

REPRODUCIBILITY OF THE ORIGINAL PAGE IS POOR.

N74-14606

PLACING THREE-DIMENSIONAL ISOPARAMETRIC ELEMENTS INTO NASTRAN

By M. B. Newman and A. W. Filstrup

Westinghouse Research and Development Center

SUMMARY

Linear (8 node), parabolic (20 node), cubic (32 node) and mixed (some edges linear, some parabolic and some cubic) have been inserted into NASTRAN, Level 15.1. First the dummy element feature was used to check out the stiffness matrix generation routines for the linear element in NASTRAN. Then, the necessary modules of NASTRAN were modified to include the new family of elements. The matrix assembly was changed so that the stiffness matrix of each isoparametric element is only generated once as the time to generate these higher order elements tends to be much longer than the other elements in NASTRAN. This paper presents some of the experiences and difficulties of inserting a new element or family of elements into NASTRAN.

INTRODUCTION

In solving many structural problems at Westinghouse, it has become apparent that in order to obtain the accuracy required, three-dimensional finite elements would be required. It also became apparent that three-dimensional finite elements based on constant strain tetrahedra like the CTETRA, CWEDGE, CHEX1 and CHEX2 elements in NASTRAN are too stiff to give accurate results at a reasonable cost for many problems.

Because of this, a Westinghouse proprietary program, WISEC, was developed for heat conduction and static linear elastic analysis using three dimensional isoparametric elements. Because of the large general purpose capability of NASTRAN, both for types of problems solved and for types of elements used, it was decided to place these elements into NASTRAN.

Even though three-dimensional isoparametric elements were then to be and now have been placed into NASTRAN by Dr. E. I. Field and Mr. S. E. Johnson of Universal Analytics (see Ref. 1), and are to be included in Level 16 NASTRAN now scheduled to be released in 1974, it was decided to place three-dimensional isoparametric elements into Level 15.1 NASTRAN. First we would have use of this element in NASTRAN at an earlier date than we would if we waited until Level 16 was released. Second, we would gain experience and familiarity with NASTRAN which would enable us to more easily make any future modifications which we would desire. A third benefit, which we didn't realize at the time, is the

PRECEDING PAGE BLANK NOT FILMED

fact that the family of elements we added can have different number of grid or nodal points on the various edges as shown in figure 1. As we understand, Level 16 NASTRAN will have elements which are either linear (2 points on each edge), parabolic (3 points on each edge) or cubic (4 points on each edge). Mixed elements, like that shown in figure 1, can be used to reduce the number of degrees of freedom in portions of the structure not requiring the higher order elements without introducing incompatibilities between adjacent elements. The order of an element is taken to be that of its highest ordered edge.

As the theory of three-dimensional isoparametric elements is explained elsewhere, for instance in Refs. 1 to 4, it will not be repeated here.

At the present time, the stiffness and mass matrices have been successfully inserted and tested. The differential stiffness matrix is due to be added shortly.

The work described in this report was performed with Level 15.1 NASTRAN on an IBM 370-165. It is planned to insert the changes into Level 15.5 NASTRAN on a CDC 6600.

RECOMMENDATIONS

1. For anyone making changes in NASTRAN, an up-to-date Programmer's Manual is of great aid. Unfortunately, the latest available Programmer's Manual is not always for the latest available level of NASTRAN.
2. Many of the tables present in Level 15.1 NASTRAN are too short to permit elements with as many degrees of freedom as the isoparametric elements. These tables should be increased in length to permit easier insertion of new elements.

METHOD AND EXPERIENCES

NASTRAN is an extremely large system comprised of fifteen super links with approximately 850 subprograms whose source statements are on over 200,000 card images. NASTRAN is indeed a very large and complex program and, at first glance, a dense forest that seems too difficult to enter. As one starts to review the large NASTRAN Programmer's Manual (approximately 15 centimeters thick) and examine the materials the authors of NASTRAN have distributed, the forest does not seem as dense. This section of the paper describes our experiences in adding new elements to the NASTRAN system.

The three-dimensional isoparametric elements added presented many problems that the usual NASTRAN elements did not encounter. The tables were much larger, for example. The number of nodes that described our cubic isoparametric element varies from ten to thirty-two nodes. This number forced us to expand the Element Connection and Properties Tables and other array sizes that dealt with nodes. The concept of a variable number of nodes per element was also a departure from the usual NASTRAN practice of a constant number of nodes per element type.

For these higher order elements, the computer time necessary to create the element matrices was quite large; hence, a procedure to create the element matrices once and to save them had to be incorporated into the element level subprograms.

The release we used to incorporate the new elements was level 15.1. The computer used was an IBM 370-165 operating under the ASP system. The Programmer's Manual we had was for Level 12 which caused some difficulty but not too much. We will outline the procedure we used in adding the new elements.

First one should review the materials distributed with the Level 15 system. Figure 2 is a VTOC (Volume Table of Contents) of the distributed system. Pages 5.3-13 and 5.3-14 of the Programmer's Manual (ref. 5) describes each of the data sets of the distributed system. The data sets which are most useful to us are SOUL, the partitioned data set containing the FORTRAN source programs, SUBSYS, also a partitioned data set containing the linkage editor control cards for the fifteen super links of NASTRAN; the partitioned data set OBJ, which contains all the load modules of each individual subroutine of the system; the partitioned data set NSTNLMOD, which contain the fifteen link-edited super links which constitute the NASTRAN executable set.

