### 16. Abstract

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 storagescheme 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

0.0 INTRODUCTION ................................................................. 1

0.1 Program Architecture .................................................. 1

0.2 Rules and Conventions ................................................... 3

0.3 Subroutines and Common Blocks ....................................... 5

0.4 System Dependency ........................................................ 10

1.0 CONTROL STRUCTURE .......................................................... 17

1.1 Overview .............................................................................. 17

1.2 Execution Supervisor and Analysis Drivers ............................. 22

1.3 Control Variables ................................................................. 29

2.0 DATA STORAGE SCHEME .......................................................... 39

2.1 Overview .............................................................................. 39

2.2 Memory Allocation Subprogram .......................................... 40

2.3 Counters and Pointers ......................................................... 96

3.0 THE FILE SYSTEM ................................................................. 104

3.1 Overview .............................................................................. 104

3.2 User Interface ...................................................................... 105

3.3 Restart File ......................................................................... 136

3.4 Post-Processing Data File .................................................... 149

APPENDIX Subroutines ............................................................... A-1

(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:

SnmmN  Shape functions for elements consisting of mm-nodal points in nn-geometrical configuration.

DnmmN  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:

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Subroutine</th>
<th>Subroutine</th>
<th>Subroutine</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCLIN</td>
<td>ADAPIC</td>
<td>ADAPTID</td>
<td>ADAPTS</td>
</tr>
<tr>
<td>ADD</td>
<td>ADDGANC</td>
<td>ADDSIM</td>
<td>ADDINC</td>
</tr>
<tr>
<td>ADDINV</td>
<td>ADDPUL</td>
<td>ADDSMU</td>
<td>ANPLAS</td>
</tr>
<tr>
<td>ASMVEC</td>
<td>ASSEM1</td>
<td>ASSEM2</td>
<td>ASSEM3</td>
</tr>
<tr>
<td>ASSE44</td>
<td>ASSEM5</td>
<td>ATTRIB</td>
<td>BASSUB</td>
</tr>
<tr>
<td>BANDER</td>
<td>BANNER</td>
<td>BASAXS</td>
<td>BASEIN</td>
</tr>
<tr>
<td>BNSPSN</td>
<td>BASPSTL</td>
<td>BASSOL</td>
<td>BAXSYM</td>
</tr>
<tr>
<td>BEAMIN</td>
<td>BEFSRLH</td>
<td>BFGSRH</td>
<td>BFGSW</td>
</tr>
<tr>
<td>BFLOAD</td>
<td>BLOCK01</td>
<td>BMSTRS</td>
<td>BNDTRM</td>
</tr>
<tr>
<td>BODYIN</td>
<td>BOUND1</td>
<td>BOUND2</td>
<td>BOUND3</td>
</tr>
<tr>
<td>BOUNDN</td>
<td>BOUNDF</td>
<td>BOUNIN</td>
<td>BPSRN</td>
</tr>
<tr>
<td>BPSSTRS</td>
<td>BREAD</td>
<td>BSHDKQ</td>
<td>BSHELL</td>
</tr>
<tr>
<td>BSOLID</td>
<td>BTEAM</td>
<td>BUCKLE</td>
<td>BULKIN</td>
</tr>
<tr>
<td>BWRITE</td>
<td>CENMAS</td>
<td>CENT2D</td>
<td>CENT3D</td>
</tr>
<tr>
<td>CNTAX</td>
<td>CENTBM</td>
<td>CENTSH</td>
<td>CHARIN</td>
</tr>
<tr>
<td>CHCHAR</td>
<td>CHKEM</td>
<td>CLENUP</td>
<td>CLOSEF</td>
</tr>
<tr>
<td>CNODEL</td>
<td>CNNSMA</td>
<td>CNSTNM</td>
<td>COLRED</td>
</tr>
<tr>
<td>COMPDF</td>
<td>COMPPB</td>
<td>COMPRO</td>
<td>CONDSE</td>
</tr>
<tr>
<td>CONJUG</td>
<td>CONNIN</td>
<td>CONTIN</td>
<td>COORIN</td>
</tr>
<tr>
<td>COPY</td>
<td>COPYCH</td>
<td>COPYDS</td>
<td>COPYIN</td>
</tr>
<tr>
<td>COPYSD</td>
<td>CORDDTR</td>
<td>COROUT</td>
<td>CPXCKX1</td>
</tr>
<tr>
<td>CPXBKR2</td>
<td>CPXDIV</td>
<td>CPXEXEC</td>
<td>CPXKAC</td>
</tr>
<tr>
<td>CPXFOR</td>
<td>CPXMUL</td>
<td>CPXREA</td>
<td>CPXRES</td>
</tr>
<tr>
<td>CRPLAW</td>
<td>CRPSNN</td>
<td>CUTHIL</td>
<td>D2D04N</td>
</tr>
<tr>
<td>D2D09N</td>
<td>D3D08N</td>
<td>D3D27N</td>
<td>DAMP3N</td>
</tr>
<tr>
<td>DASHIN</td>
<td>DAT1</td>
<td>DAT2</td>
<td>DAT3</td>
</tr>
<tr>
<td>DAT5</td>
<td>DATBMS</td>
<td>DATCG1</td>
<td>DATER</td>
</tr>
<tr>
<td>DATIN1</td>
<td>DATIN2</td>
<td>DATIN3</td>
<td>DATOH1</td>
</tr>
<tr>
<td>DATOU1</td>
<td>DATOU4</td>
<td>DAX04N</td>
<td>DAX09N</td>
</tr>
<tr>
<td>DEMO2N</td>
<td>DECINT</td>
<td>DECOMP</td>
<td>DECREA</td>
</tr>
<tr>
<td>Command</td>
<td>Command</td>
<td>Command</td>
<td>Command</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
</tr>
<tr>
<td>DEFGUP</td>
<td>DERIV</td>
<td>DIAM</td>
<td>DIRECT</td>
</tr>
<tr>
<td>DISPIN</td>
<td>DISTIN</td>
<td>DIV2Q2</td>
<td>DIV2X2</td>
</tr>
<tr>
<td>DMATIN</td>
<td>DMP1NG</td>
<td>DOT</td>
<td>DSH04N</td>
</tr>
<tr>
<td>DSHELL</td>
<td>DSLOAD</td>
<td>DUPLIN</td>
<td>DXOUT</td>
</tr>
<tr>
<td>DYNAMC</td>
<td>DYNAMT</td>
<td>EIGENV</td>
<td>EIGENV</td>
</tr>
<tr>
<td>ELVULV</td>
<td>EQVC</td>
<td>EQVS</td>
<td>EQVSTR</td>
</tr>
<tr>
<td>ERROR</td>
<td>ETTRANS</td>
<td>EXITN</td>
<td>EFIXT</td>
</tr>
<tr>
<td>FILINT</td>
<td>FILL</td>
<td>FIRST1</td>
<td>FEGEI</td>
</tr>
<tr>
<td>FOLOIN</td>
<td>FORRES</td>
<td>FREDOM</td>
<td>F'REFOR</td>
</tr>
<tr>
<td>FRNTNL</td>
<td>FRNTOP</td>
<td>FRONTB</td>
<td>FRONTF</td>
</tr>
<tr>
<td>FRONTR</td>
<td>FRONTS</td>
<td>GENNOD</td>
<td>GEMAT</td>
</tr>
<tr>
<td>GBM02N</td>
<td>GO2GLO</td>
<td>GO2ROT</td>
<td>GRAV2D</td>
</tr>
<tr>
<td>GETBSP</td>
<td>GRAVAX</td>
<td>GRAVBM</td>
<td>GRAVSH</td>
</tr>
<tr>
<td>GRAV3D</td>
<td>HARMIN</td>
<td>HEAD</td>
<td>HEADCR</td>
</tr>
<tr>
<td>GSH04N</td>
<td>HOLEIN</td>
<td>HOLEM</td>
<td>HOLEM</td>
</tr>
<tr>
<td>HOLEDF</td>
<td>HOST</td>
<td>ICLEAR</td>
<td>INCIN</td>
</tr>
<tr>
<td>HOOKGW</td>
<td>INIMOD</td>
<td>INITDF</td>
<td>INITFR</td>
</tr>
<tr>
<td>INCSEC</td>
<td>INITI2</td>
<td>INITIN</td>
<td>INTISE</td>
</tr>
<tr>
<td>INTI1</td>
<td>INRDIR</td>
<td>INRFRC</td>
<td>INSIDE</td>
</tr>
<tr>
<td>INTI2</td>
<td>INTETP</td>
<td>INTETR</td>
<td>INTETR</td>
</tr>
<tr>
<td>INV3</td>
<td>INVERT</td>
<td>ITERIN</td>
<td>INTOQ</td>
</tr>
<tr>
<td>JT</td>
<td>KEY</td>
<td>L2NORM</td>
<td>JACOBI</td>
</tr>
<tr>
<td>LAYINT</td>
<td>LELAST</td>
<td>LETCM</td>
<td>LAUXSYM</td>
</tr>
<tr>
<td>LINES</td>
<td>LINES2</td>
<td>LINESR</td>
<td>LINESU</td>
</tr>
<tr>
<td>LMPMAS</td>
<td>LOCEVC</td>
<td>LPSRC</td>
<td>LPSTRES</td>
</tr>
<tr>
<td>LSHELL</td>
<td>LSOLID</td>
<td>LTBEM</td>
<td>LMBAD</td>
</tr>
<tr>
<td>MAIN</td>
<td>MASMAT</td>
<td>MASSIN</td>
<td>MATINV</td>
</tr>
<tr>
<td>MAIONE</td>
<td>MATPRT</td>
<td>MATSUM</td>
<td>MAXCON</td>
</tr>
<tr>
<td>MAXIM</td>
<td>MESURE</td>
<td>MID</td>
<td>MIDDLE</td>
</tr>
<tr>
<td>MFKAKE</td>
<td>MODAL</td>
<td>MULT</td>
<td>MULTT</td>
</tr>
<tr>
<td>NEWACC</td>
<td>NEWADD</td>
<td>NEWMARK</td>
<td>NEWRHS</td>
</tr>
<tr>
<td>NEWVEL</td>
<td>NODPRE</td>
<td>NODST</td>
<td>NOTION</td>
</tr>
<tr>
<td>NRMNRM</td>
<td>NUL</td>
<td>NULINT</td>
<td>NULNRM</td>
</tr>
<tr>
<td>OPTTEST</td>
<td>OPTIM</td>
<td>ORIENT</td>
<td>OUTPRO</td>
</tr>
<tr>
<td>PAGE</td>
<td>PAGE2</td>
<td>PAGE2S</td>
<td>PAGE3</td>
</tr>
<tr>
<td>PAGE3S</td>
<td>PERDIN</td>
<td>PERDOP</td>
<td>PJOOP</td>
</tr>
<tr>
<td>PLASTD</td>
<td>PLASTS</td>
<td>PNTN</td>
<td>POLD2D</td>
</tr>
<tr>
<td>POL3D</td>
<td>POLICE</td>
<td>POSTEN</td>
<td>POSTOU</td>
</tr>
<tr>
<td>POSTPR</td>
<td>PREFIN</td>
<td>PRELEM</td>
<td>PRESET</td>
</tr>
<tr>
<td>PRESIN</td>
<td>PRFRMT</td>
<td>PRINCV</td>
<td>PRININ</td>
</tr>
<tr>
<td>PRINO1</td>
<td>PRINO2</td>
<td>PRINO3</td>
<td>PRINOU</td>
</tr>
<tr>
<td>PRINSU</td>
<td>PRINTM</td>
<td>PRINTS</td>
<td>PRNSHL</td>
</tr>
<tr>
<td>PRNTIL</td>
<td>PRININO</td>
<td>PRIETR</td>
<td>PRWARN</td>
</tr>
<tr>
<td>PSDIN</td>
<td>PULSIN</td>
<td>PUXDUP</td>
<td>PUTTE</td>
</tr>
<tr>
<td>Block</td>
<td>Block</td>
<td>Block</td>
<td>Block</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>QNODEL</td>
<td>QPSTRS</td>
<td>QUIT</td>
<td>R3DTEN</td>
</tr>
<tr>
<td>RAMDSK</td>
<td>RATIO</td>
<td>RBF</td>
<td>READEX</td>
</tr>
<tr>
<td>RASPR</td>
<td>REDTAG</td>
<td>RELDGF</td>
<td>RELDGF</td>
</tr>
<tr>
<td>RESCHK</td>
<td>RESCON</td>
<td>RESDYN</td>
<td>RESQ1</td>
</tr>
<tr>
<td>RESSEQ2</td>
<td>RESID</td>
<td>RESOLV</td>
<td>RESTRT</td>
</tr>
<tr>
<td>ROTBAK</td>
<td>ROTDMT</td>
<td>ROTFOR</td>
<td>ROTPRF</td>
</tr>
<tr>
<td>ROTTEN</td>
<td>ROW</td>
<td>RSHOLL</td>
<td>RBEAM</td>
</tr>
<tr>
<td>RTDOLF</td>
<td>RUNCMD</td>
<td>S1D02N</td>
<td>S2D04N</td>
</tr>
<tr>
<td>S2D09N</td>
<td>S3D08N</td>
<td>S3D27N</td>
<td>S4VER</td>
</tr>
<tr>
<td>SBCIN</td>
<td>SCALER</td>
<td>SEARCH</td>
<td>SEARCI</td>
</tr>
<tr>
<td>SELECT</td>
<td>SETCCM</td>
<td>SETHEN</td>
<td>SETOLR</td>
</tr>
<tr>
<td>SETOMD</td>
<td>SETRMS</td>
<td>SETUP</td>
<td>SHIFIN</td>
</tr>
<tr>
<td>SHOEIE</td>
<td>SHTRAN</td>
<td>SIMPLE</td>
<td>SK</td>
</tr>
<tr>
<td>SMASTR</td>
<td>SMULT</td>
<td>SNODEL</td>
<td>SOLUT1</td>
</tr>
<tr>
<td>SOLUT2</td>
<td>SOLUTN</td>
<td>SOLVER</td>
<td>SOLVIT</td>
</tr>
<tr>
<td>SPRIN</td>
<td>SPSTRS</td>
<td>SSOLID</td>
<td>STATIC</td>
</tr>
<tr>
<td>STICKIO</td>
<td>STIFF</td>
<td>STRAIN</td>
<td>STRESS</td>
</tr>
<tr>
<td>STRING</td>
<td>STRIPB</td>
<td>STRSBC</td>
<td>STRUCT</td>
</tr>
<tr>
<td>SUBALC</td>
<td>SUBCK</td>
<td>SUBDER</td>
<td>SUBDIV</td>
</tr>
<tr>
<td>SUBELV</td>
<td>SUBFEM</td>
<td>SUBGLB</td>
<td>SUBGLD</td>
</tr>
<tr>
<td>SUBINC</td>
<td>SUBINT</td>
<td>SUBRES</td>
<td>SUBSIN</td>
</tr>
<tr>
<td>SUBSO1</td>
<td>SUBSO2</td>
<td>SUBSOL</td>
<td>SUBSPC</td>
</tr>
<tr>
<td>SUBST1</td>
<td>SUBSTN</td>
<td>SUBSTS</td>
<td>SUBT</td>
</tr>
<tr>
<td>SUBVAL</td>
<td>SUPER</td>
<td>SYSEQN</td>
<td>T2D04N</td>
</tr>
<tr>
<td>T2D04P</td>
<td>T3D08N</td>
<td>TB02N</td>
<td>TEMPLN</td>
</tr>
<tr>
<td>TFULL2</td>
<td>THRSTN</td>
<td>TIMEIN</td>
<td>TIMER</td>
</tr>
<tr>
<td>TIMOUT</td>
<td>TMULT</td>
<td>TMULTV</td>
<td>TNPSRD</td>
</tr>
<tr>
<td>TRACIN</td>
<td>TRANTN</td>
<td>TRANS1</td>
<td>TRANS2</td>
</tr>
<tr>
<td>TRANS4</td>
<td>TSH04N</td>
<td>TSHIFT</td>
<td>TYIN1</td>
</tr>
<tr>
<td>TYING2</td>
<td>TYING3</td>
<td>TYININ</td>
<td>TYPETN</td>
</tr>
<tr>
<td>UBOUN</td>
<td>UCOEF</td>
<td>UCOOR</td>
<td>UDURIV</td>
</tr>
<tr>
<td>UFORCE</td>
<td>UFXORD</td>
<td>UHOOK</td>
<td>UNITST</td>
</tr>
<tr>
<td>UPRESS</td>
<td>UTX</td>
<td>UPTXL</td>
<td>USXX</td>
</tr>
<tr>
<td>UTEMP</td>
<td>UTERM</td>
<td>VALINT</td>
<td>VDSKIO</td>
</tr>
<tr>
<td>VELCIN</td>
<td>VMULT</td>
<td>VSH04N</td>
<td>VSWEIL</td>
</tr>
<tr>
<td>VMULT</td>
<td>VULVRG</td>
<td>VVMULT</td>
<td>WALCON</td>
</tr>
<tr>
<td>WALKEQ</td>
<td>WKSLP</td>
<td>WORKIN</td>
<td>WRITEX</td>
</tr>
<tr>
<td>YIEL</td>
<td>YIELIN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAY, IPLD, IRSTRT, JCREAD, IPVARS, IPSETS, IFLEX,
COMMON / AUTOIN / CURPER, TOTPER, ARCLEN, ATOLER, BTOLE, CTOLE,
1 JADAP, NCREEP, SCALE
COMMON / BODYFR / POINTS(3, 2)
COMMON / BSECT / ISECT, KSECT
COMMON / CONTROL / JEND, JJITER, JTEMP, JPRINT, JP, JSUB,
1 JINC, JRENT, JSAVE, JREDIN, JAUTO, JPOST,
2 JBACK, JOPTIM, JCREEP, JDIST, JCONST, JDYN,
3 NONISO, ITHERM, ITRIG, IDYNM, JREDOT, JANG,
4 JHERM, JFORCE, JTEMP, JUCOE, JDISTS, JHOCK,
5 JDERIV, JBOUN, IDSTOP, INTSTR, JLPLN, JBDND,
6 JFRONT, JDIFOR, JEMBED, JTEST, JDSP, JIFBFSG,
7 IPSCM, IFLINE, IFPRINT, ICOMPS, IPCONJ, JIEGEN,
8 IFBODY, IFGRAV, IFCENT, JDAMP, IDYN, JSTAT,
9 JFSXX, JSTIF, JCEMCM, JFINIT, JLARGE, JFOLLOW,
+ JWSLP, JFRS, JCDUM2, JCDUM3
COMMON / COMPND / NSTAT, NXSOLV, NXSOLG, NXMODI, NBECKL, NXSUPR,
1 NXR0N, NXNUM1, NXNUM2, NXNUM3, NXNUM4
COMMON / COUNT / LININC, LINTOT, NBECHO
COMMON / CTITLE / TITLE (20), IDAT (5), IDATE2, ICodF,
1 IPCRAY
COMMON / DAMP / DAMPF(3)
COMMON / EIGEN / IECNC, IGNMS, IOMEV, IMENO, IDYNM, JISTR2,
1 IPFAR, IPFBR, IPTVED, IMADF, IOMEG
COMMON / MODSUP / IMPORT, IMDISO, IMVEL0, IMPORT, IMDIS1, IMVEL1
COMMON / ELEMEN / IC, IEL, IDF, JLOW, IPATH, JASEM,
1 JRULE, JCART, JEL009, JEL010, JEL011, JEL012
COMMON / ELTYP / NELCRD, NELNFR, NELNOD, NELSTR, NELCHR, NELPR,
1 NELINT, NELIV, NELLAY, NDI, NSHEAR, NELCMP
COMMON / ERRORS / IERR
COMMON / FREE / IA (80), IBEGIN (16), ILEN (16),
1 NSTRIN, IS, ICOL, NEW
LOGICAL
NEW
COMMON / HARMON / OMEG, IJHARM, KHARM, OMEGB, IBASE, KBASE,
1 ICFOR, ICMFOR, ICMRES, ICSEH, ICBFEN, ICBEXEC,
2 ICCMAT
COMMON / INCON / FACTOR, INCFSA(21)
COMMON / MACHIN / IDF
COMMON / MAXIMA / MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
1 MAXLAY, MAXINT, MAXWRK, MAXNL, NSUMAX, MAXCMP,
2 MAXBP, MAXCMR, MAXTEM, MAXELM, MAXWK, MAXDTM,
3 MAXFUN, MAXSET, MAXVAR, MAXSET, MAXEAM, MAXORD,
4 MAX025, MAX026, MAX027, MAX028, MAX029, MAX030
COMMON / LOUBIN / JLOUB, JITER, JEXTRA, JWEIGE, JSUBRE, JISTRN,
1 JCITR, JHRGNS, JGMAM, LOUB03, LOUB04, LOUB05
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
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 / VRIDS K / ISETUP, MAVAIL, LENREC, NUMREC, LENBLK, NUMBLK,
1 TVPAGE, NVPAGE, IVSTRI, IKRO, ILOL, IPIVCO,
2 IHEDER, IFRNRH, IPIVOT, IPIVRO, IPVKOL, IVEND,
3 ISRECD, IERECD

CHARACTER*20 NAMEIN , NAMEPR , NAMEME , LOGFIL , NAMESC ,
1 NAME08 , NAME09

--- DEFINE THE BUFFER SIZE FOR SCRATCH FILE -----------------------------
MAVAIL = 3 000
NEUFFR = MAVAIL * 4 + 12

OPEN FILES : NAMES READ FROM A SPECIAL FILE 'TSNAMES'
```
C
OPEN( UNIT = 18 , FILE = 'T$FILES' , FORM = 'FORMATTED' )
C
READ( 18 , 1000 ) NAMEIN
READ( 18 , 1000 ) NAMEPR
READ( 18 , 1000 ) NAMEME
READ( 18 , 1000 ) LOGFIL
READ( 18 , 1000 ) NAMESC
READ( 18 , 1000 ) NAME08
READ( 18 , 1000 ) NAME09
C
OPEN( 5 , FILE = NAMEIN )
OPEN( 6 , FILE = NAMEPR )
OPEN( 19 , FILE = NAMEME )
C
OPEN( 8 , FORM = 'UNFORMATTED' ,
1 RECL = 256 , FILE = NAME08 )
OPEN( 9 , FILE = NAME09 )
OPEN( 12 , FORM = 'UNFORMATTED' ,
1 RECL = NBUFFR , FILE = NAMESC )
C
CLOSE( 18 , STATUS = 'DELETE' )
C
******************************************************************************
C
1000 FORMAT( A20 )
C
******************************************************************************
C
RETURN
END

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:

```fortran
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:

```fortran
C=SUBROUTINE= HEAD CALLED BY 'DATIN1'
SUBROUTINE HEAD
1 (VERNO , MONTH , JDATE , IPRNT, 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**

Page: 13

**MHOST Version 4.2**
**IMPLICIT REAL*8 ( A-H, O-Z )**

**COMMON / CTITLE / TITLE ( 20), IDAT ( 5), IDATE2, ICLOCK, IFCRAY**

**CALL CLOCK ( ICLOCK )**
**CALL DATE ( IDATE2 )**

**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)**

*** THE FOLLOWING STATEMENT MAY CAUSE A WARNING MESSAGE WHEN EXECUTED ***
*** ON IBM CMS WITH 'RUNHOST' PROCEDURE FILE. ***

**WRITE (ICONSL,1100) IDATE2, ICLOCK, Verno, MONTH, JDATE, TITLE**
**WRITE (ICONSL,1002)**

**SYSTEMS' MANUAL**
**Page : 14**
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.

