL RESPONSE IN VARIOUS COORDINATE SYSTEMS FOR THE
C     PARTICULAR TIME IN QUESTION
      DO 206 J=1,NEP
      XNR(J)=EK(J,I)
      XNR1(J)=XQFT(J)*TIFS/P(J)/P(J)
  206 XN2(J)=XNR(J)-XNR1(J)
      DO 500 K=1,N
      DO 500 L=1,5
  500 TP(K,L)=0.0
      DO 501 K=1,NN
  501 TU2(K)=0.0
      REWIND 8
      DO 507 J=1,NEP
      READ (8) ((P2(II,JJ),II=1,N),JJ=1,5)
      READ (8) (U2(II),II=1,NN)
      DO 207 K=1,N
      DO 207 L=1,5
  207 TP(K,L)=TP(K,L)+P2(K,L)*XN2(J)
      DO 226 K=1,NN
  226 TU2(K)=TU2(K)+U2(K)*XN2(J)
  507 CONTINUE
      DO 222 K=1,2
      TSO2(K)=0.0
      DO 222 J=1,NEP
  222 TSO2(K)=          TSO2(K)+SSO2(J,K)*XN2(J)
      DO 223 J=1,2
  223 TSO(J)=TSO2(J)+SO3(J)
      DO 227 J=1,NN
  227 TU(J)=U3(J)+TU2(J)
      DO 209 K=1,N
      DO 209 L=1,5
  209 TP(K,L)= P3(K,L)+TP (K,L)
      PRINT 228, I
      PRINT 229, (TU(J),J=1,NN)
      PRINT 240
      PRINT 230, TSO
      CALL PRINTM(TP,N,5,MAXN)
```

```
      DO 710 J=1,N
      AI(2,5)=-XSJ(J)/XSI(J)*XCALP(J)
      AI(3,5)= XSJ(J)/XSI(J)*XSALP(J)
      AJ(2,5)=XCALP(J)
      AJ(3,5)=-XSALP(J)
      CI(2,2)=XSALP(J)
      CI(3,3)=XSALP(J)
      CI(2,3)=XCALP(J)
      CI(3,2)=-XCALP(J)
      DO 69 K=1,3
      AIS(K)=0.0
      DO 69 L=1,5
   69 AIS(K)=AIS(K)+AI(K,L)*TP(J,L)
      DO 70 K=1,3
      AJS(K)=0.0
      DO 70 L=1,5
   70 AJS(K)=AJS(K)+AJ(K,L)*TP(J,L)
      JM=J-1
      IF (J-NASM) 800,801,802
  800 DO 44 M=1,3
      MJ=M+J*3
      MJM=M+JM*3
      DI(M)=TU(MJM)
   44 DJ(M)=TU(MJ)
      GO TO 95
  801 DO 20 M=1,3
      MJM=M+JM*3
   20 DI(M)=TU(MJM)
      DO 21 M=1,NFRSC
      MJ=M+J*3
   21 DJ(M)=TU(MJ)
      DO 22 M=NFRSCP,3
   22 DJ(M)=0.0
      GO TO 95
  802 IF (J-NAS) 104,105,106
  104 STOP
  105 DO 23 M=1,3
      MJ=NASA+M
      DI(M)=DJ(M)
   23 DJ(M)=TU(MJ)
      GO TO 95
  106 DO 24 M=1,3
      MJ=(J-2)*3+NFRSC+M
      MJM=(J-1)*3+NFRSC+M
      DI(M)=TU(MJ)
   24 DJ(M)=TU(MJM)
   95 DO 701 K=1,3
      CIS(K)=0.0
      CJS(K)=0.0
      CID(K)=0.0
      CJD(K)=0.0
      DO 701 L=1,3
      CIS(K)=CIS(K)+CI(K,L)*AIS(L)
      CJS(K)=CJS(K)+CI(K,L)*AJS(L)
      CID(K)=CID(K)+CI(K,L)*DI(L)
  701 CJD(K)=CJD(K)+CI(K,L)*DJ(L)
```

```
      PRINT 14, (J,(CIS(K),K=1,3),(CJS(K),K=1,3))
   14 FORMAT(2X,2HN=I4,9H CIS,CJS=6E16.8)
      PRINT 15, (CID(K),K=1,3),(CJD(K),K=1,3)
   15 FORMAT (8X,9H CID,CJD=6E16.8)
  710 CONTINUE
  205 CONTINUE
  228 FORMAT (5X,8HI(TIME)=I4)
  229 FORMAT (5X,3HTU=6E16.8/(8X,6E16.8))
  230 FORMAT (4X,4HTSO=2E16.8)
  240 FORMAT(5X,34HSHELL INTERNAL STRESS RESPONSE ARE)
      RETURN
      END
```

REFERENCES

(1) Lu, Z.A., Penzien, J. and Popov, E. P., "Finite Element Solution for Thin Shells of Revolution," I.E.R. Technical Report SESM 63-3, University of California, Berkeley, California, September 1963. Also published as: NASA Contractor Report 37, National Aeronautics and Space Administration, Washington, D.C., July 1964.

(2) Popov, E.P., Penzien, J. and Lu, Z.A., "Finite Element Solution for Axisymmetric Shells," Journal of the Engineering Mechanics Division, Proceedings of the ASCE, Vol. 90, No. EM 5, October 1964, pp. 119-145.

(3) Popov, E.P. and Lu, Z.A., "Edge Disturbances in Axisymmetric Shells of Revolution," I.E.R. Technical Report SESM 64-3, University of California, Berkeley, California, September 1964.

(4) Chow, H.Y. and Popov, E.P., "Finite Element Solution of Axisymmetric Dynamic Problems of Shells of Revolution," Structural Engineering Laboratory Report SESM 66-3, University of California, Berkeley, California, April 1966.

(5) Timoshenko, S.P. and Woinowsky-Krieger, S., Theory of Plates and Shells, Second Edition, McGraw-Hill, New York, 1959.

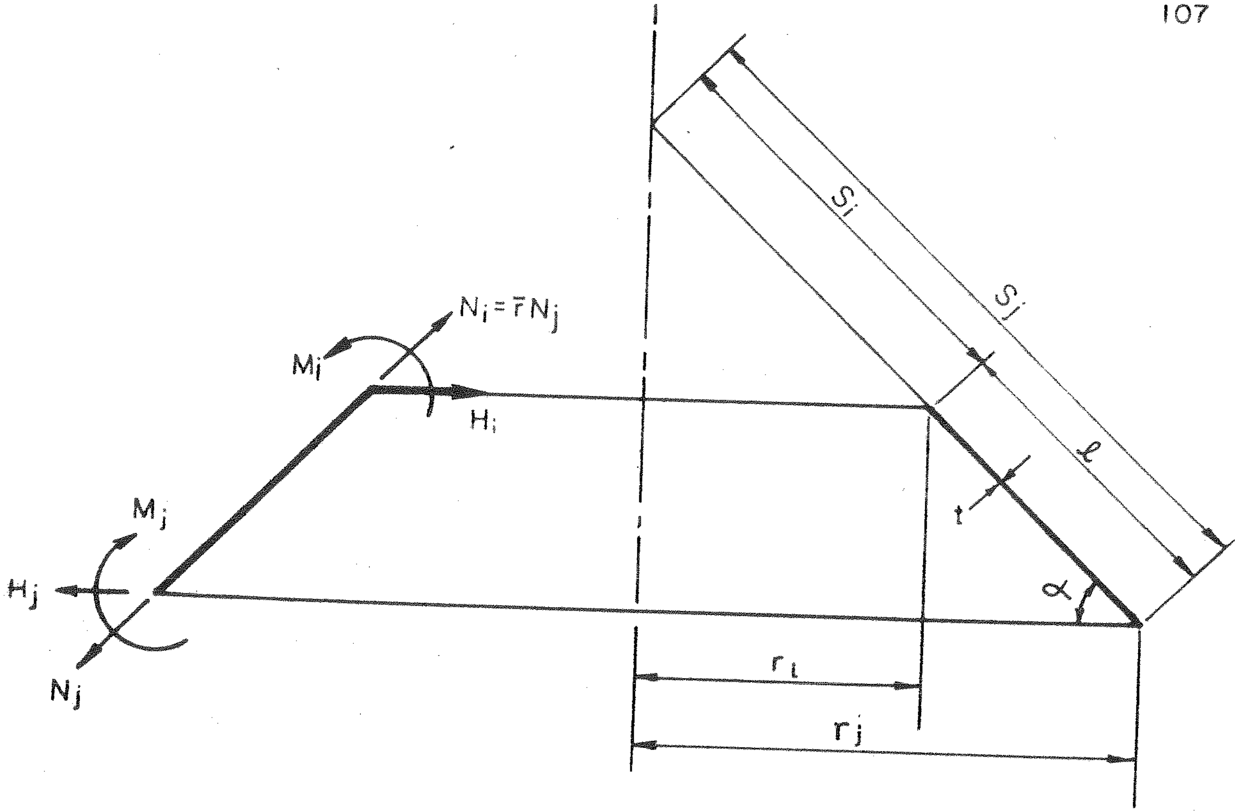
(6) Timoshenko, S.P., Strength of Materials, Part II, Third Edition, Van Nostrand Co., Princeton, New Jersey, 1956.

(7) Flügge, W., Stresses in Shells, Springer-Verlag, Berlin, 1962.

