NASA Contract Report CR-182236

The MHOST Finite Element Program:
3-D Inelastic Analysis Methods for Hot Section Components
Volume III - Systems' Manual

Shohei Nakazawa
The Finite Element Factory
Sunnyvale, California

July 1989

Prepared for
Lewis Research Center
Under Contract NAS3 - 23697

NASA
National Aeronautics and
Space Administration
This document discusses the internal structure of the MHOST finite element program designed for three-dimensional inelastic analysis of gas turbine hot section components. This computer code is the first implementation of the mixed iterative solution strategy for improved efficiency and accuracy over the conventional finite element method.

This document covers the control structure of the program, the data storage scheme and the memory allocation procedure and the file handling facilities including the read/write sequences.
This document discusses the internal structure of the MHOST finite element program designed for three-dimensional inelastic analysis of gas turbine engine hot section components. The computer code is the first implementation of the mixed iterative solution strategy for improved efficiency and accuracy over the conventional finite element method. The formulation of the mixed iterative solution method is an original development under the HOST project (contract NAS3-23697) and detailed documentation is available in the MHOST Theoretical Manual and other publications. The computer program has been written, tested and maintained at MARC Analysis Research Corporation, Advanced Project Group as a subcontractor to the United Technology Pratt and Whitney Aircraft.

The complete computer program consists of about 450 subroutines and a total of 47,000 lines of Fortran 77 statements. The current version 4.2 is no longer compatible with the ANSI Fortran 66 standard.

This document covers:

(i) the control structure of the program;

(ii) the data storage scheme and the memory allocation procedure;

(iii) the file handling facilities including the read/write sequences.

A brief note on the control variables in the labelled common blocks is given in the control structure section. Pointers for the working arrays in the common block are described in the second section of this document. The files used internally and those produced to communicate with other softwares (such as the graphic post-processing systems) are discussed in detail in the last section.

The appendix includes the brief description of each subroutine in conjunction with the names of common block referenced therein.

The MHOST code has been developed on PRIME 9955 at MARC running under Primos operating system (Rev.19.4.2 of F77 compiler has been used) and recently ported over to Alliant FX/8 running under unix operating system. This version is portable to any other unix-based computers with minimum amount of the conversion work. Versions are available on IBM mainframes with VS-Fortran Compiler (tested at United Technology Pratt and Whitney Aircraft) and CRAY X-MP COS using CFT compiler (the installation at NASA Lewis Research Center).
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0.1 Program Architecture</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0.2 Rules and Conventions</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>0.3 Subroutines and Common Blocks</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>0.4 System Dependency</td>
<td>10</td>
</tr>
<tr>
<td>1.0</td>
<td>CONTROL STRUCTURE</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>1.1 Overview</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>1.2 Execution Supervisor and Analysis Drivers</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>1.3 Control Variables</td>
<td>29</td>
</tr>
<tr>
<td>2.0</td>
<td>DATA STORAGE SCHEME</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td>2.1 Overview</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td>2.2 Memory Allocation Subprogram</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>2.3 Counters and Pointers</td>
<td>96</td>
</tr>
<tr>
<td>3.0</td>
<td>THE FILE SYSTEM</td>
<td>104</td>
</tr>
<tr>
<td></td>
<td>3.1 Overview</td>
<td>104</td>
</tr>
<tr>
<td></td>
<td>3.2 User Interface</td>
<td>105</td>
</tr>
<tr>
<td></td>
<td>3.3 Restart File</td>
<td>136</td>
</tr>
<tr>
<td></td>
<td>3.4 Post-Processing Data File</td>
<td>149</td>
</tr>
<tr>
<td></td>
<td>APPENDIX</td>
<td>A-1</td>
</tr>
<tr>
<td></td>
<td>Subroutines</td>
<td></td>
</tr>
</tbody>
</table>

(ii)
0.0 INTRODUCTION

The MHOST program has evolved from a small finite element code for testing new and innovative ideas into a versatile package usable as a research and development tool in solid and structural mechanics. Several code developers have worked on this program at various times over four years; more than 47,000 lines of extensively commented Fortran 77 code now make up the program. This document describes the internal architecture of the code. Particular emphasis is placed here on the facilities by which users can modify the code to incorporate new ideas such as different elements or constitutive equations.

The concept of three libraries plays a central role in the MHOST code. These libraries are:

The element library. Almost all the element specific operations are coded in this library with a common interface subroutine. Note that the MHOST code is written based on the mixed iterative solution concept and the element library is accessed significantly more often than in the case for conventional displacement method codes.

The material library. All the constitutive equations built into the MHOST program are accessed through an interface routine for the material library. The nodal evaluation of the constitutive equations in the mixed iterative solution process enables this operation to occur independently of the loop structure for the element formulation.

The solution algorithm library. The MHOST code uses a number of modern iterative solution algorithms and two types of solvers for a linearized system of algebraic equations. These options are accessed directly from the finite element driver routines; from a programming point of view, the routines in the solution algorithm library are not as clearly identifiable as other library routines.

0.1 Program Architecture

The MHOST program is designed and constructed to perform inelastic finite element computations in a reliable manner. A number of analysis driver routines are coded to perform individual clearly defined tasks. Inter-relations between analysis tasks are controlled by the execution supervisor subprogram and the utilities attached to it. All three libraries discussed briefly in the previous section are executed by each analysis driver routine. The program architecture of the MHOST program is illustrated in Figure 1.
Figure 1 Architecture of MHOST Program Version 4.2
The concept of multiple-drivers prevents users from combining unreasonable options, and avoids program troubles. Also this coding strategy allows code developers to add new analysis capabilities without affecting existing functions of the MHOST program.

The nonlinear finite element computations are arranged in a nested loop structure. The outer loops which control the algorithmic operations are coded in the analysis driver routines, while the inner loops involving incremental data addressing are coded in the element assembly submodules, a level below the analysis drivers.

A unique feature of MHOST is that all information is stored at nodes. The finite element nodal database resides in the lower address of working space in the blank common block. The storage allocation for this nodal database is virtually independent of analysis and solution algorithms and takes place before reading the finite element model data.

The working space required to perform the solution is allocated in the higher address portion of the same working space. The allocation of this working space is analysis dependent and takes place inside of the finite element driver routine.

0.2 Rules and Conventions

Coding rules are established to maintain the readability and maintainability of the MHOST program.

Subroutine names are selected as close as possible to a plain English word representing the operation performed by the subroutine. Because of limitations set by the old Fortran standard of six characters, some subroutine names can be cryptic.

A special rule has been implemented for subroutines in the element library:

- **SnnmmN**  Shape functions for elements consisting of mm-nodal points in nn-geometrical configuration.
- **DnnmmN**  Cartesian derivatives for elements consisting of mm-nodal points in nn-geometrical configuration.

<table>
<thead>
<tr>
<th>nn</th>
<th>Geometry</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D</td>
<td>Two dimensional</td>
</tr>
<tr>
<td>AX</td>
<td>Axi-symmetric</td>
</tr>
</tbody>
</table>
Some of the other rules are:

- The subroutine names with string IN are used for data read from the main input channel.

- The string INIT is used to indicate subprograms involving memory allocation, except for INITST which generates the initial stress terms in the stiffness equations.

- The string SUB appears in subroutine names for subroutines involving the subelement scheme and the local-global analysis.

- The subroutine names starting with U indicate user subroutines.

Variable names are chosen as close as possible to a plain English word representing the information contained in the variables. Because of limitations set by the old Fortran standard of six characters, some variable names can be often cryptic.

Note that the implicit data types of Fortran specification are assumed. Variable names starting with I, J, K, L, M and N are integers except for those specifically declared as logical variables.

Some of the variable name conventions are:

- The string MAX is used for the maximum number of entries in adjustable dimensioned arrays.

- The string NEL is used for the actual number of entries in adjustable dimensioned arrays containing element related information.

Statement Numbers

Four-digit statement numbers are used for better visibility of the Fortran program listing. Codes which are ported from other software do not necessarily follow the conventions defined below.

The format statements referred from the read statements are labelled by statement numbers from 1000 to 1999. The format statements referred from the write statements are labelled by statement numbers from 2000 to 2999.
Ends of DO loops are labelled by statement numbers from 5000 to 5999. Statements to which control is transferred by the conditional GO TO statements are labelled by statement numbers from 8000 to 8999.

Efforts have been made to avoid arithmetic IF and GO TO statements in the MHOST code. Instead the block IF structure is implemented wherever possible. Note that this approach has made the code incompatible with Fortran 66 compilers.

0.3 Subroutines and Common Blocks

All the subroutines included in the MHOST program Version 4.2 are listed below in alphabetical order:

ACCLIN  ADAPTC  ADAPTD  ADAPTS
ADD    ADDBAN    ADDIAG    ADDINC
ADDINV  ADDPUL    ADDSMU    ANPLAS
ASMEVE  ASSEM1    ASSEM2    ASSEM3
ASSEME  ASSEM5    ATTRIB    BASUB
BANDER  BANNER    BASAXS    BASEIN
BASP5N  BASPST    BASOSL    BAXSYM
BEAMIN  BFGSLH    BFGSRH    BFGSW
BFLOAD  BLCK01    BMSTRE    ENDTRM
BODYIN  BOUND1    BOUND2    BOUND3
BOUNDN  BOUNFR    BOUNIN    BPSTRN
BPSTRE  BREAD     BSHDKQ    BSHELL
BSOLID  BTEBEAM    BUCKLE    BULKIN
BWRITE  CENMAS     CENT2D    CENT3D
CENTAX  CENTEM     CENTSH    CHARIN
CHCHAR  CHELM     CLENUP    CLOSEF
CNODEL  CNSMAS     CNSTNM    COLED
COMPDF  COMPIN     COMPRO    CONDSE
CONJUG  CONNIN     CONTIN    COORIN
COPY    COPYCH     COPYDS    COPYN
COPYSD  CORDTR     COROUT    CPXEX
CPXER2  CPXDIV     CPXEXEC   CPXFAC
CPXFOR  CPXMLUL    CPXREA    CPXRES
CPLAW   CRPSTN     CUTHIL    D2D04N
D2D09N  D3D08N    D3D27N    DAMPIN
DASHIN  DAT1      DAT2      DAT3
DAT5    DATBMS     DATCG1    DATER
DATIN1  DATIN2     DATIN3    DATOH1
DATOU1  DATOU4     DAX04N    DAX09N
DEM02N  DECINT     DECOMP    DECREA
| 3ST Version 4.2 | SYSTEMS' MANUAL | Page : 6 | MHOST Version 4.2 |
All the common blocks included in the MHOST program Version 4.2 are listed below in alphabetical order:

COMMON / ADDVAL / ISPRI, KSPRI, IDASH, KDASH, IMASS, KMASS
COMMON / ALGEM / ICREAD, IJPRNT, JPRNT, ICONSL, IPOSTF, ISCRAF,

1

SYSTEMS' MANUAL

Page : 7
COMMON / AUTOIN / CURPER,TOTPER,ARCLEN,ATOLER,BTOLER,CTOLER,
  JADAP,NCREEP,SCALE
COMMON / BODYFR / POINTS( 3, 2)
COMMON / BSECT / BSECT,KBSECT
COMMON / CONTRO / JEND, JITER, JTEMP, JPRINT, JP, JSUB,
  JINC, JREST, JSAVE, JREDIN, JAUTO, JPOST,
  JBACK, JOPTIM, JCREEP, JDIST, JCONST, JDYN,
  JNISO, JTHERM, JTRIG, JIDYN, JIRET, JTANGE,
  JHERM, JFORCE, JUTEMP, JUOF, JDIST, JUHOCK,
  JDERIV, JBIND, JINDEX, JSTATE, JINST, JJSTOP,
  JFRONT, JDIFOR, JIFORB, JITEST, JIDISP, JFFRGS,
  IFSCNT, IFLINE, IFPRINT, IFCOMPS, IFCONJ, IFCONT,
  IFBODY, IFGRAV, IPHEAV, JDEP, JSTOR, JSTATE,
  JFRONT, JINST, JINDEX, JFINT, JFRG, JFOLH,
  JIKSLL, JFRES, JCDUM2, JCDUM3
COMMON / COMPND / NXSTAT, NXSOLV, NXINST, NXMODL, NXECKL, NXSUPR,
  NXREQN, NXDUM1, NXDUM2, NXDUM3, NXDUM4
COMMON / COUNT / LININC, LININT, NOECCHO
COMMON / CTITLE / TITLE ( 20), IDAT ( 5), IDATE2, ICIFX,
  IPOSH, IPRES, JCDUM3, JCDUM2
COMMON / DAMP / DAMP(3)
COMMON / EIGEN / IEQVC, IGNMS, IQMEG, IMOENO, IDYNMD, IDRT2,
  IPITR, IPTBR, IPTVD, IOMD , IQMEG
COMMON / MODSUP / IFORO, IMISO, IMVELO, IFPOR, IMMIS, IMVEL1
COMMON / ELEMP / IC, ISEL, IFL , JLAN, IPAD, JASEM,
  JSMU, JCR, JREL09, JREL10, JREL11, JREL12
COMMON / ELTYP / NELCRD, NELNFR, NELNOD, NELSTR, NELCHR, NELFR,
  NELINT, NELV, NEMLAY, NDI, NSHEAR, NELCMP
COMMON / ERRORS / IERR
COMMON / FREE / IA ( 80), IBEGIN ( 16), ILENG ( 16),
  NSTRIN, IS, ICOL, NEW
LOGICAL / NEW
COMMON / HARMON / OMEG1, HARM1, OMEGB, OMEG3, OMEGB, OMEG3
COMMON / INCCON / FACTOR, INCFLG(21)
COMMON / MACHT / IDP
COMMON / MAXIMA / MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
  MAXLAY, MAXINT, MAXRKG, MAXNL, NSUMAX, MAXCM,
  MAXSP, MAXCMR, MAXTEM, MAXELM, MAXKM, MAXDLM,
  MAXFN, MAXBET, MAXVAR, MAXSET, MAXEAN, MAXORD,
  MAXO25, MAXO26, MAXO27, MAXO28, MAXO29, MAXO30
COMMON / LOUBIN / JLOB, JINTER, JEXTRA, JWRITE, JSUBRE, JISTRN,
  JCITER, JHRGLS, JGRAM, JLOB03, JLOB04, JLOB05

SYSTEMS' MANUAL
MHOST Version 4.2
Page : 8
COMMON / PAGCNT / NPAGE1,NPAGE2
COMMON / PARAM / NTYPE,NELEM,NNODE,NBC,NTIE,NMAX,
                   NITRAN,NITRAC,NPD,NEAN,FXPER,NSUB,
                   NPRINT,NPOST,NSBC,NDUP,NSIZE,NESECT,
                   NSHIFT,NSFBS,NSMRS,NSPRI,NMSS,NSDASH,
                   NDYNMD,NSPDEC,NPDCS,NIHARM,NBASE,NINC,
                   NSMCH,NDICON,NNOUT,NSPDT,NSHOT,NSPLT,
                   NSPDA,NSPER,NSPDD,NSPLH,NSPLS,NSZP,
                   NSPLR,NSPERL,NSPFR,NSPLB,NSPDD,NSPLN,
                   NSPLT,NSPERL,NSPFL,NSPLR,NSPERL,NSPFR,
                   NSPLB,NSPDD,NSPLN,NSPERL,NSPFL,NSPLR,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
                   NSPERL,NSPFL,NSPERL,NSPFL,NSPERL,NSPFL,
Most of the variables and arrays in the common blocks are used to store control variables and counters. The exceptions are START1 to START8 which contain all the pointer values for working storage allocation.

0.4 System Dependency

The MHOST Version 4.2 program is written in Fortran 77, as closely as possible to ANSI specification. The code has been developed on the PRIME 9955 under PRIMOS F77 Rev. 19.4.2.
In the original version, integer and single precision real word lengths of 32 bits are assumed and double precision is specified by using

```
IMPLICIT REAL*8 (A-H, O-Z)
```

for all real variables except for working storage arrays. The ratio of integer and double precision word length is stored in the variable IDP in common block MACHIN. The value is set at 2 in the main program of the original version.

To generate versions for 64 bits/word machines such as CRAYs and CYBERs, the value of IDP must be set 1 (one) in the main program.

The file opening and error handling procedures are system dependent. The subroutine INTINT is called from the main program to perform these functions. An example of this routine for the PRIME version is included here.

C SUBROUTINE = INTINT CALLED FROM THE MAIN PROGRAM
SUBROUTINE INTINT

```
C **********************************************************************
C IMPLICIT REAL*8 ( A-H , O-Z )
REAL*4 RWORK
C **********************************************************************
C COMMON / VRIDSK / ISETUP,MAVAIL,LENREC,NUMREC,LENBLK,NUMBLK,
1 IVPAGE,NVPAGE,IVSTRI,IKRO ,ILCOL ,IPIVCO,
2 IHEDER,IFNRH,IPIVOT,IPIVRO,IPVKOL,IVEND ,
3 ISRECD,IEREC
C CHARACTER*20 NAMEIN , NAMEPR , NAMEME , LOGFIL , NAMESC ,
1 NAME08 , NAME09
C --- DEFINE THE BUFFER SIZE FOR SCRATCH FILE ------------------------
C MAVAIL = 3 000
NEUFFR = MAVAIL * 4 + 12
C **********************************************************************
C OPEN FILES : NAMES READ FROM A SPECIAL FILE 'T$NAMES'
```
On the PRIME a user will have to prepare a file called T$FILES in the working directory containing names of all the files to be opened as Fortran I/O units.

On other systems, this operation can be performed either by issuing commands at the operating system level or by duplicating the PRIME version approach directly in the MHOST program.

Note it is recommended that on IBM machines the error and warning message print-outs be disabled for double precision floating point operations. This is due to the fact that the solution of an algebraic system of equations involves double precision variables with fictitiously large exponents.
The real time clock and calendar routine and the CPU clock routine both involve system calls and differ depending on the machinery and operating system. The organization of common block CTITLE in which the date and time are stored may need to be modified depending on how the system returns these values. The actual call to the system's clock and calendar is made through SUBROUTINE DATER. The following code is used to communicate with the PRIME/PRIMDS:

C SUBROUTINE DATER
   SUBROUTINE DATER( IDAT, IDAT4)
C
C DATE ROUTINE FOR THE PRIME
C
C * * * * * * * *
INTEGER*2 ARRAYS(15),NUM,IDAT(6)
DIMENSION IDAT4(2)
NUM=4
CALL TIMDAT(ARRAYS,NUM)
IDAT(1)=ARRAYS(1)
IDAT(2)=ARRAYS(1)
IDAT(3)=ARRAYS(2)
IDAT(4)=ARRAYS(2)
IDAT(5)=ARRAYS(3)
IDAT(6)=ARRAYS(3)
I0=ARRAYS(4)
I1=I0/60
I2=I0-60*I1
IDAT4(1)=I1
IDAT4(2)=I2
RETURN
END

On the CRAY, this operation is simplified to two system calls in SUBROUTINE HEAD as shown below:

C=SUBROUTINE= HEAD CALLED BY 'DATIN1'
SUBROUTINE HEAD
   1 (VERNO ,MONTH ,JDATE ,ILPRNT,ICONSL)
C
C ************************************************************************
C
C PICK UP TODAY'S DATE AND TIME AND THEN PRINT 'HOST' LOGO ON THE
C FIRST PAGE OF LISTING. NOTE THIS OCCURS EVEN WHEN VACANT DATA FILE
C IS ASSIGNED ( ON PRIME ).

SYSTEMS' MANUAL

MHOST Version 4.2
C
C ******************************************************************************
C
IMPLICIT REAL*8 ( A-H , O-Z )
C ***
COMMON / CTITLE / TITLE ( 20),IDAT ( 5),IDATE2,ICLOCK,
        1 IPCRAY
C
******************************************************************************
C
CALL CLOCK ( ICLOCK )
CALL DATE ( IDATE2 )
C
WRITE(ILPRNT,1000)
WRITE(ILPRNT,2000)
WRITE(ILPRNT,3000)
WRITE(ILPRNT,1100) IDATE2,ICLOCK,VERNO,MONTH,JDATE,TITLE
WRITE(ICONSL,1001)
WRITE(ICONSL,2000)
WRITE(ICONSL,3000)
C
*** THE FOLLOWING STATEMENT MAY CAUSE A WARNING MESSAGE WHEN EXECUTED
*** ON IBM CMS WITH 'RUNHOST' PROCEDURE FILE.
C
WRITE(ICONSL,1100) IDATE2,ICLOCK,VERNO,MONTH,JDATE,TITLE
WRITE(ICONSL,1002)
C
******************************************************************************
C
1000 FORMAT(1H1,20(/))
1001 FORMAT(20(/))
1002 FORMAT(/)
2000 FORMAT(
*2X,'MA' RC CC CC CCCCCCCC CCCCCCCC CCCCCCCC CC CCCCCCCC CC /
*2X,'MAZ' ZRC CC CC CCCCCCCC CCCCCCCC CCCCCCCC CC CC CC CC '/
*2X,'MAZZ' ZZZRC CC CC CC CC CC CC CC CC CC CC CC CC /
*2X,'MA ZZ ZZ RC CC CC CC CC CC CC /
*2X,'MA ZZZ RC CCCCCCCCC CC CC CCCCCCCC CC CC CC /
)
3000 FORMAT(
*2X,'MA' Z RC CCCCCCCCC CC CC CCCCCCCC CC CC '/
*2X,'MA' RC CC CC CC CC CC CC CC CC /
*2X,'MA' RC CC CC CC CC CC CC CC /
*2X,'MA' RC CC CCCCCCCC CCCCCCCC CC CC /
*2X,'MA' RC CC CC CCCCCCCC CCCCCCCC CC /
)
1100 FORMAT(/,2X,6HDATE: ,A8,3X,A8,//2X, SHE
Calls to the CPU clock for the execution time are also system dependent. SUBROUTINE TIME in the original PRIME version interfaces with the system's clock and returns the time in seconds as a double precision real variable.

SUBROUTINE TIMER
SUBROUTINE TIMER(CU)
C
C* * * * *
C
C CPTIME RUNTIME FOR PRIME
C
C* * * * *
C
IMPLICIT REAL*8 (A-H,O-Z)
INTEGER*2 I(15),NUM
NUM=15
CALL TIMDAT(I,NUM)
CU=I(7)+(I(8)/330.)
RETURN
END

On the CRAY, this operation is simplified and performed at a routine one level higher as:

C=SUBROUTINE=TIMOUT UTILITY SUBROUTINE
SUBROUTINE TIMOUT(IA1,IA2,IA3,IA4,IA5,IA6)
C
C************************************************************
C
C OBTAINS AND PRINTS CPU TIME
C
C************************************************************
C
IMPLICIT REAL*8 (A-H,O-Z)
C
DIMENSION NAME(6)
C
On the CRAY, this operation is simplified and performed at a routine one level higher as:
COMMON / ALGEM / IREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1 IJLPRNT, ISPRT, PI , LINE , LINE2
COMMON / CONTRO / JEND , JITER , JTEMP , JPRINT , JP , JSUB ,
1 JINC , JREST , JSAVE , JREDIM , JAUTO , JPOST ,
2 JBACK , JOPTIM , JCREEP , JPRES , JCONST , JDYN ,
3 NONISO , ITERM , ITRIG , IDYNM , JREROT , JTANGE ,
4 JTHRM , JFORCE , JUTEMP , JUOEF , JPRESS , JUHOOK ,
5 JDERIV , JUBOUN , IDSTOP , INISTR , JPLAST , JBAND ,
6 JFRONT , JDEFOR , JEMBED , JTEST , JDISP , IFBFGS ,
7 IFSCNT , IFLINE , IFPRINT , JCOMPS

COMMON / TIMLOC / CPU0

C
C *** GET 'DELTA' VALUES - CALL TO CARY LIBRARY SUBROUTINE
C
CALL SECOND ( CPU )
C
IF ( CPU0 .EQ. 0.D0 ) CPU0 = CPU
   CPU = CPU - CPU0
C
NAME(1) = IA1
NAME(2) = IA2
NAME(3) = IA3
NAME(4) = IA4
NAME(5) = IA5
NAME(6) = IA6
C
CALL LINES( 3 , 3 )
   WRITE(ICONSL,1010) NAME, JINC, JITER, CPU
IF(CPU.NE.0.D0) WRITE(ILPRNT,1000) NAME, JINC, JITER, CPU
C
1000 FORMAT//(//,2X,6A4,9HINCREMENT,I4,10H ITERATION,I3,8H CPTIME=,F9.3,
1 4H SEC)
1010 FORMAT(2X,6A4,9HINCREMENT,I4,10H ITERATION,I3,8H CPTIME=,F9.3,
1 4H SEC)
RETURN
END

To be able to obtain a reasonably accurate estimate of the CPU time, it is
recommended that these routines be modified for the particular system on
which the MHOST program is installed.
1.0 CONTROL STRUCTURE

In this chapter, a description of major subprograms is given. This is preceded by a brief note on the architecture of the MHOST program. The third section here is devoted to the definition of control variables appearing in the common blocks.

1.1 Overview

The architecture of the MHOST code is schematically shown in Figure 1. The execution supervisor routine (SUBROUTINE HOST) controls several analysis modules in a consistent manner. In Version 4.2 of the MHOST code, a pair of subroutines, LETCMD and RUMCMD, have been added to check the consistency of the Parameter Data and to generate internal flags for the selection of analysis modules. The intention here is to stop users from combining features of the program package not designed to be used together.

C=SUBROUTINE=HOST CALLED FROM 'HOST: MAIN' PROGRAM
SUBROUTINE HOST
   1 (RWORK, IWORK, ISIZE, VERSNO, MONTH, JDATE)
C
C **********************************************************************
C
IMPLICIT REAL*8 ( A-H, O-Z )
REAL*4 RWORK
C
C **********************************************************************
C
COMMON / CONTRO / JEND, JITER, JTEMP, JPRINT, JP, JSUB,
1 JINC, JREST, JSAVE, JREDIM, JAUTO, JPPOST,
2 JBACK, JOPTIM, JCREEP, JDIST, JCONST, JDYN,
3 NONISO, ITERM, ITRIG, IDYN, JREPO, JTANG,
4 JTERM, JFORCE, JTEMP, JUOEF, JDISTS, JUHOC,
5 JDERIV, JUBOUN, IDSTOP, IMSTR, JPLAST, JBAND,
6 JFRONT, JDEFOR, JEMBED, JTEST, JDISP, IFBGFS,
7 IPSCM, IFLINE, JFRNT, JCOMP, IPO, JCNJNG, JEIGEN,
8 IFBODY, IFGRAV, IFCENT, JDAMP, LDYN, ISTAT,
9 JFDSXX, JISTIF, JCNM, JFINIT, JLEAR, JFOLON,
+ JWKSLP, JPRE, JDUM, JCDUM3,
COMMON / ALGEM / ICREAD, ILPRNT, JLFRT, ICONSL, JPOST, ISCRAF,
1 IPILOT, IRSTR, ICREAD, IPVARS, IPSETS, IPFLX,
2 PI, LINE, LINE2
COMMON / ADDVAL / ISPRI, KSPRI, IDASH, KDASH, IMASS, KMmass
COMMON / TMARCH / DAX, DBETA, DGAMMA
COMMON / AUTOIN / CURPER, TUPPER, ARCLEN, ATOLER, BTOLER, CTOLER,
1 JADAP ,NCREEP,SCALE
COMMON / PERIOD / JPEROD( 2 ),IPDISP,IPFORC,INDISP,INFORC
COMMON / POWER / IEPH,IELSEM,IEPSMO,ISIGMD,ITN,M,IPBP,ISPP,ISFF,ISQO,ICQQ,ITN,M,
IPSF,IPSF
COMMON / PULSES / IPULSE,KPULSE,IPDTIM,IPDFOR
COMMON / EIGEN / IEGNMC,IGNMS,IMOEG,IMOENO,IDYNMD,ISTR21,
ITPBR,ITPVEV,IMDAM ,ICBEXC,
COMMON / ERRORS / IERR
COMMON / MODSUP / IMFOR0,INDIS0,IMVE0,IMFOR1,IMDIS1,IMVE1
COMMON / HARMON / OMEG ,KHARM ,OMEGB ,IBASE ,KBASE ,
ICNFOR,ICMFOR,ICMRES,ICALLP,ICBFPN,ICBEXC,
COMMON / LOUBIN / ILOUB,JINTER,JEXTRA,JWEIGH,JSUBRE,JSTRN,
JCITER, JHRGLS, JGRAM,LOUB03,LOUB04,LOUB05
COMMON / START5 / IRL,IREAC,IES,IBQM,ISRL,
IBITLC,ISRM,ILAST,ILRL,INDINC,IFORIN,
1 IP,P,IPAM,IMASMT,IDIAG,ΙUPTR,ΙCOLEPT,
3 IMASDI,IMASUP,IST521,IST522,IST523,IST524
COMMON / START4 / IDINC,IDTOT,IPORC,IRESDN,ΙIWNO,ISIGNO,
1 IEPSNO,IPSTRN,ΙCSTR,ΙSTRN,ΙSTRS,ΙSTRN,
2 IIPSTN,ΙCSTR,ΙΤΣΤΝ,ΙΣΤΣΝΟ,ΙΣΤΣΝΟ,ΙΣΤΣΝΟ,ΙΔΜΑΤ,
3 IDIMNO,ΙΕΩΣΤ,ΙΟΜΕΝΟ,ΙΟΜΕΝΟ,ΙΟΔΝΟ,ΙΟΔΝΟ,
5 IDYNV,ΙΔΥΝΑ,ΙΔΣΧ1,ΙΔΣΧ2,ΙΔΣΤΗ,ΙΣΩΣΛ,
6 IEQCSI,IPREF,ΙΔΣΧ3,ΙΥΙΛΟΔ,ΙΠΙΝΚ,ΙΔΡΟΟΤ,
7 ΙΣΤ443,ΙΣΤ444,ΙΣΤ445,ΙΣΤ446,ΙΣΤ447,ΙΣΤ448
COMMON / PARAM / ΝΤΥΡΕ,ΝΕΩΜ,ΝΝΟΔ,ΝΒΕ,ΝΜΑΧ,
1 ΝΤΡΑΝ,ΝΤΡΑΚ,ΝΦΩ,ΝΒΑΝΔ,ΝΕΚΤ,ΝSUB,
2 ΝPRINT,ΝΠΩΤ,ΝΒΕ,ΝΝΗ΢,ΝΒΣΟΤ,ΝΒΣΟΔ,ΝΒΣΟΣ,ΝΒΣΟΝ,ΝΒΣΟ,ΝΒΣΟ,ΝΙΝΗ,
3 NSHIFT,ΝΦΒΟΣ,ΝΓΜΡΝ,ΝΦΡΙΝ,ΝΜΑΣΧ,ΝΟΔΑΙ,
4 NDYNMD,ΝΣΒΝΟ,ΝΙΝΗΡ,ΗΝΗΡ,ΝΒΟΣ,ΝΙΝΗ,
5 NITER,ΝΦΠΣΤ,ΝΦΠΣΤ,ΝΦΠΣΤ,ΝΦΠΣΤ,ΝΝΟΜΟΔ,
6 NSUMZH,ΝΑΝΜΕ,ΝΜΟΝΙΤ,ΝΠΑΡ40,ΝΠΑΡ41,ΝΠΑΡ42,
7 ΝΠΑΡ43,ΝΠΑΡ44,ΝΠΑΡ45,ΝΠΑΡ46,ΝΠΑΡ47,ΝΠΑΡ48
COMMON / BSECT / IBSECT,JBSECT
COMMON / INCON / FACTOR,INCFLG(21)
COMMON / MACIN / IDP
COMMON / SUBSTR / NLVSUB( 10),NFSUB( 10)
COMMON / SUBELM / ISUBEL,ISUENP,ISUBPT,NSDATA,ISUBITY,IEMBED
COMMON / SHIPT / ISHIFT,KSHIFT,IFREQ,LFREQ,NOFFST,NFOUN
DATA IZ / 1H /
DATA MAXSUB / 10 /

C
C ******************************************

SYSTEMS' MANUAL
Page : 18
WRITE(ICONSL,2000) IZ
CALL TIMDUT('BE','GIN','EXEC','UTIO','N','')

*** JOB PARAMETER INPUT **********************************************

CALL DATINI

*** SET THE RUNSTREAM FLAGS FOR MULTIPLE OPTION EXECUTIONS **************

*** BULK FINITE ELEMENT DATA INPUT ***********************************
IPFLG = 1
IF( IDYNM .EQ. 1 .AND. JDYN .EQ. 2 ) IPFLG = 2

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

IF( JREST .EQ. 0 ) CALL BULKIN( RWORK, IWORK, ISIZE, IPFLG )

*** RESTART FILE INPUT *** NOT SUPPORTED IN VERSION 2.0 **************

IF( JREST .NE. 0 ) CALL RESTRT( RWORK, IWORK, ISIZE )

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

CALL FINITE ELEMENT SOLUTION DRIVER ROUTINES

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

DO 5000 ICC_PD = 1, D

CALL RUNCMD
1  (ICC_PD, ICOMPD, IFSTAT, IFSOLV, IFINIG, IFMODL, IFBCKL, IFSUPR, IFREQN,
   IFDUM1, IFDUM2, IFDUM3, IERR )

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

IF( IFSOLV .EQ. 1 ) THEN
   CALL STATIC( RWORK, IWORK, ISIZE )
ELSE IF( IFSTAT .EQ. 1 ) THEN
   CALL FRONTIS( RWORK, IWORK, ISIZE )
ELSE IF( IFMODL .EQ. 1 ) THEN
   CALL MODAL( RWORK, IWORK, ISIZE )
ELSE IF( IFSUPR .EQ. 1 ) THEN
   CALL SUPER( RWORK, IWORK, ISIZE )
ELSE IF( IFBCKL .EQ. 1 ) THEN
   CALL BUCKLE( RWORK, IWORK, ISIZE )
ELSE IF( IFREQN .EQ. 1 ) THEN
   CALL FREDCM( RWORK, IWORK, ISIZE )
ELSE IF( IFINIG .EQ. 1 ) THEN
   CALL DYNAMO( HWORK, I%DRK, ISIZE )

SYSTEMS' MANUAL
MHOSm Version 4.2

Page: 20
The analysis modules represent the control structure for the incremental iterative algorithms implemented in the MHOST code. The source code with comments is designed to serve as the schematic flow chart of the computational process.

The schematic flow of the element and nodal data manipulations is coded in the element assembly submodules identified by the string ASSEM in the subroutine names which are entered from analysis modules. The element assembly submodules perform operations independent of element types and constitutive models. The assumptions introduced in the mechanics aspects of the formulations are explicitly coded at this level. These submodules call the librarian routines for elements and constitutive models.

The element librarian subprogram (SUBROUTINE DERIV) generates quantities unique to the element used in an analysis. The MHOST code uses standard finite element matrix notation as in ZIENKIEWICZ (1977). The element specific information returned from the librarian subprogram is the strain-displacement array referred to as the B matrix in previous writeups for 1987.

The constitutive equation librarian subprogram (SUBROUTINE STRESS and BMSTRS) is designed to incorporate the nodal storage of stress and strain values. The STRESS subroutine sets up the loop over the points at which
constitutive equations are evaluated. The current implementation is a nested double loop with the outer loop being over the nodes and the inner loop over the integration layers through the thickness. Note that the conventional displacement method can be recovered by restructuring this loop in conjunction with a few minor modifications in the core allocation for stresses and strains.

The librarian subprogram utilizes the constitutive equation package (SUBROUTINE NODSTR) from which individual subprograms for initial strains, stress recovery and the material tangent are called. The librarian subprogram also controls the pre-integration of stresses, strains and the material tangent over thickness for the shell element.

Note that application of the constitutive equation represents one of the most costly operations in nonlinear finite element computations. An attempt is made in MHOST to optimize the execution of this process which occurs, for example, once every iteration during the recovery of the residual vector. At the beginning of each increment, this process may be executed to evaluate the material tangent and to determine the contribution of initial strain terms which are often necessary for proper displacement pre-conditioning.

Except for a small amount of information related to convergence of the iterative solution, all report generation is performed at the end of an increment. Optionally, post-processing and restart files are written at the end of user-specified increments. Generic reporting subprograms are called from analysis modules inside the increment loop.

1.2 Execution Supervisor and Analysis Drivers

This section describes major subprograms in the MHOST package. For the sake of clarity no listings are attached in this section. Also this section does not discuss the arguments and common blocks appearing in each subroutine.

Execution Supervisor

MAIN PROGRAM - declares work space in blank common as an integer array. Also defines system parameters which are machine independent. Then passes control to actual execution supervisor SUBROUTINE HOST.

SUBROUTINE HOST - controls the sequence of execution for the analysis modules. First, this routine executes the control parameter data reader SUBROUTINE DATINI and checks for consistency by entering SUBROUTINE LETCMD and RUNCMD. The current structure of the code allows certain combinations...
of two analysis modules to be executed sequentially. Analysis modules called from the execution supervisor are:

SUBROUTINE STATIC for a quasi-static incremental iterative solution.

SUBROUTINE DYNAMT for a transient time integration of the dynamic equilibrium equations in an incremental-iterative manner.

SUBROUTINE MODAL for an eigenvalue extraction in vibration mode analysis. This subsystem may be executed after quasi-static analysis for modal analysis of prestressed structures.

SUBROUTINE BUCKLE for an eigenvalue extraction in buckling load calculations. This subsystem is executable only after quasi-static analysis.

SUBROUTINE FRONTS for a quasi-static incremental iterative solution by the out-of-core frontal approach. Note that certain options are not available in this subsystem.

SUBROUTINE SUPER for linear dynamic response calculations by the method of mode superposition.

Input Data Reader

There are three major subprograms:

SUBROUTINE DATIN1 - Reads and interprets the parameter data input called by the execution supervisor. All the default values for control variables are set in this routine.

SUBROUTINE BULKIN - Reads and interprets the finite element model definition data and prints the mesh and loading data for the initial increment (number 0). The bulk data reader is entered before the execution of analysis modules. This subroutine utilizes the following lower level routines:

SUBROUTINE INITII for memory allocation of integer workspace in blank common to store nodal and element data.

SUBROUTINE DATIN2 for model data input.

SUBROUTINE DATOU1 for reporting model definition data.

SUBROUTINE CHKELM for the detection of clockwise element connectivity definition which results in a negative Jacobian.
This routine automatically corrects the connectivity table to a counterclockwise direction.

SUBROUTINE SUBDIV for memory allocation of subelement mesh data and automatic mesh generation of subelements.

SUBROUTINE INCRIN - Reads and interprets the loading and constraint data for each increment. This routine is invoked by individual analysis modules from inside the loop over the increments.

SUBROUTINE DATINI3, a small subset of the bulk data reader SUBROUTINE DATINI1 is used for actual operations including initialization of arrays at the beginning of an increment.

Algebraic Operation Subsystem

There are three groups of routines for the profile solver, frontal solver and eigenvalue extraction, respectively. The profile solution package consists of:

SUBROUTINE CCMPRO - Sets up the integer array for the profile of global stiffness equations to be stored in a profile form.

SUBROUTINE ASSEMS - Assembles the element stiffness equations into the global equation system stored in a profile form.

SUBROUTINE SOUT1 - Controls the iterative solution processes including the vector update required for the quasi- and secant-Newton iterations.

SUBROUTINE DECOMP - Factorizes the global stiffness equations stored in a profile form.

SUBROUTINE SOLVER - Performs the back substitution and generates the update vector for the incremental displacement.

The frontal solution package consists of:

SUBROUTINE FRONTW - Estimates the front matrix size to be accommodated in the core memory.

SUBROUTINE INFR - Allocates memory for the work space required for the frontal solution.

SUBROUTINE PRFRNT - Sets up the elimination table for the frontal solution.
SUBROUTINE FRONIF - Assembles and factorizes the global stiffness equation simultaneously.

SUBROUTINE FRONTB - Performs the back substitution and generates the updates for the displacement vector.

SUBROUTINE FRONTR - Controls the iterative solution processes including the vector update required for the quasi- and secant-Newton iterations. Calls SUBROUTINE FRONTB for the incremental displacement update vector.

SUBROUTINE VDSKIO - Controls the data stream stored in the in-core buffer area and the actual out-of-core storage devices.

The eigenvalue analysis package consists of:

SUBROUTINE EIGENV - Controls the execution of the eigenvalue extraction subsystem.

SUBROUTINE INIMDP - Initializes the array for eigenvectors.

SUBROUTINE SUBSPC - Performs the subspace iteration and generates a specified number of eigenvalues and eigenvectors.

SUBROUTINE JACOBI - Solves the eigenvalue problem in subspace by the Jacobi iteration.

There are a number of subprograms used commonly by the braic Operation Subsystem:

SUBROUTINE STRUCT - Controls the memory allocation for the global algebraic manipulations at the beginning of every increment.

SUBROUTINE INITI2 - Allocates memory required for the storage of global stiffness matrix and other vectors required in the linear algebraic manipulation of finite element equations.

SUBROUTINE LINESR - Calculates the search distance when the line search option is turned on.

Element Assembly Submodules

These are subprograms that construct vectors and matrices appearing in the algorithmic description of mixed iterative process discussed in the previous subsections.
SUBROUTINE ASSEMI - Assembles the displacement stiffness matrix for preconditioning purposes. All the kinematic and constitutive options are tested in this module.

SUBROUTINE ASSEM2 - Assembles the coefficient matrix for transient time integration by the Newmark family of algorithms. This routine has evolved from SUBROUTINE ASSEMI and contains all the same options.

SUBROUTINE ASSEM3 - Assembles the coefficient matrix for quasi-static analysis using the frontal solution subsystem. Large displacement, stress stiffening and centrifugal mass terms are not available in this package.

SUBROUTINE ASSEM4 - Calculates the nodal strain and recovers the residual vector in a mixed form. The subelement solution package is entered from this subprogram.

Element Loop Structure and Library Routines

In the element assembly submodules, element arrays are generated in the loops over elements. The protocol for accessing the element library is designed and implemented to involve a sequence of subroutine calls:

SUBROUTINE ELVULV - Sets up the current element parameters (See Table 2 for variables and values) from the element library table.

SUBROUTINE CNODEL - Pulls out quantities for the current element from the global nodal array and restores them in the element workspace. Coordinate transformations necessary for beam and shell elements are performed in this subprogram.

SUBROUTINE DERIV - Sets up the displacement-strain matrix for the current element by calling the element library subroutines. Those are:

SUBROUTINE BPSTRS for plane stress elements, types 3 and 101.

SUBROUTINE BPSTRN for plane strain elements, types 11 and 102.

SUBROUTINE BSOLID for three-dimensional solid element, type 7.

SUBROUTINE BSHELL for three-dimensional shell element, type 75.

SUBROUTINE BAXSYM for axisymmetric solid-of-revolution elements, types 10 and 103.
SUBROUTINE BTBEAM for linear Timoshenko beam element, type 98.

SUBROUTINE BASPST for the assumed stress plane stress element, type 151.