```fortran
SUBROUTINE TIME(CU)
C * * * * *
C CPTIME RUNTIME FOR PRIME
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:

```fortran
C=SUBROUTINE=TIMDUT UTILITY SUBROUTINE
SUBROUTINE TIMDUT(IA1,IA2,IA3,IA4,IA5,IA6)
C * * * * *
C OBTAINS AND PRINTS CPU TIME
C * * * * *
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION NAME(6)
```

SYSTm_S' MANUAL MHOST Version 4.2
Page : 15
C *** GET 'DELTA' VALUES - CALL TO CARY LIBRARY SUBROUTINE
C
C COMMON / ALGEM / ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1   IPLQT, IRSTR, PI, LINE, LINE2
C COMMON / CONTRO / JEND, JITER, JTEMP, JPRINT, JP, JSUB,
1   JINC, JREST, JSAVE, JREDIM, JAUTO, JPOST,
2   JBACK, JOPTIM, JCREEP, JRES, JCONST, JDIFF,
3   NONISO, ITERM, JITRIG, JIDYN, JREPOT, JTANGE,
4   JTERM, JFORCE, JUTEMP, JUCOEF, JPRESS, JUHOOK,
5   JDERIV, JUBOUN, IDSTOP, INISTR, JPLAST, JBAND,
6   JFRONT, JDEFOR, JEMBED, JTEST, JDISP, JFBFGS,
7   JFRONT, JDEFOR, JEMBED, JTEST, JDISP, JFBFGS,
C
COMMON / TIMLOC / CPU0
C
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) = IAA
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,6A4,9HINCREM_,I4,10H ITERATION, I3,8H CPTIME=,F9.3, 1 4H SEC)
1010 FORMAT(2X,6A4,9HINCREM_,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 ,JPOST ,
   2 JBACK ,JOPTIM,JCREEP,JDIST ,JCONST,JDYN ,
   3 NONISO ,ITERM ,ITRIG ,IDYN ,JREPORT,JITANGE ,
   4 JTERM ,JFORCE ,JUTEMP,JOCEQ,JDIST ,JHOOK ,
   5 JDERIV,JUBOUN,IDSTOP,IMSTR,JPRAST,JBAND ,
   6 JFRONT,JDEFOR,JUTEMP,JTERM,JOUTE,JDIST ,JUHOO ,
   7 IPSCMD,IFLINE,IFPRINT,ICOMP,IPCON,JEIGEN ,
   8 IFBODY,IFGRAV,IFCENT,JDAMP,LDYN ,ISTAT ,
   9 JFRONT,JINST,JCNST,JFINIT,JLARGE,JFOLLOW ,
+ JWKSPL,IPRES ,JCDUM2,JCDUM3
COMMON / ALGEM / ICREAD,ILPRNT,JLPRNT,ICONSL,IPFBGS,IPOL,IPR,
   1 IPLOTB,IPRTB,ICREAD,IPVARS,IPSETS,IPFILE ,
   2 PI ,LINE ,LINE2
COMMON / ADDVAL / ISPRI ,ISPRI ,IDASH ,KDASH ,IMASS ,KMASS
COMMON / TMARCH / DALPHA,DBETA ,DGAMMA
COMMON / AUTOIN / CURPER,TOIPER,ARCLEN,ATOI,ATOR,BTOL,CTOL,
COMMON / PERIOD / JPERIOD(2), IPDISP, IPFORC, INDISP, INFORC
COMMON / POWER / IEIPHI, IEITNM, IEPSMO, ISIGMD, IHFN, IHFC,
  1 IFBP, ISPP, ISFF, ISQQ, ICQQ, ITTNM,
  2 IPSF, IPSD
COMMON / PULSES / IPULSE, IPULS, IPDTIM, IPDFOR
COMMON / EIGEN / IEIGMVC, IGNS, IOMEG, IOMENO, IDYNMD, ISTR12,
  1 IPTAR, IPTBR, IPIVDE, IMDAM, IOMEGD
COMMON / ERRORS / IERR
COMMON / MODSUP / IMFOR0, IMDIS0, IMVELO, IMFOR1, IMDIS1, IMVE1
COMMON / HARMON / OMEGH, KHARM, OMGB, IBASE, KBASE,
  1 ICNFOR, ICMFOR, ICMRES, ICHHFN, ICBHFN, ICBEXEC,
  2 ICMMAT
COMMON / LOUBIN / JLOUB, JINTER, JEXTRA, JWEIGH, JSUBRE, JISTRN,
  1 JCITER, JHRGLS, JGRAM, LOUB03, LOUB04, LOUB05
COMMON / START5 / IRL, IREAC, IES, IAEP, IBQM, ISRL,
  1 IBMLC, ISK, ILAST, IRLB, IDINC, IFORIN,
  2 IOP, IDAM, IDMSMT, IDIAG, IUPTR, ICOLPT,
  3 IDASDI, IMASUP, IST521, IST522, IST523, IST524
COMMON / START4 / IDINC, IDIOT, IPFORCE, IRESID, IWINOD, ISIGNO,
  1 IEPSNO, IPSTRN, ICSTRN, ISTRN, ISTRS, IISTRN,
  2 IIPSTR, IICSTR, IITSTR, IPSTNO, ICSTNO, ISTSTNO,
  3 IISNO, IISNNO, IIPSNO, IICSNO, IITSNO, IDMAT,
  4 IMENO, IEQCSI, IOMENO, IOMNNO, ITDSNO, IVTDS0,
  5 IDYNV, IDYN, IDSN, ICDS2, IISTR, ISWELL,
  6 IEQCSI, IPREF, IDSS, IYIELD, IDPFNC, IDPTOT,
  7 IST443, IST444, IST445, IST446, IST447, IST448
COMMON / PARAM / NTYPE, NELEM, NNODE, NBC, NTIE, NMAX,
  1 NTRAN, NTRAC, NFD, NBAND, NEXT, NSUB,
  2 NPRINT, NPOST, NSBC, NDUP, NSIZE, NBSEC,
  3 NSHIFT, NSEF, NMR, NSPR, NNS, NDASH,
  4 NDYNMD, NSBNC, NSUP, NHARB, NBSENC, NINC,
  5 NITER, NFPST, NDPTS, NPULSE, NPDPTS, NHARD,
  6 NSUPCH, NDIMEN, NMNOT, NP40, NP41, NP42,
  7 NP43, NP44, NP45, NP46, NP47, NP48
COMMON / BSECT / IBSECT, JBSECT
COMMON / INCON / FACTOR, INFLG(21)
COMMON / MACHIN / IDP
COMMON / SUBSTR / NLVSTJB(10), NFRR(10)
COMMON / SUBELM / ISUBEL, ISUBET, ISUBPT, NSD, ISUBTY, IEMBED
COMMON / SHIFT / ISHIFT, KSHIFT, IFREQ, LFREQ, NOFFST, NFOUND
DATA / IZ / 1H /
DATA / MAXSUB / 10 /
WRITE(ICONSL, 2000) IZ
2000 FORMAT(A1)
CALL TIMDUT( ' BE', 'GIN ', 'EXEC', 'UTIO', 'N', ' ', )

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

CALL DATINI
1 (RWORK , IWORK , ISIZE , VERSNO , MONTH , JDATE , NELEM , NNODE ,
2 NBC , NTIE , NMAX , NTRAN , NTRAC , NPOST , NLVSB , NFRSUB ,
3 NEXT , JDYN , JTEMP , NPRINT , JREST , JINC , NINC , JLOUB ,
4 JINTER , JEXTRA , JWEIGH , NSTBC , NTYPE , NMAXSUB , ILAST , JSUB ,
5 NSUB , ISTAT , IDYNM , ITEST , JOPTIM , JCREEP , JDIST , NONISO ,
6 NDTMD , IDYNMD , NPOSMD , THERM , JCONST , NUP , JREPOT , JTANGE ,
7 JTEMP , DPCALPHA , DDETA , DCGAMMA , JMEGEN , JFORCE , JUTEMP , JUCOEF ,
8 JDISTS , JUHOCK , JDERIV , JUBOUN , JPEROD , NSBNC , JCREEP , ATOLER ,
9 BTOLE , CTOLER , JPOST , INTSTR , JBAND , JFRONT , JDEFOR , NMR ,
+ JEMBED , NBSECT , JDSIP , NSHIFT , NSUPER , JSUBRE , IFBFGS , NSPRI ,
1 NDASH , NMASS , NSBFGS , IFSCNT , IFLINE , IFPRNT , NHCFL , NMEG ,
2 NBASE , NMEG , ICMP , NFDP , NPULSE , IPCON , NSSPTS , NSHOCK ,
3 NSPTS , NFDP , LDYN , JFPSXX , JSTIF , JCONS , JHARD , JFINT ,
4 JLARGE , JFROW , JWKL , JSTRN , JCITER , JHGLS , NDIM , JGRAM ,
5 JPERD , JMONIT )

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

CALL LETCMD
1 (NCOMPD , IFSTAT , IFSOLV , IFINTG , IFMDDL , IFSUPR , IFREQN ,
2 IFDUM , IFDUM2 , IFDUM3 , IFDUM4 , IERR )

*** BULK FINITE ELEMENT DATA INPUT ******************************************************

IF ( JREST ) IS SET ZERO, START FROM SCRATCH AND THE DATA IS READ
FROM THE MAIN CARD READER CHANNEL
IF ( JREST ) IS NON-ZERO, THE RESTART FILE IS RECOVERED FROM THE
RESTART FILE

FLAG 'IPFLG' IS USED FOR CONTROLLING THE PRINTOUT FOR THE INITIAL
VALUES FOR TRANSIENT DYNAMICS

IPFLG = 0 .... ONLY INITIAL CONDITION IS PRINTED
1 .... ONLY THE REST OF THE MODEL DEFINITION DATA ARE PRINTED
2 .... ALL OF THE INFORMATION IS PRINTED
IPFLG = 1
IF( IDYNM .EQ. 1 .AND. JDYN .EQ. 2 ) IPFLG = 2

C **********************************************************************
C
C DO 5000 ICOMPD = 1 ,NCOMPD
C
CALL RUNCMD
1   (ICOMPD,IFSTAT,IFSOLV,IFINIG,IFMODL,IFBCKL,IFSUPR,IFREQN,
     2   IFDUM1,IFDUM2,IFDUM3,IFDUM4,IERR )
C
C **********************************************************************
C
ELSE IF( IFSTAT .EQ. 1 ) THEN
    CALL STATIC( RWORK, IWORK, ISIZE )
C
ELSE IF( IFSOLV .EQ. 1 ) THEN
    CALL FRONTS( RWORK, IWORK, ISIZE )
C
ELSE IF( IFMODL .EQ. 1 ) THEN
    CALL MODAL( RWORK, IWORK, ISIZE )
C
ELSE IF( IFSUPR .EQ. 1 ) THEN
    CALL SUPER( RWORK, IWORK, ISIZE )
C
ELSE IF( IFBCKL .EQ. 1 ) THEN
    CALL BUCKLE( RWORK, IWORK, ISIZE )
C
ELSE IF( IFREQN .EQ. 1 ) THEN
    CALL FREDOM( RWORK, IWORK, ISIZE )
C
ELSE IF( IFINIG .EQ. 1 ) THEN
    CALL DYNAMT( RWORK, IWORK, ISIZE )
C
C **********************************************************************
C
SYSTEMS' MANUAL

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 subroutine 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 INIT1 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 DATIN3, a small subset of the bulk data reader SUBROUTINE DATINI 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 COMPRO - Sets up the integer array for the profile of global stiffness equations to be stored in a profile form.

SUBROUTINE ASSEM5 - Assembles the element stiffness equations into the global equation system stored in a profile form.

SUBROUTINE SOLUT1 - 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 INTFR - Allocates memory for the work space required for the frontal solution.

SUBROUTINE PRFRNT - Sets up the elimination table for the frontal solution.
SUBROUTINE FRONTF - 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 INMDP - 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 ASSEMN - 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 ASSEMM - 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:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDVAL</td>
<td>Variables related to additional point masses, dashpots and springs.</td>
</tr>
<tr>
<td>ALGEM</td>
<td>Fortran unit numbers and record number counters. Also the value of p is stored in this common block.</td>
</tr>
<tr>
<td>AUTOIN</td>
<td>Real variables used for automatic load increments. Also control variables for adaptive load increments by the arc length method are stored here.</td>
</tr>
<tr>
<td>BODYFR</td>
<td>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.</td>
</tr>
<tr>
<td>BSECT</td>
<td>Pointers and counters for beam section definitions.</td>
</tr>
<tr>
<td>CONTRO</td>
<td>Control variables and binary switches to activate options built into the MHOST program.</td>
</tr>
<tr>
<td>COMPNDS</td>
<td>Flags for sequential execution of multiple analysis drivers.</td>
</tr>
</tbody>
</table>
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  An 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>IRSTRT</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 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>JDIST</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for the distributed load option.</td>
</tr>
<tr>
<td>JUONST</td>
<td>CONTRO</td>
<td>0</td>
<td>Selector for the constitutive equation =</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>N_ISO</td>
<td>CONTRO</td>
<td>0</td>
<td>Flag for temperature dependent material response.</td>
</tr>
<tr>
<td>ITHERM</td>
<td>CONTRO</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>NXSTAT</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>NXSGLV</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for modal analysis for the second step.</td>
</tr>
<tr>
<td>NXINYG</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for buckling analysis for the second step.</td>
</tr>
<tr>
<td>NXMODL</td>
<td>COMPND</td>
<td>0</td>
<td>Flag for modal super-position for the second step of the analysis.</td>
</tr>
</tbody>
</table>

0 for linear elasticity
1 for secant elasticity/simplified plasticity
2 plasticity
3 unified creep plasticity
<table>
<thead>
<tr>
<th>Variable</th>
<th>Common Block</th>
<th>Initial Value</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<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>Variable Name</td>
<td>Common Block</td>
<td>Initial Value</td>
<td>Content</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</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>NELLV</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 I_3amina 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>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 INITT2.</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 LOBIN 1
Selector for the integration rule used in the residual force calculation.

= 1
= 2
Full Gaussian quadrature
One point reduced integration

JWEIGH LOBIN 3
Selector for the integration rule used to generate the stiffness equations.

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

= 3
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>
</tbody>
</table>
| NBC           | PARAM        | 0             | 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
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.

2.0 DATA STORAGE SCHEME

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 INITI 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 (IWOR x, RWOR x, IERR, NTYPE, ILAST, ILPRNT, NDIMEN)
C
C **********************************************************************
C C RENGINE FOR OBTAINING INPUT FOR DIFFERENT TYPES OF ELEMENTS
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 RWOR x
C
C **********************************************************************
C
DIMENSION IWOR x ( 1 ) , RWOR x ( 1 )
DIMENSION ISTORE( 13 , 16 )
PARAMETERS TO CHARACTERIZE ELEMENTS AVAILABLE IN THE HOST PROGRAM

<table>
<thead>
<tr>
<th>NO.</th>
<th>TYPE</th>
<th>NCRD</th>
<th>NDOF</th>
<th>NODE</th>
<th>NSTR</th>
<th>NCHR</th>
<th>NINT</th>
<th>NLLV</th>
<th>NLAY</th>
<th>NCMP</th>
<th>NDI</th>
<th>NSHR</th>
<th>JLAW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>6</td>
<td>6</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>75</td>
<td>7</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>72</td>
<td>7</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>98</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>6</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>101</td>
<td>2</td>
<td>2</td>
<td>9</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>102</td>
<td>2</td>
<td>2</td>
<td>9</td>
<td>4</td>
<td>6</td>
<td>9</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>103</td>
<td>2</td>
<td>2</td>
<td>9</td>
<td>4</td>
<td>6</td>
<td>9</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>104</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>27</td>
<td>6</td>
<td>6</td>
<td>27</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>151</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>152</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>153</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>154</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>6</td>
<td>6</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

**FREE FORMAT READER FOR ELEMENT TYPE DEFINITION**

101 CONTINUE
CALL FPREFOR(IWORK(ILAST+1),IWORK(ILAST+1),13,0,0,IERR,JKEY)

IF( JKEY .EQ. 1 ) GO TO 501
IF( NTYPE .EQ. 0 ) GO TO 301

DO 201 J = 1 , NTYPE

K = ILAST + 1 - 13*J
IF(IWORK(ILAST + 1) .NE. IWORK(K)) GO TO 201
CALL COPYIN(IWORK(ILAST + 1),IWORK(K), 13 )
LL = IWORK(K)

GO TO 401

201 CONTINUE
301 CONTINUE

DO 351 LL = 1 , NTYPES

IF( ISTORE(1,LL).EQ.IWORK(ILAST + 1)) THEN
   CALL COPYIN(ISTORE(1,LL),IWORK(ILAST + 1), 13 )
   IF( ISTORE(13,LL).LE.4 ) NDIMEN = 2
ENDIF

22 CONTINUE

NTYPE = NTYPE + 1
ILAST = ILAST + 13

C 351 CONTINUE
C 401 CONTINUE
501 CONTINUE

GO TO 101

RETURN
END

As seen in the following example, the element data copied by the above subprogram is accessed every time the element loop is incremented:

CALL NUL(REA(IMASNO),NNODE)
CALL NUL(REA(IGSINO),NNODE*NELSTR)
CALL NUL(REA(IGEPNO),NNODE*NELSTR)

CALL NUL(REA(IEQPSI) , NELLAY*NEI/gOD)
CALL NUL(REA(IIPSNO) , NELCMP*NELIAY*NELNOD)

CALL NUL( REA(IIEQPSI) , NELLAY*NEINOD )
CALL NUL( REA(IIPSN0) , NELCMP*NELLAY*NEINOD )
C >> START THE FIRST ELEMENT LOOP >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

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(I1) , CTRANS , IELCNT , IC , -1 , -1 )
C
CALL LMPMAS
1 ( REA(IMASNO), REA(IDET) , INT(I1) , NELNOD , NELINT , 
2 NNODE , IC , REA(ICH) , REA(ICOR) , NELCRD , 
3 IELCNT , NELEM , REA(INOD) , MAXCRD , CTRANS , 
4 JGRAM )
C
IF ( JRES .GT. 0 ) CALL BOUNDN
1 ( REA(IBNORM), REA(ICOR ) , INT(I1 ) , IELCNT , IC , 
2 NNODE , NELNOD , NELCRD )
C
5500 CONTINUE
C
C *** CLOSE THE FIRST ELEMENT LOOP <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
C
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 ************************************************* **********************************************
C
C CONTINUE
C
C 8500
C
SUBROUTINE ELVULV as shown below accesses the entries defined by SUBROUTINE TYPEIN:
C
C=SUBROUTINE=ELVULV CALLED BY SUBROUTINES 'ASSEM1', 'ASSEM4'
SUBROUTINE ELVULV
1 ( IWORK , IC1 , IERR )
C
C ************************************************* **********************************************
C
C EXTRACTS INFORMATION ABOUT A CERTAIN ELEMENT TYPE
C
C IC1 ELEMENT TYPE

SYSTEMS' MANUAL    MHOST Version 4.2
Page : 43
C IERR ERROR FLAG

C **********************************************************************
C
C IMPLICIT REAL*8 ( A-H , O-Z )
REAL*4 R_3RK
C
C **********************************************************************
C
DIMENSION IWORK(1)
C
C **********************************************************************
C
COMMON / ALGEM / IREAD,ILPRNT,JLPRNT,ICONS,N,JPOSTF,ICSCRAF,
1 IPLTRB,ISTRT,JCREAD,IPVARS,IPSETS,IIFILE,
2 PI ,LINE ,LINE2
COMMON / CONTRO / JEND ,JITER ,JTEMP ,JPRINT,JP ,JSUB ,
1 JINC ,JREST ,JSAVE ,JREDIM,JAUTO ,JPOST ,
2 JBACK ,JOPTIM,JCREEP,JDIST ,JCONST,JMIN ,
3 NONISO,ITHERM,ITRIG ,IDYN ,JREPO,TJANGE ,
4 JTHEM,JFORCE,JUTEMP,JUCOE,JDIST ,JUNI ,
5 JDERIV,JUBOUND,IDSTOP,INTSTR ,JPLAST,JBAND ,
6 JFRONT,JDEFOR,JEZVED,ITEST ,JDISP ,IFBFGS ,
7 IFSCNT,IFLINE ,IPFRNT ,ICOMPS ,IPCONJ,JEIGEN ,
8 IFBODY,IFGRAV,IFCENT ,JDAMP ,LDYN ,ISTAT ,
9 JFDXX ,JSTIF,JCENIM,JKINIT,JKERL ,JPOLO ,
+ JWKSLP,JPRES ,JCDUM2,JCDUM3
COMMON / ELTYP / NELCRD,NELNFR,NELNOD,NELSTR,NELCHR,NELPR ,
1 NELINT,NELNV ,NELLAY ,NDI ,NSHEAR,NELCMP ,
COMMON / START1 / IELEM,INEL ,INCHAR ,IPRES ,ISTRS ,
1 ISTRN ,ICOOP ,PRINT,PPOST ,JDIST ,ILEAN ,
2 IPRES,IBNORM,IMONIT,IST116 ,IST117,IST118
COMMON / PARAM / NTYPE ,NELEM ,NNODE ,NBC ,NTIE ,NMAX ,
1 NTRAN ,NTRAC ,NFD ,NBAND ,NEXT ,NSUB ,
2 NPINT,NPOST ,NSBC ,NDUP ,NSIZE ,NBSEC ,
3 NSHIFT,NBSFCS,NMPS ,NSPRI ,NMASS ,NDASH ,
4 NDYNMD ,NSBNC ,NSUPER,NHARM ,NBASE ,NINC ,
5 NITER ,NPSPTS,NPDPTS,NPULSE,NPDPTS,NHARD ,
6 NSUMCH,NDIMEN,NMONT,NDP40 ,NP41 ,NP42 ,
7 NPAR43,NPAR44,NPAR45 ,NPAR46 ,NPAR47 ,NPAR48
COMMON / ELEMEN / IC ,IEL ,IDF ,JLAW ,IPATH ,IASSEM ,
1 JRULE ,IJCAJ ,JEL009 ,JEL010 ,JEL011 ,JEL012

C
C **********************************************************************
C
SYSTEMS' MANUAL  MHOST Version 4.2
Page : 44
DO 5000 II = 1 , NTYPE

C
I1 = 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
IERR = IERR+1

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)
NELAY = IWORK(IS1+7)
NELCMP = IWORK(IS1+8)
NDI = IWORK(IS1+9)
NSHEAR = IWORK(IS1+10)
JLAW = IWORK(IS1+11)
NELV = NELNFR*NELNOD
IDF = NELNFR*NELNOD

C

C
SPECIAL TREATMENT FOR THE NINE POINT INTEGRATION OF THE QUADRATIC
LAGRANGIAN ELEMENTS ( ONLY IN CASE OF MATRIX ASSEMBLY )

C
IF ( IASSEM .EQ. 0 .AND. NELINT .EQ. 9 ) NELINT = 4

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:

```fortran
SUBROUTINE CNODEL
  REAL*4 REA
  DIMENSION REA(1),INT(1),ISLV(1),CTRANS(3,3)
  DIMENSION VTRANS(2,2),TTRANS(3,3),ETRANS(3,3)

  IMPLICIT REAL*8 (A-H,O-Z)
  REAL*4 REAL*4
  DIMENSION REA(1),INT(1),ISLV(1),CTRANS(3,3)
  DIMENSION VTRANS(2,2),TTRANS(3,3),ETRANS(3,3)

  COMMON / TMARCH / DALPHA, DBETA ,DGamma
  COMMON / ALGEM / ICREAD, ILPRNT, JLPRT, ICONSL, IPOSTF, ISCRAF,
      IPI, LINE , LINE2
  COMMON / CONTRO / JEND , JITER , JTEMP , JPRINT, JP , JSUB ,
```

SYSTEMS' MANUAL  MHOST Version 4.2
Page : 46
COMMON / ELTYP / NELCD, NELFR, NELNO, NELST, NELCH, NELPR,
1 NELINT, NELLV, NELLAY, NDI, NSHEAR, NELCMP,
2 COMMON / MAXIMA / MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
1 MAXLAY, MAXINT, MAXVRK, MAXNLV, NSUMAX, MAXCMP,
2 MAXBSP, MAXGMR, MAXTEM, MAXELM, MAXMT,
3 MAXFRN, MAXBET, MAXVAR, MAXSET, MAXEAN, MAXORD,
4 MAX025, MAX026, MAX027, MAX028, MAX029, MAX030
1 COMMON / PARAM / NTYPE, NELEM, NNODE, NBC, NTIE, NMAX,
1 NTRAN, NTRAC, NFP, NBAND, NEXT, NSUB,
2 NPRINT, NPOST, NSBC, NDUP, NSIZE, NSDEC,
3 NSHIFT, NSMBGS, NSMPR, NSPRI, NDASH, NINC,
4 NDYNMD, NSBNC, NSUPER, NHARM, NSUB, NHC,
5 NTIER, NPSPTS, NPPRT, NSBFGS, NSPRI, NSBNC, NSUPER,
6 NSMCC, NDM2, NDM3, NSUMAX, MAXCMP, MAXLWK, MAXDMP,
7 COMMON / START1 / IELPRT, IYT, INEL, ICHAR, IPRES, ISTRS,
1 ISTRN, ICOP, IPRINT, IPOST, IDIST, ILEAN,
2 COMMON / START2 / INOD, ITEM, INLV, IPOSU, ITEMP, IDUP,
1 COMMON / START4 / IDINC, IDTOT, IFORCE, IRESID, IWNOO,
1 COMMON / START5 / IRL, IREAC, IES, IAB, IDCH, IBQM,
1 COMMON / START6 / IEVL, ICOR, ISIG, IEPS, IPOSU, ISNOD,
1
NELST2 = NELSTR * NELSTR
MAXST2 = MAXSTR * MAXSTR
IPP = IPRES + (IEL-1) * MAXPRS * IDP
MAXDIM = MAXCMP * MAXLAY
NELDIM = NELCMP * NELLAY

CALL SEARCH
1 (INT(IELV), 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(IXRL), REA(IDTOT), INT(IELV), 1, NFD, 1, NELLV, 1)
CALL SEARCH
1 (REA(IXIRL), REA(IDINC), INT(IELV), 1, NFD, 1, NELLV, 1)
CALL SEARCH
1 (REA(KDIMNO), REA(IDMNO), ISLV, MAXST2, NNODE, 1, NELNOD, NELST2)
CALL SEARCH
1 (REA(KGEPNO), REA(IGEPNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KINEGNO), REA(IISGNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KIMENO), REA(ISWNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
CALL SEARCH
1 (REA(KIDN3NO), REA(IIDN3NO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)
1 (REA(KIDSNO), REA(ITDSNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)  
CALL SEARCH  
1 (REA(KEQPSI), REA(IEQPSI), ISLV, 1, NNODE, 1, NELNOD, 1)  
CALL SEARCH  
1 (REA(KEQPSI), REA(IEQPSI), ISLV, 1, NNODE, 1, NELNOD, 1)  
CALL SEARCH  
1 (REA(KIPSNO), REA(IIPSNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)  
CALL SEARCH  
1 (REA(KICSNO), REA(IICSNO), 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, 1, NNODE, 1, NELNOD, 1)  
CALL SEARCH  
1 (REA(KIQMNO), REA(IIQMNO), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)  
CALL SEARCH  
1 (REA(KSWFNO), REA(ISWELL), ISLV, 1, NNODE, 1, NELNOD, 1)  
CALL SEARCH  
1 (REA(KTDFNO), REA(ITEMDF), ISLV, 1, NNODE, 1, NELNOD, NELSTR)  
CALL SEARCH  
1 (REA(KIOMNO), REA(IICMND), ISLV, MAXSTR, NNODE, 1, NELNOD, NELSTR)  
CALL SEARCH  
1 (REA(KYIELD), REA(IYIELD), ISLV, N3HARD, NNODE, 1, NELNOD, N3HARD)  
CALL INTERP  
1 (REA(ICH), REA(KISTRS), 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(KISTRS), 1, NELNOD, NELCHR)  
CALL SEARCH  
D-MATRIX ROTATION FOR ANISOTROPIC MATERIALS  
IF (NONISO .EQ. 1) CALL ROTMT  
1 (REA(KDOMNO), REA(IPREF), ISLV, NELNOD, NELSTR, NNODE, NDI, NSHEAR)  
CALL SEARCH  
*** DYNAMIC CALCULATIONS: QUANTITIES ASSOCIATED WITH NODAL TIME DERIVATIVES  
IF (IDYMN .EQ. 1 .AND. JDYN .EQ. 1) THEN  
CALL SEARCH  
1 (REA(IAELM), REA(IDYNA), INT(IELV), 1, NFD, 1, NELLV, 1)  
CALL SEARCH  
1 (REA(IVELM), REA(IDYNV), INT(IELV), 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, _S, REA( ICOR), I, *
* MAXCRD, NELNOD, SIGN, IFLAG)

II= NELSTR * IDP *(I - 1)

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(KIGSNO+I1), VTRANS, TTRANS, SIGN, NELSTR)
CALL SHTRAN( REA(KGTDNO+I1), VTRANS, TTRANS, SIGN, NELSTR)
CALL SHTRAN( REA(KIGSNO+I1), 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.EQ.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

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

SYSTEMS' MANUAL

Page : 51

MHOST Version 4.2
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(IYRL),NDIMEN*NELNOD)

IF( ISTEP.EQ.0 ) THEN

CALL ADDSMU( REA(ICOR),ONE,REA(ICOR),HALF,REA(IYRL),
            &     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(IYRL),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
IIXRL = IXRL + (I-1)*MAXNFR*IDP
IIXIRL = IXIRL + (I-1)*MAXNFR*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 )

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

I1 = NELSTR * IDP *(I - 1)

CALL SHTRAN(REA(KGEPhK3+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(KIGSNO+I1), VTRANS, TTRANS, SIGN, NELSTR)
CALL SHTRAN(REA(K_IDNO+I1), VTRANS, TTRANS, SIGN, NELSTR)
CALL SHTRAN(REA(KTDShD+I1), VTRANS, TTRANS, SIGN, NELSTR)

DO 15000 K = 1, MAXSTR
   J1= NELSTR * IDP * (NELSTR *(I - 1)+(K - 1))
   CALL SHTRAN(REA(KDMINO+J1), VTRANS, TTRANS, SIGN, NELSTR)
CONTINUE

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

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

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

ELSE IF( IC.EQ.98 ) THEN

UPDATE BEAM TRANSFORMATION

END IF

ELSE IF( NDIMEN.GT.MAXNFR ) THEN

SOMETHING IS WRONG!!!!!!!!!!

