THE UNIVERSITY OF ALBERTA

SER84

FINITE ELEMENT PROGRAMS

FOR

FRAME ANALYSIS

BY

MOHAMED H. EL-ZANATY

AND

DAVID W. MURRAY

Department of Civil Engineering

University of Alberta

April, 1980

ACKNOWLEDGEMENTS

This report was produced in the Department of Civil Engineering at the University of Alberta. Funds provided by the Civil Engineering Department and the National Research Council of Canada is acknowledged with thanks.

TABLE OF CONTENTS

	Page
CHAPTER I INTRODUCTION	1
CHAPTER II DEVELOPMENT OF PROGRAMS	3
2.1 Introduction to Program Development	3
2.2 Program ESTANF	4
2.2.1 Logic Flow of ESTANF	4
2.3 Program NONELA	7
2.3.1 Logic Flow of NONELA	7
2.4 Program INPLAF	10
2.4.1 Logic Flow of INPLAF	10
2.5 Program PLAFIT	13
2.5.1 Logic Flow of PLAFIT	13
2.6 Program INSTAF	16
2.6.1 Logic Flow of INSTAF	16
CHAPTER III APPLICATION	17
3.1 Introduction	17
3.2 Buckling Analysis	17
3.3 Nonlinear Elastic Analysis	17
3.4 Inelastic Analysis	20
LIST OF REFERENCES	22
APPENDIX A USER'S MANUAL FOR ESTANF	23
APPENDIX B USER'S MANUAL FOR NONELA	34
APPENDIX C USER'S MANUAL FOR INPLAF AND PLAFIT	45
APPENDIX D USER'S MANUAL FOR INSTAF	55

Page

APPENDIX E PROGRAM DESCRIPTIONS AND LISTINGS 67

E.1 Program ESTANF 68

E.2 Program NONELA 98

E.3 Program INPLAF 122

E.4 Program PLAFIT 142

E.5 Program INSTAF 161

E.6 The Data Managing Package 203

E.7 Data Storage and Retrieval Package 210

E.8 Common Input Subroutines 214

E.9 Common Subroutines in the Equation Solving Package. . 222

E.10 Common Subroutines in Formulation and Output 225

APPENDIX F EXAMPLE INPUT AND OUTPUT 241

F.1 Buckling Analysis 242

F.2 Nonlinear Elastic Analysis 247

F.3 Incremental Elastic-Plastic Analysis 253

F.4 Iterative Elastic-Plastic Analysis 260

F.5 Inelastic Analysis 267

LIST OF FIGURES

FIGURE		PAGE
3-1	Example Frame	18
3-2	Load-Deformation Curves for Elastic Frame	19
3-3	Load-Deformation Curves for Inelastic Frame	21

CHAPTER I

INTRODUCTION

The collection of programs reported herein was developed during the program of study of M. EL-Zanaty at the University of Alberta, in order to investigate the behavior of frames. This report discusses the development of the computer programs presented throughout that study. It also gives the user's manuals, logic flows, and listings of all programs.

Five different computer programs (ESTANF, NONELA, INPLAF, PLAFIT, and INSTAF) are presented in this report. Program ESTANF performs an eigenvalue analysis (buckling problems). NONELA is a program to evaluate the load-deformation characteristics of frames with linearly elastic material.

Two computer programs (INPLAF and PLAFIT) have been developed to solve for the load-deformation characteristics of frames by plastic analysis. INPLAF performs an incremental plastic analysis while PLAFIT is based on an iterative technique. Each program has the option to perform first or second order analysis.

The program INSTAF has been developed to evaluate the load-deformation characteristics of inelastic structures. It has the provision of including residual stresses in the cross-sections and strain hardening of the material in the solution.

All programs reported herein are capable of analyzing unbraced as well as braced frames.

Details of formulation and typical results for a wide variety of problems are contained in Structural Engineering Report No. 83, "Inelastic Behavior of Multistory Steel Frames" by EL-Zanaty et al. (1980).

CHAPTER II

DEVELOPMENT OF PROGRAMS

2.1 Introduction to Program Development

All programs developed during this investigation handle only plane frame type problems with no limitations on the size. In general, the programs handle two sets of load one of which is constant while the other is variable. Source language is FORTRAN IV. Double precision is used throughout.

The equation solver used is skyline Cholesky decomposition. All programs use the equation solving package developed by Elwi and Murray (1977) with a few modifications. A type of dynamic array allocation proposed by McCormick (1975), is used. This simple data manager identifies an array by its FORTRAN name, its length, and a logical number which designates the common block to which the array belongs.

The MAIN program segment is composed of a number of comment statements; a number of statements where the lengths of the common blocks are specified and passed to the data manager; a call statement to the main executive routine MAINMG; and an END card.

The executive routine MAINMG calls all other subroutines which read the data, formulate and solve the problem, and output the final results.

2.2 Program ESTANF: Eigenvalue Stability Analysis of Frames

This program solves for the lowest eigenvalue and the associated eigenvector, of a frame, using inverse power iteration technique.

The program includes an automatic recording of the present variables and last converged-at solution for added flexibility in restarting the iterative process. A flag is used as a control parameter to specify if the present run is original or a restart one.

2.2.1 Logic Flow of ESTANF

The sequence of events coded into the main executive routine MAINMG of the program is as follows. Details of the required input are given in Appendix A.

1. Read problem control parameters (Sub:INPUT1).
2. If the problem is not a restart, go to 5.
3. Read from FILE1 all information stored.
4. Go to 19.
5. Read nodal geometry, member properties and connectivities (Sub: INPUT2)
6. Read external boundary conditions (Sub: BOUND).
7. Read load control parameters, load factors, and load variable flag (Sub: LOADCV).
8. Read nodal loads, and/or prescribed displacements (Sub: JLOAD1).
9. Read member loads and compute the corresponding nodal loads (Sub: MLOAD1).
10. Form the column heights array (Sub: COLHT).
11. Form the diagonal component addressing array (Sub: ADDRES).

12. Form and assemble the structure elastic stiffness matrix. Also form geometric element stiffness for unit axial forces and write the stiffnesses on FILE1. (Subs: STIFF, ELEMS, MODIFY, and ASSEMB).
13. Add external boundary conditions (Sub: BOUND1).
14. Use Cholesky decomposition to partition $[K_{ii}]$ and reduce the load partition $\{R_i\}$ (Sub: EQSBST).
15. Solve for nodal displacements (Sub: BKSB1).
16. Loop over number of load cases to find nodal displacements and member forces (Subs: DISPL2 and STRES2).
17. Read the inverse iteration control parameters and the assumed eigenvector (Sub: INPUTI).
18. Assemble the geometric stiffness matrix arising due to the variable set of axial forces into array B. Also assemble the constant geometric stiffness into the elastic stiffness matrix. This constant geometric stiffness is due to the constant set of external forces which develop a set of constant axial forces in the members (Sub: STIFF1).
19. Calculate the smallest eigenvalue and the corresponding eigenvector by inverse power iteration using Cholesky decomposition (Subs: EIGV and MULT1).
20. If eigenvalue is obtained without exceeding the maximum number of iterations specified, go to 22.
21. Store all information necessary for a restart of the problem on FILE1.
22. Return to MAIN.

The functions of subroutines INPUT1, INPUT2, BOUND, LOADCV, JLOAD1, MLOAD1, and INPUTI are described in steps 1, 5, 6, 7, 8, 9, and 17 respectively. Subroutines COLHT and ADDRES form part of the equation solving package and are common to all programs and are listed in Sect. E.9. Subroutines ELEMS, MODIFY, ASSEMB, and BOUND1 are common to programs ESTANF, INPLAF, and PLAFIT and are listed in Sect. E.10. Also, the data storage and retrieval package and the data manager are common to all programs and are listed in Sects. E.7 and E.6 respectively. The remaining subroutines of program ESTANF are listed in Sect. E.1.

2.3 Program NONELA: Nonlinear Elastic Analysis

This program solves for the load-deformation characteristics of frames using a Newton-Raphson method. The program is capable of analyzing braced and unbraced frames. The Re-start capability described in Sect. 2.2 is also used. A flag is used to indicate if the re-start is due to a new load increment or due to a default in convergence. Two sets of applied loads are used, one is constant while the other increases monotonically or both can vary independently from one another.

2.3.1 Logic Flow of NONELA

The sequence of events coded into the main executive subroutine, MAINMG, is as follows. Details of the required input are given in Appendix B.

1. Read the problem control parameters (Sub: INPUT1)
2. If the problem is not a re-start go to 10.
3. Read from FILE10 all the information stored from the previous step.
4. Assemble the two sets of applied loads into a new load vector using the load factors introduced at step 1.
5. Read from FILE9 the number of this load increment.
6. If the re-start is due to default in convergence at the previous load increment go to 8.
7. Read from FILE9 the displacement vector obtained from the previous load increment, then go to 18.
8. Read from FILE10 the displacement vector obtained from the last converged solution.

9. Go to 18.
10. Read nodal geometry, member properties and connectivities (Sub: INPUT2).
11. Read external boundary conditions (Sub: BOUND).
12. Read nodal loads in two sets (Sub: JLOAD2).
13. Read member loads and compute the corresponding nodal loads (Sub: MLOAD2).
14. Calculate the total loads on the frame.
15. Form the column heights array. (Sub: COLHT).
16. Form the diagonal component addressing array (Sub: ADDRES).
17. Evaluate the derivatives of the shape vector for each Gauss point in a beam element (Sub: SHAPE).
18. Form and assemble the tangent stiffness matrix for flexural elements. Also assemble resisting forces together with external forces to find the vector of unbalanced forces (Subs: STIFF, TR, TR1, and ASSEMB).
19. If the frame is unbraced, go to 21.
20. Form and assemble the tangent stiffness matrix for the bracing members and add the matrix to the structure stiffness matrix (Subs: STIFFB and ASSEMB).
21. Add external boundary conditions (Sub: BOUND2).
22. Check convergence in unbalanced forces and nodal displacements. If both vectors converge go to 28 (Sub: CONVER).
23. If the number of iterations exceeded the maximum number allowed, go to 28.

24. Solve for nodal displacements $\{\Delta r\}$ (Subs: EQSBST and BKSB1).
25. Output the incremental nodal displacements $\{\Delta r\}$ (Sub: DISPL1).
26. Output the total nodal displacements $\{r\}$ and evaluate convergence in displacement vector (Sub: DINCR).
27. Go to 18.
28. Write on FILE9 and FILE10 all the necessary information for re-start of the problem.
29. Return to MAIN.

2.4 Program INPLAF: Incremental Plastic Analysis of Frames

This program solves for the load-deformation characteristics of frames using an incremental step-by-step method described by EL-Zanaty et al. (1980). The program has the capability of performing first or second order elastic-plastic analysis. The input description is found in Appendix C, and the listing in Sect. E.3.

2.4.1 Logic Flow of INPLAF

The sequence of events coded into the main executive routine MAINMG of the program is as follows.

1. Read the problem control parameters (Sub: INPUT1).
2. Read nodal geometry, member properties and connectivities (Sub: INPUT2).
3. Read the external boundary conditions (Sub: BOUND).
4. Read nodal loads (Sub: JLOAD2).
5. Read member loads and compute the corresponding nodal loads, then add 4 and 5 (Sub: MLOAD2).
6. Form the column heights array (Sub: COLHT).
7. Form the diagonal component addressing array (Sub: ADDRES).
8. Compute the maximum number of hinges that can be formed at a joint without causing a joint mechanism (Sub: MNHING).
9. Form and assemble the structure tangent stiffness matrix. For a second order analysis include the geometric stiffness matrix (Subs: STIFFP, ELEMS, MODIFY, and ASSEMB).
10. Add the external boundary conditions (Sub: BOUND1).

11. If conducting a second order analysis store the structure stiffness matrix on FILE1.
12. Use Choleski decomposition to partition $[K_{ii}]$ and reduce the load partition $\{R_i\}$ (Sub: EQSBST).
13. If the determinant of the stiffness matrix is not negative, go to 18.
14. If performing first order analysis, go to 35.
15. Change the sign of the incremental load.
16. Retrieve the stiffness matrix from FILE1.
17. Decompose the stiffness matrix [K] and reduce the load vector $\{\Delta R\}$ (Sub: EQSBST).
18. Solve for incremental nodal displacements (Sub: BKSB1).
19. If the number of load cases is equal to one go to 23.
20. Set the number of load cases equal to one.
21. Output nodal displacements for constant load vector (Sub: DISPL1).
22. Output the member end forces (Sub: STRES1).
23. Output nodal displacements for incremental load vector (Sub: DISPL1).
24. Check if any nodal displacement due to the small increment in load is greater than 1×10^6. If it is go to 36, otherwise continue.
25. Output the incremental member end forces (Sub: STRES1).
26. If the analysis is first order, go to 33.
27. Form equivalent load vector taking sway forces into account (Sub: CORRB).
28. Retrieve the stiffness matrix from FILE1.

29. Decompose the matrix [K] and reduce the load vector $\{\Delta R\}$ (Sub: EQSBST).
30. Solve for nodal displacements (Sub: BKSB1).
31. Output nodal displacements corresponding to the equivalent load vector (Sub: DISPL1).
32. Output the member end forces (Sub: STRES1).
33. Search for the location of the next plastic hinge. Place a real hinge at this location. Then update load vector, nodal displacements and member end forces and output the results (Sub: HINGE).
34. Assume a new load increment $\{\Delta R\}$.
35. Go to 9.
36. Return to MAIN.

2.5 Program PLAFIT: Plastic Analysis of Frames Using An Iterative Technique

This program solves for the load-deformation curve of frames using an iterative method described by EL-Zanaty et al. (1980). A flag is used to specify if the analysis is a first or second order. The input description is found in Appendix C, and the listing in Sect. E.4.

2.5.1 Logic Flow of PLAFIT

The sequence of events coded into the main executive subroutine, MAINMG, of the program is as follows.

1. Read the problem control parameters (Sub: INPUT1)
2. Read nodal geometry, member properties and connectivities (Sub: INPUT2).
3. Read the external boundary conditions (Sub: BOUND).
4. Read nodal loads (Sub: JLOAD2).
5. Read member loads and compute the corresponding nodal loads, then assemble the result into the vector of nodal loads (Sub: MLOAD2).
6. Form the column heights array (Sub: COLHT).
7. Form the diagonal component addressing array (Sub: ADDRES).
8. Compute the maximum number of hinges that can be formed at each joint without causing a joint mechanism (Sub: MNHING).
9. Form and assemble the tangent stiffness matrix. Also modify the constant load vector by assembling an applied moment equal to the plastic moment capacity at the location where a hinge has formed (Subs: STIFFP, ELEMS, MODIFY, and ASSEMB).

10. Add the external boundary conditions (Sub:BOUND1).

11. Decompose the matrix [K] and reduce the load vector {R} (Sub: EQSBST).

12. Solve for nodal displacements (Sub: BKSB1).

13. Check the sign of the determinant of the stiffness matrix. If the sign is negative set a flag NV=1.

14. If conducting a first order analysis and NV=1, go to 29.

15. If the number of load cases is equal to 1, go to 17.

16. Output nodal displacements due to the constant load vector (this consists of the constant applied loads in addition to the plastic moment capacities acting as applied moments at the locations of hinge formation). Also output member end forces for the same set of loads (Sub: DISPL1 and Sub: STRES1).

17. Output nodal displacements and member end forces due to total variable applied loads (Sub: DISPL1 and Sub: STRES1).

18. Find the location of the new hinge and update the load vector, nodal displacements, and member end forces using a load factor λ (Sub: HINGE).

19. Check the convergence of λ. If it has converged insert a real hinge at the particular section where M has just attained M_{pc} and set a flag IT = 999. If convergence does not occur go to 23.

20. If the problem is first order analysis, go to 24.

21. If IT = 999, go to 24.

22. IT = IT +1.

23. go to 9.

24. Set the number of load sets equal to 2.
25. Number of hinges formed = No of hinges +1.
26. IT = 0
27. If NV=1, go to 29.
28. go to 9.
29. Return to MAIN.

2.6 Program INSTAF: Inelastic Stability Analysis of Frames

The program solves for the load-deformation characteristic of an inelastic frame. It has the provision of including residual stresses of the cross sections and/or the strain hardening of the material in the analysis. It has also the re-starting capability presented in Sect. 2.2.

2.6.1 Logic Flow of INSTAF

The sequence of events coded into the main executive routine, MAINMG, of the program is similar to that listed in Sect. 2.3.1, with a few changes that can be stated as follows.

1. At step 10, the cross-sectional residual strain distribution is given (Sub: INPUT2).
2. In formulating and assembling the tangent stiffness matrix for flexural elements (Step 18), each member is subdivided into 3 elements. A substructure scheme developed by Elwi and Murray (1977) is then used to eliminate the internal degrees of freedom and assemble only the member end degrees of freedom into the structure stiffness matrix.
3. A gradient test is performed at step 24 to determine if the stiffness matrix is still positive definite. If the sign of the determinant of the tangent stiffness matrix is negative the program stops. A re-start run is then performed with a decrease in the applied loads.

CHAPTER III

APPLICATION

3.1 Introduction

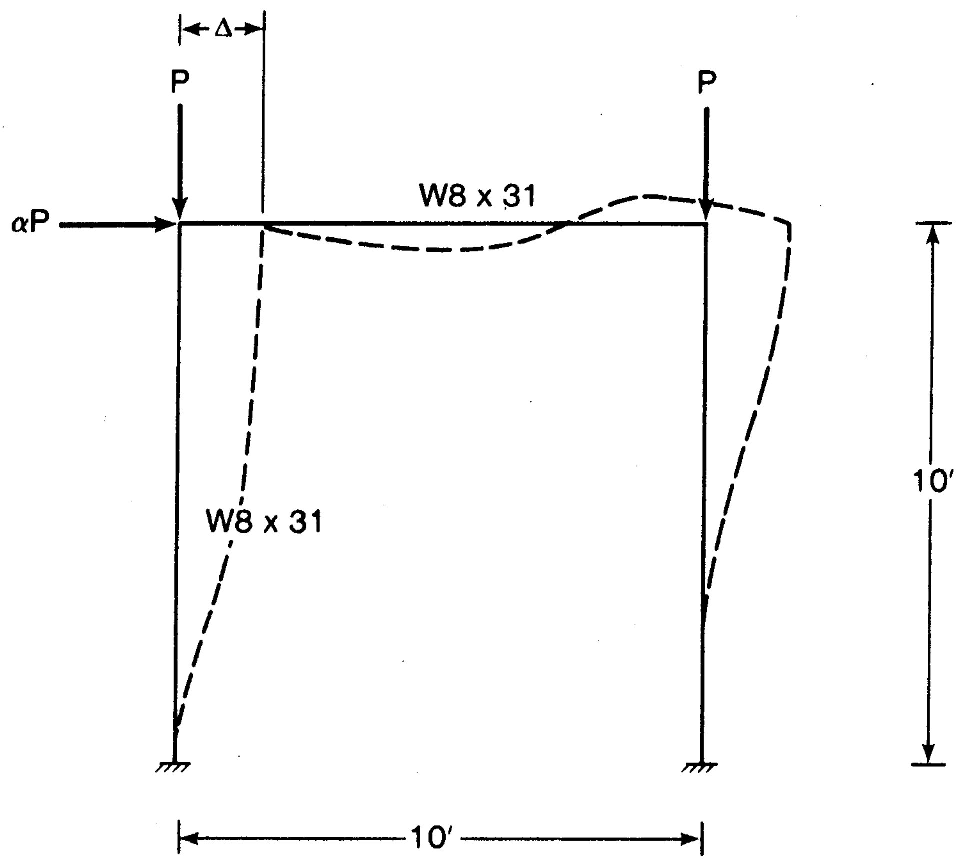
In this chapter an example problem is solved using the different programs developed. The structure chosen for illustration is a one bay-one story frame with fixed bases as shown in Fig. 3-1. The input file and sample output for each analysis performed are given in Appendix F.

3.2 Buckling Analysis

In this analysis each member was treated as one element. The classical solutions tabulated by Lu (1962) give a critical load of 1689.6 compared to 1696.67 obtained from the present analysis. The difference in results is about 0.42%. The input data file and the output are given in Sect. F.1.

3.3 Nonlinear Elastic Analysis

The results obtained for the example frame of Fig. 3-1 are shown in Fig. 3-2 for different values of α where α is a fraction of the vertical load. The input data file for α equal 0.5 and sample output for the problem are given in Sect. F.2.



$E = 30{,}000$ Ksi

$\sigma_y = 36$ Ksi

Figure 3-1. Example Frame

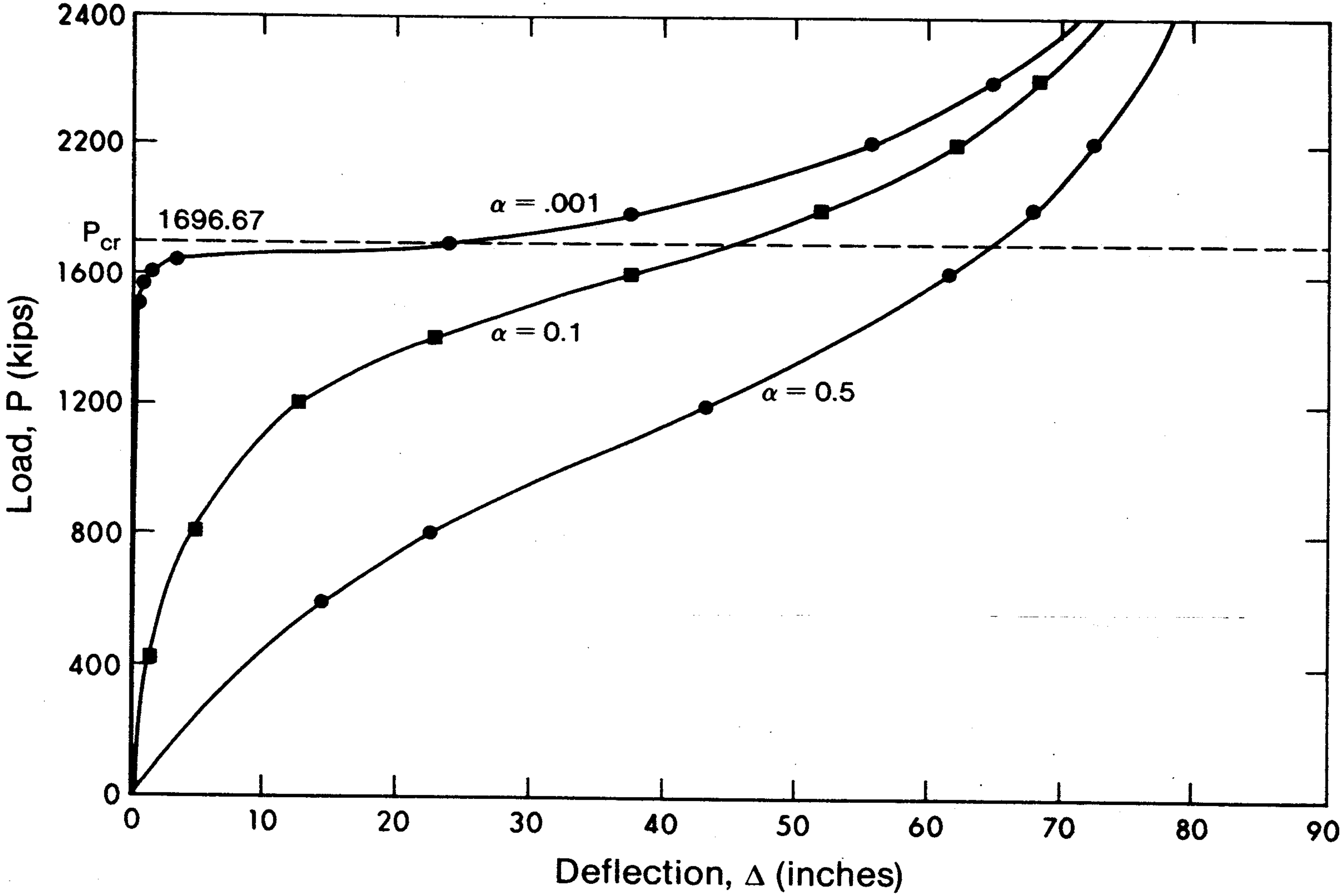


Figure 3-2. Load-Deformation Curves for Elastic Frame.

3.4 Inelastic Analysis

Figure 3-3 illustrates the results obtained for the example frame using the different plastic analyses. The load-deformation curves (1), (2) and (3) represent the inelastic analysis, the second order iterative elastic-plastic analysis, and the second order incremental elastic-plastic analysis, respectively. In the inelastic analysis (curve 1), the effects of residual stresses and strain hardening were not considered. The input files and sample outputs for the three cases are given in Sects. F.3, F.4, and F.5.

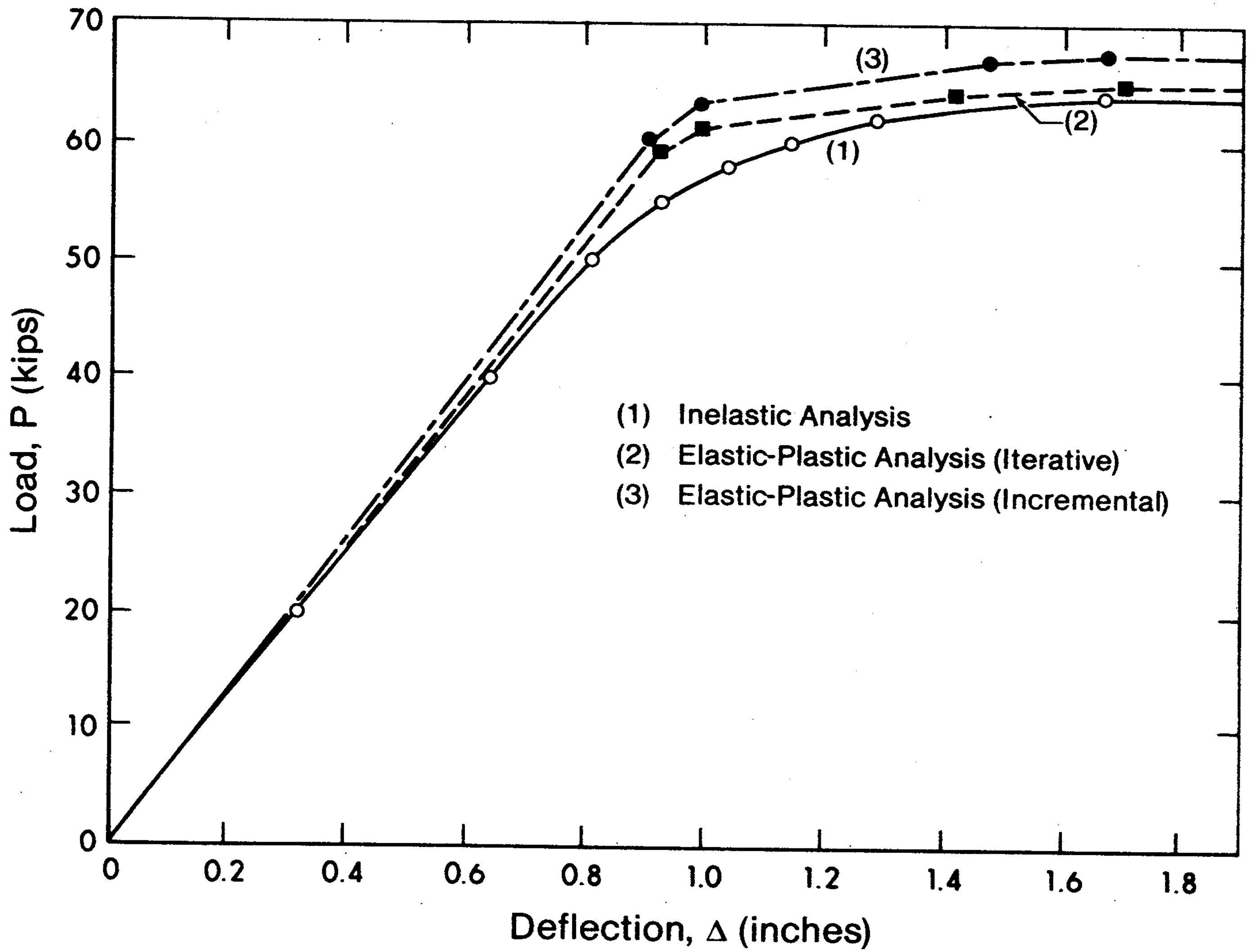


Figure 3-3. Load-Deformation Curves for Inelastic Frame.

LIST OF REFERENCES

1. Elwi, A.A., and Murray, D.W., (1974),
"Substructure Analysis of Plane Frames",
Structural Engineering Report No. 64,
Department of Civil Engineering, University of Alberta, June.

2. EL-Zanaty, M.H., Murray, D.W. and Bjorhovde, R., (1980),
"Inelastic Behavior of Multistory Steel Frames",
Structural Engineering Report No. 83,
Department of Civil Engineering, University of Alberta, April

3. Lu, L.W., (1962),
"A Survey of Literature on the Stability of Frames",
Welding Research Council, Bulletin 81, September.

4. McCormick, J.M., (1975),
"Data Handling in FORTRAN Programs",
Committee on Computers and Information Systems, American Society of Civil Engineers, National Structural Engineering Convention.

APPENDIX A

USER'S MANUAL FOR ESTANF

APPENDIX A

User's Manual For ESTANF

A.1 Problem Identification Cards

A.1.1 Heading Card (20A4)

One card which contains any title for the problem.

A.1.2 Re-start Identification Card (I5)

5
IREF

IREF: If 1, this is a re-start run (continuation to the previous run).

If 0, this is a new problem.

A.2 Problem Control Card (6I4, F12.0)

4	8	12	16	20	24	36
NJ	NE	NEBEL	NLC	NLT	IFEIG	UNIT

NJ : Number of joints.

NE : Number of elements

NEBEL: Number of external boundary elements

NLC : Number of load cases

NLT : Number of load types

IFEIG: Eigenproblem run flag,

If 0, this is a simple analysis

If 1, this is an eigenvalue analysis.

UNIT : Length conversion factor; e.g. 12.0 if section and material properties are entered in 'inch' units, while the nodal geometry is entered in 'foot' units.

A.3 Nodal Geometry Cards (I5, 2F12.0, I5)

One card/node, unless automatic generation of nodal data is used.

5	17	29	34
N	X(N)	Y(N)	INC

N : Node number.

X(N) : X-coordinate of the node.

Y(N) : Y-coordinate of the node.

INC : If non zero, automatic nodal generation will be initiated. The generated nodes will have numbers (NOLD + K* INC), where NOLD is the nodal number on the preceding card, and K is a positive integer. Generation terminates when (NOLD + K * INC) = N. The coordinates of the generated nodes are linearly interpolated between node NOLD and node N. INC should be positive.

Note: The first nodal geometry card must have INC=0, and the last must have N=NJ.

A.4 Member Data Cards

A.4.1 Member Property Default Card (3F12.0)

12	24	36
ADEF	RDEF	YDEF

ADEF : Default value for member area.

RDEF : Default value for moment of inertia of member.

YDEF : Default value for Young's modulus.

Note: If UNIT=12.0 on card A.2, ADEF, RDEF, and YDEF should be entered in in^2, in^4 and k/in^2, respectively. If UNIT=1.0 any consistent set of units may be used.

A.4.2 Member Property and Connectivity Cards (8I5, 3F12.0)

One card/member, unless automatic member data generation is used.

5	10	15	20	25	30	35	40	52	64	76
M	NOD(I,M)	NOD(J,M)	MKODE(M)	MFLAG(M)	INC	NOD1	NOD2	AREA (M)	RI (M)	YMOD (M)

M : Member number. The last member card must have M=NE.

NOD(I,M) : The node number at end I of member M.

NOD(J,M) : The node number at end J of member M.

MKODE(M) : 0, if member is continuous at both ends.
1, if member is hinged at end I only.
2, if member is hinged at end J only.
3, if member is hinged at both ends.

MFLAG(M) : If 0, the geometric stiffness matrix is not considered. If 1, the geometric stiffness matrix for member M is included in the analysis. (Note that for eigenvalue analysis MFLAG=1 for all columns).

INC : If nonzero, automatic generation of member data will be initiated. The new members will have numbers (MOLD + K* INC), where MOLD is the member number on the preceding card, and K is a positive integer. These members will have the cross section and material properties of the member on the card initiating the generation (i.e. the present card). The end nodes of the new members will be (NOD(I,MOLD) + K*NOD1) and (NOD(J,MOLD) + K*NOD2) for nodes at end I and J, respectively. INC must be zero for the first member card and zero or positive otherwise.

NOD1 : Incrementation value for nodal number of end I.

NOD2 : Incrementation value for nodal number of end J.

AREA(M) : Area of member section, if other than ADEF.

RI(M) : Moment of inertia of member section, if other than RDEF.

YMOD(M) : Young's modulus of member material, if other than YDEF.

Note: Units of AREA, RI, and YMOD are the same as for ADEF, RDEF, and YDEF, respectively.

A.4.3 <u>Echo Check Flag Card</u> (I5)

5
K

K : If 1, the complete nodal geometry and member data will be contained in the output.

If 0, the output will not contain an echo check of the completed data.

A.5 External Boundary Elements Cards (3I6, F12.0)

One card per external boundary element. A boundary element is defined as a spring with very high stiffness attached to a node. Each 'element' can restrain one or more of the degrees of freedom at a node. Several elements with different sitffnesses in different directions can be attached to one node.

6	12	18	30
N	NPB(N)	KODE(N)	BES(N)

N : Number of the external boundary element.

NPB(N) : Number of the node to which this element is attached.

KODE(N) : A three digit number of the form ijk, corresponding to degrees of freedom u, v, and r, respectively. If any of these is 1, the corresponding degree of freedom will be restrained, otherwise the digit must be 0.

BES(N) : The spring stiffness of this element, if other than 10^{20}.

A.6 Load Data Cards

A.6.1 Load Flag Card (2I5)

5	10
JLFLAG	MLFLAG

JLFLAG : If 0, no joint loads will be prescribed.
If 1, joint loads will be prescribed.

MLFLAG : If 0, no member loads will be prescribed.
If 1, member loads will be prescribed.

A.6.2 Load Identification Cards (2I5, F12.0)

One card for each load type

5	10	22
LT	LVF(LT)	FL(LT)

LT : Load type number (The first card must have LT = 1 and the last LT = NLT).

LVF(LT) : If 0, this load is a constant set of load.
If 1, the load is a variable set of load.

FL(LT) : The load factor corresponding to LT type of load.

A.6.3 Joint Load/Displacement Cards

This group consists of a group of cards for each load type. Each load type group consists of a number of cards per each loaded node. The first card in a node group is a nodal identification card. The rest of the cards for this node, each describes a case of loading.

A.6.3.1 Nodal Identification Card (6I5)

5	10	15	20	25	30
N	IDESP	NBEL(1)	NBEL(2)	NBEL(3)	INC

N : Node number (not necessarily in order)

IDESP : If non-zero, displacements will be specified

(NBEL(I), I = 1,3) : The boundary element numbers attached to this node, in case non-zero displacements are specified. These three numbers correspond to degrees of freedom u, v, and r, respectively. In case this node is restrained in 3 directions with one element, and non-zero displacements are specified only in 2 directions, then this boundary element number appears only twice in the corresponding locations.

INC : If non-zero, automatic loading generation will be initiated. Nodes with numbers (NOLD + K*INC) will be assigned the same loads as on the node (N) initiating the generation.

A.6.3.2 Nodal Load Cards (I5, 3F12.0)

5	17	29	41
JKODE	U	V	R

JKODE: A three digit number of the form ijk, the digits corresponding to u,v, and r, respectively. If any digit is 1, the corresponding u, v, or r will be interpreted as a displacement in the direction of the corresponding NBEL of card A.6.3.1, otherwise it must be zero.

U : Load in the X-direction

V : Load in the Y-direction

R : Moment in the X-Y plane, positive anti-clockwise.

Note: If UNIT=12, and YMOD is in K/in^2, U and V must be in kips and R in K-ft. Notice that the U, V, and R values are assembled directly into the load array B.

A.6.3.3 Termination Card

One blank card at the end of each load type of the 'joint load/ displacement card group'.

A.6.4 Member Load Specification Cards

This group consists of a number of cards for each load type. One card is assigned for each element on which non-zero loads are applied for this particular type. Load generation is carried out for one load type for one case of loading at a time. Load types are entered in order. However, in cases other than when automatic load generation is used, the cards for a specific load type may be entered in any order, since any one card defines completely the member number, the load, its position, and orientation with respect to the member.

A.6.4.1 Member Load Cards (3I5, 2F12.0, 4F10.0)

One card per member per load per load type, unless automatic load generation is used.

5	10	15	27	39	49	59	69	79
M	INC	K	CL	CN	QI	AI	QJ	AJ

M : Member number

INC : If non zero, automatic load generation will be initiated for load type LT. The load on the card initiating the generation, or its end effects, will be assigned to members with numbers (MOLD + L*INC), where MOLD is the member number on the preceding card, and L is a positive integer.

K	:	If 1, the load is concentrated.
		If 2, the load is uniformly distributed.
		If 3, the load is trapezoidal.
CL	:	Projection along the member of a vector in direction of load.
CN	:	Projection normal to the member of a vector in direction of load.
QI	:	Magnitude of load at distance (AI* length) from end I.
AI	:	Fraction of length from end I to beginning of load.
QJ	:	Magnitude of load at distance (AJ* length) from end I.
AJ	:	Fraction of length from end I to end of load.

Note: If the load is concentrated, QJ and AJ may be omitted. If the load is uniform, QJ may be omitted. Units of load are in K, or K/ft, if UNIT=12.0.

A.6.4.2 Termination Card

One blank card at the end of each group of member load cards assigned to a particular load type.

A.7 Inverse Power Iteration Cards

A.7.1 Eigen Problem Control Parameters Card (F12.6, 3I5)

12	17	22	27
TOL	ITLIM	IFPR	INFLAG

TOL : Tolerance limit; an arbitrarily small value which specifies desirable accuracy for convergence criterion.

ITLIM	:	Maximum number of iterations. The program stops when this number is reached, whether convergence is attained or not.
IFPR	:	If 0, only the final eigenvalue and the corresponding eigenvector are printed. If 1, the eigenvalue and the corresponding eigenvector obtained after each iteration are printed.
INFLAG	:	Input flag. If 0, an assumed eigenvector is generated consisting of a unit vector. If 1, the assumed eigenvector is read from the next sequence of cards.

A.7.2 <u>Assumed Eigenvector Cards</u> (8F10.0)

These cards are to be omitted if INFLAG=0

10	20	80
		VO(I), I=1,NN

{VO(I)}	:	An assumed eigenvector
NN	:	Number of equations of equilibrium = NJ*IDOF.

APPENDIX B

USER'S MANUAL FOR NONELA

APPENDIX B

User's Manual for NONELA

B.1 Problem Identification Cards

B.1.1 Heading Card (20A4)

One card which contains any title for the problem.

B.1.2 Re-start Control Card (2I5, 2F12.0)

5	10	22	34
IREF	IRCO	FL1	FL2

IREF : If 0, this is a new problem.

If 1, this is a re-start run corresponding to a new load increment.

IRCO : a correction run flag.

equal 0, if convergence is obtained in the previous load increment.

equal 1, if the previous run does not converge.

FL1,FL2 : Load factors 1 and 2 respectively.

B.2 Problem Control Card (6I4, 2F12.0)

4	8	12	16	20	24	36	48
NJ	NFE	NE	NEBEL	NMT	ITLIM	TOL	UNIT

NJ : Number of joints

NFE : Number of flexural elements (elements subjected to axial force and bending moment).

NE : Total number of elements (flexural elements and truss elements; truss elements are subjected to axial force only).

NEBEL: Number of external boundary elements.

NMT : Number of different member types.

1, if all the members are flexural elements (in this case NFE=NE)

2, if the two types of elements are present (in this case NE>NFE)

ITLIM: Maximum number of iterations. The program stops when this number is reached whether convergence is attained or not.

TOL : Tolerance limit; an arbitrarily small value which specifies described accuracy for convergence criterion.

UNIT : Length conversion factor; e.g. 12.0 if section and material properties are entered in 'inch' units, while the nodal geometry is entered in 'foot' units.

B.3 Nodal Geometry Cards (I5, 2F12.0, I5)

One card/node, unless automatic generation of nodal data is used.

5	17	29	34
N	X(N)	Y(N)	INC

N : Node number

X(N) : X-coordinate of the node.

Y(N) : Y-Coordinate of the node.

INC : If non-zero, automatic nodal generation will be initiated. The generated nodes will have numbers (NOLD + K* INC), where NOLD is the nodal number on the preceding card, and K is a positive integer. Generation terminates when (NOLD + K * INC) = N. The coordinates of the generated nodes are linearly interpolated between node NOLD and node N. INC should be positive.

Note: The first nodal geometry card must have INC=0, and the last must have N=NJ.

B.4 Member Data Cards

B.4.1 Member Property Default Card (4F12.0)

12	24	36	48
ADEF	RDEF	A4MDEF	YDEF

ADEF : Default value for member area

RDEF : Default value for moment of inertia of member

A4MDEF : Default value for forth moment of area and can be calculated as $\int_A y^4 dA$.

YDEF : Default value for Young's modulus.

Note: If UNIT=12.0 on card B.2, ADEF, RDEF, A4MDEF, and YDEF should be entered in in^2, in^4, in^6 and k/in^2, respectively. If UNIT=1.0 any consistent set of units may be used.

B.4.2 Member Property and Connectivity Cards (6I4, 4F12.0)

One card per member, unless automatic member data generation is used.

4	8	12	16	20	24	36	48	60	72
M	NOD(I,M)	NOD(J,M)	INC	NOD1	NOD2	AREA(M)	RI(M)	FMA(M)	YMOD(M)

M : Member number; the last member card must have M=NE.

NOD(I,M) : The node number at end I of member M.

NOD(J,M) : The node number at end J of member M.

INC : If non-zero automatic generation of member data will be initiated. The new members will have numbers (MOLD + K * INC), where MOLD is the member number on the preceding card, and K is a positive integer. These members will have the cross section and material properties of the member on the card initiating the generation (i.e. the present card). The end nodes of the new members will be (NOD(I,MOLD) + K * NOD1) and (NOD(J,MOLD) + K * NOD2) for nodes at end I and J, respectively. INC must be zero for the first member card and zero or positive otherwise.

NOD1 : Incrementation value for nodal number of end I.

NOD2 : Incrementation value for nodal number of end J.

AREA(M) : Area of member section, if other than ADEF

RI(M) : Moment of inertia of member section, if other than RDEF

FMA(M) : Forth moment of area of member section, if other than A4MDEF.

YMOD(M) : Young's modulus of member material, if other than YDEF.

Note: Units of AREA, RI, FMA, and YMOD are the same as for ADEF, RDEF, A4MDEF, and YDEF, respectively.

B.4.3 <u>Echo Check Flag Card</u> (I5)

5
K

K : If 1, the complete nodal geometry and member data will be contained in the output.
If 0, the output will not contain an echo check of the completed data.

B.5 <u>External Boundary Elements Cards</u> (3I6, F12.0)

One card per external boundary element. The definition of a boundary element is given in Sect. A.5.

6	12	18	30
N	NPB(N)	KODE(N)	BES(N)

N : Number of the external boundary element.

NPB(N) : Number of the node to which this element is attached.

KODE(N) : A five digit number of the form ijklm, corresponding to degrees of freedom u, v, r, $\frac{\partial u}{\partial x}$, and $\frac{\partial v}{\partial y}$, respectively. If any of these is 1, the corresponding degree of freedom will be restrained, otherwise the digit must be 0. Notice that the degree of freedom corresponding to strain perpendicular to the member should be restrained.

BES(N) : The spring stiffness of this element, if other than 10^{20}.

B.6 Load Data Cards

B.6.1 Load Flag Card (2I5)

5	10
JLFLAG	MLFLAG

JLFLAG : If 0, no joint loads will be prescribed.
If 1, joint loads will be prescribed.

MLFLAG : If 0, no member loads will be prescribed
If 1, member loads will be prescribed.

B.6.2 Joint Load/Displacement Cards

This group consists of a number of cards per loaded node, unless automatic load generation is used. The first card in a nodal load group is a node identification card. The rest of the cards for this node, each describes a case of loading. If JLFLAG=0 on card B.6.1, this group is omitted.

B.6.2.1 Nodal Identification Card (6I5)

One card at the head of a nodal load group.

5	10	15	20	25	30
N	IDESP	NBEL(1)	NBEL(2)	NBEL(3)	INC

N : Node number (not necessarily in order).

IDESP : If non-zero, displacements will be specified.

(NBEL(I), I=1,3)	:	The boundary element numbers attached to this node, in case non-zero displacements are specified. These three numbers correspond to degrees of freedom u, v, and r, respectively. In case this node is restrained in 3 directions with one element, and non-zero displacements are specified only in 2 directions, then this boundary element number appears only twice in the corresponding locations.
INC	:	If non-zero, automatic loading generation will be initiated. Nodes with numbers (NOLD + K * INC) will be assigned the same loads as on the node N initiating the generation.

B.6.2.2 Nodal Load Cards (2I5, 3F12.0)

5	10	22	34	46
JKODE	LVF	U	V	R

JKODE: A three digit number of the form ijk, the digits correspond to u, v, and r, respectively. If any digit is 1, the corresponding u, v, or r will be interpreted as a displacement in the direction of the corresponding NBEL of card B.6.2.1, otherwise it must be zero.

LVF : A flag indicating the variability of load. It could be either 0, or 1. Thus, two different sets of loads can be applied to the structure. Each set is increased independently using load factors FL1 and FL2 introduced in Sect. B.1.2. Notice that one of the sets can be kept constant by setting the corresponding load factor equal to 1.0.

U : Load in the X-direction.

V : Load in the Y-direction

R : Moment in the X-Y plane, positive anti-clockwise.

Note: If UNIT=12.0, and YMOD is in K/in^2, U and V must be in kips and R in k-ft. Notice that U, V, and R values are assembled directly into the load array B.

B.6.2.3 Termination Card

One blank card at the end of the joint load/displacement card group.

B.6.3 Member Load Specification Cards

This group consists of a number of cards for each member on which non-zero loads are applied, unless automatic load generation is used. Two sets of load can be applied to the structure one is kept constant and the other is increased monotonically or both can be increased monotonically. Load generation is carried out for one load for one case of loading at a time. In cases other than when automatic load generation is used, the cards may be entered in any order, since any one card defines completely the member number, the case of loading, the load, and its type, position, and orientation with respect to the member. If MLFLAG=0 on card B.6.1 this group is omitted.

B.6.3.1 <u>Member Load Cards</u> (4I4, 2F12.0, 4F10.0)

One card per member per load per load type, unless automatic load generation is used.

4	8	12	16	28	40	50	60	70	80
M	INC	LVF	K	CL	CN	QI	AI	QJ	AJ

M : Member number

INC : If non-zero, automatic load generation will be initiated. The load on the card, inititing the generation, or its end effects, will be assigned to members with numbers (MOLD + L * INC), where MOLD is the member number on the preceding card, and L is a positive integer.

LVF : A load variability flag defined in Sect. B.6.2.2

K : If 1, the Load is concentrated.
If 2, the load is uniformly distributed.
If 3, the load is trapezoidal.

CL : Projection along the member of a vector in direction of load.

CN : Projection normal to the member of a vector in direction of load.

QI : Magnitude of load at distance (AI* length) from end I.

AI : Fraction of length from end I to beginning of load.

QJ : Magnitude of load at distance (AJ* length) from end I.

AJ : Fraction of length from end I to end of load.

Note: If the load is concentrated, QJ and AJ may be omitted. If the load is uniform, QJ may be omitted. Units of load are in K, or K/ft, if UNIT=12.0.

B.6.3.2 Termination Card

One blank card at the end of the group of member load cards.