The next step of the procedure should be to set up a development disk with at least two data sets which we named NEWOBJ and NADEV. NEWOBJ corresponds to OBJ, and NADEV corresponds to NSTNLMOD. It would be advisable to set up a data set corresponding to SOUL but we elected to keep all of our new source programs in card-deck form. The IBM 370 utility program IEHMOVE or IEBCOPY can be used to move the fifteen link edited links from NSTNLMOD to the development pack. NADEV's initial allocation should be as large as possible as this data set will be modified frequently. An alternate approach, which we did not use but one that could have saved us some grief would be to set up fifteen different data sets rather than one partitioned data set with fifteen members. Then each time we needed to link-edit, we would scratch the particular data set and recreate the new link edited data set (instead of member). This procedure would keep us from using up all the extents of a partitioned data set and not having to compress the partitioned set which we had to do approximately every twenty to twenty-five re-link edits. Figure 3 is the VTOC of our development disk. The other utility that we made quite frequent use of was IEBTPCH. With the use of this utility we can either list or punch a member of SOUL or any of the other partitioned data sets. The JCL for PUNCHIT is given in Figure 4 and for PRINIT in Figure 5. With these two decks we can list or punch subroutines from SOUL. The punched routine could then be modified for our new element. Another utility which could have been used for modifying source decks is IEBUPDATE which we did not use. The next step in the process is to compile either a modified subprogram of the NASTRAN system or to compile one of our new subprograms. The compiled program is placed into our partitioned data set NEWOBJ. The JCL for this procedure is shown in Figure 6. When all the decks for one of the links has been compiled, the next step is to link edit this link.

The link editor allows one to specify a group of libraries of programs via the LKED.XXX DD cards. In our case, we described two libraries, OBJ which

contained all the original unmodified or distributed load modules and NEWOBJ, the modified and new load modules. Each library is given a DD name, for the partitioned set OBJ the name LKED.LIB is used and for NEWOBJ we chose to use the name LKED.LIP. The overlay control cards can be punched and listed from the data set SUBSYS for this link. The control card deck is then modified to reflect the modifications made to the link. See Figure 7 for the JCL and modified control deck for the Link Edit step. The SYSLMOD DD card defines the output load module for the linkage editor and places the load module in the data set NADEV.

The next step is to run a NASTRAN problem to test the procedure implemented. Alters can be made to the DMAP program to extract contents of tables or of generated matrices. In addition, print statements can be made within the modified programs to print out calculated results. If these print statements are used, they should be activated by a specific DIAG that is not in use by the NASTRAN system. See pages 3.3-15 and 3.3-15a of the NASTRAN Programmer's Manual for DIAGS not in use by the system. Figure 8 gives an example of the use of alter statements and demonstrates the use of DIAG setting for controlling debug printing. In debugging a modified link, a dump is quite helpful on the occurrence of a system fatal error. The most important part of the dump is the save area trace which lists the routines last used when the error occurred. Usually this is sufficient and a full dump is not necessary. NASTRAN has built into the system a use of the SNAP macro which dumps only the save areas. Use of DIAG 1 will produce a full dump.

The link edit procedure for NASTRAN links is rather costly on the IBM 370-165 because of the extremely large number of segments in each of the links. Hence, whenever possible, we did as much checking of a modified module with a nonoverlaid FORTRAN run. For example, in checking out the stiffness matrix routines for the isoparametric elements we ran a simple model in NASTRAN, and with the Alter statements we printed the contents of the ECPT (Element Connection and Properties Tables). A main program which simulates SMA1 was written to supply the proper ECPT to the element stiffness matrix routines and the element matrices were generated and printed out. When we were satisfied that the routines operated properly we modified our link-edit control deck and link edited the new element stiffness matrix routines into our data set NADEV. A run of the same model would then produce the element stiffness matrices, displacements, and stresses. Figure 7 is our JCL and control deck for the insertion of stiffness matrix routines into link 3.

The procedure for the variable number of grid points for mixed elements (one that is not full) was implemented in the following manner. The connection cards for the element were left blank at positions where grid points were missing from the full element. A modification of TALA and TALB was made to enter a zero as the grid point number for the missing grid points. For the grid points present, the degree of freedom for that grid point (a nonzero value) was entered as the grid point number. All tables such as ECPT, EST (Element Summary Table) have nonzero values for grid points present in the element and zeros for missing grid points. The length of the grid point table is fixed for each element type, for example, twenty for CSOLID2, the quadratic isoparametric. This table is then used as a guide to all processing of the mixed element. The modifications to TALA and TALB were supplied to us by Carl Hennrich of MacNeal-Schwendler Company.

The procedure used for saving the element matrices and not recreating them each time they are needed was as follows: A scratch tape was assigned to be used in the element matrix subroutine. This tape had to be assigned a GINO buffer at a level where all buffers are assigned for this module. Also an array had to be assigned for record keeping of saved elements. Initialization of counters had to be done at the level where the buffer was assigned. At the element level the routine would first ask if this element had previously been encountered. This is done by a search through the table of all elements that have been saved. If found, the tape record number is extracted from the table and the tape is positioned by GINO commands to the proper record. The record is read into core and the sub matrices needed for this call are assembled from the total element matrix and given to the subroutine which is assembling the total mass or stiffness matrix. If this element has not been encountered, the element matrix is calculated, and the tape is positioned to the end of last element written, and the new element matrix is written on the scratch tape. The element number and record number is entered into the table. This procedure was suggested in the Programmers Manual, Page 4.87-1, last two sentences of paragraph 3. GINO proved very useful here in that the records saved were of variable length because of the three types of elements and because of the use of mixed elements within a type. The variable length could be stored in the record and using GINO's capability of reading and writing segmented or partial records we could read the number of words for the variable length record. To add scratch tapes to an existing module the MPL (Module Property List) had to be modified by recompiling the block data program XMPLBD, see pages 2.4-21 and 2.4-22 of the Programmers Manual.