SYSTEMS' MANUAL

Page : 54
CALL QUIT( 'NO.', 'DIME', 'NS', 'GT.', 'NO.', 'DOF', 0 )
ENDIF
ENDIF

RETURN
END

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:

CALL SUBROUTINE 'BKWd_' SUBROUTINE INITII
1 (WORK, WORK, ISIZE )

** ALLOCATES CORE FOR DATA INPUT AND ZEREOES OUT CORE **

** 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
IBTLC /START5/ 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
ICONFOR /HARMON/ POINTER TO THE COMPLEX MODAL FORCE VECTOR
ICOLPT /START5/ POINTER TO ARRAY DEFINING THE LOCATIONS OF THE
COLUMN ELEMENTS THAT ARE JUST ABOVE THE DIAGONAL
ICON /START7/ ELEMENT CONNECTIVITY ARRAY
ICOI /START1/
ICOR /START6/   NODAL COORDINATE ARRAY
ICSTNO /START4/   NODAL CREEP STRAIN
IDASH /ADVAL/   POINTER TO THE ADDED DASHPOT DAMPING VALUES
IDET /START7/   DETERMINANTS ARRAY AT INTEGRATION POINTS
IDFINC /START4/   INCREMENTAL DEFORMATION GRADIENT ARRAY
IDFTOT /START4/   TOTAL DEFORMATION GRADIENT ARRAY
IDIG /START5/   POINTER TO DIAGONAL COMPONENTS OF GLOBAL STIFFNESS ARRAY STORED IN PROFILE FORM
IDINC /START4/   INCREMENTAL DISPLACEMENT ARRAY
IDINCP /START5/   ITERATION VECTOR FOR INCREMENTAL DISPLACEMENT
IDIST /START1/   DISTRIBUTED LOAD ARRAY
IDMINO /START4/   NODAL DMATRIX ARRAY ALLOCATION
IDP /MACHIN/   WORD LENGTH ( 1 REAL WORD = IDP INTEGER WORD )
IDSITR /START4/   DISTRIBUTED LOAD INPUT ARRAY
IDSX1 /START4/   WORKING ARRAY FOR DYNAMICS
IDSX2 /START4/   WORKING ARRAY FOR DYNAMICS
IDTOT /START4/   TOTAL DISPLACEMENT VECTOR
IDUP /START2/   Duplicated node connectivity
IDYNA /START4/   GLOBAL ACCELERATION ARRAY
IEQCS /START4/   INCREMENTAL NODAL EQUIVALENT PLASTIC STRAIN BY LAYER
IEQPSI /START6/   TOTAL NODAL EQUIVALENT PLASTIC STRAIN BY LAYER
IEQPSI /START6/   TOTAL NODAL EQUIVALENT PLASTIC STRAIN BY LAYER
IETM /START6/   TOTAL NODAL EQUIVALENT PLASTIC STRAIN BY LAYER
IEXT /START6/   TOTAL NODAL EQUIVALENT PLASTIC STRAIN BY LAYER
IFORCE /START4/   DISTRIBUTED LOAD INPUT ARRAY
IFORIN /START5/   DISTRIBUTED LOAD INPUT ARRAY
IFREQ /SHIFT/   HIGH FREQUENCY BOUNDS FOR POWER SHIFT
IGEPNO /START8/   TOTAL GENERALIZED NODAL STRAIN COMPONENTS
IGNMS /EIGEN/   TOTAL GENERALIZED NODAL STRESS COMPONENTS
IGSINO /START8/   TOTAL GENERALIZED NODAL STRESS COMPONENTS
IGTINO /START8/   TOTAL GENERALIZED NODAL STRESS COMPONENTS
IHARM /HARMON/   HARMONIC NODAL FORCE MAGNITUDE AND PHASE
IICSNO /START4/   INCREMENTAL GENERALIZED NODAL STRAIN COMPONENTS
IIGENO /START8/   INCREMENTAL GENERALIZED NODAL STRESS COMPONENTS
IIGSNO /START8/   INCREMENTAL GENERALIZED NODAL STRESS COMPONENTS
IIMNO /START4/   INCREMENTAL NODAL BACKSTRESS (SHIFT TENSOR) COMPONENTS BY LAYER
IIPSNO /START4/   INCREMENTAL NODAL PLASTIC STRAIN COMPONENTS BY LAYER
IISNNO /START4/   INCREMENTAL NODAL STRAIN COMPONENTS FOR SHELL ELEMENTS BY LAYER
IISTNO /START4/   INCREMENTAL NODAL STRESS COMPONENTS
FOR SHELL ELEMENTS BY LAYER
INCREMENTS 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
NODEL 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 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
POINTER TO UPPER TRIANGULAR PART OF GLOBAL
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
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KIGSNO</td>
<td>/START8/</td>
</tr>
<tr>
<td>KIPSMO</td>
<td>/START16/</td>
</tr>
<tr>
<td>KIFSTR</td>
<td>/START6/</td>
</tr>
<tr>
<td>KISNNO</td>
<td>/START6/</td>
</tr>
<tr>
<td>KISTNO</td>
<td>/START6/</td>
</tr>
<tr>
<td>KISTRN</td>
<td>/START6/</td>
</tr>
<tr>
<td>KISTRS</td>
<td>/START6/</td>
</tr>
<tr>
<td>KITSMO</td>
<td>/START6/</td>
</tr>
<tr>
<td>KITSTR</td>
<td>/START6/</td>
</tr>
<tr>
<td>KNASS</td>
<td>/ADDVAL/</td>
</tr>
<tr>
<td>KPSTNO</td>
<td>/START6/</td>
</tr>
<tr>
<td>KPSRSTRN</td>
<td>/START6/</td>
</tr>
<tr>
<td>KSFILT</td>
<td>/SHIFT/</td>
</tr>
<tr>
<td>KSPRI</td>
<td>/ADDVAL/</td>
</tr>
<tr>
<td>KIDSMN</td>
<td>/START6/</td>
</tr>
<tr>
<td>KITSTNO</td>
<td>/START6/</td>
</tr>
<tr>
<td>KITSTRN</td>
<td>/START6/</td>
</tr>
<tr>
<td>LFREQ</td>
<td>/SHIFT/</td>
</tr>
<tr>
<td>MAXBET</td>
<td>STACK</td>
</tr>
<tr>
<td>MAXBSP</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXCHS</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXCMF</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXCRD</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXDMT</td>
<td>STACK</td>
</tr>
<tr>
<td>MAXINT</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXLAY</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXNFR</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXNLV</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXNOD</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXPRS</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXSTR</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>MAXWRK</td>
<td>/MAXIMA/</td>
</tr>
<tr>
<td>NBASE</td>
<td>/PARAM/</td>
</tr>
<tr>
<td>NBC</td>
<td>/PARAM/</td>
</tr>
<tr>
<td>NBSECT</td>
<td>/PARAM/</td>
</tr>
<tr>
<td>NDASH</td>
<td>/PARAM/</td>
</tr>
<tr>
<td>NDUP</td>
<td>/PARAM/</td>
</tr>
<tr>
<td>NELEM</td>
<td>/PARAM/</td>
</tr>
<tr>
<td>NEXT</td>
<td>/PARAM/</td>
</tr>
<tr>
<td>NFOUND</td>
<td>/SHIFT/</td>
</tr>
<tr>
<td>NFRSUB</td>
<td>/SUBSTR/</td>
</tr>
<tr>
<td>NHARM</td>
<td>/PARAM/</td>
</tr>
<tr>
<td>NLIUB</td>
<td>/SUBSTR/</td>
</tr>
<tr>
<td>NMAX</td>
<td>/PARAM/</td>
</tr>
</tbody>
</table>
C NNODE /PARAM/ NUMBER OF NODE IN THE MESH
C NOFFST /SHIFT/ NUMBER OF OFFSETS FOR NEW EIGENVALUES/EIGENVECTORS
C NPRINT /PARAM/ NUMBER OF PRINT OPTIONS
C NSBC /PARAM/ NUMBER OF STRESS BOUNDARY CONDITIONS
C NSBNC /PARAM/ NUMBER OF STRESS BOUNDARY CONDITIONS
C NSHIFT /PARAM/ NUMBER OF POINTS FOR POWER SHIFT
C NSPRI /PARAM/ NUMBER OF D.O.F. WITH ADDED STIFFNESS
C NSUB /PARAM/ NOT USED IN VERSION 1.7 AND UP
C NSUMAX /MAXIMA/ NOT USED IN VERSION 1.7 AND UP
C NSUMCH /PARAM/ SUM OF PROFILE COLUMN HEIGHTS FOR UPPER
C NTIE /PARAM/ TRIANGULAR PART OF GLOBAL ARRAY STORED IN PROFILE FORM
C NTRAC /PARAM/ NUMBER OF TYING CONSTRAINT DATA SETS
C NTRAN /PARAM/ NUMBER OF NODAL FORCE VECTOR INPUT
C NTYPE /PARAM/ NUMBER OF NODAL COORDINATE TRANSFORMATION DATA
C OMEGB /HARMON/ FREQUENCY OF EXCITATION FOR HARMONIC BASE MOTION
C OMEGH /HARMON/ FREQUENCY OF EXCITATION FOR HARMONIC NODAL LOADS

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

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

*** -1- ******************************************************************************

COMMON / ADDVAL / ISPRI ,KSPRI ,IDASH ,KDASH ,IMASS ,KMASS
COMMON / ALGEM / ICREAD ,ILPRNT ,JLPRNT ,ICONSL ,IPOSTF ,JSCRAF,
1 IPILOTB ,JISTR ,JRCREAD ,IPVARS ,IPSETS ,IFILEX,
2 PI ,LINE ,LINE2
COMMON / CONTRO / JEND ,JITER ,JTEMP ,JPRINT ,JP ,JSUB ,
1 JINC ,JREST ,JSAVE ,JREDIM ,JPOST ,
2 JBACK ,JOPTIM ,JCREEP ,JDIST ,JCONST ,JDNY ,
3 NONISO ,IHERM ,ITRIG ,IDYN ,JREPOT ,JTANGE,
4 JHERM ,JFORCE ,JUTEMP ,JUOEF ,JDIST ,JHOCK ,
5 JDERIV ,JUBOUN ,IDSTOP ,INTSTR ,JPLAST ,JBAND ,
6 JFRONT ,JDEFOR ,JMEMED ,JTEST ,JDISP ,JFBFGS ,
7 JSCNTM ,JLINE ,IPPRNT ,ICOMP ,IPONJ ,JEIGEN ,
8 JBODY ,IFGRAV ,IPCENT ,JDAMP ,JDNY ,IJSTAT ,
9 JFSXX ,JISTIF ,JCENTM ,JINIT ,JLARGE ,JFOLLOW ,
+ JWKSLP ,JPRNS ,JCDUM2 ,JCDUM3
COMMON / DAMP / DAMPF(3)
COMMON / EIGEN / IEIGNC ,IGNMS ,IQMEG ,IMENO ,IDYNMD ,JSTRT2 ,
1 IPTAR ,IPTBR ,IPTVMD ,IMDM ,IQMEG
COMMON / MODSUP / IMPOR0 ,IMDIS0 ,IMVE0 ,IMPOR1 ,IMDIS1 ,IMVEL1
COMMON / HARMON / OMEGH ,IHARM ,KHARM ,OMEGB ,IBASE ,KBASE ,
COMMON / PERPAR / IPTYPE( 32), NTYPE, NPVAR, NPSET, JPERT , NESSUS
1 NVPCON, NPP009, NPP010, NPP011, NPP012 NESSUS
COMMON / PERPTR / IMEANS, ISTDEV, IPDATA, IVTYPE, ISKIP , IREDIF, NESSUS
1 IDINC0, IREAC0, IRED0, ISTIF0, IMASS0, NESSUS
2 IPP013, IOMEG0, IOMEGP, IETAK, IZETAK NESSUS
COMMON / PERDAT / IXCOORD, IXCHAR, IXFORC, KXFORC, IXDIST, KXDIST, NESSUS
1 IXTEMP, JXTEMP, IXBEAM, IXFVEC, IXSPRI, IXSPRI, NESSUS
2 IXPREF, IXP015, IXP016, IPWBEG, IPWEND NESSUS
COMMON / POWER / IELPHI, IELTNM, IEPNSM, ISIGNO, IHFN, IHFC, NESSUS
1 IEPB, ISP, ISFF, ISQQ, ICOQ, ITNM , NESSUS
2 IPSF, IPSD
COMMON / PULSES / IPULSE, KPULSE, IPD TIM, IPD FOR
COMMON / MAXIMA / MAXCRD, MAXFR, MAXNOD, MAXCHR, MAXDRS, NESSUS
1 MAXLAY, MAXINT, MAXWKR, MAXLN, NSUMAX, MAXCMP, NESSUS
2 MAXBSP, MAXCMR, MAXTEM, MAXELE, MAXL, MAXMT, NESSUS
3 MAXFRN, MAXBET, MAXVAR, MAXSET, MAPEX, MAXORD, NESSUS
4 MAX025, MAX026, MAX027, MAX028, MAX029, MAX030 NESSUS
COMMON / PARAM / NTPE, NELEM, NNODE, NBC, NITE, NMAX , NESSUS
1 NTRAN, NTRAC, NDP, NBD, NEXT, NSUB , NESSUS
2 NPRINT, NPOST, NBEC, NDU, NSE, NEBECT , NESSUS
3 NSHIFT, NSERG, NMR, NSPI, NMAX, NDASH, NESSUS
4 NDYNMD, NSBNC, NSUP, NHARM, NEBASE , NESSUS
5 NSUB, NSPIT, NFDPIT, NPULSE, NFPOPS, NHAIRD, NESSUS
6 NSUMC, NIMEN, NMONIT, NPAR40, NPAR41, NPAR42, NESSUS
7 NPAR43, NPAR44, NPAR45, NPAR46, NPAR47, NPAR48 NeSSUS
COMMON / TMARCH / DALPHA, DBETA, DGAMMA NESSUS
COMMON / PERIOD / JPEROD( 2 ), IPDISP, IPFORC, INDISP, IPF O R C NESSUS
COMMON / SUBELM / ISUBEL, ISUBNP, ISUBPT, NSDATA, ISUBTY, IMBED NESSUS
COMMON / SHIFT / ISHT, KSHIF T, IFREQ, LIFREQ, NOFFST, NOFOUND NESSUS
COMMON / SUBTYP / NSUCRD, NSUNFR, NSUND, NSUBR, NSUBR, NSUPR , NESSUS
1 NSUSE, NSUSEL, NSUND, NSUSE, NSUIDF NESSUS
COMMOM / BSGCT / IBSECT, KSBECT NESSUS
COMMON / START1 / IELPRM, ITPP, INEL, ICHAR, IPRES, ISTRS , NESSUS
1 ISTRN, ICP, IPRINT, IPOST, IDIST, ILEAN , NESSUS
2 IBRES, IBNORM, IMONIT, IST116, IST117, IST118 NESSUS
COMMON / START2 / INOD, ITEM, INLV, IPGUS, ITMDF, IDUP NESSUS
COMMON / START3 / IIEC, ITI, ITR, ITRAC, IEXT , NESSUS
1 ISBC , ISBCR
COMMON / START4 / IDINC, IDITOT, IPORCE, IREDISD, IWINOD, ISIGNO, NESSUS
1 IEPNSO, IEPSTN, ICSTRN, ICSTRN, IISTR, IISTRN, NESSUS
2 IIPSTR, IICSTR, IITSF, IIPSTN, ISTN, ISTNO, NESSUS
3 IISNO, IISNSO, IIPNSO, IICNSO, IITSNO, IDMAT, NESSUS
4 IIDMNO, IEQST, IOEMNO, IIOEMNO, ITDSNO, IVSWTO, NESSUS
IDYNV, IDYNA, IDSX1, IDSX2, IDSITR, ISWELL,
IEQCSI, IPREF, IDSX3, IYIELD, IDFINC, IDFTOT,
IST443, IST444, IST445, IST446, IST447, IST448
COMMON / START5 / IRL, IREAC, IES, IAB, IBQM, ISRL,
IBTL, ISKM, ILAST, IRLB, IDINCP, IFORIN,
IOP, IDAM, IMASMT, IDIAG, IUPTRI, ICOLFT,
IMASDI, IMASUP, IST521, IST522, IST523, IST524
COMMON / START6 / IELV, ICOR, ISIG, IEPS, IWNOD, ISNOD,
IENOD, IETM, ICH, IPP, IXRL, IXIRL,
IXP, LK, KPSTRN, KCSTRN, KISTRN, KLSTRS,
KISTRN, KIPSTR, KICSTR, KITSTR, KPSTNO, KCSTNO,
KISTNO, KISTNO, KSNNO, KPSNO, KCSTNO, KISTNO,
IMASNO, IMNOD, IEQPST, IEQPSI, KEQPST, KEQPSI,
KDMAT, KDMN0, KTDSNO, KITDST, LXM, LXC,
IVEDM, IAELO, IMASEL, KYIELD, IST547, IST548,
IST549, IST550, IST551, IST552, IST553, IST554
COMMON / START7 / ICON, IKBAR, ITRACR, ITTRANR, IBETA, IDET
COMMON / START8 / KGEPS, KIGEPS, KKGIG, KGIGSIG, KGIDST,
UGEPS, IGIGSIG, IIGSNO, IIGSNO, IGIDNO,
KGEPS, KIGIGSIG, KIGSIGSIG, KIGSIGSIG, KGIDNO
COMMON / START9 / KEQCSI, KIQMNO, KSWLNO, KTFMNO, KDEMMY,
KEQCSS, KOMENO, KVSWIO, IST910, IST911, IST912
COMMON / SUBSTR / NLVSUB( 10), NFRSUB( 10)
COMMON / MACHIN / IDP
COMMON / ERRORS / IERR

**--------------------------------------------------------**
**--------------------------------------------------------**

C DIMENSION RWORK ( ISIZE ) , IWORK ( ISIZE )
C
C DIMENSION YIELDC( 3 )
C
C *** -2-  *****************************************************
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 MAXWRF = 1
C MAXDRT = 1
MAXNLV = 1
MAXCMP = 1
MAXBSP = 6
MAXLWK = 1

MAXELM = NELEM + NDUP + NTIE

YIELDC(1) = 1.00D+36
YIELDC(2) = 0.00D+00
YIELDC(3) = 0.00D+00

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

MAXNLV = NSUMAX
IELPRM = 1

***

IF(NTYPE.EQ.0) GO TO 2

***

DO 1 I = 1 ,NTYPE
   IC = I
   IS1 = IELPRM + (IC-1)*13 + 1

   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)

   NWRK = IWORK(IS1+3)* MAXINT
   MAXWRK = NWRK

   MNLV = IWORK(IS1+1)*IWORK(IS1+2)
   MAXNLV = MNLV

   MWBE = IWORK(IS1+1)*IWORK(IS1+2)
   MAXBE = MWBE

   MLWK = MAXINT *IWORK(IS1+7)
   MMLWT = MWBE *IWORK(IS1+8)

1
   IF(MAXLWK.LT.MLWK)
      MAXLWK = MLWK

      MDWT = IWORK(IS1+3)*IWORK(IS1+3)
1 * MAXINT
   IF(MDMT.GT.MAXDMT) MAXDMT = MDMT
C
C **********************************************************************
C *** CORE ALLOCATION FOR BEAM SECTION PROPERTY SETS ***
C **********************************************************************

   IF ( NBSECT .EQ. 0 ) GO TO ii0

   IBSEC9 = IIAST
   KBSECT = IBSEC9 + NBSECT * MAXBSP * IDP
   IIAST = KBSECT + NBSECT

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

110 CONTINUE

C **********************************************************************
C *** CORE ALLOCATION FOR POWER SHIFT IN EIGEN ANALYSIS ***
C **********************************************************************

   IF ( NSHIFT .EQ. 0 ) GO TO 120

   ISHIFT = IIAST
   IFREQ = ISHIFT + NSHIFT * IDP
   LFREQ = IFREQ + NSHIFT * IDP
   KSHIFT = LFREQ + NSHIFT * IDP
   IIAST = 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 + NPDPTS * IDP
KPULSE = IPULSE + NPULSE * 2 * IDP
ILAST = KPULSE + NPULSE * 2

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

CONTINUE

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

IF (JEMBED .GT. 0)

IEMBED = ILAST
ISUBEL = IEMBED
ISUBNP = ISUBEL
ISUBPT = ISUBNP
ISUBY = ISUBPT

NSDATA = 34

ICHAR = ISUBY + MAXELM
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
KEEPS = KDMAT + MAXDMT * IDP
KIGEPS = KEEPS + MAXWRK * IDP
KGSIG = KIGEPS + MAXWRK * IDP
KIGSIG = KGSIG + MAXWRK * IDP
KGTSTRN = KIGSIG + MAXWRK * IDP
KISTRS = KGTSTRN + MAXWRK * IDP
ILAST = KISTRS + MAXCHR * IDP * MAXNOD

IPRINT = ILAST
IMONIT = IPRINT + 12 * NPRINT
IPOST = IMONIT + 4 * NMONIT
ILAST = IPOST

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

C **********************************************************************
C *** CORE _IC_ FOR NODAL QUANTITIES ***
C **********************************************************************

INOD = IIAST
IWINOD = INOD + MAXCRD * NNODE * IDP
IBPRES = IWINOD + NNODE * IDP
IBNORM = IBPRES + JRES * NNODE * IDP
ISIGNO = IBNORM + JRES * NNODE * IDP
IEPSNO = ISIGNO + MAXCMP * NNODE * IDP * 3
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 = IIISTNO + MAXCMP * NNODE * IDP * MAXLAY
IIPSNO = IISNNO + MAXCMP * NNODE * IDP * MAXLAY
IICSNO = IIPSNO + MAXCMP * NNODE * IDP * MAXLAY
IITSNO = IICSNO + MAXCMP * NNODE * IDP * MAXLAY
ITDSNO = IITSNO + MAXCMP * NNODE * IDP * MAXLAY
IOMENO = ITDSNO + MAXCMP * NNODE * IDP * MAXLAY
IIOQNO = IOMENO + MAXCMP * NNODE * IDP * MAXLAY
IGEPNO = IIOQNO + MAXCMP * NNODE * IDP * MAXLAY
IGENO = IGEPNO + MAXSTR * NNODE * IDP
IGSINO = IGENO + MAXSTR * NNODE * IDP
II-IITNO = IIGSMNO + MAXSTR * NNODE * IDP
IIDSNO = II-IITNO + MAXSTR * NNODE * IDP
IIMENO = IIDSNO + MAXSTR * NNODE * IDP
II-IQNO = IIMENO + MAXSTR * NNODE * IDP
IGQNO = II-IQNO + MAXSTR * NNODE * IDP
IIGENO = IGQNO + MAXSTR * NNODE * IDP
IMASNO = IIGENO + MAXSTR * NNODE * IDP
IEQPSI = IMASNO + NNODE * IDP
IEQPST = IEQPSI + NNODE * IDP * MAXLAY
IEQCST = IEQPST + NNODE * IDP * MAXLAY
IEQCSI = IEQCST + NNODE * IDP * MAXLAY
IVSWTO = IEQCSI + NNODE * IDP * MAXLAY
ISWELL = IVSWTO + NNODE * IDP * MAXLAY

SYSTEMS’ MANUAL

Page: 67

MHOST Version 4.2
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

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

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

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

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

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

C
***

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

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

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

C
IMDAM = ILAST
IOMEGD = IMDAM + NDYNMD * IDP
IMFOR0 = IOMEGD + NDYNMD * IDP
IMDIS0 = IMPOR0 + NDYNMD * IDP
IMVEL0 = IMDIS0 + NDYNMD * IDP
IMFOR1 = IMVEL0 + NDYNMD * IDP
IMDIS1 = IMPOR1 + NDYNMD * IDP
IMVEL1 = IMDIS1 + NDYNMD * IDP
ILAST = IMVEL1 + NDYNMD * IDP

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

C
ICMFOR = ILAST
ICMRES = ICMFOR + 2 * NSUPER * IDP
ICHHFN = ICMRES + 2 * NSUPER * IDP
ICBFHN = ICHHFN + 2 * NSUPER * IDP
ICCMAT = ICBFHN + 2 * NSUPER * IDP
ICBEXC = ICCMAT + 2 * NBASE * IDP * 2 * NBASE * IDP
ILAST = ICBEXC + 2 * NBASE * IDP