APPENDIX C

USER'S MANUAL FOR INPLAF AND PLAFIT

APPENDIX C

User's Manual for INPLAF and PLAFIT

C.1 Problem Identification and Control Cards

C.1.1 Heading Card (20A4)

One card which contains any title for the problem.

C.1.2 Problem Control Card (6I4, F12.0)

4	8	12	16	20	24	36
NJ	NE	NEBEL	NLC	ITAF	MNHE	UNIT

NJ : Number of joints.

NE : Number of elements.

NEBEL: Number of external boundary elements.

NLC : 1, if one set of load is applied to the frame
2, if two different sets of loads are applied to the frame.

ITAF: If 0, a first order analysis is produced.
If 1, a second order analysis is produced.

MNHE : An estimated maximum number of hinges that may be formed to achieve a mechanism. This number is just a limit which will cause the program to stop if it is exceeded.

UNIT : Length conversion factor (see Sect. A.2)

C.2 Nodal Geometry Cards (I5, 2F12.0, I5)

One card per node, unless automatic generation of nodal data is used.

5	17	29	34
N	X(N)	Y(N)	INC

N : Node number

X(N) : X-coordinate of the node

Y(N) : Y-coordinate of the node.

INC : If non-zero, automatic nodal generation will be initiated. The generated nodes will have numbers (NOLD + K * INC), where NOLD is the nodal number on the preceding card, and K is a positive integer. Generation terminates when (NOLD + K * INC) = N. The coordinates of the generated nodes are linearly interpolated between node NOLD and node N. INC should be positive.

Note: The first nodal geometry card must have INC=0, and the last must have N=NJ.

C.3 Member Data Cards

C.3.1 Member Property Default Card (5F12.0)

12	24	36	48	60
ADEF	RDEF	SMDEF	YSDEF	YDEF

ADEF : Default value for member cross sectional area.

RDEF : Default value for moment of inertia of member cross section.

SMDEF : Default value for plastic section modulus.

YSDEF : Default value for yield stress of the member material.

YDEF : Default value for Young's modulus.

Note: If UNIT=12.0 on card C.1.2, then ADEF, RDEF, SMDEF, YSDEF, and YDEF should be entered in in^2, in^4, in^3, k/in^2 and k/in^2, respectively. If UNIT=1.0 any consistent set of units may be used.

C.3.2 Member Property and Connectivity Cards

Two cards are provided for each member, unless automatic data generation is used. The first card in a member group describes the member end conditions and the connectivity while the second card specifies the member properties.

C.3.2.1 Member Connectivity Card (8I4)

4	8	12	16	20	24	28	32
M	NOD(I,M)	NOD(J,M)	MKODE(M)	MFLAG(M)	INC	NOD1	NOD2

M : Member number. The last member card must have M=NE.

NOD(I,M) : The node number at end I of member M.

NOD(J,M) : The node number at end J of member M.

MKODE(M) : 0, if member is continuous at both ends.
1, if member is hinged at end I only
2, if member is hinged at end J only
3, if member is hinged at both ends.

MFLAG(M) : If 0, the geometric stiffness matrix is not considered.
If 1, the geometric stiffness matrix for member M is included in the analysis.

INC : If non-zero, automatic generation of member data will be initiated. The new members will have numbers

(MOLD + K * INC), where MOLD is the member number on the preceding card, and K is a positive integer. These members will have the cross section and material properties of the member on the card initiating the generation. The end nodes of the new members will be (NOD(I,MOLD) + K * NOD1) AND (NOD(J,MOLD) + K * NOD2) for nodes at end I and J, respectively. INC must be zero for the first member card and zero or positive otherwise.

NOD1 : Incrementation value for nodal number of end I.

NOD2 : Incrementation value for nodal number of end J.

C.3.2.2 Member Property Card (5F12.0)

12	24	36	48	60
AREA(M)	RI(M)	PSM(M)	YS(M)	YMOD(M)

AREA(M) : Area of member section, if other than ADEF.

RI(M) : Moment of inertia of member section, if other than RDEF.

PSM(M) : Plastic section modulus of member section, if other than SMDEF.

YS(M) : Yield stress of the material of member, if other than YSDEF.

YMOD(M) : Young's modulus of member material, if other than YDEF.

C.3.3 Echo Check Flag Card (I5)

5
K

K : If 1, the complete nodal geometry and member data will be contained in the output.

If 0, the output will not contain an echo check for the completed data.

C.4 External Boundary Elements Cards (3I6, F12.0)

One card per external boundary element.

6	12	18	30
N	NPB(N)	KODE(N)	BES(N)

N : Number of the external boundary element.

NPB(N) : Number of the node to which this element is attached.

KODE(N) : A three digit number of the form ijk, corresponding to degrees of freedom u, v, and r, respectively. If any of these is 1, the corresponding degree of freedom will be restrained, otherwise the digit must be zero.

BES(N) : The spring stiffness of this element, if other than 10^{20}.

C.5 Load Data Cards

C.5.1 Load Flag Card (2I5)

5	10
JLFLAG	MLFLAG

JLFLAG : If 0, no joint loads will be prescribed.

If 1, joint loads will be prescribed.

MLFLAG : If 0, no member loads will be prescribed

If 1, member loads will be prescribed.

C.5.2 Joint Load/Displacement Cards

This group consists of a number of cards per loaded node, unless automatic load generation is used. The first card in a nodal load group is a node identification card. The rest of the cards for this node, each describes a case of loading. If JLFLAG=0 on card C.5.1 this group is omitted.

C.5.2.1 Nodal Identification Card (6I5)

One card at the head of a nodal load group.

5	10	15	20	25	30
N	IDESP	NBEL(1)	NBEL(2)	NBEL(3)	INC

N : Node number (not necessarily in order).

IDESP : If non-zero, displacements will be specified.

(NBEL (I), I=1,3) : The boundary element numbers attached to this node, in case non-zero displacements are specified. These three numbers correspond to degrees of freedom u, v, and r, respectively. In case this node is restrained in 3 directions with one element, and non-zero displacements are specified only in 2 directions, then this boundary element number appears only twice in the corresponding locations.

INC : If non-zero, automatic loading generation will be initiated. Nodes with numbers (NOLD + K * INC) will be assigned the same loads as on the node N initiating the generation.

C.5.2.2 <u>Nodal Load Cards</u> (2I5, 3F12.0)

5	10	22	34	46
JKODE	LVF	U	V	R

JKODE: A three digit number of the form ijk, the digits correspond to u, v, and r, respectively. If any digit is 1, the corresponding u, v, or r will be interpreted as a displacement in the direction of the corresponding NBEL of card C.5.2.1, otherwise it must be zero.

LVF : A flag indicating the variability of load. It could be either 0, or 1. Thus, two different sets of loads can be applied to the structure. One of the sets is kept constant while the other is increased monotonically.

U : Load in the X-direction.

V : Load in the Y-direction.

R : Moment in the X-Y plane, positive anti-clockwise

<u>Note</u>: If UNIT=12.0, and YMOD is in K/in^2, U and V must be in kips and R in K-ft. Notice that the U, V, and R values are assembled directly into the load array B.

C.5.2.3 <u>Termination Card</u>

One blank card at the end of the joint load/displacement card group.

C.5.3 Member Load Specification Cards

This group consists of a number of cards for each member on which non-zero loads are applied, unless automatic load generation is used. Two sets of load can be applied to the structure, one is kept constant and the other is increased monotonically. Load generation is carried out for one load for one case of loading at a time. In cases other than when automatic load generation is used, the cards may be entered in any order, since any one card defines completely the member number, the case of loading, the load, and its type, position, and orientation with respect to the member. If MLFLAG=0 on card C.5.1 this group is omitted.

C.5.3.1 Member Load Cards (4I4, 2F12.0, 4F10.0)

One card per member per load per load type, unless automatic load generation is used.

4	8	12	16	28	40	50	60	70	80
M	INC	LVF	K	CL	CN	QI	AI	QJ	AJ

M : Member number.

INC : If non-zero, automatic load generation will be initiated. The load on the card initiating the generation, or its end effects, will be assigned to members with numbers (MOLD + L * INC), where MOLD is the member number on the preceding card, and L is a positive integer.

LVF : A load variability flag defined in Sect. C.5.2.2

K : If 1, the load is concentrated.
If 2, the load is uniformly distributed.
If 3, the load is trapezoidal.

CL : Projection along the member of a vector in direction of load.

CN : Projection normal to the member of a vector in direction of load.

QI : Magnitude of load at distance (AI * length) from end I.

AI : Fraction of length from end I to beginning of load.

QJ : Magnitude of load at distance (AJ * length) from end I.

AJ : Fraction of length from end I to end of load.

<u>Note</u>: If the load is concentrated, QJ and AJ may be omitted. If the load is uniform, QJ may be omitted. Units of load are in K, or K/ft, if UNIT=12.0.

C.5.3.2 <u>Termination Card</u>

One blank card at the end of each group of member load cards assigned to a particular load type.

APPENDIX D

USER'S MANUAL FOR INSTAF

APPENDIX D

User's Manual for INSTAF

D.1 Problem Identification Cards

D.1.1 Heading Card (20A4)

One card which contains a title for the problem.

D.1.2 Re-start Control Card (3I5, 2F12.0)

5	10	15	27	39
IREF	IRCO	IC	FL1	FL2

IREF : If 0, this is a new problem.

If 1, this is a re-start run corresponding to a new load increment.

IRCO : a correction run flag.

0, if convergence is obtained in the previous load increment.

1, if the previous run does not converge.

IC : 0, at the start of the problem and also at each new load increment as long as ILL=1 from the last output.

1, if ILL=999 in the last output (first time only).

2, thereafter

FL1,FL2 : Load factors 1 and 2, respectively.

D.2 Problem Control Card (6I4, 2F12.0)

4	8	12	16	20	24	36	48
NJ	NFE	NE	NEBEL	NMT	ITLIM	TOL	UNIT

NJ : Number of joints.

NFE : Number of flexural elements (elements subjected to axial force and bending moment).

NE : Total number of elements (flexural elements and truss elements; truss elements are subjected to axial force only).

NEBEL: Number of external boundary elements.

NMT : Number of different member types.

1, if all the members are flexural elements (in this case NFE=NE)

2, if the two types of elements are present (in this case NE>NFE)

ITLIM: Maximum number of iterations. The program stops when this number is reached.

TOL : Tolerance limit; an arbitrarily small value which specifies described accuracy for convergence criterion.

UNIT : Length conversion factor; e.g. 12.0 if section and material properties are entered in 'inch' units, while the nodal geometry is entered in 'foot' units.

D.3 Number of Different Variable Cards (3I5)

5	10	15
NDXST	NDMT	NDRT

NDXST : Number of different types of cross-section.

NDMT : Number of different types of material property.

NDRT : Number of different types of residual stress distributions.

D.4 Nodal Geometry Cards (I5, 2F12.0, I5)

One card per node, unless automatic generation of nodal data is used.

5	17	29	34
N	X(N)	Y(N)	INC

N : Node number

X(N) : X-coordinate of the node.

Y(N) : Y-coordinate of the node.

INC : If non-zero, automatic nodal generation will be initiated. The generated nodes will have numbers (NOLD + K * INC), where NOLD is the nodal number on the preceding card, and K is a positive integer. Generation terminates when (NOLD + K * INC) = N. The coordinates of the generated nodes are linearly interpolated between node NOLD and node N. INC should be positive.

D.5 Member Data Cards

D.5.1 Member Cross Section Geometry Cards (I5, 4F12.0)

One card per each different type of cross section of members (There are as many cards as NDXST).

5	17	29	41	53
NN	H(NN)	W(NN)	TF(NN)	TW(NN)

NN : Type number of the member cross section.

H(NN) : Total height of the cross section (I-beam).

W(NN) : Width of the flange

TF(NN) : Flang thickness of the cross section

TW(NN) : Web thickness of the cross section

D.5.2 Member Material Property Cards (I5, 6F12.0)

One card per each different type of material property (There are as many cards as NDMT).

5	17	29	41	53	65	77
MN	EY(MN)	EP(MN)	EULT(MN)	YS(MN)	YPS(MN)	ULTS(MN)

MN : Type number of member material.

EY(MN) : The yield strain (ε_y) of the material.

EP(MN) : Plastic strain (ε_p) of the material (usually called strain hardening strain).

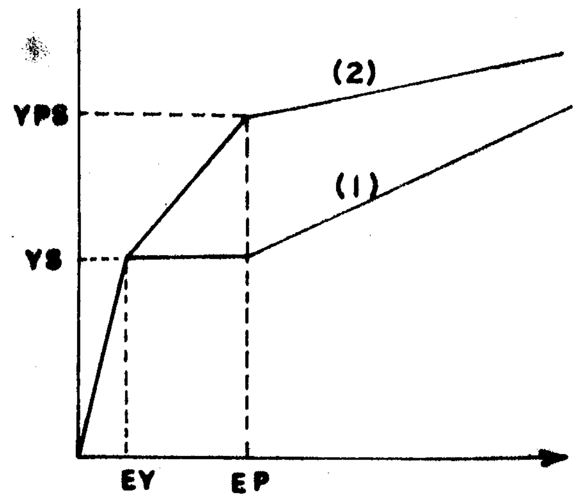
EULT(MN) : Ultimate strain of the material.

YS(MN) : Yield stress for this type of material.

YPS(MN) : Strain hardening stress

ULTS(MN) : Ultimate stress of the material.

For the above definitions see the figure below.



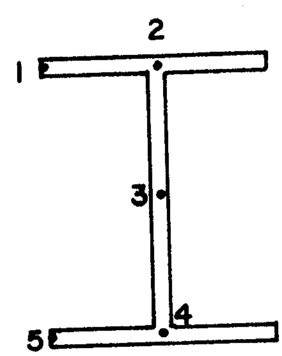
D.5.3 Member Residual Strains Cards (I5, 5F12.0)

One card for each different type of residual strain distribtuion (As many cards as NDRT).

5	17	65
NR		EPR(NR,I), I=1,5

NR : Type number of the section residual strain distribution.

EPR(NR,I) : The value of residual strain at location I of the cross section (see figure below).



D.5.4 Member Property and Connectivity Cards (9I5)

One card per member, unless automatic member data generation is used. The data are arranged so that the flexural elements are introduced first followed by the truss elements (bracing members).

5	10	15	20	25	30	35	40	45
M	NOD(I,M)	NOD(J,M)	MXST(M)	MT(M)	MRT(M)	INC	NOD1	NOD2

M : Member number; the last member card must have M = NE.

NOD(I,M) : The node number at end I of member M

NOD(J,M) : The node number at end J of member M

MXST(M) : Type number of the cross section for member M

MT(M) : Material type number for the member

MRT(M) : Residual strain distribution type number for the member cross section.

INC : If non-zero, automatic generation of member data will be initiated. The new members will have numbers (MOLD + K * INC), where MOLD is the member number on the preceding card, and K is a positive integer. These members will have the cross section and material properties of the member on the card initiating the generation (i.e. the present card). The end nodes of the new members will be (NOD(I,MOLD) + K * NOD1) and (NOD(J,MOLD) + K * NOD2) for nodes at end I and J, respectively. INC must be zero for the first member card and zero or positive otherwise.

NOD1 : Incrementation value for nodal number of end I.

NOD2 : Incrementation value for nodal number of end J.

D.5.5 <u>Echo Check Flag Card</u> (I5)

5
K

K : If 1, the complete nodal geometry and member data will be contained in the output.

If 0, the output will not contain an echo check of the completed data.

D.6 External Boundary Elements Cards (3I6, F12.0)

One card per external boundary element.

6	12	18	30
N	NPB(N)	KODE(N)	BES(N)

N : Number of the external boundary element.

NPB(N) : Number of the node to which this element is attached.

KODE(N) : A five digit number of the form ijklm, corresponding to degrees of freedom u, v, r, $\frac{\partial u}{\partial x}$, and $\frac{\partial v}{\partial y}$, respectively. If any of these is 1, the corresponding degree of freedom will be restrained, otherwise the digit must be 0. Notice that the degree of freedom corresponding to strain perpendicular to the member should be restrained.

BES(N) : The spring stiffness of this element, if other than 10^{20}.

D.7 Load Data Cards

D.7.1 Load Flag Card (2I5)

5	10
JLFLAG	MLFLAG

JLFLAG : If 0, no joint loads will be prescribed.
If 1, joint loads will be prescribed.

MLFLAG : If 0, no member loads will be prescribed.
If 1, member loads will be prescribed.

D.7.2 Joint Load/Displacement Cards

This group consists of a number of cards per loaded node, unless automatic load generation is used. The first card in a nodal load group is a node identification card. The rest of the cards for this node, each describes a case of loading IF JLFLAG = 0 on card D.7.1, this group is omitted.

D.7.2.1 Nodal Identification Card (6I5)

One card at the head of a nodal load group.

5	10	15	20	25	30
N	IDESP	NBEL(1)	NBEL(2)	NBEL(3)	INC

N : Node number (not necessarily in order)

IDESP : If non-zero, displacements will be specified.

(NBEL(I), I=1,3) : The boundary element numbers attached to this node, in case non-zero displacements are specified. These three numbers correspond to degrees of freedom u, v, and r, respectively. In case this node is restrained in 3 directions with one element, and non-zero displacements are specified only in 2 directions, then this boundary element number appears only twice in the corresponding locations.

INC : If non-zero, automatic loading generation will be initiated. Nodes with numbers (NOLD + K * INC) will be assigned the same loads as on the node N initiating the generation.

D.7.2.2 Nodal Load Cards (2I5, 3F12.0)

5	10	22	34	46
JKODE	LVF	U	V	R

JKODE : A three digit number of the form ijk, the digits correspond to u,v, and r, respectively. If any digit is 1, the corresponding u, v, or r will be interpreted as a displacement in the direction of the corresponding NBEL of card D.7.2.1, otherwise it must be zero.

LVF : A flag indicating the variability of load. It could be either 0, or 1. Thus, two different sets of loads can be applied to the structure. Each set is increased independently using load factors FL1 and FL2 introduced in Sect. D.1.2. Notice that one of the sets can be kept constant by setting the corresponding load factor equal to 1.0.

U : Load in the x-direction

V : Load in the y-direction.

R : Moment in the x-y plane, positive anti-clockwise.

NOTE: If UNIT = 12.0, and YMOD is in K/in^2, U and V must be in kips and R in K-ft. Notice that the U, V, and R values are assembled directly into the load array B.

D.7.2.3 Termination Card

One blank card at the end of the joint load/displacement card group.

D.7.3 Member Load Specification Cards

This group consists of a number of cards for each member on which non-zero loads are applied, unless automatic load generation is used. Two sets of load can be applied to the structure, one is kept constant and the other is increased monotonically or both can be increased monotonically. Load generation is carried out for one load for one case of loading at a time. In cases other than when automatic load generation is used, the cards may be entered in any order, since any one card defines completely the member number, the case of loading, the load, and its type, position, and orientation with respect to the member. If MLFLAG=0 on card D.7.1 this group is omitted.

D.7.3.1 Member Load Cards (4I4, 2F12.0, 4F10.0)

One card per member per load per load type, unless automatic load generation is used.

4	8	12	16	28	40	50	60	70	80
M	INC	LVF	K	CL	CN	QI	AI	QJ	AJ

M : Member number.

INC : If non-zero, automatic load generation will be initiated. The load on the card initiating the generation, or its end effects, will be assigned to members with numbers (MOLD + L * INC), where MOLD is the member number on the preceding card, and L is a positive integer.

LVF : A load variability flag defined in Sect. D.7.2.2

K : If 1, the load is concentrated.
If 2, the load is uniformly distributed
If 3, the load is trapezoidal.

CL : Projection along the member of a vector in direction of load.

CN : Projection normal to the member of a vector in direction of load.

QI : Magnitude of load at distance (AI * length) from end I.

AI : Fraction of length from end I to beginning of load.

QJ : Magnitude of load at distance (AJ * length) from end I.

AJ : Fraction of length from end I to end of load.

NOTE: If the load is concentrated, QJ and AJ may be omitted. If the load is uniform, QJ may be omitted. Units of load are in K, or K/ft, if UNIT = 12.0.

D.7.3.2 Termination Card

One blank card at the end of each group of member load cards assigned to a particular load type.

APPENDIX E

PROGRAM DESCRIPTIONS AND LISTINGS

APPENDIX E

PROGRAM DESCRIPTIONS AND LISTINGS

E.1 Program ESTANF

Program ESTANF consists of the following parts.

1. MAIN
2. MAINMG, the main executive subroutine
3. The input package

   INPUT1, INPUT2, BOUND, LOADCV, INPUTI, JLOAD1, and MLOAD1.
4. The data managing package

   ISPAC, LOCOM, REMOV, REMOV2, and BLOCK DATA.
5. The data storage and retrieval package CLEAR, ICLEAR, RTRV1, RTRV2, STORE1, and STORE2.
6. The equation solving package

   (a) ADDRES and COLHT.

   (b) EQSBST and BKSB1
7. The formulation and output package

   (a) ASSEMB, ELEMS, MODIFY, MULT1 and BOUND1

   (b) STIFF, STIFF1, DISPL2, STRES2, and EIGV.

The listing for packages, 1, 2, 3, 6b, and 7b follows. The listing for packages 4, 5, 6a, and 7a can be found in Sections E.6, E.7, E.9, and E.10, respectively. Descriptions of the subroutines can be found in the listing.

This program uses two line-files, one of which is permanent (FILE1) and the other is temporary (FILE2). They are used in sequential I/O operations.

```
C*************************************************************
C
C                     ESTANF
C                     ******
C
C       THIS IS A PROGRAM FOR EIGENVALUE STABILITY ANALISIS
C     OF FRAMES . THE PROGRAM SOLVES FOR LOWEST EIGENVALUE
C     USING INVERSE POWER ITERATION TECHNIQUE. THE EQUATION
C     SOLVER USED IS CHOLESKY DECOMPOSITION OF SKYLINE
C     INCORE TYPE.
C
C     *******************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 NAMES
C
      COMMON/PROBIA/ NNN(500)
      COMMON/IA1/    MMM(200)
      COMMON/IA2/    KKK(1000)
      COMMON/RA1/    AAA(4000)
      COMMON/RA2/    BBB(25000)
      COMMON/DIMCOM/LAST1,LAST2,LAST3,LAST4,LAST5,MXDIM
     *,NAMES(5,20),IPT(5,21),ICOM(5)
C
C     *******************************************************
C
      ICOM(1) = 500
      ICOM(2) = 4000
      ICOM(3) = 200
      ICOM(4) = 25000
      ICOM(5) = 1000
C
      CALL MAINMG
C
      END
```

```
C*************************************************************
C
                      SUBROUTINE MAINMG
C                     ****************
C
C          THIS IS THE MAIN MANEGER OF PROGRAM ESTANF. IT
C        CONTROLS THE READING , THE FORMULATION , THE SOLUTION
C        AND THE OUTPUT OF THE RESULTS.
C
C       ******************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
        REAL*8 NAMES,NAME
C
        COMMON/PROBCV/ UNIT,NJ,NE,NLC,IN,IO,NEBEL,IFEIG,
       *NLT,IREF
C
        COMMON/PROBIA/ NNN(1)
        COMMON/IA1/    MMM(1)
        COMMON/IA2/    KKK(1)
        COMMON/RA1/    AAA(1)
        COMMON/RA2/    BBB(1)
        COMMON/DIMCOM/LA1,LA2,LA3,LA4,LA5,MXDIM,NAMES(5,20),
       *IPT(5,21),ICOM(5)
C
C       ******************************************************
C
        CALL TIME(0,0)
        IN = 5
        IO = 6
C
C         READ PROBLEM CONTROL VARIABLES
C         ------------------------------C
        CALL INPUT 1
C
        IF(IREF.EQ.0) GO TO 20
        WRITE(IO,2000)
        REWIND 1
        READ (1) L1,L2,L3,L4,L5,M1
        READ (1) NN,NWA,ILOAD,TOL,ITLIM,IFPR,II4,II5
        CALL RTRV2 (1,BBB(1),II4)
        CALL RTRV1 (1,KKK(1),II5)
        GO TO 200
20      NODE  = 2
        IDOF  = 3
        ILOAD = 1
        CALL CLEAR (NAMES(1,1),100)
        REWIND 1
        REWIND 2
C
C
C READ NODAL GEOMETRY AND MEMBER PROP. AND CONNECTIVITIES
C
        J1 = ISPAC (1HX,NJ,2)
```

```
      J2 = ISPAC (1HY,NJ,2)
      J3 = ISPAC (4HAREA,NE,2)
      J4 = ISPAC (2HRI,NE,2)
      J5 = ISPAC (4HYMOD,NE,2)
      I1 = ISPAC (3HNOD,(NE*NODE),1)
      I2 = ISPAC (5HMKODE,NE,1)
      I3 = ISPAC (5HMFLAG,NE,1)
C
      CALL INPUT 2 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J5),
     *NNN(I1),NNN(I2),NNN(I3))
C
      J6 = ISPAC (3HFEF,(IDOF*NODE*NE*NLC),2)
      CALL CLEAR (AAA(J6),(IDOF*NODE*NE*NLC))
C
C        READ EXTERNAL BOUNDARY CONDITIONS
C
      J7 = ISPAC (3HBES,NEBEL,2)
      K1 = ISPAC (3HNPB,NEBEL,3)
      K2 = ISPAC (4HKODE,NEBEL,3)
      CALL BOUND (AAA(J7),MMM(K1),MMM(K2),NEBEL,IN,IO)
C
      L1 = ISPAC (1HB,(IDOF*NJ*NLC),4)
      CALL CLEAR (BBB(L1),(IDOF*NJ*NLC))
C
C       READ LOADS IF ANY
C
      J8 = ISPAC (2HFL,NLT,2)
      J9 = ISPAC (3HLVF,NLT,2)
C
      CALL LOADCV (AAA(J8),AAA(J9),NLT,JLFLAG,MLFLAG,IN,IO)
C
C   READ JOINT LOADS AND / OR DISPLACEMENTS
C
      IF(JLFLAG.EQ.0) GO TO 50
      CALL JLOAD1 (BBB(L1),AAA(J7),AAA(J8),AAA(J9),NLT,
     *NLC,IN,IO)
C  READ AND PREPARE MEMBER LOADS IF ANY
C
50    IF(MLFLAG.EQ.0) GO TO 60
      CALL MLOAD1 (AAA(J1),AAA(J2),AAA(J6),AAA(J8),AAA(J9),
     *BBB(L1),NNN(I1),NNN(I2))
60    CONTINUE
      CALL REMOV (3HLVF,2)
      CALL REMOV (2HFL,2)
C
C        FORM COLUMN HEIGHTS AND ADDRESSING ARRAY
C
      M1 = ISPAC (4HMAXA,(IDOF*NJ+1),5)
      M2 = ISPAC (3HMHT,(IDOF*NJ),5)
      CALL ICLEAR (KKK(M2),(IDOF*NJ))
      DO 70 I=1,NE
      CALL COLHT (NODE,(NODE*IDOF),IDOF,I,
     *KKK(M2),NNN(I1))
70    CONTINUE
```

```
C
      NEQ = NJ * IDOF
      CALL ADDRES (KKK(M1),KKK(M2),NEQ,NWA)
      CALL REMOV (3HMHT,5)
C
C      RESERVE AND CLEAR SPACE FOR ELASTIC STIFFNESS MATRIX
C
      L2 = ISPAC (1HA,NWA,4)
      CALL CLEAR (BBB(L2),NWA)
C
C      FORM ELEMENT STIFFNESSES AND ASSEMBLE ELASTIC
C    STIFFNESS INTO A.
C
      CALL STIFF (NNN(I1),NNN(I2),NNN(I3),KKK(M1),AAA(J1),
     *AAA(J2),AAA(J3),AAA(J4),AAA(J5),BBB(L2),
     *UNIT,NE,IFEIG)
C
C
C       ADD EXTERNAL BOUNDARY CONDITIONS
C
      CALL BOUND1 (BBB(L2),KKK(M1),MMM(K1),MMM(K2),AAA(J7),
     *NEBEL)
      CALL STORE2 (2,BBB(L2),NWA)
C
100   CALL TIME (3,3)
C
      CALL EQSBST (BBB(L2),BBB(L1),KKK(M1),NEQ,NLC,ILOAD,0)
      CALL BKSB1 (BBB(L1),BBB(L2),KKK(M1),NEQ,NLC,0)
C
      WRITE (IO,2100)
      CALL TIME (3,3)
      J8 = ISPAC (4HFEF2,(IDOF*NODE*NE*NLC),2)
      CALL CLEAR (AAA(J8),(IDOF*NODE*NE*NLC))
      DO 110 I=1,NLC
      CALL DISPL2 (AAA(J8),BBB(L1),AAA(J6),I,NJ,NE,NLC,
     *IN,IO)
      CALL STRES2 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J5),
     *BBB(L1),AAA(J8),NNN(I1),NNN(I2),NE,I,NLC,UNIT,IN,IO)
110   CONTINUE
C
      CALL TIME (3,3)
      IF(IFEIG.EQ.0) GO TO 900
      ILOAD = 2
      CALL CLEAR (BBB(L2),NWA)
      L3 = ISPAC (3HAGV,NWA,4)
      CALL CLEAR (BBB(L3),NWA)
      REWIND 2
      CALL RTRV2 (2,BBB(L2),NWA)
      NN = NEQ
      L4 = ISPAC (2HVO,NN,4)
      L5 = ISPAC (2HTT,NN,4)
      CALL CLEAR (BBB(L5),NN)
      CALL INPUTI (BBB(L4),MMM(K1),MMM(K2),NEBEL,NN,TOL,
     *ITLIM,IFPR,IN,IO)
```

```
C
      CALL STIFF1 (NNN(I1),NNN(I3),BBB(L2),BBB(L3),KKK(M1),
     *AAA(J8),NE,NLC)
      REWIND 2
      CALL STORE2 (2,BBB(L3),NWA)
      CALL STORE2 (2,BBB(L2),NWA)
      II4 = IPT(4,LA4+1) - 1
      II5 = IPT(5,LA5+1) - 1
200   CALL EIGV (BBB(L2),BBB(L3),BBB(L5),BBB(L4),KKK(M1),
     *NN,NWA,ILOAD,TOL,ITLIM,IFPR,IREF,IR,IN,IO)
C   END OF PROBLEM
      CALL TIME (3,3)
      IF(IR.EQ.0) GO TO 900
      REWIND 1
      WRITE (1) L1,L2,L3,L4,L5,M1
      WRITE (1) NN,NWA,ILOAD,TOL,ITLIM,IFPR,II4,II5
      CALL STORE2 (1,BBB(1),II4)
      CALL STORE1 (1,KKK(1),II5)
      CALL TIME (3,3)
900   RETURN
C
C        FORMAT STATEMENTS
C
1000  FORMAT (I5)
1100  FORMAT (2I5)
2000  FORMAT (5X,'RE-START OF PROBLEM'//,5X,19(1H*))
2100  FORMAT (//,'BACK SUBSTITUTION OF STRUCTURE',
     *' PROBLEM IS COMPLETED'//)
C
      END
```

```
C*************************************************************
C
                    SUBROUTINE INPUT 1
C                   *****************
C
C
C        THIS SUBROUTINE READS THE PROBLEM CONTROL VARIABLES
C       FOR PROGRAM ESTANF.
C
C      ***************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON/PROBCV/ UNIT,NJ,NE,NLC,IN,IO,NEBEL,IFEIG,
     *NLT,IREF
C
      DIMENSION HED(20)
C
      READ(IN,1000) HED
      WRITE(IO,2000) HED
      READ (IN,1100) IREF
      IF(IREF.EQ.1) RETURN
      READ (IN,1200) NJ,NE,NEBEL,NLC,NLT,IFEIG,UNIT
      WRITE(IO,2100) NJ,NE,NEBEL,NLC,NLT,IFEIG,UNIT
C
      RETURN
C
1000  FORMAT (20A4)
1100  FORMAT (I5)
1200  FORMAT (6I4,F12.0)
2000  FORMAT (///,30X,20A4//,30X,20(1H*)///)
2100  FORMAT (5X,'PROBLEM CONTROL VARIABLES'/,5X,25(1H*)//,
     *10X,'NUMBER OF JOINTS                        =',I5//,
     *10X,'NUMBER OF ELEMENTS                      =',I5//,
     *10X,'NO. OF EXTERNAL BOUND. ELEMENTS         =',I5//,
     *10X,'NUMBER OF LOAD CASES                    =',I5//,
     *10X,'NUMBER OF LOAD TYPES                    =',I5//,
     *10X,'EIGEN PROBLEM RUN FLAG                  =',I5//,
     *10X,'LENGTH CONVERSION FACTOR                =',F12.6//,
     *10X,50(1H*))
C
      END
```

```
C*************************************************************
C
       SUBROUTINE INPUT 2 (X,Y,AREA,RI,YMOD,NOD,MKODE,MFLAG)
C      *****************
C
C         THIS SUBROUTINE READS THE NODAL GEOMETRY AND
C      MEMBER PROPERTIES . PROGRAM ESTANF
C
C     ***************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON/PROBCV/ UNIT,NJ,NE,NLC,IN,IO,NEBEL,IFEIG,
     *NLT,IREF
      DIMENSION X(1),Y(1),AREA(1),RI(1),YMOD(1),NOD(2,1),
     *MKODE(1),MFLAG(1)
C
      WRITE(IO,2000)
50    READ (IN,1000) N,X(N),Y(N),INC
      IF(INC.EQ.0) GO TO 200
      NINT = (N-NOLD) / INC
      RN   = NINT
      IF(RN.LT.FLOAT(N-NOLD)/FLOAT(INC)-0.001) GO TO 999
      DX = (X(N)-X(NOLD))/RN
      DY = (Y(N)-Y(NOLD))/RN
      L = NOLD
      M = NINT - 1
      DO 100 J=1,M
      LL = L + INC
      X(LL) = X(L) + DX
      Y(LL) = Y(L) + DY
      L = LL
100   CONTINUE
200   WRITE (IO,2100) N,X(N),Y(N),INC
      NOLD = N
      IF(N.LT.NJ) GO TO 50
C
      READ (IN,1100) ADEF,RDEF,YDEF
      WRITE(IO,2200) ADEF,RDEF,YDEF
C
      WRITE(IO,2300)
300   READ (IN,1200) M,(NOD(I,M),I=1,2),MKODE(M),MFLAG(M),
     *INC,NOD1,NOD2,AREA(M),RI(M),YMOD(M)
      IF(AREA(M).EQ.0) AREA(M) = ADEF
      IF(RI(M).EQ.0)   RI(M)   = RDEF
      IF(YMOD(M).EQ.0) YMOD(M) = YDEF
      IF(INC.EQ.0) GO TO 500
      NINT = (M-MOLD) / INC - 1
      L = MOLD
      DO 400 I=1,NINT
      LL         =L + INC
      AREA(LL)   = AREA(M)
      RI(LL)     = RI(M)
      YMOD(LL)   = YMOD(M)
```

```
      MKODE(LL) = MKODE(M)
      MFLAG(LL) = MFLAG(M)
      NOD(1,LL) = NOD(1,L) + NOD1
      NOD(2,LL) = NOD(2,L) + NOD2
400   L = LL
C
500   WRITE(IO,2400) M,NOD(1,M),NOD(2,M),MKODE(M),MFLAG(M),
     *INC,NOD1,NOD2,AREA(M),RI(M),YMOD(M)
      MOLD = M
      IF(M.LT.NE) GO TO 300
      READ (IN,1300) K
      IF(K.EQ.0) GO TO 600
      WRITE(IO,2700)
      WRITE(IO,2750) (N,X(N),Y(N),N=1,NJ)
      WRITE(IO,2800)
      WRITE(IO,2850) (M,(NOD(I,M),I=1,2),MKODE(M),MFLAG(M),
     *AREA(M),RI(M),YMOD(M),M=1,NE)
C
600   RETURN
C
999   WRITE (IO,9999) N
9999  FORMAT('NODAL GEOMETRY DATA INPUT ERROR',I5)
      STOP
C
C        FORMAT STATEMENTS
C
1000  FORMAT(I5,2F12.0,I5)
1100  FORMAT(3F12.0)
1200  FORMAT(8I5,3F12.0)
1300  FORMAT(I5)
2000  FORMAT (///,5X,'NODAL GEOMETRY DATA AS INPUT'/,5X,
     *28(1H*)//,10X,1HN,7X,1HX,14X,1HY,9X,3HINC/)
2100  FORMAT (6X,I5,2D15.6,I5)
2200  FORMAT (///,5X,'MEMBER PROPERTIES DEFAULT VALUES'/,
     *5X,32(1H*)//,10X,'AREA',11X,' =',D15.6,//,
     *10X,'M. INERTIA',6X,'=',D15.6,//,
     *10X,'Y. MODULUS',6X,'=',D15.6)
2300  FORMAT (///,5X,'MEMBER DATA AS INPUT'/,5X,20(1H*)//,
     *9X,1HM,4X,1HI,4X,1HJ,2X,4HCODE,2X,4HFLAG,2X,3HINC,
     *1X,4HINCI,1X,4HINCJ,5X,4HAREA,13X,1HI,13X,4HYMOD/)
2400  FORMAT (5X,8I5,3D15.6)
2700  FORMAT (///,5X,'COMPLETED NODAL GEOMETRY DATA'/,
     *5X,30(1H*)//,4X,1HN,7X,1HX,14X,1HY/)
2750  FORMAT (I5,2D15.6)
2800  FORMAT (///,5X,'COMPLETED MEMBER DATA'/,5X,21(1H*),
     *5X,1HM,4X,1HI,4X,1HJ,2X,4HCODE,2X,4HFLAG,5X,4HAREA,
     *13X,1HI,12X,4HYMOD/)
2850  FORMAT (5I5,3D15.6)
C
      END
```

```
C*****************************************************
C
                SUBROUTINE BOUND (BES,NPB,KODE,NEBEL,IN,IO)
C               ***************
C
C        THIS SEGMENT READS THE EXTERNAL BOUNDARY CONDITIONS
C
C     ************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION BES(1),NPB(1),KODE(1)
C
      DO 100 J=1,NEBEL
      READ (IN,1000) N,NPB(N),KODE(N),BES(N)
100   IF(BES(N).EQ.0) BES(N) = 1.0E20
      WRITE(IO,2000)
      WRITE(IO,2100) (N,NPB(N),KODE(N),BES(N),N=1,NEBEL)
C
      RETURN
C
C   FORMAT  STATEMENTS
C
1000  FORMAT (3I6,F12.0)
2000  FORMAT (///,T30,'EXTERNAL BOUNDARY ELEMENTS DATA'/,
     *T30,31(1H*)//,
     *3(4X,1HN,3X,3HNPB,1X,4HKODE,7X,2HEB,6X)/)
2100  FORMAT  (3(3I5,D15.6))
C
      END
```

```
C*********************************************************************
C
            SUBROUTINE LOADCV (FL,LVF,NLT,JLFLAG,MLFLAG,IN,IO)
C           *****************
C
C              THIS SUBROUTINE READS CONTROL PARAMETERS,LOAD
C           FACTORS, AND LOAD VARIABLE FLAG.
C
C           ******************************************************
C
            IMPLICIT REAL*8(A-H,O-Z)
C
            DIMENSION FL(1),LVF(1)
C
            WRITE(IO,2000)
            READ (IN,1000) JLFLAG,MLFLAG
            DO 100 LT=1,NLT
            READ (IN,1100) LT,LVF(LT),FL(LT)
100         WRITE(IO,2100) LT,LVF(LT),FL(LT)
C
            RETURN
C
C    FORMAT STATEMENTS
C
1000        FORMAT (2I5)
1100        FORMAT (2I5,F12.0)
2000        FORMAT (//,5X,'LOAD CONTROL PARAMETERS'/,5X,23(1H*)
           *//,10X,'LOAD TYPE   LVF  LOAD FACTOR'/)
2100        FORMAT (10X,I5,4X,I5,2X,F12.6)
C
            END
```

```
C***********************************************************
C
            SUBROUTINE INPUTI (VO,NPB,KODE,NEBEL,NN,TOL,
     *                              ITLIM,IFPR,IN,IO)
C           ****************
C
C         THIS ROUTINE READS INFORMATION NECESSARY FOR INVERSE
C       ITERATION.
C
C       ****************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION VO(1),NPB(1),KODE(1)
C
        WRITE(IO,2000)
        READ (IN,1000) TOL,ITLIM,IFPR,INFLAG
        WRITE(IO,2100) TOL,ITLIM,IFPR,INFLAG
        WRITE(IO,2200)
C
        K = INFLAG + 1
        GO TO (100,500),K
C
100     DO 40 I=1,NN
40      VO(I) = 1.0
        DO 400 KK=1,NEBEL
        N = NPB(KK)
        KOD = KODE(KK)
        NE  = 3 * N
        IF((KOD-100).LT.0) GO TO 200
        VO(NE-2) = 0.0
        KOD = KOD - 100
200     IF((KOD-10).LT.0) GO TO 300
        VO(NE-1) = 0.0
        KOD = KOD-10
300     IF(KOD.EQ.0) GO TO 400
        VO(NE) = 0.0
400     CONTINUE
        GO TO 600
500     READ (IN,1100) (VO(I) ,I=1,NN)
C
600     WRITE(IO,2300) (VO(I),I=1,NN)
C
        RETURN
C
C   FORMAT STATEMENTS
C
1000    FORMAT (F12.6,3I5)
1100    FORMAT (8F10.0)
2000    FORMAT (///,5X,'EIGEN PROBLEM BEGINS'/
     *,5X,20(1H*)//)
2100    FORMAT (10X,'EIGEN PROBLEM CONTROL PARAMETERS'/
     *,10X,32(1H-)//,
     *10X,'TOLERANCE LIMIT                  =',F12.6//,
```

```
     *10X,'MAXIMAUM NO. OF ITERATIONS    =',I5//,
     *10X,'PRINT CODE                    =',I5//,
     *10X,'INPUT CODE                    =',I5//,
     *10X,40(1H*))
2200  FORMAT (//,5X,'ASSUMED EIGENVECTOR'//)
2300  FORMAT (8F12.6)
C
      END
```

```
C*********************************************************
C
            SUBROUTINE JLOAD1 (B,BES,FL,LVF,NLT,NLC,IN,IO)
C           ****************
C
C           THIS SEGMENT READS JOINT LOADS AND/OR DISPLACEMENTS
C
C     ****************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION B(NLC,1),BES(1),FL(1),LVF(1),NBEL(3)
C
C
      WRITE(IO,2000)
      DO 900 LT=1,NLT
200   READ (IN,1000) N,IDESP,(NBEL(I),I=1,3),INC
      IF(N.LE.0) GO TO 900
      N3 = 3 * N
      N2 = N3 - 1
      N1 = N3 - 2
      READ (IN,1100) JKODE,U,V,R
      IF(IDESP.EQ.0) GO TO 300
C
      K = JKODE
      IF((K-100).LT.0) GO TO 225
      B(J,N1) = U * BES(NBEL(1))
      K = K - 100
225   IF((K-10).LT.0) GO TO 250
      B(J,N2) = V * BES(NBEL(2))
      K = K - 10
250   IF(K.EQ.0) GO TO 275
      B(J,N3) = R * BES(NBEL(3))
275   CONTINUE
C
300   WRITE(IO,2100)N,(NBEL(I),I=1,3),INC
      WRITE(IO,2200) JKODE,LT,LVF(LT),U,V,R
      U = U * FL(LT)
      V = V * FL(LT)
      R = R * FL(LT)
      IF(LVF(LT).EQ.0) GO TO 325
      LC = 1
      GO TO 350
325   LC = 2
350   B(LC,N1) = B(LC,N1) + U
      B(LC,N2) = B(LC,N2) + V
      B(LC,N3) = B(LC,N3) + R
      IF(INC.EQ.0) GO TO 800
      NINT = (N-NOLD) / INC - 1
      L = NOLD
      DO 700 I=1,NINT
      LL = L + INC
      L3 = LL * 3
      L2 = L3 - 1
```

```
      L1 = L3 - 2
      B(LC,L1) = B(LC,L1) + U
      B(LC,L2) = B(LC,L2) + V
      B(LC,L3) = B(LC,L3) + R
700   L = LL
800   NOLD = N
      GO TO 200
900   CONTINUE
      RETURN
C
C        FORMAT STATEMENTS
C
1000  FORMAT (6I5)
1100  FORMAT (I5,3F12.0)
2000  FORMAT (///,T30,'JOINT LOADS DATA AS INPUT'/,T30,25
     *(1H*)//,'     N   NBU  NBV  NBR  INC    CODE  LT  LVF',
     *7X,'U',14X,'V',14X,'R'/)
2100  FORMAT (5I5)
2200  FORMAT (27X,3I5,3D15.6)
C
      END
```

```
C*******************************************************************
C
            SUBROUTINE MLOAD1 (X,Y,FEF,FL,LVF,B,NOD,MKODE)
C           ****************
C
C           THIS SUBROUTINE READS MEMBER LOADS AND CALCULATES
C         THE END FORCES. THE END FORCES ARE STORED IN VECTOR
C         B AS EQUVALENT LOADS, AND IN VECTOR FEF AS MEMBER
C         END FORCES. THE JOINT LOADS ARE THEN ADDED TO B .
C
C     ***************************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON/PROBCV/ UNIT,NJ,NE,NLC,IN,IO,NEBEL,IFEIG,
     *NLT,IREF
      DIMENSION X(1),Y(1),FEF(NLC,1),B(NLC,1),NOD(2,1),
     *MKODE(1),FL(1),LVF(1),FQ(6),FK(6)
C
C      MODIFY UNITS FOR LOAD VECTOR
C
      DO 200 J=1,NLC
      DO 200 I=1,NJ
      II = 3 * I
      B(J,II) = B(J,II) * UNIT
200   CONTINUE
C
C      READ AND PREPARE MEMBER LOADS
C
      C13 = 1. / 3.
      WRITE(IO,2000)
      DO 800 LT=1,NLT
250   READ (IN,1000) M,INC,K,CL,CN,QI,AI,QJ,AJ
      IF(M.LE.0) GO TO 800
      WRITE(IO,2100) M,LT,LVF(LT),INC,K,CL,CN,QI,AI,QJ,AJ
      KKK = 0
      MM  = M
      QI = QI * FL(LT)
      QJ = QJ * FL(LT)
275   CONTINUE
      I = NOD(1,MM)
      J = NOD(2,MM)
      DX = (X(J) - X(I))
      DY = (Y(J) - Y(I))
      XL = DSQRT(DX**2+DY**2)
      CT = DX / XL
      ST = DY / XL
      DC = 1. / DSQRT(CL**2+CN**2)
      COSG = DC * CL
      SING = DC * CN
      AIM = 1. - AI
C
      GO TO (300,400,500),K
C
```

```
C        FIXED END FORCES FOR CONCENTRATED LOADS
C
300   COSG = COSG * QI
      SING = SING * QI
      FQ(3) = -AIM**2*AI*XL*SING
      FQ(6) =  AIM*AI**2*XL*SING
      DV    = (FQ(3) + FQ(6)) / XL
      FQ(1) = -AIM*COSG
      FQ(2) = -AIM * SING + DV
      FQ(4) = -AI * COSG
      FQ(5) = -AI * SING - DV
      GO TO 600
C
C       FIXED END FORCES FOR UNIFORM LOADS
C
400   D1    = AJ - AI
      D2    = 0.5 * (AJ**2 - AI**2)
      D3    = C13 * (AJ**3 - AI**3)
      D4    = 0.25 *(AJ**4 - AI**4)
      COSG  = COSG * QI * XL
      SING  = SING * QI * XL
      FQ(3) = -XL * SING * (D2 - 2.*D3 + D4)
      FQ(6) =  XL * SING * (D3 - D4)
      DV    =  (FQ(3) + FQ(6)) / XL
      FQ(1) = -COSG * (D1 - D2)
      FQ(2) = -SING * (D1 - D2) + DV
      FQ(4) = -COSG * D2
      FQ(5) = -SING * D2 - DV
      GO TO 600
C
C        FIXED END FORCES FOR TRAPEZOIDAL LOADS
C
500   AL    = XL / (AJ - AI)
      COSGI = COSG * QI * AL
      COSGJ = COSG * QJ * AL
      SINGI = SING * QI * AL
      SINGJ = SING * QJ * AL
      D1    = AJ - AI
      D2    = 0.5 * (AJ**2 - AI**2)
      D3    = C13 * (AJ**3 - AI**3)
      D4    = .25 * ( AJ**4 - AI**4)
      D5    = 0.2 * (AJ**5 - AI**5)
      D6    = D2 - 2.*D3 + D4
      D7    = D3 - 2.*D4 + D5
      AII   = AJ*D1 - (1.+AJ)*D2 +D3
      BII   = AJ*D6 - D7
      AIJ   = D2 - D3 - AI*(D1-D2)
      BIJ   = D7 - AI*D6
      AJI   = AJ*D2 - D3
      BJI   = AJ*(D3-D4) - (D4-D5)
      AJJ   = D3 - AI*D2
      BJJ   = D4 - D5 - AI*(D3-D4)
C
      FQ(3) = -XL * (SINGI*BII + SINGJ*BIJ)
```

```
      FQ(6) =  XL * (SINGI*BJI + SINGJ*BJJ)
      DV    = (FQ(3) + FQ(6)) / XL
      FQ(1) = -(COSGI*AII + COSGJ*AIJ)
      FQ(2) = -(SINGI*AII + SINGJ*AIJ) + DV
      FQ(4) = -(COSGI*AJI + COSGJ*AJJ)
      FQ(5) = -(SINGI*AJI + SINGJ*AJJ) - DV
C
C        MODIFY END FORCES VECTOR FQ FOR HINGED ENDS
C
600   CONTINUE
      KI = MKODE(MM) + 1
      DO 605 KK = 1,6
605   FK(KK) = FQ(KK)
      GO TO (650,620,630,610),KI
610   FK(3) = 0.0
      FK(6) = 0.0
      GO TO 640
620   FK(6) = FK(6) - 0.5*FK(3)
      DV    = 1.5*FK(3) / XL
      FK(3) = 0.0
      GO TO 640
630   FK(3) = FK(3) - 0.5*FK(6)
      DV    = 1.5*FK(6) / XL
640   FK(2) = FK(2) - DV
      FK(5) = FK(5) + DV
650   FK(3) = FK(3) * UNIT
      FK(6) = FK(6) * UNIT
C
C        ASSEMBLE FQ INTO FEF
C
      IF(LVF(LT).EQ.0) GO TO 660
      LC = 1
      GO TO 670
660   LC = 2
670   KEF = 6 * MM - 6
      DO 675 KK =1,6
      KEF = KEF + 1
      FEF(LC,KEF) = FEF(LC,KEF) + FK(KK)
675   CONTINUE
C
C       ASSEMBLE FQ INTO LOAD VECTOR B
C
      II   = 3*NOD(1,MM)
      B(LC,II-2) = B(LC,II-2) - FK(1)*CT + FK(2)*ST
      B(LC,II-1) = B(LC,II-1) - FK(1)*ST - FK(2)*CT
      B(LC,II)   = B(LC,II)   - FK(3)
      JJ   = 3*NOD(2,MM)
      B(LC,JJ-2) = B(LC,JJ-2) - FK(4)*CT + FK(5)*ST
      B(LC,JJ-1) = B(LC,JJ-1) - FK(4)*ST - FK(5)*CT
      B(LC,JJ)   = B(LC,JJ)   - FK(6)
C
700   IF(INC.EQ.0) GO TO 750
      MOLD = MOLD +  IABS(INC)
      MM = MOLD
```

```
      IF(MM.GT.M) GO TO 999
      IF(MM.EQ.M) GO TO 750
      IF(INC.EQ.0) GO TO 750
      GO TO 600
750   CONTINUE
      MOLD = MM
      GO TO 250
C
800   CONTINUE
C
      RETURN
C
999   WRITE(IO,9999) MM,M
      STOP
C
C        FORMAT STATEMENTS
C
1000  FORMAT (3I5,2F12.0,4F10.0)
2000  FORMAT (///,T30,'MEMBER LOADS AS INPUT'/,4X,
     *'M   LT  LVF  INC CODE',6X,'CL',13X,'CN',13X,'QI',
     *13X,'AI',13X,'QJ',13X,'AJ'/)
2100  FORMAT (5I5,6D15.6)
9999  FORMAT ('GENERATION INCREMENT ERROR    MM=',I5,
     *'     M=',I5)
C
      END
```

```
C*********************************************************
C
            SUBROUTINE  EQSBST (A,B,MAXA,NB,NLC,ILOAD,IT)
C           *****************
C
C        THIS SUBROUTINE TRIANGULARIZES A STIFFNESS MATRIX
C     STORED COLUMN-WISE UNDER A SKYLINE AND REDUCES THE
C     CORRESPONDING LOAD VECTOR, DOWN TO EQUATION NB.
C
C     *******************************************************
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION A(1),B(NLC,1),MAXA(1)
C
C
      DO  1000 N=1,NB
      KN = MAXA(N)
      KL = KN + 1
      KU = MAXA(N+1) - 1
      KH = KU - KL
      IF(KH) 900,500,100
100   IF(ILOAD.EQ.2.AND.IT.GT.0) GO TO 650
      K  = N - KH
      IC = 0
      KLT= KU
      DO 400 J=1,KH
      IC = IC + 1
      KLT= KLT - 1
      KI = MAXA(K)
      ND = MAXA(K+1) - KI - 1
      IF(ND) 400,400,200
200   KK = IC
      IF(KK.GT.ND) KK = ND
      C  = 0.0
      DO 300 L = 1,KK
300   C  = C + A(KI+L)*A(KLT+L)
      A(KLT) = A(KLT) - C
400   K  = K + 1
500   IF(ILOAD.EQ.2.AND.IT.GT.0) GO TO 650
      K  = N
C
      C  = 0.0
      DO 600 KK = KL,KU
      K  = K - 1
      KI = MAXA(K)
      D  = A(KK)/A(KI)
      C  = C + D*A(KK)
      A(KK) = D
600   CONTINUE
      A(KN) = A(KN) - C
C
650   DO 800 IC=1,NLC
      K  = N
      C = 0.0
```

```
      DO 700 KK = KL,KU
      K  = K - 1
      C  = C + A(KK)*B(IC,K)
700   CONTINUE
      B(IC,N) = B(IC,N) - C
800   CONTINUE
C
900   IF(A(KN)) 950,950,1000
950   WRITE(6,3000) N,A(KN)
      STOP
C
1000  CONTINUE
C
      RETURN
C
3000  FORMAT('ZERO OR NEGATIVE ELEMENT ON MAIN DIAGONAL NO.'
     *,I4,D15.6)
C
      END
```

```
C***********************************************************************
C
                 SUBROUTINE BKSB1 (B,A,MAXA,NN,NLC,KB)
C                ****************
C
C           THIS SUBROUTINE IS A PART OF THE PARTIAL REDUCTION
C       PACKAGE. IT FORMS THE LAST STAGE OF BACKSUBSTITUTION,
C       THE OPERATION (L(-1)(T)*RI*) .   IN CASE OF AN
C       UNSUBSTRUCTURED PROBLEM IT PERFORMS THE OPERATION
C       //D(-1)*L(-1)(T)*R//
C
C       ***********************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION A(1),B(NLC,1),MAXA(1)
C
        IF(KB.NE.0) GO TO 200
        DO 100 I=1,NLC
        DO 100 J=1,NN
        K = MAXA(J)
100     B(I,J) = B(I,J)/A(K)
C
200     N  = NN
        DO 600 L=2,NN
        KL = MAXA(N) + 1
        KU = MAXA(N+1) - 1
        IF(KU-KL) 600,300,300
300     DO 500 I=1,NLC
        IF(B(I,N).EQ.0.) GO TO 500
        K  = N
        DO 400 KK=KL,KU
        K = K - 1
400     B(I,K) = B(I,K) - A(KK)*B(I,N)
500     CONTINUE
600     N  = N - 1
C
        RETURN
C
        END
```

```
C*****************************************************************
C
          SUBROUTINE STIFF (NOD,MKODE,MFLAG,MAXA,X,Y,AREA,RI,
     *                                    YMOD,A,UNT,NE,IFEIG)
C         ****************
C
C          THIS SUBROUTINE FORMS THE ELASTIC STIFFNESS MATRIX
C       OF 6 DOF PLANE FRAME MEMBER. THE UPPER TRIANGLE IS
C       STORED COLUMN-WISE IN A VECTOR.
C
C       ***************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
C
        DIMENSION NOD(2,1),MKODE(1),MFLAG(1),MAXA(1),
       *X(1),Y(1),AREA(1),RI(1),YMOD(1),A(1),LM(6)
       *,AS(7),AC(7),S(21)
C
        REWIND 1
C
        DO 700 M=1,NE
        K = MKODE(M)
        I = NOD(1,M)
        J = NOD(2,M)
        DX = (X(J) - X(I)) * UNT
        DY = (Y(J) - Y(I)) * UNT
        XL = DSQRT (DX**2 + DY**2)
        XLI = 1. / XL
        CO = DX * XLI
        SI = DY * XLI
        CL = CO * XLI
        SL = SI * XLI
        CL2 = CL ** 2
        SL2 = SL ** 2
        CSL = CL * SL
        C2  = CO ** 2
        S2  = SI ** 2
        SC  = CO * SI
        ALP = YMOD(M) * RI(M) * XLI
        BETA= YMOD(M) * AREA(M) * XLI
C
C     FORM LM ARRAY
C
        LM(3) = 3 * I
        LM(2) = LM(3) - 1
        LM(1) = LM(3) - 2
        LM(6) = 3 * J
        LM(5) = LM(6) - 1
        LM(4) = LM(6) - 2
C
C         FORMULATION OF ELASTIC STIFFNESS MATRIX
C
        AS(1) = BETA
```

```
      AS(2) = 12.0 * ALP
      AS(3) = 6.0 * ALP
      AS(4) = AS(3)
      AS(5) = 4.0 * ALP
      AS(6) = 2.0 * ALP
      AS(7) = AS(5)
      IF(K.NE.0) CALL MODIFY(K,AS)
C
C       FORM ELASTIC STIFFNESS MATRIX
C
      CALL ELEMS(AS,S,CL,SL,CL2,SL2,CSL,C2,S2,SC)
      CALL ASSEMB (MAXA,S,A,LM,6)
      IF(IFEIG.EQ.0) GO TO 700
C
C          FORMULATION OF GEOMETRIC ELEMENT STIFFNESS
      L = MFLAG(M)
      IF(L.EQ.0) GO TO 700
      AC(1) = XLI
      AC(2) = 6.0/5.0 * XL
      AC(3) = 1.0/10. * XL
      AC(4) = AC(3)
      AC(5) = 2.0/15. * XL
      AC(6) = -1./30. * XL
      AC(7) = AC(5)
      CALL ELEMS (AC,S,CL,SL,CL2,SL2,CSL,C2,S2,SC)
      CALL STORE2 (1,S,21)
700   CONTINUE
C
      RETURN
C
      END
```

```
C*************************************************************
C
            SUBROUTINE STIFF1 (NOD,MFLAG,A,AGV,MAXA,PP,NE,NLC)
C           ****************
C
C              THIS SUBROUTINE ASSEMBLES THE GEOMETRIC STIFFNESS
C        MATRIX AND ADD THE CONSTANT GEOMETRIC STIFFNESS
C        MATRIX INTO THE ELASTIC STIFFNESS MATRIX.
C
C        *******************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION MFLAG(1),MAXA(1),A(1),AGV(1),PP(NLC,1),
     *S(21),NOD(2,1),LM(6),S1(21)
C
      REWIND 1
      DO 600 M=1,NE
      I = NOD(1,M)
      J = NOD(2,M)
      LM(3) = 3 * I
      LM(2) = LM(3) - 1
      LM(1) = LM(3) - 2
      LM(6) = 3 * J
      LM(5) = LM(6) - 1
      LM(4) = LM(6) - 2
      L=MFLAG(M)
      IF(L.EQ.0) GO TO 600
      CALL RTRV2(1,S,21)
      M1 = 6 * M - 5
      DO 500 LC=1,NLC
      IF(LC.EQ.1) GO TO 400
      DO 300 KK=1,21
300   S(KK) = -S(KK) * PP(LC,M1)
      CALL ASSEMB (MAXA,S,A,LM,6)
      GO TO 500
400   DO 450 JJ=1,21
450   S1(JJ) =  S(JJ) * PP(LC,M1)
      CALL ASSEMB (MAXA,S1,AGV,LM,6)
500   CONTINUE
600   CONTINUE
C
      RETURN
C
      END
```

```
C*****************************************************************
C
              SUBROUTINE DISPL2 (FEF2,B,FEF,K,NJ,NE,NLC,IN,IO)
C             ****************
C
C         THIS SUBROUTINE OUTPUTS NODAL DISPLACEMENTS FOR LOAD
C     CASE (K).  IT PREPARES DUMMY VECTOR FEF2 TO CALCULATE
C     MEMBER END FORCES.
C
C     ****************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION FEF2(NLC,1),B(NLC,1),FEF(NLC,1)
C
      WRITE(IO,2000) K
      DO 100 N=1,NJ
      N3 = N * 3
      N1 = N3 - 2
100   WRITE(IO,2100) N,(B(K,NN),NN=N1,N3)
      LC = K
      DO 300 II=1,NE
      III = 6*II + 1
      DO 300 JJ=1,6
      FEF2(LC,III-JJ) = FEF(LC,III-JJ)
300   CONTINUE
C
      RETURN
C
C       FORMAT STATEMENTS
C
2000  FORMAT ('1','LOAD CASE NO. ',I4,//
     *,'NODAL DISPLACEMENTS',//
     *,4X,'N',7X,'U',14X,'V',14X,'R'/)
2100  FORMAT (I4,3D15.6)
C
      END
```

```
C***********************************************************
C
            SUBROUTINE STRES2 (X,Y,AREA,RI,YMOD,B,FEF2,NOD,
     *                        MKODE,NE,K,NLC,UNIT,IN,IO)
C           ****************
C
C              SUBROUTINE STRES2 COMPUTES AND OUTPUTS MEMBER
C            END FORCES FOR A PLANE FRAME PROBLEM.
C
C      ******************************************************
C
       IMPLICIT REAL*8(A-H,O-Z)
C
       DIMENSION X(1),Y(1),AREA(1),RI(1),YMOD(1),B(NLC,1)
      *,FEF2(NLC,1),NOD(2,1),MKODE(1)
C
       DO 500 M=1,NE
       I   = NOD(1,M)
       J   = NOD(2,M)
       DX  = (X(J) - X(I)) * UNIT
       DY  = (Y(J) - Y(I)) * UNIT
       XLI = 1. / DSQRT(DX**2+DY**2)
       COST= DX * XLI
       SINT= DY * XLI
       C1  = 2. * YMOD(M) * RI(M) * XLI
       C2  = AREA(M) * YMOD(M) * XLI
C
       DU  = B(K,(3*J-2)) - B(K,(3*I-2))
       DV  = B(K,(3*J-1)) - B(K,(3*I-1))
       DP  = C2 * (DU*COST + DV*SINT)
       ROT = 3. * (DV*COST - DU*SINT) * XLI
       DMI = C1 * (2.*B(K,(3*I)) + B(K,(3*J)) - ROT)
       DMJ = C1 * (2.*B(K,(3*J)) + B(K,(3*I)) - ROT)
       DV  = (DMI + DMJ) * XLI
C
C         MODIFY END FORCES FOR MEMBER TYPE
C
       KK = MKODE(M)
       IF(KK.EQ.0) GO TO 400
       GO TO (100,200,300),KK
100    DMJ = DMJ - DMI*.5
       DV  = DV  - 1.5*DMI*XLI
       DMI = 0.0
       GO TO 400
200    DMI = DMI - 0.5*DMJ
       DV  = DV  - 1.5*DMJ*XLI
       DMJ = 0.0
       GO TO 400
300    DV  = DV - (DMI + DMJ)*XLI
       DMI = 0.0
       DMJ = 0.0
C
400    MM = 6*M
       FEF2(K,MM-5) = FEF2(K,MM-5) - DP
```

```
      FEF2(K,MM-4) = FEF2(K,MM-4) + DV
      FEF2(K,MM-3) =(FEF2(K,MM-3) + DMI)/UNIT
      FEF2(K,MM-2) = FEF2(K,MM-2) + DP
      FEF2(K,MM-1) = FEF2(K,MM-1) - DV
      FEF2(K,MM)   =(FEF2(K,MM)   + DMJ)/UNIT
500   CONTINUE
C
      WRITE(IO,2000)
      DO 600 M=1,NE
      M2 = 6 * M
      M1 = M2 - 5
      WRITE(IO,2100) M,NOD(1,M),NOD(2,M),(FEF2(K,I),I=M1,M2)
600   CONTINUE
C
      RETURN
C
C        FORMAT STATEMENTS
C
2000  FORMAT (///,'MEMBER END FORCES',//,3X,'MEM',3X,'I',4X,
     *'J',7X,'NI',13X,'VI',13X,'MI',13X,'NJ',13X,'VJ',13X,
     *'MJ'/)
2100  FORMAT (3I5,6D15.6)
C
      END
```

```
C**********************************************************
C
            SUBROUTINE EIGV (A,AG,TT,VO,MAXA,NN,NWA,ILOAD,
     *                        TOL,ITLIM,IFPR,IREF,IR,IN,IO)
C           ***************
C
C             THIS SUBROUTINE CALCULATES THE SMALLEST EIGENVALUE
C       FOR THE EQUATION  AX = (LANDA)BX  BY INVERSE ITERATION
C       USING CHOLESKY DECOMPOSITION.
C
C       *****************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION A(1),AG(1),TT(1),VO(1),MAXA(1)
C
        IT = 0
        EOLD = 0.0
        IF(IREF.EQ.1) IT=1
100     REWIND 2
        CALL RTRV2 (2,AG,NWA)
        CALL RTRV2 (2,A,NWA)
        CALL MULT1 (TT,AG,VO,MAXA,NN,NWA)
        CALL EQSBST (A,TT,MAXA,NN,1,ILOAD,IT)
        CALL BKSB1 (TT,A,MAXA,NN,1,0)
        IF(IT.GT.0) GO TO 150
        REWIND 2
        CALL RTRV2 (2,AG,NWA)
        CALL STORE2 (2,A,NWA)
150     K = 0
        ABMAX = 0.0
        DO 200 I=1,NN
        C = TT(I)
        IF(DABS(C).LT.ABMAX) GO TO 200
        K = I
        ABMAX = DABS(C)
200     CONTINUE
        EIGV = TT(K)
        C = 1. / EIGV
        DIF = 0.0
        DO 300 I=1,NN
        VS = VO(I)
        VO(I) = TT(I) * C
        DIF = DIF + DABS(VS-VO(I))
300     CONTINUE
        DIFE = DABS(EIGV-EOLD)
        EOLD = EIGV
        IT = IT + 1
        IF(IFPR.EQ.0) GO TO 40
        WRITE(IO,2000) IT,DIF,EIGV,C
        WRITE(IO,2100) (VO(I),I=1,NN)
40      IF(DIF.LT.TOL) GO TO 50
        IF(DIFE.LT.TOL) GO TO 50
        IF(IT.GT.ITLIM) GO TO 999
```

```
      GO TO 100
50    WRITE(IO,2200) EIGV,C,(VO(I),I=1,NN)
      IR = 0
C
      RETURN
C
999   WRITE(IO,2300) IT,EIGV,C
      IR = 1
      RETURN
C
C         FORMAT STATEMENTS
C
2000  FORMAT(//,5X,'ITERATE',2X,I3,3X,'HAS BEEN COMPLETED'
     *,2X,' WITH TOLERANCE OF',2X,E12.4//,5X,
     *'EIGENVALUE = ',E12.5,5X,'LOAD FACTOR =',F12.6,//,
     *5X,'THE EIGENVECTOR IS :',/)
2100  FORMAT (5X,8F10.6)
2200  FORMAT(//,5X,'EIGENVALUE  =',E12.5,5X,'LOAD FACTOR =',
     *F12.6//,5X,'AND THE VECOR IS : '/(5X,8E12.4))
2300  FORMAT ('THE NO. OF ITERATES HAS REACHED ',I5//,
     *'EIGENVALUE  = ',E12.5,5X,'LOAD FACTOR =',F12.6)
C
      END
```