DISCUSSION

The new Programmer's Manual for version 15.1 has an excellent chapter on adding a structural element. This was an update of a NASA Fourteenth Quarterly Report for NASTRAN, January 1970. This chapter gives a step-by-step procedure of all routines and tables that have to be modified to accommodate a new element. The Fourteenth Quarterly Report aided us greatly in getting through most of the input problems of NASTRAN.

From this step-by-step procedure, one can see that adding a new element to NASTRAN is not that difficult because of the excellent documentation and supplies that have been distributed.

ACKNOWLEDGMENTS

The authors would like to thank S-Y Lien and W. VanBuren for providing the isoparametric routines and C. W. Hennrich, R. Gillian, and M. M. Hurwitz for NASTRAN consultation.

REFERENCES

1. Field, E. I., and Johnson, S. E.: Addition of Three-Dimensional Isoparam Elements to NASA Structural Analysis Program (NASTRAN). NASA CR-112269 1973.
2. Zienkiewicz, O. C.: The Finite Element Method in Engineering Science. McGraw-Hill Book Co., Inc., 1971, pp. 129-153.
3. Zienkiewicz, O. C., Irons, B. M., Ergatoudis, J., Ahmad, S. and Scott, F. Iso-Parametric and Associated Element Families for Two- and Three-Dimensional Analysis. Chapter 13 of Finite Elements in Stress Analysis I. Holand and K. Bell, eds., Tapir Forlag (Trondheim, Norway) 1969.
4. MacNeal, R. H., ed.: The NASTRAN Theoretical Manual. NASA SP-221(01), 1972.
5. Anon.: The NASTRAN Programmer's Manual. NASA SP-223(01), 1972.

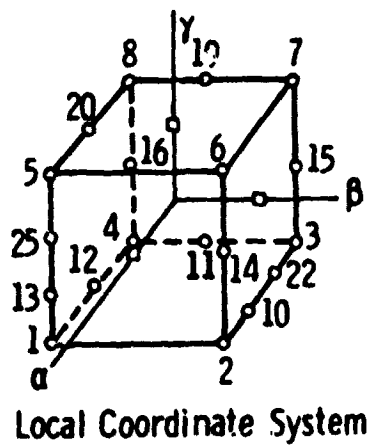
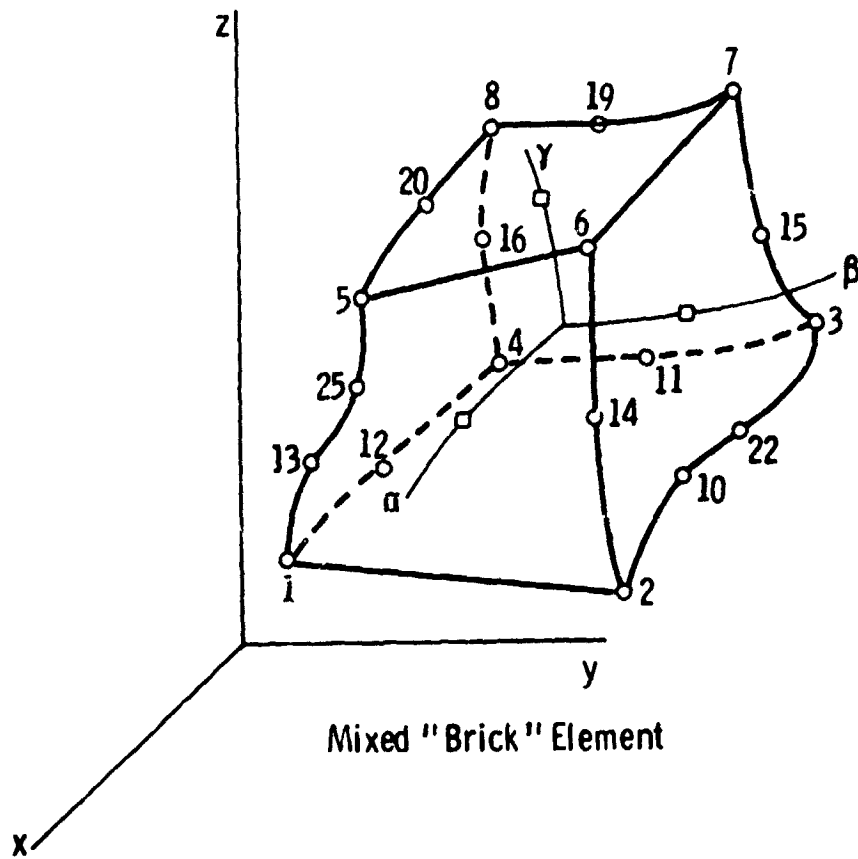


Figure 1.- Mixed 3-D isoparametric element.
Variable number of grid points.