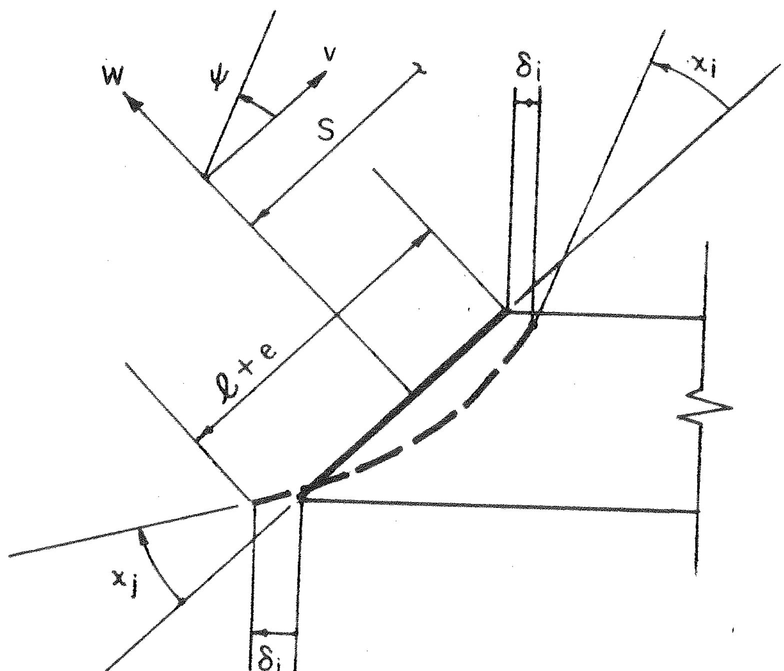
(8) Archer, J.S., "Consistent Matrix Formulations for Structural Analysis Using Finite Element Techniques," AIAA Journal, Vol. 3, No. 10, October 1965, pp. 1910-1918.

(9) Hurty, W.C. and Rubinstein, M.F., Dynamics of Structures, Prentice-Hall, Englewood Cliffs, New Jersey, 1964.

(10) Abramowitz, M. and Stegun, I.A., editors, Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, National Bureau of Standards, Applied Mathematical Series No. 55, June 1964, pp. 379-389.

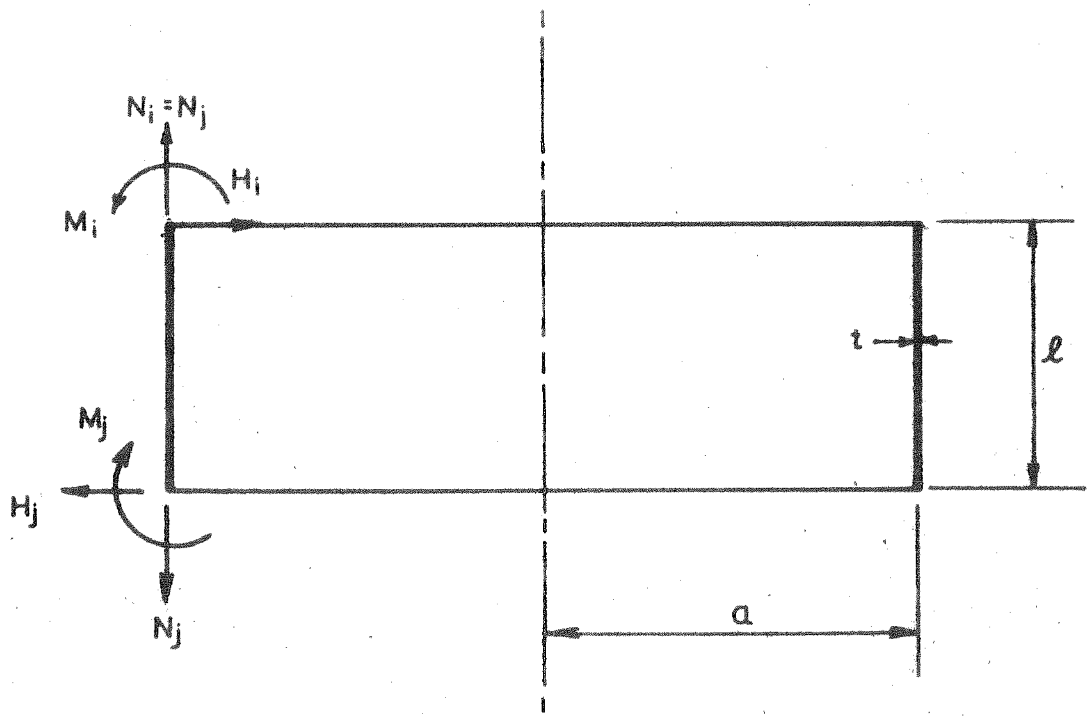


(a) Element Forces and Geometry

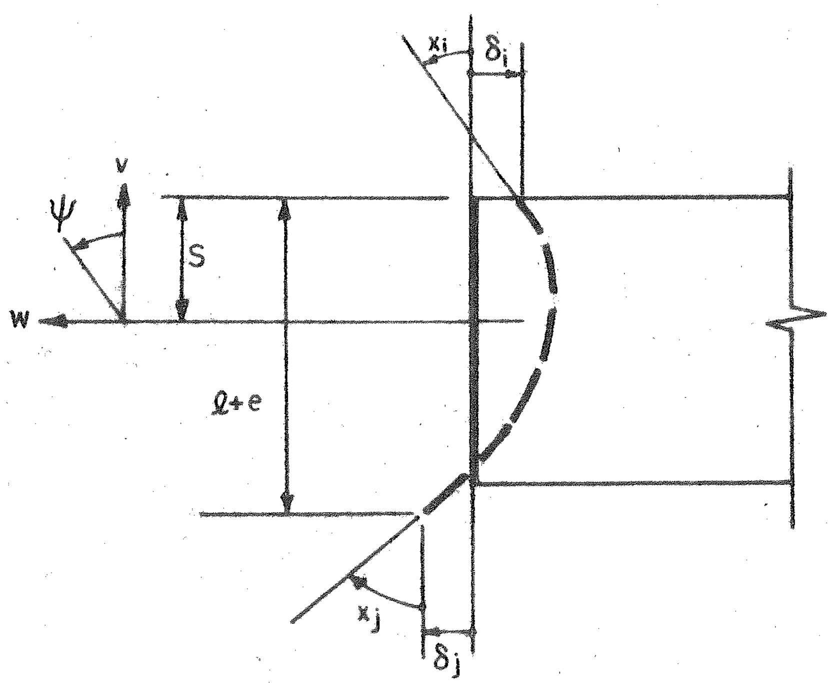


(b) Element Displacements

FIG. I CONICAL ELEMENT

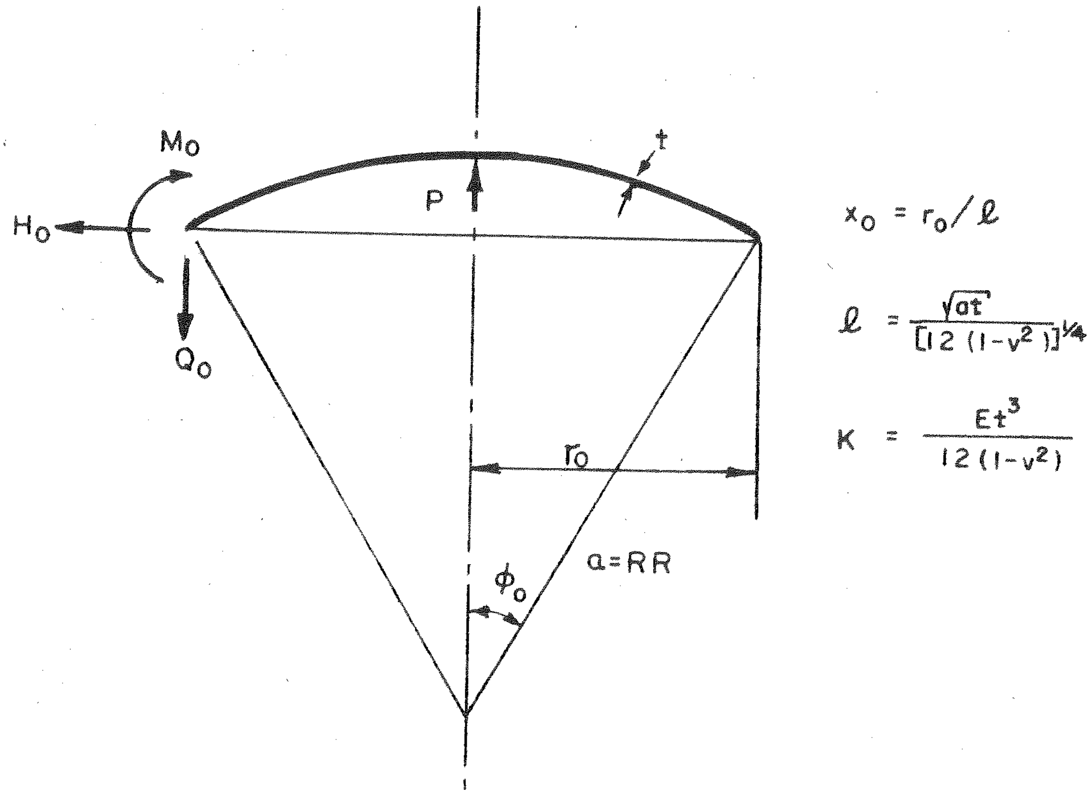


(a) Element Forces and Geometry

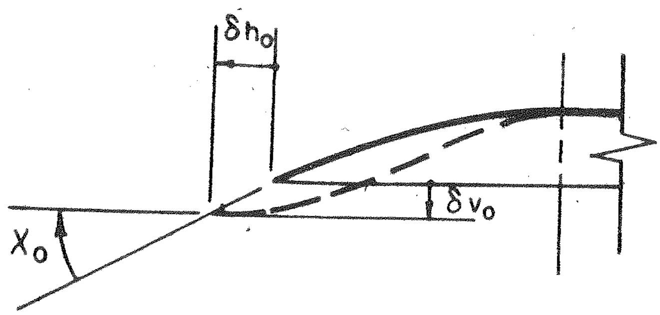


(b) Element Displacements

FIG. 2 CYLINDRICAL ELEMENT

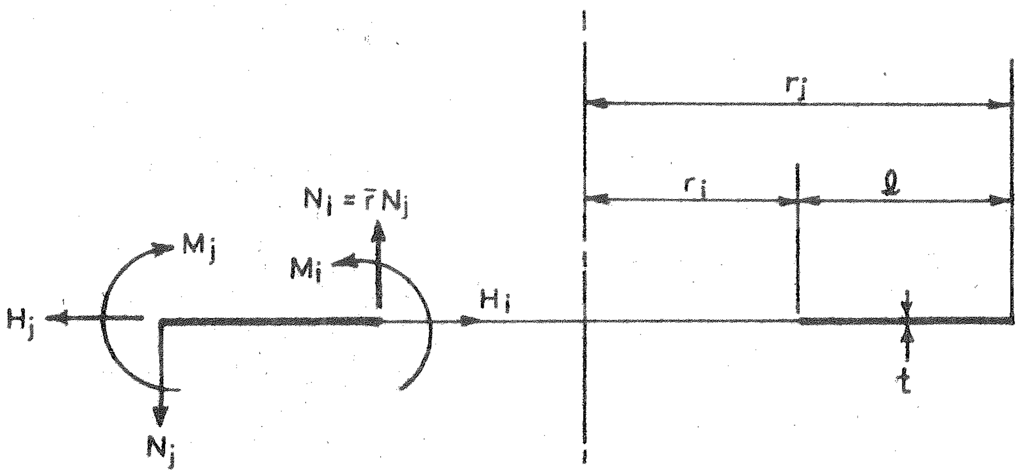


