AFATL-TR-77-16, Volume II

Residual-Strength Analysis of Primary Aircraft Structure Damaged Due to Projectile Impact and Penetration

Volume II User Guides for the Computer Program

LEVEL II

GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA, GEORGIA 4 30332

FEBRUARY 1977

FINAL REPORT FOR PERIOD NOVEMBER 1975-FEBRUARY 1977

Approved for public release; distribution unlimited

Air Force Armament Laboratory
AIR FORCE SYSTEMS COMMAND★UNITED STATES AIR FORCE★EGLIN AIR FORCE BASE, FLORIDA

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFATL-TR-77-16-Volume II-2	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): RESIDUAL-STRENGTH ANALYSIS OF PRIMARY AIRCRAFT STRUCTURE DAMAGED DUE TO PROJECTILE IMPACT AND PENETRATION. VOLUME II, USER GUIDES FOR THE COMPUTER PROGRAMS.	5. TYPE OF REPORT & PERIOD COVERED: Final Report November 1975 to February 1977
	6. PERFORMING ORG. REPORT NUMBER: GIT-E-23-616-Vol-2
7. AUTHOR(s): J. A. Aberson, J. M. Anderson, W. W. King	8. CONTRACT OR GRANT NUMBER(s): F08635-76-C-0136
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Georgia Institute of Technology, Atlanta, Georgia 30332	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: Project No. 2549; Task No. 01; Work Unit No. 018
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Armament Laboratory, Armament Development and Test Center, Eglin Air Force Base, Florida 32542	12. REPORT DATE: February 1977
	13. NUMBER OF PAGES: 211
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Available in DDC.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Projectile Impact and Penetration	Finite-Element Program, CRAKD
Residual-Strength Analysis	Finite-Element Simulations
Primary Aircraft Structure Damage	Lumped Mass
Loading-to-Failure Aluminum Channels	Consistent Mass

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

High-strength aluminum channels, damaged by projectile impacts, have been loaded to failure in pure bending. In cases where damage included cracks of significant lengths, failure loads have been compared with those predicted by linear fracture mechanics through a finite-element computer program (CRAKD) which contains special crack-tip stress-singularity elements.

CRAKD has been employed for the computation of time-dependent stress-intensity factors in several problems of elastodynamic fracture. The results

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

153 800

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. (CONCLUDED) demonstrate the importance of specimen inertia in impact tests designed for the determination of dynamic fracture toughness.

PREFACE

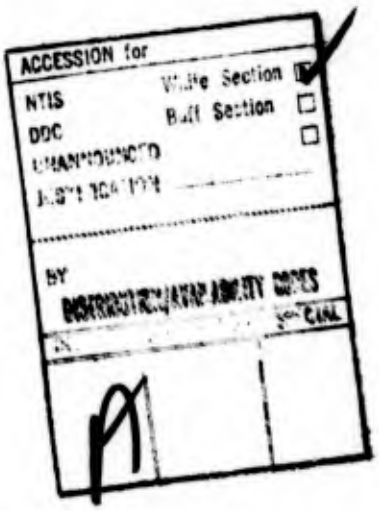
This report was prepared by the Georgia Institute of Technology, Atlanta, Georgia 30332, under Contract No. F08635-76-C-0136 with the Air Force Armament Laboratory, Armament Development and Test Center, Eglin Air Force Base, Florida 32542. Dr. Kevin T. McArdle (DLYV) managed the program for the Armament Laboratory. This effort was conducted during the period from November 1975 to February 1977.

This report consists of two volumes. Volume I contains the Test and Analysis, and Volume II contains User Guides for the Computer Program. This is Volume II.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER:

J. R. MURRAY
Chief, Analysis Division



(The reverse of this page is blank)

LIST OF FIGURES

LIST OF TABLES

SECTION I

COMPUTER PROGRAMS, USER GUIDES, AND SAMPLE PROBLEMS

In this section the computer programs CRAKD and GENER will be explained through a detailed description of their structure and the subprograms comprising them. CRAKD is a displacement-method finite-element analysis computer program which is capable of performing static, harmonic, and transient analyses of structures which contain cracks. Such analyses are possible because of the unique 8- and 10-node cracked elements contained in CRAKD. GENER is a utility program designed to generate the bulk of the input data required by CRAKD to establish and analyze a finite-element model.

Since, in the process of evolving a model and analyzing it, GENER is used first, it is described and illustrated in the following paragraphs. CRAKD is described and illustrated through two examples in paragraph 3.2 and subsequent paragraphs.

Also, in paragraph 3.3 the versions of the 8- and 10-node cracked elements for incorporation into NASTRAN at the Armament Laboratory are described, and the necessary subroutines to accomplish the insertion are listed.

1.1 GENER - A DATA GENERATOR FOR CRAKD

The effort to successfully construct a finite-element model of a cracked structure for subsequent analysis using CRAKD at Eglin AFB can be reduced by using the data generator program GENER.

GENER is an all-Fortran program intended to generate bulk data for CRAKD. Execution of GENER with the CDC Scope operating system produces a file, MODEL,

containing the bulk data for a finite element model in the format used by CRAKD. The bulk data which can be generated are (a) node numbers, degrees-of-freedom numbers, and node coordinates, (b) displacement restraint specifications, (c) triangular element numbers and node specifications, and (d) triangular element material and section properties. Once the bulk data is generated, selected editing of MODEL completes the establishment of a finite-element model which can then be input to CRAKD which performs static, harmonic, or transient analyses of linear-elastic cracked structures.

The structure of GENER is shown in Figure 1. A brief description of each of the subprograms comprising the program follows, and a complete listing of the program and all subprograms is contained in subparagraph 3.1.4.

1.1.1 DESCRIPTION OF THE SUBPROGRAMS

PROGRAM GENER (INPUT, OUTPUT, MODEL, TAPE5=INPUT, TAPE6=OUTPUT, TAPE10=MODEL)

GENER is the main program and calls the subprograms needed to generate bulk model data in a format compatible with the dynamic finite-element analysis program. The files INPUT, OUTPUT, and MODEL are used for Fortran read and write commands on units 5, 6, and 10, respectively. The generation data, which is explained in the User's Input Guide to GENER (paragraph 1.2), is stored on INPUT, the model bulk data generated will be written on MODEL, and optional user specified messages and data listings will be written on OUTPUT.

GENER calls subroutines CORGEN, RSTGEN, DOF, TRIGEN, and OUT. GENER also reads the element numbers, node specifications, material constants, and section properties for any 8- and 10-node cracked elements in the model.

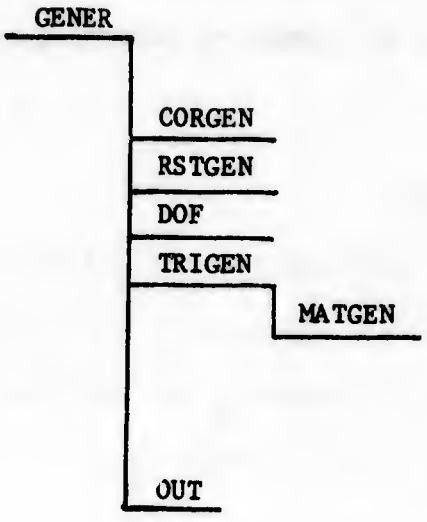


Figure 1. GENER Program Structure

SUBROUTINE CORGEN (COORD, IN)

This subprogram reads node generation data from file INPUT and generates node numbers, IN, and node coordinate, COORD. It will also, with the appropriate user specified options, print (a) the vector of node numbers, IN, as generated and (b) note the number of nodes and the largest node number generated.

SUBROUTINE RSTGEN (NF, IN)

The purpose of this subprogram is to read and generate model node restraint specifications, NF. Restraints can be imposed zero displacements (rigid supports) for static, harmonic, and transient analyses and imposed non-zero displacements for static analyses.

SUBROUTINE DOF (NF, IN, NDOFGN)

This subroutine is used to generate node freedom numbers, NF, in the order that the nodes are input (or generated) into vector IN. Freedom numbers are the row identifiers used to solve the model's nodal equations of motion. The stiffness [K] and mass [M] matrices and the nodal displacement vector {U} can be partitioned as:

$$\begin{bmatrix} K_{II} & K_{IR} \\ K_{RI} & K_{RR} \end{bmatrix}, \quad \begin{bmatrix} M_{II} & M_{IR} \\ M_{RI} & M_{RR} \end{bmatrix}$$

and

$$\begin{Bmatrix} U_I \\ U_R \end{Bmatrix}$$

where the U_I are the unknown independent degrees of freedom numbered 1 through NTDOF (total number of independent unknown displacements in the model) and the U_R are the restrained (known) node displacements (numbered ≤ 0).

SUBROUTINE TRIGEN (JT, IN)

This subroutine generates triangle element numbers and node specifications, JT. The vector IN is used in the generation sequence. User-specified options can cause this subprogram to print the sequence of element numbers as generated and stored in IN and/or print the number of triangles and largest element number generated.

SUBROUTINE MATGEN (JT, TPRP)

This subprogram generates the material constants E, ν, and ρ and the thickness t for each of the triangles elements generated by TRIGEN. The properties are stored in array TPRP.

SUBROUTINE OUT (COORD, NF, JT, TPRP, IN, IOPRT, IOUNIT)

This subroutine writes the arrays COORD and NF and JT and TPRP onto unit IOUNIT. IOUNIT is read by GENER and must be 10 to be consistent with the program statement in the main program where the file MODEL is synonymous with unit 10. The parameter IOPRT is used to allow the user to get a printed listing (on file OUTPUT) of MODEL if needed.

1.1.2 USER'S INPUT GUIDE TO GENER

To create a data file to execute GENER, the following sequence of cards must be input as listed. Several examples of data generator features for

node and element specifications are included to attempt to clarify the written instructions. A complete example is given in paragraph 3.1.3.

GENER - Input Data Sequence

Card	Description
(1)	TITLE
	FORMAT (8A10)
	Input any descriptive title (1 → 80 characters)
(2)	NDOFN, NRSTGN, NDOFGN, NTRIGN, NMATGN, NC8, NC10, NLUMPS, NLDNDS, ISTATE FORMAT (1015)

NDOFN =
- 0 - no nodes generated
- 2 - nodes and x,y coordinate generated (2-D model)
- 3 - nodes and x,y,z coordinate generated (3-D model)

NOTE: NDOFN used to generally restrain 3-D to 2-D

NRSTGN =
- 0 - no displacement restraints generated
- 1 - displacement restraints generated

NDOFGN =
- 0 - no node freedom numbers generated
- 2 - node freedom numbers generated in generated sequence
- 1 - node freedom numbers generated in input sequence

NTRIGN =
- 0 - no triangular elements generated
- 1 - triangular elements generated

NMATGN =
- 0 - no triangular element material properties generated
- 1 - triangular element material properties generated

NC8 = number of 8-node cracked elements input (not generated - but can be input in GENER data)

NC10 = number of 10-node cracked elements input (not generated - but can be input in GENER data)

NLUMPS = number of lumped masses to be input - edit MODEL

NLDNDS = number of loaded nodes for which loads are to be input - edit MODEL

ISTATE =
- 1 plane-stress element
- 0 plane-strain element

(3) 10 INPRT IOPRT 21 22

FORMAT (515)

INPRT = 0 - no option - see CRAKD input guide
1 - no option - see CRAKD input guide
2 - node and triangle numbers printed in order of generation

IOPRT = 0 - 1 - 2 - 3 - no options - see CRAKD input guide

(4) NTYPE

FORMAT (A10)

NTYPE = STATIC - static analysis
HCMASS - harmonic consistent mass analysis
HLMASS - harmonic lumped-mass analysis
TCMASS - transient consistent mass analysis
TLMASS - transient lumped-mass analysis

(5) NODE, X, Y, Z

FORMAT (15, 15X, 3E10.0)

NODE = number of node whose coordinate are X, Y, and Z (Input Z only if NDOFN = 3)

To generate nodes the following form of (5) is used

(5) - 1 NBUMP, NUMND, NDINC, DELTAX, DELTAY, DELTAZ

FORMAT (415, 3E 10.0)

- 1 indicates that this is a generator card

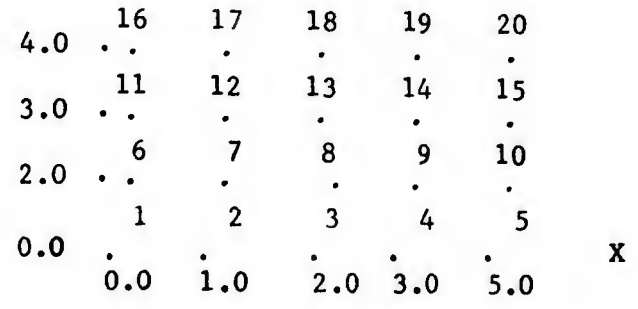
NBUMP = number of new node sets to be generated

NUMND = number of nodes in set to be generated

NDINC = number to be added to node numbers in the set (NUMND) to generate new nodes

DELTAX, DELTAY, DELTAZ = increment of X, Y and Z coordinates added to coordinates of nodes in the set to generated new node coordinates (Input DELTAZ only if NDOFN = 3)

Example:



The 20 nodes and their coordinates shown can be generated by either of the two following sequences:

Column →	5	10	15	20	25	35	45
	1				0.0	0.0	
	-1	3	1	1	1.0	0.0	
	5				5.0	0.0	
	-1	1	5	5	0.0	2.0	
	-1	2	5	5	0.0	1.0	

NOTE: the first card creates node 1; card 2 creates nodes 2,3,4; card 3 creates node 5, card 4 creates nodes 6,7,8,9,10; and card 5 creates nodes 11 → 30.

or

1				0.0	0.0
-1	1	1	5	0.0	2.0
-1	2	1	5	0.0	1.0
-1	3	4	1	1.0	0.0
-1	1	4	1	2.0	0.0

NOTE: card 1 creates node 1, card 2 creates node 6, card 3 creates nodes 11, 16, card 4 creates nodes 2,7,12,17,3,8,13,18,4,9,14,19; and card 5 creates nodes 5,10,15,20.

(6) 0 (zero)

FORMAT (15)

This card terminates node generation.

Input cards (5) and (6) only if NDOFN > 0.

(7) NODE, NRX, NRY, NRZ, NBUMP, NDINC

FORMAT (6I5)

NODE = node number at which displacement restraints are specified

NRX, NRY, NRZ =
- -1 - non-zero displacement is to be imposed
- 0 - no imposed displacement imposed
- 1 - zero displacement imposed

(Input NRZ only if NDOFN = 3)

NBUMP = number of other nodes which have the same displacement restraint specifications

NDINC = node number increment used to establish NBUMP nodes with same restraints, i.e., NODE, NODE + NDINC, NODE + 2 x 2 X NDINC, ... , NODE + NBUMP & NDINC all have same restraint specifications

(8) 0 (Zero)

FORMAT (I5)

this card terminates restraint specifications input

Input cards (7) and (8) only if NRSTGN > 0

(9) NODE, NBUMP, NDINC

FORMAT (3I5)

NODE = number of node

NBUMP = number of node number to be generated from NODE

NDINC = node number increment, i.e., NODE, NODE + NDINC, NODE + 2 X NDINC, ... , NODE + NBUMP * NDINC

This card(s) is used to generate a new node number sequence to control the bandwidth of the stiffness matrix and the mass matrix in CRAKD

(10) 0 (Zero)

FORMAT (I5)

This card terminates alternate node sequence.

Input cards (9) and (10) only if NDOFGN > 0

(11) NELT, JT1, JT2, JT3, NBUMP, NUMTRI, INC

FORMAT (7I5)

NELT = > 0 leave NBUMP, NUMTRI, INC, blank and JT1, JT2, and JT3 are the node numbers defining triangle number NELT

-1 NBUMP sets of NUMTRI triangular elements will be generated with element numbers incremented by INC and node numbers incremented by JT1, JT2, and JT3, respectively

NOTE: always define triangles in counter-clockwise sense. Node 1 → Node 2 establishes the local x-axis for the element.

Example: The pattern of 33 triangular elements shown can be generated as follows (Figure 2).

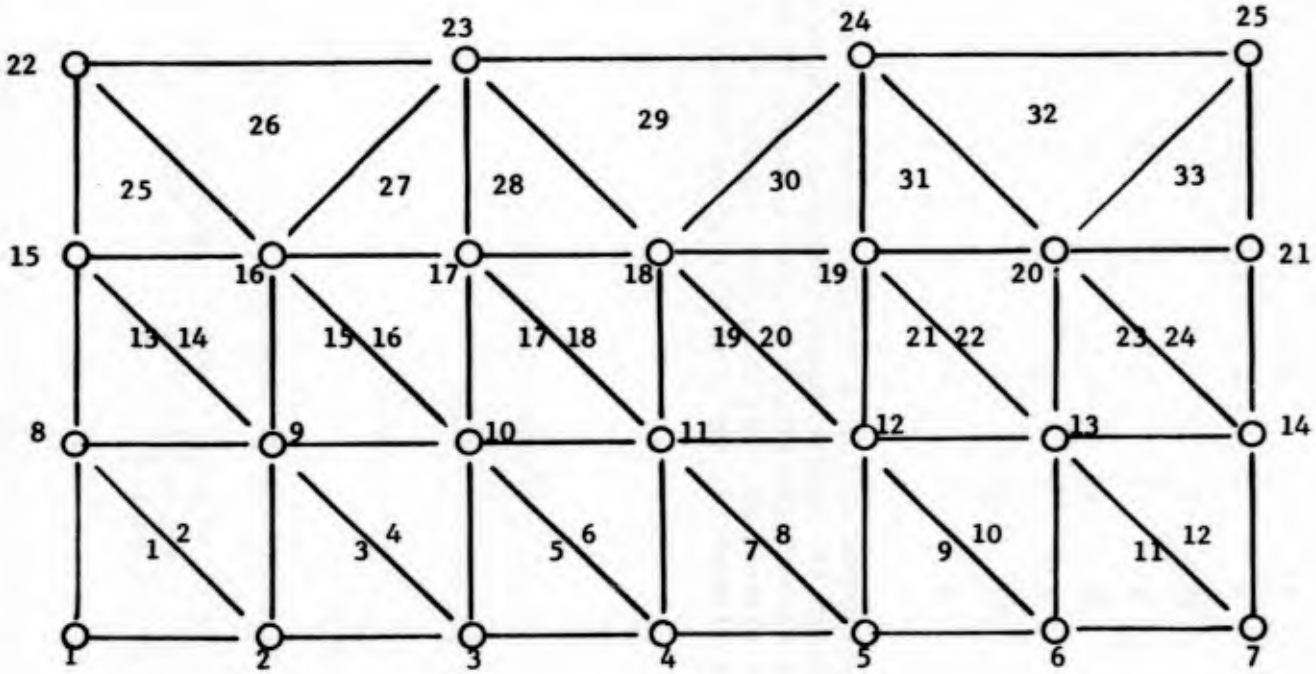


Figure 2. GENER Example

Card No.	Column No.						
	5	10	15	20	25	30	35
1	1	1	2	8			
2	2	9	8	2			
3	-1	1	1	1	5	2	2
4	-1	7	7	7	1	12	12
5	25	15	16	22			
6	-1	2	2	1	2	1	3
7	26	23	22	16			
8	-1	1	1	2	2	1	3
9	27	16	17	23			
10	-1	2	2	1	2	1	3

NOTE: card 1 creates element 1; card 2 creates element 2; card 3 creates elements 3 through 12; card 4 creates elements 13 through 24; card 5 creates element 25; card 6 creates elements 28 and 31; card 7 creates element 26; card 8 creates element 29 and 32; card 9 creates element 27; and card 10 creates elements 30 and 33.

(12) 0 (zero)

FORMAT (I5)

This card terminates triangular element generation.

Input cards (11) and (12) only if NTRIGN > 0

(13) NELT, NBUMP, E, ν, t, ρ

FORMAT (2I5, 4E 10.0)

NELT = number of elements which has properties E, ν, t, ρ

NBUMP = number of elements following which have the same properties, i.e., triangles NELT, NELT + 1, NELT + 2, .., NELT + NBUMP all have E, ν, t, ρ

(14) 0 (zero)

This card terminates material input generation for triangles.

Input cards (13) and (14) only if NMATGN > 0
(if NTRIGN > 0 then NMATGN > 0)

(15a) N8ELT, JC1, JC2, ... , JC8, ISTRES

FORMAT (10I5)

N8ELT = number of 8-node cracked elements

JC1→JC8 = nodes defining the 8-node cracked element as shown (Figure 3)

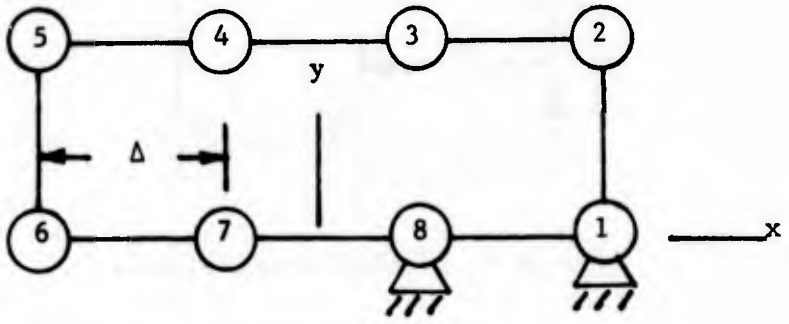


Figure 3. 8-Node Crack Element

ISTRES = 1 plane stress properties
0 plane strain properties

(15b) E, ν, t, ρ

FORMAT (4E 10.0)

Input (15a) and (15b) NC8 times

(16) 0 (zero)

This card terminates 8-node element input

Input (15a,b) and (16) only if NC8 > 0

(17a) N10ELT, JC1, JC2, . . . , JC10, ISTRES

FORMAT (1215)

N10ELT = number of 10-node cracked element

JC1→JC10 = nodes defining the 1-node cracked element as shown (Figure 4)

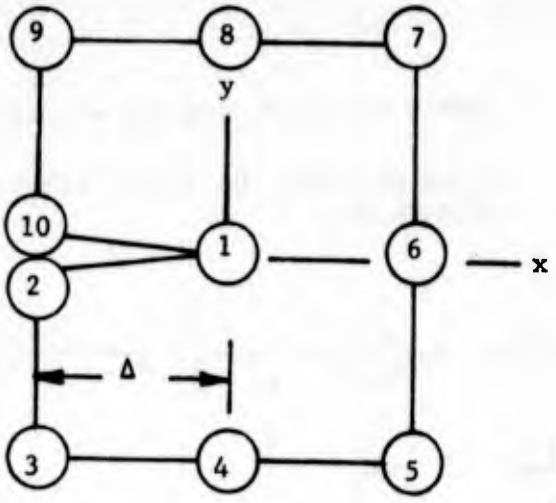


Figure 4. 10-Node Crack Element

ISTRES = 1 plane stress properties
0 plane strain properties

(17b) E, ν, t, ρ

FORMAT (4E 10.0)

Input (17a,b) NC10 times

(18) 0 (zero)

FORMAT (I5)

This card terminates 10-node element input

Input (17a,b) and (18) only if NC10 > 0

1.1.3 GENER - A SAMPLE PROBLEM

To illustrate how GENER establishes finite-element model bulk data for CRAKD, the problem shown on Figure 5 is used. This problem consists of 163 nodes, 273 triangle elements, and one 8-node cracked element. It has imposed zero displacements (rigid restraints at nodes 39 and 154 through 160). The cracked element is shown as ABCDE on the figure. This problem is one of the examples later analyzed and presented in paragraph 1.2.3. The data needed by GENER (a compiled version of GENER is permanently stored in file GENXQ at the Armament Laboratory to eliminate the unnecessary recompilation of GENER for each problem solution) is as given in Table 1. The output, i.e., file MODEL, resulting from input of these data to GENER (GENXQ) is listed in Table 2.

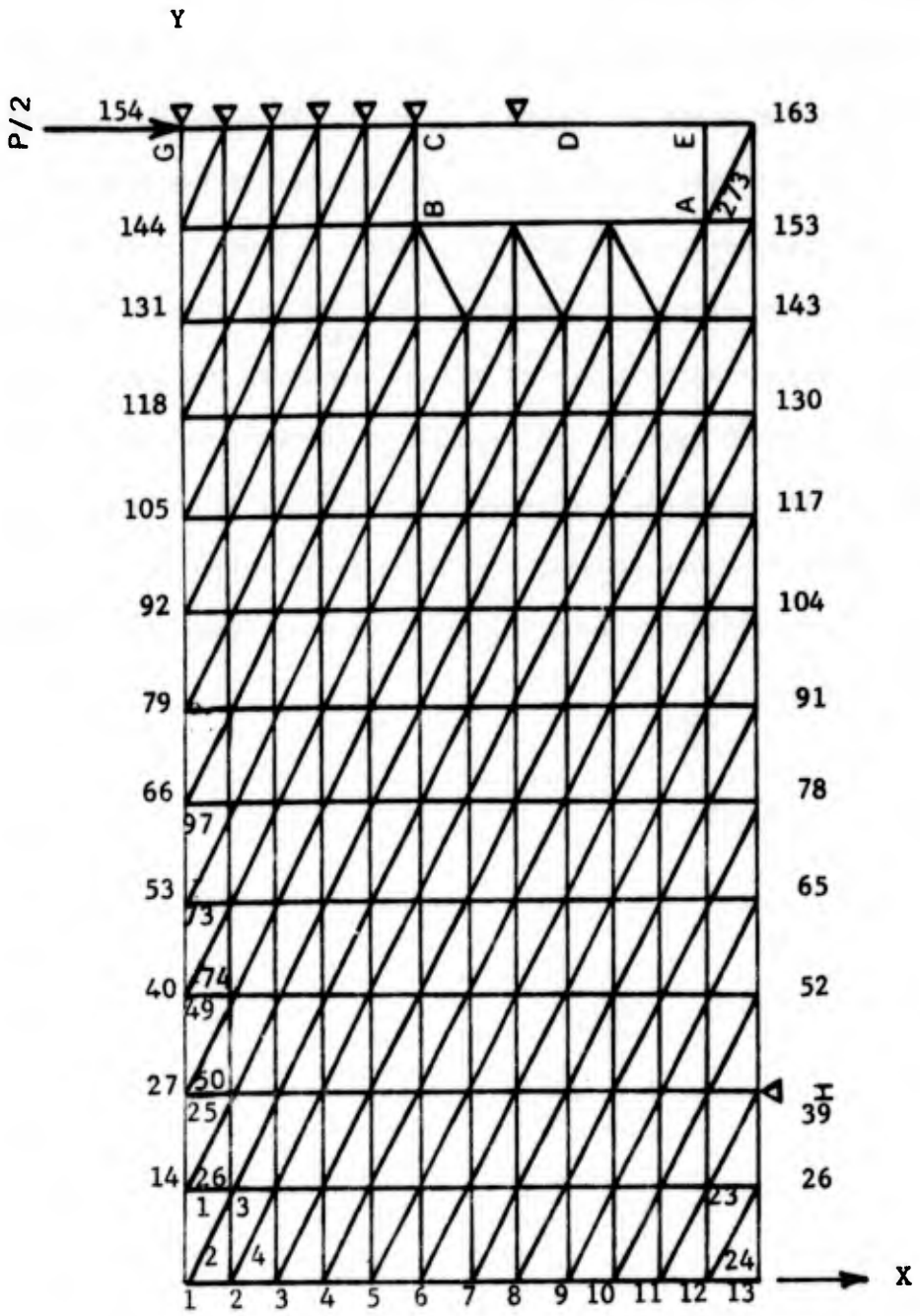


Figure 5. Finite Element Model of Irwin Dynamic-Tear Test Specimen (subparagraph 4.2, Vol. I)

TABLE 1. DATA FILE TO INPUT TO GENXQ TO GENERATE FILE MODEL FOR IRWIN PROBLEM

```
IRWIN MODEL WITH SINGLE POINT LOAD      A0 = 1.0 IN.  2/6/76    JAA
 2     1     0     1     1     1     1     0     0
10     1     3    21    22
TRANSIENT
 1
-1    12     1     1    1.0          0.0
-1    10    13    13                 2.0
144                                   22.0
-1     4     1     1    1.0
-1     3     1     1    2.0
-1     1     1     1    1.0
-1     1    10    10                 2.0
163                    12.0          24.0
 0                     0.25          0.25
 39    1
154     0     1     0     6     1
 0
 1    15    14     1
 2     1     2    15
-1     1     1     1    11     2     2
-1    13    13    13     9    24    24
241  145   144   131
242  131   132   145
-1     1     1     1     4     2     2
251  136   137   149
252  150   149   137
253  137   138   150
254  138   139   150
255  151   150   139
256  139   140   151
257  140   141   151
258  152   151   141
259  141   142   152
260  153   152   142
261  142   143   153
262  155   154   144
263  144   145   155
-1     1     1     1     4     2     2
-1     4     4     4     1     2     2
 0
 1   272    30.0E06       0.30          1.0          7.3E-04
 0
 1   159   149   150   151   152   162   161   160     1
30.0E06       0.30          1.0          7.3E-04
 0
```

TABLE 2. MODEL FOR IRWIN PROBLEM AS GENERATED BY GENXQ

```
IRWIN MODEL WITH SINGLE POINT LOAD      A0 = 1.0 IN.  2/6/76    JAA
163  273  0    1    0    0    0    8    2    1
  1                   3                   21   22
TRANSIENT
  1    0  0          0.0000    0.0000
  2    0  0           .2500    0.0000
  3    0  0           .5000    0.0000
  4    0  0           .7500    0.0000
  5    0  0          1.0000    0.0000
  6    0  0          1.2500    0.0000
  7    0  0          1.5000    0.0000
  8    0  0          1.7500    0.0000
  9    0  0          2.0000    0.0000
 10    0  0          2.2500    0.0000
 11    0  0          2.5000    0.0000
 12    0  0          2.7500    0.0000
 13    0  0          3.0000    0.0000
 14    0  0          0.0000     .5000
 15    0  0           .2500     .5000
 16    0  0           .5000     .5000
 17    0  0           .7500     .5000
 18    0  0          1.0000     .5000
 19    0  0          1.2500     .5000
 20    0  0          1.5000     .5000
 21    0  0          1.7500     .5000
 22    0  0          2.0000     .5000
 23    0  0          2.2500     .5000
 24    0  0          2.5000     .5000
 25    0  0          2.7500     .5000
 26    0  0          3.0000     .5000
 27    0  0          0.0000    1.0000
 28    0  0           .2500    1.0000
 29    0  0           .5000    1.0000
 30    0  0           .7500    1.0000
 31    0  0          1.0000    1.0000
 32    0  0          1.2500    1.0000
 33    0  0          1.5000    1.0000
 34    0  0          1.7500    1.0000
 35    0  0          2.0000    1.0000
 36    0  0          2.2500    1.0000
 37    0  0          2.5000    1.0000
 38    0  0          2.7500    1.0000
 39    1  0          3.0000    1.0000
 40    0  0          0.0000    1.5000
 41    0  0           .2500    1.5000
 42    0  0           .5000    1.5000
 43    0  0           .7500    1.5000
 44    0  0          1.0000    1.5000
 45    0  0          1.2500    1.5000
 46    0  0          1.5000    1.5000
 47    0  0          1.7500    1.5000
 48    0  0          2.0000    1.5000
 49    0  0          2.2500    1.5000
```

TABLE 2. MODEL FOR IRWIN PROBLEM AS GENERATED BY GENXQ (CONTINUED)

50	0	0	2.5000	1.5000
51	0	0	2.7500	1.5000
52	0	0	3.0000	1.5000
53	0	0	0.0000	2.0000
54	0	0	.2500	2.0000
55	0	0	.5000	2.0000
56	0	0	.7500	2.0000
57	0	0	1.0000	2.0000
58	0	0	1.2500	2.0000
59	0	0	1.5000	2.0000
60	0	0	1.7500	2.0000
61	0	0	2.0000	2.0000
62	0	0	2.2500	2.0000
63	0	0	2.5000	2.0000
64	0	0	2.7500	2.0000
65	0	0	3.0000	2.0000
66	0	0	0.0000	2.5000
67	0	0	.2500	2.5000
68	0	0	.5000	2.5000
69	0	0	.7500	2.5000
70	0	0	1.0000	2.5000
71	0	0	1.2500	2.5000
72	0	0	1.5000	2.5000
73	0	0	1.7500	2.5000
74	0	0	2.0000	2.5000
75	0	0	2.2500	2.5000
76	0	0	2.5000	2.5000
77	0	0	2.7500	2.5000
78	0	0	3.0000	2.5000
79	0	0	0.0000	3.0000
80	0	0	.2500	3.0000
81	0	0	.5000	3.0000
82	0	0	.7500	3.0000
83	0	0	1.0000	3.0000
84	0	0	1.2500	3.0000
85	0	0	1.5000	3.0000
86	0	0	1.7500	3.0000
87	0	0	2.0000	3.0000
88	0	0	2.2500	3.0000
89	0	0	2.5000	3.0000
90	0	0	2.7500	3.0000
91	0	0	3.0000	3.0000
92	0	0	0.0000	3.5000
93	0	0	.2500	3.5000
94	0	0	.5000	3.5000
95	0	0	.7500	3.5000
96	0	0	1.0000	3.5000
97	0	0	1.2500	3.5000
98	0	0	1.5000	3.5000
99	0	0	1.7500	3.5000
100	0	0	2.0000	3.5000
101	0	0	2.2500	3.5000
102	0	0	2.5000	3.5000

TABLE 2. MODEL FOR IRWIN PROBLEM AS GENERATED BY GENXQ (CONTINUED)

103	0	0	2.7500	3.5000
104	0	0	3.0000	3.5000
105	0	0	0.0000	4.0000
106	0	0	.2500	4.0000
107	0	0	.5000	4.0000
108	0	0	.7500	4.0000
109	0	0	1.0000	4.0000
110	0	0	1.2500	4.0000
111	0	0	1.5000	4.0000
112	0	0	1.7500	4.0000
113	0	0	2.0000	4.0000
114	0	0	2.2500	4.0000
115	0	0	2.5000	4.0000
116	0	0	2.7500	4.0000
117	0	0	3.0000	4.0000
118	0	0	0.0000	4.5000
119	0	0	.2500	4.5000
120	0	0	.5000	4.5000
121	0	0	.7500	4.5000
122	0	0	1.0000	4.5000
123	0	0	1.2500	4.5000
124	0	0	1.5000	4.5000
125	0	0	1.7500	4.5000

TABLE 2. MODEL FOR IRWIN PROBLEM AS GENERATED BY GENXQ (CONTINUED)

```
126    0    0          2.0000     4.5000
127    0    0          2.2500     4.5000
128    0    0          2.5000     4.5000
129    0    0          2.7500     4.5000
130    0    0          3.0000     4.5000
131    0    0          0.0000     5.0000
132    0    0           .2500     5.0000
133    0    0           .5000     5.0000
134    0    0           .7500     5.0000
135    0    0          1.0000     5.0000
136    0    0          1.2500     5.0000
137    0    0          1.5000     5.0000
138    0    0          1.7500     5.0000
139    0    0          2.0000     5.0000
140    0    0          2.2500     5.0000
141    0    0          2.5000     5.0000
142    0    0          2.7500     5.0000
143    0    0          3.0000     5.0000
144    0    0          0.0000     5.5000
145    0    0           .2500     5.5000
146    0    0           .5000     5.5000
147    0    0           .7500     5.5000
148    0    0          1.0000     5.5000
149    0    0          1.5000     5.5000
150    0    0          2.0000     5.5000
151    0    0          2.5000     5.5000
152    0    0          2.7500     5.5000
153    0    0          3.0000     5.5000
154    0    1          0.0000     6.0000
155    0    1           .2500     6.0000
156    0    1           .5000     6.0000
157    0    1           .7500     6.0000
158    0    1          1.0000     6.0000
159    0    1          1.5000     6.0000
160    0    1          2.0000     6.0000
161    0    0          2.5000     6.0000
162    0    0          2.7500     6.0000
163    0    0          3.0000     6.0000
  0
  1   15   14    1300000000.0     .30000   1.00000  .7300E-03
  2    1    2    1530000000.0     .30000   1.00000  .7300E-03
  3   16   15     230000000.0     .30000   1.00000  .7300E-03
  4    2    3    1630000000.0     .30000   1.00000  .7300E-03
  5   17   16     330000000.0     .30000   1.00000  .7300E-03
  6    3    4    1730000000.0     .30000   1.00000  .7300E-03
  7   18   17     430000000.0     .30000   1.00000  .7300E-03
  8    4    5    1830000000.0     .30000   1.00000  .7300E-03
  9   19   18     530000000.0     .30000   1.00000  .7300E-03
 10    5    6    1930000000.0     .30000   1.00000  .7300E-03
 11   20   19     630000000.0     .30000   1.00000  .7300E-03
 12    6    7    2030000000.0     .30000   1.00000  .7300E-03
 13   21   20     730000000.0     .30000   1.00000  .7300E-03
 14    7    8    2130000000.0     .30000   1.00000  .7300E-03
 15   22   21     830000000.0     .30000   1.00000  .7300E-03
 16    8    9    2230000000.0     .30000   1.00000  .7300E-03
 17   23   22     930000000.0     .30000   1.00000  .7300E-03
 18    9   10    2330000000.0     .30000   1.00000  .7300E-03
 19   24   23    1030000000.0     .30000   1.00000  .7300E-03
 20   10   11    2430000000.0     .30000   1.00000  .7300E-03
 21   25   24    1130000000.0     .30000   1.00000  .7300E-03
 22   11   12    2530000000.0     .30000   1.00000  .7300E-03
 23   26   25    1230000000.0     .30000   1.00000  .7300E-03
 24   12   13    2630000000.0     .30000   1.00000  .7300E-03
 25   28   27    1430000000.0     .30000   1.00000  .7300E-03
 26   14   15    2830000000.0     .30000   1.00000  .7300E-03
```

TABLE 2. MODEL FOR IRWIN PROBLEM AS GENERATED BY GENXQ (CONTINUED)

27	29	28	1530000000.0	.30000	1.00000	.7300E-03
28	15	16	2930000000.0	.30000	1.00000	.7300E-03
29	30	29	1630000000.0	.30000	1.00000	.7300E-03
30	16	17	3030000000.0	.30000	1.00000	.7300E-03
31	31	30	1730000000.0	.30000	1.00000	.7300E-03
32	17	18	3130000000.0	.30000	1.00000	.7300E-03
33	32	31	1830000000.0	.30000	1.00000	.7300E-03
34	18	19	3230000000.0	.30000	1.00000	.7300E-03
35	33	32	1930000000.0	.30000	1.00000	.7300E-03
36	19	20	3330000000.0	.30000	1.00000	.7300E-03
37	34	33	2030000000.0	.30000	1.00000	.7300E-03
38	20	21	3430000000.0	.30000	1.00000	.7300E-03
39	35	34	2130000000.0	.30000	1.00000	.7300E-03
40	21	22	3530000000.0	.30000	1.00000	.7300E-03
41	36	35	2230000000.0	.30000	1.00000	.7300E-03
42	22	23	3630000000.0	.30000	1.00000	.7300E-03
43	37	36	2330000000.0	.30000	1.00000	.7300E-03
44	23	24	3730000000.0	.30000	1.00000	.7300E-03
45	38	37	2430000000.0	.30000	1.00000	.7300E-03
46	24	25	3830000000.0	.30000	1.00000	.7300E-03
47	39	38	2530000000.0	.30000	1.00000	.7300E-03
48	25	26	3930000000.0	.30000	1.00000	.7300E-03
49	41	40	2730000000.0	.30000	1.00000	.7300E-03
50	27	28	4130000000.0	.30000	1.00000	.7300E-03
51	42	41	2830000000.0	.30000	1.00000	.7300E-03
52	28	29	4230000000.0	.30000	1.00000	.7300E-03
53	43	42	2930000000.0	.30000	1.00000	.7300E-03
54	29	30	4330000000.0	.30000	1.00000	.7300E-03
55	44	43	3030000000.0	.30000	1.00000	.7300E-03
56	30	31	4430000000.0	.30000	1.00000	.7300E-03
57	45	44	3130000000.0	.30000	1.00000	.7300E-03
58	31	32	4530000000.0	.30000	1.00000	.7300E-03
59	46	45	3230000000.0	.30000	1.00000	.7300E-03
60	32	33	4630000000.0	.30000	1.00000	.7300E-03
61	47	46	3330000000.0	.30000	1.00000	.7300E-03
62	33	34	4730000000.0	.30000	1.00000	.7300E-03
63	48	47	3430000000.0	.30000	1.00000	.7300E-03
64	34	35	4830000000.0	.30000	1.00000	.7300E-03
65	49	48	3530000000.0	.30000	1.00000	.7300E-03
66	35	36	4930000000.0	.30000	1.00000	.7300E-03
67	50	49	3630000000.0	.30000	1.00000	.7300E-03
68	36	37	5030000000.0	.30000	1.00000	.7300E-03
69	51	50	3730000000.0	.30000	1.00000	.7300E-03
70	37	38	5130000000.0	.30000	1.00000	.7300E-03
71	52	51	3830000000.0	.30000	1.00000	.7300E-03
72	38	39	5230000000.0	.30000	1.00000	.7300E-03
73	54	53	4030000000.0	.30000	1.00000	.7300E-03
74	40	41	5430000000.0	.30000	1.00000	.7300E-03
75	55	54	4130000000.0	.30000	1.00000	.7300E-03
76	41	42	5530000000.0	.30000	1.00000	.7300E-03
77	56	55	4230000000.0	.30000	1.00000	.7300E-03
78	42	43	5630000000.0	.30000	1.00000	.7300E-03
79	57	56	4330000000.0	.30000	1.00000	.7300E-03
80	43	44	5730000000.0	.30000	1.00000	.7300E-03
81	58	57	4430000000.0	.30000	1.00000	.7300E-03
82	44	45	5830000000.0	.30000	1.00000	.7300E-03
83	59	58	4530000000.0	.30000	1.00000	.7300E-03
84	45	46	5930000000.0	.30000	1.00000	.7300E-03
85	60	59	4630000000.0	.30000	1.00000	.7300E-03
86	46	47	6030000000.0	.30000	1.00000	.7300E-03
87	61	60	4730000000.0	.30000	1.00000	.7300E-03
88	47	48	6130000000.0	.30000	1.00000	.7300E-03
89	62	61	4830000000.0	.30000	1.00000	.7300E-03
90	48	49	6230000000.0	.30000	1.00000	.7300E-03
91	63	62	4930000000.0	.30000	1.00000	.7300E-03

TABLE 2. MODEL FOR IRWIN PROBLEM AS GENERATED BY GENXQ (CONTINUED)

92	49	50	6330000000.0	.30000	1.00000	.7300E-03
93	64	63	5030000000.0	.30000	1.00000	.7300E-03
94	50	51	6430000000.0	.30000	1.00000	.7300E-03
95	65	64	5130000000.0	.30000	1.00000	.7300E-03
96	51	52	6530000000.0	.30000	1.00000	.7300E-03
97	67	66	5330000000.0	.30000	1.00000	.7300E-03
98	53	54	6730000000.0	.30000	1.00000	.7300E-03
99	68	67	5430000000.0	.30000	1.00000	.7300E-03
100	54	55	6830000000.0	.30000	1.00000	.7300E-03
101	69	68	5530000000.0	.30000	1.00000	.7300E-03
102	55	56	6930000000.0	.30000	1.00000	.7300E-03
103	70	69	5630000000.0	.30000	1.00000	.7300E-03
104	56	57	7030000000.0	.30000	1.00000	.7300E-03
105	71	70	5730000000.0	.30000	1.00000	.7300E-03
106	57	58	7130000000.0	.30000	1.00000	.7300E-03
107	72	71	5830000000.0	.30000	1.00000	.7300E-03
108	58	59	7230000000.0	.30000	1.00000	.7300E-03
109	73	72	5930000000.0	.30000	1.00000	.7300E-03
110	59	60	7330000000.0	.30000	1.00000	.7300E-03
111	74	73	6030000000.0	.30000	1.00000	.7300E-03
112	60	61	7430000000.0	.30000	1.00000	.7300E-03
113	75	74	6130000000.0	.30000	1.00000	.7300E-03
114	61	62	7530000000.0	.30000	1.00000	.7300E-03
115	76	75	6230000000.0	.30000	1.00000	.7300E-03
116	62	63	7630000000.0	.30000	1.00000	.7300E-03
117	77	76	6330000000.0	.30000	1.00000	.7300E-03
118	63	64	7730000000.0	.30000	1.00000	.7300E-03
119	78	77	6430000000.0	.30000	1.00000	.7300E-03
120	64	65	7830000000.0	.30000	1.00000	.7300E-03
121	80	79	6630000000.0	.30000	1.00000	.7300E-03
122	66	67	8130000000.0	.30000	1.00000	.7300E-03
123	81	80	6730000000.0	.30000	1.00000	.7300E-03
124	67	68	8130000000.0	.30000	1.00000	.7300E-03
125	82	81	6830000000.0	.30000	1.00000	.7300E-03
126	68	69	8230000000.0	.30000	1.00000	.7300E-03
127	83	82	6930000000.0	.30000	1.00000	.7300E-03
128	69	70	8330000000.0	.30000	1.00000	.7300E-03
129	84	83	7030000000.0	.30000	1.00000	.7300E-03
130	70	71	8430000000.0	.30000	1.00000	.7300E-03
131	85	84	7130000000.0	.30000	1.00000	.7300E-03
132	71	72	8530000000.0	.30000	1.00000	.7300E-03
133	86	85	7230000000.0	.30000	1.00000	.7300E-03
134	72	73	8630000000.0	.30000	1.00000	.7300E-03
135	87	86	7330000000.0	.30000	1.00000	.7300E-03
136	73	74	8730000000.0	.30000	1.00000	.7300E-03
137	88	87	7430000000.0	.30000	1.00000	.7300E-03
138	74	75	8830000000.0	.30000	1.00000	.7300E-03
139	89	88	7530000000.0	.30000	1.00000	.7300E-03
140	75	76	8930000000.0	.30000	1.00000	.7300E-03
141	90	89	7630000000.0	.30000	1.00000	.7300E-03
142	76	77	9030000000.0	.30000	1.00000	.7300E-03
143	91	90	7730000000.0	.30000	1.00000	.7300E-03
144	77	78	9130000000.0	.30000	1.00000	.7300E-03
145	93	92	7930000000.0	.30000	1.00000	.7300E-03
146	79	80	9330000000.0	.30000	1.00000	.7300E-03
147	94	93	8030000000.0	.30000	1.00000	.7300E-03
148	80	81	9430000000.0	.30000	1.00000	.7300E-03
149	95	94	8130000000.0	.30000	1.00000	.7300E-03
150	81	82	9530000000.0	.30000	1.00000	.7300E-03
151	96	95	8230000000.0	.30000	1.00000	.7300E-03
152	82	83	9630000000.0	.30000	1.00000	.7300E-03
153	97	96	8330000000.0	.30000	1.00000	.7300E-03
154	83	84	9730000000.0	.30000	1.00000	.7300E-03
155	98	97	8430000000.0	.30000	1.00000	.7300E-03
156	84	85	9830000000.0	.30000	1.00000	.7300E-03

TABLE 2. MODEL FOR IRWIN PROBLEM AS GENERATED BY GENXQ (CONTINUED)

157	99	98	85300000000.0	.30000	1.00000	.7300E-03
158	85	86	99300000000.0	.30000	1.00000	.7300E-03
159	100	99	86300000000.0	.30000	1.00000	.7300E-03
160	86	87	10030000000.0	.30000	1.00000	.7300E-03
161	101	100	87300000000.0	.30000	1.00000	.7300E-03
162	87	88	10130000000.0	.30000	1.00000	.7300E-03
163	102	101	88300000000.0	.30000	1.00000	.7300E-03
164	88	89	10230000000.0	.30000	1.00000	.7300E-03
165	103	102	89300000000.0	.30000	1.00000	.7300E-03
166	89	90	10330000000.0	.30000	1.00000	.7300E-03
167	104	103	90300000000.0	.30000	1.00000	.7300E-03
168	90	91	10430000000.0	.30000	1.00000	.7300E-03
169	106	105	92300000000.0	.30000	1.00000	.7300E-03
170	92	93	10630000000.0	.30000	1.00000	.7300E-03
171	107	106	93300000000.0	.30000	1.00000	.7300E-03
172	93	94	10730000000.0	.30000	1.00000	.7300E-03
173	108	107	94300000000.0	.30000	1.00000	.7300E-03
174	94	95	10830000000.0	.30000	1.00000	.7300E-03
175	109	108	95300000000.0	.30000	1.00000	.7300E-03
176	95	96	10930000000.0	.30000	1.00000	.7300E-03
177	110	109	96300000000.0	.30000	1.00000	.7300E-03
178	96	97	11030000000.0	.30000	1.00000	.7300E-03
179	111	110	97300000000.0	.30000	1.00000	.7300E-03
180	97	98	11130000000.0	.30000	1.00000	.7300E-03
181	112	111	98300000000.0	.30000	1.00000	.7300E-03
182	98	99	11230000000.0	.30000	1.00000	.7300E-03
183	113	112	9930000000.0	.30000	1.00000	.7300E-03
184	99	100	11330000000.0	.30000	1.00000	.7300E-03
185	114	113	10030000000.0	.30000	1.00000	.7300E-03
186	100	101	11430000000.0	.30000	1.00000	.7300E-03
187	115	114	10130000000.0	.30000	1.00000	.7300E-03
188	101	102	11530000000.0	.30000	1.00000	.7300E-03
189	116	115	10230000000.0	.30000	1.00000	.7300E-03
190	102	103	11630000000.0	.30000	1.00000	.7300E-03
191	117	116	10330000000.0	.30000	1.00000	.7300E-03
192	103	104	11730000000.0	.30000	1.00000	.7300E-03
193	119	118	10530000000.0	.30000	1.00000	.7300E-03
194	105	106	11930000000.0	.30000	1.00000	.7300E-03
195	120	119	10630000000.0	.30000	1.00000	.7300E-03
196	106	107	12030000000.0	.30000	1.00000	.7300E-03
197	121	120	10730000000.0	.30000	1.00000	.7300E-03
198	107	108	12130000000.0	.30000	1.00000	.7300E-03
199	122	121	10830000000.0	.30000	1.00000	.7300E-03
200	108	109	12230000000.0	.30000	1.00000	.7300E-03
201	123	122	10930000000.0	.30000	1.00000	.7300E-03
202	109	110	12330000000.0	.30000	1.00000	.7300E-03
203	124	123	11030000000.0	.30000	1.00000	.7300E-03
204	110	111	12430000000.0	.30000	1.00000	.7300E-03
205	125	124	11130000000.0	.30000	1.00000	.7300E-03
206	111	112	12530000000.0	.30000	1.00000	.7300E-03
207	126	125	11230000000.0	.30000	1.00000	.7300E-03
208	112	113	12630000000.0	.30000	1.00000	.7300E-03
209	127	126	11330000000.0	.30000	1.00000	.7300E-03
210	113	114	12730000000.0	.30000	1.00000	.7300E-03
211	128	127	11430000000.0	.30000	1.00000	.7300E-03
212	114	115	12830000000.0	.30000	1.00000	.7300E-03
213	129	128	11530000000.0	.30000	1.00000	.7300E-03
214	115	116	12930000000.0	.30000	1.00000	.7300E-03
215	130	129	11630000000.0	.30000	1.00000	.7300E-03
216	116	117	13030000000.0	.30000	1.00000	.7300E-03
217	132	131	11830000000.0	.30000	1.00000	.7300E-03
218	118	119	13230000000.0	.30000	1.00000	.7300E-03
219	133	132	11930000000.0	.30000	1.00000	.7300E-03
220	119	120	13330000000.0	.30000	1.00000	.7300E-03
221	134	133	12030000000.0	.30000	1.00000	.7300E-03