VOLUME FORMAT 4 DSCB AT 0/ 2/ 1 (CC/MM/SS) 14/31/09-90 73-165
 EXTENT OF VTOC - FROM 0/ 2 TO 0/ 4 SEQUENCE NUMBER = 0 CODE = 01 3 TRACKS ALLOCATED
 DEVICE CHARACTERISTICS NUMBER OF LOGICAL CYLINDERS = 203
 NUMBER OF TRACKS PER LOGICAL CYLINDER = 20
 NUMBER OF PAGES PER TRACK = 2294
 NUMBER OF FULL CLOCKS PER TRACK = 25
 NUMBER OF PDS DIRECTORY BLOCKS PER TRACK = 17

FORMAT 5 DSCB AT 0/ 2/ 2 (CC/MM/SS) EXTENTS OF AVAILABLE SPACE
 STARTING AT RELATIVE TRACK 1 1 OF 11 THERE ARE 0 CYLINDERS AND AN ADDITIONAL 1 TRACKS AVAILABLE
 STARTING AT RELATIVE TRACK 31 OF 31 THERE ARE 1 CYLINDERS AND AN ADDITIONAL 12 TRACKS AVAILABLE
 A TOTAL OF 33 TRACKS AVAILABLE

INSTALMENT FORMAT 1 DSCB AT 0/ 2/ 3 (CC/MM/SS) RECORD FORMAT IS L
 NUMBER OF EXTENTS = 1 DATA SET ORGANIZATION IS PARTITIONED
 BLOCKSIZE = 7296 LOGICAL RECORD LENGTH = 0
 SECONDARY ALLOCATION - CODE = 00 QUANTITY = 20
 THE LAST BLOCK WAS WRITTEN ON RELATIVE TRACK 50 AS RECORD NUMBER 0
 THERE ARE 5744 BYTES REMAINING ON THIS TRACK FOLLOWING THE BLOCK
 EXTENT 1 FROM 517 15 TO 517 16 SEQUENCE NUMBER = 0 CODE = 01 1032 TRACKS ALLOCAT

OBJ FORMAT 1 DSCB AT 0/ 2/ 4 (CC/MM/SS) RECORD FORMAT IS U
 NUMBER OF EXTENTS = 2 DATA SET ORGANIZATION IS PARTITIONED
 BLOCKSIZE = 7296 LOGICAL RECORD LENGTH = 7296
 SECONDARY ALLOCATION - CODE = 00 QUANTITY = 2
 THE LAST BLOCK WAS WRITTEN ON RELATIVE TRACK 50 AS RECORD NUMBER 9
 THERE ARE 3197 BYTES REMAINING ON THIS TRACK FOLLOWING THE BLOCK
 EXTENT 1 FROM 517 15 TO 517 16 SEQUENCE NUMBER = 0 CODE = 01 508 TRACKS
 EXTENT 2 FROM 517 17 TO 517 18 SEQUENCE NUMBER = 1 CODE = 01 2 TRACKS
 TOTAL OF 508 TRACKS ALLOCAT

SOU1 FORMAT 1 DSCB AT 0/ 2/ 5 (CC/MM/SS) RECORD FORMAT IS PB
 NUMBER OF EXTENTS = 1 DATA SET ORGANIZATION IS PARTITIONED
 BLOCKSIZE = 7296 LOGICAL RECORD LENGTH = 80
 SECONDARY ALLOCATION - CODE = 00 QUANTITY = 2
 THE LAST BLOCK WAS WRITTEN ON RELATIVE TRACK 20 AS RECORD NUMBER 2
 THERE ARE 4822 BYTES REMAINING ON THIS TRACK FOLLOWING THE BLOCK
 EXTENT 1 FROM 517 15 TO 517 16 SEQUENCE NUMBER = 0 CODE = 01 2017 TRACKS ALLOCAT

SOU2 FORMAT 1 DSCB AT 0/ 2/ 6 (CC/MM/SS) RECORD FORMAT IS PB
 NUMBER OF EXTENTS = 1 DATA SET ORGANIZATION IS PARTITIONED
 BLOCKSIZE = 7296 LOGICAL RECORD LENGTH = 80
 SECONDARY ALLOCATION - CODE = 00 QUANTITY = 1
 THE LAST BLOCK WAS WRITTEN ON RELATIVE TRACK 6 AS RECORD NUMBER 2
 THERE ARE 5088 BYTES REMAINING ON THIS TRACK FOLLOWING THE BLOCK

Figure 2.- VTOC distributed system V15.

VOLUME FCPAT 4 DSCB AT 0/ 1/ 1 (CC/HH/R) 14731/13-13 73-165

EXTENT OF VTCC - FROM 0/ 1 TO 0/ 4 SEQUENCE NUMBER = 0 CODE = 01 4 TRACKS ALLOCATED

DEVICE CHARACTERISTICS

 NUMBER OF LOGICAL CYLINDERS = 203

 NUMBER OF TRACKS PER LOGICAL CYLINDER = 20

 NUMBER OF BYTES PER TRACK = 7294

 NUMBER OF FULL DSCBS PER TRACK = 25

 NUMBER OF PDS DIRECTORY BLOCKS PER TRACK = 17

FCPAT 5 DSCB AT 0/ 1/ 2 (CC/HH/R) EXTENTS OF AVAILABLE SPACE

 STARTING AT RELATIVE TRACK 3840 (192/ 3) THERE ARE 6 CYLINDERS AND AN ADDITIONAL 15 TRACKS AVAILABLE

 A TOTAL OF 175 TRACKS AVAILABLE

HEMORJ FCPAT 1 DSCB AT 0/ 1/ 3 (CC/HH/R) RECORD FORMAT IS FB

 NUMBER OF EXTENTS = 1 DATA SET ORGANIZATION IS PARTITIONED

 BLOCKSIZE = 3120 LOGICAL RECORD LENGTH = 30

 SECONDARY ALLOCATION - CODE = CO QUANTITY = 8

 THE LAST BLOCK WAS WRITTEN ON RELATIVE TRACK 269 AS RECORD NUMBER 3

 THERE ARE 2485 BYTES REMAINING ON THAT TRACK FOLLOWING THE BLOCK

 EXTENT 1 FROM 1/ 0 TO 15/ 19 SEQUENCE NUMBER = 0 CODE = 81 300 TRACKS ALLOCAT

 EXTENT 2

 EXTENT 3

 THIS DSCB IS CHAINED TO FCPAT 3 DSCB AT 0/ 1/ 5 (CC/HH/R)