E.2 Program NONELA

Program NONELA consists of the following parts.

1. MAIN
2. MAINMG, the main executive subroutine
3. The input package
   (a) INPUT1 and INPUT2
   (b) BOUND, JLOAD2, and MLOAD2.
4. The data managing package
   ISPAC, LOCOM, REMOV, REMOV2, and BLOCK DATA.
5. The data storage and retrieval package CLEAR, ICLEAR, RTRV1, RTRV2, STORE1, and STORE2.
6. The equation solving package
   (a) ADDRES and COLHT
   (b) EQSBST and BKSB1
7. The formulation and output package
   (a) ASSEMB, BOUND2, DISPL1, SHAPE, DINCR, and CONVER.
   (b) STIFF, STIFFB, TR, and TR1

The listing for packages 1, 2, 3a, 6b, and 7b follows. The listing for parts 3b, 4, 5, 6a and 7a can be found in Sects. E.8, E.6, E.7, E.9, and E.10, respectively. Descriptions of the subroutines can be found in the listing.

Five files (line-files) are used in this program exclusively in sequential I/O operations. FILES 1, 2, and 3 are temporary files while FILES 9 and 10 are permanent. FILE9 contains the present trial vector while FILE10 contains the last converged-at solution.

```
C*****************************************************
C
C                     NONELA
C                     ******
C
C          THIS IS A PROGRAM FOR ELASTIC STABILITY ANALYSIS
C       OF PLANER FRAMED TYPE STRUCTURES. THE PROGRAM SOLVE
C       FOR LOWEST EIGENVALUE AND/OR EVALUATES THE NONLINEAR
C       RESPONSE OF THE STRUCTURE.
C
C       *****************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 NAMES
C
      COMMON/PROBIA/ NNN(200)
      COMMON/IA1/    MMM(200)
      COMMON/RA1/    AAA(4000)
      COMMON/RA2/    BBB(1000)
      COMMON/RA3/    CCC(1000)
      COMMON/DIMCOM/LAST1,LAST2,LAST3,LAST4,LAST5,MXDIM,
     *NAMES(5,20),IPT(5,21),ICOM(5)
C
C     *****************************************************
C
      ICOM(1) = 200
      ICOM(2) = 4000
      ICOM(3) = 200
      ICOM(4) = 1000
      ICOM(5) = 1000
C
      CALL MAINMG
C
      END
```

```
C*****************************************************************
C
                SUBROUTINE MAINMG
C               *****************
C
C          THIS IS THE MAIN MANAGER OF PROGRAM NONELA. IT
C        CONTROLS THE READING , THE FORMULATION ,THE SOLUTION
C        AND THE OUTPUT OF THE RESULTS.
C
C        *************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 NAMES,NAME
C
      COMMON/PROBCV/ UNIT,FL1,FL2,TOL,NJ,NE,NFE,NMT,NEBEL,
     *IN,IO,ITLIM,IREF,IRCO
C
      COMMON/PROBIA/ NNN(1)
      COMMON/IA1   / MMM(1)
      COMMON/RA1   / AAA(1)
      COMMON/RA2   / BBB(1)
      COMMON/RA3   / CCC(1)
      COMMON/DIMCOM/ LA1,LA2,LA3,LA4,LA5,MXDIM,NAMES(5,20)
     *,IPT(5,21),ICOM(5)
C
      DIMENSION XGAUS(6,4)
C
      DATA XGAUS/-1.0,0.0,1.0,3*0.,-1.0,-0.5773503,0.5773503
     *,1.0,2*0.0,-1.0,-0.7745967,0.0,0.7745967,1.0,0.0,
     *-1.0,-.86113631,-.33998104,.33998104,.86113631,1.0/
C
C     ***************************************************
C
      CALL TIME (0,0)
      IN = 5
      IO = 6
C
      REWIND 1
      REWIND 2
      REWIND 9
      REWIND 10
C
C   READ PROBLEM CONTROL VARIABLES
C   ------------------------------C
C
      CALL INPUT1
C
      IF(IREF.EQ.0) GO TO 40
C
      READ (10) NJ,NFE,NE,IDOF,NGAUS,NMT,NEQ,INN,NEBEL,
     *ITLIM,TOL,UNIT
      READ (10) L1,L2
      CALL RTRV2 (10,BBB(L1),NEQ)
      CALL RTRV2 (10,BBB(L2),NEQ)
```

```
      DO 10 I=1,NEQ
10    BBB(I) = BBB(I) * FL1 + BBB(I+NEQ) * FL2
      CALL STORE2 (1,BBB(L1),NEQ)
      READ (10) NWA
      READ (10) I1,I3,K1,K2,II1,II3
      READ (10) J1,J2,J3,J4,J5,J6,J7,J8,J9,J10,II2
      READ (10) L1,L2,L3,M0,M1,M2,M3,II5
      CALL RTRV1 (10,NNN(1),II1)
      CALL RTRV1 (10,MMM(1),II3)
      CALL RTRV2 (10,AAA(1),II2)
      CALL RTRV2 (10,CCC(1),II5)
C
      DO 12 I=1,INN
12    AAA(J8+I-1) = AAA(J8+I-1) * FL1 + AAA(J9+I-1) * FL2
C
      READ (9) LI
      IRCO = IRCO + 1
      GO TO (15,20) ,IRCO
15    CALL RTRV2 (9,BBB(L2),NEQ)
      LI = LI + 1
      CALL STORE2 (10,BBB(L2),NEQ)
      GO TO 25
20    CALL RTRV2 (10,BBB(L2),NEQ)
C
25    WRITE(IO,2000) LI
      WRITE(IO,2100)
      DO 30 N=1,NJ
      N5 = N * 5
      N1 = N5 - 4
30    WRITE(IO,2200) N,(BBB(NN),NN=N1,N5)
C
      CALL STORE2 (2,BBB(L2),NEQ)
      GO TO 450
C
40    NODE  = 2
      IDOF  = 5
      NGAUS = 4
      NEQ   = NJ * IDOF
      LI    = 1
      INN   = IDOF * NE * NODE
C
      CALL CLEAR (NAMES(1,1),100)
C
C  READ NODAL GEOMETRY AND MEMBER PROPER. AND CONNECTIVITIES
C  --------------------------------------------------------C
      J1 = ISPAC (1HX,NJ,2)
      J2 = ISPAC (1HY,NJ,2)
      J3 = ISPAC (4HAREA,NE,2)
      J4 = ISPAC (2HRI,NFE,2)
      J5 = ISPAC (3HFMA,NFE,2)
      J6 = ISPAC (4HYMOD,NE,2)
      I1 = ISPAC (3HNOD,(NE*NODE),1)
      I2 = ISPAC (5HMKODE,NE,1)
C
```

```
      CALL INPUT 2 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J5),
     *AAA(J6),NNN(I1))
C
      CALL ICLEAR (NNN(I2),NE)
C
C    READ EXTERNAL BOUNDARY CONDITIONS
C
      J7 = ISPAC (3HBES,NEBEL,2)
      K1 = ISPAC (3HNPB,NEBEL,3)
      K2 = ISPAC (4HKODE,NEBEL,3)
      CALL BOUND (AAA(J7),MMM(K1),MMM(K2),NEBEL,IN,IO)
C
      WRITE(10) NJ,NFE,NE,IDOF,NGAUS,NMT,NEQ,INN,NEBEL,
     *ITLIM,TOL,UNIT
C
      J8 = ISPAC (4HFEFC,INN,2)
      J9 = ISPAC (4HFEFV,INN,2)
      CALL CLEAR (AAA(J8),(INN*2))
C
      L1 = ISPAC (2HBC,NEQ,4)
      L2 = ISPAC (2HBV,NEQ,4)
      CALL CLEAR (BBB(L1),(NEQ*2))
C
C    READ LOADS IF ANY
C
      READ(IN,1000) JLF,MLF
      IF(JLF.EQ.0) GO TO 200
      CALL JLOAD2 (BBB(L1),BBB(L2),AAA(J7),IDOF,IN,IO)
200   IF(MLF.EQ.0) GO TO 250
      CALL MLOAD2 (AAA(J1),AAA(J2),AAA(J8),AAA(J9),BBB(L1),
     *BBB(L2),NNN(I1),NNN(I2),UNIT,IDOF,NJ,IN,IO)
C
      DO 220 I=1,INN
220   AAA(J8+I-1) = AAA(J8+I-1) + AAA(J9+I-1)
C
250   CONTINUE
      WRITE(10) L1,L2
      CALL STORE2 (10,BBB(L1),NEQ)
      CALL STORE2 (10,BBB(L2),NEQ)
      DO 300 I=1,NEQ
300   BBB(I) = BBB(I) + BBB(I+NEQ)
      CALL REMOV (2HBV,4)
      CALL STORE2 (1,BBB(L1),NEQ)
C
C    FORM COLUMN HEIGHTS AND ADDRESSING ARRAY
C
      I3 = ISPAC (4HMAXA,(NEQ+1),1)
      I4 = ISPAC (3HMHT,NEQ,1)
      CALL ICLEAR (NNN(I4),NEQ)
      DO 350 I=1,NE
      CALL COLHT (NODE,(NODE*IDOF),IDOF,I,NNN(I4),
     *NNN(I1))
350   CONTINUE
C
```

```
      CALL ADDRES (NNN(I3),NNN(I4),NEQ,NWA)
      CALL REMOV (3HMHT,1)
C   RESERVE SPACE FOR DERIVATIVES OF SHAPE VECTOR
C
      NGAUS2 = NGAUS + 2
      M0 = ISPAC (3HPHI,(4*NGAUS2),5)
      M1 = ISPAC (4HPHIP,(4*NGAUS2),5)
      M2 = ISPAC (5HPHIDP,(4*NGAUS2),5)
C
C   RESERVE AND CLEAR SPACE FOR TANGENT STIFFNESS MATRIX
C
      M3 = ISPAC (2HTK,NWA,5)
C
      DO 400 IG=1,NGAUS2
      XI = XGAUS (IG,NGAUS)
      CALL SHAPE (XI,CCC(M0),CCC(M1),CCC(M2),IG)
400   CONTINUE
C
      L2 = ISPAC (4HDISP,NEQ,4)
      L3 = ISPAC (5HTDISP,NEQ,4)
      CALL CLEAR (BBB(L2),NEQ)
      J10 = ISPAC (4HFEF2,(IDOF*NODE*NE),2)
450   IT = 0
      IF = 1
C   FORM AND ASSEMBLE TANGENT STIFFNESS MATRIX
C
600   CALL CLEAR (CCC(M3),NWA)
      CALL STIFF (NNN(I1),NNN(I3),AAA(J1),AAA(J2),AAA(J3),
     *AAA(J4),AAA(J5),AAA(J6),CCC(M0),CCC(M1),CCC(M2),
     *CCC(M3),BBB(L1),BBB(L2),NFE,IDOF,UNIT,NGAUS,NEQ,NJ,IO)
      IF(NMT.EQ.1) GO TO 650
      CALL STIFFB (NNN(I1),NNN(I3),AAA(J1),AAA(J2),
     *AAA(J3),AAA(J6),CCC(M3),BBB(L1),BBB(L2),NE,NFE,
     *IDOF,UNIT,NJ,IO)
C
C
C  ADD EXTERNAL BOUNDARY CONDITIONS
C
650   CALL BOUND2 (CCC(M3),BBB(L1),NNN(I3),MMM(K1),MMM(K2),
     *AAA(J7),NJ,NEBEL,IO)
C
      IF(IT.GT.1) CALL CONVER (BBB(L1),C,CONVD,IT,TOL,
     *ITLIM,NEQ,IF,IO)
      IF(IF.EQ.0) GO TO 800
C
      CALL EQSBST (CCC(M3),BBB(L1),BBB(L1),NNN(I3),
     *NEQ,1,1,0)
      CALL BKSB1 (BBB(L1),BBB(L1),CCC(M3),NNN(I3),NEQ,1,0)
      CALL DISPL1 (AAA(J10),BBB(L1),AAA(J8),NJ,NE,IDOF,
     *IN,IO)
      REWIND 3
      CALL STORE2 (3,BBB(L1),NEQ)
      REWIND 1
      REWIND 3
```

```
      CALL RTRV2 (1,BBB(L1),NEQ)
      CALL RTRV2 (3,BBB(L2),NEQ)
      IT = IT + 1
      CALL TIME(3,3)
      IF(IT.GT.1.OR.IREF.EQ.1) GO TO 700
      CALL STORE2(2,BBB(L2),NEQ)
      GO TO 600
700   REWIND 2
      CALL RTRV2 (2,BBB(L3),NEQ)
      CALL DINCR (BBB(L1),BBB(L2),BBB(L3),NEQ,IT,C,CONVD,
     *NJ,IO)
      REWIND 2
      CALL STORE2 (2,BBB(L2),NEQ)
      GO TO 600
800   REWIND 9
      WRITE(9) LI
      CALL STORE2 (9,BBB(L2),NEQ)
      IF(LI.GT.1) GO TO 900
C
      II1 = IPT (1,LA1+1) - 1
      II2 = IPT (2,LA2+1) - 1
      II3 = IPT (3,LA3+1) - 1
      II5 = IPT (5,LA5+1) - 1
C
      WRITE (10) NWA
      WRITE (10) I1,I3,K1,K2,II1,II3
      WRITE (10) J1,J2,J3,J4,J5,J6,J7,J8,J9,J10,II2
      WRITE (10) L1,L2,L3,M0,M1,M2,M3,II5
      CALL STORE1 (10,NNN(1),II1)
      CALL STORE1 (10,MMM(1),II3)
      CALL STORE2 (10,AAA(1),II2)
      CALL STORE2 (10,CCC(1),II5)
900   RETURN
C
1000  FORMAT (2I5)
2000  FORMAT (10X,'LOAD INCREMENT NO. ',I5//)
2100  FORMAT ('NODAL LOADS',//,4X,'N',7X,'FU',13X,'FV',
     *13X,'FR',12X,'FDU',12X,'FDV'/)
2200  FORMAT (I4,5D15.6/)
C
      END
```

```
C*************************************************************
C
                  SUBROUTINE INPUT1
C                 *****************
C
C           THIS SUBROUTINE READS THE PROBLEM CONTROL VARIABLES
C         FOR PROGRAM NONELA.
C
C         ****************************************************
C
          IMPLICIT REAL*8(A-H,O-Z)
C
          COMMON/PROBCV/ UNIT,FL1,FL2,TOL,NJ,NE,NFE,NMT,NEBEL,
         *IN,IO,ITLIM,IREF,IRCO
C
          DIMENSION HEAD(20)
C
          READ (IN,1000) HEAD
          WRITE(IO,2000) HEAD
          READ (IN,1100) IREF,IRCO,FL1,FL2
          IF(IREF.EQ.1) RETURN
          READ (IN,1200) NJ,NFE,NE,NEBEL,NMT,ITLIM,TOL,UNIT
          WRITE(IO,2100) NJ,NFE,NE,NEBEL,NMT,ITLIM,TOL,UNIT
C
          RETURN
C
C       FORMAT STATEMENTS
C
1000      FORMAT (20A4)
1100      FORMAT (2I5,2F12.0)
1200      FORMAT (6I4,2F12.0)
2000      FORMAT (//,30X,20A4//,30X,20(1H*)////)
2100      FORMAT (5X,'PROBLEM CONTROL VARIABLES'/,5X,25(1H*)//,
         *10X,'NUMBER OF JOINTS                     =',I5//,
         *10X,'NUMBER OF FLEXURAL ELEMENTS          =',I5//,
         *10X,'NUMBER OF ELEMENTS                   =',I5//,
         *10X,'NO. OF EXTERNAL BOUNDARY ELEMENTS    =',I5//,
         *10X,'NUMBER OF DIFFERENT MEMBER TYPES     =',I5//,
         *10X,'MAXIMUM NO. OF ITERATIONS            =',I5//,
         *10X,'TOLERANCE LIMIT                      =',F12.6//,
         *10X,'LENGTH CONVERGENCE FACTOR            =',F12.6//,
         *10X,50(1H*))
C
          END
```

```
C*****************************************************************
C
               SUBROUTINE INPUT 2 (X,Y,AREA,RI,FMA,YMOD,NOD)
C              ******************
C
C            THIS SUBROUTINE READS THE NODAL GEOMETRY AND
C        MEMBER PROPERTIES FOR PROGRAM NONELA
C
C        *****************************************************
C
         IMPLICIT REAL*8(A-H,O-Z)
C
         COMMON/PROBCV/ UNIT,FL1,FL2,TOL,NJ,NE,NFE,NMT,NEBEL,
        *IN,IO,ITLIM,IREF,IRCO
         DIMENSION X(1),Y(1),AREA(1),RI(1),FMA(1),YMOD(1),
        *NOD(2,1)
C
         WRITE(IO,2000)
50       READ (IN,1000) N,X(N),Y(N),INC
         IF(INC.EQ.0) GO TO 200
         NINT = (N-NOLD) / INC
         RN   = NINT
         IF(RN.LT.FLOAT(N-NOLD)/FLOAT(INC)-0.001) GO TO 999
         DX = (X(N)-X(NOLD))/RN
         DY = (Y(N)-Y(NOLD))/RN
         L = NOLD
         M = NINT - 1
         DO 100 J=1,M
         LL = L + INC
         X(LL) = X(L) + DX
         Y(LL) = Y(L) + DY
         L = LL
100      CONTINUE
200      WRITE (IO,2100) N,X(N),Y(N),INC
         NOLD = N
         IF(N.LT.NJ) GO TO 50
C
         READ (IN,1100) ADEF,RDEF,A4MDEF,YDEF
         WRITE(IO,2200) ADEF,RDEF,A4MDEF,YDEF
C
         WRITE(IO,2300)
300      READ (IN,1200) M,(NOD(I,M),I=1,2),INC,
        *NOD1,NOD2,AREA(M),RI(M),FMA(M),YMOD(M)
         IF(AREA(M).EQ.0) AREA(M) = ADEF
         IF(RI(M).EQ.0)   RI(M)   = RDEF
         IF(FMA(M).EQ.0)  FMA(M)   =A4MDEF
         IF(YMOD(M).EQ.0) YMOD(M) = YDEF
         IF(INC.EQ.0) GO TO 500
         NINT = (M-MOLD) / INC - 1
         L = MOLD
         DO 400 I=1,NINT
         LL         =L + INC
         AREA(LL)   = AREA(M)
         RI(LL)     = RI(M)
```

```
      FMA(LL)    = FMA(M)
      YMOD(LL)   = YMOD(M)
      NOD(1,LL) = NOD(1,L) + NOD1
      NOD(2,LL) = NOD(2,L) + NOD2
400   L = LL
C
500   WRITE(IO,2400) M,NOD(1,M),NOD(2,M),INC,
     *NOD1,NOD2,AREA(M),RI(M),FMA(M),YMOD(M)
      MOLD = M
      IF(M.LT.NFE) GO TO 300
      IF(NMT.EQ.1) GO TO 550
      READ (IN,1400) ADEFB,YDEFB
      WRITE(IO,2500) ADEFB,YDEFB
      WRITE(IO,2600)
520   READ (IN,1500) M,(NOD(I,M),I=1,2),INC,NOD1,NOD2,
     *AREA(M),YMOD(M)
      IF(AREA(M).EQ.0) AREA(M) = ADEFB
      IF(YMOD(M).EQ.0) YMOD(M) = YDEFB
      IF(INC.EQ..0) GO TO 540
      NINT = (M-MOLD) / INC - 1
      L = MOLD
      DO 530 I=1,NINT
      LL = L + INC
      AREA(LL) = AREA(M)
      YMOD(LL) = YMOD(M)
      NOD(1,LL) = NOD(1,L) + NOD1
      NOD(2,LL) = NOD(2,L) + NOD2
530   L = LL
540   WRITE(IO,2650) M,NOD(1,M),NOD(2,M),INC,NOD1,NOD2,
     *AREA(M),YMOD(M)
      MOLD = M
      IF(M.LT.NE) GO TO 520
550   READ (IN,1300) K
      IF(K.EQ.0) GO TO 600
      WRITE(IO,2700)
      WRITE(IO,2750) (N,X(N),Y(N),N=1,NJ)
      WRITE(IO,2800)
      WRITE(IO,2850) (M,(NOD(I,M),I=1,2),
     *AREA(M),RI(M),FMA(M),YMOD(M),M=1,NE)
C
600   RETURN
C
999   WRITE (IO,9999) N
9999  FORMAT('NODAL GEOMETRY DATA INPUT ERROR',I5)
      STOP
C
C        FORMAT STATEMENTS
C
1000  FORMAT(I5,2F12.0,I5)
1100  FORMAT(4F12.0)
1200  FORMAT(6I4,4F12.0)
1300  FORMAT(I5)
1400  FORMAT (2F12.0)
1500  FORMAT (6I4,2F12.0)
```

```
2000  FORMAT (///,5X,'NODAL GEOMETRY DATA AS INPUT'/,5X,
     *28(1H*)//,10X,1HN,7X,1HX,14X,1HY,9X,3HINC/)
2100  FORMAT (6X,I5,2D15.6,I5)
2200  FORMAT (///,5X,'FLEXURAL MEMBER PROP. DEFAULT VALUES'
     */,5X,36(1H*)//,10X,'AREA',16X,' =',D15.6,//,
     *10X,'M. INERTIA',11X,'=',D15.6,//,
     *10X,'FOURTH MOM. OF AREA',2X,'=',D15.6,//,
     *10X,'Y. MODULUS',11X,'=',D15.6)
2300  FORMAT (///,5X,'FLEXURAL MEMBER DATA '/,5X,20(1H*)//
     *,9X,1HM,4X,1HI,4X,1HJ,3X,3HINC,1X,4HINCI,
     *1X,4HINCJ,5X,4HAREA,13X,1HI,14X,2HI4,12X,4HYMOD/)
2400  FORMAT (5X,6I5,4D15.6)
2500  FORMAT (///,5X,'BRACING MEMBER PROP. DEFAULT VALUES'
     */,5X,40(1H*)//,10X,'AREA              =',D15.6//,
     *10X,'YOUNG MODULUS   =',D15.6)
2600  FORMAT (///,5X,'BRACING MEMBERS DATA'/,5X,30(1H*)//,
     *9X,1HM,4X,1HI,4X,1HJ,2X,3HINC,1X,4HINCI,1X,
     *4HINCJ,6X,4HAREA,12X,4HYMOD/)
2650  FORMAT (5X,6I5,2D15.6)
2700  FORMAT (///,5X,'COMPLETED NODAL GEOMETRY DATA'/,5X,
     *30(1H*)//,4X,1HN,7X,1HX,14X,1HY/)
2750  FORMAT (I5,2D15.6)
2800  FORMAT (///,5X,'COMPLETED MEMBER DATA'/,5X,21(1H*)//
     *,4X,1HM,4X,1HI,4X,1HJ,5X,4HAREA,10X,1HI,10X,
     *2HI4,13X,4HYMOD/)
2850  FORMAT (3I5,4D15.6)
C
      END
```

```
C*************************************************************
C
            SUBROUTINE EQSBST (A,BC,BV,MAXA,NB,NLC,ILOAD,IT)
C           ****************
C
C              THIS SUBROUTINE TRIANGULARIZES A STIFFNESS MATRIX
C        STORED COLUMNWISE UNDER A SKYLINE AND REDUCES THE
C        CORRESPONDING LOAD VECTOR, DOWN TO EQUATION NB.
C
C        *****************************************************
C
         IMPLICIT REAL*8(A-H,O-Z)
C
         DIMENSION A(1),BC(1),BV(1),MAXA(1)
C
C
         DO  1000 N=1,NB
         KN = MAXA(N)
         KL = KN + 1
         KU = MAXA(N+1) - 1
         KH = KU - KL
         IF(KH) 900,500,100
100      IF(ILOAD.EQ.2.AND.IT.GT.0) GO TO 650
         K  = N - KH
         IC = 0
         KLT= KU
         DO 400 J=1,KH
         IC = IC + 1
         KLT= KLT - 1
         KI = MAXA(K)
         ND = MAXA(K+1) - KI - 1
         IF(ND) 400,400,200
200      KK = IC
         IF(KK.GT.ND) KK = ND
         C  = 0.0
         DO 300 L = 1,KK
300      C  = C + A(KI+L)*A(KLT+L)
         A(KLT) = A(KLT) - C
400      K  = K + 1
500      IF(ILOAD.EQ.2.AND.IT.GT.0) GO TO 650
         K  = N
C
         C  = 0.0
         DO 600 KK = KL,KU
         K  = K - 1
         KI = MAXA(K)
         D  = A(KK)/A(KI)
         C  = C + D*A(KK)
         A(KK) = D
600      CONTINUE
         A(KN) = A(KN) - C
C
650      CONTINUE
         K = N
```

```
      C = 0.0
      DO 700 KK=KL,KU
      K = K - 1
700   C = C + A(KK) * BV(K)
      BV(N) = BV(N) - C
      IF(NLC.EQ.1) GO TO 900
      K = N
      C1 = 0.0
      DO 800 KK=KL,KU
      K = K - 1
800   C1 = C1 + A(KK) * BC(K)
      BC(N) = BC(N) - C1
C
900   IF(A(KN)) 950,950,1000
950   WRITE(6,3000) N,A(KN)
      STOP
C
1000  CONTINUE
C
      RETURN
C
3000  FORMAT('ZERO OR NEGATIVE ELEMENT ON MAIN DIAGONAL NO.'
     *,I4,D15.6)
C
      END
```

```
C*****************************************************
C
            SUBROUTINE BKSB1 (BC,BV,A,MAXA,NN,NLC,KB)
C           ***************
C
C         THIS SUBROUTINE IS A PART OF THE PARTIAL REDUCTION
C      PACKAGE. IT FORMS THE LAST STAGE OF BACKSUBSTITUTION,
C      THE OPERATION (L(-1)(T)*RI*) .   IN CASE OF AN
C      UNSUBSTRUCTUREDPROBLEM IT PERFORMS THE OPERATION
C      //D(-1)*L(-1)(T)*R//
C
C      *****************************************************
C
       IMPLICIT REAL*8(A-H,O-Z)
C
       DIMENSION A(1),BC(1),BV(1),MAXA(1)
C
       IF(KB.NE.0) GO TO 200
       DO 100 J=1,NN
       K = MAXA(J)
       BV(J) = BV(J) / A(K)
       IF(NLC.EQ.1) GO TO 100
       BC(J) = BC(J) / A(K)
100    CONTINUE
200    N = NN
       DO 600 L=2,NN
       KL = MAXA(N) + 1
       KU = MAXA(N+1) - 1
       IF(KU-KL) 600,300,300
300    IF(BV(N).EQ.0) GO TO 500
       K = N
       DO 400 KK=KL,KU
       K = K - 1
400    BV(K) = BV(K) - A(KK) * BV(N)
500    IF(NLC.EQ.1) GO TO 600
       IF(BC(N).EQ.0) GO TO 600
       K = N
       DO 550 KK=KL,KU
       K = K - 1
550    BC(K) = BC(K) - A(KK) * BC(N)
600    N = N - 1
C
       RETURN
C
       END
```

```
C*****************************************************
C
            SUBROUTINE STIFF (NOD,MAXA,X,Y,AREA,RI,FMA,YMOD,
     *                     PHI,PHIP,PHIDP,TK,B,DISP,NE,IDO,
     *                                 UNIT,NGAUS,ND,NJ,IO)
C
C            THIS SUBROUTINE FORMS THE TANGENT STIFFNESS
C         MATRIX OF 10 D.O.F. PLANE FRAME MEMBER.
C
C        ***********************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION NOD(2,1),MAXA(1),X(1),Y(1),AREA(1),RI(1),
     *FMA(1),YMOD(1),PHI(4,1),PHIP(4,1),PHIDP(4,1),TK(1),
     *B(1),DISP(1),D(10),DM(8),PHIL(4),PHIPL(4),PHIDPL(4),
     *PHITP(4),DIP(8),DIM(8),DIMS(8),PHIP1(8),PHIP2(8),
     *PHID1(8),BB(8),S(8,8),WGAUS(6,4),T(8,10),TEMP(8,10),
     *SS(10,10),A(55),LM(10)
C
      DATA WGAUS/0.0,2.0,5*0.0,2*1.0,4*0.0,0.5555556,
     *0.8888889,0.5555556,3*0.0,0.34785485,0.65214515,
     *0.65214515,0.34785485,0.0/
C
C     ***************************************************
C
      WRITE (IO,2200)
C
      DO 900 M=1,NE
      I   = NOD(1,M)
      J   = NOD(2,M)
      III = I
      JJJ = J
      DX  = (X(J) - X(I)) * UNIT
      DY  = (Y(J) - Y(I)) * UNIT
      XL  = DSQRT (DX**2 + DY**2)
      XLI = 1. / XL
      CO  = DX * XLI
      SI  = DY * XLI
      CL  = CO * XLI
      SL  = SI * XLI
      CL2 = CL ** 2
      SL2 = SL ** 2
      CSL = CL * SL
      C2  = CO ** 2
      S2  = SI ** 2
      SC  = CO * SI
      YA  = YMOD(M) * AREA(M)
      YR  = YMOD(M) * RI(M)
      YR4 = YMOD(M) * FMA(M)
      DO 50 KK=1,IDO
      LM(KK)     = IDO * I - IDO + KK
50    LM(KK+IDO) = IDO * J - IDO + KK
C
```

```
      II = (I-1) * 5
      JJ = (J-1) * 5
      DO 100 I=1,5
      J  = II + I
      J1 = JJ + I
      D(I)   = DISP(J)
      D(I+5) = DISP(J1)
100   CONTINUE
      CALL TR (D,CO,SI,SC,C2,S2,DM)
C
      DO 150 I=1,8
      DO 150 J=1,8
      S(I,J) = 0.0
150   CONTINUE
C
      DO 175 I=1,8
175   BB(I) = 0.0
C
      PHITP(1) = 12. * XLI**3.
      PHITP(2) = 6. * XLI * XLI
      PHITP(3) = -12. * XLI**3.
      PHITP(4) = 6. * XLI * XLI
      VTP = 0.0
      DO 180 I=1,4
180   VTP = VTP + PHITP(I) * DM(I+4)
      NGAUS2 = NGAUS + 2
      DO 600 IG=1,NGAUS2
      WG = 0.5 * XL * WGAUS(IG,NGAUS)
      DO 200 I=1,4
      IF(I.EQ.1.OR.I.EQ.3) C1 = XLI
      PHIL(I)   = PHI(I,IG)  * 1./(XLI-C1+1.)
      PHIPL(I)  = PHIP(I,IG) * C1
      PHIDPL(I) = PHIDP(I,IG) * XLI * C1
      C1 = 1.0
200   CONTINUE
      U   = 0.0
      V   = 0.0
      UP  = 0.0
      UDP = 0.0
      VP  = 0.0
      VDP = 0.0
C   AT GAUSS POINT EVALUATE U',V',V"
      DO 300 I=1,4
      U   = U   + PHIL(I)  * DM(I)
      V   = V   + PHIL(I)  * DM(I+4)
      UP  = UP  + PHIPL(I) * DM(I)
      UDP = UDP + PHIDPL(I)* DM(I)
      VP  = VP  + PHIPL(I) * DM(I+4)
      VDP = VDP + PHIDPL(I)* DM(I+4)
300   CONTINUE
C
      UP2  = UP * UP
      VP2  = VP * VP
      VDP2 = VDP * VDP
```

```
      VPI  = 1. / (1-VP2)
      RVPI = DSQRT (VPI)
      VPVD = VP * VDP
      A1   = 1 + UP
      A2   = UP + 0.5*(UP2+VP2)
      A3   = 1. + VP2 * VPI
      A4   = VDP * A3
      A5   = A3 * VDP2
      A6   = A1 + VP2*RVPI
      A7   = VDP * A6
      A8   = A3 + 1.
      A9   = VP * RVPI * A8
      A10  = VPVD * VPI * A3
      A11  = 2. * A4 * RVPI
      A12  = A9 * VDP
      A13  = 3. * A10 * RVPI * VP
      A14  = A5 * VPI
      A15  = 4. * A10 * VPVD * VPI
      A16  = VTP * A1
      A17  = VDP * UDP
      A18  = A12 * VDP
      A19  = RVPI * VP2 * VTP
      A20  = A1 + VP2
      A20  = DSQRT(A20)
      A20 = 1. / A20
C
C
      PP = YA * A2 + 0.5 * YR * A5
      OM = -YR * A7
      SM = YR * A2 + 0.5 * YR4 * A5
      VV  = -YR * (A16 + A17 + A18 + A19) * A20
      IF(IG.EQ.1) GO TO 350
      GO TO 375
C
350   P1  = PP
      OM1 = OM
      SM1 = SM
C
375   IF(IG.EQ.1.OR.IG.EQ.NGAUS2) GO TO 600
C
      FE1 = PP * A1 - OM * VDP
      FE2 = PP * VP + VDP * (-OM*A9 + SM*A10)
      FE3 = -OM * A6 + SM * A4
C
      DO 400 I=1,4
      BB(I)  = BB(I)   + WG * FE1 * PHIPL(I)
      BB(I+4)= BB(I+4) + WG *(FE2*PHIPL(I) + FE3*PHIDPL(I))
C
      DIP(I)   = YA * A1 * PHIPL(I)
      DIP(I+4) = YA * VP * PHIPL(I)
     *            + YR * VDP * (A10*PHIPL(I) + A3*PHIDPL(I))
      DIM(I)   = -YR * VDP * PHIPL(I)
      DIM(I+4) = -YR * (A12 * PHIPL(I) + A6 * PHIDPL(I))
      DIMS(I)   = YR * A1 * PHIPL(I)
```

```
      DIMS(I+4) = YR * VP * PHIPL(I)
     *            +YR4 * VDP * ( A10*PHIPL(I) + A3*PHIDPL(I))
      PHIP1(I)   = PHIPL(I)
      PHIP1(I+4) = 0.0
      PHIP2(I)   = 0.0
      PHIP2(I+4) = PHIPL(I)
      PHID1(I)   = 0.0
      PHID1(I+4) = PHIDPL(I)
400   CONTINUE
C
C   FORM ELEMENT STIFFNESS MATRIX
C
      DO 500 I=1,4
      DO 500 J=1,8
      S(I,J)    = S(I,J)   +WG * PHIPL(I) *
     *                     ( A1 * DIP(J) - VDP * DIM(J)
     *                     + PP * PHIP1(J) - OM * PHID1(J))
      S(I+4,J)= S(I+4,J)
     *          + WG * (PHIPL(I) *
     *          ( VP * DIP(J) - VDP * (A9*DIM(J)-A10*DIMS(J))
     *          + PP*PHIP2(J)-OM*(A9*PHID1(J)
     *                       +(A11 + A13) * PHIP2(J))
     *          + SM * (2.*A10*PHID1(J)+(A14+A15)* PHIP2(J)))
     *          + PHIDPL(I) *
     *          ( -A6 * DIM(J) + A4 * DIMS(J)
     *          -OM * ( PHIP1(J) + A9 * PHIP2(J))
     *          + SM * (2. * A10 * PHIP2(J) + A3 * PHID1(J))))
500   CONTINUE
600   CONTINUE
C
      WRITE (IO,2300) M,III,JJJ,P1,OM1,SM1,PP,OM,SM
C
      CALL TR1 (BB,D,CO,SI,SC,C2,S2)
      DO 650 I=1,5
      J  = II + I
      J1 = JJ + I
      B(J)  = B(J)  - D(I)
      B(J1) = B(J1) - D(I+5)
650   CONTINUE
      DO 700 J=1,10
      DO 700 I=1,8
700   T(I,J)  = 0.0
      T(1,1)  = CO
      T(1,2)  = SI
      T(2,4)  = C2
      T(2,5)  = S2
      T(3,6)  = CO
      T(3,7)  = SI
      T(4,9)  = C2
      T(4,10) = S2
      T(5,1)  = SI
      T(5,2)  = -CO
      T(6,3)  = -1.
      T(6,4)  = SC
```

```
      T(6,5)  = -SC
      T(7,6)  = SI
      T(7,7)  = -CO
      T(8,8)  = -1.
      T(8,9)  = SC
      T(8,10) = -SC
C
C   FORM ELEMENT STIFFNESS MATRIX IN GLABOL COORDINATES
C
      DO 720 L=1,10
      DO 720 K=1,8
      D1 = 0.0
      DO 710 N=1,8
710   D1 = D1 + S(K,N) * T(N,L)
720   TEMP(K,L) = D1
      DO 750 L=1,10
      DO 750 K=1,10
      D1 = 0.0
      DO 740 N=1,8
740   D1 = D1 + T(N,L) * TEMP(N,K)
      SS(L,K) = D1
750   SS(K,L) = D1
      K = 1
      DO 760 I=1,10
      DO 760 J=I,10
      A(K) = SS(I,J)
      K = K + 1
760   CONTINUE
C
      CALL ASSEMB (MAXA,A,TK,LM,10)
C
900   CONTINUE
C
      RETURN
C
2200  FORMAT (//,'STRESS RESULTANTS',//,5X,'MEMBER NO.',2X,
     *'NODI',2X,'NODJ',10X,'PI',13X,'MI',12X,'MI*',12X,
     *'PJ',13X,'MJ',12X,'MJ*',/)
2300  FORMAT (5X,I5,4X,I5,2X,I5,2X,6D15.6)
C
      END
```

```
C*************************************************************
C
            SUBROUTINE STIFFB (NOD,MAXA,X,Y,AREA,YMOD,TK,B,
     *                         DISP,NE,NFE,IDO,UNIT,NJ,IO)
C           ****************
C
C          THIS SUBROUTINE FORMS THE TANGENT STIFFNESS
C     MATRIX FOR A BRACING MEMBER IN A FRAME.
C
C     *******************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION NOD(2,1),MAXA(1),X(1),Y(1),AREA(1),YMOD(1),
     *TK(1),B(1),DISP(1),LM(4),D(4),DM(4),S(4,4),BB(4),
     *PHIP(2),PHIP1(4),PHIP2(4),DIP(4),A(10),SS(4,4),
     *T(4,4),TEMP(4,4)
C
      NNE = NFE + 1
C
      WRITE (IO,2000)
C
      DO 900 M=NNE,NE
      I  = NOD(1,M)
      J  = NOD(2,M)
      DX = (X(J) - X(I)) * UNIT
      DY = (Y(J) - Y(I)) * UNIT
      XL = DSQRT (DX*DX + DY*DY)
      XLI= 1.D0 / XL
      CO = DX * XLI
      SI = DY * XLI
      YA = YMOD(M) * AREA(M)
C
      DO 50 KK=1,2
      LM(KK)     = IDO * I - IDO + KK
50    LM(KK+2) = IDO * J - IDO + KK
C
      II = (I-1) * 5
      JJ = (J-1) * 5
      DO 100 I=1,2
      J  = II + I
      J1 = JJ + I
      D(I)   = DISP(J)
      D(I+2) = DISP(J1)
100   CONTINUE
C
      DM(1) = D(1) * CO + D(2) * SI
      DM(2) = D(3) * CO + D(4) * SI
      DM(3) = D(1) * SI - D(2) * CO
      DM(4) = D(3) * SI - D(4) * CO
C
      DO 200 J=1,4
      DO 200 I=1,4
      S(I,J) = 0.0D0
```

```
200   CONTINUE
C
      DO 250 I=1,4
      BB(I) = 0.0D0
250   CONTINUE
C
      PHIP(1) = -XLI
      PHIP(2) =  XLI
      UP  = PHIP(1) * DM(1) + PHIP(2) * DM(2)
      VP  = PHIP(1) * DM(3) + PHIP(2) * DM(4)
      UP2 = UP * UP
      VP2 = VP * VP
      A1  = 1 + UP
      A2  = UP + 0.5D0 * (UP2 + VP2)
      PP  = YA * A2
      WRITE (IO,2100) M,PP
      FE1 = XL * PP * A1
      FE2 = XL * PP * VP
      YAA = YA * A1
      YAV = YA * VP
C
      DO 300 I=1,2
      BB(I)       = FE1 * PHIP(I)
      BB(I+2)     = FE2 * PHIP(I)
      DIP(I)      = YAA * PHIP(I)
      DIP(I+2)    = YAV * PHIP(I)
      PHIP1(I)    = PHIP(I)
      PHIP1(I+2)= 0.0D0
      PHIP2(I)    = 0.0D0
      PHIP2(I+2)= PHIP(I)
300   CONTINUE
C
C  FORM ELEMENT STIFFNESS MATRIX
C
      DO 400 I=1,2
      DO 400 J=1,4
      S(I,J)    = S(I,J)    + XL * PHIP(I) * (A1*DIP(J)
     *                                       + PP*PHIP1(J))
      S(I+2,J) = S(I+2,J)  + XL * PHIP(I) * (VP*DIP(J)
     *                                       + PP*PHIP2(J))
400   CONTINUE
C
      D(1) = BB(1) * CO + BB(3) * SI
      D(2) = BB(1) * SI - BB(3) * CO
      D(3) = BB(2) * CO + BB(4) * SI
      D(4) = BB(2) * SI - BB(4) * CO
C
      DO 500 I=1,2
      J  = II + I
      J1 = JJ + I
      B(J)  = B(J) - D(I)
      B(J1) = B(J1) - D(I+2)
500   CONTINUE
C
```

```
      DO 550 J=1,4
      DO 550 I=1,4
550   T(I,J) = 0.0D0
C
      T(1,1) = CO
      T(1,2) = SI
      T(2,3) = CO
      T(2,4) = SI
      T(3,1) = SI
      T(3,2) = -CO
      T(4,3) = SI
      T(4,4) = -CO
C
C   FORM ELEMENT STIFFNESS MATRIX IN GLOBAL COORDINATES
C
      DO 650 L=1,4
      DO 650 K=1,4
      D1 = 0.0D0
      DO 600 N=1,4
600   D1 = D1 + S(K,N) * T(N,L)
650   TEMP(K,L) = D1
C
      DO 750 L=1,4
      DO 750 K=1,4
      D1 = 0.0D0
      DO 700 N=1,4
700   D1 = D1 + T(N,L) * TEMP(N,K)
      SS(L,K) = D1
750   SS(K,L) = D1
      K = 1
      DO 800 I=1,4
      DO 800 J=I,4
      A(K) = SS(I,J)
      K = K + 1
800   CONTINUE
      NR = 4
      CALL ASSEMB (MAXA,A,TK,LM,NR)
C
900   CONTINUE
C
      RETURN
C
C   FORMAT STATEMENTS
C
2000  FORMAT (5X,'BRACING MEMBER NO.',5X,'AXIAL FORCE'/)
2100  FORMAT (10X,I5,15X,D15.6)
C
      END
```

```
C*****************************************************************
C
              SUBROUTINE TR (D,CO,SI,SC,C2,S2,DM)
C             **************
C
C          THIS SUBROUTINE TRANSFORMS THE NODAL DISPLACEMENTS
C      FROM SYSTEM COORDINATES TO LOCAL ONS.
C
C      *****************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION D(10),DM(8)
C
      DM(1) = D(1) * CO + D(2) * SI
      DM(2) = D(4) * C2 + D(5) * S2
      DM(3) = D(6) * CO + D(7) * SI
      DM(4) = D(9) * C2 + D(10)* S2
      DM(5) = D(1) * SI - D(2) * CO
      DM(6) = -D(3) + D(4) * SC - D(5) * SC
      DM(7) = D(6) * SI - D(7) * CO
      DM(8) = -D(8) + D(9) * SC - D(10) * SC
C
      RETURN
C
      END
```

```
C*****************************************************
C
                 SUBROUTINE TR1 (BB,DD,CO,SI,SC,C2,S2)
C                **************
C
C            THIS SUBROUTINE TRANSFORMS THE NODAL FORCES
C      FROM LOCAL TO GLABOL COORDINATES.
C
C      ************************************************
C
       IMPLICIT REAL*8(A-H,O-Z)
C
       DIMENSION BB(8),DD(10)
C
       DD(1)  = BB(1) * CO + BB(5) * SI
       DD(2)  = BB(1) * SI - BB(5) * CO
       DD(3)  = -BB(6)
       DD(4)  = BB(2) * C2 + BB(6) * SC
       DD(5)  = BB(2) * S2 - BB(6) * SC
       DD(6)  = BB(3) * CO + BB(7) * SI
       DD(7)  = BB(3) * SI - BB(7) * CO
       DD(8)  = -BB(8)
       DD(9)  = BB(4) * C2 + BB(8) * SC
       DD(10) = BB(4) * S2 - BB(8) * SC
C
       RETURN
C
       END
```