(a) Element Forces and Geometry
(Singular Case)

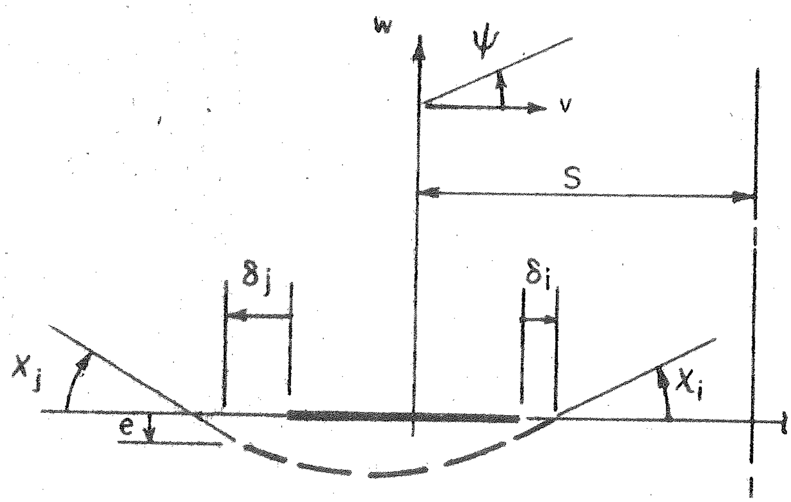


(b) Element Displacements

FIG. 3 SPHERICAL CAP ELEMENT

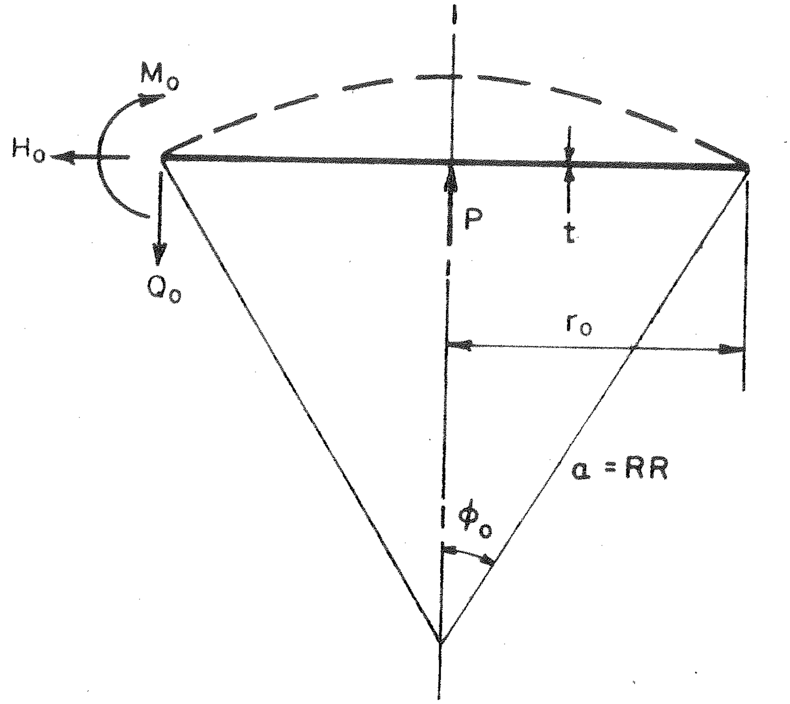


(a) Element Forces and Geometry

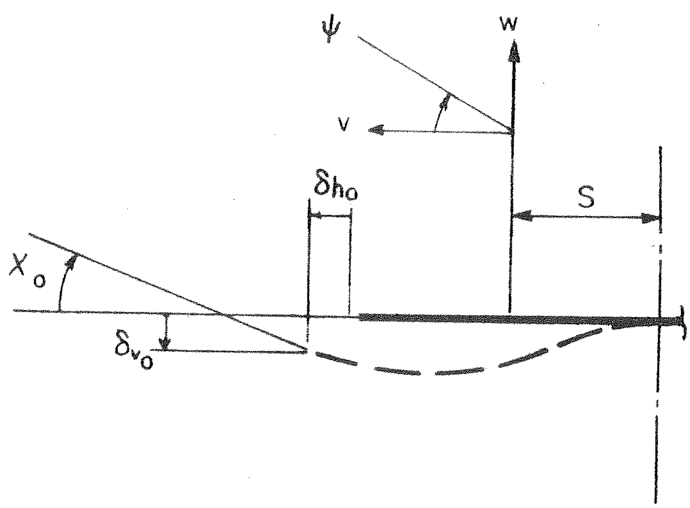


(b) Element Displacements

FIG. 4 ANNULAR RING ELEMENT (CASE $\alpha = 180°$)



(a) Element Forces and Geometry
(Singular Case)



(b) Element Displacements

FIG. 5 DISC ELEMENT

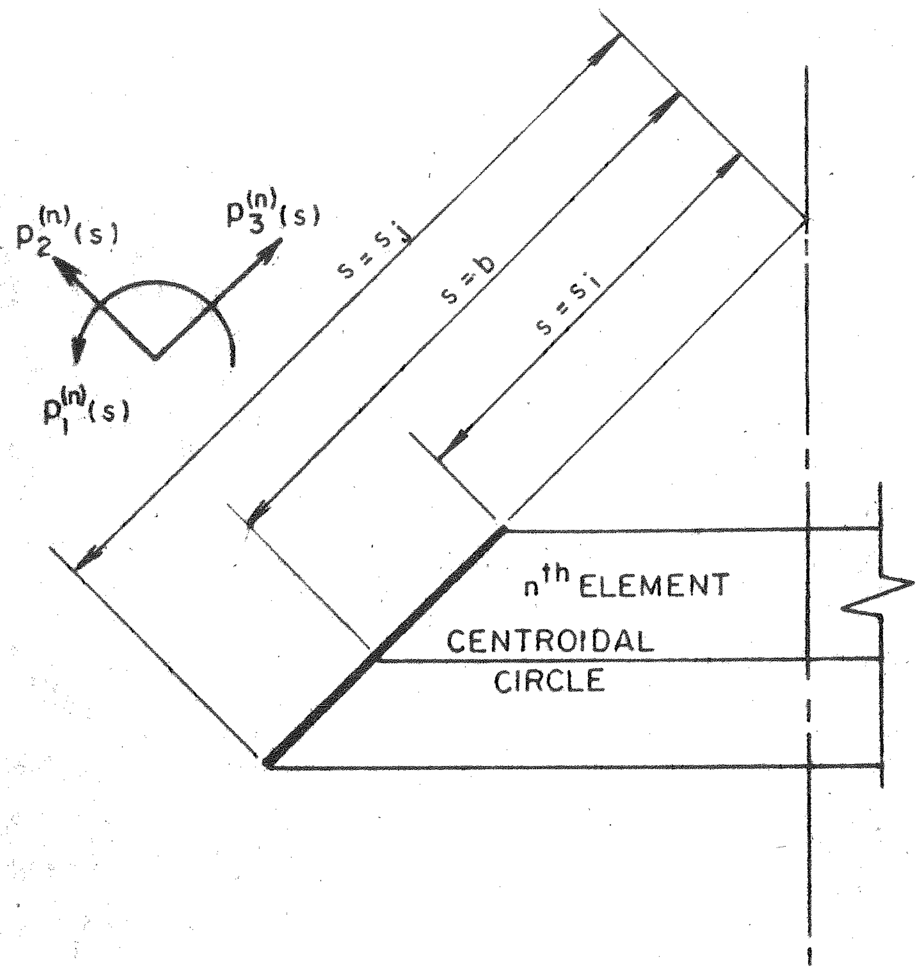
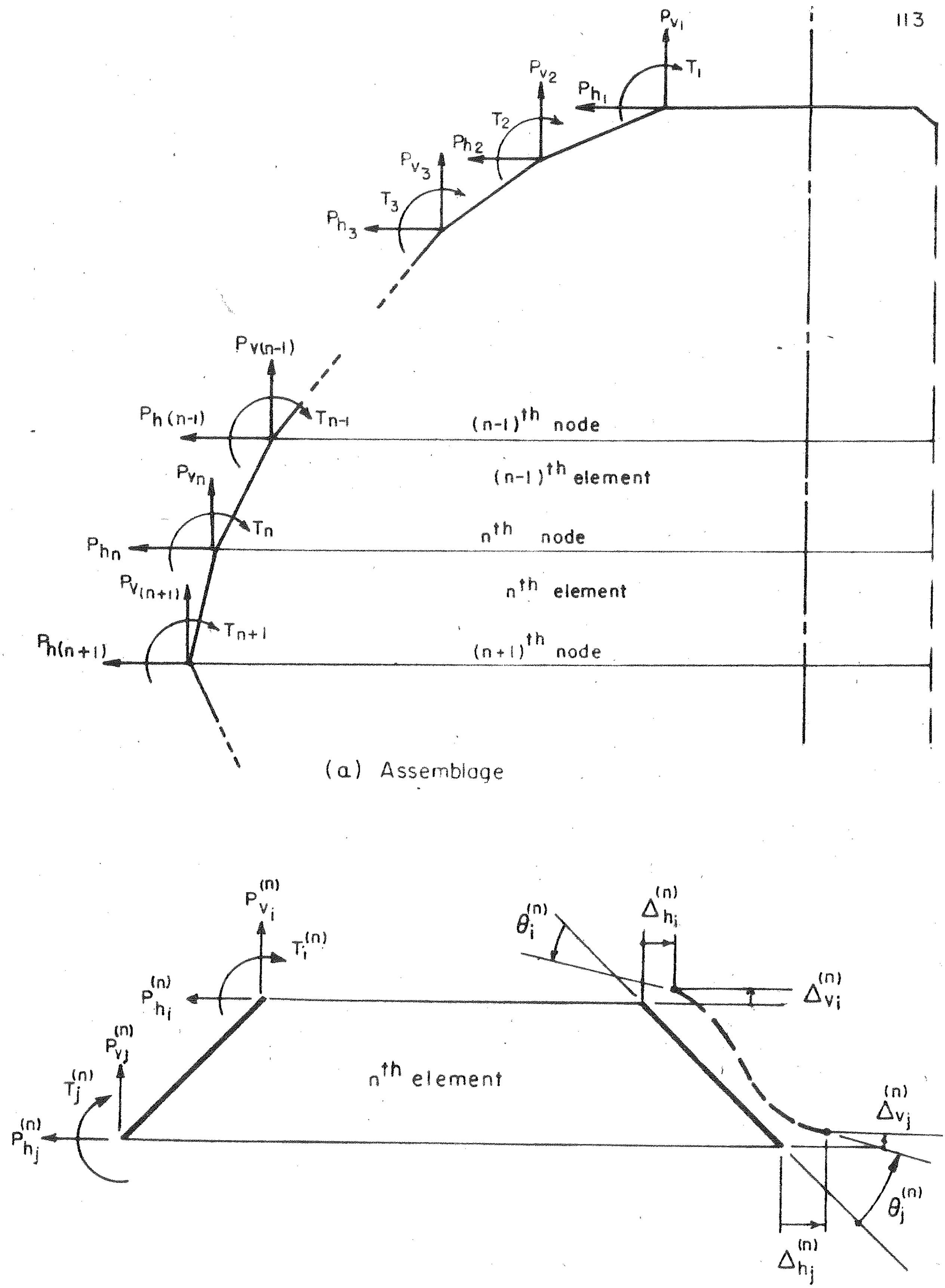


FIG. 6 TRIBUTARY JOINT LOADS



(a) Assemblage

(b) Redefined Element Forces and Displacements

FIG. 7 DISPLACEMENTS AND NODAL FORCES FOR AN ASSEMBLAGE

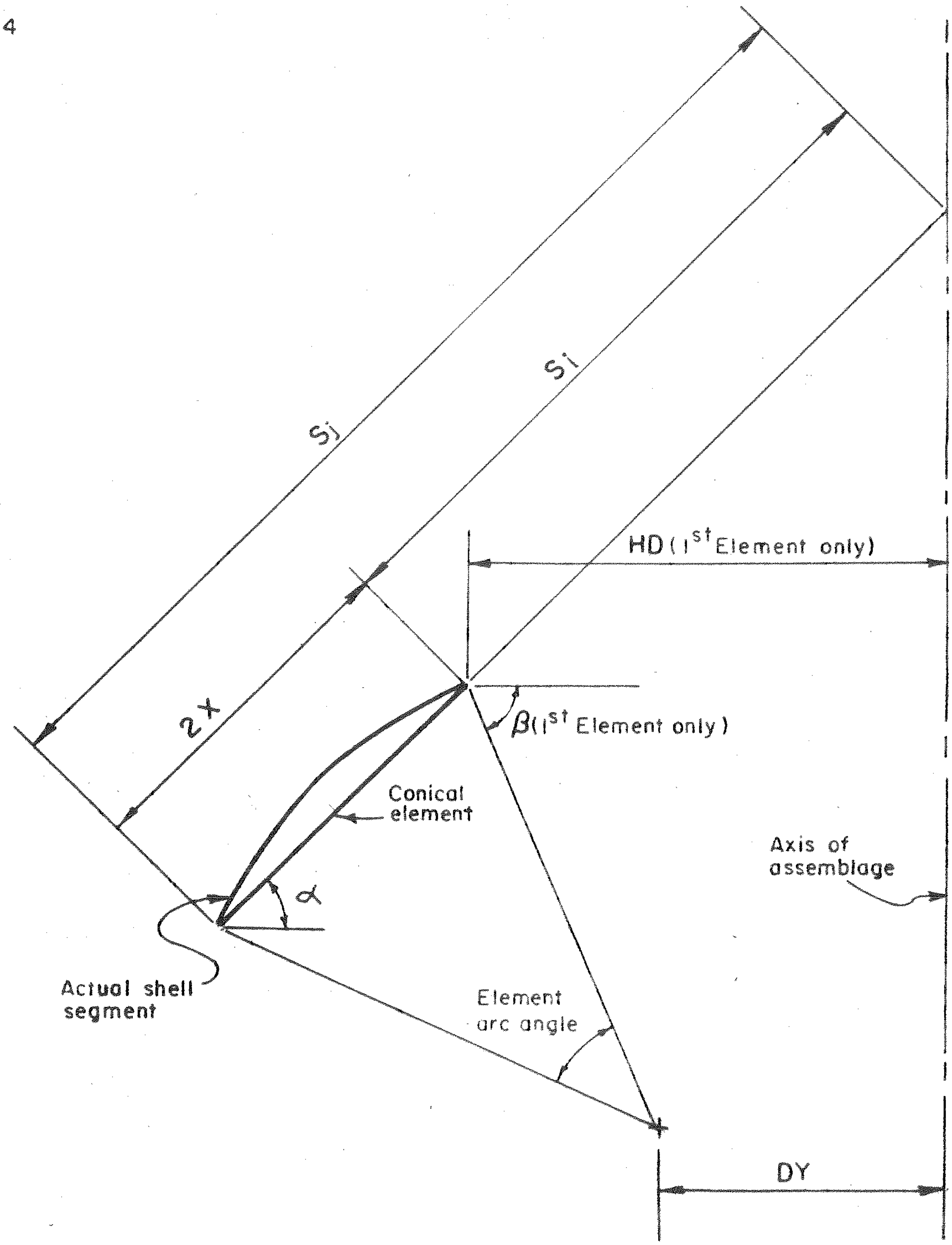


FIG. 8 GEOMETRY OF SHELL ELEMENTS FITTING A CIRCULAR ARC

(POSITIVE GAUSSIAN CURVATURE CASE)

APPENDICES

A. Integration Formulae

For the ith normal mode, the equation to be integrated is