TABLE 2. MODEL FOR IRWIN PROBLEM AS GENERATED BY GENXQ (CONCLUDED)

```
222  120  121  134300000000.0     .30000    1.00000  .7300E-03
223  135  134  121300000000.0     .30000    1.00000  .7300E-03
224  121  122  135300000000.0     .30000    1.00000  .7300E-03
225  136  135  122300000000.0     .30000    1.00000  .7300E-03
226  122  123  136300000000.0     .30000    1.00000  .7300E-03
227  137  136  123300000000.0     .30000    1.00000  .7300E-03
228  123  124  137300000000.0     .30000    1.00000  .7300E-03
229  138  137  124300000000.0     .30000    1.00000  .7300E-03
230  124  125  138300000000.0     .30000    1.00000  .7300E-03
231  139  138  125300000000.0     .30000    1.00000  .7300E-03
232  125  126  139300000000.0     .30000    1.00000  .7300E-03
233  140  139  126300000000.0     .30000    1.00000  .7300E-03
234  126  127  140300000000.0     .30000    1.00000  .7300E-03
235  141  140  127300000000.0     .30000    1.00000  .7300E-03
236  127  128  141300000000.0     .30000    1.00000  .7300E-03
237  142  141  128300000000.0     .30000    1.00000  .7300E-03
238  128  129  142300000000.0     .30000    1.00000  .7300E-03
239  143  142  129300000000.0     .30000    1.00000  .7300E-03
240  129  130  143300000000.0     .30000    1.00000  .7300E-03
241  145  144  131300000000.0     .30000    1.00000  .7300E-03
242  131  132  145300000000.0     .30000    1.00000  .7300E-03
243  146  145  132300000000.0     .30000    1.00000  .7300E-03
244  132  133  146300000000.0     .30000    1.00000  .7300E-03
245  147  146  133300000000.0     .30000    1.00000  .7300E-03
246  133  134  147300000000.0     .30000    1.00000  .7300E-03
247  148  147  134300000000.0     .30000    1.00000  .7300E-03
248  134  135  148300000000.0     .30000    1.00000  .7300E-03
249  149  148  135300000000.0     .30000    1.00000  .7300E-03
250  135  136  149300000000.0     .30000    1.00000  .7300E-03
251  136  137  149300000000.0     .30000    1.00000  .7300E-03
252  150  149  137300000000.0     .30000    1.00000  .7300E-03
253  137  138  150300000000.0     .30000    1.00000  .7300E-03
254  138  139  150300000000.0     .30000    1.00000  .7300E-03
255  151  150  139300000000.0     .30000    1.00000  .7300E-03
256  139  140  151300000000.0     .30000    1.00000  .7300E-03
257  140  141  151300000000.0     .30000    1.00000  .7300E-03
258  152  151  141300000000.0     .30000    1.00000  .7300E-03
259  141  142  152300000000.0     .30000    1.00000  .7300E-03
260  153  152  142300000000.0     .30000    1.00000  .7300E-03
261  142  143  153300000000.0     .30000    1.00000  .7300E-03
262  155  154  144300000000.0     .30000    1.00000  .7300E-03
263  144  145  155300000000.0     .30000    1.00000  .7300E-03
264  156  155  145300000000.0     .30000    1.00000  .7300E-03
265  145  146  156300000000.0     .30000    1.00000  .7300E-03
266  157  156  146300000000.0     .30000    1.00000  .7300E-03
267  146  147  157300000000.0     .30000    1.00000  .7300E-03
268  158  157  147300000000.0     .30000    1.00000  .7300E-03
269  147  148  158300000000.0     .30000    1.00000  .7300E-03
270  159  158  148300000000.0     .30000    1.00000  .7300E-03
271  148  149  159300000000.0     .30000    1.00000  .7300E-03
272  163  162  152300000000.0     .30000    1.00000  .7300E-03
273  152  153  163300000000.0     .30000    1.00000  .7300E-03
  0
  1  159  149  150  151  152  162  161  160    1
  130000000.0     .30000    1.00000  .00073000
  0
```

1.1.4 GENER - PROGRAM LISTING

A complete listing of the source statements contained in GENER (file GENER at Armament Laboratory) follows.

```
C ***** PROGRAM TO GENERATE NODES AND TRIANGLES FOR CRAK1 AND CRAK4 *****
C
C
      PARAMETER KORE=15000,IN=3000
      DIMENSION A(KORE),B(IN),C(IN),TITLE(20),TYPE(5),ND(10),P(4)
      COMMON NODES,NTRIS,NSPRNG,NCRK8,NCRK10,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      READ(5,100) (TITLE(I),I=1,20)
  100 FORMAT(20A4)
      READ(5,200) NDOF,NRSTGN,NDOFGN,NTRIGN,MATGEN,ISTATE,NC8,NC10,NLUMP
      READ(5,200) IOUNIT,INPRT,IOPRT,IOUT1,IOUT2
      READ(5,100) (TYPE(I),I=1,5)
  200 FORMAT(10I5)
      N1=1
      N2=1
      N3=1
      IZERO=0
      IF(NDOF)20,20,10
   10 CALL CORGEN(A,B)
      IF(NODES,GT,IN) WRITE(6,400)
  400 FORMAT(10X,30HNODES GENERATED EXCEED STORAGE/)
      N1=NOOF*NODES+1
      N2=N1
   20 IF(NRSTGN)40,40,30
   30 CALL RSTGEN(A (N1),B)
      N2=N1+NDOF*NODES
      N3=N2
      IF(NDOFGN)40,40,35
   35 CALL DOF(A(N1),B,NDOFGN)
   40 IF(NTRIGN)60,60,50
   50 CALL TRIGEN(A(N2),C)
      IF(NTRIS,GT,IN) WRITE(6,500)
  500 FORMAT(10X,34HTRIANGLES GENERATED EXCEED STORAGE/)
      N3=N2+3*NTRIS
   60 WRITE(IOUNIT,100) (TITLE(I),I=1,20)
      WRITE(IOUNIT,200) NODES,NTRIS,NSPRNG,NC8,NC10,NLUMP,
     *                  NLDNDS,NRSTR,NDOF,ISTATE
      WRITE(IOUNIT,300) INPRT,IOPRT,IOUT1,IOUT2
  300 FORMAT(I5,15X,I5,10X,2I5)
      WRITE(IOUNIT,100) (TYPE(I),I=1,5)
      CALL OUT(A,A(N1),A(N2),A(N3),B,IOPRT,IOUNIT)
      IF(NC8,LE.0) GO TO 80
      NCRK8=0
   70 READ(5,200) NO,(ND(I),I=1,8),ISTRES
      IF(NO.LE.0) GO TO 75
      READ(5,600) (P(I),I=1,4)
  600 FORMAT(4E10.0)
      NCRK8=NCRK8+1
      WRITE(IOUNIT,200) NO,(ND(I),I=1,8),ISTRES
      WRITE(IOUNIT,700) NO,(P(I),I=1,4)
  700 FORMAT(I5,F10.1,2F10.5,F10.8)
      GO TO 70
```

PROGRAM TO GENERATE NODES AND TRIANGLES FOR CRAK1 AND CRAK4 (CONTINUED)

```
75 WRITE(IOUNIT,200) IZERO
80 IF(NC10.LE.0) GO TO 90
90 CONTINUE
   STOP
   END
```

```
      SUBROUTINE CORGEN(COORD,IN)
      DIMENSION COORD(1),XINC(3),X(3),IN(1)
      COMMON NODE,NTRIS,NSPRNG,NCRK8,NCRK10,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      NSCALE=0
      NODE=0
      NDLRG=0
   10 READ(5,100) ND,NBUMP,NUMND,NDINC,X(1),X(2),X(3)
  100 FORMAT(4I5,3E10.0)
      IF(ND)25,20,30
   25 XINC(1)=X(1)
      XINC(2)=X(2)
      XINC(3)=X(3)
      GO TO 80
   30 NODE=NODE+1
      IN(NODE)=ND
      IF(ND.GT.NDLRG) NDLRG=ND
      LOC=NDOF*(ND-1)
      DO 50 J=1,NDOF
      COORD(LOC+J)=X(J)
   50 CONTINUE
      GO TO 10
   80 CONTINUE
      DO 60 I=1,NBUMP
      DO 60 J=1,NUMND
      NODE=NODE+1
      ND=IN(NODE-NUMND)
      LOC=NDOF*(ND-1)
      ND1=ND+NDINC
      IN(NODE)=ND1
      IF(ND1.GT.NDLRG) NDLRG=ND1
      LOC1=NDOF*(ND1-1)
      DO 70 K=1,NDOF
      COORD(LOC1+K)=COORD(LOC+K)+XINC(K)
   70 CONTINUE
   60 CONTINUE
      GO TO 10
   20 CONTINUE
      DO 21 I=1,NDOF
      IF(ABS(X(I)).GT.0) GO TO 24
      X(I)=1.0
      GO TO 21
   24 NSCALE=1
   21 CONTINUE
      IF(NSCALE)26,26,27
   27 DO 28 I=1,NODE
      LOC=NDOF*(I-1)
      DO 28 J=1,NDOF
      COORD(LOC+J)=X(J)*COORD(LOC+J)
   28 CONTINUE
   26 IF(INPRT.GT.1) WRITE(6,300) (IN(I),I=1,NODE)
  300 FORMAT(14I5)
      IF(INPRT)160,160,52
   52 WRITE(6,200) NODE,NDLRG
  200 FORMAT(10X,21HNODES GENERATED      =,I5,
     *      /10X,21HLARGEST NODE NUMBER =,I5)
  160 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE RSTGEN(NF,IN)
      DIMENSION NF(1),IN(1),IFR(3)
      COMMON NNODES,NTRIS,NSPRNG,NCRK0,NCRK1D,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      NID=0
      NRR=0
   10 READ(5,100) ND,IFR(1),IFR(2),IFR(3),NBUMP,NDINC
  100 FORMAT(6I5)
      IF(ND)30,30,15
   15 LOC=NDOF*(ND-1)
      NBUMP=NBUMP+1
      DO 20 I=1,NBUMP
      INCLOC=LOC+NDOF*NDINC*(I-1)
      DO 20 J=1,NDOF
      NF(INCLOC+J)=IFR(J)
   20 CONTINUE
      GO TO 10
   30 DO 40 I=1,NNODES
      ND=IN(I)
      LOC=NDOF*(ND-1)
      DO 40 J=1,NDOF
      IF(NF(LOC+J))41,40,43
   41 NID=NID+1
      GO TO 40
   43 NRR=NRR+1
   40 CONTINUE
      NRSTR=NRR+NID
      RETURN
      END
```

```
      SUBROUTINE DOF(NF,IN,NDOFGN)
      DIMENSION NF(1),IN(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,
     *       NLDNOS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      IF(NDOFGN.LE.1) GO TO 40
      NSEQ=0
   10 READ(5,100) ND,NBUMP,NDINC
  100 FORMAT(10I5)
      IF(ND)50,50,20
   20 NBUMP=NBUMP+1
      DO 30 I=1,NBUMP
      ND1=ND+NDINC*(I-1)
      NSEQ=NSEQ+1
      IN(NSEQ)=ND1
   30 CONTINUE
      GO TO 10
   40 NSEQ=NNODES
   50 IF(NSEQ-NNODES)60,70,60
   60 WRITE(6,200) NSEQ,NNODES
  200 FORMAT(10X,29HNUMBER OF RESEQUENCED NODES =,I5,
     *      /10X,29HNUMBER OF GENERATED NODES   =,I5/)
      IFLAG=IFLAG+1
   70 NER=0
      NID=0
      NRR=0
      DO 80 I=1,NNODES
      ND=IN(I)
      LOC=NDOF*(ND-1)
      DO 80 J=1,NDOF
      IF(NF(LOC+J))81,82,83
   81 NID=NID+1
      GO TO 80
   82 NER=NER+1
      NF(LOC+J)=NER
      GO TO 80
   83 NRR=NRR+1
      NF(LOC+J)=0
   80 CONTINUE
      NIR=-NER
      LOC=NDOF*(NNODES)
      DO 91 I=1,LOC
      IF(NF(I))90,91,91
   90 NIR=NIR-1
      NF(I)=NIR
   91 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TRIGEN(JT,IN)
      DIMENSION JT(1),IN(1),JTEMP(3)
      COMMON NNODES,NT,NSPRNG,NCRK8,NCRK10,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      NT=0
      NOTRI=0
   10 READ(5,100) NO,(JTEMP(I),I=1,3),NBUMP,NUMTRI,INC
  100 FORMAT(7I5)
      IF(NO)40,20,30
   30 NT=NT+1
      IN(NT)=NO
      LOC=3*(NO-1)
      DO 35 I=1,3
      JT(LOC+I)=JTEMP(I)
   35 CONTINUE
      IF(JT(LOC+1).EQ.JT(LOC+2).OR.JT(LOC+1).EQ.JT(LOC+3).OR.
     *   JT(LOC+2).EQ.JT(LOC+3)) WRITE(6,220) NO
      IF(JT(LOC+1).GT.NNODES) WRITE(6,220) NO
      IF(JT(LOC+2).GT.NNODES) WRITE(6,220) NO
      IF(JT(LOC+3).GT.NNODES) WRITE(6,220) NO
  220 FORMAT(10X,9HTRIANGLE ,I5,17H  IS MISSPECIFIED/)
      IF(NO.GT.NOTRI) NOTRI=NO
      GO TO 10
   40 DO 45 I=1,NBUMP
      DO 45 J=1,NUMTRI
      NT=NT+1
      NT1=IN(NT-NUMTRI)
      NO=NT1+INC
      IN(NT)=NO
      LOC=3*(NO-1)
      LOC1=3*(NT1-1)
      DO 50 K=1,3
      JT(LOC+K)=JT(LOC1+K)+JTEMP(K)
   50 CONTINUE
      IF(JT(LOC+1).EQ.JT(LOC+2).OR.JT(LOC+1).EQ.JT(LOC+3).OR.
     *   JT(LOC+2).EQ.JT(LOC+3)) WRITE(6,220) NO
      IF(JT(LOC+1).GT.NNODES) WRITE(6,220) NO
      IF(JT(LOC+2).GT.NNODES) WRITE(6,220) NO
      IF(JT(LOC+3).GT.NNODES) WRITE(6,220) NO
      IF(NO.GT.NOTRI) NOTRI=NO
   45 CONTINUE
      GO TO 10
   20 IF(INPRT)70,70,60
   60 WRITE(6,200) NT,NOTRI
  200 FORMAT(10X,32HNUMBER OF TRIANGLES GENERATED  =,I4,
     *      /10X,32HLARGEST TRIANGLE NO. GENERATED =,I4)
   70 CONTINUE
      CALL MATGEN(JT,JT(3*NT+1))
      IF(INPRT.GT.1) WRITE(6,300) (IN(I),I=1,NT)
  300 FORMAT(14I5)
      RETURN
      END
```

```
      SUBROUTINE MATGEN(JT,TPRP)
      DIMENSION JT(1),TPRP(1),TEMP(4)
      COMMON NNODES,NT,NSPRNG,NCRK8,NCRK10,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      KOUNT=0
   10 READ(5,100) I,NBUMP,(TEMP(J),J=1,4)
  100 FORMAT(2I5,4E10.0)
      IF(I)20,20,30
   30 NBUMP=NBUMP+1
      LOC=4*(I-1)
      DO 15 J=1,NBUMP
      KOUNT=KOUNT+1
      LOC1=LOC+4*(J-1)
      DO 15 K=1,4
      TPRP(LOC1+K)=TEMP(K)
   15 CONTINUE
      GO TO 10
   20 IF(KOUNT-NT)25,35,25
   25 WRITE(6,200) KOUNT,NT
  200 FORMAT(10X,"TRI-PROPERTIES INPUT = ",I5,
     *      /10X,"TRIANGLES INPUT       = ",I5/)
   35 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE OUT(COORD,NF,JT,TPRP,IN,IOPRT,IOUNIT)
      DIMENSION COORD(1),NF(1),JT(1),TPRP(1),IN(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE
      IZERO=0
      IF(NNODES.LE.0) GO TO 20
      IF(NDOF.GT.2) GO TO 10
      DO 5 J=1,NNODES
      I=IN(J)
    5 WRITE(IOUNIT,100) (I,NF(2*I-1),NF(2*I),COORD(2*I-1),
     *                   COORD(2*I))
  100 FORMAT(3I5,5X,2F10.4)
      WRITE(IOUNIT,100) IZERO
      GO TO 20
   10 DO 15 J=1,NNODES
      I=IN(J)
   15 WRITE(IOUNIT,200) (I,NF(3*I-2),NF(3*I-1),NF(3*I),
     *                   COORD(3*I-2),COORD(3*I-1),COORD(3*I))
  200 FORMAT(4I5,3F10.4)
      WRITE(IOUNIT,100) IZERO
   20 IF(NTRIS.LE.0) GO TO 30
      WRITE(IOUNIT,300) (I,JT(3*I-2),JT(3*I-1),JT(3*I),
     *                   TPRP(4*I-3),TPRP(4*I-2),TPRP(4*I-1),
     *                   TPRP(4*I),I=1,NTRIS)
  300 FORMAT(4I5,F10.1,2F10.5,E10.4)
      WRITE(IOUNIT,100) IZERO
   30 CONTINUE
      RETURN
      END
```

1.2 CRAKD - A FINITE-ELEMENT ANALYSIS PROGRAM

CRAKD, as documented in this report, is an all-Fortran program designed to be contained and executed totally in-core on the Control Data Corporation 6600 digital computer at the Armament Development and Test Center, Eglin Air Force Base. Only standard I/O statements (READ and WRITE) are used. The program performs linear-elastic, displacement-method finite-element static, harmonic, and transient analyses of cracked structures. Four types of elements are available to establish structural models: uniform plane stress/strain isotropic triangles, massless axial/shear springs, uniform 8-node symmetric and 10-node unsymmetric isotropic cracked elements, and lumped (point) masses.

The program, whose organization is illustrated by Figure 6, consists of a main program, CRAKD, which acts as the driver for the four major program divisions. These divisions are input, equations formulation, equations solution, and output. The planned program structure (Figure 6) makes it convenient to implement new element types as needed or desired.

Additionally, owing to the way central memory is allocated in the input segment, core storage requests can be set by the user for each model by changing two statements in the main program, CRAKD. The two statements which control storage requests are found on lines 8 and 17 of CRAKD. They are:

```
DIMENSION A(XXXXX), N(16)
```

and

```
NKORE = XXXXX
```

where XXXXX is the number of central memory words required for storage of model data and analysis formulation and solution.

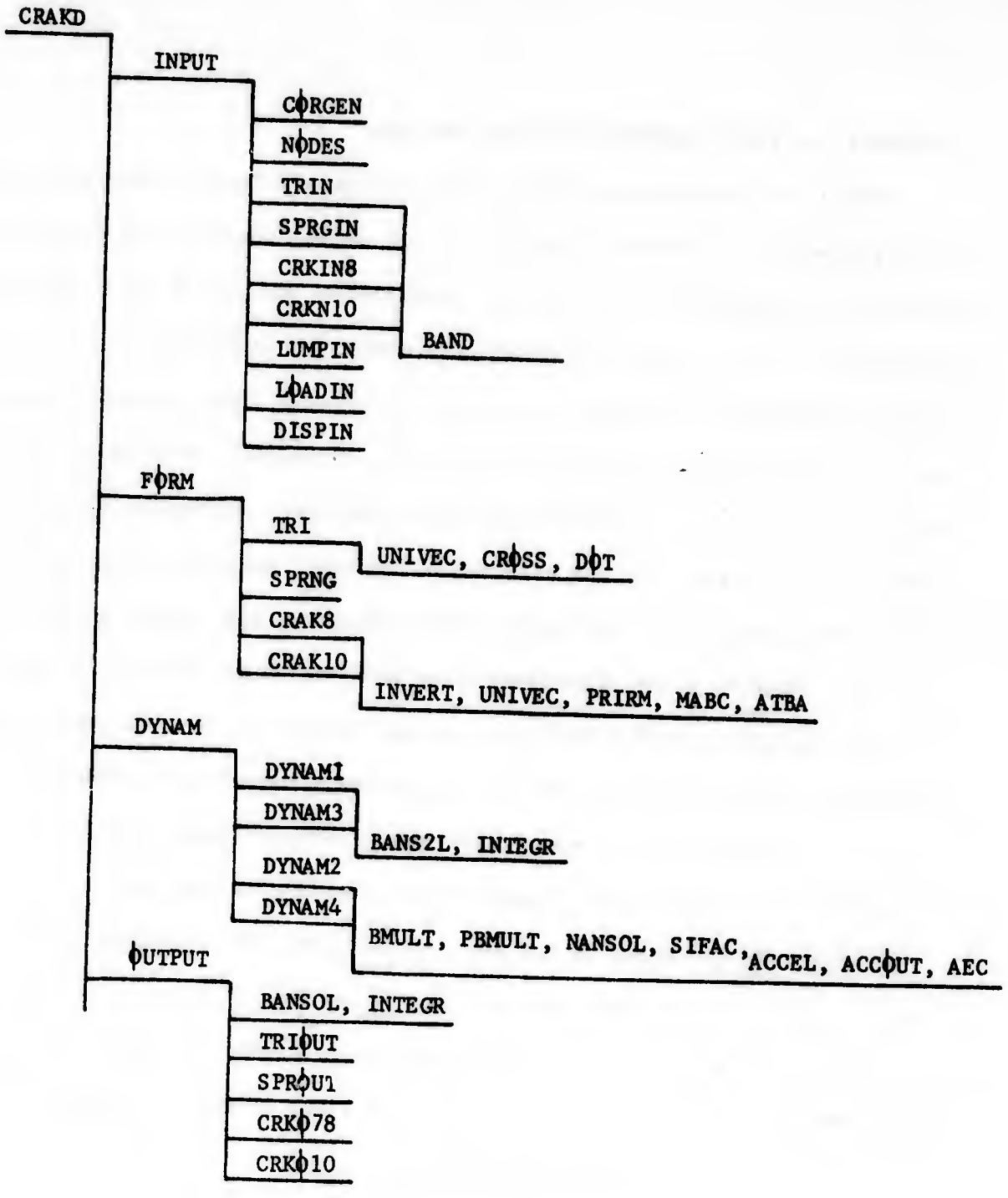


Figure 6. CRAKD Program Structure

The individual subprograms which comprize the four major analysis divisions are described in detail in subparagraph 3.2.1. A complete listing of the source statements of CRAKD and all its subprograms is contained in subparagraph 3.2.4.

1.2.1 DESCRIPTION OF THE SUBPROGRAMS OF CRAKD

PROGRAM CRAKD (INPUT=100, OUTPUT=100, DATA=100, TAPE5=INPUT, TAPE6=OUTPUT, TAPE21=DATA, TAPE25, TAPE26, TAPE28, TAPE29)

CRAKD controls the core storage requests through the DIMENSION A(xxxxx), N(16) statement and calls the four major program subroutines INPUT, FORM, DYNAM, and OUTPUT. The input file where model specififcation data is located is TAPE5. The file MODEL (see paragraph 3.1) is taken to be TAPE5. This procedure is illustrated in the examples contained in subparagraphs 3.2.2 and 3.2.3. The files TAPE6, TAPE21, TAPE25, TAPE26, TAPE28, and TAPE29 will be introduced and discussed in the descriptions of the subprograms where they are used.

SUBROUTINE INPUT (A,N)

This subroutine controls all of the input associated with establishing a finite-element model and specifying the mass characterization and the type of analysis to be made. INPUT also organizes all data contiguously from the beginning of the array A such that no storage is left unused, unless A has been originally specified too large for the particular problem addressed. The locator array, N, contains the 16 starting addresses within A of various model data. As a last task, INPUT computes and prints both the problem's requirements for the size of A and the program's current A dimension. The user can then modify the A specification for subsequent program executions

to reduce central memory requirements and thereby save money.

During the input phase the subprograms called by INPUT--CORIN, NODES, TRIN, SPRGIN, CEKIN8, CRKN10, LUMPIN, DISPIN, and LOADIN--all perform checking of the data for some commonly occurring user errors. When such errors occur, descriptive program messages will be issued to enable the user to correct the deficiencies, and an error flag, IFLAG which is initially set to zero, will be positively incremented by one. If IFLAG > zero at the conclusion of INPUT, the message

**********PROGRAM STOPS**********

will be printed and INPUT terminates the run. This prevents analyses attempts with obviously inconsistent data.

SUBROUTINE CORIN (COORD, NF, NDOF)

This subroutine reads the node numbers, degree-of-freedom numbers, NF, and node coordinates COORD as they are listed on file MODEL and stores them in ascending node order in the portions of the A array (dimensioned in CRAKD) assigned by INPUT to COORD and NF (A(1) and A(N(1)), respectively). The parameter NDOF is input by the user as either 1, 2 or 3 to designate 1-D, 2-D, or 3-D model (see subparagraph 3.2.2 for Input Guide).

Checks are made on the total number of nodes and restraints (as determined from examination of the freedom numbers) to determine if they are consistent with the specifications entered by the user.

SUBROUTINE NODES (COORD, NF, NDOF)

This subroutine lists node numbers, freedom numbers NF, and node coordinates COORD in ascending node order on TAPE21 which is assigned the

name DATA by the PROGRAM CRAKD statement. Also, should the user desire (see subparagraph 1.2.2 for Input Guide), a separate listing with heading can be obtained during the program execution onto TAPE6 which is the printer file.

SUBROUTINE TRIN (JT, TPRP, NF, NDOF)

This routine reads the element number, node or joint specifications, and the material and section properties as they are listed for each triangle element from file MODEL. These are stored into the JT (joint or node numbers defining the element) and TPRP (material constants, E, ν and ρ, and element thickness, t) portions of the overall A array as assigned by INPUT (A(N(2)) and A(N(3)) respectively) in its call to TRIN.

Checks are made on the total number of elements input to insure consistency with user specifications.

Also, TRIN calls BAND which uses the JT and NF arrays.

The element specifications are always listed on TAPE21 (DATA) for possible future usage, and, should the user specify the appropriate option in the input sequence, a similar listing with heading can be written onto TAPE6, the printer file.

SUBROUTINE SPRGIN (JS, SPRP, NF, NDOF)

This subprogram reads spring element data from the input file MODEL and stores them into the JS (joint or node numbers defining the element and its reference directions) and the SPRP (linear spring constants) array portions of the A array which are assigned by INPUT (A(N(4)) and A(N(5)),respectively) through its call to SPRGIN. Also, SPRGIN calls BAND which uses JS and NF.

Checks or the total number of spring elements are made to insure data consistency with the user's specifications.

A listing with heading is always written onto TAPE21 (file DATA), but an appropriate user input option can cause a similar listing to appear in the printed output of the programs execution.

SUBROUTINE CRKIN8 (JC, CPRP, NF, NDOF)

SUBROUTINE CKRN10 (JC, CPRP, NF, NDOF)

The purpose of these subprograms is to read from the input file, MODEL, the element number node definition, plane stress/strain specification, material constants and thickness for, respectively, each 8-node and 10-node cracked element in the model. These are then stored in ascending element order in the JC (element node definition and plane stress/strain indicator) and CPRP (material constant, E, ν, and ρ, and thickness, t) array portions of the A array. These portions are determined by INPUT (A(N(6)), A(N(8)) and A(N(7)), A(N(9)), respectively) to make these data contiguous with previously stored information, CRKIN8 and CRKN10 call BAND which uses JC and NF.

Checks are made to insure that the total number of 8-node and 10-node elements are consistent with the specifications of the user.

A listing is made on TAPE21 (file DATA) for each cracked element, and should the user specify the appropriate input option, a separate similar listing will appear in the printed output from the program's execution.

SUBROUTINE LUMPIN (NF, SM, NDOF, NER)

This subprogram reads from the input file the node number and the mass which is to be lumped at that node. Any such masses are added into the

structural mass matrix, SM, for each of the active degrees-of-freedom at the designated node.

Checks are made to insure that the total number of lumped masses entered are consistent with user specifications.

Messages are printed for each point mass read and summed into SM.

The location, A(N(12)), of the beginning of SM is assigned by INPUT to be contiguous to the location of the structural data and the space allotted for the formation of the structural stiffness matrix.

SUBROUTINE LOADIN (NF, P, NDOF)

SUBROUTINE DISPIN (NF, P, NDOF)

These two subprograms read from the input file, MODEL, the node numbers and the values, respectively, of the applied node loads and imposed non-zero node displacements for static analyses. The loads and displacements are placed in the vector P whose location, A(N(10)), is assigned by INPUT to place it immediately following previous model data. For transient analyses, values placed in the P vector by LOADIN are used only as pointers to nodes where the calculation of accelerations is required.

Checks are made to guarantee consistency between number of loads and/or imposed displacements entered and the number which the user specifies.

A listing of the P vector occurs on TAPE21 (file DATA) if the user inputs the appropriate option (see Input Guide, subparagraph 1.2.2).

SUBROUTINE BAND (ND, NDE, NF, NDOF, NB)

The purpose of this routine which is called by TRIN, SPRGIN, CRKIN8, and CRKN10 is to calculate the maximum bandwidth, NB, of the structural stiffness and mass (if the consistent mass characterization is chosen)

matrices. The quantity NB is required by FORM (equations formulation segment of the program).

Basically, for each element the maximum difference occurring between the degree-of-freedom numbers, NF, for all the nodes, ND, is determined (ND has NDE numbers where NDE = 3 for triangles, 2 for springs, 8 for 8-node elements, and 10 for 10-node elements). The maximum of these for all elements in a model is NB-1.

SUBROUTINE FORM (A, N)

Subroutine FORM is the controlling program for assembling the structural stiffness and mass matrices. The subprograms TRI, SPRNG, CRAK8, and CRAK10 called by FORM compute element stiffness and mass matrices for triangles, springs, 8-node cracked elements, and 10-node cracked elements, respectively. The structural stiffness ST and structural mass SM are dimensioned NB Columns (bandwidth as determined by BAND in the INPUT segment by NER rows (total independent degrees-of-freedom in the model) and are contained within the A array. The starting addresses within A for ST and SM are contained in the locator array N at N(11) and N(12).

SUBROUTINE TRI (CO, NF, JT, TPRP, P, ST, SM, NROW)

This subroutine forms the elemental stiffness matrices for uniform constant-plane-stress or plane-strain triangles, and then assembles them into the banded (NROW x NB) matrices ST and SM. The node coordinates CO, element node definitions JT, and the constants E, ν, ρ and t (Young's modulus, Poisson's ratio, mass density, and element thickness) contained in TPRP are used to generate element matrices. The freedom numbers NF point where the elemental stiffness and masses are to be summed into ST and SM. The load/imposed displacement vector P is available for the calculation of modified node forces which occur whenever non-zero displacements are imposed (static analyses only). These modified forces arise as follows: The equations to be solved are

$$\begin{bmatrix} K_{II} & K_{IR} \\ K_{IR} & K_{RR} \end{bmatrix} \begin{Bmatrix} U_I \\ U_R \end{Bmatrix} = \begin{Bmatrix} P_I \\ P_R \end{Bmatrix}$$

where U_I are the unknown independent node displacements and U_R are the imposed (known) zero and non-zero displacements. The equations, which are solved for the U_I, are symbolically

$$K_{II}\ U_I + K_{IR}\ U_R = P_I \ ,$$

or, since the U_R are known,

$$K_{II}\ U_I = P_I - K_{IR}\ U_R$$

If all the U_R are zero, the right-hand side remains unaltered. However, if non-zero U_R are imposed, the term $K_{IR}\ U_R$ modifies the right-hand side thus

$$K_{II}\ U_I = P_I^*$$

where

$$P_I^* = P_I - K_{IR}\ U_R \ .$$

TRI calls the subprograms UNIVEC, CROSS, DOT and ATBA.

SUBROUTINE SPRNG (CO, NF, JS, SPRP, P, ST, NBW)

This subprogram uses CO, JS, and SPRP to compute element stiffness matrices for massless linear axial/shear springs, and, using NF, assembles these elemental stiffnesses into the banded (NBW X NB) structural stiffness ST. P, the load/imposed displacement vector, is available to establish a modified force vector (see description in SUBROUTINE TRI) if any of the spring elements connect nodes which have imposed non-zero restraints.

SUBROUTINE CRAK8 (CO, NF, JC, CPRP, P, ST, SM, NDOF, NROW, S1, S2, S3, T1, T2, T3)

SUBROUTINE CRAK10 (CO, NF, JC, CPRP, P, ST, SM, NDOF, NROW, S1, S2, S3, T1, T2, T3)

These subroutines use CO, JC, and CPRP to compute the elemental matrices S1, S2, S3, T1, T2, and T3 for uniform 8-node, isotropic symmetric crack and uniform 10-node isotropic, unsymmetric crack elements. The 8-node element is intended for modelling mode I structures, i.e., those which are symmetric about the line of the crack, while the 10-node element is intended for modelling unsymmetric (modes I and II) cracked structures. The elemental stiffness and mass are assembled into the banded (NROW x NB) ST and SM matrices. Contributions to a modified force vector is computed if any of the 8-node elements connect nodes with imposed non-zero restraints.

CRAK8 and CRAK10 also contain Fortran DATA statements which are permanently stored dimensionless element stiffness and mass matrices. These dimensionless forms, which are possible to establish because of the fixed shape and node positions chosen for the cracked elements (see subparagraph 1.1.2, Card(s) (15a) and (17a) or subparagraph 1.2.2, Card(s) 11 and 14), make efficient the inclusion of a cracked element into a finite-element analysis.

Intermediate element matrices can be printed during matrix assembly if the appropriate user options are specified (Card 3, subparagraph 1.2.2).

The subprograms UNIVEC, PRTRM, INVERT, MABC, and ATBA are called. Element dimensions, thickness, shear modulus, and element matrices are written onto TAPE25 (8-node) and TAPE26 (10-node) to extract mode I stress-intensity factors K_I (8-node) and mode I and mode II stress-intensity factors K_I and

K_{II} (10-node).

The load/displacement vector P is used to compute contributions to the modified force vector if any cracked elements connect nodes with imposed non-zero displacements.

SUBROUTINE INVERT (A, B, N, NA)

This subroutine inverts the upper left N x N submatrix of the matrix A and stores it in the upper left of B. The overall size of both A and B is NA x NA. The submatrix of A is destroyed during the inversion process.

SUBROUTINE PRTRM (AR, A, PR, PC, NFILE)

This subroutine writes the upper left PC x PC submatrix of A, whose overall size is AR x AR, onto the unit whose number is NFILE. NFILE is now set to be 21 so the matrix listing is on TAPE 21 which has been assigned the name DATA by the PROGRAM CRAKD statement. The listed form of the matrix is first the row number followed by the first PC terms of that row--listed 10 terms per line.

FUNCTION DOT (V1, V2, NO)

This subprogram computes the scalar (DOT) product for the two vectors V1 and V2. Each vector has N components.

SUBROUTINE CROSS (A, B, AXB)

This subroutine calculates the components of the vector (cross) product of the two vectors A and B. The vectors are all assumed to have 3 components.

SUBROUTINE ATBA (AR, A, BR, B, CR, C, N, NB)

This subroutine forms the N x N submatrix of C from the triple matrix product XYZ. X is the transpose of the upper left NB x N submatrix of A, Y is the upper left NB x NB submatrix of B, and Z is the upper left NB x N submatrix of A. The overall dimensions of A, B, and C are, respectively, AR x AR, BR x BR, and CR x CR.

SUBROUTINE MABC (AR, A, BR, B, CR, C, NRA, NCARB, NCB)

This subroutine forms the upper left NRA x NCB submatrix of C from the matrix product XY. X is the upper left NRA x NCARB submatrix of A, and Y is the upper left NCARB x NCB submatrix of B. The overall sizes of A, B, and C are, respectively, AR x AR, BR x BR, and CR x CR.

SUBROUTINE UNIVEC (A, N, XL)

This subroutine computes the length XL of the N-component vector A. The vector A is then converted to a unit-length vector prior to the subroutine return. UNIVEC calls DOT to compute XL^2 as A·A.

SUBROUTINE DYNAM (A, N)

This subroutine controls the solution of the equations of motion for harmonic and transient analyses and the output of the solution--displacements, accelerations, and stress-intensity factors. For harmonic analyses, subprograms DYNAM1 and DYNAM3 are called. For transient analyses, DYNAM2 and DYNAM4 are called.

Return of DYNAM is to the main program CRAKD.

SUBROUTINE DYNAM1 (NF, JC8, JC10, P, ST, SM, DYN, P2, U, NDOF, NROW)

SUBROUTINE DYNAM3 (NF, JC8, JC10, P, ST, SM, DYN, P2, U, NDOF, NROW)

The purpose of these two subroutines is to solve the system of equations

$$(K - \omega^2 M)\ U = 0$$

where K and M are the structural stiffness, ST, and structural mass, SM, matrices assembled in the FORM segment, and the U are the unknown node displacements. The finite-element model's mass distribution can be characterized by either a consistent-mass distribution (SM is NROW x NB) or a lumped-mass distribution (SM is NROW x 1). DYNAM1 is used for consistent-mass characterizations and DYNAM3 for lumped-mass.

Both subprograms use the matrices DYN, P2, and U during the solution. Both routines call BANS2L and INTEGR. BANS2L, the actual equation solver, will be described following.

Each value of ω in the system equations is established from three quantities which are read from the input file by DYNAM1 and DYNAM3. These quantities are ω_i and ω_f, the initial and final value of the applied frequency, and NSTEP, the number of equal $\Delta\omega$'s to be added to ω_i to establish ω_f, i.e.,

$$\omega_f = \omega_i + (\text{NSTEP})\ \Delta\omega \quad .$$

Harmonic response (displacements and stress-intensity factors) will be calculated for ω_i, $\omega_i + \Delta\omega$, $\omega_i + 2\Delta\omega$, ... , ω_f.

For each ω, node displacements and stress-intensity factors are listed on TAPE21, and printed in the output (TAPE6) of the program execution.

SUBROUTINE BANS2L (AK, R, NEQ, IBAND, NDIM)

This subroutine solves the system of equations [AK] {unknowns} = {R} where AK is in banded form, i.e., NDIM x IBAND, and the number of unknowns is NEQ. The numerical procedure is Gaussian elimination.

SUBROUTINE DYNAM2 (NF, NCB, NC10, P, ST, SM, F, U, U2, NP1, NDOF, NROW)

This subroutine solves the equations of motion

$$\ddot{MU} + KU = F$$

for the transient response of a finite-element model where K and M are the structural stiffness matrix, ST(NROW x NB), and the structural consistent mass matrix, SM(NROW x NB); U and $\ddot{U}$ are the unknown node displacements and accelerations, and F, the time-dependent, user-specified node forces. The Newmark-β ($\beta = 1/4$) time-integration algorithm is chosen to solve the system equations.

The algorithm (1) has two equations: a starting or first step and a general step. The starting step is

$$\left(M + \frac{h^2}{4}K\right) U_1 = \left(M - \frac{h^2}{4}K\right) U_o + h\dot{MU}_o + \frac{h^2}{4}\left(F_o + F_1\right)$$

where the U_o and $\dot{U}_o$ are the user-specified initial conditions (node displacements and velocities at t = 0) and h is the specified time step to t_o t_1 (i.e., from U_o to U_1 and F_o to F_1). The only unknowns in the equation are the U_1. The general step, once U_o and U_1 are known, is (h a specified constant)

$$\left(M + \frac{h^2}{4}K\right) U_{n+2} = 2\left(M - \frac{h^2}{4}K\right) U_{n+1} - \left(M + \frac{h^2}{4}K\right) U_n$$

$$+ \frac{h^2}{4}\left(F_n + 2F_{n+1} + F_{n+2}\right) .$$

DYNAM2 reads from the input file U_o, $\dot{U}_o$, h, and the number of time steps NTS to be considered. Solutions are then generated for times $t_1 = h$, $t_2 = 2h, \ldots, t_f = (NTS)\ h$. Once the displacements U are known at a time step, stress-intensity factors and accelerations at user-designated nodes may be calculated and listed onto TAPE28 and TAPE29, respectively. These quantities are later transferred to TAPE21 and, additionally, the stress-intensity factors are printed in the output (TAPE6) of the program execution. The frequency of calculation of stress-intensities and accelerations is controlled by the two quantities NTSIF and NTACC read from the input file by DYNAM2. Stress-intensities will be computed every NTSIF time steps and accelerations every NTACC time steps.

DYNAM2 uses U and U2 as scratch matrices during the time-integration process. The time-dependent forces, F, are established by calls to a user-constructed subroutine FVEC which generates appropriate forces at the time and position required to simulate the force environment of the model. This subroutine is illustrated in the sample problems (subparagraphs 1.2.3 and 1.2.4).

DYNAM2 also calls BMULT, PBMULT, BANSOL, SOLVE (a second entry to BANSOL), SIFAC, ACCEL and ACCOUT.

SUBROUTINE DYNAM4 (NF, JC8, JC10, P, ST, SM, F, U, U2, BP1, NDOF, NROW)

This subroutine solves the equations of motion

$$M\ddot{U} + KU = F$$

for the transient response of a finite-element model. K and M are the structural stiffness matrix, ST(NROW x NB) and the structural lumped-mass matrix, SM(NROW x 1); $\ddot{U}$ and U are the unknown node accelerations and displacements and

F are the user-specified, time-dependent applied node forces. For the lumped-mass characterization, the time-integration to solve the system equations is accomplished by a central-difference algorithm. The starting step is

$$U_1 = U_o + h\dot{U}_o + \frac{h^2}{4} M^{-1} \left(F_o - KU_o \right)$$

where U_o and $\dot{U}_o$ are the user input initial conditions (node displacements and velocities at time t = 0) and h is the specified time step from t_o to t_1. Since the lumped-mass matrix is a vector, inversion is trivial and the solution for the unknown displacements at t_1, U_1, is quite simple. The general step t_{n+1} once U_1 is known is

$$U_{n+1} = 2U_n - U_{n-1} + h^2 M^{-1} \left(F_n - KU_n \right) \quad .$$

DYNAM4 reads from the input file U_o, $\dot{U}_o$, h, the number of time steps NTS and the calculation frequency NTSIF. Solutions for U will be generated for t = h, t = 2h, ... , t = NTS(h) and stress-intensity factors will be computed every NTSIF time steps. Stress-intensity factors will be listed on TAPE28, TAPE21, and printed in the output (TAPE6) of the program execution.

DYNAM4 uses U and U_2 as scratch matrices, and the forces F are generated through calls to the user-constructed subroutine FVEC. This subroutine is illustrated in subparagraphs 1.2.3 and 1.2.4.

DYNAM4 also calls BMULT and SIFAC.

SUBROUTINE BANSOL (AK, R, NEQ, JBAND, NDIM)

This subroutine solves the equations

[AK] {unknowns} = {R}

where AK is in banded form (NDIM x IBAND) and NEQ is the number of unknowns. The solution obtained by Gaussian elimination is returned in {R}.

SUBROUTINE BMULT (N, IBW, A, X, B)

This subroutine multiplies a vector X by a banded symmetric matrix A (upper portion). The product is returned in B. IBW is the upper bandwidth of A, and N is the number of columns of A.

SUBROUTINE PBMULT (N, IBW, A, X, B, P)

This subroutine is the same as BMULT above, but the product vector B is scaled by the factor P.

SUBROUTINE SIFAC (NF, JC8, JC10, P2, U, NDOF)

This subroutine uses the displacements, P2, as determined by the time-integration subprograms DYNAM2 and DYNAM4 to compute the stress-intensity factors for the 8- and 10-node elements. SIFAC reads information placed on TAPE25 (8-node) and TAPE26 (10-node) by CRAK8 and CRAK10 during the FORM segment and writes the stress-intensities onto TAPE28.

SUBROUTINE ACCEL (P, U2, U1, U, DT)

This subprogram uses the three displacement vectors U2 ($t = t_2$), U_1($t = t_2$ - DT), and U ($t = t_2$ - 2DT) to compute the acceleration (transient consistent-mass analyses only) of nodes which are designated by pointers placed in the P vector. The accelration is

$$\ddot{U} = (U2 - 2(U1) + U) \; / \; DT * DT \quad .$$

The value is then written onto TAPE 29.

SUBROUTINE ACCOUT (NF, P, KF, IT3, IT5, NDOF)

This subroutine transfers accelerations generated by ACCEL (transient analyses only) and stored on TAPE29 to TAPE21 and, if user specified, to printed output (TAPE6).

SUBROUTINE OUTPUT (A, N)

This subroutine controls the solution and subsequent output for static analyses. It calls BANS2L, INTEGR, TRIOUT, SPROUT, CRKOT8, and CRKO10. Node displacements, triangle element stresses, spring element forces, and node force equilibrium checks are listed on TAPE21. Appropriate user-specified values of the output print option will result in a similar listing in the printed output (TAPE6).

SUBROUTINE INTEGR (I, INTGR)

This subroutine extracts the integer I from the location INTGR of the A array in OUTPUT.

SUBROUTINE TRIOUT (CO, NF, JT, TPRP, P, CKFOR, SX, SY, SXY, NDOF)

The purpose of TRIOUT is to use the calculated node displacements, P, to compute the stresses σ_x, σ_y, and τ_{xy}, and store them into SX, SY, and SXY for each constant-plane-stress or plane-strain triangular element. The element forces are also computed and summed into CKFOR, the node-equilibrium-check array. The element stresses are output in elemental coordinates, i.e., the reference + x - direction which is established by the vector from node 1 to node 2 of the triangle's definition.

SUBROUTINE SPROUT (CO, NF, JS, SPRP, P, CKFOR, ELFOR)

This subprogram computes and stores into ELFOR the spring forces associated with the spring constants, K_1, K_2, and K_3 for any spring elements contained in the finite-element model. It also calculates the contribution of each spring to the node-equilibrium-check array CKFOR.

SUBROUTINE CRKOT8 (NF, JC, CPRP, P, CKFOR, CUMKC, DINV, DICOS, SIF1)

SUBROUTINE CRKO10 (NF, JC, CPRP, P, CKFOR, CUMKC, DIWV, DICOS, SIF1, SIF2)

These subroutines use the calculated node displacements, P, and information written on TAPE25 (CRKOT8) and TAPE26 (CRKO10) to compute the stress-intensity factors SIF1 for the 8-node symmetric elements and SIF1 and SIF2 for the 10-node unsymmetric elements.

1.2.2 USER'S INPUT GUIDE TO CRAKD

Following is the complete input sequence, with card description, for establishing a correct input file for CRAKD.

CRAKD - Input Data Sequence

Card — **Description**

1. [Title (72 characters maximum)]

   one card

   read by 12A6

2. [(1) NNODES, (2) NTRIS, (3) NSPRNG, (4) NCRK8, (5) NCRK10, (6) NLUMPS, (7) NLDNDS, (8) NRSTR, (9) NDOF, (10) SFLAG]

   one card

   read by 10I5

   (1) NNODES - Number of nodes

   (2) NTRIS - Number of (constant strain) triangular elements

   (3) NSPRNG - Number of spring elements

   (4) NCRK8 - Number of 8-node cracked elements

   (5) NCRK10 - Number of 10-node cracked elements

   (6) NLUMPS - Number of lumped masses

   (7) NLDNDS

   Note:
   - a) Static - applied loads
   - b) Harmonic - applied load amplitudes - sign indicates 180^{o} phase shift
   - c) Transient - nodes at which accelerations are to be calculated

   (8) NRSTR

   a) Static - number of restraints plus number of imposed displacements

   b) Dynamic - presently only restraints can be handled

   (9) NDOF - Number of degrees of freedom at each nodal point

   a) 2 = plane structure

   b) 3 = 3-D structure: elements are, of course, 2-D

   (10) SFLAG - Stress state code for TRIANGULAR elements

   a) 0 = plane strain

   b) 1 = plane stress

CRAKD - INPUT DATA

3. (1) (2) (3) (4) (5) (6) (7) (8) (10) (11) (12)
[INPRT, IKPRT, ITPRT, ICPRT, IOPRT, IIN8, IIN10, IOUT1, IOUT2, IOUT4, IOUT5]

(subject to modification)

one card

read by 12I5

(1) INPRT - Input data echo control flag

(2) IKPRT - Stiffness formation echo flag

(3) ITPRT - Triangular element data echo flag

(4) ICPRT - Cracked element echo flag

(5) IOPRT - Output data control flag

Analysis Type	Minimum Out	Data Check	Maximum Out
STATIC	0,0,0,0,3	1,0,0,0,4	3,3,3,3,0
HARMONIC	0,0,0,0,0	1,0,0,0,0	3,3,3,3,4
TRANSIENT	0,0,0,0,0	1,0,0,0,0	3,3,3,3,4

(6) IIN8 -

(7) IIN10 -

(8) IOUT1 - Logical unit where main data output file resides
(Only one used now Mar. 77)

(9) IOUT2 -

(10) IOUT3 -

(11) IOUT4 -

(12) IOUT5 -

4. [Type]

One card

read by 6A1

STATIC : Static analysis

HCMASS : Harmonic consistent-mass analysis

HLMASS : Harmonic lumped-mass analysis

TCMASS : Transient consistent-mass analysis

TLMASS : Transient lumped-mass analysis

5. [(1) (2) (3) (4) (5) (6) (7)
Node, Fnx, Fny, Fnz, X , Y , Z]

one card for each node

read by 415, 3E10.0

NODE - Nodal point number

Fn_{α} - Freedom number for the (global) α - direction at NODE,

freedom code is:

- 1 = > imposed displacement
0 = > elastic freedom
+ 1 = > rigid restraint

X - Global X coordinate of Node

Y - Global Y coordinate of Node

Z - Global Z coordinate of Node

6. [0]

one card

read by 15

0 - indicator to INPUT that end of Nodal Point data has been reached

CRAKD-INPUT DATA

NOTE: If any of the numbers

NTRIS, NSPRNG, NCRK8, NCRK10, NLDNDS, NLUMPS

Is zero, omit all input data pertaining to that quantity - including the 0 card marking the end-of-data

7. [(1) TRINR, (2) NODE1, (3) NODE2, (4) NODE3, (5) E, (6) ν, (7) t, (8) ρ]

one card per Triangular Element

Read by 415, 4E10.0

TRINR - Triangular element number

NODE1, NODE2, NODE3 - node numbers of nodes defining vertices of that element

a) specify in a counter-clockwise order

b) the local X axis is defined by the vector from NODE1 to NODE2, i.e., $\overrightarrow{\text{NODE 2 - NODE1}}$

c) the plane of element defined by the vectors NODE1-NODE2 and $\overrightarrow{\text{NODE1-NODE3}}$

E - modulus of elasticity (FL^{-2})

ν - Poisson's ratio

t - thickness (L)

ρ - density (FT^2L^{-1})

{Whatever units are used must, of course, be consistent}

8. [0]

one card

read by 15

0 - indicator to INPUT that end of triangular data has been reached.

9. [(1) NSPRNG, (2) NODE1, (3) NODE2, (4) NODE3, (5) NODE4, (6) K1 , (7) K2 , (8) K3]

one card per spring element

read by 515, 5X, 3E10.0

NSPRNG - Spring element number

NODE1, NODE2 - Define the attachment points of the element

NODE1, NODE3 - Define the direction in which K1 acts

NODE1, NODE4 - Defines, with $\overrightarrow{\text{NODE3-NODE1}}$, the plane in which K1 and K2 act

(a) All K's act between Node 1 and Node 2

(b) K2 acts normal to K1, in the plane $\overrightarrow{\text{(NODE3-NODE1)}} \times \overrightarrow{\text{(NODE4-NODE1)}}$

(c) The K1, K2, K3 directions are right-handed

K1, K2, K3 - Spring constants

10. [0]

one card

read by 15

0 - indicator to INPUT that end of Spring data has been reached.

11. [(1) N8ELT, (2) NODE1, (3) NODE2, (4) NODE3, (5) NODE4, (6) NODE5, (7) NODE6, (8) NODE7, (9) NODE8, (10) SFLAG]

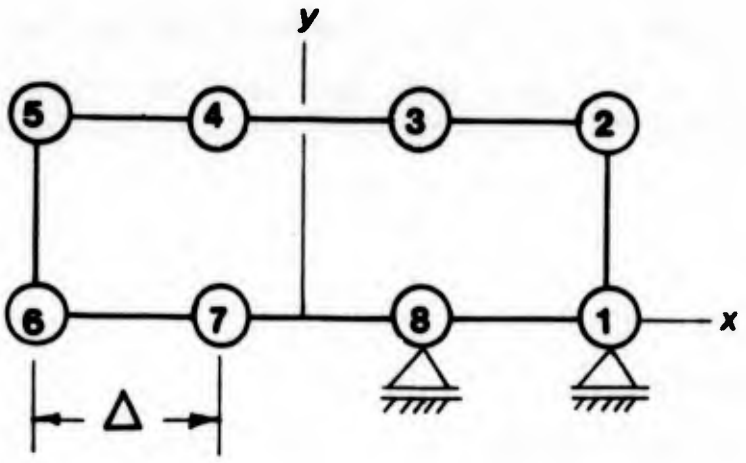
12. [(1) N8ELT, (2) E , (3) ν , (4) t , (5) ρ]

read by 15, 4E10.0

Note: two cards per 8-node element

N8ELT - 8-node cracked element identification number

NODE1-NODE8 - the node numbers of the nodes defining the element, where the order must be as indicated below



NOTE: (a) The element's aspect ratio is 3 x 1 as indicated where the local X-axis is defined by the vector, $\overrightarrow{\text{NODE8-NODE1}}$, and the local Y-axis is defined by the vector, $\overrightarrow{\text{NODE8-NODE3}}$. The lengths are also used for a "rectangularity" check

(b) Support conditions: Zero y displacement at nodes 8 and 1 are essential since the element is intended to represent only symmetric cases.

SFLAG

0 = > plane strain

1 = > plane stress

E - modulus of elasticity (FL^{-2})

ν - Poisson's ratio

t - thickness (L)

ρ - density (FT^2L^{-1})

13. [0]

one card

read by I5

0 - indicator to INPUT that the end of 8-node element data has been reached.

14. [(1) (2) (3) (4) (5) (6) (7) (8) (9) (10) (11) (11)
N10ELT,NODE1,NODE2,NODE3,NODE4,NODE5,NODE6,NODE7,NODE8,NODE9,NODE10,SFLAG]

read by 12I5

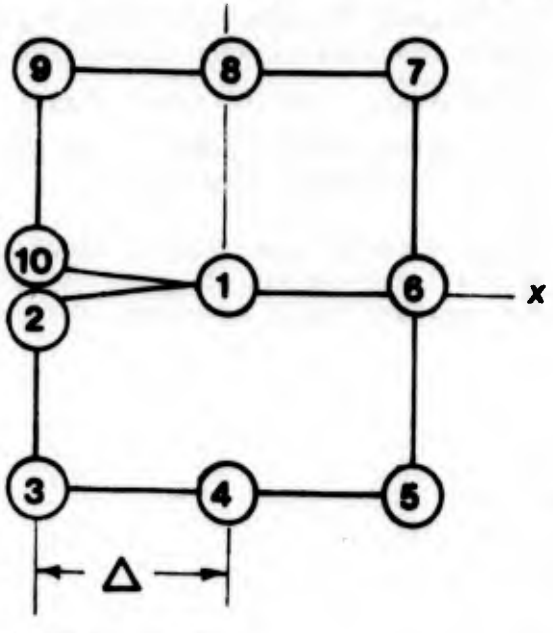
15. [N10ELT,E,ν,t,ρ]

read by I5, 4E10.0

Note: two cards per 10-node element

N10ELT - 10-node element number

NODE1 -,NODE10 - the node numbers of the nodes defining the element, where the order must be as indicated below.