E.3 Program INPLAF

Program INPLAF consists of the following parts.

1. MAIN
2. MAINMG, the main executive subroutine.
3. The input package
   (a) INPUT1 and INPUT2
   (b) BOUND, JLOAD2, and MLOAD2.
4. The data managing package
   ISPAC, LOCOM, REMOV, REMOV2, and BLOCK DATA.
5. The data storage and retrieval package
   CLEAR, ICLEAR, RTRV1, RTRV2, STORE1, and STORE2.
6. The equation solving package
   (a) ADDRES and COLHT
   (b) EQSBST and BKSB1
7. The formulation and output package
   (a) ASSEMB, ELEMS, MODIFY, BOUND1, DISPL1, STRES1, MULTI1, and MNHING
   (b) STIFFP, HINGE, and CORRB.

The listing for packages 1, 2, 3a, 6b, and 7b follows. The listing for parts 3b, 4, 5, 6a, and 7a can be found in Sects. E.8, E.6, E.7, E.9, and E.10, respectively. Descriptions of the subroutines can be found in the listing.

Two temporary files (1 and 2) are used in sequential I/O operations. Both files are line-files.

```
C*****************************************************
C
C                    INPLAF
C                    ******
C
C          THIS IS A PROGRAM FOR ELASTO-PLASTIC STABILITY
C      ANALYISIS OF PLANER FRAMED TYPE STRUCTURES. THE
C      PROGRAM EVALUATES THE RESPONSE OF THE STRUCTURE
C      USING INCREMENTAL TECHNIQUE.
C
C      ***************************************************
C
       IMPLICIT REAL*8(A-H,O-Z)
       REAL*8 NAMES
C
       COMMON/PROBIA/ NNN(200)
       COMMON/IA1/    MMM(200)
       COMMON/RA1/    AAA(4000)
       COMMON/RA2/    BBB(1000)
       COMMON/RA3/    CCC(1000)
       COMMON/DIMCOM/LAST1,LAST2,LAST3,LAST4,LAST5,MXDIM,
      *NAMES(5,20),IPT(5,21),ICOM(5)
C
C      ****************************************************
C
       ICOM(1) = 200
       ICOM(2) = 4000
       ICOM(3) = 200
       ICOM(4) = 1000
       ICOM(5) = 1000
C
       CALL MAINMG
C
       END
```

```
C*************************************************************
C
                 SUBROUTINE MAINMG
C                ****************
C
C           THIS IS THE MAIN MANAGER OF PROGRAM INPLAF. IT
C         CONTROLS THE READING , THE FORMULATION ,THE SOLUTION
C         AND THE OUTPUT OF THE RESULTS.
C
C         *****************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 NAMES,NAME
C
      COMMON/PROBCV/ UNIT,NJ,NE,NEBEL,MNHE,ITAF,NLC,IN,IO
C
      COMMON/PROBIA/ NNN(1)
      COMMON/IA1   / MMM(1)
      COMMON/RA1   / AAA(1)
      COMMON/RA2   / BBB(1)
      COMMON/RA3   / CCC(1)
      COMMON/DIMCOM/ LA1,LA2,LA3,LA4,LA5,MXDIM,NAMES(5,20)
     *,IPT(5,21),ICOM(5)
C
C     *****************************************************
C
      CALL TIME (0,0)
      IN = 5
      IO = 6
C
      REWIND 1
C
C   READ PROBLEM CONTROL VARIABLES
C   ------------------------------C
C
      CALL INPUT1
C
      NODE  = 2
      IDOF  = 3
      ILOAD = 1
      NEQ   = NJ * IDOF
      CALL CLEAR (NAMES(1,1),100)
C
C   READ NODAL GEOMETRY AND MEMBER PROP. AND CONNECTIVITIES
C   ------------------------------------------------------C
      J1 = ISPAC (1HX,NJ,2)
      J2 = ISPAC (1HY,NJ,2)
      J3 = ISPAC (4HAREA,NE,2)
      J4 = ISPAC (2HRI,NE,2)
      J5 = ISPAC (2HPM,(NE*2),2)
      J6 = ISPAC (4HYMOD,NE,2)
      J7 = ISPAC (3HPSM,NE,2)
      J8 = ISPAC (2HYS,NE,2)
      I1 = ISPAC (3HNOD,(NE*NODE),1)
```

```
         I2 = ISPAC (5HMKODE,NE,1)
         I3 = ISPAC (5HMFLAG,NE,1)
C
         CALL INPUT 2 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J5),
        *AAA(J6),AAA(J7),AAA(J8),NNN(I1),NNN(I2),NNN(I3))
C
C
C     READ EXTERNAL BOUNDARY CONDITIONS
C
         J9 = ISPAC (3HBES,NEBEL,2)
         K1 = ISPAC (3HNPB,NEBEL,3)
         K2 = ISPAC (4HKODE,NEBEL,3)
C
         CALL BOUND (AAA(J9),MMM(K1),MMM(K2),NEBEL,IN,IO)
C
C
         INN = IDOF * NODE * NE
         J10 = ISPAC (4HFEFC,INN,2)
         J11 = ISPAC (4HFEFV,INN,2)
         CALL CLEAR (AAA(J10),(INN*2))
C
         L1 = ISPAC (2HBC,NEQ,4)
         L2 = ISPAC (2HBV,NEQ,4)
         CALL CLEAR (BBB(L1),(NEQ*2))
C
C     READ LOADS IF ANY
C
         READ (IN,1000) JLF,MLF
         IF(JLF.EQ.0) GO TO 50
         CALL JLOAD2 (BBB(L1),BBB(L2),AAA(J9),IDOF,IN,IO)
50       IF(MLF.EQ.0) GO TO 60
         CALL MLOAD2 (AAA(J1),AAA(J2),AAA(J10),AAA(J11),
        *BBB(L1),BBB(L2),NNN(I1),NNN(I2),UNIT,IDOF,NJ,IN,IO)
60       CONTINUE
C
C     FORM COLUMN HEIGHTS AND ADDRESSING ARRAY
C
         I5 = ISPAC (4HMAXA,(NEQ+1),1)
         I6 = ISPAC (3HMHT,NEQ,1)
         CALL ICLEAR (NNN(I6),NEQ)
         DO 70 I=1,NE
         CALL COLHT (NODE,(NODE*IDOF),IDOF,I,NNN(I6),
        *NNN(I1))
70       CONTINUE
C
         CALL ADDRES (NNN(I5),NNN(I6),NEQ,NWA)
         CALL REMOV (3HMHT,1)
C
         L3  = ISPAC (2HBT,NEQ,4)
         L4  = ISPAC (2HDT,NEQ,4)
         CALL CLEAR (BBB(L3),(NEQ*2))
         L5  = ISPAC (2HFL,(NE*2),4)
         J12 = ISPAC (5HFEFC2,INN,2)
         J13 = ISPAC (5HFEFV2,INN,2)
```

```
      CALL CLEAR (AAA(J12),(INN*2))
      K3  = ISPAC (5HMAXNH,NJ,3)
      CALL ICLEAR (MMM(K3),NJ)
      CALL MNHING (NNN(I1),NNN(I2),MMM(K1),MMM(K3),NE,
     *NJ,NEBEL)
C
      DO 80 I=1,NEQ
      BBB(L3+I-1) = BBB(L1+I-1)
80    CONTINUE
      REWIND 2
      CALL STORE2 (2,BBB(L2),NEQ)
C RESERVE AND CLEAR SPACE FOR ELASTIC
C    STIFFNESS MATRIX
C
      NH = 0
      NV = 0
      M1 = ISPAC (1HA,NWA,5)
      M2 = ISPAC (2HAG,NWA,5)
      M3 = ISPAC (2HUF,NEQ,5)
      M4 = ISPAC (2HTT,NEQ,5)
      CALL CLEAR (CCC(M3),(2*NEQ))
90    CALL CLEAR (CCC(M1),(NWA*2))
      CALL CLEAR (BBB(L5),(NE*2))
C
C      FORM ELEMENT STIFFNESS AND ASSEMBLE
C   ELASTIC STIFFNESS INTO A.
C
      CALL STIFFP (NNN(I1),NNN(I2),NNN(I3),NNN(I5),
     *AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J6),CCC(M1),
     *AAA(J12),AAA(J13),UNIT,NE,ITAF)
C
C      ADD EXTERNAL BOUNDRAY CONDITIONS
C
      CALL BOUND1 (CCC(M1),NNN(I5),MMM(K1),MMM(K2),
     *AAA(J9),NEBEL)
C
      IF(ITAF.EQ.0) GO TO 91
      REWIND 1
      CALL STORE2 (1,CCC(M1),NWA)
91    CALL EQSBST (CCC(M1),BBB(L1),BBB(L2),NNN(I5),NEQ,
     *NLC,ILOAD,0)
      IF(ILOAD.NE.999) GO TO 95
      IF(ITAF.EQ.0) GO TO 900
      IF(NV.EQ.1) GO TO 95
      REWIND 2
      CALL RTRV2 (2,BBB(L2),NEQ)
      DO 92 I=1,NEQ
      BBB(L2+I-1) = -BBB(L2+I-1)
92    CONTINUE
      REWIND 1
      REWIND 2
      CALL RTRV2 (1,CCC(M1),NWA)
      CALL STORE2 (2,BBB(L2),NEQ)
      NV = 1
```

```
      CALL EQSBST (CCC(M1),BBB(L1),BBB(L2),NNN(I5),NEQ,
     *NLC,ILOAD,0)
95    CALL BKSB1 (BBB(L1),BBB(L2),CCC(M1),NNN(I5),NEQ,NLC,0)
      IF(NLC.EQ.1) GO TO 200
      NLC = 1
      CALL DISPL1 (AAA(J12),BBB(L1),AAA(J10),NJ,NE,IDOF,
     *IN,IO)
      CALL STRES1 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J6),
     *BBB(L1),AAA(J12),AAA(J13),NNN(I1),NNN(I2),NE,IDOF,
     *UNIT,IN,IO)
      DO 100 I=1,NEQ
      BBB(L4+I-1) = BBB(L1+I-1)
100   CONTINUE
C
200   CALL DISPL1 (AAA(J13),BBB(L2),AAA(J11),NJ,NE,IDOF,
     *IN,IO)
      DO 250 I=2,NEQ,3
      C = DABS(BBB(L2+I))
      IF(C.GT.1.0E+6) GO TO 900
250   CONTINUE
      CALL STRES1 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J6),
     *BBB(L2),AAA(J13),AAA(J12),NNN(I1),NNN(I2),NE,IDOF,
     *UNIT,IN,IO)
      IF(ITAF.EQ.0) GO TO 400
      REWIND 2
      CALL RTRV2 (2,BBB(L2),NEQ)
      CALL CORRB (NNN(I1),NNN(I2),NNN(I3),NNN(I5),AAA(J1),
     *AAA(J2),AAA(J3),AAA(J4),AAA(J6),CCC(M2),CCC(M4),
     *BBB(L2),BBB(L4),AAA(J13),UNIT,NEQ,NWA,NE)
      CALL STORE2 (2,BBB(L2),NEQ)
      REWIND 1
      CALL RTRV2 (1,CCC(M1),NWA)
      CALL EQSBST (CCC(M1),BBB(L1),BBB(L2),NNN(I5),NEQ,
     *NLC,ILOAD,0)
      CALL BKSB1 (BBB(L1),BBB(L2),CCC(M1),NNN(I5),NEQ,NLC,0)
      CALL DISPL1 (AAA(J13),BBB(L2),AAA(J11),NJ,NE,IDOF,
     *IN,IO)
      CALL STRES1 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J6),
     *BBB(L2),AAA(J13),AAA(J12),NNN(I1),NNN(I2),NE,IDOF,
     *UNIT,IN,IO)
400   REWIND 2
      CALL RTRV2 (2,BBB(L1),NEQ)
      IF(ITAF.NE.0) CALL RTRV2 (2,CCC(M1),NEQ)
C
      CALL HINGE (AAA(J3),AAA(J7),AAA(J8),NNN(I1),NNN(I2),
     *BBB(L5),BBB(L1),BBB(L3),BBB(L2),BBB(L4),CCC(M1),
     *CCC(M3),AAA(J13),AAA(J12),MMM(K3),NE,NJ,NEQ,INN,
     *IDOF,ITAF,UNIT,IN,IO)
      IF(NH.GT.MNHE) GO TO 900
      NH = NH + 1
      WRITE (IO,2000) NH
      CALL TIME (3,3)
      DO 500 I=1,NEQ
      BBB(L2+I-1) = BBB(L1+I-1)
```

```
500    CONTINUE
       GO TO 90
C
900    RETURN
C
C  FORMAT STATEMENTS
C
1000   FORMAT (2I5)
2000   FORMAT (//,10X,30(1H*)/,10X,'HINGE NO.',
     *I3,3X,'HAS BEEN FORMED'/,10X,30(1H*)//)
C
       END
```

```
C*************************************************************
C
                SUBROUTINE INPUT1
C               ****************
C
C          THIS SUBROUTINE READS THE PROBLEM CONTROL VARIABLES
C      FOR PROGRAM INPLAF.
C
C      ***********************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON/PROBCV/ UNIT,NJ,NE,NEBEL,MNHE,ITAF,NLC,IN,IO
C
      DIMENSION HEAD(20)
C
      READ (IN,1000) HEAD
      WRITE(IO,2000) HEAD
      READ (IN,1100) NJ,NE,NEBEL,NLC,ITAF,MNHE,UNIT
      WRITE(IO,2100) NJ,NE,NEBEL,NLC,ITAF,MNHE,UNIT
C
      RETURN
C
C   FORMAT STATEMENTS
C
1000  FORMAT (20A4)
1100  FORMAT (6I4,F12.0)
2000  FORMAT (//,30X,20A4//,30X,30(1H*)////)
2100  FORMAT (5X,'PROBLEM CONTROL VARIABLES'/,5X,25(1H*)//,
     *10X,'NUMBER OF JOINTS                         =',I5//,
     *10X,'NUMBER OF ELEMENTS                       =',I5//,
     *10X,'NO. OF EXTERNAL BOUNDARY ELEMENTS        =',I5//,
     *10X,'NUMBER OF LOAD CASES                     =',I5//,
     *10X,'IDENTIF. OF TYPE OF ANALYSIS FLAG        =',I5//,
     *10X,'MAXIMUM NO. OF HINGES ESTIMATED          =',I5//,
     *10X,'LENGTH CONVERSION FACTOR                 =',F12.6//,
     *10X,50(1H*))
C
      END
```

```
C*****************************************************************
C
          SUBROUTINE INPUT 2 (X,Y,AREA,RI,PM,YMOD,PSM,YS,NOD,
     *                                       MKODE,MFLAG)
C
C          THIS SUBROUTINE READS THE NODAL GEOMETRY AND
C       MEMBER PROPERTIES FOR PROGRAM INPLAF.
C
C     ***********************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON/PROBCV/ UNIT,NJ,NE,NEBEL,MNHE,ITAF,NLC,IN,IO
      DIMENSION X(1),Y(1),AREA(1),RI(1),YMOD(1),NOD(2,1),
     *MKODE(1),MFLAG(1),PSM(1),YS(1),PM(2,1)
C
      WRITE(IO,2000)
50    READ (IN,1000) N,X(N),Y(N),INC
      IF(INC.EQ.0) GO TO 200
      NINT = (N-NOLD) / INC
      RN   = NINT
      IF(RN.LT.FLOAT(N-NOLD)/FLOAT(INC)-0.001) GO TO 999
      DX = (X(N)-X(NOLD))/RN
      DY = (Y(N)-Y(NOLD))/RN
      L = NOLD
      M = NINT - 1
      DO 100 J=1,M
      LL = L + INC
      X(LL) = X(L) + DX
      Y(LL) = Y(L) + DY
      L = LL
100   CONTINUE
200   WRITE (IO,2100) N,X(N),Y(N),INC
      NOLD = N
      IF(N.LT.NJ) GO TO 50
C
      READ (IN,1100) ADEF,RDEF,SMDEF,YSDEF,YDEF
      WRITE(IO,2200) ADEF,RDEF,SMDEF,YSDEF,YDEF
C
      WRITE(IO,2300)
300   READ (IN,1200) M,(NOD(I,M),I=1,2),MKODE(M),MFLAG(M),
     *INC,NOD1,NOD2
      READ (IN,1300) AREA(M),RI(M),PSM(M),YS(M),YMOD(M)
      IF(AREA(M).EQ.0) AREA(M) = ADEF
      IF(RI(M).EQ.0)   RI(M)   = RDEF
      IF(PSM(M).EQ.0) PSM(M)   = SMDEF
      IF(YS(M).EQ.0)  YS(M)    = YSDEF
      IF(YMOD(M).EQ.0) YMOD(M) = YDEF
      PM(1,M) = PSM(M) * YS(M)
      PM(2,M) = PM(1,M)
      NN = MKODE(M) + 1
      GO TO (340,310,320,330),NN
310   PM(1,M) = 0.0
      GO TO 340
```

```
 320   PM(2,M) = 0.0
       GO TO 340
 330   PM(1,M) = 0.0
       PM(2,M) = 0.0
 340   IF(INC.EQ.0) GO TO 500
       NINT = (M-MOLD) / INC - 1
       L = MOLD
       DO 400 I=1,NINT
       LL         =L + INC
       AREA(LL)   = AREA(M)
       RI(LL)     = RI(M)
       PSM(LL)    = PSM(M)
       YS(LL)     = YS(M)
       YMOD(LL)   = YMOD(M)
       MKODE(LL)  = MKODE(M)
       MFLAG(LL)  = MFLAG(M)
       NOD(1,LL)  = NOD(1,L) + NOD1
       NOD(2,LL)  = NOD(2,L) + NOD2
 400   L = LL
C
 500   WRITE(IO,2400) M,NOD(1,M),NOD(2,M),MKODE(M),MFLAG(M),
      *INC,NOD1,NOD2,AREA(M),RI(M),PSM(M),YS(M),YMOD(M)
       MOLD = M
       IF(M.LT.NE) GO TO 300
       READ (IN,1400) K
       IF(K.EQ.0) GO TO 600
       WRITE(IO,2700)
       WRITE(IO,2750) (N,X(N),Y(N),N=1,NJ)
       WRITE(IO,2800)
       WRITE(IO,2850) (M,(NOD(I,M),I=1,2),MKODE(M),MFLAG(M),
      *AREA(M),RI(M),YMOD(M),M=1,NE)
C
600    RETURN
C
999    WRITE (IO,9999) N
9999   FORMAT('NODAL GEOMETRY DATA INPUT ERROR',I5)
       STOP
C
C         FORMAT STATEMENTS
C
1000   FORMAT(I5,2F12.0,I5)
1100   FORMAT(5F12.0)
1200   FORMAT(8I4)
1300   FORMAT (5F12.0)
1400   FORMAT(I5)
2000   FORMAT (///,5X,'NODAL GEOMETRY DATA AS INPUT'/,
      *5X,28(1H*)//,10X,1HN,8X,1HX,14X,1HY,9X,3HINC/)
2100   FORMAT (6X,I5,2D15.6,I5)
2200   FORMAT (///,5X,'MEMBER PROPERTIES DEFAULT VALUES'/,
      *5X,32(1H*)//,10X,'AREA',20X,' =',D15.6,//,
      *10X,'M. INERTIA',15X,'=',D15.6,//,
      *10X,'PLASTIC SECTION MODULUS  =',D15.6//,
      *10X,'YIELD STRESS             =',D15.6//,
      *10X,'Y. MODULUS',15X,'=',D15.6)
```

```
2300  FORMAT (///,5X,'MEMBER DATA AS INPUT'/,5X,20(1H*)//,
     *4X,1HM,4X,1HI,4X,1HJ,2X,4HCODE,1X,4HFLAG,2X,3HINC,
     *1X,4HINCI,1X,4HINCJ,3X,4HAREA,10X,1HI,11X,3HPSM,
     *9X,2HYS,11X,4HYMOD/)
2400  FORMAT (8I5,5D12.4)
2700  FORMAT (///,5X,'COMPLETED NODAL GEOMETRY DATA'/,
     *5X,29(1H*)//,4X,1HN,7X,1HX,14X,1HY/)
2750  FORMAT (I5,2D15.6)
2800  FORMAT (///,5X,'COMPLETED MEMBER DATA'/,5X,20(1H*)//,
     *4X,1HM,4X,1HI,4X,1HJ,1X,4HCODE,5X,4HAREA,
     *10X,1HI,13X,4HYMOD/)
2850  FORMAT (5I5,3D15.6)
C
      END
```

```
C*************************************************************
C
            SUBROUTINE  EQSBST (A,BC,BV,MAXA,NB,NLC,ILOAD,IT)
C           *****************
C
C             THIS SUBROUTINE TRIANGULARIZES A STIFFNESS MATRIX
C         STORED COLUMNWISE UNDER A SKYLINE AND REDUCES THE
C         CORRESPONDING LOAD VECTOR, DOWN TO EQUATION NB.
C
C         ***************************************************
          IMPLICIT REAL*8(A-H,O-Z)
C
          DIMENSION A(1),BC(1),BV(1),MAXA(1)
C
C
          DO  1000 N=1,NB
          KN = MAXA(N)
          KL = KN + 1
          KU = MAXA(N+1) - 1
          KH = KU - KL
          IF(KH) 900,500,100
  100     IF(ILOAD.EQ.2.AND.IT.GT.0) GO TO 650
          K  = N - KH
          IC = 0
          KLT= KU
          DO 400 J=1,KH
          IC = IC + 1
          KLT= KLT - 1
          KI = MAXA(K)
          ND = MAXA(K+1) - KI - 1
          IF(ND) 400,400,200
  200     KK = IC
          IF(KK.GT.ND) KK = ND
          C  = 0.0
          DO 300 L = 1,KK
  300     C  = C + A(KI+L)*A(KLT+L)
          A(KLT) = A(KLT) - C
  400     K  = K + 1
  500     IF(ILOAD.EQ.2.AND.IT.GT.0) GO TO 650
          K  = N
C
          C  = 0.0
          DO 600 KK = KL,KU
          K  = K - 1
          KI = MAXA(K)
          D  = A(KK)/A(KI)
          C  = C + D*A(KK)
          A(KK) = D
  600     CONTINUE
          A(KN) = A(KN) - C
C
  650     CONTINUE
          K = N
          C = 0.0
```

```
      DO 700 KK=KL,KU
      K = K - 1
700   C = C + A(KK) * BV(K)
      BV(N) = BV(N) - C
      IF(NLC.EQ.1) GO TO 900
      K = N
      C1 = 0.0
      DO 800 KK=KL,KU
      K = K - 1
800   C1 = C1 + A(KK) * BC(K)
      BC(N) = BC(N) - C1
C
900   IF(A(KN)) 975,950,1000
950   WRITE(6,3000) N,A(KN)
      STOP
975   ILOAD = 999
C
1000  CONTINUE
C
      RETURN
C
3000  FORMAT('ZERO OR NEGATIVE ELEMENT ON MAIN DIAGONAL NO.'
     *,I4,D15.6)
C
      END
```

```
C*****************************************************************
C
              SUBROUTINE BKSB1 (BC,BV,A,MAXA,NN,NLC,KB)
C             ****************
C
C        THIS SUBROUTINE IS A PART OF THE PARTIAL REDUCTION
C      PACKAGE. IT FORMS THE LAST STAGE OF BACKSUBSTITUTION,
C      THE OPERATION (L(-1)(T)*RI*) .   IN CASE OF AN
C      UNSUBSTRUCTURED PROBLEM IT PERFORMS THE OPERATION
C      //D(-1)*L(-1)(T)*R//
C
C      ***********************************************************
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION A(1),BC(1),BV(1),MAXA(1)
C
      IF(KB.NE.0) GO TO 200
      DO 100 J=1,NN
      K = MAXA(J)
      BV(J) = BV(J) / A(K)
      IF(NLC.EQ.1) GO TO 100
      BC(J) = BC(J) / A(K)
100   CONTINUE
200   N = NN
      DO 600 L=2,NN
      KL = MAXA(N) + 1
      KU = MAXA(N+1) - 1
      IF(KU-KL) 600,300,300
300   IF(BV(N).EQ.0) GO TO 500
      K = N
      DO 400 KK=KL,KU
      K = K - 1
400   BV(K) = BV(K) - A(KK) * BV(N)
500   IF(NLC.EQ.1) GO TO 600
      IF(BC(N).EQ.0) GO TO 600
      K = N
      DO 550 KK=KL,KU
      K = K - 1
550   BC(K) = BC(K) - A(KK) * BC(N)
600   N = N - 1
C
      RETURN
C
      END
```

```
C***************************************************************
C
          SUBROUTINE STIFFP (NOD,MKODE,MFLAG,MAXA,X,Y,AREA,RI,
     *                      YMOD,A,PPC,PPV,UNT,NE,ITAF)
C
C
C          THIS SUBROUTINE FORMS THE ELASTIC STIFFNESS MATRIX
C       OF 6 DOF PLANE FRAME MEMBER. THE UPPER TRIANGLE IS
C       STORED COLUMN-WISE IN A VECTOR.
C
C       ***************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
C
        DIMENSION NOD(2,1),MKODE(1),MFLAG(1),MAXA(1),
       *X(1),Y(1),AREA(1),RI(1),YMOD(1),A(1),PPC(1),PPV(1),
       *LM(6),AS(7),AC(7),S(21)
C
C
        DO 700 M=1,NE
        K = MKODE(M)
        I = NOD(1,M)
        J = NOD(2,M)
        DX = (X(J) - X(I)) * UNT
        DY = (Y(J) - Y(I)) * UNT
        XL = DSQRT (DX**2 + DY**2)
        XLI = 1. / XL
        CO = DX * XLI
        SI = DY * XLI
        CL = CO * XLI
        SL = SI * XLI
        CL2 = CL ** 2
        SL2 = SL ** 2
        CSL = CL * SL
        C2  = CO ** 2
        S2  = SI ** 2
        SC  = CO * SI
        ALP = YMOD(M) * RI(M) * XLI
        BETA= YMOD(M) * AREA(M) * XLI
C
C     FORM LM ARRAY
C
        LM(3) = 3 * I
        LM(2) = LM(3) - 1
        LM(1) = LM(3) - 2
        LM(6) = 3 * J
        LM(5) = LM(6) - 1
        LM(4) = LM(6) - 2
C
C         FORMULATION OF ELASTIC STIFFNESS MATRIX
C
        AS(1) = BETA
        AS(2) = 12.0 * ALP
```

```
      AS(3) = 6.0 * ALP
      AS(4) = AS(3)
      AS(5) = 4.0 * ALP
      AS(6) = 2.0 * ALP
      AS(7) = AS(5)
      IF(K.NE.0) CALL MODIFY(K,AS)
C
C       FORM ELASTIC STIFFNESS MATRIX
C
      CALL ELEMS(AS,S,CL,SL,CL2,SL2,CSL,C2,S2,SC)
      CALL ASSEMB (MAXA,S,A,LM,6)
      IF(ITAF.EQ.0) GO TO 700
C
C         FORMULATION OF GEOMETRIC ELEMENT STIFFNESS
      L = MFLAG(M)
      IF(L.EQ.0) GO TO 700
      M1 = 6 * M - 5
      AC(1) = 0.0
      AC(2) = 6.0/5.0 * XL
      AC(3) = 0.0
      AC(4) = AC(3)
      AC(5) = 0.0
      AC(6) = 0.0
      AC(7) = AC(5)
      CALL ELEMS (AC,S,CL,SL,CL2,SL2,CSL,C2,S2,SC)
      DO 100 KK=1,21
100   S(KK) = - S(KK) * PPC(M1)
      CALL ASSEMB (MAXA,S,A,LM,6)
700   CONTINUE
C
      RETURN
C
      END
```

```
C*****************************************************
C
              SUBROUTINE HINGE (AREA,PSM,YS,NOD,MKOD,FL,
     *                  B,BT,D,DT,UF,UFT,EMF,EMFT,MAXNH,
     *                  NE,NJ,NEQ,INN,IDO,ID,UNIT,IN,IO)
C
C        THIS SUBROUTINE EVALUATES THE ABSOLUTE VALUE OF
C     M /(PM - M) AT EACH NODE FOR EACH MEMBER AND FIND THE
C     LARGEST VALUE AND INSERT HINGE AT THIS NODE.  IT ALSO
C     UPDATES LOADS , DISPLACEMENTS AND MEMBER FORCES.
C
C     ***************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION PSM(1),YS(1),NOD(2,1),MKOD(1),B(1),BT(1),
     *D(1),DT(1),AREA(1),UF(1),UFT(1),EMF(1),EMFT(1),
     *FL(2,1),MAXNH(1)
C
      DO 100 M=1,NE
      KK = 3
      PMC = PSM(M) * YS(M) / UNIT
      PY  = AREA(M) * YS(M)
      MI  = 6 * M - 5
      PPY = EMFT(MI)  / PY
      IF(DABS(PPY).GT.0.15) PMC= 1.18*(1-PPY)*PMC
      DO 50 I=1,2
      MJ  = 6 * M - KK
      PMC1 = PMC
      IF(EMF(MJ).GT.0) PMC1 = -PMC
      PMMJ = PMC1 + EMFT(MJ)
      IF(DABS(EMF(MJ)).LT..0000001) GO TO 50
      FL(I,M) = EMF(MJ) / PMMJ
50    KK = 0
100   CONTINUE
      K  = 0
      K1 = 0
      ABMAX = 0.0
      FLC   = 0.0
      DO 200 M=1,NE
      DO 200 I=1,2
      C = FL(I,M)
      IF(DABS(C).LT.ABMAX) GO TO 200
      K  = M
      K1 = I
      ABMAX = DABS(C)
200   CONTINUE
      FLC = - 1. / FL(K1,K)
      MKOD(K) = MKOD(K) + K1
      FL(K1,K) = 0.0
      JN       = NOD(K1,K)
      MAXNH(JN)= MAXNH(JN) - 1
      ABMAX = ABMAX - .0000001
      DO 250 M=1,NE
```

```
      DO 250 L=1,2
      C = FL(L,M)
      I = NOD(L,M)
      IF(DABS(C).LT.ABMAX) GO TO 250
      IF(MAXNH(I).EQ.0) GO TO 250
      MKOD(M) = MKOD(M) + L
      MAXNH(I) = MAXNH(I) - 1
250   CONTINUE
      DO 300 I=1,NEQ
      BT(I) = BT(I) + B(I) * FLC
      DT(I) = DT(I) + D(I) * FLC
      IF(ID.EQ.0) GO TO 300
      UFT(I)= UFT(I)+ UF(I)* FLC
300   CONTINUE
      DO 400 I=1,INN
      EMFT(I) = EMFT(I) + EMF(I) * FLC
400   CONTINUE
      WRITE(IO,2000)
      DO 500 N=1,NJ
      N2 = N * IDO
      N1 = N2 - IDO + 1
500   WRITE(IO,2100) N,(BT(NN),NN=N1,N2)
575   WRITE(IO,2200)
      DO 600 N=1,NJ
      N2 = N * IDO
      N1 = N2 - IDO + 1
600   WRITE(IO,2100) N,(DT(NN),NN=N1,N2)
      WRITE(IO,2300)
      DO 700 M=1,NE
      M2 = 6 * M
      M1 = M2 - 5
700   WRITE(IO,2400) M,NOD(1,M),NOD(2,M),(EMFT(I),I=M1,M2)
C
      RETURN
C
C  FORMAT STATEMENTS
C
2000  FORMAT (///'TOTAL EXTERNAL NODAL LOADS',//,4X,
     *'N',7X,'FU',13X,'FV',13X,'FR'/)
2100  FORMAT (I5,3D15.6)
2200  FORMAT (//,'TOTAL NODAL DISPLACEMENTS',//,4X,'N',7X,
     *'U',14X,'V',14X,'R'/)
2300  FORMAT (//,'TOTAL MEMBER END FORCES',//,3X,'MEM',3X,
     *'I',4X,'J',7X,'NI',13X,'VI',13X,'MI',13X,'NJ',13X,
     *'VJ',13X,'MJ'/)
2400  FORMAT (3I5,6D15.6)
C
      END
```

```
C*************************************************************
C
            SUBROUTINE CORRB (NOD,MKODE,MFLAG,MAXA,X,Y,AREA,RI,
     *                        YMOD,AG,TT,B,D,PPV,UNT,NEQ,NWA,NE)
C
C
C            THIS SUBROUTINE FORMS THE ELASTIC STIFFNESS MATRIX
C         OF 6 DOF PLANE FRAME MEMBER. THE UPPER TRIANGLE IS
C         STORED COLUMN-WISE IN A VECTOR.
C
C         *****************************************************
C
          IMPLICIT REAL*8(A-H,O-Z)
C
C
          DIMENSION NOD(2,1),MKODE(1),MFLAG(1),MAXA(1),
         *X(1),Y(1),AREA(1),RI(1),YMOD(1),AG(1),B(1),D(1),
         *PPV(1),TT(1),LM(6),AC(7),S(21)
C
C
          DO 700 M=1,NE
          K = MKODE(M)
          I = NOD(1,M)
          J = NOD(2,M)
          DX = (X(J) - X(I)) * UNT
          DY = (Y(J) - Y(I)) * UNT
          XL = DSQRT (DX**2 + DY**2)
          XLI = 1. / XL
          CO = DX * XLI
          SI = DY * XLI
          CL = CO * XLI
          SL = SI * XLI
          CL2 = CL ** 2
          SL2 = SL ** 2
          CSL = CL * SL
          C2  = CO ** 2
          S2  = SI ** 2
          SC  = CO * SI
          ALP = YMOD(M) * RI(M) * XLI
          BETA= YMOD(M) * AREA(M) * XLI
C
C     FORM LM ARRAY
C
          LM(3) = 3 * I
          LM(2) = LM(3) - 1
          LM(1) = LM(3) - 2
          LM(6) = 3 * J
          LM(5) = LM(6) - 1
          LM(4) = LM(6) - 2
C
C            FORMULATION OF GEOMETRIC ELEMENT STIFFNESS
          L = MFLAG(M)
          IF(L.EQ.0) GO TO 700
          M1 = 6 * M - 5
```

```
      AC(1) = 0.0
      AC(2) = 6.0/5.0 * XL
      AC(3) = 0.0
      AC(4) = AC(3)
      AC(5) = 0.0
      AC(6) = 0.0
      AC(7) = AC(5)
      CALL ELEMS (AC,S,CL,SL,CL2,SL2,CSL,C2,S2,SC)
      DO 100 KK=1,21
100   S(KK) =  S(KK) * PPV(M1)
      CALL ASSEMB (MAXA,S,AG,LM,6)
700   CONTINUE
      CALL MULT1 (TT,AG,D,MAXA,NEQ,NWA)
      DO 800 I=1,NEQ
      B(I)   = B(I) +  TT(I)
800   CONTINUE
C
      RETURN
C
      END
```

E.4 Program PLAFIT

The program consists of the following parts.