SUBROUTINE BASPSN for the assumed stress plane strain element, type 152.

SUBROUTINE BASSOL for the assumed stress three-dimensional solid element, type 154.

SUBROUTINE UDERIV - A slot for user coded element B matrix routine.

The following subprograms are used to calculate terms appearing in the finite element equations:

SUBROUTINE LMPMAS - Calculates nodal weight factor for the strain projection.

SUBROUTINE STIFF - Performs matrix triple products to assemble the element stiffness matrix and the element load vector associated with the initial strain terms.

SUBROUTINE STRAIN - Calculates element strain at specified sampling points and projects to nodes.

SUBROUTINE CNSMAS - Assembles the consistent mass matrix for modal and transient analysis.

SUBROUTINE INITST - Generates initial stress terms for quasi-static, buckling and modal analysis of prestressed structures.

SUBROUTINE CENMAS - Evaluates the centrifugal mass terms for rotating structures at speed.

SUBROUTINE RESID - Calculates the element residual vector for the global element.

SUBROUTINE SUBFEM - Performs the subelement solution and calculates the element residual vector for the subelement mesh.

SUBROUTINE RELDFG - Calculates the relative deformation gradient at the element sampling points and projects to nodes.

SUBROUTINE RESDYN - Calculates the contribution of mass and damping
terms in the element residual vector when the transient dynamics option is used.

Material Library

A system of subroutines is included in the MHOST code which covers a wide range of material models and initial strain assumptions:

SUBROUTINE SIMPLE - Integrates the stress over the increment assuming the elastic-plastic response of the material is modeled by the total secant modulus approach. Also generates the material modulus matrix.

SUBROUTINE PLASTS - Integrates the stress over the increment and calculates the plastic strain by using the radial return algorithm.

SUBROUTINE PLASTD - Calculates the consistent elastic-plastic modulus if the incremental equivalent plastic strain is positive.

SUBROUTINE WALKEQ - Integrates the Walker unified creep plasticity constitutive equation and also generates a material modulus based on the temperature dependent elasticity assumption.

SUBROUTINE LELAST - Calculates the stress for the constant material modulus given as data.

SUBROUTINE THRSTN - Calculates the thermal strain.

SUBROUTINE CRPSTN - Calculates the creep strain.

Report Generation Subsystem

A system of subroutines is included in the MHOST code to generate reports on the line printer image file and the formatted post-processing file readable by MENTAT, Version 5 and up (a proprietary interactive pre- and post-processor program developed by and available from MARC Analysis Research Corporation). Note that the format of the post-processing file is similar to that of the MARC General Purpose Finite Element Program, Version K.1 and the commercially available post-processing packages compatible with this program can easily be modified for use with MHOST.

SUBROUTINE PRININ - Reads and restores the user instructions for the print options. This routine is entered from the model data reader and/or the incremental reader. This subprogram interprets user instructions including the type of information, the locations (either nodes or element integration points) at which the information is...
printed and specified by character strings, and the range of nodes or elements specified by integer numbers.

SUBROUTINE PRINOU - Writes the reports on the line printer image file. A loop is constructed over the instruction set restored by PRININ.

SUBROUTINE PRINSU - Generates the reports for the subelement solution on the line printer image file. The same control structure as the global solution writer PRINOU is implemented. This routine also follows the instruction set generated by PRININ.

SUBROUTINE POSTOU - Writes formatted records on the post-processing file, Fortran Unit 19. All the information generated by the analysis is written. The user has no control over the records to be written on this file. The nodal values of stress and strain invariants and components are written in this subroutine.

SUBROUTINE POSTEN - Packs the nodal value record buffer and writes out the record when the buffer is filled.

1.3 Control Variables

The control variables effective globally through a number of subprograms are stored in the following common blocks:

ADDVAL Variables related to additional point masses, dashpots and springs.

ALGEM Fortran unit numbers and record number counters. Also the value of p is stored in this common block.

AUTOIN Real variables used for automatic load increments. Also control variables for adaptive load increments by the arc length method are stored here.

BODYFR A real array to store the direction of the body force vector for gravity loading or the position of two points defining the axis of rotation for centrifugal loading.

BSECT Pointers and counters for beam section definitions.

CONTRO Control variables and binary switches to activate options built into the MHOST program.

COMPND Flags for sequential execution of multiple analysis drivers.
COUNT  Counters for the line on the current page of the line printer image file.

CTITLE  The title line and the date and time when the execution is initiated.

DAMP  A damping factor array for transient dynamic analyses.

EIGEN  Counters for eigenvalue extraction by the subspace iteration method.

MODSUP  Counters for transient response calculation by vibration mode superposition.

ELEMEN  A element type indicator and flags for element array manipulations.

ELTYP  Counters for the element variables which are currently being processed. The entries in this common block are redefined every time the element loop is incremented.

ERRORS  Data input error counter.

FREE  Working arrays for the free format reader.

HARMON  Control variables for harmonic analysis.

INCON  Variables for the incremental data reader.

MACHIN  Machine dependent ratio of the integer and real word lengths.

MAXIMA  Maximum number of element variables appearing in the current problem.

LOUBIN  Flags for the selection of integration processes for various element arrays.

PAGCNT  Counters for the page numbers of the line printer image output file.

PARAM  Mesh parameters for the total number of data entries.

PERIOD  Parameters to define time periodic loading and boundary conditions.
POSTPN  Control variables used for generation of the post-processing file.

POWER  Control variables for the power spectrum.

PULSES  Counters and pointers for pulse leads in dynamic analysis.

RESULT  Control variables for generation of the line printer image output file.

SUBELM  Control variables for the subelement analysis option.

SUBTYP  Element variables for the subelement analysis option. The information stored in this common block is the subelement counterpart of ELTYP and entries are updated every time the loop over the subelements is incremented.

SUBSTR  Space allocated for the control variables of the substructure option (not used in Version 4.2).

SHIFT  Control variables for the power-shift option in the eigenvalue extraction for modal analysis.

TIME  Time parameters for transient dynamics and rate dependent calculations.

TIMLOC  The initial value for the CPU clock.

TMARCH  Parameters to define the time integration operator.

TRANSF  Local coordinate transformation arrays. Typically updated in an element-by-element manner.

TOLER  Error and tolerance for the global iterative solution.

VRTDSK  Control variables for the out-of-core solution by the frontal process.

ZPRINT  Control variables for a definition of a record in the line printer output file.

Control variables which play a significant role are summarized below:

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Common Block</th>
<th>Initial Value</th>
<th>Content</th>
</tr>
</thead>
</table>

Page: 31
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICREAD</td>
<td>ALGEM</td>
<td>5</td>
<td>Fortran unit number for the main data input device.</td>
</tr>
<tr>
<td>ILPRNT</td>
<td>ALGEM</td>
<td>6</td>
<td>Fortran unit number for the line printer output file (Execution log).</td>
</tr>
<tr>
<td>JLPRNT</td>
<td>ALGEM</td>
<td>6</td>
<td>Fortran unit number for the line printer output file (Bulk report).</td>
</tr>
<tr>
<td>ICONSL</td>
<td>ALGEM</td>
<td>1</td>
<td>Fortran unit number for the command input and the execution log output.</td>
</tr>
<tr>
<td>IPOSTF</td>
<td>ALGEM</td>
<td>19</td>
<td>Fortran unit number for the formatted post-processing file.</td>
</tr>
<tr>
<td>ISCRAF</td>
<td>ALGEM</td>
<td>10</td>
<td>Fortran unit number for the working on-line storage (binary).</td>
</tr>
<tr>
<td>IPOSTB</td>
<td>ALGEM</td>
<td>9</td>
<td>Fortran unit number for plotting data output file (binary - not used in Version 4.2).</td>
</tr>
<tr>
<td>IRRSTR</td>
<td>ALGEM</td>
<td>8</td>
<td>Fortran unit number for binary restart file.</td>
</tr>
<tr>
<td>JCREAD</td>
<td>ALGEM</td>
<td>12</td>
<td>Fortran unit number for alternate input data file.</td>
</tr>
<tr>
<td>LINE</td>
<td>ALGEM</td>
<td>0</td>
<td>Line counter for the output unit ILPRNT.</td>
</tr>
<tr>
<td>LINE2</td>
<td>ALGEM</td>
<td>0</td>
<td>Line counter for the output unit JLPRNT.</td>
</tr>
<tr>
<td>CURPER</td>
<td>AUTOIN</td>
<td>0.0</td>
<td>Load factor of the current arc-length iteration step.</td>
</tr>
<tr>
<td>TOTPER</td>
<td>AUTOIN</td>
<td>0.0</td>
<td>Total load factor including the current step of arc-length iteration.</td>
</tr>
<tr>
<td>ARCLEN</td>
<td>AUTOIN</td>
<td>0.0</td>
<td>The arc-length.</td>
</tr>
<tr>
<td>JADP</td>
<td>AUTOIN</td>
<td>0</td>
<td>Flag for the adaptive load increment option.</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Common Block</td>
<td>Initial Value</td>
<td>Content</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>NCREEP</td>
<td>AUTOIN</td>
<td>0</td>
<td>Maximum number of updates for the adaptive time step control for creep strain.</td>
</tr>
<tr>
<td>JEND</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for end-of-iteration.</td>
</tr>
<tr>
<td>JITER</td>
<td>CONTRO</td>
<td>0</td>
<td>Current iteration step of the global solution.</td>
</tr>
<tr>
<td>JTEMP</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for temperature loading.</td>
</tr>
<tr>
<td>JPRINT</td>
<td>CONTRO</td>
<td>0</td>
<td>Full line printer report generation interval for incremental analysis.</td>
</tr>
<tr>
<td>JP</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for pressure loading.</td>
</tr>
<tr>
<td>JSUB</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for substructure.</td>
</tr>
<tr>
<td>JINC</td>
<td>CONTRO</td>
<td>0</td>
<td>Current increment.</td>
</tr>
<tr>
<td>JREST</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for restart jobs.</td>
</tr>
<tr>
<td>JSAVE</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for check-point job to write the restart file.</td>
</tr>
<tr>
<td>JREDIM</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for memory allocation of global arrays for the next solution step.</td>
</tr>
<tr>
<td>JAUTO</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for automatic load incrementation.</td>
</tr>
<tr>
<td>JPOST</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for post file generation.</td>
</tr>
<tr>
<td>JBACK</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for back substitution to avoid the global stiffness assembly and factorization.</td>
</tr>
<tr>
<td>JOPTIM</td>
<td>CONTRO</td>
<td>0</td>
<td>Number of Cuthil-McGee iterations for bandwidth optimization.</td>
</tr>
<tr>
<td>JCREEP</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for the creep strain option.</td>
</tr>
<tr>
<td>Variable</td>
<td>Common Block</td>
<td>Initial Value</td>
<td>Content</td>
</tr>
<tr>
<td>----------</td>
<td>--------------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>JDIST</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for the distributed load option.</td>
</tr>
<tr>
<td>JCONST</td>
<td>CONTRO</td>
<td>0</td>
<td>Selector for the constitutive equation =</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 for linear elasticity</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 for secant elasticity/simplified plasticity</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 plasticity</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 unified creep plasticity</td>
</tr>
<tr>
<td>JDYN</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for the dynamic analysis option.</td>
</tr>
<tr>
<td>NONISO</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for anisotropic material response.</td>
</tr>
<tr>
<td>IThERM</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for temperature dependent material response.</td>
</tr>
<tr>
<td>NXSTAT</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for the static analysis module as the driver for the second step of the analysis.</td>
</tr>
<tr>
<td>NXSOLV</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for the optional frontal solution for the second step of the analysis.</td>
</tr>
<tr>
<td>NXYNTG</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for the direct time integration for the second step of the analysis.</td>
</tr>
<tr>
<td>NXMODL</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for modal analysis for the second step.</td>
</tr>
<tr>
<td>NXBCKL</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for buckling analysis for the second step.</td>
</tr>
<tr>
<td>NXSUPR</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for modal super-position for the second step of the analysis.</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Common Block</td>
<td>Initial Value</td>
<td>Content</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>TITLE</td>
<td>CTITLE</td>
<td>' '</td>
<td>An array to store the character string representing the title of the current analysis job.</td>
</tr>
<tr>
<td>IDYNMD</td>
<td>EIGEN</td>
<td>0</td>
<td>Number of eigenvalues and eigenvectors to be extracted.</td>
</tr>
<tr>
<td>IPTAR</td>
<td>EIGEN</td>
<td>0</td>
<td>Pointer to the workspace required for the subspace iteration. If the BFGS update is invoked in the static analysis, this variable is used as a pointer for an update vector.</td>
</tr>
<tr>
<td>IPTBR</td>
<td>EIGEN</td>
<td>0</td>
<td>Pointer to the workspace required for the subspace iteration. If the BFGS update is invoked in the static analysis, this variable becomes a pointer for one of the update vectors.</td>
</tr>
<tr>
<td>IC</td>
<td>ELEMEN</td>
<td>0</td>
<td>Current element type. This variable is updated when the element loop is incremented.</td>
</tr>
<tr>
<td>IEL</td>
<td>ELEMEN</td>
<td>0</td>
<td>Current element number.</td>
</tr>
<tr>
<td>IDF</td>
<td>ELEMEN</td>
<td>0</td>
<td>Number of d.o.f. per element being processed.</td>
</tr>
<tr>
<td>JLAW</td>
<td>ELEMEN</td>
<td>0</td>
<td>Constitutive equation type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>= 7</td>
</tr>
<tr>
<td>IPATH</td>
<td>ELEMEN</td>
<td>0</td>
<td>Flag for the integration method options for assembly and recovery.</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Common Block</td>
<td>Initial Value</td>
<td>Content</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>IASSEM</td>
<td>ELEMEN</td>
<td>0</td>
<td>Flag to indicate element assembly operations.</td>
</tr>
<tr>
<td>NELCRD</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of coordinate entries per node for the current element.</td>
</tr>
<tr>
<td>NELNFR</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of d.o.f. per node for the current element.</td>
</tr>
<tr>
<td>NELNOD</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of nodes per current element.</td>
</tr>
<tr>
<td>NELSTR</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of generalized stress/strain components per node for the current element.</td>
</tr>
<tr>
<td>NELPR</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of distributed load types for the current element.</td>
</tr>
<tr>
<td>NELINT</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of integration points for the current element.</td>
</tr>
<tr>
<td>NELLLV</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of d.o.f. per current element.</td>
</tr>
<tr>
<td>NELLAT</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of integration layers for the shell element to calculate resultant quantities.</td>
</tr>
<tr>
<td>NDI</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of direct strain/stress components for the current element.</td>
</tr>
<tr>
<td>NSHEAR</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of shear strain/stress components for the current element.</td>
</tr>
<tr>
<td>NELCMP</td>
<td>ELTYP</td>
<td>0</td>
<td>Number of LDamina strain/stress components (differs from NELSTR only for shells).</td>
</tr>
<tr>
<td>IA</td>
<td>FREE</td>
<td></td>
<td>Buffer for the card image read from ICREAD file.</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Common Block</td>
<td>Initial Value</td>
<td>Content</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>MAXCRD</td>
<td>MAXIMA</td>
<td>1</td>
<td>Maximum number of coordinate entries in the current mesh.</td>
</tr>
<tr>
<td>MAXNFR</td>
<td>MAXIMA</td>
<td>2</td>
<td>Maximum number of d.o.f. per node in the current mesh.</td>
</tr>
<tr>
<td>MAXNOD</td>
<td>MAXIMA</td>
<td>1</td>
<td>Maximum number of nodes per element in the current mesh.</td>
</tr>
<tr>
<td>MAXCHR</td>
<td>MAXIMA</td>
<td>1</td>
<td>Maximum number of material data entries for the element definition in the current mesh.</td>
</tr>
<tr>
<td>NSUMAX</td>
<td>MAXIMA</td>
<td>0</td>
<td>Maximum number of substructures (not-used in version 4.2)</td>
</tr>
<tr>
<td>MAXBSP</td>
<td>MAXIMA</td>
<td>6</td>
<td>Maximum number of entries in the beam section definition in the mesh.</td>
</tr>
<tr>
<td>Variable Name</td>
<td>Common Block</td>
<td>Initial Value</td>
<td>Content</td>
</tr>
<tr>
<td>MAXDMT</td>
<td>MAXIMA</td>
<td>1</td>
<td>Maximum dimension of the constitutive resultant matrix.</td>
</tr>
<tr>
<td>MAXFRN</td>
<td>MAXIMA</td>
<td>0</td>
<td>Maximum front matrix size (valid when the frontal solution option is invoked).</td>
</tr>
<tr>
<td>MAXEAN</td>
<td>MAXIMA</td>
<td>0</td>
<td>Maximum number of entries in the inverse connectivity table. The value is set in INITI2.</td>
</tr>
<tr>
<td>MAXBET</td>
<td>MAXIMA</td>
<td>0</td>
<td>Maximum number of entries in the element strain-displacement matrix for all the integration points.</td>
</tr>
<tr>
<td>JLOUB</td>
<td>LOUBIN</td>
<td>1</td>
<td>Not used in Version 4.2.</td>
</tr>
<tr>
<td>JINTER</td>
<td>LOUBIN</td>
<td>3</td>
<td>Selector for the integration rule used to generate the strain-displacement equation.</td>
</tr>
</tbody>
</table>

\[=1\]

\[=2\]

Reduced integration
Selective integration
(one point shear sampling)
Selective integration with the element coordinate transformation.

JEXTRA | LOUBIN | 1  
---|---|---
Selector for the integration rule used in the residual force calculation.

= 1  
= 2  
Full Gaussian quadrature 
One point reduced integration

JWEIGH | LOUBIN | 3  
---|---|---
Selector for the integration rule used to generate the stiffness equations.

= 1  
= 2  
= 3  
Full Gaussian quadrature 
Selective integration (one point shear sampling without coordinate transformation). 
Selective integration with the element coordinate transformation.

NPAGE1 | PAGCNT | 0  
---|---|---
Page counter for the line printer output on ILPRNT.

NPAGE2 | PAGCNT | 0  
---|---|---
Page counter for the line printer output on JLPRNT.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Common Block</th>
<th>Initial Value</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTYPE</td>
<td>PARAM</td>
<td>0</td>
<td>Number of different element types in the current mesh.</td>
</tr>
<tr>
<td>NELEM</td>
<td>PARAM</td>
<td>0</td>
<td>Total number of elements in the current mesh.</td>
</tr>
<tr>
<td>NBC</td>
<td>PARAM</td>
<td>0</td>
<td>Total number of displacement boundary conditions in the current mesh. The user specified value in the parameter data section is used for memory allocation. The value is updated to the actual value when the boundary displacement data is</td>
</tr>
<tr>
<td>PARAM</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>NNODE</td>
<td></td>
<td>Total number of nodes in the present mesh.</td>
<td></td>
</tr>
<tr>
<td>NTIE</td>
<td></td>
<td>Total number of tying equations in the present mesh.</td>
<td></td>
</tr>
<tr>
<td>NMAX</td>
<td></td>
<td>Maximum number of terms in the tying equations in the present mesh.</td>
<td></td>
</tr>
<tr>
<td>NTRAN</td>
<td></td>
<td>Total number of nodal coordinate transformations specified by the user in the current mesh.</td>
<td></td>
</tr>
<tr>
<td>NSBFGS</td>
<td></td>
<td>Number of update vectors for the BFGS quasi-Newton method.</td>
<td></td>
</tr>
</tbody>
</table>

2.0 DATA STORAGE SCHEME

This chapter discusses the dynamic core allocation scheme implemented in the MHOST code. The first section outlines the design principle and is followed by the code actually allocating memory in the second section. The third section is devoted to the definition of pointers.

2.1 Overview

A dynamic core allocation strategy is implemented in the MHOST code. An integer array is defined as an entry to blank common. This array is used as the working storage. Except for the frontal solution and transient dynamic analysis options, all the information required for the finite element computation is stored in this work space. First parameters defining the number of variables for element types in use is copied from the array residing in SUBROUTINE TYPEIN.

The first section of this space stores the nodal variables and element arrays. The amount of memory required to store these data is found from the number of nodes, elements and attributes given by the user as part of the parameter data. This portion of working storage and information stored therein is often referred to as the finite element database. The pointers for the finite element database are defined in SUBROUTINE INITI1 before reading the model data section of the input data file. The model data is read directly into the space allocated for the finite element database.
In finite element computations, space is required to store arrays where size varies depending on the analysis type and mesh topology. For instance, in most of the analysis options, the global stiffness matrix is stored in a profile form. After the model data is read, the mesh topology (element connectivity) is swept through and column heights are calculated for each degree of freedom in SUBROUTINE COMPRO. The pointers for the global arrays are then calculated in INITI2.

For the local-global analysis inherent to the subelement procedure, an indirect accessing scheme is implemented. When the subelement option is invoked by the parameter data, element arrays to store subelement data pointers are prepared as a part of the finite element database. The pointers are calculated only for the global elements which are divided into subelements. The mechanism of this indirect addressing scheme is implemented in the memory allocation routine INITSE and the subelement solution driver routine SUBFEM.

2.2 Memory Allocation Subprograms

The following SUBROUTINE TYPEIN substitutes the element characteristics array into the workspace:

C=SUBROUTINE-TYPEIN CALLED FROM SUBROUTINE 'DATINI'
SUBROUTINE TYPEIN
1 (IWORK ,RWORK ,IERR ,NTYPE ,ILAST ,ILPRNT,NDIMEN)

C **********************************************************************
C C ROUTINE FOR OBTAINING INPUT FOR DIFFERENT TYPES OF ELEMENTS
C C IERR ERROR FLAG
C NTYPE NUMBER OF ELEMENT TYPES
C ILAST LAST WORD FOR INPUT
C ILPRNT OUTPUT DEVICE
C NDIMEN DIMENSIONS OF THE ANALYSIS PROBLEM
C **********************************************************************
C IMPLICIT REAL*8 ( A-H , O-Z )
REAL*4 RWORK
C **********************************************************************
C DIMENSION IWORK ( 1 ) , RWORK ( 1 )
DIMENSION ISTORE( 13 , 16 )
PARAMETERS TO CHARACTERIZE ELEMENTS AVAILABLE IN THE HOST PROGRAM

DATA ISTORE

1/ 9, 3, 2, 1, 6, 2, 0, 1, 1, 1, 0, 1,
2 3, 2, 4, 3, 6, 4, 3, 1, 3, 2, 1, 2,
3 11, 2, 4, 4, 6, 4, 3, 1, 4, 3, 1, 3,
4 10, 2, 4, 4, 6, 4, 3, 1, 4, 3, 1, 4,
5 7, 3, 8, 6, 6, 8, 3, 1, 6, 3, 3, 5,
6 75, 7, 6, 9, 6, 4, 4, 5, 5, 2, 3, 6,
7 72, 7, 6, 9, 6, 4, 4, 5, 5, 2, 3, 6,
8 98, 6, 6, 2, 6, 6, 1, 3, 1, 6, 3, 3, 7,
9 101, 2, 2, 9, 3, 6, 9, 3, 1, 3, 2, 1, 2,
10 102, 2, 2, 9, 3, 6, 9, 3, 1, 4, 3, 1, 3,
11 103, 2, 2, 9, 3, 6, 9, 3, 1, 4, 3, 1, 4,
2 104, 3, 27, 6, 6, 27, 3, 1, 6, 3, 3, 5,
3 151, 2, 2, 4, 3, 6, 4, 3, 1, 3, 2, 1, 2,
4 152, 2, 2, 4, 3, 6, 4, 3, 1, 4, 3, 1, 3,
5 153, 2, 2, 4, 3, 6, 4, 3, 1, 4, 3, 1, 4,
6 154, 3, 8, 6, 6, 8, 3, 1, 6, 3, 3, 3, 5/
As seen in the following example, the element data copied by the above subprogram is accessed every time the element loop is incremented:

```plaintext
CALL NUL( REA(IMASNO), NNODE)
CALL NUL( REA(IGSINO), NNODE*NELSTR)
CALL NUL( REA(IEQPSI), NELLAY*NELNOD )
CALL NUL( REA(IIPSNO), NELCMP*NELLAY*NELNOD )
```
C >>> START THE FIRST ELEMENT LOOP >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

C DO 5500 IELCNT = 1 ,NELEM
   IC = INT( ITYP + IELCNT - 1 )
   II = INEL + MAXNOD*( IELCNT - 1 )
C ***
   IF( IC .EQ. 0 ) GO TO 5500
C ***
   CALL ELVULV( INT , IC , IERR )
   CALL CNODEL( REA ,INT ,INT(II) ,CTRANS ,IELCNT ,IC , -1 , -1 )
C CALL LMPMAS
1 ( REA(IMASNO),REA(IDET),INT(II),NELNOD,NELINT,
2 NNODE ,IC ,REA(ICH),REA(ICOR),NELCRD ,
3 IELCNT ,NELEM ,REA(INOD),MAXCRD ,CTRANS ,
4 JGRAM )
C IF (JPRES .GT. 0) CALL BOUNDN
1 ( REA(IBLENORM),REA(ICOR),INT(II),IELCNT ,IC ,
2 NNODE ,NELNOD ,NELCRD )
C 5500 CONTINUE
C <<< CLOSE THE FIRST ELEMENT LOOP <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
C ...
NORMALIZE THE NODAL NORMALS FOR BEAMS AND SHELLS
C IF (JLAW.EQ.6 .OR. JLAW.EQ.9) CALL NRMNRM(REA(INOD),MAXCRD,NNODE)
C ************************************************************
C 8500 CONTINUE

SUBROUTINE ELVULV as shown below accesses the entries defined by SUBROUTINE TYPEIN:

C=SUBROUTINE=ELVULV CALLED BY SUBROUTINES 'ASSEM','ASSEM4'
SUBROUTINE ELVULV
1 ( IWORK ,IC1 ,IERR )
C ************************************************************
C EXTRACTS INFORMATION ABOUT A CERTAIN ELEMENT TYPE
C IC1 ELEMENT TYPE
IERR  ERROR FLAG

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

IMPLICIT REAL*8 ( A-H , O-Z )
REAL*4  RWORK

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

DIMENSION IWORK(I)

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

COMMON / ALGEM / IREAD,ILPRNT,ILPRNT,ICONS,IPRINT,JPRINT,ISCRAP,
   1 IPLOTB,IRSTRT,ICREAD,IPVARS,IPSETS,IFILEX,
   2 PI ,LINE ,LINE2
COMMON / CONTRO / JEND ,JTER ,JTEMP ,JPRINT,JP ,JSUB ,
   1 JINC ,JREST ,JSAVE ,JREDIM,JAUTO ,JPOST ,
   2 JBACK ,JOPTIM,JCREEP,JDIST ,JCONST,JDYN ,
   3 NONISO ,IHERM ,ITRIG ,IDYN ,JREPOT,JTANGE ,
   4 JHERM ,JFORCE,JELEMENT,JCOREF,JDISTS ,JHOOK ,
   5 JDIV ,JUBOUN,IDSTOP,INTSTP,JINTER,JPLAST ,
   6 JFRONT,JECON,JEADD ,JTEST ,JINDP ,IFBFGS ,
   7 IPSCIB ,ILINE ,IPRINT,ICOMP ,IPCONS,IPIONS,
   8 IFBOD ,IFGRAV ,IFCENT,JDAMP ,IHYD ,IEST ,
   9 JFDXX,JISTIF,JCONIM,JPRINT,JLARGE,JPOLOH ,
 + JWKSFL,JPRES ,JCDUM2,JCDUM3
COMMON / ELITYP / NELCD,ANELTD,ANELT2,ANELT3,ANELT4,ANELT5,
   1 NELCD,ANELT6,ANELT7,ANELT8,ANELT9,ANELT10,
COMMON / START1 / IELPRM,ITYP ,INEL ,ICHAR ,IPRES ,ITRNS ,
   1 ISTRN,ICOP ,IPRINT,IPRES ,IDIST ,ISEAM ,
   2 IPRES,IBNORM,IBMAT ,ISTIL6,ISTIL7,ISTIL8
COMMON / PARAM / NTYPE ,NELEM ,NNODE ,NBC ,NTIE ,NMAX ,
   1 NTRAN ,NTRAN ,NFD ,NBAND ,NEXT ,NSUB ,
   2 NPRINT,NPRES ,NSBC ,NDUP ,NSIZE ,NBECT ,
   3 NSHIFT,NSBCS,NGMRS ,NSPRI ,NMAX ,NDASH ,
   4 NDYNMD,NSBNC ,NSUPER,NHARM ,NBASE ,NINC ,
   5 NITER ,NPSITS,NPDPTS ,NPULSE ,NPDPTS ,NHARD ,
   6 NSMCH,NDIMN,NMINT,NPAR40,NPAR41,NPAR42 ,
   7 NPAR43,NPAR44,NPAR45,NPAR46,NPAR47,NPAR48
COMMON / ELEMEN / IC ,IEL ,IDF ,JLAW ,IPATH ,JASSEM ,
   1 JRULE ,JCARL ,JEL009,JEL010,JEL011,JEL012
DO 5000 II = 1 ,NTYPE
C
II = IWORK(IELPRM + 13*(II - 1))
IF(I1 .EQ. IC1) GO TO 8000
5000 CONTINUE
CALL LINES( 1 , 1 )
WRITE(ILPRNT,9999) IC1
WRITE(ICONSL,9999) IC1
9999 FORMAT(2X,28H***ERROR*** ELEMENT TYPE,I5,12H NOT DEFINED)
C
C **********************************************
C
IERR = IERR+1
C
C **********************************************
C
8000 CONTINUE
IS1 = IELPRM + 13*(II - 1) + 1
NELCRD = IWORK(IS1)
NELNFR = IWORK(IS1+1)
NELNOD = IWORK(IS1+2)
NELSTR = IWORK(IS1+3)
NELCHR = IWORK(IS1+4)
NELINT = IWORK(IS1+5)
NELPR = IWORK(IS1+6)
NELLAY = IWORK(IS1+7)
NELCNP = IWORK(IS1+8)
NDI = IWORK(IS1+9)
NSHEAR = IWORK(IS1+10)
JLAW = IWORK(IS1+11)
NELLV = NELNFR*NELNOD
IDF = NELNFR*NELNOD
C
C **********************************************
C
SPECIAL TREATMENT FOR THE NINE POINT INTEGRATION OF THE QUADRATIC
C LAGRANGIAN ELEMENTS ( ONLY IN CASE OF MATRIX ASSEMBLY )
C
IF ( IASSEM .EQ. 0 .AND. NELINT .EQ. 9 ) NELINT = 4
C
C **********************************************
C
IF (JPRINT.LT.3 .OR. NBAND.EQ.0) GO TO 9000
C
CALL LINES( 4 , 4 )
In the element loops, information related to the element being processed is pulled out from the global finite element database and substituted into the element work arrays. The following SUBROUTINE CNODEL is called before the element data manipulation:

C=SUBROUTINE CNODEL = CALLED BY 'ASSEMI','ASSEM4','MASMAT'
    'DIV2Q2','ASSEM3','INITST','NEWRHS',
    'PERTRB','RESTOR'
SUBROUTINE CNODEL
1 (REA ,INT ,ISLV ,CTRANS,IEL ,IC ,IFLAG ,ISTEP )
C
C **********************************************************************
C
C EXTRACTS EI/KMENT QUANTITIES AND UPDATES GEOMETRY
C
C ISLV CC_STITUTIVEIAW_
C IEL PO_TOELEMENTNUMBER
C
C **********************************************************************
C
IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 REA
C
DIMENSION REA(1),INT(1),ISLV(1),CTRANS(3,3)
DIMENSION VTRANS(2,2),TIRANS(3,3),ETRANS(3,3)
C
C **********************************************************************
C
DALPHA, DBETA ,DGAMMA
ALGEM
ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1 IPILOTB, IISTRRT, JCREAD, IFVARS, IIPSETS, IFILEX,
2 PI ,LINE ,LINE2
COMMON / TMARCH / DALPHA, DBETA ,DGAMMA
COMMON / ALGEM / ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1 IPILOTB, IISTRRT, JCREAD, IFVARS, IIPSETS, IFILEX,
2 PI ,LINE ,LINE2
COMMON / CONTRO / JEND ,JITER ,JTEMP ,JPRINT, JP ,JSUB ,
COMMON / ELTYP / NELCRD, NELNFR, NELNOD, NELSTR, NELCHR, NELFR,
    NELINT, NELLV, NELLAY, NDI, NSHEAR, NELCMP
COMMON / MAXMA / MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
    MAXLAY, MAXINT, MAXWRK, MAXNLV, NSUMAX, MAXCMP,
    MAXBSP, MAXGMR, MAXTEM, MAXELM, MAXXK, MAXMT,
    MAXFRN, MAXBET, MAXVAR, MAXSET, MAXEAN, MAXORD,
    MAX025, MAX026, MAX027, MAX028, MAX029, MAX030
COMMON / PARAM / NTYPE, NELEM, NNODE, NBC, NITE, NMAX,
    NTRAN, NTRAC, NPD, NBAND, NEXT, NSUB,
    NPRINT, NPOST, NSBC, NDUP, NSIZE, NBSECT,
    NSHIFT, NSBFCS, NQMF, NSP, NCP, NDASH,
    NDF, NSBC, NSU, NCS, NSUMAX, MAXCMP,
    NSUM2H, NDIM, NMDNIT, NPAR40, NPAR41, NPAR42,
    NPAR43, NPAR44, NPAR45, NPAR46, NPAR47, NPAR48
COMMON / START1 / IELPRM, ITYP, INEL, ICHAR, IPRES, ISTRS,
    ISTRN, ICOP, IPRINT, IPOST, IDIST, ILEAN,
    IBPRES, IBNF, IMONIT, IST116, IST117, IST118
COMMON / START2 / INOD, ITEM, INLV, IPOSO, ITEMDP, IDUP
COMMON / START4 / IDINC, IDTOT, IFORCE, IRESID, IWNUM, ISIGNO,
    IEPSNO, IPSTN, ICSTRN, IST, ISTR, ISTRN,
    ITPSTN, IICSTR, ITSTR, IPSTNO, ICSTRNO, ISTNO,
    IT, IIP, IICSNO, IIPSTNO, IICSTRNO, ISTNO, IIPST,
    IDMN, IBCST, IOMINO, IOMNO, ITDSN, ICSNO,
    IDYNV, IDYNA, IDSX1, IDSX2, IDSX3, IDYNA, ISWELL,
    IDQCSI, IPREF, IDsX3, IDYNA, IDYNA, IST523, IST524
COMMON / START5 / IRL, IREAC, IES, IAB, IBQM, ISRL,
    IBLC, ISKM, ILAST, IRLB, IINCP, IFORIN,
    IOP, IDAM, IMASM, IDIAG, IUPTRI, ICLMP,
    IMASDI, IMASUP, IST521, IST522, IST523, IST524
COMMON / START6 / IELV, ICOR, ISIG, IEP, IWNUM, ISIGNO,
    IENOD, IETM, ICH, IPP, IXRL, IXRL,
    IXP, IXX, KSTRN, KSTRN, KSTRN, KSTRN, KSTRN,
C
KISTINO, KISTINO, KISNNO, KIPSNO, KICSNO, KITSNO, IMAISO, IMNOD, IEQPST, IEQPSI, KEQPST, KEQPSI, KDMAT, KDMINO, KTDSNO, KITDST, IXM, IXC, IVELM, IAEIM, IMASEL, KYIELD, IST647, IST648, IST649, IST650, IST651, IST652, IST653, IST654

COMMON / START7 / ICON, IXBKR, ITRACR, ITRANR, IBETA, IDET
COMMON / START8 / KGEPS, IGEPS, KGSIG, KGSIG, KGDST,
1 IGEPSNO, IIGENO, IGSINO, IIGSNO, IIGTNO,
2 KGEPSNO, KIGENO, KGSINO, KIGSNO, KIGTNO
COMMON / START9 / KEQCSI, KIOMNO, KSWLNO, KIPNNO, KITDNO, KIDUMMY,
1 KEQCST, KOMENO, KVSNO, IST910, IST911, IST912
COMMON / MACHIN / IDP

C
**********************************************************************
C
NELST2 = NELSTR * NELSTR
MAXST2 = MAXSTR * MAXSTR
IPP = IPRES + (IEL-1) * MAXPRS * IDP
MAXDIM = MAXCMP * MAXLAY
NELDIM = NELCMP * NELLAY

C
**********************************************************************
C
CALL SEARCH
1 (INT(IENV), INT(INLV), ISLV, MAXNFR, NNODE, 1, NELNOD, NELNFR)
CALL SEARCH
1 (REA(ICOR), REA(INOD), ISLV, MAXCRD, NNODE, 1, NELNOD, NELCRD)
CALL SEARCH
1 (REA(ISXL), REA(IDTOT), INT(IENV), 1, NFD, 1, NELLV, 1)
CALL SEARCH
1 (REA(ISXL), REA(IDINC), INT(IENV), 1, NFD, 1, NELLV, 1)
CALL SEARCH
1 (REA(KDMINO), REA(IDMINO), ISLV, MAXST2, NNODE, 1, NELNOD, NELST2)
CALL SEARCH
1 (REA(KGEPSNO), REA(IGEPSNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KIGENO), REA(IIGENO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KGSINO), REA(IIGSINO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KITDNO), REA(IIGTNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
CALL SEARCH
1 (REA(KIDSNO), REA(ITDSNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KEQPSI), REA(IEQPSI), ISLV, NNODE, 1, NELNOD, 1)
CALL SEARCH
1 (REA(KEQPST), REA(IEQPST), ISLV, NNODE, 1, NELNOD, 1)
CALL SEARCH
1 (REA(KIPSNO), REA(IIPSNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KICSN0), REA(IICSN0), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KITSNO), REA(IITSNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KEQCSI), REA(IEQCSI), ISLV, NNODE, 1, NELNOD, 1)
CALL SEARCH
1 (REA(KIOMNO), REA(IIOOMNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KSWLNO), REA(ISWELL), ISLV, NNODE, 1, NELNOD, 1)
CALL SEARCH
1 (REA(KIMPNO), REA(ITEM), ISLV, NNODE, 1, NELNOD, 1)
CALL SEARCH
1 (REA(KIDFNO), REA(ITEMDF), ISLV, NNODE, 1, NELNOD, 1)
CALL SEARCH
1 (REA(KISTR), REA(ICHAR), ISLV, MAXCHR, NNODE, 1, NELNOD, NELCHR)
CALL SEARCH
N3HARD = NHARD * 3
CALL SEARCH
1 (REA(KYIELD), REA(IYIELD), ISLV, N3HARD, NNODE, 1, NELNOD, N3HARD)
CALL INTERP
1 (REA(ICH), REA(KISTR), NIELNOD, NELCHR)

C
C
D-MATRIX ROTATION FOR ANISOTROPIC MATERIALS
C
IF (NONISO .EQ. 1) CALL ROTMT
1 (REA(KDMNO), REA(IPREF), ISLV, NIELNOD, NELSTR, NMODE, NDI, N3HARD)
C
C *** DYNAMIC CALCULATIONS: QUANTITIES ASSOCIATED WITH NODAL TIME DERIVATIVES
C
IF (IDYN .EQ. 1 .AND. JDYN .EQ. 1) THEN
CALL SEARCH
1 (REA(IAELM), REA(IDYNA), INT1IELV, 1, NFD, 1, NELLV, 1)
CALL SEARCH
1 (REA(IVELM), REA(IDYNV), INT1IELV, 1, NFD, 1, NELLV, 1)
END IF

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

IF( IC.EQ.75 .AND. JINC.LT.JLARGE ) THEN

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

DO 90 I = 1, NELNOD
CALL NUL (VTRANS, 4 )
CALL NUL (TTRANS, 9 )
CALL TSH04N(VTRANS,TTRANS,ETRANS,ETRANS,REA(ICOR), I,
* MAXCRD,NELNOD, SIGN,IFLAG)

II = NELSTR * IDP *(I - 1)

CALL SHTRAN(REA(KGEPNO+II),VTRANS,ETRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGNO+II),VTRANS,ETRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGNO+II),VTRANS,TTRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGNO+II),VTRANS,TTRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGNO+II),VTRANS,TTRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGNO+II),VTRANS,TTRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGNO+II),VTRANS,TTRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGNO+II),VTRANS,TTRANS,SIGN,NELSTR)

***

DO 70 K = 1, MAXSTR
   K1 = NELSTR * IDP * (NELSTR *(I - 1)+(K - 1))

CALL SHTRAN(REA(KDMINO+K1),VTRANS,TTRANS,SIGN,NELSTR)
70 CONTINUE

I2 = NELSTR * NELSTR * IDP *(I - 1)

CALL TRANSP(REA(KDMINO+I2),NELSTR)
CALL TRANSP(ETRANS,3)
CALL MATINV(ETRANS,3,3,ISW)

DO 80 J = 1, MAXSTR
   J1 = NELSTR * IDP * (NELSTR *(I - 1)+(J - 1))
   CALL SHTRAN(REA(KDMINO+J1),VTRANS,ETRANS,SIGN,NELSTR)
80 CONTINUE

90 CONTINUE

**********************************************************************
ELSE IF ( IC.EQ.98 .AND. JINC.LT.JLARGE ) THEN

******************************************************************************
CALL TBM02N(CTRANS,REA(ICOR),MAXCRD,NELNOD)
******************************************************************************
ENDIF

IF( JINC.GE.JLARGE ) THEN

******************************************************************************
UPDATE GEOMETRY FOR LARGE DEFORMATION ANALYSIS
******************************************************************************

BEGINNING OF INCREMENT (ISTEP = -1):
           Xi   = X0 + Ui

MID-INCREMENT (ISTEP = 0 ):
           Xi+1/2 = X0 + Ui + 1/2*(si)*(Ui+1 - Ui)

END OF INCREMENT (ISTEP = 1 ):
           Xi+1 = X0 + Ui + (si)*(Ui+1 - Ui)

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

HALF    = 0.5D0
ONE     = 1.0D0

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

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

IF( NDIMEN.EQ.MAXNFR ) THEN
NUMBER OF COORDINATES IS EQUAL TO THE NUMBER OF DEGREES-OF-FREEDOM (CONTINUUM ELEMENTS)

------------------------------------------------------------------

COMPUTE GEOMETRY AT THE BEGINNING OF THE INCREMENT
------------------------------------------------------------------

CALL ADD( REA(ICOR),REA(ICOR),REA(IXHR),NDIMEN*NELNOD)

IF( ISTEP.EQ.0 ) THEN

------------------------------------------------------------------

COMPUTE HALF OF INCREMENTAL DISPLACEMENTS TO OBTAIN MID-INCREMENT GEOMETRY
------------------------------------------------------------------

CALL ADDSMU( REA(ICOR),ONE,REA(ICOR),HALF,REA(IXHR),
&
 & NDIMEN*NELNOD)

ELSE IF( ISTEP.EQ.1 ) THEN

------------------------------------------------------------------

USE TOTAL INCREMENTAL DISPLACEMENTS TO OBTAIN END OF INCREMENT GEOMETRY
------------------------------------------------------------------

CALL ADD( REA(ICOR),REA(ICOR),REA(IXHR),NDIMEN*NELNOD)

ENDIF

ELSE IF( MAXNFR.GT.NDIMEN ) THEN

------------------------------------------------------------------

NUMBER OF DEGREES-OF-FREEDOM IS GREATER THAN THE NUMBER OF COORDINATES (BEAM AND SHELL ELEMENTS)
------------------------------------------------------------------

DO 10000 I=1,NELNOD

IICOR = ICOR + (I-1)*NELCRD*IDP

SYSTEMS' MANUAL

Page : 52
\[
I_{IXRL} = IXRL + (I-1) \times \text{MAXNFR} \times IDP \\
I_{IXIRL} = IXIRL + (I-1) \times \text{MAXNFR} \times IDP 
\]

---

**COMPUTE GEOMETRY AT THE BEGINNING OF THE INCREMENT**

---

```
CALL ADD( REA(IICOR),REA(IICOR), 
          REA(IIXRL),NDIMEN )
```

**IF** (ISTEP.EQ.0) **THEN**

---

**COMPUTE HALF OF INCREMENTAL DISPLACEMENTS TO OBTAIN MID-INCREMENT GEOMETRY**

---

```
CALL ADDSMU( REA(IICOR),ONE ,REA(IICOR), 
              HALF,REA(IIXIRL),NDIMEN)
```

**ELSE IF** (ISTEP.EQ.1) **THEN**

---

**USE TOTAL INCREMENTAL DISPLACEMENTS TO OBTAIN END OF INCREMENT GEOMETRY**

---

```
CALL ADD( REA(IICOR),REA(IICOR), 
          REA(IIXIRL),NDIMEN )
```

**ENDIF**

---

```
10000 CONTINUE
```

**IF** (IC.EQ.75) **THEN**

---

**UPDATE TRANSFORMATIONS**

---

```
DO 20000 I = 1 , NELNOD
   CALL NUL (VTRANS, 4 )
```

---

**SYSTEMS' MANUAL**

Page : 53
CALL NUL (TTRANS, 9 )
CALL TSH04N(VTRANS,TTRANS,ETRANS,CTRANS,REA(ICOR),I,
& MAXCRD,NELNOD, SIGN,IFLAG)

CALL SHTRAN(REA(KGEPNO+I1),VTRANS,ETRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KIGENO+I1),VTRANS,ETRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGSINO+I1),VTRANS,TTRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KGIDNO+I1),VTRANS,TTRANS,SIGN,NELSTR)
CALL SHTRAN(REA(KTDSNO+I1),VTRANS,TTRANS,SIGN,NELSTR)

DO 15000 K = 1 ,MAXSTR
   K1= NELSTR * IDP * (NELSTR*(I - i)+(K - i))
   CALL SHTRAN(REA(KDMI%Z}+KI),VTRANS,TTRANS,SIGN,NELSTR)
15000 CONTINUE

DO 16000 J = 1 ,MAXSTR
   J1= NELSTR * IDP * (NELSTR*(I - i)+(J - i))
   CALL SHTRAN(REA(KDMI_O+JI),VTRANS,ETRANS,SIGN,NELSTR)
16000 CONTINUE

ELSE IF( IC.EQ.98 ) THEN

UPDATE BEAM TRANSFORMATION

END IF

ELSE IF( NDIMEN.GT.MAXNFR ) THEN

SOMETHING IS WRONG!!!!!!!!!!
Note that the coordinate transformations necessary for the shell elements take place in this subprogram.