(a) The element's aspect ratio is 1 x 1 as indicated and the local X-axis is defined by the vector, $\overrightarrow{\text{NODE6-NODE1}}$, and the local Y-axis is defined by the vector, $\overrightarrow{\text{NODE8-NODE1}}$. These lengths are also used for a rectangularity check.

(b) This element can represent general (planar) loading.

CRAKD-INPUT DATA

SFLAG 0 = plane strain

1 = plane stress

E - modulus of elasticity (FL^{-2})

ν - Poisson's ratio

t - thickness (L)

ρ - density ($FT^2\ L^{-1}$)

16. [0]

one card

read by I5

0 - indicator to INPUT that the end of 10-node element data has been reached.

17. [NODENR,MASS]

one card per lumped mass

read by I5, E15.8

NODENR - node number

MASS - lumped mass to act there ($FT^2\ L^{-1}$)

18. [0]

one card

read by I5

0 - indicator to INPUT that the end of lumped-mass data has been reached.

19. [(1) (2) (3)
NODENR,DISPX,DISPY,DISPZ]

one card per node with enforced displacement

read by I5, 5X, 3E10.0

NODENR - node number

DISPX - imposed displacement of node, in global coordinate system, in the X-direction

DISPY - displacement in Y-direction

DISPZ - displacement in Z-direction

20. [0]

one card

read by I5

0 - indicator to INPUT that the end of DISPIN data has been reached.

Note: Imposed displacements only for STATIC analysis

21. [(1) NODENR, (2) LOADX, (3) LOADY, (4) LOADZ]

one card per loaded node

read by I5, 5X, 3E10.0

NODENR - node number

LOADX - applied load at node, in global coordinate system, in the X-direction

LOADY - load in the Y-direction

LOADZ - load in the Z-direction

22. [0]

one card

read by I5

0 - indicator to INPUT that the end of LOADIN has been reached.

Note: If a Transient analysis

NODENR - Node number at which acceleration is to be calculated in Transient analysis

LOADX
LOADY } (Code)
LOADZ

- 0 = do not calculate acceleration for this freedom
- 1 = calculated acceleration for this freedom

Note: Following cards only for Transient analysis

23. [NODE, U_{ox}, U_{oy}, U_{oz}]

one card per node (Specify at unrestrained freedoms only)

read by I5, 3E15.8

NODE - node number

U_{ox} - initial (global) x-displacement

U_{oy} - initial (global) y-displacement

U_{oz} - initial (global) z-displacement

24. [0]

one card

read by I5

0 - indicator to DYNAM2 or DYNAM4 that the end of displacement initial conditions has been reached.

25. [NODE, V_{ox}, V_{oy}, V_{oz}]

one card per node (unrestrained freedoms only)

read by I5, 3E15.8

NODE - node number

V_{ox} - initial (global) x - velocity

V_{oy} - initial (global) y - velocity

V_{oz} - initial (global) z - velocity

26. [0]

one card

read by I5

0 - indicator to DYNAM1 or DYNAM4 that the end of velocity initial conditions has been reached.

CRAKD-INPUT DATA: Explanatory Summary

(1) STATIC ANLYSIS (IDYNAM = 0)

The program does an equilibrium analysis and outputs diagnostics (determined by INPRT,IKPRT,ITPRT,KPRT,lOPRT) and stress intensity factor(s).

Also available: triangle stresses
nodal equilibrium check

(2) HARMONIC ANALYSIS (IDYNAM = 1 for HCMASS and IDYNAM = 3 for HLMASS)

The applied loads are interpreted as harmonic amplitudes.

The harmonic analysis subprograms, DYNAM1 and DYNAM3, solicit input of

ω_I - initial excitation frequency,

ω_F - final excitation frequency, and

NSTEP - number of steps in the interval, including end points,

i.e., $\omega_F = \omega_I + \text{NSTEP}(\Delta\omega)$

DYNAM1 also offers the option of rerunning without reassembly.

(3) TRANSIENT ANALYSIS (IDYNAM = 2 for TCMASS and IDYNAM = 4 for TLMASS)

DYNAM2 and DYNAM4 solicit terminal input of initial displacements and velocities and

DT - time step length for integration

NTS - number of time steps to be taken

NTSIF - interval for calculating stress-intensity factors
stress-intensity factors calculated every NTSIF time steps

NTACC - interval for calculating accelerations at selected (by LOAD data) nodes
accelerations are calculated every NTACC time steps

3.2.3 CRAKD - Sample Problem

To execute the computer program CRAKD, an understanding of the related permanent files now existing at ADTC, Eglin AFB, is necessary. Currently the following listed files (with brief description) exist:

FILE	DESCRIPTION
CRAKD	source statements (Fortran) for the finite element program which presently contains the force vector (subroutine FVEC) for the first check problem. This file contains all the statements in one record.
CRAKDAT	contains the data necessary to generate bulk data for two check problems. First card of each data set is title card; Set 1 title is Livermore/Chen Model . . . . Set 2 title is Irvin Model . . . .
GENER	source statements (Fortran) for the bulk data generator program--uses any data set from CRAKDAT to generate model bulk data.
GENXQ	binary form, i.e., compiled form of GENER. Since little or no modification of GENER is necessary, Fortran compilation is not needed each time GENER is used.

MODEL	complete finite-element model which is input to CRAKD--it now contains (CYCLE = 1) the complete model for first set of data in CRAKDAT, i.e., Livermore/ Chen Model . . . .
CRAKRUN	file for submitting remote BATCH job.

The general procedure to follow in conceiving, constructing, and analyzing a finite-element model is as follows:

Step 1. The user edits, appends, or replaces CRAKDAT so that it contains the data required by GEN XQ to generate bulk data for the desired model (to construct such a data set, see paragraph 3.1).

Step 2. The user executes GENXQ using as input the data set in CRAKDAT. The local file, MODEL, will be created by GENXQ and will contain the bulk data for the desired finite-element model (see subparagraph 3.2.1).

Step 3. The user edits MODEL to complete the finite-element model and to provide analysis parameters. Usually for static analysis the user specifies loads and/or imposed displacements, and for harmonic and transient analysis the user specifies time step, number of time steps, etc. (see subparagraph 3.2.2).

Step 4. The user replaces the previous permanent file MODEL with the new complete MODEL.

Step 5. If a transient analysis is to be performed, then the user will need to replace the force vector contained in

CRAKD with one which matches the model to be analyzed. This means that the final subroutine in CRAKD, i.e., SUBROUTINE FVEC (NF,T,F) must be modified. Currently, the FVEC subroutine is the one required for the data set Livermore/Chen Model . . . Basically, the user is required to write the subroutine FVEC such that it will place the correct time dependent forces, F, onto the proper displacement degrees of freedom, NF, at the specified time T. The flexibility gained by this procedure allows the user to specify any kind of load functions, e.g., constant, pulse, ramp, step, tabular, etc. The example following Step 7, below, illustrates how to write a force vector for the second set of data in CRAKDAT, i.e., Irwin Model.

Step 6. If the user modifies FVEC in Step 5, then the updated CRAKD continuing the new FVEC must replace the previous CRAKD.

Step 7. The user executes CRAKD using MODEL as input by submitting a remote Batch job.

3.2.3.1. Sample Problem - Irwin's Problem

To illustrate 2 steps 1 → 7 of subparagraph 3.2.3, the second set of data in CRAKDAT will be used. The model as conceived is shown on Figure 5, and the data needed by GENXQ to generate the model is given in Table 1. The file commands, and an explanation of the purpose of each command follows for the sequence, Step 1 through Step 7.

	Command	Explanation
Step 1:	(a) ATTACH, CRAKDAT, ID=E6	The user gets the file CRAKDAT for modification--i.e., the user wants to establish a file with only the Irwin data.
	(b) EDITOR	The user begins to edit the data
	(c) EDIT, CRAKDAT, S	set for input to data generator
	(d) LIST,/IRWIN/	GENXQ lists line number where Irwin model data generator set begins, i.e., LN
	(e) DELETE, 10, LN1	LN1-LN-10 (lines sequenced by 10's)
	(f) SAVE, DATAGEN	edited file which contains only Irwin data saved under the local name DATAGEN (see Table 1).
	(g) REWIND, DATAGEN	
Step 2:	(a) ATTACH, GENXQ, ID=E6	The user gets the compiled form of GENER
	(b) GENXQ, DATAGEN	The user executes the data generator program using DATAGEN as input. GENXQ creates file MODEL (see Table 2).
Step 3:	(a) EDIT, MODEL, S	The user modifies and completes the file MODEL to make it suitable for analysis by CRAKD.

	(b) LIST, L	The user determines the last line and number, LN, of lines of the data set. After this line add the lines below in (c). See instruction set: CRAKD - Input Data; entries 23, 24, 25, 26 and additional data for Transient Analysis (IDYNAM=2).
	(c) LN+10=0000	LN+10 equals last line number listed in (b) + 10.0 is a blank space. The 0 (zero) means no specified initial displacement the 0 (zero) no specified initial velocities.
	LN+20=0000	
	LN+30=1.0E-05,60	time step and number of time steps
	LN+40=NO	No, do not want to change time step
	LN+50=1	calculate stress-intensity factor every time step.
	LN+60=NO	No, do not want to change stress-intensity factor calculation frequency
	(d) SAVE, MODELI	The user saves the complete Irwin edited Model under the name MODELI.
Step 4:	(a) CATALOG, MODELI, MODEL, ID=E6	The user replaces the Chen model with the Irwin model.
Step 5:	(a) ATTACH, CRAKD, ID=E6	The user gets CRAKD to modify the force vector subroutine FVEC for analysis of the Irwin model.
	(b) EDIT, CRAKD, S	

(c) LIST,/SUBROUTINE FVEC/ User determines the line number where FVEC begins, i.e., line LNFVEC.

(d) The present subroutine in CRAKD is as follows:

```
      SUBROUTINE FVEC (NF,T,F)
      DIMENSION F(1)
      COMMON NNODES, NTRIS, NSPRNG, NCRK8, NCRK10, NLUMPS,
     *       NLDNDS, NRSTR, NER, NRR, NID,IFLAG, NG, NDOF


      DO 10 I = 1, NER
   10 F(I)= 3.2E-04
      F(139)= 3.2E-04
      F(140)= 6.4E-04
      F(142)= 6.4E-04
      F(144)= 6.4E-04
      F(146)= 6.4E-04
      F(148)= 7.2E-04
      F(150)= 4.0E-04
      CONTINUE
      RETURN
      END
```

The lines beginning at

```
F(139)= 3.2E-04
```

and going through the line

```
F(150)= 4.0E-04
```

are to be replaced with the following two lines

```
F(306)= (2.75E07)*T

IF (T.GT. 1.0E-04) F(306)=

    2.013E03 + (0.737E07)*T
```

This produces a subroutine which will place the force P/2 (see Figure 7) onto the node where the hammer strikes (see Figure 5).

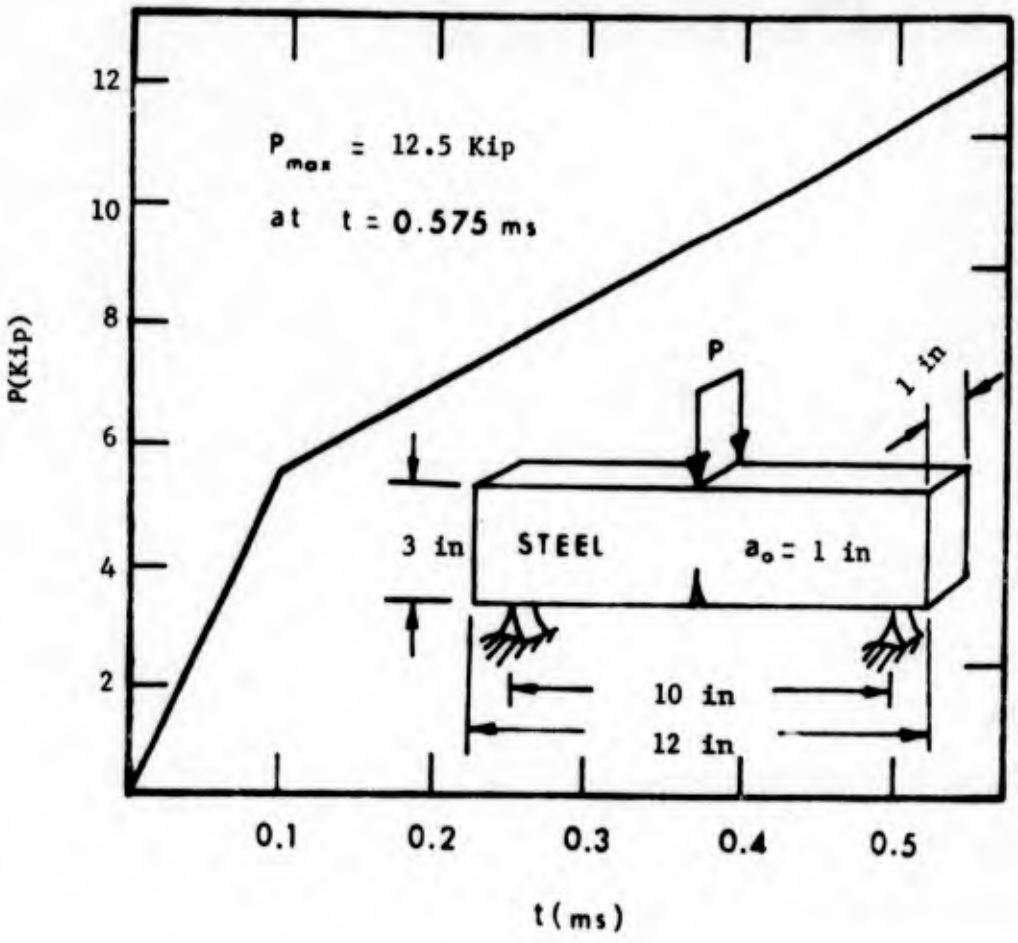


Figure 7. Load-Time Curve Used to Construct FVEC for Analysis of the Irwin Test

Step 6:	(a) SAVE, CRAKD1,	The user saves local edited film under the name CRAKD1.
	(b) CATALOG, CRAKD1, CRAKD,	User replaces CRAKD for analysis of the Irwin problem.
	(c) RETURN, CRAKD, MODEL	
Step 7:	(a) ATTACH, RUN, CRAKRUN, ID=E6	The user gets the file containing statements necessary for submitting a BATCH job (refer to instruction set titled: Instruction to execute (at time-sharing terminal) CRAKD at Armament Laboratory.
	(b) REWIND, RUN	
	(c) BATCH, RUN, INPUT, HERE	The user submits a remote Batch job for execution of CRAKD using Irwin model.

The results of the remote Batch submittal is given in Table 3.

TABLE 3. CRAKD RESULTS FOR IRWIN PROBLEM

```
1

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

   IRWIN MODEL WITH SINGLE POINT LOAD    A0 = 1.0 IN.  2/6/76   JAA

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

                    TRANSIENT LUMPED-MASS ANALYSIS

          MODEL GEOMETRY AND RESTRAINT SPECIFICATIONS
                  ARE ECHOED ON  FILE  21

          TRIANGULAR ELEMENT PROPERTIES ARE ECHOED ON FILE  21

          BANDWIDTH AFTER TRIANGULAR ELEMENTS INPUT IS  30

          8-NODE ELEMENT PROPERTIES ARE ECHOED ON FILE  21

          BANDWIDTH AFTER 8-NODE ELEMENTS INPUT IS....  30

                  MODEL INPUT SPECIFICATIONS
             NUMBER OF NODES.........................163
             NUMBER OF TRIANGULAR ELEMENTS...........273
             TRIANGULAR ELEMENTS ARE IN..... PLANE STRESS
             NUMBER OF 8-NODE CRACKED ELEMENTS.......  1
             NUMBER OF RIGID RESTRAINTS..............  8
             NUMBER OF TRANSLATIONAL FREEDOMS PER NODE  2
             TOTAL DEGREES OF FREEDOM IN MODEL.......318

                  COMPUTED  BANDWIDTH    =   30
                  AVAILABLE BANDWIDTH    =   38
                  REQUIRED  CORE STORAGE = 24534
                  AVAILABLE CORE STORAGE = 30000

          ENTER:  DELTA T, NUMBER OF TIME STEPS

          THE PARAMETERS ARE: DT =   .10000000E-04  NTS =    60

          IF YOU WISH TO CORRECT THIS,TYPE: YES

                                                                    76
          HOW OFTEN MUST I CALCULATE S.I. FACTORS?
```

TABLE 3. CRAKD RESULTS FOR IRWIN PROBLEM (CONTINUED)

```
 S.I. FACTOR WILL BE CALCULATED EVERY      1  TIME STEPS

 IF YOU WISH TO CORRECT THIS,TYPE: YES

 TIME STEP      1       T =  1.000E-05
    8-NODE ELT      1 K-ONE =  2.847E+00

 TIME STEP      2       T =  2.000E-05
    8-NODE ELT      1 K-ONE =  3.766E+01

 TIME STEP      3       T =  3.000E-05
    8-NODE ELT      1 K-ONE =  1.737E+02

 TIME STEP      4       T =  4.000E-05
    8-NODE ELT      1 K-ONE =  4.552E+02

 TIME STEP      5       T =  5.000E-05
    8-NODE ELT      1 K-ONE =  8.563E+02

 TIME STEP      6       T =  6.000E-05
    8-NODE ELT      1 K-ONE =  1.335E+03

 TIME STEP      7       T =  7.000E-05
    8-NODE ELT      1 K-ONE =  1.890E+03

 TIME STEP      8       T =  8.000E-05
    8-NODE ELT      1 K-ONE =  2.580E+03

 TIME STEP      9       T =  9.000E-05
    8-NODE ELT      1 K-ONE =  3.438E+03

 TIME STEP     10       T =  1.000E-04
    8-NODE ELT      1 K-ONE =  4.409E+03

 TIME STEP     11       T =  1.100E-04
    8-NODE ELT      1 K-ONE =  5.497E+03

 TIME STEP     12       T =  1.200E-04
    8-NODE ELT      1 K-ONE =  6.766E+03

 TIME STEP     13       T =  1.300E-04
    8-NODE ELT      1 K-ONE =  8.176E+03

 TIME STEP     14       T =  1.400E-04
    8-NODE ELT      1 K-ONE =  9.638E+03

 TIME STEP     15       T =  1.500E-04
    8-NODE ELT      1 K-ONE =  1.114E+04

 TIME STEP     16       T =  1.600E-04
    8-NODE ELT      1 K-ONE =  1.277E+04

 TIME STEP     17       T =  1.700E-04
    8-NODE ELT      1 K-ONE =  1.457E+04

 TIME STEP     18       T =  1.800E-04
    8-NODE ELT      1 K-ONE =  1.648E+04

 TIME STEP     19       T =  1.900E-04
    8-NODE ELT      1 K-ONE =  1.843E+04

 TIME STEP     20       T =  2.000E-04
```

TABLE 3. CRAKD RESULTS FOR IRWIN PROBLEM (CONTINUED)

```
   8-NODE ELT     1 K-ONE =  2.049E+04

 TIME STEP    21      T =  2.100E-04
   8-NODE ELT     1 K-ONE =  2.264E+04

 TIME STEP    22      T =  2.200E-04
   8-NODE ELT     1 K-ONE =  2.472E+04

 TIME STEP    23      T =  2.300E-04
   8-NODE ELT     1 K-ONE =  2.667E+04

 TIME STEP    24      T =  2.400E-04
   8-NODE ELT     1 K-ONE =  2.855E+04

 TIME STEP    25      T =  2.500E-04
   8-NODE ELT     1 K-ONE =  3.043E+04

 TIME STEP    26      T =  2.600E-04
   8-NODE ELT     1 K-ONE =  3.220E+04

 TIME STEP    27      T =  2.700E-04
   8-NODE ELT     1 K-ONE =  3.379E+04

 TIME STEP    28      T =  2.800E-04
   8-NODE ELT     1 K-ONE =  3.528E+04

 TIME STEP    29      T =  2.900E-04
   8-NODE ELT     1 K-ONE =  3.664E+04

 TIME STEP    30      T =  3.000E-04
   8-NODE ELT     1 K-ONE =  3.778E+04

 TIME STEP    31      T =  3.100E-04
   8-NODE ELT     1 K-ONE =  3.876E+04

 TIME STEP    32      T =  3.200E-04
   8-NODE ELT     1 K-ONE =  3.959E+04

 TIME STEP    33      T =  3.300E-04
   8-NODE ELT     1 K-ONE =  4.025E+04

 TIME STEP    34      T =  3.400E-04
   8-NODE ELT     1 K-ONE =  4.081E+04

 TIME STEP    35      T =  3.500E-04
   8-NODE ELT     1 K-ONE =  4.125E+04

 TIME STEP    36      T =  3.600E-04
   8-NODE ELT     1 K-ONE =  4.152E+04

 TIME STEP    37      T =  3.700E-04
   8-NODE ELT     1 K-ONE =  4.162E+04

 TIME STEP    38      T =  3.800E-04
   8-NODE ELT     1 K-ONE =  4.152E+04

 TIME STEP    39      T =  3.900E-04
   8-NODE ELT     1 K-ONE =  4.118E+04

 TIME STEP    40      T =  4.000E-04
   8-NODE ELT     1 K-ONE =  4.067E+04

 TIME STEP    41      T =  4.100E-04
   8-NODE ELT     1 K-ONE =  3.993E+04
```

TABLE 3. CRAKD RESULTS FOR IRWIN PROBLEM (CONCLUDED)

```
TIME STEP     42      T =  4.200E-04
   8-NODE ELT     1 K-ONE =  3.911E+04

TIME STEP     43      T =  4.300E-04
   8-NODE ELT     1 K-ONE =  3.826E+04

TIME STEP     44      T =  4.400E-04
   8-NODE ELT     1 K-ONE =  3.726E+04

TIME STEP     45      T =  4.500E-04
   8-NODE ELT     1 K-ONE =  3.617E+04

TIME STEP     46      T =  4.600E-04
   8-NODE ELT     1 K-ONE =  3.512E+04

TIME STEP     47      T =  4.700E-04
   8-NODE ELT     1 K-ONE =  3.403E+04

TIME STEP     48      T =  4.800E-04
   8-NODE ELT     1 K-ONE =  3.292E+04

TIME STEP     49      T =  4.900E-04
   8-NODE ELT     1 K-ONE =  3.185E+04

TIME STEP     50      T =  5.000E-04
   8-NODE ELT     1 K-ONE =  3.086E+04

TIME STEP     51      T =  5.100E-04
   8-NODE ELT     1 K-ONE =  2.998E+04

TIME STEP     52      T =  5.200E-04
   8-NODE ELT     1 K-ONE =  2.912E+04

TIME STEP     53      T =  5.300E-04
   8-NODE ELT     1 K-ONE =  2.832E+04

TIME STEP     54      T =  5.400E-04
   8-NODE ELT     1 K-ONE =  2.762E+04

TIME STEP     55      T =  5.500E-04
   8-NODE ELT     1 K-ONE =  2.696E+04

TIME STEP     56      T =  5.600E-04
   8-NODE ELT     1 K-ONE =  2.635E+04

TIME STEP     57      T =  5.700E-04
   8-NODE ELT     1 K-ONE =  2.590E+04

TIME STEP     58      T =  5.800E-04
   8-NODE ELT     1 K-ONE =  2.568E+04

TIME STEP     59      T =  5.900E-04
   8-NODE ELT     1 K-ONE =  2.571E+04

TIME STEP     60      T =  6.000E-04
   8-NODE ELT     1 K-ONE =  2.592E+04
```

3.2.3.2 SAMPLE PROBLEM - CHEN'S PROBLEM

The first data set contained in CRAKDAT (Table 4) can be used to generate the finite-element model of the problem shown on Figure 8. Input of the data in Table 4 to GENER will produce a file, MODEL, which, when edited for use by CRAKD, is given in Table 5. The force vector FVEC in CRAKD is changed to the one given in Table 6. Use of MODEL (Table 5) and CRAKD with the Chen FVEC (Table 6) produces the results shown in Table 7. For a discussion of this problem and lumped-mass variations, see Section III, Volume I.

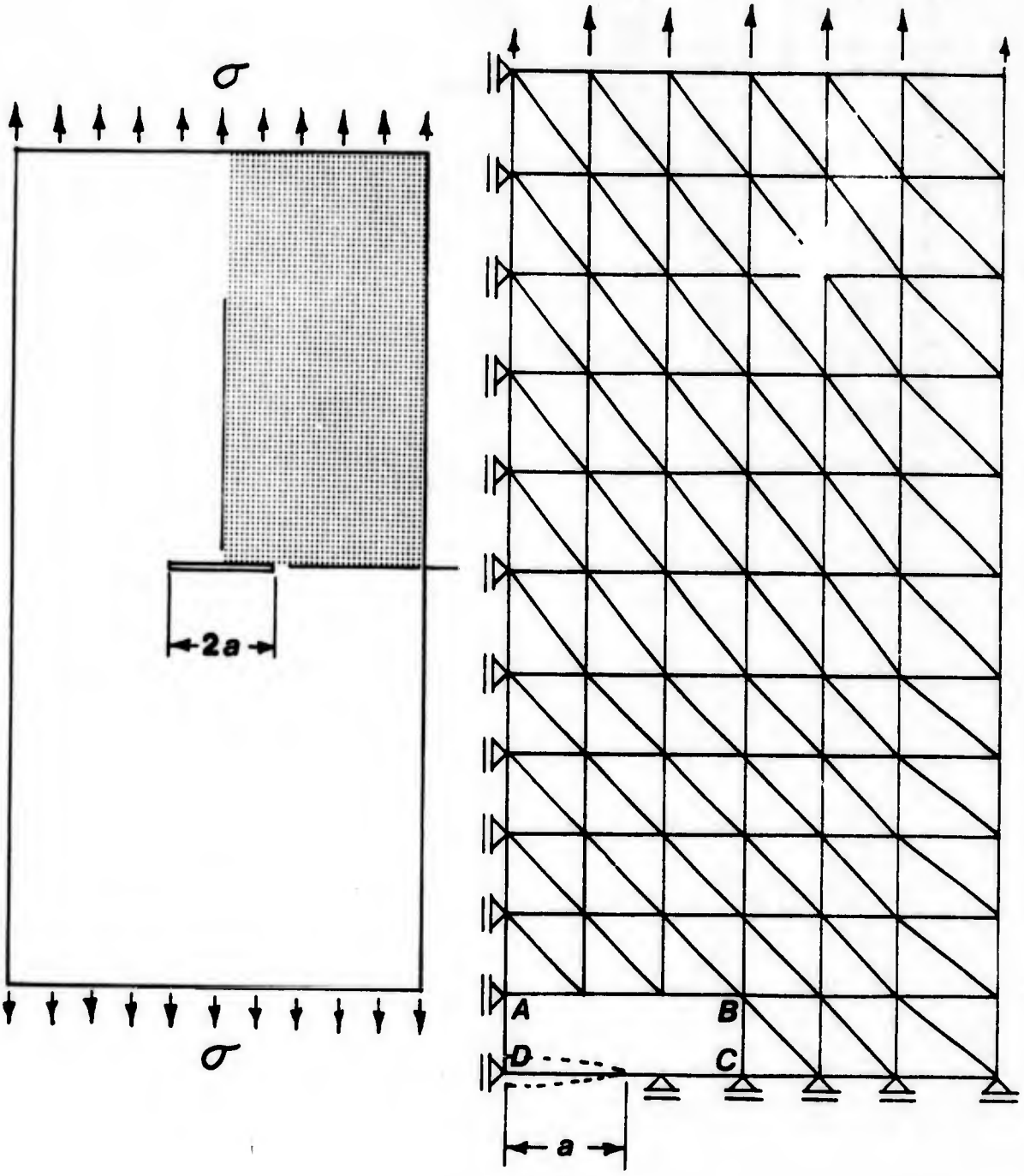


Figure 8. Chen's Problem and Finite-Element Model

TABLE 4. DATA FILE TO INPUT TO GENXQ TO GENERATE FILE MODEL FOR CHEN PROBLEM

```
LIVERMORE/CHEN MODEL   UNITS LENGTH=CM UNITS STRESS=MBAR  1/29/76  JAA
   2     1     0     1     1     0     1     0     0
  10     1     3    21    22
TRANSIENT
   1
  -1     5     1     1     0.0         4.0
   7                        0.0        25.0
  -1     5     7     7     4.0
  -1     6     7     7     5.0
   0                        0.04        0.04
   0
   1     0     1     0    11     7
   3     1     0     0     4     1
   0
   1     4    11     5
   2    12     5    11
  -1     1     1     1     2     2     2
   7     8    15     9
   8    16     9    15
  -1     1     1     1     5     2     2
  -1     7     7     7     9    12    12
   0
   1   125     2.0         0.3         1.0         5.0E-12
   0
   1     4    11    10     9     8     1     2     3     0
   2.0         0.3         1.0         5.0E-12
   0
```

TABLE 5. EDITED MODEL FOR CHEN PROBLEM TO INPUT TO CRAKD

```
LIVERMORE/CHEN MODEL    UNITS LENGTH=CM UNITS STRESS=MBAR  1/29/76  JAA
  84  126     0     1     0     0     0    17     2     0
   1                      3                21    22
TLMASS
    1     0     1          0.0000     0.0000
    2     0     0          0.0000      .1600
    3     1     0          0.0000      .3200
    4     1     0          0.0000      .4800
    5     1     0          0.0000      .6400
    6     1     0          0.0000      .8000
    7     1     0          0.0000     1.0000
    8     0     1           .1600     0.0000
    9     0     0           .1600      .1600
   10     0     0           .1600      .3200
   11     0     0           .1600      .4800
   12     0     0           .1600      .6400
   13     0     0           .1600      .8000
   14     0     0           .1600     1.0000
   15     0     1           .3200     0.0000
   16     0     0           .3200      .1600
   17     0     0           .3200      .3200
   18     0     0           .3200      .4800
   19     0     0           .3200      .6400
   20     0     0           .3200      .8000
   21     0     0           .3200     1.0000
   22     0     1           .4800     0.0000
   23     0     0           .4800      .1600
   24     0     0           .4800      .3200
   25     0     0           .4800      .4800
   26     0     0           .4800      .6400
   27     0     0           .4800      .8000
   28     0     0           .4800     1.0000
   29     0     1           .6400     0.0000
   30     0     0           .6400      .1600
   31     0     0           .6400      .3200
   32     0     0           .6400      .4800
   33     0     0           .6400      .6400
   34     0     0           .6400      .8000
   35     0     0           .6400     1.0000
   36     0     1           .8000     0.0000
   37     0     0           .8000      .1600
   38     0     0           .8000      .3200
   39     0     0           .8000      .4800
   40     0     0           .8000      .6400
   41     0     0           .8000      .8000
   42     0     0           .8000     1.0000
   43     0     1          1.0000     0.0000
   44     0     0          1.0000      .1600
   45     0     0          1.0000      .3200
   46     0     0          1.0000      .4800
   47     0     0          1.0000      .6400
   48     0     0          1.0000      .8000
   49     0     0          1.0000     1.0000
   50     0     1          1.2000     0.0000
   51     0     0          1.2000      .1600
   52     0     0          1.2000      .3200
   53     0     0          1.2000      .4800
   54     0     0          1.2000      .6400
   55     0     0          1.2000      .8000
   56     0     0          1.2000     1.0000
   57     0     1          1.4000     0.0000
   58     0     0          1.4000      .1600
   59     0     0          1.4000      .3200
   60     0     0          1.4000      .4800
```

TABLE 5. EDITED MODEL FOR CHEN PROBLEM TO INPUT TO CRAKD (CONTINUED)

61	0	0	1.4000	.6400	
62	0	0	1.4000	.8000	
63	0	0	1.4000	1.0000	
64	0	1	1.6000	0.0000	
65	0	0	1.6000	.1600	
66	0	0	1.6000	.3200	
67	0	0	1.6000	.4800	
68	0	0	1.6000	.6400	
69	0	0	1.6000	.8000	
70	0	0	1.6000	1.0000	
71	0	1	1.8000	0.0000	
72	0	0	1.8000	.1600	
73	0	0	1.8000	.3200	
74	0	0	1.8000	.4800	
75	0	0	1.8000	.6400	
76	0	0	1.8000	.8000	
77	0	0	1.8000	1.0000	
78	0	1	2.0000	0.0000	
79	0	0	2.0000	.1600	
80	0	0	2.0000	.3200	
81	0	0	2.0000	.4800	
82	0	0	2.0000	.6400	
83	0	0	2.0000	.8000	
84	0	0	2.0000	1.0000	
0					
1	4	11	5	2.0	.30000 1.00000 .5000E-11
2	12	5	11	2.0	.30000 1.00000 .5000E-11
3	5	12	6	2.0	.30000 1.00000 .5000E-11
4	13	6	12	2.0	.30000 1.00000 .5000E-11
5	6	13	7	2.0	.30000 1.00000 .5000E-11
6	14	7	13	2.0	.30000 1.00000 .5000E-11
7	8	15	9	2.0	.30000 1.00000 .5000E-11
8	16	9	15	2.0	.30000 1.00000 .5000E-11
9	9	16	10	2.0	.30000 1.00000 .5000E-11
10	17	10	16	2.0	.30000 1.00000 .5000E-11
11	10	17	11	2.0	.30000 1.00000 .5000E-11
12	18	11	17	2.0	.30000 1.00000 .5000E-11
13	11	18	12	2.0	.30000 1.00000 .5000E-11
14	19	12	18	2.0	.30000 1.00000 .5000E-11
15	12	19	13	2.0	.30000 1.00000 .5000E-11
16	20	13	19	2.0	.30000 1.00000 .5000E-11
17	13	20	14	2.0	.30000 1.00000 .5000E-11
18	21	14	20	2.0	.30000 1.00000 .5000E-11
19	15	22	16	2.0	.30000 1.00000 .5000E-11
20	23	16	22	2.0	.30000 1.00000 .5000E-11
21	16	23	17	2.0	.30000 1.00000 .5000E-11
22	24	17	23	2.0	.30000 1.00000 .5000E-11
23	17	24	18	2.0	.30000 1.00000 .5000E-11
24	25	18	24	2.0	.30000 1.00000 .5000E-11
25	18	25	19	2.0	.30000 1.00000 .5000E-11
26	26	19	25	2.0	.30000 1.00000 .5000E-11
27	19	26	20	2.0	.30000 1.00000 .5000E-11
28	27	20	26	2.0	.30000 1.00000 .5000E-11
29	20	27	21	2.0	.30000 1.00000 .5000E-11
30	28	21	27	2.0	.30000 1.00000 .5000E-11
31	22	29	23	2.0	.30000 1.00000 .5000E-11
32	30	23	29	2.0	.30000 1.00000 .5000E-11
33	23	30	24	2.0	.30000 1.00000 .5000E-11
34	31	24	30	2.0	.30000 1.00000 .5000E-11
35	24	31	25	2.0	.30000 1.00000 .5000E-11
36	32	25	31	2.0	.30000 1.00000 .5000E-11
37	25	32	26	2.0	.30000 1.00000 .5000E-11
38	33	26	32	2.0	.30000 1.00000 .5000E-11
39	26	33	27	2.0	.30000 1.00000 .5000E-11

TABLE 5. EDITED MODEL FOR CHEN PROBLEM TO INPUT TO CRAKD (CONTINUED)

40	34	27	33	2.0	.30000	1.00000	.5000E-11
41	27	34	28	2.0	.30000	1.00000	.5000E-11
42	35	28	34	2.0	.30000	1.00000	.5000E-11
43	29	36	30	2.0	.30000	1.00000	.5000E-11
44	37	30	36	2.0	.30000	1.00000	.5000E-11
45	30	37	31	2.0	.30000	1.00000	.5000E-11
46	38	31	37	2.0	.30000	1.00000	.5000E-11
47	31	38	32	2.0	.30000	1.00000	.5000E-11
48	39	32	38	2.0	.30000	1.00000	.5000E-11
49	32	39	33	2.0	.30000	1.00000	.5000E-11
50	40	33	39	2.0	.30000	1.00000	.5000E-11
51	33	40	34	2.0	.30000	1.00000	.5000E-11
52	41	34	40	2.0	.30000	1.00000	.5000E-11
53	34	41	35	2.0	.30000	1.00000	.5000E-11
54	42	35	41	2.0	.30000	1.00000	.5000E-11
55	36	43	37	2.0	.30000	1.00000	.5000E-11
56	44	37	43	2.0	.30000	1.00000	.5000E-11
57	37	44	38	2.0	.30000	1.00000	.5000E-11
58	45	38	44	2.0	.30000	1.00000	.5000E-11
59	38	45	39	2.0	.30000	1.00000	.5000E-11
60	46	39	45	2.0	.30000	1.00000	.5000E-11
61	39	46	40	2.0	.30000	1.00000	.5000E-11
62	47	40	46	2.0	.30000	1.00000	.5000E-11
63	40	47	41	2.0	.30000	1.00000	.5000E-11
64	48	41	47	2.0	.30000	1.00000	.5000E-11
65	41	48	42	2.0	.30000	1.00000	.5000E-11
66	49	42	48	2.0	.30000	1.00000	.5000E-11
67	43	50	44	2.0	.30000	1.00000	.5000E-11
68	51	44	50	2.0	.30000	1.00000	.5000E-11
69	44	51	45	2.0	.30000	1.00000	.5000E-11
70	52	45	51	2.0	.30000	1.00000	.5000E-11
71	45	52	46	2.0	.30000	1.00000	.5000E-11
72	53	46	52	2.0	.30000	1.00000	.5000E-11
73	46	53	47	2.0	.30000	1.00000	.5000E-11
74	54	47	53	2.0	.30000	1.00000	.5000E-11
75	47	54	48	2.0	.30000	1.00000	.5000E-11
76	55	48	54	2.0	.30000	1.00000	.5000E-11
77	48	55	49	2.0	.30000	1.00000	.5000E-11
78	56	49	55	2.0	.30000	1.00000	.5000E-11
79	50	57	51	2.0	.30000	1.00000	.5000E-11
80	58	51	57	2.0	.30000	1.00000	.5000E-11
81	51	58	52	2.0	.30000	1.00000	.5000E-11
82	59	52	58	2.0	.30000	1.00000	.5000E-11
83	52	59	53	2.0	.30000	1.00000	.5000E-11
84	60	53	59	2.0	.30000	1.00000	.5000E-11
85	53	60	54	2.0	.30000	1.00000	.5000E-11
86	61	54	60	2.0	.30000	1.00000	.5000E-11
87	54	61	55	2.0	.30000	1.00000	.5000E-11
88	62	55	61	2.0	.30000	1.00000	.5000E-11
89	55	62	56	2.0	.30000	1.00000	.5000E-11
90	63	56	62	2.0	.30000	1.00000	.5000E-11
91	57	64	58	2.0	.30000	1.00000	.5000E-11
92	65	58	64	2.0	.30000	1.00000	.5000E-11
93	58	65	59	2.0	.30000	1.00000	.5000E-11
94	66	59	65	2.0	.30000	1.00000	.5000E-11
95	59	66	60	2.0	.30000	1.00000	.5000E-11
96	67	60	66	2.0	.30000	1.00000	.5000E-11
97	60	67	61	2.0	.30000	1.00000	.5000E-11
98	68	61	67	2.0	.30000	1.00000	.5000E-11
99	61	68	62	2.0	.30000	1.00000	.5000E-11
100	69	62	68	2.0	.30000	1.00000	.5000E-11
101	62	69	63	2.0	.30000	1.00000	.5000E-11
102	70	63	69	2.0	.30000	1.00000	.5000E-11
103	64	71	65	2.0	.30000	1.00000	.5000E-11

TABLE 5. EDITED MODEL FOR CHEN PROBLEM TO INPUT TO CRAKD (CONCLUDED)

```
104    72    65    71          2.0      .30000    1.00000  .5000E-11
105    65    72    66          2.0      .30000    1.00000  .5000E-11
106    73    66    72          2.0      .30000    1.00000  .5000E-11
107    66    73    67          2.0      .30000    1.00000  .5000E-11
108    74    67    73          2.0      .30000    1.00000  .5000E-11
109    67    74    68          2.0      .30000    1.00000  .5000E-11
110    75    68    74          2.0      .30000    1.00000  .5000E-11
111    68    75    69          2.0      .30000    1.00000  .5000E-11
112    76    69    75          2.0      .30000    1.00000  .5000E-11
113    69    76    70          2.0      .30000    1.00000  .5000E-11
114    77    70    76          2.0      .30000    1.00000  .5000E-11
115    71    78    72          2.0      .30000    1.00000  .5000E-11
116    79    72    78          2.0      .30000    1.00000  .5000E-11
117    72    79    73          2.0      .30000    1.00000  .5000E-11
118    80    73    79          2.0      .30000    1.00000  .5000E-11
119    73    80    74          2.0      .30000    1.00000  .5000E-11
120    81    74    80          2.0      .30000    1.00000  .5000E-11
121    74    81    75          2.0      .30000    1.00000  .5000E-11
122    82    75    81          2.0      .30000    1.00000  .5000E-11
123    75    82    76          2.0      .30000    1.00000  .5000E-11
124    83    76    82          2.0      .30000    1.00000  .5000E-11
125    76    83    77          2.0      .30000    1.00000  .5000E-11
126    84    77    83          2.0      .30000    1.00000  .5000E-11
  0
  1     4    11    10     9     8     1     2     3     0
  1         2.0     .30000   1.01000   5.0E-12
  0
  0
  0
0.1E-07,1500
NO
20
NO
```

TABLE 6. FORCE VECTOR FOR THE CHEN PROBLEM

```
CFVCHEN
      SUBROUTINE FVEC(NF,T,F)
      DIMENSION F(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF
      DO 10 I=1,NER
   10 F(I)=0.0
      F(139)=3.2E-04
      F(140)=6.4E-04
      F(142)=6.4E-04
      F(144)=6.4E-04
      F(146)=6.4E-04
      F(148)=7.2E-04
      F(150)=4.0E-04
      CONTINUE
      RETURN
      END
```

1

TABLE 7. CRAKD RESULTS FOR THE CHEN PROBLEM

```
**************************************************************************

  LIVERMORE/CHEN MODEL   UNITS LENGTH=CM UNITS STRESS=MBAR  1/29/76  JAA

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

                        TRANSIENT LUMPED-MASS ANALYSIS


          MODEL GEOMETRY AND RESTRAINT SPECIFICATIONS
                    ARE ECHOED ON  FILE  21

          TRIANGULAR ELEMENT PROPERTIES ARE ECHOED ON FILE  21

          BANDWIDTH AFTER TRIANGULAR ELEMENTS INPUT IS  15

          8-NODE ELEMENT PROPERTIES ARE ECHOED ON FILE  21

          BANDWIDTH AFTER 8-NODE ELEMENTS INPUT IS....  15

                    MODEL INPUT SPECIFICATIONS
               NUMBER OF NODES.......................... 84
               NUMBER OF TRIANGULAR ELEMENTS.............126
               TRIANGULAR ELEMENTS ARE IN..... PLANE STRAIN
               NUMBER OF 8-NODE CRACKED ELEMENTS.........  1
               NUMBER OF RIGID RESTRAINTS................. 17
               NUMBER OF TRANSLATIONAL FREEDOMS PER NODE  2
               TOTAL DEGREES OF FREEDOM IN MODEL.........151

                    COMPUTED  BANDWIDTH     =   15
                    AVAILABLE BANDWIDTH     =   40
                    REQUIRED  CORE STORAGE  =  6196
                    AVAILABLE CORE STORAGE  = 10000
ENTERED DYNAM

          ENTER%  DELTA T, NUMBER OF TIME STEPS

          THE PARAMETERS ARE% DT =   .1000000E-07  NTS =  1500

          IF YOU WISH TO CORRECT THIS,TYPE% YES

          HOW OFTEN MUST I CALCULATE S.I. FACTORS#

          S.I. FACTOR WILL BE CALCULATED EVERY    20  TIME STEPS
```

TABLE 7. CRAKD RESULTS FOR THE CHEN PROBLEM (CONTINUED)

```
IF YOU WISH TO CORRECT THIS,TYPE% YES

TIME STEP     1     T =  0.
   8-NODE ELT    1 K-ONE =  0.

TIME STEP    20     T =  1.900E-07
   8-NODE ELT    1 K-ONE = -2.365E-27

TIME STEP    40     T =  3.900E-07
   8-NODE ELT    1 K-ONE =  9.196E-21

TIME STEP    60     T =  5.900E-07
   8-NODE ELT    1 K-ONE =  1.572E-16

TIME STEP    80     T =  7.900E-07
   8-NODE ELT    1 K-ONE =  1.082E-13

TIME STEP   100     T =  9.900E-07
   8-NODE ELT    1 K-ONE =  1.475E-11

TIME STEP   120     T =  1.190E-06
   8-NODE ELT    1 K-ONE =  7.180E-10

TIME STEP   140     T =  1.390E-06
   8-NODE ELT    1 K-ONE =  1.687E-08

TIME STEP   160     T =  1.590E-06
   8-NODE ELT    1 K-ONE =  2.288E-07

TIME STEP   180     T =  1.790E-06
   8-NODE ELT    1 K-ONE =  2.012E-06

TIME STEP   200     T =  1.990E-06
   8-NODE ELT    1 K-ONE =  1.244E-05

TIME STEP   220     T =  2.190E-06
   8-NODE ELT    1 K-ONE =  5.719E-05

TIME STEP   240     T =  2.390E-06
   8-NODE ELT    1 K-ONE =  2.035E-04

TIME STEP   260     T =  2.590E-06
   8-NODE ELT    1 K-ONE =  5.766E-04

TIME STEP   280     T =  2.790E-06
   8-NODE ELT    1 K-ONE =  1.327E-03

TIME STEP   300     T =  2.990E-06
   8-NODE ELT    1 K-ONE =  2.515E-03

TIME STEP   320     T =  3.190E-06
   8-NODE ELT    1 K-ONE =  3.967E-03

TIME STEP   340     T =  3.390E-06
   8-NODE ELT    1 K-ONE =  5.261E-03

TIME STEP   360     T =  3.590E-06
   8-NODE ELT    1 K-ONE =  5.962E-03

TIME STEP   380     T =  3.790E-06
   8-NODE ELT    1 K-ONE =  5.988E-03

TIME STEP   400     T =  3.990E-06
   8-NODE ELT    1 K-ONE =  5.716E-03
```

TABLE 7. CRAKD RESULTS FOR THE CHEN PROBLEM (CONTINUED)

```
TIME STEP   420      T =  4.190E-06
   8-NODE ELT     1 K-ONE =  5.650E-03

TIME STEP   440      T =  4.390E-06
   8-NODE ELT     1 K-ONE =  5.916E-03

TIME STEP   460      T =  4.590E-06
   8-NODE ELT     1 K-ONE =  6.199E-03

TIME STEP   480      T =  4.790E-06
   8-NODE ELT     1 K-ONE =  6.193E-03

TIME STEP   500      T =  4.990E-06
   8-NODE ELT     1 K-ONE =  6.031E-03

TIME STEP   520      T =  5.190E-06
   8-NODE ELT     1 K-ONE =  6.143E-03

TIME STEP   540      T =  5.390E-06
   8-NODE ELT     1 K-ONE =  6.739E-03

TIME STEP   560      T =  5.590E-06
   8-NODE ELT     1 K-ONE =  7.572E-03

TIME STEP   580      T =  5.790E-06
   8-NODE ELT     1 K-ONE =  8.285E-03

TIME STEP   600      T =  5.990E-06
   8-NODE ELT     1 K-ONE =  8.796E-03

TIME STEP   620      T =  6.190E-06
   8-NODE ELT     1 K-ONE =  9.219E-03

TIME STEP   640      T =  6.390E-06
   8-NODE ELT     1 K-ONE =  9.545E-03

TIME STEP   660      T =  6.590E-06
   8-NODE ELT     1 K-ONE =  9.576E-03

TIME STEP   680      T =  6.790E-06
   8-NODE ELT     1 K-ONE =  9.235E-03

TIME STEP   700      T =  6.990E-06
   8-NODE ELT     1 K-ONE =  8.767E-03

TIME STEP   720      T =  7.190E-06
   8-NODE ELT     1 K-ONE =  8.498E-03

TIME STEP   740      T =  7.390E-06
   8-NODE ELT     1 K-ONE =  8.445E-03

TIME STEP   760      T =  7.590E-06
   8-NODE ELT     1 K-ONE =  8.379E-03

TIME STEP   780      T =  7.790E-06
   8-NODE ELT     1 K-ONE =  8.188E-03

TIME STEP   800      T =  7.990E-06
   8-NODE ELT     1 K-ONE =  7.904E-03

TIME STEP   820      T =  8.190E-06
   8-NODE ELT     1 K-ONE =  7.485E-03
```

TABLE 7. CRAKD RESULTS FOR THE CHEN PROBLEM (CONTINUED)

```
TIME STEP   840      T =  8.390E-06
   8-NODE ELT     1 K-ONE =  6.361E-03

TIME STEP   860      T =  8.590E-06
   8-NODE ELT     1 K-ONE =  6.110E-03

TIME STEP   880      T =  8.790E-06
   8-NODE ELT     1 K-ONE =  5.345E-03

TIME STEP   900      T =  8.990E-06
   8-NODE ELT     1 K-ONE =  4.503E-03

TIME STEP   920      T =  9.190E-06
   8-NODE ELT     1 K-ONE =  3.484E-03

TIME STEP   940      T =  9.390E-06
   8-NODE ELT     1 K-ONE =  2.447E-03

TIME STEP   960      T =  9.590E-06
   8-NODE ELT     1 K-ONE =  1.763E-03

TIME STEP   980      T =  9.790E-06
   8-NODE ELT     1 K-ONE =  1.722E-03

TIME STEP  1000      T =  9.990E-06
   8-NODE ELT     1 K-ONE =  2.013E-03

TIME STEP  1020      T =  1.019E-05
   8-NODE ELT     1 K-ONE =  2.183E-03

TIME STEP  1040      T =  1.039E-05
   8-NODE ELT     1 K-ONE =  2.098E-03

TIME STEP  1060      T =  1.059E-05
   8-NODE ELT     1 K-ONE =  1.998E-03

TIME STEP  1080      T =  1.079E-05
   8-NODE ELT     1 K-ONE =  2.047E-03

TIME STEP  1100      T =  1.099E-05
   8-NODE ELT     1 K-ONE =  2.117E-03

TIME STEP  1120      T =  1.119E-05
   8-NODE ELT     1 K-ONE =  1.994E-03

TIME STEP  1140      T =  1.139E-05
   8-NODE ELT     1 K-ONE =  1.607E-03

TIME STEP  1160      T =  1.159E-05
   8-NODE ELT     1 K-ONE =  1.032E-03

TIME STEP  1180      T =  1.179E-05
   8-NODE ELT     1 K-ONE =  3.648E-04

TIME STEP  1200      T =  1.199E-05
   8-NODE ELT     1 K-ONE = -3.000E-04

TIME STEP  1220      T =  1.219E-05
   8-NODE ELT     1 K-ONE = -8.330E-04

TIME STEP  1240      T =  1.239E-05
   8-NODE ELT     1 K-ONE = -1.166E-03

TIME STEP  1260      T =  1.259E-05
```

TABLE 7. CRACK RESULTS FOR THE CHEN PROBLEM (CONCLUDED)

```
   8-NODE ELT     1 K-ONE = -1.326E-03

TIME STEP  1280      T =  1.279E-05
   8-NODE ELT     1 K-ONE = -1.340E-03

TIME STEP  1300      T =  1.299E-05
   8-NODE ELT     1 K-ONE = -1.197E-03

TIME STEP  1320      T =  1.319E-05
   8-NODE ELT     1 K-ONE = -9.252E-04

TIME STEP  1340      T =  1.339E-05
   8-NODE ELT     1 K-ONE = -6.740E-04

TIME STEP  1360      T =  1.359E-05
   8-NODE ELT     1 K-ONE = -6.220E-04

TIME STEP  1380      T =  1.379E-05
   8-NODE ELT     1 K-ONE = -7.325E-04

TIME STEP  1400      T =  1.399E-05
   8-NODE ELT     1 K-ONE = -7.354E-04

TIME STEP  1420      T =  1.419E-05
   8-NODE ELT     1 K-ONE = -4.737E-04

TIME STEP  1440      T =  1.439E-05
   8-NODE ELT     1 K-ONE = -1.082E-04

TIME STEP  1460      T =  1.459E-05
   8-NODE ELT     1 K-ONE =  1.329E-04

TIME STEP  1480      T =  1.479E-05
   8-NODE ELT     1 K-ONE =  2.803E-04
```