1. MAIN
2. MAINMG, the main executive subroutine
3. The input package
   (a) INPUT1 and INPUT2
   (b) BOUND, JLOAD2, and MLOAD2
4. The data managing package
   ISPAC, LOCOM, REMOV, REMOV2, and BLOCK DATA
5. The data storage and retrieval package
   CLEAR, ICLEAR, RTRV1, RTRV2, STORE1, and STORE2.
6. The equation solving package
   (a) ADDRES and COLHT
   (b) EQSBST and BKSB1
7. The formulation and output package
   (a) ASSEMB, ELEMS, MODIFY, BOUND1, DISPL1, STRES1 and MNHING
   (b) STIFFP and HINGE

The listing for packages 1, 2, 3a, 6b, and 7b follows. The listing of parts 3b, 4, 5, 6a, and 7a can be found in Sects. E.8, E.6, E.7, E.9, and E.10, respectively. Descriptions of the subroutines can be found in the listing.

FILES 1 and 2 are temporary files that are used in sequential I/O operations. Both are line-files.

```
C*************************************************************
C
C                       PLAFIT
C                       ******
C
C         THIS IS A PROGRAM FOR ELASTO-PLASIC ANALYSIS
C       OF PLANER FRAMED TYPE STRUCTURES. THE PROGRAM
C       EVALUATES THE RESPONSE OF THE STRUCTURE USING AN
C       ITERATIVE TECHNIQUE.
C
C       ***********************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 NAMES
C
      COMMON/PROBIA/ NNN(400)
      COMMON/IA1/    MMM(200)
      COMMON/RA1/    AAA(4000)
      COMMON/RA2/    BBB(1000)
      COMMON/RA3/    CCC(2000)
      COMMON/DIMCOM/LAST1,LAST2,LAST3,LAST4,LAST5,MXDIM,
     *NAMES(5,20),IPT(5,21),ICOM(5)
C
C     ****************************************************
C
      ICOM(1) = 400
      ICOM(2) = 4000
      ICOM(3) = 200
      ICOM(4) = 1000
      ICOM(5) = 2000
C
      CALL MAINMG
C
      END
```

```
C*************************************************************
C
                SUBROUTINE MAINMG
C               ****************
C
C            THIS IS THE MAIN MANAGER OF PROGRAM PLAFIT. IT
C        CONTROLS THE READING , THE FORMULATION ,THE SOLUTION
C        AND THE OUTPUT OF THE RESULTS.
C
C        ******************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 NAMES,NAME
C
      COMMON/PROBCV/ UNIT,NJ,NE,NEBEL,MNHE,ITAF,NLC,IN,IO
C
      COMMON/PROBIA/ NNN(1)
      COMMON/IA1   / MMM(1)
      COMMON/RA1   / AAA(1)
      COMMON/RA2   / BBB(1)
      COMMON/RA3   / CCC(1)
      COMMON/DIMCOM/ LA1,LA2,LA3,LA4,LA5,MXDIM,NAMES(5,20)
     *,IPT(5,21),ICOM(5)
C
C     ******************************************************
C
      CALL TIME (0,0)
      IN = 5
      IO = 6
C
C
C   READ PROBLEM CONTROL VARIABLES
C   ------------------------------C
C
      CALL INPUT1
C
      NODE  = 2
      IDOF = 3
      ILOAD = 1
      NEQ   = NJ * IDOF
      CALL CLEAR (NAMES(1,1),100)
C
C   READ NODAL GEOMETRY AND MEMBER PROP. AND CONNECTIVITIES
C   ----------------------------------------------------C
      J1 = ISPAC (1HX,NJ,2)
      J2 = ISPAC (1HY,NJ,2)
      J3 = ISPAC (4HAREA,NE,2)
      J4 = ISPAC (2HRI,NE,2)
      J5 = ISPAC (2HPM,(2*NE),2)
      J6 = ISPAC (4HYMOD,NE,2)
      J7 = ISPAC (3HPSM,NE,2)
      J8 = ISPAC (2HYS,NE,2)
      I1 = ISPAC (3HNOD,(NE*NODE),1)
      I2 = ISPAC (5HMKODE,NE,1)
```

```
      I3 = ISPAC (5HMFLAG,NE,1)
C
      CALL INPUT 2 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J5),
     *AAA(J6),AAA(J7),AAA(J8),NNN(I1),NNN(I2),NNN(I3))
C
C
C    READ EXTERNAL BOUNDARY CONDITIONS
C
      J9 = ISPAC (3HBES,NEBEL,2)
      K1 = ISPAC (3HNPB,NEBEL,3)
      K2 = ISPAC (4HKODE,NEBEL,3)
      CALL BOUND (AAA(J9),MMM(K1),MMM(K2),NEBEL,IN,IO)
C
C
      INN = IDOF * NODE * NE
      J10 = ISPAC (4HFEFC,INN,2)
      J11 = ISPAC (4HFEFV,INN,2)
      CALL CLEAR (AAA(J10),(INN*2))
C
      L1 = ISPAC (2HBC,NEQ,4)
      L2 = ISPAC (2HBV,NEQ,4)
      CALL CLEAR (BBB(L1),(NEQ*2))
C
C    READ LOADS IF ANY
C
      READ (IN,1000) JLF,MLF
      IF(JLF.EQ.0) GO TO 50
      CALL JLOAD2 (BBB(L1),BBB(L2),AAA(J9),IDOF,IN,IO)
50    IF(MLF.EQ.0) GO TO 60
      CALL  MLOAD2 (AAA(J1),AAA(J2),AAA(J10),AAA(J11),
     *BBB(L1),BBB(L2),NNN(I1),NNN(I2),UNIT,IDOF,NJ,IN,IO)
60    CONTINUE
C
C    FORM COLUMN HEIGHTS AND ADDRESSING ARRAY
C
      I5 = ISPAC (4HMAXA,(NEQ+1),1)
      I6 = ISPAC (3HMHT,NEQ,1)
      CALL ICLEAR (NNN(I6),NEQ)
      DO 70 I=1,NE
      CALL COLHT (NODE,(NODE*IDOF),IDOF,I,NNN(I6),
     *NNN(I1))
70    CONTINUE
C
      CALL ADDRES (NNN(I5),NNN(I6),NEQ,NWA)
      CALL REMOV (3HMHT,1)
C
      L3  = ISPAC (2HBT,NEQ,4)
      L4  = ISPAC (2HDT,NEQ,4)
      CALL CLEAR (BBB(L3),(NEQ*2))
      L5  = ISPAC (2HFL,(NE*2),4)
      L6  = ISPAC (3HBTT,NEQ,4)
      J12 = ISPAC (5HFEFC2,INN,2)
      J13 = ISPAC (5HFEFV2,INN,2)
      J16 = ISPAC (3HTEF,INN,2)
```

```
C
      CALL CLEAR (AAA(J12),(INN*2))
      K3 = ISPAC (5HMAXNH,NJ,3)
      CALL ICLEAR (MMM(K3),NJ)
      CALL MNHING (NNN(I1),NNN(I2),MMM(K1),MMM(K3),NE,
     *NJ,NEBEL)
C
      REWIND 1
      CALL STORE2 (1,BBB(L1),NEQ)
      CALL STORE2 (1,AAA(J10),INN)
      REWIND 2
      CALL STORE2 (2,BBB(L2),NEQ)
      REWIND 2
C RESERVE AND CLEAR SPACE FOR ELASTIC
C     STIFFNESS MATRIX
C
      NH = 0
      NV = 0
      IT = 0
      M1 = ISPAC (1HA,NWA,5)
90    CALL CLEAR (CCC(M1),NWA)
      CALL CLEAR (BBB(L5),(NE*2))
C      FORM ELEMENT STIFFNESS AND ASSEMBLE
C   ELASTIC STIFFNESS INTO A.
      CALL STIFFP (NNN(I1),NNN(I2),NNN(I3),NNN(I5),AAA(J1),
     *AAA(J2),AAA(J3),AAA(J4),AAA(J6),AAA(J8),AAA(J5),
     *CCC(M1),BBB(L1),AAA(J10),AAA(J16),UNIT,NE,ITAF,IT)
C
C      ADD EXTERNAL BOUNDRAY CONDITIONS
C
      CALL BOUND1 (CCC(M1),NNN(I5),MMM(K1),MMM(K2),
     *AAA(J9),NEBEL)
      CALL EQSBST (CCC(M1),BBB(L1),BBB(L2),NNN(I5),NEQ,
     *NLC,ILOAD,0)
95    CALL BKSB1 (BBB(L1),BBB(L2),CCC(M1),NNN(I5),NEQ,NLC,0)
      IF(ILOAD.EQ.999) NV = 1
      IF(ITAF.EQ.0.AND.ILOAD.EQ.999) GO TO 900
      IF(NLC.EQ.1) GO TO 200
      CALL DISPL1 (AAA(J12),BBB(L1),AAA(J10),NJ,NE,IDOF,
     *IN,IO)
      CALL STRES1 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J6),
     *BBB(L1),AAA(J12),AAA(J16),NNN(I1),NNN(I2),NE,
     *IDOF,UNIT,IN,IO)
      DO 100 I=1,NEQ
      BBB(L4+I-1) = BBB(L1+I-1)
100   CONTINUE
C
200   CALL DISPL1 (AAA(J13),BBB(L2),AAA(J11),NJ,NE,IDOF,
     *IN,IO)
      CALL STRES1 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J6),
     *BBB(L2),AAA(J13),AAA(J16),NNN(I1),NNN(I2),NE,
     *IDOF,UNIT,IN,IO)
      IF(NH.EQ.0.AND.IT.EQ.0) CALL RTRV2 (2,BBB(L3),NEQ)
      REWIND 1
```

```
      CALL RTRV2 (1,BBB(L1),NEQ)
      CALL RTRV2 (1,AAA(J10),INN)
C
C
350   CALL HINGE (AAA(J3),AAA(J7),AAA(J8),NNN(I1),NNN(I2),
     *BBB(L5),BBB(L3),BBB(L1),BBB(L6),BBB(L2),BBB(L4),
     *AAA(J13),AAA(J12),AAA(J16),MMM(K3),NE,NJ,NEQ,INN,
     *IDOF,ITAF,UNIT,NH,IT,IO)
C
C
C
      DO 500 I=1,NEQ
      BBB(L2+I-1) = BBB(L3+I-1)
500   CONTINUE
      CALL CLEAR (BBB(L4),NEQ)
      IF(ITAF.EQ.0) GO TO 550
      IF(IT.EQ.999) GO TO 550
      IT = IT + 1
      GO TO 90
550   IF(NH.GT.MNHE) GO TO 900
      NLC = 2
      NH = NH + 1
      IT = 0
      WRITE(IO,2000) NH
      CALL TIME (3,3)
      IF(NV.EQ.1) GO TO 900
      GO TO 90
C
900   RETURN
C
C  FORMAT STATEMENTS
C
1000  FORMAT (2I5)
2000  FORMAT (//,10X,30(1H*)/,10X,'HINGE NO.',I4,2X,
     *'HAS BEEN FORMED'/,10X,30(1H*)//)
C
      END
```

```
C***********************************************************************
C
                 SUBROUTINE INPUT1
C                ****************
C
C
C           THIS SUBROUTINE READS THE PROBLEM CONTROL VARIABLES
C        FOR PROGRAM PLAFIT.
C
C        ******************************************************
C
         IMPLICIT REAL*8(A-H,O-Z)
C
         COMMON/PROBCV/ UNIT,NJ,NE,NEBEL,MNHE,ITAF,NLC,IN,IO
C
         DIMENSION HEAD(20)
C
         READ (IN,1000) HEAD
         WRITE(IO,2000) HEAD
         READ (IN,1100) NJ,NE,NEBEL,NLC,ITAF,MNHE,UNIT
         WRITE(IO,2100) NJ,NE,NEBEL,NLC,ITAF,MNHE,UNIT
C
         RETURN
C
C    FORMAT STATEMENTS
C
1000     FORMAT (20A4)
1100     FORMAT (6I4,F12.0)
2000     FORMAT (//,30X,20A4//,30X,30(1H*)////)
2100     FORMAT (5X,'PROBLEM CONTROL VARIABLES'/,5X,25(1H*)//,
        *10X,'NUMBER OF JOINTS                          =',I5//,
        *10X,'NUMBER OF ELEMENTS                        =',I5//,
        *10X,'NO. OF EXTERNAL BOUNDARY ELEMENTS         =',I5//,
        *10X,'NUMBER OF LOAD CASES                      =',I5//,
        *10X,'IDENTIF. OF TYPE OF ANALYSIS FLAG         =',I5//,
        *10X,'MAXIMUM NO. OF HINGES ESTIMATED           =',I5//,
        *10X,'LENGTH CONVERSION FACTOR                  =',F12.6//,
        *10X,50(1H*))
C
         END
```

```
C*************************************************************
C
            SUBROUTINE INPUT 2 (X,Y,AREA,RI,PM,YMOD,PSM,YS,NOD,
     *                                              MKODE,MFLAG)
C
C             THIS SUBROUTINE READS THE NODAL GEOMETRY AND
C         MEMBER PROPERTIES FOR PROGRAM PLAFIT.
C
C     ******************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON/PROBCV/ UNIT,NJ,NE,NEBEL,MNHE,ITAF,NLC,IN,IO
      DIMENSION X(1),Y(1),AREA(1),RI(1),YMOD(1),NOD(2,1),
     *MKODE(1),MFLAG(1),PSM(1),YS(1),PM(2,1)
C
      WRITE(IO,2000)
50    READ (IN,1000) N,X(N),Y(N),INC
      IF(INC.EQ.0) GO TO 200
      NINT = (N-NOLD) / INC
      RN   = NINT
      IF(RN.LT.FLOAT(N-NOLD)/FLOAT(INC)-0.001) GO TO 999
      DX = (X(N)-X(NOLD))/RN
      DY = (Y(N)-Y(NOLD))/RN
      L = NOLD
      M = NINT - 1
      DO 100 J=1,M
      LL = L + INC
      X(LL) = X(L) + DX
      Y(LL) = Y(L) + DY
      L = LL
100   CONTINUE
200   WRITE (IO,2100) N,X(N),Y(N),INC
      NOLD = N
      IF(N.LT.NJ) GO TO 50
C
      READ (IN,1100) ADEF,RDEF,SMDEF,YSDEF,YDEF
      WRITE(IO,2200) ADEF,RDEF,SMDEF,YSDEF,YDEF
C
      WRITE(IO,2300)
300   READ (IN,1200) M,(NOD(I,M),I=1,2),MKODE(M),MFLAG(M),
     *INC,NOD1,NOD2
      READ (IN,1300) AREA(M),RI(M),PSM(M),YS(M),YMOD(M)
      IF(AREA(M).EQ.0) AREA(M) = ADEF
      IF(RI(M).EQ.0)   RI(M)   = RDEF
      IF(PSM(M).EQ.0) PSM(M)   = SMDEF
      IF(YS(M).EQ.0)  YS(M)    = YSDEF
      IF(YMOD(M).EQ.0) YMOD(M) = YDEF
      PM(1,M) = PSM(M) * YS(M)
      PM(2,M) = PM(1,M)
      NN = MKODE(M) + 1
      GO TO (340,310,320,330),NN
310   PM(1,M) = 0.0
      GO TO 340
```

```
320   PM(2,M) = 0.0
      GO TO 340
330   PM(1,M) = 0.0
      PM(2,M) = 0.0
340   IF(INC.EQ.0) GO TO 500
      NINT = (M-MOLD) / INC - 1
      L = MOLD
      DO 400 I=1,NINT
      LL        =L + INC
      AREA(LL)  = AREA(M)
      RI(LL)    = RI(M)
      PSM(LL)   = PSM(M)
      YS(LL)    = YS(M)
      YMOD(LL)  = YMOD(M)
      MKODE(LL) = MKODE(M)
      MFLAG(LL) = MFLAG(M)
      NOD(1,LL) = NOD(1,L) + NOD1
      NOD(2,LL) = NOD(2,L) + NOD2
400   L = LL
C
500   WRITE(IO,2400) M,NOD(1,M),NOD(2,M),MKODE(M),MFLAG(M),
     *INC,NOD1,NOD2,AREA(M),RI(M),PSM(M),YS(M),YMOD(M)
      MOLD = M
      IF(M.LT.NE) GO TO 300
      READ (IN,1400) K
      IF(K.EQ.0) GO TO 600
      WRITE(IO,2700)
      WRITE(IO,2750) (N,X(N),Y(N),N=1,NJ)
      WRITE(IO,2800)
      WRITE(IO,2850) (M,(NOD(I,M),I=1,2),MKODE(M),MFLAG(M),
     *AREA(M),RI(M),YMOD(M),M=1,NE)
C
600   RETURN
C
999   WRITE (IO,9999) N
9999  FORMAT('NODAL GEOMETRY DATA INPUT ERROR',I5)
      STOP
C
C        FORMAT STATEMENTS
C
1000  FORMAT(I5,2F12.0,I5)
1100  FORMAT(5F12.0)
1200  FORMAT(8I4)
1300  FORMAT (5F12.0)
1400  FORMAT(I5)
2000  FORMAT (///,5X,'NODAL GEOMETRY DATA AS INPUT'/,
     *5X,28(1H*)//,10X,1HN,8X,1HX,14X,1HY,9X,3HINC/)
2100  FORMAT (6X,I5,2D15.6,I5)
2200  FORMAT (///,5X,'MEMBER PROPERTIES DEFAULT VALUES'/,
     *5X,32(1H*)//,10X,'AREA',20X,' =',D15.6,//,
     *10X,'M. INERTIA',15X,'=',D15.6,//,
     *10X,'PLASTIC SECTION MODULUS  =',D15.6//,
     *10X,'YIELD STRESS             =',D15.6//,
     *10X,'Y. MODULUS',15X,'=',D15.6)
```

```
2300  FORMAT (///,5X,'MEMBER DATA AS INPUT'/,5X,20(1H*)//,
     *4X,1HM,4X,1HI,4X,1HJ,2X,4HCODE,1X,4HFLAG,2X,3HINC,
     *1X,4HINCI,1X,4HINCJ,3X,4HAREA,10X,1HI,
     *11X,3HPSM,9X,2HYS,10X,4HYMOD/)
2400  FORMAT (8I5,5D12.4)
2700  FORMAT (///,5X,'COMPLETED NODAL GEOMETRY DATA'/,
     *5X,29(1H*)//,4X,1HN,7X,1HX,14X,1HY/)
2750  FORMAT (I5,2D15.6)
2800  FORMAT (///,5X,'COMPLETED MEMBER DATA'/,5X,
     *20(1H*)//,4X,1HM,4X,1HI,4X,1HJ,1X,4HCODE,5X,
     *4HAREA,10X,1HI,13X,4HYMOD/)
2850  FORMAT (5I5,3D15.6)
C
      END
```

```
C*************************************************************
C
           SUBROUTINE  EQSBST (A,BC,BV,MAXA,NB,NLC,ILOAD,IT)
C          *****************
C
C            THIS SUBROUTINE TRIANGULARIZES A STIFFNESS MATRIX
C         STORED COLUMNWISE UNDER A SKYLINE AND REDUCES THE
C         CORRESPONDING LOAD VECTOR, DOWN TO EQUATION NB.
C
C         *****************************************************
          IMPLICIT REAL*8(A-H,O-Z)
C
          DIMENSION A(1),BC(1),BV(1),MAXA(1)
C
C
          DO  1000 N=1,NB
          KN = MAXA(N)
          KL = KN + 1
          KU = MAXA(N+1) - 1
          KH = KU - KL
          IF(KH) 900,500,100
 100      IF(ILOAD.EQ.2.AND.IT.GT.0) GO TO 650
          K  = N - KH
          IC = 0
          KLT= KU
          DO 400 J=1,KH
          IC = IC + 1
          KLT= KLT - 1
          KI = MAXA(K)
          ND = MAXA(K+1) - KI - 1
          IF(ND) 400,400,200
 200      KK = IC
          IF(KK.GT.ND) KK = ND
          C  = 0.0
          DO 300 L = 1,KK
 300      C  = C + A(KI+L)*A(KLT+L)
          A(KLT) = A(KLT) - C
 400      K  = K + 1
 500      IF(ILOAD.EQ.2.AND.IT.GT.0) GO TO 650
          K  = N
C
          C  = 0.0
          DO 600 KK = KL,KU
          K  = K - 1
          KI = MAXA(K)
          D  = A(KK)/A(KI)
          C  = C + D*A(KK)
          A(KK) = D
 600      CONTINUE
          A(KN) = A(KN) - C
C
 650      CONTINUE
          K = N
          C = 0.0
```

```
      DO 700 KK=KL,KU
      K = K - 1
700   C = C + A(KK) * BV(K)
      BV(N) = BV(N) - C
      IF(NLC.EQ.1) GO TO 900
      K = N
      C1 = 0.0
      DO 800 KK=KL,KU
      K = K - 1
800   C1 = C1 + A(KK) * BC(K)
      BC(N) = BC(N) - C1
C
900   IF(A(KN)) 975,950,1000
950   WRITE(6,3000) N,A(KN)
      STOP
975   ILOAD = 999
C
1000  CONTINUE
C
      RETURN
C
3000  FORMAT('ZERO OR NEGATIVE ELEMENT ON MAIN DIAGONAL NO.'
     *,I4,D15.6)
C
      END
```

```
C*****************************************************
C
            SUBROUTINE BKSB1 (BC,BV,A,MAXA,NN,NLC,KB)
C           ****************
C
C          THIS SUBROUTINE IS A PART OF THE PARTIAL REDUCTION
C       PACKAGE. IT FORMS THE LAST STAGE OF BACKSUBSTITUTION,
C       THE OPERATION (L(-1)(T)*RI*) .    IN CASE OF AN
C       UNSUBSTRUCTURED PROBLEM IT PERFORMS THE OPERATION
C       //D(-1)*L(-1)(T)*R//
C
C       *****************************************************
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION A(1),BC(1),BV(1),MAXA(1)
C
        IF(KB.NE.0) GO TO 200
        DO 100 J=1,NN
        K = MAXA(J)
        BV(J) = BV(J) / A(K)
        IF(NLC.EQ.1) GO TO 100
        BC(J) = BC(J) / A(K)
  100   CONTINUE
  200   N = NN
        DO 600 L=2,NN
        KL = MAXA(N) + 1
        KU = MAXA(N+1) - 1
        IF(KU-KL) 600,300,300
  300   IF(BV(N).EQ.0) GO TO 500
        K = N
        DO 400 KK=KL,KU
        K = K - 1
  400   BV(K) = BV(K) - A(KK) * BV(N)
  500   IF(NLC.EQ.1) GO TO 600
        IF(BC(N).EQ.0) GO TO 600
        K = N
        DO 550 KK=KL,KU
        K = K - 1
  550   BC(K) = BC(K) - A(KK) * BC(N)
  600   N = N - 1
C
        RETURN
C
        END
```

```
C***********************************************************
C
        SUBROUTINE STIFFP (NOD,MKODE,MFLAG,MAXA,X,Y,AREA,RI,
     *                 YMOD,YS,PM,A,B,FEFC,PPC,UNT,NE,ITAF,IT)
C
C
C          THIS SUBROUTINE FORMS THE ELASTIC STIFFNESS
C       MATRIX OF 6 DOF PLANE FRAME MEMBER. THE UPPER
C       TRIANGLE IS STORED COLUMN-WISE IN A VECTOR.
C
C       ***********************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
C
        DIMENSION NOD(2,1),MKODE(1),MFLAG(1),MAXA(1),
     *X(1),Y(1),AREA(1),RI(1),YMOD(1),A(1),PPC(1),LM(6)
     *,YS(1),FEFC(1),B(1),AS(7),AC(7),S(21),PM(2,1)
C
C
      DO 700 M=1,NE
      K = MKODE(M)
      I = NOD(1,M)
      J = NOD(2,M)
      PMCI = PM(1,M)
      PMCJ = PM(2,M)
      MI = 6 * M - 3
      MJ = 6 * M
      MII = MJ - 5
      PY  = AREA(M) * YS(M)
      PPY = PPC(MII) / PY
      IF(DABS(PPY).LE.0.15) GO TO 50
      PMCI = (1-PPY) * PMCI * 1.18
      PMCJ = (1-PPY) * PMCJ * 1.18
50    IF(PPC(MI).GT.0.0) PMCI = - PMCI
      IF(PPC(MJ).GT.0.0) PMCJ = - PMCJ
      DX = (X(J) - X(I)) * UNT
      DY = (Y(J) - Y(I)) * UNT
      XL = DSQRT (DX**2 + DY**2)
      XLI = 1. / XL
      CO = DX * XLI
      SI = DY * XLI
      CL = CO * XLI
      SL = SI * XLI
      CL2 = CL ** 2
      SL2 = SL ** 2
      CSL = CL * SL
      C2  = CO ** 2
      S2  = SI ** 2
      SC  = CO * SI
      ALP = YMOD(M) * RI(M) * XLI
      BETA= YMOD(M) * AREA(M) * XLI
C
C   FORM LM ARRAY
```

```
C
      LM(3) = 3 * I
      LM(2) = LM(3) - 1
      LM(1) = LM(3) - 2
      LM(6) = 3 * J
      LM(5) = LM(6) - 1
      LM(4) = LM(6) - 2
C
C        FORMULATION OF ELASTIC STIFFNESS MATRIX
C
      AS(1) = BETA
      AS(2) = 12.0 * ALP
      AS(3) = 6.0 * ALP
      AS(4) = AS(3)
      AS(5) = 4.0 * ALP
      AS(6) = 2.0 * ALP
      AS(7) = AS(5)
      IF(K.NE.0) CALL MODIFY(K,AS)
C
C        FORM ELASTIC STIFFNESS MATRIX
C
      CALL ELEMS(AS,S,CL,SL,CL2,SL2,CSL,C2,S2,SC)
      CALL ASSEMB (MAXA,S,A,LM,6)
      IF(ITAF.EQ.0) GO TO 150
C
C         FORMULATION OF GEOMETRIC ELEMENT STIFFNESS
      L = MFLAG(M)
      IF(L.EQ.0) GO TO 700
      M1 = 6 * M - 5
      AC(1) = 0.0
      AC(2) =  XL
      AC(3) = 0.0
      AC(4) = AC(3)
      AC(5) = 0.0
      AC(6) = 0.0
      AC(7) = AC(5)
      CALL ELEMS (AC,S,CL,SL,CL2,SL2,CSL,C2,S2,SC)
      DO 100 KK=1,21
100   S(KK) = - S(KK) * PPC(M1)
C
C
      CALL ASSEMB (MAXA,S,A,LM,6)
C
150   IF(K.EQ.0) GO TO 700
      II = LM(3)
      JJ = LM(6)
      GO TO (200,300,400),K
200   AC(3) = PMCI
      AC(2) =  1.5 * XLI * PMCI
      AC(6) =  0.5 * PMCI
      AC(5) = - AC(2)
      GO TO 450
300   AC(3) =  0.5 * PMCJ
      AC(2) = 1.5 * PMCJ * XLI
```

```
      AC(6) =  PMCJ
      AC(5) = - 1.5 * PMCJ * XLI
      GO TO 450
400   AC(3) = PMCI
      AC(2) = XLI * (PMCI + PMCJ)
      AC(6) = PMCJ
      AC(5) = - AC(2)
450   CONTINUE
      B(II)   = B(II)   + AC(3)
      B(II-1) = B(II-1) + AC(2) * CO
      B(II-2) = B(II-2) - AC(2) * SI
      B(JJ)   = B(JJ)   + AC(6)
      B(JJ-1) = B(JJ-1) + AC(5) * CO
      B(JJ-2) = B(JJ-2) - AC(5) * SI
      KEF = 6 * M - 6
      DO 500 KK=1,6
      KEF = KEF + 1
      FEFC(KEF) = FEFC(KEF) - AC(KK)
500   CONTINUE
700   CONTINUE
C
      RETURN
C
      END
```

```
C*********************************************************************
C
                SUBROUTINE HINGE (AREA,PSM,YS,NOD,MKOD,FL,B,BC,
     *                             BT,D,DT,EMF,EMFT,TEF,MAXNH,NE,
     *                           NJ,NEQ,INN,IDO,ID,UNIT,NH,IT,IO)
C
C          THIS SUBROUTINE EVALUATES THE ABSOLUTE VALUE OF
C       M /(PM - M) AT EACH NODE FOR EACH MEMBER AND FIND THE
C       LARGEST VALUE AND INSERT HINGE AT THIS NODE.  IT ALSO
C       UPDATES LOADS , DISPLACEMENTS AND MEMBER FORCES.
C
C       *****************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION PSM(1),YS(1),NOD(2,1),MKOD(1),B(1),BT(1),
     *D(1),DT(1),AREA(1),EMF(1),EMFT(1),FL(2,1),MAXNH(1),
     *TEF(1),BC(1)
C
        DO 100 M=1,NE
        KK = 3
        DO 50 I=1,2
        PMC = PSM(M) * YS(M) / UNIT
        MJ  = 6 * M - KK
        IF(DABS(EMF(MJ)).LT.0.0000001) GO TO 50
        PY  = AREA(M) * YS(M)
        MI = 6 * M - 5
        PPY = (EMFT(MI) + EMF(MI)) / PY
        IF(DABS(PPY).GT.0.15) PMC =(1-PPY) *PMC * 1.18
20      IF(NH.EQ.0.AND.IT.EQ.0) GO TO 30
        IF(TEF(MJ).GT.0.0) PMC = -PMC
        GO TO 40
30      IF(EMF(MJ).GT.0) PMC = -PMC
40      PMMJ = PMC + EMFT(MJ) + EMF(MJ)
        IF(PMMJ.EQ.0.0) GO TO 50
        FL(I,M) = EMF(MJ) / PMMJ
50      KK = 0
100     CONTINUE
        IF(IT.EQ.0) GO TO 150
        K = KOLD
        K1 = K1OLD
        ABMAX = DABS(FL(K1,K))
        GO TO 225
150     K  = 0
        K1 = 0
        ABMAX = 0.0
        DO 200 M=1,NE
        DO 200 I=1,2
        C = FL(I,M)
        IF(DABS(C).LT.ABMAX) GO TO 200
        K  = M
        K1 = I
        ABMAX = DABS(C)
200     CONTINUE
```

```
      KOLD = K
      K1OLD = K1
225   FLC = - 1. / FL(K1,K)
      IF(ID.EQ.0) GO TO 230
      IF(DABS(FLC).LT.0.0001) IT = 999
      IF(IT.NE.999) GO TO 275
230   MKOD(K) = MKOD(K) + K1
      FL(K1,K) = 0.0
      JN       = NOD(K1,K)
      MAXNH(JN)= MAXNH(JN) - 1
      ABMAX = ABMAX - .0000001
      DO 250 M=1,NE
      DO 250 L=1,2
      C = FL(L,M)
      I = NOD(L,M)
      IF(DABS(C).LT.ABMAX) GO TO 250
      IF(MAXNH(I).EQ.0) GO TO 250
      MKOD(M) = MKOD(M) + L
      MAXNH(I) = MAXNH(I) - 1
250   CONTINUE
275   DO 300 I=1,NEQ
      B(I)  = B(I) * (FLC + 1)
      BT(I) = BC(I) + B(I)
      DT(I) = DT(I) + D(I) * ( FLC + 1 )
300   CONTINUE
      DO 400 I=1,INN
      TEF(I) = EMFT(I) + EMF(I) * ( FLC + 1 )
400   CONTINUE
      WRITE(IO,2000)
      DO 500 N=1,NJ
      N2 = N * IDO
      N1 = N2 - IDO + 1
500   WRITE(IO,2100) N,(BT(NN),NN=N1,N2)
575   WRITE(IO,2200)
      DO 600 N=1,NJ
      N2 = N * IDO
      N1 = N2 - IDO + 1
600   WRITE(IO,2100) N,(DT(NN),NN=N1,N2)
      WRITE(IO,2300)
      DO 700 M=1,NE
      M2 = 6 * M
      M1 = M2 - 5
700   WRITE(IO,2400) M,NOD(1,M),NOD(2,M),(TEF(I),I=M1,M2)
C
      RETURN
C
C  FORMAT STATEMENTS
C
2000  FORMAT (///'TOTAL EXTERNAL NODAL LOADS',//,4X,'N',7X,
     *'FU',13X,'FV',13X,'FR'/)
2100  FORMAT (I5,3D15.6)
2200  FORMAT (//,'TOTAL NODAL DISPLACEMENTS',//,4X,'N',7X,
     *'U',14X,'V',14X,'R'/)
2300  FORMAT (//,'TOTAL MEMBER END FORCES'//,3X,'MEM',3X,
```

```
     *'I',4X,'J',7X,'NI',13X,'VI',13X,'MI',13X,'NJ',13X,
     *'VJ',13X,'MJ'/)
2400  FORMAT (3I5,6D15.6)
C
      END
```

E.5 Program INSTAF

The program consists of the following parts.

1. MAIN
2. MAINMG, the main executive subroutine
3. The input package.
   (a) INPUT1 and INPUT2
   (b) BOUND, JLOAD2, and MLOAD2
4. The data managing package
   ISPAC, LOCOM, REMOV, REMOV2, and BLOCK DATA
5. The data storage and retrieval package
   CLEAR, ICLEAR, RTRV1, RTRV2, STORE1, and STORE2.
6. The equation solving package
   (a) ADDRES and COLHT
   (b) EQSBST, BKSB1, EQFT, EQKBB, and BKSB2
7. The formulation and output package
   (a) ASSEMB, BOUND2, DISPL1, MULT2, SHAPE, DINCR, and CONVER
   (b) STIFF, TR1, TR2, STIFFB, STRAIN, STEP, and STEP1

The listing for packages 1, 2, 3a, 6b, and 7b follows. The listing for parts 3b, 4, 5, 6a, and 7a are given in Sects. E.8, E.6, E.7, E.9, and E.10, respectively. Descriptions of the subroutines can be found in the listing.