$$\ddot{\eta}_i(t) + \omega_i^2 \eta_i(t) = P_i^*(t) = P_i f(t). \tag{A1-1}$$

The time increment is chosen such that

$$\Delta t \leq (1/32)\ (2\pi/\omega_i). \tag{A1-2}$$

The initial values are

$$\eta_i(0) = \dot{\eta}_i(0) = 0; \quad \ddot{\eta}_i(0) = P_i f(0). \tag{A1-3}$$

Then the general integration formulae are

$$\begin{aligned}
\ddot{\eta}_i(t+\Delta t) &= \{P_i[f(t+\Delta t) - f(t)] - \omega_i^2[\eta_i(t) + \Delta t\dot{\eta}_i(t) + \frac{\Delta t^2}{3}\ddot{\eta}_i(t)]\}/F \\
\dot{\eta}_i(t+\Delta t) &= \dot{\eta}_i(t) + \frac{\Delta t}{2}[\ddot{\eta}_i(t) + \ddot{\eta}_i(t+\Delta t)] \\
\eta_i(t+\Delta t) &= \eta_i(t) + \Delta t\dot{\eta}_i(t) + \frac{\Delta t^2}{3}[\ddot{\eta}_i(t) + \ddot{\eta}_i(t+\Delta t)/2]
\end{aligned} \tag{A1-4}$$

where

$$F = 1 + \frac{\Delta t^2\ \omega_i^2}{6}$$

This integration method assumes a linear acceleration distribution over the increment Δt.