1.2.4 CRAKD-PROGRAM LISTING

Following is a complete listing of all Fortran statements for the finite-element analysis program CRAKD and all its associated subprograms:

```
CCRAKD
      PROGRAM CRAKD(INPUT=100,OUTPUT=100,DATA=100,TAPE5=INPUT,
     *TAPE6=OUTPUT,TAPE21=DATA,TAPE10,TAPE25,TAPE26,TAPE28,
     *TAPE29)
C ******* DYNAMIC CRACKED ELEMENT PROGRAM  EGLIN AFB  MARCH 1976 *******
C ******* LUMPED MASS MODIFICATION SEPTEMBER 1976 *******
C *******    PLANE MEMBRANE TRIANGLES AND SPRING ELEMENTS    *******
C *******    EIGHT AND TEN NODE CRACKED ELEMENTS              *******
C ******* J. A. ABERSON 404/894-2775, OR J. M. ANDERSON 404/894-2774, OR
C ******* W. W. KING 404/894-2772 OF SCHOOL OF ENGINEERING SCIENCE AND
C ******* MECHANICS, GEORGIA INSTITUTE OF TECHNOLOGY 225 NORTH AVENUE,
C ******* ATLANTA, GEORGIA 30332  *******
      DIMENSION A(10000),N(16)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      COMMON/ADR/NNF,NJT,NTP,NJS,NSP,NJ8,N8P,NJ10,N10P,NP1,NST,NSM,
     *           NDY,NP2,NUV,NUN
      EQUIVALENCE (NNF,N)
      NKORE=10000
      DO 10 I=1,NKORE
      A(I)=0.0
   10 CONTINUE
      CALL INPUT(A,N)
      CALL FORM(A,N)
      IF(IDYNAM.LE.0) GO TO 400
      CALL DYNAM(A,N)
      GO TO 600
400   CALL OUTPUT(A,N)
600   ENDFILE IOUT1
      REWIND IOUT1
      STOP
      END
```

```
CINPTLM
      SUBROUTINE INPUT(A,NN)
      DIMENSION A(1),NN(16),TITLE(12)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,ICUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DATA ISTRN,ISTRS/6HSTRAIN,6HSTRESS/
      IFLAG=0
      READ(5,600) TITLE
  600 FORMAT(12A6)
      WRITE(6,700) TITLE
  700 FORMAT(1H1,////,72(1H*)//,12A6,//,72(1H*))
      READ(5,100) NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *            NLONDS,NRSTR,NDOF,ISTATE
      READ(5,100) INPRT,IKPRT,ITPRT,ICPRT,IOPRT,IIN8,IIN10,IOUT1,IOUT2,
     *            IOUT3,IOUT4,IOUT5
  100 FORMAT(12I5)
      IDYNAM=0
      TYPE=1H
      READ(5,301) TYPE
  301 FORMAT(A6)
      IF(TYPE.NE.6HSTATIC) GO TO 199
      WRITE(6,1100)
 1100 FORMAT(//32X,15HSTATIC ANALYSIS//)
      IDYNAM=0
      GO TO 302
  199 IF(TYPE.NE.6HHCMASS) GO TO 198
      WRITE(6,1200)
 1200 FORMAT(//23X,33HHARMONIC CONSISTENT-MASS ANALYSIS//)
      IDYNAM=1
      GO TO 302
  198 IF(TYPE.NE.6HHLMASS) GO TO 197
      WRITE(6,1300)
 1300 FORMAT(//25X,29HHARMONIC LUMPED-MASS ANALYSIS//)
      IDYNAM=3
      GO TO 302
  197 IF(TYPE.NE.6HTCMASS) GO TO 196
      WRITE(6,1400)
 1400 FORMAT(//23X,34HTRANSIENT CONSISTENT-MASS ANALYSIS//)
      IDYNAM=2
      GO TO 302
  196 IF(TYPE.NE.6HTLMASS) GO TO 195
      WRITE(6,1500)
 1500 FORMAT(//25X,30HTRANSIENT LUMPED-MASS ANALYSIS//)
      IDYNAM=4
      GO TO 302
  195 WRITE(6,1600)
 1600 FORMAT(/10X,5HERROR,5X,28HTYPE OF ANALYSIS UNSPECIFIED/)
      IFLAG=IFLAG+1
  302 CONTINUE
      REWIND IOUT1
      NN(1)=NDOF*NNODES + 1
      NN(2)=NN(1) + NDOF*NNODES
      NN(3)=NN(2) + 3*NTRIS
      NN(4)=NN(3) + 4*NTRIS
      NN(5)=NN(4) + 4*NSPRNG
      NN(6)=NN(5) + 3*NSPRNG
      NN(7)=NN(6) + 9*NCRK8
      NN(8)=NN(7) + 4*NCRK8
      NN(9)=NN(8) + 11*NCRK10
      NN(10)=NN(9) + 4*NCRK10
      NN(11)=NN(10) + NDOF*NNODES
```

```
C
      NB=0
      N1=NN(1)
      CALL CORIN(A,A(N1),NDOF)
C
      IF (INPRT.GT.0) CALL NODES (A,A(N1),NDOF)
C
      N2=NN(2)
      N3=NN(3)
      IF (NTRIS.GT.0) CALL TRIN (A(N2),A(N3),A(N1),NDOF)
C
      N2=NN(4)
      N3=NN(5)
      IF (NSPRNG.GT.0) CALL SPRGIN (A(N2),A(N3),A(N1),NDOF)
C
      N2=NN(6)
      N3=NN(7)
      IF (NCRK8.GT.0) CALL  CRKIN8 (A(N2),A(N3),A(N1),NDOF)
C
      N2=NN(8)
      N3=NN(9)
      IF (NCRK10.GT.0) CALL CRKN10 (A(N2),A(N3),A(N1),NDOF)
      NN(12)=NN(11)+NB*NER
      NN(13)=NN(12)
      IF(IDYNAM.GT.0) NN(13)=NN(12)+NB*NER
      N2=NN(12)
      IF(IDYNAM.GT.0.AND.NLUMPS.GT.0)CALL LUMPIN(A(N1),A(N2),NDOF,NER)
      N2=NN(10)
      IF(NID.GT.0)CALL DISPIN(A(N1),A(N2),NDOF)
      IF(NLDNDS.GT.0)CALL LOADIN(A(N1),A(N2),NDOF)
      NN(14)=NN(13)
      IF(IDYNAM.EQ.0) GO TO 194
      IF(IDYNAM.EQ.1) NN(14)=NN(13)+NB*NER
      IF(IDYNAM.EQ.2) NN(14)=NN(13)+3*NER
      IF(IDYNAM.EQ.3) NN(14)=NN(13)+NB*NER
      IF(IDYNAM.EQ.4) NN(14)=NN(13)+1*NER
194   NN(15)=NN(14)
      IF(IDYNAM.EQ.1) NN(15)=NN(14)+NDOF*NNODES
      IF(IDYNAM.EQ.2) NN(15)=NN(14)+3*NER
      IF(IDYNAM.EQ.3) NN(15)=NN(14)+NDOF*NNODES
      IF(IDYNAM.EQ.4) NN(15)=NN(14)+3*NER
      NN(16)=NN(15)
      IF(IDYNAM.GT.0) NN(16)=NN(15) + NNODES*NDOF
      IF (ISTATE.EQ.0) ISTAT1=ISTRN
      IF (ISTATE.EQ.1) ISTAT1=ISTRS
      WRITE(6,800)
      IF(NNODES.GT.0) WRITE(6,801) NNODES
      IF(NTRIS.GT.0) WRITE(6,802) NTRIS
      IF(NTRIS.GT.0) WRITE(6,811) ISTAT1
      IF(NSPRNG.GT.0) WRITE(6,803) NSPRNG
      IF(NCRK8.GT.0) WRITE(6,804) NCRK8
      IF(NCRK10.GT.0) WRITE(6,805) NCRK10
      IF(NLUMPS.GT.0) WRITE(6,812) NLUMPS
      IF((NLDNDS.GT.0).AND.(IDYNAM.NE.2.OR.IDYNAM.NE.4))
     +WRITE(6,806) NLDNDS
      IF((NLDNDS.GT.0).AND.(IDYNAM.EQ.2.OR.IDYNAM.EQ.4))
     +WRITE(6,813) NLDNDS
      IF(NRR.GT.0) WRITE(6,807) NRR
      IF(NID.GT.0) WRITE(6,808) NID
      WRITE(6,809) NDOF
      WRITE(6,810) NER
800   FORMAT(//20X,26HMODEL INPUT SPECIFICATIONS)
801   FORMAT(13X,41HNUMBER OF NODES.....,I3)
802   FORMAT(13X,41HNUMBER OF TRIANGULAR ELEMENTS.....,I3)
```

```
  803  FORMAT(13X,41HNUMBER OF TRANSLATIONAL SPRING ELEMENTS..,I3)
  804  FORMAT(13X,41HNUMBER OF 8-NODE CRACKED ELEMENTS.......,I3)
  805  FORMAT(13X,41HNUMBER OF 10-NODE CRACKED ELEMENTS......,I3)
  806  FORMAT(13X,41HNUMBER OF LOADED NODES..................,I3)
  807  FORMAT(13X,41HNUMBER OF RIGID RESTRAINTS..............,I3)
  808  FORMAT(13X,41HNUMBER OF IMPOSED NON-ZERO RESTRAINTS....,I3)
  809  FORMAT(13X,41HNUMBER OF TRANSLATIONAL FREEDOMS PER NODE,I3)
  810  FORMAT(13X,41HTOTAL DEGREES OF FREEDOM IN MODEL........,I3)
  811  FORMAT(13X,38HTRIANGULAR ELEMENTS ARE IN..... PLANE ,A6)
  812  FORMAT(13X,41HNUMBER OF LUMPED MASSES..................,I3)
  813  FORMAT(13X,41HNODAL ACCELERATIONS TO BE CALCULATED.....,I3)
       IF(INPRT.GT.1) WRITE (6,1000) (NN(I),I=1,16)
 1000 FORMAT(//2X,"NF@",I5," JT @",I5," TP @",I5," JS @",I5," SP @",
      *I5," J 8@",I5," 8P @",I5,/1X,"J10@",I5," 10P@",I5,
      *"  P@",I5," ST @",I5," SM @",I5," DYN@",I5," P2 @",I5,
      */1X,"FRE@",I5," END@",I5/)
       MAXKOR=NN(11)-1
       IF(IDYNAM.GT.0) GO TO 133
       MAXKOR=MAXKOR+MAX0((3*NTRIS),(1216+NCRK8),(1900+2*NCRK10))
       N2=(NKORE-MAXKOR)/(NER)
       MAXKOR=MAXKOR+NER*NB
       GO TO 134
  133  IF(IDYNAM.GT.1) GO TO 136
       MAXKOR=MAXKOR+2*NDOF*NNODES
       N2=(NKORE-MAXKOR)/(3*NER)
       MAXKOR=MAXKOR+3*NER*NB
       GO TO 134
  136  IF(IDYNAM.GT.2) GO TO 137
       N1=NDOF*NNODES + 6*NER
       IF(NCRK8.GT.0) N2=MAX0(N1,2432)
       IF(NCRK10.GT.0) N2=MAX0(N1,3800)
       MAXKOR=MAXKOR+N2
       N2=(NKORE-MAXKOR)/(2*NER)
       MAXKOR=MAXKOR+2*NER*NB
       GO TO 134
  137  IF(IDYNAM.GT.3) GO TO 138
       MAXKOR=MAXKOR+2*NDOF*NNODES+NER
       N2=(NKORE-MAXKOR)/(2*NER)
       MAXKOR=MAXKOR+2*NB*NER
       GO TO 134
  138  N1=NDOF*NNODES+6*NER+NER
       IF(NCRK8.GT.0) N2=MAX0(N1,2432)
       IF(NCRK10.GT.0) N2=MAX0(N1,3800)
       MAXKOR=MAXKOR+N2
       N2=(NKORE-MAXKOR)/NER
       MAXKOR=MAXKOR+NB*NER
  134  IF(NKORE-MAXKOR) 132,131,131
  132  WRITE(6,500)
  500  FORMAT(//5X,"***** AVAILABLE STORAGE EXCEEDED *****"//)
       WRITE(6,501) NB,N2,MAXKOR,NKORE
  501  FORMAT(//18X,25HCOMPUTED  BANDWIDTH    = ,I4,
      *       /18X,25HAVAILABLE BANDWIDTH    = ,I4,
      *       /18X,25HREQUIRED  CORE STORAGE = ,I5,
      *       /18X,25HAVAILABLE CORE STORAGE = ,I5)
       STOP
  131  IF(IFLAG.LE.0) GO TO 130
       WRITE(6,200) IFLAG
  200  FORMAT(//10X,31HNUMBER OF INPUT ERRORS FOUND = ,I3,
      *       /10X,"********** PROGRAM STOPS *********"///)
       STOP
  130  WRITE(6,501) NB,N2,MAXKOR,NKORE
       RETURN
       END
```

```
CCORIND
      SUBROUTINE CORIN(COORD,NF,NDOF)
      DIMENSION COORD(NDOF,1),NF(NDOF,1),N(3),X(3)
      DIMENSION XX(3),NN(3),DL(3)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOFN
      NODE=0
      NER=0
      NRR=0
      NID=0
      NNL=0
10    READ(5,100) ND,(NN(J),J=1,3),(XX(J),J=1,3)
100   FORMAT(4I5,3E10.0)
      IF(ND)20,20,30
30    NODE=NODE+1
      IF(ND-NODE) 9998,800,801
 801  ZL=FLOAT(ND-NNL)
      DO 11 J=1,NDOF
 11   DL(J)=(XX(J)-X(J))/ZL
      GO TO 32
 31   NODE=NODE+1
 32   IF(ND-NODE) 9998,800,802
 802  DO 12 J=1,NDOF
 12   X(J)=X(J)+DL(J)
      GO TO 803
 800  DO 13 J=1,NDOF
      X(J)=XX(J)
 13   N(J)=NN(J)
 803  DO 40 I=1,NDOF
      IF(N(I))41,42,43
41    NID=NID+1
      NF(I,NODE)=-1
      GO TO 40
42    NER=NER+1
      NF(I,NODE)=NER
      GO TO 40
43    NRR=NRR+1
      NF(I,NODE)=0
40    COORD(I,NODE)=X(I)
      NNL=NODE
      IF(ND-NNL) 9998,10,31
 20   NIR=-NER
      DO 50 I=1,NDOF
      DO 50 J=1,NDOF
      IF(NF(I,J))51,50,50
51    NIR=NIR-1
      NF(I,J)=NIR
50    CONTINUE
      NR=NRR+NID
120   IF(NODE-NNODES)130,140,130
130   IFLAG=IFLAG+1
      WRITE(6,500) NODE,NNODES
500   FORMAT(//10X,39HNUMBER OF NODAL RESTRAINTS INPUT     = ,I5,
     *       //10X,39HNUMBER OF NODAL RESTRAINTS SPECIFIED = ,I5)
140   IF(NRR+NID-NRSTR)150,160,150
150   IFLAG=IFLAG+1
      WRITE(6,600) NR,NRSTR
600   FORMAT(//10X,39HNUMBER OF  DISPLACEMENTS   INPUT     = ,I5,
     *       //10X,39HNUMBER OF  DISPLACEMENTS   SPECIFIED = ,I5)
160   RETURN
9998  WRITE(6,601) ND
601   FORMAT(//10X,"****NODE NUMBER ",I5," IS OUT OF SEQUENCE****"//)
      STOP
```

```
      END

CDISPIN
      SUBROUTINE DISPIN(NF,P,NDOF)
      DIMENSION NF(NDOF,1),P(1),PP(3)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOFN
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      N=0
 10   READ(5,200) I,PP(1),PP(2),PP(3)
 200  FORMAT(I5,5X,3E10.0)
      IF(I)50,50,20
 20   DO 30 J=1,NDOF
      IFR=NF(J,I)
      IF(IFR)31,30,30
 31   IFR=-IFR
      N=N+1
      P(IFR)=PP(J)
 30   CONTINUE
      GO TO 10
 50   IF(N-NID)60,70,60
 60   IFLAG=IFLAG+1
      WRITE(6,500)N,NID
500   FORMAT(//10X,39HNUMBER OF  DISPLACEMENTS   INPUT     = ,I3,
     *       /10X,39HNUMBER OF  DISPLACEMENTS   SPECIFIED = ,I3)
 70   IF (INPRT.LE.0) GO TO 80
      LOC=NDOF*NNODES
      WRITE (IOUT1,100)
 100  FORMAT(//10X,"THE P VECTOR AFTER DISPIN FOLLOWS")
      WRITE (IOUT1,*) (P(I),I=1,LOC)
 80   RETURN
      END
```

```
CLOADIN
      SUBROUTINE LOADIN(NF,P,NDOF)
      DIMENSION NF(NDOF,1),P(1),PP(3)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNOS,NRSTR,NER,NRR,NID,
     *       IFLAG,N8,NDOFN
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      NLD=0
   10 READ(5,200) I,PP(1),PP(2),PP(3)
  200 FORMAT(I5,5X,3E10.0)
      IF(I)60,60,20
   20 NLD=NLD+1
      DO 30 J=1,NDOF
      IFR=NF(J,I)
      IF(IFR)30,30,40
   40 P(IFR)=PP(J)
   30 CONTINUE
      GO TO 10
   60 IF(NLD-NLDNDS)70,80,70
   70 IFLAG=IFLAG+1
      WRITE(6,500)NLD,NLDNDS
  500 FORMAT(//10X,39HNUMBER OF     LOADED NODES  INPUT      = ,I3,
     *       /10X,39HNUMBER OF     LOADED NODES  SPECIFIED = ,I3)
   80 IF (INPRT.LE.0) GO TO 90
      LOC=NDOF*NNODES
      WRITE (IOUT1,100)
  100 FORMAT(//10X,"THE P VECTOR AFTER LOADIN FOLLOWS")
      WRITE (IOUT1,*) (P(I),I=1,LOC)
   90 RETURN
      END
```

```
CNODES
      SUBROUTINE NODES (COORD,NF,NDOF)
      DIMENSION COORD(NDOF,1),NF(NDOF,1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNDS,NRSTR,NER,NFR,NID,
     *       IFLAG,NB,NDOFN
      COMMON/BLK2/INPRT,IKPRT,ITPRT,IGPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,ICUT1,IOUT2,IOUT3,IOUT4,IOUT5
      WRITE (6,24) IOUT1
      IF (IFLAG.GT.0) WRITE (6,23) IFLAG
      WRITE (IOUT1,20)
      IF (NDOF.EQ.3) WRITE (IOUT1,21) (I,(NF(J,I),J=1,NDOF),
     *(COORD(J,I),J=1,NDOF),I=1,NNODES)
      IF (NDOF.EQ.2) WRITE (IOUT1,22) (I,(NF(J,I),J=1,NDOF),
     *(COORD(J,I),J=1,NDOF),I=1,NNODES)
20    FORMAT(14X,"MODEL GEOMETRY AND RESTRAINT SPECIFICATIONS"/,
     *10X,"NODE",3X,"RX",3X,"RY",3X,"RZ",5X,"X-COORD",4X,
     *"Y-COORD",4X,"Z-COORD"/)
21    FORMAT(10X,I4,2X,I3,2X,I3,2X,I3,2X,1P3E11.3)
22    FORMAT(10X,I4,2X,I3,2X,I3,7X,1P2E11.3)
23    FORMAT(1X,"IFLAG = ",I3,"  DATA ERRORS IN NODE SPECIFICATIONS"//)
24    FORMAT(//10X,"MODEL GEOMETRY AND RESTRAINT SPECIFICATIONS ",
     *       /20X,"ARE ECHOED ON  FILE ",I3)
      RETURN
      END
```

CLUMPIN

```
      SUBROUTINE LUMPIN(NF,SM,NDOF,NER)
      DIMENSION NF(NDOF,1),SM(NER,1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NOER,NRR,NIO,
     *       IFLAG,N8,NDOFN,ISTATE,NKORE,IDYNAM
      NL=0
  101 READ(5,200) I,XM
  200 FORMAT(I5,E15.8)
      IF(I.LE.0) GO TO 102
      NL=NL+1
      DO 103 J=1,NDOF
      IFR=NF(J,I)
      IF(IFR.GT.0) GO TO 104
      WRITE(6,201)  I,IFR
  201 FORMAT(//10X,"NODE ",I5," FREEDOM NUMBER ",I5,
     *       /10X,"HAD A LUMPED MASS INPUT"/)
      GO TO 103
  104 SM(IFR,1)=SM(IFR,1)+XM
  103 CONTINUE
      GO TO 101
  102 IF(NL-NLUMPS) 105,106,105
  105 WRITE(6,202) NL,NLUMPS
  202 FORMAT(//10X,39HNUMBER OF LUMPED MASSES INPUT          = ,I5,
     *       /10X,39HNUMBER OF LUMPED MASSES SPECIFIED     = ,I5)
      IFLAG=IFLAG+1
      NLUMPS=NL
  106 RETURN
      END
```

```
CTRIND
      SUBROUTINE TRIN(JT,TPRP,NF,NDOF)
      DIMENSION JT(3,1),TPRP(4,1),JTEMP(3),TTEMP(4),NF(NDOF,1)
      DIMENSION JJTEMP(3),TTTEMP(4)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOFN
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      NT=0
 10   READ(5,100) I,(JJTEMP(J),J=1,3),(TTTEMP(J),J=1,4)
      NI=0
100   FORMAT(4I5,4E10.0)
      IF(I)20,20,30
 30   NT=NT+1
      IF(I-NT) 9998,800,801
 800  DO 11 J=1,3
      JTEMP(J)=JJTEMP(J)
      TTEMP(J)=TTTEMP(J)
 11   JT(J,NT)=JTEMP(J)
      TTEMP(4)=TTTEMP(4)
      GO TO 802
 801  NI=NI+1
      DO 12 J=1,3
 12   JT(J,NT)=JTEMP(J)+NI
 802  CALL BAND(JT(1,NT),3,NF,NDOF,NB)
      IF(ITPRT.GE.3) WRITE(IOUT1,101) NT,NB
 101  FORMAT(10X,"TRI NR ",I5,"  NB = ",I5)
      DO 50 K=1,4
 50   TPRP(K,NT)=TTEMP(K)
      IF(I-NT) 9998,10,30
 20   IF(NT-NTRIS)60,70,60
 60   WRITE(6,200) NT,NTRIS
200   FORMAT(//10X,39HNUMBER OF    CST   ELEMENTS INPUT     = ,I5,
     *       /10X,39HNUMBER OF    CST   ELEMENTS SPECIFIED = ,I5)
      IFLAG=IFLAG+1
      NTRIS=NT
 70   IF(INPRT)90,90,80
 80   WRITE(6,301) IOUT1
 301  FORMAT(//10X,"TRIANGULAR ELEMENT PROPERTIES ARE ECHOED ON FILE ",I
     *3)
      WRITE(IOUT1,300)
300   FORMAT(//10X,"TRIANGULAR ELEMENT PROPERTIES",
     */1X,"TRI",1X,17HNODE1 NODE2 NODE3,7X,1HE,10X,2HNU,9X,1HT,
     *9X,3HRHO/)
      DO 110 I=1,NTRIS
      WRITE(IOUT1,400) I,(JT(J,I),J=1,3),(TPRP(K,I),K=1,4)
 110  CONTINUE
400   FORMAT(1X,I3,3(1X,I5),1X,1P4E11.3)
 90   WRITE(6,500) NB
500   FORMAT(//10X,45HBANDWIDTH AFTER TRIANGULAR ELEMENTS INPUT IS ,I3)
      RETURN
9998  WRITE(6,501) I
 501  FORMAT(//10X,"****TRI ELEMENT NUMBER ",I5,"  IS OUT OF ",
     *"SEQUENCE****"//)
      STOP
      END
```

```
CSPRGIN
      SUBROUTINE SPRGIN(JS,SPRP,NF,NDOF)
      DIMENSION JS(4,1),SPRP(3,1),NS(4),SP(3),NF(NDOF,1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOFN,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      NSS=0
   10 READ(5,100)I,(NS(M),M=1,4),SP(1),SP(2),SP(3)
  100 FORMAT(5I5,5X,3E10.0)
      IF(I)30,30,20
   20 NSS=NSS+1
      DO 40 M=1,4
   40 JS(M,I)=NS(M)
      CALL BAND(JS(1,I),2,NF,NDOF,NB)
      DO 41 M=1,3
41    SPRP(M,I)=SP(M)
      GO TO 10
   30 IF(NSS-NSPRNG)50,60,50
   50 IFLAG=IFLAG+1
      WRITE(6,200)NSS,NSPRNG
  200 FORMAT(//10X,39HNUMBER OF  SPRING ELEMENTS INPUT     = ,I3,
     *       /10X,39HNUMBER OF  SPRING ELEMENTS SPECIFIED = ,I3)
      NSPRNG=NSS
   60 IF(INPRT) 90,90,80
80    WRITE(6,301) IOUT1
301   FORMAT(//10X,"SPRING ELEMENT PROPERTIES ARE ECHOED ON FILE ",I3)
      WRITE(IOUT1,300)
  300 FORMAT(//10X,45HSPRING ELEMENT NODE DEFINITION AND PROPERTIES
     *       ,/10X,50HNOTE VECTORS 1-3 AND 1-4 DEFINE PLANE OF K1 AND K2
     *       ,/10X,"ELT",3X,"N1",3X,"N2",3X,"N3",3X,"N4",6X,"K1"
     *       ,  9X,"K2",9X,"K3"/)
      DO 70 M=1,NSPRNG
      WRITE(IOUT1,400) M,(JS(N,M),N=1,4),(SPRP(N,M),N=1,3)
  400 FORMAT(8X,5(1X,I4),3(1X,1PE10.3))
   70 CONTINUE
   90 WRITE(6,500)NB
  500 FORMAT(//10X,45HBANDWIDTH AFTER SPRING ELEMENTS INPUT IS.... ,I3)
      RETURN
      END
```

```
CCRKIN8
      SUBROUTINE CRKIN8(JC,CPRP,NF,NDOF)
      DIMENSION JC(9,1),CPRP(4,1),NC(9),CP(4),NF(NDOF,1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOFN
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DATA ISTRN,ISTRS/6HSTRAIN,6HSTRESS/
      NCC=0
   10 READ(5,100) I,(NC(M),M=1,9)
  100 FORMAT(10I5)
      IF(I) 30,30,20
   20 READ(5,101) I1,(CP(J),J=1,4)
  101 FORMAT(I5,4E10.0)
      IF(I-I1) 51,61
   51 IFLAG=IFLAG+1
      WRITE(6,201) I,I1
  201 FORMAT(///7X,"DATA CARDS FOR 8-NODE ELEMENTS ",I3,
     *" AND ",I3," ARE SCRAMBLED")
      GO TO 30
   61 NCC=NCC+1
      DO 40 M=1,9
   40 JC(M,I)=NC(M)
      CALL BAND(JC(1,I),8,NF,NDOF,NB)
      DO 41 J=1,4
   41 CPRP(J,I)=CP(J)
      GO TO 10
   30 IF(NCC-NCRK8)50,60,50
   50 IFLAG=IFLAG+1
      WRITE(6,200)NCC,NCRK8
  200 FORMAT(//10X,39HNUMBER OF  8-NODE ELEMENTS INPUT     = ,I3,
     *       /10X,39HNUMBER OF  8-NODE ELEMENTS SPECIFIED = ,I3)
      NCRK8=NCC
   60 IF(INPRT) 90,90,80
   80 WRITE (6,301) IOUT1
  301 FORMAT(//10X,"8-NODE ELEMENT PROPERTIES ARE ECHOED ON FILE ",I3)
      WRITE(IOUT1,300)
  300 FORMAT(//14X,"8-NODE ELEMENT NODES AND MATERIAL PROPERTIES",
     */1X,"ELT",4X,"1",4X,"2",4X,"3",4X,"4",4X,"5",4X,"6",4X,"7",4X,
     *"8"," PLANE ",6X,"E",10X,"NU",9X,"T",9X,"RHO")
      DO 70 M=1,NCRK8
      IF (JC(9,M).EQ.0) ISTAT=ISTRN
      IF (JC(9,M).EQ.1) ISTAT=ISTRS
  120 WRITE(IOUT1,400) M,(JC(N,M),N=1,8),ISTAT,(CPRP(N,M),N=1,4)
  400 FORMAT(1X,I3,8I5,1X,A6,1P4E11.3)
   70 CONTINUE
   90 WRITE(6,500) NB
  500 FORMAT(//10X,45HBANDWIDTH AFTER 8-NODE ELEMENTS INPUT IS.... ,I3)
      RETURN
      END
```

```
CCRKN10
      SUBROUTINE CRKN1C(JC,CPRP,NF,NDOF)
      DIMENSION JC(11,1),CPRP(4,1),NC(11),CP(4),NF(NDOF,1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNOS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOFN
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN1C,ICUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DATA ISTRN,ISTRS/6HSTRAIN,6HSTRESS/
      NCC=0
   10 READ(5,100) I,(NC(M),M=1,11)
  100 FORMAT(12I5)
      IF(I) 30,30,20
   20 READ(5,101) I1,(CP(J),J=1,4)
  101 FORMAT(I5,4E10.0)
      IF(I-I1) 51,61
   51 IFLAG=IFLAG+1
      WRITE(6,201) I,I1
  201 FORMAT(//7X,"DATA CARDS FOR 10-NODE ELEMENTS ",I3,
     *" AND ",I3," ARE SCRAMBLED")
      GO TO 30
   61 NCC=NCC+1
      DO 40 M=1,11
   40 JC(M,I)=NC(M)
      CALL BAND(JC(1,I),10,NF,NDOF,NB)
      DO 41 J=1,4
   41 CPRP(J,I)=CP(J)
      GO TO 10
   30 IF(NCC-NCRK10)50,60,50
   50 IFLAG=IFLAG+1
      WRITE(6,200)NCC,NCRK10
  200 FORMAT(//10X,39HNUMBER OF 10-NODE ELEMENTS INPUT      = ,I3,
     *       /10X,39HNUMBER OF 10-NODE ELEMENTS SPECIFIED = ,I3)
      NCRK10=NCC
   60 IF(INPRT) 90,90,80
   80 WRITE(6,301) IOUT1
  301 FORMAT(//10X,"10-NODE ELEMENT PROPERTIES ARE ECHOED ON FILE ",I3)
      WRITE(IOUT1,300)
  300 FORMAT(//14X,"10-NODE ELEMENT NODES AND MATERIAL PROPERTIES",
     */1X,"ELT",4X,"1",4X,"2",4X,"3",4X,"4",4X,"5",4X,"6",4X,"7",4X,
     *"8",4X,"9",3X,"10","  PLANE ",6X,"E",10X,"NU",9X,"T",9X,"RHO"/)
      DO 70 M=1,NCRK10
      IF (JC(11,M).EQ.0) ISTAT=ISTRN
      IF (JC(11,M).EQ.1) ISTAT=ISTRS
  120 WRITE(IOUT1,400) M,(JC(J,M),J=1,10),ISTAT,(CPRP(N,M),N=1,4)
  400 FORMAT(1X,I3,10I5,1X,A6,1P4E11.3)
   70 CONTINUE
   90 WRITE(6,500) NB
  500 FORMAT(//10X,45HBANDWIDTH AFTER 10-NODE ELEMENTS INPUT IS... ,I3)
      RETURN
      END
```

```
CBAND
      SUBROUTINE BAND(ND,NDE,NF,NDOF,NB)
      DIMENSION ND(1),NF(NDOF,1),L(10)
      NS=0
      DO 10 K=1,NDE
   10 L(K)=ND(K)
      DO 20 K=1,NDE
      LK=L(K)
      DO 20 J=K,NDE
      LJ=L(J)
      DO 30 KK=1,NDOF
      IF(NF(KK,LK))30,30,40
   40 DO 50 KJ=1,NDOF
      IF(NF(KJ,LJ))50,50,60
   60 NS=IABS(NF(KK,LK)-NF(KJ,LJ))+1
      IF(NS-NB)50,50,70
   70 NB=NS
   50 CONTINUE
   30 CONTINUE
   20 CONTINUE
      RETURN
      END
```

```
CFORMEX
      SUBROUTINE FORM(A,N)
      DIMENSION A(1),N(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      COMMON/ADR/NNF,NJT,NTP,NJS,NSP,NJ8,N8P,NJ10,N10P,NP1,NST,NSM,
     *           NDY,NP2,NUV,NUN
      IF(NTRIS)25,25,10
10    CALL TRI(A,A(NNF),A(NJT),A(NTP),A(NP1),A(NST),A(NSM),NDOF,NER)
25    IF(NSPRNG)20,20,15
15    CALL SPRNG(A,A(NNF),A(NJS),A(NSP),A(NP1),A(NST),NER)
20    IF(NCRK8)40,40,30
30    N1=NDY+256
      N2=N1+384
      N3=N2+256
      N4=N3+384
      N5=N4+576
      CALL CRAK8(A,A(NNF),A(NJ8),A(N8P),A(NP1),A(NST),A(NSM),NDOF,NER,
     *           A(NDY),A(N1),A(N2),A(N3),A(N4),A(N5))
40    IF(NCRK10)60,60,50
50    N1=NDY+400
      N2=N1+600
      N3=N2+400
      N4=N3+600
      N5=N4+900
      CALL CRAK10(A,A(NNF),A(NJ10),A(N10P),A(NP1),A(NST),A(NSM),NDOF,
     *           NER,A(NDY),A(N1),A(N2),A(N3),A(N4),A(N5))
60    IF(IOPRT.GT.3) CALL PRTRM(NEP,A(NST),NER,NB,IOUT1)
      IF(IOPRT.GT.3) CALL PRTRM(NER,A(NSM),NER,NB,IOUT1)
      RETURN
      END
```

```
CINVERT
      SUBROUTINE INVERT(A,B,N,NA)
C  MATRIX INVERSION
C  GLOBAL VARIABLES%
C       NA  IS THE DIMENSIONED ORDER OF THE MATRICES A AND B
C       N     IS THE NUMBER OF ROWS ACTUALLY USED IN THE A AND B MATRICES
C       AINVERSE IS COMPUTED AND STORED IN B
C       THE MATRIX A IS DESTROYED
      DOUBLE PRECISION D,T
      DIMENSION A(1),B(1)
      DO 3 J=1,N
      DO 3 I=1,N
      L=(J-1)*NA+I
      IF(I-J) 5,4,5
 5    B(L)=0.
      GO TO 3
 4    B(L)=1.
 3    CONTINUE
      DO 7 K=1,N
      J=(K-1)*NA+K
      L=K
      DO 15 I=K,N
      J1=(K-1)*NA+I
      IF(ABS(A(J1))-ABS(A(J))) 15,15,16
 16   J=J1
      L=I
 15   CONTINUE
      D=A(J)
      IF(D) 2,6,2
 6    STOP
 2    DO 23 I=1,N
      J1=(I-1)*NA+K
      J2=(I-1)*NA+L
      IF(I-K) 21,20,20
 20   T=A(J2)/D
      A(J2)=A(J1)
      A(J1)=T
 21   T=B(J2)/D
      B(J2)=B(J1)
 23   B(J1)=T
      DO 7 I=K,N
      IF(I-K) 8,7,8
 8    L=(K-1)*NA+I
      T=A(L)
      IF(T) 9,7,9
 9    DO 10 J=1,N
      J1=(J-1)*NA+I
      J2=(J-1)*NA+K
      IF(J-K) 10,12,12
 12   A(J1)=A(J1)-A(J2)*T
 10   B(J1)=B(J1)-B(J2)*T
 7    CONTINUE
      DO 1 K=2,N
      L=N-K+1
      L1=L+1
      DO 1 I=1,N
      T=0.
```

```
      DO 13 J=L1,N
      J1=(J-1)*NA+L
      J2=(I-1)*NA+J
13    T=T+A(J1)*B(J2)
      J1=(I-1)*NA+L
1     B(J1)=B(J1)-T
      RETURN
      END
```

```
CTRILM
      SUBROUTINE TRI (CO,NF,JT,TPRP,P,ST,SM,NDOFN,NROW)
      DIMENSION CO(NDOFN,1),NF(NDOFN,1),JT(3,1),TPRP(4,1),P(1),
     *          ST(NROW,1),SM(NROW,1),AKT(6,9),AK(9,9),AM(9,9),
     *          T(6,9),LOCATE(9),STV(21),DD(3,3),BB(3,6),
     *           A(3),B(3),C(3),D(3)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
-     COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,N8,NDOF,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      REAL NU
      DATA STV/ 2.,0.,1.,0.,1.,0.,2.,0.,1.,0.,1.,2.,0.,1.,0.,2.,0.,1.,
     *2.,0.,2./
      DO 310 I=1,NTRIS
      IF(TPRP(1,I).LE.0) GO TO 310
      IF(TPRP(3,I).LE.0) GO TO 310
      DO 19 J=1,3
      A(J)=0.0
      B(J)=0.0
      C(J)=0.0
      D(J)=0.0
      DO 19 K=1,3
      DD(J,K)=0.0
   19 CONTINUE
      DO 20 J=1,9
      DO 20 K=1,9
   20 AM(J,K)= 0.0
      DO 21 J=1,6
      DO 21 K=1,9
      T(J,K)=0.0
   21 CONTINUE
      DO 22 J=1,3
      DO 22 K=1,6
      BB(J,K)=0.0
   22 CONTINUE
      J1=JT(1,I)
      J2=JT(2,I)
      J3=JT(3,I)
      A(3)=0.
      B(3)=0.
      DO 11 J=1,NDOF
      A(J)=CO(J,J2)-CO(J,J1)
   11 B(J)=CO(J,J3)-CO(J,J1)
      CALL UNIVEC(A,NDOF,XL12)
      CALL UNIVEC(B,NDOF,XL13)
      CALL CROSS(A,B,C)
      CALL UNIVEC(C,3,XLZ)
      CALL CROSS(C,A,D)
      DO 14 J=1,3
      L1=(2*J)-1
      L2=NDOF*(J-1)
      DO 14 K=1,NDOF
      T(L1,L2+K)=A(K)
      T(L1+1,L2+K)=D(K)
   14 CONTINUE
      K1=3*NDOF
      C(2)=-XL13*DOT(B,A,NDOF)
      C(3)=XL12
      C(1)=-C(2)-C(3)
      IF(ABS(-C(2)-C(3)).LT.(5.E-06*(ABS(C(2))+C(3)))) C(1)=0.0
      A(2)= XL13*DOT(B,D,NDOF)
      A(1)=-A(2)
```

```
      A(3)= 0.
      C2=-C(2)
C
C     COMPUTE  STIFFNESS MATRIX   KG=TT*BT*(D*B*T)
C
      EX=TPRP(1,I)
      NU=TPRP(2,I)
      TSK=TPRP(3,I)
      TOV4A=TSK/(2.*XL12*A(2))
      G=EX/(2.*(1.+NU))
C
C   GENERATE THE CONSTITUTIVE MATRIX  DD
C
      IF (ISTATE.GT.0) GO TO 401
      DD=(TOV4A*2.*G)/(1.-2.*NU)
      DD(1,1)=DD*(1.-NU)
      DD(2,2)=DD(1,1)
      DD(1,2)=DD*NU
      DD(2,1)=DD(1,2)
      GO TO 410
 401  DD=(TOV4A*EX)/(1.-NU*NU)
      DD(1,1)=DD
      DD(2,2)=DD
      DD(1,2)=DD*NU
      DD(2,1)=DD(1,2)
 410  DD(3,3)=G*TOV4A
C
C   FORM THE STRAIN-DISPL MATRIX   BB
C
      DO 15 J=1,3
      JJ=2*J
      JI=JJ-1
      BB(1,JI)=A(J)
      BB(2,JJ)=C(J)
      BB(3,JI)=C(J)
 15   BB(3,JJ)=A(J)
C
C   FORM THE PRODUCT DD*BB*T  FOR USE IN STRESS RECOVERY
C
      DO 16 J=1,3
      DO 16 M=1,K1
      AKT(J,M)=0.
      DO 16 K=1,3
      DO 16 L=1,6
 16   AKT(J,M)=AKT(J,M)+DD(J,K)*BB(K,L)*T(L,M)
      DO 17 J=1,K1
      DO 17 M=1,K1
      AK(J,M)=0.
      DO 17 K=1,6
      DO 17 L=1,3
 17   AK(J,M)=AK(J,M) + T(K,J)*BB(L,K)*AKT(L,M)
C
C   ASSEMBLE THE LOCAL MASS MATRIX
C
      IF(IDYNAM.EQ.0) GO TO 18
      RHO=TPRP(4,I)
      IF(RHO.LE.0) GO TO 18
      IF(ABS(RHO).LE.0.0) GO TO 18
      RAT12=(RHO*TSK*XL12*A(2))/24.
      IF(IDYNAM.GT.2) GO TO 23
      L1=0
      DO 10 J=1,5
      L1=L1+1
      AKT(J,J)=RAT12*STV(L1)
```

```
      L2=J+1
      DO 10 K=L2,6
      L1=L1+1
      AKT(J,K)=RAT12*STV(L1)
      AKT(K,J)=AKT(J,K)
 10   CONTINUE
      L1=L1+1
      AKT(6,6)=RAT12*STV(L1)
C
      CALL ATBA(6,T,6,AKT,9,AM,E,K1)
      GO TO 18
   23 CONTINUE
      DO 24 J=1,9
      AM(J,J)=4.*RAT12
   24 CONTINUE
 18   CONTINUE
      DO 326 J=1,NDOF
      LOCATE(J)=NF(J,J1)
      LOCATE(J+NDOF)=NF(J,J2)
 326  LOCATE(J+2*NDOF)=NF(J,J3)
      DO 327 J=1,K1
      IF(LOCATE(J).LE.0) GO TO 327
      J1=LOCATE(J)
      DO 335 K=1,K1
      IF(LOCATE(K)) 329,335,330
 329  J2=-LOCATE(K)
      P(J1)=P(J1)-AK(J,K)*P(J2)
      GO TO 335
 330  J2=LOCATE(K)
      IF(J2.LT.J1) GO TO 335
      J3=J2-J1+1
      IF(IDYNAM.GT.2) J2=1
      ST(J1,J3)=ST(J1,J3) + AK(J,K)
      SM(J1,J2)=SM(J1,J2) + AM(J,K)
 335  CONTINUE
 327  CONTINUE
 310  CONTINUE
      RETURN
      END
```

```
CSPRNG
      SUBROUTINE SPRNG(CO,NF,JS,SPRP,P,ST,NBW)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLCNDS,NPSTP,NEP,NEP,NTD,
     *       IFLAG,NB,NDOF,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      DIMENSION CO(1),NF(1),JS(1),SPRP(1),P(1),ST(NBW,1),DICOS(6,6),
     1   AK(6,6),AKT(6,6),LOCATE(6)
      REAL K1,K2,K3
      DO 10 KOUNT=1,NSPRNG
      DO 5 I=1,6
      DO 5 J=1,6
      DICOS(I,J)=0.0
      AK(I,J)=0.0
      AKT(I,J)=0.0
    5 CONTINUE
      LOC=4*(KOUNT-1)
C
C FORM DIRECTION COSINES FOR K1-K2 PLANE
C
      LOC1=NDOF*(JS(LOC+1)-1)
      LOC3=NDOF*(JS(LOC+3)-1)
      LOC4=NDOF*(JS(LOC+4)-1)
C
C FORM COMPONENTS OF VECTORS FROM NODES 1 TO 3 AND 1 TO 4
C VECTOR FROM NODES 1 TO 3 DEFINES DIRECTION OF K1
C
      DX=CO(LOC3+1)-CO(LOC1+1)
      DY=CO(LOC3+2)-CO(LOC1+2)
      DZ=0.0
      GO TO (20,20,15),NDOF
   15 DZ=CO(LOC3+3)-CO(LOC1+3)
   20 DL13=SQRT(DX*DX+DY*DY+DZ*DZ)
      A1=DX/DL13
      A2=DY/DL13
      A3=DZ/DL13
      DX=CO(LOC4+1)-CO(LOC1+1)
      DY=CO(LOC4+2)-CO(LOC1+2)
      DZ=0.0
      GO TO (30,30,25),NDOF
   25 DZ=CO(LOC4+3)-CO(LOC1+3)
   30 DL14=SQRT(DX*DX+DY*DY+DZ*DZ)
      B1=DX/DL14
      B2=DY/DL14
      B3=DZ/DL14
      C1=A2*B3-A3*B2
      C2=A3*B1-A1*B3
      C3=A1*B2-A2*B1
      DLC=SQRT(C1*C1+C2*C2+C3*C3)
      C1=C1/DLC
      C2=C2/DLC
      C3=C3/DLC
      B1=C2*A3-C3*A2
      B2=C3*A1-C1*A3
      B3=C1*A2-A1*C2
      DICOS(1,1)=A1
      DICOS(1,2)=A2
      DICOS(1,3)=A3
      DICOS(2,1)=B1
      DICOS(2,2)=B2
      DICOS(2,3)=B3
      GO TO(32,32,31),NDOF
   31 DICOS(3,1)=C1
      DICOS(3,2)=C2
```

```
      DICOS(3,3)=C3
   32 DO 35 I=1,NDOF
      DO 35 J=1,NDOF
   35 DICOS(I+NDOF,J+NDOF)=DICOS(I,J)
C
C FORM ELEMENTS OF LOCAL STIFFNESS MATRIX
C
      K1=SPRP(3*KOUNT-2)
      K2=SPRP(3*KOUNT-1)
      K3=SPRP(3*KOUNT)
C
C FORM LOCAL STIFFNESS MATRIX, AK
C
      M=2*NDOF
      AK(1,1)=K1
      AK(1,NDOF+1)=-K1
      AK(1+NDOF,1+NDOF)=K1
      AK(2,2)=K2
      AK(2,NDOF+2)=-K2
      AK(2+NDOF,2+NDOF)=K2
      GO TO (40,40,36),NDOF
   36 AK(3,3)=K3
      AK(3,6)=-K3
      AK(6,6)=K3
   40 IJ=M-1
      DO 45 I=1,IJ
      JI=I+1
      DO 45 J=JI,M
      AK(J,I)=AK(I,J)
   45 CONTINUE
C
C TRANSFORM AK TO GLOBAL COORDINATES
C
      DO 50 I=1,M
      DO 50 J=1,M
      AKT(I,J)=0.0
      DO 50 K=1,M
      AKT(I,J)=AKT(I,J)+AK(I,K)*DICOS(K,J)
   50 CONTINUE
      DO 55 I=1,M
      DO 55 J=1,M
      AK(I,J)=0.0
      DO 55 K=1,M
      AK(I,J)=AK(I,J)+DICOS(K,I)*AKT(K,J)
   55 CONTINUE
C
C STORE INTO STRUCTURAL STIFFNESS MATRIX, ST
C
      DO 60 I=1,2
      NODE=JS(LOC+I)
      LOCN=NDOF*(NODE-1)
      JI=NDOF*(I-1)
      DO 60 J=1,NDOF
   60 LOCATE(J+JI)=NF(LOCN+J)
      DO 65 I=1,M
      IF(LOCATE(I))65,65,70
   70 LI=LOCATE(I)
      DO 75 J=1,M
      IF(LOCATE(J))80,75,85
   80 LJ=-LOCATE(J)
      P(LI)=P(LI)-AK(I,J)*P(LJ)
      GO TO 75
   85 LJ=LOCATE(J)
      IF(LJ-LI)75,90,90
```

```
90 LIBAND=LJ-LI+1
   ST(LI,LIBAND)=ST(LI,LIBAND)+AK(I,J)
75 CONTINUE
65 CONTINUE
10 CONTINUE
   RETURN
   END
```

```
CCRAK8D
      SUBROUTINE CRAK8 (CO,NF,JC,CPRP,P,ST,SM,NDOF,NROW,S1,S2,S3,T1,T2,
     *                  T3)
      DIMENSION CO(NDOF,1),NF(NDOF,1),JC(9,1),CPRP(4,1),P(1),
     *          ST(NROW,1),SM(NROW,1)
      DIMENSION S1(16,16),S2(16,24),T1(16,24),T2(24,24),
     *          T3(24,24),A(3),B(3),LOCATE(24),S3(16,16)
      DIMENSION DK1(16,16),DK2(16,16),DM11(16,16),DM12(16,16),
     *          DM22(16,16),DA1(16,16),DA2(16,16)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLCNDS,NRSTR,NER,NRR,NIO,
     *       IFLAG,N8,NDOFN,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,ICUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DATA DL/0.005/
      DATA ((DK1(I,J),J=1,16),I= 1, 2)/
     *  .74918292E+05, -.34405470E+05, -.21599952E+06, -.33150230E+05,
     * -.67540618E+05, -.13656206E+05, -.75603812E+05, -.66084340E+05,
     * -.72113795E+05,  .33583761E+05,  .65785202E+05, -.61792838E+05,
     * -.14160356E+06, -.16122800E-03,  .51319907E+02, -.29428780E-01,
     * -.34405470E+05,  .76799831E+05,  .21806068E+06, -.38833618E-02,
     * -.11655353E+06,  .19201443E+05,  .75308031E+05,  .26245117E-02,
     * -.38659665E+06, -.53278712E+06, -.51428760E+06, -.72631836E-02,
     * -.11928898E+06, -.12040138E-04, -.46678156E-14, -.23695873E-04/
      DATA ((DK1(I,J),J=1,16),I= 3, 4)/
     * -.21599952E+06,  .21806068E+06,  .12245796E+07,  .26004660E+06,
     *  .35000448E+06,  .22992243E+06,  .54040894E+06,  .48843752E+06,
     * -.29276585E+06, -.10909787E+07, -.20193788E+07, -.67446061E+06,
     * -.16958450E+07, -.63909041E-02,  .14383888E+00,  .14293194E-01,
     * -.33150230E+05, -.38833618E-02,  .26004660E+06,  .43519982E+06,
     *  .67627168E+06,  .32257080E-01, -.74783684E+06,  .13056326E+06,
     *  .49659966E+06,  .41503906E-02, -.30757978E+07, -.42768359E+07,
     * -.40322366E+07, -.20887852E-02, -.47391746E-04,  .73464159E-02/
      DATA ((DK1(I,J),J=1,16),I= 5, 6)/
     * -.67540618E+05, -.11655353E+06,  .35000448E+06,  .67627168E+06,
     *  .19586981E+07,  .74512984E+06, -.20286459E+06,  .28624285E+06,
     *  .18211592E+07,  .31062565E+07, -.11820282E+07, -.67598636E+07,
     * -.10982185E+08, -.14919598E-01,  .11122704E-01, -.25386810E-01,
     * -.13656206E+05,  .19201443E+05,  .22992243E+06,  .32257080E-01,
     *  .74512984E+06,  .20822341E+07,  .26953048E+07,  .83007813E-02,
     * -.30016365E+07,  .15082815E+07,  .35397209E+07, -.15136719E-01,
     * -.16728412E+08, -.16212463E-04,  .14086947E-01, -.60009211E-03/
      DATA ((DK1(I,J),J=1,16),I= 7, 8)/
     * -.75603812E+05,  .75308031E+05,  .54040894E+06, -.74783684E+06,
     * -.20286459E+06,  .26953048E+07,  .72730057E+07,  .36422817E+07,
     * -.11362507E+07,  .97540634E+06,  .93734054E+07,  .18213988E+08,
     * -.11306448E+07, -.71194526E-01, -.30632019E-01,  .29551601E+00,
     * -.66084340E+05,  .26245117E-02,  .48843752E+06,  .13056326E+06,
     *  .28624285E+06,  .83007813E-02,  .36422817E+07,  .97627335E+07,
     *  .11470787E+08,  .47607422E-02, -.10329097E+08,  .11776049E+08,
     *  .22616133E+08, -.35675049E+00, -.80658495E-03,  .70429754E+00/
      DATA ((DK1(I,J),J=1,16),I= 9,10)/
     * -.72113795E+05, -.38659665E+06, -.29276585E+06,  .49659966E+06,
     *  .18211592E+07, -.30016365E+07, -.11362507E+07,  .11470787E+08,
     *  .30187835E+08,  .18290617E+08, -.11373638E+07,  .61486831E+07,
     *  .45521573E+08, -.60854594E+00,  .33526039E+00,  .70967101E+00,
     *  .33583761E+05, -.53278712E+06, -.10909787E+07,  .41503906E-02,
     *  .31062565E+07,  .15082815E+07,  .97540634E+06,  .47607422E-02,
     *  .18290617E+08,  .45704249E+08,  .51106232E+08, -.21484375E-01,
     * -.36214069E+08,  .49018860E-03,  .88392596E+00, -.43737888E-03/
      DATA ((DK1(I,J),J=1,16),I=11,12)/
     *  .65785202E+05, -.51428760E+06, -.20193788E+07, -.30757978E+07,
     * -.11820282E+07,  .35397209E+07,  .93734054E+07, -.10329097E+08,
     * -.11373638E+07,  .51106232E+08,  .12727866E+09,  .84333936E+08,
```

```
*   .56215047E+07,  .87467202E+00,  .79859351E+00,  .10492592E+01,
* -.61792838E+05, -.72631836E-02, -.67446061E+06, -.42768359E+07,
* -.67598636E+07, -.15136719E-01,  .18213988E+08,  .11776049E+08,
*   .61486831E+07, -.21484375E-01,  .84333936E+08,  .20581100E+09,
*   .22507218E+09, -.10224724E+01, -.12266040E-02,  .76087311E+01/
 DATA ((DK1(I,J),J=1,16),I=13,14)/
* -.14160356E+06, -.11928898E+06, -.16958450E+07, -.40322366E+07,
* -.10982185E+08, -.16728412E+08, -.11306448E+07,  .22616133E+08,
*   .45521573E+08, -.36214069E+08,  .56215047E+07,  .22507218E+09,
*   .53144446E+09, -.32404133E+01,  .11597366E+01,  .10001343E+02,
* -.16122800E-03, -.12040139E-04, -.63909041E-02, -.20887852E-02,
* -.14919598E-01, -.16212463E-04, -.71194526E-01, -.35675049E+00,
* -.60854594E+00,  .49018860E-03,  .87467202E+00, -.10224724E+01,
* -.32404133E+01,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK1(I,J),J=1,16),I=15,16)/
*   .51319907E+02, -.46678156E-14,  .14383888E+00, -.47391746E-04,
*   .11122704E-01,  .14086947E-01, -.30632019E-01, -.80658495E-03,
*   .33526039E+00,  .88392596E+00,  .79859351E+00, -.12266040E-02,
*   .11597366E+01,  .00000000   ,  .00000000   ,  .00000000   ,
* -.29428780E-01, -.23695873E-04,  .14293194E-01,  .73464159E-02,
* -.25386810E-01, -.60009211E-03,  .29551601E+00,  .70429754E+00,
*   .70967101E+00, -.43737888E-03,  .10492592E+01,  .76087311E+01,
*   .10001343E+02,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,16),I= 1, 2)/
* -.12283052E+06,  .80388109E+05,  .43199903E+06,  .60179566E+05,
*   .10249154E+06,  .64832466E+05,  .15120530E+06,  .40606926E+05,
* -.79724767E+05, -.10272339E+06, -.13156257E+06, -.12830288E+06,
* -.42925245E+06,  .00000000   ,  .00000000   ,  .00000000   ,
*   .80388109E+05, -.76799831E+05, -.34089634E+06,  .29945374E-02,
*   .14763560E+05, -.96001635E+05, -.26030594E+06, -.22888184E-02,
*   .27149402E+06,  .11839207E+06, -.11823852E+06, -.10375977E-02,
*   .58186712E+06,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,16),I= 3, 4)/
*   .43199903E+06, -.34089634E+06, -.20911018E+07, -.33634035E+06,
* -.70000333E+06, -.46210810E+06, -.10482137E+07, -.27035940E+06,
*   .58551734E+06,  .74595316E+06,  .99135445E+06,  .10181528E+07,
*   .33911229E+07,  .00000000   ,  .00000000   ,  .00000000   ,
*   .60179566E+05,  .29945374E-02, -.33634035E+06, -.23040097E+06,
* -.56329904E+06, -.13015747E-01,  .22833898E+06, -.16128537E+06,
* -.66416634E+06,  .11596680E-02,  .11548741E+07,  .91964966E+06,
*   .30076162E+06,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,16),I= 5, 6)/
*   .10249154E+06,  .14763560E+05, -.70000333E+06, -.56329904E+06,
* -.17279662E+07, -.27049064E+06,  .40572141E+06, -.23229154E+06,
* -.18345239E+07, -.69081886E+06,  .23637737E+07,  .31608362E+07,
*   .20776460E+07,  .00000000   ,  .00000000   ,  .00000000   ,
*   .64832466E+05, -.96001635E+05, -.46210810E+06, -.13015747E-01,
* -.27049064E+06, -.49248380E+06, -.10059140E+07, -.10864258E-01,
*   .60337268E+06, -.19789696E+06, -.12191725E+07,  .13427734E-01,
*   .24391649E+07,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,16),I= 7, 8)/
*   .15120530E+06, -.26030594E+06, -.10482137E+07,  .22833898E+06,
*   .40572141E+06, -.10059140E+07, -.35280243E+07, -.75704441E+06,
*   .22722668E+07,  .76047193E+06, -.46936707E+07, -.45046931E+07,
*   .22604767E+07,  .00000000   ,  .00000000   ,  .00000000   ,
*   .40606926E+05, -.22888184E-02, -.27035940E+06, -.16128537E+06,
* -.23229154E+06, -.10864258E-01, -.75704441E+06, -.11699978E+07,
* -.16375402E+07, -.13183594E-01, -.26048774E+06, -.23525129E+07,
* -.43076167E+07,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,16),I= 9,10)/
* -.79724767E+05,  .27149402E+06,  .58551734E+06, -.66416634E+06,
* -.18345239E+07,  .60337268E+06,  .22722668E+07, -.16375402E+07,
* -.88921951E+07, -.44785472E+07,  .22739846E+07,  .19007510E+07,
* -.12955745E+08,  .00000000   ,  .00000000   ,  .00000000   ,
```

```
*  -.10272339E+06,  .11039207E+06,  .74595316E+06,  .11596680E-02,
*  -.69081886E+06, -.19789696E+06,  .76647193E+06, -.13183594E-01,
*  -.44785472E+07, -.54888420E+07, -.51756151E+07,  .17578125E-01,
*  -.60206947E+07,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,16),I=11,12)/
*  -.13156257E+06, -.11823852E+06,  .99135445E+06,  .11548741E+07,
*   .23637737E+07, -.12191725E+07, -.46936707E+07, -.26048774E+06,
*   .22739846E+07, -.51756151E+07, -.24411038E+08, -.19043161E+08,
*  -.11239140E+08,  .00000000   ,  .00000000   ,  .00000000   ,
*  -.12830288E+06, -.10375977E-02,  .10181528E+07,  .91964966E+06,
*   .31608362E+07,  .13427734E-01, -.45046931E+07, -.23525129E+07,
*   .19017510E+07,  .17578125E-01, -.19043161E+08, -.26753233E+08,
*  -.26760124E+08,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,16),I=13,14)/
*  -.42925245E+06,  .58186712E+06,  .33911229E+07,  .30076162E+06,
*   .20776460E+07,  .24391649E+07,  .22604767E+07, -.43076167E+07,
*  -.12955745E+08, -.60206947E+07, -.11239140E+08, -.26760124E+08,
*  -.70022721E+08,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,16),I=15,16)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM11(I,J),J=1,16),I= 1, 2)/
*   .10097322E+04, -.68202992E+02,  .11699500E+04,  .44909976E+01,
*  -.61218292E+03, -.63649955E+02,  .74353500E+03,  .56819625E+03,
*   .22129602E+01, -.37512852E+03,  .20411586E+03,  .13837981E+04,
*   .10627419E+04,  .27257699E+02,  .36660563E+02, -.38565776E+02,
*  -.68202992E+02,  .14399555E+03,  .39916299E+03, -.63332653E-15,
*  -.24916942E+03, -.93591111E+02, -.45291356E+02,  .23414430E-14,
*  -.30715993E+03, -.55152892E+03, -.59271159E+03,  .89993371E-14,
*   .34896260E+03,  .51879074E-16,  .10191500E-16, -.25292404E-16/
 DATA ((DM11(I,J),J=1,16),I= 3, 4)/
*   .11699500E+04,  .39916299E+03,  .43020579E+04,  .19831411E+03,
*  -.16064028E+04, -.46676275E+03,  .16765731E+04,  .14104800E+04,
*  -.37010019E+03, -.25014061E+04, -.11177082E+04,  .35702160E+04,
*   .50396234E+04,  .31219080E+02,  .25287230E+02, -.89038743E+02,
*   .44909976E+01, -.63332653E-15,  .19831411E+03,  .15596371E+03,
*   .20155116E+03, -.87843415E-15, -.13589093E+03,  .18221974E+02,
*   .80438956E+02,  .37022252E-14, -.35321889E+03, -.46622771E+03,
*  -.40192043E+03, -.20005555E+01, -.19306422E-16, -.29986111E+01/
 DATA ((DM11(I,J),J=1,16),I= 5, 6)/
*  -.61218292E+03, -.24916942E+03, -.16064028E+04,  .20155116E+03,
*   .20787677E+04,  .68949543E+03, -.19783537E+03, -.35687358E+03,
*   .17672949E+04,  .30145601E+04,  .24911917E+04, -.15581543E+04,
*  -.26065318E+04, -.87173121E+01, -.36474195E+02,  .12652744E+02,
*  -.63649955E+02, -.93591111E+02, -.46676275E+03, -.87843415E-15,
*   .68949543E+03,  .59883833E+03,  .57966149E+03, -.63976602E-14,
*   .34000156E+03,  .13991566E+04,  .17938732E+04, -.81532003E-14,
*  -.21157888E+04,  .43855978E-16, -.59991666E+01,  .22537853E-16/
 DATA ((DM11(I,J),J=1,16),I= 7, 8)/
*   .74353500E+03, -.45291356E+02,  .16765731E+04, -.13589093E+03,
*  -.19783537E+03,  .57966149E+03,  .32887169E+04,  .22713371E+04,
*   .20907238E+04,  .99151479E+03,  .36273033E+04,  .62378002E+04,
*   .54129358E+04,  .30752067E+02, -.16416833E+01, -.67009740E+02,
*   .56819625E+03,  .23414430E-14,  .14104800E+04,  .18221974E+02,
```

```
*  -.35687358E+03, -.63976602E-14,  .22713371E+04,  .29183181E+04,
*   .30508636E+04, -.67806871E-13,  .17527806E+03,  .54102119E+04,
*   .78757377E+04,  .33298611E+02,  .13099873E-15, -.47843194E+02/
 DATA ((DM11(I,J),J=1,16),I= 9,10)/
*   .22129602E+01, -.30715993E+03, -.37010019E+03,  .80438956E+02,
*   .17672949E+04,  .34000156E+03,  .20907238E+04,  .30508636E+04,
*   .73587435E+04,  .51224205E+04,  .50056132E+04,  .52719451E+04,
*   .91702361E+04,  .31787652E+02, -.42748841E+02, -.45195283E+02,
*  -.37512852E+03, -.55152892E+03, -.25014061E+04,  .37022252E-14,
*   .30145601E+04,  .13991566E+04,  .99151479E+03, -.67806871E-13,
*   .51224205E+04,  .97256226E+04,  .11093757E+05, -.24436009E-12,
*  -.56303478E+04, -.57554872E-15, -.36995139E+02,  .84155773E-15/
 DATA ((DM11(I,J),J=1,16),I=11,12)/
*   .20411586E+03, -.59271159E+03, -.11177082E+04, -.35321889E+03,
*   .24911917E+04,  .17938732E+04,  .36273033E+04,  .17527806E+03,
*   .50056132E+04,  .11093757E+05,  .19283040E+05,  .81614225E+04,
*  -.47629891E+03,  .60760561E+01, -.38365731E+02, -.58503434E+02,
*   .13837981E+04,  .89893371E-14,  .35702160E+04, -.46622771E+03,
*  -.15581543E+04, -.81532003E-14,  .62378002E+04,  .54102119E+04,
*   .52719451E+04, -.24436009E-12,  .81614225E+04,  .22980518E+05,
*   .27942478E+05,  .55663110E+02,  .83808828E-15, -.12521087E+03/
 DATA ((DM11(I,J),J=1,16),I=13,14)/
*   .10627419E+04,  .34896260E+03,  .50396234E+04, -.40192043E+03,
*  -.26065318E+04, -.21157888E+04,  .54129358E+04,  .78757377E+04,
*   .91702361E+04, -.56303478E+04, -.47629891E+03,  .27942478E+05,
*   .54884463E+05,  .75682183E+02, -.36012386E+02, -.15407803E+03,
*   .27257699E+02,  .51879074E-16,  .31219080E+02, -.20005555E+01,
*  -.87173121E+01,  .43855978E-16,  .30752067E+02,  .33298611E+02,
*   .31787652E+02, -.57554872E-15,  .60760561E+01,  .55663110E+02,
*   .75682183E+02,  .30000000E+01,  .00000000    , -.15000000E+01/
 DATA ((DM11(I,J),J=1,16),I=15,16)/
*   .36660563E+02,  .10191500E-16,  .25287230E+02, -.19306422E-16,
*  -.36474195E+02, -.59991666E+01, -.16416833E+01,  .13099873E-15,
*  -.42748841E+02, -.36995139E+02, -.38365731E+02,  .83808828E-15,
*  -.36012386E+02,  .00000000    ,  .30000000E+01, -.69617638E-17,
*  -.38565776E+02, -.25292404E-16, -.89038743E+02, -.29986111E+01,
*   .12652744E+02,  .22537853E-16, -.67009740E+02, -.47843194E+02,
*  -.45195283E+02,  .84155773E-15, -.58503434E+02, -.12521087E+03,
*  -.15407803E+03, -.15000000E+01, -.69617638E-17,  .32498611E+01/
 DATA ((DM12(I,J),J=1,16),I= 1, 2)/
*  -.25092885E+04,  .39260900E+03, -.15599333E+04, -.68004177E+02,
*   .11646474E+04,  .36861665E+03, -.99138000E+03, -.11562927E+04,
*  -.58827299E+03,  .37466822E+03, -.27215448E+03, -.22278646E+04,
*  -.24821790E+04, -.42612174E+02, -.42612174E+02,  .44414135E+02,
*   .39260900E+03, -.28799111E+03, -.92702811E+03,  .17178912E-14,
*   .18812667E+03,  .48055552E+01, -.20870166E+03, -.16219665E-14,
*   .39370354E+03,  .51000000E+03,  .37687876E+03, -.72272917E-14,
*  -.17056402E+03, -.55253653E-16,  .12000000E+02,  .00000000    /
 DATA ((DM12(I,J),J=1,16),I= 3, 4)/
*  -.15599333E+04, -.92702811E+03, -.95207454E+04, -.85594745E+03,
*   .21418704E+04,  .80788005E+03, -.30015897E+04, -.23607410E+04,
*   .49346692E+03,  .40546035E+04,  .21763304E+04, -.42547831E+04,
*  -.67194979E+04, -.39545661E+01,  .39545661E+01,  .86318804E+02,
*  -.68004177E+02,  .17178912E-14, -.85594745E+03, -.25435333E+03,
*  -.38006875E+03,  .17582507E-14, -.14623571E+03, -.16360846E+03,
*  -.39264930E+03, -.12542051E-14, -.48243266E+02, -.30395577E+03,
*  -.63232297E+03,  .99999999E+01, -.53356299E-16,  .14998889E+02/
 DATA ((DM12(I,J),J=1,16),I= 5, 6)/
*   .11646474E+04,  .18812667E+03,  .21418704E+04, -.38006875E+03,
*  -.36290713E+04, -.12128232E+04,  .26378050E+03,  .73440917E+03,
*  -.25783118E+04, -.37678335E+04, -.33215889E+04,  .17889939E+04,
*   .24528193E+04,  .30127671E+02,  .30127671E+02, -.15671050E+01,
*   .36861665E+03,  .48055552E+01,  .80788005E+03,  .17582507E-14,
*  -.12128232E+04, -.61297535E+03, -.54064857E+03,  .86874952E-14,
```

