MSC/NASTRAN Based Component Mode Synthesis Analysis Without The Use of DMAPs

by

Tarun Ghosh
Rockwell International Corporation
Canoga Park, California

Abstract

Component mode synthesis method of analyzing large structures is a very powerful and efficient tool available in MSC/NASTRAN. For many years the method based on MSC/NASTRAN has been widely used in the aerospace and automotive industries where different physical components are often designed and modelled by different organizations, departments or groups. For forward and backward flow of data it is important to have a standard, yet flexible, method. Unfortunately, different organizations have their own Direct Matrix Abstraction Program (DMAP) based method of component mode synthesis. This becomes an obstacle to the free flow of data, increases the chance of errors, imposes restrictions and creates a major task of updating and verifying the DMAPs with MSC/NASTRAN revisions.

The purpose of this paper is to show through an actual example how component mode synthesis can be performed in MSC/NASTRAN without the use of elaborate DMAPs. The analyst can do such tasks as plotting, data recovery, apply loads, transfer data forward and backward, without the use of a single structured solution based DMAP. The net result is an efficient process that reduces the time and cost of the analysis.

INTRODUCTION

The initial drive behind component mode synthesis was its ability to solve large size finite element models on small core computers with limited speeds. The computers of today have memories and speeds many times more than those of the 1960s and 1970s. But modal synthesis is still used, even more than before, because of the other advantages associated with it. One great benefit is to be able to partition a large model into smaller models thereby making the problem manageable. Also, different organizations can develop component models. In today's world, big projects of building a car, ship, airplane or space structure are often the combined effort of different organizations. The physical components or their finite element models are often built by different modelling groups. The integrator then combines the different models. If dynamic analysis is to be performed, as is often the case, modal synthesis is preferred.

Probably MSC/NASTRAN is used in modal synthesis more than any other finite element code. The author's experience shows that modal synthesis requires the use of database files (.DBALL and .MASTER) which are binary. Often different organizations use different versions of MSC/NASTRAN and different kind of computers making it potentially difficult to transfer database files. Therefore, analysts performing modal synthesis mostly use their own structured solution based DMAPs (Direct Matrix Abstraction Programs) which generate coded (readable) mass, stiffness, data recovery, load transformation matrices. These files can be transferred and be used with versions of MSC/NASTRAN different from the one that created them. This method of performing modal synthesis on one hand solves a problem and on the other hand creates a problem. Every organization has to have its own user manual for using these DMAPs. As these DMAPs are based on structured solutions, they have to be constantly updated and verified with changes in version of MSC/NASTRAN. This takes a lot of effort, time and money. These DMAPs and programmer imposed restrictions make the modal synthesis

technique more complex than it is. Also, the chance of errors in these DMAPs should not be over looked.

MSC/NASTRAN based component mode synthesis using internal superelements without the use of DMAPs can be found in Reference-1. The purpose of this paper is to show that the method can be extended to work for component mode synthesis using external superelements without the use of structured solution based DMAPs.

METHOD

For any method to be of wide use, it must:

- be capable of applying external loads
- allow plotting of deformed and undeformed system
- perform data recovery
- allow transferring data
- allow duplication of components

In component mode synthesis analysis, a system model is an assembly of component models. Each component consists of an interior and a boundary. The dynamic behavior of the interior is expressed in terms of the boundary. The boundary information is carried to the system level. Eigenvalue extraction, load application and boundary condition imposition can be done at the system level. In a way, a component mode model is like a huge element (super element). The only difference between a component mode model and a regular element (such as a beam element) is that while a regular element has only physical degrees of freedom a component mode model has physical degrees of freedom as well as generalized degrees of freedom associated with the dynamic behavior. From mathematics it can be shown that more the number of modes

(generalized degrees of freedom) in the components, closer is the solution based on modal synthesis to the solution had no super elements been used.

Since at the system level, only boundary information of the components is carried through, to apply loads and do displacement plot of a degree of freedom either of two things can be done. The displacement or applied load at a degree of freedom can be expanded in terms of the boundary, or the boundary can be moved to that degree of freedom (i.e. define it as a boundary degree of freedom).

Data recovery of elements/grids that are part of the residual structure (also called boundary) is straight forward. Data recovery of elements/grids in the interior of components requires special care. The following example illustrates this problem.