B. Element Stress-Strain Formulae

Given the element forces in local co-ordinates, it is desired to find the stress resultants normally used in engineering. For a spherical cap, these quantities are M_θ, N_θ, N_ϕ, Q_ϕ and M_ϕ and the stresses and strains may in turn be calculated from these using the equations

$$\sigma_\phi = \frac{N_\phi}{t} \pm \frac{6M_\phi}{t^2} \; ; \qquad \sigma_\theta = \frac{N_\theta}{t} \pm \frac{6M_\theta}{t^2}$$

$$\epsilon_\phi = \frac{1}{E}(\sigma_\phi - \nu\sigma_\theta) \; ; \quad \epsilon_\theta = \frac{1}{E}(\sigma_\theta - \nu\,\sigma_\phi). \tag{A2-1}$$

The subscripts ϕ and θ represent meridional and circumferential directions respectively. For a conical or cylindrical element, the desired resultants are N_s, Q_s, M_s, M_θ and N_θ. The stresses ~~and~~ strains in this case are

$$\sigma_s = \frac{N_s}{t} \pm \frac{6M_s}{t^2} \; ; \qquad \sigma_\theta = \frac{N_\theta}{t} \pm \frac{6M_\theta}{t^2}$$

$$\epsilon_s = \frac{1}{E}(\sigma_s - \nu\,\sigma_\theta) \; ; \quad \epsilon_\theta = \frac{1}{E}(\sigma_\theta - \nu\,\sigma_s). \tag{A2-2}$$

The subscripts s and θ represent meridional and circumferential directions respectively.

1. Stress Resultants for a Spherical Cap

The stress resultants for a spherical cap can be found from M_o, H_o and Q_o from the following formulae

$$M_\phi = M_o$$

$$N_\phi = Q_o \sin \phi_o + H_o \cos \phi_o$$

$$Q_o = Q_o \cos \phi_o - H_o \sin \phi_o$$

$$N_\theta = (Et/a) \{[C_{11}M_o + C_{12}H_o + C_{13}Q_o] [\text{ber } x_o - \frac{\text{bei}'x_o}{x_o}] + + [C_{21}M_o + C_{22}H_o + C_{23}Q_o] [\text{bei } x_o + \frac{\text{ber}'x_o}{x_o}] - \frac{Et\ell Q_o}{x_o \ell^2 a^2} \}$$

$$M_\theta = \frac{\ell^2 ET}{a^2} \{[C_{11}M_o + C_{12}H_o + C_{13}Q_o] [(1-\nu) \frac{\text{ber}'x_o}{x_o} - \nu \text{ bei } x_o] + + [C_{21}M_o + C_{22}H_o + C_{23}Q_o] [(1-\nu) \frac{\text{bei}'x_o}{x_o} + \nu \text{ ber } x_o] \}. \quad \text{(A2-3)}$$

The C_{ij} factors are the same as are used in calculating the element flexibilities and expressions for these can be found in Appendix G.

2. Stress Resultants for a Conical Element

The stress resultants for a conical element can be found from M_i, M_j, H_i, H_j, N_j and $N_i = \bar{r} N_j$ from the following formulae

$$N_{s_k} = N_k + H_k \cos\alpha$$

$$M_{s_k} = M_k$$

$$Q_{s_i} = -H_i \sin\alpha$$

$$Q_{s_j} = H_j \sin\alpha$$

$$\begin{aligned} M_{\theta_k} = (2/y_k^2) \ \{ & [A_{11}H_k + A_{12}M_k + A_{13}H_1 + A_{14}M_1] \ [\nu y_k \mathrm{bei}'y_k + \\ & + 2(1-\nu) \ (\mathrm{bei}\ y_k + \frac{2\ \mathrm{ber}'y_k}{y_k})] - [A_{21}H_k + A_{22}M_k + \\ & + A_{23}H_1 + A_{24}M_1] \ [\nu y_k \mathrm{ber}'y_k + 2(1-\nu) \ (\mathrm{ber}\ y_k + \\ & - \frac{2\ \mathrm{bei}'y_k}{y_k})] + [B_{11}H_k + B_{12}M_k + B_{13}H_1 + B_{14}M_1]. \\ & [\nu y_k \mathrm{kei}'y_k + 2(1-\nu)(\mathrm{kei}\ y_k \frac{2\ \mathrm{kei}'y_k}{y_k})] - [B_{21}H_k + \\ & + B_{22}M_k + B_{23}H_1 + B_{24}M_1] \ [\nu y_k \mathrm{ker}'y_k + 2(1-\nu) \ (\mathrm{ker}\ y_k + - \frac{2\ \mathrm{kei}'y_k}{y_k})]\} \end{aligned} \qquad (A2\text{-}4)$$

$$\begin{aligned} N_{\theta_k} = -(\cot\alpha/2s_k) \ \{ & [A_{11}H_k + A_{12}M_k + A_{13}H_1 + A_{14}M_1] \ [y_k \mathrm{ber}'y_k + \\ & - 2(\mathrm{ber}\ y_k - \frac{2\ \mathrm{bei}'y_k}{y_k})] + [A_{21}H_k + A_{22}M_k + A_{23}H_1 + \\ & + A_{24}M_1][y_k \mathrm{bei}'y_k - 2(\mathrm{bei}\ y_k + \frac{2\ \mathrm{bei}'y_k}{y_k})] + [B_{11}H_k + \\ & + B_{12}M_k + B_{13}H_1 + B_{14}M_1] \ [y_k \mathrm{ker}'y_k - 2(\mathrm{ker}\ y_k - \frac{2\ \mathrm{kei}'y_k}{y_k})] + \\ & + [B_{21}H_k + B_{22}M_k + B_{23}H_1 + B_{24}M_1] \ [y_k \mathrm{kei}'y_k + \\ & - 2(\mathrm{kei}\ y_k + \frac{2\ \mathrm{ker}'y_k}{y_k})]\} + p_r \cot\alpha\ s_k, \end{aligned}$$

where p_r is the internal pressure and $k = i,j$. When $k = i$ then $l = j$ and vice versa. The A_{ij}, B_{ij} and y_k factors are the same as used in calculating the element flexibilities and expressions for these can be found in Appendix E.

C. Thomson Function Formulae

Depending upon the magnitude of the argument y, there are three different sets of formulae that are used to compute the Thomson functions [Reference (10)]. Computation procedures outlining the application of the different sets are given in Appendix D below.

1. Series Formulation

$$\text{ber } y = \sum_{m=0}^{\infty} \frac{(-1)^m}{[(2m)!]^2} \left(\frac{y}{2}\right)^{4m}$$

$$\text{bei } y = \sum_{m=0}^{\infty} \frac{(-1)^m}{[(2m+1)!]^2} \left(\frac{y}{2}\right)^{4m+2}$$

$$\text{ker } y = -\ln\left(\frac{y}{2}\right) \text{ber } y + \frac{\pi}{4} \text{bei } y + \sum_{m=0}^{\infty} \frac{(-1)^m}{[(2m)!]^2} \left(\frac{y}{2}\right)^{4m} \psi(2m+1)$$

$$\text{kei } y = -\ln\left(\frac{y}{2}\right) \text{bei } y - \frac{\pi}{4} \text{ber } y + \sum_{m=0}^{\infty} \frac{(-1)^m}{[(2m+1)!]^2} \left(\frac{y}{2}\right)^{4m+2} \psi(2m+2) \qquad \text{(A3-1)}$$

$$\text{ber}'y = \sum_{m=1}^{\infty} 2m \left(\frac{y}{2}\right)^{4m-1} \frac{(-1)^m}{[(2m)!]^2}$$

$$\text{bei}'y = \sum_{m=0}^{\infty} (2m+1) \left(\frac{y}{2}\right)^{4m+1} \frac{(-1)^m}{[(2m+1)!]^2}$$

$$\text{ker}'y = -\ln\left(\frac{y}{2}\right) \text{ber}'y - \frac{\text{ber } y}{y} + \frac{\pi}{4} \text{bei}'y + \sum_{m=1}^{\infty} 2m \left(\frac{y}{2}\right)^{4m-1} \frac{(-1)^m}{[(2m)!]^2} \psi(2m+1)$$

$$\text{kei}'y = -\ln\left(\frac{y}{2}\right) \text{bei}'y - \frac{\text{bei } y}{y} - \frac{\pi}{4} \text{ber}'y + \sum_{m=0}^{\infty} (2m+1) \left(\frac{y}{2}\right)^{4m+1} \frac{(-1)^m}{[(2m+1)!]^2} \psi(2m+2)$$

where ψ is the logarithmic derivative of the gamma function given by

$$\psi(m+1) = 1 + \frac{1}{2} + \frac{1}{3} + \ldots\ldots + \frac{1}{m} - \gamma$$

$$\psi(1) = -\gamma \qquad \text{(A3-2)}$$

γ = Euler's constant = 0.577215665....