120

```
     * -.12299278E+04, -.18217146E+04, -.23438226E+04,  .31308289E-13,
     *  .33494710E+03, -.97856022E-16,  .21000000E+02, -.16593714E-15/
      DATA ((DM12(I,J),J=1,16),I= 7, 8)/
     * -.99138000E+03, -.20870166E+03, -.30015897E+04, -.14623571E+03,
     *  .26378050E+03, -.54064857E+03, -.43199739E+04, -.30076649E+04,
     * -.27876317E+04, -.67651295E+03, -.40290682E+04, -.61330846E+04,
     * -.72172477E+04, -.21595833E+02,  .21595833E+02,  .44456133E+02,
     * -.11562927E+04, -.16219665E-14, -.23607410E+04, -.16360846E+03,
     *  .73440917E+03,  .86874952E-14, -.30076649E+04, -.27328509E+04,
     * -.26534150E+04,  .64212524E-13, -.21853203E+04, -.60728802E+04,
     * -.85369041E+04, -.68999999E+01, -.21990331E-15,  .41446111E+02/
      DATA ((DM12(I,J),J=1,16),I= 9,10)/
     * -.58827299E+03,  .39370354E+03,  .49346692E+03, -.39264930E+03,
     * -.25783118E+04, -.12299278E+04, -.27876317E+04, -.26534150E+04,
     * -.71602367E+04, -.54834114E+04, -.66741509E+04, -.52607939E+04,
     * -.76055061E+04,  .73074595E+01,  .73074595E+01,  .39145987E+02,
     *  .37466822E+03,  .51000000E+03,  .40546035E+04, -.12542051E-14,
     * -.37678335E+04, -.18217146E+04, -.67651295E+03,  .64212524E-13,
     * -.54834114E+04, -.81126053E+04, -.92006689E+04,  .20106139E-12,
     *  .23431129E+04,  .70852612E-16,  .19750000E+02, -.58991440E-15/
      DATA ((DM12(I,J),J=1,16),I=11,12)/
     * -.27215448E+03,  .37687876E+03,  .21763304E+04, -.48243266E+02,
     * -.33215889E+04, -.23438226E+04, -.40290682E+04, -.21853203E+04,
     * -.66741509E+04, -.92006689E+04, -.13938659E+05, -.42771067E+04,
     *  .63506522E+03, -.29627858E+02,  .29627858E+02,  .12700228E+02,
     * -.22278646E+04, -.72272917E-14, -.42547831E+04, -.30395577E+03,
     *  .17889939E+04,  .31308289E-13, -.61330846E+04, -.60728802E+04,
     * -.52607939E+04,  .20106139E-12, -.42771067E+04, -.12492875E+05,
     * -.17314736E+05, -.31875000E+02, -.22196329E-15,  .43253442E+02/
      DATA ((DM12(I,J),J=1,16),I=13,14)/
     * -.24821790E+04, -.17056402E+03, -.67194979E+04, -.63232297E+03,
     *  .24528193E+04,  .33494710E+03, -.72172477E+04, -.85369041E+04,
     * -.76055061E+04,  .23431129E+04,  .63506522E+03, -.17314736E+05,
     * -.29877070E+05, -.26446532E+02, -.26446532E+02,  .65925238E+02,
     * -.42612174E+02, -.55253653E-16, -.39545661E+01,  .99999999E+01,
     *  .30127671E+02, -.97856022E-16, -.21595833E+02, -.68999999E+01,
     *  .73074595E+01,  .70852612E-16, -.29627858E+02, -.31875000E+02,
     * -.26446532E+02,  .00000000   ,  .00000000   ,  .00000000   /
      DATA ((DM12(I,J),J=1,16),I=15,16)/
     * -.42612174E+02,  .12000000E+02,  .39545661E+01, -.53356299E-16,
     *  .30127671E+02,  .21000000E+02,  .21595833E+02, -.21990331E-15,
     *  .73074595E+01,  .19750000E+02,  .29627858E+02, -.22196329E-15,
     * -.26446532E+02,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .44414135E+02,  .00000000   ,  .86318804E+02,  .14998889E+02,
     * -.15671050E+01, -.16593714E-15,  .44456133E+02,  .41446111E+02,
     *  .39145987E+02, -.58991440E-15,  .12700228E+02,  .43253442E+02,
     *  .65925238E+02,  .00000000   ,  .00000000   ,  .00000000   /
      DATA ((DM22(I,J),J=1,16),I= 1, 2)/
     *  .16728590E+04, -.35531308E+03,  .11812166E-13,  .18170999E+01,
     * -.77643159E+03, -.42311465E+03,  .39694810E-14,  .41000409E+03,
     *  .39218200E+03, -.15299007E+03, -.89398974E-14,  .75239178E+03,
     *  .16547860E+04,  .00000000   ,  .00000000   ,  .00000000   ,
     * -.35531308E+03,  .20799111E+03,  .69055043E+03, -.12519282E-14,
     *  .12536840E+02,  .15598889E+03,  .35564907E+03, -.10407257E-14,
     * -.31316789E+03, -.12813008E+03,  .10160182E+03,  .33879149E-14,
     * -.52740190E+03,  .00000000   ,  .00000000   ,  .00000000   /
      DATA ((DM22(I,J),J=1,16),I= 3, 4)/
     *  .11812166E-13,  .69055043E+03,  .63471636E+04,  .86624867E+03,
     * -.22806843E-13,  .62842814E+02,  .20010598E+04,  .14125128E+04,
     * -.60576544E-14, -.14816501E+04, -.14508869E+04,  .48578081E+03,
     *  .47917399E-13,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .18170999E+01, -.12519282E-14,  .86624867E+03,  .32876148E+03,
     *  .53404627E+03, -.35227897E-14,  .39417225E+02,  .32094222E+03,
     *  .64564069E+03, -.37222829E-14, -.64694084E+03, -.31950262E+03,
```

```
*  .18258679E+03,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM22(I,J),J=1,16),I= 5, 6)/
* -.77643159E+03,  .12536840E+02, -.22806843E-13,  .53404627E+03,
*  .24193809E+04,  .73326864E+03, -.22320687E-13,  .80389340E+02,
*  .17188745E+04,  .14148764E+04, -.10411810E-13, -.16015529E+04,
* -.16352129E+04,  .00000000   ,  .00000000   ,  .00000000   ,
* -.42311465E+03,  .15598889E+03,  .62842814E+02, -.35227897E-14,
*  .73326864E+03,  .62052015E+03,  .80014317E+03, -.81670781E-14,
*  .81147462E+02,  .69233663E+03,  .13094259E+04, -.40939474E-14,
* -.14514624E+04,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM22(I,J),J=1,16),I= 7, 8)/
*  .39694810E-14,  .35564907E+03,  .20010598E+04,  .39417225E+02,
* -.22320687E-13,  .80014317E+03,  .28799826E+04,  .11588431E+04,
* -.39857007E-13,  .13557898E+03,  .26860455E+04,  .24675462E+04,
* -.20814947E-13,  .00000000   ,  .00000000   ,  .00000000   ,
*  .41000409E+03, -.10407257E-14,  .14125128E+04,  .32094222E+03,
*  .80389340E+02, -.81670781E-14,  .11588431E+04,  .12997608E+04,
*  .14935683E+04, -.26603719E-13,  .15593578E+03,  .15564717E+04,
*  .28560818E+04,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM22(I,J),J=1,16),I= 9,10)/
*  .39218200E+03, -.31316789E+03, -.60576544E-14,  .64564069E+03,
*  .17188745E+04,  .81147462E+02, -.39857007E-13,  .14935683E+04,
*  .47734911E+04,  .22402944E+04, -.92516272E-13,  .23997977E+03,
*  .50703374E+04,  .00000000   ,  .00000000   ,  .00000000   ,
* -.15299007E+03, -.12813008E+03, -.14816501E+04, -.37222829E-14,
*  .14148764E+04,  .69233663E+03,  .13557898E+03, -.26603719E-13,
*  .22402944E+04,  .29162885E+04,  .31305806E+04, -.62602701E-13,
*  .29120448E+03,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM22(I,J),J=1,16),I=11,12)/
* -.89398974E-14,  .10160182E+03, -.14508869E+04, -.64694084E+03,
* -.10411810E-13,  .13094259E+04,  .26860455E+04,  .15593578E+03,
* -.92516272E-13,  .31305806E+04,  .92924391E+04,  .48145082E+04,
* -.22207236E-12,  .00000000   ,  .00000000   ,  .00000000   ,
*  .75239178E+03,  .33879149E-14,  .48578081E+03, -.31950262E+03,
* -.16015529E+04, -.40939474E-14,  .24675462E+04,  .15564717E+04,
*  .23997977E+03, -.52602701E-13,  .48145082E+04,  .68746440E+04,
*  .70607005E+04,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM22(I,J),J=1,16),I=13,14)/
*  .16547860E+04, -.52740190E+03,  .47917399E-13,  .18258679E+03,
* -.16352129E+04, -.14514624E+04, -.20814947E-13,  .28560818E+04,
*  .50703374E+04,  .29120448E+03, -.22207236E-12,  .70607005E+04,
*  .19918047E+05,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM22(I,J),J=1,16),I=15,16)/
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DA1(I,J),J=1,16),I= 1, 2)/
*  .14696938E+02, -.12000000E+02, -.36742346E+02, -.18000000E+02,
* -.25719642E+02, -.27000000E+02, -.29761300E+02, -.40500000E+02,
* -.38973131E+02, -.60750000E+02, -.53735681E+02, -.91125000E+02,
* -.76094234E+02,  .10000000E+01,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .10000000E+01,  .15000000E+01/
```

```
      DATA ((DA1(I,J),J=1,16),I= 3, 4)/
     *  .16715716E+02, -.12000000E+02, -.19097607E+02, -.20000000E+01,
     *  .39504816E+02,  .45000000E+02,  .13592416E+03,  .15150000E+03,
     *  .27504444E+03,  .29925000E+03,  .36140388E+03,  .34587500E+03,
     *  .11176328E+03,  .10000000E+01,  .00000000   , -.99999999E+00,
     *  .50611116E+01,  .20816682E-16, -.36195786E+02,  .48572257E-16,
     * -.45143671E+02, -.80000000E+01, -.89035451E+02, -.48000000E+02,
     * -.18895818E+03, -.16400000E+03, -.36918703E+03, -.39600000E+03,
     * -.58626328E+03,  .00000000   ,  .10000000E+01,  .15000000E+01/
      DATA ((DA1(I,J),J=1,16),I= 5, 6)/
     *  .13776677E+02, -.40000000E+01,  .18800111E+02,  .14000000E+02,
     *  .38957191E+02,  .23000000E+02,  .20063841E+02, -.50000000E+00,
     * -.30748407E+02, -.46250000E+02, -.61074916E+02, -.43125000E+02,
     * -.19501362E+02,  .10000000E+01,  .00000000   , -.10000000E+01,
     *  .85144969E+01, -.17347235E-17, -.20112396E+02,  .20816682E-16,
     * -.11484510E+02, -.80000000E+01, -.16763464E+02, -.16000000E+02,
     * -.11032829E+02, -.40000000E+01,  .18104859E+02,  .28000000E+02,
     *  .46430887E+02,  .00000000   ,  .10000000E+01,  .50000000E+00/
      DATA ((DA1(I,J),J=1,16),I= 7, 8)/
     *  .11497691E+02,  .40000000E+01,  .50531615E+02,  .14000000E+02,
     * -.20264413E+01, -.23000000E+02, -.38136535E+02, -.50000000E+00,
     *  .41060637E+02,  .40250000E+02,  .11936560E+02, -.43125000E+02,
     * -.73179153E+02,  .10000000E+01,  .00000000   , -.10000000E+01,
     *  .18603654E+02, -.52041704E-17,  .12430145E+02, -.13877788E-16,
     * -.45036505E+01, -.80000000E+01, -.36481124E+01,  .16000000E+02,
     *  .22445131E+02, -.40000000E+01, -.29657418E+02, -.28000000E+02,
     * -.11517958E+01,  .00000000   ,  .10000000E+01, -.50000000E+00/
      DATA ((DA1(I,J),J=1,16),I= 9,10)/
     *  .89459798E+01,  .12000000E+02,  .75925743E+02, -.20000000E+01,
     * -.76419656E+02, -.45000000E+02,  .61114630E+02,  .15150000E+03,
     *  .61782797E+02, -.29925000E+03, -.41008641E+03,  .34587500E+03,
     *  .10607633E+04,  .10000000E+01,  .00000000   , -.99999999E+00,
     *  .29546564E+02,  .13877788E-16,  .73212942E+02, -.55511151E-16,
     * -.51798820E+02, -.79999999E+01,  .56372657E+02,  .48000000E+02,
     * -.39786443E+02, -.16400000E+03, -.79865569E+02,  .39600000E+03,
     *  .47450000E+03,  .00000000   ,  .10000000E+01, -.15000000E+01/
      DATA ((DA1(I,J),J=1,16),I=11,12)/
     *  .88233911E-17,  .12000000E+02,  .16178215E-15, -.18000000E+02,
     * -.31144112E-15,  .27000000E+02,  .37030279E-15, -.40500000E+02,
     * -.22748158E-15,  .60750000E+02,  .17298098E-14, -.91125000E+02,
     * -.55109783E-14,  .10000000E+01,  .00000000   , -.15510549E-17,
     *  .29393877E+02,  .00000000   ,  .73484692E+02, -.46837534E-16,
     * -.51439284E+02,  .00000000   ,  .59522601E+02, -.28102520E-15,
     * -.77946262E+02,  .63230671E-15,  .10747136E+03, -.63230671E-15,
     * -.15218847E+03,  .00000000   ,  .10000000E+01, -.15000000E+01/
      DATA ((DA1(I,J),J=1,16),I=13,14)/
     *  .50941873E-17,  .40000000E+01,  .31134989E-16, -.20000000E+01,
     * -.19978957E-16,  .10000000E+01,  .79183117E-17, -.50000000E+00,
     * -.16214389E-17,  .25000000E+00,  .41099019E-17, -.12500000E+00,
     * -.43645608E-17,  .10000000E+01,  .00000000   , -.51701829E-18,
     *  .16970563E+02,  .00000000   ,  .14142136E+02, -.52041704E-17,
     * -.32998316E+01,  .00000000   ,  .12727922E+01, -.34694470E-17,
     * -.55558389E+00,  .26020852E-17,  .25534412E+00, -.86736174E-18,
     * -.12052956E+00,  .00000000   ,  .10000000E+01, -.50000000E+00/
      DATA ((DA1(I,J),J=1,16),I=15,16)/
     *  .84852813E+01, -.40000000E+01, -.70710678E+01, -.20000000E+01,
     * -.16499158E+01, -.10000000E+01, -.63639610E+00, -.50000000E+00,
     * -.27779195E+00, -.25000000E+00, -.12767206E+00, -.12500000E+00,
     * -.60264782E-01,  .10000000E+01,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .10000000E+01,  .50000000E+00/
      DATA ((DA2(I,J),J=1,16),I= 1, 2)/
```

```
*  -.29393877E+02,  .12000000E+02,  .73484692E+02,  .18000000E+02,
*    .51439284E+02,  .27000000E+02,  .59522601E+02,  .40500000E+02,
*    .77946262E+02,  .60750000E+02,  .10747136E+03,  .91124999E+02,
*    .15218847E+03,  .00000000    ,  .00000000    ,  .00000000    ,
*    .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*    .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*    .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*    .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DA2(I,J),J=1,16),I= 3, 4)/
*  -.30841520E+02,  .12000000E+02,  .61540366E+02,  .10000000E+02,
*    .81960025E+01, -.89999999E+01, -.53034337E+02, -.59500000E+02,
*   -.15683866E+03, -.14925000E+03, -.29417963E+03, -.25437500E+03,
*   -.39084218E+03,  .00000000    ,  .00000000    ,  .00000000    ,
*   -.93380610E+01,  .79999999E+01,  .74747686E+02,  .24000000E+02,
*    .81042217E+02,  .46000000E+02,  .10010005E+03,  .60000000E+02,
*    .84783587E+02,  .30500000E+02, -.27266250E+02, -.10350000E+03,
*   -.31633333E+03,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DA2(I,J),J=1,16),I= 5, 6)/
*  -.21586889E+02,  .40000000E+01, -.42466444E+01, -.60000000E+01,
*   -.22969021E+02, -.11000000E+02, -.15807969E+02, -.35000000E+01,
*    .55355577E+01,  .10250000E+02,  .20944999E+02,  .14625000E+02,
*    .13768772E+02,  .00000000    ,  .00000000    ,  .00000000    ,
*   -.13341431E+02,  .80000000E+01,  .47096008E+02,  .80500000E+01,
*    .90073011E+01, -.20000000E+01, -.14244714E+02, -.12000000E+02,
*   -.20018538E+02, -.95000000E+01, -.41121995E+01,  .54999999E+01,
*    .17832177E+02,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DA2(I,J),J=1,16),I= 7, 8)/
*  -.13341431E+02, -.40000000E+01, -.47096008E+02, -.60000000E+01,
*    .90073011E+01,  .11000000E+02,  .14244714E+02, -.35000000E+01,
*   -.20018538E+02, -.10250000E+02,  .41121995E+01,  .14625000E+02,
*    .17832177E+02,  .00000000    ,  .00000000    ,  .00000000    ,
*   -.21586889E+02,  .80000000E+01,  .42466444E+01, -.79999999E+01,
*   -.22969021E+02, -.20000000E+01,  .15807969E+02,  .12000000E+02,
*    .55355577E+01, -.94999999E+01, -.20944999E+02, -.55000000E+01,
*    .13768772E+02,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DA2(I,J),J=1,16),I= 9,10)/
*  -.93380610E+01, -.12000000E+02, -.74747686E+02,  .10000000E+02,
*    .81042217E+02,  .89999999E+01, -.10010005E+03, -.59500000E+02,
*    .84783587E+02,  .14925000E+03,  .27266250E+02, -.25437500E+03,
*   -.31633333E+03,  .00000000    ,  .00000000    ,  .00000000    ,
*   -.30841520E+02,  .79999999E+01, -.61540366E+02, -.24000000E+02,
*    .81960025E+01,  .46000000E+02,  .53034337E+02, -.60000000E+02,
*   -.15683866E+03,  .30500000E+02,  .29417963E+03,  .10350000E+03,
*   -.39084218E+03,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DA2(I,J),J=1,16),I=11,12)/
*  -.15197172E-16, -.12000000E+02, -.11397879E-15,  .18000000E+02,
*    .22220819E-15, -.27000000E+02, -.52518567E-15,  .40500000E+02,
*    .10387729E-14, -.60750000E+02, -.11705106E-14,  .91124999E+02,
*    .12868998E-14,  .00000000    ,  .00000000    ,  .00000000    ,
*   -.29393877E+02,  .12408439E-16, -.73484692E+02, -.37225317E-16,
*    .51439284E+02,  .83756964E-16, -.59522601E+02, -.30802653E-15,
*    .77946262E+02,  .31408861E-15, -.10747136E+03, -.15138196E-14,
*    .15218847E+03,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DA2(I,J),J=1,16),I=13,14)/
*  -.87740914E-17, -.40000000E+01, -.21935229E-16,  .20000000E+01,
*    .14254662E-16, -.10000000E+01, -.11230226E-16,  .50000000E+00,
*    .74041455E-17, -.25000000E+00, -.27810477E-17,  .12500000E+00,
*    .10191933E-17,  .00000000    ,  .00000000    ,  .00000000    ,
*   -.16970563E+02,  .41361464E-17, -.14142136E+02, -.41361464E-17,
*    .32998316E+01,  .31021098E-17, -.12727922E+01, -.38027967E-17,
*    .55558389E+00,  .12925457E-17, -.25534412E+00, -.20765700E-17,
*    .12052956E+00,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DA2(I,J),J=1,16),I=15,16)/
*  -.16970563E+02,  .40000000E+01,  .14142136E+02,  .20000000E+01,
```

```
     *   .32999316E+01,  .10000000E+01,  .12727922E+01,  .50000000E+00,
     *   .55558389E+00,  .25000000E+00,  .25534412E+00,  .12500000E+00,
     *   .12052956E+00,  .00000000    ,  .00000000    ,  .00000000    ,
     *   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
     *   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
     *   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
     *   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
      NP=8
      IP=2*NP
      IP2=3*NP
      DO 800 KT=1,NCRK8
      NORHO=0
      IF(CPRP(1,KT).LE.0) GO TO 800
      IF(CPRP(3,KT).LE.0) GO TO 800
      IF(CPRP(4,KT).LE.0) NORHO=1
      SIG=CPRP(2,KT)
      G=CPRP(1,KT)/(2.*(1.+SIG))
      IF(JC(9,KT).GT.0) SIG=SIG/(1.+SIG)
      THICK=CPRP(3,KT)
      RHO=CPRP(4,KT)
      WTS=DL*G*THICK
      IF(ICPRT.GT.0) WRITE(IOUT1,2001)
     *CPRP(1,KT),G,CPRP(2,KT),SIG,RHO,THICK
 2001 FORMAT(//10X,"E,G,NU,SIG,RHO,THICK",/10X,1P6E10.3/)
      J1=JC(1,KT)
      J2=JC(2,KT)
      J8=JC(8,KT)
      A(3)=0.
      B(3)=0.
      DO 170 I=1,NDOF
      A(I)=CO(I,J1)-CO(I,J8)
      B(I)=CO(I,J2)-CO(I,J1)
  170 CONTINUE
      CALL UNIVEC (A,NDOF,DEL)
      CALL UNIVEC (B,NDOF,DEL1)
      IF (ABS(DEL1-DEL).GT.(5.E-06*(DEL+DEL1))) WRITE (6,1001) KT
 1001 FORMAT(//10X,"**** WARNING ****  8-NODE ELEMENT NUMBER ",I3,
     *        //27X,"IS NOT RECTANGULAR WITHIN (-06)")
      WTM=THICK*DEL*DEL*RHO
      DO 160 I=1,IP
      DO 160 J=1,IP
      S2(I,J)=DA1(I,J)+SIG*DA2(I,J)
  160 T1(I,J)=S2(I,J)
      IF(ICPRT.LT.3) GO TO 201
      WRITE(IOUT1,2003)
      CALL PRTRM(IP,S2,IP,IP,IOUT1)
 2003 FORMAT(//10X,"A IS"/)
  201 CONTINUE
      CALL INVERT (S2,S1,IP,IP)
      IF(ICPRT.LT.3) GO TO 202
      WRITE(IOUT1,2004)
      CALL PRTRM(IP,S1,IP,IP,IOUT1)
      WRITE(IOUT1,2005)
      CALL MABC(IP,S1,IP,T1,IP2,T3,IP,IP,IP)
      CALL PRTRM(IP2,T3,IP,IP,IOUT1)
 2004 FORMAT(//10X,"AINV IS"/)
 2005 FORMAT(//10X,"A*AINV IS"/)
      DO 159 I=1,IP2
      DO 159 J=1,IP2
  159 T3(I,J)= 0.
  202 CONTINUE
      DO 161 I=1,IP
      DO 161 J=1,IP2
  161 S2(I,J)=0.
```

```
      DO 162 I=1,NP
      LOC=2*I-1
      LOC1=NDOF*(I-1)
      DO 162 J=1,NDOF
      S2(LOC,LOC1+J)=A(J)
      S2(LOC+1,LOC1+J)=B(J)
  162 CONTINUE
      LOC=NDOF*NP
      IF(ICPRT.LT.2) GO TO 203
      WRITE(IOUT1,2020)
      CALL PRTRM(IP,S2,IP,LOC,IOUT1)
 2020 FORMAT(//10X,"THE T MATRIX IS"/)
  203 CONTINUE
      CALL MABC (IP,S1,IP,S2,IP,T1,IP,IP,LOC)
C (AINV*T) IS IN T1
      IF(IDYNAM.GT.0) WRITE (25) DEL,G,THICK,((T1(J,K),J=1,IP),K=1,L
     *OC)
      IF(ICPRT.LT.3)GO TO 303
      WRITE(IOUT1,3030)
 3030 FORMAT(//10X,"THE AINV*T MATRIX IS"/)
      CALL PRTRM(IP,T1,IP,IP,ICUT1)
  303 CONTINUE
      DO 163 I=1,IP
      DO 163 J=1,IP
  163 S3(I,J)=(DK1(I,J)+SIG*DK2(I,J))*WTS
      CALL AT3A(IP,T1,IP,S3,IP2,T2,LOC,IP)
      IF(ICPRT.LT.3) GO TO 204
      WRITE(IOUT1,2021)
      CALL PRTRM(IP2,T2,LOC,LOC,IOUT1)
 2021 FORMAT(//10X,"THE GK MATRIX IS"/)
  204 CONTINUE
      IF(IDYNAM.EQ.0) WRITE(25) (DEL,G,THICK,S1,S2,T2)
      IF(NORHO.EQ.1) GO TO 165
      IF(IDYNAM.LE.0.OR.IDYNAM.GT.2)GO TO 165
      SIGSQ=SIG*SIG
      DO 164 I=1,IP
      DO 164 J=1,IP
  164 S2(I,J)=(DM11(I,J)+SIG*DM12(I,J)+SIGSQ*DM22(I,J))*WTM
C  CM IS IN S2
      CALL AT3A (IP,T1,IP,S2,IP2,T3,LOC,IP)
      IF(ICPRT.LT.3) GO TO 205
      WRITE(IOUT1,2006)
      CALL PRTRM(IP2,T3,LOC,LOC,IOUT1)
 2006 FORMAT(//10X,"THE GM MATRIX IS"/)
  205 CONTINUE
      GO TO 167
  165 DO 166  I=1,IP2
      DO 166  J=1,IP2
  166 T3(I,J)=0.
      IF(IDYNAM.LT.3)GO TO 167
      DO 169 I=1,NP
      DO 168 J=1,NDOF
      K=J+NDOF*(I-1)
      T3(K,K)=WTM/8.0
      IF(I.EQ.3.OR.I.EQ.4.OR.I.EQ.7.OR.I.EQ.8)T3(K,K)=(5.0*WTM)/8.0
  168 CONTINUE
  169 CONTINUE
  167 CONTINUE
      DO 91 I=1,NP
      LOC=JC(I,KT)
      LOC2=NDOF*(I-1)
      DO 91 J=1,NDOF
      LOCATE(J+LOC2)=NF(J,LOC)
   91 CONTINUE
```

```
      LOC=NP*NDOF
      IF(ICPRT.GT.1)  WRITE(IOUT1,2007)KT,(LOCATE(J),J=1,LOC)
 2007 FORMAT(//1X,"  LOCATE(J)  ",/1X,I5,10I5,2(/6X,10I5)/)
      DO 112 I=1,LOC
      IF(LOCATE(I).LE.0) GO TO 112
      IR=LOCATE(I)
      DO 111 J=1,LOC
      IF(LOCATE(J)) 130,111,140
  130 IC= -LOCATE(J)
      P(IR)=P(IR)-T2(I,J)*P(IC)
      GO TO 111
  140 IC=LOCATE(J)
      IF(IC.LT.IR) GO TO 111
      IC=IC-IR+1
      ST(IR,IC)=ST(IR,IC) + T2(I,J)
      IF(IDYNAM.GT.2)IC=1
      IF(IDYNAM.GT.0)SM(IR,IC)=SM(IR,IC) + T3(I,J)
  111 CONTINUE
  112 CONTINUE
  800 CONTINUE
      END FILE 25
      RETURN
      END
```

```
CCRK100
      SUBROUTINE CRAK10 (CO,NF,JC,CPRP,P,ST,SM,NDOF,NROW,S1,S2,S3,T1,T2,
     *                   T3)
      DIMENSION CO(NDOF,1),NF(NDOF,1),JC(11,1),CPRP(4,1),P(1),
     *          ST(NPOW,1),SM(NROW,1)
      DIMENSION S1(20,20),S2(20,30),T1(20,30),T2(30,30),
     *          T3(30,30),A(3),B(3),LOCATE(30),S3(20,20)
      DIMENSION DK1(20,20),DK2(20,20),DM11(20,20),DM12(20,20),
     *          DM22(20,20),DA1(20,20),DA2(20,20)
      DIMENSION SUMODD(10),SUMEVN(10)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOFN,ISTATE,NKORE,IOYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DATA DL/0.0025/
      DATA ((DK1(I,J),J=1,20),I= 1, 2)/
     *  .25383440E+06, -.98855985E+05, -.57599873E+06, -.85996789E+05,
     * -.10096276E+06,  .15459537E+05, -.18716508E+01, -.63836080E+05,
     * -.91795146E+05, -.25800067E+05,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     * -.98855985E+05,  .20479911E+06,  .56252495E+06, -.48942566E-02,
     * -.37124756E+06, -.20479875E+06, -.16776250E+06, -.20751953E-02,
     * -.20712976E+06, -.54613320E+06,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
      DATA ((DK1(I,J),J=1,20),I= 3, 4)/
     * -.57599873E+06,  .56252495E+06,  .27546923E+07,  .53494079E+06,
     *  .11512607E+01, -.33150449E+06, -.60755052E+06,  .17109200E+05,
     * -.72413776E+06, -.16650697E+07,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     * -.85996789E+05, -.48942566E-02,  .53494079E+06,  .81919811E+06,
     *  .12028359E+07,  .41320801E-01, -.15058143E+07, -.98304222E+06,
     * -.73726304E+06,  .42114258E-02,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
      DATA ((DK1(I,J),J=1,20),I= 5, 6)/
     * -.10096276E+06, -.37124756E+06,  .11512607E+01,  .12028359E+07,
     *  .35512267E+07,  .16787275E+07, -.50177133E+06, -.15462737E+07,
     * -.18019692E+07,  .12474602E+07,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .15459537E+05, -.20479875E+06, -.33150449E+06,  .41320801E-01,
     *  .16787275E+07,  .30310258E+07,  .36196245E+07,  .34332275E-02,
     * -.42159821E+07, -.24576278E+07,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
      DATA ((DK1(I,J),J=1,20),I= 7, 8)/
     * -.18716508E+01, -.16776250E+06, -.60755052E+06, -.15058143E+07,
     * -.50177133E+06,  .36196245E+07,  .94687366E+07,  .58304316E+07,
     * -.49429396E+02, -.40367372E+07,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     * -.63836080E+05, -.20751953E-02,  .17109200E+05, -.98304222E+06,
     * -.15462737E+07,  .34332275E-02,  .58304316E+07,  .10111179E+08,
     *  .11038290E+08,  .26794434E-01,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
      DATA ((DK1(I,J),J=1,20),I= 9,10)/
     * -.91795146E+05, -.20712976E+06, -.72413776E+06, -.73726304E+06,
     * -.18019692E+07, -.42159821E+07, -.49429396E+02,  .11038290E+08,
     *  .25498746E+08,  .16782835E+08,  .00000000   ,  .00000000   ,
```

```
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  -.25800087E+05, -.54613320E+06, -.16650697E+07,  .42114258E-02,
*   .12474602E+07, -.24576278E+07, -.40367372E+07,  .26794434E-01,
*   .16782835E+08,  .29595078E+08,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DK1(I,J),J=1,20),I=11,12)/
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .50767049E+05, -.26595127E-01,
*  -.38400039E+05, -.65903989E+05, -.64290989E+05, -.24603010E+05,
*  -.10462570E+01, -.21151595E+05, -.11033629E+06, -.72109782E+05,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.26595127E-01,  .00000000    ,
*  -.59609652E-01, -.16723107E-02, -.16697644E+00,  .16997594E-02,
*  -.13772718E+01, -.14231573E-01,  .35532953E+01,  .48591950E+01/
 DATA ((DK1(I,J),J=1,20),I=13,14)/
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.38400039E+05, -.59609652E-01,
*   .37463849E+06,  .22500996E+06, -.67652892E+00, -.38075740E+06,
*  -.45868195E+06, -.17387021E+06,  .92162676E+05, -.26951269E+06,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.65903989E+05, -.16723107E-02,
*   .22500996E+06,  .27306564E+06,  .31821190E+06, -.36588764E-01,
*  -.18640235E+06, -.10922431E+06,  .26361505E+06, -.17137395E-01/
 DATA ((DK1(I,J),J=1,20),I=15,16)/
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.64290989E+05, -.16697644E+00,
*  -.67652892E+00,  .31821190E+06,  .16557517E+07,  .11283011E+07,
*   .11946951E+06, -.14118704E+07, -.15126550E+07, -.61334600E+06,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.24603010E+05,  .16997594E-02,
*  -.38075740E+06, -.36588764E-01,  .11283011E+07,  .15564757E+07,
*   .17359410E+07,  .21810786E-01, -.12108889E+07, -.93621288E+06/
 DATA ((DK1(I,J),J=1,20),I=17,18)/
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.10462570E+01, -.13772718E+01,
*  -.45868195E+06, -.18640235E+06,  .11946951E+06,  .17359410E+07,
*   .57441307E+07,  .38933584E+07,  .29252335E+02, -.46937592E+07,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.21151595E+05, -.14231573E-01,
*  -.17387021E+06, -.10922431E+06, -.14118704E+07,  .21810786E-01,
*   .38933584E+07,  .56173560E+07,  .59460887E+07,  .53173252E-02/
 DATA ((DK1(I,J),J=1,20),I=19,20)/
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.11033629E+06,  .35532953E+01,
*   .92162676E+05,  .26361505E+06, -.15126550E+07, -.12108889E+07,
*   .29252335E+02,  .59460887E+07,  .17339775E+08,  .12122642E+08,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    , -.72109782E+05,  .48591950E+01,
*  -.26951269E+06, -.17137395E-01, -.61334600E+06, -.93621288E+06,
*  -.46937592E+07,  .53173252E-02,  .12122642E+08,  .17892150E+08/
 DATA ((DK2(I,J),J=1,20),I= 1, 2)/
*  -.40613456E+06,  .24522038E+06,  .11519975E+07,  .10942898E+06,
```

```
*   .30727256E+00, -.20286492E+05,  .26878636E+01, -.39866835E+05,
* -.24478131E+06, -.25002525E+06,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .24522038E+06, -.20479911E+06, -.84122187E+06,  .22888184E-04,
*   .19253631E+06, -.10617332E+01, -.14640797E+06,  .10833740E-02,
*   .28641534E+06,  .27306972E+06,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,20),I= 3, 4)/
*   .11519975E+07, -.84122187E+06, -.44075024E+07, -.44769433E+06,
* -.64425506E+00,  .82768106E+05, -.34918625E+00,  .25004498E+06,
*   .14482529E+07,  .14765234E+07,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .10942898E+06,  .22888184E-04, -.44769433E+06, -.27306684E+06,
* -.62811273E+06, -.14007568E-01,  .44525378E+06,  .21845617E+06,
*   .51624801E+05,  .57220459E-03,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,20),I= 5, 6)/
*   .30727256E+00,  .19253631E+06, -.64425506E+00, -.62811273E+06,
* -.21853709E+07, -.36223273E+06,  .10035298E+07,  .74489177E+06,
*   .93715927E-01, -.60047948E+06,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
* -.20286492E+05, -.10617332E+01,  .82768106E+05, -.14007568E-01,
* -.36223273E+06, -.32767705E+06, -.47646843E+06, -.83923340E-03,
*   .21218618E+06,  .30545914E+02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,20),I= 7, 8)/
*   .26878636E+01, -.14640797E+06, -.34918625E+00,  .44525378E+06,
*   .10035298E+07, -.47646843E+06, -.26120713E+07, -.12689389E+07,
*   .49677034E+02,  .80903916E+06,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
* -.39886835E+05,  .10833740E-02,  .25004498E+06,  .21845617E+06,
*   .74489177E+06, -.83923340E-03, -.12689389E+07, -.11234422E+07,
* -.11301048E+07, -.61035156E-02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,20),I= 9,10)/
* -.24478131E+06,  .28641534E+06,  .14482529E+07,  .51624801E+05,
*   .93715927E-01,  .21218618E+06,  .49677034E+02, -.11301048E+07,
* -.38488772E+07, -.26284711E+07,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
* -.25002525E+06,  .27306972E+06,  .14765234E+07,  .57220459E-03,
* -.60047948E+06,  .30545914E+02,  .80903916E+06, -.61035156E-02,
* -.26284711E+07, -.30688064E+07,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DK2(I,J),J=1,20),I=11,12)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   , -.45126239E+05,  .26595127E-01,
*   .76799973E+05,  .75687028E+05,  .23059845E-01, -.42746934E+05,
*   .10921021E+01,  .59776670E+05,  .51923222E+05, -.17922027E+05,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .26595127E-01,  .00000000   ,
*   .59609652E-01,  .16723107E-02,  .16697644E+00, -.16997594E-02,
*   .13772718E+01,  .14231573E-01, -.35532953E+01, -.48591950E+01/
```

```
      DATA ((DK2(I,J),J=1,20),I=13,14)/
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     * -.17630077E+06, -.18855737E+06,  .76799973E+05,  .59609652E-01,
     * -.52368164E-01, -.19301194E+06, -.18432335E+06,  .49753546E+05,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     * -.18855737E+06, -.27306564E+06,  .75687028E+05,  .16723167E-02,
     * -.36159741E+05, -.21845590E+06, -.15299461E+06,  .35781347E-01,
      DATA ((DK2(I,J),J=1,20),I=15,16)/
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .84013747E+00, -.15299461E+06,  .23059845E-01,  .16697644E+00,
     * -.23893582E+06, -.81607571E+04, -.40139515E+06, -.45901607E+06,
     *  .00000000  ,  .00000000  ,  .80108643E-02, -.92438511E+05,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .12113144E+06,  .35781347E-01, -.42746934E+05, -.16997594E-02,
     * -.60847487E+06, -.14434443E-01, -.45901607E+06, -.81919998E+06,
      DATA ((DK2(I,J),J=1,20),I=17,18)/
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     * -.52368164E-01, -.36159741E+05, -.23893582E+06, -.60847487E+06,
     * -.80619600E+06, -.52026312E+06, -.29633286E+02,  .18805824E+06,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     * -.19301194E+06, -.21845590E+06, -.81607571E+04, -.14434443E-01,
     * -.52026312E+06, -.11234874E+07, -.98803061E+06, -.53869262E-02,
      DATA ((DK2(I,J),J=1,20),I=19,20)/
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     * -.18432335E+06, -.26180845E+06, -.80108643E-02,  .31313270E+06,
     * -.29633286E+02, -.98803061E+06, -.15586368E+07, -.69795602E+06,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .49753546E+05,  .94341020E-02, -.92438511E+05, -.48591950E+01,
     *  .18805824E+06, -.53869262E-02, -.69795602E+06, -.18814744E+02,
      DATA ((DM11(I,J),J=1,20),I= 1, 2)/
     *  .10467904E+04, -.42590781E+02,  .95999999E+03,  .13588686E+03,
     * -.10673532E+03, -.63112885E+02,  .13824000E+03,  .29615729E+03,
     *  .22763046E+03,  .35162547E+02,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     * -.42590781E+02,  .85333332E+02,  .23137123E+03,  .00000000  ,
     * -.16312094E+03, -.11946667E+03, -.11817772E+03,  .00000000  ,
     *  .31588391E+02,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
      DATA ((DM11(I,J),J=1,20),I= 3, 4)/
     *  .95999999E+03,  .23137123E+03,  .28850316E+04,  .35939278E+03,
     * -.29866666E+03, -.43602921E+03,  .95803365E+01,  .66980764E+03,
     *  .86204081E+03,  .49728679E+02,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .00000000  ,  .00000000  ,  .00000000  ,  .00000000  ,
     *  .13588686E+03,  .00000000  ,  .35939278E+03,  .14222222E+03,
     *  .16590692E+03,  .00000000  , -.47433564E+02,  .39009523E+02,
     *  .80424940E+02,  .10666667E+02,  .00000000  ,  .00000000  ,
```

```
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM11(I,J),J=1,20),I= 5, 6)/
* -.10673582E+03, -.16312094E+03, -.29866666E+03,  .16590692E+03,
*   .97975420E+03,  .53491285E+03,  .46080000E+03, -.12339307E+02,
* -.61592458E+02,  .97048905E+01,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
* -.63112885E+02, -.11946667E+03, -.43602921E+03,  .00000000   ,
*   .53491285E+03,  .48274285E+03,  .52842294E+03,  .00000000   ,
* -.26112312E+03,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM11(I,J),J=1,20),I= 7, 8)/
*   .13824000E+03, -.11817772E+03,  .95803365E+01, -.47433564E+02,
*   .46080000E+03,  .52842294E+03,  .10715664E+04,  .56251285E+03,
*   .28964571E+03,  .82408572E+01,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .29615729E+03,  .00000000   ,  .66980764E+03,  .39099523E+02,
* -.12339307E+02,  .00000000   ,  .56251285E+03,  .84520634E+03,
*   .95079768E+03,  .17066666E+02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM11(I,J),J=1,20),I= 9,10)/
*   .22763046E+03,  .31588391E+02,  .86204081E+03,  .80424940E+02,
* -.61592458E+02, -.26112312E+03,  .28964571E+03,  .95079768E+03,
*   .15821252E+04,  .20087732E+02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .35162547E+02,  .00000000   ,  .49728679E+02,  .10666667E+02,
*   .97048905E+01,  .00000000   ,  .82408572E+01,  .17066666E+02,
*   .20087732E+02,  .40000000E+01,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM11(I,J),J=1,20),I=11,12)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .14079637E+03, -.58062991E+01,
*   .64000000E+02, -.10003922E+02, -.31320677E+02,  .68735678E+01,
*   .76799999E+02,  .19051143E+02, -.99111829E+02, -.14608700E+02,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   , -.58062991E+01,  .40000000E+01,
* -.47869254E+01, -.10666667E+02, -.77919343E+01,  .00000000   ,
*   .17518769E+01,  .00000000   , -.76690757E+01,  .00000000   /
 DATA ((DM11(I,J),J=1,20),I=13,14)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .64000000E+02, -.47869254E+01,
*   .17561062E+03,  .11025337E+03,  .76799999E+02, -.21551297E+02,
* -.55669486E+02, -.90517002E+02, -.10971428E+03, -.12572403E+02,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   , -.10003922E+02, -.10666667E+02,
*   .11025337E+03,  .16497778E+03,  .16721456E+03,  .00000000   ,
* -.12528272E+03, -.11702857E+03,  .13998608E+02,  .00000000   /
 DATA ((DM11(I,J),J=1,20),I=15,16)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   , -.31320677E+02, -.77919343E+01,
*   .76799999E+02,  .16721456E+03,  .28278620E+03,  .91375647E+02,
* -.10971428E+03, -.20471403E+03, -.88247643E+02,  .94434061E+00,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
```

```
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .68735678E+01,  .00000000   ,
* -.21551297E+02,  .00000000   ,  .91375647E+02,  .13165714E+03,
*   .14424284E+03,  .00000000   , -.12340741E+03, -.42666666E+01/
 DATA ((DM11(I,J),J=1,20),I=17,18)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .76799999E+02,  .17518769E+01,
* -.55669486E+02, -.12528272E+03, -.10971428E+03,  .14424284E+03,
*   .49910220E+03,  .29044056E+03, -.17066666E+03, -.12214004E+02,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .19051143E+02,  .00000000   ,
* -.90517002E+02, -.11702857E+03, -.20471403E+03,  .00000000   ,
*   .29044056E+03,  .35628698E+03,  .22982660E+03,  .00000000   /
 DATA ((DM11(I,J),J=1,20),I=19,20)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   , -.99111829E+02, -.76690757E+01,
* -.10971428E+03,  .13998608E+02, -.88247643E+02, -.12340741E+03,
* -.17066666E+03,  .22982660E+03,  .91858730E+03,  .18425132E+02,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   , -.14608700E+02,  .00000000   ,
* -.12572403E+02,  .00000000   ,  .94434061E+00, -.42666666E+01,
* -.12214004E+02,  .00000000   ,  .18425132E+02,  .26666666E+01/
 DATA ((DM12(I,J),J=1,20),I= 1, 2)/
* -.26445231E+04,  .34465863E+03, -.12800000E+04, -.31555515E+03,
*   .18042182E-12,  .10019021E+03, -.18432000E+03, -.50426580E+03,
* -.57506644E+03, -.52581444E+02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .34465863E+03, -.17066666E+03, -.55259457E+03,  .00000000   ,
*   .15840115E+03,  .10240000E+03,  .61377748E+02,  .00000000   ,
* -.22540745E+02,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM12(I,J),J=1,20),I= 3, 4)/
* -.12800000E+04, -.55259457E+03, -.60209355E+04, -.71131093E+03,
*   .39822222E+03,  .70071186E+03,  .00000000   , -.83306949E+03,
* -.11493877E+04, -.25794052E+02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
* -.31555515E+03,  .00000000   , -.71131093E+03, -.13653333E+03,
* -.17294342E+03,  .00000000   , -.93396413E+02, -.14628571E+03,
* -.20521405E+03,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM12(I,J),J=1,20),I= 5, 6)/
*   .18042182E-12,  .15840115E+03,  .39822222E+03, -.17294342E+03,
* -.13436629E+04, -.64920827E+03, -.61439999E+03, -.73842180E+01,
*   .00000000   ,  .84762894E+01,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .10019021E+03,  .10240000E+03,  .70071186E+03,  .00000000   ,
* -.64920827E+03, -.40959999E+03, -.44875663E+03,  .00000000   ,
*   .41289770E+02,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM12(I,J),J=1,20),I= 7, 8)/
* -.18432000E+03,  .61377748E+02,  .00000000   , -.93396413E+02,
* -.61439999E+03, -.44875663E+03, -.93518523E+03, -.44641607E+03,
* -.38619428E+03, -.11394236E+02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
```

```
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
* -.50426580E+03,  .00000000    , -.83306949E+03, -.14628571E+03,
* -.73842180E+01,  .00000000    , -.44641607E+03, -.55133459E+03,
* -.67334881E+03, -.85333332E+01,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DM12(I,J),J=1,20),I= 9,10)/
* -.57506644E+03, -.22540745E+02, -.11493877E+04, -.20521405E+03,
*  .00000000    ,  .41289770E+02, -.38619428E+03, -.67334881E+03,
* -.91496400E+03, -.80363275E+01,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
* -.52581444E+02,  .00000000    , -.25794052E+02,  .00000000    ,
*  .84762894E+01,  .00000000    , -.11394236E+02, -.85333332E+01,
* -.80363275E+01,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DM12(I,J),J=1,20),I=11,12)/
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    , -.29383590E+03,  .17527148E+02,
* -.85333332E+02,  .95756644E+01,  .00000000    , -.56007752E+02,
* -.10240000E+03,  .44711761E+01,  .12198379E+03,  .14484738E+02,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .17527148E+02,  .00000000    ,
* -.51588103E+01,  .00000000    ,  .36326954E+01,  .00000000    ,
* -.63301309E+01, -.85333332E+01, -.51140266E+01,  .00000000    /
 DATA ((DM12(I,J),J=1,20),I=13,14)/
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    , -.85333332E+02, -.51588103E+01,
* -.24083742E+03, -.16487320E+03, -.10240000E+03, -.23475053E+02,
*  .00000000    ,  .61659740E+02,  .14628571E+03,  .13193634E+02,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .95756644E+01,  .00000000    ,
* -.16487320E+03, -.27306666E+03, -.23773603E+03,  .00000000    ,
*  .17638674E+03,  .14628571E+03,  .12359798E+02,  .00000000    /
 DATA ((DM12(I,J),J=1,20),I=15,16)/
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .36326954E+01,
* -.10240000E+03, -.23773603E+03, -.24679523E+03, -.49958477E+02,
*  .14628571E+03,  .13978589E+03,  .00000000    , -.47150583E+01,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    , -.56007752E+02,  .00000000    ,
* -.23475053E+02,  .00000000    , -.49958477E+02, -.11702857E+03,
* -.10635868E+03,  .00000000    ,  .12718586E+03,  .00000000    /
 DATA ((DM12(I,J),J=1,20),I=17,18)/
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    , -.10240000E+03, -.63301309E+01,
* -.28863742E+03, -.14709837E+03,  .22755555E+03,  .39758303E+01,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .44711761E+01, -.85333332E+01,
* -.14709837E+03, -.18724571E+03, -.19860843E+02,  .00000000    ,
 DATA ((DM12(I,J),J=1,20),I=19,20)/
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
```

```
*   .00000000    ,  .00000000    ,  .12198379E+03, -.51140266E+01,
*   .14629571E+03,  .12359798E+02,  .00000000    ,  .12718586E+03,
*   .22755555E+03, -.19860843E+02, -.37052261E+03, -.83160590E+01,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .14484738E+02,  .00000000    ,
*   .13193634E+02,  .00000000    , -.47150583E+01,  .00000000    ,
*   .39758303E+01,  .00000000    , -.83160590E+01,  .00000000    /
 DATA ((DM22(I,J),J=1,20),I= 1, 2)/
*   .17630154E+04, -.34763371E+03,  .00000000    ,  .11476071E+03,
*   .00000000    , -.89404488E+02,  .00000000    ,  .17255978E+03,
*   .38337762E+03,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
* -.34763371E+03,  .17066666E+03,  .52774535E+03,  .00000000    ,
* -.88014421E+02,  .00000000    ,  .57251956E+02,  .00000000    ,
* -.10454474E+03,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DM22(I,J),J=1,20),I= 3, 4)/
*   .00000000    ,  .52774535E+03,  .40139570E+04,  .51189231E+03,
*   .00000000    , -.19444553E+03,  .00000000    ,  .17492360E+03,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .11476071E+03,  .00000000    ,  .51189231E+03,  .15928889E+03,
*   .24255361E+03,  .00000000    , -.74699907E+02,  .00000000    ,
*   .61930503E+02,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DM22(I,J),J=1,20),I= 5, 6)/
*   .00000000    , -.88014421E+02,  .00000000    ,  .24255361E+03,
*   .89577527E+03,  .25525935E+03,  .00000000    , -.10626095E+03,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
* -.89404488E+02,  .00000000    , -.19444553E+03,  .00000000    ,
*   .25525935E+03,  .17554285E+03,  .21340075E+03,  .00000000    ,
* -.80384423E+02,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DM22(I,J),J=1,20),I= 7, 8)/
*   .00000000    ,  .57251956E+02,  .00000000    , -.74699907E+02,
*   .00000000    ,  .21340075E+03,  .62345683E+03,  .23706438E+03,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .17255978E+03,  .00000000    ,  .17492360E+03,  .00000000    ,
* -.10626095E+03,  .00000000    ,  .23706438E+03,  .21585270E+03,
*   .23485290E+03,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DM22(I,J),J=1,20),I= 9,10)/
*   .38337762E+03, -.10454474E+03,  .00000000    ,  .61930503E+02,
*   .00000000    , -.80384423E+02,  .00000000    ,  .23485290E+03,
*   .60997600E+03,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*   .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DM22(I,J),J=1,20),I=11,12)/
```

```
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .19589060E+03,  .00000000   ,
*   .00000000   , -.38253571E+02,  .00000000   ,  .29801496E+02,
*   .00000000   , -.57519926E+02, -.81322526E+02,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DM22(I,J),J=1,20),I=13,14)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .16055828E+03,  .10237846E+03,  .00000000   , -.38889106E+02,
*   .00000000   ,  .34984719E+02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   , -.38253571E+02,  .00000000   ,
*   .10237846E+03,  .15928889E+03,  .10395155E+03,  .00000000   ,
* -.41499949E+02,  .00000000   ,  .39410320E+02,  .00000000   /
 DATA ((DM22(I,J),J=1,20),I=15,16)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .10395155E+03,  .16453015E+03,  .10939686E+03,
*   .00000000   , -.45540409E+02,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .29801496E+02,  .00000000   ,
* -.38889106E+02,  .00000000   ,  .10939686E+03,  .17554285E+03,
*   .11855597E+03,  .00000000   , -.51153724E+02,  .00000000   /
 DATA ((DM22(I,J),J=1,20),I=17,18)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   , -.41499949E+02,  .00000000   ,  .11855597E+03,
*   .19242495E+03,  .13170243E+03,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .34984719E+02,  .00000000   , -.57519926E+02,  .00000000   ,
*   .13170243E+03,  .21585270E+03,  .14945185E+03,  .00000000   /
 DATA ((DM22(I,J),J=1,20),I=19,20)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   , -.81322526E+02,  .00000000   ,
*   .00000000   ,  .39410320E+02,  .00000000   , -.51153724E+02,
*   .00000000   ,  .14945185E+03,  .24701507E+03,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   /
 DATA ((DA1(I,J),J=1,20),I= 1, 2)/
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .10000000E+01,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .10000000E+01,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
```

```
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000  /
 DATA ((DA1(I,J),J=1,20), I= 3, 4)/
*  .00000000   ,  .80000000E+01,  .00000000   , -.79999999E+01,
*  .00000000   ,  .80000000E+01,  .00000000   , -.80000000E+01,
*  .00000000   ,  .10000000E+01, -.80000000E+01,  .00000000   ,
*  .80000000E+01,  .00000000   , -.80000000E+01,  .00000000   ,
*  .80000000E+01,  .00000000   , -.80000000E+01,  .00000000   ,
* -.24000000E+02,  .00000000   , -.40000000E+02,  .00000000   ,
*  .18666667E+02,  .00000000   , -.14400000E+02,  .00000000   ,
*  .12571429E+02,  .00000000   ,  .00000000   ,  .10000000E+01,
*  .00000000   , -.80000000E+01,  .00000000   ,  .80000000E+01,
*  .00000000   , -.80000000E+01,  .00000000   , -.10000000E+01/
 DATA ((DA1(I,J),J=1,20), I= 5, 6)/
*  .10122400E+02,  .80000000E+01,  .64036002E+02,  .79999999E+01,
* -.35525969E+02, -.40000000E+02, -.13668772E+02,  .64000000E+02,
*  .99341473E+02,  .99999999E+00, -.94330671E+01,  .00000000   ,
*  .24182149E+01,  .16000000E+02,  .22819177E+02, -.80000000E+01,
* -.55623538E+02, -.32000000E+02,  .58327284E+02,  .99999999E+00,
* -.24437636E+02,  .00000000   , -.39396466E+02,  .00000000   ,
*  .19264704E+02,  .79999999E+01, -.10624989E+02, -.32000000E+02,
* -.18839669E+02,  .00000000   , -.26658547E+00,  .99999999E+00,
* -.37700878E+00, -.80000000E+01, -.23535304E+01,  .80000000E+01,
*  .17891270E+02,  .00000000   , -.48654425E+02, -.10000000E+01/
 DATA ((DA1(I,J),J=1,20), I= 7, 8)/
*  .12727922E+02,  .00000000   ,  .35355339E+02,  .16000000E+02,
*  .23098821E+02,  .00000000   , -.22910259E+02, -.24000000E+02,
* -.24445691E+02,  .10000000E+01, -.70710678E+01,  .00000000   ,
* -.98994949E+01,  .00000000   ,  .12727922E+02,  .16000000E+02,
*  .15556349E+02,  .00000000   , -.18384776E+02,  .10000000E+01,
* -.12727922E+02,  .00000000   ,  .70710678E+01,  .00000000   ,
*  .32998316E+01,  .80000000E+01,  .76367532E+01,  .00000000   ,
* -.11111678E+02,  .00000000   , -.14142136E+01,  .10000000E+01,
* -.14142136E+01,  .00000000   , -.42426407E+01,  .00000000   ,
*  .70710678E+01,  .80000000E+01,  .98994949E+01,  .00000000  /
 DATA ((DA1(I,J),J=1,20), I= 9,10)/
*  .15114992E+02, -.79999999E+01,  .78081076E+00,  .79999999E+01,
*  .44749737E+02,  .40000000E+02,  .77748024E+02,  .64000000E+02,
*  .64033104E+02,  .99999999E+00, -.51944928E+01,  .00000000   ,
* -.19022296E+02, -.16000000E+02, -.24014888E+02, -.79999999E+01,
*  .24450080E+01,  .32000000E+02,  .70498738E+02,  .99999999E+00,
* -.62608347E+01,  .00000000   ,  .29190435E+02,  .00000000   ,
*  .26000341E+02,  .80000000E+01,  .41472043E+02,  .32000000E+02,
*  .65014694E+02,  .00000000   , -.37511422E+01,  .99999999E+00,
* -.53049161E+01, -.79999999E+01, -.11897021E+02, -.79999999E+01,
* -.18332963E+02,  .00000000   , -.55904472E+01,  .99999999E+00/
 DATA ((DA1(I,J),J=1,20), I=11,12)/