The following subroutine INITII calculates pointers for the finite element database:

C=SUBROUTINE INITII CALLED BY SUBROUTINE 'BULKIN'

SUBROUTINE INITII
  1 (R WORK ,I WORK ,ISIZE )

C
C **********************************************************************
C
C ALLOCATES CORE FOR DATA INPUT AND ZERNE OUT CORE
C
C **********************************************************************
C
VARIABLES

IAELM /START6/ ELEMENT ARRAY FOR NODAL ACCELERATION
IBASE /HARMON/ HARMONIC BASE MOTION MAGNITUDE AND PHASE
IBETA /START7/ ELEMENT BETA MATRIX
IBSECT /BSECT/ POINTER TO BEAM SECTION PROPERTY SETS
IBSECT /BSECT/ POINTER FOR THE LAST ADDRESS
ICH /START6/ ELEMENT MATERIAL PROPERTY ARRAY (LOCAL)
ICHAR /START1/ ELEMENT MATERIAL PROPERTY ARRAY (GLOBAL)
ICMFOR /HARMON/ POINTER TO THE COMPLEX MODAL FORCE
ICMRES /HARMON/ POINTER TO THE COMPLEX MODAL RESPONSE
ICNFOR /HARMON/ POINTER TO THE COMPLEX MODAL VELOCITY
ICNPRT /START5/ POINTER TO ARRAY DEFINING THE LOCATIONS OF THE
COLUMNS ELEMENTS THAT ARE JUST ABOVE THE DIAGONAL
ICON /START7/ ELEMENT CONNECTIVITY ARRAY
ICTOP /START1/
C ICOR /START6/ NODAL COORDINATE ARRAY
C ICSTNO /START4/ NODAL CREEP STRAIN
C IDASH /ADVAL/ POINTER TO THE ADDED DASHPOT DAMPING VALUES
C IDET /START7/ DETERMINANTS ARRAY AT INTEGRATION POINTS
C IDFINC /START4/ INCREMENTAL DEFORMATION GRADIENT ARRAY
C IDFTOT /START4/ TOTAL DEFORMATION GRADIENT ARRAY
C IDIAG /START5/ POINTER TO DIAGONAL COMPONENTS OF GLOBAL STIFFNESS
C ARRAY STORED IN PROFILE FORM
C IDINC /START4/ INCREMENTAL DISPLACEMENT ARRAY
C IDINCP /START5/ ITERATION VECTOR FOR INCREMENTAL DISPLACEMENT
C IDIST /START1/ DISTRIBUTED LOAD ARRAY
C IDMNO /START4/ NODAL DMATRIX ARRAY ALLOCATION
C IDP /MACHIN/ WORD LENGTH ( 1 REAL WORD = IDP INTEGER WORD )
C IDSTTR /START4/ DISTRIBUTED LOAD INPUT ARRAY
C IDSK1 /START4/ WORKING ARRAY FOR DYNAMICS
C IDSK2 /START4/ WORKING ARRAY FOR DYNAMICS
C IDIOT /START4/ TOTAL DISPLACEMENT VECTOR
C IDUP /START2/ DUPLICATED NODE CONNECTIVITY
C IDYNA /START4/ GLOBAL ACCELERATION ARRAY
C IEQCST /START4/ INCREMENTAL NODAL EQUIVALENT PLASTIC
C STRAIN BY LAYER
C IEQPSI /START6/ TOTAL NODAL EQUIVALENT PLASTIC STRAIN BY LAYER
C IETM /START6/ TOTAL NODAL EQUIVALENT PLASTIC STRAIN BY LAYER
C IEXT /START6/
C IFORCE /START4/
C IFORIN /START5/
C IFREQ /SHIFT/ HIGH FREQUENCY BOUNDS FOR POWER SHIFT
C IGEPSNO /START8/ TOTAL GENERALIZED NODAL STRAIN COMPONENTS
C IGEMS /EIGEN/
C IGSINO /START8/ TOTAL GENERALIZED NODAL STRESS COMPONENTS
C IGTONO /START8/
C IICHI /HARM/ HARMONIC NODAL FORCE MAGNITUDE AND PHASE
C IICSN0 /START4/
C IIGENO /START8/ INCREMENTAL GENERALIZED NODAL STRAIN
C COMPONENTS
C IIGSNO /START8/ INCREMENTAL GENERALIZED NODAL STRESS
C COMPONENTS
C IIQMNO /START4/ INCREMENTAL NODAL BACKSTRESS (SHIFT TENSOR)
C COMPONENTS BY LAYER
C IIPSNO /START4/ INCREMENTAL NODAL PLASTIC STRAIN COMPONENTS
C BY LAYER
C IISNNO /START4/ INCREMENTAL NODAL STRAIN COMPONENTS
C FOR SHELL ELEMENTS BY LAYER
C IISTNO /START4/ INCREMENTAL NODAL STRESS COMPONENTS
FOR SHELL ELEMENTS BY LAYER
INCREMENTAL THERMAL STRAIN AT NODES
POINTER FOR THE BOUNDARY CONSTRAINTS
POINTER FOR THE BOUNDARY CONSTRAINT BUFFER
LAST ADDRESS USED IN THE WORKING AREA
LINEPRINTER ID. NUMBER
POINTER TO DIAGONAL COMPONENTS OF GLOBAL
CONSISTENT MASS ARRAY STORED IN PROFILE FORM
NODAL ARRAY FOR LUMPED MASS MATRIX
POINTER TO THE ADDED MASS VALUES
POINTER TO UPPER TRIANGULAR PART OF GLOBAL
CONSISTENT MASS ARRAY STORED IN PROFILE FORM
PERIODIC DISPLACEMENT INPUT ARRAY
ELEMENT CONNECTIVITY
PERIODIC NODAL FORCE INPUT ARRAY

TOTAL NODAL BACKSTRESS (SHIFT TENSOR) COMPONENTS
BY LAYER
PERIODIC DISPLACEMENT PERIOD ARRAY
PERIODIC NODAL FORCE PERIOD ARRAY
FLAG FOR POST FILE GENERATION
FLAG FOR PRESSURE LOADING
FLAG FOR PRINT OPTION BUFFER
TOTAL NODAL PLASTIC STRAIN COMPONENTS
BY LAYER
NODAL EACTION FORCE ARRAY
NODAL RESIDUAL FORCE ARRAY
STRESS BOUNDARY CONDITION INDEX ARRAY
STRESS BOUNDARY CONDITION INPUT BUFFER
POINTER FOR THE LIST OF POWER SHIFT POINTS
NODAL TOTAL STRESS ARRAY
TOTAL NODAL STRESS COMPONENTS FOR
SHELL ELEMENTS BY LAYER
SIZE PARAMETER FOR THE CURRENT WORKING STORAGE
POINTER TO THE ADDED SPRING STIFFNESSES
POINTER FOR NODAL FORCE INTEGER VALUES
POINTER FOR NODAL FORCE REAL VALUES
POINTER ARRAY FOR NODAL TRANSFORMATION
NODAL COORDINATE TRANSFORMATION BUFFER
THERMAL STRAIN (TOTAL) AT NODES

STIFFNESS ARRAY STORED IN PROFILE FORM

NO LONGER USED IN VERSION 1.7 OR UP

HARMONIC BASE MOTION NODE AND D.O.F. LIST
POINTER TO NODAL LIST OF BEAM SECTIONS

POINTER TO THE LIST OF D.O.F. WITH ADDED DAMPING

ELEMENT WORK AREA FOR GENERALIZED INITIAL STRESS
GENERALIZED INITIAL STRESS AT INTEGRATION POINTS
HARMONIC NODAL LOAD NODE AND D.O.F. LIST
ELEMENT WORK AREA FOR CREEP STRAIN INCREMENT
CREEP STRAIN INCREMENT AT INTEGRATION POINTS
ELEMENT WORK AREA FOR GENERALIZED STRAIN
GENERALIZED STRAIN AT INTEGRATION POINTS
GENERALIZED STRESS AT INTEGRATION POINTS
ELEMENT WORK AREA FOR GENERALIZED STRESS
ELEMENT WORK AREA FOR INCREMENTAL PLASTIC STRAIN
INCREMENTAL PLASTIC STRAIN AT INTEGRATION POINTS
ELEMENT WORK AREA FOR INC. STRAIN
ELEMENT WORK AREA FOR INC. STRESS
INCREMENTAL STRAIN AT INTEGRATION POINTS
INCREMENTAL STRESS AT INTEGRATION POINTS
ELEMENT WORK AREA FOR INC. THERMAL STRAIN
INCREMENTAL THERMAL STRAIN AT INTEGRATION POINTS
POINTER TO THE LIST OF D.O.F. WITH ADDED MASS
ELEMENT WORK AREA FOR PLASTIC STRAIN
PLASTIC STRAIN AT INTEGRATION POINTS
POINTER FOR THE NUMBER OF MODES ON EACH SHIFT
POINTER TO THE LIST OF D.O.F. WITH ADDED STIFFNESS

LOW FREQUENCY BOUNDS FOR POWER SHIFT
MAXIMUM NUMBER OF BEAM SECTION PROPERTIES
MAXIMUM NUMBER OF BEAM SECTION PROPERTIES
MAXIMUM NUMBER OF INTEGRATION POINTS / ELEMENT
MAXIMUM NUMBER OF INTEGRATION LAYERS / ELEMENT
MAXIMUM NUMBER OF NODAL D.O.F. / NODE
MAXIMUM NUMBER OF NODAL D.O.F. WITH HARMONIC BASE MOTION IMPOSED
NUMBER OF BOUNDARY CONDITIONS
NUMBER OF BEAM SECTION PROPERTY SETS
NUMBER OF D.O.F. WITH ADDED DAMPING
NUMBER OF DUPLICATED NODAL POINTS
TOTAL NUMBER OF ELEMENTS
NOT USED IN VERSION 2.0
NUMBER OF EIGENVALUES/EIGENVECTORS FOUND IN P-SHIFT
NOT USED IN VERSION 2.0
NUMBER OF D.O.F. WITH HARMONIC NODAL FORCES APPLIED
NOT USED IN VERSION 2.0
NUMBER OF D.O.F. WITH ADDED MASS
C **NUMBER OF NODE IN THE MESH**
C **NUMBER OF OFFSETS FOR NEW EIGENVALUES/EIGENVECTORS**
C **NUMBER OF PRINT OPTIONS**
C **NUMBER OF STRESS BOUNDARY CONDITIONS**
C **NUMBER OF POINTS FOR POWER SHIFT**
C **NUMBER OF D.O.F. WITH ADDED STIFFNESS**
C **NOT USED IN VERSION 1.7 AND UP**
C **NOT USED IN VERSION 1.7 AND UP**
C **SUM OF PROFILE COLUMN HEIGHTS FOR UPPER**
C **TRIANGULAR PART OF GLOBAL ARRAY STORED IN PROFILE FORM**
C **NUMBER OF TYING CONSTRAINT DATA SETS**
C **NUMBER OF NODAL FORCE VECTOR INPUT**
C **NUMBER OF NODAL COORDINATE TRANSFORMATION DATA**
C **FREQUENCY OF EXCITATION FOR HARMONIC BASE MOTION**
C **FREQUENCY OF EXCITATION FOR HARMONIC NODAL LOADS**
C
C IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 WORK
C
C *** -1- **************************** ********************************
C
C COMMON / ADDVAL / ISPRI , KSPRI , IDASH , KDASH , IMASS , KMASS
COMMON / ALGEM / ICREAD , IJPRINT , JLPRINT , ICONSL , IPOSTF , ISCRAP ,
1 IJLOTB , IJSTRT , JCREAD , IPVARS , IPSETS , IFLEX ,
2 PI , IJLINE , IJLINE2
COMMON / CONTRO / IJEND , IJITER , IJTEMP , JPRINT , JP , JSUB ,
1 IJINC , IJREST , IJSAVE , JREDIM , JAUTO , JPOST ,
2 JBACK , JOPTIM , JCREEP , JDIST , JCONST , JDYN ,
3 NONISO , IJTERM , ITRIG , IDYN , JREPO , JTANGE ,
4 JITHERM , JFORCE , JUTEMP , JUDEP , JUHREST ,
5 JDERIV , JUBOUN , IDSTOP , INSTR , JPLAST , JUHOCK ,
6 JFRONT , JDIFOR , JEMBED , ITEST , JDISP , JFRGDS ,
7 IJSCNT , IJLINE , IJPRINT , ICOMP , ICOND , JEC3 ,
8 IJBODY , IJGRAW , IJCENT , JDDAMP , JDYN , JISTAT ,
9 JFDKXX , JISTIF , JECNIM , JFINIT , JUFLX , JUHERN ,
+ JUWESL , JUPRES , JCDUM2 , JCDUM3
COMMON / DAMP / DAMPF(3)
COMMON / EIGEN / IEQNC , IGNMS , IOMEG , IDMDIS , IDNMD , ISTRT2 ,
1 IIPAR , IIPITR , IIPITV , IMDAM , IOMEGD
COMMON / MDSUP / IIMPOR , IMDISO , IMVE1 , IMPO1 , IMDIS1 , IMVEL1
COMMON / HARMON / IOMEG , IJHARM , KHAIR , OMEGB , IBASE , KBASE ,
ICNFOR,ICMFOR,ICMRES,ICHIFN,ICEIFN,ICBEXC,ICMAT

COMMON / PERPAR / IPRT(32),NPRTYP,NEVARS,NPSETS,JPRT,

NPVC0, NPP008, NPP009, NPP010, NPP011, NPP012

COMMON / PERPTR / IMEAN,ISTDEV,IPDATA,IVTYPE,ISKIP,IREDEF,

IPINC0,IREAC0,IRESD0,IPDS0,IPGROUP,ISTIF,IMASS

COMMON / PERDAT / IXDOR,IXCHAR,IXFORC,KXFORC,IXDIST,KXDIST,

IXTEMP,JXTEMP,IXBEAM,IXFVEC,IXSPRI,IXSPRI,

IPRES,IPREF,IP015,IP016,IP017,IP018

COMMON / POWER / IELPHI,IELTNM,IEPSMO,ISIGNM,ITHFN,ITHFC,

IFBP,ISPP,ISFF,ISQQ,ICOQ,ITNM,

IPSF,IPSD

COMMON / PULSES / IPULSE,IPULSE,IPDM,TIPDOR

COMMON / MAXIMA / MAXC,RD,MAC,FMAX,MAXNOD,MAXSTR,MAXCHR,MAXPRS,

MAXLAY,MAXINT,MAXWRK,MAXNLV,NSUMAX,MAXMMP,

MAXBSP,MAXGMR,MAXTEM,MAXX,MAXM,NAXMT,

MAXFRN,MAXBTE,MAXVAR,MAXSET,MAXORD,

MAX025,MAX026,MAX027,MAX028,MAX029

COMMON / PARAM / NTYP,NELEM,NNODE,NCB,NTIE,NMAX,

NTRAN, NTRAC, NP, NBP, NDUMP, NDUMP, NBASE, NINC,

NP, NBASE, NBASE, NBASE, NBASE, NBASE, NINC,

NTER, NPSPT, NPDSPT, NPulse, NPDST, NPDST, NPDST,

NSUMCH, NSUMEN, NSUMNT, NPOR40, NPOR41, NPOR42,

NPOR43, NPOR44, NPOR45, NPOR46, NPOR47, NPOR48

COMMON / TMARCH / DALPHA, BBETA, DGMMA

COMMON / PERIOD / JPERIOD(2),IPDISP,IPFORC,INDISP,INFORC

COMMON / SUBELM / ISUBEL,ISUBNP,ISUBPT,NSDATA,ISUBTY,ITEMBED

COMMON / SHIFT / ISHIFT, KSHIFT, IFREQ, LFREQ, NOFFST, NFOUND

COMMON / SUBTY / NSUCRD, NSUNFR, NSUSTR, NSUCHR, NSUPR,

NSUILL, NSUILL, NSUILL, NSUILL, NSUILL, NSUILL,

COMMON / BSECT / ISBECT, KBSECT

COMMON / START1 / IELPRM, ITPP, INEL, ICHAR, IPRES, ISTRS,

ISTRN, ICP, IPRINT, IPST, IDISP, IEA,

IBP, IBP, IBP, IBP, IBP, IBP, IBP,

COMMON / START2 / INOD, ITMP, MINL, IPSU, ITMDF, IDUP

COMMON / START3 / IREAC, ITM, ITR, ITRAC, IEXT,

ITSC, ITSC, ITSC, ITSC, ITSC, ITSC,

COMMON / START4 / IPINC, IPINC, IPINC, IPINC, IPINC, IPINC,

IEPSNO, IPSTR, ISTRNO, ISTRNO, ISTRNO, ISTRNO,

IIPSTR, IICST, ITTSTR, IPSNO, ICSNO, ICSNO,

IITSNO, ITTST, ITTST, ITTST, ITTST, ITTST,

IDMNO, IDSSC, IDSSC, IDSSC, IDSSC, IDSSC,

IDMNO, IDSSC, IDSSC, IDSSC, IDSSC, IDSSC,
IDYNV, IDYNP, IDXY1, IDXY2, IDSTH, ISWELL, IDQCSI, IPREF, IDXY3, IYIELD, IDFINC, IDPIOT, IST443, IST444, IST445, IST446, IST447, IST448

COMMON / START5 / IRL, IREAC, IES, IAB, IBQM, ISRL, IBTLC, ISKM, ILAST, IRLB, IDINCP, IFORIN, IOP, IDAM, IASMST, IDTAAG, IUPTTR, ICOLFT, IMASDI, IMASUP, IST521, IST522, IST523, IST524

COMMON / START6 / IELV, IOR, ISIG, IEPS, IWNOD, ISNOD, IENOD, IEITM, ICH, IPP, IXLRL, IXIRL, IXP, LK, KPSRN, KCSRN, KISTRN, KITSTR, KIRNL, KISTNO, KISTNO, KISNO, KIPSNO, KICSNO, KITSN0, IMASNO, IMNOD, IEQPST, IEQPSI, KEPST, KEQPSI, KDMAT, KDMINO, KTDSN0, KITDST, IXM, IXC, IVELM, IAELM, IMASEL, KYIELD, IST647, IST648, IST649, IST650, IST651, IST652, IST653, IST654

COMMON / START7 / ICON, IBCHR, ITRACR, ITRANR, IBEAT, IDT

COMMON / START8 / KGEPS, KIGS, KGPST, KGEPS, KIGS, KGTDST, KICSTR, KITSTR, KITSTR, KIPSTR, KICSTR, KIPSTR, KICSTR, KIPSTR, KICSTR, KIPSTR, KICSTR

COMMON / START9 / KEQCSI, KIQMNO, KSWLN0, KIMPN0, KIDFNO, KIDMNY, KEQCSS, KIQMNO, KSWLN0, KIMPN0, KIDFNO, KIDMNY

COMMON / SUBSTR / NLSUB( 10), NFRSUB( 10)

COMMON / MACHIN / IDP

COMMON / ERRORS / IERR

C
C*****************************************************************************
C DIMENSION RWORK ( ISIZE ), IWORK ( ISIZE )
C*****************************************************************************
C DIMENSION YIELDC( 3 )
C*****************************************************************************
C MAXCRD = 1
C MAXNFR = 2
C MAXNOD = 1
C MAXSTR = 1
C MAXCHR = 1
C MAXINT = 2
C MAXPRS = 1
C MAXLAY = 1
C MAXBET = 1
C MAXWRK = 1
C MAXDFT = 1

SYSTEMS' MANUAL

Page : 62

MHOST Version 4.2
MAXNLV = 1
MAXCMP = 1
MAXBSP = 6
MAXLWK = 1

C
MAXELM = NELEM + NDUP + NTIE
C
YIELD(1) = 1.00D+36
YIELD(2) = 0.00D+00
YIELD(3) = 0.00D+00
C

C**********************************************************************

C
MAXNLV = NSUMAX
IELPRM = 1
C ***
IF(NTYPE.EQ.0) GO TO  2
C
C**********************************************************************

C
DO 1 I = 1 ,NTYPE
   IC = I
   IS1 = IELPRM + (IC-1)*13 + 1
C ***
IF(MAXCRD.LT.IWORK(IS1) ) MAXCRD = IWORK(IS1)
IF(MAXNFR.LT.IWORK(IS1+1) ) MAXNFR = IWORK(IS1+1)
IF(MAXNOD.LT.IWORK(IS1+2) ) MAXNOD = IWORK(IS1+2)
IF(MAXSTR.LT.IWORK(IS1+3) ) MAXSTR = IWORK(IS1+3)
IF(MAXCHR.LT.IWORK(IS1+4) ) MAXCHR = IWORK(IS1+4)
IF(MAXINT.LT.IWORK(IS1+5) ) MAXINT = IWORK(IS1+5)
IF(MAXPRS.LT.IWORK(IS1+6) ) MAXPRS = IWORK(IS1+6)
IF(MAXLAY.LT.IWORK(IS1+7) ) MAXLAY = IWORK(IS1+7)
IF(MAXCMP.LT.IWORK(IS1+8) ) MAXCMP = IWORK(IS1+8)
   MVRK = IWORK(IS1+3)* MAXINT
   MAXWRK = MVRK
   MNLV = IWORK(IS1+1)*IWORK(IS1+2)
   MAXNLV = MNLV
   MWBE = IWORK(IS1+1)*IWORK(IS1+2)
   * IWORK(IS1+3)* MAXINT
   IF(MWBE.GT.MAXBET) MAXBET = MWBE
   MLWK = MAXINT *IWORK(IS1+7)
   * IWORK(IS1+8)
   IF(MAXLWK.LT.MLWK) MAXLWK = MLWK
   MDMT = IWORK(IS1+3)*IWORK(IS1+3)
MAXTEM = MAXLAY
ITYP   = ILAST + 1
INEL   = ITYP + MAXELM
ICON   = INEL + MAXNOD * MAXELM
ICOP   = ICON + MAXNOD * MAXELM
IDIST  = ICOP
IF( JFRONT .NE. 0 ) IDIST = ICOP + MAXELM
IF( JFRONT .EQ. 0 ) IDIST = ICOP + NSUMAX * NSUB * 3
IF( JDIST .GT. 0 ) IDIST = IDIST + MAXEU4
ILAST  = IDIST + MAXELM

IF ( NBSECT .EQ. 0 ) GO TO ii0

IBSEC9 = IIAST
KBSECT = IBSEC9 + NBSECT * MAXBSP * IDP
ILAST  = KBSECT + NBSECT
IF ( ILAST.EQ.2*(ILAST/2)) IIAST = ILAST + 1

110 CONTINUE

IF ( NSHIFT .EQ. 0 ) GO TO 120
ISHIFT = IIAST
IFREQ = ISHIFT + NSHIFT * IDP
LFREQ = IFREQ + NSHIFT * IDP
KSHIFT = LFREQ + NSHIFT * IDP
ILAST = KSHIFT + NSHIFT
IF (ILAST.EQ.2*(ILAST/2)) ILAST = ILAST + 1

120 CONTINUE

**********************************************************************
*** CORE ALLOCATION FOR ADDED STIFFNESS, DAMPING AND MASS ***
**********************************************************************

IF (NSPRI .EQ. 0 ) GO TO 132

ISPRI = IIAST
KSPRI = ISPRI + NSPRI * IDP
ILAST = KSPRI + NSPRI * 2

IF (ILAST.EQ.2*(ILAST/2)) ILAST = ILAST + 1

132 CONTINUE

IF (NDASH .EQ. 0 ) GO TO 134

IDASH = IIAST
KDASH = IDASH + NDASH * IDP
ILAST = KDASH + NDASH * 2

IF (ILAST.EQ.2*(ILAST/2)) ILAST = ILAST + 1

134 CONTINUE

IF (NMASS .EQ. 0 ) GO TO 136

IMASS = IIAST
KMASS = IMASS + NMASS * IDP
IIAST = KMASS + NMASS * 2

IF (IIAST.EQ.2*(IIAST/2)) IIAST = IIAST + 1

136 CONTINUE

**********************************************************************
*** CORE ALLOCATION FOR PULSE LOAD DEFINITION ***
**********************************************************************

IF( NPULSE .EQ. 0 ) GO TO 138

IPDTIM = IIAST
IPDFOR = IPDTIM + NPDPTS * IDP
IPULSE = IPDFOR + NPDPPTS * IDP
KPULSE = IPULSE + NPULSE * 2 * IDP
IIAST = KPULSE + NPULSE * 2

IF (IIAST.EQ.2*(IIAST/2)) ILAST = ILAST + 1

138 CONTINUE

***************************
CORE ALLOCATION FOR FLAGS AND COUNTERS ASSOCIATED WITH EACH GLOBAL
ELEMENT FOR THE TREATMENT OF EMBEDDED SINGULARITIES BY MEANS OF
SUBELEMENT MESH REPRESENTATIONS
***************************

IF( JEMBED .GT. 0 )
IEMBED = ILAST + NELEM
ISUBEL = IEMBED
IF (JEMBED .EQ. 0) ISUBNP = ISUBEL + NELEM
ISUBPT = ISUBNP + NELEM
ISUBTY = ISUBPT
NSDATA = 34

***************************
CORE ALLOCATION FOR ELEMENT INTEGRATION POINT WORKING ARRAYS ****
***************************

ICHAR = ISUBTY + MAXELM
IF (ICHAR.EQ.2*(ICHAR/2)) ICHAR = ICHAR + 1
IPRES = ICHAR + MAXCHR * NNODE * IDP
ISIG = IPRES + 2 * MAXPRS * NNODE * IDP
IEPS = ISIG + MAXLWK * IDP
KPSTRN = IEPS + MAXLWK * IDP
KCSTRN = KPSTRN + MAXLWK * IDP
KTSTRN = KCSTRN + MAXLWK * IDP
KDMAT = KTSTRN + MAXLWK * IDP
KGEPS = KDMAT + MAXDMT * IDP
KIGEPS = KGEPS + MAXWRK * IDP
KGSIG = KIGEPS + MAXWRK * IDP
KIGSIG = KGSIG + MAXWRK * IDP
KGTSTS = KIGSIG + MAXWRK * IDP
KISTRS = KGTSTS + MAXWRK * IDP
ILAST = KISTRS + MAXCHR * IDP * MAXNOD

IPRINT = ILAST
IMONIT = IPRINT + 12 * NPRINT
IPOST = IMONIT + 4 * NMONT
ILAST = IPOST
IF (ILAST.EQ.2*(ILAST/2)) ILAST = ILAST + 1

C **********************************************************************
C *** CORE _IC_ FOR NODAL QUANTITIES ***
C **********************************************************************
C
INOD = IIAS
IWINOD = INOD + MAXCRD * NNODE * IDP
IBPRES = IWINOD + NNODE * IDP
IBNORM = IBPRES + JPRES * NNODE * IDP
ISIGNO = IBNORM + JPRES * NNODE * IDP * 3
IEPSNO = ISIGNO + MAXCMP * NNODE * IDP * MAXLAY
IPSTNO = IEPSNO + MAXCMP * NNODE * IDP * MAXLAY
ICSTNO = IPSTNO + MAXCMP * NNODE * IDP * MAXLAY
ITSTNO = ICSTNO + MAXCMP * NNODE * IDP * MAXLAY
IISTNO = ITSTNO + MAXCMP * NNODE * IDP * MAXLAY
IISNNO = IISTNO + MAXCMP * NNODE * IDP * MAXLAY
IIPSNO = IISNNO + MAXCMP * NNODE * IDP * MAXLAY
IICSNO = IIPSNO + MAXCMP * NNODE * IDP * MAXLAY
IIITSNO = IICSNO + MAXCMP * NNODE * IDP * MAXLAY
ITDSNO = IIITSNO + MAXCMP * NNODE * IDP * MAXLAY
IOMENO = ITDSNO + MAXCMP * NNODE * IDP * MAXLAY
IIOINO = IOMENO + MAXCMP * NNODE * IDP * MAXLAY
IGEPSNO = IIOINO + MAXCMP * NNODE * IDP * MAXLAY
IIGENO = IGEPSNO + MAXSTR * NNODE * IDP
IGSINO = IIGENO + MAXSTR * NNODE * IDP
IIGSINO = IGSINO + MAXSTR * NNODE * IDP
IGDINO = IIGSINO + MAXSTR * NNODE * IDP
I IMDINO = IGDINO + MAXSTR * NNODE * IDP
IPREF = IMDINO + MAXSTR * MAXSTR * NNODE * IDP
IYIELD = IPREF + NNODE * 3 * NONISO * IDP
IMASNO = IYIELD + NNODE * 3 * NHARD * IDP

C ***
IEQPSI = IMASNO + NNODE * IDP
IEQPSST = IEQPSI + NNODE * IDP * MAXLAY
IEQCST = IEQPSST + NNODE * IDP * MAXLAY
IEQCSI = IEQCST + NNODE * IDP * MAXLAY
IVSWTO = IEQCSI + NNODE * IDP * MAXLAY
ISWELL = IVSWTO + NNODE * IDP * MAXLAY
ITEM = ISWELL + NNODE * IDP * MAXLAY
ITEMDF = ITEM + NNODE * IDP * MAXLAY
INLV = ITEMDF + NNODE * IDP * MAXLAY

IF( JLARGE.NE.999999 ) THEN

    LARGE DEFORMATION ANALYSIS

    IDFTOT = INLV + NNODE*MAXNFR
    IDFINC = IDFTOT + NNODE*NDIMEN*NDIMEN*IDP
    ILAST = IDFINC + NNODE*NDIMEN*NDIMEN*IDP

ELSE

    SMALL DEFORMATION ANALYSIS

    ILAST = INLV + NNODE*MAXNFR

ENDIF

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

*** ALLOCATE STORAGE FOR BOUNDARY CONDITIONS ***
******************************************************************************

IKBC = ILAST
IKBCR = IKBC + 3 * NBC
IF (IKBCR.EQ.2*(IKBCR/2)) IKBCR = IKBCR + 1

IEXT = IKBCR + NBC * IDP
ITI = IEXT + 3 * NEXT
ITR = ITI + 3 * NTIE * (NMAX+1)
IF (ITR.EQ.2*(ITR/2)) ITR = ITR + 1

ITRAN = ITR + NTIE * NMAX * IDP
ITRANR = ITRAN + 3 * NTRAN
IF (ITRANR.EQ.2*(ITRANR/2)) ITRANR = ITRANR + 1

ITRAC = ITRANR + NTRAN * IDP
ITRACR = ITRAC + 4 * NTRAC
IF (ITRACR.EQ.2*(ITRACR/2)) ITRACR = ITRACR + 1

ISBC = ITRACR + 2 * NTRAC * IDP
ISBCR = ISBC + 2 * NSBC
IF(ISBCR.EQ.2*(ISBCR/2)) ISBCR= ISBCR+1
ILAST = ISBCR + NSBC * IDP

IDUP = ILAST
ILAST = IDUP + 2 * NDUP
IF (ILAST.EQ.2*(ILAST/2)) ILAST = ILAST + 1

*** IF HARMONIC (COMPLEX) NODAL LOADS ARE SPECIFIED, ADD... ***********

IHARM = ILAST
KHARM = IHARM + 2 * NHARM * IDP
ILAST = KHARM + 2 * NHARM
IF (ILAST.EQ.2*(ILAST/2)) ILAST = ILAST + 1

*** IF HARMONIC (COMPLEX) BASE EXCITATIONS ARE SPECIFIED, ADD... *****

IBASE = ILAST
KBASE = IBASE + 2 * NBASE * IDP
ILAST = KBASE + 2 * NBASE
IF (ILAST.EQ.2*(ILAST/2)) ILAST = ILAST + 1

***

IGNMS = ILAST
IOMEG = IGNMS + 2 * NSBNC * IDP
ILAST = IOMEG + 2 * NSBNC * IDP

******************************************************************************
*** CORE ION FOR MODAL SUPERPOSITION ANALYSIS ******
******************************************************************************

IF ( LDYN .NE. 2 .AND. NPSETS .EQ. 0 ) GO TO 180

IMDAM = ILAST
IOMEGD = IMDAM + NDYNMD * IDP
IMFOR0 = IOMEGD + NDYNMD * IDP
IMDIS0 = IMFOR0 + NDYNMD * IDP
IMVEL0 = IMDIS0 + NDYNMD * IDP
IMFOR1 = IMVEL0 + NDYNMD * IDP
IMDIS1 = IMFOR1 + NDYNMD * IDP
IMVEL1 = IMDIS1 + NDYNMD * IDP
ILAST = IMVEL1 + NDYNMD * IDP

IF ( NHARM .EQ. 0 .AND. NBASE .EQ. 0 ) GO TO 180

SYSTEMS' MANUAL

Page : 69
ICMFOR = ILAST
ICMRES = ICMFOR + 2 * NSUPER * IDP
ICCHFN = ICMRES + 2 * NSUPER * IDP
ICBHFN = ICHHFN + 2 * NSUPER * IDP
ICCMAT = ICBHFN + 2 * NSUPER * IDP
ICBEXC = ICCMAT + 2 * NBASE * IDP * 2 * NBASE * IDP
ILAST = ICBEXC + 2 * NBASE * IDP

180 CONTINUE

C **********************************************************************
C *** CORE ALLOCATION FOR FREQUENCY DOMAIN ANALYSIS ***
C **********************************************************************

C IF ( LDYN .NE. 4 ) GO TO 280

IMDAM = ILAST
IHFN = IMDAM + NSUPER * IDP
IHFC = IHFN + NSUPER * 2 * IDP
IPSF = IHFC + NSUPER * 2 * IDP
IPSD = IPSF + NPSPTS * IDP
IFBP = IPSD + NPSPTS * IDP
ISPP = IFBP + NPNTS * IDP
ISFF = ISPP + NSUPER * NSUPER * IDP
ISQQ = ISFF + NSUPER * NSUPER * IDP
ICQQ = ISQQ + NSUPER * NSUPER * IDP * 2
ISIGMO = ICQQ + NSUPER * NSUPER * IDP * 2
IEPSMO = ISIGMO + NSUPER * NNODE * IDP * MAXCMP * MAXLAY
IELPHI = IEPSMO + NSUPER * NNODE * IDP * MAXCMP * MAXLAY
IELTNM = IELPHI + MAXNOD * IDP * MAXNFR
ITNM = IELTNM + MAXNOD * IDP
ILAST = ITNM + NSUPER * NNODE * IDP

280 CONTINUE

C **********************************************************************
C *** CORE ALLOCATION FOR ELEMENT NODE WORKING STORAGE ***
C **********************************************************************

ICOLPT = ILAST
IELV = ICOLPT + MAXNFR*NNODE
ICOR = IELV + MAXNLV
IF (ICOR.EQ.2*(ICOR/2)) ICOR = ICOR+1

IEIM = ICOR + MAXNOD * MAXCRD * IDP
ICH = IETM + MAXLAY * MAXNOD * IDP
IPP = ICH + MAXCHR * IDP
IWINOD = IPP + MAXPRS * IDP
ISNOD = IWINOD + MAXCHR * IDP
IF(ISNOD.EQ.2*(ISNOD/2)) ISNOD = ISNOD + 1
IENOD = ISNOD + MAXCMP * MAXNOD * IDP * MAXLAY
KPSTNO = KSTNO + MAXCMP * MAXNOD * IDP * MAXLAY
KSTNO = KPSTNO + MAXCMP * MAXNOD * IDP * MAXLAY
KDTNOD = KSTNO + MAXCMP * MAXNOD * IDP * MAXLAY
KDSNO = KDTNOD + MAXNOD * IDP * MAXLAY
KEQPST = KEPST + MAXNOD * IDP * MAXLAY
KEQPSI = KEQPST + MAXNOD * IDP * MAXLAY
KEQCSI = KEQPSI + MAXNOD * IDP * MAXLAY
KIMNO = KIMNO + MAXSTR * MAXNOD * IDP
KSWLNO = KIMNO + MAXSTR * MAXNOD * IDP
KIPNNO = KIPNNO + MAXSTR * MAXNOD * IDP
KMDNO = KMDNO + MAXSTR * MAXNOD * IDP
KAEPLNO = KAEPLNO + MAXSTR * MAXNOD * IDP
KIGENO = KIGENO + MAXSTR * MAXNOD * IDP
KGSINO = KGSINO + MAXSTR * MAXNOD * IDP
KGTNO = KGTNO + MAXSTR * MAXNOD * IDP
KYIELD = KYIELD + NHARD * MAXNOD * IDP * 3
IXRL = IXRL + MAXNLV * IDP
IXIRL = IXIRL + MAXNLV * IDP
IXEIM = IXEIM + MAXNLV * IDP
IXELM = IXELM + MAXNLV * IDP * IDYNM
IXP = IXP + MAXNLV * IDP * IDYNM
IXK = IXK + MAXNLV * IDP * IDYNM
IDET = IDET + MAXNLV * IDP * IDYNM
IXM = IXM + MAXINT * IDP
IXC = IXC + MAXNLV * MAXNLV * IDP * IDYNM
IBETA = IBETA + MAXNLV * MAXNLV * IDP * IDYNM
IDTOUT = IDTOUT + MAXBET * IDP
IDYN = IDYN + NNODE * MAXNFR * IDP
IDYNV = IDYN + NNODE * MAXNFR * IDP * IDYNM
IRL = IRL + NNODE * MAXNFR * IDP * IDYNM
C *** PERIODIC LOADING ARRAYS ****************************************
C
NPDISP = JPEROD( 1 )
NPFORC = JPEROD( 2 )
C
IPDISP = ILAST
IPFORC = IPDISP + NPDISP* IDP *IDYNM
INDISP = IPFORC + NPFORC* IDP *IDYNM
INFORC = INDISP + NPDISP *IDYNM
ILAST = INFORC + NPFORC *IDYNM
C
IF ( ILAST.EQ.2*(ILAST/2)) ILAST = ILAST + 1
C
IF ( NHARM .EQ. 0 .AND. NBASE .EQ. 0 ) GO TO 320
ICNFOR = ILAST
ILAST = ICNFOR + NNODE * MAXNFR * IDP * 2
320 CONTINUE
C
IRL = IIAST
IFORCE = IRL + NNODE * MAXNFR * IDP * 2
IF( IFSCNT .NE. 0 ) IFORCE = IFORCE + NNODE * MAXNFR * IDP
IDINC = IFORCE + NNODE * MAXNFR * IDP
IREAC = IDINC + NNODE * MAXNFR * IDP * 2
IF( IPCONJ .NE. 0 ) IREAC = IREAC + NNODE * MAXNFR * IDP
IF( IFSCNT .NE. 0 ) IREAC = IREAC + NNODE * MAXNFR * IDP
IRLB = IREAC + NNODE * MAXNFR * IDP
IDINCP = IRLB + NNODE * MAXNFR * IDP
IFORIN = IDINCP + NNODE * MAXNFR * IDP
IBTLC = IFORIN + NNODE * MAXNFR * IDP
IEGNVC = IBTLC
IBTLC = IEGNVC + NSBNC * NNODE * MAXNFR * IDP *2
IDSITR = IBTLC + NNODE * MAXNFR * IDP *IDYNM
IDSX1 = IDSITR + NNODE * MAXNFR * IDP *IDYNM
IDSX2 = IDSX1 + NNODE * MAXNFR * IDP
IDSX3 = IDSX2 + NNODE * MAXNFR * IDP *IDYNM
ILAST = IDSX3 + NNODE * MAXNFR * IDP *IDYNM
C
C *************************************************************
C *** CORE ALLOCATION FOR PERTURBATION DATA SETS ************
C *************************************************************
C
ISKIP = ILAST
IREDEF = ISKIP + NPSETS + 1
ILAST = IREDEF + NPVARS * IDP
IF ( ILAST .EQ. 2*(ILAST /2)) ILAST = ILAST + 1