180 CONTINUE

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 + NFDPTS * IDP
ISFF = ISPP + NSUPER * NSUPER * IDP
ISQQ = ISFF + NSUPER * NSUPER * IDP
ICQQ = ISQQ + NSUPER * NSUPER * IDP * 2
ISIGMD = ICQQ + NSUPER * NSUPER * IDP * 2
IEPSMD = ISIQ4O + NSUPER * NNODE * IDP
IELPHI = IEPSMD + NSUPER * NNODE * IDP
IELTNM = IELPHI + MAXNOD * IDP
ITNM = IELTNM + MAXNOD * IDP
ILAST = ITNM + NSUPER * NNODE * IDP

280 CONTINUE

IEOLPT = ILAST
IELV = IEOLPT + 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 + MAXNFR * NNODE
IF(ISNOD.EQ.2*(ISNOD/2)) ISNOD = ISNOD + 1
IENOD = ISNOD + MAXCMP * MAXNOD * IDP * MAXLAY
KPSTNO = IENOD + MAXCMP * MAXNOD * IDP * MAXLAY
KSTNO = KPSTNO + MAXCMP * MAXNOD * IDP * MAXLAY
KTNSTNO = KSTNO + MAXCMP * MAXNOD * IDP * MAXLAY
KPSNO = KTNSTNO + MAXCMP * MAXNOD * IDP * MAXLAY
KCSNO = KPSNO + MAXCMP * MAXNOD * IDP * MAXLAY
KTSNO = KCSNO + MAXCMP * MAXNOD * IDP * MAXLAY
KTDNO = KTSNO + MAXCMP * MAXNOD * IDP * MAXLAY
KEQPST = KTDNO + MAXCMP * MAXNOD * IDP * MAXLAY
KEQPSI = KEQPST + MAXNOD * IDP * MAXLAY
KEQCSI = KEQPSI + MAXNOD * IDP * MAXLAY
KOMNO = KEQCSI + MAXNOD * IDP * MAXLAY
KSWLNO = KOMNO + MAXSTR * MAXNOD * IDP
KTPNNO = KSWLNO + MAXNOD * IDP
KIDFNO = KTPNNO + MAXNOD * IDP
KDMNO = KIDFNO + MAXNOD * IDP
KGEFNO = KDMNO + MAXSTR * MAXSTR * MAXNOD * IDP
KIGENO = KGEFNO + MAXSTR * MAXNOD * IDP
KIGSNO = KIGENO + MAXSTR * MAXNOD * IDP
KTDSNO = KIGSNO + MAXSTR * MAXNOD * IDP
KIELD = KTDSNO + MAXSTR * MAXNOD * IDP
IXRL = KIELD + NHARD * MAXNOD * IDP * 3
IXIRL = IXRL + MAXNLV * IDP
IVELM = IXIRL + MAXNLV * IDP
IAELM = IVELM + MAXNLV * IDP * IDYNM
IXP = IAELM + MAXNLV * IDP * IDYNM
IXK = IXP + MAXNLV * IDP
IDET = IXK + MAXNLV * MAXNLV * IDP
IXM = IDET + MAXINT * IDP
IXC = IXM + MAXNLV * MAXNLV * IDP * IDYNM
IBETA = IXC + MAXNLV * MAXNLV * IDP * IDYNM
IDTOT = IBETA + MAXBET * IDP
IDYN = IDTOT + NNODE * MAXNFR * IDP
IDYNV = IDYN + NNODE * MAXNFR * IDP * IDYNM
IRL = IDYNV + NNODE * MAXNFR * IDP * IDYNM
ILAST = IRL
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 = IRESID + NNODE * MAXNFR * IDP * 2
IF( IPCONJ .NE. 0 ) IREAC = IREAC + NNODE * MAXNFR * IDP
IF( IPSCTNT .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
IEBNVC = IBTLC
IBTLC = IEBNVC + NSBNC * NNODE * MAXNFR * IDP * 2
IDSTR = IDCSTR + NNODE * MAXNFR * IDP * IDYNM
IDSX1 = IDSTR + 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*NSBNC

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

IPWBEG = ILAST
IXCOOR = IPWBEG
IXCHAR = IXCOOR + IPTYPE( 1)*NNODE*MAXCRD*IDP
IXFORC = IXCOOR + IPTYPE( 2)*NNODE*MAXCHR*IDP
KXFORC = IXFORC + IPTYPE( 3)*NTRAC*2*IDP
IXDIST = KXFORC + IPTYPE( 3)*NTRAC*4
IF (IXDIST .EQ. 2*(IXDIST/2)) IXDIST = IXDIST+1
KXDIST = IXDIST + IPTYPE( 4)*NNODE*2*MAXPRS*IDP
IXTEMP = KXDIST + IPTYPE( 4)*NELEM
IF (IXTEMP .EQ. 2*(IXTEMP/2)) IXTEMP = IXTEMP+1
JXTEMP = IXTEMP + IPTYPE( 5)*NNODE*MAXLAY*IDP
IXBEAM = JXTEMP + IPTYPE( 5)*NNODE*MAXLAY*IDP
IXSPRI = IXBEAM + IPTYPE( 6)*NBSEC*MAXBSP*IDP
KXSPRI = IXSPRI + IPTYPE( 7)*NSPRI*IDP
IXFVEC = KXSPRI + IPTYPE( 7)*NSPRI*3
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
ILAST = IPWEND

680 CONTINUE

C

IBTL = ILAST
ISTRY2 = IBTL

C *** MESSAGE OUTPUT AND RETURN TO DATA INPUT ROUTINES **********************

C

ILAST1 = ILAST + 2 * NNODE + 16
NPX = ILAST1 - ITYP + 1
NSHORT = ILAST1 - ISIZE

C

CALL LINES(4,4)
WRITE(ILPRINT,2001) ILAST1
2001 FORMAT(//'NUMBER OF WORDS NEEDED IN BLANK COMMON FOR DATA,
*6H INPUT, I8, '*******************************************)
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, RWORK(INDEX), 3 )
800 CONTINUE

C RETURN
C

C ***************************************************************
C ** PRINT-OUT THE AMOUNT OF CORE REQUIRED AND TERMINATE EXECUTION **
C ***************************************************************
C
900 CONTINUE
IERR = IERR + 1
CALL LINES( 3, 3 )
WRITE(ILPRNT,2005) NSHORT
2005 FORMAT(//,2X,'INCREASE BLANK COMMON AND ISIZE WITH',I8,' WORDS')
C
CALL QUIT('MEMD', 'RY A', 'LOC', 'ATIO', 'N ', ' ', IERR)
C
******************************************************************************
C
END

The following SUBROUTINE INIT2 calculates pointers for the global arrays:

C=SUBROUTINE=INITI2 CALLED BY SUBROUTINE 'STRUCT'
SUBROUTINE INITI2(RWORK, IWORK, ISIZE)
C
******************************************************************************
C
IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 RWORK
C
******************************************************************************
C
COMMON / ALGEM / ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1      IPL01B, IRSRT, JCREAD, IPIVARS, IPSETS, IFILEX,
2      PI , LINE , LINE2
COMMON / CONTRO / JEND , JITER , JTEMP , JPRINT, JP , JSUB ,
1      JINC , JREST , JSAVE , JREDIM, JAUTO , JPOST ,
2      JBACK , JOPTIM, JCREEP, JDIST , JCONST, JDYN ,
3      JNISO, JHERM, ITRIG , JHYMN , JREPOT, JTANGE ,
4      JHERM, JFORCE, JUTEMP, JUCOEF, JDISTS, JUHOCK ,
5      JDERIV, JBOUND, JDISTOP, JINSTR, JLAST, JBAND ,
6      JFRONT, JDEFOR, JEMBED, JTEST , JDISP , JBFJG ,
7      JSCNT, IILINE, JPRINT, ICOMPS, ICOMJ, JEIGEN ,
8      IFBODY, IPGRV, IPCENT, JDAMP , LDYN , JSTAT ,
9      JFDXX, JSTIF, JCENTM, JFINIT, JUHOCK, JUHJG ,
+      JWKS , JRES , JCDUM2, JCDUM3
COMMON / PERPAX / IPTYPE( 32), NTYPE, NPVARS, NPSETS, JPERT , NESSUS
1      NPFVCON,NPP008,NPP009,NPP010,NPP011,NPP012 }
COMMON / PERPTR / IMEANS,ISTDEV,IPDATA,IVTYPE,ISKIP,IREDEF, 
1 ND:IES,IREAC0,IRESD0,IDGRP,ISTIF0,IMASS0, 
2 IDP013,IMEG0,IOEGP,IMOEGK,ITAK,IZETAK 
COMMON / PERIOD / IPEROD( 2 ),IPDISP,IPFORC,INDISP,INFORC 
COMMON / EIGEN / IEGNVC,IGNMS,IMEG0,IMOENO,IPM01D,ISTR12, 
1 IPAR,IPPER,IPTVED,IMAM,IMOEGD 
COMMON / ERRORS / IERR 
COMMON / MAXIMA / MAXCRD,MAXNFR,MAXNOD,MAXSTR,MAXCHR,MAXPRS, 
1 MAXLAI,MAXINT,MAXWRK,MAXNLV,NSUMAX,MAXCMP, 
2 MAXBSP,MAXGM,MAXTEM,MAXELM,MAXLWK,MAXRMT, 
3 MAXFRN,MAXBGT,MAXVAR,MAXSET,MAXEAN,MAXORD, 
4 MAX025,MAX026,MAX027,MAX028,MAX029,MAX030 
COMMON / PARAM / NTYPES,NELEM,NNODE,NBC,NTIE,NMAX, 
1 NTRAC,NFRA,NBAND,NEXT,NSUB, 
2 NPNRINT,NSCOUNT,NSC,NDUP,NSIZE,NSBCT, 
3 NSHIFT,NSSPFS,NGMRS,NSPRI,NMASS,NDASH, 
4 NDAYMDS,NSENC,NSUPERN,NAHARM,NBASE,NTINC, 
5 NSTF,NEFPTS,NFDPFS,NNPRES,NSPBTS,NSHRD, 
6 NSUMCH,NDIMEN,NDMite,NDIP40,NDP41,NDP42, 
7 NDIP43,NDP44,NDP45,NDP46,NDP47,NDP48 
COMMON / START1 / IELPRN,ITYP,INEL,ICHAR,IPRES,ISTR1, 
1 ISTRN,ICOP,IPRINT,IPRES,IST1, 
2 IPRES,IEBNORM,IMONIT,IST116,IST117,IST118 
COMMON / START3 / ILEC,ITI,ITR,ITRAN,ITRAN, 
1 IBSEC,ISBEC 
COMMON / START5 / IES,LAB,ILR,BM4,ISRL, 
1 IBLC,ISKM,ILAST,ILRB,IDINCPI,IFORIN, 
2 IOP,IMASMT,IMASMT,IMDIAG,IPUNTR,ICOLP, 
3 IMASDI,IMASP,IST521,IST522,IST523,IST524 
COMMON / START4 / IDINC,IDIOT,IFORCE,IRESID,IVNOD,ISIGNO, 
1 IEPSNO,IPSTN,ICSTR,ISTRN,ISTR1,ISTR1, 
2 IIIST,LTISTR,ITSPST,ITSPNO,ISTN0,ITSTN0, 
3 ITINS0,ITINSO,ITOPS0,ITCSON0,ITMAT, 
4 IIDMNO,IEBCST,IMENO,ISTMNO,IVSWS0, 
5 IDYNV,IMAYA,ISX1,ISX2,ISDTR,ISWELL, 
6 IEQCSI,IPREF,ISX3,ILYIELD,IEFINC,IFITOT, 
7 IST443,IST444,IST445,IST446,IST447,IST448 
COMMON / MACHIN / IDP 

C C *** CONSTRUCT THE REVERSE CONNECTIVITY TABLE ****************** 
C 
C IBEG = INEL 
C IEND = IBEG + NELEM*MAXNOD 
C 
C ... FIND THE MAXIMUM NUMBER OF ELEMENTS AT A NODE
C
MAXEAN = 0
DO 130 NN = I, 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... ALLOCATE SPACE FOR THE REVERSE CONNECTIVITY TABLE
C
ILEAN = ILAST
ILAST = ILEAN + NNODE*MAXEAN
C
CALL NULINT(IWORK(ILEAN), MAXEAN*NNODE)
C... CONSTRUCT THE REVERSE CONNECTIVITY TABLE
C
DO 160 NN = I, NNODE
   NF = 0
   IN = INEL-1
   DO 150 IE = I, NELEM
      DO 140 IJ = I, MAXNOD
          IN = IN+1
          IF ( IWORK(IN) .EQ. NN ) THEN
              NF = NF + 1
              IM = ILEAN+(NN-I)*MAXEAN+NF-1
              IWORK(IM) = IE
          ENDIF
140 CONTINUE
150 CONTINUE
160 CONTINUE
C
**********************************************************************
C
IADRES = ILAST
IBILC  = ILAST
ISKM   = IBILC + NEXT*NFD*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
IDIAG = IIAST
IF (IDIAG.EQ.2*(IDIAG/2) ) IDIAG = IDIAG + 1
IUPTRI = IDIAG + NFD*IDP

C
C
C
C
C
C
C
C
C

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

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

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

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

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

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
ICMEGO = IMASS0 + IH
ICMEGP = ICMEGO + NDYNMD * IDP
ILAST = ICMEGP + NDYNMD * IDP

... CHECK HOW MANY PERTURBATION TERMS THERE IS ROOM IN CORE FOR

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

... IN EITHER CASE GO ON AND ESTIMATE THE MINIMAL REQUIREMENTS

ICMEGK = ILAST
IETAK = ICMEGK + NDYNMD*MAXORD*IDP
IZETAK = IETAK + NFD*NDYNMD*MAXORD*IDP
ILAST = IZETAK + NFD*NDYNMD*MAXORD*IDP

300 CONTINUE

*** CHECK AGAINST THE SIZE OF BLANK COMMON AVAILABLE ***************

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,16HHALF-BANDWIDTH ,I15)

CALL LINES ( 4 , 4 )
ILAST1 = 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

*** WORKING STORAGE NOT SUFFICIENT !

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', 'LLOC', '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 NSUINT, NSULV, NSULAY, NSUNDI, NSUSHR, NSUIDF
COMMON / ALGEM / ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1 IPLOTB, ISTRRT, JCREAD, IPIvars, IPSETS, IFILEX,
2 PI, LINE, LINE2
C
C ***********************************************
C
FUNCTION
C
--------
C
ALLOCATES CORE STORAGE FOR THE SUBELEMENT MESH ASSOCIATED
C WITH THE IELEM-TH MASTER ELEMENT
C
**ARGUMENTS**

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

- **IWORK**  INTEGER WORKING SPACE
- **NWORK**  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(-,IELEM)**  I  ARRAY

<table>
<thead>
<tr>
<th>I</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ELEMENT CONNECTIVITY</td>
</tr>
<tr>
<td>2</td>
<td>PHYSICAL COORDINATES</td>
</tr>
<tr>
<td>3</td>
<td>ELEMENT COORDINATES</td>
</tr>
<tr>
<td>4</td>
<td>TOTAL NODAL DISPLACEMENT</td>
</tr>
<tr>
<td>5</td>
<td>INCREMENTAL NODAL DISPLACEMENT</td>
</tr>
<tr>
<td>6</td>
<td>CORRECTIONAL NODAL DISPLACEMENT</td>
</tr>
<tr>
<td>7</td>
<td>TOTAL NODAL STRAIN</td>
</tr>
<tr>
<td>8</td>
<td>TOTAL NODAL STRESS</td>
</tr>
<tr>
<td>9</td>
<td>TOTAL NODAL PLASTIC STRAIN</td>
</tr>
<tr>
<td>10</td>
<td>TOTAL NODAL CREEP STRAIN</td>
</tr>
<tr>
<td>11</td>
<td>TOTAL NODAL THERMAL STRAIN</td>
</tr>
<tr>
<td>12</td>
<td>GENERALIZED NODAL STRESS (TOTAL)</td>
</tr>
<tr>
<td>13</td>
<td>GENERALIZED NODAL STRAIN (TOTAL)</td>
</tr>
</tbody>
</table>

SYSTEMS' MANUAL

Page : 81

MHST Version 4.2
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
C IF( KEMBED .EQ. 0 ) RETURN
C
C WRITE(ICONSL,2000) IELEM
C
2000 FORMAT(' SUBELEMENT CORE ALLOCATION FOR ELEMENT NO.',I5)
C
C CALL SUBELV
C 1 (IWORK ,KSUBTY,IERR )
C
C IPOINT(  1 ) = ILAST
C IPOINT(  2 ) = IPOINT(  1 ) + NSUNOD * KSUBEL
C IPOINT(  3 ) = IPOINT(  2 ) + NSUCRD * KSUBNP * IDP
C IPOINT(  4 ) = IPOINT(  3 ) + NSUCRD * KSUBNP * IDP
C IPOINT(  5 ) = IPOINT(  4 ) + NSUNFR * KSUBNP * IDP
C IPOINT(  6 ) = IPOINT(  5 ) + NSUNFR * KSUBNP * IDP
C IPOINT(  7 ) = IPOINT(  6 ) + NSUNFR * KSUBNP * IDP
C IPOINT(  8 ) = IPOINT(  7 ) + NSUCMP * KSUBNP * IDP
C IPOINT(  9 ) = IPOINT(  8 ) + NSUCMP * KSUBNP * IDP
C IPOINT( 10 ) = IPOINT(  9 ) + NSUCMP * KSUBNP * IDP
C IPOINT( 11 ) = IPOINT( 10 ) + NSUCMP * KSUBNP * IDP
C IPOINT( 12 ) = IPOINT( 11 ) + NSUCMP * KSUBNP * IDP
C IPOINT( 13 ) = IPOINT( 12 ) + NSUSTR * KSUBNP * IDP
IPOINT( 14 ) = IPOINT( 13 ) + NSUSTR * KSUBNP * IDP
IPOINT( 15 ) = IPOINT( 14 ) + NSUCMP * KSUBNP * IDP * NSULAY
IPOINT( 16 ) = IPOINT( 15 ) + NSUCMP * KSUBNP * IDP * NSULAY
IPOINT( 17 ) = IPOINT( 16 ) + NSUCMP * KSUBNP * IDP * NSULAY
IPOINT( 18 ) = IPOINT( 17 ) + NSUCMP * KSUBNP * IDP * NSULAY
IPOINT( 19 ) = IPOINT( 18 ) + NSUSTR * KSUBNP * IDP
IPOINT( 20 ) = IPOINT( 19 ) + NSUSTR * KSUBNP * IDP
IPOINT( 21 ) = IPOINT( 20 ) + NSUSTR * KSUBNP * IDP
IPOINT( 22 ) = IPOINT( 21 ) + KSUBNP * IDP
IPOINT( 23 ) = IPOINT( 22 ) + NSUSTR * KSUBNP * IDP * NSULAY
IPOINT( 24 ) = IPOINT( 23 ) + KSUBNP * IDP
IPOINT( 25 ) = IPOINT( 24 ) + KSUBNP * IDP
IPOINT( 26 ) = IPOINT( 25 ) + KSUBNP * IDP
IPOINT( 27 ) = IPOINT( 26 ) + KSUBNP * IDP
IPOINT( 28 ) = IPOINT( 27 ) + KSUBNP * IDP
IPOINT( 29 ) = IPOINT( 28 ) + KSUBNP * IDP
IPOINT( 30 ) = IPOINT( 29 ) + NSUCMP * KSUBNP * IDP * NSULAY
IPOINT( 31 ) = IPOINT( 30 ) + NSUCMP * KSUBNP * IDP * NSULAY
IPOINT( 32 ) = IPOINT( 31 ) + NSUCMP * KSUBNP * IDP * NSULAY
IPOINT( 33 ) = IPOINT( 32 ) + NSUCMP * KSUBNP * IDP
IPOINT( 34 ) = IPOINT( 33 ) + NSUSTR * KSUBNP * IDP
KLAST = IPOINT( 34 ) + NHARD * KSUBNP * IDP * 3

CALL NULINT( IWORK( ILAST ), KLAST - ILAST + 1 )

*** UPDATE THE LAST ADDRESS
ILAST = KLAST

RETURN
END

The following SUBROUTINE SUBFEM accesses the arrays defined by the subelement pointer analysis:

C=SUBROUTINE=SUBFEM CALLED BY SUBROUTINE 'ASSE4'
SUBROUTINE SUBFEM
1 ( RWORK, IWORK, ISIZE, IPOINT, IELEM, NSUBEL, NSUBNP, NSUBTY, NSDATA, KEMBED, WORKSP, ISLV )

C

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

C
C **********************************************************************
C
C     ALGEM
C
C     ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1      IPLOTB, IPSTR, JCREAD, IPVARS, IPSETS, IFLEX,
2      PI, LINE, LINE2

C     CONTRO
1      JEND, JITR, JTEMP, JPRINT, JP, JSUB,
2      JINC, JREST, JSAVE, JREDIM, JAUTO, JPOST,
3      JBACK, JOPTIM, JCREEP, JDIST, JCONST, JDYN,
4      NONISO, ITERM, ITIRG, IDYN, JREPOT, JTANGE,
5      JTERM, JFORCE, JUTEMP, JUCOF, JDIST, JUHOOK,
6      JDERIV, JBOUND, JDISTJ, IRELJ, JBOUND,
7      JFORCE, JEMBED, JTEST, JDISP, JFBFGS,
8      IPSCNT, IFLINE, IPRINT, ICOMPS, IPCONJ, JEIGEN,
9      IPRBOD, IGRAV, IFCENT, JDA, LDYN, ISTAT,
10     JFDSXX, JISTIF, JCENTM, JFINIT, JLARGE, JFOLLOW,
11     + JWKSLP, JPRSC, JCDUM2, JCDUM3

C     ELTYP
1      NELCRD, NELNFR, NELNOD, NELSTR, NELCHR, NELPR,
2      NELINT, NELLV, NELLAY, NDI , NSHEAR, NELCMP

C     ELEMENT
1      IC, IEL, IDF, JLAW, IPATH, LASSEM,
2      JRULE, JCART, JELO09, JEL010, JEL011, JEL012

C     SUBTYP
1      NSUCRD, NSUNFR, NSUNOD, NSUSTR, NSUCHR, NSUCMP,
2      NSINT, NSULV, NSULAY, NSUSH, NSUORD

C     TRANSF
1     CTRANS( 9), XJACOB( 9)

C     LOUBIN
1     JLOUB, JINTER, JEXTRA, JWEIGH, JSUBRE, JISTRN,
2     JCIER, JHRLS, JGROM, JLOUB03, JLOUB04, JLOUB05

C     MAXIMA
1     MAXCRD, MAXNFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
2     MAXLAY, MAXINT, MAXWK, MAXLV, NSUMAX, MAXCMP,
3     MAXBSP, MAXCMR, MAXTEM, MAXLAW, MAXLV, MAXINT,
4     MAXFRN, MAXBET, MAXVAR, MAXSET, MAXEAN, MAXORD,
5     MAX025, MAX026, MAX027, MAX028, MAX029, MAX030

C     PARAM
1     NTYPE, NELM, NNODE, NBC, NTIE, NMAX,
2     NIRAN, NIRAC, NFD, NBAND, NEXT, NSUB,
3     NPRINT, NPST, NSEC, NDUP, NSIZE, NBECT,
4     NSHIFT, NSBFGS, NGMRS, NSPRI, NSAVE, NDASH,
5     NDYNDM, NSBNC, NSUPER, NHARM, NBASE, NTINC,
6     NTERR, NPSPTS, NFPDPTS, NPULS, NFDPTS, NHARD,
7     NSUMXY, NDMAN, NMONIT, NPR40, NPR41, NPR42,
8     NPR43, NPR44, NPR45, NPR46, NPR47, NPR48

C     ERRORS
1     IERR

C     MACHIN
1     IDP

C     TIME
1     TTIMINC, TOTINC, RUNTIM

C     START1
1     IELPR, ITYP, INEL, ICHAR, IPRES, ISTRS,
2     ISTRN, ICOP, IPRINT, IPOST, IDIST, ILEAN,
**COMMON / START2 /**
INOD , ITEM , INLV , IPOSU , ITEMDF , IDUP
**COMMON / START3 /**
IIBC , ITT , ITR , ITRAN , ITRAC , IEXT ,
1 ISBC , ISBCR
**COMMON / START4 /**
IDINC , IDTOT , IFORCE , IRESID , IWCONF , ISIGNO,
1 IEPENO , IPRTRN , ICSTRN , ITSTRN , IISTRN ,
2 IIPESR , IIICSTR , ITITSTR , IPSTNO , ICSTRN , ITSTRN ,
3 IISTNO , IIISNNO , IIIPSNNO , ATCSNO , ITISNO , IDMAT ,
4 IDMINO , IEQCSI , ICSTRN , IIOMNO , ITDSNO , IVSTM ,
5 IDINV , IDNYA , IDSX1 , IDSX2 , IDSTR , ISWELL ,
6 IEQCSI , IPREF , IDSX3 , IYIELD , IFORCE , IDFTOT ,
7 IST443 , IST444 , IST445 , IST446 , IST447 , IST448
**COMMON / START5 /**
/ RLR , IREC , IES , IAB , IBCM , ISRL ,
1 IETLC , ICDM , ILAST , IRLB , IDINCP , IFORIN ,
2 IOP , IDAM , IMAQMT , IDIAG , IUPFR , ICOFLPT ,
3 IMASDI , IMASUP , IST521 , IST522 , IST523 , IST524
**COMMON / START6 /**
IENOD , IEITM , ICH , IPP , IXRL , IXMLR ,
1 IXP , IXK , KPSTNO , KITSNNO , KITSR ,
2 KISTRO , KIPSTNO , KICSTRN , ITSTNO ,
3 KISNO , KISTNO , KINNO , KIPSNO , KICSNOS ,
4 IMASNO , IMNOD , IEQPST , IEQPST , KEQPST , KEQPST ,
5 KDMT , KDMNO , KTSNNO , KITSN , IXML , IXC ,
7 IVEEL , IEELM , IEASEL , KIYELD , IST647 , IST648 ,
8 IST649 , IST650 , IST651 , IST652 , IST653 , IST654
**COMMON / START7 /**
ICON , IBCMR , ITRANR , ITRAN , IBETA , IDET
**COMMON / START8 /**
KEEPS , KIEPS , KGNSG , KIGHG , KGNSg ,
1 KGNSN , KISGNO , KISNO , KIKNO ,
2 KGSNO , KGNSO , KGNSN , KGSNO ,
**COMMON / START9 /**
KEQCZI , KSOMNO , KSUSO , KKSNO , KDINO ,
1 KEQCST , KSOMNO , KSUSO , KKSNO , KDINO ,
**COMMON / TOLE /**
RELERR , ABSERR , REACMX , RESDXM , DISERR , DISTR ,
1 ENGTOR , ENGNRM