MADEVZ FCPAT 1 DSCB AT 0/ 1/ 4 (CC/HH/R) RECORD FORMAT IS U

 NUMBER OF EXTENTS = 11 DATA SET ORGANIZATION IS PARTITIONED

 BLOCKSIZE = 7294 LOGICAL RECORD LENGTH = 0

 SECONDARY ALLOCATION - CODE = CO QUANTITY = 20

 THE LAST BLOCK WAS WRITTEN ON RELATIVE TRACK 2081 AS RECORD NUMBER 15

 THERE ARE 887 BYTES REMAINING ON THAT TRACK FOLLOWING THE BLOCK

 EXTENT 1 FROM 32/ 0 TO 100/ 19 SEQUENCE NUMBER = 0 CODE = 81 1330 TRACKS

 EXTENT 2 FROM 101/ 0 TO 120/ 19 SEQUENCE NUMBER = 1 CODE = 81 400 TRACKS

 EXTENT 3 FROM 167/ 0 TO 237/ 19 SEQUENCE NUMBER = 2 CODE = 81 100 TRACKS

 A TOTAL OF 1940 TRACKS ALLOC

FCPAT 3 DSCB AT 0/ 1/ 5 (CC/HH/R) EXTENTS CONTINUED FROM A FCPAT 1 DSCB

 EXTENT 4 FROM 247/ 0 TO 317/ 19 SEQUENCE NUMBER = 3 CODE = 81 160 TRACKS

 EXTENT 5 FROM 121/ 0 TO 143/ 19 SEQUENCE NUMBER = 4 CODE = 81 400 TRACKS

 EXTENT 6 FROM 144/ 0 TO 151/ 19 SEQUENCE NUMBER = 5 CODE = 81 100 TRACKS

 EXTENT 7 FROM 152/ 0 TO 157/ 19 SEQUENCE NUMBER = 6 CODE = 81 100 TRACKS

 EXTENT 8 FROM 160/ 0 TO 167/ 19 SEQUENCE NUMBER = 7 CODE = 81 100 TRACKS

 EXTENT 9 FROM 168/ 0 TO 173/ 19 SEQUENCE NUMBER = 8 CODE = 81 100 TRACKS

 EXTENT 10 FROM 176/ 0 TO 183/ 19 SEQUENCE NUMBER = 9 CODE = 81 100 TRACKS

 EXTENT 11 FROM 184/ 0 TO 191/ 19 SEQUENCE NUMBER = 10 CODE = 81 160 TRACKS

 A TOTAL OF 1580 TRACKS ALLOC

Figure 3.- VTOC of development system.

```

//PUNCHIT JOB (R0XXXXX,RDRD),*MBNEWMAN*,REGION=100K,TIME=(.39) JOB CARD
/*VOLREQ ID=(NASTRAN15) DISTRIBUTED SYSTEM V.15
// EXEC PGM=IEBPTPCH
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSNAME=SOU1,UNIT=SYSDA,DISP=OLD,VOL=SER=VOLNUM,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=7280)
//SYSUT2 DD SYSOUT=3
//SYSIN DD *
PUNCH TYPCRG=PO,MAXNAME=60,MAXFLDS=60
MEMBER NAME=DS1
RECORD FIELD=(80)
MEMBER NAME=DS1A
RECORD FIELD=(80)
MEMBER NAME=DS1ABD
RECORD FIELD=(80)
/

```

NOTE SOU1 IS DATA SET NAME, VOLNUM IS DISK VOLUME SERIAL NUMBER

Figure 4.- Punch source from SOU1.

```

//PRINT JOB (R0XXXXX,RDRD),*MBNEWMAN*,REGION=100K,TIME=(.39) JOB CARD
/*VOLREQ ID=(NASTRAN15) DISTRIBUTED SYSTEM V.15
//FORMAT PR,DSNAME=SYSUT2,TRAIN=HN
// EXEC PGM=IEBPTPCH
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD UNIT=SYSDA,VOLUME=SER=VOLNUM,DISP=OLD,
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=7280),DSN=SOU1
//SYSUT2 DD SYSOUT=A
//SYSIN DD *
PRINT TYPCRG=PO,MAXFLDS=80,MAXNAME=80,MAXLINE=45
MEMBER NAME=READ1
RECORD FIELD=(80)
MEMBER NAME=READ2
RECORD FIELD=(80)
/

```

Figure 5.- Prints source from SOU1.

```

//COMPILE JOB (RDXXXX,RODR),*MBNEWMAN*,REGION=200K,TIME=(,J9)
/*VOLREW ID=(NASTRANDEV)
// EXEC FORTGC,PARM,FORT='NUDECK'
//FORT,SYSLIN DD UNIT=2314,DISP=SHR,VOL=SER=VOLDEV,
// USN=NEWOBJ(IFX180)
//FORT,SYSIN DD *