EXAMPLE

Figure-1 shows a spring-mass-damper system and its partitioning into superelements. The data files for the step by step modal synthesis are given in Figures 2 through 6. In STEP-1 the component mode models of the superelements are generated and the data files are shown in Figures 2 and 3. In STEP-2 the component mode models are synthesized to generate the system model and the data file is shown in Figure-4. Boundary condition imposition and eigenvalue extraction are performed in this step using solution sequence 103. STEP-3 is a restart run using solution sequence 112 and the data file for this step is shown in Figure-5. In this step loads and damping are applied and data recovery of elements and grids in the residual structure are performed. STEP-4 is a restart run using solution sequence 112 and the data file is shown in Figure-6. Data recovery of downstream super elements is performed in this step.

In order to have at least two elastic modes per superelement, spring k2 was modelled as two springs in series with each having stiffness 2*k2 and rotational inertias and stiffnesses were

modelled. The rotational modes from rotational masses and stiffnesses, did not participate in the final solution. Also, to have at least one interior degree of freedom per superelement, the loads were applied through springs with very high stiffness.

The problem was solved using MSC/NASTRAN Version 67.5. The data files are based on Reference-2. Analytical solution to the problem was obtained based on Reference-3. There was very good agreement as shown in Table-1.

EXTERNAL DATA TRANSFER

Forward and backward flow of data is achieved as follows. The organization that builds the component needs to send the integrator the mass and stiffness matrices associated with the component. A formatted file containing mass and stiffness matrices can be generated easily using a non-structured solution sequence based DMAP as in Figure-7. A corresponding database file (.DBALL) can be generated from this formatted file by using a non-structured solution sequence based DMAP as in Figure-8.

For downstream data recovery of an external superelement, one cannot transfer the database file (.DBALL) obtained by an outside organization in STEP-1. But one could convert the database file to a readable binary file using a non-structured solution based DMAP as in Figure-7. The contents of the datasets in the binary file are given in Reference-4 and can be converted to a coded form by writing a simple FORTRAN code. This coded file can be transferred. The coded file can be converted back to a binary form by writing another simple FORTRAN code. Finally, the binary file can be converted to a database file (.DBALL) by means of a non-structured solution based DMAP as in Figure-8. In reality, only a few datasets are required for downstream processing.

DUPLICATION OF COMPONENTS

For multiple-identical superelements, one can make use of the component mode matrices of a typical superelement. So the database file (.DBALL) generated in STEP-1 above can be converted to a readable file using a non-structured solution sequence based DMAP as in Figure-7. A database file (.DBALL) with the appropriate superelement number containing the datasets required can be generated from this binary file using a non-structured solution sequence based DMAP. For upstream processing, one simply needs the mass and stiffness matrices of the component. A typical DMAP for upstream database generation is shown in Figure-9. One can identify the datasets required for downstream processing and develop a similar DMAP.

DISCUSSION

The example shows that no DMAPs are required when performing modal synthesis in MSC/NASTRAN. But in cases where one has duplicate components, or one would like to communicate with external sites, one would need DMAPs based on non-structured solution sequences. These DMAPs are generally independent of MSC/NASTRAN version changes. In more complicated cases, where users wish to use selective modes, or users wish to include non-linear stiffening, or one has to interact with programs other than MSC/NASTRAN, one may have to use structured solution based DMAPs.

The method of defining degrees of freedom where displacement plot is required, or where an applied load exists should be carefully used. Increasing boundary degrees of freedom, also increases computation time and size of matrices. Degrees of freedom defined as boundary to obtain displacement plots to check system undeformed geometry and modes, should be removed when transient solution is required. One may even find a way to expand the displacement and

applied load at a interior degree of freedom in terms of boundary degrees of freedom without using any structured solution based DMAP.

One good feature of this method is that except the grid numbers at the boundary, the same grid, element, material numbers may exist in more than one component model.

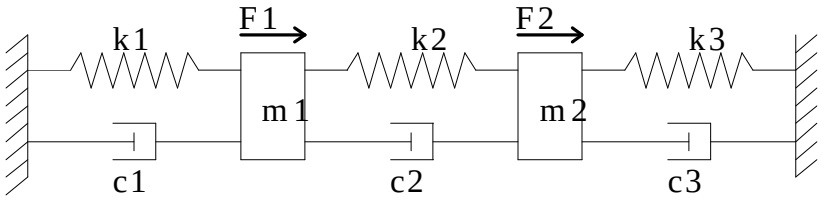
The idea brought out in this paper is to stay away from structured solution based DMAPs as much as possible when one is interacting with different organizations.