*  .12000000E+02, -.80000000E+01, -.20000000E+02, -.79999999E+01,
* -.93333333E+01, -.80000000E+01, -.72000000E+01, -.80000000E+01,
* -.62857143E+01,  .10000000E+01,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*  .00000000   ,  .00000000   , -.40000000E+01,  .10000000E+01,
* -.40000000E+01, -.80000000E+01, -.40000000E+01, -.80000000E+01,
* -.40000000E+01, -.80000000E+01, -.40000000E+01,  .10000000E+01/
 DATA ((DA1(I,J),J=1,20), I=13,14)/
*  .15114992E+02, -.79999999E+01,  .78081076E+00,  .79999999E+01,
*  .44749737E+02,  .40000000E+02,  .77748024E+02,  .64000000E+02,
*  .64033104E+02,  .99999999E+00,  .51944928E+01,  .00000000   ,
*  .19022296E+02,  .16000000E+02,  .24014888E+02,  .79999999E+01,
* -.24450080E+01, -.32000000E+02, -.70498738E+02, -.99999999E+00,
*  .62608347E+01,  .00000000   , -.29190435E+02,  .00000000   ,
* -.26000341E+02, -.80000000E+01, -.41472043E+02, -.32000000E+02,
```

```
* -.65014694E+02,  .00000000    , -.37511422E+01,  .99999999E+00,
* -.53049161E+01, -.79999999E+01, -.11897021E+02, -.79999999E+01,
* -.18332963E+02,  .00000000    , -.55904472E+01,  .99999999E+00/
 DATA ((DA1(I,J),J=1,20),I=15,16)/
*  .12727922E+02,  .00000000    ,  .35355339E+02,  .16000000E+02,
*  .23098821E+02,  .00000000    , -.22910259E+02, -.24000000E+02,
* -.24445691E+02,  .10000000E+01,  .70710678E+01,  .00000000    ,
*  .98994949E+01,  .00000000    , -.12727922E+02, -.16000000E+02,
* -.15556349E+02,  .00000000    ,  .18384776E+02, -.10000000E+01,
*  .12727922E+02,  .00000000    , -.70710678E+01,  .00000000    ,
* -.32998316E+01, -.80000000E+01, -.76367532E+01,  .00000000    ,
*  .11111678E+02,  .00000000    , -.14142136E+01,  .10000000E+01,
* -.14142136E+01,  .00000000    , -.42426407E+01,  .00000000    ,
*  .70710678E+01,  .80000000E+01,  .98994949E+01,  .00000000    /
 DATA ((DA1(I,J),J=1,20),I=17,18)/
*  .10122400E+02,  .80000000E+01,  .64036002E+02,  .79999999E+01,
* -.35525969E+02, -.40000000E+02, -.13668772E+02,  .64000000E+02,
*  .99341473E+02,  .99999999E+00,  .94330671E+01,  .00000000    ,
* -.24182149E+01, -.16000000E+02, -.22819177E+02,  .80000000E+01,
*  .55623538E+02,  .32000000E+02, -.58327284E+02, -.99999999E+00,
*  .24437636E+02,  .00000000    ,  .39396466E+02,  .00000000    ,
* -.19264704E+02, -.79999999E+01,  .10624989E+02,  .32000000E+02,
*  .18839069E+02,  .00000000    , -.26658547E+00,  .99999999E+00,
* -.37703878E+00, -.80000000E+01, -.23535304E+01,  .80000000E+01,
*  .17891270E+02,  .00000000    , -.48654425E+02, -.10000000E+01/
 DATA ((DA1(I,J),J=1,20),I=19,20)/
*  .00000000    ,  .80000000E+01,  .00000000    , -.79999999E+01,
*  .00000000    ,  .80000000E+01,  .00000000    , -.80000000E+01,
*  .00000000    ,  .10000000E+01,  .80000000E+01,  .00000000    ,
* -.80000000E+01,  .00000000    ,  .80000000E+01,  .00000000    ,
* -.80000000E+01,  .00000000    ,  .80000000E+01,  .00000000    ,
*  .24000000E+02,  .00000000    ,  .40000000E+02,  .00000000    ,
* -.18666667E+02,  .00000000    ,  .14400000E+02,  .00000000    ,
* -.12571429E+02,  .00000000    ,  .00000000    ,  .10000000E+01,
*  .00000000    , -.80000000E+01,  .00000000    ,  .80000000E+01,
*  .00000000    , -.80000000E+01,  .00000000    , -.10000000E+01/
 DATA ((DA2(I,J),J=1,20),I= 1, 2)/
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .00000000    ,  .00000000    ,  .00000000    /
 DATA ((DA2(I,J),J=1,20),I= 3, 4)/
*  .00000000    , -.80000000E+01,  .00000000    ,  .79999999E+01,
*  .00000000    , -.80000000E+01,  .00000000    ,  .79999999E+01,
*  .00000000    ,  .00000000    ,  .80000000E+01,  .00000000    ,
* -.80000000E+01,  .00000000    ,  .80000000E+01,  .00000000    ,
* -.80000000E+01,  .00000000    ,  .80000000E+01,  .00000000    ,
*  .24000000E+02,  .00000000    ,  .40000000E+02,  .00000000    ,
* -.18666667E+02,  .00000000    ,  .14400000E+02,  .00000000    ,
* -.12571429E+02,  .00000000    ,  .00000000    ,  .00000000    ,
*  .00000000    ,  .80000000E+01,  .00000000    ,  .00000000    ,
*  .00000000    ,  .80000000E+01,  .00000000    , -.80000000E+01,
 DATA ((DA2(I,J),J=1,20),I= 5, 6)/
* -.10922157E+02, -.80000000E+01, -.62150959E+02,  .00000000    ,
*  .41017540E+02,  .16000000E+02, -.18535514E+02, -.32000000E+02,
* -.22884519E+02,  .00000000    ,  .87894728E+01,  .00000000    ,
* -.51487540E+01, -.16000000E+02, -.72814378E+01,  .16000000E+02,
*  .24860384E+02,  .00000000    , -.35157891E+02,  .00000000    ,
```

```
*   .26368419E+02, -.79999999E+01,  .25743770E+02,  .16000000E+02,
*   .16990021E+02, -.16000000E+02, -.44748690E+02,  .00000000   ,
*   .55248115E+02,  .00000000   ,  .36407189E+01,  .00000000   ,
* -.12430192E+02,  .00000000   ,  .17578946E+02,  .16000000E+02,
* -.10297508E+02, -.32000000E+02, -.14562876E+02,  .00000000   /
 DATA ((DA2(I,J),J=1,20),I= 7, 8)/
* -.16970563E+02,  .00000000   , -.28284271E+02, -.79999999E+01,
* -.13199327E+02,  .00000000   ,  .10182338E+02,  .79999999E+01,
*   .88893423E+01,  .00000000   ,  .56568542E+01,  .00000000   ,
*   .56568542E+01,  .00000000   , -.56568542E+01, -.80000000E+01,
* -.56568542E+01,  .00000000   ,  .56568542E+01,  .00000000   ,
*   .16970563E+02, -.80000000E+01, -.28284271E+02,  .00000000   ,
*   .13199327E+02,  .80000000E+01,  .10182338E+02,  .00000000   ,
* -.88893423E+01,  .00000000   ,  .56568542E+01,  .00000000   ,
* -.56568542E+01, -.80000000E+01, -.56568542E+01,  .00000000   ,
*   .56568542E+01,  .80000000E+01,  .56568542E+01,  .00000000   /
 DATA ((DA2(I,J),J=1,20),I= 9,10)/
* -.26368419E+02,  .79999999E+01,  .25743770E+02,  .00000000   ,
* -.16990021E+02, -.16000000E+02, -.44748690E+02, -.32000000E+02,
* -.55248115E+02,  .00000000   ,  .36407189E+01,  .00000000   ,
*   .12430192E+02,  .16000000E+02,  .17578946E+02,  .16000000E+02,
*   .10297508E+02,  .00000000   , -.14562876E+02,  .00000000   ,
*   .10922157E+02, -.79999999E+01, -.62150959E+02, -.16000000E+02,
* -.41017540E+02, -.16000000E+02, -.18535514E+02,  .00000000   ,
*   .22884519E+02,  .00000000   ,  .87894728E+01,  .00000000   ,
*   .51487540E+01,  .00000000   , -.72814378E+01, -.16000000E+02,
* -.24860384E+02, -.32000000E+02, -.35157891E+02,  .00000000   /
 DATA ((DA2(I,J),J=1,20),I=11,12)/
* -.24000000E+02,  .80000000E+01,  .48000000E+02,  .79999999E+01,
*   .18666667E+02,  .80000000E+01,  .14400000E+02,  .79999999E+01,
*   .12571429E+02,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .00000000   ,  .00000000   ,
*   .00000000   ,  .00000000   ,  .80000000E+01,  .00000000   ,
*   .80000000E+01,  .80000000E+01,  .80000000E+01,  .80000000E+01,
*   .80000000E+01,  .80000000E+01,  .80000000E+01,  .00000000   /
 DATA ((DA2(I,J),J=1,20),I=13,14)/
* -.26368419E+02,  .79999999E+01,  .25743770E+02,  .00000000   ,
* -.16990021E+02, -.16000000E+02, -.44748690E+02, -.32000000E+02,
* -.55248115E+02,  .00000000   , -.36407189E+01,  .00000000   ,
* -.12430192E+02, -.16000000E+02, -.17578946E+02, -.16000000E+02,
* -.10297508E+02,  .00000000   ,  .14562876E+02,  .00000000   ,
* -.10922157E+02,  .79999999E+01,  .62150959E+02,  .16000000E+02,
*   .41017540E+02,  .16000000E+02,  .18535514E+02,  .00000000   ,
* -.22884519E+02,  .00000000   ,  .87894728E+01,  .00000000   ,
*   .51487540E+01,  .00000000   , -.72814378E+01, -.16000000E+02,
* -.24860384E+02, -.32000000E+02, -.35157891E+02,  .00000000   /
 DATA ((DA2(I,J),J=1,20),I=15,16)/
* -.16970563E+02,  .00000000   , -.28284271E+02, -.79999999E+01,
* -.13199327E+02,  .00000000   ,  .10182338E+02,  .79999999E+01,
*   .88893423E+01,  .00000000   , -.56568542E+01,  .00000000   ,
* -.56568542E+01,  .00000000   ,  .56568542E+01,  .80000000E+01,
*   .56568542E+01,  .00000000   , -.56568542E+01,  .00000000   ,
* -.16970563E+02,  .80000000E+01,  .28284271E+02,  .00000000   ,
* -.13199327E+02, -.80000000E+01, -.10182338E+02,  .00000000   ,
*   .88893423E+01,  .00000000   ,  .56568542E+01,  .00000000   ,
* -.56568542E+01, -.80000000E+01, -.56568542E+01,  .00000000   ,
*   .56568542E+01,  .80000000E+01,  .56568542E+01,  .00000000   /
 DATA ((DA2(I,J),J=1,20),I=17,18)/
* -.10922157E+02, -.80000000E+01, -.62150959E+02,  .00000000   ,
*   .41017540E+02,  .16000000E+02, -.18535514E+02, -.32000000E+02,
* -.22884519E+02,  .00000000   , -.87894728E+01,  .00000000   ,
```

```
     *  .51487540E+01,  .16000000E+02,  .72814378E+01, -.16000000E+02,
     * -.24860384E+02,  .00000000   ,  .35157891E+02,  .00000000   ,
     * -.26368419E+02,  .79999999E+01, -.25743770E+02, -.16000000E+02,
     * -.16990021E+02,  .16000000E+02,  .44748690E+02,  .00000000   ,
     * -.55248115E+02,  .00000000   ,  .36407189E+01,  .00000000   ,
     * -.12430192E+02,  .00000000   ,  .17578946E+02,  .16000000E+02,
     * -.10297508E+02, -.32000000E+02, -.14562876E+02,  .00000000   /
      DATA ((DA2(I,J),J=1,20),I=19,20)/
     *  .00000000   , -.80000000E+01,  .00000000   ,  .79999999E+01,
     *  .00000000   , -.80000000E+01,  .00000000   ,  .79999999E+01,
     *  .00000000   ,  .00000000   , -.80000000E+01,  .00000000   ,
     *  .80000000E+01,  .00000000   , -.80000000E+01,  .00000000   ,
     *  .80000000E+01,  .00000000   , -.80000000E+01,  .00000000   ,
     * -.24000000E+02,  .00000000   , -.40000000E+02,  .00000000   ,
     *  .18666667E+02,  .00000000   , -.14400000E+02,  .00000000   ,
     *  .12571429E+02,  .00000000   ,  .00000000   ,  .00000000   ,
     *  .00000000   ,  .80000000E+01,  .00000000   , -.80000000E+01,
     *  .00000000   ,  .80000000E+01,  .00000000   ,  .00000000   /
      NP=10
      IP=2*NP
      IP2=3*NP
      DO 800 KT=1,NCRK10
      NORHO=0
      IF(CPRP(1,KT).LE.0) GO TO 800
      IF(CPRP(3,KT).LE.0) GO TO 800
      IF(CPRP(4,KT).LE.0) NORHO=1
      SIG=CPRP(2,KT)
      G=CPRP(1,KT)/(2.*(1.+SIG))
      IF(JC(11,KT).GT.0) SIG=SIG/(1.+SIG)
      THICK=CPRP(3,KT)
      RHO=CPRP(4,KT)
      WTS=DL*G*THICK
      IF(ICPRT.GT.0) WRITE(IOUT1,2001)
     *                    CPRP(1,KT),G,CPRP(2,KT),SIG,RHO,THICK
 2001 FORMAT(//10X,"E,G,NU,SIG,RHO,THICK",/10X,1P6E10.3/)
      J1=JC(1,KT)
      J6=JC(6,KT)
      J8=JC(8,KT)
      A(3)=0.
      B(3)=0.
      DO 170 I=1,NDOF
      A(I)=CO(I,J6)-CO(I,J1)
      B(I)=CO(I,J8)-CO(I,J1)
  170 CONTINUE
      CALL UNIVEC (A,NDOF,DEL)
      CALL UNIVEC (B,NDOF,DEL1)
      IF (ABS(DEL1-DEL).GT.(5.E-06*(DEL+DEL1))) WRITE (6,1001) KT
 1001 FORMAT(//10X,"**** WARNING ****  10-NODE ELEMENT NUMBER ",I3,
     *       //27X,"IS NOT RECTANGULAR WITHIN (-06)")
      WTM=THICK*DEL*DEL*RHO
      DO 160 I=1,IP
      DO 160 J=1,IP
      S2(I,J)=DA1(I,J)+SIG*DA2(I,J)
  160 T1(I,J)=S2(I,J)
      IF(ICPRT.LT.3) GO TO 201
      WRITE(IOUT1,2003)
      CALL PRTRM(IP,S2,IP,IP,IOUT1)
 2003 FORMAT(//10X,"A IS"/)
  201 CONTINUE
      CALL INVERT (S2,S1,IP,IP)
      IF(ICPRT.LT.3) GO TO 202
      WRITE(IOUT1,2004)
      CALL PRTRM(IP,S1,IP,IP,IOUT1)
      WRITE(IOUT1,2005)
```

```
      CALL MABC(IP,S1,IP,T1,IP2,T3,IP,IP,IP)
      CALL PRTRM(IP2,T3,IP,IP,IOUT1)
 2004 FORMAT(//10X,"AINV IS"/)
 2005 FORMAT(//10X,"A*AINV IS"/)
      DO 159 I=1,IP2
      DO 159 J=1,IP2
  159 T3(I,J)= 0.
  202 CONTINUE
      DO 161 I=1,IP
      DO 161 J=1,IP2
  161 S2(I,J)=0.
      DO 162 I=1,NP
      LOC=2*I-1
      LOC1=NDOF*(I-1)
      DO 162 J=1,NDOF
      S2(LOC,LOC1+J)=A(J)
      S2(LOC+1,LOC1+J)=B(J)
  162 CONTINUE
      LOC=NDOF*NP
      IF(ICPRT.LT.2) GO TO 203
      WRITE(IOUT1,2020)
      CALL PRTRM(IP,S2,IP,LOC,IOUT1)
 2020 FORMAT(//10X,"THE T MATRIX IS"/)
  203 CONTINUE
      CALL MABC (IP,S1,IP,S2,IP,T1,IP,IP,LOC)
C (AINV*T) IS IN T1
      IF(IDYNAM.GT.0) WRITE (26) DEL,G,THICK,((T1(J,K),J=1,IP),K=1,LO
     *C)
      DO 163 I=1,IP
      DO 163 J=1,IP
  163 S3(I,J)=(DK1(I,J)+SIG*DK2(I,J))*WTS
      DO 505 I=1,IP
      S3(10,I)=0.0
      S3(I,10)=0.0
      S3(12,I)=0.0
      S3(I,12)=0.0
      S3(20,I)=0.0
      S3(I,20)=0.0
  505 CONTINUE
      CALL ATBA(IP,T1,IP,S3,IP2,T2,LOC,IP)
      IF(ICPRT.LT.3) GO TO 204
      WRITE(IOUT1,2021)
      CALL PRTRM(IP2,T2,LOC,LOC,IOUT1)
 2021 FORMAT(//10X,"THE GK MATRIX IS"/)
  204 CONTINUE
      IF(IDYNAM.EQ.0) WRITE (26) DEL,G,THICK,S1,S2,T2
      IF(NORHO.EQ.1) GO TO 165
      IF(IDYNAM.LE.0.OR.IDYNAM.GT.2)GO TO 165
      SIGSQ=SIG*SIG
      DO 164 I=1,IP
      DO 164 J=1,IP
  164 S2(I,J)=(DM11(I,J)+SIG*DM12(I,J)+SIGSQ*DM22(I,J))*WTM
C  CM IS NOW IN S2
      CALL ATBA (IP,T1,IP,S2,IP2,T3,LOC,IP)
      IF(ICPRT.LT.3) GO TO 508
      SUMO=0.
      SUME=0.
      DO 506 I=1,NP
      SUMODD(I)=0.
      SUMEVN(I)=0.
      I2=2*I
      I1=I2-1
      DO 506 J=1,NP
      J2=2*J
```

```
      J1=J2-1
      SUMODD(I)=SUMODD(I)+T3(J1,I1)
      SUMEVN(I)=SUMEVN(I)+T3(J2,I2)
  506 CONTINUE
      DO 507 J=1,NP
      SUMO=SUMO+SUMODD(J)
      SUME=SUME+SUMEVN(J)
  507 CONTINUE
      DO 509 I=1,NP
      SUMODD(I)=(SUMODD(I)/SUMO)*100.
  509 SUMEVN(I)=(SUMEVN(I)/SUME)*100.
      WRITE(6,730) SUMO,SUME
      WRITE(6,731) (SUMODD(K),K=1,NP)
      WRITE(6,732) (SUMEVN(K),K=1,NP)
  730 FORMAT(//10X,"XMASS IS%  ",1PE11.3,/10X,"YMASS IS%  ",
     *1PE11.3/)
  731 FORMAT(//10X,"U MASSES ARE (PERCENT)%  ",2(/10X,1P5E11.3)/)
  732 FORMAT(//10X,"V MASSES ARE (PERCENT)%  ",2(/10X,1P5E11.3)/)
  508 CONTINUE
      IF(ICPRT.LT.3) GO TO 205
      WRITE(IOUT1,2006)
      CALL PRTRM(IP2,T3,LOC,LOC,IOUT1)
 2006 FORMAT(//10X,"THE GM MATRIX IS"/)
  205 CONTINUE
      GO TO 167
  165 DO 166  I=1,IP2
      DO 166  J=1,IP2
  166 T3(I,J)=0.
      IF(IDYNAM.LT.3)GO TO 167
      DO 169 I=1,NP
      DO 168 J=1,NDOF
      K=J+NDOF*(I-1)
      T3(K,K)=(20.0*WTM)/9.0
      IF(I.EQ.1)T3(K,K)=(2.0*WTM)/9.0
  168 CONTINUE
  169 CONTINUE
  167 CONTINUE
      DO 91 I=1,NP
      LOC=JC(I,KT)
      LOC2=NDOF*(I-1)
      DO 91 J=1,NDOF
      LOCATE(J+LOC2)=NF(J,LOC)
   91 CONTINUE
      LOC=NP*NDOF
      IF(ICPRT.GT.1)  WRITE(IOUT1,2007)KT,(LOCATE(J),J=1,LOC)
 2007 FORMAT(//1X,"  LOCATE(J)  ",/1X,I5,10I5,2(/6X,10I5)/)
      DO 112 I=1,LOC
      IF(LOCATE(I).LE.0) GO TO 112
      IR=LOCATE(I)
      DO 111 J=1,LOC
      IF(LOCATE(J)) 130,111,140
  130 IC= -LOCATE(J)
      P(IR)=P(IR)-T2(I,J)*P(IC)
      GO TO 111
  140 IC=LOCATE(J)
      IF(IC.LT.IR) GO TO 111
      IC=IC-IR+1
      ST(IR,IC)=ST(IR,IC) + T2(I,J)
      IF(IDYNAM.GT.2)IC=1
      IF(IDYNAM.GT.0)SM(IR,IC)=SM(IR,IC) + T3(I,J)
  111 CONTINUE
  112 CONTINUE
  800 CONTINUE
      END FILE 26
```

```
      RETURN
      END

PRTRM
CPRTRM
      SUBROUTINE PRTRM(AR,A,PR,PC,NFILE)
      INTEGER PR,PC,AR
      DIMENSION A(AR,1)
201   FORMAT(1H ,I3,1P10E11.3)
202   FORMAT(1H ,6X,I4,9(7X,I4))
      KS = PC/10
      IF (KS.EQ.0) GO TO 555
      IREM = 10*KS + 1
      DO 1003 K = 1,KS
      L1 = 10*K - 9
      L2 = 10*K
      WRITE(NFILE,202) (I,I=L1,L2)
      DO 1003 I=1,PR
1003  WRITE(NFILE,201) I,(A(I,J),J=L1,L2)
      IF (IREM.GT.PC) GO TO 556
      WRITE(NFILE,202) (I,I=IREM,PC)
      DO 1005 I=1,PR
1005  WRITE(NFILE,201) I,(A(I,J),J=IREM,PC)
      GO TO 556
555   WRITE(NFILE,202) (I,I=1,PC)
      DO 1006 I=1,PR
1006  WRITE(NFILE,201) I,(A(I,J),J=1,PC)
556   CONTINUE
      RETURN
      END
```

```
DOT
CDOT
      FUNCTION DOT (V1, V2, N)
C
C  THIS FUNCTION EVALUATES THE DOT PRODUCT OF THE VECTORS V1 AND V2
C
      DOUBLE PRECISION DOTD
      DIMENSION V1(1), V2(1)
      DOTD=0.0
      DO 10 I= 1,N
10    DOTD= DOTD + DBLE(V1(I))*DBLE(V2(I))
      DOT=DOTD
      RETURN
      END
```

```
CROSS
CCROSS
      SUBROUTINE CROSS (A,B,AXB)
      DIMENSION A(1),B(1),AXB(1)
      DOUBLE PRECISION Z
      Z=A(2)*B(3)-A(3)*B(2)
      AXB(1)=Z
      Z=A(3)*B(1)-A(1)*B(3)
      AXB(2)=Z
      Z=A(1)*B(2)-A(2)*B(1)
      AXB(3)=Z
      RETURN
      END
```

```
ATBA
CATBA
      SUBROUTINE ATBA (AR,A,BR,B,CR,C,N,NB)
C  THIS SUBPROGRAM FORMS THE MATRIX PRODUCT (AT*B*A=C)
      INTEGER AR,BR,CR
      DIMENSION A(AR,1),B(BR,1),C(CR,1)
      DOUBLE PRECISION Z
      DO 10 I= 1,N
      DO 10 J= 1,N
      Z=0.
      DO 11 K= 1,NB
      DO 11 L= 1,NB
 11   Z= Z + A(K,I)*B(K,L)*A(L,J)
 10   C(I,J)= Z
      RETURN
      END
```

```
MABC
CMABC
      SUBROUTINE MABC (AR,A,BR,B,CR,C,NRA,NCARB,NCB)
C  THIS SUBPROGRAM FORMS THE MATRIX PRODUCT (A*B=C)
      INTEGER AR,BR,CR
      DOUBLE PRECISION Z
      DIMENSION A(AR,1),B(BR,1),C(CR,1)
      DO 10 I= 1,NRA
      DO 10 K= 1,NCB
      Z=0.
      DO 11 J= 1,NCARB
11    Z= Z + A(I,J)*B(J,K)
10    C(I,K)=Z
      RETURN
      END
```

```
UNIVEC
CUNIVEC
      SUBROUTINE UNIVEC (A,N,XL)
      DIMENSION A(1)
      DOUBLE PRECISION XL1
      XL1=SQRT(DOT(A,A,N))
      XL=XL1
      IF(XL.LT.(1.E-08)) GO TO 11
      DO 10 I=1,N
 10   A(I)=SNGL(DBLE(A(I))/XL1)
      GO TO 12
 11   WRITE(6,100)
 100  FORMAT(//"*** PROBABLE ZERO VECTOR IN UNIVEC ***"//)
 12   CONTINUE
      RETURN
      END
```

```
DYNAM
COYNME4
      SUBROUTINE DYNAM (A,N)
      DIMENSION A(1),N(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRKS,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE,IDYNAM
      COMMON/ADR/NNF,NJT,NTP,NJS,NSP,NJ8,N8P,NJ10,N10P,NP1,NST,NSM,
     *           NDY,NP2,NUV,NUN
      IF(IDYNAM.EQ.1) CALL DYNAM1(A(NNF),A(NJ8),A(NJ10),A(NP1),A(NST),
     *                            A(NSM),A(NDY),A(NP2),A(NUV),
     *                           NDOF,NER)
      IF(IDYNAM.EQ.2) CALL DYNAM2(A(NNF),A(NJ8),A(NJ10),A(NP1),A(NST),
     *                            A(NSM),A(NDY),A(NP2),A(NUV),NP1,
     *                           NDOF,NER)
      IF(IDYNAM.EQ.3)CALL DYNAM3(A(NNF),A(NJ8),A(NJ10),A(NP1),A(NST),
     *                               A(NSM),A(NDY),A(NP2),A(NUV),
     *                          NDOF,NER)
      IF(IDYNAM.EQ.4)CALL DYNAM4(A(NNF),A(NJ8),A(NJ10),A(NP1),A(NST),
     *                             A(NSM),A(NDY),A(NP2),A(NUV),NP1,
     *                        NDOF,NER)
      RETURN
      END
```

```
DYNAM1
COYNAM1
      SUBROUTINE DYNAM1 (NF,JC8,JC10,P,ST,SM,DYN,P2,U,NDOF,NROW)
      DIMENSION NF(NDOF,1),P(1),ST(NROW,1),SM(NROW,1),DYN(NROW,1),
     *          P2(1),U(NDOF,1),JC8(9,1),JC10(11,1)
      DIMENSION V(30),S2(20,30),IN(3),C(20)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,N8,NDOFN,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DATA PI/3.1415926/
  300 WRITE(6,200)
      A1=1HN
  200 FORMAT(/5X,"ENTER% INITIAL FREQ, FINAL FREQ, NR OF STEPS",
     *       //5X,"FREQ IN HZ, FREE FORMAT, ORDER INDICATED")
      READ(5,*) WI,WF,NSTEP
      WRITE(6,201) WI, WF, NSTEP
  201 FORMAT(/5X,"THE PARAMETERS ARE%",/6X,"WI",9X,"WF",4X,"NSTEP",
     *       //1X,1P2E11.3,I5/)
      WRITE(6,202)
  202 FORMAT(/5X,"IF YOU WISH TO CORRECT THIS, TYPE% YES")
      READ(5,102) A1
  102 FORMAT(A1)
      IF(A1.EQ.1HY) GO TO 300
      WI=WI*PI*2.
      WF=WF*PI*2.
      DELW=WF-WI
      IF(NSTEP.LE.1) Z1=1.
      IF(NSTEP.LE.1) GO TO 799
      Z1=1./FLOAT(NSTEP-1)
  799 CONTINUE
      DO 800 KT=1,NSTEP
C CALCULATE OMEGA SQUARED
      W=WI+(KT-1)*DELW*Z1
      W2=W*W
C INITIALIZE P2 AND DYN
      DO 801 I=1,NER
      P2(I)=P(I)
      DO 801 J=1,N8
  801 DYN(I,J)=ST(I,J) - W2*SM(I,J)
C CALCULATE NODAL DISPLACEMENTS, RETURNED IN P2
      CALL BANS2L(DYN,P2,NER,N8,NER)
C RECOVER NODAL DISPLACEMENTS, IN NODAL POINT ORDER
      DO 802 I=1,NNODES
C ZERO THE NODAL DISPLACEMENTS
      DO 803 J=1,NDOF
  803 U(J,I)=0.
C RECOVER THE FREEDOM NUMBERS OF THE NODAL DISPLACEMENTS
      DO 804 J=1,NDOF
      CALL INTEGR(IN(J),NF(J,I))
  804 IN(J)=IABS(IN(J))
C IN(J) CONTAINS FREEDOM NUMBERS OF NODAL DISPL, OR ZERO
C RECOVER THE NODAL DISPL FROM P2
      DO 805 J=1,NDOF
      K=IN(J)
      IF(K.EQ.0) GO TO 805
      U(J,I)=P2(K)
  805 CONTINUE
  802 CONTINUE
C WRITE OUT THE DISPLACEMENT VECTOR
      W=W/(2.*PI)
      WRITE(IOUT1,203) W
      WRITE(6,203) W
```

```
  203  FORMAT(/5X,"THE DATA FOR OMEGA= ",1PE11.3," FOLLOWS%")
       WRITE(IOUT1,204)
       IF(IOPRT.GT.0) WRITE(6,204)
  204  FORMAT(/5X,"NODAL POINT DISPLACEMENTS",
      *       /5X,"NODE",7X,"U",10X,"V",10X,"W"/)
       IF(NDOF.EQ.2) GO TO 8051
       DO 8052 I=1,NNODES
       IF(IOPRT.GT.0) WRITE(6,205) I,(U(J,I),J=1,3)
  8052 WRITE(IOUT1,205) I,(U(J,I),J=1,3)
  205  FORMAT(/3X,I5,2X,1P3E11.3)
       GO TO 8053
  8051 DO 8054 I=1,NNODES
       IF(IOPRT.GT.0) WRITE(6,206) I,(U(J,I),J=1,2)
  8054 WRITE(IOUT1,206) I,(U(J,I),J=1,2)
  206  FORMAT(/3X,I5,2X,1P2E11.3)
  8053 I1=8*NDOF
       IF(NCRK8.LE.0) GO TO 806
       REWIND 25
       WRITE(IOUT1,207)
  207  FORMAT(/6X,"8-NODE ELEMENT STRESS INTENSITY FACTOR",
      *       /6X,"ELEMENT",7X,"K ONE")
C NODAL POINT DISP NOW IN U IN NODAL POINT ORDER
C RECOVER GEN. DISP. AND CALC STRESS INTENSITY FACTORS
       DO 807 I=1,NCRK8
C READ IN THE PARAMETERS AND GEN COORD RECOVERY MATRIX
C GENERATE THE ELEMENT DISPLACEMENT VECTOR
       DO 808 J=1,8
C FIND THE NODE NUMBERS FOR  THE ELEMENT
       K=JC8(J,I)
       I2=NDOF*(J-1)
C STUFF  DISPLACEMENTS INTO THE V VECTOR
       DO 809 L=1,NDOF
  809  V(I2+L)=U(L,K)
  808  CONTINUE
       READ(25) DEL,G,THK,((S2(J,K),J=1,16),K=1,I1)
C  MULTIPLY C=AINV*T*V
       DO 810 J=1,16
       C(J)=0.
       DO 810 K=1,I1
  810  C(J)=C(J)+S2(J,K)*V(K)/DEL
C THE C(J) ARE THE GENL COORD FOR THE 8-NODE ELEMENT, C(1) IS THE
C LEADING COEF IN THE WILLIAMS SERIES
C CALCULATE THE STRESS INTENSITY FACTOR, K ONE
       SIF=12.*SQRT(2.*PI*DEL)*G*C(1)
       WRITE(IOUT1,209) I,SIF
       WRITE(6,209) I,SIF
  209  FORMAT(/6X,"8-NODE ELEMENT ",I4,3X,"K ONE = ",1PE11.3/)
  807  CONTINUE
  806  CONTINUE
C WRITE HEADING PREPARATORY TO 10-NODE CALCULATIONS
       I1=10*NDOF
       IF(NCRK10.LE.0) GO TO 811
       REWIND 26
       WRITE(IOUT1,210)
  210  FORMAT(/5X,"10-NODE ELEMENT STRESS INTENSITY FACTORS",
      *       /5X,"ELEMENT",7X,"K ONE",10X,"K TWO")
       DO 812 I=1,NCRK10
C READ IN  PARAMETERS AND  GENL COORD RECOVERY MATRIX
C GENERATE THE ELEMENT DISPL VECTOR
       DO 813 J=1,10
C FIND NODE NUMBERS FOR THE ELEMENT
       K=JC10(J,I)
       I2=NDOF*(J-1)
C STUFF  DISPL INT THE V VECTOR
```

```
      DO 814 L=1,NDOF
  814 V(I2+L)=U(L,K)
  813 CONTINUE
      READ(26) DEL,G,THK,((S2(J,K),J=1,20),K=1,I1)
C MULTIPLY C=(AINV*T)*V
      DO 815 J=1,20
      C(J)=0.
      DO 815 K=1,I1
  815 C(J)=C(J)+S2(J,K)*V(K)/DEL
C FOR 10-NODE ELEMENT, C(1) AND C(11) ARE THE RELEVANT GENL COORD
      SIF=12.*SQRT(2.*PI*DEL)*G*C(1)
      THK=4.*SQRT(2.*PI*DEL)*G*C(11)
      WRITE(IOUT1,211) I,SIF,THK
  211 FORMAT(/6X,I4,5X,1PE11.3,4X,1PE11.3)
      WRITE(6,212) I,SIF,THK
  212 FORMAT(/5X,"10-NODE ELEMENT ",I4,3X,"K ONE = ",
     *       1PE11.3," K TWO = ",1PE11.3)
  812 CONTINUE
  811 CONTINUE
  800 CONTINUE
      WRITE(6,213)
  213 FORMAT(//5X,"IF YOU WISH TO RUN DYNAM1 AGAIN,TYPE% YES"/)
      READ(5,102) A1
      IF(A1.EQ.1HY) GO TO 300
      RETURN
      END
```

```
DYNAM2
COYNAM2
      SUBROUTINE DYNAM2(NF,JC8,JC10,P,ST,SM,F,U,U2,NP1,NDOF,NROW)
      DIMENSION NF(NDOF,1),JC8(9,1),JC10(11,1),ST(NROW,1),SM(NROW,1),
     *          F(NROW,1),U(NROW,1),U2(NDOF,1),P(1)
      DIMENSION PP(3)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLCNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,N8,NDOFN,ISTATE,NKORE,IDYNAM
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      A1=1HN
      DO 297 I=1,3
      DO 297 J=1,NER
      U(J,I)=0.0
      F(J,I)=0.0
 297  CONTINUE
 97   FORMAT (A1)
 98   FORMAT(//10X,"IF YOU WISH TO CORRECT THIS,TYPE: YES")
      IFLAG=0
 3051 READ(5,103) I,(PP(J),J=1,NDOF)
      IF(I.LE.0) GO TO 305
      DO 306 J=1,NDOF
      N0=NF(J,I)
      IF(N0.GT.0) GO TO 307
      IF(PP(J).LE.1.E-06) GO TO 306
      IFLAG=IFLAG+1
      WRITE(6,109) I,N0
 109  FORMAT(/10X,"****NODE ",I5,"  FREEDOM NUMBER ",I5,
     *       /10X,"HAS A SPECIFIED INITIAL DISPLACEMENT"/)
      GO TO 306
C  UZERO IS IN U(J,1)
 307  U(N0,1)=PP(J)
 306  CONTINUE
      GO TO 3051
 305  CONTINUE
 300  READ(5,103) I,(PP(J),J=1,NDOF)
 103  FORMAT(I5,3E15.8)
      IF(I.LE.0) GO TO 301
      DO 302 J=1,NDOF
      N0=NF(J,I)
      IF(N0.GT.0) GO TO 303
      IF(PP(J).LE.1.E-06) GO TO 302
      IFLAG=IFLAG+1
      WRITE(6,104) I,N0
 104  FORMAT(//5X,"****NODE ",I5,"  FREEDOM NUMBER  ",I5,
     *       /5X,"HAS A SPECIFIED INITIAL VELOCITY****"//)
      GO TO 302
C  VZERO IS NOW IN U(I,3)
 303  U(N0,3)=PP(J)
 302  CONTINUE
      GO TO 300
 301  CONTINUE
C  SAVE ST,SM,UZERO,VZERO FOR RERUN WITH DIFFERENT TIME STEP
C  ACTIVATE WHEN  NEEDED% WILL NEED TO REINITIALIZE U0 AND V0
C     WRITE(27) ST,SM,(U(J,1),J=1,NER),(U(J,3),J=1,NER)
      IF(INPRT.LT.3) GO TO 108
      WRITE(6,106)
 106  FORMAT(//10X,"FREEDOM NUMBER",7X,"UZERO",7X,"VZERO"/)
      WRITE(6,107) ((J,U(J,1),U(J,3)),J=1,NER)
 107  FORMAT(14X,I5,10X,1PE10.3,2X,1PE10.3)
 108  IF(IFLAG.EQ.0) GO TO 298
      WRITE(6,1111) IFLAG
 1111 FORMAT(//10X,"****THERE WERE ",I5,"  ERRORS IN INITIAL CONDITIONS"
```

```
     *       ,/10X,"PROGRAM TERMINATES****"//)
      STOP
  298 CONTINUE
C  INPUT INTEGRATION INTERVAL AND STEP LENGTH
      WRITE (6,101)
  101 FORMAT(//10X,"ENTER:  DELTA T, NUMBER OF TIME STEPS")
      READ (5,*) DT,NTS
      WRITE (6,99) DT,NTS
   99 FORMAT(//10X,"THE PARAMETERS ARE: DT = ",E15.8,
     *          "  NTS = ",I5)
      WRITE (6,98)
      READ (5,97) A1
      IF(A1.EQ.1HY) GO TO 298
C  SET UP STRESS INTENSITY FACTOR CALC FREQUENCY
      IF((NCRK8.NE.0).OR.(NCRK10.NE.0)) GO TO 312
      NTSIF=0
      GO TO 356
  312 WRITE(6,112)
      A1=1HN
  112 FORMAT(//10X,"HOW OFTEN MUST I CALCULATE S.I. FACTORS?")
      READ(5,*) NTSIF
      WRITE(6,113) NTSIF
  113 FORMAT(//10X,"S.I. FACTOR WILL BE CALCULATED EVERY ",I5,"  TIME ST
     *EPS")
      WRITE(6,98)
      READ(5,97) A1
      IF(A1.EQ.1HY) GO TO 312
C  SET UP ACCELERATION CALC FREQUENCY
  356 IF(NLONDS.NE.0) GO TO 350
      NTACC=0
      GO TO 340
  350 WRITE(6,120)
      A1=1HN
  120 FORMAT(//10X,"HOW OFTEN MUST I CALCULATE ACCELERATIONS?")
      READ(5,*) NTACC
      WRITE(6,121) NTACC
  121 FORMAT(//10X,"ACCELERATIONS WILL BE CALCULATED EVERY ",I5,
     *"  TIME STEPS")
      WRITE(6,98)
      READ(5,97) A1
      IF(A1.EQ.1HY) GO TO 350
      IT5=0
      N0=NDOF*NNODES
      DO 341 I=1,N0
  341 IF(IFIX(P(I)).EQ.1) IT5=IT5+1
C  COUNTS THE NUMBER OF ACCELERATIONS TO BE CALCULATED
      IT3=NTS/NTACC
      IF(MOD(NTS,NTACC).EQ.0) IT3=IT3-1
C  WRITE ON 29 1+IT5 TIMES EACH ACCEL CALL
  340 CONTINUE
      HSQOV4=(DT*DT)/4.
C  BEGIN FIRST STEP CALCULATION: F(I,3)=DT*M*VZERO
      CALL PBMULT(NER,NB,SM,U(1,3),F(1,3),DT)
C  FORM D=M + HSQOV4*K, B=2*(M - HSQOV4*K)
      DO 309 I=1,NER
      DO 309 J=1,NB
      ZM=SM(I,J)
      ZT=ST(I,J)*HSQOV4
C  D MATRIX FORMED IN SM
      SM(I,J)=ZM+ZT
C  B MATRIX FORMED IN ST
  309 ST(I,J)=2.*(ZM-ZT)
C  CONTINUE CALC FOR RHS FIRST STEP: F(I,1)=0.5*B*UZERO
      CALL PBMULT(NER,NB,ST,U(1,1),F(1,1),0.5)
```

```
      DO 310 I=1,NER
      U(I,2)=F(I,3)+F(I,1)
C  ZERO APPROPRIATELY
      F(I,3)=0.0
      F(I,1)=0.0
 310  U(I,3)=0.0
C  NOW CALC FZERO, FONE
      T=0.
C
C  FVEC *MUST* RETURN NODAL FORCES AT TIME T IN FREEDOM NUMBER ORDER
C
      CALL FVEC(NF,T,F(1,1))
      T=T+DT
      CALL FVEC(NF,T,F(1,2))
C  COMPLETE CALC OF RHS IN FIRST STEP
      DO 311 I=1,NER
 311  U(I,2)=U(I,2)+HSQOV4*(F(I,2)+F(I,1))
C  TRIANGULARIZE D
      CALL BANSOL(SM,U(1,2),NER,NB,NER)
C  SOLVE D*UONE=RHS
      CALL SOLVE(SM,U(1,2),NER,NB,NER)
C  UONE IS IN U(I,2)
C  VECTOR UTILIZATION: UZERO IN U(I,1), UONE IN U(I,2)
C                      FZERO IN F(I,1), FONE IN F(I,2)
C
C  INITIALIZE INDICES FOR GENERAL STEP OF INTEGRATION
C
      IT1=NTSIF
      IT=1
      IF(IT.NE.IT1)GO TO 326
      WRITE(28)IT,T
      CALL SIFAC(NF,JC8,JC10,U(1,2),U2,NDOF)
 326  CONTINUE
      IT4=NTACC+1
      N0=0
      N1=3
      N2=2
      N3=1
C  THIS IS THE GENERAL INTEGRATION STEP LOOP
      DO 313 IT=2,NTS
C  BEGIN RHS CALCULATION U(I,N1)= B*U(I,N2)
      CALL BMULT(NER,NB,ST,U(1,N2),U(1,N1))
      T=T+DT
      CALL FVEC(NF,T,F(1,N1))
      DO 314 J=1,NER
 314  U(J,N1)=U(J,N1)+HSQOV4*(F(J,N1)+2.*F(J,N2)+ F(J,N3))
C  USTAR= USUB(N+2) + USUB(N)
C  CALCULATE D*USTAR= RHS
      CALL SOLVE(SM,U(1,N1),NER,NB,NER)
      DO 315 I=1,NER
 315  U(I,N1)=U(I,N1)-U(I,N3)
C  AT THIS POINT,UN+2 IS ALWAYS IN U(I,N1)
C  U(I,N3) IS NOW AVAILABLE FOR WORKING SPACE
C  CALCULATE THE S.I. FACTORS AT SPECIFIED STEPS
      IF(NTSIF.EQ.0) GO TO 316
      IF(IT.LT.IT1) GO TO 316
      IT1=IT1+NTSIF
      WRITE(28) IT,T
      CALL SIFAC(NF,JC8,JC10,U(1,N1),U2,NDOF)
 316  IF(NTACC.EQ.0) GO TO 360
      IF(IT.LT.IT4) GO TO 360
      IT4=IT4+NTACC
      TM1=T-DT
      I=IT-1
```

```
      WRITE(29) I,TM1
      CALL ACCEL(P,U(1,N1),U(1,N2),U(1,N3),DT)
 360  CONTINUE
C  CYCLE INDICES FOR NEXT STEP
      N0=N3
      N3=N2
      N2=N1
      N1=N0
C  INITIALIZE U(I,N1) AND F(I,N1)
      DO 317 I=1,NER
      U(I,N1)=0.
 317  F(I,N1)=0.
C  END OF INTEGRATION LOOP
 313  CONTINUE
C  RECOVER STRESS INTENSITY FACTORS AND WRITE ON IOUT1, FORMATTED
      IF(NTSIF.EQ.0) GO TO 325
      REWIND 28
 114  FORMAT(/10X,"TIME STEP ",I5,5X,"T = ",1PE10.3)
 115  FORMAT(13X,"8-NODE ELT  ",I3," K-ONE = ",1PE10.3)
 116  FORMAT(13X,"10-NODE ELT ",I3," K-ONE = ",1PE10.3," K-TWO = ",1PE10
     *.3)
      IT2=NTS/NTSIF
      DO 319 I=1,IT2
      READ(28) IT1,T
      IF(IOPRT.GT.1) WRITE(6,114) IT1,T
      WRITE(IOUT1,114) IT1,T
      IF(NCRK8.EQ.0) GO TO 320
      DO 321 J=1,NCRK8
      READ(28) IT1,T
      IF(IOPRT.GT.1) WRITE(6,115) IT1,T
 321  WRITE(IOUT1,115) IT1,T
 320  IF(NCRK10.EQ.0) GO TO 319
      DO 323 J=1,NCRK10
      READ(28) IT1,T,DT
      IF(IOPRT.GT.1) WRITE(6,116) IT1,T,DT
 323  WRITE(IOUT1,116) IT1,T,DT
 319  CONTINUE
C  RECOVER ACCELERATIONS AND WRITE ON IOUT1, FORMATTED
 325  IF(NTACC.EQ.0) GO TO 370
      REWIND 29
      CALL ACCOUT(NF,P,F,IT3,IT5,NDOF)
 370  CONTINUE
      RETURN
      END
```

```
DYNAM3
COYNAM3
      SUBROUTINE DYNAM3 (NF,JC8,JC10,P,ST,SM,DYN,P2,U,NDOFN,NROW)
      DIMENSION NF(NDOFN,1),P(1),ST(NROW,1),SM(1),DYN(NROW,1),
     *          P2(1),U(NDOFN,1),JC8(9,1),JC10(11,1),
     *          V(30),S2(20,30),IN(3),C(20)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,N8,NDOF,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DATA PI/3.1415926/
  300 WRITE(6,200)
      A1=1HN
  200 FORMAT(/5X,"ENTER% INITIAL FREQ, FINAL FREQ, NR OF STEPS",
     *      //5X,"FREQ IN HZ, FREE FORMAT, ORDER INDICATED")
      READ(5,*) WI,WF,NSTEP
      WRITE(6,201) WI, WF, NSTEP
  201 FORMAT(/5X,"THE PARAMETERS ARE%",/6X,"WI",9X,"WF",4X,"NSTEP",
     *      //1X,1P2E11.3,I5/)
      REWIND 26
      WRITE(6,202)
  202 FORMAT(/5X,"IF YOU WISH TO CORRECT THIS, TYPE% YES")
      READ(5,102) A1
  102 FORMAT(A1)
      IF(A1.EQ.1HY) GO TO 300
      WI=WI*PI*2.
      WF=WF*PI*2.
      DELW=WF-WI
      IF(NSTEP.LE.1) Z1=1.
      IF(NSTEP.LE.1) GO TO 799
      Z1=1./FLOAT(NSTEP-1)
  799 CONTINUE
      DO 800 KT=1,NSTEP
C CALCULATE OMEGA SQUARED
      W=WI+(KT-1)*DELW*Z1
      W2=W*W
C  INITIALIZE P2 AND DYN
      DO801I=1,NER
      P2(I)=P(I)
      DO801J=1,N8
      IF(I.EQ.J) GO TO 801
      DYN(I,J)=ST(I,J)
      GO TO 8011
  801 DYN(I,J)=ST(I,J)-W2*SM(I)
C CALCULATE NODAL DISPLACEMENTS, RETURNED IN P2
 8011 CALL BANS2L(DYN,P2,NER,NB,NER)
C RECOVER NODAL DISPLACEMENTS, IN NODAL POINT ORDER
      DO 802 I=1,NNODES
C ZERO THE NODAL DISPLACEMENTS
      DO 803 J=1,NDOF
  803 U(J,I)=0.
C RECOVER THE FREEDOM NUMBERS OF THE NODAL DISPLACEMENTS
      DO 804 J=1,NDOF
      CALL INTEGR(IN(J),NF(J,I))
  804 IN(J)=IABS(IN(J))
C IN(J) CONTAINS FREEDOM NUMBERS OF NODAL DISPL, OR ZERO
C RECOVER THE NODAL DISPL FROM P2
      DO 805 J=1,NDOF
      K=IN(J)
      IF(K.EQ.0) GO TO 805
      U(J,I)=P2(K)
  805 CONTINUE
  802 CONTINUE
```

```
C WRITE OUT THE DISPLACEMENT VECTOR
      W=W/(2.*PI)
      WRITE(IOUT1,203) W
      WRITE(6,203) W
  203 FORMAT(/5X,"THE DATA FOR OMEGA= ",1PE11.3," FOLLOWS%")
      WRITE(IOUT1,204)
      IF(IOPRT.GT.0) WRITE(6,204)
  204 FORMAT(/5X,"NODAL POINT DISPLACEMENTS",
     *       /5X,"NODE",7X,"U",10X,"V",10X,"W"/)
      IF(NDOF.EQ.2) GO TO 8051
      DO 8052 I=1,NNODES
      IF(IOPRT.GT.0) WRITE(6,205) I,(U(J,I),J=1,3)
 8052 WRITE(IOUT1,205) I,(U(J,I),J=1,3)
  205 FORMAT(/3X,I5,2X,1P3E11.3)
      GO TO 8053
 8051 DO 8054 I=1,NNODES
      IF(IOPRT.GT.0) WRITE(6,206) I,(U(J,I),J=1,2)
 8054 WRITE(IOUT1,206) I,(U(J,I),J=1,2)
  206 FORMAT(/3X,I5,2X,1P2E11.3)
 8053 I1=8*NDOF
      IF(NCRK8.LE.0) GO TO 806
      REWIND 25
      WRITE(IOUT1,207)
  207 FORMAT(/6X,"8-NODE ELEMENT STRESS INTENSITY FACTOR",
     *       /6X,"ELEMENT",7X,"K ONE")
C NODAL POINT DISP NOW IN U IN NODAL POINT ORDER
C RECOVER GEN. DISP. AND CALC STRESS INTENSITY FACTORS
      DO 807 I=1,NCRK8
C READ IN THE PARAMETERS AND GEN COORD RECOVERY MATRIX
C GENERATE THE ELEMENT DISPLACEMENT VECTOR
      DO 808 J=1,8
C FIND THE NODE NUMBERS FOR  THE ELEMENT
      K=JC8(J,I)
      I2=NDOF*(J-1)
C STUFF DISPLACEMENTS INTO THE V VECTOR
      DO 809 L=1,NDOF

  809 V(I2+L)=U(L,K)
  808 CONTINUE
      READ(25) DEL,G,THK,((S2(J,K),J=1,16),K=1,I1)
C  MULTIPLY C=AINV*T*V
      DO 810 J=1,16
      C(J)=0.
      DO 810 K=1,I1
  810 C(J)=C(J)+S2(J,K)*V(K)/DEL
C THE C(J) ARE THE GENL COORD FOR THE 8-NODE ELEMENT, C(1) IS THE
C LEADING COEF IN THE WILLIAMS SERIES
C CALCULATE THE STRESS INTENSITY FACTOR, K ONE
      SIF=12.*SQRT(2.*PI*DEL)*G*C(1)
      WRITE(IOUT1,209) I,SIF
      WRITE(6,209) I,SIF
  209 FORMAT(/6X,"8-NODE ELEMENT ",I4,3X,"K ONE = ",1PE11.3/)
  807 CONTINUE
  806 CONTINUE
C WRITE HEADING PREPARATORY TO 10-NODE CALCULATIONS
      I1=10*NDOF
      IF(NCRK10.LE.0) GO TO 811
      WRITE(IOUT1,210)
  210 FORMAT(/5X,"10-NODE ELEMENT STRESS INTENSITY FACTORS",
     *       /5X,"ELEMENT",7X,"K ONE",10X,"K TWO")
      DO 812 I=1,NCRK10
C READ IN  PARAMETERS AND  GENL COORD RECOVERY MATRIX
C GENERATE THE ELEMENT DISPL VECTOR
      DO 813 J=1,10
```

```
C FIND NODE NUMBERS FOR THE ELEMENT
      K=JC10(J,I)
      I2=NDOF*(J-1)
C STUFF DISPL INT THE V VECTOR
      DO 814 L=1,NDOF
 814  V(I2+L)=U(L,K)
 813  CONTINUE
      READ(26) DEL,G,THK,((S2(J,K),J=1,20),K=1,I1)
C MULTIPLY C=(AINV*T)*V
      DO 815 J=1,20
      C(J)=0.
      DO 815 K=1,I1
 815  C(J)=C(J)+S2(J,K)*V(K)/DEL
C FOR 10-NODE ELEMENT, C(1) AND C(11) ARE THE RELEVANT GENL COORD
      SIF=12.*SQRT(2.*PI*DEL)*G*C(1)
      THK=4.*SQRT(2.*PI*DEL)*G*C(11)
      WRITE(IOUT1,211) I,SIF,THK
 211  FORMAT(/6X,I4,5X,1PE11.3,4X,1PE11.3)
      WRITE(6,212) I,SIF,THK
 212  FORMAT(/5X,"10-NODE ELEMENT ",I4,3X,"K ONE = ",
     *         1PE11.3," K TWO = ",1PE11.3)
 812  CONTINUE
 811  CONTINUE
 800  CONTINUE
      WRITE(6,213)
 213  FORMAT(//5X,"IF YOU WISH TO RUN DYNAM1 AGAIN,TYPE% YES"/)
      READ(5,102) A1
      IF(A1.EQ.1HY) GO TO 300
      RETURN
      END
```

```
DYNAM4
CDYNAM4
      SUBROUTINE DYNAM4(NF,JC8,JC10,P,ST,SM,F,U,U2,NP1,NDOFN,NROW)
      DIMENSION NF(NDOFN,1),JC8(9,1),JC10(11,1),ST(NROW,1),SM(1),
     *          F(1),U(NROW,1),U2(NDOFN,1),P(1),PP(3)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE,IDYNAM
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      WRITE(6,9999)
 9999 FORMAT(1X,"ENTERED DYNAM4")
      A1=1HN
C  STORES ST,SM,UZERO,VZERO FOR FUTURE USE% ACTIVATE WHEN NEEDED
C     NO=2*NER*(NB+1)
C     DEFINE FILE 27(1,NO,V,IP)
C
C  INITIALIZE F AND U
C
      DO 297 J=1,NER
      F(J)=0.0
      DO 297 I=1,3
      U(J,I)=0.0
  297 CONTINUE
   97 FORMAT (A1)
   98 FORMAT(//10X,"IF YOU WISH TO CORRECT THIS,TYPE% YES")
      IFLAG=0
 3051 READ(5,103) I,(PP(J),J=1,NDOF)
      IF(I.LE.0) GO TO 305
      DO 306 J=1,NDOF
      NO=NF(J,I)
      IF(NO.GT.0) GO TO 307
      IF(PP(J).LE.1.E-06) GO TO 306
      IFLAG=IFLAG+1
      WRITE(6,109) I,NO
  109 FORMAT(/10X,"****NODE ",I5,"  FREEDOM NUMBER ",I5,
     *       /10X,"HAS A SPECIFIED INITIAL DISPLACEMENT"/)
      GO TO 306
C  UZERO IS IN U(J,1)
  307 U(NO,1)=PP(J)
  306 CONTINUE
      GO TO 3051
  305 CONTINUE
  300 READ(5,103) I,(PP(J),J=1,NDOF)
  103 FORMAT(I5,3E15.8)
      IF(I.LE.0) GO TO 301
      DO 302 J=1,NDOF
      NO=NF(J,I)
      IF(NO.GT.0) GO TO 303
      IF(PP(J).LE.1.E-06) GO TO 302
      IFLAG=IFLAG+1
      WRITE(6,104) I,NO
  104 FORMAT(//5X,"****NODE ",I5,"  FREEDOM NUMBER ",I5,
     *       /5X,"HAS A SPECIFIED INITIAL VELOCITY****"//)
      GO TO 302
C  VZERO IS NOW IN U(I,3)
  303 U(NO,3)=PP(J)
  302 CONTINUE
      GO TO 300
  301 CONTINUE
C  SAVE ST,SM,UZERO,VZERO FOR RERUN WITH DIFFERENT TIME STEP
C  ACTIVATE WHEN  NEEDED% WILL NEED TO REINITIALIZE U0 AND V0
C     WRITE(27'IP) ST,SM,(U(J,1),J=1,NER),(U(J,3),J=1,NER)
      IF(INPRT.LT.3) GO TO 108
```

```
      WRITE(6,106)
  106 FORMAT(//10X,"FREEDOM NUMBER",7X,"UZERO",7X,"VZERO"/)
      WRITE(6,107) ((J,U(J,1),U(J,3)),J=1,NER)
  107 FORMAT(14X,I5,10X,1PE10.3,2X,1PE10.3)
  108 IF(IFLAG.EQ.0) GO TO 298
      WRITE(6,1111) IFLAG
 1111 FORMAT(//10X,"****THERE WERE ",I5," ERRORS IN INITIAL CONDITIONS"
     *       ,/10X,"PROGRAM TERMINATES****"//)
      STOP
  298 CONTINUE
C  INPUT INTEGRATION INTERVAL AND STEP LENGTH
      WRITE (6,101)
  101 FORMAT(//10X,"ENTER%  DELTA T, NUMBER OF TIME STEPS")
      READ (5,*) DT,NTS
      WRITE (6,99) DT,NTS
   99 FORMAT(//10X,"THE PARAMETERS ARE% DT = ",E15.8,
     *          "  NTS = ",I5)
      WRITE (6,98)
      READ (5,97) A1
      IF(A1.EQ.1HY) GO TO 298
C  SET UP STRESS INTENSITY FACTOR CALC FREQUENCY
      IF((NCRK8.NE.0).OR.(NCRK10.NE.0)) GO TO 312
      NTSIF=0
      GO TO 340
  312 WRITE(6,112)
      A1=1HN
  112 FORMAT(//10X,"HOW OFTEN MUST I CALCULATE S.I. FACTORS#")
      READ(5,*) NTSIF
      WRITE(6,113) NTSIF
  113 FORMAT(//10X,"S.I. FACTOR WILL BE CALCULATED EVERY ",I5,"  TIME ST
     *EPS")
      WRITE(6,98)
      READ(5,97) A1
      IF(A1.EQ.1HY) GO TO 312
C  SET UP FILE FOR SIFAC
C
C  START OF CENTRAL DIFFERENCE TIME INTEGRATION FOR
C  SINGLE 2ND ORDER ORDINARY DIFFERENTIAL EQUATION
  340 HSQOV2=DT**2/2.
C  BEGIN FIRST STEP CALCULATION%  U(I,2)=ST*UZERO
      CALL BMULT(NER,NB,ST,U(1,1),U(1,2))
C  NOW CALCULATE FZERO
      T=0.0
C  FVEC *MUST* RETURN NODAL FORCES AT TIME T IN FREE-
C  DOM NUMBER ORDER
      CALL FVEC(NF,T,F)
C  CALCULATE UONE
      DO309I=1,NER
C  CALCULATE INVERSE OF M
      SM(I)=1./SM(I)
C  U(I,2)=FZERO-K*UZERO
      U(I,2)=F(I)-U(I,2)
C  U(I,2)=(DT**2)*MINV*(FZERO-K*UZERO)
      U(I,2)=HSQOV2*SM(I)*U(I,2)
C  UONE=UZERO+DT*VZERO+(DT**2)/2.*MINV*(FZERO-K*UZERO)
      U(I,2)=U(I,1)+DT*U(I,3)+U(I,2)
  309 U(I,3)=0.0
      WRITE(10,1001)
 1001 FORMAT(10X,"UZERO",10X,"U ONE")
      WRITE(10,1000) (U(I,1),U(I,2),I=1,NER)
 1000 FORMAT(5X,2E15.8)
C  UONE IS IN U(I,2)
C  VECTOR UTILIZATION%  UZERO IN U(I,1),
C                       UONE IN U(I,2)
```

```
C
C
C   CALCULATE STRESS INTENSITY FACTORS FOR FIRST TIME STEP
C
      IF(NTSIF.LE.0) GO TO 310
      IT=1
      WRITE(28) IT,T
      CALL SIFAC(NF,JC8,JC10,U(1,2),U2,NDOFN)
  310 CONTINUE
C
C  INITIALIZE INDICES FOR GENERAL STEP OF INTEGRATION
C
      IT1=NTSIF
      N0=0
      N1=3
      N2=2
      N3=1
C  THIS IS THE GENERAL INTEGRATION STEP LOOP
      HSQ=DT**2
      DO313IT=2,NTS
C  BEGIN RHS CALCULATION%  U(J,N1)=K*U(N)
      CALL BMULT(NER,NB,ST,U(1,N2),U(1,N1))
      T=T+DT
      CALL FVEC(NF,T,F)
      DO314J=1,NER
C  U(J,N1)=F(N)-K*U(N)
      U(J,N1)=F(J)-U(J,N1)
C  U(J,N1)=(DT**2)*MINV*(F(N)-K*U(N))
      U(J,N1)=HSQ*SM(J)*U(J,N1)
C  U(N+1)=2.*U(N)-U(N-1)+(DT**2)*MINV*(F(N)-K*U(N))
  314 U(J,N1)=2.*U(J,N2)-U(J,N3)+U(J,N1)
C  AT THIS POINT, U(N+1) IS ALWAYS IN U(J,N1)
C  U(J,N3) IS NOW AVAILABLE FOR WORKING SPACE
C  CALCULATE THE S.I. FACTORS AT SPECIFIED STEPS
      IF(NTSIF.EQ.0) GO TO 316
      IF(IT.LT.IT1) GO TO 316
      IT1=IT1+NTSIF
      WRITE(10,1002) IT
 1002 FORMAT(5X,//,"DISPLACEMENTS FOR NT = ",I5/)
      WRITE(10,1003) (I,U(I,N1),I=1,NER)
 1003 FORMAT(1X,5(" U(",I3,")=",E15.8))
      WRITE(28) IT,T
      CALL SIFAC(NF,JC8,JC10,U(1,N1),U2,NDOFN)
C  CYCLE INDICES FOR NEXT STEP
  316 N0=N3
      N3=N2
      N2=N1
      N1=N0
C  INITIALIZE U(I,N1) AND F(I)
      DO 317 I=1,NER
      U(I,N1)=0.
  317 F(I)=0.0
C  END OF INTEGRATION LOOP
  313 CONTINUE
C  RECOVER STRESS INTENSITY FACTORS AND WRITE ON IOUT1, FORMATTED
      IF(NTSIF.EQ.0) GO TO 370
      REWIND 28
  114 FORMAT(/10X,"TIME STEP ",I5,5X,"T = ",1PE10.3)
  115 FORMAT(13X,"8-NODE ELT  ",I3," K-ONE = ",1PE10.3)
  116 FORMAT(13X,"10-NODE ELT ",I3," K-ONE = ",1PE10.3," K-TWO = ",1PE10
     *.3)
      IT2=NTS/NTSIF
      DO 319 I=1,IT2
      READ(28) IT1,T
```

```
      IF(IOPRT.GT.1) WRITE(6,114) IT1,T
      WRITE(IOUT1,114) IT1,T
      IF(NCRK8.EQ.0) GO TO 320
      DO 321 J=1,NCRK8
      READ(28) IT1,T
      IF(IOPRT.GT.1) WRITE(6,115) IT1,T
  321 WRITE(IOUT1,115) IT1,T
  320 IF(NCRK10.EQ.0) GO TO 319
      DO 323 J=1,NCRK10
      READ(28) IT1,T,DT
      IF(IOPRT.GT.1) WRITE(6,116) IT1,T,DT
  323 WRITE(IOUT1,116) IT1,T,DT
  319 CONTINUE
  370 CONTINUE
      RETURN
      END
```

```
BANSOL
CBANSOL
      SUBROUTINE  BANSOL(AK,R,NEQ,IBAND,NDIM)
C  SYMMETRIC BAND MATRIX EQUATION SOLVER
C
C  ENTRY BANSOL TRIANGULARIZES  THE BAND MATRIX AK
C  ENTRY SOLVE SOLVES FOR RIGHT HAND SIDE R, SOLUTION RETURNS IN R
C
      DOUBLE PRECISION PIVOT,CP
      DIMENSION AK(NDIM,1),R(1)
      IF(IBAND.EQ.1) GO TO 121
      NRS = NEQ - 1
      NR = NEQ
C  TRIANGULARIZE MATRIX
      DO 120 N = 1,NRS
      M = N - 1
      MR = MINO(IBAND,NR-M)
      PIVOT = AK(N,1)
      DO 120 L = 2,MR
      CP = AK(N,L)/PIVOT
      I = M + L
      J = 0
      DO 110 K = L,MR
      J = J + 1
  110 AK(I,J) = SNGL(DBLE(AK(I,J))-CP*AK(N,K))
  120 AK(N,L) = CP
  121 RETURN
      ENTRY SOLVE
C  REDUCE VECTOR
      IF(IBAND.EQ.1) GO TO 321
      DO 220 N = 1,NRS
      M = N - 1
      MR = MINO(IBAND,NR-M)
      CP = R(N)
      R(N) = CP/AK(N,1)
      DO 220 L = 2,MR
      I = M + L
  220 R(I) = SNGL(DBLE(R(I))-AK(N,L)*CP)
      R(NR) = SNGL(DBLE(R(NR))/DBLE(AK(NR,1)))
C  BACK SUBSTITUTION
      DO 320 I = 1,NRS
      N = NR - I
      M = N - 1
      MR = MINO(IBAND,NR-M)
      DO 320 K = 2,MR
      L = M + K
  320 R(N) = SNGL(DBLE(R(N))-DBLE(AK(N,K))*DBLE(R(L)))
      GO TO 322
  321 CP=DBLE(R(1))/DBLE(AK(1,1))
      R(1)=SNGL(CP)
  322 RETURN
      END
```

```
BMULT
CBMULT
      SUBROUTINE BMULT (N,IBW,A,X,B)
C
C  THIS SUBPROGRAM MULTIPLIES A BANDED SYMMETRIC MATRIX WITH A
C  VECTOR.  ONLY THE UPPER-BAND PORTION OF THE MATRIX IS STORED.
C
C  GLOBAL PARAMETERS
C       N   = ORDER OF BAND MATRIX
C       IBW = UPPER BAND WIDTH OF MATRIX
C       A   = UPPER BAND OF MATRIX
C       X   = VECTOR TO BE MULTIPLIED
C       B   = PRODUCT VECTOR
C
      REAL A(N,1),X(1),B(1)
      DOUBLE PRECISION SUM,XD
      N1 = N- IBW +1
      DO 300 I= 1,N
      SUM= 0.0
      J1= IBW
      IF (I.GT.N1) J1= N- I+ 1
      DO 100 J= 1,J1
      XD= X(I+J-1)
100   SUM= SUM+ A(I,J)*XD
      IF ((I.EQ.1).OR.(IBW.EQ.1)) GO TO 300
      J1=1
      J2=I-1
      IF (I.GT.IBW) J1=I-IBW+1
      DO 200 J= J1,J2
      XD= X(J)
200   SUM= SUM+ A(J,I-J+1)*XD
300   B(I)= SUM
      RETURN
      END
```

```
PBMULT
CPBMULT
      SUBROUTINE PBMULT (N,IBW,A,X,B,P)
C
C  THIS SUBPROGRAM MULTIPLIES A BANDED SYMMETRIC MATRIX WITH A
C  VECTOR.  ONLY THE UPPER-BAND PORTION OF THE MATRIX IS STORED.
C  P IS A SCALE FACTOR%  B= P*A*X
C
C  GLOBAL PARAMETERS
C       N   = ORDER OF BAND MATRIX
C       IBW = UPPER BAND WIDTH OF MATRIX
C       A   = UPPER BAND OF MATRIX
C       X   = VECTOR TO BE MULTIPLIED
C       B   = PRODUCT VECTOR
C       P   = SCALE FACTOR
C
      REAL A(N,1),X(1),B(1)
      DOUBLE PRECISION SUM,XD
      N1 = N- IBW +1
      DO 300 I= 1,N
      SUM= 0.0
      J1= IBW
      IF (I.GT.N1) J1= N- I+ 1
      DO 100 J= 1,J1
      XD= X(I+J-1)
100   SUM= SUM+ A(I,J)*XD
      IF ((I.EQ.1).OR.(IBW.EQ.1)) GO TO 300
      J1=1
      J2=I-1
      IF (I.GT.IBW) J1=I-IBW+1
      DO 200 J= J1,J2
      XD= X(J)
200   SUM= SUM+ A(J,I-J+1)*XD
300   B(I)= SUM*P
      RETURN
      END
```

```
SIFAC
CSIFAC
      SUBROUTINE SIFAC(NF,JC8,JC10,P2,U,NDOF)
      DIMENSION NF(NDOF,1),JC8(9,1),JC10(11,1),P2(1),U(NDOF,1),
     *          C(20),V(30),S2(20,30),IN(3)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOFN,ISTATE,NKORE,IDYNAM
      COMMON/BLK3/IIN8,IIN10,ICUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DATA PI/3.1415926/
C RECOVER NODAL DISPLACEMENTS, IN NODAL POINT ORDER
      DO 802 I=1,NNODES
C ZERO THE NODAL DISPLACEMENTS
      DO 803 J=1,NDOF
 803  U(J,I)=0.
C RECOVER THE FREEDOM NUMBERS OF THE NODAL DISPLACEMENTS
      DO 804 J=1,NDOF
      CALL INTEGR(IN(J),NF(J,I))
 804  IN(J)=IABS(IN(J))
C IN(J) CONTAINS FREEDOM NUMBERS OF NODAL DISPL, OR ZERO
C RECOVER THE NODAL DISPL FROM P2
      DO 805 J=1,NDOF
      K=IN(J)
      IF(K.EQ.0) GO TO 805
      U(J,I)=P2(K)
 805  CONTINUE
 802  CONTINUE
 8053 I1=8*NDOF
      IF(NCRK8.LE.0) GO TO 806
      REWIND 25
C NODAL POINT DISP NOW IN U IN NODAL POINT ORDER
C RECOVER GEN. DISP. AND CALC STRESS INTENSITY FACTORS
      DO 807 I=1,NCRK8
C READ IN THE PARAMETERS AND GEN COORD RECOVERY MATRIX
      READ(25) DEL,G,THK,((S2(J,K),J=1,16),K=1,I1)
C GENERATE THE ELEMENT DISPLACEMENT VECTOR
      DO 808 J=1,8
C FIND THE NODE NUMBERS FOR  THE ELEMENT
      K=JC8(J,I)
      I2=NDOF*(J-1)
C STUFF DISPLACEMENTS INTO THE V VECTOR
      DO 809 L=1,NDOF
 809  V(I2+L)=U(L,K)
 808  CONTINUE
C  MULTIPLY C=AINV*T*V
      DO 810 J=1,16
      C(J)=0.
      DO 810 K=1,I1
 810  C(J)=C(J)+S2(J,K)*V(K)/DEL
C THE C(J) ARE THE GENL COORD FOR THE 8-NODE ELEMENT, C(1) IS THE
C LEADING COEF IN THE WILLIAMS SERIES
C CALCULATE THE STRESS INTENSITY FACTOR, K ONE
      SIF=12.*SQRT(2.*PI*DEL)*G*C(1)
      WRITE(28) I,SIF
 807  CONTINUE
 806  CONTINUE
      I1=10*NDOF
      IF(NCRK10.LE.0) GO TO 811
      REWIND 26
      DO 812 I=1,NCRK10
C READ IN  PARAMETERS AND  GENL COORD RECOVERY MATRIX
      READ(26) DEL,G,THK,((S2(J,K),J=1,20),K=1,I1)
C GENERATE THE ELEMENT DISPL VECTOR
      DO 813 J=1,10
```

```
C FIND NODE NUMBERS FOR THE ELEMENT
      K=JC10(J,I)
      I2=NDOF*(J-1)
C STUFF DISPL INT THE V VECTOR
      DO 814 L=1,NDOF
 814  V(I2+L)=U(L,K)
 813  CONTINUE
C MULTIPLY C=(AINV*T)*V
      DO 815 J=1,20
      C(J)=0.
      DO 815 K=1,I1
 815  C(J)=C(J)+S2(J,K)*V(K)/DEL
C FOR 10-NODE ELEMENT, C(1) AND C(11) ARE THE RELEVANT GENL COORD
      SIF=12.*SQRT(2.*PI*DEL)*G*C(1)
      THK=4.*SQRT(2.*PI*DEL)*G*C(11)
      WRITE(28) I,SIF,THK
 812  CONTINUE
 811  CONTINUE
      RETURN
      END
```

```
ACCEL
CACCEL
      SUBROUTINE ACCEL(P,U2,U1,U,DT)
      DIMENSION P(1),U2(1),U1(1),U(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NPR,NID,
     *       IFLAG,NB,NDOF,ISTATE,NKORE,IDYNAM
      DOUBLE PRECISION DTSQ,AD
      N0=NDOF*NNODES
      DTSQ=DT*DT
      DO 10 I=1,N0
      IF(IFIX(P(I)).NE.1) GO TO 10
      AD=(U2(I)-2.*U1(I)+U(I))/DTSQ
      ACCLER=SNGL(AD)
      WRITE(29) ACCLER
 10   CONTINUE
      RETURN
      END
```