```

JOB CARD
DEVELOPMENT DISK

```

BLOCK DATA
CIFX180
COMMON /IFPX1/ N,11(100),12(100),13(100),14(100),15(100),16(120),
* 17(64)
DATA N/310/
DATA 11/4HGRID,4H ,4HGRDS,4HET ,4HADUM,4HI ,4HSEUG,4HP
5 ,4HCORD,4H1R ,4HCORD,4H1C ,4HCORD,4H1S ,4HCORD,4H2R
9 ,4HCORD,4H2C ,4HCORD,4H2S ,4HPLOT,4HEL ,4HSPCL,4H
3 ,4HSPCA,4HDD ,4HSUPD,4HRT ,4HOMIT,4H ,4HSPC ,4H
7 ,4HMPC ,4H ,4HFORC,4HE ,4HMUME,4HNT ,4HFUNC,4HE1
1 ,4HMOME,4HNT1 ,4HFORC,4HE2 ,4HMUME,4HNT2 ,4HPLUA,4HD
5 ,4HSLQA,4HD ,4HGKAV,4H ,4HTEMP,4H ,4HGENE,4ML
9 ,4HPRUD,4H ,4HPTUB,4HE ,4HPVIS,4HC ,4HADUM,4H2
DATA 15/4HTEMP,4HP1 ,4HTEMP,4HP2 ,4HTEMP,4HP3 ,4HTEMP,4HRB
5 ,4HGRID,4HB ,4HFSLI,4HST ,4HRING,4HFL ,4HPRES,4HPT
9 ,4HCFLU,4HID2 ,4HCFLU,4HID3 ,4HCFLU,4HID4 ,4HAXIF,4H
C 3 ,4HBDYL,4HIST ,4HFREE,4HPT ,4HSELE,4HCT ,4HSELE,4HCT1
3 ,4HBDYL,4HIST ,4HFREE,4HPT ,4HASET,4H ,4HASET,4H1
7 ,4HCTET,4HRA ,4HCWED,4HGE ,4HCHEX,4HA1 ,4HCHEX,4HA2
1 ,4HDMIA,4HA ,4HFLSY,4HM ,4HAXSL,4HOT ,4HCAXI,4HF2
5 ,4HCAXI,4HF3 ,4HCAXI,4HF4 ,4HCALO,4HT3 ,4HCALO,4HT4
9 ,4HGKID,4HF ,4HGRIU,4HS ,4HSLBU,4HT ,4HCHBU,4HT
3 ,4HQMBD,4HY ,4HMAT4,4H ,4HMAT5,4H ,4HSAME,4H
C ADD CSOL101,CSOL102,CSOL103 NAMES TO DIRECTORY OF CONNECTION CARD NAMES MAN
7 ,4HSAME,4H1 ,4HINPU,4HT ,4HOUTP,4HUT ,4HCAL,4HID1
1 ,4HCAL,4HID2 ,4HCAL,4HID3 ,4H0000,4H0000,4H0000,4H0000
5 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
9 ,4H0000,4H0000,4H0000,4H0000 /
DATA 16/4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
5 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
9 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
3 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
7 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
1 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
5 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
9 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
3 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
7 ,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000,4H0000
1 ,4HCPRU,4HD ,4HCPTU,4HBE ,4HCPRU,4HAD1 ,4HCPRU,4HAD2
5 ,4HCPTU,4HIA1 ,4HCPTU,4HIA2 ,4HCPRU,4HPLT ,4HCPTU,4HPLT
9 ,4HCPTU,4HSC ,4HCOPU,4HMASS,4HGNDP,4HNT ,4HHTHA,4HSS
3 ,4HRES,4H ,4HLFRE,4HQ ,4HMFRE,4HQ ,4HMOD,4HES
7 ,4HG ,4H ,4HW3 ,4H ,4HW4 ,4H ,4HMUDA,4HCC /
DATA 17/4HMPCS,4H ,4HSPCS,4H ,4HLOAD,4HS ,4HMLTH,4HDS
5 ,4HDEFD,4HRMS ,4HTEMP,4HLD5 ,4HTEMP,4HMTS ,4HNICS ,4H
9 ,4HAOUT,4HS ,4HLOOP,4HS ,4HLOOP,4HS ,4HDLQA,4HDS
3 ,4HFREQ,4HS ,4HTFS ,4H ,4HPLOT,4HS ,4HTSTE,4HFS
7 ,4HPOUT,4HS ,4HTEMP,4HMXS ,4HDSU,4HS ,4HKKPP,4HS
1 ,4HMZPP,4HS ,4HMZPP,4HS ,4HCNET,4HMUDS,4HSUAM,4HPS
5 ,4HPLCO,4HS ,4HNLFO,4HNCES,4HXYOU,4HTS ,4HDELE,4HTES
9 ,4HRAND,4HMS ,4HAXYO,4HUTS ,4HNULU,4HOPS ,4H0000,4H0000/
END