**********************************************************************
**DIMENSION RWORK ( ISIZE ) , IWORK ( ISIZE ) , ISLV ( 1 )
**DIMENSION NSUBEL( NELEM ) , NSUBNP( NELEM ) , NSUBTY( NELEM)
**DIMENSION WORKSP( ISIZE )
**DIMENSION IPOINT( NSDATA , NELEM )

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

---

**RWORK** | 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
---

POINTERS

---

**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 |
**NODEL TEMP_ TEMPERATURE DIFFERENCE**

**EQUIVALENT PLASTIC STRAIN (TOTAL)**

**EQUIVALENT PLASTIC STRAIN (INCREMENT)**

**EQUIVALENT CREEP STRAIN (TOTAL)**

**EQUIVALENT CREEP STRAIN (INCREMENT)**

**TOTAL SHIFT TENSOR**

**INCREMENTAL SHIFT TENSOR**

**TVSEWL**

**SWELL**

**GENERALIZED INITIAL STRESS**

**YIELD STRESS DEFINED AT NODES**

**Call TLIMOUT**

```
1 ('SUBL', 'LEME', 'NT L', 'OOP', 'ENTE', 'RED')
```

**WRITE(ICONSL, 200) IELEM**

**FORMAT(30X, 'IELEM=', I5)**

**CALL LIMIT**

```
KSUBLEL = NSUBEL(IELEM)
KSUBNP = NSUBNP(IELEM)
KSUBTY = NSUBTY(IELEM)
LBACK = 0
JMASTR = 1
KAW = JAW
ICONV = 0
CTOLER = RELERR
```

**Initialize here the number of integration points / element to be able to use some 'tricks' of 'reduced' integration at later stage**

```
IF(JWEIG .GE. 2) IGAUS = 0
IF(JWEIG .GE. 4) IGAUS = 3
ITRANS = 0
IF(JWEIG .GE. 3) ITRANS = 1
IGAU = 0
ITRAIN = 0
```
* *** 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 ***

**JTRANS** = 0
IF ( JTRANS .EQ. 2 ) JTRANS = 1
IF ( JTRANS .EQ. 2 ) JTRANS = 1

**KGAUS**: CONTROL VARIABLE FOR THE 'RECOVERY' INTEGRATION

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

**KGAUS**: CONTROL VARIABLE FOR THE 'RESIDUAL' INTEGRATION

```c
KGAUS = 0
IF ( JEXTRA .EQ. 2 ) KGAUS = 1
IF ( JWEIGHT .EQ. 2 ) KGAUS = 3
INTRSD = NELINT
IF ( JEXTRA .EQ. 2 ) INTRSD = 1
KTRANS = 0
IF ( JWEIGHT .EQ. 3 ) KTRANS = 1
```

* ARRANGE SUBELEMENT POINTERS FOR THE GLOBAL WORKAREA
* * *

```c
J CONNC = IPOINT(1, IELEM)
J PCORD = IPOINT(2, IELEM)
J ECORD = IPOINT(3, IELEM)
J DISPL = IPOINT(4, IELEM)
```

SYSTEMS' MANUAL

Page: 88
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 GSTR1 = IPOINT(19, IELEM )
J GSTRN = 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 EQCSI = IPOINT(27, IELEM )
J OMEGT = IPOINT(28, IELEM )
J OMEGI = IPOINT(29, IELEM )
J TSWL = IPOINT(30, IELEM )
J SWELL = IPOINT(31, IELEM )
J TDSTR = IPOINT(32, IELEM )
J YIELD = IPOINT(34, IELEM )

**********************************************************************
PATCHY CODE FOR POINTERS ASSOCIATED WITH THE ELEMENT BUFFER
**********************************************************************
C
KEQCSI = KEQPST
KSWINO = KPSINO
KIONO = KCIINO

**********************************************************************
WORKSPACE ALLOCATION FOR THE CURRENT SUBELEMENT SOLUTION
**********************************************************************
C
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, KSIZE, KSIZE, JELEPS, JELIMP, JTDWK, JELSTR, IEBETA, IELDET,
5 JELGID, JREACT, JRESID)

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

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

CALL NUL( _ORK(JLUMPM) , KSUBNP )

DO 4900 ISUBEL = 1 , KSUBEL

CALL SNODEL
1 ( ISUBEL , IWORK (JCONNECT), RWORK (JPCORD), RWORK (JDMATX),
2 WORKSP (IELCOR), WORKSP (IELDMT), KSUBEL , KSUBNP ,
3 NSUNOD , NSUSTR , NSUCRD , NSUNFR ,
4 RWORK (JDISPI), RWORK (JDISPL), WORKSP (IELIDS), WORKSP (IELDMT),
5 KCONN , JCONNECT , RWORK (JTDSTR), WORKSP (IELGTD),
6 RWORK (JGSTR), WORKSP (IELSTR))

CALL IMPMAS
1 ( RWORK (JLUMPM), WORKSP (IELDET), IWORK (KCONNECT), NSUNOD ,
2 NSUINT, KSUBNP , KSUBTY , RWORK (ICCHAR),
3 WORKSP (IELCOR), NSUCRD, ISUBEL , KSUBEL ,
4 RWORK (JPCORD), NSUCRD, CTRANS, JGRAM )

CONTINUE
CONTINUE

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

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

MIXED ITERATIVE SOLUTION FOR THE SUBELEMENT GRID

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

***************
(I) TOTAL NUMBER OF ITERATIONS AND CONVERGENCE TOLERANCE ARE SAME AS
THOSE DEFINED FOR THE GLOBAL MESH

(II) IN CASE OF NO CONVERGENCE ACHIEVED (IF NOT DIVERGE), THIS
ROUTINE RETURNS NORMALLY TO THE MAIN FINITE ELEMENT SOLUTION
WITH A LITTLE WARNING MESSAGE

******************************************************************************
PHASE 1 : NODAL STRESS / MATERIAL TANGENT CALCULATION FOR THE
DISPLACEMENT PRECONDITIONING
******************************************************************************

\[ N3\text{HARD} = 3 \times NH\text{ARD} \]

CALL SUBINT
1 (RWORK (JECORD),NELCRD, ,KSUBNP ,RWORK (KYSPEL)),
2  N3\text{HARD} ,NELNOD ,RWORK (KYSPEL))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (IXIRL)),
2  NSUNFR ,NELNOD ,RWORK (JDISPL))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KIGENO)),
2  NSUCMP ,NELNOD ,RWORK (JSTRNI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KICSNR)),
2  NSUCMP ,NELNOD ,RWORK (JCREPI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KITSNO)),
2  NSUCMP ,NELNOD ,RWORK (JTHIRI))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KISTRS)),
2  NSUCHR ,NELNOD ,WORKSP (JSUCHR))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KIGSNR)),
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 (KIOINO)),
2  NELSTR ,NELNOD ,RWORK (JOME))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KSWLNO)),
2  1 ,NELNOD ,RWORK (JSWELL))
CALL SUBINT
1 (RWORK (JECORD),NELCRD ,KSUBNP ,RWORK (KIMINO)),
2  1 ,NELNOD ,RWORK (JTEMPS))
CALL SUBINT
1 (RWORK (JECORD), NELCRD, KSUBNP, RWORK (KTDFNO),
2 1 , NELNOD, RWORK (JMPDF))
CALL SUBT
1 (WORKSP (JTDWRK), RWORK (JGSTR), RWORK (JTDSTR), KSUBNP*NSUSSTR )
CALL STRESS
1 (RWORK (JSTRAN), RWORK (JSTRI), RWORK (JSTRT), RWORK (JSTRST),
2 RWORK (JCREPT), RWORK (JCREPI), RWORK (JTPLAS), RWORK (JPLASI),
3 RWORK (JITHERT), RWORK (JITHERI), RWORK (JTEMPR), WORKSP (JSUCHR),
4 NSUSTR , NSULAY , NSUCHR , NSUNOD ,
5 KSUBNP , NSUNI , NSUSHR , JLAW ,
6 RWORK (JEQPST), RWORK (JEQPSI), KLAW , RWORK (JDMATX),
7 JMATRIX , JTEMP , JCREEP , JITER ,
8 JINC , NONISO , RWORK (JMPDF), RWORK (JEQCST),
9 RWORK (JTVSWL), RWORK (JTDSTR), JCONST , RWORK (JOMEGI),
+ RWORK (JOMEGI), ITHERM , TIMINC , RUNITM ,
1 RWORK (JGSTR), RWORK (JGSTRN), RWORK (JGSTRS), RWORK (JGSTR),
2 NSUSTR , NSUCMP , LBACK , RWORK (JSWELL),
3 RWORK (JEQCSI), RWORK (JPCORD), NSUCRD , JPLAST ,
4 ICOMPS , RWORK (IPREF), RWORK (JYIELD), NHARD ,
5 JUHOK , 1 , JWKLSP )
CALL ADD
1 (RWORK (JTDSTR), RWORK (JTDSTR), WORKSP (JTDWRK), NSUSTR*KSUBNP )

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

DO 8000 ISITER = 1 , NSTOPS
MCITER = ISITER - 1

C ******************************************************************************
C FIRST ELEMENT LOOP : STIFFNESS MATRIX ASSEMBLY
C ******************************************************************************
C
IF ( ISITER .GT. 1 ) GO TO 5001

DO 5000 ISUBEL = 1 , KSUBEL
IASSEM = 1

CALL SUBELV
CALL SNODE
   ( ISUBEL , IWORK ( JCONNC ) , RWORK ( JCOORD ) , RWORK ( JDMATX ) ,
   WORKSP ( IELCOR ) , WORKSP ( IELMT ) , KSUBEL , KSUBNP ,
   NSUNOD , NSUSTR , NSUCRD , NSUNFR ,
   RWORK ( JDISPI ) , RWORK ( JDISPL ) , WORKSP ( IELIDS ) , WORKSP ( IELTDS ) ,
   KCONNC , JCONNC , RWORK ( JDISIR ) , WORKSP ( IELGID ) ,
   RWORK ( JGSTR ) , WORKSP ( IELSTR ) )

CALL SUBDER
   ( WORKSP ( IEBETA ) , WORKSP ( IELDET ) , WORKSP ( IELCOR ) , WORKSP ( IELTDS ) ,
   WORKSP ( IELIDS ) , RWORK ( ICHAR ) , IGAUS , ITRANS ,
   CTRANS )

CALL INTERP
   ( WORKSP ( JELGID ) , WORKSP ( IELMT ) , NSUINT ,
   NSUNOD )

CALL INTERP
   ( WORKSP ( JELDMT ) , WORKSP ( IELMT ) , NSUINT ,
   NSUNOD ,
   NSUSTR*NSUSTR )

IF( ISITER .EQ. 1 ) CALL STIFF
   ( WORKSP ( IELSTF ) , WORKSP ( ILOAD ) , WORKSP ( JELGID ) , WORKSP ( IEBETA ) ,
   WORKSP ( IELDET ) , NSUSTR , NSUINT , NSULV ,
   NSUIDF , MCITER , WORKSP ( JELDMT ) , 1 ,
   1 , 'BACK' , 0 , 0 )

IF( ISITER .EQ. 1 ) CALL SYSEQN
   ( WORKSP ( ISTIFF ) , WORKSP ( ILOADV ) , WORKSP ( IELSTF ) , WORKSP ( ILOAD ) ,
   IWORK ( JCONNC ) , NSUNOD , NSUNFR , KSUBNP ,
   KSUBEL , KSIZE , KESIZE , ISUBEL )

C 5000 5001 CONTINUE

C CALL SUBSOL
   ( WORKSP ( ISTIFF ) , WORKSP ( ILOADV ) , KSIZE ,
   RWORK ( JDMIS ) ,
   NSUNFR , KSUBNP , RWORK ( JDISPI ) , NELNFR ,
   NELNOD , NSUNOD , KEMBED , ISITER ,
   WORKSP ( JREACT ) , KSUBEL )

CALL ADD
   * CALL PJOP
   * 1 ( RWORK ( JDISPI ) , 'DISINC' , NSUNFR*KSUBNP )
   * CALL PJOP
   * 1 ( RWORK ( JDISPC ) , 'DISCOR' , NSUNFR*KSUBNP )

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

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

DO 5100 ISUBEL = 1 , KSUBEL

IASSEM = 0

CALL SUBELV
1 (IWORK , KSUBTY , IERR)

CALL SNOVEL
1 (ISUBEL , IWORK (JCONNCC), RWORK (JPORD), RWORK (JDMATX),
2 WORKSP (IELCOR), WORKSP (IELMDT), KSUBEL , KSUBNP ,
3 NSUNOD , NSUSTR , NSUCRD , NSUNFR ,
4 RWORK (JDISPL), RWORK (JDISPL), WORKSP (IELIDS), WORKSP (IELIDS),
5 KCONNECT , KCONNECT , RWORK (JIDSTR), WORKSP (IELGID),
6 RWORK (JGSTRI), WORKSP (IELSTR))

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

CALL STRAIN
1 (RWORK (JGSTNI), WORKSP (JELEPS), WORKSP (IELIDS), WORKSP (IEBETA),
2 NSUSTR , NSUINT , NSUNFR , NSUNOD ,
3 NSUSTR , NSUSTR , RWORK (JLMPM), WORKSP (IELDET),
4 RWORK (KCONNECT), JGAUS , WORKSP (JELMPE), JTEMP ,
5 RWORK (JTHEI), RWORK (ICCHAR), NSUCHR , JAW ,
6 NSUSTR , KSUBTY , WORKSP (IELCOR), NSUCHR ,
7 CTRANS , RWORK (JSTRI), NSUSTR , NSUCMP ,
8 WORKSP , WORKSP , RWORK (JCREPT), RWORK (JTHEI),
9 JSTRN , JCITER )

CONTINUE

CALL STRESS
1 (RWORK (JSTRAIN), RWORK (JSTRI), RWORK (JSTRT), RWORK (JSTRT),
2 RWORK (JCREPT), RWORK (JCREPT), RWORK (JPLAS), RWORK (JPLAS),
3 RWORK (JTHEI), RWORK (JTHEI), RWORK (JTHEI), WORKSP (JSTOR),
4 NSUSTR , NSUSTR , NSUSTR , NSUCHR ,
5 KSUBNP , NSUSTR , NSUCHR , JAW ,
6 RWORK (JEQPST), RWORK (JEQPSI), KLAW , RWORK (JDMATX),
C

DO 5200 ISUBEL = 1 , KSUBEL

CALL SNODEL
   (ISUBEL , IW (JCONNC) , RWORK (JPCORD) , RWORK (JDMATX) ,
   WORKSP (IELCOR) , WORKSP (IELMT) , KSUBEL , KSUBNP ,
   NSUNOD , NSUSTR , NSUCRD , NSUNFR ,
   RWORK (JDISPI) , RWORK (JDISPL) , WORKSP (IELIDS) , WORKSP (IELIDS) ,
   KCONNC , JCONNC , RWORK (JTDSTR) , WORKSP (JELGID) ,
   RWORK (JGSTRN) , WORKSP (JELSTR) )

CALL SUBDER
   (WORKSP (JELGID) , WORKSP (JELGID) , NSUINT ,
   WORKSP (JELSTR) , WORKSP (JELSTR) , NSUINT ,
   CALL INTERP
      (ISUBEL , WORKSP (IELLOAD) , WORKSP (JELGID) , WORKSP (JELSTR) ,
      IW (KCONNC) , NSUNOD , NSUSTR , NSUNFR ,
      NSUINT , WORKSP (IELLOAD) , WORKSP (JEBETA) , KSIZE ,
      KSUBNP , RWORK (JDISPI) , KSIZE , INCNVG ,
      CTOLER , WORKSP (IELDET) , IW (JCONNC) , ILPRINT ,
      ICONSL , KSUBEL , McITER , ENGTOT ,
      WORKSP (JREACT) , WORKSP (JRESID) )

C

CONTINUE

CALL SUBCHK
   (RWORK (JDISPC) , RWORK (JDISPI) , KSIZE ,
   ISITER , INCNVG , )

C

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

S _$'r_MS ' MANUAL MHOST Version 4.2

Page : 95
C CHECK THE DISPLACEMENT CONVERGENCE
C******************************************************************************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
C 8100   CONTINUE
C
CALL SUBINT
1  (RWORK (JECOND),NELCRD ,KSUBNP ,RWORK (KIGSNO),
2  NELSTR ,NELNOD ,RWORK (JGSTRI))
IF( JSUBRE .EQ. 0 ) CALL SUBGLB
1  (IC ,RWORK (IXP) ,RWORK (JSTRST),RWORK (JSTRI),
2  IWORK (JCONN),NSUNOD ,NSUSTR ,NSUNFR ,
3  NSUINT ,IDF ,WORKSP ,RWORK (IBETA) ,
4  RWORK (JECOND),RWORK (ICOR) ,RWORK (JLUMP),NELSTR ,
5  NSUCRD ,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 (JSTRST),RWORK (JSTRI),
2  IWORK (JCONN),NSUNOD ,NSUSTR ,NSUNFR ,
3  NSUINT ,IDF ,RWORK (KIGSNO),RWORK (IBETA) ,
4  RWORK (JECOND),RWORK (ICOR) ,RWORK (JLUMP),NELSTR ,
5  NSUCRD ,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 ,WORKSP ,RWORK (JGSTRI),WORKSP(JTDWRK) )

C******************************************************************************C
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.

IWINOD  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.

IISTNO  Pointer to the incremental nodal stress array.

IISNNO  Pointer to the incremental nodal strain array.
**IIPSN0**  Pointer to the incremental nodal plastic strain array.

**IICSNO**  Pointer to the incremental nodal creep strain array.

**IITSNO**  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.
IXIRC  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.
KPSINO   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.
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.

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 IRSTRT. 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 SUBROUTINE KEY
C     SUBROUTINE KEY(NAME,NOPT,IOPT,NN,IN,INERR)
C
C* * * * * *
C
C CHECKS STRING FOR KEYWORD
C
C NAME KEYWORD NAMES
C NOPT NUMBER OF KEYWORDS
C IOPT KEYWORD FLAG
C     INTEGER PARAMETERS
C     NUMBER OF INTEGERS
C     ERROR FLAG
C
C* * * * * *
C
C     IMPLICIT REAL*8 (A-H,O-Z)
C
C     DIMENSION NAME(1),NN(4)
C     COMMON / ALGEM / IREAD,ILPRNT,JLPRNT,ICONSL,IPOSTF,ISCRAF,
C                   IPOINT,IRSTRT,JCREAD,IPVARS,IPSETS,IFILEX,
C               1     PI ,LINE ,LINE2
C     COMMON/FREE/IA(80),IBEGIN(16),ILEN(16),NSTRIN,IS,ICOL,NEW
C     LOGICAL NEW
1 CONTINUE
     JKEY=1
     CALL STRING(IERR,JKEY,1)
     IS=IS+1
     IB=IBEGIN(IS)
     IL=ILEN(IS)
     IF(IL.LT.3) GOTO 99
     IF(IL.GT.4) IL=4
     I1=4
     IB1=IB-1
     DO 3 I=I,NOPT
       I1=I1+4
     DO 2 J=1,IL
       IF(IA(IB1+J).NE.NAME(I1+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=ILEN(IS)
       CALL DECINT(NNI,IA(IB),IL,IERR)
       NNI(I)=NNI
5 CONTINUE
     GOTO 100
C
CONTINUE
IERR=IERR+1
CALL LINES(1,1)
L=IB-1+IL
WRITE(ILPRNT,1001) (IA(I),I=IB,IL)

FORMAT(2X,11H***ERROR***,5X,25HSTRING IS NOT A KEYWORD: ,78A1)
GOTO 1

CONTINUE
RETURN
END

SUBROUTINE FREFOR(INIW,REAW,NINT,NREA,NVAR,IERR,JKEY)

FREE FORMAT ROUTINE

INTEGER WORKSPACE
REAL WORKSPACE
NUMBER OF INTEGERS
NUMBER OF REALS
ERROR FLAG
KEY

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

COMMON / ALGEM / ICREAD,ILPRNT,ICONS,L,IOSTF,ISCRAF,
1 IPLOTB,IRSTRT,JCREAD,IPVARS,IPSETS,IPLEX,
2 PI ,LINE ,LINE2
COMMON/FREE/IA(80),IBEGIN(16),ILEN(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

IF(NINT.LE.0) GOTO 10
DO 1 I=1,NINT
IF(ICOL.LE.76) GOTO 2

Page : 107
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(IDUM,IA(IB),IL,IERR)
INIW(IC)=IDUM
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
M10T=IABS(NVAR)
IF(ICOL.LE.76) GOTO 23
JKEY=-1
CALL STRING(IERR,JKEY,1)
IF(JKEY.EQ.1) GOTO 30
JKEY=0
23 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.LREA)) GOTO 24
JKEY=-1
CALL STRING(IERR,JKEY,1)
IF(JKEY.EQ.1) GO TO 30
JKEY=0
21 CONTINUE
IC=IC+1
IS=IS+1
ICOL=ICOL+5
IB=IBEGIN(IS)
IL=ILENGT(IS)
IF(LREA) 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)
C
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 'H O S T' OR 'F E M'
C
SUBROUTINE DATINI
1 (RWORK, IWORK, ISIZE ,VERSNO, MONTH, JDATE, NELEM, NNODE ,
2 NBC , NTIE , NMAX , NTRAN , NTRAC , NPOST, NLVSUB, NFRSUB,
C READ THE CONTROL DATA AND PARAMETERS ASSOCIATED WITH THE CORE STORAGE ALLOCATION

C IMPLICIT REAL*8 ( A-H, O-Z )
REAL*4 RWORK
C
C 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 JPEROD ( 2 )
C
EQUIVALENCE ( NAME( 1, 1 ), NAME1( 1, 1 ) )
EQUIVALENCE ( NAME( 1, 35 ), NAME2( 1, 1 ) )
EQUIVALENCE ( NAME( 1, 71 ), NAME3( 1, 1 ) )
C
COMMON / ALGEM / IREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
   1 IPLOTB, IRSTRT, JCREAD, IPVARS, IPSETS, IFILEX,
   2 PI , LINE , LINE2
COMMON / COUNT / LININC, LINTOT, NOECHO
COMMON / CTITLE / TITLE ( 20 ), IDAT ( 5 ), IDATE2, ICLOCK,
* IFCRAY

COMMON / ERRORS / IERR
COMMON / FREE / IA (80), IBEGIN(16), ILENGTH(16),
NSTRIN, IS, ICOL, NEW

LOGICAL NEW

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

DATA NAME1
*
/1HE, 1HL, 1HE, 1HM, 1HN, 1HO, 1HD, 1HE, 1HB, 1HO, 1HU, 1HN,
1HT, 1HY, 1HI, 1HN, 1HT, 1HR, 1HA, 1HN, 1HF, 1HO, 1HR, 1HC,
1HP, 1HO, 1HS, 1HT, 1HS, 1HU, 1HB, 1HS,
1HE, 1HX, 1HT, 1HE, 1HP, 1HR, 1HE, 1HS, 1HT, 1HE, 1HM, 1HP,
1HP, 1HR, 1HT, 1HN, 1HR, 1HE, 1HS, 1HT, 1HL, 1HO, 1HU, 1HB,
1HS, 1HT, 1HR, 1HE, 1HE, 1HN, 1HD, 1H,
1HT, 1HE, 1HS, 1HT, 1HD, 1HY, 1HN, 1HA, 1HO, 1HP, 1HT, 1HT,
1HT, 1HR, 1HA, 1HC, 1HC, 1HR, 1HE, 1HE, 1HA, 1HN, 1HI, 1HS,
1HM, 1HO, 1HD, 1HA, 1HB, 1HU, 1HC, 1HK, 1HT, 1HH, 1HE, 1HR,
1HC, 1HO, 1HN, 1HS, 1HD, 1HI, 1HS, 1HT, 1HD, 1HU, 1HP, 1HL,
1HR, 1HE, 1HO, 1HO, 1HT, 1HA, 1HN, 1HG, 1HU, 1HT, 1HH, 1HE,
1HS, 1HC, 1HH, 1HE, 1HU, 1HF, 1HO, 1HR, 1HU, 1HT, 1HE, 1HM/

DATA NAME2
*
/1HU, 1HC, 1HO, 1HE, 1HU, 1HP, 1HR, 1HE, 1HU, 1HH, 1HO, 1HO,
1HU, 1HD, 1HE, 1HR, 1HU, 1HB, 1HO, 1HU, 1HP, 1HE, 1HR, 1HI,
1HB, 1HA, 1HN, 1HD, 1HP, 1HR, 1HO, 1HN, 1HD, 1HE, 1HF, 1HO,
1HE, 1HM, 1HB, 1HE, 1HG, 1LM, 1HR, 1HS, 1HB, 1HE, 1HA, 1HM,
1HD, 1HT, 1HS, 1HE, 1HS, 1HH, 1HT, 1HF, 1HG, 1HS,
1HS, 1HP, 1HR, 1HT, 1HD, 1HA, 1HS, 1HM, 1HA, 1HS, 1HS,
1HS, 1HE, 1HC, 1HA, 1HL, 1HI, 1HN, 1HE, 1HN, 1HA, 1HR, 1HM,
1HB, 1HA, 1HS, 1HE, 1HC, 1HO, 1HM, 1HP, 1HU, 1HL, 1HS,
1HC, 1HO, 1HN, 1HU, 1HS, 1HH, 1HO, 1KC, 1HP, 1HO, 1HW, 1HE,
1HN, 1HO, 1HE, 1HC, 1HP, 1HE, 1HR, 1HT, 1HS, 1HT, 1HI, 1HF,
1HC, 1HE, 1HN, 1HT, 1HH, 1HA, 1HR, 1HD, 1HF, 1HI, 1HN, 1HI,
1HL, 1HA, 1HR, 1HG, 1HF, 1HO, 1HL, 1HL, 1HW, 1HK, 1HS, 1HL/