ACCOUT
CACCOUT

```
      SUBROUTINE ACCOUT(NF,P,KF,IT3,IT5,NDOF)
      DIMENSION NF(NDOF,1),P(1),KF(3,1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,N8,NDOFN,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      IT=0
      DO 9 J=1,IT5
      DO 9 I=1,3
 9    KF(I,J)=0
      DO 10 I=1,NER
      IF(IFIX(P(I))) 10,10,11
 11   IT=IT+1
      KF(1,IT)=I
 10   CONTINUE
      DO 12 I=1,IT
      DO 12 J=1,NNODES
      DO 12 K=1,NDOF
      IF(NF(K,J).NE.KF(1,I)) GO TO 12
      KF(2,I)=J
      KF(3,I)=K
 12   CONTINUE
      DO 13 I=1,IT3
      READ(29) IT,T
      IF(IOPRT.GT.1) WRITE(6,200) IT,T
      WRITE(IOUT1,200) IT,T
      DO 13 J=1,IT5
      READ(29) AX
      IF(KF(3,J).NE.1) GO TO 14
      WRITE(IOUT1,201) KF(2,J),AX
      IF(IOPRT.GT.1) WRITE(6,201) KF(2,J),AX
      GO TO 13
 14   IF(KF(3,J).NE.2) GO TO 15
      WRITE(IOUT1,202) KF(2,J),AX
      IF(IOPRT.GT.1) WRITE(6,202) KF(2,J),AX
      GO TO 13
 15   IF(KF(3,J).NE.3) GO TO 13
      WRITE(IOUT1,203) KF(2,J),AX
      IF(IOPRT.GT.1) WRITE(6,203) KF(2,J),AX
 13   CONTINUE
200   FORMAT(/10X,"TIME STEP ",I5,"  TIME = ",1PE10.3)
201   FORMAT(13X,"NODE ",I5,"  X-ACCEL = ",1PE10.3)
202   FORMAT(13X,"NODE ",I5,"  Y-ACCEL = ",1PE10.3)
203   FORMAT(13X,"NODE ",I5,"  Z-ACCEL = ",1PE10.3)
      RETURN
      END
```

```
OUTPUT
COUTPUT
      SUBROUTINE OUTPUT(A,N)
      DIMENSION A(1),N(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NID,
     *       IFLAG,NB,NDOF
      COMMON/BLK1/NLC,NTCOMP,NCCOMP
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      COMMON/ADR/NNF,NJT,NTP,NJS,NSP,NJ8,N8P,NJ10,N10P,NP1,NST,NSM,
     *           NDY,NP2,NUV,NUN
      IF(IOPRT.GT.0) WRITE(IOUT1,1400) NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,
     *                                 NRR,NID
 1400 FORMAT(10X,"NNODES = ",I5/,
     *       10X,"NTRIS  = ",I5/,
     *       10X,"NSPRNG = ",I5/,
     *       10X,"NCRK8  = ",I5/,
     *       10X,"NCRK10 = ",I5/,
     *       10X,"NRR    = ",I5/,
     *       10X,"NID    = ",I5)
      N12=NDY+NDOF*NNODES
      CALL BANS2L(A(NST),A(NP1),NER,NB,NER)
      IF(IOPRT.GT.2)GO TO 67
      WRITE(6,600)
      GO TO 65
600   FORMAT(1H1/,40X,19HNODAL DISPLACEMENTS//,
     *11X,4HNODE,12X,2HDX,13X,2HDY,13X,2HDZ/)
   67 WRITE(IOUT1,650)
  650 FORMAT(40X,"NODAL DISPLACEMENTS")
   65 DO 66 I=1,NNODES
      LOC=NDOF*(I-1)
      CALL INTEGR(IFR1,A(NNF+LOC))
      CALL INTEGR(IFR2,A(NNF+LOC+1))
      IFR1=IABS(IFR1)
      DX=A(NP1+IFR1-1)
      IF(IFR1.EQ.0) DX=0.0
      IFR2=IABS(IFR2)
      DY=A(NP1+IFR2-1)
      IF(IFR2.EQ.0) DY=0.0
      DZ=0
      GO TO (60,60,50),NDOF
 50   CALL INTEGR(IFR3,A(NNF+LOC+2))
      IFR3=IABS(IFR3)
      IF(IFR3.EQ.0) GO TO 60
      DZ=A(NP1+IFR3-1)
   60 IF(IOPRT.LT.3)GO TO 61
      WRITE(IOUT1,500) I,DX,DY,DZ
      GO TO 66
   61 WRITE(6,500) I,DX,DY,DZ
   66 CONTINUE
C
C    ZERO OUT CKFOR,SX,SY,SXY ARRAYS
C
      LOC=NDOF*NNODES
      DO 70 I=1,LOC
 70   A(NDY+I-1)=0.0
      IF(NTRIS)78,78,76
 76   N13=N12+NTRIS
      N14=N13+NTRIS
      DO 80 I=1,NTRIS
      A(N12+I-1)=0.0
      A(N13+I-1)=0.0
 80   A(N14+I-1)=0.0
```

```
      IF(IOPRT)86,86,85
 85   WRITE(6,1200)
1200  FORMAT(/20X,14HENTERED TRIOUT)
 86   CALL TRIOUT(A,A(NNF),A(NJT),A(NTP),A(NP1),A(NDY),A(N12),A(N13),
     *            A(N14),NDOF)
      IF(IOPRT.GT.0)GO TO 77
      WRITE(6,100)
 100  FORMAT(1H1/,40X,27HTRIANGULAR ELEMENT STRESSES)
      IF(NTCOMP)10,10,20
 10   WRITE(6,200)
 200  FORMAT(45X,18HLOCAL  COORDINATES//)
      GO TO 30
 20   WRITE(6,300)
 300  FORMAT(45X,18HGLOBAL COORDINATES//)
 30   WRITE(6,400)
 400  FORMAT(10X,8HTRIANGLE,9X,7HSIGMA X,8X,7HSIGMA Y,8X,8HSIGMA XY/)
      WRITE(6,500) (I,A(N12+I-1),A(N13+I-1),A(N14+I-1),I=1,NTRIS)
      GO TO 75
   77 WRITE(IOUT1,690)
  690 FORMAT(40X,"TRIANGLE STRESSES")
      WRITE(IOUT1,500) (I,A(N12+I-1),A(N13+I-1),A(N14+I-1),I=1,NTRIS)
 500  FORMAT(1X,I4,3E15.8)
   78 IF(NSPRNG)75,75,79
   79 IF(IOPRT)87,87,88
   88 WRITE(6,1210)
 1210 FORMAT(1H1/,5X,14HENTERED SPROUT)
   87 CALL SPROUT(A,A(NNF),A(NJS),A(NSP),A(NP1),A(NDY),A(N12))
      IF(IOPRT.GT.0)GO TO 89
      WRITE(6,1220)
 1220 FORMAT(1H1/,40X,13HSPRING FORCES//9X,7HELEMENT,11X,5HF(K1),10X,
     *       5HF(K2),10X,5HF(K3),10X,5HR(12)/)
      WRITE(6,1230) (I,A(N12+4*I-4),A(N12+4*I-3),A(N12+4*I-2),
     *               A(N12+4*I-1),I=1,NSPRNG)
 1230 FORMAT(10X,I5,5X,4E15.8)
      GO TO 75
   89 WRITE(IOUT1,1220)
      WRITE(IOUT1,1230) (I,A(N12+4*I-4),A(N12+4*I-3),A(N12+4*I-2),
     *                   A(N12+4*I-1),I=1,NSPRNG)
 75   IF(NCRK8)110,110,120
120   N13=N12+24*24
      N14=N13+16*16
      N15=N14+16*24
      IF(IOPRT)126,126,125
125   WRITE(6,1300)
1300  FORMAT(/20X,14HENTERED CRKOT8)
126   CALL CRKOT8(A(NNF),A(NJ8),A(N8P),A(NP1),A(NDY),A(N12),A(N13),
     *            A(N14),A(N15))
      WRITE(6,1000)
1000  FORMAT(1H1/,10X,47H8-NODE CRACKED ELEMENT STRESS INTENSITY FACTORS
     *       //,10X,7HELEMENT,11X,8HMODE 1 K/)
      WRITE(6,1100) (I,A(N15+I-1),I=1,NCRK8)
1100  FORMAT(11X,I5,5X,1E15.8)
110   IF(NCRK10)140,140,150
150   N13=N12+30*30
      N14=N13+20*20
      N15=N14+20*30
      N16=N15+NCRK10
      IF(IOPRT.GT.0)WRITE(6,1500)
1500  FORMAT(/20X,14HENTERED CRKO10)
      CALL CRKO10(A(NNF),A(NJ10),A(N10P),A(NP1),A(NDY),A(N12),A(N13),
     *            A(N14),A(N15),A(N16))
 140  IF(IOPRT.GT.1) GO TO 145
      WRITE(6,700)
700   FORMAT(1H1/,40X,23HNODE EQUILIBRIUM CHECKS//,
```

```
     *       11X,4HNODE,9X,7HX-FORCE,8X,7HY-FORCE,8X,7HZ-FORCE/)
      IF(NDOF.GT.2) GO TO 90
      WRITE(6,800) (I,A(NDY+2*I-2),A(NDY+2*I-1),I=1,NNODES)
800   FORMAT(10X,I5,5X,2E15.8)
      GO TO 130
 90   WRITE(6,900) (I,A(NDY+3*I-3),A(NDY+3*I-2),A(NDY+3*I-1),I=1,NNODES)
900   FORMAT(10X,I5,5X,3E15.8)
      GO TO 130
 145  WRITE(IOUT1,675)
 675  FORMAT(40X,"NODAL EQUILIBRIUM CHECKS")
      GO TO(146,146,147),NDOF
 146  WRITE(IOUT1,800) (I,A(NDY+2*I-2),A(NDY+2*I-1),I=1,NNODES)
      GO TO 130
 147  WRITE(IOUT1,900)(I,A(NDY+3*I-3),A(NDY+3*I-2),A(NDY+3*I-1),I=1,
     *NNODES)
130   CONTINUE
      RETURN
      END
```

```
BANS2L
CBANS2L
      SUBROUTINE  BANS2L(AK,R,NEQ,IBAND,NDIM)
C  SYMMETRIC BAND MATRIX EQUATION SOLVER
      DIMENSION AK(NDIM,1), R(1)
      DOUBLE PRECISION CP,PIVOT
      IF(IBAND.EQ.1) GO TO 321
      NRS = NEQ - 1
      NR = NEQ
      DO 120 N = 1,NRS
      M = N - 1
      MR = MIN0(IBAND,NR-M)
      PIVOT = AK(N,1)
      DO 120 L = 2,MR
      CP = AK(N,L)/PIVOT
      I = M + L
      J = 0
      DO 110 K = L,MR
      J = J + 1
  110 AK(I,J) = AK(I,J) - CP*AK(N,K)
  120 AK(N,L) = CP
      DO 220 N = 1,NRS
      M = N - 1
      MR = MIN0(IBAND,NR-M)
      CP = R(N)
      R(N) = CP/AK(N,1)
      DO 220 L = 2,MR
      I = M + L
  220 R(I) = R(I) - AK(N,L)*CP
      R(NR) = R(NR)/AK(NR,1)
      DO 320 I = 1,NRS
      N = NR - I
      M = N - 1
      MR = MIN0(IBAND,NR-M)
      DO 320 K = 2,MR
      L = M + K
C  STORE COMPUTED DISPLACEMENTS IN LOAD VECTOR R
  320 R(N) = R(N) - AK(N,K)*R(L)
      GO TO 322
 321  CP=R(1)/AK(1,1)
      R(1)=CP
 322  RETURN
      END
```

```
INTEGR
CINTEGR
      SUBROUTINE INTEGR(I,INTGR)
      I=INTGR
      RETURN
      END
```

```
TRIOUT
CTRIOUT
      SUBROUTINE TRIOUT(CO,NF,JT,TPRP,P,CKFOR,SX,SY,SXY,NDOF)
      DIMENSION CO(NDOF,1),JT(3,1),TPRP(4,1),P(1),CKFOR(NDOF,1),SX(1),
     * NF(NDOF,1),SY(1),SXY(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NIO,
     *       IFLAG,NB,NDOFN,ISTATE
      COMMON/BLK1/NLC,NTCOMP,NCCOMP
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      DIMENSION AK(9,9),AKT(9,9),T(9,9),XLOC(3),YLOC(3)
      DIMENSION FNODE(9),DF(9),U(3),V(3)
      REAL NU
      DO 310 I=1,NTRIS
      IF(TPRP(1,I).LE.0)GO TO 310
      IF(TPRP(3,I).LE.0)GO TO 310
      NTCOMP=0
      LOC=3*NDOF
      DO 311 J=1,LOC
      DO 311 K=1,LOC
      AKT(J,K)=0.
      AK(J,K)=0.
311   T(J,K)=0.
      J1=JT(1,I)
      J2=JT(2,I)
      J3=JT(3,I)
C
C   COMPUTE DIRECTION COSINE MATRIX T
C
      DX=CO(1,J2)-CO(1,J1)
      DY=CO(2,J2)-CO(2,J1)
      DZ=0
      GO TO (312,312,313),NDOF
313   DZ=CO(3,J2)-CO(3,J1)
312   XL12=SQRT(DX*DX+DY*DY+DZ*DZ)
      A1=DX/XL12
      A2=DY/XL12
      A3=DZ/XL12
      DX=CO(1,J3)-CO(1,J1)
      DY=CO(2,J3)-CO(2,J1)
      DZ=0
      GO TO (314,314,315),NDOF
315   DZ=CO(3,J2)-CO(3,J1)
314   XL13=SQRT(DX*DX+DY*DY+DZ*DZ)
      D1=DX/XL13
      D2=DY/XL13
      D3=DZ/XL13
      C1=A2*D3-A3*D2
      C2=A3*D1-A1*D3
      C3=A1*D2-A2*D1
      XLZ=SQRT(C1*C1+C2*C2+C3*C3)
      C1=C1/XLZ
      C2=C2/XLZ
      C3=C3/XLZ
      B1=C2*A3-C3*A2
      B2=C3*A1-C1*A3
      B3=C1*A2-C2*A1
      T(1,1)=A1
      T(1,2)=A2
      T(1,3)=A3
      T(2,1)=B1
      T(2,2)=B2
      T(2,3)=B3
```

```
      DO 318 J=1,NDOF
      DO 318 K=1,NDOF
      T(J+2,K+NDOF)=T(J,K)
318   T(J+4,K+2*NDOF)=T(J,K)
C
C   COMPUTE LOCAL COORDINATES
      DY=CO(2,J2)-CO(2,J1)
      DZ=0
      GO TO (319,319,320),NDOF
320   DZ=CO(3,J2)-CO(3,J1)
319   XLOC(1)=0
      YLOC(1)=0
      XLOC(2)=DX*T(1,1)+DY*T(1,2)+DZ*T(1,3)
      YLOC(2)=0
      DX=CO(1,J3)-CO(1,J1)
      DY=CO(2,J3)-CO(2,J1)
      DZ=0
      GO TO (321,321,322),NDOF
322   DZ=CO(3,J2)-CO(3,J1)
321   XLOC(3)=DX*T(1,1)+DY*T(1,2)+DZ*T(1,3)
      YLOC(3)=DX*T(2,1)+DY*T(2,2)+DZ*T(2,3)
C
C   COMPUTE ELEMENT LOCAL STIFFNESS MATRIX
C
      EX=TPRP(1,I)
      NU=TPRP(2,I)
      TSK=TPRP(3,I)
      G=EX/(2.*(1+NU))
      TOV4A=TSK/(2.*XLOC(2)*YLOC(3))
C
C  ISTATE = 0 (PLANE STRAIN) ; ISTATE = 1 (PLANE STRESS)
C
      IF(ISTATE.GT.0) GO TO 401
      D0=2.*G/(1.-2.*NU)
      D11=D0*(1.-NU)*TOV4A
      D12=D0*NU*TOV4A
      D22=D11
      D33=G*TOV4A
      GO TO 410
401   D0=EX/(1.-NU*NU)
      D11=D0*TOV4A
      D12=D0*NU*TOV4A
      D22=(EX/(1.-NU*NU))*TOV4A
      D33=G*TOV4A
410 CONTINUE
      AK(1,1)= D11*YLOC(3)*YLOC(3)+D33*(XLOC(3)-XLOC(2))*(XLOC(3)
     1         -XLOC(2))
      AK(2,1)=-(D12+D33)*(XLOC(3)-XLOC(2))*YLOC(3)
      AK(2,2)= D22*(XLOC(3)-XLOC(2))*(XLOC(3)-XLOC(2))+D33*YLOC(3)
     1         *YLOC(3)
      AK(3,1)=-D11*YLOC(3)*YLOC(3)-D33*XLOC(3)*(XLOC(3)-XLOC(2))
      AK(3,2)= D12*YLOC(3)*(XLOC(3)-XLOC(2))+D33*XLOC(3)*YLOC(3)
      AK(3,3)= D11*YLOC(3)*YLOC(3)+D33*XLOC(3)*XLOC(3)
      AK(4,1)= D12*XLOC(3)*YLOC(3)+D33*YLOC(3)*(XLOC(3)-XLOC(2))
      AK(4,2)=-D22*XLOC(3)*(XLOC(3)-XLOC(2))-D33*YLOC(3)*YLOC(3)
      AK(4,3)=-(D12+D33)*XLOC(3)*YLOC(3)
      AK(4,4)= D22*XLOC(3)*XLOC(3)+D33*YLOC(3)*YLOC(3)
      AK(5,1)= D33*XLOC(2)*(XLOC(3)-XLOC(2))
      AK(5,2)=-D33*XLOC(2)*YLOC(3)
      AK(5,3)=-D33*XLOC(3)*XLOC(2)
      AK(5,4)=-AK(5,2)
      AK(5,5)= D33*XLOC(2)*XLOC(2)
      AK(6,1)=-D12*XLOC(2)*YLOC(3)
      AK(6,2)= D22*XLOC(2)*(XLOC(3)-XLOC(2))
```

```
      AK(6,3)=-AK(6,1)
      AK(6,4)=-D22*XLOC(2)*XLOC(3)
      AK(6,5)= 0.
      AK(6,6)= D22*XLOC(2)*XLOC(2)
      DO 323 J=2,6
      M=J-1
      DO 323 K=1,M
323   AK(K,J)=AK(J,K)
C
C   COMPUTE ELEMENT GLOBAL STIFFNESS,AK
C
      M=3*NDOF
      DO 324 J=1,M
      DO 324 K=1,M
      DO 324 L=1,M
324   AKT(J,K)=AKT(J,K)+AK(J,L)*T(L,K)
      DO 325 J=1,M
      DO 325 K=1,M
      AK(J,K)=0.0
      DO 325 L=1,M
325   AK(J,K)=AK(J,K)+T(L,J)*AKT(L,K)
      DO 332 J=1,NDOF
      IFR1=IABS(NF(J,J1))
      IFR2=IABS(NF(J,J2))
      IFR3=IABS(NF(J,J3))
      DF(J)        =P(IFR1)
      IF(IFR1.EQ.0) DF(J)=0.0
      DF(J+NDOF)   =P(IFR2)
      IF(IFR2.EQ.0) DF(J+NDOF)=0.0
      DF(J+2*NDOF)=P(IFR3)
332   IF(IFR3.EQ.0) DF(J+2*NDOF)=0.0
      DO 333 J=1,M
      FNODE(J)=0.0
      DO 333 K=1,M
333   FNODE(J)=FNODE(J)+AK(J,K)*DF(K)
C
C     COMPUTE NODAL FORCE SUMMATIONS
C
      DO 334 J=1,NDOF
      CKFOR(J,J1)=CKFOR(J,J1)-FNODE(J)
      CKFOR(J,J2)=CKFOR(J,J2)-FNODE(J+NDOF)
334   CKFOR(J,J3)=CKFOR(J,J3)-FNODE(J+2*NDOF)
      IF(NTCOMP)340,340,310
340   D11=2.*D11/TSK
      D12=2.*D12/TSK
      D33=2.*D33/TSK
      DO 335 K=1,3
      LOC=NDOF*(K-1)
      U(K)=0.0
      V(K)=0.0
      DO 335 J=1,NDOF
      U(K)=U(K)+DF(LOC+J)*T(1,J)
335   V(K)=V(K)+DF(LOC+J)*T(2,J)
      U21=U(2)-U(1)
      U31=U(3)-U(1)
      V12=V(1)-V(2)
      V31=V(3)-V(1)
C
C   COMPUTE ELEMENT STRESSES IN LOCAL COORDINATES
C
      SGX=D11*YLOC(3)*U21+D12*(XLOC(3)*V12+XLOC(2)*V31)
      SGY=D12*YLOC(3)*U21+D11*(XLOC(3)*V12+XLOC(2)*V31)
      TXY=-D33*(YLOC(3)*V12+XLOC(3)*U21-XLOC(2)*U31)
      IF(NTCOMP)336,336,337
```

```
336     SX(I)=SGX
        SY(I)=SGY
        SXY(I)=TXY
        GO TO 310
C
C     TRANSFORM TO GLOBAL COORDINATES
C
337     IF(NDOF-2)338,338,339
339     WRITE(6,100)
100     FORMAT(1H1/,10X,44HSTRESSES WILL BE OUTPUT IN LOCAL COORDINATES)
        NTCOMP=1
        GO TO 310
338     SX(I)=T(1,1)*T(1,1)*SGX+T(2,1)*T(2,1)*SGY+T(1,1)*T(2,1)*TXY
        SY(I)=T(1,2)*T(1,2)*SGX+T(2,2)*T(2,2)*SGY+T(1,2)*T(2,2)*TXY
        SXY(I)=T(1,1)*T(1,2)*SGX+T(2,1)*T(2,2)*SGY+T(1,1)*T(2,2)*TXY
310     CONTINUE
        RETURN
        END
```

```
SPROUT
CSPROUT
      SUBROUTINE SPROUT(CO,NF,JS,SPRP,P,CKFOR,ELFOR)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLONDS,NRSTR,NER,NRR,NIO,
     *       IFLAG,NB,NDOF,ISTATE,NKORE,IDYNAM
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      DIMENSION CO(1),NF(1),JS(1),SPRP(1),P(1),CKFOR(1),ELFOR(1)
     1   ,DICOS(6,6),AKT(6,6),AK(6,6),DF(6),FNODE(6)
      REAL K1,K2,K3
      DO 10 KOUNT=1,NSPRNG
      DO 5 I=1,6
      DO 5 J=1,6
      DICOS(I,J)=0.0
      AK(I,J)=0.0
      AKT(I,J)=0.0
    5 CONTINUE
      LOC=4*(KOUNT-1)
C
C FORM DIRECTION COSINES FOR K1-K2 PLANE
C
      LOC1=NDOF*(JS(LOC+1)-1)
      LOC3=NDOF*(JS(LOC+3)-1)
      LOC4=NDOF*(JS(LOC+4)-1)
C
C FORM COMPONENTS OF VECTORS FROM NODES 1 TO 3 AND 1 TO 4
C
      DX=CO(LOC3+1)-CO(LOC1+1)
      DY=CO(LOC3+2)-CO(LOC1+2)
      DZ=0.0
      GO TO (20,20,15),NDOF
   15 DZ=CO(LOC3+3)-CO(LOC1+3)
   20 DL13=SQRT(DX*DX+DY*DY+DZ*DZ)
      A1=DX/DL13
      A2=DY/DL13
      A3=DZ/DL13
      DX=CO(LOC4+1)-CO(LOC1+1)
      DY=CO(LOC4+2)-CO(LOC1+2)
      DZ=0.0
      GO TO (30,30,25),NDOF
   25 DZ=CO(LOC4+3)-CO(LOC1+3)
   30 DL14=SQRT(DX*DX+DY*DY+DZ*DZ)
      B1=DX/DL14
      B2=DY/DL14
      B3=DZ/DL14
      C1=A2*B3-A3*B2
      C2=A3*B1-A1*B3
      C3=A1*B2-A2*B1
      DLC=SQRT(C1*C1+C2*C2+C3*C3)
      C1=C1/DLC
      C2=C2/DLC
      C3=C3/DLC
      B1=C2*A3-C3*A2
      B2=C3*A1-C1*A3
      B3=C1*A2-A1*C2
      DICOS(1,1)=A1
      DICOS(1,2)=A2
      DICOS(1,3)=A3
      DICOS(2,1)=B1
      DICOS(2,2)=B2
      DICOS(2,3)=B3
      GO TO(32,32,31),NDOF
   31 DICOS(3,1)=C1
      DICOS(3,2)=C2
```

```
      DICOS(3,3)=C3
   32 DO 35 I=1,NDOF
      DO 35 J=1,NDOF
   35 DICOS(I+NDOF,J+NDOF)=DICOS(I,J)
C
C FORM ELEMENTS OF LOCAL STIFFNESS MATRIX
C
      K1=SPRP(3*KOUNT-2)
      K2=SPRP(3*KOUNT-1)
      K3=SPRP(3*KOUNT)
C
C FORM LOCAL STIFFNESS MATRIX, AK
C
      M=2*NDOF
      AK(1,1)=K1
      AK(1,NDOF+1)=-K1
      AK(1+NDOF,1+NDOF)=K1
      AK(2,2)=K2
      AK(2,NDOF+2)=-K2
      AK(2+NDOF,2+NDOF)=K2
      GO TO (40,40,36),NDOF
   36 AK(3,3)=K3
      AK(3,6)=-K3
      AK(6,6)=K3
   40 IJ=M-1
      DO 45 I=1,IJ
      JI=I+1
      DO 45 J=JI,M
      AK(J,I)=AK(I,J)
   45 CONTINUE
      DO 50 I=1,M
      DO 50 J=1,M
      AKT(I,J)=0.0
      DO 50 K=1,M
      AKT(I,J)=AKT(I,J)+AK(I,K)*DICOS(K,J)
   50 CONTINUE
C
C COMPUTE ELEMENT FORCES
C
      DO 55 I=1,2
      LOC1=NDOF*(JS(LOC+I)-1)
      IJ=NDOF*(I-1)
      DO 55 J=1,NDOF
      LOC2=NF(LOC1+J)
      DF(IJ+J)=P(LOC2)
      IF(IABS(NF(LOC1+J)).EQ.0)DF(IJ+J)=0.0
   55 CONTINUE
      DO 60 I=1,NDOF
      ELFOR(LOC+I)=0.0
      DO 60 J=1,M
      ELFOR(LOC+I)=ELFOR(LOC+I)+AKT(I,J)*DF(J)
   60 CONTINUE
      ELFOR(LOC+4)=SQRT(ELFOR(LOC+1)*ELFOR(LOC+1)+ELFOR(LOC+2)
     *                 *ELFOR(LOC+2))
C
C COMPUTE NODAL EQUILIBRIUM CHECKS
C
      DO 65 I=1,M
      DO 65 J=1,M
      AK(I,J)=0.0
      DO 65 K=1,M
      AK(I,J)=AK(I,J)+DICOS(K,I)*AKT(K,J)
   65 CONTINUE
      DO 70 I=1,M
```

```
      FNODE(I)=0.0
      DO 70 J=1,M
      FNODE(I)=FNODE(I)+AK(I,J)*DF(J)
   70 CONTINUE
      DO 75 I=1,2
      IJ=NDOF*(I-1)
      LOC1=NDOF*(JS(LOC+I)-1)
      DO 75 J=1,NDOF
      CKFOR(LOC1+J)=CKFOR(LOC1+J)-FNODE(IJ+J)
   75 CONTINUE
   10 CONTINUE
      RETURN
      END
```

```
CRKOT8
CCRKOT8
      SUBROUTINE CRKOT8(NF,JC,CPRP,P,CKFOR,CUMKC,DINV,DICOS,SIF1)
      DIMENSION NF(1),JC(1),CPRP(1),CKFOR(1),CUMKC(24,24),
     *          DINV(16,16),SIF1(1),C(16),LOCATE(24),DF(24),FNODE(24),
     *          P(1),D(24),DICOS(16,24)
      COMMON NNODES,NTRIS,NSPRNG,NCRKS8,NCRK10,NLUMPS,
     *       NLDNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,N8,NDOF
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,IOUT1,IOUT2,IOUT3,IOUT4,IOUT5
      IF(ICPRT.GT.1)WRITE(6,400)
  400 FORMAT(10X,15HCRAKOT8 ENTERED/)
      REWIND 25
      DO 900 KOUNT=1,NCRKS8
      READ(25) (DEL,G,THICK,DINV,DICOS,CUMKC)
      IF(THICK)5,5,6
    5 SIF1(KOUNT)=0.0
      GO TO 900
    6 CONTINUE
      LOC=9*(KOUNT-1)
      DO 10 I=1,8
      LOCI=NDOF*(I-1)
      NODE=JC(LOC+I)
      LOCND=NDOF*(NODE-1)
      DO 10 J=1,NDOF
   10 LOCATE(LOCI+J)=NF(LOCND+J)
      IF(ICPRT.GT.1)WRITE(IOUT1,100)(LOCATE(I),I=1,16)
  100 FORMAT(1X,9HLOCATE(I),5X,8I5/15X,8I5)
      DO 20 I=1,8
      LOCI=NDOF*(I-1)
      DO 20 J=1,NDOF
      LOCND=IABS(LOCATE(LOCI+J))
      DF(LOCI+J)=P(LOCND)
      IF(IABS(LOCND).EQ.0)DF(LOCI+J)=0.0
   20 CONTINUE
      IF(ICPRT.GT.1)WRITE(IOUT1,200)(DF(I),I=1,16)
  200 FORMAT(1X,5HDF(I),4X,1P8E11.3,/10X,1P8E11.3)
C
C       COMPUTE THE C(I)'S AND STRESS INTENSITY FACTOR K(1)
C
      II=8*NDOF
      DO 25 I=1,16
      D(I)=0.0
      DO 25 J=1,II
      D(I)=D(I)+DICOS(I,J)*DF(J)
   25 CONTINUE
      IF(ICPRT.GT.1)WRITE(IOUT1,200) (D(I),I=1,16)
      DO 30 I=1,16
      C(I)=0.0
      DO 30 J=1,16
   30 C(I)=C(I)+DINV(I,J)*D(J)/DEL
      PI=4.0*ATAN(1.0)
      SIF1(KOUNT)=12.*SQRT(2.*PI*DEL)*G*C(1)
      IF(ICPRT.GT.1)WRITE(IOUT1,300)(C(I),I=1,16)
  300 FORMAT(1X,4HC(I),5X,8E15.8,/10X,8E15.8)
C
C       COMPUTE ELEMENT NODAL FORCES
C
      DO 40 I=1,II
      FNODE(I)=0.0
      DO 40 J=1,II
   40 FNODE(I)=FNODE(I)+CUMKC(I,J)*DF(J)
      DO 50 I=1,8
```

```
      NODE=JC(LOC+I)
      LOCND=NDOF*(NODE-1)
      LOCI=NDOF*(I-1)
      DO 50 J=1,NDOF
   50 CKFOR(LOCND+J)=CKFOR(LOCND+J)-FNODE(LOCI+J)
  900 CONTINUE
      RETURN
      END
```

```
CRKO1C
CCRKO10
      SUBROUTINE CRKO10(NF,JC,CPRP,P,CKFOR,CUMKC,DINV,DICOS,SIF1,SIF2)
      DIMENSION NF(1),JC(1),CPRP(1),CKFOR(1),CUMKC(30,30),
     *          DINV(20,20),SIF1(1),C(20),LOCATE(30),DF(30),FNODE(30),
     *          P(1),D(30),DICOS(20,30),SIF2(1)
      COMMON NNODES,NTRIS,NSPRNG,NCRK8,NCRK10,NLUMPS,
     *       NLCNDS,NRSTR,NER,NRR,NID,
     *       IFLAG,N8,NDOF
      COMMON/BLK2/INPRT,IKPRT,ITPRT,ICPRT,IOPRT
      COMMON/BLK3/IIN8,IIN10,ICUT1,IOUT2,IOUT3,IOUT4,IOUT5
      IF(ICPRT.GT.1)PRINT 400
400   FORMAT(/20X,15HCRAKO10 ENTERED)
      REWIND 26
      DO 900 KOUNT=1,NCRK10
      READ(26) DEL,G,THICK,DINV,DICOS,CUMKC
      IF(THICK)5,5,6
    5 SIF1(KOUNT)=0.0
      SIF2(KOUNT)=0.0
      GO TO 900
    6 CONTINUE
      LOC=11*(KOUNT-1)
      DO 10 I=1,10
      LOCI=NDOF*(I-1)
      NODE=JC(LOC+I)
      LOCND=NDOF*(NODE-1)
      DO 10 J=1,NDOF
 10   LOCATE(LOCI+J)=NF(LOCND+J)
      II=10*NDOF
      IF(ICPRT.GT.1)WRITE(IOUT1,100) (LOCATE(I),I=1,II)
100   FORMAT(1X,9HLOCATE(I),5X,10I5)
      DO 20 I=1,10
      LOCI=NDOF*(I-1)
      DO 20 J=1,NDOF
      LOCND=IABS(LOCATE(LOCI+J))
      DF(LOCI+J)=P(LOCND)
      IF(IABS(LOCND).EQ.0)DF(LOCI+J)=0.0
 20   CONTINUE
      IF(ICPRT.GT.1)WRITE(IOUT1,200) (DF(I),I=1,II)
200   FORMAT(1X,5HDF(I),4X,8E15.8,/10X,8E15.8)
C
C       COMPUTE THE C(I)"S AND STRESS INTENSITY FACTOR K(1)
C
      DO 25 I=1,20
      D(I)=0.0
      DO 25 J=1,II
      D(I)=D(I)+DICOS(I,J)*DF(J)
   25 CONTINUE
      IF(ICPRT.GT.1)WRITE(IOUT1,200) (D(I),I=1,20)
      DO 30 I=1,20
      C(I)=0.0
      DO 30 J=1,20
 30   C(I)=C(I)+DINV(I,J)*D(J)/DEL
      PI=4.0*ATAN(1.0)
      SIF1(KOUNT)=12.*SQRT(2.*PI*DEL)*G*C(1)
      SIF2(KOUNT)=4.*SQRT(2.*PI*DEL)*G*C(11)
      IF(ICPRT.GT.1)WRITE(IOUT1,300) (C(I),I=1,20)
300   FORMAT(1X,4HC(I),5X,8E15.8,/10X,8E15.8,/10X,4E15.8)
C
C       COMPUTE ELEMENT NODAL FORCES
C
      DO 40 I=1,II
      FNODE(I)=0.0
      DO 40 J=1,II
```

```
 40   FNODE(I)=FNODE(I)+CUMKC(I,J)*DF(J)
      DO 50 I=1,10
      NODE=JC(LOC+I)
      LOCND=NDOF*(NODE-1)
      LOCI=NDOF*(I-1)
      DO 50 J=1,NDOF
 50   CKFOR(LOCND+J)=CKFOR(LOCND+J)-FNODE(LOCI+J)
900   CONTINUE
      PRINT 500
 500  FORMAT(1H1,/10X,"10-NODE CRACKED ELEMENT STRESS INTENSITY FACTORS"
     *       //10X,"ELEMENT",11X,"MODE 1 K",7X,"MODE 2 K"/)
      PRINT 600, (I,SIF1(I),SIF2(I),I=1,NCRK10)
 600  FORMAT(10X,I5,12X,1PE10.3,5X,1PE10.3/)
      RETURN
      END
```