2. Asymptotic Formulation

$$\text{ber } y = A(y)\ [\Lambda(y)\cos\alpha - \chi(y)\sin\alpha]$$

$$\text{bei } y = A(y)\ [\chi(y)\cos\alpha + \Lambda(y)\sin\alpha]$$

$$\text{ber}'y = A(y)\ [\phi(y)\cos\beta - \Omega(y)\sin\beta]$$

$$\text{bei}'y = A(y)\ [\Omega(y)\cos\beta + \phi(y)\sin\beta] \qquad \text{(A3-3)}$$

$$\text{ker } y = B(y)\ [\Lambda(-y)\cos\beta + \chi(-y)\sin\beta]$$

$$\text{kei } y = B(y)\ [\chi(-y)\cos\beta - \Lambda(-y)\sin\beta]$$

$$\text{ker}'y = -B(y)\ [\phi(-y)\cos\alpha + \Omega(-y)\sin\alpha]$$

$$\text{kei}'y = -B(y)\ [\Omega(-y)\cos\alpha - \phi(-y)\sin\alpha]$$

where

$$A(y) = e^{y/2^{\frac{1}{2}}}/(2\pi y)^{\frac{1}{2}} \quad ; \quad B(y) = (\pi/2y)^{\frac{1}{2}}\, e^{-y/2^{\frac{1}{2}}}$$

$$\alpha = y/2^{\frac{1}{2}} - \pi/8 \quad ; \quad \beta = y/2^{\frac{1}{2}} + \pi/8$$

$$\Lambda(y) = 1 + \sum_{c=1}^{N} \frac{(-1)^c(-1^2)(-3^2)\ldots[-(2c-1)^2]}{c!\;(8y)^c} \cos(\pi c/4)$$

$$\chi(y) = \sum_{c=1}^{N} \frac{(-1)^c(-1^2)(-3^2)\ldots[-(2c-1)^2]}{c!\;(8y)^c} \sin(\pi c/4) \qquad \text{(A3-4)}$$

$$\phi(y) = 1 + \sum_{c=1}^{N} \frac{(-1)^c(-1^2)(-3^2)\ldots[-(2c-3)^2](2c+1)(2c-1)}{c!\;(8y)^c} \cos(\pi c/4)$$

$$\Omega(y) = \sum_{c=1}^{N} \frac{(-1)^c(-1^2)(-3^2)\ldots[-(2c-3)^2](2c+1)(2c-1)}{c!\;(8y)^c} \sin(\pi c/4)$$

3. Modified Asymptotic Formulation

The quantities defined in Equations (A3-4) remain unchanged and the ker group function in Equations (A3-3) remain the same. However, the ber group function formulae are modified as follows

$$\text{ber } y = A(y)\,[\Lambda(y)\cos\alpha - \chi(y)\sin\alpha] - (\text{kei } y)/\pi$$

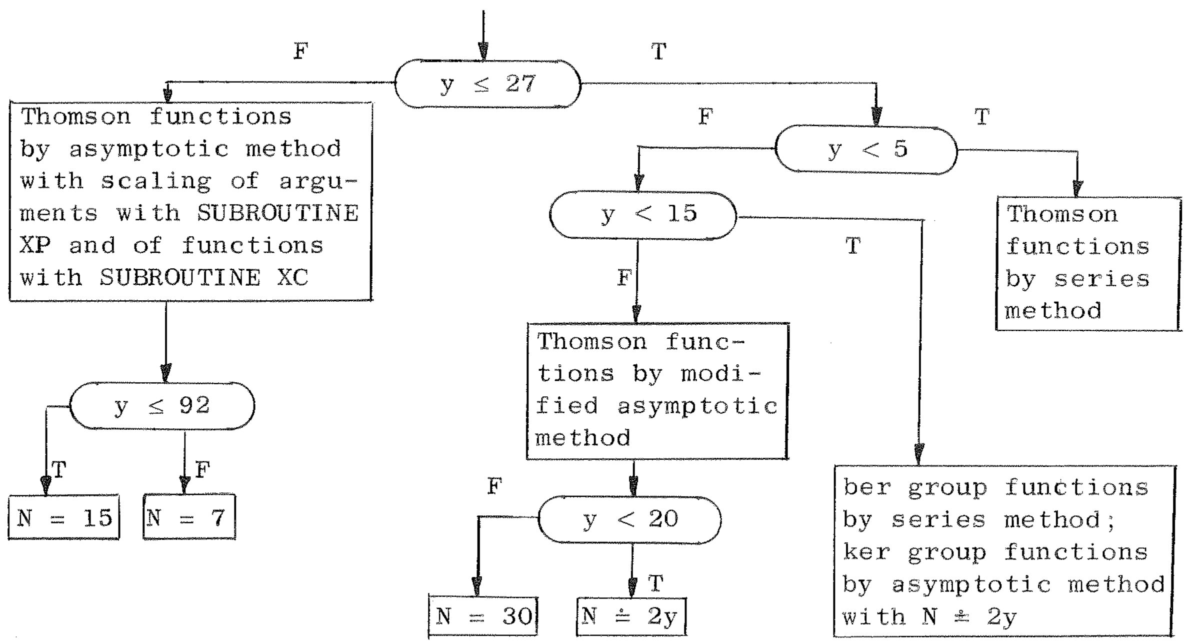
$$\text{bei } y = A(y)\,[\chi(y)\cos\alpha + \Lambda(y)\sin\alpha] + (\text{ker } y)/\pi$$

$$\text{ber}'y = A(y)\,[\phi(y)\cos\beta - \Omega(y)\sin\beta] - (\text{kei}'y)/\pi \qquad \text{(A3-5)}$$

$$\text{bei}'y = A(y)\,[\Omega(y)\cos\beta + \phi(y)\sin\beta] + (\text{ker}'y)/\pi$$

D. Thomson Function Computation Procedures

If y is the argument and N is the upper limit of summations in the asymptotic methods then



In the series method, calculations are continued until the magnitude of the calculated term is less than 10^{-8} times the current function value.

E. Conical Element Flexibility Formulae

Four fifths of the flexibility coefficients in Equation (13) can be expressed in terms of the functions

$$f = Cp_q \left[a_{lm} b_n(y_i) - a_{rs} b_t(y_i) + b_{lm} k_n(y_i) - b_{rs} k_t(y_i)\right]$$

$$= F(i;\ q;\ l,\ m,\ n;\ r,\ s,\ t) \qquad \text{(A5-1)}$$

$$f = Cp_q \left[a_{lm} b_n(y_i) + a_{rs} b_t(y_i) + b_{lm} k_n(y_i) + b_{rs} k_t(y_i)\right]$$

$$= G(i;\ q;\ l,\ m,\ n;\ r,\ s,\ t).$$

The individual flexibility coefficients are then given by the following relationships

$$f_{11} = -F(i;\ 1;\ 1,\ 2,\ 2;\ 2,\ 2,\ 1)$$

$$f_{12} = -F(i;\ 1;\ 1,\ 4,\ 2;\ 2,\ 4,\ 1)$$

$$f_{13} = -F(i;\ 1;\ 1,\ 1,\ 2;\ 2,\ 1,\ 1)$$

$$f_{14} = -F(i;\ 1;\ 1,\ 3,\ 2;\ 2,\ 3,\ 1)$$

$$f_{15} = (\bar{r}\ \cot\ \alpha)/(Et)$$

$$f_{21} = F(j;\ 1;\ 1,\ 2,\ 2;\ 2,\ 2,\ 1)$$

$$f_{22} = F(j;\ 1;\ 1,\ 4,\ 2;\ 2,\ 4,\ 1)$$

$$f_{23} = F(j;\ 1;\ 1,\ 1,\ 2;\ 2,\ 1,\ 1)$$

$$f_{24} = F(j;\ 1;\ 1,\ 3,\ 2;\ 2,\ 3,\ 1) \qquad \text{(A5-2)}$$

$$f_{25} = -(\cot\ \alpha)/(Et)$$

$$f_{31} = -G(i;\ 3;\ 1,\ 2,\ 5;\ 2,\ 2,\ 6)$$

$$f_{32} = -G(i;\ 3;\ 1,\ 4,\ 5;\ 2,\ 4,\ 6)$$

$$f_{33} = -G(i;\ 3;\ 1,\ 1,\ 5;\ 2,\ 1,\ 6)$$

$$f_{34} = -G(i;\ 3;\ 1,\ 3,\ 5;\ 2,\ 3,\ 6)$$

$$f_{35} = (\nu\ s_j\ \cos\ \alpha)/(Et)$$

$f_{41} = G(j;\ 3;\ 1,\ 2,\ 5;\ 2,\ 2,\ 6)$

$f_{42} = G(j;\ 3;\ 1,\ 4,\ 5;\ 2,\ 4,\ 6)$

$f_{43} = G(j;\ 3;\ 1,\ 1,\ 5;\ 2,\ 1,\ 6)$

$f_{44} = G(j;\ 3;\ 1,\ 3,\ 5;\ 2,\ 3,\ 6)$

$f_{45} = -(\nu\ s_j\ \cos\ \alpha)/(Et)$

$f_{51} = G(j;\ 2;\ 1,\ 2,\ 9;\ 2,\ 2,\ 10) - G(i;\ 2;\ 1,\ 2,\ 9;\ 2,\ 2,\ 10)$

$f_{52} = G(j;\ 2;\ 1,\ 4,\ 9;\ 2,\ 4,\ 10) - G(i;\ 2;\ 1,\ 4,\ 9;\ 2,\ 4,\ 10)$

$f_{53} = G(j;\ 2;\ 1,\ 1,\ 9;\ 2,\ 1,\ 10) - G(i;\ 2;\ 1,\ 1,\ 9;\ 2,\ 1,\ 10)$

$f_{54} = G(j;\ 2;\ 1,\ 3,\ 9;\ 2,\ 3,\ 10) - G(i;\ 2;\ 1,\ 3,\ 9;\ 2,\ 3,\ 10)$

$f_{55} = (s_j\ \ln\ \bar{r})/(Et)$

where

$$Cp_1 = \frac{2[3(1-\nu^2)]^{\frac{1}{2}}\ \cot\ \alpha}{Et^2}\ ;\quad Cp_2 = \frac{\cot\ \alpha}{Et}\ ;\quad Cp_3 = \frac{\cot\ \alpha\ \cos\ \alpha}{Et}$$

$$\begin{Bmatrix} a_{mn} \\ b_{mn} \end{Bmatrix} = \frac{H_{mn}}{\Delta} \begin{Bmatrix} d_{mn} \\ d_{(m+2)n} \end{Bmatrix} \quad \text{for } m = 1,2 \text{ and } n = 1,2,3,4 \qquad \text{(A5-3)}$$

$H_{21} = -H_{11} = s_i \sin\ \alpha;$ $\qquad H_{22} = -H_{21} = y_i^2/2$

$H_{23} = -H_{13} = s_j \sin\ \alpha;$ $\qquad H_{24} = -H_{14} = y_j^2/2$

and d_{mn} is the minor of item nm of the following determinate

$$\Delta = \begin{vmatrix} b_1(y_i) & b_2(y_i) & k_1(y_i) & k_2(y_i) \\ b_4(y_i) & -b_3(y_i) & k_4(y_i) & -k_3(y_i) \\ b_1(y_j) & b_2(y_j) & k_1(y_j) & k_2(y_j) \\ b_4(y_j) & -b_3(y_j) & k_4(y_j) & -k_3(y_j) \end{vmatrix} \qquad \text{(A5-4)}$$

and the remaining quantities are given by

$$\begin{aligned}
b_1(y) &= \text{ber } y - 2y^{-1}\text{bei}'y; \quad b_2(y) = \text{bei } y + 2y^{-1}\text{ber}'y \\