DATA NAME3
*
/1HH, 1HO, 1HU, 1HR, 1HM, 1HO, 1HN, 1HI/

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

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 TYIN FLAG THE TYING OPTION WITH NUMBER OF TYING

SYSTEMS' MANUAL

Page : 111
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

*SUES 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)

*DYNA INVOKE TRANSIENT TIME INTEGRATION

*OPTI FLAG THE 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

*REPORT REPORT GENERATION INTERVAL

*TANG MODIFIED NEWTON OPTION

*UHE USER SUBROUTINE 'UHERM' OPTION

*SCHE TIME INTEGRATION SCHEME OPTION

*UFOR USER SUBROUTINE 'UFORCE' OPTION

*UTEM USER SUBROUTINE 'UTEMP' OPTION

*UCOE USER SUBROUTINE 'UCOEFF' OPTION

*UPRE USER SUBROUTINE 'UPRESS' OPTION

*UHOO USER SUBROUTINE 'UHOOK' OPTION

*UDER USER SUBROUTINE 'UDERIV' OPTION

*UCOU USER SUBROUTINE 'UCOINC' OPTION

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

*PRON FRONTAL SOLUTION SUBSYSTEM (OPTIONAL)

*DEFO EIGENVALUE EXTRACTION FOR THE STIFFNESS

*EMBE SUBELEMENT MESH ANALYSIS OPTION

*GMRS MULTIPLE GENERIC MODELLING REGIONS OPTION
*BEAM BEAM SECTION PARAMETER OPTION
*DISP CONVENTIONAL DISPLACEMENT METHOD
*SHIF POWER SHIFT FOR EIGEN EXTRACTION
*BEFGS BFGS UPDATE FOR THE NONLINEAR SOLUTION
*SPRI ADDED STIFFNESS, GROUND SPRING
*DASH ADDED DAMPING, DASHPOT TO GROUND
*MMASS 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
*WKSU USER SUBROUTINE 'WKLSP'
*HOUR (GLASS CONTROL) AS THE NAME INDICATES ....
*MONI INVOICES MONITOR UTILITY

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

NSHIFT = 0
JPOST = 0
NSPRI = 0
NDASH = 0
NMASS = 0
NHARM = 0
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NBASE</td>
<td>0</td>
</tr>
<tr>
<td>NTIE</td>
<td>0</td>
</tr>
<tr>
<td>NDUP</td>
<td>0</td>
</tr>
<tr>
<td>JEMBED</td>
<td>0</td>
</tr>
<tr>
<td>ITERM</td>
<td>0</td>
</tr>
<tr>
<td>NCREEP</td>
<td>1</td>
</tr>
<tr>
<td>JEMBED</td>
<td>0</td>
</tr>
<tr>
<td>JDISP</td>
<td>0</td>
</tr>
<tr>
<td>JHRGLS</td>
<td>0</td>
</tr>
<tr>
<td>JWKS LP</td>
<td>0</td>
</tr>
<tr>
<td>JISTRN</td>
<td>0</td>
</tr>
<tr>
<td>JCITER</td>
<td>0</td>
</tr>
<tr>
<td>JREPOT</td>
<td>1</td>
</tr>
<tr>
<td>ISTAT</td>
<td>1</td>
</tr>
<tr>
<td>NSBF GS</td>
<td>0</td>
</tr>
<tr>
<td>IDYNM</td>
<td>0</td>
</tr>
<tr>
<td>NGRMS</td>
<td>1</td>
</tr>
<tr>
<td>IPCONJ</td>
<td>0</td>
</tr>
<tr>
<td>JSUB</td>
<td>0</td>
</tr>
<tr>
<td>NSUB</td>
<td>0</td>
</tr>
<tr>
<td>JFRONT</td>
<td>0</td>
</tr>
<tr>
<td>JREST</td>
<td>0</td>
</tr>
<tr>
<td>JCREEP</td>
<td>0</td>
</tr>
<tr>
<td>JTEMP</td>
<td>0</td>
</tr>
<tr>
<td>NEXT</td>
<td>0</td>
</tr>
<tr>
<td>JUBOUN</td>
<td>0</td>
</tr>
<tr>
<td>NONISO</td>
<td>0</td>
</tr>
<tr>
<td>IFBF GS</td>
<td>0</td>
</tr>
<tr>
<td>IFSCNT</td>
<td>0</td>
</tr>
<tr>
<td>IFLINE</td>
<td>0</td>
</tr>
<tr>
<td>NDYNMD</td>
<td>0</td>
</tr>
<tr>
<td>IDYNMD</td>
<td>100000</td>
</tr>
<tr>
<td>NBGMD</td>
<td>0</td>
</tr>
<tr>
<td>JTERM</td>
<td>0</td>
</tr>
<tr>
<td>JCONST</td>
<td>2</td>
</tr>
<tr>
<td>JDYN</td>
<td>0</td>
</tr>
<tr>
<td>JEIGEN</td>
<td>0</td>
</tr>
<tr>
<td>JDEFOR</td>
<td>0</td>
</tr>
<tr>
<td>NBSECT</td>
<td>0</td>
</tr>
<tr>
<td>JFORCE</td>
<td>0</td>
</tr>
<tr>
<td>JPEROD(1)</td>
<td>0</td>
</tr>
<tr>
<td>JPEROD(2)</td>
<td>0</td>
</tr>
<tr>
<td>JUTEMP</td>
<td>0</td>
</tr>
<tr>
<td>JUCEOF</td>
<td>0</td>
</tr>
<tr>
<td>JDISTS</td>
<td>0</td>
</tr>
</tbody>
</table>
C READ TITLE CARD AND PRINT THE USUAL PROBLEM HEADER
C
C READ(ICREAD, 1000, END=3001) TITLE
1000 FORMAT (20A4)
C
C CALL HEAD (VERSNO, MONTH, JDATE, ILPRNT, ICONSL)
C CALL HEADER (VERSNO, MONTH, JDATE, ILPRNT, ICONSL, 2)
C CALL LINES(70,0)
C WRITE(ILPRNT, 1001) TITLE
1001 FORMAT (10X, 20A4)
C
C NEW = .TRUE.
C
C continue 
C *** KEY-WORD INTERPRETER ****************************
C CALL KEY( NAME, NOPT, IOPT, NN, 6, IERR )
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, 48, 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

!] WORK , !WORK , IERR , NTYPE , ILAST , IPRNT , NDIMEN)

GO TO 998

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

2 CONTINUE

NNODE = NN(1)

GO TO 998

OPTION 3 : *BOUN - MAXIMUM NUMBER OF BOUNDARY CONDITIONS

3 CONTINUE

NBC = NN(1)

GO TO 998

OPTION 4 : *TYIN - 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 **********************************************************************
C OPTION 6 : *FORC - MAXIMUM NUMBER OF NODAL FORCE ENTRIES
C **********************************************************************

6 CONTINUE
NTRAC = NN(1)
GO TO 998

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

9 CONTINUE
NEXT = NN(1)
GO TO 998

C****************************************************************************
C        OPTION 10 : *PRES - NODAL PRESSURE DEFINITION
C****************************************************************************
C
10 CONTINUE
    JPRES = 1
    GO TO 998

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

C****************************************************************************
C        OPTION 12 : *PRIN - INCREASE THE NUMBER OF PRINT OPTIONS
C****************************************************************************
C
12 CONTINUE
    IF (NN(1).LT.0) IFPRNT = 1
    IF (NN(1).LT.0) NPRINT = IABS(NN(1))
    IF (NN(1).GT.0) NPRINT = NN(1)
    GO TO 998

C****************************************************************************
C        OPTION 13 : *REST - RESTART TAPE FLAG TO BE SET
C****************************************************************************
C
13 CONTINUE
    JREST = 1
C+    JINC = NN(1)
C+    NINC = NN(2)

C *** EXIT IMMEDIATELY WITHOUT READING *END CARD **********************
C
    GO TO 16

C****************************************************************************
C        OPTION 14 : *LOUB - SELECTORS FOR THE NUMERICAL QUADRATURES
C****************************************************************************
C
SYSTEMS' MANUAL

Page : 118
14 CONTINUE
    JLOUB = 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
    NSTBRC = 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

MHOST Version 4.2
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**********************************************************************
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**********************************************************************
C
20 CONTINUE
JDIST = 1
GO TO 998

C**********************************************************************
C OPTION 21 : *CREE - CREEP AND ITS TIME STEP CONTROL PARAMETERS
C**********************************************************************
C
21 CONTINUE
JCREEP = 1
NCREEP = 3
ATOLER = 0.5D0
BITOLER = 0.5D-1
CTOLER = 0.5D-1

C
IF( NN(1) .EQ. 0 ) GO TO 2101
C
NCREEP = NN(1)
C
CALL FREFOR
1 ( IWORK(ILAST+1), RWORK(ILAST+1), 0, 3, 0, IERR, JKEY )
C
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), BITOLER, 1 )
J = ILAST + 3

IF( RWORK(J) .NE. 0.0 ) CALL COPYSD ( RWORK(J), CTOLER, 1 )

2101 CONTINUE
   GO TO 998

C ************************************************************
C OPTION 21 : *ANIS - ANISOTROPIC ELASTICITY
C ************************************************************
C 22 CONTINUE
   NONISO=1
   GO TO 998

C ************************************************************
C OPTION 23 : *MODAL - MODAL ANALYSIS OPTION AND PARAMETER SET
C *************************************************************
C 23 NDYNMD=NN(1)
   NSBNC =NN(2)
   INTSTR=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 NULINP(NN,4)
   JKEY = 0
   CALL FREFOR(NN,NN,1,0,0,IER,JKEY)
   IF ( JKEY .EQ. 1 ) GO TO 998
   NSUPER = NN(1)
   LDYN = 2
   GO TO 998

C *****************************************************
C OPTION 24 : *BUCK - BUCKLING ANALYSIS AND PARAMETERS
C *****************************************************
C 24 NDYNMD = NN(1)
NSBNC = NN(2)
INTSTR = 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
******************************************
C OPTION 25 : *THER - TEMPERATURE DEPENDENT PROPERTIES
C ******************************************
C
25 CONTINUE
   IThERM = 1
   GO TO 998
C
******************************************
C OPTION 25 : *CONS - CONSTITUTIVE LAW SELECTION
C ******************************************
C
26 CONTINUE
   JCONST = NN(1)
   IF ( JCONST .LT. 0 .OR. JCONST .GT. 4 )
      JCONST = 2
   GO TO 998
C
******************************************
C OPTION 27 : *DIST - DISTRIBUTED LOAD ( SAME AS *TRAC AND *PRES )
C ******************************************
C
27 CONTINUE
   JDIST = 1
   GO TO 998
C
******************************************
C OPTION 28 : *DUPL - MAXIMUM NUMBER OF DUPLICATED NODES
C ******************************************
C
28 CONTINUE
   NDUP = NN(1)
   GO TO 998
OPTION 29: *REPO - REPORT GENERATION INTERVAL TO BE SET

29 CONTINUE
   JREPORT = NN(1)
   IF(NN(1).eq.0) JREPORT=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: *SCHE - TIME STEPPING SCHEME PARAMETER OPTION

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

OPTIONS 33, 34, 35, 36, 37, 38, 39 - FLAGS FOR USER SUBROUTINES
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  ) = N(  1  )
JPEROD(  2  ) = N(  2  )
GO TO 998

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

41 CONTINUE
JBAND = 1
JFRONT = 0
GO TO 998

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

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

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

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

MSBNC = 8 + NDYNMD
IF ( NDYNMD .GT. 8 ) NSBNC = MSBNC

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

C **********************************************************************
C OPTION 44 : *EMBE - EMBEDDED MULTIPLE SINGULARITIES
C **********************************************************************

44 CONTINUE
JEMBED = 1

IF( NN( 1 ) .NE. 0 ) JSUBRE = 1

GO TO 998

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

45 CONTINUE
NGMRS = NN( 1 )
IF( NGMRS .EQ. 0 ) NGMRS = 1
GO TO 998
C
C **************************************************************
C OPTION 46 : *BEAM - MAXIMUM NUMBER OF BEAM SECTION DATA
C **************************************************************
C
46 CONTINUE
   NSHIFT = 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 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 OPTION 52: *MASS - ADDED MASS
C ...........................................................................................................
C 52 CONTINUE
NMASS = NN(1)
IF ( NMASS .EQ. 0 ) NMASS = 1
GO TO 998
C
C ...........................................................................................................
C OPTION 53: *SECA - SECANT NEWTON METHOD OPTION
C ...........................................................................................................
C 53 CONTINUE
IFSCNT = 1
GO TO 998
C
C ...........................................................................................................
C OPTION 54: *LINE - LINE SEARCH OPTION
C ...........................................................................................................
C 54 CONTINUE
IFLINE = 1
GO TO 998
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),RWORK(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 ******************************************************

56 CONTINUE
NBASE = NN(1)
IF (NBASE .EQ. 0) NBASE = i
CALL FREFOR(IWORK(ILAST+I),R3RK(IIASI_I),0,1,0,IERR,JKEY)
CALL COPYSD(RWORK(ILAST+1),OMEGB,1)
GO TO 998

C ******************************************************
C OPTION 57 : *COMPOSITE LAMINATE OPTION
C ******************************************************

57 CONTINUE
ICOMPS = 1
CALL CMPPDF(IWORK,RORK,IERR,NTYPE,ILAST,ILPRINT)
GO TO 998

C ******************************************************
C OPTION 58 : *PULS - PULSE LOAD OPTION
C ******************************************************

58 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 ******************************************************

59 CONTINUE
IPCONJ = 1
IILINE = 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 : _PWR - 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

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

SYSTEMS' MANUAL

Page : 129

MHOST Version 4.2
OPTION 64 : *STIF - STRESS STIFFENING OPTION

64 CONTINUE
  JSTIF = NN(1)
  IF ( NN(1) .EQ. 0 ) JSTIF = 1
  GO TO 998

OPTION 65 : *CENT - CENTRIFUGAL MASS STIFFNESS OPTION

65 CONTINUE
  JCENIM = NN(1)
  GO TO 998

OPTION 66 : *HARD - WORK-HARDENING OPTION FOR PLASTICITY

66 CONTINUE
  NHARD = NN(1)
  IF ( NN(1) .EQ. 0 ) NHARD = 1
  GO TO 998

OPTION 67 : *FINIT - FINIT STRAIN OPTION

67 CONTINUE
  JFINIT = NN(1)
  GO TO 998

OPTION 68 : *LARG - LARGE DISPLACEMENTS AND ROTATIONS OPTION

68 CONTINUE
  JLARGE = NN(1)
  GO TO 998

OPTION 69 : *FOLL - FOLLOWER FORCE OPTION
CONTINUE
   IFBFGS = NN(1)
   GO TO 998

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

OPTION 70 : *WLSKP : FLAGS THE USER SUBROUTINE FOR WORKHARDENING
*****************************************************************************

70 CONTINUE
   JWKSLP = 1
   GO TO 998

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

OPTION 71 : *HOUR ... HOURGLASS CONTROL FLAG IN A SPECIAL WAY
*****************************************************************************

71 CONTINUE
   JHRGLS = 1
   GO TO 998

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

OPTION 72 : *MONI - TURN ON THE MONITOR UTILITY
*****************************************************************************

72 CONTINUE
   NMONIT = NN(1)
   IF (NMONIT .LT. 1) NMONIT = 1
   GO TO 998

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

*** NORMAL EXIT BY READING '*END' CARD *******************************

16 CONTINUE
   IF( LOECHO .NE. 0 ) NOECHO = 1

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

*** A FEW POSSIBLE CONTRADICTIONS IN PARAMETER DATA ARE CHECKED HERE

   IF( IFBFGS .EQ. 1 .AND. IPSCNT .EQ. 1 ) CALL QUIT
   1 ('CONT','RADI','CTIO','N DE','TECT','ED', 1 )

   IF( IFBFGS .EQ. 1 .AND. IPCONJ .EQ. 1 ) CALL QUIT
   1 ('CONT','RADI','CTIO','N DE','TECT','ED', 1 )

   IF( IPCONJ .EQ. 1 .AND. IPSCNT .EQ. 1 ) CALL QUIT
C

IF( JFINIT.LT.JLARGE ) CALL QUIT
& ( 'FIN', 'IST', 'ARTS', 'B4', 'LAR', 'GE', 0 )
C
IF( JLARGE.EQ.999999 ) THEN
IF( JFOLLOW.NE.999999 ) CALL QUIT
& ( 'FOL', 'BU', 'NO', 'LAR', 'G', ' ', 0 )
IF( JFINIT.NE.999999 ) CALL QUIT
& ( 'FIN', 'BU', 'NO', 'LAR', 'G', ' ', 0 )
ENDIF

C
IF( JLARGE.NE.999999 .AND. JISTIF.EQ.999999 ) CALL PRWARN
& ( 'LARGE DISPL OPTION WITHOUT INITIAL STRESS OPTION')
C
RETURN
C
*** CONTROL DATA ERROR
C
3001 CONTINUE
WRITE(ICONSL,9000)
CALL QUIT('INPU', 'T ', ' ', ' ', ' ', ' ', ' ', 1)
C
*** FORMAT STATEMENTS
C
6000 FORMAT(/,1X,23H***ERROR*** OPTION ,I2,3H *,A4,11H NOT ACTIVE)
9000 FORMAT(/,28H ********** ERROR **********,/14H NO INPUT FILE,//)
C
STOP
END

An example of a typical bulk data reader is the subprogram to read and store the nodal coordinates:
C
C=SUBROUTINE=COORIN CALLED BY SUBROUTINE 'DATIN2'
C
SUBROUTINE COORIN
1 (INT ,REA ,IERR ,NCRD ,ILAST ,ILPRNT,NNODE,INOD ,
2 MAXCRD,ICHECK)
C
C
READ IN COORDINATE DATA
C
IERR ERROR FLAG
C
NCRD NUMBER OF COORDINATES PER NODE
C     LAST WORD FOR INPUT
C     KW     OUTPUT DEVICE
C     NNODE  NUMBER OF NODES IN MESH
C     INOD   POINTER TO COORDINATES
C     MAXCRD MAXIMUM NUMBER OF COORDINATES PER NODE
C     ICHECK ENABLE/DISABLE NODAL THICKNESS CHECK
C
C******************************************************************************
C
IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 REAL
C******************************************************************************
C
COMMON / MACHIN / IDP
C******************************************************************************
C
      DIMENSION INT( 1 ), REAL( 1 )
C******************************************************************************
C
      IS1 = INOD - MAXCRD*IDP
      JKEY = -1
C
*** CHECK IF THE CORE IS ALLOCATED FOR THE COORDINATE DATA **************
C
      IF ( NNODE .GT. 0 ) GO TO 208
C
     CALL LINES( 1 , 1 )
     WRITE(ILPRNT,9108)
     9108 FORMAT(2X,48H***ERROR***
     1 5H DATA)
     IE = IERR + 1
C
*** CHECK THE NUMBER OF DATA ENTRIES *************************************
C
      208 CONTINUE
      IF ( NCRD .EQ. 0 ) NCRD = MAXCRD
C
*** MCRD: COUNTER OF THE NUMBER OF ACTUAL COORDINATE DATA ENTRIES ****
C
      MCRD = NCRD
      IF ( MAXCRD .EQ. 7 ) MCRD = 3
IF ( NCRD .LE. MAXCRD ) GO TO 607
CALL LINES( 1 , 1 )
WRITE(ILPRNT,9208) NCRD,MAXCRD
9208 FORMAT(2X,47H***ERROR*** \ NUMBER OF COORDINATE DIRECTIONS,I5, \ *13H GREATER THAN,I5)
       IERR = IERR + 1

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

607   ICOUNT = 0
       TNEW = 0.D0
       TOLD = 0.D0

608 CONTINUE
       ICOUNT = ICOUNT + 1

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

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

   IF ( JKEY .EQ. 1 ) GO TO 108

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

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

*** IF THICKNESS IS NOT GIVEN THE LAST NONZERO VALUE IS TAKEN ********

   IF ( TNEW .EQ. 0.D0 ) TNEW = TOLD

*** RANGE CHECK **********************************************

   IF ( K .GT. 0 .AND. K .LE. NNODE ) GO TO 308

*** ERROR MESSAGES **********************************************

   CALL LINES( 1 , 1 )
   IF ( K .LE. 0 ) WRITE(ILPRNT,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)
C                         IERR = IERR + 1

308 CONTINUE
C
C *** THICKNESS DATA CHECK FOR SHELL ELEMENTS **************************
C
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 ',
   1          'FIRST DATA LINE')                           IERR = IERR + 1
C
C *** ADRESSES 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)
C
   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 RESTRT to read the restart file:

C ... SUBROUTINE SAVER ... RESTART FILE WRITER
C
SUBROUTINE SAVER( RWORK, IWORK, ISIZE )
C
C **********************************************************************
C ** **
C ** WRITES A BINARY FILE FOR LATER RESTART **
C ** **
C **********************************************************************
C
IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 RDRK
C
DIMENSION IWORK(ISIZE), RWORK(ISIZE)
C
COMMON / ADDVAL / ISPRI, KSPRI, IDASH, KDASH, IMASS, KMASS
COMMON / ALGEM / ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF,
1 IPR07, IRSTRT, JCREAD, IIPVARS, IIPSET, IFILEX,
2 PI, LINE, LINE2
COMMON / AUTOIN / CURPER, TOTPER, ARCLEN, ATOLER, ETOLER, CTOLER,
1 JADAP, NCREEP, SCALE
COMMON / BODYFR / POINTS( 3, 2)
COMMON / BSECT / IBSECT, KBSECT
COMMON / CONTRO / JEND, JITERT, JTEMP, JPRINT, JP, JSUB,
1 JINC, JREST, JSAVE, JREDIM, JAUTO, JPOST,
2 JBACK, JOPTIM, JCREEP, JDIST, JCONST, JDYN,
3 NONISO, ITERM, ITIRG, IDYNM, JREPOT, JTANGE,
COMMON / COMND / NXSTAT, NXSOVL, NXINIG, NXMODL, NXBCKL, NXSUPR,
1  NXREQN, NXDUM1, NXDUM2, NXDUM3, NXDUM4
COMMON / COUNT / LININC, LINTOT, NOECHO
COMMON / DAMP / DAMPP(3)
COMMON / EIGEN / IGSCV, IGENS, IOMEG, IDENO, IDYNMD, ISTRT2,
1  IPIAR, IPTBR, IPTVED, IAMAD, IOMEGD
COMMON / MDSSUP / IFMOR0, IMDIS0, INVLO, IMFOR1, IMDIS1, IMVEL1
COMMON / ELEMEN / IC, IEL, IDF, JLAW, IASSEM, 1
1  JPERED, JCAIT, JEL009, JEL010, JEL011, JEL012
COMMON / HARMON / OMEGH, IHEM, KHARM, OMEGB, IBASE, KBASE,
1  ICFLF0, ICMFOR, ICMRES, ICHHFN, ICHHFN, ICBEKC,
2  ICBMAT
COMMON / INCOCON / FACTOR, INCFLG(21)
COMMON / MAXIMA / MAXCRD, MAXFR, MAXNOD, MAXSTR, MAXCHR, MAXPRS,
1  MAXLAV, MAXINT, MAXWRK, MAXLVL, NSUMAX, MAXCMP,
2  MAXBSP, MAXGMR, MAXTEM, MAXLWK, MAXDMT,
3  MAXFRN, MAXBET, MAXSET, MAXEAN, MAXORD,
4  MAX025, MAX026, MAX027, MAX028, MAX029, MAX030
COMMON / LOUBIN / JLQUB, JLINTER, JEXTRA, JWEIGH, JSUBRE, JISTRN,
1  JPERED, JHRLS, JGRAM, LOUB03, LOUB04, LOUB05
COMMON / PARAM / NTYPE, NELEM, NNOD, NBC, NTIE, NMAX,
1  NPRINT, NTRAC, NFD, NBAND, NEXT, NSUB,
2  NPSPTS, NFDPTS, NSBC, NSUPER, NPAR43, NPAR44, NPAR45,
3  NDYNMO, NSUPR, NBASE, NSIZE
4  NITER, NSBC, NSUPER, NPAR4, NPAR42, NPAR47, NPAR48
COMMON / PERPAR / IPTYPE(32), NPTYPE, NPVARS, NPSETS, JPERT,
1  NPAR43, NPAR44, NPAR45, NPAR46, NPAR47, NPAR48
COMMON / PERPRTR / IMAENS, ISTEY, IPOST, IMDATA, IVTYPE, IKIP, IREDEF,
1  IDINCO, IREA0, IRESO, IDGR, ISTIF0, IMASS0,
2  IFP013, IOMEGO, IOMEGP, IOMEGK, IETAK, IZETAK
COMMON / PERDAS / IXCOOR, IXCHAR, IXFORC, KXFORC, IXDIST, KKDIST
1  IXTEMP, JXTEM, IXBEAM, IXFVEC, IXSPI, KXSPI,
2  IXRES, IXPREF, IXP015, IXP016, IPWBE, IPWEND
COMMON / PERIOD / JPEROD(2), IPDIS, IPFORD, INDISP, INFORC
COMMON / POSTPN / IPINT, JPINT, KPOINT, NDATA, PRNTBF(6)
COMMON / POWER / IELPHI,IELINM,IEPSNO,ISIGNO,IHFN ,IHFC ,
1 IFBP ,ISPP ,ISFF ,ISQQ ,ICQQ ,ITNM ,
2 IPSF ,IPSD
COMMON / PREPAR / NRFPTS,NRFSDS,NRFLIN,NRFINT,NRFREA,NRFSEL, NESSUS
1 NRFCMP,NRFCDF,NRFSD2,IRFEND,IRFTP,JCOUNT, NESSUS
2 JCFLAG,JRFSEL,IRFDOM,NKPT,MORE, NESSUS
COMMON / PREPTR / IRFPTS,IRFIN,IRFMNR,IRFSDR,IRFSEL,IRFSET, NESSUS
1 IRFMNV,IRFSDV,IRFPTV,IRFVEC,IRFVAL,IRFNRN, NESSUS
2 IRFCOR,IRFID,IRFWRK, NESSUS
COMMON / PULSES / IPULSE,KPULSE,IDTIM,IPDFOR
COMMON / RESULT / MANVAR( 7 ),JPR ,ICOM1 ,NCOMP
COMMON / START1 / IELPRM,ITYP ,INEL ,ICHAR ,IPRES ,ISTR, 1
1 ISTRN ,ICOP ,IPRINT,IPPOST ,IDIST ,ILEAN ,
2 IPRES,IBNORM,IMONIT,IST116,IST117,IST118
COMMON / START2 / IMOD ,ITEM ,INLV ,IPOSU ,ITEMDF,IDUP
COMMON / START3 / IKBC ,ITI ,ITR ,ITRAN ,ITRAC ,IEXT ,
1 ISBC ,ISBCR
COMMON / START4 / IDINC ,IDTOT ,IFORCE ,IRESID ,IWIND ,ISIGNO,
1 IEPSNO,IPSTRN,ICSTRN,ITSTRN,ISTS,ISTRN,
2 IIATPY,IIATPH,IIATIR,IIATSM,IIATNO,IIATNO,
3 IIATNO,IIATNO,IIATNO,IIATNO,IIATNO,IIATNO,
4 IDMNO,IEQCSI,IMEQNO,ITESNO,IVSNO,
5 IDMNO,IEQCSI,IMEQNO,ITESNO,IVSNO,
6 IEQCSI,IPREF ,IDsX3 ,ITYELD,IDFINC,IDFTOT,
7 IST443,IST444,IST445,IST446,IST447,IST448
COMMON / START5 / IRL ,IREAC ,IES ,IAB ,IBQM ,ISR, 1
1 IBTLC ,ISKM ,ILAST ,IRLE ,IDINCP,FORIN,
2 IOP ,IDAM ,IMASMT,IDIAG ,ITRANR,ISUBPT,
3 IMASDI,IMASUP,IST521,IST522,IST523,IST524
COMMON / START6 / IELV ,ICOR ,ISIG ,IEPS ,IWIND ,ISIGNO,
1 IENOD ,IPET ,ICH ,IPP ,IXRL ,IXIRL ,
2 IXP ,IXK ,KIPSTR,IKSSTR,IKSSTR,IKSSTR,
3 KISTR,KISTR,KISTR,KISTR,KISTR,KISTR,
4 KISTR,KISTR,KISTR,KISTR,KISTR,KISTR,
5 IMASNO,IMNO ,IEQPSI,IEQPSI,IEQPSI,
6 KMAT ,KDMNO,KTDSNO,KTDSST,IXM ,IXC ,
7 IVELM ,IAMEL ,IMASL,KYIELD,IST647,IST648,
8 IST649,IST650,IST651,IST652,IST653,IST654
COMMON / START7 / ICT ,IKBCR ,ITRACR,ITRANR,IBETA ,IDET
COMMON / START8 / KEEPS ,KEEPS,KGSSIG ,KGSSIG,KGSSDST,
1 IGEPNO,IIGENO,IGSNO,IGSNO,IGSNO,
2 KGEPNO,KIGENO,KGSSNO,KGSSNO,KGSSNO
COMMON / START9 / KEQCSI,KMEMO,KSWMNO,KTMPSN,KIDFINO,KDUMMY,
1 KEQCSI,KMEMO,KVSNO,IST910,IST911,IST912
COMMON / SUBELM / ISUBEL,ISUBNP,ISUPTD,NSDATA,ISUBTY,ISUBEME