1.3 NASTRAN SUBROUTINES

Following are listings of the subroutines which are to be inserted as dummy elements into the Armament Laboratory's version of NASTRAN. These subroutines compute the stiffness matrix (subroutines KDUM1, KDMK1A, KDMK2A, KDMK2A, KDMK1B, and KDMK2B), the lumped-mass matrix (subroutine MDUM1), and the time-dependent Mode I and Mode II stress-intensity factors (K_I and K_{II}) subroutines SDUM 11 and SDUM 12) for any 10-node crack element contained in a NASTRAN model. For analysis of a purely symmetric (Mode I) crack problem, multipoint constraint options (MPC) are specified to guarantee that the nodal displacements for the 10-node element are symmetric about the line of the crack. For such analyses K_{II} will be calculated to be zero.

```
      SUBROUTINE KDUM1
C*****
C THIS ROUTINE COMPUTES THE TEN 6X6 MATRICES K(NPVT,NPVT) AND K(NPVT,J)
C J=1,2,3,4,5,6,7,8,9 FOR A 10-NODE CRACK ELEMENT
C*****
C CALLS FROM THIS ROUTINE ARE MADE TO THE FOLLOWING
C         MESAGE - ERROR MESSAGE WRITER
C         MAT    - MATERIAL DATA ROUTINE
C         SMALB  - INSERTION ROUTINE
C         INVERD - DOUBLE PRECISION INVERSE ROUTINE
C         TRANSD - DOUBLE PRECISION TRANSFORMATION SUPPLIER
C         GMMATD - DOUBLE PRECISION MATRIX MULTIPLY AND TRANSPOSE
C*****
C
      LOGICAL HEAT
C
      DOUBLE PRECISION KE,A,B,DZ(1),PI,FR2,UR,UT
C
      DIMENSION IZ(1),IECPT(1)
C
      REAL NU
C
      COMMON ICOM
C
      COMMON  /SYSTEM/ ISYS,NO
C
      COMMON  /SMA1X / RZ(1)
C
      COMMON  /SMA1BK/ ICSTM ,NCSTM ,IGPCT ,IPOINT,NPOINT
     1,                I6X6K ,N6X6K ,I6X64 ,N6X64
C
      COMMON  /SMA1IO/ IFCSTM,IPMPT ,IFDIT ,IDUM1 ,IFECPT,IGECPT
     1,                IFGPCT,IGGPCT,IFGEI ,IGGEI ,IFKGG ,IGKGG
     2,                IF4GG ,IG4GG ,IFGPST,IGGPST,INRW  ,OUTRW
     3,                CLSNRW,CLSRW ,NEOR  ,EOR   ,MCBKGG(7),MCB4GG(7)
C
      COMMON  /SMA1CL/ IOPT4 ,K4GGWS,NPVT  ,LEFT  ,FROWIC,LROWIC
     1,                NROWSC,TNROWS,JMAX  ,NLINKS,LINK(10),IDETCK
     2,                DODET ,NOGO
C
      COMMON  /SMA1HT/ HEAT
C
      COMMON  /SMA1ET/ ECPT(56),IND(44)
C
      COMMON  /MATIN / MATID ,INFLAG,TEMP  ,PLAARG,SINTH ,COSTH
C
      COMMON  /MATOUT/ E     ,G     ,NU    ,RHO   ,ALPHA ,TSUBO
     1,                GSUBE ,SIGT  ,SIGC  ,SIGS
C
C                     THERMAL CONDUCTIVITY
      COMMON  /HMTOUT/ FK
C
      COMMON  /SMA1DP/ KE(36),A(219),B(45)
C
      EQUIVALENCE (RZ(1),IZ(1),DZ(1)),(ECPT(1),IECPT(1))
C
      DATA PI,FR2/3.141592653589793200,1.189207115002721000/
```

```
C
C*****                                                          119,120
C
C CHECK ELEMENT GEOMETRY AND LOCATE PIVOT POINT
C
      NOPVT=0
      DO 10 I=2,11
   10 IF (IECPT(I),EQ.NPVT) NOPVT=I-1
      IF (NOPVT.EQ.0) GO TO 2010
      IF (ECPT(13) 2020,2020,20
   20 IF ( (IECPT(15).LT.1).OR .(IECPT(15).GT.2)) GO TO 2020
      DO 30 I=16,52,4
      J=(I/4-4)*3
      A(J+1)=ECPT(I+1)
      A(J+2)=ECPT(I+2)
   30 A(J+3)=ECPT(I+3)
      A(216)=DSQRT((A(28)-A(1))*(A(28)-A(1))+(A(29)-A(2))*(A(29)-A(2))
     1            +(A(30)-A(3))*(A(30)-A(3)))
      A(217)=(a(5)-A(2))*(a(9)-A(3))-(A(8)-A(2))*(A(6)-A(3))
      A(218)=(A(7)-A(1))*(A(6)-A(3))-(A(4)-A(1))*(A(9)-A(3))
      A(219)=(A(4)-A(1))*(A(8)-A(2))-(A(7)-A(1))*(A(5)-A(2))
      DO 40 I=2,10
   40 A(I+29)=DSQRT((A(3*I-2)-A(3*I-5))*(A(3*I-2)-A(3*I-5))+
     1              (A(3*I-1)-A(3*I-4))*(A(3*I-1)-A(3*I-4))+
     2              (A(3**I )-A(3*I-3))*(A(3*I  )-A(3*I-3)))
      DO 50 I=3,9
   50 A(I+37)=DSQRT((A(3*I-2)-A(1))*(A(3*I-2)-A(1))+
     1              (A(3*I-1)-A(2))*(A(3*I-1)-A(2))+
     2              (A(3*I  )-A(3))*(A(3*I  )-A(3)))
      DO 60 I=4,10
   60 A(I+43)=A(217)*(A(3*I-2)-A(1))+A(218)*(A(3*I-1)-A(2))+
     1        A(219)*(A(3*I  )-A(3))
      A(54)=DSQRT((A(28)-A(4))*(A(28)-A(4))+(A(29)-A(5))*(A(29)-A(5))
     1           +(A(30)-A(6))*(A(30)-A(6)))
C
C ARE NODES 2 AND 10 COIN CIDENT
C
      IF (A(54)/A(216).GT.1.0D-4) GO TO 2020
C
C IS NODE 1 AT CENTER OF ELEMENT
C
      DO 70 I=41,45,2
   70 IF (DABS(1.0-A(I)/A(216)).GT.1.0D-4) GO TO 2020
C
C ARE NODES EQUALLY SPACED
C
      DO 80 I=31,39
   80 IF (DABS(1.0-A(I)/A(216)).GT.1.0D-4) GO TO 2020
C
C IS ELEMENT SQUARE
C
      DO 90 I=40,46,2
   90 IF (DABS(DSQRT(2.0D0)-A(I)/A(216)).GT.1.0D-4) GO TO 2020
C
```

CHECK ELEMENT GEOMETRY AND LOCATE PIVOT POINT (CONTINUED)

```
C IS ELEMENT PLANAR
C
      DO 100 I=47,53
  100 IF (DABS(A(I))/A(216)/A(216)/A(216),GT.1.00-2) GO TO 2020
C
  ELEMENT GEOMETRY CHECK COMPLETE
```

```
C
C*****
C
C SET UP ELEMENT LOCAL COORDINATE SYSTEM DIRECTION COSINES
      A(212)=A(43)
      A(214)=(A(16)-A(1))/A(43)
      A(217)=(A(22)-A(1))/A(45)
      A(215)=(A(17)-A(2))/A(43)
      A(218)=(A(23)-A(2))/A(45)
      A(216)=(A(18)-A(3))/A(43)
      A(219)=(A(24)-A(3))/A(45)
      IF (DABS(A(214)*A(217)+A(215)*A(218)+A(216)*A(219)).GT.1.0D-5)
     1   GO TO 2020
C
C LOCATE THE ELEMENT MATERIAL PROPERTIES  (E,G,NU)
C
      MATID=IECPT(12)
      INFLAG=1
      TEMP=ECPT(56)
      CALL MAT(IECPT(1))
C
C BRANCH ON HEAT FORMULATION
C
      IF (HEAT) GO TO 500
C
C CALCULATE PLANE STRESS OR PLANE STRAIN PARAMETER
C
      IF (IECPT(15).EQ.1) A(213)=DBLE(NU)/(1.0D0+DBLE(NU))
      IF (IECPT(15).EQ.2) A(213)=DBLE(NU)
C
C CALCULATE ELEMENT LOCAL COORDINATE SYSTEM STIFFNESS MATRIX
C
C*****
C
      DO 200 M=1,2
      DO 110 J=1,5
      I=J+5
      DO 110 N=1,9
      GO TO (101,102),M
  101 UR=(N+2.0D0)/(N+2.0D0*(-1.0D0)**N)
      UR=-UR*(6.0D0-8.0D0*A(213)-N)*DCOS((N-2.0D0)*(I-6.0D0)*PI/8.0D0)
      UR= UR-(N+2.0D0)*DCOS((N+2.0D0)*(I-6.0D0)*PI/8.0D0)
      UT=(N+2.0D0)/(N+2.0D0*(-1.0D0)**N)
      UT=-UT*(6.0D0-8.0D0*A(213)+N)*DSIN((N-2.0D0)*(I-6.0D0)*PI/8.0D0)
      UT= UT+(N+2.0D0)*DSIN((N+2.0D0)*(I-6.0D0)*PI/8.0D0)
      GO TO 103
  102 UR=(N+2.0D0)*((N-2.0D0*(-1.0D0)**N)/(N+2.0D0))*DSIN((N+2.0D0)*
     1   (I-6.0D0)*PI/8.0D0)+(6.0D0-8.0D0*A(213)-N)*DSIN((N-2.0D0)*
     2   (I-6.0D0)*PI/8.0D0)
      UT=(N+2.0D0)*((N-2.0D0*(-1.0D0)**N)/(N+2.0D0))*DCOS((N+2.0D0)*
     1   (I-6.0D0)*PI/8.0D0)-(6.0D0-8.0D0*A(213)+N)*DCOS((N-2.0D0)*
     2   (I-6.0D0)*PI/8.0D0)
  103 IF (I-(I/2)*2) 104,105,104
  104 UR=(FR2**N)*UR
```

```
SET UP ELEMENT LOCAL COORDINATE SYSTEM DIRECTION COSINES  (CONTINUED)
      UT=(FR2**N)*UT
105   A(10*N+2*J+89)=UR*DCOS((I-6)*PI/4.0D0)-UT*DSIN((I-6)*PI/4.0D0)
110   A(10*N+2*J+90)=UR*DSIN((I-6)*PI/4.0D0)+UT*DCOS((I-6)*PI/4.0D0)
      IF (M.EQ.2) GO TO 120
      DO 111 I=102,182,10
      A(I)=A(I-1)
```

```
111 A(I-1)=0.0D0
    DO 112 I=191,200,2
    A(I)=1,0D0
112 A(I+1)=0.0D0
    A(192)=1.0D0
    GO TO 130
120 DO 121 I=101,181,10
121 A(I)=0.0D0
    DO 122 I=111,120,2
    A(I)=0.0D0
122 A(I+1)=1.0D0
    A(111)=1.0D0
    A(191)= 0.0D0
    A(192)= 1.0D0
    A(193)=-1.0D0
    A(194)= 1.0D0
    A(195)=-1.0D0
    A(196)= 0.0D0
    A(197)=-1.0D0
    A(198)=-1.0D0
    A(199)= 0.0D0
    A(200)=-1.0D0
130 CALL INVERD(10,A(101),10,A(201),0,A(211),IND(31),IND(1))
    IF (IND(31).NE.1) CALL MESAGE(-30,26,IECPT(1))
    IF (M,EQ.1) CALL KDMK1A(B)
    IF (M.EQ.2) CALL KDMK1B(B)
    DO 135 I=1,45
135 A(I)=B(I)
    IF (M.EQ.1) CALL KDMK2A(B)
    IF (M.EQ.2) CALL KDMK2B(B)
    DO 140 I=1,45
140 B(I)=A(213)*B(I)+A(I)
    DO 150 I=1,10
    DO 150 J=1,10
    A(10*J+I-10)=0.0D0
    DO 150 K=1,9
    DO 150 L=1,9
    IF (K.GE.L) KL=L
    IF (L.GE.K) KL=K
    N=-9
    DO 145 LK=1,KL
145 N=N+(11-LK)
    N=N+IABS(K-L)
150 A(10*J+I-10)=A(10*J+I-10)+A(10*I+K+90)*B(N)*A(10*J+L+90)
    IF (M.EQ.2) GO TO 175
    GO TO (155,160,160,160,160,165,170,170,170,170),NOPVT
155 DO 156 I=1,10
    KE(2*I-1)=A(I)
156 KE(2*I  )=0.0D0
    GO TO 200
160 J=120-20*NOPVT
    DO 161 I=1,10
    N=J+I
    KE(2*I-1)=0.5D0*A(N)
```

```
          (CONTINUED)
161 KE(2*I   )=-.5D0*A(N+10)
    GO TO 200
165 DO 166 I=1,10
    KE(2*I-1)=A(I+10)
166 KE(2*I   )=0.0D0
    GO TO 200
```

```
170 J=20*NOPVT-120
    DO 171 I=1,10
    N=J+I
    KE(2*I-1)=0.5D0*A(N)
171 KE(2*I  )=0.5D0*A(N+10)
    GO TO 200
175 GO TO (180,185,185,185,185,190,195,195,195,195),NOPVT
180 DO 181 I=1,10
    A(2*I+119)=0.0D0
181 A(2*I+120)=A(I)
    GO TO 200
185 J=120-20*NOPVT
    DO 186 I=1,10
    N=J+I
    A(2*I+119)=-.5D0*A(N)
186 A(2*I+120)=0.5D0*A(N+10)
    GO TO 200
190 DO 191 I=1,10
    A(2*I+119)=0.0D0
191 A(2*I+120)=A(I+10)
    GO TO 200
195 J=20*NOPVT-120
    DO 196 I=1,10
    N=J+I
    A(2*I+119)=0.5D0*A(N)
196 A(2*I+120)=0.5D0*A(N+10)
200 CONTINUE
    DO 210 I=1,20
210 A(I+100)=KE(I)
    DO 245 I=1,20
    GO TO (215,220,225,227,225,227,225,227,225,227,230,235,240,240,
   1       240,240,240,240,240,240),I
215 A(1)=A(101)
    A(2)=A(102)
    GO TO 245
220 A(3)=A(121)
    A(4)=A(122)
    GO TO 245
225 J=12-I
    A(2*I-1)=0.5D0*A(2*J+ 99)-.5D0*A(2*J+119)
    A(2*I  )=0.5D0*A(2*J+100)-.5D0*A(2*J+120)
    GO TO 245
227 J=14-I
    A(2*I-1)=-.5D0*A(2*J+ 99)+.5D0*A(2*J+119)
    A(2*I  )=-.5D0*A(2*J+100)+.5D0*A(2*J+120)
    GO TO 245
230 A(21)=A(103)
    A(22)=A(104)
    GO TO 245
235 A(23)=A(123)
    A(24)=A(124)
    GO TO 245
240 A(2*I-1)=0.5D0*A(2*I+79)+0.5D0*A(2*I+ 99)
    A(2*I  )=0.5D0*A(2*I+80)+0.5D0*A(2*I+100)
245 CONTINUE
```

(CONTINUED)

```
C
C ELEMENT LOCAL COORDINATE SYSTEM STIFFNESS MATRIX COMPLETE  A(1),A(40)
C*****
C
C TRANSFORM TO GLOBAL COORDINATES AND STORE INTO KGG MATRIX
```

```
C
      DO 400 I=1,10
      DO 310 J=1,36
  310 KE(J)=0.0D0
      A(41)=.25D-2*DBLE(G)*DBLE(ECPT(13))*A(4*I-3)
      A(42)=.25D-2*DBLE(G)*DBLE(ECPT(13))*A(4*I-2)
      A(43)=.25D-2*DBLE(G)*DBLE(ECPT(13))*A(4*I-1)
      A(44)=.25D-2*DBLE(G)*DBLE(ECPT(13))*A(4*I  )
      CALL GMMATD(A(214),2,3,1,A(41),2,2,0,A(51))
      CALL GMMATD(A(51),3,2,0,A(214),2,3,0,A(61))
      J=4*I+12
      CALL TRANSD(ECPT(J),A(71))
      CALL GMMATD(A(71),3,3,1,A(61),3,3,0,A(81))
      J=4*NOPVT+12
      CALL TRANSD(ECPT(J),A(91))
      CALL GMMATD(A(81),3,3,0,A(91),3,3,0,A(101))
      KE( 1)=A(101)
      KE( 2)=A(104)
      KE( 3)=A(107)
      KE( 7)=A(102)
      KE( 8)=A(105)
      KE( 9)=A(108)
      KE(13)=A(103)
      KE(14)=A(106)
      KE(15)=A(109)
      CALL SMA1B(KE,ECPT(I+1),-1,IFKGG,0.0D0)
      IF (IOPT4) 320,400,320
  320 IF (GSUBE) 330,400,330
  330 K4GGSW=1
      A(111)=GSUBE
      CALL SMA1B(KE,ECPT(I+1),-1,IF4GG,A(111))
  400 CONTINUE
      RETURN
C*****
C HEAT FORMULATION. FIRST GET MATERIAL PROPERTY -K- FROM HMAT ROUTINE
C*****
  500 CALL HMAT(IECPT(1),RZ(1))
C
C VARIABLE FK IN LABELED COMMON BLOCK HMTOUT CONTAINS THE MATERIAL
C THERMAL CONDUCTIVITY.
C
C THE THERMAL PROPERTIES ARE NOT YET DEFINED FOR THE 10-NODE CRAKED
C ELEMENT.
C
      WRITE(NO,501) IECPT(1)
  501 FORMAT(50H0***USER WARNING MESSAGE ----, THERMAL PROPERTIES ,
     1       12HFOR ELEMENT ,I8,18H ARE NOT INCLUDED.)
      RETURN
C
C OUTPUT FATAL ERROR MESSAGES
C
 2010 CALL  MESAGE(-30,34,IECPT(1))
 2020 CALL  MESAGE(-30,31,IECPT(1))
      RETURN
      END
```

```
      SUBROUTINE KOMK1A(A)
      DOUBLE PRECISION B(45),A(1)
      DATA (B(I),I=1,45) /
     * .25383561957813D+06,-.98856075210379D+05,-.57600000000000D+06,
     *-.85996816085674D+05,-.10096257071230D+06, .15459507520659D+05,
     *-.26999999998253D+01,-.63836548779353D+05,-.91795452131392D+05,
     * .20480000000000D+06, .56252632760132D+06,-.12478018618367D-10,
     *-.37124809795006D+06,-.20479904000000D+06,-.16776262913973D+06,
     * .16818724191125D-09,-.20712896652826D+06, .27547041373293D+07,
     * .53494133527037D+06, .39666666664398D+01,-.33150365061498D+06,
     *-.60755026157961D+06, .17108397973991D+05,-.72414068568232D+06,
     * .81920064000000D+06, .12028394569211D+07, .29095037490378D-09,
     *-.15058161263357D+07,-.98304341332960D+06,-.73726359701372D+06,
     * .35512344751464D+07, .16787277958873D+07,-.50177231998438D+06,
     *-.15462743847720D+07,-.18019692857643D+07, .30310336000056D+07,
     * .36196337612239D+07, .63933569549590D-09,-.42159867293081D+07,
     * .94687770306281D+07, .58304429249355D+07,-.51216000000515D+02,
     * .10111215664795D+08, .11038321364003D+08, .25498829464971D+08/
      DO 5 I=1,45
    5 A(I)=B(I)
      RETURN
      END
```

```
      SUBROUTINE KOMK2A(A)
      DOUBLE PRECISION B(45),A(1)
      DATA (B(I),I=1,45) /
     *-.40613699132501D+06, .24522101908062D+06, .11520000000000D+07,
     * .10942906659240D+06, .12406994960966D-09,-.20286515128225D+05,
     * .36719999998764D+01,-.39885896735624D+05,-.24478028678264D+06,
     *-.20480000000000D+06,-.84122586184631D+06, .48365755844770D-11,
     * .19253626755961D+06,-.11200000005684D+01,-.14640838218524D+06,
     *-.31239544284744D-09, .28641576772196D+06,-.44075266197269D+07,
     *-.44769557453282D+06,-.41999999997471D+01, .82767339556603D+05,
     *-.20474261789552D-08, .25004532853863D+06, .14482562285349D+07,
     *-.27306752000000D+06,-.62811409819395D+06,-.29022295677805D-09,
     * .44525453192187D+06, .21845631999580D+06, .51624824549484D+05,
     *-.21853750616285D+07,-.36223237053838D+06, .10035311999346D+07,
     * .74489142797886D+06,-.27470629406205D-09,-.32767680000420D+06,
     *-.47646906603319D+06,-.48818371567449D-09, .21218622357182D+06,
     *-.26120764222422D+07,-.12689385071979D+07, .51744000000977D+02,
     *-.11234413590513D+07,-.11301034786502D+07,-.38488799192592D+07/
      DO 5 I=1,45
    5 A(I)=B(I)
      RETURN
      END
```

```
      SUBROUTINE KDMK16(A)
      DOUBLE PRECISION B(45),A(1)
      DATA (B(I),I=1,45) /
     * .5076712391562730+05, .0000000000000000+00,-.3840000000000000+05,
     *-.6590407489728700+05,-.6429084575649950+05,-.2460312674884610+05,
     *-.1140000000115660+01,-.2115168457420780+05,-.1103359652658510+06,
     * .0000000000000000+00, .0000000000000000+00, .0000000000000000+00,
     * .0000000000000000+00, .0000000000000000+00, .0000000000000000+00,
     * .0000000000000000+00, .0000000000000000+00, .3746397626767940+06,
     * .2250103437302650+06,-.9800000000170520+00,-.3807583684718480+06,
     *-.4586822046814770+06,-.1738695314925000+06, .9216335999593120+05,
     * .2730662400000000+06, .3182116921499870+06,-.1865322373795770-09,
     *-.1864025920430740+06,-.1092241066704000+06, .2636146322054930+06,
     * .1655756105366550+07, .1128304709167690+07, .1194695999952820+06,
     *-.1411872129116880+07,-.1512657024466510+07, .1556481279997200+07,
     * .1735942552399300+07,-.5427155659555250-11,-.1210888403375450+07,
     * .5744149539653060+07, .3893367388086150+07, .3005333333469370+02,
     * .5617376548590090+07, .5946105832285730+07, .1733984021461180+08/
      DO 5 I=1,45
    5 A(I)=B(I)
      RETURN
      END
```

```
      SUBROUTINE KDMK28(A)
      DOUBLE PRECISION B(45),A(1)
      DATA (B(I),I=1,45) /
     *-.4512633236944640+05, .0000000000000000+00, .7680000000000000+05,
     * .7566714383713080+05,-.1151567730622000-10,-.4274684423896970+05,
     * .1320000000072830+01, .5977677466443160+05, .5192309113571320+05,
     * .0000000000000000+00, .0000000000000000+00, .0000000000000000+00,
     * .0000000000000000+00, .0000000000000000+00, .0000000000000000+00,
     * .0000000000000000+00, .0000000000000000+00,-.1763010647890790+06,
     *-.1885576514471950+06, .1000000000029760+01, .1211317478458460+06,
     * .3001590087592370-09,-.1930122911198000+06,-.1843235199953630+06,
     *-.2730662400000000+06,-.1529941155979200+06, .1815101361551100-09,
     *-.3615978080822000+05,-.2184563199956000+06,-.2618083205588010+06,
     *-.4013954194828010+06,-.4590168805832420+06,-.2389359999960630+06,
     *-.8160862648341400+04, .4621174554131350-09,-.8192019199965000+06,
     *-.6084748649690190+06, .1733866746196370-10, .3131324300597800+06,
     *-.8061964266179740+06,-.5202633757062150+06,-.3024000000110870+02,
     *-.1123489645718020+07,-.9880312845300250+06,-.1558637322651740+07/
      DO 5 I=1,45
    5 A(I)=B(I)
      RETURN
      END
```

```
      SUBROUTINE MDUM1
C *****
C THIS SUBROUTINE CALCULATES THE LUMPED-MASS MATRIX FOR THE 10-NODE
  CRACK ELEMENT
C *****
      DOUBLE PRECISION X,Y,Z,AREA,DMASS,AMASS,MASS
      DIMENSION IECPT(1)
      COMMON /SMA2IO/ DUMMY1(10),IFMGG,DUMMY2(25)
      COMMON /SMA2CL/ DUMMY3(2),NPVT
      COMMON /SMA2DP/ X,Y,Z,AREA,DMASS,AMASS(10),MASS(36),NOPVT
      COMMON /SMA2ET/ ECPT(56),DUMMY4(44)
      COMMON /MATIN / MATIDC,MATFLG,ELTEMP,DUMMY5(6)
      COMMON /MATOUT/ RHO
      EQUIVALENCE (ECPT(1),IECPT(1))
      MATIDC=IECPT(12)
      MATFLG=4
      ELTEMP=ECPT(56)
      CALL MAT(IECPT(1))
      X=DBLE(ECPT(37))-DBLE(ECPT(17))
      Y=DBLE(ECPT(38))-DBLE(ECPT(18))
      Z=DBLE(ECPT(39))-DBLE(ECPT(19))
      AREA=4.0D0*(X*X+Y*Y+Z*Z)
      DMASS=AREA*(DBLE(RHO)*DBLE(ECPT(13))+DBLE(ECPT(14)))
      AMASS( 1)=5.0*DMASS/9.0
      DO 10 I=2,10
      AMASS( I)=DMASS/18.0
   10 CONTINUE
      NOPVT=0
      DO 20 I=2,11
   20 IF(IECPT(I).EQ.NPVT)NOPVT=I-1
      IF(NOPVT.EQ.0)CALL MESAGE(-30,34,IECPT(1))
      DO 30 I=1,36
   30 MASS(I)=0.0D0
      MASS(1)=AMASS(NOPVT)
      MASS(8)=AMASS(NOPVT)
      MASS(15)=AMASS(NOPVT)
      CALL SMAB2(MASS(1),NPVT,-1,IFMGG,0.0D0)
      RETURN
      END
```

```
      SUBROUTINE SDUM11
C*****
C THIS ROUTINE IS PHASE 1 OF STRESS DATA RECOVERY FOR THE 10 NODE
C CRACK ELEMENT
C*****
C
      DIMENSION IECPT(1)
      COMMON /SDR2X5/ ECPT(56),INDEX(44)
     1, IELID     ,ISILNO(10),S1(30)     ,S2(30)     ,SIGVEC(29)
     2, FORVEC(25)
C
C
C
C
      COMMON /SDR2X6/ A(10,10),B(10,10),C(10),ISING,DETERM
     1, SIGMA,PI,FR2,UR,UT,DELTA,A1(20),A2(20),TL(6),TG(9),X(6),Y(6)
C
      COMMON /MATOUT/ E,G,ANU,RHO,ALPHA,TSUBO,GSUBE,SIGT,SIGC,SIGS
C
      COMMON /MATIN / MATIDC,MATFLG,ELTEMP,STRESS,SINTH,COSTH
C
      EQUIVALENCE (IECPT(1),ECPT(1))
C
      DATA PI,FR2/3.14159265,1.18920712/
C
C CALL MAT TO GET MATERIAL PROPERTIES
C
      MATIDC=IECPT(12)
      MATFLG=1
      ELTEMP=ECPT(56)
      CALL MAT(IECPT(1))
      IF (IECPT(15).EQ.1) SIGMA=ANU/(1.0+ANU)
      IF (IECPT(15).EQ.2) SIGMA=ANU
C
C CALCULATE 11 PARTITION OF D MATRIX AND INVERT
C
      DELTA=SQRT(SQRT((ECPT(37)-ECPT(17))**2+(ECPT(38)-ECPT(18))**2+
     1                  (ECPT(39)-ECPT(19))**2))
      DO 10 J=1,10,2
      DO 10 N=1,9
      I=(J-1)/2+6
      UR=-(N+2)*COS((N+2)*(I-6)*PI/8.0)-((N+2.)/(N+2*(-1)**N))*(6.0-8.0*
     1    SIGMA-N)*COS((N-2)*(I-6)*PI/8.0)
      UT= (N+2)*SIN((N+2)*(I-6)*PI/8.0)-((N+2.)/(N+2*(-1)**N))*(6.0-8.0*
     1    SIGMA+N)*SIN((N-2)*(I-6)*PI/8.0)
      IF (I-(I/2)*2) 4,5,4
    4 UR=(FR2**N)*UR
      UT=(FR2**N)*UT
    5 UR=(DELTA**N)*UR
      UT=(DELTA**N)*UT
      A(J  ,N)=UR*COS((I-6)*PI/4.0)-UT*SIN((I-6)*PI/4.0)
   10 A(J+1,N)=UR*SIN((I-6)*PI/4.0)+UT*COS((I-6)*PI/4.0)
      DO 20 I=1,9
      A(2,I)=A(1,I)
   20 A(1,I)=0.0
      DO 30 I=1,10,2
      A(I  ,10)=4.0**6
   30 A(I+1,10)=0.0
      A(2,10)=4.0**6
      CALL INVERS(10,A,10,C,0,DETERM,ISING,INDEX)
      IF (ISING.NE.1) CALL MESAGE(-30,26,IECPT(1))
```

```
      DO 40 J=1,10,2
      DO 40 N=1,9
      I=(J-1)/2+6
      UR=(N+2)*((N-2+(-1)**N)/(N+2.))*SIN((N+2)*(I-6)*PI/8.0)+(6.0-8.0*
     1   SIGMA-N)*SIN((N-2)*(I-6)*PI/8.0)
      UT=(N+2)*((N-2+(-1)**N)/(N+2.))*COS((N+2)*(I-6)*PI/8.0)-(6.0-8.0*
     1   SIGMA+N)*COS((N-2)*(I-6)*PI/8.0)
      IF (I-(I/2)*2) 44,45,44
   44 UR=(FR2**N)*UR
      UT=(FR2**N)*UT
   45 UR=(DELTA**N)*UR
      UT=(DELTA**N)*UT
      B(J  ,N)=UR*COS((I-6)*PI/4.0)-UT*SIN((I-6)*PI/4.0)
   40 B(J+1,N)=UR*SIN((I-6)*PI/4.0)+UT*COS((I-6)*PI/4.0)
      DO 50 I=1,9
   50 B(1,I)=0.0
      DO 60 I=1,10,2
      B(I  ,2)=0.0
   60 B(I+1,2)=4.0*G
      B(1,2)=4.0*G
      B( 1,10)= 0.0
      B( 2,10)= 4.0*G
      B( 3,10)=-4.0*G
      B( 4,10)= 4.0*G
      B( 5,10)=-4.0*G
      B( 6,10)= 0.0
      B( 7,10)=-4.0*G
      B( 8,10)=-4.0*G
      B( 9,10)= 0.0
      B(10,10)=-4.0*G
      CALL INVERS(10,B,10,C,0,DETERM,ISING,INDEX)
      IF (ISING.NE.1) CALL  MESAGE(-30,26,IECPT(1))
C
C EXTRACT ROWS FOR FIRST SYMMETRIC AND ANTI-SYMMETRIC TERM
C
      A1(1)=A(1,1)*4.0*G
      A1(2)=0.0
      A2(1)=0.0
      A2(2)=B(1,1)*4.0*G
      DO 70 I=3,10,2
      A1(I)= A(1,-I+12)*2.0*G
   70 A2(I)=-B(1,-I+12)*2.0*G
      DO 80 I=4,10,2
      A1(I)=-A(1,-I+14)*2.0*G
   80 A2(I)= B(1,-I+14)*2.0*G
      A1(11)=A(1,2)*4.0*G
      A1(12)=0.0
      A2(11)=0.0
      A2(12)=B(1,2)*4.0*G
      DO 90 I=3,10
      A1(I+10)=A(1,I)*2.0*G
   90 A2(I+10)=B(1,I)*2.0*G
C
C DERIVE KI AND KII RECOVERY FACTORS
C
      DELTA=DELTA**2
      TL(1)=(ECPT(37)-ECPT(17))/DELTA
      TL(2)=(ECPT(38)-ECPT(18))/DELTA
```

```
      TL(3)=(ECPT(39)-ECPT(19))/DELTA
      TL(4)=(ECPT(45)-ECPT(17))/DELTA
      TL(5)=(ECPT(46)-ECPT(18))/DELTA
      TL(6)=(ECPT(47)-ECPT(19))/DELTA
      J1=1
      J2=1
      DO 100 I=1,10
      CALL GMMATS(A1(J1),1,2,0,TL(1),2,3,0,X(1))
      CALL GMMATS(A2(J1),1,2,0,TL(1),2,3,0,Y(1))
      J=4*I+12
      CALL TRANSS (IECPT(J),TG)
      CALL GMMATS(X(1),1,3,0,TG(1),3,3,0,S1(J2))
      CALL GMMATS(Y(1),1,3,0,TG(1),3,3,0,S2(J2))
      J1=J1+2
  100 J2=J2+3
      DO 110 I=1,30
      S1(I)=7.51988982*S1(I)
  110 S2(I)=2.50662827*S2(I)
C
C STORE ELEMENT ID AND SIL NUMBERS
C
      IELID=IECPT(1)
      DO 120 I=1,10
  120 ISILNO(I)=IECPT(I+1)
      RETURN
      END
```

```
      SUBROUTINE SDUM12
C*****
C THIS ROUTINE IS PHASE 2 OF STRESS DATA RECOVERY FOR THE 10 NODE
C CRACK ELEMENT
C*****
C
      COMMON /SDR2XX/ ZZ(1)
C
      COMMON /SDR2X4/ DUMMY(35),IVEC
C
      COMMON /SDR2X7/ IELID,ISILNO(10),S1(30),S2(30),DUMMY1(29)
     1,               JSELID,STRES(9),DUMMY2(90)
     2,               JFELID,FORCE(9),DUMMY3(15)
C
      COMMON /SDR2X8/ SUMK1(10),SUMK2(10),IUTA,IUTB
C
      IDISP=IVEC-1
      IUTB=-2
      DO 10 I=1,10
      IUTA=IDISP+ISILNO(I)
      IUTB=IUTB+3
      CALL GMMATS(S1(IUTB),1,3,0,ZZ(IUTA),3,1,0,SUMK1(I))
   10 CALL GMMATS(S2(IUTB),1,3,0,ZZ(IUTA),3,1,0,SUMK2(I))
      DO 20 I=1,9
   20 STRES(I)=0.0
      DO 30 I=1,10
      STRES(1)=STRES(1)+SUMK1(I)
   30 STRES(2)=STRES(2)+SUMK2(I)
      DO 40 I=1,9
   40 FORCE(I)=STRES(I)
      JSELID=IELID
      JFELID=IELID
      RETURN
      END
```

INITIAL DISTRIBUTION

Hq USAF/SAMI	1
Hq USAFE/DOQ	1
Hq PACAF/DOOFQ	3
Hq TAC/DRA	1
ASD/ENFEA	2
AUL/LSE 71-249	1
OO/ALC/MMWMP	2
AFIS/INT	1
DDC-TC	2
AFATL/DLODL	2
AFATL/DL	1
AFATL/DLD	1
AFATL/DLY	1
ASD/XRP	1
US Army TRADOC Sys Ana Act/ ATAA-SL	1
COMIPAC/I-232	1
School of Engineering Science and Mechanics/Georgia Inst of Tech	15
Off of Engineering Research/ Oklahoma State University	1
AMSAA/DRXSY-J	1
Inst of Fracture and Solid Mech/ Lehigh University	1
Systems, Science and Software	1
AFFDL/FBR	1
AFFDL/FBR(ASIAC)	1
NSWC/DL	1
PACMISTESTCEN/Code 1245	1
BRL-BLV	1
NWC/Code 3261	1
Dept of Engineering Sciences/ University of Florida	1
Univ of Florida Grad Center	1
AFATL/DLYV	10
TAC/INAT	1
AFFDL/FES	1
NWC/Code 318	2

(The reverse of this page is blank)