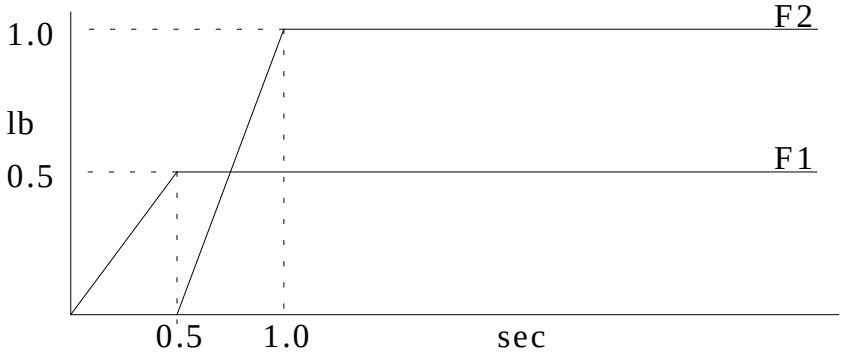
ACKNOWLEDGEMENT

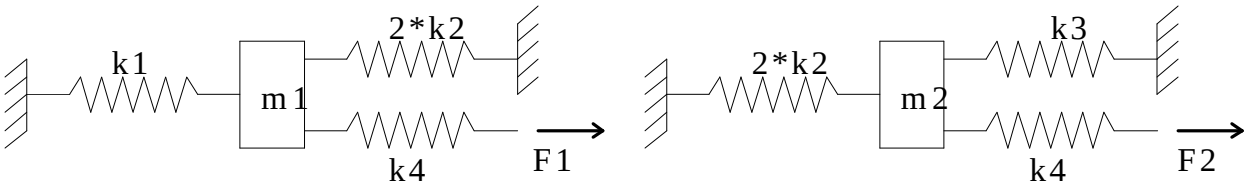
This work was performed at Rocketdyne Division, Rockwell International Corporation. The material presented here is based on work performed for the International Space Station under NASA contract HX3210. The author wishes to acknowledge the support of Rocketdyne, NASA management and Ted Rose of The MacNeal-Schwendler Corporation.

REFERENCES

1. MSC/NASTRAN Superelement Analysis Seminar Notes, Los Angeles, California, August 1994.
2. MSC/NASTRAN User's Manual, Version 67, Volumes I & II, The MacNeal Schwendler Corp., Los Angeles, California, August 1991.
3. Meirovitch, L., "Elements of Vibration Analysis", Second Edition, McGraw-Hill Book Co, NewYork, 1986.
4. MSC/NASTRAN Programmer's Manual, Volumes IV, The MacNeal Schwendler Corp., Los Angeles, California, March 1986.







SUPERELEMENT 1 **SUPERELEMENT 2**

m1 = 1.0 slug	k1 = 1.0 lb/in	c1 = 0.05 lb-sec/in
m2 = 2.0 slug	k2 = 1.0 lb/in	c2 = 0.05 lb-sec/in
	k3 = 2.0 lb/in	c3 = 0.10 lb-sec/in
	k4 = 1000.0 lb/in	