SYSTEMS' MANUAL

Page : 138

MHKHOST Version 4.2
** Rewind the restart file to overwrite with the newest data **

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

WRITE (IRSTRT) ISPRI, KSPI, IDASH, KDASH, IMASS, KMASS
WRITE (IRSTRT) ICREAD, IPRINT, JLPRINT, ICCONS, IPOSTF, ISCRAF
WRITE (IRSTRT) IPLOTB, IRSTRT, JCREAD, IPVARS, IPSETS, IFILEX
WRITE (IRSTRT) PI, LINE, LINE2
WRITE (IRSTRT) CURPER, TOTPER, ARCLEN, ATOLER, BTOLER, CTOLER
WRITE (IRSTRT) JADAP, NCREEP, SCALE
WRITE (IRSTRT) POINTS (3, 2)
WRITE (IRSTRT) IBSECT, KSEECT
WRITE (IRSTRT) JEND, JITEX, JTTEMP, JPRINT, JP, JSUB,
WRITE (IRSTRT) JINC, JREST, JSAVE, JREDIM, JALTO, JPOST,
WRITE (IRSTRT) JBACK, JOPTIM, JCREEP, JDIST, JCONST, JDYN,
WRITE (IRSTRT) NONISO, JHERM, ITTRIG, IDYNM, JREPOT, JTANGE,
WRITE (IRSTRT) JHERM, JFORCE, JUTEMP, JUCEED, JDISTS, JUHOC,
WRITE (IRSTRT) JDERIV, JUBOUN, JSTOP, JINTS, JPLAST, JBAND,
WRITE (IRSTRT) JFRONT, JDEFOR, JEMBED, JTEST, JDISP, JFBFGS,
WRITE (IRSTRT) IPSCTN, IFLINE, IPPRINT, ICOMPS, IPOCONU, JEIGEN,
WRITE (IRSTRT) IFBODY, IFGRAV, IFCENT, IJAMP, LDYN, ISTAT,
WRITE (IRSTRT) NSFXX, JISTIF, JCENTM, JFINIT, JLARGE, JFOLLOW,
WRITE (IRSTRT) JWKSPE, JPRES, JCDUM2, JCDUM3
WRITE (IRSTRT) NXSTAT, NXSOLV, NXINTG, NXMODL, NXBKCL, NXSUPR,
<table>
<thead>
<tr>
<th>WRITE (IRSTR)</th>
<th>LININC, LINTOT, NOECHO</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRITE (IRSTR)</td>
<td>DAMPF(3)</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IECQVC, IGNMS, IOMEG, IOMENO, IDYNMD, ISTRG1,</td>
</tr>
<tr>
<td></td>
<td>ISTRG2</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IMPO, IPTBR, IPTV, IMMD, IOMEG</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IC, IEL, IDP, IJAW, IPATH, IASSEM,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>JRULE, JCART, JEL009, JEL010, JEL011, JEL012</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>OMGEH, IHARM, KHARM, OMEGB, IBASE, KBASE,</td>
</tr>
<tr>
<td></td>
<td>IGNFOR, ICMFOR, ICMRES, ICHFHN, ICBFHN, ICBEXEC,</td>
</tr>
<tr>
<td></td>
<td>ICCMAT</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>FACTOR, INCFLG(20)</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>MAXCRD, MAXFR, MAXIOD, MAXSTR, MAXCHR, MAXPRS,</td>
</tr>
<tr>
<td></td>
<td>MAXLAY, MAXINT, MAXWR, MAXLV, NSUMAX, MAXCM,</td>
</tr>
<tr>
<td></td>
<td>MAXBS, MAXGMR, MAXTM, MAXEM, MAXLINK, MAXDM,</td>
</tr>
<tr>
<td></td>
<td>MAXPRN, MAXSET, MAXVAR, MAXKEN, MAXORD,</td>
</tr>
<tr>
<td></td>
<td>MAX025, MAX026, MAX027, MAX028, MAX029, MAX030</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>JLOUB, JINTER, JEXTRA, JWEEH, JSUBRE, JISTRN,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>JCITER, JHRLS, JGRAM, LOUB03, LOUB04, LOUB05</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>NTYPES, NELEM, NNODE, NBC, NTIE, NMAX,</td>
</tr>
<tr>
<td></td>
<td>NMRAN, NTRAC, NFD, NBAND, NEXT, NSUB,</td>
</tr>
<tr>
<td></td>
<td>NPRINT, NPPOST, NSEC, NDUP, NSIZE, NESB,</td>
</tr>
<tr>
<td></td>
<td>NSHIFT, NSBGS, NQAMS, NISPRI, NMASS, NDASH,</td>
</tr>
<tr>
<td></td>
<td>NDYNMD, NSBN, NSUPER, NHARM, NBASE, NINC,</td>
</tr>
<tr>
<td></td>
<td>NITER, NSPTS, NFPTS, NPULS, NPULS, NHARD,</td>
</tr>
<tr>
<td></td>
<td>NSUMCH, NPAR0, NPAR01, NPAR02, NPAR03, NPAR04, NPAR05</td>
</tr>
<tr>
<td></td>
<td>NPAR06, NPAR07, NPAR08, NPAR09, NPAR10, NPAR11, NPAR12</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IPTYPE(32), NTYPE, NPVA, NPSETS, JPART,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>NPP008, NPP009, NPP010, NPP011, NPP012</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IMEPS, ISDEV, IPDATA, IVTYPE, ISKIP, IREDX,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>INICO, IREACO, IRESCD, ISTRP, ISTR0, IASS0,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IPP013, IOMEG, IOMEG, IOMEG, IETAK, IZETAK,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>LXCOOR, IXCHAR, IXFOR, IXFORC, ISDIST, KDIST,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IXTEMP, JXTEMP, IXBEAM, IXVEC, IXSPRI, IXSPRI,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IXP015, IXP016, IXP017, IXP018,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>JPERIO(2), JPDIS, JPFOC, JPFOC, JPFOC, JPFOC,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IPOINT, JPOINTER, KPOINTER, NDATA, PRMTE(6)</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>SLPHI, SLPHI, SLPHI, SLPHI, SLPHI, SLPHI</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IFBP, ISPP, ISFP, ISQO, ICQO, ITIM,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IPSF, IPSD</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>NRP0TS, NRFS0S, NRFLN, NRPINT, NRFRA, NRPSEL,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>NRCMP, NRP20, NRSIZ, NRPEND, JRFTP, JOUNT,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>JCPAL, JRPSEL, JRF0D, NVKEEP, MORE</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IRFPTS, IRFIN, IRFMRN, IRFSR, IRFSE, IRFSET,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IRF0N, IRF0D, IRF0T, IRF0V, IRFSE, IRF0V, IRF0N,</td>
</tr>
<tr>
<td>WRITE (IRSTR)</td>
<td>IRF0C, IRF0D, IRF0R,</td>
</tr>
</tbody>
</table>

Systems' Manual

MHOST Version 4.2

Page: 140
WRITE (IRSTR)  IPULSE, KPULSE, IPOTIM, IPDFOR
WRITE (IRSTR)  MANVAR( 7 ), JPR, ICOM1, NOCMP
WRITE (IRSTR)  IELEM, ITP, INEL, ICHAR, IPRES, ISTRS,
  1  ISTRN, ICOM1, IPRINT, IPOST, IDIST, ILEAN,
  2  IELEM, IBNORM, IMONIT, IST116, IST117, IST118
WRITE (IRSTR)  INOD, ITMP, INLV, IPOSU, ITEMPF, IDUP
WRITE (IRSTR)  ISBC, ITI, ITAN, IDTAN, IEXT,
  1  ISBC, ISBCR
WRITE (IRSTR)  IDINC, IDROT, IFORCE, IPRESID, IWINOD, ISIGNO,
  1  IEPSNO, IPSTRN, ICSTRN, ITSTRN, ISTRS, ISTRN,
  2  IIIPSTR, IIICSTR, IIITSTR, IIPTSTR, IIPTSTR, IICSTRN,
  3  IIICSTRN, IIICSTRN, IICSTRN, IIICSTRN, IDMAT,
  4  IDMIN, IECST, IOMNO, IOMNO, IODSNO, IVSWTO,
  5  IDYNV, IDYNV, IDSV1, IDSV2, IDSTHR, IESWELL,
  6  IEVQSI, IPREF, IDSV3, IYIELD, IST441, IST442,
  7  IST443, IST444, IST445, IST446, IST447, IST448,
WRITE (IRSTR)  IRL, IRENC, IES, IAB, IBQM, ISRL,
  1  IRL, IRLC, ISRM, ILAST, ILRL, IDINCP, IFORIN,
  2  IOP, IDAN, IMASMT, IDIAG, IUPTR, ICOPT,
  3  IMASDI, IMASUP, IST521, IST522, IST523, IST524,
WRITE (IRSTR)  IELV, IORC, ISIG, IEPS, IWINOD, ISIGNO,
  1  INOD, ITII, ICH, IPP, IXRL, IXRIL,
  2  IXP, IXK, IKPSTR, ICSTRN, KISTRN, KISTR,
  3  KISTRN, KIPSTR, KICSTR, KITSTRN, KISTRN, KCSTRN,
  4  KISTRN, KISTRN, KICSTR, KITSTRN, KISTRN, KCSTRN,
  5  IMASNO, IMNO, IEQPST, IEQPSI, KEQPST, KEQPSI,
  6  KIDMAT, KIDMNO, KITDSNO, KITDST, IXC,
  7  IVELM, IAEML, IMASEL, KYIELD, IST647, IST648,
  8  IST649, IST650, IST651, IST652, IST653, IST654
WRITE (IRSTR)  ICON, IKBRCR, ITRACR, ITRASE, IBETA, IDECT
WRITE (IRSTR)  KGEPS, KIGEPS, KGSG, KGIG, KGIDST,
  1  IGEPSNO, IIGENNO, IGSGNO, IGSGNO, IGSGNNO,
  2  KGEPSNO, IIGENNO, KGISNO, KGIGSNO, KGIDST
WRITE (IRSTR)  KEQCSI, KEQMNO, KEQSN, KIPFNO, KDUMM,
  1  KEQCSI, KEQMNO, KSVWTO, KEQCN, KEQCS, KEQCSI
WRITE (IRSTR)  ISUBEL, ISUBPSTOP, ISUBPSTOP, NSDATA, ISUBET, IEMBED
WRITE (IRSTR)  NSUER, NSUNFR, NSUNOD, NSUSTR, NSUCHR, NSUPR,
  1  NJUIN, NSULV, NSUTEM, NSUEND, NSUSTR, NSUIDF
WRITE (IRSTR)  NLVSUB( 10 ), NFRSUB( 10 ), MHOST Version 4.2
WRITE (IRSTR)  NSHIFT, KSHIFT, IFREQ, LFR, NOFST, NOFOUND
WRITE (IRSTR)  TMINC, TOTINC, RUNTIM
WRITE (IRSTR)  DALPHA, DBETA, DGAMMA
WRITE (IRSTR)  RELERR, ABSERR, REACMX, RESIMX, DISERR, DISERR,
  1  ENGTOR, ENGRM
WRITE (IRSTR)  ISETUP, MAVAIL, LENREC, NUMREC, LEMN, NUMBLK,
WRITE (IRSTRT) MAXCOL, NCOL, LAYPR, JTENSR

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

NWORDS = 64
NRECDS = ILAST / NWORDS

DO 600 JREC = 1, NRECDS
  IBEG = 1+(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

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
C  ****************************************************************************
C
C IMPLICIT REAL*8 (A-H,O-Z)    REAL*4 RWORK
C
C DIMENSION IWORK(ISIZE), RWORK(ISIZE)
C
COMMON / ADDVAL / ISPRI , KSPI , IDASH , KDASH , IMASS , KMASS
COMMON / ALGEM / IREAD , ILPRINT , JLPRINT , ICONSL , IPOSTF , ISCRAF,
  1  IPL0T8 , ISTR8 , ICREAD , IPARS , IPSETS , IFILEX,
  2  PI , LINE , LINE2
COMMON / AUTOIN / CURPER , TOPER , ARCLEN , ATOLER , BTOLER , CTOLER ,
  1  JADAP , NCREEP , SCALE
COMMON / BODYFR / POINTS ( 3 , 2 )
COMMON / BSECT / ISECT , KSECT
COMMON / CONTRO / JEND , JITER , JTEMP , JPRINT , JP , JSUB ,
  1  JINC , JREST , JSAVE , JREDIM , JAVIO , JPOST ,
  2  JBACK , JOPTIM , JCREEP , JDIST , JCONST , JDYN ,
  3  NONISO , ITERM , ITRIG , IDYNM , JREPOT , JTAU ,
  4  JHERM , JFORCE , JUTEMP , JUOEF , JDISTS , JUOCR ,
  5  JDEVI , JUBON , IDSTOP , INSTR , JLAST , JBAND ,
  6  JFROJ , JDEFOF , JEMBED , ITEST , JDISP , IFBFGS ,
  7  IPSCTNT , IPLINE , IPFRNT , ICINPS , IPCONJ , IEIGEN ,
  8  IBODY , IPGRAV , IPCENT , JDAMP , LDYN , ISPAT ,
  9  JFDSXX , JISTIF , JCENM , JPRINT , JLARGE , JFOLLOW ,
    JNKSLP , JPRES , JCDUM2 , JCDUM3
COMMON / COMPND / NXSTAT , NXSOLV , NXINTG , NXMDSL , NXBCKL , NXSUPR ,
  1  NXREQN , NXDUM1 , NXDUM2 , NXDUM3 , NXDUM4
COMMON / COUNT / LININC , LINTOT , NOECHO
COMMON / DAMP / DAMPF ( 3 )
COMMON / EIGEN / IECNV , IQS , IOMEG , IOMENO , IDYNMD , ISTR12 ,
  1  IPTAR , IPTBR , IPTVED , IMDAM , IOMEG
COMMON / MODSUP / IMPORO , IMDIS0 , IMVELO , IMPOR1 , IMDIS1 , IMVE11
COMMON / ELEMEM / IC , IEL , IDF , JLAW , IPATH , IASSEM ,
  1  JRULE , JCAIT , JREL09 , JREL010 , JREL011 , JREL012
COMMON / HARMON / OMEGL , IHARM , KHARM , OMEGB , IBASE , KBASE ,
  1  ICNFOR , ICMPOR , ICMPRS , ICHFH , ICBBF , ICBBCC ,
  2  ICI2MAT
COMMON / INCCON / FACTOR , INCFILG ( 21 )
COMMON / MAXIMA / MAXCRO , MAXNFR , MAXNOD , MAXSTR , MAXCHR , MAXPRS ,
  1  MAXLAY , MAXINT , MAXWKR , MAXNLV , NSUMAX , MAXMCP ,
  2  MAXSRP , MAXGMR , MAXTEM , MAXLM , MAXLW , MAXINT ,
  3  MAXFRI , MAXBET , MAXVAR , MAXSET , MAXEAN , MAXORD ,
  4  MAXO25 , MAXO26 , MAXO27 , MAXO28 , MAXO29 , MAXO30
COMMON / LOUBIN / JLOUB , JITNTER , JEXTRA , JWEIGH , JSUBRE , JISTRN,
COMMON / PARAM / NTYPE, NELEM, NNODE, NSEC, NIE, NMAX,
1 NTR, NTRAC, NF, NBAND, NEXT, NSUB,
2 NPRINT, NPOST, NSEC, NUP, NSIZE, NSCET,
3 NSHIFT, NSECFS, NGMFS, NSPI, NMASS, NDASH,
4 NDYNMD, NSEC, NSUPER, NHARM, NBACE, NINC,
5 NTER, NPDPTS, NPDPTS, NPULSE, NPDPTS, NHARD,
6 NSUMCH, NDIMEN, NMONIT, NPAR40, NPAR41, NPAR42,
7 NPAR43, NPAR44, NPAR45, NPAR46, NPAR47, NPAR48,

COMMON / PERPAR / IPTYPE(32), NPTYPE, NPVARS, NPSETS, JPERT,
1 NOSCON, NNP008, NNP009, NNP010, NNP011, NNP012,
2 NDIC0, INREAC0, IRESD0, IDGRP, IISTIF0, IMASS0,
3 NP013, IOMEG0, IOMEGP, IOMEGK, IETAK, IETAK,
4 NEXP, NEXP, NEXP, NEXP, NEXP, NEXP,
5 JPERIOD, JPERIOD, JPERIOD, JPERIOD, JPERIOD,
6 COMMON / POSTN / IOPOINT, IOPOINT, KPOINT, NDATA, PRNBF(6),
7 COMMON / POWER / IEPL, IEPL, IEPL, IEPL, IEPL,
8 COMMON / PREPAR / NRFPTS, NRFDP, NRFDT, NRFSDS, NRFSEL,
9 JCFIP, JCFIP, JCFIP, JCFIP, JCFIP,
10 COMMON / PREPTR / IRFPTS, IRFINT, IRFPLUS, IRFSEL, IRFSET,
11 IRFMIN, IRFSDV, IRFPIT, IRFVEC, IRFPIT, IRFSET,
12 IRFCOR, IRFPL, IRFPL, IRFPL, IRFPL,
13 COMMON / PULSES / IPULSE, KPULSE, IDPF, IPFDT,
14 COMMON / RESULT / MVAR(7), JPR, JCOM, JCOMP,
15 COMMON / START1 / IEPL, IYPE, IENL, IECHAR, IPRES, ISTRS,
16 ISTRN, ITOP, IPND, IDIST, IEAN,
17 IEPPS, IEINM, IMONIT, IST116, IST117, IST118,
18 COMMON / START2 / IDOD, ITIM, INLV, IPOSU, ISTRM, IDUP,
19 COMMON / START3 / IEBS, IIT, ITRAN, ITRAC, IEXT,
20 COMMON / START4 / IDIB, IITOT, IIFOR, IDRES, IWMON, IDIGNO,
21 IESPNO, IESPST, ICTRN, ICSTR, ICTRN, ICTRN,
22 IIPSTR, IIISTR, IIISTR, IIISTR, IIISTR,
23 IIISTR, IIISTR, IIISTR, IIISTR, IIISTR,
24 IIISTRN, IISNO, IIISPNO, IIICSN0, IITSSD, IIDMAT,
25 IDMNNO, IIEQCS, IIOMENO, IIOMNO, ITDSNO, IIWSW0,
26 IDYNV, IDTNA, IDSSX1, IDSSX2, IDSTR, ISWELL,
27 IEQCSI, IIPREF, ISDX3, IYFIELD, IDFNC, IDFTOT,
28 COMMON / START5 / IRL, IREAC, IES, IAB, IBQM, ISRL,
** READ BACK THE CONTENTS OF THE COMMON BLOCKS **

```plaintext
READ (IRSTRT) ISPRI, KSPRI, IDASH, KDASH, IMASS, KMASS
READ (IRSTRT) ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF, 1
  IPLOTB, IRSTRT, JCREAD, IPVARS, IPSETS, IPFILEX, 2
  PI, LINE, LINE2
READ (IRSTRT) CURPER, TOTPER, ARCLEN, ATOLER, BIOLER, CTOLER, 1
  JADAP, NCREEP, SCALE
READ (IRSTRT) POINTS(3, 2)
READ (IRSTRT) IBSECT, KBSECT
```

** READ BACK THE CONTENTS OF THE COMMON BLOCKS **

```plaintext
READ (IRSTRT) ISPRI, KSPRI, IDASH, KDASH, IMASS, KMASS
READ (IRSTRT) ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF, 1
  IPLOTB, IRSTRT, JCREAD, IPVARS, IPSETS, IPFILEX, 2
  PI, LINE, LINE2
READ (IRSTRT) CURPER, TOTPER, ARCLEN, ATOLER, BIOLER, CTOLER, 1
  JADAP, NCREEP, SCALE
READ (IRSTRT) POINTS(3, 2)
READ (IRSTRT) IBSECT, KBSECT
```

** READ BACK THE CONTENTS OF THE COMMON BLOCKS **

```plaintext
READ (IRSTRT) ISPRI, KSPRI, IDASH, KDASH, IMASS, KMASS
READ (IRSTRT) ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF, 1
  IPLOTB, IRSTRT, JCREAD, IPVARS, IPSETS, IPFILEX, 2
  PI, LINE, LINE2
READ (IRSTRT) CURPER, TOTPER, ARCLEN, ATOLER, BIOLER, CTOLER, 1
  JADAP, NCREEP, SCALE
READ (IRSTRT) POINTS(3, 2)
READ (IRSTRT) IBSECT, KBSECT
```