IF ( NPSETS .EQ. 0 ) GO TO 680

*** ARRAYS NEEDED IN ANY TYPE OF PERTURBATION ANALYSIS ***************

IMEANS = ILAST
ISTDEV = IMEANS + NPVARS*IDP
IPDATA = ISTDEV + NPVARS*IDP
IVTYPE = IPDATA + NPVARS*NPSETS*IDP
ILAST = IVTYPE + NPVARS
IF ( ILAST .EQ. 2*(ILAST /2)) ILAST = ILAST + 1

*** ARRAYS USED FOR THE PERTURBED STATIC PROBLEM **********************

IDINC0 = ILAST
IREAC0 = IDINC0 + NNODE*MAXNFR*IDP
IRESD0 = IREAC0 + NNODE*MAXNFR*IDP
ILAST = IRESD0 + NNODE*MAXNFR*IDP

*** ARRAYS USED FOR THE PERTURBED EIGENVALUE PROBLEM *******************

IDGRP = ILAST
ILAST = IDGRP + 2*NSETN

*** PERTURBATION WORKSPACE USED FOR VARIABLE MANIPULATIONS *************

IPWBEG = ILAST
IXCOOR = IPWBEG
IXCHAR = IXCOOR + IPTYPE( 1)*NNODE*MAXCRD*IDP
IXFORC = IXCHAR + IPTYPE( 2)*NNODE*MAXCHR*IDP
KXFORC = IXFORC + IPTYPE( 3)*NTRAC*2*IDP
IXDIST = KXFORC + IPTYPE( 3)*NTRAC*4
IF (IXDIST .EQ. 2*(IXDIST/2)) IVDIST = IVDIST+1
KXDIST = IVDIST + IPTYPE( 4)*NNODE*2*MAXPRS*IDP
IXTEMP = KXDIST + IPTYPE( 4)*NELEM
IF (IXTEMP .EQ. 2*(IXTEMP/2)) IEXTMP = IEXTMP+1
JXTEMP = IEXTMP + IPTYPE( 5)*NNODE*MAXLAY*IDP
IXBEAM = JXTEMP + IPTYPE( 5)*NNODE*MAXLAY*IDP
IXSPR = IXBEAM + IPTYPE( 6)*NBSECT*MAXBSP*IDP
KXSPI = IXSPR + IPTYPE( 7)*NSPRI*IDP
IXFV = KXSPI + IPTYPE( 7)*NSPRI*3
IXVF = IXFV + IPTYPE( 8)*NSPRI*IDP
IXFVEC = IXVF + IPTYPE( 9)*NSPRI*IDP
IF (IXFVEC .EQ. 2*(IXFVEC/2)) IXFVEC = IXFVEC+1
IXPRES = IXFVEC + IPTYPE( 3)*NNODE*MAXNFR*IDP
IXPREF = IXPRES + IPTYPE( 8)*NNODE*IDP
IPWEND = IXREF + IPTYPE( 9 ) * NNODE * 3 * IDP
LAST = IPWEND
680 CONTINUE

C
IBTLC = LAST
ISTRM2 = IBTLC
C
*** MESSAGE OUTPUT AND RETURN TO DATA INPUT ROUTINES ***************
C
LAST1 = LAST + 2 * NNODE + 16
NPX = LAST1 - ITYP + 1
NSHORT = LAST1 - ISIZE
C
CALL LINES(4,4)
WRITE(ILPRINT,2001) LAST1
2001 FORMAT(//,2X,47HNUMBER OF WORDS NEEDED IN BLANK COMMON FOR DATA, *6H INPUT, I8, //, 2X, 43H******************************************************, *18H******************************* )
   IF(NSHORT .GT. 0) GO TO 900
C
C ******************************************************
C ** INITIALIZE DATA IN BLANK COMMON AND CONTINUE EXECUTION **
C ******************************************************
C
CALL NULINT( IWORK(ITYP), NPX )
C
SPECIAL INITIALIZATION FOR YIELD FUNCTION ARRAY
C
ISTART = IYIELD
IEND = IYIELD + NNODE * 3 * NHARD * IDP
ISTEP = 3 * NHARD * IDP
C
DO 800 INDEX = ISTART, IEND, ISTEP
   CALL COPY( YIELDC, 3 )
800 CONTINUE
C
RETURN
C
*************************************************************************
C ** PRINT-OUT THE AMOUNT OF CORE REQUIRED AND TERMINATE EXECUTION **
C*************************************************************************
C
900 CONTINUE
   IERR = IERR + 1
   CALL LINES( 3, 3 )
WRITE(1,1205) NSHORT
1205 FORMAT('INCREASE BLANK COMMON AND ISIZE WITH',18,' WORDS')
C
CALL QUIT('MEMD','RY A','LOC','ATIO','N',',',IERR)
C
**********************************************************************
C
END
C
C The following SUBROUTINE INIT2 calculates pointers for the global arrays:

C=SUBROUTINE=INITI2 CALLED BY SUBROUTINE 'STRUCT'
C SUBROUTINE INITI2(RWORK,IWORK,ISIZE)
C
C **********************************************************************
C
C CORE FOR GLOBAL STIFFNESS
C
C **********************************************************************
C
IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 RWORK
C
COMMON / ALGEM / ICREAD,ILPRNT,JLPRNT,IONS,L,IPRF,JPOST,ISCRAF,
1 IPLOT,JLPRNT,IPRINT,JP,JSUB,
2 PI,LINE,LINE2
COMMON / CONTRO / JEND ,JITER ,JTEMP ,JPRINT ,JP ,JSUB ,
1 JINC ,JRES ,JSAVE ,JREDIM,JAUTO ,JPOST ,
2 JBACK ,JPOSTM ,JCRETEP ,JDIST ,JCONST ,JERYN ,
3 JNOISO ,JISER ,JTRIG ,JENM ,JREPOT ,JTANGE ,
4 JHERM ,JFORCE ,JUTEMP ,JUEROE ,JDMS ,JUHCK ,
5 JDERIV ,JUBOUND ,JSTOP ,JINT ,JPLSUB ,JBASE ,
6 JFRONT ,JDEFOR ,JEMBED ,JTEST ,JDISP ,JFBFGS ,
7 IPSCAT ,IPLINE ,JPRNT ,ICOMP ,IPCONJ ,JEISEN ,
8 JBODY ,JGRAV ,JECENT ,JDAMP ,JDEY ,JISTAT ,
9 JFDSXX ,JISTIF ,JCENTM ,JFINIT ,JLARGE ,JFOLOW ,
JWKSLL ,JIPRES ,JCDUM2 ,JCDUM3
COMMON / PERPAR / IPTYPE(32),NPRTYPE,NPVARS,NPSETS,JPERT ,
1 NPVCN ,NPPOO8 ,NPPOO9 ,NPPOO10 ,NPPOO11 ,NPPOO12

NESSUS
NESSUS
**COMMON / PERPTR** / IMEANS, ISTDEV, IPDATA, IVTYPE, ISKIP, IREDEF, NESSUS
1 IDINCO, IREACO, IRESDO, IDGRF, ISTIFO, IMASS0, NESSUS
2 IPOP13, IOMEGO, IOMEGP, IOMEGK, IPEAK, IPEAK, NESSUS
**COMMON / PERIOD** / JPEROD( 2 ), IPDISP, IPFORC, INDISP, INFORC
**COMMON / EIGEN** / IBOGNC, IIGNMS, IOMEG, IMENO, IDYNMD, ISTRP2,
1 IPTAR, IPTER, IPTVED, IDAM0, IOMEG0
**COMMON / ERRORS** / IERR
**COMMON / MAXIMA** / MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
1 MAXLAY, MAXINT, MAXWRK, MAXNLV, NSUMAX, MAXCMP,
2 MAXBSP, MAXGR, MAXTEM, MAXELM, MAXLW, MAXMT,
3 MAXFRN, MAXBET, MAXVAR, MAXSET, MAXEA, MAXORD,
4 MAX025, MAX026, MAX027, MAX028, MAX029, MAX030
**COMMON / PARAM** / NTYPE, NELEM, NNODE, NBC, NITE, NMAX,
1 NIRAN, NTRAC, NFDR, NNBAND, NEXT, NSUB,
2 NPRINT, NPOST, NSBC, NDUP, NSIZE, NSECT,
3 NSHIFT, NSBFGS, NGMRS, NSPI, NMASS, NDASH,
4 NDYNMD, NSBEC, NSUPER, NHARM, NBASE, NINC,
5 NITER, NPSPTS, NFDPPTS, NPULE, NPDPPTS, NHIARD,
6 NSUMCH, NDIMEN, NMINT, NP401, NP41, NP42,
7 NP43, NP444, NP445, NP446, NP447, NP448
**COMMON / START1** / IELPRM, ITYP, INEL, ICHAR, IPRES, ISTRS,
1 ISTRN, IOCP, IPRINT, IPPOST, IDIST, ILEAN,
2 IPRES, IBNORM, IMONIT, IST116, IST117, IST118
**COMMON / START3** / IBRC, ITI, ITR, ITIRAN, ITIRAC, IEXT,
1 ISBC, ISBCR
**COMMON / START5** / IRL, IREAC, IES, IAB, IBOM, ISRL,
1 IBHLC, ISKM, ILAST, ISRL, IDINCP, IPORIN,
2 IOP, IDAMMT, IMASMT, IGRAPH, IUPTRI, ICOND,
3 IMASDI, IMASUP, IST521, IST522, IST523, IST524
**COMMON / START4** / IDINC, IDIOT, IPORC, IRESID, IMNOD, ISIGNO,
1 ITEPSNO, IPSTRN, ICSTRN, ITSTRN, ISTRS, ISTRN,
2 ITIPRC, ICSSTR, ITSRS, IPSNO, ICSTNO, ITSTNO,
3 IISNO, IISNO, IPSTN, IICSN, ITCSNO, IDMAT,
4 ILMN, IECQST, IOMENO, IONM, IODSN, IVSWTO,
5 IDYNV, IDYNA, IDX, IEXI, IEXI, ISTR, ISISW,
6 IEQCSI, IPREF, IDX2, IYIELD, IDFINC, IDFTOT,
7 IST443, IST444, IST445, IST446, IST447, IST448
**COMMON / MACHIN** / IDP
C
C *** CONSTRUCT THE REVERSE CONNECTIVITY TABLE ***************************
C
C IBEG = INEL
IEND = IBEG + NELEM*MAXNOD
C
C ... FIND THE MAXIMUM NUMBER OF ELEMENTS AT A NODE
C
MAXEAN = 0
DO 130 NN = 1, NNODE
   NF = 0
   DO 120 IN = IBEG, IEND
      IF ( IWORK(IN) .EQ. NN ) NF = NF+1
   CONTINUE
   IF ( MAXEAN .LT. NF ) MAXEAN = NF
130 CONTINUE
C
C ... ALLOCATE SPACE FOR THE REVERSE CONNECTIVITY TABLE
C
ILEAN = ILAST
ILAST = ILEAN + NNODE*MAXEAN
C
CALL NULIN( IWORK(ILEAN), MAXEAN*NNODE )
C
C ... CONSTRUCT THE REVERSE CONNECTIVITY TABLE
C
DO 160 NN = 1, NNODE
   NF = 0
   IN = INEL-1
   DO 150 IE = 1, NELEM
      DO 140 IJ = 1, MAXNOD
         IN = IN+1
         IF ( IWORK(IN) .EQ. NN ) THEN
            NF = NF+1
            IM = ILEAN+(NN-1)*MAXEAN+NF-1
            IWORK(IM) = IE
         ENDIF
      CONTINUE
   CONTINUE
160 CONTINUE
C
C**********************************************************************
C
IADRES = ILAST
IBILC = ILAST
ISKM = IBILC + NEXT*NPD*IDP
IBQM = ISKM + NEXT+NEXT*IDP
ISRL = IBQM + NEXT*IDP
IAB = ISRL + NEXT*IDP
ILAST = IAB + (4*NBAND + 2)*IDP
IF( MOD( ILAST, 2 ) .EQ. 0 ) ILAST = ILAST + 1

C
C **********************************************************************
C
IDIAG = IIAST
IF ( IDIAG .EQ. 2*(IDIAG/2) ) IDIAG = IDIAG + 1
IUPTRI = IDIAG + NFD*IDP

C
C
IH = ( NFD + NSUMCH ) * IDP
NSIZE = NFD + NSUMCH

C
ILAST = ILAST + IH
IOP = ILAST
IF(MOD(IOP,2) .EQ. 0) IOP = IOP + 1

C
IF( JDYN .EQ. 1 ) THEN
IESIZE = ( MAXNFR * MAXMOD )**2 * IDP
IDAMMT = IOP
IMASMT = IDAMMT + IESIZE
ILAST = IMASMT + IESIZE
END IF

C
IF( JEIGEN .EQ. 1 ) THEN
IMASDI = IOP
IMASUP = IMASDI + NFD * IDP
IPTAR = IMASDI + IH
IPTBR = IPTAR + NSBNC * NFD * IDP
IPTVED = IPTBR + NSBNC * NFD * IDP
ILAST = IPTVED + NSBNC * NFD * IDP
END IF

C
IF( IFBFGS .EQ. 1 ) THEN
IPTAR = IOP
IPTBR = IPTAR + NFD * NSBFGS * IDP
ILAST = IPTBR + NFD * NSBFGS * IDP
END IF

C
*** ADDITIONAL ALLOCATION FOR THE EIGENPROBLEM PERTURBATION **********
C
IF ( JEIGEN .EQ. 0 .OR. NPSETS .EQ. 0 ) GO TO 300

C
ISTIF0 = ILAST
IMASS0 = ISTIF0 + IH
IOME0 = IMASS0 + IH
IOMEGP = IOME0 + NDYNMD * IDP
ILAST = IOMEGP + NDYNMD * IDP

C ... CHECK HOW MANY PERTURBATION TERMS THERE IS ROOM IN CORE FOR
C
ITERM = (1+2*NFD)*NDYNMD*IDP
ILEFT = ISIZE-ILAST
IF ( ILEFT .LE. 0 ) ILEFT = 0
MAXORD = ILEFT/ITERM
IF ( MAXORD .GT. (NITER+2)) MAXORD = NITER+2
IF ( MAXORD .LE. 2 ) MAXORD = 3

C ... IN EITHER CASE GO ON AND ESTIMATE THE MINIMAL REQUIREMENTS
C
IOMEGK = ILAST
IETAK = IOMEGK + NDYNMD*MAXORD*IDP
IZETAK = IETAK + NFD*NDYNMD*MAXORD*IDP
ILAST = IZETAK + NFD*NDYNMD*MAXORD*IDP

CONTINUE

C *** CHECK AGAINST THE SIZE OF BLANK COMMON AVAILABLE ***************
C
CALL LINES ( 7 , 7 )
WRITE(ILPRNT,2000) NFD , NBAND
2000 FORMAT(//,2X,21HDATA ON SYSTEM MATRIX,/, *
  2X,21H************************************,//,
  * 5X,16HNUMBER OF D.O.F.,I15,/, *
  5X,16HTRANS.BANDWIDTH ,I15)

CALL LINES ( 4 , 4 )
LAST1 = ILAST + 16
NPX = ILAST1 - IADRES + 1
WRITE(ILPRNT,2001) ILAST1
2001 FORMAT(//,2X,44HTOTAL NUMBER OF WORDS NEEDED IN BLANK COMMON,I8,/, *
2X,52H******************************) NSHORT = ILAST1 - ISIZE
IF ( NSHORT .LE. 0 ) GO TO 200

C
C *** WORKING STORAGE NOT SUFFICIENT !
C
IERR = IERR + 1
CALL LINES ( 3 , 3 )
WRITE(ILPRNT,2005) NSHORT
2005 FORMAT(//,2X,35HINCREASE BLANK COMMON AND IBLK WITH,18,6H WORDS)
CALL QUIT('MEMD','RY A','LOC','ATIO','N ',' ',IERR)

C
C ***
C
200 CONTINUE
CALL NULLINT ( IWORK(IADRES) , NPX )
RETURN
END

The following SUBROUTINE INITSE defines the pointers for the subelement data storage:

C=SUBROUTINE=INITSE CALLED BY SUBRoutines 'SUBEIN','SUBDIV'
SUBROUTINE INITSE
1 ( IWORK ,RWORK ,ISIZE ,KEMBED,KSUBEL,KSUBNP,KSUBTY,NNODE ,
2 NELEM,NHARD ,IERR ,IPOINT,NSDATA,IELEM ,ILAST ,IDP )
C
C **********************************************************************
C
IMPLICIT REAL*8 ( A-H , O-Z )
REAL*4 RWORK
C
C **********************************************************************
C
DIMENSION IWORK ( ISIZE ) ,RWORK ( ISIZE)
DIMENSION IPOINT(NSDATA)
C
C **********************************************************************
C
COMMON / SUBTYP / NSUCRD,NSUNFR,NSUNOD,NSUSTR,NSUCHR,NSUCMP,
1 NSUIN,NSULV ,NSULAY,NSUNDI,NSUSR,NSUIDF
COMMON / ALGEM / ICREAD,ILPRNT,JLPRNT,ICONSL,IPOSTF,ISCRAF,
1 IPLOTB,IRSTRT,JCREAD,IPVARS,IPSETS,IFILEX,
2 PI ,LINE ,LINE2
C
C **********************************************************************
C
FUNCTION
C
--------
C
ALLOCATES CORE STORAGE FOR THE SUBELEMENT MESH ASSOCIATED
C WITH THE IEELM-TH MASTER ELEMENT
ARGUMENTS

----
IWORK  INTEGER WORKING SPACE
RWORK  SINGLE PRECISION REAL WORK SPACE
ISIZE  TOTAL SIZE OF THE WORKING ARRAY
KEMBED THE FLAGS INDICATING THE PRESENCE OF SUB-ELEMENT DIVISION
= 1 FOR UNIFORM 2X2 SUBELEMENT GRID
= 10 FOR USER DEFINED SUBELEMENT GRID
< 0 SUBELEMENT MESH DEFINED BUT THE CORE IS NOT ALLOCATED
KSUBEL NUMBER OF SUBELEMENTS
KSUBNP NUMBER OF NODES IN THE SUBELEMENT NODES
KSUBTY SUBELEMENT TYPE
NNODE NUMBER OF NODES IN THE GLOBAL MESH
NELEM NUMBER OF ELEMENTS IN THE GLOBAL MESH
IERR ERROR COUNTER ( IF IERR .GT. 0 ) EVENTUALLY THE JOB SHOULD QUIT
IPOINT ARRAY FOR SUBELEMENT ARRAY POINTERS
NSDATA NUMBER OF ENTRIES PER ELEMENT FOR THE SUBELEMENT POINTER ARRAY
IELEM COUNTER FOR THE CURRENT ELEMENT
ILAST LAST ADDRESS OF THE WORKING STORAGE ( UPDATED IN THIS ROUTINE )
IDP RATIO OF THE REAL / INTEGER WORD LENGTH

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

POinters

----
IPOINT( 1 ,IELEM)  I  ARRAY

1  ELEMENT CONNECTIVITY
2  PHYSICAL COORDINATES
3  ELEMENT COORDINATES
4  TOTAL NODAL DISPLACEMENT
5  INCREMENTAL NODAL DISPLACEMENT
6  CORRECTIONAL NODAL DISPLACEMENT
7  TOTAL NODAL STRAIN
8  TOTAL NODAL STRESS
9  TOTAL NODAL PLASTIC STRAIN
10  TOTAL NODAL CREEP STRAIN
11  TOTAL NODAL THERMAL STRAIN
12  GENERALIZED NODAL STRESS (TOTAL)
13  GENERALIZED NODAL STRAIN (TOTAL)
C 14 I INCREMENTAL NODAL STRAIN
C 15 I INCREMENTAL NODAL STRESS
C 16 I INCREMENTAL NODAL PLASTIC STRAIN
C 17 I INCREMENTAL NODAL CREEP STRAIN
C 18 I INCREMENTAL NODAL THERMAL STRAIN
C 19 I GENERALIZED NODAL STRESS (INCREMENTAL)
C 20 I GENERALIZED NODAL STRAIN (INCREMENTAL)
C 21 I LUMPED MASS COEFFICIENT FOR SMOOTHING
C 22 I NODAL MATERIAL TANGENT
C 23 I NODAL TEMPERATURE
C 24 I TEMPERATURE DIFFERENCE
C 25 I EQUIVALENT PLASTIC STRAIN (TOTAL)
C 26 I EQUIVALENT PLASTIC STRAIN (INCREMENT)
C 27 I EQUIVALENT CREEP STRAIN (TOTAL)
C 28 I EQUIVALENT CREEP STRAIN (INCREMENT)
C 29 I TOTAL SHIFT TENSOR
C 30 I INCREMENTAL SHIFT TENSOR
C 31 I TVSEWL
C 32 I SWELL
C 33 I GENERALIZED INITIAL STRESS
C 34 I YIELD FUNCTION DEFINED AT NODES
C
C **********************************************************************
C IF( KEMBED .EQ. 0 ) RETURN
C
WRITE(ICONSL,2000) IELEM
2000 FORMAT(' _COREAI/IDCATIONFORELEMENTNO.',I5)
C
CALL SUBELV
1 (IWORK,KSUBTY,IERR )
C
IPOINT( 1 ) = ILAST
IPOINT( 2 ) = IPOINT( 1 ) + NSUNOD * KSUBEL
IPOINT( 3 ) = IPOINT( 2 ) + NSUCRD * KSUBNP * IDP
IPOINT( 4 ) = IPOINT( 3 ) + NSUCRD * KSUBNP * IDP
IPOINT( 5 ) = IPOINT( 4 ) + NSUNFR * KSUBNP * IDP
IPOINT( 6 ) = IPOINT( 5 ) + NSUNFR * KSUBNP * IDP
IPOINT( 7 ) = IPOINT( 6 ) + NSUNFR * KSUBNP * IDP
IPOINT( 8 ) = IPOINT( 7 ) + NSUCMP * KSUBNP * IDP
IPOINT( 9 ) = IPOINT( 8 ) + NSUCMP * KSUBNP * IDP
IPOINT(10 ) = IPOINT( 9 ) + NSUCMP * KSUBNP * IDP
IPOINT(11 ) = IPOINT(10 ) + NSUCMP * KSUBNP * IDP
IPOINT(12 ) = IPOINT(11 ) + NSUCMP * KSUBNP * IDP
IPOINT(13 ) = IPOINT(12 ) + NSUSSTR * KSUBNP * IDP

SYSTEMS' MANUAL
MHOST Version 4.2
Page : 82
The following SUBROUTINE SUBFEM accesses the arrays defined by the subelement pointer analysis:

```
C=SUBROUTINE=SUBFEM CALLED BY SUBROUTINE 'ASSEM4'

SUBROUTINE SUBFEM
1 (FWORK, IWORK, ISIZE, IPOINT, IELEM, NSUBEL, NSUBNP, NSUBTY,
  2 NSDATA, KEMBED, WORKSP, ISLV )

C
C **********************************************************************

C IMPLICIT REAL*8 (A-H, O-Z)

SYSTEMS' MANUAL
Page : 83
```
REAL*4 RWORK

C
************************************************************
C
C      ALGEM       ICREAD, ILPRNT, JLPRNT, ICONSL, IPPOST, ISCRAF,
C 1     IPLOT, ISRSTR, JCREAD, IPVARS, IPSETS, IFILEX,
C 2     LINE , LINE2
C
C      CONTRO     JEND , JITTER, JTEMP , JPRINT, JP , JSUB ,
C 1     JINC , JREST, JSAVE , JREDIM, JAUTO , JPOST ,
C 2     JBACK , JOPTIM, JCREEP, JDIST , JCONST, JDYN ,
C 3     NONISO, ITERM, ITTRIG, IDYN , JREDIP, JTANGE ,
C 4     JTERM, JFORCE, JUTEMP, JCOEF, JDISTS, JFREEK ,
C 5     JDERIV, JBOUND, IDSTOP, INSTR, JPLAST, JBDND ,
C 6     JFRONT, JDEFOR, JEMBED, ITEST , JDISP , IFBFGS ,
C 7     IFSINT, IFLINE, IPRTNT, ICOMP, IPCONJ, JGEIGEN ,
C 8     IFBODY, IFGRAV, IFCENT, Jadamp, IDYN , ISTAT ,
C 9     JFRONT, JDEFOR, JEMBED, ITEST , JDISP , IFBFGS ,
+     JFRONT, JDEFOR, JEMBED, ITEST , JDISP , IFBFGS ,
C
C      ELTYP       NELCRD, NELNFR, NELNOD, NELSTR, NELCHG, NELPR ,
C 1     NELINT, NELLV , NELLAY, NDI , NSHEAR, NELCMF ,
C
C      ELEMEN      IC , IEL , IDF , JLAB , IPATH , JASSEM ,
C 1     JRULE , JCAST, JELO09, JEL010, JEL011, JEL012
C
C      SUBTYP      NSUCRD, NSUNFR, NSUNOD, NSUSTR, NSUCHG, NSUCMP ,
C 1     NSUINT, NSULV , NSULAY, NSUSD, NSUSD , NSUSD ,
C
C      TRANSF      CTRANS( 9), XJACOB( 9)
C
C      LOUBIN      JLOUB , JINTEK, JEXTRA, JWEIG, JSUBRE, JINTEK ,
C 1     JLOUB , JINTEK, JEXTRA, JWEIG, JSUBRE, JINTEK ,
C
C      MAXIMA      MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHG, MAXPRS ,
C 1     MAXLAI, MAXINT, MAXWK, MAXLV, NSMAX, MAXCMF ,
C 2     MAXBSP, MAXCMF, MAXTEM, MAXEM, MAXLV, MAXINT ,
C 3     MAXFRN, MAXBET, MAXVAR, MAXSET, MAXH , MAXORD ,
C 4     MAX025, MAX026, MAX027, MAX028, MAX029, MAX030
C
C      PARAM       NTYPE , NELEM , NNODE , NBC , NTIE , NMAX ,
C 1     NRAN, NIRAC, NFD , NBAND , NEXT , NSUB ,
C 2     NPRINT, NPOST, NSBEC, NDUP , NSIZE , NSBEC ,
C 3     NSHEFT, NSBFGS, NGMRS , NSPRI , NMSASS , NDASH ,
C 4     NDMCMX, NSBNC , NSUPER, NHARM , NBASE , NTINC ,
C 5     NTTER , NPSPTS, NPPTS, NPPULSE, NPPPTS, NHARD ,
C 6     NSUMCH, NDIMEN, NMONIT, NP40, NP41, NP42 ,
C 7     NP43, NP44, NP45, NP46, NP47, NP48
C
C      ERRORS       IERR
C      MACHIN       IDP
C      TIME         TTINC, TTINC, RTIM
C      START1       IELPRM, IYP , INEL , ICHAR, IPRES , ISTRS ,
C 1     ISTRN , ICOP , IPRINT, IPPOST , IDIST , ILEAN ,

SYSTEMS' MANUAL

Page : 84

MHOST Version 4.2
IBPRES, IHBRM, IDMONT, IST116, IST117, IST118
COMMON / START2 / INOD, ITEM, INLV, IPOSU, ITEMDF, IDUP
COMMON / START3 / IKBC, ITI, ITR, ITRAN, ITRAC, IEXT,
COMMON / START4 / IDINC, IDTOT, IFC, IRESID, IWIND, ISIGNO,
COMMON / START4 / IEPSNO, IPSTRN, ICS, ITSTRN, IISTRN.
COMMON / START4 / I1PSTN, I2CSTR, I3ITSTR, I4PSNO, ICSTNO, I5TSNO,
COMMON / START4 / I6TSNO, I7IDMN, I8EQCS, IGENDNO, I9IDMN, I10ITR,
COMMON / START4 / IDIVN, IDYN, I12ISX1, I13ISX2, I14KSTR, I15SHELL,
COMMON / START4 / IEPSNO, IU/OT, IEQCSI, ISIGNO, IEPSNO, IPSTRN, ICS, ITSTRN,
COMMON / START5 / INOD, ITEM, INLV, IPOSU, ITEMDF, IDUP
COMMON / START5 / IDINC, IDTOT, IFC, IRESID, IWIND, ISIGNO,
COMMON / START5 / IEPSNO, IPSTRN, ICS, ITSTRN, IISTRN.
COMMON / START5 / I1PSTN, I2CSTR, I3ITSTR, I4PSNO, ICSTNO, I5TSNO,
COMMON / START5 / I6TSNO, I7IDMN, I8EQCS, IGENDNO, I9IDMN, I10ITR,
COMMON / START5 / IDIVN, IDYN, I12ISX1, I13ISX2, I14KSTR, I15SHELL,
COMMON / START5 / IEPSNO, IU/OT, IEQCSI, ISIGNO, IEPSNO, IPSTRN, ICS, ITSTRN,
COMMON / START6 / INOD, ITEM, INLV, IPOSU, ITEMDF, IDUP
COMMON / START6 / IDINC, IDTOT, IFC, IRESID, IWIND, ISIGNO,
COMMON / START6 / IEPSNO, IPSTRN, ICS, ITSTRN, IISTRN.
COMMON / START6 / I1PSTN, I2CSTR, I3ITSTR, I4PSNO, ICSTNO, I5TSNO,
COMMON / START6 / I6TSNO, I7IDMN, I8EQCS, IGENDNO, I9IDMN, I10ITR,
COMMON / START6 / IDIVN, IDYN, I12ISX1, I13ISX2, I14KSTR, I15SHELL,
COMMON / START6 / IEPSNO, IU/OT, IEQCSI, ISIGNO, IEPSNO, IPSTRN, ICS, ITSTRN,
COMMON / START7 / INOD, ITEM, INLV, IPOSU, ITEMDF, IDUP
COMMON / START7 / IDINC, IDTOT, IFC, IRESID, IWIND, ISIGNO,
COMMON / START7 / IEPSNO, IPSTRN, ICS, ITSTRN, IISTRN.
COMMON / START7 / I1PSTN, I2CSTR, I3ITSTR, I4PSNO, ICSTNO, I5TSNO,
COMMON / START7 / I6TSNO, I7IDMN, I8EQCS, IGENDNO, I9IDMN, I10ITR,
COMMON / START7 / IDIVN, IDYN, I12ISX1, I13ISX2, I14KSTR, I15SHELL,
COMMON / START7 / IEPSNO, IU/OT, IEQCSI, ISIGNO, IEPSNO, IPSTRN, ICS, ITSTRN,
COMMON / START8 / INOD, ITEM, INLV, IPOSU, ITEMDF, IDUP
COMMON / START8 / IDINC, IDTOT, IFC, IRESID, IWIND, ISIGNO,
COMMON / START8 / IEPSNO, IPSTRN, ICS, ITSTRN, IISTRN.
COMMON / START8 / I1PSTN, I2CSTR, I3ITSTR, I4PSNO, ICSTNO, I5TSNO,
COMMON / START8 / I6TSNO, I7IDMN, I8EQCS, IGENDNO, I9IDMN, I10ITR,
COMMON / START8 / IDIVN, IDYN, I12ISX1, I13ISX2, I14KSTR, I15SHELL,
COMMON / START8 / IEPSNO, IU/OT, IEQCSI, ISIGNO, IEPSNO, IPSTRN, ICS, ITSTRN,
COMMON / TOLER / RELERR, ABSEERR, REACMX, RESIMX, DISERR, DISTOR,
COMMON / TOLER / ENGTO, ENGNRM

FUNCTION
COMPLETE MIXED FINITE ELEMENT SOLUTION FOR THE SUBELEMENT MESH
SUBDIVISION. RESULTS ARE FED BACK TO THE GLOBAL MESH AS A
MODIFIED RESIDUAL FORCE VECTOR, AND NODAL QUANTITIES ASSOCIATED
WITH THE SUBELEMENT GRIDS ARE RESTORED FOR FURTHER DATA
MANIPULATIONS.

ARGUMENTS

WORK SINGLE PRECISION REAL WORKAREA
IWORK INTEGER WORKAREA
ISIZE TOTAL SIZE FOR THE WORKAREA
IPOINT POINTER ARRAY FOR THE SUBELEMENT DATA STORAGE
IELEM INDEX FOR THE CURRENT 'MASTER' ELEMENT
NSUBEL ARRAY FOR THE NUMBER OF SUBELEMENT ELEMENTS
NSUBNP ARRAY FOR THE NUMBER OF SUBELEMENT NODES
NSUBTY ARRAY FOR THE SUBELEMENT ELEMENT TYPES
NSDATA NUMBER-OF-ENTRIES/ELEMENT IN IPOINT
KEMBED FLAG FOR THE EMBEDDED SUBELEMENTS
WORKSP DOUBLE PRECISION REAL WORKSPACE

POINTER

IPOINT( - ,IELEM) I ARRAY

1 I ELEMENT CONNECTIVITY
2 I PHYSICAL COORDINATES
3 I ELEMENT COORDINATES
4 I TOTAL NODAL DISPLACEMENT
5 I INCREMENTAL NODAL DISPLACEMENT
6 I CORRECTIONAL NODAL DISPLACEMENT
7 I TOTAL NODAL STRAIN
8 I TOTAL NODAL STRESS
9 I TOTAL NODAL PLASTIC STRAIN
10 I TOTAL NODAL CREEP STRAIN
11 I TOTAL NODAL THERMAL STRAIN
12 I GENERALIZED NODAL STRESS (TOTAL)
13 I GENERALIZED NODAL STRAIN (TOTAL)
14 I INCREMENTAL NODAL STRAIN
15 I INCREMENTAL NODAL STRESS
16 I INCREMENTAL NODAL PLASTIC STRAIN
17 I INCREMENTAL NODAL CREEP STRAIN
18 I INCREMENTAL NODAL THERMAL STRAIN
19 I GENERALIZED NODAL STRESS (INCREMENTAL)
20 I GENERALIZED NODAL STRAIN (INCREMENTAL)
21 I LUMPED MASS COEFFICIENT FOR SMOOTHING
CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)

CALL TIMOUT
1 ('SUBLE', 'LEME', 'NTL', 'OOP', 'ENTE', 'RED')

WRITE ICONSL, 2000 IELEM
FORMAT (30X, 'IELEM = ', I5)
*** SET UP CONTROL VARIABLES ASSOCIATED WITH THE 'VARIATIONAL' ***
*** STRAIN RECOVERY AND THE NODAL 'RESIDUAL' CALCULATION FOR THE ***
*** AUGMENTED LAGRANGIAN TYPE ITERATION ***
S.N. 12-09-83/01-31-84/03-21-84 ***

*** JINTER = 1 'REDUCED' INTEGRATION FOR THE RECOVERY PROCESS ***
*** = 2 'FULL' INTEGRATION ***
*** = 3 'TRAPEZOIDAL' INTEGRATION ***
*** = 4 'SELECTIVE' GAUSS INTEGRATION ***

*** JEXTRA = 1 'FULL' INTEGRATION FOR THE RESIDUALS ***
*** = 2 'REDUCED' INTEGRATION ***

KGALJS: CONTROL VARIABLE FOR THE 'RESIDUAL' INTEGRATION

JGAUS = 1
IF ( JINTER .EQ. 2 ) JGAUS = 0
IF ( JINTER .EQ. 3 ) JGAUS = 2
IF ( JINTER .EQ. 4 ) JGAUS = 3
IF ( JWEIG. .EQ. 3 ) JTRANS = 0
JTRANS = 1

KGAUS = 0
IF ( JEXTRA .EQ. 2 ) KGAUS = 1
IF ( JWEIG. .GE. 2 ) KGAUS = 3
INTRSD = NELINT
IF ( JEXTRA .EQ. 2 ) INTRSD = 1
KTRANS = 0
IF ( JWEIG. .EQ. 2 ) KTRANS = 0

ARRANGE SUBELEMENT POINTERS FOR THE GLOBAL WORKAREA

J CONNC = IPOINT( 1 , IELEM )
J PCORD = IPOINT( 2 , IELEM )
J ECORD = IPOINT( 3 , IELEM )
J DISPL = IPOINT( 4 , IELEM )
J DISPI = IPOINT( 5 , IELEM )
J DISPC = IPOINT( 6 , IELEM )
J STRAN = IPOINT( 7 , IELEM )
J STRST = IPOINT( 8 , IELEM )
J TPLAS = IPOINT( 9 , IELEM )
J CREPT = IPOINT( 10 , IELEM )
J THERT = IPOINT( 11 , IELEM )
J GSTRS = IPOINT( 12 , IELEM )
J GSTRN = IPOINT( 13 , IELEM )
J STRNI = IPOINT( 14 , IELEM )
J STRSI = IPOINT( 15 , IELEM )
J PLASI = IPOINT( 16 , IELEM )
J CREPI = IPOINT( 17 , IELEM )
J THERI = IPOINT( 18 , IELEM )
J GSTRI = IPOINT( 19 , IELEM )
J GSTINI = IPOINT( 20 , IELEM )
J LUMP = IPOINT( 21 , IELEM )
J DMATX = IPOINT( 22 , IELEM )
J TEMPR = IPOINT( 23 , IELEM )
J TMPDF = IPOINT( 24 , IELEM )
J EQPST = IPOINT( 25 , IELEM )
J EQPSI = IPOINT( 26 , IELEM )
J EQCST = IPOINT( 27 , IELEM )
J EQCSI = IPOINT( 28 , IELEM )
J OMEGT = IPOINT( 29 , IELEM )
J OMEGI = IPOINT( 30 , IELEM )
J TSWL = IPOINT( 31 , IELEM )
J SWELL = IPOINT( 32 , IELEM )
J TDSSTR = IPOINT( 33 , IELEM )
J YIELD = IPOINT( 34 , IELEM )

PATCHY CODE FOR POINTERS ASSOCIATED WITH THE ELEMENT BUFFER

WORKSPACE ALLOCATION FOR THE CURRENT SUBELEMENT SOLUTION

IASSEM = 1
CALL SUBELV
1  ( IWORK, KSUBTY, IERR )
IASSEM = 0

CALL SUBALC
1  ( WORKSP, ISTIFF, IELSTF, ILOADV, ILOAD, ISIZE, KSUBNP, KSUBEL,
2  NSUNFR, NSUNOD, NSUSTR, NSUINT, IEBETA, IELDET, IELCOR, IELDMT,
3  KLAST, ILAST, IDP, NSUCRD, IELIDS, IELGTD, JELGID,
4  JELDMT, KGSIZE, KESIZE, JELEPS, JELMP, JIDWRK, JELSTR, IELSTR,
5  JSUCHR, NSUCHR, JREACT, JRESID)

CALL SMOOT
1  ( RWORK, IWORK, ISLV, CTRANS, IELEM, IC, 0 )

IF( JINC .GE. 1 .AND. JITER .GE. 1 ) GO TO 7900

C **********************************************************************
C PHASE 0 : PREPARATION OF LUMPED 'GRAMM' MATRIX FOR STRAIN
C SMOOTHING
C **********************************************************************

CALL NUL( RWORK(JLUMP) , KSUBNP )

DO 4900 ISUBEL = 1 , KSUBEL

CALL SNODEL
1  ( ISUBEL, IWORK, JCONNCC, RWORK(JPCORD), RWORK(JDMATX),
2  WORKSP(IELCOR), WORKSP(IELDMT), KSUBEL, KSUBNP, NSUNOD,
3  NSUSTR, NSUCRD, NSUNFR, RWORK(JDISPL), RWORK(JDISPI),
4  WORKSP(IELODTS), WORKSP(IELODS), KCONNCC, JCONNCC,
5  RWORK(JTDSTR), RWORK(JTDST), WORKSP(IELOID),
6  RWORK(JGSTMH), WORKSP(IELOST))

CALL IMPMAS
1  ( RWORK(JLUMP), WORKSP(IELOD), IWORK(JCONN), NSUNOD,
2  NSUINT, KSUBTP, KSUBEL, RWORK(ICHAR),
3  WORKSP(IELCOR), NSUCRD, ISUBEL, KSUBEL,
4  RWORK(JPCORD), NSUCRD, CTRANS, JGRAM )

CONTINUE

CALL NUL
1  ( RWORK(JTDSTR), NSUSTR*KSUBNP)

C **********************************************************************
C MIXED ITERATIVE SOLUTION FOR THE SUBELEMENT GRID
C **********************************************************************
C (I) TOTAL NUMBER OF ITERATIONS AND CONVERGENCE TOLERANCE ARE SAME AS
C THOSE DEFINED FOR THE GLOBAL MESH
C (II) IN CASE OF NO CONVERGENCE ACHIEVED (IF NOT DIVERGE), THIS
C ROUTINE RETURNS NORMALLY TO THE MAIN FINITE ELEMENT SOLUTION
C WITH A LITTLE WARNING MESSAGE
C
C **********************************************************************
C PHASE 1 : NODAL STRESS / MATERIAL TANGENT CALCULATION FOR THE
C DISPLACEMENT PRECONDITIONING
C **********************************************************************
C
N3HARD = 3 * NHARD

CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KYIELD),
2 N3HARD ,NELNOD ,RWORK (JYIELD))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (IXIRL),
2 NSUNFR ,NELNOD ,RWORK (JDISPI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KICENO),
2 NSUCMP ,NELNOD ,RWORK (JSTRI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KICSNO),
2 NSUCMP ,NELNOD ,RWORK (JCREPI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KITSNO),
2 NSUCHR ,NELNOD ,RWORK (JITHERI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KISTS),
2 NSUCMP ,NELNOD ,WORKSP (JSUCHR))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KIGSNO),
2 NELSTR ,NELNOD ,RWORK (JGSTRI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KEQCSI),
2 1 ,NELNOD ,RWORK (JEQCSI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KIQMN),
2 NELSTR ,NELNOD ,RWORK (JOMEGI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KSWLNO),
2 1 ,NELNOD ,RWORK (JSWELL))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KIMPNO),
2 1 ,NELNOD ,RWORK (JTEMPR))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KTDFO)),
2 1 ,NELNOD ,RWORK (JIMPDF))

CALL SUBT
1 (RWORKSP(JTDWRK),RWORK (JGSTRI),RWORK (JTDSTR),KSUBNP*NSUSTR )

CALL STRESS
1 (RWORK (JSTRAN),RWORK (JSTRNI),RWORK (JSTRST),RWORK (JSTRSI),
2 RWORK (JCREPT),RWORK (JCREPI),RWORK (JTPLAS),RWORK (JPLASI),
3 RWORK (JITHERI),RWORK (JITHERI),RWORK (JTEMPI),RWORKSP(JSUCHR),
4 NSUSTR ,NSULAY ,NSUCHR ,KSUBNP,
5 KSUBNP ,NSUBLISH ,NSUSHR ,JLAW,
6 RWORK (JEQPSI),RWORK (JEQPSI),KLAU ,RWORK (JDMATX),
7 JMATRX ,JTEMP ,JCREEP ,JITER,
8 JINC ,JNONISO ,RWORK (JIMPDF),RWORK (JEQCSSI),
9 RWORK (JTVSWL),RWORK (JTDSTR),JCONST ,RWORK (JOMEIGT),
+ JWORK (JOMEIGT),JITHERM ,JTIMINC ,RUNTIN,
1 RWORK (JGSTRN),RWORK (JGSTRN),RWORK (JGSTRS),RWORK (JGSTRI),
2 NSULAY ,NSUCMP ,LBACK ,RWORK (JSWELL),
3 RWORK (JEQCSSI),RWORK (JPCORD),NSUCRP ,JPLAST,
4 ICOMPS ,RWORK (IPREF ),RWORK (JYIELD),NHARD,
5 JUHOK ,JINC ,JYIELD ,JWKSPLP )