FIGURE-1: PROBLEM AND MODELLING

```
DBDIR
ID TGHOSH,SE10
SOL  103
TIME  1
CEND
ECHO     = UNSORT
TITLE    = CREATE EXTERNAL SUPERELEMENT MODEL OF S.E. 10
LABEL    = CMS OF SUPERELEMENT 10
SUBCASE 1
 METHOD   = 1
 SUPER    = 10
 DISP(PUNCH) = ALL
 STRESS(PUNCH) = ALL
 FORCE(PUNCH) = ALL
 SPCF(PUNCH) = ALL
$K2GG     = KAAEXT
$M2GG     = MAAEXT
BEGIN BULK
PARAM   GRDPNT  1
PARAM   LFREQ   .01
PARAM   HFREQ   10.
PARAM   FIXEDB  -1
PARAM   DBDICT   2
EIGRL   1       +0.00   2000.   10
SEQSET1 10      0       1001    THRU    1010
SPOINT  1001    THRU    1010
$ DEFINE BOUNDARY POINTS, POINTS WITH APPLIED LOADS AND POINTS FOR PLOTTING
$ AS BOUNDARY POINTS
SEBSET1,10,14,1,5
SECSET1,10,1,22
CELAS2,22,1000.,2,1,22,1
$
$ DEFINE SUPERELEMENT SEID=10
SESET   10      2
$
GRID,1,,0.,0.,0.,,2356    $ GRID WITH SINGLE POINT CONSTRAINT
GRID,2,,10.,0.,0.,,2356
GRID,22,,10.,0.,0.,,23456
GRID,5,,15.,0.,0.,,2356
$
CBEAM,1,1,1,2,10.,10.,0.  $ BEAM
CBEAM,2,1,2,5,10.,10.,0.
CONM2,4,2,,1.0,,,,,       $ CONCENTRATED MASS
,1.0
PBEAM,1,1,1.0,1.0,1.0,0.0,1.0,0.0,+PBEAM1
+PBEAM1,,,,,,
MAT1,1,10.,,0.3
ENDDATA
```

FIGURE-2: GENERATING COMPONENT MODE MODEL OF SUPERELEMENT 1

```
ID TGHOSH,SE20
SOL  103
TIME  1
CEND
ECHO    = UNSORT
TITLE   = CREATE EXTERNAL SUPERELEMENT MODEL OF S.E. 20
LABEL   = CMS OF SUPERELEMENT 20
SUBCASE 1
 SUPER  = 20
 METHOD = 1
 DISP   = ALL
 STRESS = ALL
 FORCE  = ALL
 SPCF   = ALL
BEGIN BULK
PARAM   GRDPNT  1
PARAM   LFREQ   .01
PARAM   HFREQ   10.
PARAM   FIXEDB  -1
EIGRL   1       +0.00   2000.   10
SEQSET1 20      0       2001    THRU    2010
SPOINT  2001    THRU    2010
SEBSET1,20,14,4,5
SECSET1,20,14,33
CELAS2,33,1000.,3,1,33,1
CORD2R        20       0   0.000   0.000   0.000   0.000   0.000   1.000+
+         -1.000   0.000   0.000
$
$ DEFINE SUPERELEMENT SEID=20
$
SESET   20      3
$
GRID,3,,-20.,0.,0.,20,2356
GRID,33,,-20.,0.,0.,,23456
GRID,4,,-25.,0.,0.,20,2356
GRID,5,,-15.,0.,0.,20,2356
$
CBEAM,3,1,3,4,-10.,-10.,0.  $ BEAM
CBEAM,6,1,3,5,-10.,-10.,0.
CONM2,5,3,,2.0,,,,,       $ CONCENTRATED MASS
,1.0
$
PBEAM,1,1,1.0,1.0,1.0,0.0,1.0,0.0,+PBEAM1
+PBEAM1,,,,,,
MAT1,1,10.,,0.3
ENDDATA
```