FILES 1, 2, 4, 9, and 10 are all line-files used in this program. FILES 1 and 2 are temporary files while FILES 4, 9, and 10 are permanent. Direct access is used on file 4. Remaining files are used exclusively in sequential I/O operations.

```
C*****************************************************
C
C                      INSTAF
C                      ******
C
C          THIS IS A PROGRAM FOR NONLINEAR INELASTIC
C       STABILITY ANALYSIS OF PLANER FRAMED TYPE STRUCTURES.
C
C       ***************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
        REAL*8 NAMES
C
        COMMON/PROBIA/ NNN(2000)
        COMMON/IA1/    MMM(200)
        COMMON/RA1/    AAA(10000)
        COMMON/RA2/    BBB(1000)
        COMMON/RA3/    CCC(10000)
        COMMON/DIMCOM/LAST1,LAST2,LAST3,LAST4,LAST5,MXDIM,
       *NAMES(5,20),IPT(5,21),ICOM(5)
C
C       ****************************************************
C
        ICOM(1) = 2000
        ICOM(2) = 10000
        ICOM(3) = 200
        ICOM(4) = 1000
        ICOM(5) = 10000
C
        CALL MAINMG
C
        END
```

```
C*********************************************************
C
              SUBROUTINE MAINMG
C             ****************
C
C         THIS IS THE MAIN MANAGER OF PROGRAM INSTAF. IT
C      CONTROLS THE READING , THE FORMULATION ,THE SOLUTION
C      AND THE OUTPUT OF THE RESULTS.
C
C      ***************************************************
C
       IMPLICIT REAL*8(A-H,O-Z)
       REAL*8 NAMES,NAME
C
       COMMON/PROBCV/ UNIT,FL1,FL2,TOL,NJ,NE,NEBEL,NDXST,
      *NDMT,NDRT,IN,IO,ITLIM,IREF,IRCO,IC,NFE,NMT
C
       COMMON/PROBIA/ NNN(1)
       COMMON/IA1   / MMM(1)
       COMMON/RA1   / AAA(1)
       COMMON/RA2   / BBB(1)
       COMMON/RA3   / CCC(1)
       COMMON/DIMCOM/ LA1,LA2,LA3,LA4,LA5,MXDIM,NAMES(5,20)
      *,IPT(5,21),ICOM(5)
C
       DIMENSION XGAUS(4)
C
       DATA XGAUS/-.86113631,-.33998104,.33998104,.86113631/
C
C       ***************************************************
C
       CALL TIME (0,0)
       IN = 5
       IO = 6
C
       REWIND 1
       REWIND 2
       REWIND 4
       REWIND 9
       REWIND 10
C
C    READ PROBLEM CONTROL VARIABLES
C    ------------------------------C
C
       CALL INPUT1
C
       IF(IREF.EQ.0) GO TO 40
C
       READ (10) NJ,NFE,NE,NMT,IDOF,NGAUS,NEQ,NEBEL,ITLIM,
      *TOL,UNIT
       READ (10) L1,L2
       CALL RTRV2 (10,BBB(L1),NEQ)
       CALL RTRV2 (10,BBB(L2),NEQ)
       DO 10 I=1,NEQ
```

```
10    BBB(I) = BBB(I) * FL1 + BBB(I+NEQ) * FL2
      CALL STORE2 (1,BBB(L1),NEQ)
      READ (10) NWA,NWK
      READ (10) I1,I3,I4,I5,I6,K0,K1,K2,K3,II1,II3
      READ (10) J1,J2,J3,J4,J5,J6,J7,J8,J9,J10,
     *J11,J12,J13,J14,J15,J16,II2
      READ (10) L1,L2,L3,L4,M0,M1,M2,M3,M4,II5
      CALL RTRV1 (10,NNN(1),II1)
      CALL RTRV1 (10,MMM(1),II3)
      CALL RTRV2 (10,AAA(1),II2)
      CALL RTRV2 (10,CCC(1),II5)
C
      READ (9) LI,ILL
      IRCO = IRCO + 1
      GO TO (15,20) ,IRCO
15    CALL RTRV2 (9,BBB(L2),NEQ)
      CALL RTRV2 (9,BBB(L4),NEQ)
      LI = LI + 1
      CALL STORE2 (10,BBB(L2),NEQ)
      CALL STORE2 (10,BBB(L4),NEQ)
      GO TO 25
20    CALL RTRV2 (10,BBB(L2),NEQ)
      CALL RTRV2 (10,BBB(L4),NEQ)
25    IF(ILL.EQ.999.AND.IC.EQ.1) CALL CLEAR (BBB(L4),NEQ)
C
      WRITE(IO,2000) LI
      WRITE(IO,2100)
      DO 30 N=1,NJ
      N5 = N * 5
      N1 = N5 - 4
30    WRITE(IO,2200) N,(BBB(NN),NN=N1,N5)
C
      CALL STORE2 (2,BBB(L2),NEQ)
      GO TO 85
C
40    NODE  = 2
      IDOF  = 5
      NGAUS = 4
      NEQ   = NJ * IDOF
      NES   = 3
      NJS   = 4
      IDOF1 = 4
      NEQ1  = IDOF1 * NJS
      CALL CLEAR (NAMES(1,1),100)
      LI = 1
C
C   READ NODAL GEOMETRY AND MEMBER PROP. AND CONNECTIVITIES
C   ------------------------------------------------------C
      J1  = ISPAC (1HX,NJ,2)
      J2  = ISPAC (1HY,NJ,2)
      J3  = ISPAC (1HH,NDXST,2)
      J4  = ISPAC (1HW,NDXST,2)
      J5  = ISPAC (2HTF,NDXST,2)
      J6  = ISPAC (2HTW,NDXST,2)
```

```
      J7  = ISPAC (2HEY,NDMT,2)
      J8  = ISPAC (2HEP,NDMT,2)
      J9  = ISPAC (4HEULT,NDMT,2)
      J10 = ISPAC (2HYS,NDMT,2)
      J11 = ISPAC (3HYPS,NDMT,2)
      J12 = ISPAC (4HULTS,NDMT,2)
      J13 = ISPAC (3HEPR,(5*NDRT),2)
      I1  = ISPAC (3HNOD,(NE*NODE),1)
      I2  = ISPAC (5HMKODE,NE,1)
      I3  = ISPAC (4HMXST,NE,1)
      I4  = ISPAC (2HMT,NE,1)
      I5  = ISPAC (3HMRT,NE,1)
      K0  = ISPAC (3HNPS,(NES*NODE),3)
C
      CALL INPUT 2 (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J5),
     *AAA(J6),AAA(J7),AAA(J8),AAA(J9),AAA(J10),AAA(J11),
     *AAA(J12),AAA(J13),NNN(I1),NNN(I3),NNN(I4),NNN(I5),
     *MMM(K0))
C
      J14 = ISPAC (3HFEF,(IDOF*NODE*NE),2)
      CALL CLEAR (AAA(J14),(IDOF*NODE*NE))
      CALL ICLEAR (NNN(I2),NE)
C
C   READ EXTERNAL BOUNDARY CONDITIONS
C
      J15 = ISPAC (3HBES,NEBEL,2)
      K1 = ISPAC (3HNPB,NEBEL,3)
      K2 = ISPAC (4HKODE,NEBEL,3)
      CALL BOUND (AAA(J15),MMM(K1),MMM(K2),NEBEL,IN,IO)
C
      WRITE(10) NJ,NFE,NE,NMT,IDOF,NGAUS,NEQ,NEBEL,ITLIM,
     *TOL,UNIT
C
      L1 = ISPAC (2HBC,NEQ,4)
      L2 = ISPAC (2HBV,NEQ,4)
      CALL CLEAR (BBB(L1),NEQ)
      CALL CLEAR (BBB(L2),NEQ)
C
C   READ LOADS IF ANY
C
      READ (IN,1000) JLF,MLF
      IF(JLF.EQ.0) GO TO 50
      CALL JLOAD2 (BBB(L1),BBB(L2),AAA(J15),IDOF,IN,IO)
50    IF(MLF.EQ.0) GO TO 60
      CALL MLOAD2 (AAA(J1),AAA(J2),AAA(J14),AAA(J14),
     *BBB(L1),BBB(L2),NNN(I1),NNN(I2),UNIT,IDOF,NJ,IN,IO)
60    CONTINUE
      WRITE(10) L1,L2
      CALL STORE2 (10,BBB(L1),NEQ)
      CALL STORE2 (10,BBB(L2),NEQ)
      DO 65 I=1,NEQ
65    BBB(I) = BBB(I) + BBB(I+NEQ)
      CALL REMOV (2HBV,4)
      CALL STORE2 (1,BBB(L1),NEQ)
```

```
C
C    FORM COLUMN HEIGHTS AND ADDRESSING ARRAY
C
      I6 = ISPAC (4HMAXA,(NEQ+1),1)
      I7 = ISPAC (3HMHT,NEQ,1)
      CALL ICLEAR (NNN(I7),NEQ)
      DO 70 I=1,NE
      CALL COLHT (NODE,(NODE*IDOF),IDOF,I,NNN(I7),
     *NNN(I1))
70    CONTINUE
C
      CALL ADDRES (NNN(I6),NNN(I7),NEQ,NWA)
      CALL REMOV (3HMHT,1)
C
      NEQ1 = 16
      K3 = ISPAC (4HMAXS,(NEQ1+1),3)
      K4 = ISPAC (3HMHT,NEQ1,3)
      CALL ICLEAR (MMM(K4),NEQ1)
      DO 77 I=1,3
      CALL COLHT (NODE,(NODE*IDOF1),IDOF1,I,MMM(K4),MMM(K0))
77    CONTINUE
      CALL ADDRES (MMM(K3),MMM(K4),NEQ1,NWK)
      CALL REMOV (3HMHT,3)
C    RESERVE SPACE FOR DERIVATIVES OF SHAPE VECTOR
C
      M0 = ISPAC (3HPHI,(4*NGAUS),5)
      M1 = ISPAC (4HPHIP,(4*NGAUS),5)
      M2 = ISPAC (5HPHIDP,(4*NGAUS),5)
C
C    RESERVE AND CLEAR SPACE FOR TANGENT STIFFNESS MATRIX
C
      M3 = ISPAC (2HTK,NWA,5)
      M4 = ISPAC (3HTKS,NWK,5)
C
      DO 80 IG=1,NGAUS
      XI = XGAUS (IG)
      CALL SHAPE (XI,CCC(M0),CCC(M1),CCC(M2),IG)
80    CONTINUE
C
      L2 = ISPAC (4HDISP,NEQ,4)
      L3 = ISPAC (5HTDISP,NEQ,4)
      L4 = ISPAC (2HSD,NEQ,4)
      CALL CLEAR (BBB(L2),NEQ)
      CALL CLEAR (BBB(L4),NEQ)
      J16 = ISPAC (4HFEF2,(IDOF*NODE*NE),2)
      CALL CLEAR (AAA(J16),(IDOF*NODE*NE))
85    IT = 0
      IF = 1
C    FORM AND ASSEMBLE TANGENT STIFFNESS MATRIX
C
90    CALL CLEAR (CCC(M3),NWA)
      CALL CLEAR (CCC(M4),NWK)
      CALL TIME (3,3)
C
```

```
      CALL STIFF (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J5),
     *AAA(J6),AAA(J7),AAA(J8),AAA(J9),AAA(J10),AAA(J11),
     *AAA(J12),AAA(J13),NNN(I1),NNN(I3),NNN(I4),
     *NNN(I5),NNN(I6),CCC(M0),CCC(M1),CCC(M2),CCC(M3),
     *BBB(L1),BBB(L2),IDOF,NGAUS,NFE,NJ,UNIT,IO,MMM(K3),
     *CCC(M4),MMM(K0),BBB(L4),NWK,IREF,IT,ILL,IC)
C
      IF(NMT.EQ.1) GO TO 92
C
      CALL STIFFB (AAA(J1),AAA(J2),AAA(J3),AAA(J4),AAA(J5),
     *AAA(J6),AAA(J7),AAA(J8),AAA(J9),AAA(J10),AAA(J11),
     *AAA(J12),AAA(J13),NNN(I1),NNN(I3),NNN(I4),NNN(I5),
     *NNN(I6),CCC(M3),BBB(L1),BBB(L2),IDOF,NFE,NE,UNIT,IO)
C
      IF(IC.EQ.1) IC = 2
      CALL TIME (3,3)
C
C  ADD EXTERNAL BOUNDARY CONDITIONS
C
92    CALL BOUND2 (CCC(M3),BBB(L1),NNN(I6),MMM(K1),MMM(K2),
     *AAA(J15),NJ,NEBEL,IO)
C
      IF(IT.GT.1) CALL CONVER (BBB(L1),C,CONVD,IT,TOL,
     *ITLIM,NEQ,IF,IO)
C
      CALL TIME (3,3)
      ILL = 1
      CALL EQSBST (CCC(M3),BBB(L1),NNN(I6),NEQ,ILL)
      IF(ILL.EQ.999.AND.IC.EQ.0) GO TO 200
      CALL BKSB1 (BBB(L1),CCC(M3),NNN(I6),NEQ,0)
      CALL TIME (3,3)
      CALL DISPL1 (AAA(J16),BBB(L1),AAA(J14),NJ,NE,IDOF,
     *IN,IO)
      DO 95 I=1,NEQ
      BBB(L2+I-1) = BBB(L1+I-1)
      BBB(L4+I-1) = BBB(L1+I-1)
95    CONTINUE
      REWIND 1
      CALL RTRV2 (1,BBB(L1),NEQ)
      IT = IT + 1
      CALL TIME(3,3)
      IF(IT.GT.1.OR.IREF.EQ.1) GO TO 100
      CALL STORE2(2,BBB(L2),NEQ)
      GO TO 90
100   REWIND 2
      CALL RTRV2 (2,BBB(L3),NEQ)
      CALL DINCR (BBB(L1),BBB(L2),BBB(L3),NEQ,IT,C,CONVD,
     *NJ,IO)
      REWIND 2
      CALL STORE2 (2,BBB(L2),NEQ)
      IF(IF.EQ.0) GO TO 200
      GO TO 90
200   REWIND 9
      WRITE(9) LI,ILL
```

```
      CALL STORE2 (9,BBB(L2),NEQ)
      CALL STORE2 (9,BBB(L4),NEQ)
      WRITE(IO,2300) ILL
      IF(LI.GT.1) GO TO 300
C
      II1 = IPT (1,LA1+1) - 1
      II2 = IPT (2,LA2+1) - 1
      II3 = IPT (3,LA3+1) - 1
      II5 = IPT (5,LA5+1) - 1
C
      WRITE (10) NWA,NWK
      WRITE (10) I1,I3,I4,I5,I6,K0,K1,K2,K3,II1,II3
      WRITE (10) J1,J2,J3,J4,J5,J6,J7,J8,J9,J10,
     *J11,J12,J13,J14,J15,J16,II2
      WRITE (10) L1,L2,L3,L4,M0,M1,M2,M3,M4,II5
      CALL STORE1 (10,NNN(1),II1)
      CALL STORE1 (10,MMM(1),II3)
      CALL STORE2 (10,AAA(1),II2)
      CALL STORE2 (10,CCC(1),II5)
C
300   RETURN
C
1000  FORMAT (2I5)
2000  FORMAT (10X,'LOAD INCREMENT NO. ',I5//)
2100  FORMAT ('NODAL LOADS',//,4X,'N',7X,'FU',13X,'FV',
     *13X,'FR',12X,'FDU',12X,'FDV'//)
2200  FORMAT (I4,5D15.6)
2300  FORMAT (5X,'ILL  =',I5)
C
      END
```

```
C*************************************************************
C
                SUBROUTINE INPUT1
C               ****************
C
C           THIS SUBROUTINE READS THE PROBLEM CONTROL VARIABLES
C        FOR PROGRAM INSTAF.
C
C        *****************************************************
C
         IMPLICIT REAL*8(A-H,O-Z)
C
         COMMON/PROBCV/ UNIT,FL1,FL2,TOL,NJ,NE,NEBEL,NDXST,
        *NDMT,NDRT,IN,IO,ITLIM,IREF,IRCO,IC,NFE,NMT
C
         DIMENSION HEAD(20)
C
         READ (IN,1000) HEAD
         WRITE(IO,2000) HEAD
         READ (IN,1100) IREF,IRCO,IC,FL1,FL2
         IF(IREF.EQ.1) RETURN
         READ (IN,1200) NJ,NFE,NE,NEBEL,NMT,ITLIM,TOL,UNIT
         READ (IN,1300) NDXST,NDMT,NDRT
         WRITE(IO,2100) NJ,NFE,NE,NEBEL,NMT,NDXST,NDMT,NDRT,
        *ITLIM,TOL,UNIT
C
         RETURN
C
C     FORMAT STATEMENTS
C
1000     FORMAT (20A4)
1100     FORMAT (3I5,2F12.0)
1200     FORMAT (6I4,2F12.0)
1300     FORMAT (3I5)
2000     FORMAT (//,30X,20A4//,30X,20(1H*)////)
2100     FORMAT (5X,'PROBLEM CONTROL VARIABLES'/,5X,25(1H*)//,
        *10X,'NUMBER OF JOINTS                        =',I5//,
        *10X,'NUMBER OF FLEXURAL ELEMENTS             =',I5//,
        *10X,'NUMBER OF ELEMENTS                      =',I5//,
        *10X,'NO. OF EXTERNAL BOUNDARY ELEMENTS       =',I5//,
        *10X,'NUMBER OF DIFFERENT MEMBER TYPES        =',I5//,
        *10X,'NUMBER OF DIFFERENT X-SEC TYPES         =',I5//,
        *10X,'NUMBER OF DIFFERENT MATERIAL TYPES      =',I5//,
        *10X,'NUMBER OF DIFFERENT RESIDUAL TYPES      =',I5//,
        *10X,'MAXIMUM NO. OF ITERATIONS               =',I5//,
        *10X,'TOLERANCE LIMIT                         =',F12.6//,
        *10X,'LENGTH CONVERGENCE FACTOR               =',F12.6//,
        *10X,50(1H*))
C
         END
```

```
C***************************************************************
C
            SUBROUTINE INPUT2 (X,Y,H,W,TF,TW,EY,EP,EULT,YS,YPS,
     *                         ULTS,EPR,NOD,MXST,MT,MRT,NPS)
C           ****************
C
C             THIS SUBROUTINE READS THE NODAL GEOMETRY AND
C        MEMBER PROPERTIES . PROGRAM INSTAF
C
C      *****************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON/PROBCV/ UNIT,FL1,FL2,TOL,NJ,NE,NEBEL,NDXST,
     *NDMT,NDRT,IN,IO,ITLIM,IREF,IRCO,IC,NFE,NMT
      DIMENSION X(1),Y(1),H(1),W(1),TF(1),TW(1),EY(1),EP(1),
     *EULT(1),YS(1),YPS(1),ULTS(1),EPR(1,5),NOD(2,1),
     *MXST(1),MT(1),MRT(1),NPS(2,1)
C
      WRITE(IO,2000)
 50   READ (IN,1000) N,X(N),Y(N),INC
      IF(INC.EQ.0) GO TO 200
      NINT = (N-NOLD) / INC
      RN   = NINT
      IF(RN.LT.FLOAT(N-NOLD)/FLOAT(INC)-0.001) GO TO 999
      DX = (X(N)-X(NOLD))/RN
      DY = (Y(N)-Y(NOLD))/RN
      L = NOLD
      M = NINT - 1
      DO 100 J=1,M
      LL = L + INC
      X(LL) = X(L) + DX
      Y(LL) = Y(L) + DY
      L = LL
 100  CONTINUE
 200  WRITE (IO,2100) N,X(N),Y(N),INC
      NOLD = N
      IF(N.LT.NJ) GO TO 50
C
      WRITE(IO,2200)
C
      DO 225 NN=1,NDXST
      READ (IN,1100) NN,H(NN),W(NN),TF(NN),TW(NN)
 225  WRITE(IO,2300) NN,H(NN),W(NN),TF(NN),TW(NN)
C
      WRITE(IO,2400)
      DO 250 MN=1,NDMT
      READ (IN,1200) MN,EY(MN),EP(MN),EULT(MN),YS(MN),
     *YPS(MN),ULTS(MN)
 250  WRITE(IO,2500) MN,EY(MN),EP(MN),EULT(MN),YS(MN),
     *YPS(MN),ULTS(MN)
      WRITE(IO,2600)
      DO 275 NR=1,NDRT
      READ (IN,1300) NR,(EPR(NR,I),I=1,5)
```

```
275    WRITE(IO,2700) NR,(EPR(NR,I),I=1,5)
C
280    WRITE(IO,2800)
300    READ (IN,1400) M,(NOD(I,M),I=1,2),MXST(M),
      *MT(M),MRT(M),INC,NOD1,NOD2
       IF(INC.EQ.0) GO TO 500
       NINT = (M-MOLD) / INC - 1
       L = MOLD
       DO 400 I=1,NINT
       LL          =L + INC
       NOD(1,LL) = NOD(1,L) + NOD1
       NOD(2,LL) = NOD(2,L) + NOD2
       MXST(LL)  = MXST(M)
       MT(LL)    = MT(M)
       MRT(LL)   = MRT(M)
400    L        = LL
C
500    WRITE(IO,2900) M,NOD(1,M),NOD(2,M),MXST(M),
      *MT(M),MRT(M),INC,NOD1,NOD2
       MOLD = M
       IF(M.LT.NE) GO TO 300
       NPS(1,1) = 1
       NPS(2,1) = 2
       NPS(1,2) = 3
       NPS(2,2) = 1
       NPS(1,3) = 2
       NPS(2,3) = 4
       READ (IN,1500) K
       IF(K.EQ.0) GO TO 600
       WRITE(IO,3000)
       WRITE(IO,3100) (N,X(N),Y(N),N=1,NJ)
       WRITE(IO,3200)
       WRITE(IO,3300) (M,(NOD(I,M),I=1,2),MXST(M),
      *MT(M),MRT(M),M=1,NE)
C
600    RETURN
C
999    WRITE (IO,9999) N
9999   FORMAT('NODAL GEOMETRY DATA INPUT ERROR',I5)
       STOP
C
C          FORMAT STATEMENTS
C
1000   FORMAT(I5,2F12.0,I5)
1100   FORMAT(I5,4F12.0)
1200   FORMAT(I5,6F12.0)
1300   FORMAT(I5,5F12.0)
1400   FORMAT(9I5)
1500   FORMAT(I5)
2000   FORMAT (///,5X,'NODAL GEOMETRY DATA AS INPUT'/,5X,
      *28(1H*)//,10X,1HN,7X,1HX,14X,1HY,10X,3HINC//)
2100   FORMAT (6X,I5,2D15.6,I5)
2200   FORMAT(///,5X,'DIFFERENT TYPES OF X-SEC PROPERTIES'/,
      *5X,35(1H*)//,10X,'TYPE',6X,'H',14X,'W',14X,'TF',
```

```
     *13X,'TW'/)
2300  FORMAT(7X,I5,4D15.6)
2400  FORMAT(///,5X,'DIFFERENT MATERIAL PROPERTIES'/,5X,
     *29(1H*)//,10X,'TYPE',5X,'EY',13X,'EP',13X,'EULT',
     *12X,'YS',12X,'YPS',12X,'ULTS'/)
2500  FORMAT(7X,I5,6D15.6)
2600  FORMAT(///,5X,'DIFFERENT TYPES OF RESIDUAL STRAINS',
     */,5X,35(1H*)//,10X,'TYPE',5X,'EPR1',11X,'EPR2',10X,
     *'EPR3',10X,'EPR4',12X,'EPR5',/)
2700  FORMAT(7X,I5,3X,5D15.6)
2800  FORMAT(///,5X,'MEMBER DATA AS INPUT'/,5X,20(1H*)//,
     *9X,1HM,4X,1HI,4X,1HJ,2X,3HXST,3X,2HMT,3X,2HRT,3X,
     *3HINC,1X,4HINCI,1X,4HINCJ/)
2900  FORMAT(5X,9I5)
3000  FORMAT(///,5X,'COMPLETED NODAL GEOMETRY DATA'/,5X,
     *29(1H*)//,4X,1HN,7X,1HX,14X,1HY/)
3100  FORMAT(I5,2D15.6)
3200  FORMAT(///,5X,'COMPLETED MEMBER DATA'/,5X,21(1H*)//,
     *4X,1HM,4X,1HI,4X,1HJ,3X,3HXST,2X,2HMT,3X,2HRT/)
3300  FORMAT(6I5)
C
      END
```

```
C*****************************************************************
C
                SUBROUTINE EQSBST (A,B,MAXA,NB,IL)
C               ****************
C
C          THIS SUBROUTINE TRIANGULARIZES A STIFFNESS MATRIX
C       STORED COLUMNWISE UNDER A SKYLINE AND REDUCES THE
C       CORRESPONDING LOAD VECTOR, DOWN TO EQUATION NB.
C
C       ***********************************************************
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION A(1),B(1),MAXA(1)
C
C
        DO  1000 N=1,NB
        KN = MAXA(N)
        KL = KN + 1
        KU = MAXA(N+1) - 1
        KH = KU - KL
        IF(KH) 900,500,100
 100    K  = N - KH
        IC = 0
        KLT= KU
        DO 400 J=1,KH
        IC = IC + 1
        KLT= KLT - 1
        KI = MAXA(K)
        ND = MAXA(K+1) - KI - 1
        IF(ND) 400,400,200
 200    KK = IC
        IF(KK.GT.ND) KK = ND
        C  = 0.0
        DO 300 L = 1,KK
 300    C  = C + A(KI+L)*A(KLT+L)
        A(KLT) = A(KLT) - C
 400    K  = K + 1
 500    K  = N
C
        C  = 0.0
        DO 600 KK = KL,KU
        K  = K - 1
        KI = MAXA(K)
        D  = A(KK)/A(KI)
        C  = C + D*A(KK)
        A(KK) = D
 600    CONTINUE
        A(KN) = A(KN) - C
C
        K  = N
        C = 0.0
        DO 700 KK = KL,KU
        K  = K - 1
        C  = C + A(KK)*B(K)
```

```
700   CONTINUE
      B(N) = B(N) - C
C
900   IF(A(KN)) 975,950,1000
950   WRITE(6,3000) N,A(KN)
      STOP
975   IL = 999
C
1000  CONTINUE
C
      RETURN
C
3000  FORMAT('ZERO ELEMENT ON MAIN DIAGONAL NO.',I4,
     *D15.6)
C
      END
```

```
C*******************************************************
C
                 SUBROUTINE BKSB1 (B,A,MAXA,NN,KB)
C                ***************
C
C           THIS SUBROUTINE IS A PART OF THE PARTIAL REDUCTION
C        PACKAGE. IT FORMS THE LAST STAGE OF BACKSUBSTITUTION,
C        THE OPERATION (L(-1)(T)*RI*) .   IN CASE OF AN
C        UNSUBSTRUCTURED PROBLEM IT PERFORMS THE OPERATION
C        //D(-1)*L(-1)(T)*R//
C
C        *************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION A(1),B(1),MAXA(1)
C
      IF(KB.EQ.1) GO TO 200
      DO 100 J=1,NN
      K = MAXA(J)
100   B(J) = B(J)/A(K)
C
200   N  = NN
      DO 600 L=2,NN
      KL = MAXA(N) + 1
      KU = MAXA(N+1) - 1
      IF(KU-KL) 600,300,300
300   IF(B(N).EQ.0.) GO TO 600
      K  = N
      DO 400 KK=KL,KU
      K = K - 1
400   B(K) = B(K) - A(KK)*B(N)
600   N  = N - 1
C
      RETURN
C
      END
```

```
C*****************************************************
C
              SUBROUTINE EQFT (A,MAXA,NP,ID,NN)
C             ***************
C
C         THIS ROUTINE IS A PART OF THE PARTIAL REDUCTION
C       PACKAGE. IT FORMS F(T) FOR AN UPPER TRIANGULAR
C       MATRIX STORED COLUMNWISE UNDER A SKYLINE.
C
C       ************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION A(1),MAXA(1)
C
        NS = NP*ID + 1
        NB = NN*ID
        NI = NP*ID
C
        DO 700 N=NS,NB
        KN =MAXA(N)
        KL = KN + N - NI
        KU = MAXA(N+1) - 1
        KH = KU - KL
        IF (KH) 700,500,100
100     K  = N - (KU-KN) + 1
        IC = 0
        KLT = KU
        DO 400 J=1,KH
        IC  = IC + 1
        KLT = KLT - 1
        KI  = MAXA(K)
        ND  = MAXA(K+1) - KI - 1
        IF (ND) 400,400,200
200     KK = IC
        IF(KK.GT.ND) KK=ND
        C  = 0.0
        DO 300 L=1,KK
300     C  = C + A(KI+L)*A(KLT+L)
        A(KLT) = A(KLT) - C
400     K  = K + 1
500     K  = NS
C
        DO 600 KK=KL,KU
        K  = K - 1
        KI = MAXA(K)
        A(KK) = A(KK) / A(KI)
600     CONTINUE
700     CONTINUE
C
        RETURN
C
        END
```

```
C*****************************************************
C
              SUBROUTINE EQKBB (A,B,MAXA,NP,ID,NN)
C             ***************
C
C      THIS SUBROUTINE IS A PART OF THE PARTIAL REDUCTION
C    PACKAGE. IT FORMS THIS PARTITION KBB* , AND RBB* (THE
C    INTER-BOUNDARY SUBARRAYS).
C
C      ***************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION A(1),MAXA(1),B(1)
C
      NS = NP*ID + 1
      NB = NN*ID
      NI = NP*ID
C
      DO 600 N=NS,NB
      KN  = MAXA(N)
      KU1 = KN + N - NS
      KU2 = MAXA(N+1) - 1
      IF(KU1.GE.KU2) GO TO 600
      KH1 = KU2 - KU1
      KH2 = KU1 - KN + 1
      K   = N - KH2 + 1
      KLT = KU1 + 1
C
      DO 300 J=1,KH2
      KLT = KLT - 1
      KI  = MAXA(K)
      KI1 = KI + K - NS
      KI2 = MAXA(K+1) - 1
      KHI = KI2 - KI1
      IF (KHI) 300,300,100
100   KK  = KHI
      IF(KK.GT.KH1) KK = KH1
      C  = 0.0
      L1 = KI + J + KK
      L2 = KLT + J + KK
      L3 = NI - KK
      DO 200 L=1,KK
200   C  = C + A(L1-L)*A(L2-L)*A(MAXA(L3+L))
      A(KLT) = A(KLT) - C
300   K  = K + 1
C
      L4 = NI - KH1
      L5 = KU2 + 1
      C = 0.0
      DO 400 L=1,KH1
400   C  = C + A(L5-L)*B(L4+L)
      B(N) = B(N) - C
600   CONTINUE
```

```
C
      RETURN
C
      END
```

```
C*************************************************************
C
                 SUBROUTINE BKSB2 (A,BA,MAXA,NP,NB,ID)
C                ****************
C
C           THIS SUBROUTINE IS PART OF THE PARTIAL REDUCTION
C       PACKAGE . IT FORMS A PART OF THE BACKSUBSTITUTION
C       SCHEME NAMELY THE QUANTITTY //D(-1)*RI* - F(T)*RBB //
C
C       *******************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION A(1),BA(1),MAXA(1)
C
        NS = NP * ID + 1
        NN = NB * ID
        NI = NP * ID
C
        DO 100 N=1,NI
        KN = MAXA(N)
        BA(N) = BA(N) / A(KN)
100     CONTINUE
C
        DO 400 N=NS,NN
        KN  = MAXA(N)
        KU1 = KN + N - NI
        M1  = MAXA(N+1)
        KU2 = M1 - 1
        IF(KU1.GT.KU2) GO TO 400
        KH  = KU2 - KU1 + 1
        K   = N - KU2 + KN
C
        DO 300 J=1,KH
        AR = A(M1 - J)
        IF(AR.EQ.0.) GO TO 300
        BA(K) = BA(K) - BA(N)*AR
300     K = K + 1
C
400     CONTINUE
C
        RETURN
C
        END
```

```
C*****************************************************
C
            SUBROUTINE STIFF (X,Y,H,W,TF,TW,EY,EP,EULT,YS,YPS,
     *                 ULTS,EPR,NOD,MXST,MT,MRT,MAXA,PHI,PHIP,
     *                 PHIDP,TK,B,DISP,IDO,NGAUS,NE,NJ,UNIT,IO,
     *                      MAXS,TKS,NPS,SD,NWK,IREF,IT,ILL,IC)
C
C           THIS SUBROUTINE FORMS THE TANGENT STIFFNESS
C        MATRIX OF 10 D.O.F. PLANE FRAME MEMBER.
C
C       ************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION X(1),Y(1),H(1),W(1),TF(1),TW(1),EY(1),
     *EP(1),EULT(1),YS(1),YPS(1),ULTS(1),EPR(1,5),
     *NOD(2,1),MXST(1),MT(1),MRT(1),MAXA(1),BAT(16),
     *PHI(4,1),PHIP(4,1),PHIDP(4,1),TK(1),B(1),DISP(1),
     *EPS(5),MAXS(1),NPS(2,1),TKS(1),BS(16),LMS(8),SD(1),
     *INFO(4),D(10),DM(8),PHIL(4),PHIPL(4),PHIDPL(4),
     *PHITP(4),DIP(8),DIM(8),DIMS(8),PHIP1(8),PHIP2(8),
     *PHID1(8),BB(8),S(8,8),WGAUS(4),T(8,10),TEMP(8,10),
     *SS(10,10),LM(10),A(55),BA(16)
C
      DATA WGAUS/.173927425,.326072575,.326072575,
     *.173927425/
C
C     ************************************************
C
      NI   = 2
      NB   = 8
      ID   = 4
      NN   = 4
      ND   = 10
      NREC = 0
C
      WRITE(IO,2200)
      DO 900 M=1,NE
      KXS = MXST(M)
      KMT = MT(M)
      KRT = MRT(M)
      YSM = YS(KMT)
      YPSM = YPS(KMT)
      ULTSM = ULTS(KMT)
      EYM   = EY(KMT)
      EPM   = EP(KMT)
      EULTM = EULT(KMT)
      HH    = H(KXS)
      WW    = W(KXS)
      TFF   = TF(KXS)
      TWW   = TW(KXS)
C
      YMOD  = YSM / EYM
      YPL   = (YPSM - YSM) / (EPM - EYM)
```

```
      YMOST = (ULTSM - YPSM) / (EULTM - EPM)
      I   = NOD(1,M)
      J   = NOD(2,M)
      III = I
      JJJ = J
      DX = (X(J) - X(I)) * UNIT
      DY = (Y(J) - Y(I)) * UNIT
      XL =DSQRT (DX**2 + DY**2)
      XLI= 1  / XL
      CO = DX * XLI
      SI = DY * XLI
      C2 = CO ** 2
      S2 = SI ** 2
      SC = CO * SI
      DO 50 KK=1,IDO
      LM(KK)      = IDO * I - IDO + KK
50    LM(KK+IDO) = IDO * J - IDO + KK
      II = (I-1) * 5
      JJ = (J-1) * 5
C
      CALL CLEAR (BAT,16)
C
      DO 75 I=1,16
75    BS(I) = 0.0
C
      XL1  = 0.8D0 * XL
      XLI1 = 1  / XL1
C
      DO 100 I=1,5
      J  = II + I
      J1 = JJ + I
      D(I)   = SD(J)
      D(I+5) = SD(J1)
100   CONTINUE
      CALL CLEAR (BA,16)
      IF(IREF.EQ.0.AND.IT.EQ.0) GO TO 130
      CALL TR2 (D,CO,SI,SC,C2,S2,DM)
C
      NREC = NREC + 1000
      READ(4'NREC) (BAT(LL),LL=1,16)
      NREC = NREC + 1000
      READ(4'NREC)(TKS(LL),LL=1,NWK)
      NREC = NREC + 1000
      READ(4'NREC) (BS(LL),LL=1,16)
      NREC = NREC - 3000
      DO 125 I=1,8
      BA(I)   = BS(I)
      BA(I+8) = DM(I)
125   CONTINUE
      IF(ILL.EQ.999.AND.IC.EQ.1) CALL CLEAR (BA,16)
      CALL BKSB2 (TKS,BA,MAXS,NI,NN,ID)
      CALL BKSB1 (BA,TKS,MAXS,(NI*ID),1)
      CALL CLEAR (TKS,NWK)
      CALL CLEAR (BS,16)
```

```
      DO 128 I=1,16
      BAT(I) = BAT(I) + BA(I)
128   CONTINUE
130   NREC = NREC + 1000
      WRITE(4'NREC) (BAT(LL),LL=1,16)
C
      DO 625 IJK=1,3
      IJ = NPS(1,IJK)
      JI = NPS(2,IJK)
      DO 140 I = 1,2
      J  = (IJ-1) * 4 + I
      J1 = (JI-1) * 4 + I
      DM(I)   = BAT(J)
      DM(I+2) = BAT(J1)
      DM(I+4) = BAT(J+2)
      DM(I+6) = BAT(J1+2)
140   CONTINUE
C
      DO 150 J=1,8
      DO 150 I=1,8
      S(I,J) = 0.0D0
150   CONTINUE
C
      DO 175 I=1,8
175   BB(I) = 0.0D0
C
      PHITP(1) = 12 * XLI1**3
      PHITP(2) = 6 * XLI1 * XLI1
      PHITP(3) = -12 * XLI1**3
      PHITP(4) = 6 * XLI1 * XLI1
      VTP = 0.0D0
      DO 180 I=1,4
180   VTP = VTP + PHITP(I) * DM(I+4)
      DO 600 IG=1,NGAUS
      WG  =  XL1 * WGAUS(IG)
      DO 200 I=1,4
      IF(I.EQ.1.OR.I.EQ.3) C1 = XLI1
      PHIL(I)   = PHI(I,IG)  * 1./(XLI1-C1+1.)
      PHIPL(I)  = PHIP(I,IG) * C1
      PHIDPL(I) = PHIDP(I,IG) * XLI1 * C1
      C1 = 1.0
200   CONTINUE
      UP  = 0.0D0
      UDP = 0.0D0
      VP  = 0.0D0
      VDP = 0.0D0
C   AT GAUSS POINT EVALUATE U',V',V"
      DO 300 I=1,4
      UP   = UP  + PHIPL(I) * DM(I)
      UDP  = UDP + PHIDPL(I)* DM(I)
      VP   = VP  + PHIPL(I) * DM(I+4)
      VDP  = VDP + PHIDPL(I)* DM(I+4)
300   CONTINUE
C
```

```
      UP2  = UP * UP
      VP2  = VP * VP
      VDP2 = VDP * VDP
      VPI  = 1 / (1-VP2)
      IF(VPI) 310,320,320
 310  RVPI = -DSQRT(-VPI)
      GO TO 330
 320  RVPI = DSQRT(VPI)
 330  VPVD = VP * VDP
      A1   = 1 + UP
      A2   = UP + 0.5D0 *(UP2+VP2)
      A3   = 1 + VP2 * VPI
      A4   = VDP * A3
      A5   = A3 * VDP2
      A6   = A1 + VP2*RVPI
      A7   = VDP * A6
      A8   = A3 + 1
      A9   = VP * RVPI * A8
      A10  = VPVD * VPI * A3
      A11  = 2 * A4 * RVPI
      A12  = A9 * VDP
      A13  = 3 * A10 * RVPI * VP
      A14  = A5 * VPI
      A15  = 4 * A10 * VPVD * VPI
C
      Y1   = 0.5D0 * (HH - TFF)
      Y12  = Y1 * Y1
      A21  = Y1 * A7
      A22  = 0.5D0 * Y12 * A5
C
      EPS(1) = A2 + A21 + A22
      EPS(2) = EPS(1)
      EPS(3) = A2
      EPS(4) = A2 - A21 + A22
      EPS(5) = EPS(4)
C
      DO 350 I=1,5
      EPS(I) = EPS(I) + EPR(KRT,I)
 350  CONTINUE
C
      CALL STEP (EPS,HH,WW,TFF,TWW,YMOD,YPL,YMOST,EYM,EPM,
     *YSM,YPSM,PP,OM,SM,TAREA,R1,R2,R3,R4)
C
      YA  = YMOD * TAREA
      YR1 = YMOD * R1
      YR  = YMOD * R2
      YR3 = YMOD * R3
      YR4 = YMOD * R4
      IF(IJK.EQ.2.AND.IG.EQ.1) GO TO 360
      GO TO 375
 360  P1  = PP
      OM1 = OM
      SM1 = SM
C
```

```
375   FE1 = PP * A1 - OM * VDP
      FE2 = PP * VP + VDP * (-OM*A9 + SM*A10)
      FE3 = -OM * A6 + SM * A4
C
      DO 400 I=1,4
      BB(I)     = BB(I)   + WG * FE1 * PHIPL(I)
      BB(I+4)   = BB(I+4) + WG * ( FE2 * PHIPL(I)
     *                            + FE3 * PHIDPL(I) )
C
      DIP(I)    = (YA*A1 - YR1*VDP) * PHIPL(I)
      DIP(I+4)  = (YA*VP - YR1*A12 + YR*A10*VDP) * PHIPL(I)
     *                  + ( YR*A4 - YR1*A6) * PHIDPL(I)
      DIM(I)    = (YR1*A1 - YR*VDP) * PHIPL(I)
      DIM(I+4)  = (YR1*VP - YR*A12 + YR3*A10*VDP) * PHIPL(I)
     *                    + (YR3*A4 - YR*A6) * PHIDPL(I)
      DIMS(I)   = (YR*A1 - YR3*VDP) * PHIPL(I)
      DIMS(I+4) = (YR*VP - YR3*A12 + YR4*A10*VDP) * PHIPL(I)
     *                       + (YR4*A4 - YR3*A6) * PHIDPL(I)
      PHIP1(I)   = PHIPL(I)
      PHIP1(I+4) = 0.0D0
      PHIP2(I)   = 0.0D0
      PHIP2(I+4) = PHIPL(I)
      PHID1(I)   = 0.0D0
      PHID1(I+4) = PHIDPL(I)
400   CONTINUE
C
C   FORM ELEMENT STIFFNESS MATRIX
C
      DO 500 I=1,4
      DO 500 J=1,8
      S(I,J)    = S(I,J)   +WG * PHIPL(I) *
     *                     ( A1 * DIP(J) - VDP * DIM(J)
     *                     + PP * PHIP1(J) - OM * PHID1(J))
      S(I+4,J) = S(I+4,J)
     *           + WG * (PHIPL(I) *
     *           ( VP * DIP(J) - VDP*(A9*DIM(J) - A10*DIMS(J))
     *           + PP*PHIP2(J) - OM*(A9*PHID1(J)
     *                            +(A11+A13)*PHIP2(J) )
     *           + SM * (2*A10*PHID1(J) + (A14+A15)*PHIP2(J)))
     *           + PHIDPL(I) *
     *           ( -A6 * DIM(J) + A4 * DIMS(J)
     *           -OM * ( PHIP1(J) + A9 * PHIP2(J))
     *           + SM * (2 * A10 * PHIP2(J) + A3 * PHID1(J))))
500   CONTINUE
600   CONTINUE
C
      XL1  = 0.1D0 * XL
      XLI1 = 1 / XL1
C
      DO 610 I=1,2
      J  = (IJ - 1) * 4 + I
      J1 = (JI - 1) * 4 + I
      BS(J)     = BS(J)   - BB(I)
      BS(J+2)   = BS(J+2) - BB(I+4)
```

```
      BS(J1)   = BS(J1) - BB(I+2)
      BS(J1+2) = BS(J1+2) - BB(I+6)
610   CONTINUE
C
      K = 1
      DO 620 I=1,8
      DO 620 J=I,8
      A(K) = S(I,J)
620   K = K + 1
      DO 622 KK=1,2
      LMS(KK)   = 4 * IJ - 4 + KK
      LMS(KK+2) = 4 * JI - 4 + KK
      LMS(KK+4) = 4 * IJ - 4 + KK+2
622   LMS(KK+6) = 4 * JI - 4 + KK+2
      CALL ASSEMB (MAXS,A,TKS,LMS,NB)
625   CONTINUE
      WRITE(IO,2300) M,III,JJJ,P1,OM1,SM1,PP,OM,SM
C
      ILM = 1
      CALL EQSBST (TKS,BS,MAXS,NB,ILM)
      CALL EQFT (TKS,MAXS,NI,ID,NN)
      CALL EQKBB (TKS,BS,MAXS,NI,ID,NN)
      NREC = NREC + 1000
      WRITE(4'NREC) (TKS(LL),LL=1,NWK)
      NREC = NREC + 1000
      WRITE(4'NREC) (BS(LL),LL=1,16)
      DO 630 I=1,8
      BB(I) = BS(I+8)
630   CONTINUE
C
      CALL TR1 (BB,D,CO,SI,SC,C2,S2)
      DO 650 I=1,5
      J  = II + I
      J1 = JJ + I
      B(J)  = B(J)  + D(I)
      B(J1) = B(J1) + D(I+5)
650   CONTINUE
      DO 700 J=1,10
      DO 700 I=1,8
700   T(I,J)  = 0.0D0
      T(1,1)  = CO
      T(1,2)  = SI
      T(2,4)  = C2
      T(2,5)  = S2
      T(3,1)  = SI
      T(3,2) = -CO
      T(4,3) = -1.0D0
      T(4,4) = SC
      T(4,5) = -SC
      T(5,6) = CO
      T(5,7) = SI
      T(6,9) = C2
      T(6,10) = S2
      T(7,6)  = SI
```

```
      T(7,7)   = -CO
      T(8,8)   = -1.0D0
      T(8,9)   = SC
      T(8,10)  = -SC
C
C    FORM ELEMENT STIFFNESS MATRIX IN GLABOL COORDINATES
C
      NR = 8
      NC = 10
      CALL MULT2 (TEMP,TKS,T,MAXS,NR,NC)
      DO 750 L=1,10
      DO 750 K=1,10
      D1 = 0.0D0
      DO 740 N=1,8
740   D1 = D1 + T(N,L) * TEMP(N,K)
      SS(L,K) = D1
750   SS(K,L) = D1
      K = 1
      DO 760 I=1,10
      DO 760 J=I,10
      A(K) = SS(I,J)
      K = K + 1
760   CONTINUE
C
      CALL ASSEMB (MAXA,A,TK,LM,ND)
C
      CALL CLEAR (TKS,NWK)
900   CONTINUE
C
      RETURN
C
C      FORMAT STATEMENTS
C
2200  FORMAT (///,'STRESS RESULTANTS'//,5X,'MEM',2X,
     *'NODI',2X,'NODJ',10X,'PI',13X,'MI',12X,'MI*',12X,
     *'PJ',13X,'MJ',12X,'MJ*'/)
2300  FORMAT (I7,2X,I5,2X,I5,2X,6D15.6)
C
      END
```

```
C*****************************************************
C
                SUBROUTINE TR1 (BB,DD,CO,SI,SC,C2,S2)
C               **************
C
C           THIS SUBROUTINE TRANSFORMS THE NODAL FORCES
C       FROM LOCAL TO GLABOL COORDINATES.
C
C       *********************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION BB(8),DD(10)
C
        DD(1)  = BB(1) * CO + BB(3) * SI
        DD(2)  = BB(1) * SI - BB(3) * CO
        DD(3)  = -BB(4)
        DD(4)  = BB(2) * C2 + BB(4) * SC
        DD(5)  = BB(2) * S2 - BB(4) * SC
        DD(6)  = BB(5) * CO + BB(7) * SI
        DD(7)  = BB(5) * SI - BB(7) * CO
        DD(8)  = -BB(8)
        DD(9)  = BB(6) * C2 + BB(8) * SC
        DD(10) = BB(6) * S2 - BB(8) * SC
C
        RETURN
C
        END
```

```
C*****************************************************
C
              SUBROUTINE TR2 (D,CO,SI,SC,C2,S2,DM)
C             *************
C
C           THIS SUBROUTINE TRANSFORMS THE NODAL DISPLACEMENTS
C       FROM SYSTEM COORDINATES TO LOCAL ONS.
C
C       *************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION D(10),DM(8)
C
        DM(1) = D(1) * CO + D(2) * SI
        DM(2) = D(4) * C2 + D(5) * S2
        DM(5) = D(6) * CO + D(7) * SI
        DM(6) = D(9) * C2 + D(10)* S2
        DM(3) = D(1) * SI - D(2) * CO
        DM(4) = -D(3) + D(4) * SC - D(5) * SC
        DM(7) = D(6) * SI - D(7) * CO
        DM(8) = -D(8) + D(9) * SC - D(10) * SC
C
        RETURN
C
        END
```

```
C*********************************************************************
C
               SUBROUTINE STIFFB (X,Y,H,W,TF,TW,EY,EP,EULT,YS,
     *                           YPS,ULTS,EPR,NOD,MXST,MT,MRT,
     *                     MAXA,TK,B,DISP,IDO,NFE,NE,UNIT,IO)
C
C
C           THIS ROUTINE FORMS THE TANGENT STIFFNESS MATRIX FOR
C      A MEMBER SUBJECTED TO AXIAL FORCE ONLY.
C
C     ****************************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION X(1),Y(1),H(1),W(1),TF(1),TW(1),EY(1),EP(1),
     *EULT(1),YS(1),YPS(1),ULTS(1),EPR(1,5),NOD(2,1),MXST(1)
     *,MT(1),MRT(1),MAXA(1),TK(1),B(1),DISP(1),D(4),DM(4),
     *A(10),S(4,4),BB(4),DIP(4),PHIP(2),PHIP1(4),PHIP2(4),
     *T(4,4),TEMP(4,4),SS(4,4),LM(4),EPS(5)
C
CC
C
      NNE = NFE + 1
      NR  = 4
      WRITE(IO,2000)
C
      DO 900 M=NNE,NE
      KXS = MXST(M)
      KMT = MT(M)
      KRT = MRT(M)
      YSM = YS(KMT)
      YPSM = YPS(KMT)
      ULTSM = ULTS(KMT)
      EYM   = EY(KMT)
      EPM   = EP(KMT)
      EULTM = EULT(KMT)
      HH    = H(KXS)
      WW    = W(KXS)
      TFF   = TF(KXS)
      TWW   = TW(KXS)
C
      YMOD  = YSM / EYM
      YPL   = (YPSM - YSM) / (EPM - EYM)
      YMOST = (ULTSM - YPSM) / (EULTM - EPM)
      I  = NOD(1,M)
      J  = NOD(2,M)
      III = I
      JJJ = J
      DX = (X(J) - X(I)) * UNIT
      DY = (Y(J) - Y(I)) * UNIT
      XL =DSQRT (DX**2 + DY**2)
      XLI= 1  / XL
      CO = DX * XLI
      SI = DY * XLI
```

```
      C2 = CO ** 2
      S2 = SI ** 2
      SC = CO * SI
      DO 50 KK=1,2
      LM(KK)   = IDO * I - IDO + KK
50    LM(KK+2) = IDO * J - IDO + KK
      II = (I-1) * 5
      JJ = (J-1) * 5
      DO 100 I=1,2
      J  = II + I
      J1 = JJ + I
      D(I)   = DISP(J)
      D(I+2) = DISP(J1)
100   CONTINUE
C
      DM(1) = D(1) * CO + D(2) * SI
      DM(2) = D(3) * CO + D(4) * SI
      DM(3) = D(1) * SI - D(2) * CO
      DM(4) = D(3) * SI - D(4) * CO
C
      DO 200 J=1,4
      DO 200 I=1,4
      S(I,J) = 0.0D0
200   CONTINUE
C
C
      PHIP(1) = -XLI
      PHIP(2) =  XLI
      UP  = PHIP(1) * DM(1) + PHIP(2) * DM(2)
      VP  = PHIP(1) * DM(3) + PHIP(2) * DM(4)
      UP2 = UP * UP
      VP2 = VP * VP
      A1  = 1 + UP
      A2  = UP + 0.5D0 * (UP2 + VP2)
C
      DO 275 I=1,5
      EPS(I) = A2 + EPR(KRT,I)
275   CONTINUE
C
      CALL STEP1 (EPS,HH,WW,TFF,TWW,YMOD,YPL,YMOST,
     *EYM,EPM,YSM,YPSM,PP,TAREA)
C
      YA  = YMOD * TAREA
      WRITE (IO,2100) M,PP
      FE1 = XL * PP * A1
      FE2 = XL * PP * VP
      YAA = YA * A1
      YAV = YA * VP
C
      DO 300 I=1,2
      BB(I)     = FE1 * PHIP(I)
      BB(I+2)   = FE2 * PHIP(I)
      DIP(I)    = YAA * PHIP(I)
      DIP(I+2)  = YAV * PHIP(I)
```

```
      PHIP1(I)   = PHIP(I)
      PHIP1(I+2)= 0.0D0
      PHIP2(I)   = 0.0D0
      PHIP2(I+2)= PHIP(I)
300   CONTINUE
C
C  FORM ELEMENT STIFFNESS MATRIX
C
      DO 400 I=1,2
      DO 400 J=1,4
      S(I,J)    = S(I,J)    + XL * PHIP(I) * (A1*DIP(J)
     *                                        + PP*PHIP1(J) )
      S(I+2,J) = S(I+2,J)  + XL * PHIP(I) * (VP*DIP(J)
     *                                        + PP*PHIP2(J) )
400   CONTINUE
C
      D(1) = BB(1) * CO + BB(3) * SI
      D(2) = BB(1) * SI - BB(3) * CO
      D(3) = BB(2) * CO + BB(4) * SI
      D(4) = BB(2) * SI - BB(4) * CO
C
      DO 500 I=1,2
      J  = II + I
      J1 = JJ + I
      B(J)  = B(J) - D(I)
      B(J1) = B(J1) - D(I+2)
500   CONTINUE
C
      DO 550 J=1,4
      DO 550 I=1,4
550   T(I,J) = 0.0D0
C
      T(1,1) = CO
      T(1,2) = SI
      T(2,3) = CO
      T(2,4) = SI
      T(3,1) = SI
      T(3,2) = -CO
      T(4,3) = SI
      T(4,4) = -CO
C
C   FORM ELEMENT STIFFNESS MATRIX IN GLOBAL COORDINATES
C
      DO 650 L=1,4
      DO 650 K=1,4
      D1 = 0.0D0
      DO 600 N=1,4
600   D1 = D1 + S(K,N) * T(N,L)
650   TEMP(K,L) = D1
C
      DO 750 L=1,4
      DO 750 K=1,4
      D1 = 0.0D0
      DO 700 N=1,4
```

```
700    D1 = D1 + T(N,L) * TEMP(N,K)
       SS(L,K) = D1
750    SS(K,L) = D1
       K = 1
       DO 800 I=1,4
       DO 800 J=I,4
       A(K) = SS(I,J)
       K = K + 1
800    CONTINUE
       CALL ASSEMB (MAXA,A,TK,LM,NR)
C
900    CONTINUE
C
       RETURN
C
C    FORMAT STATEMENTS
C
2000   FORMAT (5X,'BRACING MEMBER NO.',5X,'AXIAL FORCE')
2100   FORMAT (10X,I5,15X,D15.6)
C
       END
```

```
C*********************************************************
C
            SUBROUTINE STRAIN (EPS,RE,TR,ST,I,J,DDXX,TT,
     *                    YMOD,YPL,YMOST,EPM,EYM,YSM,YPSM)
C
C        THIS SUBROUTINE DETERMINES THE TRANSFORMED SECTION
C       AND THE STRESSES IN A REGION. REGIONS IN A PLATE
C       SEGMENT ARE DETERMINED FOR ANY ONE OF THE VARIOUS
C       STRAIN DISTRIBUTION (18 TYPES FOR TRILINEAR STRESS
C       - STRAIN CURVE.
C
C       ****************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION EPS(5),RE(5),TR(5),ST(6)
C
        UMC = 1.0
        P   = EPS(I)
        Q   = EPS(J)
        IF(P.LT.0.0) UMC = -1.0
        IF(Q.EQ.0.0) GO TO 25
        DIV = P / Q
        GO TO 50
25      DIV = DABS(P)
50      PP  = TT * YMOST / YMOD
        QQ  = TT * YPL   / YMOD
        DO 100 N =1,5
        RE(N) = 0.0
        TR(N) = 0.0
100     CONTINUE
        TR(2) = QQ
        TR(4) = QQ
        K = 1
        IF(DIV.LT.0.0) GO TO 200
        L = 0
        GO TO 225
200     L = 9
225     P   = DABS(P)
        Q   = DABS(Q)
        PMQ = P - Q
        PPQ = P + Q
        PEP = P - EPM
        PEY = P - EYM
        QEY = Q - EYM
        QEP = Q - EPM
C
        ST(1) = ( YMOST * PEP + YPSM ) * UMC
        ST(2) = YPSM * UMC
        ST(3) = YSM  * UMC
        IF(DIV.LT.0.0) UMC = - UMC
        ST(4) = YSM  * UMC
        ST(5) = YPSM * UMC
        ST(6) = ( YMOST * QEP + YPSM ) * UMC
```

```
C
      IF(P.EQ.Q) GO TO 250
      C1 = DDXX * PEP / PMQ
      C2 = DDXX * PEY / PMQ
      GO TO 275
250   C1 = 0.0
      C2 = 0.0
275   IF(PPQ.EQ.0.0) GO TO 300
      D1 = DDXX * PEP / PPQ
      D2 = DDXX * PEY / PPQ
      D4 = DDXX * QEY / PPQ
      D5 = DDXX * QEP / PPQ
C
300   IF(P.GE.EPM.AND.Q.GE.EPM)                 KK = K + L
      IF(P.GE.EPM.AND.Q.GE.EYM.AND.Q.LE.EPM)    KK = 2*K + L
      IF(P.GE.EPM.AND.Q.GE.0.0.AND.Q.LE.EYM)    KK = 3*K + L
      IF(P.GE.EYM.AND.P.LE.EPM.AND.Q.GE.EPM)    KK = 4*K + L
      IF(P.GE.EYM.AND.P.LE.EPM.AND.Q.GE.EYM.
     *AND.Q.LE.EPM)                             KK = 5*K + L
      IF(P.GE.EYM.AND.P.LE.EPM.AND.Q.GE.0.0.
     *AND.Q.LE.EYM)                             KK = 6*K + L
      IF(P.GE.0.0.AND.P.LE.EYM.AND.Q.GE.EPM)    KK = 7*K + L
      IF(P.GE.0.0.AND.P.LE.EYM.AND.Q.GE.EYM.
     *AND.Q.LE.EPM)                             KK = 8*K + L
      IF(P.GE.0.0.AND.P.LE.EYM.AND.Q.GE.0.0.
     *AND.Q.LE.EYM)                             KK = 9*K + L
      GO TO (301,302,303,304,305,306,307,308,309,310,311,
     *312,313,314,315,316,317,318) ,KK
301   RE(1) = DDXX
      TR(1) = PP
      ST(2) = ( YMOST * QEP + YPSM ) * UMC
      ST(3) = 0.0
      ST(4) = 0.0
      ST(5) = 0.0
      ST(6) = 0.0
      GO TO 400
302   RE(1) = C1
      TR(1) = PP
      RE(2) = DDXX - RE(1)
      ST(3) = (YPL * QEY + YSM ) * UMC
      ST(4) = 0.0
      ST(5) = 0.0
      ST(6) = 0.0
      GO TO 400
303   RE(1) = C1
      TR(1) = PP
      RE(2) = C2 - C1
      RE(3) = DDXX - RE(1) - RE(2)
      TR(3) = TT
      ST(4) = YMOD * Q * UMC
      ST(5) = 0.0
      ST(6) = 0.
      GO TO 400
304   RE(4) = C1
```

```
      RE(5) = DDXX - RE(4)
      TR(5) = PP
      ST(4) = ( YPL * PEY + YSM ) * UMC
      ST(1) = 0.0
      ST(2) = 0.0
      ST(3) = 0.0
      GO TO 400
305   RE(4) = DDXX
      ST(1) = 0.0
      ST(2) = 0.0
      ST(3) = 0.0
      ST(4) = (YPL * PEY + YSM) * UMC
      ST(5) = (YPL * QEY + YSM) * UMC
      ST(6) = 0.0
      GO TO 400
306   RE(2) = C2
      RE(3) = DDXX - C2
      TR(3) = TT
      ST(1) = 0.0
      ST(2) = (YPL * PEY + YSM) * UMC
      ST(4) = YMOD * Q * UMC
      ST(5) = 0.0
      ST(6) = 0.0
      GO TO 400
307   RE(3) = C2
      TR(3) = TT
      RE(4) = C1 - RE(3)
      RE(5) = DDXX - RE(3) - RE(4)
      TR(5) = PP
      ST(1) = 0.0
      ST(2) = 0.0
      ST(3) = YMOD * P * UMC
      GO TO 400
308   RE(3) = C2
      TR(3) = TT
      RE(4) = DDXX - RE(3)
      ST(1) = 0.0
      ST(2) = 0.0
      ST(3) = YMOD * P * UMC
      ST(5) = (YPL * QEY + YSM) * UMC
      ST(6) = 0.0
      GO TO 400
309   RE(3) = DDXX
      TR(3) = TT
      ST(1) = 0.0
      ST(2) = 0.0
      ST(3) = YMOD * P * UMC
      ST(4) = YMOD * Q * UMC
      ST(5) = 0.0
      ST(6) = 0.0
      GO TO 400
310   RE(1) = D1
      TR(1) = PP
      RE(2) = D2 - RE(1)
```

```
      RE(5) = D5
      TR(5) = PP
      RE(4) = D4 - RE(5)
      RE(3) = DDXX - RE(1) - RE(2) - RE(4) - RE(5)
      TR(3) = TT
      GO TO 400
 311  RE(1) = D1
      TR(1) = PP
      RE(2) = D2 - RE(1)
      RE(4) = D4
      RE(3) = DDXX - RE(1) - RE(2) - RE(4)
      TR(3) = TT
      ST(5) = (YPL * QEY + YSM) * UMC
      ST(6) = 0.0
      GO TO 400
 312  RE(1) = D1
      TR(1) = PP
      RE(2) = D2 - RE(1)
      RE(3) = DDXX - RE(1) - RE(2)
      TR(3) = TT
      ST(4) = YMOD * Q * UMC
      ST(5) = 0.0
      ST(6) = 0. 0
      GO TO 400
 313  RE(2) = D2
      RE(5) = D5
      TR(5) = PP
      RE(4) = D4 - RE(5)
      RE(3) = DDXX - RE(2) - RE(5) - RE(4)
      TR(3) = TT
      ST(1) = 0.0
      ST(2) = -(YPL * PEY + YSM) * UMC
      GO TO 400
 314  RE(2) = D2
      RE(4) = D4
      RE(3) = DDXX - RE(2) - RE(4)
      TR(3) = TT
      ST(1) = 0.0
      ST(2) = -(YPL * PEY + YSM) * UMC
      ST(5) =   (YPL * QEY + YSM) * UMC
      ST(6) = 0.0
      GO TO 400
 315  RE(2) = D2
      RE(3) = DDXX - RE(2)
      TR(3) = TT
      ST(1) = 0.0
      ST(2) = -(YPL * PEY + YSM) * UMC
      ST(4) =  YMOD * Q * UMC
      ST(5) = 0.0
      ST(6) = 0.0
      GO TO 400
 316  RE(5) = D5
      TR(5) = PP
      RE(4) = D4 - RE(5)
```

```
      RE(3) = DDXX - RE(5) - RE(4)
      TR(3) = TT
      ST(1) = 0.0
      ST(2) = 0.0
      ST(3) = -YMOD * P * UMC
      GO TO 400
  317 RE(4) = D4
      RE(3) = DDXX - RE(4)
      TR(3) = TT
      ST(1) = 0.0
      ST(2) = 0.0
      ST(3) = -YMOD * P * UMC
      ST(5) = (YPL * QEY + YSM) * UMC
      ST(6) = 0.0
      GO TO 400
  318 RE(3) = DDXX
      TR(3) = TT
      ST(1) = 0.0
      ST(2) = 0.0
      ST(3) = -YMOD * P * UMC
      ST(4) =  YMOD * Q * UMC
      ST(5) = 0.0
      ST(6) = 0.0
C
400   CONTINUE
C
      RETURN
C
      END
```

```
C*****************************************************
C
            SUBROUTINE STEP (EPS,HH,WW,TFF,TWW,YMOD,YPL,
     *                 YMOST,EYM,EPM,YSM,YPSM,PP,OM,SM,
     *                  TAREA,EIX1G,EIX2G,EIX3G,EIX4G)
C
C          THIS ROUTINE COMPUTES THE TRANSFORMED SECTION
C       PROPERTIES AND STRESS RESULTANTS FOR I- BEAM.
C
C       *****************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION EPS(5),DDX(4),T(4),RE(5),TR(5),ST(6)
C
        DDX(1) = 0.5 * WW
        DDX(2) = 0.5 * (HH - TFF)
        DDX(3) = DDX(2)
        DDX(4) = DDX(1)
        T(1)   = TFF
        T(2)   = TWW
        T(3)   = T(2)
        T(4)   = T(1)
C
        TAREA = 0.0
        EIX1G = 0.0
        EIX2G = 0.0
        EIX3G = 0.0
        EIX4G = 0.0
        PP    = 0.0
        OM    = 0.0
        SM    = 0.0
C
        DO 900 M=1,4
        GO TO (10,10,20,20),M
10      I = M + 1
        J = I - 1
        GO TO 25
20      I = M
        J = I + 1
25      TT   = T(M)
        DDXX = DDX(M)
C
        CALL STRAIN (EPS,RE,TR,ST,I,J,DDXX,TT,YMOD,YPL,
       *YMOST,EPM,EYM,YSM,YPSM)
C
        TARER = 0.0
        EIX1R = 0.0
        EIX2R = 0.0
        EIX3R = 0.0
        EIX4R = 0.0
        EIY1R = 0.0
        EIY2R = 0.0
        EIY3R = 0.0
```

```
      EIY4R = 0.0
C
      DO 200 N=1,5
      IF(N.GT.1) GO TO 50
      SS = 0.5 * RE(1)
      GO TO 100
50    SS = SS + 0.5 * (RE(N) + RE(N-1))
100   TARER = TARER + RE(N) * TR(N)
      IF(M.EQ.1.OR.M.EQ.4) GO TO 150
      EIX1R = EIX1R + RE(N) * TR(N) * SS
      EIX2R = EIX2R + TR(N) * RE(N) ** 3 / 12.0
     *              + TR(N) * RE(N) * SS ** 2.0
      EIX3R = EIX3R + TR(N) * RE(N) * SS *
     *                (0.25 * RE(N) ** 2 + SS ** 2)
      EIX4R = EIX4R + TR(N) * RE(N) ** 5 / 80.0
     *              + SS**2 * (TR(N) * RE(N)**3 / 2.0
     *                          +TR(N) * RE(N) * SS**2 )
      IF(M.EQ.2.OR.M.EQ.3) GO TO 200
150   EIY2R = EIY2R + RE(N) * TR(N)**3 / 12.0
      EIY4R = EIY4R + RE(N) * TR(N)**5 / 80.0
C
200   CONTINUE
C
      GO TO (310,320,330,310),M
310   HY    = -DDX(2)
      EIX1  = 0.0
      EIX2  = 2. * EIY2R
      EIX3  = 0.0
      EIX4  = 2. * EIY4R
      TARER = 2. * TARER
      IF(M.EQ.4) HY = -HY
      YI    = HY
      C1    = 2.0
      GO TO 400
C
320   HY    =   0.0
      EIX1  = - EIX1R
      EIX2  =   EIX2R
      EIX3  = - EIX3R
      EIX4  =   EIX4R
      YI    = 0.0
      C1    = 1.0
      GO TO 400
C
330   HY    = 0.0
      EIX1  = EIX1R
      EIX2  = EIX2R
      EIX3  = EIX3R
      EIX4  = EIX4R
      YI    = 0.0
      C1    = 1.0
      GO TO 400
C
C
```

```
 400    TAREA = TAREA + TARER
        EIX1G = EIX1G + EIX1 + TARER * HY
        EIX2G = EIX2G + EIX2 + TARER * HY*HY
        EIX3G = EIX3G + EIX3 + HY * (3.*EIX2 + TARER * HY**2)
        EIX4G = EIX4G + EIX4 + HY**2 * (6.*EIX2+TARER*HY**2)
 C
        RPP = 0.0
        ROM = 0.0
        RSM = 0.0
        SUM = 0.0
        STI = ST(1)
 C
        DO 700 N=1,5
        STJ = ST(N+1)
        C2  = RE(N) ** 2
        IF(M.EQ.2) SUM = -RE(N)
        IF(M.EQ.3) SUM =  RE(N)
        IF(M.EQ.1.OR.M.EQ.4) C2 = TT*TT / 2.
        YJ = YI + SUM
 C
        RPP = RPP + C1 * .5 * RE(N) * TT * (STI + STJ)
        ROM = ROM + C1 * RE(N) * TT * (STI*(2.*YI+YJ)+
       *                              STJ*(2.*YJ+YI))/6.0
        RSM = RSM + C1 * RE(N) * TT*(STI*(4.*YI**2+2*YJ**2-C2)
       *                     + STJ*(4.*YJ**2+2.*YI**2-C2))/12.
        STI = STJ
        YI  = YJ
 C
 700    CONTINUE
 C
        PP = PP + RPP
        OM = OM + ROM
        SM = SM + RSM
 C
 900    CONTINUE
 C
        RETURN
 C
        END
```

```
C*******************************************************
C
            SUBROUTINE STEP1 (EPS,HH,WW,TFF,TWW,YMOD,YPL,
     *                     YMOST,EYM,EPM,YSM,YPSM,PP,TAREA)
C
C          THIS ROUTINE COMPUTES THE TRANSFORMED SECTION
C      PROPERTIES AND STRESS RESULTANTS FOR I- BEAM.
C
C      ****************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION EPS(5),DDX(4),T(4),RE(5),TR(5),ST(6)
C
      DDX(1) = 0.5 * WW
      DDX(2) = 0.5 * (HH - TFF)
      DDX(3) = DDX(2)
      DDX(4) = DDX(1)
      T(1)   = TFF
      T(2)   = TWW
      T(3)   = T(2)
      T(4)   = T(1)
C
      TAREA = 0.0
      PP    = 0.0
C
      DO 900 M=1,4
      GO TO (10,10,20,20),M
10    I = M + 1
      J = I - 1
      GO TO 25
20    I = M
      J = I + 1
25    TT   = T(M)
      DDXX = DDX(M)
C
      CALL STRAIN (EPS,RE,TR,ST,I,J,DDXX,TT,YMOD,YPL,
     *YMOST,EPM,EYM,YSM,YPSM)
C
      TARER = 0.0
C
      DO 200 N=1,5
      TARER = TARER + RE(N) * TR(N)
200   CONTINUE
C
      IF(M.EQ.1.OR.M.EQ.4) TARER = 2 * TARER
      TAREA = TAREA + TARER
C
      RPP = 0.0
      STI = ST(1)
      C1 = 1.0D0
      IF(M.EQ.1.OR.M.EQ.4) C1 = 2.0D0
C
      DO 700 N=1,5
```

```
      STJ = ST(N+1)
C
      RPP = RPP + C1 * .5 * RE(N) * TT * (STI + STJ)
      STI = STJ
C
700   CONTINUE
C
      PP = PP + RPP
C
900   CONTINUE
C
      RETURN
C
      END
```