CALL ADD
1 (RWORK (JTDSTR),RWORK (JTDSTR),RWORKSP (JTDWRK),NSUSTR*KSUBNP )

C ******************************************************
C MAIN ITERATION LOOP
C ******************************************************
C
ISTART = 1
NSTOPS = NITER + 1

DO 8000 ISITER = 1 , NSTOPS

MCITER = ISITER - 1

DO 5000 ISUBEL = 1 , KSUBEL

CALL SUBELV

SYSTEMS' MANUAL

Page : 92
CALL SNODEL
1 (ISUBEL,IWORK(JCONNC),WORKSP(JPCORD), WORKSP(JDMATX),
2 WORKSP(IELCOR), WORKSP(IELDST), KSUBEL, KSUBNP,
3 N S U N O D , N S U T R , N S U C R D , N S U N F R ,
4 RWORK (JDISPI), RWORK (JDISPL), WORKSP (IELDIS), WORKSP (IELDTS),
5 KCONNC, JCONNC, RWORK (JDISST), WORKSP (IELGID),
6 RWORK (JGSTR), WORKSP (IELSTR))

CALL SUBDER
1 (WORKSP(IEBETA), WORKSP(IELDET), WORKSP(IELCOR), WORKSP(IELDTS),
2 WORKSP(IELIDS), RWORK (ICHA R), IGAUS, ITRANS ,
3 CTRANS )

CALL INTERP
1 (WORKSP(JELGID), WORKSP(IELGID),NSUINT, NSUNOD,
2 N S U T R )

CALL INTERP
1 (WORKSP(JELDMT), WORKSP(IELDMT),NSUINT, NSUNOD,
2 N S U T R * N S U T R )

IF ( ISITER .EQ. 1 ) CALL STIFF
1 (WORKSP(IESTF), WORKSP(ILOAD), WORKSP(JELGID), WORKSP(IEBETA),
2 WORKSP(IELDET), N S U T R , N S U N T , N S U L V ,
3 N S UIDF, MCI TER, WORKSP(JELDMT), 1 ,
4 1 ' LEAK', 0 , 0 )

IF ( ISITER .EQ. 1 ) CALL SYSEQN
1 (WORKSP(IESTF), WORKSP(ILOAD), WORKSP(IELSTF), WORKSP(ILOAD),
2 IWORK (JCONNC), N S U N O D , N S U N F R , KSUB NP ,
3 KSUBEL, KSIZE, KSIZE, ISUBEL )

C 5000 CONTINUE
5001 CONTINUE

C CALL SUBSOL
1 (WORKSP(IESTF), WORKSP(ILOADV), KSIZE, RWORK (JDISPC),
2 N S U N F R , KSUB NP, RWORK (JDISPI), NELNFR ,
3 NELNOD, N S U N O D , KEMBED, ISITER ,
4 WORKSP(JREACT), KSUBEL )

CALL ADD
1 (RWORK (JDISPI), RWORK (JDISPI), RWORK (JDISPC), N S U N F R * KSUB NP )

CALL PJOP
* 1 (RWORK (JDISPI), 'DISINC', N S U N F R * KSUB NP)

CALL PJOP
* 1 (RWORK (JDISPC), 'DISCOR', N S U N F R * KSUB NP)

C ********************************************
C PHASE 2 : NODAL STRAIN RECOVERY
SECOND ELEMENT LOOP :

CALL NUL
1 (RWORK (JGSTNI), KSUB* = KSUBNP )

DO 5100 ISUBEL = 1 , KSUBEL

IASSEM = 0

CALL SUBELV
1 ( IWORK , KSUBTY , IERR )

CALL SNODEL
1 ( ISUBEL , IWORK (JCONNC), RWORK (JPCORD), RWORK (JDMATX),
2 WORKSP (IELCOR), WORKSP (IELMT), KSUBEL , KSUBNP ,
3 NSUNOD , NSUSTR , NSUCRD , NSUNFR ,
4 RWORK (JDISPI), RWORK (JDISPL), WORKSP (IELIDS), WORKSP (IELTDS),
5 KCONNC , JCONNC , RWORK (JTDSTR), WORKSP (IELGID),
6 RWORK (JGSTRI), WORKSP (IELSTR) )

CALL SUBDER
1 ( WORKSP (IEBETA), WORKSP (IELDET), WORKSP (IELCOR), WORKSP (IELTDS),
2 WORKSP (IELIDS), RWORK (ICR), JGAUS , JTRANS ,
3 CTRANS )

CALL STRAIN
1 ( RWORK (JGSTNI), WORKSP (JLEPS), WORKSP (IELIDS), WORKSP (JEBETA),
2 NSUSTR , NSUINT , NSUNFR , NSUNOD ,
3 NUSDIF , KSUBNP , RWORK (JLIMP), WORKSP (IELDET),
4 RWORK (KCONNC), JGAUS , WORKSP (JELIMP), JTEMP ,
5 RWORK (JTHERI), RWORK (ICR), NSUCHR , JLA ,
6 NSUSTR , KSUBTY , WORKSP (IELCOR), NSUCRD ,
7 CTRANS , RWORK (JSTRNI), NSULAY , NSUCMP ,
8 WORKSP , WORKSP , RWORK (JCREP), RWORK (JTHERT),
9 JSTRNI , JCITER )

CONTINUE

CALL STRESS
1 ( RWORK (JSTRA), RWORK (JSTRI), RWORK (JSTRST), RWORK (JSTRT),
2 RWORK (JCREP), RWORK (JCREP), RWORK (JPLAS), RWORK (JPLST),
3 RWORK (JTHERI), RWORK (JHERT), RWORK (JTEMP), WORKSP (JSUCHR),
4 NSUSTR , NSULAY , NSUCHR , NSUNOD ,
5 KSUBNP , NSUNDI , NSUSHR , JLA ,
6 RWORK (JEQPS), RWORK (JEPSI), KLAW , RWORK (JDMATX),
C

CALL SNODEL
1 (ISUBEL ,IWORK (JCONNC),IWORK (JDMATX),IWORK (JPCORD),IWORK (JRESID),
2 WORKSP(IELCOR),WORKSP(IELDMT),KSUBEL ,KSUBNP ,
3 NSUNOD ,NSUSTR ,NSUCRD ,NSUNFR ,
4 WORK (JDISPL),WORK (JDISPI),WORK (JELCOR),WORK (JELDMT),
5 KCONNC ,KCONNC ,KWORK (JDMATX),KWORK (JPCORD),
6 WORK (JGSTRN),WORK (JGSTR),WORK (JGSTRN),WORK (JGSTR),
7 WORK (JTCOR),WORK (JTCOR),)

CALL SUBDER
1 (WORKSP(JELSTR),WORKSP(JELSTR),NSUINT ,NSUNOD ,
2 WORKSP(IELSTR),WORKSP(IELSTR),NSUINT ,NSUNOD ,
3 CTRANS ,)

CALL INTERP
1 (WORKSP(JELSTR),WORKSP(JELSTR),NSUINT ,NSUNOD ,
2 NSUSTR ,)

CALL SUBRES
1 (ISUBEL ,WORKSP(ISOLOAD),WORKSP(JELSTR),WORKSP(JELSTR),
2 IWORK (KCONNC),NSUNOD ,NSUSTR ,NSUNFR ,
3 NSUINT ,WORKSP(ISOLOAD),WORKSP(JELSTR),KSIZE ,
4 KSUBNP ,WORK (JDISPL),KSIZE ,ICONV ,
5 CTOLER ,WORKSP(ISOLOAD),IWORK (JCONNC),IMPRNT ,
6 ICONSL ,KSUBEL ,MCITER ,ENGTOT ,
7 WORKSP(JREACT),WORKSP(JRESID))

5200 CONTINUE

CALL SUBCHK
1 (IWORK (JDISPC),WORK (JDISPI),KSIZE ,DISTOR ,
2 ISITER ,ICONV ,)

C *******************************************************************************
C CHECK THE DISPLACEMENT CONVERGENCE
C******************************************************************************
C
IF( ICONVG .EQ. 1 )      GO TO 8100
C
8000          CONTINUE
C
C******************************************************************************
C CALCULATE THE CONTRIBUTION OF SUBELEMENT REFINEMENT IN TERMS OF
C THE GLOBAL RESIDUAL VECTOR
C******************************************************************************
C
8100          CONTINUE
C
CALL SUBINT
1 ( RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KIGSNO),
2 NELSTR ,NELNOD ,RWORK (JGSTRI))
IF( JSUBRE .EQ. 0 ) CALL SUBGLB
1 ( IC ,RWORK (IXP) ,RWORK (JSTSTR),RWORK (JSTRSI),
2 IWORK (JCONN),NSUNOD ,NSUSTR ,NSUNFR ,
3 NSWINT ,IDF ,WORKSP ,RWORK (IBETA) ,
4 RWORK (JECORD),RWORK (ICOR) ,RWORK (JLUMP),NELSTR ,
5 NSELCD ,NELCRD ,NELNOD ,KSUBNP ,
6 KSUBEL ,NFD ,RWORK (IREAC),IWORK (IELV) ,
7 RWORK (ISRL) ,NEXT ,INEXT ,RWORK (ICH) ,
8 NELCHR ,NELNFR ,RWORK (JGSTRI),WORKSP (JTDWRK),
9 KEMBED )
IF( JSUBRE .NE. 0 ) CALL SUBGLD
1 ( IC ,RWORK (IXP) ,RWORK (JSTSTR),RWORK (JSTRSI),
2 IWORK (JCONN),NSUNOD ,NSUSTR ,NSUNFR ,
3 NSWINT ,IDF ,RWORK (KIGSNO),RWORK (IBETA) ,
4 RWORK (JECORD),RWORK (ICOR) ,RWORK (JLUMP),NELSTR ,
5 NSELCD ,NELCRD ,NELNOD ,KSUBNP ,
6 KSUBEL ,NFD ,RWORK (IREAC),IWORK (IELV) ,
7 RWORK (ISRL) ,NEXT ,INEXT ,RWORK (ICH) ,
8 NELCHR ,NELNFR ,RWORK (IDET) ,NELINT ,
9 RWORK (KIGSIG),INFRSD ,IC ,KSUBTY ,
+ KEMBED ,WORKSP ,RWORK (JGSTRI),WORKSP (JTDWRK) )
C
RETURN
END

2.3 Counters and Pointers
Most of the variables used as either counters or pointers are documented in the Fortran source program. In this section, we discuss the counter and pointer variables stored in the common blocks:

START 1

Stores pointers for storing global data and some of the element data. Variables are:

IELPRM Pointer to the array storing the element parameters for all element types used in the present mesh.
ITYP Pointer to the element type identifier array.
INEL Pointer to the element connectivity array.
ICHAR Pointer to the nodal material property data array.
IPRES Flag for pressure loading.
IPRINT Pointer to the line printer output control data array.
IBPRES Pointer to the nodal pressure definition array.
IBNORM Pointer to the nodal array storing the components of the nodal point normal.

START 2

Stores pointers for the global mesh data. Variables are:

INOD Pointer to the nodal coordinate definition array.
ITEM Pointer to the nodal total temperature array.
ITEMDF Pointer to the nodal incremental temperature array.
IDUP Pointer to the array storing the duplicate node input data.

START 3

Stores the pointers for the global nodal constraint and loading data arrays. Variables are:

IKBC Starting address for the nodal displacement constrain
array.

**ITI**  
Pointer to the integer array for defining the tying equations.

**ITR**  
Pointer to the real array for defining the tying equations.

**ITRAN**  
Starting address of the array for nodal coordinate transformation input.

**ISBC**  
Starting address of the integer array for storing nodal stress boundary condition data input.

**ISBCR**  
Pointer to the real array for the prescribed nodal stress values.

**START 4**

Stores the pointer for nodal data storage. Variables are:

**IDINC**  
Pointer to the incremental displacement array.

**IDTOT**  
Pointer to the total displacement array.

**IFORCE**  
Pointer to the total load vector due to the mechanical loading.

**IRESID**  
Pointer to the residual vector.

**IWIND**  
Not used in version 4.2.

**ISIGNO**  
Pointer to the total nodal stress array.

**IEPSNO**  
Pointer to the total nodal strain array.

**IFSTINO**  
Pointer to the total nodal plastic strain array.

**ICSTINO**  
Pointer to the total nodal creep strain array.

**ITSTINO**  
Pointer to the total nodal thermal strain array.

**IISTINO**  
Pointer to the incremental nodal stress array.

**IISNNO**  
Pointer to the incremental nodal strain array.
IIPSNO  Pointer to the incremental nodal plastic strain array.

IIICSNO  Pointer to the incremental nodal creep strain array.

IIITSNO  Starting address for the incremental thermal strain defined at nodes.

IDMINO  Starting address for the material tangent arrays defined at nodes. In case of shells, the constitutive resultant array is stored.

IOMENO  Starting address for the total shift tensor defined at nodes. This array is used for the kinematic hardening and the unified viscoplastic constitutive models.

ITDSNO  Starting address for the total initial stress terms due to the initial strain terms defined at nodes.

IDYNA  Starting address for the total nodal acceleration array.

IDYNV  Pointer to the total nodal velocity array. IDYNA and IDYNV are defined only when the transient analysis option is invoked.

IPREF  Pointer to the material orientation vector for anisotropic material response defined at a node.

IDFINC  Pointer to the nodal array for the incremental deformation gradient. This array is allocated and used for finite deformation analysis.

IDFTOT  Pointer to the nodal array for the total deformation gradient.

IYIELD  Pointer to the array used to define nodally the strain hardening slope table.

START 5

Stores pointers for global arrays. Variables are:

IRL  Starting address for a vector storing the nodal loads including the residual.

IREAC  Starting address for a vector storing the nodal reactions.
IES  Starting address for the space of banded global stiffness matrix. Not used in Version 4.2.

IAB  Starting address for the space to pull a row of the global stiffness array.

ILAST  Pointer to the last word used in the blank common workspace.

IDIAG  Pointer to the array for diagonal entries of the global stiffness array stored in profile form.

IUPTRI  Pointer to the array storing the upper triangular part of the global stiffness array in profile form.

ICOLPT  Pointer to the integer array storing the column height for each degree of freedom of the global stiffness matrix in profile form.

IMASDI  Pointer to the space reserved for the diagonal entry of the global mass matrix. This space is allocated only when the modal analysis option is invoked.

IMASUP  Pointer to the upper triangular entries of the mass matrix stored in profile form.

START 6

Stores pointers for the element workspace. Variables are:

IELV  Starting address for the connectivity array of the element being processed.

ICOR  Starting address for the nodal coordinate array for the current element.

ISIG  Starting address for the element total stress array at integration points.

IEPS  Starting address for the element total strain array at integration points.

IXRL  Starting address for the total nodal displacement array defined for the current element.
Starting address for the incremental nodal displacement array.

IXP
Starting address for the workspace to store the element load vector.

IXK
Starting address for the workspace for the element stiffness array.

KPSTNO
Pointer to the nodal plastic strain array for the current element.

KCSTNO
Pointer to the nodal creep strain array for the current element.

KTSTNO
Pointer to the nodal thermal strain array for the current element.

KISTNO
Pointer to the nodal incremental stress for the current element.

KISNNO
Pointer to the nodal incremental strain for the current element.

KIPSNO
Pointer to the nodal incremental plastic strain for the current element.

KICSNO
Pointer to the nodal incremental creep strain for the current element.

KITSNO
Pointer to the nodal incremental thermal strain for the current element.

KDMAT
Pointer to the element material tangent array defined at the integration points for the current element.

KDMINO
Pointer to the nodal material tangent array for the current element.

KTDSNO
Pointer to the nodal array for the initial stress due to initial strains such as thermal and creep effects defined for the current element.

KITDST
Pointer to the element initial stress vector.
START 7

Stores more pointers for global and element arrays. Variables are:

ICON Not used in Version 4.2.

IKBCR Starting address for a double precision real array storing the prescribed displacement values.

ITRANR Starting address for a double precision real array storing the angle of rotation for the user specified nodal coordinate transformations.

ITRACR Starting address for a double precision real array storing the values of user specified nodal coordinate transformations.

IBETA Starting address for a double precision real array storing the strain displacement matrix at integration points of the current element.

IDET Starting address for a double precision real array storing the determinant of the Jacobian for isoparametric mapping at integration points of the current element.

START 8

Stores pointers for the resultant quantities. Variables are:

KGEPS Pointer to the strain resultant array at element integration points.

KIGEPS Pointer to the incremental strain resultant array at element integration points.

KGEPS Pointer to the strain resultant array at element integration points.

KIGEPS Pointer to the incremental strain resultant array at element integration points.

KGSIG Pointer to the stress resultant array at element
integration points.

KIGSIG Pointer to the incremental stress resultant array at element integration points.

KGIDST Pointer to the total resultant for initial stress at element integration points.

IGEPNO Pointer to the global array for the strain resultants.

IIGENO Pointer to the global array for the incremental strain resultants.

IGSINO Pointer to the global array for the stress resultants.

IIGSNO Pointer to the global array for the incremental stress resultants.

IGIDNO Pointer to the global array for total resultants of initial stress.

KGEPNO Starting address for the nodal array of strain resultants of the current element.

KIGENO Starting address for the nodal array of incremental strain resultants of the current element.

KGSINO Starting address for the nodal array of stress resultants of the current element.

KIGSNO Starting address for the nodal array of incremental stress resultants of the current element.

KGIDNO Starting address for the nodal array of total resultant for initial stresses of the current element.

START 9

Stores pointers for creep related quantities. Variables are:

KEQCSI Pointer to the equivalent incremental creep strain at nodes of the current element.

KEQCST Pointer to the equivalent total creep strain at nodes of
3.0 THE FILE SYSTEM

This chapter is devoted to a discussion of the file system for supporting a series of structural analyses by the MHOST program. In the next section, the overview of this file system is presented followed by a detailed discussion on the user interface (input and output files), restart file, and post-processing data file.

3.1 Overview

The MHOST program is a batch processor and when execution is initiated the user does not need to interact with the program until the Fortran STOP statement is executed (or an execution error is detected by the operating system). Under an interactive operational environment, the code produces a summary of the execution log on the terminal screen. Note that the same information is written on the main output file.

The file system is composed of three parts: the standard input and output files referred to as the user interface; the restart file; and the post processing data file. The main report part is referred to as Fortran unit number ILPRNT (see Block Data Subprogram, common block ALGEM) which consists of the input data echo print (card image print out), the input data as it is interpreted by the MHOST code, and the summary of execution such as the usage of memory and time consumed by CPU. The analysis result part is referred to as Fortran unit number JLPRNT. To make this logical distinction clear, separate header records and page counters are provided to these units.

The user interface files are the standard input file, the log file and the line printer output file. All the instructions to accomplish the analysis task are written in the input file. The MHOST program assumes that the input file is a card image file. That is, the input file is a sequentially accessed, formatted file, and the record length is fixed to 80 bytes. Under most operating systems (perhaps IBM systems are the only exceptions), it is not necessary to declare explicitly the record length. As shown in the next section, records are individually read into the read buffer as 80 separate characters and the user interface routine decodes each line.

The output file is assumed to contain 132 bytes/record which is the standard width for most line printers (as well as other printers driven in the same manner as the line printer). Again, except for IBM systems, this file is opened without explicitly specifying the record length. The line
printer file is logically separated into two parts. As the default setting, the code is delivered with the same Fortran unit number.

The log file is the standard output unit on which the system's messages are printed. This file is defined as Fortran unit ICONSL and is handled in a manner which depends on the computer systems default value for interactive terminal output.

Typically, on PRIME systems, ICONSL is the unit number 1 (one) and under the UNIX environment, unit number 6 (six) is assigned to ICONSL.

The restart file is a sequentially accessed binary file. The record length is fixed to 256 bytes/record. The contents of common block and the work space are written and read by the MHOST code. This file is referred to as Fortran unit number Iivating. The unit number 8 (eight) is assigned as default value when the MHOST code is delivered.

The post-processing file is a sequentially written formatted file. A record length of 80 bytes is implicitly assumed but not necessarily declared explicitly when the file is opened (under most operating systems). Except for minor differences in the format to write nodal stresses and strains, construction of this file is compatible with the post-processing file output of the MARC general purpose finite element package version K1. Any commercially available finite element post-processing package can easily be modified to read this file. The MENTAT interactive finite element pre- and post-processing package available from MARC is shipped with the MHOST post-processing interface as a standard feature.

3.2 User Interface

The free format reader is written in Fortran 66 to interpret data coming in through the input data file in a line-by-line manner. Two utilities are available. One is the keyword interpreter SUBROUTINE KEY and another is the free format numeric data reader SUBROUTINE FREFOR:

```c
C SUBROUTINE KEY
  SUBROUTINE KEY(NAME,NOPT,IOPT,NN,IN,IERR)
C
C* * * * * *
C
C CHECKS STRING FOR KEYWORD
C
C NAME KEYWORD NAMES
C NOPT NUMBER OF KEYWORDS
C IOPT KEYWORD FLAG
```

MHST Version 4.2
C  NN       INTEGER PARAMETERS
C  IN       NUMBER OF INTEGERS
C  IERR     ERROR FLAG
C
C* * * * * *
C
C  IMPLICIT REAL*8 (A-H,O-Z)
C
C
DIMENSION NAME(1),NN(4)
COMMON / ALGEM / ICREAD,ILPRNT,JLPRNT,ICONSL,IPPOST,ISCRAF,
1       IPILOTB,IRSTRT,JCREAD,IPVARS,IPSETS,IFILEX,
2       PI ,LINE ,LINE2
COMMON/FREE/IA(80),IBEGIN(16),ILENGT(16),NSTRIN,IS,IOC,NEW
LOGICAL NEW
1 CONTINUE
JKEY=1
CALL STRING(IERR,JKEY,1)
IS=IS+1
IB=IBEGIN(IS)
IL=ILENGT(IS)
IF(IL.LT.3) GOTO 99
IF(IL.GT.4) IL=4
II=-4
IBI=IB-1
DO 3 I=1,NOPT
II=II+4
DO 2 J=1,IL
IF(IA(IBI+J).NE.NAME(II+J)) GOTO 3
2 CONTINUE
IOPT=I
GOTO 4
3 CONTINUE
GOTO 99
4 CONTINUE
C
IF(IN.EQ.0) GOTO 100
DO 5 I=1,IN
IS=IS+1
IB=IBEGIN(IS)
IL=ILENGT(IS)
CALL DECINT(NNI,IA(IB),IL,IERR)
NN(I)=NNI
5 CONTINUE
GOTO 100
C
99 CONTINUE
IERR=IERR+1
CALL LINES(1,1)
L=IB-1+IL
WRITE(ILPRNT,1001) (IA(I),I=IB,IL)
1001 FORMAT(2X,11H***ERROR***,5X,25HSTRING IS NOT A KEYWORD: ,78A1)
GOTO 1
100 CONTINUE
RETURN
END

C SUBROUTINE FREFOR
SUBROUTINE FREFOR(INIW,REAW,NINT,NREA,NVAR,IERR,JKEY)
C
C* * * * * * *
C
FREE FORMAT ROUTINE
C
C INIW INTEGER WORKSPACE
C REAW REAL WORKSPACE
C NINT NUMBER OF INTEGERS
C NREA NUMBER OF REALS
C NVAR
C IERR ERROR FLAG
C JKEY
C
C* * * * * * *
C
IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 REAW
C
COMMON / ALGEM / ICREAD,ILPRNT,ILPRNT,ICONSIL,IPARF,IPSETF,ISCRAF,
1 IPLOTB,IRSRT,JCREAD,IPVARS,IPSETS,IFILEX,
2 PI ,LINE ,LINE2
COMMON/FREE/IA(80),IBEGIN(16),ILENGT(16),NSTRIN,IS,ICOL,NEW
DIMENSION INIW(1),REAW(1)
LOGICAL LREA,NEW
IC=0
CALL STRING(IERR,JKEY,1)
IF(JKEY.EQ.1) GOTO 30
JKEY=0
C
IF(NINT.LE.0) GOTO 10
DO 1 I=1,NINT
IF(ICOL.LE.76) GOTO 2
JKEY=-1
CALL STRING(IERR, JKEY, 1)
IF(JKEY.EQ.1) GOTO 30
JKEY=0
2 CONTINUE
IC=IC+1
IS=IS+1
ICOL=ICOL+5
IB=IBEGIN(IS)
IL=ILENGT(IS)
CALL DECINT(IDT, IA(IB), IL, IERR)
INIW(IC)=IDT
1 CONTINUE
10 CONTINUE
C
IF(NREA.LE.0) GOTO 20
DO 11 I=1,NREA
IF(ICOL.LE.71) GOTO 12
JKEY=-1
CALL STRING(IERR, JKEY, 1)
IF(JKEY.EQ.1) GOTO 30
JKEY=0
12 CONTINUE
IC=IC+1
IS=IS+1
ICOL=ICOL+10
IB=IBEGIN(IS)
IL=ILENGT(IS)
C
REAW(IC)=0.
C
IF(IL .NE. 0) CALL DECREA(REAW(IC), IA(IB), IL, IERR)
C
11 CONTINUE
20 CONTINUE
C
IF(NVAR.EQ.0) GOTO 30
LREA=NVAR.GT.0
MTOT=IABS(NVAR)
IF(ICOL.LE.76) GOTO 23
JKEY=-1
CALL STRING(IERR, JKEY, 1)
IF(JKEY.EQ.1) GOTO 30
JKEY=0
CONTINUE
IS=IS+1
ICOL=ICOL+5
IB=IBEGIN(IS)
IL=ILENGT(IS)
CALL DECINT(NVAR,IA(IB),IL,IERR)
IF(NVAR.LE.0) GOTO 30

C
NTOT=NTOT*NVAR
DO 21 I=1,NTOT
IF(ICOL.LE.71.OR.(ICOL.LE.76.AND..NOT.IREA)) GOTO 24
JKEY=-1
CALL STRING(IERR,JKEY,1)
IF(JKEY.EQ.1) GO TO 30
JKEY=0
24 CONTINUE
IC=IC+1
IS=IS+1
ICOL=ICOL+5
IB=IBEGIN(IS)
IL=ILENGT(IS)
IF(IREA) GOTO 22
CALL DECINT(IDUM,IA(IB),IL,IERR)
INIW(IC)=IDUM
GOTO 21
22 CONTINUE
ICOL=ICOL+5
C
REAW(IC)=0.
C
IF(IL.NE.0)CALL DECREA(REAW(IC),IA(IB),IL,IERR)
21 CONTINUE
30 CONTINUE
RETURN
END

An example of usage of the above utilities is the parameter data reader
SUBROUTINE DATINI:

C ... SUBROUTINE DATINI ... CALLED BY SUBROUTINE 'HOST' OR 'FEM'
C
SUBROUTINE DATINI
1 (RWORK ,IWORK ,ISIZE ,VERSNO,MONTH ,JDATE ,NELEM ,NNODE ,
2 NEC ,NTIE ,NMAX ,NTRAN ,NTRAC ,NPOST ,NLVSUB,NFRSUB,

SYSTEMS' MANUAL
Page : 109
MHOST Version 4.2
**READ THE CONTROL DATA AND PARAMETERS ASSOCIATED WITH THE CORE STORAGE ALLOCATION**

**IMPLICIT REAL*8 ( A-H, O-Z )
  REAL*4 RWORK**

**DIMENSION RWORK ( ISIZE ), IWORK ( ISIZE)
DIMENSION NFRSUB(MAXSUB), NLVSUB(MAXSUB)
DIMENSION NAME ( 4, 72), NN ( 6)
DIMENSION NAME1 ( 4, 34), NAME2 ( 4, 36)
DIMENSION NAME3 ( 4, 2)
DIMENSION JPERIOD( 2)**

**EQUIVALENCE (NAME( 1, 1), NAME1( 1, 1))
EQUIVALENCE (NAME( 1,35), NAME2 ( 1, 1))
EQUIVALENCE (NAME( 1,71), NAME3 ( 1, 1))**

**COMMON / ALGEM / IREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRF,
  1   IPLOTB, IPRINT, JCREAD, IPvARS, IPSETS, IFILEX,
  2   PI , LINE , LINE2
COMMON / COUNT / LININC, LINTOT, NOECHO
COMMON / CTITLE / TITLE (  20), IDAT (  5), IDATE2, ICLOCK**
**PARAMETER DATA OPTIONS**

1. *ELEM MAXIMUM NUMBER AND THE TYPE OF ELEMENT*
2. *NODE MAXIMUM NUMBER OF NODES*
3. *BOUND MAXIMUM NUMBER OF DISPLACEMENT CONSTRAINT*
4. *TYING FLAG THE TYING OPTION WITH NUMBER OF TYING*
DEGREE OF FREEDOMS

*TRAN COORDINATE TRANSFORMATION OPTION FLAGGED WITH
THE NUMBER OF POINTS SUBJECTED TO THIS OPER.

*FORC MAXIMUM NUMBER OF NODAL FORCE DATA

*POST FLAG THE POST PROCESSING TAPE GENERATION
OPTION

*SUSI FLAG THE SUBSTRUCTURING OPTION WITH THE NUMBER
OF SUBSTRUCTURES

*EXTE

*PRES FLAG THE NODAL PRESSURE DEFINITION OPTION

*TEMP FLAG FOR THERMAL LOADING

*PRIN FLAG FOR PRINT OUTPUT

*REST FLAG FOR RESTART RUN

*LOUB SET UP NUMERICAL INTEGRATION

*STRE FLAG FOR STRESS BOUNDARY CONDITIONS

*END ......OBVIOUS.............

*TEST (RESERVED)

*DYN INVOKE TRANSIENT TIME INTEGRATION

*OPTI FLAG BAND WIDTH OPTIMIZATION

*TRAC FLAG THE DISTRIBUTED LOADING

*CREE FLAG THE CREEP STRAIN OPTION

*ANIS FLAG ANISOTROPY OPTION

*MODA MODAL ANALYSIS OPTION

*BUCK BUCKLING ANALYSIS OPTION

*THER TEMPERATURE DEPENDENT ELASTICITY OPTION

*CONS CONSTITUTIVE EQUATION SELECTION

*DIST FLAG FOR DISTRIBUTED LOAD

*DUPL DUPLICATED NODE OPTION

*REPO REPORT GENERATION INTERVAL

*ANG MODIFIED NEWTON OPTION

*UHE USER SUBROUTINE 'UHERM' OPTION

*SCH TIME INTEGRATION SCHEME OPTION

*UFOR USER SUBROUTINE 'UFORC' OPTION

*UTEM USER SUBROUTINE 'UTEMP' OPTION

*UCOE USER SUBROUTINE 'UCOEF' OPTION

*UPRE USER SUBROUTINE 'UPRESS' OPTION

*UHOO USER SUBROUTINE 'UHOOK' OPTION

*UDER USER SUBROUTINE 'UDEIV' OPTION

*UBOU USER SUBROUTINE 'UBOUN' OPTION

*PERI PERIODIC LOADING CONDITION OPTION FOR THE
BAND BAND SOLVER (DEFAULT)

*FRON FRONTAL SOLUTION SUBSYSTEM (OPTIONAL)

*DEFI EIGENVALUE EXTRACTION FOR THE STIFFNESS

*EME SUBELEMENT MESH ANALYSIS OPTION

*GMRS MULTIPLE GENERIC MODELLING REGIONS OPTION
**BEAM SECTION PARAMETER OPTION**
**DISP CONVENTIONAL DISPLACEMENT METHOD**
**SHIF POWER SHIFT FOR EIGEN EXTRACTION**
**BFGS BFGS UPDATE FOR THE NONLINEAR SOLUTION**
**SPRI ADDED STIFFNESS, GROUND SPRING**
**DASH ADDED DAMPING, DASHPOT TO GROUND**
**MASS ADDED MASS**
**SCEN SECANT NEWTON METHOD**
**LINE LINE SEARCH**
**HARM HARMONIC NODAL FORCE LOADING**
**BASE HARMONIC BASE EXCITATION**
**COMP COMPOSITE LAMINATE OPTION FOR ELEMENT 75**
**PULS PULSE LOAD OPTION**
**CONJ CONJUGATE GRADIENT ITERATION**
**SHOC SHOCK SPECTRA OPTION**
**SHIP POWER SPECTRAL DENSITY OPTION**
**NOEC SUPRESS THE MODAL DATA ECHO PRINT**
**PERT SET UP PERTURBATION SIZE FLAGS**
**STIF STRESS STIFFENING OPTION**
**CENT CENTRIFUGAL MASS STIFFNESS OPTION**
**HARD WORK-HARDENING OPTION FOR PLASTICITY**
**FINI FINITE STRAIN OPTION**
**LARG LARGE DISPLACEMENTS & ROTATIONS OPTION**
**FOLL FOLLOWER FORCES OPTION**
**WKSL USER SUBROUTINE 'WKSL'**
**HOUR (GLASS CONTROL) AS THE NAME INDICATES ....**
**MONI INVOKES MONITOR UTILITY**

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

**NOPT** = 72
**JSUBRE** = 0
**LOECHO** = 0
**IFPRNT** = 0

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

**SET DEFAULT VALUES**

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

**NSHIFT** = 0
**JPOST** = 0
**NSPRI** = 0
**NDASH** = 0
**NMASS** = 0
**NHARM** = 0
NBASE = 0
NTIE = 0
NDUP = 0
JEMBED = 0
ITHERM = 0
NCREEP = 1
JEMBED = 0
JDISP = 0
JHRGLS = 0
JWKSLP = 0
JISTRN = 0
JCITER = 0
JREPO = 1
ISTAT = 1
NSBFGS = 0
IDYNM = 0
NGMRS = 1
IPCONJ = 0
JSUB = 0
NSUB = 0
JFRONT = 0
JREST = 0
JCREEP = 0
JTEMP = 0
NEXT = 0
JUBOUN = 0
NONISO = 0
IFBFGS = 0
IFSCNT = 0
IFLINE = 0
NDYNMD = 0
IDYNMD = 100000
NPOSMD = 0
JOTHERM = 0
JCONST = 2
JDYN = 0
JEIGEN = 0
JDEFOR = 0
NBSECT = 0
JFORCE = 0
JPEROD(1) = 0
JPEROD(2) = 0
JUTEMP = 0
JUCOEF = 0
JDISTS = 0
![Image]
GO TO (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72), IOPT

OPTION 1: *ELEM - MAXIMUM NUMBER OF ELEMENTS IN MODEL

1 CONTINUE
NELEM = NN(1)
CALL TYPEIN
1 (IWORK, FWORK, IERR, NTYPES, ILAST, ILPRINT, NDIMEN)
GO TO 998

OPTION 2: *NODE - MAXIMUM NUMBER OF NODES IN MODEL

2 CONTINUE
NNODE = NN(1)
GO TO 998

OPTION 3: *BOUND - MAXIMUM NUMBER OF BOUNDARY CONDITIONS

3 CONTINUE
NBC = NN(1)
GO TO 998

OPTION 4: *TYING - MAXIMUM NUMBER OF TYING CONSTRAINT

4 CONTINUE
NTIE = NN(1)
NMAX = NN(2)
GO TO 998

OPTION 5: *TRAN - NUMBER OF NODAL COORDINATE TRANSFORMATIONS
5 CONTINUE
NTRAN = NN(1)
GO TO 998

C **********************************************************************
C OPTION 6 : *FORC - MAXIMUM NUMBER OF NODAL FORCE ENTRIES
C **********************************************************************

6 CONTINUE
NTRAC = NN(1)
GO TO 998

C **********************************************************************
C OPTION 7 : *POST - POST PROCESSING
C **********************************************************************

7 CONTINUE
JPOST = 1
NPOST = 1
IF (NN(1).GT.0) NPOST = NN(1)
GO TO 998

C **********************************************************************
C OPTION 8 : *SUBS - INACTIVE IN VERSION 2.0
C **********************************************************************

8 CONTINUE
JSUB = NN(1)
NSUB = NN(2)
IF(JSUB.NE.2) GO TO 998
DO 108 J = I,NSUB
CALL FREFOR(NN, NN, 3,0,0, IERR, JKEY)
NLVSUB(NN(1)) = NN(2)
NFRSUB(NN(1)) = NN(3)
108 CONTINUE
GO TO 998

C **********************************************************************
C OPTION 9 : *NEXT - EXTRNAL D.O.F. INACTIVE IN VERSION 2.0
C **********************************************************************

9 CONTINUE
NEXT = NN(1)
GO TO 998

C
C ***************************************************************************************************
C OPTION 10 : *PRES - NODAL PRESSURE DEFINITION
C ***************************************************************************************************

10 CONTINUE
   JPRES = 1
   GO TO 998
C
C ***************************************************************************************************
C OPTION 11 : *TEMP - TEMPERATURE LOAD FLAG TO BE SET
C ***************************************************************************************************

11 CONTINUE
   JTEMP = 1
   GO TO 998
C
C ***************************************************************************************************
C OPTION 12 : *PRIN - INCREASE THE NUMBER OF PRINT OPTIONS
C ***************************************************************************************************

12 CONTINUE
   IF (NN(1).LT.0) IFPRINT = 1
   IF (NN(1).LT.0) NPRINT = IABS(NN(1))
   IF (NN(1).GT.0) NPRINT = NN(1)
   GO TO 998
C
C ***************************************************************************************************
C OPTION 13 : *REST - RESTART TAPE FLAG TO BE SET
C ***************************************************************************************************

13 CONTINUE
   JREST = 1
   JINC = NN(1)
   NINC = NN(2)
C
C *** EXIT IMMEDIATELY WITHOUT READING *END CARD ****************************
C GO TO 16
C
C ***************************************************************************************************
C OPTION 14 : *LOUB - SELECTORS FOR THE NUMERICAL QUADRATURES
C ***************************************************************************************************

SYSTEMS' MANUAL

MH O S T  V e rsion 4 .2

Page : 118
14 CONTINUE
   JLOBB = 1
   JINTER = NN(1)
   JEXTRA = NN(2)
   JWEIGH = NN(3)
   JGRAM = NN(4)
   IF(JINTER.LT.1.OR.JINTER.GT.4) JINTER = 2
   IF(JEXTRA.LT.1.OR.JEXTRA.GT.3) JEXTRA = 1
   IF(JWEIGH.LT.1.OR.JWEIGH.GT.5) JWEIGH = 1
   IF(JGRAM .LT.0.OR.JGRAM .GT.1) JGRAM = 0

C --- SPECIAL TRICKS FOR INITIAL STRAIN AND CONSISTENT MASS ITERATION --
C
C IF( NN( 6 ) .NE. 0 ) JISTRN = 1
C IF( NN( 5 ) .NE. 0 ) JCITER = NN( 5 )
GO TO 998

C**********************************************************************
C OPTION 15 : *STRE - MAXIMUM NUMBER OF STRESS BOUNDARY CONDITIONS
C**********************************************************************
C
15 CONTINUE
   NSTRBC = NN(1)
GO TO 998

C**********************************************************************
C OPTION 17 : *TEST FOR THE INTERNAL USE AT MARC DEVELOPMENT
C GROUP ONLY - TO INVOKE THIS IS POTENTIALLY DANGEROUS
C**********************************************************************
C
17 CONTINUE
   ISTAT = 0
   IDYNM = 0
   ITEST = 1
GO TO 998

C**********************************************************************
C OPTION 18 : *DYNA - TRANSIENT TIME INTEGRATION PARAMETER
C COULD BE SPECIFIED IN VERSION 2.0 OR UP
C**********************************************************************
C
18 CONTINUE
   JDYN = NN( 1 )
   IF(JDYN .LE. 0) JDYN = 1

SYSTEMS' MANUAL
Page : 119
IF(JDYN .GT. 2) JDYN = 2
ISTAT = 0
IDYNM = 1
ITEST = 0
GO TO 998

C **********************************************************************
C OPTION 19 : *OPTI - BANDWIDTH OPTIMIZER ITERATION CYCLES
C **********************************************************************

19 CONTINUE
JOPTIM = NN(1)
IF(JOPTIM.EQ.0) JOPTIM = 10
GO TO 998

C **********************************************************************
C OPTION 20 : *TRAC - DUMMY - SAME AS OPTION 10
C **********************************************************************

20 CONTINUE
JDIST = 1
GO TO 998

C **********************************************************************
C OPTION 21 : *CREE - CREEP AND ITS TIME STEP CONTROL PARAMETERS
C **********************************************************************

21 CONTINUE
JCREEP = 1
NCREEP = 3
ATOLER = 0.5D0
BTOLER = 0.5D-1
CTOLER = 0.5D-1

IF( NN(1) .EQ. 0 ) GO TO 2101
NCREEP = NN(1)

CALL FREFOR
1 ( IWORK(ILAST+1), RWORK(ILAST+1), 0, 3, 0, IERR, JKEY )

J = ILAST + 1
IF( RWORK(J) .NE. 0.0 ) CALL COPYSD ( RWORK(J), ATOLER, 1 )
J = ILAST + 2
IF( RWORK(J) .NE. 0.0 ) CALL COPYSD ( RWORK(J), BTOLER, 1 )
\[ J = \text{ILAST} + 3 \]

\[
\text{IF}( RWORK(J) \neq 0.0 ) \text{ CALL COPYSD ( RWORK(J), CTOLER, 1 )}
\]

2101 CONTINUE
GO TO 998

C**********************************************************************
C \text{OPTIC \_} 21 : \text{*ANIS - ANISOTROPIC ELASTICITY}
C**********************************************************************

22 CONTINUE
NONISO=1
GO TO 998

C**********************************************************************
C \text{OPTION} 23 : \text{*MODAL - MODAL ANALYSIS OPTION AND PARAMETER SET}
C**********************************************************************

23 NDYNMD=NN(1)
NSBNC = NN(2)
INSTR=NN(3)
IF ( NDYNMD .EQ. 0 ) NDYNMD = 1
IF ( NSBNC .EQ. 0 ) NSBNC = NDYNMD * 2
MDYNMD = NDYNMD + 8

IF ( NSBNC .GT. MDYNMD) NSBNC = MDYNMD

JEIGEN = 1
LDYN = 1
IDYNM = 1
ISTAT = 0

CALL NULINT(NN,4)
JKEY = 0
CALL FREFOR(NN,NN,1,0,0,IERR,JKEY)
IF ( JKEY .EQ. 1 ) GO TO 998
NSUPER = NN(1)