FIGURE-3: GENERATING COMPONENT MODE MODEL OF SUPERELEMENT 2

```
ASSIGN SE10='se10.MASTER'
ASSIGN SE20='se20.MASTER'
DBLOCATE DB=* LOGI=se10 WHERE(SEID=10 OR PEID=10)
DBLOCATE DB=* LOGI=se20 WHERE(SEID=20 OR PEID=20)
ID TKGHOSH,SYSTEM
SOL    103
TIME    10
CEND
TITLE    = SYSTEM LEVEL SOLUTION
SUBCASE 1
METHOD = 1
SUPER = 10
SUBCASE 2
METHOD = 1
SUPER = 20
SUBCASE 3
METHOD   = 2
SPC      = 1
DISP(PLOT)= ALL
OUTPUT(PLOT)
SET 1    = ALL
SEPLOT 0  $ PLOT RESIDUAL STRUCTURE ONLY
PTITLE = RESIDUAL STRUCTURE
AXES X,Y,Z
VIEW 34.27,23.17,0.0
FIND SCALE ORIGIN 1    SET 1
PLOT SET 1  ORIGIN 1
PLOT MODAL DEFORMATION SET 1 ORIGIN 1
BEGIN BULK
$ THE FOLLOWING PARAMS ARE REQUIRED FOR PLOTTING
PARAM,PLOTSUP,0
PARAM,POST,0
PARAM,NEWSEQ,-1
EIGRL   2       +0.00   2000.   10
SPC1    1       123456  1       4
EIGRL   1       +0.00   2000.   10
CORD2R       100       0   0.000   0.000   0.000   0.000   0.000   1.000+
+          1.000   0.000   0.000
CORD2R       200       0   0.000   0.000   0.000   0.000   0.000   1.000+
+         -1.000   0.000   0.000
SPOINT  1001    THRU    1010
CSUPER  10      0       1       5       22      1001    1002    1003
      1004    1005    1006    1007    1008    1009    1010
CSUPER  20      0       4       5       33      2001    2002    2003
      2004    2005    2006    2007    2008    2009    2010
GRID,1,100,0.,0.,0.,,2356    $ GRID WITH SINGLE POINT CONSTRAINT
GRID,22,100,10.,0.,0.,,23456
GRID,5,100,15.,0.,0.,,2356
SPOINT  2001    THRU    2010
GRID,33,200,-20.,0.,0.,,23456
GRID,4,200,-25.,0.,0.,,2356
PLOTEL  101     1       22
PLOTEL  102     22      5
PLOTEL  201     4       33
PLOTEL  202     33      5
ENDDATA
```

FIGURE-4: GENERATING SYSTEM MODEL

```
NASTRAN BUFFSIZE=12289
INIT DBALL LOGICAL=(DBALL(25000))
ASSIGN SE10='se10.MASTER'
ASSIGN SE20='se20.MASTER'
DBLOCATE DB=* LOGI=SE10 WHERE(SEID=10 OR PEID=10)
DBLOCATE DB=* LOGI=SE20 WHERE(SEID=20 OR PEID=20)
RESTART VERSION=1, KEEP
ID TKGHOSH,RESP
SOL  112
TIME  10
CEND
TITLE    = MODAL TRANSIENT FOR SUPERELEMENT MODEL
PARAM,SERST,SEMI
SET 100 = 0
SELG = 100
SELR = 100
SUBCASE 1000
 LOADSET= 10
 SDAMP  = 1
 TSTEP  = 201
 DLOAD  = 10
 METHOD   = 2
 DISP     =ALL
 SPC      = 1
BEGIN BULK
$
$   LOADSET   DAREA   FORCE
$     SET      SET     ID
$     .....   ......  .....
LSEQ    10      1       101
LSEQ    10      2       102
$
TABDMP1,1,CRIT,,,,,,,+AAA1
+AAA1,.1591549,.025,.2516398,.03959,ENDT
$
TSTEP,201,8,.25,1
$
DLOAD       10      1.      1.     101      1.     102
$
TLOAD1,101,1,,,1
TLOAD1,102,2,,,2
$
FORCE,101,22,0,1.0,1.0,0.0,0.0
FORCE,102,33,0,1.0,1.0,0.0,0.0
$
TABLED1*             1                                  *C     0
*C     0                                               *C     1
*C     1  0.00000000E+00  0.00000000E+00  0.50000000E+00  0.50000000E+00*C     2
*C     2  1.00000000E+00  0.50000000E+00  1.50000000E+00  0.50000000E+00*C     3
*C     3  2.00000000E+00  0.50000000E+00ENDT
TABLED1*             2                                  *C  1877
*C  1877                                               *C  1878
*C  1878  0.00000000E+00  0.00000000E+00  0.50000000E+00  0.00000000E+00*C  1879
*C  1879  1.00000000E+00  1.00000000E+00  1.50000000E+00  1.00000000E+00*C  1880
*C  1880  2.00000000E+00  1.00000000E+00ENDT
ENDDATA
```