** READ BACK THE CONTENTS OF THE COMMON BLOCKS **

```plaintext
READ (IRSTRT) ISPRI, KSPRI, IDASH, KDASH, IMASS, KMASS
READ (IRSTRT) ICREAD, ILPRNT, JLPRNT, ICONSL, IPOSTF, ISCRAF, 1
  IPLOTB, IRSTRT, JCREAD, IPVARS, IPSETS, IPFILEX, 2
  PI, LINE, LINE2
READ (IRSTRT) CURPER, TOTPER, ARCLEN, ATOLER, BIOLER, CTOLER, 1
  JADAP, NCREEP, SCALE
READ (IRSTRT) POINTS(3, 2)
READ (IRSTRT) IBSECT, KBSECT
```
READ (IRSTRT)  JEND, JITER, JTEMP, JPRINT, JP, JSUB, 
1 JINC, JREST, JSAVE, JREDIM, JAUTO, JPOST, 
2 JBACK, JUPTIM, JCREEP, JDIST, JCONST, JDYN, 
3 NONISO, ITERM, ITRIG, IDYNM, JREPO, JFACE, 
4 JITERM, JFORCE, JUTEMP, JUOE, JDISTS, JHooke, 
5 JDERIV, JBOUND, IDSTOR, INISTR, JPLAST, JFBAND, 
6 JFACE, JDEFOR, JEMBED, JTEST, JDISP, JPBEG, 
7 IFSCMT, IFLINE, IFPRNT, IMCON, JDIAG, JEIGEN, 
8 IPBODY, IPGRAD, IPCENT, JDAEM, IDYN, ISTAT, 
9 JFDGXX, JFSTRF, JCENT, JINIT, JLARGE, JFAROW, 
1 JWSLIP, JPRES, JCDUM2, JCDUM3, 
+ READ (IRSTRT)  NXSTAT, NXSOLV, NXINTG, NXMODL, NXXSL, NXSUPR, 
1 NREGQ, NDXDUM1, NDXDUM2, NDXDUM3, NDXDUM4, 
READ (IRSTRT)  LINC, LININT, NOECO, 
READ (IRSTRT)  DMAPF(3), 
READ (IRSTRT)  IEGQVC, IGNMS, IOMEG, IOMEO, IDYNM, ISTRU2, 
1 IPTC, IPTBR, IPTVE, IDAM, IOMEG, 
READ (IRSTRT)  IMPOR, IMDIS0, IMVE0, IMPOR1, IMDIS1, IMVE1, 
READ (IRSTRT)  IC, IEL, IDF, JLAH, IPATH, IASSEM, 
1 JRULE, JCAR, JEL009, JEL010, JEL011, JEL012, 
READ (IRSTRT)  OMEGH, JHARM, JHARM, JOMEG, JBASE, JBASE, 
1 INCFOR, JCMFOR, JCMRES, ICHHFN, ICEHFN, ICSEX, 
2 ICCMAT, 
READ (IRSTRT)  FACTOR, INCFLG(20), 
READ (IRSTRT)  MAXCRD, MAXNFR, MAXMOD, MAXSTR, MAXCHR, MAXPS, 
1 MAXL, MAXINT, MAXWK, MAXLW, NSUMP, MAXCP, 
2 MAXBSP, MAXGR, MAXTEM, MAXLM, MAXLW, MAXDT, 
3 MAXFRN, MAXET, MAXVAR, MAXSET, MAXEAN, MAXORD, 
4 MAX025, MAX026, MAX027, MAX028, MAX029, MAX030, 
READ (IRSTRT)  JLOUB, JINTER, JEXTRA, JWEIGH, JSBRE, JISTRN, 
1 JCTTER, JHICLS, JGRAM, JLOUB03, JLOUB04, JLOUB05, 
READ (IRSTRT)  NTYPE, NELEM, NNODE, NBC, NTIE, NNMA, 
1 NTRAN, NTAC, NFD, NBAND, NEXT, NSUB, 
2 NPRINT, NPST, NSBC, NDUP, NSIZE, NSECT, 
3 NSHIFT, NSBFGS, NSGR, NSPR, NMASS, NDASH, 
4 NDYNM, NSNC, NSUPER, NHARM, NBASE, NPC, 
5 NITER, NPSPTS, NFPTS, NPULSE, NDPTS, NHARD, 
6 NSUCH, NPRAR38, NMOIT, NPRAR40, NPRAR41, NPRAR42, 
7 NPRAR43, NPRAR44, NPRAR45, NPRAR46, NPRAR47, NPRAR48, 
READ (IRSTRT)  ITYPE(32), NTYPE, NPVAR, NPSET, JPER, 
1 NPWIN, NPPO08, NPPO09, NPPO10, NPPO11, NPPO12, 
READ (IRSTRT)  IMEANS, ISDEV, IPDATA, ITYPE, ISKIP, IREDEF, 
1 IDINC0, IREAC0, IRESD0, IDGRP, ISTITF0, IMASS0, 
2 IPPO13, IOMEG0, IOMEG1, IOMEG2, IETAK, IZETAK, 
READ (IRSTRT)  IXCOOR, IXCHAR, IXFORC, KXFORC, KXDIST, KXDIST, 

SYSTEMS' MANUAL

Page : 146
DO 600 JREC = 1, NRECDS
    IBEG = 1+(JREC-1)*NWORDS
    IEND = JREC*NWORDS
    READ (IRSTR) (IWORK(JJ), JJ=IBEG, IEND )

CONTINUE

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

*----------------------------------------------------------------------*
* ** PRINT MESSAGE AND PROCEED WITH THE ANALYSIS AS USUAL            **
*----------------------------------------------------------------------*
C+    CALL QUIT ("REST", "ART", "FILE", "REA", "DER", ",1)
    CALL TIMOUT ("REST", "ART", "FILE", "WAS", ", REA", "D ")
C
*----------------------------------------------------------------------*
* ** STEP UP THE INCREMENT COUNTER                                    **
*----------------------------------------------------------------------*

SYSTEMS' MANUAL  MHOST Version 4.2
Page : 148
```
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>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 (1- given)</td>
</tr>
</tbody>
</table>
```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8.</td>
<td>IPSTYP</td>
<td>Type of tape (1= Formatted)</td>
</tr>
<tr>
<td>9.</td>
<td>NCRD</td>
<td>Number of coordinates per node</td>
</tr>
<tr>
<td>10.</td>
<td>NNODMX</td>
<td>Maximum number of nodes per element</td>
</tr>
<tr>
<td>11.</td>
<td>IANTYP</td>
<td>Analysis Type Flag:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = Displacement - With Reaction Forces</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = Dynamics - With Reaction Forces</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15 = Eigenvector (Modal)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16 = Eigenvector (Buckle)</td>
</tr>
<tr>
<td>12.</td>
<td>ICOMPL</td>
<td>Set to 0 if real analysis</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set to 1 if complex analysis</td>
</tr>
<tr>
<td>13.</td>
<td>NBCTRA</td>
<td>Number of nodes with transformations</td>
</tr>
<tr>
<td>14.</td>
<td>POSTRV</td>
<td>Post tape revision number, 1 for this release (Version 4.2)</td>
</tr>
<tr>
<td></td>
<td>IDM4</td>
<td>Not used; reserved for future expansion</td>
</tr>
<tr>
<td></td>
<td>IDM5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IDM6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IDM7</td>
<td></td>
</tr>
</tbody>
</table>

**Note:**

\[ \text{INOD} = \{\text{NDEG}\} \times \text{JNODE} \]

If \( \text{IANTYP} = 2 \) on 2

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

\( \text{IANTYP} = 15 \) only appears during subincrements

\( \text{IANTYP} = 16 \) only appears during subincrements

If \( \text{IANTYP} = 8 \) and \( \text{ICOMPL} = 1 \) \( \text{JNODE} = 2 \)

If \( \text{IANTYP} = 9 \) and \( \text{ICOMPL} = 1 \) \( \text{JNODE} = 4 \)

Block II

**CODE NUMBER ASSOCIATED WITH ELEMENT VARIABLES**

\( \text{INUM} \) Records; \( J=1, \text{INUM} \)

**SYSTEMS' MANUAL**

MHOST Version 4.2

Page : 150
**Element Variable Code**

<table>
<thead>
<tr>
<th>Block IV</th>
<th>CONNECTIVITY LIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>NR;6I3;JPLOT(J)</td>
<td>If IPSTCC is zero this block will be omitted.</td>
</tr>
<tr>
<td>Block V</td>
<td>COORDINATE LIST</td>
</tr>
<tr>
<td>NR;6I3;LM(1)</td>
<td>MNUM Records; J=1,MNUM</td>
</tr>
<tr>
<td>LM(2)</td>
<td>Element Type</td>
</tr>
<tr>
<td>LM(3)</td>
<td>Number of Nodes in this Element</td>
</tr>
<tr>
<td>LM(NNODMX+2)</td>
<td>1st Node of Element J</td>
</tr>
<tr>
<td>Block VI</td>
<td>TRANSFORMATION LIST</td>
</tr>
<tr>
<td>NR;6E13.6; SUM(1)</td>
<td>If IPSTCC is zero this block will be omitted.</td>
</tr>
<tr>
<td>SUM(2)</td>
<td>LNUM Records; J=1, LNUM</td>
</tr>
<tr>
<td>SUM(NCRD)</td>
<td>1st Coordinate of Jth Node</td>
</tr>
<tr>
<td>Block VII</td>
<td>TRANSFORMATION - DIRECTION COSINES</td>
</tr>
<tr>
<td>NR;6I13; LM(I)</td>
<td>If NBCTRA is zero this block will be omitted.</td>
</tr>
<tr>
<td></td>
<td>List of nodes which have transformations applied</td>
</tr>
</tbody>
</table>

---

SYSTEMS' MANUAL

Page : 151

MHOST Version 4.2
1 record per each node listed in Block VI if binary tape

\[
\begin{align*}
NR; 6E13.6; & \quad D(1,1) \\
D(2,1) & \\
D(3,1) & \\
D(1,2) & \\
D(2,2) & \\
D(3,2) & \\
D(1,1) & \\
D(2,3) & \\
D(3,3) & \\
\end{align*}
\]

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)**

\[
\begin{align*}
X1(2) & \\
X1(3) & \\
X1(4) & \\
X1(5) & \\
X1(6) & \\
\end{align*}
\]

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.

**MNUNM*NSTRES Records**

**NR; 6E13.6;**

\[
\begin{align*}
VALUE((CI,J), \; I=1,\; INUM), \; J=1,\; INODE) & \\
\end{align*}
\]

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>
18 Mean normal stress (tensile positive) (for Mohr-Coulomb).
19 User definable quantity to write on post tape.
20 User definable quantity to write on post tape.
21-26 Physical components of the total plastic strain.
27 Total equivalent plastic strain.
29 Second state variable.
31-36 Physical components of total creep strain.
37 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  2  4  2  0
  0  0  0  0  0

17
11
12
13
39
1
2
3
7
21
22
23
8
31
32
33
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>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>4</td>
<td>14</td>
<td>15</td>
<td>18</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>16</td>
<td>17</td>
<td>20</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>17</td>
<td>18</td>
<td>21</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>19</td>
<td>20</td>
<td>23</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>20</td>
<td>21</td>
<td>24</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>22</td>
<td>23</td>
<td>26</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>23</td>
<td>24</td>
<td>27</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>25</td>
<td>26</td>
<td>29</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>26</td>
<td>27</td>
<td>30</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>28</td>
<td>29</td>
<td>32</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>29</td>
<td>30</td>
<td>33</td>
<td>32</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<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>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>2.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>4.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.0000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>6.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.0000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>8.0000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.0000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>10.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>12.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>14.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14.000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>16.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>18.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18.000</td>
<td>0.00000</td>
<td></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>
<td></td>
</tr>
<tr>
<td>20.000</td>
<td>0.50000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20.000</td>
<td>0.00000</td>
<td></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>
<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>
<td></td>
</tr>
</tbody>
</table>
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 / ICREAD, JLPRNT, JLPRNT, ICONSL, IPPOST, ISCRAF,
1 IPLOTB, IP3TRT, JCREAD, IPVARS, IPSETS, IFILEX,
2 IP , LINE , LINE2
COMMON / PARAM / NTYPE , NELEM , NNODE , NBC , NTIE , NMAX ,
1 NIRAN , NIRAC , NFD , NBAND , NEXT , NSUB ,
2 NPRINT , NSUP , NBAND , NSUB , NSECT ,
3 NSHIFT , NSBFGS , NGRFS , NSPRL , NMAX , NDASH ,
4 NDYNMD , NSCNC , NSUPR , NBASE , NTNC ,
5 NTMPTR , NSPT , NFPTS , NFPRPT , NFPRPT , NFPRPT , NFPRPT , NFPRPT ,
6 NSUMCH , NDIMEN , NMONIT , NPAR40 , NPAR41 , NPAR42 ,
7 NPAR43 , NPAR44 , NPAR45 , NPAR46 , NPAR47 , NPAR48
COMMON / CTTITLE / TITLE ( 20 ) , IDAT ( 5 ) , IDATE2 , ICLK ,
1 IPCRAY
COMMON / EIGEN / IEQVNC , IEQMS , IOMSQ , IOENO , IDYNMD , ISTAT2 ,
1 IPTAR , IPTBR , IPTRED , IMDAM , IOMEGD
COMMON / ELEMEN / ICP , IEL , IDP , JELAW , IPAT , ISASSEM ,
1 JRULE , JCART , JEL009 , JEL010 , JEL011 , JEL012
COMMON / ELTYP / NELCRD , NELNFR , NELNOD , NELSTR , NELCHR , NELPR ,
1 NELINC , NELLV , NELLAY , NDI , NSHEAR , NELCMP
COMMON / ERRORS / IERR
COMMON / MACHIN / IDP
COMMON / MAXIMA / MAXCRD , MAXNFR , MAXNOD , MAXSTR , MAXCHR , MAXPRS ,
1 MAXLAD , MAXINT , MAXWRT , MAXNLV , NSUMAX , MAXCEM ,
2 MAXESP , MAXCHR , MAXTEM , MAXELM , MAXLW , MAXMT ,
3 MAXFRN , MAXBTC , MAXVAR , MAXSET , MAXEAN , MAXORD ,
4 MAX025 , MAX026 , MAX027 , MAX028 , MAX029 , MAX030
COMMON / CONTRO / JEND , JITER , JTEMP , JPRINT , JP , JSUB ,
1 JINC , JREST , JSAVE , JREDIM , JAUTO , JPOST ,
2 JBACK , JOPTIM , JRCEPL , JDIST , JCONST , JDYN ,
3 NONISO , ITHRM , ITRIG , IDYN , JJREVOT , JTHANG ,
4 JTHRM , JFORCE , JUFTMP , JUCQEF , JDISTS , JUHOOK ,
5 JDERIV , JUBLASS , JSTP , JINTSTR , JPLAST , JNAND ,
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 IPTOF FILE AND THEN IPTOF 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
NNODMX  =  NEINOD
IANTYP  =  2
IF( JEIGEN .EQ. 1 .AND. IDYNM .EQ. 1 ) IANTYP  =  15
IF( JEIGEN .EQ. 1 .AND. IDYNM .EQ. 0 ) IANTYP  =  16
ICOMPL  =  0
NBCTRA  =  0
IPOSTR  =  1
IDM4  =  0
IDM5  =  0
IDM6  =  0
IDM7  =  0

WRITE(IPOSTF, 1010) INUM ,I2/t_ ,MNUM ,NDEG ,NSTRES,JNOD ,
1 IPSTCC,IPSTYP,NCRD ,NNODMX,IANTYP,ICOMPL,
2 NBCTRA,IPSTOR,IDM4 ,IDM5 ,IDM6 ,IDM7
1010 FORMAT(6I13)

C --- CODE NUMBER ASSOCIATED WITH ELEMENT VARIABLES -------------------
C
IF( JEIGEN .EQ. 1 ) GO TO 3000
C
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  =  MINCMP + 1
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

SYSTEMS' MANUAL

Page : 162
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

WRITE(IPOSTF,1030) (JPLOT(K), K=1,INDX)
1030 FORMAT(113)

300 CONTINUE

--- DATA BLOCK IV : ELEMENT CONNECTIVITY DATA

DO 6000 IEL = 1, NELEM
    IC = IWORK(ITYP + IEL - 1)
    NELNOD = 0
    IF( IC .EQ. 0) GO TO 6007

CALL ELVULV1 (IWORK, IC, IERR)
    IADRES = INEL + MAXNOD * (IEL - 1)
    JADRES = IADRES + NELNOD - 1

... SPECIAL TRICK ASSOCIATED WITH THE FRONTAL SOLUTION

DO 6005 INTNPNT = 1, NELNOD
    JNPNT = INTNPNT + IADRES - 1
    6005 KPLOT(INTNPNT) = IABS(IWORK(JNPNT))

CONTINUE

... REPLACE THE ASSUMED STRAIN ELEMENTS WITH THE MATCHING MARC/MENTAT ELEMENT NUMBERS (A LITTLE INNOCENT LIE THAT WON'T HURT ANYBODY)

IC1 = IC
    IF ( IC .EQ. 151) IC1 = 3
    IF ( IC .EQ. 152) IC1 = 11
IF ( IC .EQ. 153 ) ICl = 10
IF ( IC .EQ. 154 ) ICl = 7

WRITE(IPOSTF,1010) ICI ,NELNOD,(KFLT(K),K=1,NELNOD)

CONTINUE

--- DATA BLOCK V : NODAL COORDINATE DATA -----------------------

DO 6010 INODE = 1 , NNODE

IADRES = INOD + MAXCRD * ( INODE - 1 ) * IDP

CALL POSTPR

1 (RWORK(IADRES),NCRD,IPOSTF)

CONTINUE

CONTINUE

--- INCREMENTAL HEADER RECORD --------------------------

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 = IOMEG + ( ISUBIN - 1 ) * IDP
CALL COPY( RWORK(INDX), FREQUN, 1 )
IF( IDYNM .EQ. 1 ) RANTYP = 15
IF( IDYNM .EQ. 0 ) RANTYP = 16

CONTINUE

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

1020 FORMAT(6G13.6)

IF( JEIGEN .EQ. 1 ) GO TO 7500
C
INUM = (MINCMP + 1) * 4
INITAL = 0
DO 7000 INODE = 1, NNODE
C
IREC = 0
C
CALL NUL( VALUE, MAXVAL )
C
*** VALUES CALCULATED AT THE LAYER LEVEL ******************************
C
INILAY = 1
IF( NELLAY .NE. 1 ) INILAY = 3
C
*** FIRST DATA OF THE SERIES IS TEMPERATURE ***************************
C
LOCTEM = 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
C
DO 5200 INTQNT = 1, MAXQNT
C
INTLOC = LOCATE( INTQNT ) + (( INODE - 1 ) * MAXLAY + INILAY - 1 )
1 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
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
1
    CALL TRANS2(NTRAN,RWORK(ITRANR),IWORK(ITRAN),RWORK(IDTOT),
    NFD,-1.DO,NELNFR,JIWAR)
    CALL TRANS2(NTRAN,RORK(ITRANR),IWORK(ITRAN),RWORK(IDINC),
    NFD,-1.DO,NELNFR,JIWAR)
9550
C
    IDVEC = IDTOT
    IF( JEIGEN .EQ. 1 ) IDVEC = IEIGMVC + ( ISUBIN - 1 ) * NFD * IDP
C
DO 5000 INODE = 1 , NNODE
C    ... TOTAL DISPLACEMENTS OR EIGENVECTOR
C
    JNTLOC = INLV + ( INODE - 1 ) * MAXNFR
    KNITLOC = IDVEC + ( IWORK(JNTLOC) - 1 )*IDP
    CALL COPY( RWORK(KNITLOC) , VALUE , MAXNFR )
    NENTRY = MAXNFR
C
C    ... NODAL REACTION VECTOR
C
    INDX = NELNFR + 1
    LNTLOC = IRESID + ( IWORK(LNTLOC) - 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
C    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)

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

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

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

DIMENSION ARRAY (NENTRY)

COMMON /POSTPN/ IPOINT,JPOINT,KPOINT,NDATA,PRNTBF(6)

IF (INITAL.NE.0) GO TO 4950
  IPOINT = I
  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

JPOINT = 1

5000 CONTINUE

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
APPENDIX SUBROUTINES

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, SUBEIM, 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,
              SUBEIM, 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,
              SUBEIM, 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,
              SUBEIM, 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,
MACHIN, MAXIMA, SUBELM, SUBTYP, LOUBIN, PARAM, START1, START2, START3, START4, START5, START6, START7, START8, TRANSF, TIME

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.

BNDTRM
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.

BPSTRES
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, INCCON, 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: CONTRO PARAM, SUBELM

External reference: INITII, 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, VSHO4N, TBM02N, S1DO2N,
S3DO8N, D2DO4N, D3DO8N, DAXO4N, GSHO4N, DMSO2N, 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.

CENTBM

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, ELYTP

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, ROTDMT, TSH04N, SHTRAN, TRANSF, MATINV, TBM02N, ADD, ADDSMU, NUL

CNMAS
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.
CPXBK1, 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

SUBROUTINES
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

DATEMS
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, INCON, 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, INCON

DATOU1
Prints out model data as interpreted on the line printer.

Common block: ALGEM, CONTRO, PARAM, INCON, 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, INCON, TOLER, MACHIN

DATOU4
Prints incremental data as interpreted on the line printer.

Common block: ALGEM, CONTRO, PARAM, INCON, TOLER

SUBROUTINES
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, CONTRO, 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, INCNON, ELEMEN, ELTYP, 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, ELTYP, PARAM, MACHIN, START3, START4, START5, START7

ELVULV
Pulls out parameters defining element characteristics from the workspace.

Common block: ALGEM, CONTRO, ELTYP, 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

FOLOIN  
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, ELEMEN, MAXIMA, ELTYP, CONTRO, SUBSTR, ERRORS, PARAM, SHIFT

External reference: STRUCT, TIMOUT, ASSEM1, SOLUT1, ASSEM4, RESCON, ADDINC, PRINOU, POSTOU, COPY, NUL, INITST, SUBT, MASMAT, MADD, EIGENV, CNSTNM, PNTINM, INISQQ, SETRMS, QUIT

FREFOR  
Reads free format numeric data line from the card reader.

Common block: ALGEM, FREE
FRINTBL
Sets up the elimination table for the frontal solution and calculates maximum front matrix size.

Common block: ALGEM, PARAM, MAXIMA, MACHIN

FRONTOP

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, INCON, MACHIN, ELEMEN, MAXIMA, ELTYP, CONTO, 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.

GEM02N
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, HOLELM

**HOLEIN**

Reads parameters to define a embedded hole in a global element.

**Common block:** ELTYP, SUBELM, MACHIN

**HOLELM**

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.
HOOKLW

Calculates the linear elastic stress-strain matrix for all element types other than beams.

External reference: NUL, UTTEMP, UHOOK

HOST

Supervises execution of the MHOST code. See Section 1.2 for detail.

Common block: CONTRO, ALGEM, ADDVAL, TPMARCH, AUTOIN, PERIOD, POWER, PULSES, EIGEN, ERRORS, MODSUP, HARMON, LOUBIN, START5, START4, PARAM, BSECT, INCON, 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: ADUVAL, ALGEM, CONTRO, DAMP, EIGEN, MODSUP, HARMON, POWER, PULSES, MAXIMA, PARAM, TMARCH, PERIOD, SUBELM, SHIFT, SUBTYP, BSEC, START1, START2, START3, START4, START5, START6, START7, START8, START9, SUBSTR, MACHIN, ERRORS

INITI2
Allocations 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, START2, START4

INITIN
Reads parameters defining initial conditions for transient analysis.

Common block: ALGEM, ERRORS, CONTRO, MAXIMA, ELEMEN, ETYPE, 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, ELTYP, 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

MHOST Version 4.2
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.

INTIDYN

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

INTISQ

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 (ILPRINT).

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, CONTRO, 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, PAGE3S

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

SUBROUTINES

MHOST Version 4.2
Page : A - 25
PAGE25
Same as PAGE2. Used for subelement solution.

PAGE35
Same as PAGE3. Used for subelement solution.

NUL
Zero - clears a double precision real array.

NULINT
Zero - clears an integer array.

NULNRM
Zero - clears space allocated for nodal normals of shell coordinate definition.

OPTIM
Controls execution of the bandwidth optimization option.

External reference: LINES, MKFAKE, MAXCON, CURNIL, QUIT

ORIENT
Generates the coordinate transformation matrix to the preferred orientation of an anisotropic material.

OUTPRO
Performs the outer product of two three-dimensional vectors.

Common block: START2, START3, START4, START5, START6, START7, START8, SUBSTR, TIME, TRANSF, MACHIN

NEWVEL
Updates the velocity vector at the beginning of a time integration step.

NODPRE
Computes the nodal pressure and directly adds it into the global load vector.

NODSTR
Computes stress at a given point and generates material tangent.

External reference: NUL, COPY, ROTPREF, THRSTN, CRPSTN, SUBT, SIMPLE, MULT, LElast, PLASTS, PLASTD, WALKEQ

SUBROUTINES
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, CONTR, SUBSTR, ERRORS, PARAM, SHIFT, START8

External reference: STRUCT, TIMDUT, ASSEM1, MASMAT, MADD, EIGENV, POSTOU

PNTINM
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, START5, 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.

PRNTEL      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:  VRIDSK, 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.

READX

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.
matrix.

RELDIFG

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, ELTYP, 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

RESCHK

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

RESDYN

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

Renumerates nodes for the reduction of front width. Not functional
in Version 4.2.

RESEQ2

Renumerates elements for the reduction of front width. Not
functional in Version 4.2.

RESID

MHOST Version 4.2
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: ADDVAL, ALGEM, AUTOIN, BODYPF, BSECTION, CONTROL, COMPOUND, 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,
VRITDSK, ZPRINT

ROTBAK

Rotates the stress state back to the global coordinate system in finite deformation analysis.

ROTDIMT

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.

ROTFRF

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.

RTFOLF  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, INCCON, MAXIMA, LOUBIN, PARAM, PERIOD, POSTPN, POWER, PULSES, RESULT, START1, START2, START3, START4, START5, START6, START7, START8, START9, SUBELM, SUBTYP, SUBSTR, SHIFT, TIME, TMARCH, TOLER, VRIDSK, 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, RTOFLF, ADDPUL, SUBT, TYING1, BOUND1, DECOMP, BFSGW, BFSGRH, COPY, SOLVER, MADD, MATSUM, BFGLH, 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.

SPRIN
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

STRIPB  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, CONTR0, 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, CONTR0, SUBTYP, ELEMEN, ERRORS

SUBDIV  Controls the subelement mesh generation.

Common blocks: ALGEM, CONTR0, 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

SUBPfEM  Supervises the mixed iterative solution in the subelement mesh. Called by ASSEM4.

Common block: ALGEM, CONTR0, ELTYP, ELEMEN, SUBTYP, TRANSF, LOUBIN,

SUBROUTINES
MAXIMA, PARAM, ERRORS, MACHIN, TIME, START1, START2, START3,
START4, START5, START6, START7, START8, START9, TOLER

External reference: TIMOUT, SUBELV, SUBALC, SMASTR, NUL, SNODEL, LMPMAS,
SUBINT, SUBT, STRESS, ADD, SUBDER, INTERP, STIFF,
SYSEQN, SUBSOL, STRAIN, SUBRES, SUBCHK, SUBGLB, SUBGLD

SUBGLB, SUBGLD
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

SUBS01, 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, QUITT, SMULT,
LINES

SUBROUTINES
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.
TBM02N
    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

TMLT
Multiplies the transpose of an array by a matrix.

TMLTV
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, UTEMD, 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, VRIDSK

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