k_1(y) &= \text{ker } y - 2y^{-1}\text{kei}'y; \quad k_2(y) = \text{kei } y + 2y^{-1}\text{ker}'y \\
b_3(y) &= y\ \text{ber}'y - 2(1-\nu)b_1(y); \ b_4(y) = y\ \text{bei}'y - 2(1-\nu)b_2(y) \\
k_3(y) &= y\ \text{ker}'y - 2(1-\nu)k_1(y); \ k_4(y) = y\ \text{kei}'y - 1(1-\nu)k_2(y) \\
b_5(y) &= -\tfrac{1}{2}\ [y\ \text{ber}'y - 2(1+\nu)b_1(y)] \\
b_6(y) &= -\tfrac{1}{2}\ [y\ \text{bei}'y - 2(1+\nu)b_2(y)] \\
k_5(y) &= -\tfrac{1}{2}\ [y\ \text{ker}'y - 2(1+\nu)k_1(y)] \\
k_6(y) &= -\tfrac{1}{2}\ [y\ \text{kei}'y - 2(1+\nu)k_2(y)] \\
b_9(y) &= \text{ber } y - 2(1+\nu)\ y^{-1}\ \text{bei}'y \\
b_{10}(y) &= \text{bei } y + 2(1+\nu)\ y^{-1}\ \text{ber}'y \\
k_9(y) &= \text{ker } y - 2(1+\nu)\ y^{-1}\ \text{kei}'y \\
k_{10}(y) &= \text{kei } y + 2(1+\nu)\ y^{-1}\ \text{ker}'y.
\end{aligned} \qquad \text{(A5-5)}$$

The definitions of y and $\bar{r}$ are

$$\begin{aligned}
y &= 2(3(1-\nu^2))^{\frac{1}{4}}\ ((2\ \tan\alpha)/t)^{\frac{1}{2}}\ s^{\frac{1}{2}} \\
\bar{r} &= s_j/s_i.
\end{aligned} \qquad \text{(A5-6)}$$

F. Cylindrical Element Flexibility Formulae

Employing the following definitions

$$\kappa = (3(1-\nu^2)a^2/t^2)^{\frac{1}{4}} \; ; \qquad \lambda = \kappa \ell / a$$

$$Q = \kappa / a \; ; \qquad P = e^{\lambda} ; \qquad N = e^{-\lambda} \qquad \text{(A6-1)}$$

$$L = \cos\lambda + \sin\lambda \; ; \qquad M = \cos\lambda - \sin\lambda$$

the individual flexibility coefficients in Equation (14) for a cylindrical element may be written

$$f_{1k} = Q(C_{1k} - C_{2k} - C_{3k} - C_{4k}) \text{ for } k=1,2,3,4$$

$$f_{15} = 0$$

$$f_{2k} = Q(-NLC_{1k} + NMC_{2k} + PMC_{3k} + PLC_{4k}) \text{ for } k = 1,2,3,4$$

$$f_{25} = 0$$

$$f_{3k} = -C_{1k} - C_{3k} \text{ for } k = 1,2,3,4$$

$$f_{35} = (\nu a)/(Et) \qquad \text{(A6-2)}$$

$$f_{4k} = N(\cos\lambda C_{1k} + \sin\lambda C_{2k}) + P(\cos\lambda C_{3k} + \sin\lambda C_{4k}) \text{ for } k = 1,2,3,4$$

$$f_{45} = -(\nu a)/(Et)$$

$$f_{51} = f_{52} = 0 ; \qquad f_{55} = \ell/(Et)$$

$$f_{5k} = (\nu/2\kappa)\ (NM-1)C_{1k} + (NL-1)C_{2k} - (PL-1)C_{3k} + (PM-1)C_{4k} \text{ for } k = 3,4$$

Here the C_{mn} quantitites are given by

$$C_{mn} = (-1)^{m+n} \frac{D_{nm}}{\Delta} \frac{a^2}{2K^2} \quad \text{for } m = 1,2,3,4 \text{ and } n = 1,2$$

$$C_{mn} = (-1)^{m+n} \frac{D_{nm}}{\Delta} \frac{a^3}{2K^3} \quad \text{for } m = 1,2,3,4 \text{ and } n = 3,4 \qquad \text{(A6-3)}$$

where

$$\Delta = (2\sin\lambda + P - N)(2\sin\lambda - P + N)$$

$$D_{11} = \cos 2\lambda - \sin 2\lambda - P^2 ; \quad D_{21} = -L(P-N) - 2P\sin\lambda$$

$$D_{31} = P^2 - 1 - \sin 2\lambda ; \quad D_{41} = (P-N)\cos\lambda - 2P\sin\lambda$$

$$D_{12} = M^2 + 2\sin^2\lambda - P^2 ; \quad D_{22} = -(P-N)\cos\lambda - (P+N)\sin\lambda$$

$$D_{32} = 2\sin^2\lambda ; \quad D_{42} = -(P-N)\sin\lambda \tag{A6-4}$$

$$D_{13} = -N^2 + \sin 2\lambda + \cos 2\lambda ; \quad D_{23} = M(P-N) + 2N\sin\lambda$$

$$D_{33} = 1 - N^2 - \sin 2\lambda ; \quad D_{34} = (P-N)\cos\lambda - 2N\sin\lambda$$

$$D_{14} = N^2 - 2\sin^2\lambda - \sin 2\lambda - 1 ; \quad D_{24} = -(P-N)\cos\lambda - (P+N)\sin\lambda$$

$$D_{34} = 2\sin^2\lambda ; \quad D_{44} = -(P-N)\sin\lambda .$$

G. Spherical Cap Flexibility Formulae

For a spherical cap element the following parameters are employed

$$\ell = (at)^{\frac{1}{2}}/(12(1-\nu^2))^{\frac{1}{4}} ; \quad x_o = r_o/\ell ; \quad K = Et^3/12(1-\nu^2) . \tag{A7-1}$$

1. Case with Singularity

The individual elements of Equation (16) are given by

$$f_{1k} = (C_{1k}\text{ber}'x_o + C_{2k}\text{bei}'x_o)/\ell \quad \text{for } k = 1,2$$

$$f_{13} = f_{1k}\ (k=3) - (\ell r_o\ \text{kei}'x_o)/K$$

$$f_{2k} = (\ell/a)\ \{C_{1k}[x_o\text{ber}\ x_o - (1+\nu)\text{bei}'x_o] +$$

$$+ C_{2k}[x_o\text{bei}\ x_o + (1+\nu)\text{ber}'x_o]\} \quad \text{for } k = 1,2 \tag{A7-2}$$

$$f_{23} = f_{2k}(k=3) - \ell^3 r_o\ [x_o\text{kei}\ x_o + (1+\nu)\text{ker}'x_o + (1+\nu)/x_o]/aK$$

$$f_{3k} = \frac{\ell^2 x_o}{a^2} (1+\nu) \; [-C_{1k} \text{bei}' x_o + C_{2k} \text{ber}' x_o] +$$

$$- C_{1k} \text{ber } x_o - C_{2k} \text{bei } x_o \quad \text{for } k = 1, 2$$

$$f_{33} = f_{3k}(k=3) - \frac{\ell^2 r_o}{K} \left[\frac{\ell^2 x_o}{a^2} (1+\nu) \; (\text{ker}' x_o + \frac{1}{x_o}) - \text{kei } x_o \right]$$

where

$$C_{11} = - \frac{\ell^2}{\Delta K} \; \text{ber}' x_o$$

$$C_{12} = - \frac{a^3 x_o}{\Delta E t (a^2 + r_o^2)} \; [\text{ber } x_o - \frac{(1-\nu)}{x_o} \; \text{bei}' x_o]$$

$$C_{13} = \frac{1}{\Delta} \left\{ \frac{\ell^2 r_o}{K} [\text{ber } x_o \; \text{ker}' x_o - \text{ber}' x_o \; \text{ker } x_o - \frac{(1-\nu)}{x_o} \; \text{bei}' x_o \; \text{ker}' x_o \right.$$

$$\left. + \frac{(1-\nu)}{x_o} \; \text{ber}' x_o \; \text{ker}' x_o \;] + \frac{\ell^3 a^3}{K(a^2 + r_o^2)} [\text{ber } x_o - \frac{(1-\nu)}{x_o} \; \text{bei}' x_o \;] \right\}$$

(A7-3)

$$C_{21} = - \frac{\ell^2}{\Delta K} \; \text{bei}' x_o$$

$$C_{22} = - \frac{a^3 \; x_o}{\Delta E t (a^2 + r_o^2)} \; [\text{bei } x_o + \frac{(1-\nu)}{x_o} \; \text{ber}' x_o]$$

$$C_{23} = \frac{1}{\Delta} \left\{ \frac{\ell^2 r_o}{K} \; [\text{bei } x_o \; \text{ker}' x_o - \text{bei}' x_o \; \text{ker } x_o + \frac{(1-\nu)}{x_o} \; \text{ber}' x_o \; \text{ker}' x_o \right.$$