FIGURE-5: SYSTEM MODEL TRANSIENT RUN

```
ASSIGN SE10='se10.MASTER'
ASSIGN SE20='se20.MASTER'
DBLOCATE DB=* LOGI=se10 WHERE(SEID=10 OR PEID=10)
DBLOCATE DB=* LOGI=se20 WHERE(SEID=20 OR PEID=20)
RESTART VERSION=LAST
ID TKGHOSH,RESP
SOL  112
TIME  10
CEND
TITLE    = MODAL TRANSIENT FOR SUPERELEMENT MODEL
PARAM,SERST,SEDR
SET 99 = 10,20
SEDR = 99
$SUPER = ALL
SUBCASE 1
METHOD = 1
SUPER = 10
DISP  = ALL
STRESS = ALL
FORCE = ALL
SPCF = ALL
$
SUBCASE 2
SUPER = 20
METHOD = 1
DISP = ALL
STRESS = ALL
FORCE = ALL
SPCF = ALL
SUBCASE 1000
 LOADSET= 10
 SDAMP  = 1
 TSTEP  = 201
 DLOAD  = 10
 METHOD   = 2
 DISP     =ALL
 SPC      = 1
BEGIN BULK
PARAM,DBDICT,2
$PARAM,DDRMM,-1
$
ENDDATA
```

FIGURE-6: DATA RECOVERY IN COMPONENT MODELS

```
ASSIGN OUTPUT4='craign1.f12',NEW,UNIT=12,FORMATTED
ASSIGN OUTPUT2='craign1.f11',NEW,UNIT=11,UNFORMATTED
RESTART VERSION=LAST,KEEP
DBDIR
ID GENERATE MATRICES
TIME 10
$SOL 100
COMPILER LIST,REF,DECK
DIAG 8
DIAG 31
SOL USERDMAP
COMPILE USERDMAP SOUIN=MSCSOU
ALTER 2
PUTSYS(1,125) $ ALLOW FOR LATER RESTART WITHOUT WARNING
TYPE PARM,NDDL,I,N,SEID,PEID,MPC,SPC,METH,DYRD,HIGHQUAL,NLOOP  $
TYPE PARM,NDDL,I,N,MTEMP,DESITER,MFLUID,ZUZR1,ZUZR2,ZUZR3 $
TYPE DB MAA,KAA,MLAA,KLAA $
TYPE PARM,NDDL,CHAR8,N,K2GG,M2GG
TYPE DB CMPHA,CMPHO,GOAQ,GOAT $
TYPE DB KFS,KGG,KJJ $
TYPE DB KLL,KOO,KSF,KSS $
TYPE DB KVV,LOO,MFF,KELM $
TYPE DB MGG,MJJ,MLAA1,MELM $
TYPE DB EQEXINS,EQEXINX,MPTS $
TYPE DB BGPDTS,BGPDTX,BULK,CASES $
TYPE DB CMLAMA,DYNAMICS,ECTS,ECTX $
TYPE DB EMAP,EPT,EPTS,EQEXINS $
TYPE DB EQEXINX,EST,GPDTS,GPECT $
TYPE DB GPLS,KDICT,MPT,MPTS $
TYPE DB PHG,PHQG,PVT,PVTS $
TYPE DB SILS,SILX,SLIST,TIMSIZ $
TYPE DB USET,VELEM,VGFS,MDICT $
TYPE DB GEOM1,GEOM1Q,GEOM1S,GEOM2 $
TYPE DB GEOM2S,GEOM4,GEOM4S $
$
$ SUPERELEMENT ID (SEID 10)
SEID    =   10    $
PEID    =   10    $
MPC     =   0     $
SPC     =   0     $
METH    =   1     $
DYRD    =   0     $
HIGHQUAL=   0     $
$K2GG    ='KAAEXT' $
$M2GG    ='MAAEXT' $
NLOOP   =  -1     $
```

FIGURE-7: CONVERTING DATABASE TO READABLE FILE

```
MTEMP   =   0     $
DESITER =   0     $
MFLUID  =   0     $
OUTPUT4  KAA,KLAA,MAA,MLAA//-1/12/-1 $
OUTPUT4  CMPHA,CMPHO,GOAQ,GOAT//0/12/-1 $
OUTPUT4  KFS,KGG,KJJ//0/12/-1 $
OUTPUT4  KLL,KOO,KSF,KSS//0/12/-1 $
OUTPUT4  KVV,LOO,MFF,KELM//0/12/-1 $
OUTPUT4  MGG,MJJ,MLAA1,MELM//0/12/-1 $
OUTPUT2  BGPDTS,BGPDTX,BULK,CASES,//-1/11// $
OUTPUT2  CMLAMA,DYNAMICS,ECTS,ECTX,//0/11// $
OUTPUT2  EMAP,EPT,EPTS,EQEXINS,//0/11// $
OUTPUT2  EQEXINX,EST,GPDTS,GPECT,//0/11// $
OUTPUT2  GPLS,KDICT,MPT,MPTS,//0/11// $
OUTPUT2  PHG,PHQG,PVT,PVTS,//0/11// $
OUTPUT2  SILS,SILX,SLIST,TIMSIZ,//0/11// $
OUTPUT2  USET,VELEM,MDICT,VGFS,//0/11// $
OUTPUT2  GEOM1,GEOM1Q,GEOM1S,GEOM2,//0/11// $
OUTPUT2  GEOM2S,GEOM4,GEOM4S,,//0/11// $
DBDIR  // $
CEND
TITLE = Read and store matrices for superelement 10
BEGIN BULK
ENDDATA
```