E.6 The Data Managing Package

This package is developed by Elwi and Murray (1977). It is composed of functions ISPAC and LOCOM, subroutines REMOV, and REMOV2, and a BLOCK DATA segment which initializes the common block DIMCOM.

The function ISPAC is called from subroutine MAINMG to define an array in one of five common blocks. This function then calls function LOCOM to check if this array is currently defined in the same common block. If it is not, then ISPAC enters the name of the function in the name directory (NAMES) corresponding to the required common block. ISPAC then adds the length of this array to the current length of this common block, and checks if the updated length of the common block exceeds the maximum limit specified in MAIN. Finally it returns with the position of the first element of this array in the common block. Subroutine REMOV is called from MAINMG to remove any of the last defined arrays in any common block from the name directory (NAMES), and the pointer directory (IPT). Subroutine REMOV2 is called from MAINMG to free an entire common block by initializing the corresponding column in NAMES and IPT. The arguments of these segments are as follows

ISPAC (NAME, LEN, K)

LOCOM (NAME,K)

REMOV (NAME, K)

REMOV2 (K)

E.6.1 Description of the Arguments

NAME : the name of an array to be defined

LEN : the array length

K : A logical number to designate the required common block.

The listing of this package follows.

```
C****************************************************************
C
               FUNCTION ISPAC (NAME,LENGTH,K)
C              **************
C
C       A SIMPLE MANAGER WHICH WORKS WITH 5 FIXED LENGTH
C       COMMON BLOCKS, A 5-COLUMN NAME DIRECTORY AND
C       POINTER DIRECTORY
C
C       *****************************************************
C
        REAL*8 NAMES,NAME
C
        COMMON /DIMCOM/ LAST1,LAST2,LAST3,LAST4,LAST5,MAXDIM,
     *NAMES(5,20),IPT(5,21),ICOM(5)
C
C     CHECK IF NAME ALREADY EXISTS.
C
        ISPACE = LOCOM(NAME,K)
        IF(ISPACE.EQ.0) GO TO 10
        GO TO 100
C
C     ENTER NEW NAME IN DIRECTORY.
C
10      GO TO (20,30,40,50,60),K
20      LAST1 = LAST1 + 1
        LAST  = LAST1
        GO TO 70
30      LAST2 = LAST2 + 1
        LAST  = LAST2
        GO TO 70
40      LAST3 = LAST3 + 1
        LAST  = LAST3
        GO TO 70
50      LAST4 = LAST4 + 1
        LAST  = LAST4
        GO TO 70
60      LAST5 = LAST5 + 1
        LAST  = LAST5
C
70       IF(LAST.GT.MAXDIM) GO TO 200
        NAMES(K,LAST) = NAME
        ISPACE = IPT(K,LAST)
        IPT(K,LAST+1) = ISPACE + LENGTH
        IF((IPT(K,LAST+1)-1).GT.ICOM(K)) GO TO 300
        ISPAC = ISPACE
C
        RETURN
C
C     EXITS RESULTING FROM DIAGNOSED ERRORS
C
100     WRITE(6,1000) NAME
1000    FORMAT(22H***NAME ALREADY EXISTS,10X,A8)
        GO TO 400
```

```
200   WRITE(6,2000) NAME,K
2000  FORMAT(17H***TABLE OVERFLOW ,10X,A8,I4)
      GO TO 400
300   WRITE(6,3000) NAME,K,IPT(K,LAST),LENGTH
3000  FORMAT(23H***COMMON AREA OVERFLOW ,A8,3I4)
400   CALL EXIT
C
      END
```

```
C*****************************************************************
C
                   FUNCTION LOCOM (NAME,K)
C                  **************
C
C       LOCATES INDEX OF A GIVEN NAME IN NAMES DIRECTORY.
C
C       *******************************************************
C
        REAL*8 NAME,NAMES
C
        COMMON /DIMCOM/ LAST1,LAST2,LAST3,LAST4,LAST5,MAXDIM,
       *NAMES(5,20),IPT(5,21),ICOM(5)
C
        GO TO (10,20,30,40,50),K
10      LAST = LAST1
        GO TO 60
20      LAST = LAST2
        GO TO 60
30      LAST = LAST3
        GO TO 60
40      LAST = LAST4
        GO TO 60
50      LAST = LAST5
C
60      IF(LAST.EQ.0) GO TO 200
        DO 100 M=1,LAST
        IF(NAMES(K,M).NE.NAME) GO TO 100
        LOCOM = M
        RETURN
100     CONTINUE
200     LOCOM = 0
C
        RETURN
C
        END
```

```
C*************************************************************
C
                  SUBROUTINE REMOV (NAME,K)
C                 ****************
C
C       REMOVES NAME, IF IT IS THE LAST VARIABLE IN COLUMN K,
C       IN DIRECTORY, AND UPDATES POINTERS ACCORDINGLY.
C
C       ******************************************************
C
        REAL*8 NAME,NAMES
C
        COMMON /DIMCOM/ LAST1,LAST2,LAST3,LAST4,LAST5,MAXDIM,
       *NAMES(5,20),IPT(5,21),ICOM(5)
C
        GO TO (10,20,30,40,50),K
10      LAST = LAST1
        GO TO 60
20      LAST = LAST2
        GO TO 60
30      LAST = LAST3
        GO TO 60
40      LAST = LAST4
        GO TO 60
50      LAST = LAST5
C
60      IF(NAMES(K,LAST).NE.NAME)  GO TO 150
C
C    LAST VARIABLE IN DIRECTORY COLUMN K IS NAME;  REMOVE IT.
C
        IPT(K,LAST+1) = 0
        NAMES(K,LAST) = 0
        GO TO (70,80,90,100,110),K
70      LAST1 = LAST1 - 1
        GO TO 120
80      LAST2 = LAST2 - 1
        GO TO 120
90      LAST3 = LAST3 - 1
        GO TO 120
100     LAST4 = LAST4 - 1
        GO TO 120
110     LAST5 = LAST5 - 1
C
120     RETURN
C
150     WRITE(6,1500) NAME,NAMES(K,LAST)
1500    FORMAT(37H***NAME IS NOT LAST VARIABLE IN NAMES   ,2A8)
        CALL EXIT
C
        END
```

```
C***********************************************************************
C
              SUBROUTINE REMOV2 (K)
C             *****************
C
C     INITIALISES COLUMN K IN NAMES AND IPT, AND LASTK
C
C     *******************************************************
C
      REAL*8 NAME,NAMES
C
      COMMON /DIMCOM/ LAST1,LAST2,LAST3,LAST4,LAST5,MAXDIM,
     *NAMES(5,20),IPT(5,21),ICOM(5)
C
      GO TO (10,20,30,40,50),K
10    LAST = LAST1
      LAST1=0
      GO TO 60
20    LAST = LAST2
      LAST2=0
      GO TO 60
30    LAST = LAST3
      LAST3=0
      GO TO 60
40    LAST = LAST4
      LAST4=0
      GO TO 60
50    LAST = LAST5
      LAST5=0
C
60    LASTN = LAST + 1
      DO 100 J=2,LASTN
      J1 = J - 1
      NAMES(K,J1) = 0
      IPT(K,J)    = 0
100   CONTINUE
C
      RETURN
C
      END
```

```
C*****************************************************
C
      BLOCK DATA
C
      REAL*8 NAME,NAMES
C
      COMMON /DIMCOM/ LAST1,LAST2,LAST3,LAST4,LAST5,MAXDIM,
     *NAMES(5,20),IPT(5,21),ICOM(5)
C
      DATA LAST1,LAST2,LAST3,LAST4,LAST5,MAXDIM,IPT(1,1),
     *IPT(2,1),IPT(3,1),IPT(4,1),IPT(5,1)/5*0,20,5*1/
C
      END
```

E.7 Data Storage and Retrieval Package

This package consists of six subroutines. Subroutines CLEAR and ICLEAR initialize real and integer arrays, respectively. Subroutines STORE1 and STORE2 store respectively integer and real arrays in backing storage, while subroutines RTRV1 and RTRV2 retrieve respectively integer and real arrays from backing storage. The arguments of these subroutines are as follows

CLEAR (ARRAY, LEN)

ICLEAR (IARRAY, LEN)

STORE1 (IO, IARRAY, LEN)

STORE2 (IO, ARRAY, LEN)

RTRV1 (IN, IARRAY, LEN)

RTRV2 (IN, ARRAY, LEN)

E.7.1 Description of the Arguments

ARRAY	:	the first element of a real array to be initialized, stored or retrieved.
IARRAY	:	the first element of an integer array to be initialized, stored, or retrieved.
LEN	:	the length of the array.
IN, IO	:	a logical number which designates a sequential file.

The listing of this package follows.

```
C*****************************************************
C
                SUBROUTINE CLEAR (ARRAY,LEN)
C               ****************
C
C          THIS SUBROUTINE ASSIGNES 0.0 TO A REAL ARRAY OF
C       LENGTH LEN.
C
C       *************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION ARRAY(1)
C
        DO 100 I=1,LEN
        ARRAY(I) = 0.0
100     CONTINUE
C
        RETURN
C
        END
C
C
C
C
C
C**************************************************
C
                SUBROUTINE ICLEAR (IARRAY,LENGTH)
C               ****************
C
C          THIS SEGMENT INITIALISES AN INTEGER ARRAY OF
C       LENGTH (LEN).
C
C       *********************************************
C
        DIMENSION IARRAY(1)
C
        DO 100 I=1,LENGTH
        IARRAY(I) = 0
100     CONTINUE
C
        RETURN
C
        END
```

```
C**************************************************
C
          SUBROUTINE STORE1 (IO,IARRAY,LEN)
          ****************
C
C
C     STORES AN INTEGER ARRAY OF LENGTH LEN ON FILE(IO).
C
C     ************************************************
C
      DIMENSION IARRAY(LEN)
C
      WRITE(IO) IARRAY
C
      RETURN
C
      END
C
C
C
C
C
C
C
C
C**************************************************
C
          SUBROUTINE STORE2 (IO,ARRAY,LEN)
          ****************
C
C
C     STORES A REAL ARRAY OF LENGTH LEN ON FILE(IO)
C
C     ************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION ARRAY(LEN)
C
      WRITE(IO) ARRAY
C
      RETURN
C
      END
```

```
C*****************************************************************
C
            SUBROUTINE RTRV1 (IN,IARRAY,LEN)
C           ****************
C
C              RETRIEVES AN INTEGER ARRAY OF LENGTH LEN,
C       FROM FILE(IN).
C
C       *********************************************************
C
        DIMENSION IARRAY(LEN)
C
        READ(IN) IARRAY
C
        RETURN
C
        END
C
C
C
C
C
C
C*****************************************************************
C
            SUBROUTINE RTRV2 (IN,ARRAY,LEN)
C           ****************
C
C     RETRIEVES A REAL ARRAY OF LENGTH LEN, FROM FILE (IN).
C
C       *********************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION ARRAY(LEN)
C
        READ(IN)ARRAY
C
        RETURN
C
        END
```

E.8 Common Input Subroutines

The following subroutines are common to programs NONELA, INPLAF, PLAFIT, and INSTAF. The subroutines are input subroutines which read the boundary conditions and loading. Description of the subroutines can be found in the listing which follows. These subroutines in alphabetical order are:

BOUND

JLOAD2

MLOAD2

```
C*************************************************************
C
                 SUBROUTINE BOUND (BES,NPB,KODE,NEBEL,IN,IO)
C                ****************
C
C          THIS SEGMENT READS THE EXTERNAL BOUNDARY CONDITIONS
C
C      *******************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION BES(1),NPB(1),KODE(1)
C
      DO 100 J=1,NEBEL
      READ (IN,1000) N,NPB(N),KODE(N),BES(N)
100   IF(BES(N).EQ.0) BES(N) = 1.0E20
      WRITE(IO,2000)
      WRITE(IO,2100) (N,NPB(N),KODE(N),BES(N),N=1,NEBEL)
C
      RETURN
C
C   FORMAT  STATEMENTS
C
1000  FORMAT (3I6,F12.0)
2000  FORMAT (///,T30,'EXTERNAL BOUNDARY ELEMENTS DATA'/,
     *T30,31(1H*)//,
     *3(5X,1HN,4X,3HNPB,1X,4HKODE,8X,2HEB,5X)/)
2100  FORMAT  (3(3I6,D15.6))
C
      END
```

```
C*************************************************************
C
                  SUBROUTINE JLOAD2 (BC,BV,BES,IDO,IN,IO)
C                 ****************
C
C             THIS SEGMENT READS JOINT LOADS AND/OR DISPLACEMENTS
C
C         ***************************************************
C
          IMPLICIT REAL*8(A-H,O-Z)
C
          DIMENSION BC(1),BV(1),BES(1),NBEL(3)
C
C
          WRITE(IO,2000)
100       READ (IN,1000) N,IDESP,(NBEL(I),I=1,3),INC
          IF(N.LE.0) GO TO 900
          N3 =  IDO * N - IDO + 3
          N2 = N3 - 1
          N1 = N3 - 2
          READ (IN,1100) JKODE,LVF,U,V,R
          IF(IDESP.EQ.0) GO TO 300
C
          K = JKODE
          IF((K-100).LT.0) GO TO 225
          U = U * BES(NBEL(1))
          K = K - 100
225       IF((K-10).LT.0) GO TO 250
          V = V * BES(NBEL(2))
          K = K - 10
250       IF(K.EQ.0) GO TO 275
          R = R * BES(NBEL(3))
275       CONTINUE
C
300       WRITE(IO,2100)N,(NBEL(I),I=1,3),INC
          WRITE(IO,2200) JKODE,LVF,U,V,R
          LVF = LVF + 1
          GO TO (400,500),LVF
400       BC(N1) = BC(N1) + U
          BC(N2) = BC(N2) + V
          BC(N3) = BC(N3) + R
          GO TO 550
500       BV(N1) = BV(N1) + U
          BV(N2) = BV(N2) + V
          BV(N3) = BV(N3) + R
550       CONTINUE
          IF(INC.EQ.0) GO TO 800
          NINT = (N-NOLD) / INC - 1
          L = NOLD
          DO 700 I=1,NINT
          LL = L + INC
          L3 = IDO * LL - IDO + 3
          L2 = L3 - 1
          L1 = L3 - 2
```

```
      GO TO (600,650),LVF
600   BC(L1) = BC(L1) + U
      BC(L2) = BC(L2) + V
      BC(L3) = BC(L3) + R
      GO TO 700
650   BV(L1) = BV(L1) + U
      BV(L2) = BV(L2) + V
      BV(L3) = BV(L3) + R
700   L = LL
800   NOLD = N
      GO TO 100
900   CONTINUE
C
      RETURN
C
C        FORMAT STATEMENTS
C
1000  FORMAT (6I5)
1100  FORMAT (2I5,3F12.0)
2000  FORMAT (///,T30,'JOINT LOADS DATA AS INPUT'/,T30,
     *25(1H*)//,'     N   NBU  NBV  NBR  INC    CODE LVF',
     *6X,'U',14X,'V',14X,'R'/)
2100  FORMAT (5I5)
2200  FORMAT (27X,2I5,3D15.6)
C
      END
```

```
C*****************************************************************
C
            SUBROUTINE MLOAD2 (X,Y,FEFC,FEFV,BC,BV,NOD,MKODE,
     *                                    UNIT,IDO,NJ,IN,IO)
C           *****************
C
C              THIS SUBROUTINE READS MEMBER LOADS AND CALCULATES
C       THE END FORCES. THE END FORCES ARE STORED IN VECTOR
C       B AS EQUVALENT LOADS, AND IN VECTOR FEF AS MEMBER END
C       FORCES. THE JOINT LOADS ARE THEN ADDED TO B .
C
C       ******************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION X(1),Y(1),FEFC(1),FEFV(1),BC(1),BV(1),
       *NOD(2,1),MKODE(1),FQ(6),FK(6)
C
C        MODIFY UNITS FOR LOAD VECTOR
C
        DO 200 I=1,NJ
        II = IDO * I - IDO + 3
        BV(II) = BV(II) * UNIT
        BC(II) = BC(II) * UNIT
200     CONTINUE
C
C        READ AND PREPARE MEMBER LOADS
C
        C13 = 1. / 3.
        WRITE(IO,2000)
210     READ (IN,1000) M,LVF,INC,K,CL,CN,QI,AI,QJ,AJ
        IF(M.LE.0) GO TO 800
        WRITE(IO,2100) M,LVF,INC,K,CL,CN,QI,AI,QJ,AJ
        KKK = 0
        MM  = M
        I = NOD(1,MM)
        J = NOD(2,MM)
        DX = (X(J) - X(I))
        DY = (Y(J) - Y(I))
        XL = DSQRT(DX**2+DY**2)
        CT = DX / XL
        ST = DY / XL
        DC = 1. / DSQRT(CL**2+CN**2)
        COSG = DC * CL
        SING = DC * CN
        AIM = 1. - AI
C
        GO TO (300,400,500),K
C
C          FIXED END FORCES FOR CONCENTRATED LOADS
C
300     COSG = COSG * QI
        SING = SING * QI
        FQ(3) = -AIM**2*AI*XL*SING
```

```
      FQ(6) =  AIM*AI**2*XL*SING
      DV    = (FQ(3) + FQ(6)) / XL
      FQ(1) = -AIM*COSG
      FQ(2) = -AIM * SING + DV
      FQ(4) = -AI * COSG
      FQ(5) = -AI * SING - DV
      GO TO 600
C
C        FIXED END FORCES FOR UNIFORM LOADS
C
400   D1    = AJ - AI
      D2    = 0.5 * (AJ**2 - AI**2)
      D3    = C13 * (AJ**3 - AI**3)
      D4    = 0.25 *(AJ**4 - AI**4)
      COSG  = COSG * QI * XL
      SING  = SING * QI * XL
      FQ(3) = -XL * SING * (D2 - 2.*D3 + D4)
      FQ(6) =  XL * SING * (D3 - D4)
      DV    =  (FQ(3) + FQ(6)) / XL
      FQ(1) = -COSG * (D1 - D2)
      FQ(2) = -SING * (D1 - D2) + DV
      FQ(4) = -COSG * D2
      FQ(5) = -SING * D2 - DV
      GO TO 600
C
C        FIXED END FORCES FOR TRAPEZOIDAL LOADS
C
500   AL    = XL / (AJ - AI)
      COSGI = COSG * QI * AL
      COSGJ = COSG * QJ * AL
      SINGI = SING * QI * AL
      SINGJ = SING * QJ * AL
      D1    = AJ - AI
      D2    = 0.5 * (AJ**2 - AI**2)
      D3    = C13 * (AJ**3 - AI**3)
      D4    = .25 * ( AJ**4 - AI**4)
      D5    = 0.2 * (AJ**5 - AI**5)
      D6    = D2 - 2.*D3 + D4
      D7    = D3 - 2.*D4 + D5
      AII   = AJ*D1 - (1.+AJ)*D2 +D3
      BII   = AJ*D6 - D7
      AIJ   = D2 - D3 - AI*(D1-D2)
      BIJ   = D7 - AI*D6
      AJI   = AJ*D2 - D3
      BJI   = AJ*(D3-D4) - (D4-D5)
      AJJ   = D3 - AI*D2
      BJJ   = D4 - D5 - AI*(D3-D4)
C
      FQ(3) = -XL * (SINGI*BII + SINGJ*BIJ)
      FQ(6) =  XL * (SINGI*BJI + SINGJ*BJJ)
      DV    = (FQ(3) + FQ(6)) / XL
      FQ(1) = -(COSGI*AII + COSGJ*AIJ)
      FQ(2) = -(SINGI*AII + SINGJ*AIJ) + DV
      FQ(4) = -(COSGI*AJI + COSGJ*AJJ)
```

```
      FQ(5) = -(SINGI*AJI + SINGJ*AJJ) - DV
C
C        MODIFY END FORCES VECTOR FQ FOR HINGED ENDS
C
600   CONTINUE
      KI = MKODE(MM) + 1
      DO 605 KK = 1,6
605   FK(KK) = FQ(KK)
      GO TO (650,620,630,610),KI
610   FK(3) = 0.0
      FK(6) = 0.0
      GO TO 640
620   FK(6) = FK(6) - 0.5*FK(3)
      DV    = 1.5*FK(3) / XL
      FK(3) = 0.0
      GO TO 640
630   FK(3) = FK(3) - 0.5*FK(6)
      DV    = 1.5*FK(6) / XL
640   FK(2) = FK(2) - DV
      FK(5) = FK(5) + DV
650   FK(3) = FK(3) * UNIT
      FK(6) = FK(6) * UNIT
C
C        ASSEMBLE FQ INTO FEF
C
      II = IDO * NOD(1,MM)
      JJ = IDO * NOD(2,MM)
      LVF = LVF + 1
      GO TO (660,670),LVF
660   DO 665 KK=1,6
      KEF = KEF + 1
      FEFC(KEF) = FEFC(KEF) + FK(KK)
665   CONTINUE
      BC(II-2) = BC(II-2) - FK(1)*CT + FK(2)*ST
      BC(II-1) = BC(II-1) - FK(1)*ST - FK(2)*CT
      BC(II)   = BC(II)   - FK(3)
      BC(JJ-2) = BC(JJ-2) - FK(4)*CT + FK(5)*ST
      BC(JJ-1) = BC(JJ-1) - FK(4)*ST - FK(5)*CT
      BC(JJ)   = BC(JJ)   - FK(6)
      GO TO 700
670   DO 675 KK=1,6
      KEF = KEF + 1
      FEFV(KEF) = FEFV(KEF) + FK(KK)
675   CONTINUE
      BV(II-2) = BV(II-2) - FK(1)*CT + FK(2)*ST
      BV(II-1) = BV(II-1) - FK(1)*ST - FK(2)*CT
      BV(II)   = BV(II)   - FK(3)
      BV(JJ-2) = BV(JJ-2) - FK(4)*CT + FK(5)*ST
      BV(JJ-1) = BV(JJ-1) - FK(4)*ST - FK(5)*CT
      BV(JJ)   = BV(JJ)   - FK(6)
700   IF(INC.EQ.0) GO TO 750
      MOLD = MOLD +  IABS(INC)
      MM = MOLD
      IF(MM.GT.M) GO TO 999
```

```
      IF(MM.EQ.M) GO TO 750
      IF(INC.EQ.0) GO TO 750
      GO TO 600
750   CONTINUE
      MOLD = MM
      GO TO 210
C
800   CONTINUE
C
      RETURN
C
999   WRITE(IO,9999) MM,M
      STOP
C
C        FORMAT STATEMENTS
C
1000  FORMAT (4I4,2F12.0,4F10.0)
2000  FORMAT (///,T30,'MEMBER LOADS AS INPUT'//,4X,
     *'M  LVF  INC CODE',6X,'CL',13X,'CN',13X,'QI',
     *13X,'AI',13X,'QJ',13X,'AJ'/)
2100  FORMAT (4I5,6D15.6)
9999  FORMAT ('GENERATION INCREMENT ERROR    MM=',I5,
     *'     M=',I5)
C
      END
```

E.9 Common Subroutines in the Equation Solving Package

Two subroutines in the equation solving package are common to all programs. These are ADDRES and COLHT. Description of the subroutines can be found in the listing which follows.

```
C*****************************************************
C
            SUBROUTINE ADDRES (MAXA,MHT,NEQ,NWA)
C           ****************
C
C              THIS SUBROUTINE CALCULATES THE ADDRESSES OF THE
C           DIAGONAL ELEMENTS  AND LENGTH OF A STIFFNESS MATRIX
C           UPPER TRIANGLE STORED COLUMN-WISE UNDER A SKYLINE.
C
C     ***************************************************
C
      DIMENSION MAXA(1),MHT(1)
C
      NM  = NEQ + 1
C
      MAXA(1) = 1
      MAXA(2) = 2
C
      IF(NEQ.EQ.1) GO TO 30
      DO 20 I=2,NEQ
      MAXA(I+1) = MAXA(I) + MHT(I) + 1
20    CONTINUE
C
30    NWA = MAXA(NM) - 1
      RETURN
C
      END
```

```
C*********************************************************
C
            SUBROUTINE COLHT (NB,ND,IDO,ME,MHT,NP)
C           ****************
C
C        THIS SUBROUTINE IS CALLED PER ELEMENT TO FORM AND
C     UPDATE THE COLUMN HEIGHT ARRAY(MHT).
C
C     ****************************************************
C
      DIMENSION MHT(1),NP(NB,1),LM(6)
C
      I = NP(1,ME)
      J = NP(2,ME)
      DO 100 KK=1,IDO
      LM(KK)      = IDO * I - IDO + KK
100   LM(KK+IDO) = IDO * J - IDO + KK
120   LS = 10000
C
      DO 200 I=1,ND
      IF(LM(I)-LS) 150,200,200
150   LS = LM(I)
200   CONTINUE
C
      DO 300 I=1,ND
      II = LM(I)
      MB = II - LS
      IF(MB.GT.MHT(II)) MHT(II) = MB
300   CONTINUE
C
      RETURN
C
      END
```

E.10 Common Subroutines in Formulation and Output

This group of subroutines is used in the formulation. They are common in some of the programs. Descriptions of the subroutines can be found in the listing which follows. These subroutines in alphabetical order are

ASSEMB

BOUND1

BOUND2

CONVER

DINCR

DISPL1

ELEMS

MNHING

MODIFY

MULT1

MULT2

SHAPE

STRES1

```
C*************************************************************
C
            SUBROUTINE ASSEMB (MAXA,S,A,LM,ND)
C           ****************
C
C         THIS SUBROUTINE ASSEMBLES THE ELEMENT STIFFNESS
C      MATRIX INTO THE STRUCTURE STIFFNESS MATRIX IN GLOBLE
C      COORDINATES.
C
C      ********************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION MAXA(1),A(1),S(21),LM(ND)
C
C      ADD STIFFNESS MATRIX TO STRUCTURE STIFFNESS MATRIX
C
300   NDL = 0
      DO 600 L=1,ND
      LL = LM(L)
      ML = MAXA(LL)
      KS = L
      DO 500 N=1,ND
      NN = LM(N)
      LN = LL - NN
      IF(LN) 500,400,400
400   KK = ML + LN
      KSS = KS
      IF(N.GE.L) KSS=N+NDL
      A(KK) = A(KK) + S(KSS)
500   KS = KS + ND - N
600   NDL = NDL + ND - L
C
      RETURN
C
      END
```

```
C*************************************************************
C
            SUBROUTINE BOUND1 (A,MAXA,NPB,KODE,BES,NEBEL)
C           ****************
C
C           THIS ROUTINE ADDS EXTERNAL BOUNDARY CONDITIONS
C         TO STIFFNESS MATRIX A .
C
C     ********************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION A(1),MAXA(1),NPB(1),KODE(1),BES(1)
C
      DO 300 K=1,NEBEL
      N   = NPB(K)
      KOD = KODE(K)
      EX  = BES(K)
      NN  = 3 * N
      IF((KOD-100).LT.0) GO TO 100
      NK     = MAXA(NN-2)
      A(NK) = A(NK) + EX
      KOD    = KOD - 100
100   IF((KOD-10).LT.0) GO TO 200
      NK     = MAXA(NN-1)
      A(NK) = A(NK) + EX
      KOD   = KOD - 10
200   IF(KOD.EQ.0) GO TO 300
      NK     =  MAXA(NN)
      A(NK) = A(NK) + EX
300   CONTINUE
C
      RETURN
C
      END
```

```
C*************************************************************
C
       SUBROUTINE BOUND2 (A,B,MAXA,NPB,KODE,BES,NJ,NEBEL,IO)
C      ****************
C
C        THIS ROUTINE ADDS EXTERNAL BOUNDARY CONDITIONS
C      TO STIFFNESS MATRIX A .
C
C      ******************************************************
C
       IMPLICIT REAL*8(A-H,O-Z)
C
       DIMENSION A(1),B(1),MAXA(1),NPB(1),KODE(1),BES(1)
C
       DO 500 K=1,NEBEL
       N   = NPB(K)
       KOD = KODE(K)
       EX  = BES(K)
       NN  = 5 * N
       IF((KOD-10000).LT.0) GO TO 100
       NK     = MAXA(NN-4)
       B(NN-4)= 0.0
       A(NK)  = A(NK) + EX
       KOD    = KOD - 10000
100    IF((KOD-1000).LT.0) GO TO 200
       NK     = MAXA(NN-3)
       B(NN-3)= 0.0
       A(NK)  = A(NK) + EX
       KOD    = KOD - 1000
200    IF((KOD-100).LT.0) GO TO 300
       NK     =  MAXA(NN-2)
       B(NN-2)= 0.0
       A(NK)  = A(NK) + EX
       KOD    = KOD - 100
300    IF((KOD-10).LT.0) GO TO 400
       NK     = MAXA(NN-1)
       B(NN-1)= 0.0
       A(NK)  = A(NK) + EX
       KOD    = KOD - 10
400    IF(KOD.EQ.0) GO TO 500
       NK     = MAXA(NN)
       B(NN)  = 0.0
       A(NK)  = A(NK) + EX
500    CONTINUE
       WRITE(IO,2000)
       DO 800 N=1,NJ
       N5 = N * 5
       N1 = N5 - 4
800    WRITE(IO,2100) N,(B(NN),NN=N1,N5)
       RETURN
C
C FORMAT STATEMENTS
C
2000   FORMAT (//,'NODAL UNBALANCED FORCES'//,5X,'N',6X,
```

```
     *'FU',13X,'FV',13X,'FR',13X,'FDU',13X,'FDV'/)
2100  FORMAT (I6,5D15.6)
C
      END
```

```
C*****************************************************************
C
      SUBROUTINE CONVER (DB,C,CONVD,IT,TOL,ITLIM,NEQ,IF,IO)
C     ****************
C
C        THIS SUBROUTINE TESTS THE CONVERGENCE IN LOAD
C      AND DISPLACEMENT.
C
C      ***********************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION DB(1)
C
      DF = 0.0
      DO 100 I=1,NEQ
100   DF = DF + DB(I) * DB(I)
      DF = DSQRT(DF)
      CONVB = DF / C
      IF(CONVB.LT.TOL.AND.CONVD.LT.TOL) GO TO 300
      IF(IT.GT.ITLIM) GO TO 300
      WRITE(IO,2000) IT,CONVB,CONVD
      IF = 1
      GO TO 400
300   WRITE(IO,2000) IT,CONVB,CONVD
      IF = 0
C
400   RETURN
C
C   FORMAT STATEMENT
C
2000  FORMAT (//,5X,'ITERATION NO.',I5,5X,
     *'LOAD TOLERANCE  =',F12.6,5X,
     *'DISPL. TOLERANCE  =',F12.6//,5X,85(1H*)//)
C
      END
```

```
C*****************************************************
C
              SUBROUTINE DINCR (B,DD,TD,NEQ,IT,C,CONVD,NJ,IO)
C             ***************
C
C               THIS ROUTINE EVALUATES THE SQUARE ROOT OF SUM
C            OF THE SQUARES OFTHE LOAD VECTOR. ALSO EVALUATES
C            THE CONVERGENCE IN DISPLACEMENTS.
C
C            ***********************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION B(1),DD(1),TD(1)
C
      IF(IT.GT.2) GO TO 150
      C = 0.0
      DO 100 I=1,NEQ
100   C = C + B(I) * B(I)
      C = DSQRT(C)
150   D  = 0.0
      D1 = 0.0
      DO 200 I=1,NEQ
      D = D + DD(I) * DD(I)
      DD(I) = DD(I) + TD(I)
200   D1 = D1 + DD(I) * DD(I)
      D  = DSQRT(D)
      D1 = DSQRT(D1)
      CONVD = D / D1
C
      WRITE(IO,2000)
      DO 300 N=1,NJ
      N5 = N * 5
      N1 = N5 - 4
300   WRITE(IO,2100) N,(DD(NN),NN=N1,N5)
C
      RETURN
C
C   FORMAT STATEMENTS
C
2000  FORMAT (//,'TOTAL NODAL DISPLACEMENTS',//,4X,'N',
     *7X,'U',14X,'V',14X,'R',13X,'DU',13X,'DV'/)
2100  FORMAT (I4,5D15.6)
C
      END
```

```
C**********************************************************
C
              SUBROUTINE DISPL1 (FEF2,B,FEF,NJ,NE,IDO,IN,IO)
C             ****************
C
C          THIS SUBROUTINE OUTPUTS NODAL DISPLACEMENTS FOR LOAD
C       CASE (K).  IT PREPARES DUMMY VECTOR FEF2 TO CALCULATE
C       OF MEMBER END FORCES.
C
C       ****************************************************
C
        IMPLICIT REAL*8(A-H,O-Z)
C
        DIMENSION FEF2(1),B(1),FEF(1)
C
        WRITE(IO,2000)
        DO 100 N=1,NJ
        N2 = N * IDO
        N1 = N2 - IDO + 1
100     WRITE(IO,2100) N,(B(NN),NN=N1,N2)
        DO 300 II=1,NE
        III = 6*II + 1
        DO 300 JJ=1,6
        FEF2(III-JJ) = FEF(III-JJ)
300     CONTINUE
C
        RETURN
C
C         FORMAT STATEMENTS
C
2000    FORMAT (//,'NODAL DISPLACEMENTS',//
     *,4X,'N',7X,'U',14X,'V',14X,'R',13X,'DU',13X,'DV'/)
2100    FORMAT (I5,5D15.6)
C
        END
```

```
C*****************************************************
C
                 SUBROUTINE ELEMS (A1,S,CL,SL,CL2,SL2,CSL,
     *                                          C2,S2,SC)
C                ****************
C
C           THIS SUBROUTINE FORMS THE ELEMENT STIFFNESS
C     MATRIX (ELASTIC OR GEOMETRIC)
C
C     **************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION A1(7),S(21)
C
      S(1)  = A1(1) * C2 + A1(2) * SL2
      S(2)  = A1(1) * SC - A1(2) * CSL
      S(3)  = -A1(3) * SL
      S(4)  = -S(1)
      S(5)  = -S(2)
      S(6)  = -A1(4) * SL
      S(7)  = A1(1) * S2 +A1(2) * CL2
      S(8)  = A1(3) * CL
      S(9)  = -S(2)
      S(10) = -S(7)
      S(11) = A1(4) * CL
      S(12) = A1(5)
      S(13) = -S(3)
      S(14) = -S(8)
      S(15) = A1(6)
      S(16) = S(1)
      S(17) = S(2)
      S(18) = -S(6)
      S(19) = S(7)
      S(20) = -S(11)
      S(21) = A1(7)
C
      RETURN
C
      END
```

```
C*************************************************************
C
          SUBROUTINE MNHING (NOD,MKOD,NPB,MAXNH,NE,NJ,NEBEL)
C         *****************
C
C         THIS SUBROUTINE CALCULATES THE MAXIMUM NUMBER
C      OF HINGES THAT CAN BE FORMED AT ONE JOINT.
C
C      ************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION NOD(2,1),MKOD(1),NPB(1),MAXNH(1)
C
      DO 100 M=1,NE
      I = NOD(1,M)
      J = NOD(2,M)
      K = MKOD(M) + 1
      GO TO (50,60,70,100),K
50    MAXNH(I) = MAXNH(I) + 1
      MAXNH(J) = MAXNH(J) + 1
      GO TO 100
60    MAXNH(I) = MAXNH(I)
      MAXNH(J) = MAXNH(J) + 1
      GO TO 100
70    MAXNH(I) = MAXNH(I) + 1
      MAXNH(J) = MAXNH(J)
100   CONTINUE
C
      DO 200 I=1,NJ
      MAXNH(I) = MAXNH(I) - 1
200   CONTINUE
C
      DO 300 I=1,NEBEL
      K = NPB(I)
      MAXNH(K) = MAXNH(K) + 1
300   CONTINUE
C
      RETURN
C
      END
```

```
C*********************************************************
C
                SUBROUTINE MODIFY (K,C)
C               ****************
C
C          THIS SUBROUTINE MODIFIES THE STIFFNESS MATRIX FOR
C        EACH MEMBER THAT IS NOT FIXED AT BOTH ENDS
C
C       *****************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION C(7)
C
      GO TO (100,200,100),K
100   C(2) = C(2) - C(3) * C(3) / C(5)
      C(4) = C(4) - C(3) * C(6) / C(5)
      C(7) = C(7) - C(6) * C(6) / C(5)
      C(3) = 0.0
      C(5) = 0.0
      C(6) = 0.0
      IF(K.EQ.3) GO TO 200
      GO TO 300
200   C(2) = C(2) - C(4) * C(4) / C(7)
      C(3) = C(3) - C(4) * C(6) / C(7)
      C(5) = C(5) - C(6) * C(6) / C(7)
      C(4) = 0.0
      C(6) = 0.0
      C(7) = 0.0
300   CONTINUE
C
      RETURN
C
      END
```

```
C*************************************************************
C
            SUBROUTINE MULT1 (TT,B,V,MAXA,NN,NWA)
C            ***************
C
C           THIS SUBROUTINE EVALUATES THE PRODUCT OF MATRIX
C        B (IN COMPACTED FORM) TIMES VECTOR V AND STORE
C        RESULT IN TT.
C
C        ***************************************************
C
         IMPLICIT REAL*8(A-H,O-Z)
C
         DIMENSION TT(1),B(1),V(1),MAXA(1)
C
         IF(NWA.GT.NN) GO TO 20
         DO 10 I=1,NN
10       TT(I) = B(I) * V(I)
C
         RETURN
C
20       DO 40 I=1,NN
40       TT(I) = 0.
         DO 100 I=1,NN
         KL = MAXA(I)
         KU = MAXA(I+1) - 1
         II = I + 1
         CC = V(I)
         DO 100 KK=KL,KU
         II = II - 1
100      TT(II) = TT(II) + B(KK) * CC
         IF(NN.EQ.1) RETURN
         DO 200 I=2,NN
         KL = MAXA(I) + 1
         KU = MAXA(I+1) - 1
         IF(KU-KL) 200,210,210
210      II = I
         AA = 0.0
         DO 220 KK=KL,KU
         II = II - 1
220      AA = AA + B(KK) * V(II)
         TT(I) = TT(I) + AA
200      CONTINUE
C
         RETURN
C
         END
```

```
C*********************************************************************
C
            SUBROUTINE MULT2 (TT,B,V,MAXA,NR,NC)
C           ****************
C
C              THIS SUBROUTINE EVALUATES THE PRODUCT OF MATRIX
C         B (IN COMPACTED FORM) TIMES MATRIX V AND STORE
C         RESULT IN TT.
C
C         *****************************************************
C
          IMPLICIT REAL*8(A-H,O-Z)
C
          DIMENSION TT(8,10),B(1),V(8,10),MAXA(1)
C
C
          DO 40 J=1,NC
          DO 40 I=1,NR
40        TT(I,J) = 0.
          DO 150 J=1,NC
          DO 150 I=1,NR
          IJ = I + 8
          KL = MAXA(IJ)
          KU = MAXA(IJ+1) - 1
          II = I + 1
          CC = V(I,J)
          DO 100 KK=KL,KU
          II = II - 1
          IF(II.LT.1) GO TO 150
100       TT(II,J) = TT(II,J) + B(KK) * CC
150       CONTINUE
          IF(NR.EQ.1) RETURN
          DO 200 J=1,NC
          DO 200 I=2,NR
          IJ = I + 8
          KL = MAXA(IJ) + 1
          KU = MAXA(IJ+1) - 1
          IF(KU-KL) 200,210,210
210       II = I
          AA = 0.0
          DO 220 KK=KL,KU
          II = II - 1
          IF(II.LT.1) GO TO 230
220       AA = AA + B(KK) * V(II,J)
230       TT(I,J) = TT(I,J) + AA
200       CONTINUE
C
          RETURN
C
          END
```

```
C*************************************************************
C
                SUBROUTINE SHAPE (XI,PHI,PHIP,PHIDP,IG)
C               ****************
C
C          THIS SUBROUTINE EVALUATES THE SHAPE VECTOR
C      DERIVATIVES FOR EACH GAUSS POINT IN A BEAM ELEMENT.
C
C      *******************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION PHI(4,1),PHIP(4,1),PHIDP(4,1)
      XIP1  = XI + 1.
      XIP2  = XI + 2.
      XIP12 = XIP1 * XIP1
      XIM1  = XI - 1.
      XIM2  = XI - 2.
      XIM12 = XIM1 * XIM1
      PHI(1,IG)   =  0.25  * XIP2 * XIM12
      PHI(2,IG)   =  0.125 * XIP1 * XIM12
      PHI(3,IG)   = -0.25  * XIM2 * XIP12
      PHI(4,IG)   =  0.125 * XIM1 * XIP12
      PHIP(1,IG)  = XIP2*XIM1 + .5*XIM12
      PHIP(2,IG)  = 0.5*XIP1*XIM1 + 0.25*XIM12
      PHIP(3,IG)  = -XIM2*XIP1 - 0.5*XIP12
      PHIP(4,IG)  = 0.5*XIM1*XIP1 + 0.25*XIP12
      PHIDP(1,IG) = 2. * (XIP2 + 2.*XIM1)
      PHIDP(2,IG) = XIP1 + XIM1 + XIM1
      PHIDP(3,IG) = -2. * (XIM2 + XIP1 + XIP1)
      PHIDP(4,IG) = XIM1 + XIP1 + XIP1
C
      RETURN
C
      END
```

```
C*************************************************************
C
            SUBROUTINE STRES1 (X,Y,AREA,RI,YMOD,B,FEF2,PP,NOD,
     *                       MKODE,NE,IDO,UNIT,IN,IO)
C           ****************
C
C              SUBROUTINE STRES1 COMPUTES AND OUTPUTS MEMBER
C           END FORCES FOR A PLANE FRAME PROBLEM.
C
C     *******************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      DIMENSION X(1),Y(1),AREA(1),RI(1),YMOD(1),B(1),PP(1),
     *FEF2(1),NOD(2,1),MKODE(1)
C
      DO 500 M=1,NE
      I    = NOD(1,M)
      J    = NOD(2,M)
      M1 = 6 * M - 5
      DX   = (X(J) - X(I)) * UNIT
      DY   = (Y(J) - Y(I)) * UNIT
      XLI = 1. / DSQRT(DX**2+DY**2)
      COST= DX * XLI
      SINT= DY * XLI
      C1   = 2. * YMOD(M) * RI(M) * XLI
      C2   = AREA(M) * YMOD(M) * XLI
C
      DU   = B(IDO*J-IDO+1) - B(IDO*I-IDO+1)
      DV   = B(IDO*J-IDO+2) - B(IDO*I-IDO+2)
      DP   = C2 * (DU*COST + DV*SINT)
      ROT  = 3. * (DV*COST - DU*SINT) * XLI
      DMI  = C1 * (2.*B(IDO*I-IDO+3) + B(IDO*J-IDO+3) - ROT)
      DMJ  = C1 * (2.*B(IDO*J-IDO+3) + B(IDO*I-IDO+3) - ROT)
      DV   = (DMI + DMJ) * XLI + 0.4 * PP(M1) * ROT
C
C        MODIFY END FORCES FOR MEMBER TYPE
C
      KK = MKODE(M)
      IF(KK.EQ.0) GO TO 400
      GO TO (100,200,300),KK
100   DMJ = DMJ - DMI*.5
      DV  = DV  - 1.5*DMI*XLI
      DMI = 0.0
      GO TO 400
200   DMI = DMI - 0.5*DMJ
      DV  = DV  - 1.5*DMJ*XLI
      DMJ = 0.0
      GO TO 400
300   DV  = DV - (DMI + DMJ)*XLI
      DMI = 0.0
      DMJ = 0.0
C
400   MM = 6*M
```

```
      FEF2(MM-5) = FEF2(MM-5) - DP
      FEF2(MM-4) = FEF2(MM-4) + DV
      FEF2(MM-3) =(FEF2(MM-3) + DMI)/UNIT
      FEF2(MM-2) = FEF2(MM-2) + DP
      FEF2(MM-1) = FEF2(MM-1) - DV
      FEF2(MM)   =(FEF2(MM)   + DMJ)/UNIT
500   CONTINUE
C
      WRITE(IO,2000)
      DO 600 M=1,NE
      M2 = 6 * M
      M1 = M2 - 5
      WRITE(IO,2100) M,NOD(1,M),NOD(2,M),(FEF2(I),I=M1,M2)
600   CONTINUE
C
      RETURN
C
C        FORMAT STATEMENTS
C
2000  FORMAT (///,'MEMBER END FORCES',//,3X,'MEM',3X,'I',4X,
     *'J',7X,'NI',13X,'VI',13X,'MI',13X,'NJ',13X,'VJ',13X,
     *'MJ'/)
2100  FORMAT (3I5,6D15.6)
C
      END
```