```

Figure 6.- Compiles and puts object into development data set NEWOBJ.

```

//LNKEDT JOB (R0XXXXX,RDRD),*MBNE*MAN*,REGION=275K,TIME=2          JOB CARD
/*VOLREQ ID=(NASTRAN15)                                           DISTRIBUTED SYSTEM V.15
/*VOLREQ ID=(NASTRANDEV)                                           DEVELOPMENT DISK
// EXEC FORTHL,PARM,LKED=*MAP,LIST,OVLY,DC,LET,SIZE=(262K,72K)*
//LKED.SYSLIB DD DISP=SHR,DSN=SYS1.ERRPK
//                               DD DISP=SHR,DSN=SYS1.FORTLIB
//LKED.SYSLMOD DD UNIT=SYSDA,DISP=SHR,VOL=SER=VOLDEV,DSN=NADEV2.
//                               SPACE=(CYL,(60,8,5))
//LKED.LIP DD UNIT=SYSDA,DISP=SHR,VOL=SER=VOLDEV,DSN=NEOBJ
//LKED.LIB DD UNIT=SYSDA,DISP=SHR,VOL=SER=VOLNUM,DSN=OBJ
//LKED.SYSIN DD *
    INCLUDE LIP(LIEN,KISOPK)
    CHANGE EXIT(PXIT36)
    INCLUDE LIB(PXIT)
    INCLUDE LIB(PXIT36)
    INCLUDE LIB(LJNKNSU3,XSEM3)
    INCLUDE LIB(CORSZ)
    INCLUDE LIB(SEMDD,RETURN,XEQT,MAPFNS,FMTOGO,CONMSG)
    INCLUDE LIB(MESSAGE,SSWTC,HOPEN,FREAD,CLSTAB,OPNCOR,FNAME)
    INCLUDE LIB(PRELOC,MHTTRL)
OVERLAY A
    INCLUDE LIB(PAGE)
OVERLAY A1
    INCLUDE LIB(MSGWRT,USRMSG)
OVERLAY A1
    INCLUDE LIB(WTSTRP)
    CHANGE NTRAN(PXIT),LINK(PXIT)
    INCLUDE LIB(ENDDSS)
OVERLAY ENDDSS
    INSERT ENDDSS
OVERLAY A1
    INCLUDE LIB(QPARAM)
OVERLAY A1
    INCLUDE LIB(XSAVE)
OVERLAY A1
    INCLUDE LIB(XCEI)
OVERLAY A1
    INCLUDE LIB(XCHK)
OVERLAY A1
    INCLUDE LIB(GNFIST,RPDABD,XSFA,XSOSGN,ACLEAN,XPUNP,XDPH)
    INCLUDE LIB(XPULCK,XPURGE)
    INSERT XSFA1
OVERLAY ESFA
    INSERT ESFA,USCENT
OVERLAY A1
    INCLUDE LIB(TABPT,TABPNT,MATDUM,MATPRN)
OVERLAY TABPRA
    INSERT TABPRA
OVERLAY A1
    INCLUDE LIB(PRTPRN)
OVERLAY A
    INCLUDE LIP(GPTABD)
    INCLUDE LIB(DELSCT)
    INCLUDE LIB(HMAT,HSRCH)
    INCLUDE LIB(GMMATO,PRETRD,INVERD,GMMATS,PREMAT)
    INCLUDE LIB(SAXB,DAXB,SADOTB,OADOTB)
    INSERT GPTA1
    INSERT HMTOUT
    INSERT HMATOU
    INSERT MATIN,MATOUT
    UU
    00000010
    00000020
    00000030
    00000040
    00000050
    00000060
    00000070
    00000080
    00000090
    00000100
    00000110
    00000120
    00000130
    00000140
    00000150
    00000160
    00000170
    00000180
    00000190
    00000200
    00000210
    00000220
    00000230
    00000240
    00000250
    00000260
    00000270
    00000280
    00000290
    00000300
    00000310
    00000320
    00000330
    00000340
    00000350
    00000360
    00000370
    00000380
    00000390
    00000400
    00000401
    00000410
    00000420
    00000430
    00000440
    00000450
    00000460
    00000470

```

Figure 7.- Link edit and puts execution module into NADEV development set.

FIGURE 7 CONTINUED

OVERLAY SMA1	00000480
INCLUDE LIP(SMA1B0)	00000490
INSERT SMA1CL,SMA110,SMA1BK,SMA1ET,SMA1DP	00000500
INSERT SMA1HT	00000510
CHANGE KBEAM(PEXIT)	00000520
INCLUDE LIP(SMA1A)	530
INSERT SMA1SC,APLE	
INCLUDE LIB(SMA1B,DETCK)	540
INCLUDE LIP(SMA1)	541
CHANGE KBEAM(PEXIT)	00000550
INCLUDE LIB(PLA1)	00000560
OVERLAY SMAEL	00000570
INCLUDE LIB(KROD,KBAR,KTUBE,KPANEL,KELAS)	00000580
INCLUDE LIB(KTRMEM,KQDMEM,KIRBSC,KTRPLT,KQDPLT,KTRIQU,HHBDY,HRING)	00000590
OVERLAY SMAEL	00000600
INCLUDE LIB(KCONE,KCONEX)	00000610
OVERLAY SMAEL	00000620
INCLUDE LIB(KTRIRG,KTRAPR,DKL,DKINT,DKK,DKM,DKJ,DKEF,DKBY,DK100)	00000630
INCLUDE LIB(DK211,DK219,DKJAB,KFAC)	00000640
OVERLAY SMAEL	00000650
INCLUDE LIB(KTORDR,DMATRIX,ROMBOK,D4K,D5K,D6K)	00000660
OVERLAY SMAEL	00000670
INCLUDE LIB(KFLUD2,KFLUD3,KFLUD4,KSLUT,KTEIRA,KSOLID,KPLIST)	00000680
OVERLAY SMAEL	00000690
INCLUDE LIB(KDUM1,KDUM2,KDUM3,KDUM4,KDUM5,KDUM6,KDUM7,KDUM8,KDUM9)	700
OVERLAY SMAEL	701
INSERT KISOPR	702
INSERT DTOT,FORMTO,INV3X3,JTPTN,MATERL,MBC1A,MMAT,MULTPN,MXYZ,NSELEC	703
INSERT PARTL,PARXYZ,SETCON	704
INSERT SMA1PD,ELDATA,NMAT,FRONT3,PARTIL	705
OVERLAY SMA1	00000710
CHANGE MBEAM(PEXIT)	00000720
INCLUDE LIP(SMA2A)	730
INCLUDE LIB(SMA2B)	740
INCLUDE LIP(SMA2BD,SMA2)	741
INSERT SMA2SC,MAPLE	742
INSERT SMA2CL,SMA210,SMA2BK,SMA2ET,SMA2DP	00000750
OVERLAY SMAEL2	00000760
INCLUDE LIB(MROD,MBAR,MTUBE,MASSD,BVISC,MCONMX,MCONE)	00000770
INCLUDE LIB(MSOLID,MFLUD2,MFLUD3,MFLUD4,MFKEE,MSLOT)	00000780
INCLUDE LIB(MASSTQ)	00000790
INCLUDE LIB(MCBAR,MCROD,MTRBSC,MQDPLT,MTRPLT,MTRIQU)	00000800
OVERLAY SMAEL2	00000810
INCLUDE LIB(MTRIRG,MTRAPR,DM1,DMINT,DMK,DMH,DMJ,DMEF,DMBY,DM100)	00000820
INCLUDE LIB(MFAC,DMJAB,DM219,DM211,MTORDR)	00000830
OVERLAY SMAEL2	00000840
INCLUDE LIB(MDUM1,MDUM2,MDUM3,MDUM4,MDUM5,MDUM6,MDUM7,MDUM8,MDUM9)	00000850
OVERLAY SMAEL2	851
INCLUDE LIP(MISOPR,MLIEN)	852
INSERT SMA2PD,ELDAT2,NMAT2,FRONT2,PARTI2	853
OVERLAY OEND(REGION)	00000860
INSERT SMA1X	00000870
INSERT SMA2X	00000880
OVERLAY EJDUM2	00000890
INCLUDE LIB(EJDUM2)	00000900
INSERT EJDUM2	00000910
ENTRY LINKNSUJ	00000920
NAME LINKNSUJ(H)	00000930