FIGURE-7: CONVERTING DATABASE TO READABLE FILE (CONTD.)

```
$ Create Data base for storing M,K matrices
$ for external superelement 10
$
$ Assign INPUTT4 statements for attaching M,K files
ASSIGN INPUTT4='craign1.f12',UNIT=12,FORMATTED
ASSIGN INPUTT2='craign1.f11',UNIT=11,UNFORMATTED
ID GENERATE MATRICES
TIME 10
COMPILER LIST,REF,DECK
DIAG 8
DIAG 31
SOL USERDMAP
COMPILE USERDMAP SOUIN=MSCSOU
ALTER 2
PUTSYS(1,125) $ ALLOW FOR LATER RESTART WITHOUT WARNING
TYPE PARM,NDDL,I,N,SEID,PEID,MPC,SPC,METH,DYRD,HIGHQUAL,NLOOP  $
TYPE PARM,NDDL,I,N,MTEMP,DESITER,MFLUID,ZUZR1,ZUZR2,ZUZR3 $
$TYPE PARM,NDDL,I,N,SEID,PEID $
TYPE DB MAA,KAA,MLAA,KLAA $
TYPE DB CMPHA,CMPHO,GOAQ,GOAT $
TYPE DB KFS,KGG,KJJ $
TYPE DB KLL,KOO,KSF,KSS $
TYPE DB KVV,LOO,MFF,KELM $
TYPE DB MGG,MJJ,MLAA1,MELM $
TYPE DB BGPDTS,BGPDTX,BULK,CASES $
TYPE DB CMLAMA,DYNAMICS,ECTS,ECTX $
TYPE DB EMAP,EPT,EPTS,EQEXINS $
TYPE DB EQEXINX,EST,GPDTS,GPECT $
TYPE DB GPLS,KDICT,MPT,MPTS $
TYPE DB PHG,PHQG,PVT,PVTS $
TYPE DB SILS,SILX,SLIST,TIMSIZ $
TYPE DB USET,VELEM,VGFS,MDICT $
TYPE DB GEOM1,GEOM1Q,GEOM1S,GEOM2 $
TYPE DB GEOM2S,GEOM4,GEOM4S $
$
$ SUPERELEMENT ID (SEID 10)
SEID    =   10    $
PEID    =   10    $
MPC     =   0     $
SPC     =   0     $
METH    =   1     $
DYRD    =   0     $
HIGHQUAL=   0     $
$K2GG    ='KAAEXT' $
```