APPENDIX F

Example INPUT and OUTPUT

F.1 Buckling Analysis

```
1- BUCKLING ANALYSIS
0,
4,3,2,1,1,1,12.0,
1,0,0,
2,0,10,
3,10,10,
4,10,0,
9.12,110,30000,
1,1,2,0,1,0,0,0,0,0,0,
2,2,3,0,1,0,0,0,0,0,0,
3,3,4,0,1,0,0,0,0,0,0,
0,
1,1,111,
2,4,111,
1,0,
1,1,1,
2,0,
0,0,-1,0,
3,0,
0,0,-1,0,
0,
.00001,10,1,0,
```

```
1- BUCKLING ANALYSIS

*******************

PROBLEM CONTROL VARIABLES
*************************

     NUMBER OF JOINTS                   =    4

     NUMBER OF ELEMENTS                 =    3

     NO. OF EXTERNAL BOUND. ELEMENTS    =    2

     NUMBER OF LOAD CASES               =    1

     NUMBER OF LOAD TYPES               =    1

     EIGEN PROBLEM RUN FLAG             =    1

     LENGTH CONVERSION FACTOR           =   12.000000
     **************************************************


NODAL GEOMETRY DATA AS INPUT
****************************

     N        X                 Y            INC

     1   0.0               0.0                0
     2   0.0               0.100000E+02       0
     3   0.100000E+02      0.100000E+02       0
     4   0.100000E+02      0.0                0
```

MEMBER PROPERTIES DEFAULT VALUES
****

AREA	=	0.912000E+01
M. INERTIA	=	0.110000E+03
Y. MODULUS	=	0.300000E+05

MEMBER DATA AS INPUT
****

M	I	J	CODE	FLAG	INC	INCI	INCJ	AREA	I	YMOD
1	1	2	0	1	0	0	0	0.912000E+01	0.110000E+03	0.300000E+05
2	2	3	0	1	0	0	0	0.912000E+01	0.110000E+03	0.300000E+05
3	3	4	0	1	0	0	0	0.912000E+01	0.110000E+03	0.300000E+05

EXTERNAL BOUNDARY ELEMENTS DATA
****

N	NPB	KODE	EB	N	NPB	KODE	EB	N	NPB	KODE	EB
1	1	111	0.100000E+21	2	4	111	0.100000E+21				

LOAD CONTROL PARAMETERS
****

LOAD TYPE	LVF	LOAD FACTOR
1	1	1.000000

JOINT LOADS DATA AS INPUT
****

N	NBU	NBV	NBR	INC	CODE	LT	LVF	U	V	R
2	0	0	0	0	0	1	1	0.0	-0.100000E+01	0.0
3	0	0	0	0	0	1	1	0.0	-0.100000E+01	0.0

BACK SUBSTITUTION OF STRUCTURE PROBLEM IS COMPLETED

1LOAD CASE NO. 1

NODAL DISPLACEMENTS

N	U	V	R
1	0.476838E-39	-0.100000E-19	0.586671E-37
2	-0.169616E-19	-0.438596E-03	0.317372E-21
3	-0.169407E-19	-0.438596E-03	0.864256E-22
4	-0.216840E-38	-0.100000E-19	0.138778E-36

MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.100000E+01	0.476837E-19	-0.488892E-18	-0.100000E+01	-0.476837E-19	0.965729E-18
2	2	3	-0.476837E-19	0.179754E-17	0.951694E-17	0.476837E-19	-0.179754E-17	0.845843E-17
3	3	4	0.100000E+01	-0.269388E-18	-0.114888E-17	-0.100000E+01	0.269388E-18	-0.154500E-17

EIGEN PROBLEM BEGINS
****

EIGEN PROBLEM CONTROL PARAMETERS

TOLERANCE LIMIT	=	0.000010
MAXIMAUM NO. OF ITERATIONS	=	10
PRINT CODE	=	1
INPUT CODE	=	0

****

ASSUMED EIGENVECTOR

0.0	0.0	0.0	1.000000	1.000000	1.000000	1.000000	1.000000	
1.000000	0.0	0.0	0.0					

```
ITERATE    1   HAS BEEN COMPLETED   WITH TOLERANCE OF    0.3940E+01

EIGENVALUE =  0.17881E-02     LOAD FACTOR =  559.257623

THE EIGENVECTOR IS :

  0.000000 -0.000000 -0.000000  1.000000 -0.031150  0.028074  1.000000  0.035238
  0.028074  0.000000  0.000000 -0.000000

ITERATE    2   HAS BEEN COMPLETED   WITH TOLERANCE OF    0.1302E+00

EIGENVALUE =  0.62828E-03     LOAD FACTOR = 1591.648555

THE EIGENVECTOR IS :

  0.000000  0.000000 -0.000000  1.000000  0.002000 -0.001841  1.000000 -0.001976
 -0.001841  0.000000 -0.000000 -0.000000

ITERATE    3   HAS BEEN COMPLETED   WITH TOLERANCE OF    0.1254E-01

EIGENVALUE =  0.59269E-03     LOAD FACTOR = 1687.209973

THE EIGENVECTOR IS :

  0.000000  0.000000 -0.000000  1.000000  0.005477 -0.004622  1.000000 -0.005477
 -0.004622  0.000000 -0.000000 -0.000000

ITERATE    4   HAS BEEN COMPLETED   WITH TOLERANCE OF    0.1245E-02

EIGENVALUE =  0.58939E-03     LOAD FACTOR = 1696.676774

THE EIGENVECTOR IS :

  0.000000  0.000000 -0.000000  1.000000  0.005824 -0.004898  1.000000 -0.005824
 -0.004898  0.000000 -0.000000 -0.000000


EIGENVALUE  = 0.58939E-03     LOAD FACTOR = 1696.676774

AND THE VECOR IS :
  0.1897E-33  0.1320E-18 -0.9046E-17  0.1000E+01  0.5824E-02 -0.4898E-02  0.1000E+01 -0.5824E-02
 -0.4898E-02  0.1941E-33 -0.1320E-18 -0.9046E-17
```

F.2 Nonlinear Elastic Analysis

```
2-NONLINEAR ANALYSIS
0,0,1,1,
4,3,3,2,1,10,.00001,12,
1,0,0,
2,0,10,
3,10,10,
4,10,0,
9.12,110,1500,30000,
1,1,2,0,0,0,0,0,0,0,
2,2,3,0,0,0,0,0,0,0,
3,3,4,0,0,0,0,0,0,0,
0,
1,1,11110,
2,4,11110,
1,0,
2,0,
0,1,.5,-1,0,
3,0,
0,1,0,-1,0,
0,
```

```
                    2-NONLINEAR ANALYSIS

                    ********************

PROBLEM CONTROL VARIABLES
*************************

     NUMBER OF JOINTS                      =    4

     NUMBER OF FLEXURAL ELEMENTS           =    3

     NUMBER OF ELEMENTS                    =    3

     NO. OF EXTERNAL BOUNDARY ELEMENTS     =    2

     NUMBER OF DIFFERENT MEMBER TYPES      =    1

     MAXIMUM NO. OF ITERATIONS             =   10

     TOLERANCE LIMIT                       =    0.000010

     LENGTH CONVERGENCE FACTOR             =   12.000000

     **************************************************

NODAL GEOMETRY DATA AS INPUT
****************************

     N       X               Y            INC

     1   0.0             0.0               0
     2   0.0             0.100000E+02      0
     3   0.100000E+02    0.100000E+02      0
     4   0.100000E+02    0.0               0
```

FLEXURAL MEMBER PROP. DEFAULT VALUES
****

AREA	=	0.912000E+01
M. INERTIA	=	0.110000E+03
FOURTH MOM. OF AREA	=	0.150000E+04
Y. MODULUS	=	0.300000E+05

FLEXURAL MEMBER DATA
****

M	I	J	INC	INCI	INCJ	AREA	I	I4	YMOD
1	1	2	0	0	0	0.912000E+01	0.110000E+03	0.150000E+04	0.300000E+05
2	2	3	0	0	0	0.912000E+01	0.110000E+03	0.150000E+04	0.300000E+05
3	3	4	0	0	0	0.912000E+01	0.110000E+03	0.150000E+04	0.300000E+05

EXTERNAL BOUNDARY ELEMENTS DATA
****

N	NPB	KODE	EB	N	NPB	KODE	EB	N	NPB	KODE	EB
1	1	11110	0.100000E+21	2	4	11110	0.100000E+21				

JOINT LOADS DATA AS INPUT
****

N	NBU	NBV	NBR	INC	CODE	LVF	U	V	R
2	0	0	0	0	0	1	0.500000E+00	-0.100000E+01	0.0
3	0	0	0	0	0	1	0.0	-0.100000E+01	0.0

STRESS RESULTANTS

MEMBER NO.	NODI	NODJ	PI	MI	MI*	PJ	MJ	MJ*
1	1	2	0.0	-0.0	0.0	0.0	-0.0	0.0
2	2	3	0.0	-0.0	0.0	0.0	-0.0	0.0
3	3	4	0.0	-0.0	0.0	0.0	-0.0	0.0

NODAL UNBALANCED FORCES

N	FU	FV	FR	FDU	FDV
1	0.0	0.0	0.0	0.0	0.0
2	0.500000E+00	-0.100000E+01	0.0	0.0	0.0
3	0.0	-0.100000E+01	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0

NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.250627E-20	-0.786328E-20	-0.172298E-18	-0.0	-0.287401E-05
2	0.157194E-01	-0.344881E-03	-0.797166E-04	-0.911453E-06	-0.287401E-05
3	0.156101E-01	-0.532312E-03	-0.788052E-04	-0.911453E-06	-0.443594E-05
4	0.249373E-20	-0.121367E-19	-0.171295E-18	-0.0	-0.443594E-05

STRESS RESULTANTS

MEMBER NO.	NODI	NODJ	PI	MI	MI*	PJ	MJ	MJ*
1	1	2	-0.786282E+00	-0.172298E+02	-0.948360E+01	-0.785433E+00	0.128454E+02	-0.947338E+01
2	2	3	-0.248479E+00	0.128454E+02	-0.299697E+01	-0.248499E+00	-0.127953E+02	-0.299721E+01
3	3	4	-0.121280E+01	-0.127952E+02	-0.146280E+02	-0.121363E+01	0.171295E+02	-0.146380E+02

NODAL UNBALANCED FORCES

N	FU	FV	FR	FDU	FDV
1	0.0	0.0	0.0	0.0	0.527601E-01
2	0.245938E-03	-0.323671E-02	0.712589E-04	-0.391517E-02	0.127195E-01
3	0.557295E-04	-0.319005E-02	0.266600E-03	-0.374069E-02	0.127219E-01
4	0.0	0.0	0.0	0.0	0.519961E-01

NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.150452E-23	-0.310967E-22	-0.103676E-21	-0.0	0.426385E-08
2	0.946197E-05	-0.112599E-05	-0.478628E-07	-0.176629E-08	-0.306237E-08
3	0.939200E-05	-0.121974E-05	-0.460945E-07	-0.173432E-08	-0.388942E-08
4	0.151215E-23	-0.331709E-22	-0.103678E-21	-0.0	0.329648E-08

TOTAL NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.250777E-20	-0.789438E-20	-0.172402E-18	0.0	-0.286974E-05
2	0.157289E-01	-0.346007E-03	-0.797645E-04	-0.913219E-06	-0.287707E-05
3	0.156195E-01	-0.533532E-03	-0.788513E-04	-0.913187E-06	-0.443983E-05
4	0.249525E-20	-0.121699E-19	-0.171399E-18	0.0	-0.443264E-05

STRESS RESULTANTS

MEMBER NO.	NODI	NODJ	PI	MI	MI*	PJ	MJ	MJ*
1	1	2	-0.785116E+00	-0.172401E+02	-0.946952E+01	-0.786270E+00	0.128531E+02	-0.948348E+01
2	2	3	-0.248961E+00	0.128531E+02	-0.300278E+01	-0.248973E+00	-0.128028E+02	-0.300292E+01
3	3	4	-0.121386E+01	-0.128031E+02	-0.146408E+02	-0.121272E+01	0.171399E+02	-0.146271E+02

NODAL UNBALANCED FORCES

N	FU	FV	FR	FDU	FDV
1	0.0	0.0	0.0	0.0	0.201708E-07
2	0.346784E-09	-0.122844E-08	0.167652E-08	-0.144045E-08	0.505920E-08
3	0.273275E-09	-0.122277E-08	0.178565E-08	-0.124008E-08	0.544264E-08
4	0.0	0.0	0.0	0.0	0.199253E-07

ITERATION NO. 2 LOAD TOLERANCE = 0.000000 DISPL. TOLERANCE = 0.000606

****

NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.309797E-29	-0.987808E-29	-0.210673E-27	-0.0	0.239041E-14
2	0.189125E-10	-0.343874E-12	-0.896785E-13	-0.647588E-15	-0.388181E-15
3	0.188863E-10	-0.551944E-12	-0.887954E-13	-0.610902E-15	-0.204361E-14
4	0.310262E-29	-0.146340E-28	-0.210778E-27	-0.0	0.619788E-15

TOTAL NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.250777E-20	-0.789438E-20	-0.172402E-18	0.0	-0.286974E-05
2	0.157289E-01	-0.346007E-03	-0.797645E-04	-0.913219E-06	-0.287707E-05
3	0.156195E-01	-0.533532E-03	-0.788513E-04	-0.913187E-06	-0.443983E-05
4	0.249525E-20	-0.121699E-19	-0.171399E-18	0.0	-0.443264E-05

STRESS RESULTANTS

MEMBER NO.	NODI	NODJ	PI	MI	MI*	PJ	MJ	MJ*
1	1	2	-0.785116E+00	-0.172401E+02	-0.946952E+01	-0.786270E+00	0.128531E+02	-0.948348E+01
2	2	3	-0.248961E+00	0.128531E+02	-0.300278E+01	-0.248973E+00	-0.128028E+02	-0.300292E+01
3	3	4	-0.121386E+01	-0.128031E+02	-0.146408E+02	-0.121272E+01	0.171399E+02	-0.146271E+02

NODAL UNBALANCED FORCES

N	FU	FV	FR	FDU	FDV
1	0.0	0.0	0.0	0.0	-0.888178E-15
2	0.162370E-14	0.416334E-16	0.444089E-15	0.760503E-14	0.0
3	-0.163758E-14	-0.444089E-15	0.111022E-14	0.164313E-13	-0.333067E-14
4	0.0	0.0	0.0	0.0	0.355271E-14

ITERATION NO. 3 LOAD TOLERANCE = 0.000000 DISPL. TOLERANCE = 0.000000
****

F.3 Incremental Elastic-Plastic Analysis

```
3-ELASTIC PLASTIC ANAL. (INCR)
4,3,2,1,1,5,12,
1,0,0,
2,0,10,
3,10,10,
4,10,0,
9.12,110,30.4,36,30000,
1,1,2,0,1,0,0,0,
0,0,0,0,0,
2,2,3,0,1,0,0,0,
0,0,0,0,0,
3,3,4,0,1,0,0,0,
0,0,0,0,0,
0,
1,1,111,
2,4,111,
1,0,
2,0,
0,1,.5,-1,0,
3,0,
0,1,0,-1,0,
0,
```

```
                    3-ELASTIC PLASTIC ANAL. (INCR)

                    ******************************


PROBLEM CONTROL VARIABLES
*************************

     NUMBER OF JOINTS                         =    4

     NUMBER OF ELEMENTS                       =    3

     NO. OF EXTERNAL BOUNDARY ELEMENTS        =    2

     NUMBER OF LOAD CASES                     =    1

     IDENTIF. OF TYPE OF ANALYSIS FLAG        =    1

     MAXIMUM NO. OF HINGES ESTIMATED          =    5

     LENGTH CONVERSION FACTOR                 =   12.000000

     **************************************************


NODAL GEOMETRY DATA AS INPUT
****************************

     N        X               Y             INC

     1   0.0             0.0                 0
     2   0.0             0.100000E+02        0
     3   0.100000E+02    0.100000E+02        0
     4   0.100000E+02    0.0                 0
```

MEMBER PROPERTIES DEFAULT VALUES

AREA	=	0.912000E+01
M. INERTIA	=	0.110000E+03
PLASTIC SECTION MODULUS	=	0.304000E+02
YIELD STRESS	=	0.360000E+02
Y. MODULUS	=	0.300000E+05

MEMBER DATA AS INPUT

M	I	J	CODE	FLAG	INC	INCI	INCJ	AREA	I	PSM	YS	YMOD
1	1	2	0	1	0	0	0	0.9120E+01	0.1100E+03	0.3040E+02	0.3600E+02	0.3000E+05
2	2	3	0	1	0	0	0	0.9120E+01	0.1100E+03	0.3040E+02	0.3600E+02	0.3000E+05
3	3	4	0	1	0	0	0	0.9120E+01	0.1100E+03	0.3040E+02	0.3600E+02	0.3000E+05

EXTERNAL BOUNDARY ELEMENTS DATA

N	NPB	KODE	EB	N	NPB	KODE	EB	N	NPB	KODE	EB
1	1	111	0.100000E+21	2	4	111	0.100000E+21				

JOINT LOADS DATA AS INPUT

N	NBU	NBV	NBR	INC	CODE	LVF	U	V	R
2	0	0	0	0	0	1	0.500000E+00	-0.100000E+01	0.0
3	0	0	0	0	0	1	0.0	-0.100000E+01	0.0

TOTAL EXTERNAL NODAL LOADS

N	FU	FV	FR
1	0.0	0.0	0.0
2	0.301169E+02	-0.602339E+02	0.0
3	0.0	-0.602339E+02	0.0
4	0.0	0.0	0.0

TOTAL NODAL DISPLACEMENTS

N	U	V	R
1	0.155748E-18	-0.476578E-18	-0.109440E-16
2	0.903292E+00	-0.209025E-01	-0.487427E-02
3	0.896914E+00	-0.319342E-01	-0.445609E-02
4	0.145421E-18	-0.728100E-18	-0.988172E-17

TOTAL MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.476577E+02	0.155516E+02	0.912000E+02	-0.476577E+02	-0.155516E+02	0.638405E+02
2	2	3	0.145421E+02	-0.125763E+02	-0.638405E+02	-0.145421E+02	0.125763E+02	-0.619239E+02
3	3	4	0.728100E+02	0.145001E+02	0.619239E+02	-0.728100E+02	-0.145001E+02	0.823476E+02

****
HINGE NO. 1 HAS BEEN FORMED
****

TOTAL EXTERNAL NODAL LOADS

N	FU	FV	FR
1	0.0	0.0	0.0
2	0.317589E+02	-0.635178E+02	0.0
3	0.0	-0.635178E+02	0.0
4	0.0	0.0	0.0

TOTAL NODAL DISPLACEMENTS

N	U	V	R
1	0.159193E-18	-0.499459E-18	-0.109440E-16
2	0.998464E+00	-0.219061E-01	-0.506343E-02
3	0.991517E+00	-0.338113E-01	-0.500553E-02
4	0.158397E-18	-0.770898E-18	-0.108803E-16

TOTAL MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.499459E+02	0.159193E+02	0.912000E+02	-0.499459E+02	-0.159193E+02	0.679926E+02
2	2	3	0.158397E+02	-0.135720E+02	-0.679926E+02	-0.158397E+02	0.135720E+02	-0.677273E+02
3	3	4	0.770898E+02	0.158397E+02	0.677273E+02	-0.770898E+02	-0.158397E+02	0.906693E+02

****
HINGE NO. 2 HAS BEEN FORMED
****

TOTAL EXTERNAL NODAL LOADS

N	FU	FV	FR
1	0.0	0.0	0.0
2	0.336633E+02	-0.673266E+02	0.0
3	0.0	-0.673266E+02	0.0
4	0.0	0.0	0.0

TOTAL NODAL DISPLACEMENTS

N	U	V	R
1	0.174587E-18	-0.503827E-18	-0.109440E-16
2	0.147917E+01	-0.220977E-01	-0.649601E-02
3	0.147206E+01	-0.369607E-01	-0.607419E-02
4	0.162046E-18	-0.842705E-18	-0.988172E-17

TOTAL MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.503827E+02	0.174615E+02	0.912000E+02	-0.503827E+02	-0.174615E+02	0.856838E+02
2	2	3	0.162046E+02	-0.169438E+02	-0.856838E+02	-0.162046E+02	0.169438E+02	-0.837504E+02
3	3	4	0.842705E+02	0.162640E+02	0.837504E+02	-0.842705E+02	-0.162640E+02	0.823476E+02

****
HINGE NO. 3 HAS BEEN FORMED
****

TOTAL EXTERNAL NODAL LOADS

N	FU	FV	FR
1	0.0	0.0	0.0
2	0.339338E+02	-0.678676E+02	0.0
3	0.0	-0.678676E+02	0.0
4	0.0	0.0	0.0

TOTAL NODAL DISPLACEMENTS

N	U	V	R
1	0.179045E-18	-0.503819E-18	-0.109440E-16
2	0.166877E+01	-0.220973E-01	-0.728802E-02
3	0.166174E+01	-0.374356E-01	-0.568413E-02
4	0.160293E-18	-0.853533E-18	-0.988172E-17

TOTAL MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.503819E+02	0.179077E+02	0.912000E+02	-0.503819E+02	-0.179077E+02	0.911016E+02
2	2	3	0.160293E+02	-0.174857E+02	-0.911016E+02	-0.160293E+02	0.174857E+02	-0.837504E+02
3	3	4	0.853533E+02	0.161041E+02	0.837504E+02	-0.853533E+02	-0.161041E+02	0.823476E+02

****
HINGE NO. 4 HAS BEEN FORMED
****

F.4 Iterative Elastic-Plastic Analysis

```
3-ELASTIC PLASTIC ANAL. (ITRV)
4,3,2,1,1,5,12,
1,0,0,
2,0,10,
3,10,10,
4,10,0,
9.12,110,30.4,36,30000,
1,1,2,0,1,0,0,0,
0,0,0,0,0,
2,2,3,0,1,0,0,0,
0,0,0,0,0,
3,3,4,0,1,0,0,0,
0,0,0,0,0,
0,
1,1,111,
2,4,111,
1,0,
2,0,
0,1,.5,-1,0,
3,0,
0,1,0,-1,0,
0,
```

```
3-ELASTIC PLASTIC ANAL. (ITRV)

******************************


PROBLEM CONTROL VARIABLES
*************************

     NUMBER OF JOINTS                       =    4

     NUMBER OF ELEMENTS                     =    3

     NO. OF EXTERNAL BOUNDARY ELEMENTS      =    2

     NUMBER OF LOAD CASES                   =    1

     IDENTIF. OF TYPE OF ANALYSIS FLAG      =    1

     MAXIMUM NO. OF HINGES ESTIMATED        =    5

     LENGTH CONVERSION FACTOR               =   12.000000

     ***********************************************


NODAL GEOMETRY DATA AS INPUT
****************************

     N         X                 Y             INC

     1    0.0              0.0                  0
     2    0.0              0.100000E+02         0
     3    0.100000E+02     0.100000E+02         0
     4    0.100000E+02     0.0                  0
```

MEMBER PROPERTIES DEFAULT VALUES
****

AREA	=	0.912000E+01
M. INERTIA	=	0.110000E+03
PLASTIC SECTION MODULUS	=	0.304000E+02
YIELD STRESS	=	0.360000E+02
Y. MODULUS	=	0.300000E+05

MEMBER DATA AS INPUT
****

M	I	J	CODE	FLAG	INC	INCI	INCJ	AREA	I	PSM	YS	YMOD
1	1	2	0	1	0	0	0	0.9120E+01	0.1100E+03	0.3040E+02	0.3600E+02	0.3000E+05
2	2	3	0	1	0	0	0	0.9120E+01	0.1100E+03	0.3040E+02	0.3600E+02	0.3000E+05
3	3	4	0	1	0	0	0	0.9120E+01	0.1100E+03	0.3040E+02	0.3600E+02	0.3000E+05

EXTERNAL BOUNDARY ELEMENTS DATA
****

N	NPB	KODE	EB	N	NPB	KODE	EB	N	NPB	KODE	EB
1	1	111	0.100000E+21	2	4	111	0.100000E+21				

JOINT LOADS DATA AS INPUT
****

N	NBU	NBV	NBR	INC	CODE	LVF	U	V	R
2	0	0	0	0	0	1	0.500000E+00	-0.100000E+01	0.0
3	0	0	0	0	0	1	0.0	-0.100000E+01	0.0

TOTAL EXTERNAL NODAL LOADS

N	FU	FV	FR
1	0.0	0.0	0.0
2	0.294212E+02	-0.588425E+02	0.0
3	0.0	-0.588425E+02	0.0
4	0.0	0.0	0.0

TOTAL NODAL DISPLACEMENTS

N	U	V	R
1	0.152381E-18	-0.460516E-18	-0.109440E-16
2	0.923487E+00	-0.201981E-01	-0.491389E-02
3	0.917266E+00	-0.314181E-01	-0.457457E-02
4	0.141831E-18	-0.716333E-18	-0.100964E-16

TOTAL MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.460516E+02	0.151672E+02	0.912000E+02	-0.460516E+02	-0.151672E+02	0.647251E+02
2	2	3	0.141831E+02	-0.127911E+02	-0.647251E+02	-0.141831E+02	0.127911E+02	-0.631699E+02
3	3	4	0.716333E+02	0.140736E+02	0.631699E+02	-0.716333E+02	-0.140736E+02	0.841367E+02

****
HINGE NO. 1 HAS BEEN FORMED
****

TOTAL EXTERNAL NODAL LOADS

N	FU	FV	FR
1	0.0	0.0	0.0
2	0.307435E+02	-0.614871E+02	0.0
3	0.0	-0.614871E+02	0.0
4	0.0	0.0	0.0

TOTAL NODAL DISPLACEMENTS

N	U	V	R
1	0.155209E-18	-0.479121E-18	-0.109440E-16
2	0.998442E+00	-0.210141E-01	-0.506288E-02
3	0.991766E+00	-0.329220E-01	-0.500724E-02
4	0.152227E-18	-0.750621E-18	-0.108828E-16

TOTAL MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.479121E+02	0.154412E+02	0.912000E+02	-0.479121E+02	-0.154412E+02	0.679951E+02
2	2	3	0.152227E+02	-0.135753E+02	-0.679951E+02	-0.152227E+02	0.135753E+02	-0.677401E+02
3	3	4	0.750621E+02	0.150986E+02	0.677401E+02	-0.750621E+02	-0.150986E+02	0.906900E+02

****
HINGE NO. 2 HAS BEEN FORMED
****

TOTAL EXTERNAL NODAL LOADS

N	FU	FV	FR
1	0.0	0.0	0.0
2	0.321623E+02	-0.643246E+02	0.0
3	0.0	-0.643246E+02	0.0
4	0.0	0.0	0.0

TOTAL NODAL DISPLACEMENTS

N	U	V	R
1	0.168846E-18	-0.478782E-18	-0.109440E-16
2	0.141849E+01	-0.209992E-01	-0.633630E-02
3	0.141179E+01	-0.354259E-01	-0.586381E-02
4	0.152777E-18	-0.807710E-18	-0.973681E-17

TOTAL MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.478782E+02	0.167714E+02	0.912000E+02	-0.478782E+02	-0.167714E+02	0.833055E+02
2	2	3	0.152777E+02	-0.164467E+02	-0.833055E+02	-0 152777E+02	0.164467E+02	-0.811399E+02
3	3	4	0.807709E+02	0.150877E+02	0.811399E+02	-0.807709E+02	-0.150877E+02	0.811401E+02

****
HINGE NO: 3 HAS BEEN FORMED
****

TOTAL EXTERNAL NODAL LOADS

N	FU	FV	FR
1	0.0	0.0	0.0
2	0.325334E+02	-0.650668E+02	0.0
3	0.0	-0.650668E+02	0.0
4	0.0	0.0	0.0

TOTAL NODAL DISPLACEMENTS

N	U	V	R
1	0.175622E-18	-0.478793E-18	-0.109440E-16
2	0.169895E+01	-0.209997E-01	-0.752520E-02
3	0.169238E+01	-0.360764E-01	-0.522470E-02
4	0.149712E-18	-0.822543E-18	-0.967873E-17

TOTAL MEMBER END FORCES

MEM	I	J	NI	VI	MI	NJ	VJ	MJ
1	1	2	0.478793E+02	0.174266E+02	0.912000E+02	-0.478793E+02	-0.174266E+02	0.912000E+02
2	2	3	0.149712E+02	-0.171879E+02	-0.912000E+02	-0.149712E+02	0.171879E+02	-0.806561E+02
3	3	4	0.822542E+02	0.147392E+02	0.806561E+02	-0.822542E+02	-0.147392E+02	0.806561E+02

****
HINGE NO.   4 HAS BEEN FORMED
****

F.5 Inelastic Analysis

```
5- INELASTIC ANALYSIS
0,0,0,1,1,
4,3,3,2,1,10,.00001,12,
1,1,1,
1,0,0,
2,0,10,
3,10,10,
4,10,0,
1,8,8,.433,.288,
1,.0012,.012,.12,36,36,36,
1,0,0,0,0,0,
1,1,2,1,1,1,
2,2,3,1,1,1,
3,3,4,1,1,1,
0,
1,1,11110,
2,4,11110,
1,0,
2,0,
0,1,.5,-1,0,
3,0,
0,1,0,-1,0,
0,
```

5- INELASTIC ANALYSIS

****

PROBLEM CONTROL VARIABLES

****

NUMBER OF JOINTS	=	4
NUMBER OF FLEXURAL ELEMENTS	=	3
NUMBER OF ELEMENTS	=	3
NO. OF EXTERNAL BOUNDARY ELEMENTS	=	2
NUMBER OF DIFFERENT MEMBER TYPES	=	1
NUMBER OF DIFFERENT X-SEC TYPES	=	1
NUMBER OF DIFFERENT MATERIAL TYPES	=	1
NUMBER OF DIFFERENT RESIDUAL TYPES	=	1
MAXIMUM NO. OF ITERATIONS	=	10
TOLERANCE LIMIT	=	0.000010
LENGTH CONVERGENCE FACTOR	=	12.000000

****

NODAL GEOMETRY DATA AS INPUT
****

N	X	Y	INC
1	0.0	0.0	0
2	0.0	0.100000E+02	0
3	0.100000E+02	0.100000E+02	0
4	0.100000E+02	0.0	0

DIFFERENT TYPES OF X-SEC PROPERTIES
****

TYPE	H	W	TF	TW
1	0.800000E+01	0.800000E+01	0.433000E+00	0.288000E+00

DIFFERENT MATERIAL PROPERTIES
****

TYPE	EY	EP	EULT	YS	YPS	ULTS
1	0.120000E-02	0.120000E-01	0.120000E+00	0.360000E+02	0.360000E+02	0.360000E+02

DIFFERENT TYPES OF RESIDUAL STRAINS
****

TYPE	EPR1	EPR2	EPR3	EPR4	EPR5
1	0.0	0.0	0.0	0.0	0.0

MEMBER DATA AS INPUT
****

M	I	J	XST	MT	RT	INC	INCI	INCJ
1	1	2	1	1	1	0	0	0
2	2	3	1	1	1	0	0	0
3	3	4	1	1	1	0	0	0

EXTERNAL BOUNDARY ELEMENTS DATA

N	NPB	KODE	EB	N	NPB	KODE	EB	N	NPB	KODE	EB
1	1	11110	0.100000E+21	2	4	11110	0.100000E+21				

JOINT LOADS DATA AS INPUT

N	NBU	NBV	NBR	INC	CODE	LVF	U	V	R
2	0	0	0	0	0	1	0.500000E+00	-0.100000E+01	0.0
3	0	0	0	0	0	1	0.0	-0.100000E+01	0.0

STRESS RESULTANTS

MEM	NODI	NODJ	PI	MI	MI*	PJ	MJ	MJ*
1	1	2	0.0	0.0	0.0	0.0	0.0	0.0
2	2	3	0.0	0.0	0.0	0.0	0.0	0.0
3	3	4	0.0	0.0	0.0	0.0	0.0	0.0

NODAL UNBALANCED FORCES

N	FU	FV	FR	FDU	FDV
1	0.0	0.0	0.0	0.0	0.0
2	0.500000E+00	-0.100000E+01	0.0	0.0	0.0
3	0.0	-0.100000E+01	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0

NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.250626E-20	-0.786327E-20	-0.172297E-18	-0.0	-0.287801E-05
2	0.157650E-01	-0.345361E-03	-0.799461E-04	-0.912728E-06	-0.287801E-05
3	0.156555E-01	-0.533055E-03	-0.790334E-04	-0.912728E-06	-0.444213E-05
4	0.249374E-20	-0.121367E-19	-0.171296E-18	-0.0	-0.444213E-05

STRESS RESULTANTS

MEM	NODI	NODJ	PI	MI	MI*	PJ	MJ	MJ*
1	1	2	-0.786278E+00	-0.170040E+02	-0.945050E+01	-0.785356E+00	0.126241E+02	-0.943945E+01
2	2	3	-0.248545E+00	0.126549E+02	-0.298732E+01	-0.248564E+00	-0.126049E+02	-0.298755E+01
3	3	4	-0.121272E+01	-0.125751E+02	-0.145761E+02	-0.121362E+01	0.169050E+02	-0.145869E+02

NODAL UNBALANCED FORCES

N	FU	FV	FR	FDU	FDV
1	0.0	0.0	0.0	0.0	0.410772E-02
2	0.541220E-03	-0.297032E-02	0.728174E-04	-0.106263E-02	0.261840E-02
3	0.254964E-03	-0.250747E-02	0.268540E-03	-0.103505E-02	0.259436E-02
4	0.0	0.0	0.0	0.0	0.405112E-02

NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.397746E-23	-0.263131E-22	-0.273284E-21	-0.0	0.556644E-09
2	0.249517E-04	-0.113483E-05	-0.125319E-06	-0.317428E-08	-0.312388E-08
3	0.248808E-04	-0.122960E-05	-0.123535E-06	-0.310610E-08	-0.397117E-08
4	0.398438E-23	-0.284647E-22	-0.273271E-21	-0.0	-0.371140E-09

TOTAL NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.251023E-20	-0.788958E-20	-0.172570E-18	0.0	-0.287746E-05
2	0.157900E-01	-0.346496E-03	-0.800715E-04	-0.915902E-06	-0.288114E-05
3	0.156804E-01	-0.534285E-03	-0.791570E-04	-0.915834E-06	-0.444610E-05
4	0.249773E-20	-0.121652E-19	-0.171569E-18	0.0	-0.444250E-05

STRESS RESULTANTS

MEM	NODI	NODJ	PI	MI	MI*	PJ	MJ	MJ*
1	1	2	-0.786211E+00	-0.170296E+02	-0.944971E+01	-0.786266E+00	0.126440E+02	-0.945038E+01
2	2	3	-0.249366E+00	0.126747E+02	-0.299719E+01	-0.249367E+00	-0.126246E+02	-0.299720E+01
3	3	4	-0.121386E+01	-0.125950E+02	-0.145898E+02	-0.121381E+01	0.169299E+02	-0.145892E+02

NODAL UNBALANCED FORCES

N	FU	FV	FR	FDU	FDV
1	0.0	0.0	0.0	0.0	0.385762E-08
2	0.395505E-06	-0.343238E-06	0.772076E-07	-0.617849E-08	0.393601E-08
3	0.393195E-06	0.325926E-06	0.270912E-06	-0.604889E-08	0.402542E-08
4	0.0	0.0	0.0	0.0	0.417716E-08

****
ITERATION NO. 2 LOAD TOLERANCE = 0.000001 DISPL. TOLERANCE = 0.001585
****

NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.393599E-26	-0.751975E-28	-0.269985E-24	-0.0	-0.175641E-13
2	0.246159E-07	-0.697783E-11	-0.123040E-09	-0.324118E-13	-0.287078E-13
3	0.246148E-07	-0.794945E-11	-0.121851E-09	-0.318993E-13	-0.365861E-13
4	0.395101E-26	-0.979239E-28	-0.270607E-24	-0.0	-0.250981E-13

TOTAL NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.251024E-20	-0.788958E-20	-0.172570E-18	0.0	-0.287746E-05
2	0.157900E-01	-0.346496E-03	-0.800716E-04	-0.915902E-06	-0.288114E-05
3	0.156804E-01	-0.534285E-03	-0.791571E-04	-0.915834E-06	-0.444610E-05
4	0.249773E-20	-0.121652E-19	-0.171569E-18	0.0	-0.444250E-05

STRESS RESULTANTS

MEM	NODI	NODJ	PI	MI	MI*	PJ	MJ	MJ*
1	1	2	-0.786211E+00	-0.170296E+02	-0.944971E+01	-0.786266E+00	0.126440E+02	-0.945038E+01
2	2	3	-0.249366E+00	0.126748E+02	-0.299719E+01	-0.249367E+00	-0.126247E+02	-0.299720E+01
3	3	4	-0.121386E+01	-0.125951E+02	-0.145898E+02	-0.121381E+01	0.169299E+02	-0.145892E+02

NODAL UNBALANCED FORCES

N	FU	FV	FR	FDU	FDV
1	0.0	0.0	0.0	0.0	-0.653304E-11
2	0.389761E-09	-0.333010E-09	0.762488E-10	-0.357968E-11	-0.275600E-11
3	0.389074E-09	0.329637E-09	0.267671E-09	-0.350896E-11	-0.273098E-11
4	0.0	0.0	0.0	0.0	-0.624008E-11

```
**************************************************************************************
ITERATION NO.    3     LOAD TOLERANCE  =     0.000000     DISPL. TOLERANCE  =     0.000002
**************************************************************************************
```

NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.388673E-29	-0.149025E-31	-0.266604E-27	-0.0	-0.212697E-16
2	0.243074E-10	-0.434572E-14	-0.121494E-12	-0.228380E-16	-0.231592E-16
3	0.243067E-10	-0.449250E-14	-0.120322E-12	-0.224731E-16	-0.243206E-16
4	0.390162E-29	-0.188335E-31	-0.267222E-27	-0.0	-0.219907E-16

TOTAL NODAL DISPLACEMENTS

N	U	V	R	DU	DV
1	0.251024E-20	-0.788958E-20	-0.172570E-18	0.0	-0.287746E-05
2	0.157900E-01	-0.346496E-03	-0.800716E-04	-0.915902E-06	-0.288114E-05
3	0.156804E-01	-0.534285E-03	-0.791571E-04	-0.915834E-06	-0.444610E-05
4	0.249773E-20	-0.121652E-19	-0.171569E-18	0.0	-0.444250E-05

ILL = 1