LDYN = 2
GO TO 998

C**********************************************************************
C \text{OPTION} 24 : \text{*BUCK - BUCKLING ANALYSIS AND PARAMETERS}
C**********************************************************************

24 NDYNMD = NN(1)
NSBNC = NN(2)
INISTR = NN(3)

C

IF( NDYNMD .EQ. 0 )
   NDYNMD = 1
IF( NSBNC .EQ. 0 )
   NSBNC = 2 * NDYNMD
   MSBNC = 8 + NDYNMD
   NSBNC = MIN0( NSBNC, MSBNC )

C
ISTAT = 1
IDYNM = 0
JEIGEN = 1

GO TO 998

C

OPTION 25 : *THER - TEMPERATURE DEPENDENT PROPERTIES

C
CONTINUE
I THERM = 1
GO TO 998

C

OPTION 25 : *CONS - CONSTITUTIVE LAW SELECTION

C

CONTINUE
JCONST = NN(1)
IF( JCONST.LT.0.OR.JCONST.GT.4 ) JCONST = 2
GO TO 998

C

OPTION 27 : *DIST - DISTRIBUTED LOAD ( SAME AS *TRAC AND *PRES )

C

CONTINUE
JDIST = 1
GO TO 998

C

OPTION 28 : *DUPL - MAXIMUM NUMBER OF DUPLICATED NODES

C

CONTINUE
NDUP = NN(1)
GO TO 998
OPTION 29: *REPO - REPORT GENERATION INTERVAL TO BE SET

29 CONTINUE
   JREPOT = NN(1)
   IF(NN(1) .EQ. 0) JREPOT = 1
   GO TO 998

OPTION 30: *TANG - MODIFIED NEWTON METHOD WITH TANGENT MATRIX SPECIFICATION

30 CONTINUE
   JTANG = NN(1)
   GO TO 998

31 CONTINUE
   JIHERM = 1
   GO TO 998

OPTION 32: *SCHF - TIME STEPPING SCHEME PARAMETER OPTION

32 CONTINUE
   DALPHA = 0.5D0
   DBETA = 0.25D0
   DGAMMA = 0.5D0
   CALL FREFOR(IWORK(ILAST+1),R_3RK(ILAST+1),0,3,0,IERR,JKEY)
   CALL COPYSD(R_3RK(ILAST+1),DALPHA,1)
   CALL COPYSD(R_3RK(ILAST+2),DBETA,1)
   CALL COPYSD(R_3RK(ILAST+3),DGAMMA,1)
   GO TO 998

OPTIONS 33, 34, 35, 36, 37, 38, 39 - FLAGS FOR USER SUBROUTINES

33 CONTINUE
   JFORCE = 1
   GO TO 998

34 CONTINUE
JUTEMP = 1
GO TO 998

35 CONTINUE
JUCOEF = 1
GO TO 998

36 CONTINUE
JDISTS = 1
GO TO 998

37 CONTINUE
JUHOOK = 1
GO TO 998

38 CONTINUE
JDERIV = 1
GO TO 998

39 CONTINUE
JUBOUN = 1
GO TO 998

******************************************************************************
OPTION 40 : *PERI - PERIODIC LOADING APPLICABLE ONLY FOR TRANSIENT DYNAMICS
******************************************************************************

40 CONTINUE
JPEROD( 1 ) = NN( 1 )
JPEROD( 2 ) = NN( 2 )
GO TO 998

******************************************************************************
OPTION 41 : *BAND - BAND MATRIX SOLVER ( DEFAULT )
******************************************************************************

41 CONTINUE
JBAND = 1
JFRONT = 0
GO TO 998

******************************************************************************
OPTION 42 : *FRON - FRONTAL SOLUTION
******************************************************************************
CONTINUE
JBAND = 0
JFRONT = 1
GO TO 998

OPTION 43: *DEFO - EIGENVALUE EXTRACTION FOR THE STIFFNESS
MATRX FLAGGED

CONTINUE
JDEFOR = 1
IDYNMD = NN( 1 )
NDYNMD = NN( 2 )
NSBNC = NN( 3 )
INTSTR = NN( 4 )

IF ( NDYNMD .EQ. 0 ) NDYNMD = 1
MSBNC = 2 * NDYNMD
IF ( NDYNMD .GT. 8 ) NSBNC = MSBNC

ISTAT = 1
IDYNM = 0
JEIGEN = 1
GO TO 998

OPTION 44: *EMBE - EMBEDDED MULTIPLE SINGULARITIES

CONTINUE
JEMBED = 1
JSUBRE = 1
GO TO 998

OPTION 45: *GMRS - MAXIMUM NUMBER OF MULTIPLE GENERIC REGIONS

CONTINUE
NGMRS = NN( 1 )
IF( NGMR = .EQ. 0 ) NGMR = 1
GO TO 998
C
C **********************************************************************
C OPTION 46 : *BEAM - MAXIMUM NUMBER OF BEAM SECTION DATA
C **********************************************************************
C
46 CONTINUE
NBSECT = NNODE
GO TO 998
C
C **********************************************************************
C OPTION 47 : *DISP - DISPLACEMENT METHOD OPTION
C **********************************************************************
C
47 CONTINUE
JDISP = 1
GO TO 998
C
C **********************************************************************
C OPTION 48 : *POWE - POWER SHIFT FOR EIGEN ANALYSIS
C **********************************************************************
C
48 CONTINUE
NSHIFT = NN( 1 )
IF( NSHIFT .EQ. 0 ) NSHIFT = 1
GO TO 998
C
C **********************************************************************
C OPTION 49 : *BFGS - BFGS UPDATE FOR QUASI-STATIC NONLINEAR
C ANALYSIS
C **********************************************************************
C
49 CONTINUE
IFBFGS = 1
NSBFGS = 10
IF( NN(1) .NE. 0 ) NSBFGS = NN( 1 )
GO TO 998
C
C **********************************************************************
C OPTION 50 : *SPRI - ADDED STIFFNESS, GROUND SPRING
C **********************************************************************
C
50 CONTINUE
NSPRI = NN(1)
IF ( NSPRI .EQ. 0 ) NSPRI = 1
GO TO 998
C
C*******************************************************************************
C
C OPTION 51: *DASH - ADDED DAMPING, DASHPOT TO GROUND
C*******************************************************************************
C
51 CONTINUE
    NDASH = NN(1)
    IF ( NDASH .EQ. 0 ) NDASH = 1
GO TO 998
C
C*******************************************************************************
C
C OPTION 52: *MASS - ADDED MASS
C*******************************************************************************
C
52 CONTINUE
    NMASS = NN(1)
    IF ( NMASS .EQ. 0 ) NMASS = 1
GO TO 998
C
C*******************************************************************************
C
C OPTION 53: *SECA - SECANT NEWTON METHOD OPTION
C*******************************************************************************
C
53 CONTINUE
    IFSCNT = 1
GO TO 998
C
C*******************************************************************************
C
C OPTION 54: *LINE - LINE SEARCH OPTION
C*******************************************************************************
C
54 CONTINUE
    IFLINE = 1
GO TO 998
C
C*******************************************************************************
C
C OPTION 55: *HARM - HARMONIC NODAL FORCE LOADING
C*******************************************************************************
C
55 CONTINUE
    NHARM = NN(1)
    IF ( NHARM .EQ. 0 ) NHARM = 1
    CALL FREFOR(IWORK(ILAST+1),IWKORD(ILAST+1),0,1,0,IERR,JKEY)
CALL COPYSD(RWORK(ILAST+1),OMEGH,1)
GO TO 998

C *****************************************************************************
C     OPTION 56 : *BASE - HARMONIC BASE EXCITATION
C *****************************************************************************

      CONTINUE
      NBASE = NN(1)
      IF (NBASE .EQ. 0) NBASE = 1
      CALL FREFOR(IWORK(ILAST+1),RWORK(ILAST+1),0,1,0,IERR,JKEY)
      CALL COPYSD(RWORK(ILAST+1),OMEGB,1)
      GO TO 998

C *****************************************************************************
C     OPTION 57 : _ITE IAMINATE OPTI_
C *****************************************************************************

      CONTINUE
      ICOMPS = 1
      CALL CMPDF(1,IWORK,RWORK,IERR,NTYPE,ILAST,ILPRNT)
      GO TO 998

C *****************************************************************************
C     OPTION 58 : _FULS - PULSE LOAD OPTIONS
C *****************************************************************************

      CONTINUE
      NPDPTS = NN(1)
      NPULSE = NN(2)
      IF (NPDPTS .LT. 2) NPDPTS = 2
      IF (NPULSE .EQ. 0) NPULSE = 1
      GO TO 998

C *****************************************************************************
C     OPTION 59 : PRECONDITIONED CONJUGATE GRADIENT ITERATION OPTION
C *****************************************************************************

      CONTINUE
      IPCONJ = 1
      IFLINE = 1
      GO TO 998
OPTION 60: *SHOC - SHOCK SPECTRA OPTION (INACTIVE)

```
60 CONTINUE
   CALL LINES(1,1)
   WRITE(ILPRNT,6000) 61,'SHOC'
   WRITE(ICONSL,6000) 61,'SHOC'
   IERR = IERR+1
   NSSPTS = NN(1)
   IF ( NSSPTS .EQ. 0 ) NSSPTS = 10
   LDYN = 3
   GO TO 998
```

OPTION 61: *POWE - POWER SPECTRAL DENSITY OPTION

```
61 CONTINUE
   NPSPTS = NN(1)
   NFDPTS = NN(2)
   JFDSXX = NN(3)
   IF ( NPSPTS .EQ. 0 ) NPSPTS = 10
   IF ( NFDPTS .EQ. 0 ) NFDPTS = 10
   LDYN = 4
   GO TO 998
```

OPTION 62: *LOGCH - LOG CHARTS

```
62 CONTINUE
   LOECHO = 1
   GO TO 998
```

OPTION 63: *PERT - SET UP PERTURBATION FLAGS

```
63 CONTINUE
   CALL LINES(1,1)
   WRITE(ILPRNT,6000) 63,'PERT'
   WRITE(ICONSL,6000) 63,'PERT'
   IERR = IERR+1
   CALL PERSIZ( NN(1), IERR )
   GO TO 998
```
C
C ************************************************************
C  OPTION 64 : *STIF - STRESS STIFFENING OPTION
C ************************************************************
C
64 CONTINUE
    JISTIF = NN(1)
    IF( NN(1).EQ.0 ) JISTIF = 1
    GO TO 998
C
C ************************************************************
C  OPTION 65 : *CENT - CENTRIFUGAL MASS STIFFNESS OPTION
C ************************************************************
C
65 CONTINUE
    JCENIM = NN(1)
    GO TO 998
C
C ************************************************************
C  OPTION 66 : *HARD - WORK-HARDENING OPTION FOR PLASTICITY
C ************************************************************
C
66 CONTINUE
    NHARD = NN(1)
    IF (NN(1) .EQ. 0) NHARD = 1
    GO TO 998
C
C ************************************************************
C  OPTION 67 : *FINIT - FINITE STRAIN OPTION
C ************************************************************
C
67 CONTINUE
    JFINIT = NN(1)
    GO TO 998
C
C ************************************************************
C  OPTION 68 : *LARG - LARGE DISPLACEMENTS AND ROTATIONS OPTION
C ************************************************************
C
68 CONTINUE
    JLARGE = NN(1)
    GO TO 998
C
C ************************************************************
C  OPTION 69 : *FOLL - FOLLOWER FORCE OPTION
C
69 CONTINUE
   JPOLOW  = NN(1)
   GO TO 998
C
C *************************************************
C OPTION 70 : *WLSKP : FLAGS THE USER SUBROUTINE FOR WORKHARDENING
C *************************************************
C
70 CONTINUE
   JWLSLP  = 1
   GO TO 998
C
C *************************************************
C OPTION 71 : *HOUR ... HOURGLASS CONTROL FLAG IN A SPECIAL WAY
C *************************************************
C
71 CONTINUE
   JHURGLS = 1
   GO TO 998
C
C *************************************************
C OPTION 72 : *MDNIT - _JRN ON THE MDNITOR UTILITY
C *************************************************
C
72 CONTINUE
   NMZIIT = NN(1)
   IF (NMNZIT .LT. I) NMNZIT = 1
   GO TO 998
C
C *** NORMAL EXIT BY READING '*END' CARD *******************
C
16 CONTINUE
   IF ( LOECHO .NE. 0 ) NOECHO = 1
C
C *** A FEW POSSIBLE CONTRADICTIONS IN PARAMETER DATA ARE CHECKED HERE
C
   IF( IFBG .EQ. 1 .AND. IFSCNT .EQ. 1 ) CALL QUIT
      ('CONT','RADI','CTIO','N DE','TECT','ED ', 1 )
C
   IF( IFBG .EQ. 1 .AND. IPGNJ .EQ. 1 ) CALL QUIT
      ('CONT','RADI','CTIO','N DE','TECT','ED ', 1 )
C
   IF( IPGNJ .EQ. 1 .AND. IFSCNT .EQ. 1 ) CALL QUIT
An example of a typical bulk data reader is the subroutine to read and store the nodal coordinates:

```
C=SUBROUTINE=COORIN CALLED BY SUBROUTINE 'DATIN2'
SUBROUTINE COORIN
  1 (INT ,REA ,IERR ,NCRD ,ILAST ,ILPRINT,NNODE,INOD ,
    2 MAXCRD,ICHECK)
C
C **********************************************************************
C READ IN COORDINATE DATA
C
C IERR    ERROR FLAG
C NCRD     NUMBER OF COORDINATES PER NODE
```

---

C
C IF( JFINIT.LT.JLARGE ) CALL QUIT
& ( '*FIN','I ST','ARTS',' B4 ','*LAR','GE ',' 0 )
C
IF( JFINIT.LT.JLARGE ) THEN
  IF( JFINIT.NE.999999 ) CALL QUIT
  IF( JLARGE.NE.999999 .AND. JISTIF.EQ.999999 ) CALL PRWARN
ENDIF

IF( JLARGE.EQ.999999 ) THEN
  IF( JFINIT.LT.JLARGE ) CALL QUIT
  IF( JFINIT.NE.999999 ) CALL QUIT
ENDIF

IF( JFINIT.LT.JLARGE ) THEN
  IF( JFINIT.NE.999999 ) CALL QUIT
END IF

C
C *** CONTROL DATA ERROR ***********************************************
C
C 3001 CONTINUE
WRITE(ICONSL,9000)
CALL QUIT('INPU','T ',' ',' ',' ',' ',' ',' ',' ',' ',1)

C
C *** FORMAT STATEMENTS ***********************************************
C
C 6000 FORMAT(//'1X,23H***ERROR*** OPTION ,I2,3H *,A4,11H NOT ACTIVE)
C 9000 FORMAT(//'28H ********* ERROR *********,,14H NO INPUT FILE,//)
C
```

---

SYSTEMS' MANUAL

Page : 132
LAST WORD FOR INPUT
KW OUTPUT DEVICE
NNODE NUMBER OF NODES IN MESH
INOD POINTER TO COORDINATES
MAXCRD MAXIMUM NUMBER OF COORDINATES PER NODE
ICHECK ENABLE/DISABLE NODAL THICKNESS CHECK

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

IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 REA

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

COMMON / MACHIN / IDP

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

DIMENSION INT( 1 ), REA( 1 )

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

IS1 = INOD - MAXCRD*IDP
JKEY = -1

*** CHECK IF THE CORE IS ALLOCATED FOR THE COORDINATE DATA ***********

IF ( NNODE .GT. 0 ) GO TO 208

CALL LINES( 1 , 1 )
WRITE(ILPRINT,9108)
9108 FORMAT(2X,48H***ERROR***
1 5H DATA)
IERR = IERR + 1

*** CHECK THE NUMBER OF DATA ENTRIES ****************************

CALL LINES( 1 , 1 )
WRITE(ILPRINT,9108)
9108 FORMAT(2X,48H***ERROR***
1 5H DATA)
IERR = IERR + 1

CONTINUE

IF ( NCRD .EQ. 0 ) NCRD = MAXCRD

*** MCRD: COUNTER OF THE NUMBER OF ACTUAL COORDINATE DATA ENTRIES ****

MCRD = NCRD

IF ( MAXCRD .EQ. 7 ) MCRD = 3
IF ( NCRD .LE. MAXCRD ) GO TO 607
CALL LINES( 1, 1 )
WRITE(ILPRINT,9208) NCRD,MAXCRD
9208 Format(2X,47H***ERROR*** NUMBER OF COORDINATE DIRECTIONS,I5, *13H GREATER THAN,I5) IERR = IERR + 1

C *** READ NUMERIC DATA LINE *******************************************

C 607

C

C 608 CONTINUE

C ICOUNT = ICOUNT + 1

C CALL NULINT(INT(ILAST+1),NCRD+1)
CALL FREFOR(INT(ILAST+1),REA(ILAST+1),1,NCRD,0,IERR,JKEY)

C *** IF THE CURRENT LINE IS THE KEYWORD DATA THEN RETURN ************

C IF ( JKEY .EQ. 1 ) GO TO 108

C**********************************************************************
C PROCESS THE NODAL COORDINATE DEFINITION DATA LINE
C**********************************************************************

C TOLD = TNEW
K = INT(ILAST+1)
TNEW = REA(ILAST+5)

C *** IF THICKNESS IS NOT GIVEN THE LAST NONZERO VALUE IS TAKEN ********
C IF ( TNEW .EQ. 0.D0 ) TNEW = TOLD

C *** RANGE CHECK ****************************************************
C IF ( K .GT. 0 .AND. K .LE. NNODE ) GO TO 308

C *** ERROR MESSAGES ***********************************************
C CALL LINES( 1, 1 )
IF ( K .LE. 0 ) WRITE(ILPRINT,9601) K
9601 Format(2X,27H***ERROR*** NODE NUMBER,I5,13H NON-POSITIVE)
C ***
   IF ( K .GT. NNODE ) WRITE(ILPRNT,9701) K, NNODE
9701 FORMAT(2X,27H***ERROR*** NODE NUMBER,15,13H GREATER THAN,15)  
               IERR = IERR + 1
308 CONTINUE
C
C *** THICKNESS DATA CHECK FOR SHELL ELEMENTS  *************************************************
C
   IF ( MAXCRD .NE. 7 .OR. ICHECK .EQ. 0 ) GO TO 8000
   IF ( ICOUNT .NE. 1 .OR. TNEW .NE. 0.D0 ) GO TO 8000
C
C *** ERROR MESSAGE FOR THE THICKNESS NOT DEFINED  *************************************************
C
   WRITE(ILPRNT,9753)
9753 FORMAT(2X,'***ERROR*** SHELL THICKNESS NOT SPECIFIED IN THE ',  
               'FIRST DATA LINE')  
               IERR = IERR + 1
C
C *** ADDRESSES FOR THE CURRENT (K-TH) NODAL COORDINATES  *************************************
C
8000 CONTINUE
   IS2 = IS1 + K*MAXCRD*IDP
   IS3 = IS2 + 6*IDP
C
C *** STORE THE COORDINATE DATA  **********************************************
C
   CALL COPYSD (REA(ILAST+2), REA(IS2), MCRD)
   IF ( MAXCRD .NE. 7 ) GO TO 408
C
C *** STORE THE SHELL THICKNESS DATA AT NODE  *************************************
C
   CALL COPY ( TNEW, REA(IS3), 1)
C
408 CONTINUE
508 CONTINUE
C
C *** BACK TO THE FREE FORMAT READER AND PROCESS THE NEXT LINE  *******
C
   GO TO 608
C
C *** EXIT  ***************************************
C
3.3 Restart File

The restart file is designed to store all the information necessary to resume execution of an incremental analysis or to use the final result stored in the file for starting a new set of calculations. The alteration of certain parameter data is supported by the following SUBROUTINES: SAVER to write and RESTART to read the restart file:

C ... SUBROUTINE SAVER ... RESTART FILE WRITER
C
SUBROUTINE SAVER( RWORK, IWORK, ISIZE )
C
**********************************************************************
C ** WRITES A BINARY FILE FOR LATER RESTART **
C **********************************************************************
C
IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 RWORK
C
DIMENSION IWORK(ISIZE), RWORK(ISIZE)
C
COMMON / ADDVAL / ISPRI , KSPRI , IDASH , KDASH , IMASS , KMASS
COMMON / ALGEM / ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1     IPLOTS, IRSTRT, JCREAD, IPVARS, IPSETS, IFILEX,
2     JADAP, NCREEP, SCALE
COMMON / AUTOIN / CURPER, TOTPER, ARCLEN, ATOLER, BTOLER, CTOLER,
1     JADAP, NCREEP, SCALE
COMMON / BODYFR / POINTS( 3, 2)
COMMON / BSECT / IBSECT, KBSECT
COMMON / CONTRO / JEND , JITER , JTEMP , JPRINT , JP , JSUB ,
1     JINC , JREST , JSAVE , JREDIM, JAUTO , JPOST ,
2     JBACK , JOPTIM, JCREEP, JDIST , JCONST, JDYN ,
3     NONISO, IThERM, ITRIG , IDYNM , JREPOT, JTANGE,
COMMON / COMPND / NXSTAT, NXSOLV, NXINIG, NXM00L, NXBCKL, NXSUPR,
1 NXSEQN, NXDM1, NXDM2, NXDM3, NXDM4
COMMON / COUNT / LININC, LINTOT, NOECHO
COMMON / DAMP / DAMPF(3)
COMMON / EIGEN / IEIGNC, IGEMS, IOMEG, IOMINO, IDYNMD, ISTRT2,
1 IPTAR, IPTBR, IPTVED, IMDAM, IOMEGD
COMMON / MDSUP / IMFOR0, IMDISO, IMVELO, IMFOR1, IMDIS1, IMVEL1
COMMON / ELEMEN / IC, IEL, IDF, JLAW, IPATH, IASSEM
1 COMMON / HARMON / OMEGH, IHARM, KHARM, OMEGB, IBASE, KBASE,
1 COMMON / LOUBIN / JLOUB, JINTER, JEXTRA, JWEIGH, JSUBRE, JISTRN,
1 COMMON / PARAM / NTYPC, NELEM, NNODE, NBC, NTIE, NMAX,
1 COMMON / PERPAR / IPTYPE(32), NPTYPE, NPVARS, NPSETS, JPERT,
1 COMMON / PERPTR / IMEANS, ISTDEV, IPDATA, IVTYPE, ISKIP, IREDEF,
1 COMMON / PERDAT / IXCOOR, IXCHAR, IXFORC, XXFORC, XDXDIST, XXDIST,
1 COMMON / PERIOD / JPEROD(2), IPDISP, IPFORC, INDISP, INFORC
COMMON / POSTPN / IPOINT, JPOINT, KPOINT, NDATA, PRNTBF(6)
COMMON / POWER / IELPHI,IELTMM,IEPSNO,ISIGMD,IHFN ,IHFC , IFBP ,ISPP ,ISFF ,ISQQ ,ICQQ ,ITNM, IPSF ,IPSD
COMMON / PREPAR / NRFPTS,NRFSDS,NRFLIN,NRFINT,NRFREA,NRFSEL, NRFCMP,NRFCOF,NRFSTZ, JRFPND,JRTYPE, JCFLG, JRFSEL, JRFDOM,NKPT,MORE, NESSUS
COMMON / PREPTR / IRFPTS,IRFINT,IRFMNR,IRFSDR,IRFSEL,IRFSET, NESSUS
COMMON / PULSES / IPULSE,KPULSE,IPDTIM,IPDFOR
COMMON / RESULT / MANVAR(7), JPR, ICOM1, NCOMP
COMMON / START1 / IELPRM,ITYP,INEL,ICHAR,IPRES,ISTR,S, ISTRN, ICP, IPRINT,IPOST,ISTRT ,LEAN, IBPRES,IGNORM,IMONIT,IST116,IST117,IST118
COMMON / START2 / INOD ,ITEM ,INLV ,IPOSU ,ITEMDP, IDOP
COMMON / START3 / IKBC ,TI ,ITR ,ITRAC ,IEXT , ISBC ,ISBCR
COMMON / START4 / IDINC ,IDTOT ,IFORC ,IRESID, IWINOD,ISIGNO, IEPSNO,IPSTRN,ICSTRN,ITSTRN,ISTSTRN, IIPSTN,IIIC,STR,ITPSNO,ICSTNO,ITSTNO, ILSINO,ILSSNO,ILPSNO,ICSNO,ITCSSO, IDMNO,IEQCSST,ICOMNO,ITDSNO,IVSWIO, IDYNV ,IDYNA,IDSX1 ,IDSX2, IDSTTR,ISWELL, IEQCS,IPREF,IDSX3, TYIELD, IDPSNO,-binding, IPFTOT, IST444,IST445,IST446,IST447
COMMON / START5 / IRL ,IREAC ,IES ,IAB ,IBQM ,ISRL , IBTLC ,ISKML,ILAST,IRLE, IDINC,IFORIN, IOP ,IDAM ,IMASMT,IDIAG, IQPT, IMSDX,IMASUP,IST521,IST522,IST523,IST524
COMMON / START6 / IELV ,ICOR ,ISIG ,IEPS ,IWINOD ,ISIGNO, IENOD,IEPM,ICL,IPP,IXRL ,IXTR , IXP ,IXK ,KPSRN,KCSTRN,KTSRN,KIRKAS, KISTRN,KPSRT,KCSTRK,TSTRN,KPSTNO,KCSSNO, KIPSTN,KISNO,KIPSNO,KICSSNO,KITCSS, KMASNO,IMNOD,IEQPSST,IEQPSI,KEQPSI, KDMAT,KIDNO,KDSNO,KIDST,IXM,IXC, IVELM,IAELM,IMASEL,KYIELD,IST647,IST648, IST649,IST650,IST651,IST652,IST653,IST654
COMMON / START7 / ICON ,IKBCR,ITRACR,ITRANR,IBETA, IDET
COMMON / START8 / KGEPS,KGCPS,KGCSS,KGISIG, KGSIG,KGIDST, IGEPSN,IIGEN,IKISNO,IKIDSNO,IGIDNO, IGEPSN,KIGENO,IKISNO,IKIDSNO, KGIDNO
COMMON / START9 / KEQCS,IKOMNO,KSWLNO,KIMN0,KEQPSI,KEQPS, KOMNO,KVSWIO,ISTA0,IST910,IST911,IST912
COMMON / SUBELM / ISUBEL,ISUBNP,ISUBPT,NSDATA,ISUBTY, IEMBED
** Rewind the restart file to overwrite with the newest data **

** First store the contents of the common blocks **