FIGURE-8: CONVERTING READABLE FILE TO DATABASE

```
$M2GG    ='MAAEXT' $
NLOOP   = -1     $
MTEMP   =  0     $
DESITER =  0     $
MFLUID  =  0     $
INPUTT4  /KAE1,KLAE1,MAE1,MLAE1,/4/12/-1/-1 $
$ GENERATING PARTITIONING VECTOR
MATGEN   /CPE1/6/28/28/0/0 $ FOR MUTIPLE OF 6
MERGE    KAE1,,,,CPE1,/KAA/-1//6 $
MERGE    MAE1,,,,CPE1,/MAA/-1//6 $
MERGE    KLAE1,,,,CPE1,/KLAA/-1//6 $
MERGE    MLAE1,,,,CPE1,/MLAA/-1//6 $
INPUTT4  /CMPHA,CMPHO,GOAQ,GOAT,/4/12/0/-1  $
INPUTT4  /KFS,KGG,KJJ,,/3/12/0/-1  $
INPUTT4  /KLL,KOO,KSF,KSS,/4/12/0/-1  $
$INPUTT4  /KVV,LOO,MFF,KELM,/4/12/0/-1  $
INPUTT4  /KVV,LOO,MFF,,/3/12/0/-1  $
$INPUTT4  /MGG,MJJ,MLAA1,MELM,/4/12/0/-1  $
INPUTT4  /MGG,MJJ,MLAA1,,/3/12/0/-1  $
INPUTT2  /BGPDTS,BGPDTX,BULK,CASES,/-1/11 $
INPUTT2  /CMLAMA,DYNAMICS,ECTS,ECTX,/0/11 $
INPUTT2  /EMAP,EPT,EPTS,EQEXINS,/0/11 $
INPUTT2  /EQEXINX,EST,GPDTS,GPECT,/0/11 $
INPUTT2  /GPLS,KDICT,MPT,MPTS,/0/11 $
INPUTT2  /PHG,PHQG,PVT,PVTS,/0/11 $
INPUTT2  /SILS,SILX,SLIST,TIMSIZ,/0/11 $
INPUTT2  /USET,VELEM,MDICT,,/0/11 $
$INPUTT2  /USET,VELEM,MDICT,VGFS,/0/11 $
INPUTT2  /GEOM1,GEOM1Q,GEOM1S,GEOM2,/0/11 $
INPUTT2  /GEOM2S,GEOM4,GEOM4S,,/0/11 $
DBDIR  // $
CEND
TITLE = Read and store matrices for external superelement 10
BEGIN BULK
PARAM   DBDICT  2
ENDDATA
```

FIGURE-8: CONVERTING READABLE FILE TO DATABASE (CONTD.)

```
$ Assign INPUTT4 statements for attaching M,K files
ASSIGN INPUTT4='mkiea.f12',UNIT=12,FORMATTED
TIME 10
SOL 100
COMPILE USERDMAP SOUIN=MSCSOU
ALTER 2
PUTSYS(1,125) $ ALLOW FOR LATER RESTART WITHOUT WARNING
TYPE PARM,NDDL,I,N,SEID,PEID $
TYPE DB MAA,KAA,MLAA,KLAA $
$
$ SUPERELEMENT 431 (SEID 431)
$
SEID=431 $
PEID=431 $
INPUTT4  /KAE1,KLAE1,MAE1,MLAE1,/4/12/-1/-1 $
$ GENERATING PARTITIONING VECTOR
MATGEN   /CPE1/6/120/120/0/0 $ FOR MUTIPLE OF 6
MERGE    KAE1,,,,CPE1,/KAA/-1//6 $
MERGE    MAE1,,,,CPE1,/MAA/-1//6 $
MERGE    KLAE1,,,,CPE1,/KLAA/-1//6 $
MERGE    MLAE1,,,,CPE1,/MLAA/-1//6 $
$ ######################################################################
$ ######################################################################
$
$ SUPERELEMENT 432 (SEID 432)
$
SEID=432 $
PEID=432 $
INPUTT4  /KAE2,KLAE2,MAE2,MLAE2,/4/12/-1/-1 $
$ GENERATING PARTITIONING VECTOR
MATGEN   /CPE2/6/120/120/0/0 $ FOR MUTIPLE OF 6
MERGE    KAE2,,,,CPE2,/KAA/-1//6 $
MERGE    MAE2,,,,CPE2,/MAA/-1//6 $
MERGE    KLAE2,,,,CPE2,/KLAA/-1//6 $
MERGE    MLAE2,,,,CPE2,/MLAA/-1//6 $
$ ######################################################################
DBDIR // $
$
CEND
TITLE = READ AND STORE MATRICES IN DATA BASE
BEGIN BULK
ENDDATA
```

FIGURE-9: GENERATING UPSTREAM DATABASE

TABLE-1: SYSTEM RESPONSE AT TIME = 2.0 SEC

ITEM	UNIT	MSC/NASTRAN	ANALYTICAL
m1 displacement	inch	0.47926	0.47926
m2 displacement	inch	0.38349	0.38349
k1 force	lb	0.47926	0.47926
k2 force	lb	-0.09578	-0.09578
k3 force	lb	-0.76697	-0.76697