$$\left. + \frac{(1-\nu)}{x_o} \; \text{bei}' x_o \; \text{kei}' x_o \;] + \frac{\ell^3 \; a^2}{K(a^2 + r_o^2)} \; [\text{bei } x_o + \frac{(1-\nu)}{x_o} \; \text{ber}' x_o] \right\}$$

and

$$\Delta = \begin{vmatrix} - [\text{bei } x_o + \frac{(1-\nu)}{x_o} \; \text{ber}' x_o] & [\text{ber } x_o - \frac{(1-\nu}{x_o} \; \text{bei}' x_o \;] \\ \text{bei}' \; x_o & -\text{ber}' x_o \end{vmatrix} \tag{A7-4}$$

2. Case Without Singularity

The first 2x2 portion of the flexibility matrix in Equation (18) is identical with the singular case. The additional coefficients required are

$$f_{31} = \frac{\ell^2}{\Delta K}\left[\text{ber } x_o \text{ ber}' x_o + \text{bei } x_o \text{ bei}' x_o - \left(\frac{x_o^2 \ell^2}{a^2} + 1\right) \text{ber}' x_o\right] \tag{A7-5}$$

$$f_{32} = \frac{a^3 x_o}{Et(a^2+r_o^2)} \left\{ (\text{ber}^2 x_o + \text{bei}^2 x_o) + \left[\frac{x_o \ell^2}{a^2}(1-\nu) - \frac{(1-\nu)}{x_o}\right] \right.$$

$$[\text{ber } x_o \text{ bei}' x_o - \text{bei } x_o \text{ ber}' x_o] - \frac{\ell^2}{a^2}(1-\nu^2)\,[\text{ber}'^2 x_o +$$

$$\left. + \text{bei}'^2 x_o] - \left(\frac{x_o^2 \ell^2}{a^2} + 1\right)\left[\text{ber } x_o - \frac{(1-\nu)}{x_o} \text{bei}' x_o\right] \right\}.$$

H. Flat Plate Flexibility Formulae

1. Annular Ring Element with $\alpha = 0^o$

The individual flexibility coefficients in Equation (14) for a flat plate element with $\alpha = 0^o$ are given in terms of

$$\bar{r} = r_j/r_i \ ; \quad K = (Et^3)/(12(1-\nu^2)) \ ; \quad Z = r_j^2 - r_i^2$$

and are

$$f_{11} = \frac{r_i}{KZ}\left[\frac{r_i^2}{(1-\nu)} + \frac{r_j^2}{(1-\nu)}\right] ; \quad f_{12} = \bar{r}\, f_{21}$$

$$f_{13} = f_{14} = 0 ; \qquad f_{15} = \bar{r}\, f_{51}$$

$$f_{21} = \frac{-2 r_j r_i^2}{(1-\nu^2)\, KZ} ; \qquad f_{22} = \frac{r_j}{KZ}\left[\frac{r_j^2}{(1-\nu)} + \frac{r_i^2}{(1-\nu)}\right]$$

$$f_{23} = f_{24} = 0 ; \qquad f_{25} = f_{52}$$

$$f_{31} = f_{32} = f_{35} = 0; \quad f_{34} = \bar{r}\, f_{43}$$

$$f_{33} = \frac{r_i}{EtZ}\,[r_j^2 + r_i^2 + \nu Z]$$

$$f_{41} = f_{42} = f_{45} = 0; \quad f_{43} = -\frac{2 r_j\, r_i^2}{EtZ}$$

$$f_{44} = \frac{r_j}{EtZ}\,[r_j^2 + r_i^2 - \nu Z] \tag{A8-1}$$

$$f_{51} = \frac{r_i^2}{K}\left[\frac{r_i^2 \ln \bar{r}}{(1-\nu)Z} + \frac{1}{2(1+\nu)}\right]; \quad f_{52} = -\frac{r_j^2}{K}\left[\frac{r_i^2 \ln \bar{r}}{(1-\nu)Z} + \frac{1}{2(1+\nu)}\right]$$

$$f_{53} = f_{54} = 0; \quad f_{55} = \frac{r_j}{K}\left[\frac{(\ln \bar{r})^2\, r_j^2 r_i^2 (1+\nu)}{2Z(1-\nu)} + \frac{(3+\nu)Z}{8(1+\nu)}\right]$$

2. Annular Ring Element with $\alpha = 180^\circ$

All the non-zero elements from the case $\alpha = 0^\circ$ can be used to construct the coefficients for the case $\alpha = 180^\circ$. In all cases except element f_{55} the negatives of the above formulae for $\alpha = 0^\circ$ are to be used. The formulae for f_{55} are identical. In other words

$$f_{mn}^{(180^\circ)} = -f_{mn}^{(0^\circ)} \quad \text{for } m, n = 1,2,3,4,5 \text{ except } mn = 55$$

$$f_{55}^{(180^\circ)} = f_{55}^{(0^\circ)} . \tag{A8-2}$$

3. Disc Element, Singular Case

For a disc element, the individual flexibility coefficients in Equation (17) are

$$f_{11} = \frac{r_o}{K(1+\nu)}; \quad f_{12} = 0; \quad f_{13} = \frac{r_o^2}{2(1+\nu)K}$$

$$f_{21} = 0; \quad f_{22} = \frac{(1-\nu) r_o}{Et}; \quad f_{23} = 0 \tag{A8-3}$$

$$f_{31} = f_{13}; \quad f_{32} = 0; \quad f_{33} = \frac{(3+\nu) r_o^2}{8(1+\nu)K} .$$

I. Open-Ended Element Mass Formulae

Adopting the following notation

$$\ell = \left|(r_\gamma - r_i)/\cos\alpha\right| \quad ; \quad W = 2\pi t \ell m \quad ; \quad V = \ell \cos\alpha, \qquad \text{(A9-1)}$$

the submatrices of the mass matrix in local coordinates for an open-ended element may be written as follows

$$[m_{ii}] = W \begin{bmatrix} \left(\frac{r_i}{105} + \frac{V}{280}\right)\ell^2 & -\left(\frac{11r_i}{210} + \frac{V}{60}\right)\ell & 0 \\ -\left(\frac{11r_i}{210} + \frac{V}{60}\right)\ell & \left(\frac{13r_i + 3V}{35}\right) & 0 \\ 0 & 0 & \left(\frac{r_i}{3} + \frac{V}{12}\right) \end{bmatrix}$$

$$[m_{ij}] = [m_{ji}]^T = W \begin{bmatrix} -\left(\frac{r_i}{140} + \frac{V}{280}\right)\ell^2 & -\left(\frac{13r_i}{420} + \frac{V}{60}\right)\ell & 0 \\ \left(\frac{13r_i}{420} + \frac{V}{70}\right)\ell & 9\left(\frac{r_i}{70} + \frac{V}{140}\right) & 0 \\ 0 & 0 & \left(\frac{r_i}{6} + \frac{V}{12}\right) \end{bmatrix} \qquad \text{(A9-2)}$$

$$[m_{jj}] = W \begin{bmatrix} \left(\frac{r_i}{105} + \frac{V}{168}\right)\ell^2 & \left(\frac{11r_i}{210} + \frac{V}{28}\right)\ell & 0 \\ \left(\frac{11r_i}{210} + \frac{V}{28}\right)\ell & \left(\frac{13r_i}{35} + \frac{2V}{7}\right) & 0 \\ 0 & 0 & \left(\frac{r_i}{3} + \frac{V}{4}\right) \end{bmatrix} .$$

J. End Closure Element Mass Formulae

With the following notation

$$W_o = \pi t_o m_o \tag{A10-1}$$

the mass matrix for an end closure element may be written

$$[\bar{m}_o] = \begin{bmatrix} W_o r_o^4/12 & 0 & -W_o r_o^3/4 \\ 0 & W_o r_o^2/2 & 0 \\ -W_o r_o^3/4 & 0 & W_o r_o^2 \end{bmatrix} \tag{A10-2}$$