```
WRITE (IRSTR) ISPRI, KSPRI, IDASH, KDASH, IMASS, KMSS
WRITE (IRSTR) ICREAD, IPRINT, JPRINT, IONSL, IPOSTF, ISCRAF,
                IPRINT2, JPRINT, JPRINT2, IPRINT3, IPRINT4,
                IPFIT, JPRINT, JPRINT2, JPRINT3, JPRINT4,
                JPRINT5, JPRINT6
WRITE (IRSTR) PI, LINE, LINE2
WRITE (IRSTR) CURPER, TOTPER, ARCLN, ATOLER, BTOLER, CTOLER,
                JADAP, NCREEP, SCALE
WRITE (IRSTR) POINTS( 3, 2)
WRITE (IRSTR) IBSEP, KSECT
WRITE (IRSTR) JEND, JITER, JTEMP, JPRINT, JP, JSUB,
                JINC, JREST, JSAVE, JREDIM, JALTO, JPOST,
                JBACK, JOPTIM, JCREEP, JDIST, JCONST, JDYN,
                NONISO, IHERM, ITRIG, IDYNM, JREPOT, JTANGE,
                JHERM, JFORCE, JUTEMP, JUCERF, JDIST3, JUHOCK,
                JDERIV, JBOUND, IDSTOP, INSTR, JPLAST, JEND,
                JFRONT, JDEFOR, JEMBED, ITEST, JDISP, JFBFS,
                IFSCNT, IFLINE, IPRINT, ICOMPS, ICONF, JEIGEN,
                IFBODY, IFGRAV, IFCENT, JDAMP, LDYN, ISTAT,
                JFDSXX, JISTIF, JCENTM, JFINIT, JARGE, JFOLLOW,
                JWKSLE, JPRIS, JCDUM2, JCDUM3
WRITE (IRSTR) NXSTAT, NXSOLV, NXSOLM, NXMOD, NXMODL, NXMODL, NXSUPR,
                NXSUPR, NXREQN, NXDUM1, NXDUM2, NXDUM3, NXDUM4
```
WRITE (IRSTR) LININC, LINTOT, NOECHO
WRITE (IRSTR) DAMPF(3)
WRITE (IRSTR) IEQMYC, IQEAG, IQMEG, IOMENO, IDYNMD, ISTR2,
WRITE (IRSTR) IPTAR, IPTBR, IPTVEC, IMADAM, IOMEG
WRITE (IRSTR) IMPORE, IMDISO, IMVELO, IMFORI, IMDI1, IMVEL1,
WRITE (IRSTR) IC, IEL, IDF, IJAW, IPATH, IASSEM,
WRITE (IRSTR) JRULE, JCART, JEL009, JEL010, JEL011, JEL012
WRITE (IRSTR) OMEG, IHARM, RHARM, OMGB, IBASE, KBASE,
WRITE (IRSTR) IGNFOR, ICMFOR, ICRES, ICHEFH, ICBEXC, ICMAT
WRITE (IRSTR) FACTOR, INCFLG(20)
WRITE (IRSTR) MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
WRITE (IRSTR) MAXLAY, MAXINT, MAXWRK, MAXNUL, NSUMAX, MAXCMP,
WRITE (IRSTR) MAXBSP, MAXGMR, MAXTEM, MAXHARM, MAXLINK, MAXDTM,
WRITE (IRSTR) MAXPRN, MAXSET, MAXVAR, MAXEAE, MAXORD,
WRITE (IRSTR) MAX025, MAX026, MAX027, MAX028, MAX029, MAX030,
WRITE (IRSTR) JLOUB, JINTER, JEXTRA, JWEIGH, JSUBRE, JSTRN,
WRITE (IRSTR) JCITY, JHRG, JHARM, LOUB03, LOUB04, LOUB05
WRITE (IRSTR) NTYPR, NELEM, NNODS, NEC, NTIE, NMAX,
WRITE (IRSTR) NTRAN, NTRAC, NFD, NBEAM, NEXT, NSUB,
WRITE (IRSTR) NPINT, NPOST, NSFC, NDUP, NSIZE, NSECT,
WRITE (IRSTR) NSHIFT, NSBEGS, NGMRS, NSPRI, NMASS, NDASH,
WRITE (IRSTR) NDYNMD, NSBEC, NSUPER, NHARM, NSBASE, NINC,
WRITE (IRSTR) NTIER, NPSPTS, NFDP, NPULSE, NPDP, NHARD,
WRITE (IRSTR) NSUMCH, NPR28, NMONIT, NPAR40, NPAR41, NPAR42,
WRITE (IRSTR) NPAR43, NPAR44, NPAR45, NPAR46, NPAR47, NPAR48,
WRITE (IRSTR) IPTYPR(32), NTPR, NFVAR, NPSET, JPERL,
WRITE (IRSTR) NPVON, NPPO08, NPPO09, NPPO10, NPPO11, NPPO12
WRITE (IRSTR) IMEANS, ISDEV, IPDATA, IVTYPE, ISKIP, IREDIF,
WRITE (IRSTR) IDINCO, IREDAC, IRESD0, IDGRP, ISTIF, IDASS0,
WRITE (IRSTR) IPP013, IQEAG, IOMEGP, IQMEGP, IETAK, IETAK,
WRITE (IRSTR) LXCOOR, IXCHAR, IXFORC, IXFORC, IXDIST, KXDIST,
WRITE (IRSTR) IXTEM, JXTEM, JXBEAM, IXFVEC, IXSPRI, KXSPRI,
WRITE (IRSTR) IXPRF, IXPREF, IXPO15, IXPO16, IPB, IPBEG, IPBEND,
WRITE (IRSTR) JPERQD(2), IPDISP, IPPFORC, INDISP, INFORC
WRITE (IRSTR) JPOINT, JPOINT, JPOINT, NDATA, PRMTF(6)
WRITE (IRSTR) IELPHI, IELTNM, IEPSSM, ISIGMO, IHEF, IHFC,
WRITE (IRSTR) IFBP, ISFP, ISP, ISQ, IQQQ, IQNM,
WRITE (IRSTR) IPSF, IPSD
WRITE (IRSTR) NRPPTS, NRFSDS, NRFHIN, NRFINT, NRFREA, NRFSEL,
WRITE (IRSTR) NRFCM, NRFPL, NRFSSS, LRFD, JRPF, JCOUNT,
WRITE (IRSTR) JCLLAG, JRPS, JRPSD, NMSHL, MORE
WRITE (IRSTR) IRPPTS, IRFIN, IRFMNR, IRPFR, IRPSR, IRPSR,
WRITE (IRSTR) IRFNAV, IRFSD, IRFTV, IRFVEC, IRFVAL, IRFNM,
WRITE (IRSTR) IRFCO, IRFID, IRFWRK

SYSTEM'S MANUAL

Page: 140

MHOST Version 4.2
WRITE (IRSTRT) MAXCOL, NELCOL, LAYPRT, JTENS

C**********************************************************************
C ** NEXT STORE THE DATA IN BLANK COMMON IN 64 WORD BLOCKS **
C**********************************************************************

NWORDS = 64
NRECDS = IIAST / NWORDS

DO 600 JREC = I, NRECDS
   IBEG = I+(JREC-1)*NWORDS
   IEND = JREC*NWORDS
   WRITE (IRSTRT) (IADRK(JJ), JJ=IBEG, IEND )
600 CONTINUE

IBEG = 1+NRECDS*NWORDS
IEND = ILAST
IF (IBEG .NE. IEND)
   1 WRITE (IRSTRT) (IWORK(JJ), JJ=IBEG, IEND )

C**********************************************************************
C ** PRINT MESSAGE AND GO ON TO THE NEXT INCREMENT **
C**********************************************************************

CALL TIMDUT ('REST','ART ','FILE',' GEN','ERAT','ED ')
RETURN
END

C ... SUBROUTINE RESTRT ... RESTART FILE READER
C
SUBROUTINE RESTRT( RWORK, IWORK, ISIZE )

C**********************************************************************
C **
C ** READS-IN A BINARY FILE FOR PROBLEM RESTART
C **
C**********************************************************************

IWORK : INTEGER WORKSPACE
RWORK : REAL WORKSPACE
ISIZE : SIZE OF WORKSPACE
**DIMENSION IWORK(ISIZE), RWORK(ISIZE)**

**COMMON / ADDVAL / ISPRI, KSPRI, IDASH, KDASH, IMASS, KMASH**

**COMMON / ALGEM / IREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,**

**COMMON / BODYFR / ICREAD, ILPRNT, JLPRNT, ICC_SL, IPOSTF, ISCRAF,**

**COMMON / BSECT / IBSECT, KBSECT**

**COMMON / CONTRO / JEND, JITER, JTEMP, JPRINT, JP, JSUB,**

**COMMON / COMPND / NXSTAT, NXSOLV, NxINTG, NXMODL, NXBCKL, NXSUPR,**

**COMMON / COUNT / LININC, LINTOT, NOECHO**

**COMMON / DAMP / DAMPF(3)**

**COMMON / EIGEN / IEGNVC, IQMNS, IOMEG, IMOENO, IDYNMD, ISTR12,**

**COMMON / MODSUP / IMFORO, IMDISO, IMVELO, IMFOR1, IMDIS1, IMVEL1**

**COMMON / ELEMEN / IC, IEL, IDF, JLAW, IPATH, IASSEM,**

**COMMON / HARMON / OMEG, IHARM, KHARM, OMEGB, IBASE, KBASE,**

**COMMON / HARM2 / ICNFOR, ICMPOR, ICMPRS, ICHFHN, ICHEFHN, ICBEXC,**

**COMMON / INCON / FACTOR, INCFLG(21)**

**COMMON / MAXIMA / MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRES,**

**COMMON / LOUBIN / JLOUB, JINTER, JEXTRA, JWEIGH, JSUBRE, JISTRN,**

**SYSTEMS' MANUAL**

**Page : 143**
COMMON / PARAM / NTYPE, NELEM, NNODE, NBEC, NTIE, NMAX
1 NTRN, NTRAC, NFD, NBAND, NEXT, NSUB
2 NPRINT, NPOST, NSEC, NDUP, NSIZE, NSECT
3 NSHIFT, NSEGS, NGMR, NSPR, NMASS, NDASH
4 NDYNMD, NSEC, NSUP, NBASE, NINC
5 NITER, NSFPTS, NFDPPTS, NPULSE, NPPTS, NHA RD
6 NSUMCH, NDIMEN, NMINT, NPAR40, NPAR41, NPAR42
7 NPAR43, NPAR44, NPAR45, NPAR46, NPAR47, NPAR48

COMMON / PERPAR / IPTYPE(32), NPTYPE, NPVARS, NPSETS, JPERT,
1 NPVCON, NPP008, NPP009, NPP010, NPP011, NPP012
2 IDINC0, IREAC0, IRESDO, IDGRP, ISTIF0, IMASS0
3 ITP013, IOMEGO, IOMEGP, IOMEGK, IBIK, IBIKAK
4 IDINC0, IREAC0, IRESDO, IDGRP, ISTIF0, IMASS0
5 ITP013, IOMEGO, IOMEGP, IOMEGK, IBIK, IBIKAK

COMMON / PERDAT / IXCOOR, IXCHAR, IXFORC, KXFORC, IXTEMP, JXTEMP, IXBEAM, IXVEC, IXSPR, IXSPRI
1 IXPRES, IXPRED, IXP015, IXP016, IPWBEK, IPWEND
2 IXPRES, IXPRED, IXP015, IXP016, IPWBEK, IPWEND

COMMON / PERIOD / JPERIOD(2), IPDISP, IPFORC, INDISP, INFRC
1 JPERIOD(2), IPDISP, IPFORC, INDISP, INFRC
2 JPERIOD(2), IPDISP, IPFORC, INDISP, INFRC

COMMON / POSTRN / IPJNT, KPOINT, NDATA, PRNTBF(6)
1 IPJNT, KPOINT, NDATA, PRNTBF(6)

COMMON / POWER / IELPHL, IELTMN, IEPAMO, ISIGN, IHNF, IIFB
1 IIFB, IPAM0, ISPP, ISQF, ICPQ, ICPQ, ICPQ
2 ICPQ, IPAM0, ISPP, ISQF, ICPQ, ICPQ, ICPQ

COMMON / PREPAR / NRFPTS, NRFSDS, NRFLIN, NRFINT, NRFREA, NRFSEL,
1 NRFCPM, NFRDOF, NRFST, LRFEND, JRFYP, JCOUNT,
2 JRFYP, LRFEND, JRFYP, JCOUNT,

COMMON / PREPTR / NRFPTS, NRFINT, NRFMR, IRSER, IRSF, IRSFT,
1 IRSF, IRSER, IRSFT,
2 IRSF, IRSER, IRSFT,

COMMON / PULSES / IPULSE, KPULSE, IPDTM, IPDFOR
1 IPULSE, KPULSE, IPDTM, IPDFOR

COMMON / RESULT / MANVAR(7), JPR, ICAM, NCAM
1 MANVAR(7), JPR, ICAM, NCAM

COMMON / START1 / IELPM, ITYP, IHEL, ICHAM, IPRES, ISTRN,
1 ISTRN, ICOP, IPINT, NPOST, IPOINT, IDPAY
2 ISTRN, ICOP, IPINT, NPOST, IPOINT, IDPAY

COMMON / START2 / INOD, ITEM, INLV, IPOSU, ITEMDF, IDUP
1 INLV, IPOSU, ITEMDF, IDUP

COMMON / START3 / IEC, ITI, ITR, ITRAN, ITRAC, IEXT
1 IEC, ITI, ITR, ITRAN, ITRAC, IEXT

COMMON / START4 / IDINC, IDTOM, IFORCE, IRESID, IWINOD, ISIGN
1 IWINOD, ISIGN, IWINOD, ISIGN

COMMON / START5 / IREL, IREAC, IES, IAB, IBQM, ISRL,
C **********************************************************************
C ** READ BACK THE CONTENTS OF THE COMMON BLOCKS  **
C **********************************************************************

READ (IRSTRT) ISPRI , KSPRI , IDASH , KDASH , IMASS , KMASS 
READ (IRSTRT) ICREAD , IPRINT , JPRINT , ICONSL , IPPOSTF , ISCRAF 
READ (IRSTRT) IPLTS , IRSRT , JCREAD , IPVARS , IPSETS , IFILEX 
READ (IRSTRT) PI , LINE , LINE2 
READ (IRSTRT) CURPER , TOTPER , ARCLEN , ATOLER , BIOLER , CIOLER 
READ (IRSTRT) JADAP , NCREEP , SCALE 
READ (IRSTRT) POINTS( 3 , 2 ) 
READ (IRSTRT) IBSECT , KBSECT 

C

C
READ (IRSTR)  JEND , JITER , JTEMP , JPRINT , JP , JSUB ,
1 JINC , JREST , JSAVE , JREDIM , JAUTO , JPPOST ,
2 JBACK , JUPTIM , JCREEP , JDIST , JCONST , JDYN ,
3 JNISO , JTERM , JTRIG , IDYNM , JREPOT , JFANG ,
4 JTERM , JFORCE , JUTEMP , JUOEF , JDISTS , JUHOK ,
5 JDERIV , JUBOUN , IDSTOP , INISTR , JPLAST , JRAND ,
6 JFROMI , JEFOR , JEMBED , JTEST , JDISP , JFBEYS ,
7 IFSCMT , IFLINE , IFPRINT , ICOMPS , IPONJ , JEIGEN ,
8 IFBODY , IFGRAV , IFCENT , IDAMP , JLYN , JISTAT ,
9 JFDSXX , JISTIF , JCENIM , JINIT , JLARGE , JFORO ,
+ JWORK , JPRESS , JCDUM2 , JCDUM3 ,
READ (IRSTR)  NXSTAT , NXSOLV , NXINTG , NXMOLES , NXBCKL , NXSUPR ,
1 NXREQN , NXDUM1 , NXDUM2 , NXDUM3 , NXDUM4 ,
READ (IRSTR)  LININC , LINTOT , NOECRO ,
READ (IRSTR)  DAMPF (3) ,
READ (IRSTR)  IEQNC , IGNMS , IOMEG , IDENO , IDYNMD , IISTR2 ,
1 IPTAR , IPTBR , IPTVED , IDAM , IOMEGD ,
READ (IRSTR)  IMFOR0 , IMDISO , IMVELO , IMFOR1 , IMDIS1 , IMVEL1 ,
READ (IRSTR)  IC , IEL , IDF , JAW , IPATH , IASSEM ,
1 JRULE , JCART , JEOLO9 , JEOLO10 , JEOLO11 , JEOLO12 ,
READ (IRSTR)  OMEGCH , IHARM , KHARM , OMEGB , IBASE , KBASE ,
1 ICNFOR , JCMFOR , ICARES , ICHFHN , ICBHFN , ICBEXC ,
2 ICCMAT ,
READ (IRSTR)  FACTOR , JCNLGF (20) ,
READ (IRSTR)  MAXCRD , MAXNFR , MAXNOV , MAXSTR , MAXCHR , MAXFIS ,
1 MAXLJ , MAXINT , MAXWRI , MAXLV , NSUMAX , MAXCM ,
2 MAXBSP , MAXGMR , MAXTEM , MAXELM , MAXLW , MAXDMT ,
3 MAXFOR , MAXBET , MAXVAR , MAXSET , MAXEAN , MAXORD ,
4 MAX025 , MAX026 , MAX027 , MAX028 , MAX029 , MAX030 ,
READ (IRSTR)  JLOUB , JINFER , JEXTRA , JWEIGHT , JSUBRE , JISTRN ,
1 JCIITER , JHECLS , JGGRAM , JOUB03 , JOUB04 , JOUB05 ,
READ (IRSTR)  JTYPE , NELEM , NNODE , NBC , NIE , NMX ,
1 NTRAN , NTRAC , NFD , NBAND , NEXT , NSUB ,
2 NPRINT , NPOST , NSBC , NDUP , NSIZE , NSBC2 ,
3 NSHIFT , NSBFGS , NSGAM , NSPRI , NNASS , NDASH ,
4 NDYNMD , NSENC , NSUPER , NHARM , NBASE , NINC ,
5 NITER , NPPTS , NFPTS , NPULSE , NPDPTS , NHARD ,
6 NSUMCH , NPAR38 , NMONIT , NPAR40 , NPAR41 , NPAR42 ,
7 NPAR43 , NPAR44 , NPAR45 , NPAR46 , NPAR47 , NPAR48 ,
READ (IRSTR)  NPVCNO , NPP008 , NPP009 , NPP010 , NPP011 , NPP012 ,
1 NUMANS , ISTDEV , IPDATA , JTYPE , ISKIP , IREDEF ,
2 IDINCO , IREACO , IRESDO , IDGRP , ISTIF0 , IMASSO ,
3 IPP013 , IOMEGO , IOMEGP , IOMEGK , IETAK , ITZetak ,
4 IXCOOR , LXCAR , LXFORC , KXFORC , LXDIST , KXDIST ,
| READ (IRSTR') | JPEROD( 2 ), IPDISP, IPPORC, INDISP, INFORC |
| READ (IRSTR') | IPOINT, JPOINT, KPOINT, NDATA, PRNTBF( 6 ) |
| READ (IRSTR') | IELPHI, IELTIM, IEPSMO, ISIGNO, IHFN, IHFC |
| READ (IRSTR') | IFBP, ISPP, ISFF, ISQQ, ICQQ, ITNM |
| READ (IRSTR') | IPSF, IPSD |
| READ (IRSTR') | NRFFTS, NRFSDS, NRFLIN, NRFINI, NRREAF, NRSEL, |
| READ (IRSTR') | NFRCMP, NFRODF, NFSTDS, LRFSID, JRFTYP, JCOUNT |
| READ (IRSTR') | JCFLAG, JRFSED, JRFDOM, NVKEPT, MORE |
| READ (IRSTR') | IRFFTS, IRFINI, IRMNR, IRPSDR, IRPSL, IRPSRT, |
| READ (IRSTR') | IRFNSV, IRFSDV, IRFIVD, IRFVEC, IRFVAL, IRFNM, |
| READ (IRSTR') | IFCOR, IRFID, IRFWRK |
| READ (IRSTR') | IPULSE, KPULSE, IPTIM, IPDFOR |
| READ (IRSTR') | MANVAR( 7 ), JPR, JCOM, NCOMP |
| READ (IRSTR') | IELFRM, ITYP, INEL, ICHAR, IPRES, ISTRS |
| READ (IRSTR') | ISTRN, ICOP, IPRINT, IPOST, IDIST, IALEN |
| READ (IRSTR') | IBPRES, IBNORM, IMONIT, IST116, IST117, IST118 |
| READ (IRSTR') | INOD, ITEM, INLV, IPOSU, ITEMDF, IDUP |
| READ (IRSTR') | IEC, ITI, ITR, ITRAN, ITRAC, IEXT |
| READ (IRSTR') | ISEC, ISCBR |
| READ (IRSTR') | IDINC, IDIOT, IFORCE, IRESID, IWIND, ISIGNO |
| READ (IRSTR') | IEPSNO, IEPSRN, ICSTRN, ISTSTR, ISTRS |
| READ (IRSTR') | IISPC, IICSTR, IITSTR, IPSNO, ICSTNO, ICSTO |
| READ (IRSTR') | IITSTR, IISSNO, IISNSO, IISPNO, IISCNO, IITSNO, IITM |
| READ (IRSTR') | IDMNO, IEQCS, IOMENO, IOMNO, IOMDN, IVSDN |
| READ (IRSTR') | IDYNV, IDYNA, IDSX1, IDSX2, IDSITR, ISWELL, |
| READ (IRSTR') | IBQCS, IBQDF, IBQX3, ITYELD, IST441, IST442, |
| READ (IRSTR') | IST443, IST444, IST445, IST446, IST447, IST448 |
| READ (IRSTR') | IRC, IRFA, IES, LAB, IBQ, ISRL |
| READ (IRSTR') | IBF, IBK, ILAST, ILRB, IDINCP, IFORIN |
| READ (IRSTR') | IOP, IDAM, IMASMT, IDIAG, IUPTRI, IDCTP |
| READ (IRSTR') | IMASDT, IMASUP, IST521, IST522, IST523, IST524 |
| READ (IRSTR') | IELV, IIC, ISIG, IEPS, INWIND, ISIGN |
| READ (IRSTR') | IDNOD, IEIM, ICH, IPP, IXRL, IXRL |
| READ (IRSTR') | IXP, IIXX, KPSRTN, KCSTRN, KISTRN, KisRTS |
| READ (IRSTR') | KISTRN, KIPSTR, KICSTR, KITSTR, KPSNO, KCSTNO |
| READ (IRSTR') | KITSTR, KISNNO, KIPSNO, KICSNO, KITSNO, |
| READ (IRSTR') | IMASNO, IMNOD, IEQFST, IEQPSI, KEQPS, KEPSI |
| READ (IRSTR') | KDMAT, KDMNO, KDSNO, KIDST, XM, IXC |
| READ (IRSTR') | IVELM, IAELM, IMASIL, KICYELD, IST647, IST648 |
| READ (IRSTR') | IST649, IST650, IST651, IST652, IST653, IST654 |
| READ (IRSTR') | ICON, IBCRR, ITRACR, ITRANR, IBETA, IDET |
| READ (IRSTR') | IGEPS, KGEPS, KSG, KI, SIGK, KIDST, |
| READ (IRSTR') | IGEPO, IGEPO, IGSNO, IGSNO, IGIDO, |
DO 600 JREC = 1, NRECDS
   IBEG = I+(JREC-1)*NWORDS
   IEND = JREC*NWORDS
   READ (IRSTRT) (IWORK(JJ), JJ=IBEG, IEND )
600 CONTINUE

IBEG = I+NRECDS*NWORDS
IEND = ILAST
IF (IBEG .NE. IEND)
   1 READ (IRSTRT) (IWORK(JJ), JJ=IBEG, IEND )
1

CALL QUIT ('REST','ART ','FILE',' REA','DER ',' ',1)
CALL TIMOUT ('REST','ART ','FILE',' WAS',' REA','D ')
JINC = JINC+1
C
RETURN
END

3.4 Post-Processing Data File

The MHOST program produces an industry-standard formatted post-processing data file when the user requests it. The format is almost compatible with the MARC general purpose finite element program, Versions J.3 and K.1. The difference is that the stress variables (referred to as element variables in most of the finite element programs) are produced at nodal points and written on the file in a tightly packed manner.

CONTENTS OF HOST POST TAPE - Version 4.2

The following describes the contents of the ten blocks which will be found on the Post Tape. This information is the same for either the binary or the formatted Post Tape.

<table>
<thead>
<tr>
<th>BLOCK NUMBER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block I</td>
<td>TITLE</td>
</tr>
<tr>
<td>NR;80A1;(TITLE(J),J=1,70)</td>
<td>TITLE(J) is Jth character</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Block II</th>
<th>CONTROL INFORMATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. NR;613;INUM</td>
<td>Number of variables per element</td>
</tr>
<tr>
<td>2. LNUM</td>
<td>Number of nodes in mesh</td>
</tr>
<tr>
<td>3. MNUM</td>
<td>Number of elements in mesh</td>
</tr>
<tr>
<td>4. NDEG</td>
<td>Number of degrees of freedom per node</td>
</tr>
<tr>
<td>5. NSTRES</td>
<td>Dummy</td>
</tr>
<tr>
<td>6. INOD</td>
<td>Number of nodal variables - See Note below.</td>
</tr>
<tr>
<td>7. IPSTCC</td>
<td>Connectivity and coordinate flag</td>
</tr>
<tr>
<td></td>
<td>(1- given)</td>
</tr>
</tbody>
</table>
8. IPSTYP
   Type of tape (1 = Formatted)
9. NCRD
   Number of coordinates per node
10. NNODMX
    Maximum number of nodes per element
11. IANTYP
    Analysis Type Flag:
        2 = Displacement - With Reaction Forces
        4 = Dynamics - With Reaction Forces
        15 = Eigenvector (Modal)
        16 = Eigenvector (Buckle)
12. ICOMPL
    Set to 0 if real analysis
    Set to 1 if complex analysis
13. NBCTRA
    Number of nodes with transformations
14. POSTRV
    Post tape revision number, 1 for this release (Version 4.2)

IDM4
IDM5
IDM6
IDM7

Not used; reserved for future expansion

Note:

INOD = \{NDEG\}*JNODE

If IANTYP = 2 on 2
   4 4
   15 1
   16 1

IANTYP = 15 only appears during subincrements
IANTYP = 16 only appears during subincrements
If IANTYP = 8 and ICOMPL = 1 JNODE = 2
If IANTYP = 9 and ICOMPL = 1 JNODE = 4

Block II

CODE NUMBER ASSOCIATED WITH ELEMENT VARIABLES

INUM Records; J=1, INUM
### Block IV

**Element Variable Code**

**CONNECTIVITY LIST**

If IPSTCC is zero this block will be omitted.

<table>
<thead>
<tr>
<th>MNUM Records</th>
<th>J = 1, MNUM</th>
</tr>
</thead>
</table>

**Element Type**

**Number of Nodes in this Element**

<table>
<thead>
<tr>
<th>LM(2)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>1st Node of Element J</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM(3)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Last Node of Element J</th>
</tr>
</thead>
<tbody>
<tr>
<td>LM(NNODMAX + 2)</td>
</tr>
</tbody>
</table>

### Block V

**COORDINATE LIST**

If IPSTCC is zero this block will be omitted.

<table>
<thead>
<tr>
<th>LNUM Records</th>
<th>J = 1, LNUM</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>1st Coordinate of Jth Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>NR; 6E13.6; SUM(1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Last Coordinate of Jth Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM(2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUM(NCRD)</th>
</tr>
</thead>
</table>

### Block VI

**TRANSFORMATION LIST**

If NBCTRA is zero this block will be omitted.

- Binary 1 record
- Formatted (NBCTRA-1)/6 + 1 records

**List of nodes which have transformations applied**

<table>
<thead>
<tr>
<th>IR; 6I13; LM(I)</th>
</tr>
</thead>
</table>

### Block VII

**TRANSFORMATION - DIRECTION COSINES**

If NBCTRA is zero this block will be omitted.
1 record per each node listed in Block VI if binary tape

NR;6E13.6; D(1,1)
D(2,1)
D(3,1)
D(1,2)
D(2,2)
D(3,2)
D(1,3)
D(2,3)
D(3,3)

2 records per node if formatted tape

Transformations are for local to global

Blocks VIII, IX and X are repeated for each increment.

Block VIII

INCREMENT, TIME AND FREQUENCY

NR;6E13.6 X1(1)
X1(2)
X1(3)
X1(4)
X1(5)
X1(6)

Transient Time
Increment number is a real formed as I + J/100
I is the static increment number
J is either the harmonic subincrement or the eigenvector number

Frequency
Flag to read new blocks II, III, IV, V, VI, VII. Set to 1 to read these blocks again.

IANTYP analysis type flag.
Not used; reserved for future expansion.

Block IX

VALUES OF ELEMENT VARIABLES

If INUM is zero, this block is omitted.

If IANTYP=15 or IANTYP=16, this block is omitted.

MNUM*NSTRES Records

NR;6E13.6;

VALUE((CI,J), I=1,INUM), J=1,INODE)

VALUE is the name of an array storing all the element variables at nodes.
Block X

VALUE OF NODAL VARIABLES

If INOD is zero, this block is omitted.

INUM Records

NR;6E136; SUM(1)
SUM(NDEG)
SUM(NDEG+1)
SUM(2*NDEG)
SUM(INOD)

Nodal displacements, velocities, accelerations and reactions

During subincrements:

The first NDEG quantities are:

IANTYP = 15
Nodal Components of Dynamic Mode

IANTYP = 16
Nodal Components of Buckle Eigenvector

NOTES:

NR Indicates the beginning of a new record for the binary post tape.

NR;Format Indicates the beginning of a new set of information which is to be read with the following format:

Appendix 1 provides a sample program to dump the post tape.

The table below provides codes for selecting strains and stresses for plotting:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-6</td>
<td>Components of total strain.</td>
</tr>
<tr>
<td>7</td>
<td>Equivalent plastic strain.</td>
</tr>
<tr>
<td>8</td>
<td>Equivalent creep strain.</td>
</tr>
<tr>
<td>9</td>
<td>Total temperature.</td>
</tr>
<tr>
<td>11-16</td>
<td>Component of stress.</td>
</tr>
<tr>
<td>17</td>
<td>Equivalent Mises tensile stress.</td>
</tr>
</tbody>
</table>
Mean normal stress (tensile positive)  
(for Mohr-Coulomb).

User definable quantity to write on post tape.

User definable quantity to write on post tape.

Physical components of the total plastic strain.

Total equivalent plastic strain.

Second state variable.

Physical components of total creep strain.

Total equivalent creep strain.

If several layers (shell or beam) are to be plotted, the code number should be as follows: code for variable as above + 100 x layer number.

An example of the post tape file for a 20 four node plane stress element used to model a cantilever beam is:

BEAM PROBLEM

16 33 20 2 2 4
1 1 2 4 2 0
0 1 0 0 0 0

3 4 1 2 5 4
3 4 2 3 6 5
3 4 4 5 8 7
3 4 5 6 9 8
3 4 7 8 11 10
3 4 8 9 12 11
3 4 10 11 14 13
3 4 11 12 15 14
3 4 13 14 17 16
<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>4</td>
<td>14</td>
<td>15</td>
<td>18</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>16</td>
<td>17</td>
<td>20</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>17</td>
<td>18</td>
<td>21</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>19</td>
<td>20</td>
<td>23</td>
<td>22</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>20</td>
<td>21</td>
<td>24</td>
<td>23</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>22</td>
<td>23</td>
<td>26</td>
<td>25</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>23</td>
<td>24</td>
<td>27</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>25</td>
<td>26</td>
<td>29</td>
<td>28</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>26</td>
<td>27</td>
<td>30</td>
<td>29</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>28</td>
<td>29</td>
<td>32</td>
<td>31</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>29</td>
<td>30</td>
<td>33</td>
<td>32</td>
</tr>
</tbody>
</table>

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.0000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.0000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14.000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14.000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18.000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18.000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20.000</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20.000</td>
<td>-0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>x1</td>
<td>x2</td>
<td>y1</td>
<td>y2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6003.0</td>
<td>5999.0</td>
<td>-7.9094</td>
<td>-0.16610E-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>-0.43187E-08</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.96139E-04</td>
<td>-0.16456E-02</td>
<td>0.49706E-08</td>
<td>0.54500E-09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>6003.0</td>
<td>-5999.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.61575E-02</td>
<td>-0.60014E-02</td>
<td>0.18076E-02</td>
<td>-0.42386E-08</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6003.0</td>
<td>5999.0</td>
<td>-7.9240</td>
<td>-0.29525E-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>-0.76764E-08</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.16348E-03</td>
<td>-0.30093E-02</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.61575E-02</td>
<td>-0.60014E-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6002.9</td>
<td>5998.9</td>
<td>-7.9588</td>
<td>-0.34241E-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>-0.89027E-08</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.36859E-03</td>
<td>-0.34071E-02</td>
<td>0.10247E-07</td>
<td>-0.39346E-09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>6002.9</td>
<td>-5998.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.61575E-02</td>
<td>-0.60014E-02</td>
<td>0.18076E-02</td>
<td>-0.79171E-08</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6002.8</td>
<td>5998.8</td>
<td>-7.9876</td>
<td>-0.56537E-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>-0.88140E-08</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.36859E-03</td>
<td>-0.34071E-02</td>
<td>0.16856E-07</td>
<td>-0.11461E-08</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>6002.8</td>
<td>-5998.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.61574E-02</td>
<td>-0.60013E-02</td>
<td>0.18076E-02</td>
<td>-0.88140E-08</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6002.8</td>
<td>5998.8</td>
<td>-7.9588</td>
<td>-0.56537E-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>-0.14700E-07</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.17802E-03</td>
<td>-0.56015E-02</td>
<td>0.16856E-07</td>
<td>-0.11461E-08</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>6002.8</td>
<td>-5998.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.61572E-02</td>
<td>-0.60012E-02</td>
<td>0.18076E-02</td>
<td>-0.14428E-07</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6002.8</td>
<td>5998.8</td>
<td>-7.9588</td>
<td>-0.37059E-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>-0.96353E-08</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.36859E-03</td>
<td>-0.34071E-02</td>
<td>0.10247E-07</td>
<td>-0.39346E-09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>6002.8</td>
<td>-5998.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.18076E-02</td>
<td>-0.86529E-08</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.18076E-02</td>
<td>-0.44419E-08</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.3227E-03</td>
<td>0.19007E-02</td>
<td>0.57894E-08</td>
<td>-0.13521E-10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>6002.8</td>
<td>-5998.8</td>
<td>7.9891</td>
<td>0.2093E-02</td>
</tr>
<tr>
<td>0.61573E-02</td>
<td>-0.60012E-02</td>
<td>0.18076E-02</td>
<td>0.54419E-08</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.9030</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>6002.8</td>
<td>5998.9</td>
<td>-7.9647</td>
<td>0.67373E-02</td>
<td>0.61573E-02</td>
<td>0.60012E-02</td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>0.17517E-07</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>-0.16063E-02</td>
<td>0.67225E-02</td>
<td>0.20266E-07</td>
<td>-0.41724E-09</td>
<td>-0.13365E-08</td>
<td>0.17479E-07</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>6002.9</td>
<td>5998.9</td>
<td>-7.9466</td>
<td>0.30772E-02</td>
<td>0.70974E-02</td>
<td>0.46497E-02</td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>0.80008E-08</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>6002.9</td>
<td>5998.9</td>
<td>-7.9466</td>
<td>0.30772E-02</td>
<td>0.61573E-02</td>
<td>0.60013E-02</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>6002.9</td>
<td>5998.9</td>
<td>-7.9466</td>
<td>0.30772E-02</td>
<td>0.61573E-02</td>
<td>0.60013E-02</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>6002.8</td>
<td>5998.9</td>
<td>-7.9466</td>
<td>0.30772E-02</td>
<td>0.61573E-02</td>
<td>0.60013E-02</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>-0.10450E-07</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>-0.18358E-02</td>
<td>-0.48038E-02</td>
<td>0.14573E-07</td>
<td>-0.46498E-09</td>
<td>-0.15311E-08</td>
<td>-0.12490E-07</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.61573E-02</td>
<td>-0.60012E-02</td>
<td>0.18076E-02</td>
<td>0.94669E-08</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>6002.8</td>
<td>5998.9</td>
<td>-7.9466</td>
<td>0.40192E-02</td>
<td>0.46497E-02</td>
<td>0.36411E-02</td>
</tr>
<tr>
<td>-0.18076E-02</td>
<td>-0.10450E-07</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>-0.18358E-02</td>
<td>-0.48038E-02</td>
<td>0.14573E-07</td>
<td>-0.46498E-09</td>
<td>-0.15311E-08</td>
<td>-0.12490E-07</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

**SYSTEMS' MANUAL**

Page : 157

**MHOST Version 4.2**
The write operation takes place in SUBROUTINE POSTOU:

C=SUBROUTINE=POSTOU CALLED BY SECOND LEVEL DRIVER ROUTINES
SUBROUTINE POSTOU
   1 ( IWORK, RWORK, ISIZE )

C
C******************************************************************************
C
C IMPLICIT REAL*8 ( A-H, O-Z )
REAL*4 RWORK

C
C******************************************************************************

SYSTEMS' MANUAL

MHOST Version 4.2

Page : 158
DIMENSION IWORK(ISIZE), RWORK(ISIZE)
DIMENSION JPLOT(80)
DIMENSION KPLOT(80)
DIMENSION BUFFER(50), VALUE(50)
DIMENSION i0

C
C **********************************************************************
C
C _ BLOCKS IN ALPHABETICAL ORDER
C
C **********************************************************************

COMMON / ALGEM / IREAD, ILSRNTR, ICONSL, IPOSTF, ISCRAF,
1 IPLOTO, ISTRRT, ICREAD, IPVARS, IPSETS, IPILEX,
2 PI, LINE, LINE2
COMMON / PARAM / ICTYPE, NELEM, NNODE, NBC, NITE, NMAX,
1 NIRAN, NITAC, NFD, NBAND, NEXT, NSUB,
2 NPRINT, NPOST, NSBC, NDUP, NSIZE, NSECT,
3 NSHIFT, NSBFGS, NGRPS, NSPRI, NMASS, NDASH,
4 NDYNMD, NSBNC, NSUPER, NBASE, NTNC,
5 NITER, NPSPTS, NFDPITS, NPKLSE, NPARD,
6 NSUMCH, NDMIN, NMONIT, NPAR40, NPAR41, NPAR42,
7 NPAR43, NPAR44, NPAR45, NPAR46, NPAR47, NPAR48
COMMON / CTITLE / TITLE(20), IDAT(5), IDATE2, ICLOCK,
1 IPCRAY
COMMON / EIGEN / IEQVC, I6_S, I(REG, IMDENO, IDYNMD,
1 IPTAR, IPTBR, IPTRED, IMDAM, I_EG3
COMMON / ELEMEN / IC, IEL, IDF, JLAW, IPATH, IASSEM,
1 JRULE, JCART, JEL009, JEL010, JEL011, JEL012
COMMEN / ELTYP / NCRD, NELNFR, NENDD, NELSTR, NELCHL, NELPR,
1 NELINT, NELLV, NELLAY, NDI, NSHEAR, NELCMP
COMMON / ERRORS / IERR
COMMON / MACHIN / IDP
COMMON / MAXIMA / MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
1 MAXLAY, MAXINT, MAXWRK, MAXNLV, NMAX, MAXCM,
2 MAXBSP, MAXCHR, MAXTEM, MAXELM, MAXLW, MAXINT,
3 MAXFRN, MAXBET, MAXVAR, MAXSET, MAXEAN, MAXORD,
4 MAX025, MAX026, MAX027, MAX028, MAX029, MAX030
COMMON / CONTRO / JEND, JITR, JTEMP, JPRINT, JP, JSUB,
1 JINC, JREST, JSAVE, JREDIM, JAUTO, JPOST,
2 JBACK, JOPTIM, JCREEP, JDIST, JCONST, JDYN,
3 NONISO, ITERM, ITRIG, IDYNM, JREPO, JANG,
4 JTERM, JFORCE, JUTEMP, JUOEF, JDISTS, JUHOOK,
5 JDERIV, JUBOON, IDSTOP, INTSTR, JPLAST, JBAND,
IPOSTF (19) I FORMATTED OUTPUT FILE FOR POST-PROCESSING

VERSION 2.0 GENERATES MARC COMPATIBLE POST TAPE CONTAINING
ONLY MESH DATA AND DISPLACEMENT + REACTION AT NODES.
THIS CAN BE PROCESSED BY INVOKING 'POSTDATA' COMMAND WITH 'MARC'
OPTION IN 'MENTAT' INTERACTIVE SESSION.
FULLY MENTAT COMPATIBLE POST PROCESSING FILE WILL EVENTUALLY
WRITTEN ON IPIOTF FILE AND THEN IPOSTF WILL BE USED FOR THE
TOPOLOGICAL MESH DEFINITION OUTPUT TO BE PROCESSED BY 'READ'
OPTION IN MENTAT

MINCMP = MAXCMP
IF( MAXCMP .EQ. 8 ) MINCMP = 5

IZERO = 0
MAXVAL = 50
MAXQNT = 4
LOCATE( 1 ) = ISIGNO
LOCATE( 2 ) = IEPSNO
LOCATE( 3 ) = IPSTNO
LOCATE( 4 ) = ICSTNO

--- HEADER RECORDS ARE WRITTEN IN THE POST TAPE ONLY WHEN THIS
--- ROUTINE IS ENTERED AT THE ZERO-TH INCREMENT. OTHERWISE THE NODAL
--- VALUES ARE WRITTEN IN THE POST TAPE.

IF( JINC .GT. 0 ) GO TO 9000

--- FIRST RECORD IN THE POST TAPE : TITLE -----------------------------

WRITE(IPOSTF,1000) TITLE
1000 FORMAT(20A4)

--- SECOND RECORD: CONTROL INFORMATION -------------------------------

SET VARIABLES ACCORDING TO THE MARC DOCUMENT

INUM = ( MINCMP + 1 ) * 4
IF( JEIGEN .EQ. 1 ) INUM = 0
LNUM = NNODE
MNUM = NELEM
NDEG = MAXNFR
NSTRES = 2
JNOD = MAXNFR * 2
IPSTCC = 1
IPSTYP = 1
NCRD = MAXCRD
IF( NCRD .EQ. 7 ) NCRD = 3
NNOMNX = NEINOD
IANTYP = 2
IF( JEIGEN .EQ. 1 .AND. IDYNM .EQ. 1 ) IANTYP = 15
IF( JEIGEN .EQ. 1 .AND. IDYNM .EQ. 0 ) IANIXP = ICOMPL = 0
IPOSTR = 1
IE_44 = 0
IE_5 = 0
IE_6 = 0
IEM7 = 0
WRITE(IPOSTF,1010) INUM ,I2/t_ ,MNUM ,NDEG ,NSTRES,JNOD ,
1 IPSTCC,IPSTYP,NCRD ,NNOMNX,IANTYP,ICOMPL,
2 NBCTRA,IPOSTR,IM4 ,IM5 ,IM6 ,IM7
1010 FORMAT(6113)
--- CODE NUMBER ASSOCIATED WITH ELEMENT VARIABLES ------------

IF( JEIGEN .EQ. 1 ) GO TO 3000

JPLT( 1 ) = 17
JPLT( 2 ) = 11
JPLT( 3 ) = 12
JPLT( 4 ) = 13
IF(MINCMP .GE. 4 ) JPLT( 5 ) = 14
IF(MINCMP .GE. 5 ) JPLT( 6 ) = 15
IF(MINCMP .GE. 6 ) JPLT( 7 ) = 16
INDX =
JPLT( INDX + 1 ) = 39
JPLT( INDX + 2 ) = 1
JPLT( INDX + 3 ) = 2
JPLT( INDX + 4 ) = 3
IF(MINCMP .GE. 4 ) JPLT( INDX + 5 ) = 4
IF(MINCMP .GE. 5 ) JPLT( INDX + 6 ) = 5
IF(MINCMP .GE. 6 ) JPLT( INDX + 7 ) = 6
INDX = INDX + MINCMP + 1
JPLT( INDX + 1 ) = 7
JPLT( INDX + 2 ) = 21
JPLT( INDX + 3 ) = 22
JPLT( INDX + 4 ) = 23
IF(MINCMP .GE. 4 ) JPLT( INDX + 5 ) = 24
IF(MINCMP .GE. 5 ) JPLT( INDX + 6 ) = 25
IF(MINCMP .GE. 6 ) JPLOT( INDX + 7 )= 26 
       INDX = INDX + MINCMP + 1
JPLOT( INDX + 1 )= 8 
JPLOT( INDX + 2 )= 31 
JPLOT( INDX + 3 )= 32 
JPLOT( INDX + 4 )= 33 
IF(MINCMP .GE. 4 ) JPLOT( INDX + 5 )= 34 
IF(MINCMP .GE. 5 ) JPLOT( INDX + 6 )= 35 
IF(MINCMP .GE. 6 ) JPLOT( INDX + 7 )= 36 
       INDX = INDX + MINCMP + 1
C
WRITE(IPOSTF,1030) (JPLOT(K) ,K=I, INDX)
1030 FORMAT(113)
C
3000 CONTINUE
C
--- DATA BLOCK IV : ELEMENT CONNECTIVITY DATA -----------------------
C
DO 6000 IEL = 1 , NELEM
C
       IC = IWORK( ITYP + IEL - 1 )
       NELNOD = 0
C
       IF( IC .EQ. 0 ) GO TO 6007
C
       CALL ELVULV
       1 (IWORK ,IC ,IERR )
C
       IADRES = INEL + MAXNOD * ( IEL - 1 )
       JADRES = IADRES + NELNOD - 1
C
... SPECIAL TRICK ASSOCIATED WITH THE FRONTAL SOLUTION
C
DO 6005 INTPNT = 1 , NELNOD
       JNTPNT = INTPNT + IADRES - 1
6005   KPLOT(INTPNT)= IABS(IWORK(JNTPNT))
C
6007   CONTINUE
C
... REPLACE THE ASSUMED STRAIN ELEMENTS WITH THE MATCHING MARC/MENTAT
C
     ELEMENT NUMBERS (A LITTLE INNOCENT LIE THAT WON'T HURT ANYBODY)
C
       IC1 = IC
       IF ( IC .EQ. 151 ) IC1 = 3
       IF ( IC .EQ. 152 ) IC1 = 11
IF ( IC .EQ. 153 ) IC1 = 10
IF ( IC .EQ. 154 ) IC1 = 7

WRITE(IPOSTF,1010) IC1 ,NELNOD,(KFLG(K),K=1,NELNOD)

6000 CONTINUE

C --- DATA BLOCK V : NODAL COORDINATE DATA -------------------------------------
C
DO 6010 INODE = 1 , NNODE

IADRES = IINOD + MAXCRD * ( INODE - 1 ) * IDP
CALL POSTPR
1 (RWORK(IADRES),NCRD,IPOSTF)

6010 CONTINUE

9000 CONTINUE

C --- INCREMENTAL HEADER RECORD -----------------------------------------------
C
NSUBIN = 1
IF ( JEIGEN .EQ. 1 ) NSUBIN = NDYNMD

DO 8000 ISUBIN = 1 , NSUBIN

RINCRE = JINC
FREQUN = 0
RFLAGS = 0
RANTYP = 0
RDUMMY = 0

IF ( JEIGEN .EQ. 0 ) GO TO 5100

RINCRE = RINCRE + 0.01DO * ISUBIN
INDX : ICMEG + ( ISUBIN - 1 ) * IDP
CALL COPY( RWORK(INDX) , FREQUN , 1 )
IF ( IDYNM .EQ. 1 ) RANTYP = 15
IF ( IDYNM .EQ. 0 ) RANTYP = 16

5100 CONTINUE

WRITE(IPOSTF,1020) RTIN, RINCRE, FREQUN, RFLAGS, RANTYP, RDUMMY

1020 FORMAT(6GI3.6)

IF ( JEIGEN .EQ. 1 ) GO TO 7500
C

    INUM = ( MINCMP + 1 ) * 4
    INITIAL = 0
    DO 7000 INODE = 1 , NNODE
    IREC = 0

C

    CALL NUL( VALUE , MAXVAL )
C

*** VALUES CALCULATED AT THE LAYER LEVEL ****************************
C

    INILAY = 1
    IF( NELAY .NE. 1 ) INILAY = 3
C

*** FIRST DATA OF THE SERIES IS TEMPERATURE ****************************
C

+ I = ITEM + (( INODE - 1 )*MAXLAY + INILAY - 1 )*IDP
+ IREC = 1
C

CALL COPY( RWORK(LOCTEM) , VALUE(IREC) , 1 )
C

*** TENSORIAL QUANTITIES AND THEIR INVARIANTS ****************************
C

(I)  TOTAL STRESS  
(II)  TOTAL STRAIN  
(III) PLASTIC STRAIN  
(IV)  CREEP STRAIN

DO 5200 INTQNT = 1 , MAXQNT
C

    INTLOC = LOCATE(INTQNT) + (( INODE - 1 )*MAXLAY + INILAY - 1 )
    * IDP * MAXCMP
C

CALL COPY( RWORK(INTLOC) , BUFFER , MINCMP )
IF( INTQNT .EQ. 1 ) CALL EQVS
    1 ( EQVAL , BUFFER , MINCMP , NDI , NSHEAR , JLAW , NONISO ,
    2 INODE )
IF( INTQNT .GT. 1 ) CALL EQVC
    1 ( EQVAL , BUFFER , MINCMP , NDI , NSHEAR , JLAW )
C

    IREC = IREC + 1
CALL COPY( EQVAL , VALUE(IREC) , 1 )
    IREC = IREC + 1
CALL COPY( BUFFER , VALUE(IREC) , MINCMP )
    IREC = IREC + MINCMP - 1

SYSTEMS' MANUAL  MHOST Version 4.2
Page : 165
5200 CONTINUE
C
CALL POSTEN( VALUE, INUM, IPOSTF, INODE, NNODE, INITAL )
C
7000 CONTINUE
C
7500 CONTINUE
C
--- TOTAL DISPLACEMENT AT THE CURRENT INCREMENT AND REACTION FORCE ---
C
IF(NTRAN .EQ. 0) GO TO 9550
C
CALL TRANS2(NTRAN,RWORK(ITRANR),IWORK(ITRAN),RWORK(IDTOT),
NFD,-1.D0,NELNFR,JLAW)
1
CALL TRANS2(NTRAN,RWORK(ITRANR),IWORK(ITRAN),RWORK(IDINC),
NFD,-1.D0,NELNFR,JLAW)
9550 CONTINUE
C
IDVEC = IDTOT
IF(JEIGEN .EQ. 1) IDVEC = IEGNVC + (ISUBIN - 1) * NFD * IDP
C
DO 5000 INODE = 1, NNODE
C ...
... TOTAL DISPLACEMENTS OR EIGENVECTOR
C
JNTLOC = INLV + (INODE - 1) * MAXNFR
KNTLOC = IDVEC + (IWORK(JNTLOC) - 1) * IDP
CALL COPY(RWORK(KNTLOC), VALUE, MAXNFR)
NENTRY = MAXNFR
C ...
... NODAL REACTION VECTOR
C
INDX = NELNFR + 1
LNTLOC = IRESID + (IWORK(JNTLOC) - 1) * IDP
CALL COPY(RWORK(LNTLOC), VALUE(INDX), MAXNFR)
NENTRY = NENTRY + MAXNFR
C
CALL POSTPR
1 (VALUE, NENTRY, IPOSTF)
C
5000 CONTINUE
C
IF(NTRAN .EQ. 0) GO TO 9530
C
CALL TRANS2(NTRAN,RWORK(ITRANR),IWORK(ITRAN),RWORK(IDTOT),

--- TOTAL DISPLACEMENT AT THE CURRENT INCREMENT AND REACTION FORCE ---

IF(NTRAN .EQ. 0) GO TO 9550

CALL TRANS2(NTRAN,RWORK(ITRANR),IWORK(ITRAN),RWORK(IDTOT),
NFD,-1.D0,NELNFR,JLAW)
1
CALL TRANS2(NTRAN,RWORK(ITRANR),IWORK(ITRAN),RWORK(IDINC),
NFD,-1.D0,NELNFR,JLAW)
9550 CONTINUE

IDVEC = IDTOT
IF(JEIGEN .EQ. 1) IDVEC = IEGNVC + (ISUBIN - 1) * NFD * IDP

DO 5000 INODE = 1, NNODE

... TOTAL DISPLACEMENTS OR EIGENVECTOR

JNTLOC = INLV + (INODE - 1) * MAXNFR
KNTLOC = IDVEC + (IWORK(JNTLOC) - 1) * IDP
CALL COPY(RWORK(KNTLOC), VALUE, MAXNFR)
NENTRY = MAXNFR

... NODAL REACTION VECTOR

INDX = NELNFR + 1
LNTLOC = IRESID + (IWORK(JNTLOC) - 1) * IDP
CALL COPY(RWORK(LNTLOC), VALUE(INDX), MAXNFR)
NENTRY = NENTRY + MAXNFR

CALL POSTPR
1 (VALUE, NENTRY, IPOSTF)

5000 CONTINUE

IF(NTRAN .EQ. 0) GO TO 9530

CALL TRANS2(NTRAN,RWORK(ITRANR),IWORK(ITRAN),RWORK(IDTOT),
The data record packing for the element data is carried out in the following subroutine POSTEN:

C=SUBROUTINE=POSTEN POST FILE GENERATION UTILITY
SUBROUTINE POSTEN
  1 (ARRAY ,NENTRY, IPOSTF, INODE ,NNODE ,INITAL)
C
C**********************************************************************
CIMPLICIT REAL*8 ( A-H , O-Z )
REAL*4 R_DRK
C**********************************************************************
C
DIMENSION ARRAY (NENTRY)
COMMON / POSTPN / IPOINT,JPOINT,KPOINT,NDATA ,PRNTBF( 6 )
IF( INITAL .NE. 0 ) GO TO 4950
  IPOINT = 1
  JPOINT = 1
  KPOINT = 1
  NDATA = 6
  INITAL = 1

4950 CONTINUE

IPOINT : POINTER FOR THE LAST ENTRY OF THE PRINT BUFFER
JPOINT : POINTER FOR THE CURRENT ARRAY ENTRY TO BE PRINTED
C
JPOINT = 1

C
KPOINT = JPOINT + 5
IF( KPOINT .GT. NENTRY ) GO TO 6000

CALL COPY ( ARRAY(JPOINT) , PRNTBF(IPOINT) , NDATA )
CALL POSTPR( PRNTBF , 6 , IPOSTF )

JPOINT = JPOINT + NDATA
NDATA = 6
IPOINT = 1

GO TO 5000

C --- HALF FILL THE PRINTER BUFFER AND RETURN

6000 CONTINUE

MDATA = NENTRY - JPOINT + 1
NDATA = 6 - MDATA
IPOINT = MDATA + 1

CALL COPY ( ARRAY(JPOINT) , PRNTBF( 1 ) , MDATA )

--- IF THE LAST NODE IS ENTERED WRITE THE LAST RECORD EVEN IF IT IS INCOMPLETE

IF( INODE .LT. NNODE ) GO TO 9000
IF( MDATA .EQ. 0 ) GO TO 9000

CALL NUL ( PRNTBF(IPOINT) , NDATA )
CALL POSTPR( PRNTBF , 6 , IPOSTF )

9000 RETURN
END
Subroutines included in the MHOST code, Version 4.2, are summarized in this appendix. The routine names are sorted in alphabetical order. A brief description of each routine is given, in conjunction with the names of common blocks referenced therein. Names of routines referenced in some of the vitally important routines are given in this document. Note that almost all subprograms written for the MHOST package are self-documented and further information can be obtained directly from the source listing.

ACCLIN
Reads in initial acceleration data from the main input data reader.
Common block: MACHIN

ADAPTC
Controls the adaptive time step size adjustment for creep strain evaluation
Common block: ALGEM

ADAPTD
Common block: TIME, TOLER, CONTRO

ADAPTS
Controls the adaptive load increment size adjustment for the arc length method
Common block: ALGEM

ADD
Adds two double precision real vectors.

ADDBAN
Adds the lumped values listed in SVAL with the connectivity specified in LCON to the global array GABF in band form. Not used for profile-stored global arrays.

ADDIAG
Adds the lumped values listed in SVAL to the global array. This subroutine is the profile-storage counterpart of ADDBAN.
ADDINC
Updates total quantities by adding together the values at the beginning of the increment and the incremental values.
Common block: AUTOIN, ELTYP, CONTRO, SUBELM, MAXIMA, PARAM, START1, START2, START3, START4, START6, START8, TIME

ASMVEC
Assembles global vector from the d.o.f conversion table. See also SUBST1.

ASSEM1
Assembles the displacement stiffness matrix for quasi-static analysis. Works with the profile solver.
Common block: ALGEM, ADDVAL, AUTOIN, BSECT, CONTROL, BODYFR, LOUBIN, ELEMEN, ELTYP, ERRORS, MAXIMA, PARAM, PULSES, START1, START2, START3, START4, START5, START6, START7, START8, SUBELM, SUBSTR, TIME, TRANSF, MACHIN

ASSEM2
Assembles the global time integration operator matrix for direct time integration. Works with the profile solver.
Common block: ALGEM, ADDVAL, AUTOIN, BSECT, CONTROL, BODYFR, LOUBIN, ELEMEN, ELTYP, ERRORS, MAXIMA, PARAM, PULSES, START1, START2, START3, START4, START5, START6, START7, START8, SUBELM, SUBSTR, TIME, TRANSF, MACHIN

ASSEM3
Assembles the displacement stiffness matrix for quasi-static analysis. Works with the frontal solution.
Common block: ALGEM, ADDVAL, AUTOIN, BSECT, CONTROL, BODYFR, LOUBIN, ELEMEN, ELTYP, ERRORS, MAXIMA, PARAM, PULSES, START1, START2, START3, START4, START5, START6, START7, START8, SUBELM, SUBSTR, TIME, TRANSF, MACHIN

ASSEM4
Assembles the residual vector for quasi-static and transient dynamic analyses. The routine includes the variational recovery of nodal strains, nodal stress recovery and element loop for the internal force calculation.
Common block: ALGEM, ADDVAL, BSECT, COUNTR, CONTRO, ELEMEN, ELTYP, ERRORS,
ATTRIB

Attributes the element connectivity data by the duplicate node and tying options.

Common block: ALGEM, ERRORS, MAXIMA

BACSUB

Performs back-substitution for the nodal displacement vector as a part of the frontal solution subsystem.

BANDBR

Calculates the maximum bandwidth.

Common block: ELTYP

BANNER

Prints banner on the line printer output file.

BASAXS

Generates strain-displacement matrix for axisymmetric element based on the assumed stress method.

BASEIN

Reads in parameters defining harmonic base excitation.

Common block: MACHIN

BASPSN