Figure 7.- Concluded.

```

//ALTER JOB (RDXXXXX,RDND),'MBNEWMAN',REGION=JUDK,TIME=(,59)
//VOLUME0 ID=(NASTRANDEV)
//FORMAT PR,DDNAME=FT04F001,TRAIN=HN
//FORMAT PR,DDNAME=FT06F001,TRAIN=HN
//FORMAT PR,DDNAME=SNAPSHOT,TRAIN=HN
//S2 EXEC F56THG
//PROG.NAME DD USNAME=SYS1.LINKLIB(VTOCPRT),VOLUME DEF=SYSLIB1,
//          DISP=OLD
//GO,DD2 DD UNIT=SYSDA,DISP=OLD,VOLUME=(PRIVATE,RETAIN,SER=(VOLDEV))
//GO,SYSIN DD *
VOLDEV
EOJ
// EXEC NSTMBN
//NASTRAN,STEPLIB DD UNIT=2314,VOL=SER=VOLDEV,DISP=SHR,DSN=NADEV2
//NASTRAN,SNAPSHOT DD UNIT=(CTC,,DEFER),UCB=BLKSIZE=882
//SYSIN DD *
NASTRAN BUFFSIZE=1800,SYSTEM(31)=4096,SYSTEM(9)=35,CONFIG=10
ID MARY,NEWMAN
APP DISP
SOL 1,0
TIME 5
DIAG 15
DIAG 2,8,13,14
DIAG 19,21,22
DIAG 25
ALTER 21
TABPT GPDI,.... //S
ALTER 26
TABPT GPCT,.... // S
TABPT ECPT,GPDI,EST,, // S
TABPT EUEXIN,GEOMZ,.... // S
ALTER 111
MATPRN UG,PGG,06,.... // S
ENDALTER
CEND
OLOAD#ALL
SPC#111
SPC#NALL
STRESS#ALL
ELFORCE#ALL
DISP#ALL
LOAD#100
TITLETEST OF BODY FORCE --- 1 CSOLID2 ELEMENT
ECHO#BOTH
BEGIN BULK
GRUSET
GRID 1 1. 1. 0.
GRID 2 1. 1. 0.
GRID 3 -1. 1. 0.
GRID 4 -1. -1. 0.
GRID 5 1. -1. 3.
GRID 6 1. 1. 3.

```

JOB CARD
DEVELOPMENT DISK

MONDEBUB

456

Figure 8.- VTOC example and use of ALTER for table and matrix printouts.

GRID	7		-1.	1.	3.				
GRID	8		-1.	-1.	3.				
GRID	9		1.	0.	0.				
GRID	10		0.	1.0	0.0				
GRID	11		-1.0	0.0	0.0				
GRID	12		0.0	-1.0	0.0				
GRID	13		1.0	-1.0	1.5				
GRID	14		1.0	1.0	1.5				
GRID	15		-1.0	1.0	1.5				
GRID	16		-1.0	-1.0	1.5				
GRID	17		1.0	0.0	3.0				
GRID	18		0.0	1.0	3.0				
GRID	19		-1.0	0.0	3.0				
GRID	20		0.0	-1.0	3.0				
CSOLID2	1	10							
6E1	1	2	3	4	5	6	7	8	6E1
6E2	9	10	11	12	13	14	15	16	6E2
6E3	17	18	19	20					6E3
SPC	111	5	1	-.9E-05	6	12	.9E-05		
SPC	111	5	2	.9E-05	6	3	.3E-05		
SPC	111	5	3	.3E-05	17	1	-.9E-05		
SPC	111	6	12	-.9E-05	17	2	0.0		
SPC	111	6	3	.3E-05	17	3	.15E-05		
SPC	111	7	1	.9E-05	18	1	0.0		
SPC	111	7	2	-.9E-05	18	2	-.9E-05		
SPC	111	7	3	.30E-05					
SPC	111	18	3	.15E-05					
SPC	111	19	1	.9E-05					
SPC	111	19	2	0.0					
SPC	111	19	3	.15E-05					
SPC	111	20	1	0.0					
SPC	111	20	2	.9E-05					
SPC	111	20	3	.15E-05					
MAT1	10	1.E07		.3	4.0				
GNAY	100		25.	-1.0	0.0	00.0			
ENDDATA									
/0									

Figure 8.- Concluded.