Generates strain-displacement matrix for plane strain element derived from the assumed stress method.

BASPST

Generates strain-displacement matrix for plane stress element derived from the assumed stress method.

BAXSYM

Generates strain-displacement matrix for axisymmetric element by the numerical integration.

BEAMIN

Reads in beam section properties.
Common block: MACHIN

BFGSLH
Performs the BFGS update for the left hand side of the nonlinear equations.

BFGSRH
Performs the BFGS update for the right hand side of the nonlinear equations.

BFGSW
Generates the global projection vectors required for the BFGS update.

BFLOAD
Controls the generation of element body force load vector.

BMSTRS
Evaluates constitutive equations for the beam element. Note that the current version only supports a linear elastic beam element.

ENDTRM
Preintegrates the lamina constitutive matrix to obtain the constitutive resultant with respect to the bending moment - curvature terms for shell elements.

BODYIN
Reads in parameters defining body force loading.

Common block: MACHIN

BOUND1
Applies the displacement constraint by eliminating row and column of the global profile-stored stiffness equations.

BOUND2
Applies the displacement constraint to a global vector.

BOUND3
Applies the displacement constraint to a substructure. Obsolete in Version 4.2.

BOUNDN
Calculates the weighted normal vector at nodes on the boundary.
Common block: ALGEM

BOUNFR
Applies the nodal displacement constraint to the frontal solution elimination process.

BOUNIN
Reads in boundary displacement constraint data.

Common block: MACHIN

BPSTRN
Generates strain-displacement matrix for isoparametric plane strain element by numerical integration.

BPSTRS
Generates strain-displacement matrix for isoparametric plane stress element by numerical integration.

BREAD
Reads an array from a specified Fortran I/O unit stored in binary form.

BSHELL
Generates strain-displacement matrix in resultant form for isoparametric shell element by means of selective reduced integration.

BSOLID
Generates strain-displacement matrix for isoparametric solid element by numerical integration.

BTBEAM
Generates strain-displacement matrix for the linear isoparametric Timoshenko beam element by reduced integration.

BUCKLE
Drives the buckling analysis using subspace iterations for eigenvalue extraction.

Common block: AUTOIN, MAXIMA, CTITLE, DAMP, EIGEN, INCON, MACHIN, ELEMEN, ELTYP, CONTRO, SUBSTR, ERRORS, PARAM, START1, START2, START3, START4, START5, START6, START8, TIME

BULKIN
Supervises the reading operation of the bulk (model) data.

Common block: CONTR0 PARAM, SUBELM

External reference: INIT1, DATIN2, DATOU1, CHKELM, SUBDIV

WRITE

Writes an array to a specified Fortran I/O unit in binary form.

CENMAS

Assembles the stiffness matrix entries related to the centrifugal mass terms for an element.

Common block: BODYFR

External reference: NUL, GAUSSP, COPY, S2DO4N, VSH04N, TBM02N, S1DOZN, S3DO8N, D2DO4N, D3DO8N, DAXO4N, GSH04N, DMO2N, SUBT

CENT2D

Calculates the centrifugal load vector for a two-dimensional (plane stress/strain) element.

CENT3D

Calculates the centrifugal load vector for a three-dimensional solid element.

CENTAX

Calculates the centrifugal load vector for a axisymmetric element.

CENTEM

Calculates the centrifugal load vector for a beam element.

CENTSH

Calculates the centrifugal load vector for a shell element.

CHARIN

Reads in material property data.

Common block: MACHIN, ELTYP

CHAR

Identifies the value of a number given as a single character.

CHKELM

Checks to see if the element connectivity is given in the counter
clockwise manner. If not, prints warning messages and repairs the connectivity table entries. Dangerous.

Common block:  ALGEM, AUTOIN, BSECT, CONTRO, BODYFR, LOUBIN, ELEMEN, ELTYP, ERRORS, MAXIMA, PARAM, START1, START2, START3, START4, START5, START6, START7, START8, SUBSTR, TIME, TRANSF, MACHIN

External reference:  ELVULV, QNODEL, INSIDE, LINES

CNODEL
Pulls out information related to the current element. See Section 2.2 for detail.

Common block:  TMARCH, ALGEM, CONTRO, ELTYP, MAXIMA, PARAM, START1, START2, START4, START5, START6, START7, START8, START9, MACHIN

External reference:  SEARC1, SEARCH, INTERP, ROIDMT, TSHO4N, SHTRAN, TRANSP, MATINV, TBM02N, ADD, ADDSMU, NUL

CNEMAS
Assembles the consistent mass matrix for an element.

CNSTNM
Constructs a consistent transformation from nodal to modal data for linear dynamics by modal superposition.

Common block:  EIGEN, ELTYP, MACHIN, MAXIMA, MODSUP, PARAM, POWER, START1, START2, START6

COLRED
Performs column-wise reduction for back-substitution.

COMPDF
Copies the definition of element parameters in the workspace for composite shells

Common block:  START1

COMPIN
Reads material property data for the composite laminate option with shell elements.

Common block:  ALGEM

COMPRO
Computes the profile column heights and pointers for profile-storage of the global stiffness array.

Common block: ELTYP, MAXIMA, PARAM, ALGEM

CONDSE
Removes specified row and column of a square matrix.

CONNIN
Reads element connectivity data.

Common block: ELTYP

CONTIN
Adds incremental load (defined as the nodal force) to the total load vector.

Common block: MACHIN

COORIN
Reads coordinate data for nodes.

COPY
Copies a double precision real array to another double precision real array.

COPYDS
Copies a double precision real array to a single precision real array.

COPYIN
Copies an integer array to another integer array.

COPYSD
Copies a single precision real array to a double precision real array.

CORDIR
Transforms the coordinate system defining the strain component from global to local, filters out specific components and transforms back to the original global coordinate system. Used by the strain-displacement matrix routines.

COROUT
Debug writes the nodal coordinates.
CPXBKI, CPXBK2
Back substitutes for the solution of complex matrix equations stored in the band-matrix form.

CPXDIV
Divides a complex number by another complex number.

CPXEXC
Sets up complex harmonic based excitation vector. The function is fully commented.

CPXFAC
Performs the crout decomposition of complex matrix equations stored in the banded form.

Common block: ALGEM

CPXFOR
Sets up the nodal force vector for the complex harmonic analysis.

CPXMUL
Multiplies two complex numbers.

CPXREA
Assembles the complex nodal reaction force vector due to the complex base excitation.

CPXRES
Calculates the harmonic nodal force.

CRPLAW
Defines the equivalent creep strain increment. To be used as a User Subroutine. See also the MHOST Version 4.2 Users' Manual.

CRPSIN
Determines quantities related to creep strain effects.

CUTHIL
Optimizes the band-width of global equations by Cuthil-McGee algorithm. Called by OPTIM.

D2DO4N
Calculates derivatives for two dimensional four node elements.

D2DO9N
Calculates derivatives for two dimensional nine node elements.

D3D08N
Calculates derivatives for three dimensional eight node hexahedral elements.

D3D27N
Calculates derivatives for three dimensional quadratic elements with 27 nodes.

DAMPIN
Reads in parameters defining damping terms in transient dynamic analysis.

Common block: MACHIN

DASHIN
Reads in the definition of additional damping terms in the form of discrete dashpots.

Common block: MACHIN

DAT1
Outputs an integer array on the line printer file.

Common block: ALGEM

DAT2
Outputs a real array on the line printer file.

Common block: ALGEM

DAT3, DAT5
Outputs a pair of integer and real arrays on the line printer file.

Common block: ALGEM

DATEMMS
Prints out elastic beam section properties.

Common block: ALGEM

DATCG1
Modifies parameter data during a restart job.
Common block:  ALGEM, CONTRO, ERRORS, HARMON, PARAM, PERIOD

DATER
Calls systems' routine to pull out today's date and time of the day. System dependent. See Section 0.4.

DATIN1
Supervises read operations of parameter data.

Common block:  ALGEM, COUNT, CTITLE, ERRORS, FREE

DATIN2
Supervises read operations of model data and construction of in-core database.

Common block:  ADDVAL, PULSES, POWER, EIGEN, MODSUP, HARMON, AUTOIN, BSECT, TIME, ALGEM, CONTRO, DAMP, MAXIMA, PARAM, START1, START2, START3, START4, START5, START7, SUBELM, SUBSTR, SHIFT, TOLER, INCCON, ERRORS, FREE

DATIN3
Supervises read operations of incremental data.

Common block:  ALGEM, AUTOIN, CONTRO, COUNT, EIGEN, ERRORS, FREE, MAXIMA, PARAM, START1, START2, BODYFR, START3, START4, START5, START6, START7, TIME, TOLER, INCCON

DATOU1
Prints out model data as interpreted on the line printer.

Common block:  ALGEM, CONTRO, PARAM, INCCON, BODYFR

DATOU1
Prints out parameter data as interpreted on the line printer.

Common block:  ALGEM, COUNT, START4, EIGEN, HARMON, PULSES, CONTRO, TIME, AUTOIN, BSECT, MAXIMA, TMARCH, LOUBIN, PERIOD, SHIFT, PARAM, START1, START2, START3, START5, START7, SUBSTR, INCCON, TOLER, MACHIN

DATOU4
Prints incremental data as interpreted on the line printer.

Common block:  ALGEM, CONTRO, PARAM, INCCON, TOLER
DAX04N  Calculates derivatives for four node axisymmetric elements.

DAX09N  Calculates derivatives for nine node axisymmetric elements.

DEMO2N  Calculates derivatives for two node linear Timoshenko beam elements.

DECINT  Converts a given string to an integer.
Common block:  ALGEM

DECOMP  Factorizes a symmetric matrix stored in profile form.
Common block:  PARAM

DECREA  Converts a given string to a real.
Common block:  ALGEM

DEFGUP  Updates the total deformation gradient at the end of an increment.

DERIV   Controls access to the entries in the element library.
Common block:  ALGEM, CONTO, ELTYP, ELEMEN, ERRORS

DIAM    Measures maximum bandwidth from a root node. Called by CUTHIL and used for bandwidth optimization.
Common block:  ALGEM

DIRECT  Generates three orthogonal vectors, one of which is orthogonal to the plane defined by two given vectors.
Common block:  ALGEM
DISPIN  Reads in initial displacements for transient analysis.
Common block:  MACHIN

DISTIN  Reads in distributed load data.
Common block:  MACHIN

DIV2Q2  Generates two dimensional subelement mesh composed of 9 node quadratic quadrilaterals.

DIV2X2  Generates two dimensional subelement mesh composed of 4 node linear quadrilaterals.

DMATIN  Reads in stress-strain matrix defined directly at nodes.
Common block:  MACHIN

DMPING  Sets up the modal damping matrix for the current element.

DOT (Function Subprogram)
Calculates dot-product of two double precision real vectors.

DSH04N  Calculates derivatives for four node shell elements in the lamina coordinate system.

DSHELL  Converts the notation between tensorial and matrix forms for stress-strain matrix.

DSLOAD  Controls the access to the element distributed load routines.
Common block:  CONTRO, ELTYP, ELEMEN, MAXIMA, PARAM, START1, START2, START3, START4, START5, START6, START7, MACHIN

DUPLIN  Reads in duplicate node definition data.
Common block: MACHIN

DXOUT
Prints out a double precision array for debugging purposes.

DYNAMT
Drives the transient time integration for dynamic analysis.

Common block: ALGEM, AUTOIN, PERIOD, CTITLE, DAMP, EIGEN, INCOON, ELEMEN, ELTY, MAXIMA, CONTRO, SUBSTR, ERRORS, PARAM, START1, START2, MACHIN, START3, TMARCH, START4, START5, START6, TIME

DYNOP
Assembles the element operator matrix for transient time integration.

Common block: ALGEM, TMARCH

EIGENV
Drives the eigenvalue extraction subsystem by subspace iteration.

Common block: CONTRO, ELEMEN, EIGEN, SHIFT, ELTY, PARAM, MACHIN, START3, START4, START5, START7

ELVULV
Pulls out parameters defining element characteristics from the workspace.

Common block: ALGEM, CONTRO, ELTY, START1, PARAM, ELEMEN

EQVC
Calculates the strain invariant from the given deviatoric strain tensor stored in vector form.

EQVS, EQVSTR
Calculates the stress invariant from the given deviatoric stress tensor stored in vector form.

ERROR
Prints error message on the line printer output file.

Common block: ALGEM

ETRANS
Transforms coordinates for the nodal displacement vector according
to user instructions.

Common block: ALGEM

FILINT
Fills an integer array with its address or its inverse order.

Common block: ALGEM

FILL
Fills a double precision real array with a specified number.

FIRST1
Re-numbers the nodes for Cuthill-McKee optimization.

FIXINT
Imposes constraints onto the frontal solution.

External reference: FILINT, VULVRG, COPY

FOL0IN
Reads parameters specifying the concentrated nodal follower force.

Common block: MACHIN

FORRES
Calculates forced vibration response by modal superposition.

FREDOM
Drives the linear dynamic analysis in frequency domain.

Common block: CTITLE, TIME, AUTOIN, START1, START2, START3, START4, START5, START7, START8, ALGEM, EIGEN, MODSUP, POWER, DAMP, INCON, MACHIN, ELEME, MAXIMA, EDNY, CONTRO, SUBSTR, ERRORS, PARAM, SHPT

External reference: STRUCT, TIMOUT, ASSEM1, SOLUT1, ASSEM4, RESCON, ADDINC, PRINOU, POSTOU, COPY, NUL, INITST, SUBT, MASMA, MADD, EIGENV, CNSTNM, PNTTNM, INISQ, SETRMS, QUIT

FREFOR
Reads free format numeric data line from the card reader.

Common block: ALGEM, FREE
FRNTBL
Sets up the elimination table for the frontal solution and calculates maximum front matrix size.

Common block: ALGEM, PARAM, MAXIMA, MACHIN

FRNITOP

FRONTB
Performs back substitution of global equations factorized by the frontal process.

FRONTF
Performs forward elimination of global equations by the frontal process.

FRONTR
Performs resolution by frontal solution for a newly defined load vector with a readily factorized system of equations.

FRONTS
Drives the quasi-static solution process utilizing the frontal solution scheme.

Common block: CTITLE, TIME, AUTOIN, START1, START2, START4, START5, START6, ALGEM, EIGEN, INCOON, MACHIN, ELEMEN, MAXIMA, ELTYP, CONTRO, SUBSTR, ERRORS, PARAM, START8

FRONIW
Computes the maximum front width from the element connectivity table.

Common block: ALGEM, MAXIMA, MACHIN

GAUSSP
Returns coordinates and weight factors for Gaussian quadrature.

GEN02N
Calculates derivatives for two node linear Timoshenko beam element with respect to element coordinate system.

GENCOR
Generates coordinates for subelement nodes which divide global
element uniformly.

**GENNOD**
Generates connectivity table for uniformly divided subelement mesh.

**GEOMAT**
Assembles the geometric stiffness for large displacement and buckling analyses.

External reference: GAUSSP, COPY, S2D04N, S3D08N, D2D04N, DAX04N, DSH04N, D3D08N, NOTION

**GETBSP**
Extracts data for beam section properties.

Common block: ALGEM

**G02GLO**
Rotates stresses and strains back to global configuration in finite deformation analysis.

External reference: ROTTEN, ROTBAK

**G02ROT**
Rotates stresses and strains to mid-increment configuration in finite deformation analysis.

External reference: ROTTEN, ROTFOR

**GRAV2D**
Produces body force vector due to gravity acceleration for two-dimensional elements.

Common block: ALGEM

**GRAV3D**
Produces the body force vector due to gravity acceleration for three-dimensional elements.

**GRAVAX**
Produces the body force vector due to gravity acceleration for axisymmetric elements.
Produces the body force vector due to gravity acceleration for beam elements.

GRAVSH

Produces the body force vector due to gravity acceleration for shell elements.

GSH04N

Calculates the element normal to four node shells.

HARMIN

Reads in parameters for harmonic base excitation.

Common block: ALGEM

HEAD

Prints the MHOST banner and system's information on the line printer output file and terminal screen.

Common block: CTITLE

HOLECR

Generates coordinate data for an embedded hole represented by a subelement mesh. Called by HOLEDF.

HOLEDF

Defines an element with a parameterized hole using the subelement procedure.

Common block: MAXIMA, ELTYP, SUBTYP, MACHIN, START1, TRANSF, START6

External reference: ELVULV, QNODEL, HOLECR, HOLEM

HOLEIN

Reads parameters to define a embedded hole in a global element.

Common block: ELTYP, SUBELM, MACHIN

HOLEM

Generates connectivity array for an embedded hole represented by a subelement mesh. Called by HOLEDF.

HOOKEM

Calculates the linear elastic stress-strain matrix for beam elements.
External reference: NUL, GETBSP, MUL

HOOKLW
Calculates the linear elastic stress-strain matrix for all element types other than beams.

External reference: NUL, UTEMP, UHOOK

HOST
Supervises execution of the MHOST code. See Section 1.2 for detail.

Common block: CONTRO, ALGEM, ADDVAL, TMARCH, AUTOIN, PERIOD, POWER, PULSES, EIGEN, ERRORS, MODSUP, HARMON, LOUBIN, START5, START4, PARAM, BSECT, INCCON, MACHIN, SUBSTR, SUBELM, SHIFT

ICLEAR
Zero clears an integer array.

INCRIN
Drives the reader and report writer for incremental data.

External reference: DATIN3, DATOU4, DATOU1

INCSBC
Imposes stress boundary conditions.

INIMOP
Creates initial eigenvectors for subspace iteration.

Common block: CONTRO, MAXIMA, ELTYP, ELEMEN, PARAM, EIGEN, START1, START2, START3, START4, START5, START6, START7, START8, MACHIN

External reference: QUIT, NUL, COPY, RATIO, MAXIM

INITDF
Initializes the nodal deformation gradient array.

INITFR
Allocates memory for frontal solution option.

Common block: ALGEM, CONTRO, PARAM, MACHIN, MAXIMA, EIGEN, START5, START6

INITII
Allocates memory for in-core nodal database.
Common block: ADDVAL, ALGEM, CONTRO, DAMP, EIGEN, MODSUP, HARMON, POWER, PULSES, MAXIMA, PARAM, TMARCH, PERIOD, SUBELM, SHIFT, SUBTYP, BSECT, START1, START2, START3, START4, START5, START6, START7, START8, START9, SUBSTR, MACHIN, ERRORS

INITI2
Allocates memory for the profile solution for quasi-static, dynamic and eigenvalue analyses.

Common block: ALGEM, CONTRO, PERIOD, EIGEN, ERRORS, MAXIMA, PARAM, START1, START3, START5, START4, MACHIN

INITIN
Reads parameters defining initial conditions for transient analysis.

Common block: ALGEM, ERRORS, CONTRO, MAXIMA, ELEMEN, ETYP, START1, START5, PARAM, FREE, START2, START4

External reference: KEY, DISPIN, VELCIN, ACCLIN, PERDIN, QUIT

INITSE
Allocates memory for the subelement nodal database. See Section 2.2 for detail.

Common block: SUBTYP, ALGEM

External reference: SUBELV, NULINT

INITST
Assembles the global initial stress matrix for quasi-static, dynamic and eigenvalue analysis.

Common block: AUTOIN, EIGEN, CONTRO, BODYFR, LOUBIN, ELEMEN, ETYP, ERRORS, MAXIMA, PARAM, START1, START2, START3, START4, START5, START6, START7, START8, SUBSTR, TIME, TRANSF, MACHIN

External reference: NUL, ELVULV, CNODEL, GEOMAT, ASSEM5

INRDIR
Redirects the input stream.

Common block: ALGEM, FREE

INRFRC

SUBROUTINES
Recovers the velocity and acceleration from the displacement update in transient dynamics.

Common block: TMARCH

INSIDE
Checks to see if the element is inside-out and repairs the connectivity array, if necessary, for the linear quadrilateral and hexahedral elements.

INIDYN
Initializes the scratch file for transient dynamics.

INTERP
Interpolates nodal values to the integration points.

INTINT
Initializes the file system and execution environment. System dependent. See Section 0.4. Called by MAIN program.

Common block: VRIDSK

INTSQQ
Integrates the spectral density function over a given range of frequencies.

INV3
Inverts a 3 by 3 matrix.

INVERT
Inverts a general square matrix.

Common block: ALGEM

ITERIN
Reads parameters defining convergence criteria.

Common block: TOLER

External reference: FREFOR

JACOB1
Extracts eigenvalues of a square matrix. Called by SUBSPC.

Common block: ALGEM
External reference: MAXIM, SMULT, LINES

JT (Function subprogram)
Gives the address of a specified entry in a global array stored in
band matrix form.

KEY
Checks to see if a string matches to a keyword.

Common block: ALGEM, FREE

L2NORM
Calculates 1 norm of a given vector.

LAXSYM
Calculates the load vector due to body force loading on
axisymmetric elements.

Common block: ALGEM

LAYINT
Integrates lamina quantities to resultants for thick shell
elements.

LELAST
Calculates linear elastic response of a material for a pre-defined
stress-strain matrix.

LETcmd
Sets up the integer control variables for the compound execution
of analysis drivers. See Section 1.2 for detail.

Common blocks: ALGEM, CONTRO, COMPND, PARAM, EIGEN, MACHIN

LEVEL
A part of the front matrix size optimization package. Not used in
Version 4.2.

LINES
Advances the line number of the line printer output file (ILPRNT).

Common block: ALGEM

External reference: PAGE
LINE52
Advances the line number of the line printer output file (JLPRNT).
Common block: MAXIMA, ALGEM, ZPRINT
External reference: PAGE2, PAGE3

LINESR
Calculates search distances for the line search algorithm.
Common block: MACHIN, CONTR, ALGEM, AUTOIN, PARAM, START3, START4, START5, START7, TOLER

LINESU
Advances the line number of the line printer output page used for the subelement solution.
Common block: MAXIMA, ALGEM, ZPRINT
External reference: PAGE2S, PAGE 3S

LMPMAS
Calculates the diagonalized Gramm matrix for nodal strain projections. Either Gauss-Lobatto or Gauss/now-sum algorithm can be used.

LOCVEC
Fills up the address table for the global array stored in band matrix form.
Common block: ELTYP

LPSTRN
Calculates the load vector due to body force loading on plane strain elements.

LPSTRS
Calculates the load vector due to body force loading on plane stress elements.

LSHELL
Calculates the load vector due to body force loading on shell elements.

LSOLID
Calculates the load vector due to body force loading on solid elements.

**LTBEAM**
Calculates the load vector due to body force loading on beam elements.

**MADD**
Adds a matrix to a product of another matrix and a scalar.

**MAIN**
Main program.

Common block: blank, MACHIN

**MASMAT**
Assembles the global consistent matrix stored in profile form.

Common block: AUTOIN, EIGEN, CONTRO, BODYFR, LOUBIN, ELEMEN, ELTYP, ERRORS, MAXIMA, BSECT, PARAM, START1, START2, START3, START4, START5, START6, START7, START8, SUBSTR, TIME, TRANSF, MACHIN

**MASSIN**
Reads in the connectivity and magnitude of added mass.

Common block: MACHIN

**MATINV**
Inverts a given square matrix.

**MATONE**
Fills the diagonal entry of a global matrix by unity.

**MATPRT**
Prints out a two dimensional array with a header.

Common block: ALGEM

**MATSUM**
Adds two arrays multiplied by given constants for each array.

**MAXCON**
Finds the maximum connectivity at nodes used for bandwidth optimization.
MAXIM
Pulls out the maximum value from a double precision array.

PERDIN
Reads data defining the periodic loading in transient dynamics.
Common block: MACHIN, ELTYP, PERIOD

PERDOP
Applies periodic loading to the current time integration step.
Common block: ALGEM

PJOOP
Debug - writes a double precision real variable/array.

PLASTD
Calculates a consistent tangent modulus for an elastic plastic material at a given state of stress and deformation history.
Common block: ELEMEN

PLASTS
Calculates the stress state of an elastic-plastic material by return mapping.
Common block: ELEMEN

PAGE
Advances a page of line-printer output file (ILPRNT) and prints the header.
Common block: ALGEM, CTITLE, PAGCNT

PAGE2
Advances a page of line-printer output file (JLPRNT) and prints the header for nodal variables.
Common block: ALGEM, CONTRO, CTITLE, RESULT, TIME, PAGCNT

PAGE3
Advances a page of line-printer output file (JLPRNT) and prints the header for element variables.
Common block: ALGEM, CONTRO, CTITLE, RESULT, TIME, PAGCNT
Same as PAGE2. Used for subelement solution.

Same as PAGE3. Used for subelement solution.

Zero - clears a double precision real array.

Zero - clears an integer array.

Zero - clears space allocated for nodal normals of shell coordinate definition.

Controls execution of the bandwidth optimization option.

External reference: LINES, MKFAKE, MAXCON, CUTHIL, QUIT

Generates the coordinate transformation matrix to the preferred orientation of an anisotropic material.

Performs the outer product of two three-dimensional vectors.

Common block: START2, START3, START4, START5, START6, START7, START8, SUBSTR, TIME, TRANSF, MACHIN

Updates the velocity vector at the beginning of a time integration step.

Computes the nodal pressure and directly adds it into the global load vector.

Computes stress at a given point and generates material tangent.

External reference: NUL, COPY, ROTPRF, THRSTN, CRPSTN, SUBT, SIMPLE, MULT, LELAST, PLASTS, PLASTD, WALKEQ
NOTION
Converts stresses and strains stored in vector form to tensorial form.

NRMNRM
Normalizes the nodal normal vector and stores its components as a part of the coordinate data for shell elements.

MULT
Multiplies two two-dimensional matrices.

MULTT
Multiplies a matrix by the transpose of another matrix.

NEWACC
Updates the acceleration vector at the beginning of a time integration step.

NGWADD
Allocates the acceleration and velocity vector at the beginning of a time step to temporary storage.

NEWMRK
Adds the element stiffness and mass matrices to generate the element time integration operator matrix.

External reference: MADD

NEWRHS
Forms global load vector.

Common block: AUTOIN, BSECT, CONTRO, BODYFR, LOUBIN, ELEMEN, ELTYP, ERRORS, PULSES, MAXIMA, PARAM, START1

MESURE
Calculates a distance from the integration point to a straight line defined by two points.

MID
Calculates the coordinate of the mid edge node. Not used in Version 4.2.

MIDDLE
Drives the calculation of coordinates and connectivity for mid edge nodes. Not used in Version 4.2.
MKFAKE
Sets up a fake connectivity table including tying constraint for bandwidth optimization.

MODAL
Drives the modal analysis.

Common block: CTITLE, TIME, AUTOIN, START1, START4, START5, ALGEM, EIGEN, INCON, MACHIN, ELEMEN, MAXIMA, ELTYP, CONTR0, SUBSTR, ERRORS, PARAM, SHIFT, START8

External reference: STRUCT, TIMDUT, ASSEM1, MASMAT, MADD, EIGENV, POSTOU

PMTINM
Sets up the convolution terms for nodal loads.

POLD2D
Performs polar decomposition of a two-dimensional second order tensor.

POLD3D
Performs polar decomposition of a three-dimensional second order tensor.

POLICE
Debug - writes contents of a specified common block.

POSTEN
Packs the record of nodally defined element variables for the post tape. See Section 3.4 for detail.

Common block: POSTPN

POSTOU
Writes out post processing data. See Section 3.4 for detail.

POSTPR
Writes contents of buffer to the post processing tape.

PREFIN
Reads in the parameters defining the preferred orientation of an anisotropic material.

Common block: MACHIN
PRELEM

Extracts information of the current element from the nodal database. A simplified version of CNODEL used in PRINOU.

Common block: CONTRO, ELTYP, MAXIMA, PARAM, START1, START2, START3, START4, START5, START6, START7, START8, MACHIN

PRESET

Initializes working arrays used for front matrix size optimization.

Common block: ALGEM

PRESIN

Reads parameters defining pressure loads.

Common block: MACHIN

PRFRNT

Controls the front matrix size calculation and the subsequent memory allocation.

Common block: CONTRO, PARAM, ELEMEN, ELTYP, MAXIMA, MACHIN, START1, START2, START3

PRINCV

Solves a 3 by 3 eigen problem by Jacobi iteration.

PRININ

Reads control data for line printer output.

Common block: FREE

PRIN01, PRIN02, PRIN03

Writes an array with a header.

PRINOU

Controls the line printer output of the global solution.

Common block: CONTRO, ALGEM, ELEMEN, ELTYP, RESULT, START1, START2, START4, START6, SUBELM, TIME, TRANSF, ZPRINT, MACHIN

PRINSU

Controls the line printer output of the subelement solution.
Common block: CONTRO, ALGEM, ELEMEN, ELTYP, RESULT, START1, START2, START4, START6, TIME, TRANSF, ZPRINT, MACHIN

PRINITM
Debug - writes the element stiffness matrix.

Common block: ALGEM, ELEMEN

PRINTS
Debug - writes the global stiffness matrix stored in band matrix form.

Common block: ALGEM

PRNSHL
Prints the nodal coordinate transformation matrix for shells.

PRNTL
Reports the connectivity table resulting from automatic subelement mesh generation.

PRNINO
Reports the coordinates resulting from automatic subelement mesh generation.

PRTERR
Prints error message and does not STOP but returns the control.

Common block: ALGEM

PRWARN
Prints warning message.

Common block: ALGEM

PSDIN
Reads the power spectrum definition.

Common block: MACHIN

PULSIN
Reads the pulse load definition.

Common block: MACHIN
PUTDUP  Adds the duplicate node option as a special element in the frontal solution subsystem.

PUTTIE  Adds the tying equation as a special element in the frontal solution subsystem.

QNODEL  Extracts element quantities from the global nodal database. This is yet another subset of CNODEL.

Common block: TMARCH, ALGEM, CONTRO, ELTYP, MAXIMA, PARAM, START1, START2, START4, START5, START6, START7, START8, START9, MACHIN

QUIT  Prints the error message and terminates the execution.

Common block: ALGEM

R3DTEN  Defines three-dimensional rotation matrix for decomposition of the deformation gradient.

RAMDSK  Initializes the buffer area used for the out-of-core frontal solution.

Common block: VRTDSK, ALGEM, PARAM, MACHIN, START5

RATIO  Subdivides entries of one array by corresponding entries of another array.

RBF  Rotates stress and strain vectors to preferred orientations.

READEX  Reads header and a double precision real array from a binary file.

REASPR  Calculates reaction force due to an added spring.

REDIAG  Reduces the diagonal entries of a profile store global stiffness
matrix.

RELDG

Computes the nodal incremental deformation gradient.

RELOAD

Forms the follower force at the end of an increment.

Common block: ALGEM, ADDVAL, AUTOIN, BSECT, CONTRO, BODYFR, LOUBIN, ELEMEN, ELITYP, ERRORS, MAXIMA, PARAM, PULSES, START1, START2, START3, START4, START5, START6, START7, START8, SUBELM, SUBSTR, TIME, TRANSF, MACHIN

External reference: NUL, ADD, RTFOLF, ELVULV, CNODEL, DERIV, INTERQ, STIFF, ETRANS, SUBST1, BFLOAD, DSLOAD, QUIT

RESCCHK

Tests for convergence flag associated with the residual.

RESCON

Tests for convergence in terms of residual reaction and displacement.

Common block: ALGEM, CONTRO, PARAM, START1, START2, START3, START4, START5, START6, START7, START8, SUBELM, SUBSTR, TIME, TRANSF, MACHIN

External reference: ADD, SUBT, TYING2, BOUND2, MAXIM, L2NORM, LINES, SUBT

RESID

Calculates the contribution of mass and damping terms to the residual vector. Called by ASSEM4 when the direct time integration flag is on.

Common block: TMARCH, ALGEM

RESEQ1


RESEQ2

Forms the residual vector. Called by ASSEM4.

Common block: PARAM

RESOLV

Resolves the same equation by frontal solution method for a different load vector.

Common block: ALGEM

RESTRT

Reads the restart file. See Section 3.3 for detail.

Common block: ADOVAL, ALGEM, AUTOIN, BODYFR, BSECTION, CONTROL, COMPOUND, COUNT, DAMP, EIGEN, MODSUP, ELEMEAN, HARMON, INCO, MAXIMA, LOUBIN, PARAM, PERIOD, POSTPIN, POWER, PULSES, RESULT, START1, START2, START3, START4, START5, START6, START7, START8, START9, SUBELM, SUBTYPE, SUBSTR, SHIFT, TIME, TMARCH, TOLER, VRIDSK, ZPRINT

ROTBK

Rotates the stress state back to the global coordinate system in finite deformation analysis.

ROTDMT

Rotates the element stress strain matrix into the global coordinate system for shells.

External reference: ORIENT, COPY, QUIT, MATINV, NUL, TSHIFT, TFULL2, MULT

ROTFOR

Rotates the stress state state into the deformed configuration coordinate system in finite deformation analysis.

ROTPRF

Rotates the stresses and strains defined in global coordinates to the preferred orientation of an anisotropic material.

ROTTEN

Calculates the rotation tensor from the polar decomposition of the deformation gradient in finite deformation analyses.

ROW

Pulls out a complete row from profile stored global matrix.
RSHELL Treats the random vibration input for shell elements.
RTBEAM Treats the random vibration input for beam elements.
RTPOLF Rotates load vector entries treated as follower forces.
RUNCMD Loads control parameters for compound executions.

Common block: CONTRO, COMPND, MACHIN

SID02N Shape functions for the one-dimensional two node element.
S2D04N Shape functions for the two-dimensional four node element.
S2D09N Shape functions for the two-dimensional nine node element.
S3D08N Shape functions for the three-dimensional eight node element.
S3D27N Shape functions for three-dimensional twenty-seven node element.
SAVER: Writes the restart file. See Section 3.3 for detail.

Common block: ADDVAL, ALGEM, AUTOIN, BODYFR, BSECT, CONTRO, COMPND, COUNT,
DAMP, EIGEN, MODSUP, ELEMEN, HARMON, INCON, MAXIMA, LOUBIN,
PARAM, PERIOD, POSTPN, POWER, PULSES, RESULT, START1,
START2, START3, START4, START5, START6, START7, START8,
START9, SUBelm, SUBTyp, SUBSTR, SHIFT, TIME, TMARCH, TOLER,
VRDSK, ZPRINT

SBCIN Reads in stress boundary condition data.
Common block: MACHIN

SCALER
Scales loading to a given proportion.

**Common block:**  MACHIN, START8

**SEARCH**
Pulls out double precision real element nodal quantities from the global nodal data base.

**SEARCI**
Pulls out integer element nodal quantities from the global nodal database.

**SELECT**
Reads and decodes strings for print option data input.

**SETCCM**
Sets up the complex coefficient matrix for base excitation.

**SETHFN**
Sets up the vector for complex modal damping.

**SETOLR**
Sets the limit for the search distance in the line search option.

**SETQMD**
Calculates modal damping.

**SETRMS**
Finds the root-mean-square value for a modal function from the frequency response.

**SETUP**
Computes the adjacency list for front width optimization. Not functional for Version 4.2.

**SHIFIN**
Reads parameters defining the power shift in modal dynamics.

**Common block:**  MACHIN

**SHOHEI**
Debug - writes an integer array.

**SHTRAN**
Transforms nodally defined shell stress and strain resultants to...
values in element coordinates.

**SIMPLE**

Calculates the stress and generates the tangent modulus for secant elasticity model for elastoplastic response under a monotonically increasing load.

**SK**

Integrates the element coefficient matrix into the global array stored in band matrix form.

**SMASTR**

Extracts element quantities from the global element used for subelement analysis. Yet another variation of CNODEL.

Common block: TMARCH, CONTRO, ELTYP, MAXIMA, PARAM, START1, START2, START4, START5, START6, START8, MACHIN

**SMULT**

Multiplies an array by a scalar.

**SNODEL**

Transfers global element results into the subelement mesh.

**SOLUT1**

Controls execution of the linear algebraic equation solver by the profile method.

Common block: AUTOIN, CONTRO, EIGEN, ELTYP, MAXIMA, PARAM, PULSES, START1, START3, START4, START5, START7, MACHIN, TOLER

External reference: ADD, RTFOLF, ADDPUL, SUBT, TYING1, BOUND1, DECOMP, BFGSW, BFGSRH, COPY, SOLVER, MADD, MATSUM, BFGSLH, TYING3, L2NORM, TYING2, BOUND2, ADAPTS

**SOLUT2**

Controls execution of the linear algebraic equation solver by the band matrix method. Not used in Version 4.2.

**SOLVER**

Solves linear equations for nodal displacement.

Common block: ALGEM, PARAM

**SOLVIT**
Solves a small system of algebraic equations by Gaussian elimination.

SPRIIN
Reads parameters defining the added spring stiffness.

Common block: MACHIN

SPSTRS
Generates the strain displacement matrix at a given point in the current element of plane stress type.

SSOLID
Generates the strain displacement matrix at a given point in the current element of three-dimensional continua type.

STATIC
Drives the incremental iterative solution of a quasi-static problem.

STICKIO
Controls the data flow between in-core buffer and the work file for the frontal solution.

Common block: ALGEM, MACHIN, VRTDSK

STIFF
Calculates the current element tangent stiffness matrix and the load vector.

External reference: NUL, MATPRT, SMULT, MULT, TMULT

STRAIN
Recovers strain at nodes.

Common block: COUNTR

STRESS
Controls the nodal stress recovery operation including pre-integration through the shell thickness.

STRING
Decodes the input data string.
Common block: ALGEM, FREE, COUNT

STRPB Finds non-blank entry in a given character string.

STRSBC Imposes the stress boundary condition called by ASSEM4.

STRUCT Controls the core allocation and elimination table construction for global matrix manipulation.

Common block: ALGEM, CONTRO, MAXIMA, PARAM, START1, START2, START3, START4, START5, SUBSTR

SUBALC Allocates working storage for the subelement solution.

SUBCHK Checks convergence of the subelement solution in terms of displacement update.

Common block: ALGEM, SUBCNV

SUBDER Controls access to the element library in the subelement solution.

Common blocks: ALGEM, CONTRO, SUBTYP, ELEMEN, ERRORS

SUBDIV Controls the subelement mesh generation.

Common blocks: ALGEM, CONTRO, ELTYP, ERRORS, MACHIN, MAXIMA, PARAM, START1, START5, START6, SUBTYP

SUBELV Extracts element definition variable for the current subelement.

Common block: ALGEM, SUBTYP, START1, PARAM, ELEMEN

SUBFEM Supervises the mixed iterative solution in the subelement mesh. Called by ASSEM4.

Common block: ALGEM, CONTRO, ELTYP, ELEMEN, SUBTYP, TRANSF, LOUBIN,
Calculates residual for a global element generated by the subelement solution.

SUBINC

Adds the incremental values of the subelement solution to the total array.

Common block: AUTOIN, CONTRO, SUBTYP, START1, START2, START4, START6, START8, TIME

SUBINT

Interpolates global quantities at subelement nodes.

SUBRES

Calculates the mixed residual for the subelement solution.

SUBSIN


SUBSOL, SUBS02

Back-substitutes for the factors of the global array stored in band matrix form to generate the displacement corresponding to the given load vector. Not used in Version 4.2.

SUBSOL

Solves the subelement stiffness equations for the subelement displacement.

SUBSPC

Controls execution of the subspace iteration for eigenvalue extraction.

Common block: ALGEM, SHIFT

External reference: NUL, UPIX, JACOBI, TIMOUT, COPY, MADD, QUIT, SMULT, LINES
SUBST1  Converts the element load vector to the global load vector.

SUBSTN  Adds the element nodal value to the global vector.

SUBT  Subtracts arrays.

SUBVAL  Interpolates element nodal values to a given point specified by the isoparametric coordinate system.

SUPER  Drives modal superposition for linear dynamics.

Common block:  CTITLE, TIME, AUTOIN, START1, START2, START4, START5, ALGEM, EIGEN, MODSUP, HARMON, INCON, DAMP, MACHIN, ELEMEN, MAXIMA, ELTYP, CONTRO, SUBSTR, ERRORS, PARAM, SHIFT, START8

SYSEQN  Assembles the global finite element equations for subelement analysis.

T2D04N  Calculates the local coordinate system for two-dimensional four node elements.

T2D04P  Calculates the local coordinate system for two-dimensional four node elements using the Cayley-Hamilton formula.

T3D08N  Calculates the local coordinate system for three-dimensional eight node element.

TBMO2N  Calculates the local coordinate system for two node beam element.

TEMPIN  Reads in nodal temperature definition data.

Common block:  MACHIN

TFULL2
Transforms the material modulus into fully three-dimensional form.

**THRSTN**
Calculates thermal strain.

 External reference: **UTEMP, NUL, UCOEF**

**TIMEIN**
Reads in parameters defining the time increment control.

**TIMER**
Accesses to the system dependent CPU clock routine. See Section 0.3.

**TIMOUT**
Reports the job step and elapsed CPU time.

Common block: **ALGEM, CONTRO, TIMLOC**

**TMULT**
Multiplies the transpose of an array by a matrix.

**TMULTIV**
Multiplies the transpose of an array by a vector.

**TNSPRD**
Multiplies two tensors stored in vector form.

**TRACIN**
Reads in nodal concentrated forces.

Common block: **MACHIN**

**TRANIN**
Reads nodal coordinate transformation data.

Common block: **MACHIN**

**TRANS1**
Applies a transformation to the global stiffness matrix stored in band matrix form.

**TRANS2**
Applies a transformation to the global vector.
TRANSP  Transposes a two-dimensional matrix.
TSH04N  Calculates local coordinates for four node shell elements.
TSHIFT  Sets up tensor transformation with respect to the preferred orientation of an anisotropic material.
TYING1  Imposes tying constraints on the global stiffness matrix stored in band matrix form.
TYING2  Imposes tying constraints on the global vector.
TYING3  Releases tying constraints in the global vector.
TYININ  Reads in coefficients of the tying equations.

Common block:  MACHIN

TYPEIN  Defines element parameters used in the analysis.

UBOUN,  UCOEF,  UCOOR,  UDERIV,  UFORCE,  UHOOK,  UPRESS,  UTEMP,  UTERM,  USXX,
VSWELL  User subroutines. Fully documented in MHOST Volume 1 USERS' MANUAL.

UNITST  Generates the global unit matrix in profile stored form.

UPTX  Multiplies a banded matrix by a vector.

UPTXL  Multiplies a lumped mass matrix by a vector.

VALINT  Linearly interpolates data given in a table.
VDSKID
Manages a record for out-of-core frontal solution.

Common block: ALGEM, MAXIMA, MACHIN, VRITDSK

VELCIN
Reads initial velocity definition data.

Common block: MACHIN

VMULT
Multiplies an array by a vector.

VSH04N
Generates the local coordinate system for four node shell elements.

VIMULT
Multiplies the transpose of an array by a vector.

VULVRG
Makes the conversion table for the degree of freedom from the connectivity and constraint data.

VVMULT
Multiplies two vectors.

WALCON
Calculates temperature dependent material constraints for the Walker model.

WALKEQ
Calculates stress and internal variables for the Walker model.

WORKIN
Reads in work-hardening data for a homogeneous elastic plastic material.

Common block: MACHIN

WKSLP
Calculates the hardening slope for elastoplasticity.

WRITEX
YIEL

Calculates the yield stress and hardening slope for a given equivalent plastic strain.

External reference: WKSLP

YIELIN

Reads in the nodal definition of a stress-strain curve for an elastic plastic material.

Common block: MACHIN